ESA 51
USER MANUAL

Electro Systems Associates Pvt Ltd
www.esaindia.com

PREFACE

This is the user's manual for ESA 51 microcontroller trainer. The manual describes the hardware and software components of ESA 51 and gives the interface information necessary for expanding the system.

This manual describes in detail the facilities offered by the stand-alone mode monitor program and the serial monitor program, the on-line assembler, disassembler packages. The on-board facilities: Centronics Parallel Printer Interface, DAC interface and optional ADC interface, are also described in this manual. Communication with the Host Computer is also described.

Please note that this volume is a user's guide for ESA 51 and such does not deal elaborately with the features of 8031 microcontroller family and related peripherals and their programming. Details regarding these can be obtained from the following INTEL Publication.

Microcontroller Handbook

While every effort has been made to present the information in an accurate and simple fashion, we do welcome suggestions for improving the quality and usefulness of this manual.

Please address your correspondence to:

ELECTRO SYSTEMS ASSOCIATES PVT LTD.,
4215 J.K. Complex, First Main Road, Subramanyanagar
P.O. Box No. 2139 BANGALORE - 560 021 INDIA
Fax : 91-80-3325615 Phone : 3323029 3322924
email : esaindia@vsnl.com,
www.esaindia.com

CONTENTS

APPENDICES

CHAPTER 1

INTRODUCTION

Intel's MCS-51 family of microcontrollers and its derivatives are increasingly becoming popular for instrumentation and control applications due to its speed and powerful instruction set which are essential for real-time applications. This has created the need for a good trainer and development tools. ESA 51 (an advanced version of ESA 31) provides complete solution for this requirement. It can be used as a flexible instructional aid in academic institutions and a powerful development kit in R&D Labs.

ESA 51 has on-board DAC, ADC (optional) and parallel printer interface. The system firmware provides stand-alone mode monitor, serial monitor, single line assembler, disassembler and drivers for EPROM programmer and parallel printer interfaces. ESA 51 is supported with comprehensive and user-friendly documentation.

MAIN FEATURES

- ESA 51 operates on single +5V power supply either in stand-alone mode using PC keyboard and LCD or with host PC through its RS-232-C / RS 485 interface in serial mode.
- Stand-alone and serial monitor programs support the entry of user programs, editing and debugging facilities like breakpoints (128K), single stepping and full speed execution of user programs.

- Line assembler & disassembler.
- Total on-board memory is 128K bytes of which 96K bytes RAM has battery backup provision.
- On-board parallel printer port.
- On-board 8 bit DAC using 0800.
- Optional on-board 12 bit ADC using AD1674.
- 48 I/O lines and four programmable interval timers.
- 13 port lines of 8031 brought out to the connector including INT1, RXD & TXD pins (6 lines are shared for optional ADC).
- Buffered bus signals are available through ribbon cable connector for easy system expansion.
- Driver software for file upload/download to/from host PC.

ACCESSORIES (OPTIONAL)

- Power Supply : +5V @ 3A; +12V @ 250mA; -12V @ 100mA and +30V @ 100mA
- PC keyboard and 20 X 4 LCD module for stand-alone mode of operation.
- EPROM programmer interface.
- Interface Modules for training purpose : Keyboard, Elevator, Display, ADC with DAC, Dual DAC, 8 bit-16 Channel ADC, 12 bit 8 Channel ADC, Logic Controller, Traffic Lights, Tone Generator, Stepper Motor, Opto Isolated Input, Opto Isolated Output, Relay Output etc.
- 12 bit ADC (AD1674) with 8 channel MUX.
- 3.6V Ni-Cd battery for power backup to RAM.
- Parallel printer cable.
- RS 485 interfacing cable.
- 26 core ribbon cable connector set.
- 50 core ribbon cable for bus expansion.

CENTRAL PROCESSOR

8031 MCU @ 11.0592 MHz.

MEMORY

Four 28 pin JEDEC sockets provide following

PROGRAM MEMORY

ROM : 32K bytes of system firmware using 27C256.

RAM : 32K bytes using 62256.

DATA MEMORY

RAM : 64K bytes using 62256 (32K X 2). Upper most 8K bytes are reserved for I/O addressing and I/O expansion.

PERIPHERALS

8155 : Static HMOS 256 bytes RAM with I/O ports and timer. RAM reserved for monitor, 14 bit timer is available for user and port lines are used for DAC and ADC.

8255 : PPI, Three nos. Two nos are for user, one supplied; another for user expansion. The remaining one is used for parallel printer and optional LCD.

8253 : Programmable interval timer. Three16 bit programmable timers available for user

SCN 2681 : Dual channel UART for serial, RS-232-C & RS 485 communication supporting all standard baud from 110 to 19200.

8042 : Universal Peripheral Interface (optional) used to interface PC keyboard in stand-alone mode.

ADC 1674 : 12 bit ADC, 10µs (optional).

DAC 0800 : 8 bit DAC.

INTERRUPTS

External : INT0 is used for implementing single stepping, breakpoints and user's break switch. INT1 is available to user.

Internal : Internal timer and serial interrupts are available to user.

INTERFACE SIGNALS

Bus : STD Bus compatible bus signals available through a 50 pin ribbon cable connector.

Single chip mode: MCU port lines available through a 50 pin ribbon cable connector.

Parallel I/O : 48 TTL compatible lines (2 X 8255) brought out through two 26 pin ribbon cable connectors.

Serial I/O : RS-232-C through on-board 9 Pin D-type female connector. RS 485 through on-board 9 Pin D-type male connector.

Printer : PC compatible parallel printer interface available on a 25 pin D type female connector

Timer Signals : Two 8253 and one 8155 timer signals are available at the 50 pin ribbon cable connectors.

Analog Signals : 8 analog inputs for ADC are fed through terminal blocks. DAC output is available through a test point.

POWER SUPPLY REQUIREMENT

+5V @1600mA (max)
±12V @ 250mA(max) for ADC and DAC

CHAPTER 2

CONFIGURATION AND INSTALLATION

2.1 CONFIGURATION OF ESA 51

ESA 51 Microcontroller Trainer is versatile and can be configured in three different modes which are determined by DIP switch settings (refer to the component layout diagram in appendix A to locate the DIP switch). This chapter describes all the configurable options and the installation procedures.

2.1.1 OPERATIONAL MODE SELECTION

ESA 51 can be operated either in the stand alone mode using PC keyboard and LCD or in the serial mode through RS-232-C/RS 485 interface. In the serial mode, the trainer is connected to a CRT terminal or to a host computer system (PC) through an RS-232-C/RS 485 interface. In either mode of operation, the system provides a variety of commands for program development/debugging, several features like on-line assembler, disassembler etc., The selection of the desired mode of operations is done as follows :

Sl.No.	Mode of Operation	DIP-Switch Status	
		Switch 4	Switch 5
1.	Serial mode with RS-232-C interface.	ON	OFF*
2.	Serial mode with RS 485 interface	ON	ON
3.	Stand-alone mode	OFF	×

(* factory installed option)

(× don't care)

Chapters 3 & 4 describes the commands available in stand-alone mode and serial mode respectively.

2.1.2 PRINTER ENABLE/DISABLE

ESA51 firmware includes the driver program for centronics compatible parallel printer interface. This driver can be enabled/disabled as shown below:

DIP Switch Position 6	Printer Port
OFF	Disabled.*
ON	Enabled.

(* factory installed option)

Chapter 7 describes the parallel printer interface in detail.

2.1.3 BAUD RATE SELECTION

In the serial mode of operation, ESA51 configures the on-board SCN 2681 Dual Channel UART as follows :

* Asynchronous mode
* 8 bit character length
* 2 stop bit
* no parity

Baud rate selection for RS-232-C as well as RS 485 can be set using DIP switches 1 to 3 as shown below :

S3	S2	S1	Baud Rate
ON	ON	ON	110
ON	ON	OFF	300
ON	OFF	ON	600
ON	OFF	OFF	1200
OFF	ON	ON	2400
OFF	ON	OFF	4800
OFF	OFF	ON	9600*
OFF	OFF	OFF	19,200

(* factory installed option)

2.1.4 MEMORY SELECTION

ESA 51 has four 28-Pin sockets for memory. System firmware (32K bytes) is supplied in a 27256 EPROM at the socket U9. 32K bytes of static RAM is provided by a 62256 at the socket U10 as user program memory.32Kbytes of data memory is provided at U11. The fourth socket at U12 is populated with 62256 to provide, 24K bytes of Data Memory.

2.2 INSTALLATION OF ESA 51

To install ESA51, the following accessories are required.

a) Power Supply 5V, 3A
Additionally
+12V @250mA, -12V @ 100mA for ADC & DAC circuitry.

b) For serial mode of operations :

Host PC with the driver software for host system. (Refer chapter 9 for details).

2.2.1 INSTALLATION PROCEDURES FOR SERIAL MODE OF OPERATION

a) Select serial mode of operations (Ref. Section 2.1.1)

b) Select printer if required (Ref. Section 2.1.2)

c) Set the desired baud rate (Ref. Section 2.1.3)

d) Connect ESA 51 to the host system through RS-232-C/RS485 cable (Appendix C describes the RS-232-C interface requirements) over the connector J4/J5. (Refer Appendix A for locating the connectors). Turn-on the system and execute the driver software (Ref. Chapter 9 for details)

e) Connect the appropriate power supply.

ESA 51 performs POST (Power On Self Test) operation. During the POST operation all register will be intialized to CPU's reset condition. Breakpoints in both program and data memory will be cleared and disabled. Then displays the following Sign On message followed by the command Prompt '>' in the next line.

ESA 8051 Serial monitor V x.y

(V x.y indicates version x and revision y)
>

Now ESA51 is ready for operation in serial mode.

NOTE : If the LCD module is installed the message "SERIAL" will be displayed on the display.

2.2.2 NO RESPONSE IN SERIAL MODE :

If there is no response from ESA51 in serial mode, after installing it as described in the previous section, check the following items :

a) Check the configuration of ESA51 again. (DIP Switch settings)

b) Check the power supply connections and voltages.

c) Check the baud rate of ESA51 and the host connected to it.

d) If a host system is the controlling device, make sure that the XT51 driver program is running, the RS-232-C/RS 485 cable is connected to the port and the port is working.

e) Check the RS-232-C/RS 485 connections at both the ends. (Refer Appendix C for the interfacing details)

f) Check the handshake signals of RS-232-C interface (Ref. Appendix C)

NOTE :
DIP Switch status is read only at power –ON/Reset. If the user changes the settings, these changes will be effective by pressing the RESET key or restart the trainer by switching OFF and ON the power supply. If the problem still persists, please contact the manufacturer / service center.

2.2.3 INSTALLATION PROCEDURE FOR STAND-ALONE MODE OF OPERATION

a) Select stand-alone mode operation (Ref. 2.1.1)

b) Connect the power supply of required capacity to ESA51 and switch-on the power.

c) Now the following message appears on the LCD for few seconds.

POST

Power- On Self Test is being done. Following initializations are also done.

All register will be initialized to CPU's reset condition.

Breakpoints in both program and data memory will be cleared and disabled. Then displays the following sign on message followed by the command prompt '>' in the next line.
ESA – 51
>
ESA 51 is ready for operation in the stand-alone mode.

2.2.4 NO RESPONSE IN STAND-ALONE MODE

If the correct sign – on message does not appear in the stand-alone mode, check the following items.

a) If the LCD is blank, check the power supply connections and voltages.

b) If the LCD display shows random pattern, check the configuration settings once again.

NOTE :
DIP switch is read only at power - ON/Reset. If you change the settings, either press RESET key or switch OFF and then switch ON the power supply. If the problem persists, please contact the manufacturer/service center.

CHAPTER 3

STAND-ALONE MODE MONITOR

3.1 INTRODUCTION

This chapter describes the commands supported by the stand-alone mode monitor program. The stand-alone mode monitor allows ESA 51 to be operated using a PC keyboard and LCD module.

The system must be configured for stand-alone mode of operation as described in section 2.1.1. The commands are described in this chapter.

When the system enters stand-alone mode of operation, the sign - on message “ESA-51 ” is displayed followed by prompt “>” on the next line indicating that the monitor is ready to accept commands from the user.

3.2 STRUCTURE OF MONITOR COMMANDS

Whenever the monitor is ready to accept a command from the user, it outputs a greater than symbol (` >`) as command prompt character at the beginning of a new line.

The commands entered by the user consist of a single character command mnemonic followed by a list of command parameters. This list can have four parameters depending on the command being used.

When more than one parameter is required, a single (`,`) or space is used between the parameters as a seperator.

A command is terminated by a <CR>. Commands are executed one at a time and one command in one command line.

PARAMETER ENTRY

All numeric parameters are to be entered as hexadecimal number. The valid range for one-byte parameters is 00 to FF and if more than 2 digits are entered, only the last two digits are valid (leading zeros may be omitted). Thus all one byte values are interpreted modulo 256 (decimal). The valid range for 2 byte parameter is 0000 to FFFF and longer values are evaluated modulo 64k (i.e. only the last four digits are valid).

The register name abbreviation entries required by the R commands are described later while describing the R command in detail.

RESPONSE TO ERRORS

Whenever an error is detected by the monitor (either in the command entry or in the command execution) the command is aborted "Error" is displayed on the next line along with a sign "^" attached pointing to the place where the error occurred, followed by command prompt (The possible error conditions are described while illustrating the individual commands).

Command execution occurs only after a valid delimiter (a <CR>) is entered. Hence a command entry can be cancelled anytime before the delimiter is entered by pressing <Esc>. The command gets terminated, followed by a command prompt.

3.3 MONITOR COMMANDS

Each command described in this chapter consists of one to two characters, followed by appropriate parameters and data. The commands are summarized in Table 3.1 and are described in detail in the section which follows. In the table as well as in the subsequent description, the following notations are used:

NOTATIONAL CONVENTIONS

SYMBOL	NAME	USAGE
{}	Curly braces with Ellipsis	Encloses a required argument 1 or more Times

[ ]	Square brackets	Encloses an item that appears 0 or 1 time.
[ ]…	Square brackets	Encloses an item that appears 0 or more times.
|	Vertical bar	Separates alternative items in a list
	Italics	Indicates a descriptive item that should be Replaced with an actual item.

TABLE 3.1 SUMMARY OF STAND ALONE MODE MONITOR COMMANDS

COMMAND	FUNCTION /FORMAT
A	Assembler A [address]
B	Clear/Display/Set/Breakpoint in Program memory, external Data memory B{[D{P|D}]|[{C|S}{P|D} address 1 [,address 2]]}
C	Compare a block of memory with destination block C{P|D|I {address 1, address 2, {P|D|I}, address 3}}
D	Disable breakpoint in program memory or data memory D{P|D}
E	Enable breakpoint in program memory or data memory E{P|D}
F	Fill a block of memory with a constant or search a string of Data in program memory, external data memory and internal Data memory. F {P|D|I}, address 1,address 2,data [,data[,data[,data]]],S
G	Transfer the processor control from the monitor to user Program. G [address]
H	Help. List all the commands supported by the Serial Monitor H
J	Jump to address J [address]

M	Modify/Display/Move memory contents in program memory, External data memory and internal data memory with all combinations M{P|D|I|B} address 1 [,address 2 [,{P|D|I}, address 3]].
P	Programmer
S	Execute one instruction of user program S [address]
Z	Disassembler Z[addr1[,addr2]]

3.3.1 M (MODIFY) COMMAND

FUNCTION

The M (Modify Memory) command is used to examine the contents of specified memory locations. Further if location are in RAM their contents can be altered if desired and block move contents of memory from program, data or internal memory to program, data or internal memory for all combinations.

FORMAT

M {P|D|I|B} address 1 [,address 2 {[,P|D|I}, address 3]]

OPERATION

1. Enter M followed by memory type, the address of the memory location to be examined and then enter <CR>. The monitor will now output the contents of that location.
2. To modify the contents of this location, the user can enter the new value now.
3. Enter a <CR>, either immediately or after the entry of a new value, to examine/modify the next sequential location. A <Esc> instead of the <CR> terminates the command and returns the monitor to the command entry mode.

ERROR CONDITION

1. Trying to modify the contents of non-existent or PROM location.

Examples:

Example 1 : Examine the PROM locations 11H

>MP11<CR>

>0011 FF <Esc>

>

Example 2 : Examine a few RAM location starting at 8820H and modify the contents of the location 8822H

>MD8820<CR>

8820 xx <CR>

8821 xx <CR>

8822 xx 55<CR>

8823 xx <Esc>

>

3.3.2 M (DISPLAY MEMORY) COMMAND

FUNCTION

This command is used to display the contents of the program memory, external or internal data memory.

FORMAT

M {P|D|I|}, address 1, address 2 <CR>

OPERATION

1. To use this command, enter M when prompted for command entry. After entering M, enter the memory type and then starting address of the memory block whose contents are to be displayed, then enter a comma, enter the end address of the memory block followed by a <CR>.
2. Now the monitor will output the starting address, the contents of the location from this address to the specified end address. The monitor routine displays 16 bytes in 3 lines. The number of bytes displayed on the first 3 lines are so adjusted that if the fourth line is present, its first location has address with the last nibble as zero. After the fourth line, to go to the next display section press <CR>.

Examples

Example 1 : To display the contents of program memory from location 0000H to 0015H.

>MD0000, 0015

0000: 02 00 03 02

25 CD FF FF FF FF

FF 02 FF F0 FF FF

0010: FF FF FF 02

FF F3

>

3.3.3 M (MOVE MEMORY) COMMAND

FUNCTION

This command is used to move a block of data from one area of the memory to another area.

FORMAT

M {P|D|I}, <address1>, <address2>, {P|D|I}

<destination address> <CR>

OPERATION

1. To use this command, enter M when prompted for command entry. Follow it with the type of memory starting address of the source block to be moved ("address1"), a comma, the ending address of the source block ("address2"), another comma, and then type of destination memory location, starting address of the area into which the source block is to be moved ("destination address"), followed by <CR>.
2. This operation moves the contents of memory locations from "address 1" to "address 2" to destination memory location starting from "destination address".
3. The system determines if there is any overlap between source and destination block, then the memory location will be over written from "address2" onwards.

ERROR CONDITIONS

1. Specifying an "end address" value, which is less than the value of the "start address".
2. Trying to move data into non-existent or read-only memory location.

Examples :

Example 1 : Move the contents of the location 800H through 80FH to the memory block beginning at 8840H

>MP800,80F,P8840<CR>

>

Example 2 :

>MP800,80F,P200<CR>

Error

>

An attempt to move data into ROM location produces the error message.

3.3.4 F (FILL MEMORY) COMMAND

FUNCTION

This command is used to fill a block of memory with specified constant.

FORMAT

F{P|D|I}, address1, address2, constant <CR>

OPERATION

1. To use this command enter F when prompted for command entry and enter the type of memory to be filled.
2. Now enter the starting address of the block of memory to be filled. Enter a comma. Now enter the ending address of the block. Again enter a comma. Now enter the constant. Press the <CR> to start the command execution.
3. Monitor now fills the block of memory from address1 to address 2 with the specified constant. Then the monitor displays the command prompt sign.

ERROR CONDITIONS

1. Specifying a value for the address 2 of the source block which is less than the value of the address1, of the source block.
2. Trying to fill the data in non -existent or read -only memory.

Examples :

Example1 : Filling the block of program memory with a constant 55H.

>FP8800,880F,55<CR>

>

Now you can use the M command to examine the block of memory to see that it is filled with the constant 55H.

3.3.5 C (COMPARE MEMORY) COMMAND

FUNCTION

Compare command can be used to compare the contents of one memory block with the content of another memory block.

FORMAT

C {P|D|I} address 1 of block 1, address2 of block 1

{P|D|I} address 1 of block 2 <CR>

OPERATION

1. To use this command, enter C when prompted for command entry and select the type of memory. Then enter starting address of the first block, a comma, ending address of the first block, another comma and destination type of memory and coma and then the starting address of the second block followed by the <CR>.
2. The monitor now compares the content of location beginning at address1 of block1 with the content of location beginning at address 1 of block2. This process continues till the contents of address 2 are compared with those of corresponding locations in the 2nd block. All mismatched address locations along with Data will be displayed.

Examples :

1) Compare the contents of memory locations 8000H to 8FFFH with those of a memory block beginning at 9000H

>CP8000,8FFF,9000<CR>

>

(This response showed that there is no mismatch)

2) CPA000,AFFF,P8000<CR>

ABC0 = 00 FF = 8BC0

AED8 = 48 54 = 8ED8

(This response showed that there is mismatch at two locations).

3.3.6 R (EXAMINE/MODIFY REGISTER) COMMAND

FUNCTION

This command is used to examine and optionally modify the contents of the registers.

FORMAT

R [reg] [[[new data] ,]…]<CR>

OPERATION

1. If you wish to examine/modify the contents of a particular register, Enter R (when prompted for command) followed by the register name abbreviation are shown in Table 3.2. Now the monitor will output the current contents of the specified register. The content of this register can be changed now by entering the new data value, followed by a valid terminator (a <CR> or <Esc>). If the command terminator <Esc> is pressed, the command gets terminated.

   If <CR> is pressed, next "sequential" register is displayed and allows optional modification. The sequence in which registers are displayed is shown in Table 3.2 (Note that this sequence is in closed loop).

TABLE 3.2

Register name	**Abbreviation**
Register A	A
Register B	B
Stack Pointer	SP
Flags Register	PSW
Data Pointer High	DPH
Data Pointer Low	DPL
Register TH0	TH0
Register TL0	TL0
Register TH1	TH1
Register TL1	TL1
Register P1	P1
Register P3	P3
Register Counter High	PCH
Register Counter Low	PCL
Register R0	R0
Register R1	R1
Register R2	R2
Register R3	R3
Register R4	R4
Register R5	R5

Register R6	R6
Register R7	R7

Examples :

Example 1

Examine and alter register A and then examine register B.

RA <CR>

A(E0) 24 <CR>

B(F0) 25<Esc>

>

3.3.7 J (JUMP TO ADDRESS-SET/CHANGE PC) COMMAND

FUNCTION

The J Command is used to change the program counter value to the desired address before executing a program by either GO command or SINGLE STEP command.

FORMAT

J [address] <CR>

OPERATION

To use this command, enter J when prompted for command entry. Now enter the desired starting address of the program you wish to execute. Enter <CR>. Now command prompt reappears on the next line.

Examples :

Example 1

Jump/set PC to address 100h

J [100] <CR>

>

3.3.8 G (GO) COMMAND

FUNCTION

The GO command is used to transfer the control of the system from monitor to the user's program.

FORMAT

G [address 1] <CR>

OPERATION

To use this command, enter G when prompted for command entry followed by <CR>. Execution starts from the PC value.

Now if you wish to modify the value of the PC (i.e the address to which the control is to be transferred), enter the new value. Enter <CR>. Now the user content is restored and control is transferred to the program starting at the current value of the user program counter.

A powerful debugging tool breakpointing a program is available to the user. To use this facility set one or more breakpoints in program or data memory using B command.

Now the control is transferred to the program starting at the current PC value, Upon reaching any one of the specified breakpoint addresses control is returned to the monitor. Monitor saves the present register contents and PC value, displays the current PC value and then issues a command prompt.

Notes :

1) When any one of the breakpoints is reached, control is returned to the monitor, after saving the registers.
2) Specifying more than one breakpoint address is useful when debugging a program section containing branch instructions.

Examples

Example1: Suppose the following program has been entered in the program memory .

ADDRESS	OBJECT	COMMENTS
8800	74	MOV A,#42H
8801	42	
8802	F9	MOV R1,A
8803	80	SJMP 8800H
8804	FB	

>G8800 <CR>

The program can be executed by setting up breakpoints;

The procedure is as follows :

1) Set a breakpoint in the program memory or data memory using
   BSP address <CR> or BSD address <CR>
2) If the desired breakpoints are in a range, then enter BSP addr1, addr2 <CR> or BSD addr1, addr2 <CR>
3) Enable breakpoints using EP command or ED command for program or data memory respectively.

4) Execute the program using G command (Note: single step execution of program memory disables breakpoint memory).
5) Cause of break with the program break address are displayed.
6) Enter <CR> to continue. The program starts executing from the point at which break has occurred.
7) The above procedure is repeated if program encounters another breakpoint.
8) Enter <Esc> to terminate the process.

3.3.9 S (SINGLE STEP COMMAND)

The ESA 51 trainer enables you to debug a program by single stepping the instructions. The command is used to execute a program one instruction at a time. With each instruction executed, control is returned to the monitor. Thus this command is an extremely useful debugging tool. Provision has been made for single stepping with disassembly.

3.3.9.1 S (SINGLE STEP COMMAND WITH DISASSEMBLY)

FUNCTION

This command is used to single step a program with disassembly. The register content will not be displayed.

FORMAT

>S [addr] <CR>

OPERATION

To use this command enter S when prompted for command. Only executed instruction in disassembled format are displayed and register contents are not displayed.

>S8800 <CR>

8802 MOV R1, A

3.3.10 B (BREAKPOINT) COMMANDS

BREAKPOINTS

The ESA 51 enables you to control program execution by setting break points. A breakpoint is an address that stops program execution each time the address is encountered. By setting breakpoints at key addresses in your program. You can stop program execution and examine the status of memory or registers at that point.

These commands are used to set breakpoint, clear breakpoint and display breakpoint in both program memory and data memory, enables and disables breakpoints in both the memories independently. The breakpoints can be one or more and also user can specify range of address for breakpoints.

3.3.10.1 CLEAR BREAKPOINT

FUNCTION

To clear the breakpoint(s) in the data memory or program memory.

FORMAT

BC{P|D}addr1[,addr2]

OPERATION

To clear the breakpoints enter BCP or BCD for corresponding memory and with one address or set of addresses followed by <CR>.

Example : To clear all the breakpoints in program memory.

>BCP 0, FFFF <CR>

After clearing all the breakpoints system waits for the next command entry with command prompt.

3.3.10.2 SET BREAKPOINT

FUNCTION

The set break point command is used to set breakpoints in program memory and data memory.

FORMAT

BS {P|D} addr 1 [,addr2]

OPERATION

1. Set a breakpoint in the program memory or data memory using BSP <addresses> <CR>
2. If the user wants to set more no. of break points, then enter BSP addr1, addr2 <CR>

Example

To set a breakpoint in the program memory at address 000BH enter the following command

>BSP 000B <CR>

3.3.10.3 DISPLAY BREAKPOINT

FUNCTION

To display the breakpoint which has been set using BSP or BSD Command, enter BDP or BDD for program memory or data memory respectively with <CR>.

FORMAT

BD{P|D}<CR>

OPERATION

1. Enter the command BDP or BDD and <CR> to display the preset breakpoints.
2. Enter <CR> to view remaining preset breakpoint(s) or terminate the command with <Esc>.

3. ‘NO breakpoints found’ message is displayed if no breakpoints are set.

Example

To display the breakpoint in the program memory which has been set in previous command.

>BDP <CR>

000B <Esc>

The above address is displayed to indicate that breakpoint at address 000BH is set in the program memory.

3.3.10.4 ENABLE BREAKPOINT

FUNCTION

To enable the breakpoint which has been set using BSP or BSD command

FORMAT

E {P|D} <CR>

OPERATION

Enter the command EP or ED and <CR> to enable the preset breakpoint(s).

Example:

To enable the breakpoint in the program memory which has been set in previous command

EP <CR>

>

3.3.10.5 DISABLE BREAKPOINT

FUNCTION

To disable the breakpoint which has been set using BSP or BSD command

FORMAT

D {P| D} <CR>

OPERATION

Enter the command DP or DD and <CR> to disable the preset breakpoint(s).

Example

To disable the breakpoint in the data memory

DD<CR>

>

Disable breakpoint does not clear the breakpoints.

3.3.11 A (ASSEMBLY) COMMAND

ESA 51 provides the powerful, PROM resident assembler to enhance development work. This assembler is an on-line one and supports the entire standard mnemonics and addressing modes of Intel 8031 / 8051 microcontrollers.

FUNCTION

The assembler generates the actual machine codes and stores them in the memory locations defined by the program. Also, the system will display the codes generated as well as the source statement. Any errors detected are also displayed on the screen.

OPERATION

'A' command invokes the assembler, A with optional address when prompted for the command .

A [address] <CR>

Assembly language instructions consist of 3 field, as shown below

Address Object Mnemonic

The fields may be separated by any number of blanks and tabs but must be separated by atleast one delimiter. Each instruction must be entered on a single line terminated by <CR>. No continuation lines are possible.

Opcode Field:

This required field contain mnemonic operation code for the 8051 instruction to be performed.

Operand Field:

The operand field identifies the data to be operated on by specified Opcode. Some instructions require no operands. Other require one, two or three operands. As general rule, when 2 operands are required (as in data transfer and arithmetic operations), the first operand identifies the destination (or target) of the operations result and the second operand specifies the source data and the two operands must be separated by a comma.

Examples

```
>A8000 <CR>
8000          E9          MOV A, R1 <CR>
8001          74 42       MOV A, # 42H <CR>
8003                          <Esc>
>
```

3.3.12 Z (DISASSEMBLER) COMMAND

Disassembly is an extremely useful feature, often employed during debugging.

FUNCTION

A Disassembler converts machine language codes into assembly language mnemonics, making it easy for user to understand/verify the program .

OPERATION

To use this facility, type Z when prompted for command by the Serial Monitor.

>Z [addr1[,addr2]] <CR>

Address Object Mnemonic

The disassembled code is displayed according to the above format.

NOTE: If the disassembly of the last instruction requires the reading of data from locations beyond the specified address2, the system will read them to complete the disassembly. For example, if the specified address2 is 81FFH and the code at 81FFH is 20H (which is a 3 – byte instruction), the system will read the required data from location 8200H and 8201H to complete the disassembly.

Example :

>Z 0,6<CR>

0000	02 00 30	LJMP 0030H
0003	02 2A BE	LJMP 2ABEH
0006	FF	MOV R7, A

>

3.3.13 H (HELP) COMMAND

FUNCTION

The HELP command is used to list all the commands supported by the Serial Monitor.

FORMAT

H <CR>

OPERATION

As soon as H and <CR> is entered by the user, in response to the command prompt the system lists, in alphabetical order all the commands supported by the serial monitor. The display appears as shown below :

H <CR>

Command	**Syntax**
Assemble	A [address]
Breakpoint	B {[D{P|D}]|[{C|S} {P|D} address1[,address2]]}
Compare Memory	C {P|D|I} address1, address2, {P|D|I}, address3
Disable Brkpnt	D {P|D}
Enable Brkpnt	E {P|D}
Fill / Search MEM	F {P|D|I},address1, address2,data [,data [,data [,data]],S]
Go (Execution)	G [address]
Help	H
Jump	J [address]
Memory	M {P|D|I} address1[,address2[,{P|D|I},address3]]
N (SS Count)	N count (Single Step Count = <0FFH>)
Register	R [Register]
Single Step	S [R] [address]
Z – Disassembly	Z [address1 [, address2]]

CHAPTER 4

SERIAL MONITOR

4.1 INTRODUCTION

This chapter describes the commands supported by the serial monitor program. The serial monitor allows ESA 51 to be operated from a host computer connected via the RS-232-C/RS 485 serial interface. (refer to chapter on Hardware and Appendix C on RS-232-C/RS 485 connector details).

The system must be configured for serial mode of operation as described in section 2.1.1. The commands are described in this chapter.

When the system enters serial mode of operation, the sign - on message "ESA-8051 Serial Monitor V x.y" is displayed (x is the current version number and y is the revision number) on one line and prompt ">" on the next line indicating that the monitor is ready to accept commands from the user.

4.2 STRUCTURE OF MONITOR COMMANDS

Whenever the monitor is ready to accept a command from the user, it outputs a greater than symbol (` >`) as command prompt character at the beginning of a new line.

The commands entered by the user consist of a single character command mnemonic followed by a list of command parameters. This list may consist of upto four parameters depending on the particular command being used. When more than one parameter is required, a single (`,`) or space is used between the parameters as a separator.

A command is terminated by a <CR>. Commands are executed one at a time and only one command is allowed within one command line.

PARAMETER ENTRY

All numeric parameters are to be entered as hexadecimal number. The valid range for one-byte parameters is 00 to FF and if more than 2 digits are entered, only the last two digits are valid (leading zeros may be omitted). Thus all one byte values are interpreted modulo 256 (decimal). The valid range for 2 byte parameter is 0000 to FFFF and longer values are evaluated modulo 64K (i.e. only the last four digits are valid).

All the commands except the R (examine/modify register) command require only hexadecimal values as parameters. The register name abbreviation entries required by the R command are described later while describing the R command in detail.

RESPONSE TO ERRORS

Whenever an error is detected by the monitor (either in the command entry or in the command execution) the command is aborted and "Error" is displayed on the next line with a "^" sign attached pointing to the place where the error occurred and a new command prompt is issued (The possible error conditions are described while illustrating the individual commands).

Command execution occurs only after a valid delimiter (a <CR>) is entered. Hence a command entry can be cancelled anytime before the delimiter is entered by pressing <Esc>. The command prompt character is output on a new line.

4.3 MONITOR COMMANDS

Each command described in this chapter consists of one to two character, followed by appropriate parameters and data. The commands are summarized in Table 4.1 and are described in detail in the section which follows. In the table as well as in the subsequent description, the following notation is used:

NOTATIONAL CONVENTIONS

SYMBOL	NAME	USAGE
{ }	Curly braces with Ellipsis	Encloses a required argument 1 or more Times
[ ]	Square brackets	Encloses an item that appears 0 or 1 time.
[ ]…	Square brackets	Encloses an item that appears 0 or more times.
|	Vertical bar	Separates alternative items in a list
	Italics	Indicates a descriptive item that should be Replaced with an actual item.

TABLE 4.1 SUMMARY OF SERIAL MONITOR COMMANDS

COMMAND	FUNCTION /FORMAT
A	Assembler A [address]
B	Clear/Display/Set/Breakpoint in Program memory, external Data memory B{[D{P|D}]|[{C|S}{P|D} address 1 [,address 2]]}
C	Compare a block of memory with destination block C{P|D|I {address 1, address 2, {P|D|I}, Address 3}}
D	Disable breakpoint in program memory or data memory D{P|D}
E	Enable breakpoint in program memory or data memory E{P|D}
F	Fill a block of memory with a constant or search a string of Data in program memory, external data memory and internal data memory. F {P|D|I}, address 1,address 2,data [,data[,data[,data]]],S
G	Transfer the processor control from the monitor to user Program G [address]
H	Help. List all the commands supported by the Serial Monitor H
J	Jump to address J address
M	Modify/Display/Move memory contents in program memory, External data memory and internal data memory with all combinations M{P|D|I} address 1 [,address 2 [,{P|D|I}, address 3]]
N	Execute one or more instructions specified by user. N {count}
P	Programmer
S	Execute one instruction of user program S[R] [address]

Z	Disassembler Z[addr1[,addr2]]
U & V	Commands are reserved for system usage.

4.3.1 M (MODIFY) COMMAND

FUNCTION

The M (Modify Memory) command is used to examine the contents of specified memory locations. Further if location are in RAM their contents can be altered if desired and block move contents of memory from program, data or internal memory to program, data or internal memory for all combinations.

FORMAT

M {P|D|I} address 1 [,address 2 {[,P|D|I}, address 3]]

OPERATION

1. Enter M followed by memory type, the address of the memory location to be examined and then enter <CR>. The monitor will now output the contents of that location. Note that in serial mode a `-` is always a prompt for data entry, while a ">" is the prompt for command entry.
2. To modify the contents of this location, the user can enter the new value now.
3. Enter a <CR>, either immediately after the '-' prompt by the system or after the entry of a new value, to examine/modify the next sequential location. A "<Esc>" instead of the <CR> terminates the command and returns the monitor to the command entry mode.

ERROR CONDITION

1. Trying to modify the contents of non-existent or PROM location.

Example 1 : Examine the PROM locations 11H

>MP11<CR>

>0011 FF <Esc>

>

Example 2 : Examine a series of RAM location starting at 8820H and modify the contents of the location 8822H

>MD8820<CR>

8820 xx <CR>

8821 xx <CR>

8822 xx 55<CR>

8823 xx <Esc>

>

4.3.2 M (DISPLAY MEMORY) COMMAND

FUNCTION

This command is used to display the contents of the program memory, external or internal data memory.

FORMAT

M {P|D|I}, address 1, address 2 <CR>

OPERATION

1. To use this command, enter M when prompted for command entry. After entering M, enter the memory type and then starting address of the memory block whose contents are to be displayed, then enter a comma, enter the end address of the memory block and follow it with a <CR>.
2. Now the monitor will output the starting address, the contents of the location from this address to the specified end address. The display appears in formatted lines with 16 bytes/line. The number of bytes displayed on the first line are so adjusted that if the second line is present, its first location has address with the last nibble as zero. The ASCII equivalent of the displayed data values are also shown on each line. The non-displayable character are shown as periods (“.”).

Examples

Example 1 : To display the contents of 5 bytes from location 8000H.

```
>MD8000, 8004
       0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F ASCII
8000:41  42 43  0D 31                                ABC.1
>
```

4.3.3 M (MOVE MEMORY) COMMAND

FUNCTION

This command is used to move a block data from one area of the memory to another area.

FORMAT

M {P|D|I}, <address1>, <address2>, {P|D|I}<destination address> <CR>

OPERATION

1. To use this command, enter M when prompted for command entry. Follow it with the type of memory starting address of the source block to be moved ("address1"), a comma, the ending address of the source block ("address2"), another comma, and then type of destination memory, starting address of the area into which the source block is to be moved ("destination address"). Now enter <CR>.
2. This operation moves the contents of memory location from "address 1" to "address 2" to consecutive memory location starting from "destination address".
3. The system determines if there is any overlap between source and destination block and accordingly transfer the data beginning either at the " address 1" or at the address 2" .

ERROR CONDITIONS

1. Specifying an "end address" value, which is less than the value of the "start address".
2. Trying to move data into non-existent or read-only memory location.

Examples :

Example 1 : Move the contents of the location 800H through 80FH to the memory block beginning at 8840H

>MP800,80F,P8840<CR>

>

Example 2 :

>MP800,80F,P200<CR>

Error

An attempt to move data into PROM location produces the error message.

4.3.4 F (FILL MEMORY) COMMAND

FUNCTION

This command is used to fill a block of memory with specified constant.

FORMAT

F{P|D|I}, address1, address2, constant <CR>

OPERATION

1. To use this command enter F when prompted for command entry and enter the type of memory to be filled.
2. Now enter the starting address of the block of memory to be filled. Enter a comma. Now enter the ending address of the block. Again enter a comma. Now enter the constant. Press the <CR> to start the command execution.

3. Monitor now fills the block of memory from address1 to address 2 with the specified constant. Then the monitor displays the command prompt sign.

ERROR CONDITIONS

1. Specifying a value for the address 2 of the source block which is less than the value of the address1, of the source block.
2. Trying to fill the data in non – existent or read – only memory.

EXAMPLES :

Example1 : Filling the block of program memory with a constant 55H.

>FP8800,880F,55<CR>

>

Now you can use the M command to examine the block of memory to see that it is filled with the constant 55H.

4.3.5 C (COMPARE MEMORY) COMMAND

FUNCTION

Compare command can be used to compare the contents of one memory block with the contest of another memory block.

FORMAT

C {P|D|I} address 1 of block 1, address2 of block 1

{P|D|I} address 1 of block 2 <CR>

OPERATION

1. To use this command, enter C when prompted for command entry and select the type of memory. Then enter starting address of the first block, a comma , ending address of the first block, another comma and destination type of memory and coma and then the starting address of the second block followed by a <CR>.
2. The monitor now compares the content of location beginning at address1 of block1 with the content of location beginning at address 1 of block2. This process continues till the contents of address 2 are compared with those of corresponding location in the 2^{nd} block. Any differences detected are displayed.

Examples :

1) Compare the contents of memory locations 8000H to 8FFFH with those of a memory block beginning at 9000H

>CP8000,8FFF,9000<CR>

>

(This response showed that there is no mismatch)

2) CPA000,AFFF,P8000<CR>

ABC0 = 00 FF = 8BC0

AED8 = 48 54 = 8ED8

(This response showed that there is mismatch at two locations).

4.3.6 R (EXAMINE/MODIFY REGISTER) COMMAND

FUNCTION

This command is used to examine and optionally modify the contents of the registers.

FORMAT

R [reg] [[[new data] ,]…]<CR>

OPERATION

1. To examine the contents of all the registers, enter R followed by <CR> when prompted for command entry. The monitor will now display the content of all registers.
2. If you wish to examine/modify the contents of a particular register, Enter R (when prompted for command) followed by the register name abbreviation are shown in Table 4.2. Now the monitor will output an equal sign ('=') the current contents of the specified register and data prompt character ("-"). The content of this register can be changed now by entering the new data value, followed by a valid terminator (a <CR> or <Esc>). If the terminator is <Esc>, the command is terminated.

   If the terminator is not <Esc> the next "sequential" register is displayed and opened for optional modification. The sequence in which registers are displayed is shown in Table 4.2 (Note that this sequence is circular).

TABLE 4.2

Register name	Abbreviation
Register A	A
Register B	B
Stack Pointer	SP
Flags Register	PSW

Data Pointer High	DPH
Data Pointer Low	DPL
Register TH0	TH0
Register TL0	TL0
Register TH1	TH1
Register TL1	TL1
Register P1	P1
Register P3	P3
Register Counter High	PCH
Register Counter Low	PCL
Register R0	R0
Register R1	R1
Register R2	R2
Register R3	R3
Register R4	R4
Register R5	R5
Register R6	R6
Register R7	R7

NOTE: The flag register PSW is also displayed in bit format when register command is executed. The meaning of the pattern "CAFBBOGP" is as follows.

PSW bit	**Abbreviated as**	**Functions**
PSW. 7	C	Carry Flag
PSW. 6	A	Auxiliary Flag
PSW. 5	F	Flag 0 available to the user
PSW. 4	B	Register Bank selector bit 1
PSW. 3	B	Register Bank Selector bit 0
PSW. 2	O	Overflow flag
PSW. 1	G	Usable as a general purpose flag
PSW. 0	P	Parity flag

Examples :

Example 1

>R<CR>

A	B	SP	PSW	DPH	DPL	TH0	TL0	TH1	TL1	P1	P3	PCH	PCL
(E0)	(F0)	(81)	(D0)	(83)	(82)	(8C)	(8A)	(8D)	(8B)	(90)	(B0)		
00	00	07	00	00	00	00	00	00	00	FF	FF	00	00

R0	R1	R2	R3	R4	R5	R6	R7	PSW
(00)	(01)	(02)	(03)	(04)	(05)	(06)	(07)	C A F B B O G P
00	00	00	00	00	00	00	00	0 0 0 0 0 0 0 0

Example 2

Examine and alter register A and then examine register B.

RA <CR>

A(E0) 24 <CR>

B(F0) 25<Esc>

4.3.7 J (JUMP TO ADDRESS-SET/CHANGE PC) COMMAND

FUNCTION

The J Command is used to change the program counter value to the desired address before executing a program by either GO command or SINGLE STEP command.

FORMAT

J [address] <CR>

OPERATION

To use this command, enter J when prompted for command entry. Now enter the desired starting address of the program you wish to execute. Enter <CR>. Now command prompt reappears on the next line.

4.3.8 G (GO) COMMAND

FUNCTION

The GO command is used to transfer the control of the system from monitor to the user's program.

FORMAT

G [address 1] <CR>

OPERATION

To use this command, enter G when prompted for command entry. Execution starts from the PC value.

Now if you wish to modify the value of the PC (i.e the address to which the control is to be transferred), enter the new value. Enter <CR>. Now the user context is restored and control is transferred to the program starting at the current value of the user program counter.

A powerful debugging tool breakpointing a program is available to the user. To use this facility set one or more breakpoints in program or data memory using B command.

Now the control is transferred to the program starting at the current PC value, upon reaching any one of the specified breakpoint addresses control is returned to the monitor. Monitor saves the complete user context, displays the current PC value and then issues a command prompt.

NOTES :

1. When any one of the breakpoints is reached, control is returned to the monitor, after saving the registers.
2. Specifying more than one breakpoint address is useful when debugging a program section containing branch instructions.

EXAMPLE

Enter the program presented as example 1 for the GO command from the stand-alone mode monitor (section 3.3.8).

>G8800 <CR>

The program can be executed by setting up breakpoints;

The procedure is as follows :

1) Set a breakpoint in the program memory or data memory using BSP address <CR> or BSD address <CR>
2) If the desired breakpoints are in a range, then enter BSP addr1, addr2 <CR> or BSD addr1, addr2 <CR>
3) Enable breakpoints using EP command or ED command for program or data memory respectively.
4) Execute the program using G command (Note: single step execution of program memory disables breakpoint memory).
5) Cause of break with the program break address are displayed.
6) Enter <CR> to continue. The program starts executing from the point at which break has occurred.
7) The above procedure is repeated if program encounters another breakpoint.
8) Enter <Esc> to terminate the process.

4.3.9 S (SINGLE STEP COMMAND)

The ESA 51 trainer enables you to debug a program by single stepping the instructions. The command is used to execute a program one instruction at a time. With each instruction executed, control is

returned to the monitor. Thus this command is an extremely useful debugging tool. Provision has been made for single stepping with register display, disassembly and count.

4.3.9.1 SR (SINGLE STEP WITH REGISTER DISPLAY) COMMAND

FUNCTION

This command is used to single step a program with register display.

FORMAT

SR [addr] <CR>

OPERATION

1) To use this command, enter SR when prompted for command.
2) To execute one instruction at the current value of the program counter, press <CR>, when this key is pressed, the instruction at the current PC value is executed and then all the register values are displayed.
3) To execute one instruction at the desired value of PC, enter SR and desired starting address of the program and then press the <CR>. Press <CR> whenever you want to execute one instruction at a time. Each time <CR> is pressed one instruction is executed. To terminate the command press <Esc>.

Example 1 : Suppose the program given as Example 1 to illustrate the GO command has been entered in the memory. Now this program can be single stepped as follows.

```
>SR 8800 <CR>
 A     B     SP   PSW    DPH    DPL    TH0    TL0    TH1    TL1     P1    P3    PCH    PCL
(E0) (F0) (81)  (D0)    (83)   (82)   (8C)   (8A)   (8D)   (8B)   (90)  (B0)
 00    00   07     01      00      00      00      00      00      00     FF   FB     88      02
  R0     R1    R2     R3     R4     R5     R6   R7              PSW
 (00)   (01)  (02)     (03)   (04)    (05)   (06) (07)        CAFBBOGP
  00     00     00      00     00      00     00   00          00000001
  8802           F9                  MOV R1,A
>
```

4.3.9.2 S (SINGLE STEP COMMAND WITH DISASSEMBLY)

FUNCTION

This command is used to single step a program with disassembly. The register content will not be displayed.

FORMAT

>S [addr] <CR>

OPERATION

1. To use this command enter S when prompted for command. Rest of the procedure is same as for the SR command. Only executed instruction in disassembled format are displayed and register contents are displayed.

>S8800 <CR>

8802 MOV R1, A

4.3.9.3 N (SINGLE STEP WITH COUNT)

This command is used to single step a program with count. The maximum value of the count is FF. Multiple instruction can be executed at a time.

FORMAT

N (count) <CR>

OPERATION

1. To use this command, set the PC value to the starting address of the address of the program using J command.
2. Enter N, the count and press <CR>.

If the program starting address is 9000H and the count is 20 instruction at a time, following commands have to be executed.

>J9000 <CR>

>N 14 <CR>

Now 20 instruction will be executed at a time. The register contents are displayed just like in SR command. Now the user can continue single stepping by pressing <CR> or the user can exit from the command by pressing <Esc>.

4.3.10 B (BREAKPOINT) COMMANDS

BREAKPOINTS

The ESA 51 enables you to control program execution by setting break point. A breakpoint is an address that stops program execution, each time the address is encountered. By setting breakpoints at key addresses in your program. You can “freeze” program execution and examine the status of memory or registers at that point.

These commands are used to set breakpoint, clear breakpoint and display breakpoint in program memory a well as data memory and to enable and disable breakpoints in both the memories independently. The breakpoints can be one or more and also user can specify range of address for breakpoints.

4.3.10.1 CLEAR BREAKPOINT

FUNCTION

To clear the breakpoint(s) in the data memory or program memory.

FORMAT

BC{P|D}addr2[,addr2]

OPERATION

To clear the breakpoints enter BCP or BCD for corresponding memory and with one address or range of address and <CR>.

Example : To clear the breakpoint of full program memory, enter

>BCP 0, FFFF <CR>

After clearing procedure is finished a command prompt is displayed.

4.3.10.2 SET BREAKPOINT

FUNCTION

The set break command is used to set breakpoints in program memory and data memory.

FORMAT

BS {P|D} addr 1 [,addr2]

OPERATION

1. Set a breakpoint in the program memory or data memory using BSP <addresses> <CR>
2. If the desired breakpoint is a range, then enter BSP addr1, addr2 <CR>

Example

To set a breakpoint in the program memory at address 000BH enter the following command

>BSP 000B <CR>

4.3.10.3 DISPLAY BREAKPOINT

FUNCTION

To display the breakpoint which has been set using BSP or BSD Command, enter BDP or BDD for program memory or data memory respectively with <CR>.

FORMAT

BD{P|D}<CR>

OPERATION

1. Enter the command BDP or BDD and <CR> to display the preset breakpoints.
2. Enter <CR> to view remaining preset breakpoint(s) or terminate the command with <Esc>.
3. 'NO breakpoints found' message is displayed if no breakpoints are set.

Example

To display the breakpoint in the program memory which has been set in previous command.

>BDP <CR>

000B <Esc>

The above address is displayed to indicate that breakpoint at address 000BH is set in the program memory.

4.3.10.4 ENABLE BREAKPOINT

FUNCTION

To enable the breakpoint which has been set using BSP or BSD command

FORMAT

E {P|D} <CR>

OPERATION

Enter the command EP or ED and <CR> to enable the preset breakpoint(s).

Example:

To enable the breakpoint in the program memory which has been set in previous command

EP <CR>

4.3.10.5 DISABLE BREAKPOINT

FUNCTION

To disable the breakpoint which has been set using BSP or BSD command

FORMAT

D {P| D} <CR>

OPERATION

Enter the command DP or DD and <CR> to disable the preset breakpoint(s).

Example

To disable the breakpoint in the data memory

DD<CR>

>

Disable breakpoint does not clear the breakpoints.

4.3.11 A (ASSEMBLY) COMMAND

ESA 51 provides the powerful, PROM resident assembler to simplify the user's task of program development. This assembler, available in serial mode of operation, is an on-line one and supports the entire standard mnemonics and addressing modes of Intel 8031 / 8051 microcontroller.

FUNCTION

The assembler generates the actual machine codes and stores them in the memory locations defined by the program. Also, the system will display the codes generated as well as the source statement. Any errors detected are also displayed on the screen.

OPERATION

'A' command implements the assembly facility. So, to invoke the assembler, type A with optional address when prompted for the command by the serial monitor.

A [address] <CR>

Assembly language instructions consist of 3 field, as shown below

Address Object Mnemonic

The fields may be separated by any number of blanks and tabs but must be separated by atleast one delimiter. Each instruction must be entered on a single line terminated by <CR>. No continuation lines are possible.

Opcode Field:

This required field contain mnemonic operation code for the 8051 instruction to be performed.

Operand Field:

The operand field identifies the data to be operated on by specified opcode. Some instructions require no operands. Other require one, two or three operands. As general rule, when 2 operands are required (as in data transfer and arithmetic operations), the first operand identifies the destination (or target) of the operations result and the second operand specifies the source data and the two operands must be separated by a comma.

Examples

>A8000 <CR>

Address	Opcode	Mnemonic
8000	E9	MOV A, R1 <CR>
8001	74 42	MOV A, # 42H <CR>
8003		<Esc>

>

4.3.12 Z (DISASSEMBLER) COMMAND

Disassembly is an extremely useful technique, often employed during debugging.

FUNCTION

A Disassembler converts machine language codes into assembly language mnemonics, making it easy for user to understand/verify the program .

OPERATION

To use this facility, type Z when prompted for command by the Serial Monitor.

>Z [addr1[,addr2]] <CR>

Address Object Mnemonic

The disassembled code is displayed according to the above format.

The display can be halted at any point by **Ctrl – S** and restarted by **Ctrl – Q.**

NOTE: If the disassembly of the last instruction requires the reading of data from locations beyond the specified address2, the system will read them to complete the disassembly. For example, if the specified address2 is 81FFH and the code at 81FFH is 20H (which is a 3 – byte instruction), the system will read the required data from location 8200H and 8201H to complete the disassembly.

Example :

>Z0,6 <CR>

Address	Object	Mnemonics
0000	02 00 30	LJMP 0030H
0003	02 2A BE	LJMP 2ABEH
0006	FF	MOV R7, A

>

4.3.13 H (HELP) COMMAND

FUNCTION

The HELP command is used to list all the commands supported by the serial monitor.

FORMAT

H <CR>

OPERATION

As soon as H and <CR> is entered by the user, in response to the command prompt the system lists, in alphabetical order all the commands supported by the serial monitor. The display appears as shown below :

H <CR>

Command	Syntax
Assemble	A [address]
Breakpoint	B {[D{P|D}]|[{C|S} {P|D} address1[,address2]]}
Compare Memory	C {P|D|I} address1, address2, {P|D|I}, address3
Disable Brkpnt	D {P|D}
Enable Brkpnt	E {P|D}
Fill / Search MEM	F {P|D|I},address1, address2,data [,data [,data [,data]],S]
Go (Execution)	G [address]
Help	H
Jump	J [address]
Memory	M {P|D|I} address1[,address2[,{P|D|I},address3]]
N (SS Count)	N count (Single Step Count = <0FFH>)
Register	R [Register]
Single Step	S [R] [address]
Z – Disassembly	Z [address1 [, address2]]

U & V Commands are reserved for System usage.

CHAPTER 5

HARDWARE

5.1 INTRODUCTION

This chapter describes the hardware design details of ESA 51. Appendix F gives the connector details and Appendix A has the component layout diagram. The design details are discussed in the following order:

a) CPU, Address Bus, Data Bus and Control Signals
b) Memory Addressing.
c) Keyboard/Display Interface.
d) Programmable Peripherals and Serial Interface.
e) ADC & DAC Section.
f) Bus Expansion.
g) Connector Details.

5.2 CPU, ADDRESS, DATA AND CONTROL SIGNALS

ESA 51 uses 8031/8051 Microcontroller operated with 11.0592 crystal. The on-board RESET key can provide a RESET signal to the CPU. The lower address bus is demultiplexed using a 74LS373 at U3 and the upper address bus is demultiplexed using 74LS373 at U22. The data bus is buffered using a 74LS245 at U7. All these buffered signals are available on the system connector J1 (Connector details are given at the end of this chapter).

5.3 MEMORY ADDRESSING

ESA 51 has four 28 pin JEDEC compatible slots (U9,U10,U11,U12) accepting memory devices. The socket U9 populated with a 27256 as program memory which contains the system firmware. The socket at U10 is populated with a 62256 to provide 32K bytes of static RAM of user program memory.
The sockets at U11 & U12 are populated with 62256's to provide 56K bytes of static RAM of user data memory.
The memory mapping is as follows:

TABLE 5.1 MEMORY MAP

Devices	Address range	Type of memory
27256 at U9	0000-7FFF	Program memory
62256 at U10	8000-FFFF	User program memory
62256 at U11	0000-7FFF	User Data memory
62256 at U12	8000-DFFF	User Data memory

Battery Option

The 62256 provided at U10,U11 and U12 can be backed up by an optional battery. The terminal for connecting the battery are brought out as BT.

5.4 I/O ADDRESSING

I/O decoding is implement using a E561, E562, E563, E564 and E622. Thus fold-back exists over the unused address lines. The I/O devices, their addresses and their usage is summarized below:

TABLE 5.2 I/0 ADDRESS MAP

I/O Device	**Address**	**Usage**
8255 : 1 at U42		Available to user.
Port A	E800H	
Port B	E801H	
Port C	E802H	The signal are available on connector J10.
Control Port	E803H	

8255 A : 2 U18		Available to user
Port A	E804H	
Port B	E805H	The signal are available on connector J7.
Port C	E806H	
Control Port	E807H	
8255 : 3 at U30		
Port A	E900H	Used for LCD data lines.
Port B	E901H	Used for printer.
Port C	E902H	Used for LCD control lines & printer hand shaking signals
Control Port	E903H	Configured by system firmware
8253-5 at U8		Available to user
Timer 0	EA00H	Timer 0 is available to user on connector J1.
Timer 1	EA01H	Timer 1 is available to user on connector J1.
Timer 2	EA02H	Timer 2 is available to user on connector J1.
Control Port	EA03H	
DIP Switch	E904H	Used by system for Serial / keyboard operation & printer.
2681 at U13		Used for implementing Serial communication
(addressing details are given in section 5.8)		
8042 at U31		Used for implementing keyboard/display interface.
Data Port	EC00H	
Command Port	EC01H	
8155 : at U29		Reserved for system
Internal RAM	E000H-E0FFH	Reserved for system

Command/Status	E100H	
Port A	E101H	ADC Data input
Port B	E102H	DAC data output
Port C	E103H	ADC data input
Timer Regs	E104H (Lower) & E105H (Higher)	

5.5 8042 UNIVERSAL PERIPHERALS INTERFACE

ESA 51 Trainer is interfaced with a PC keyboard and LCD for operation in stand-alone mode. Keyboard interface is controlled by an 8042 Universal Peripheral Interface microcomputer. The UPI allows the user to develop customized solution for peripheral device control.

The addressing information for 8042 UPI onboard ESA 51 is given earlier in this chapter. The UPI uses clock inputs from the 11.0592 MHz crystal oscillator. The keyboard reading is implemented by polling the command/status port of 8042. User can read the keyboard in polling mode by checking the status of output buffer register. The keyboard sends scan codes for the respective keys pressed. The scan codes for the keys can be referred in the PC AT reference manual. The UPI is programmed for encoding either 101 or 84 keys PC keyboard.

5.6 LCD INTERFACE

In the stand-alone mode, an LCD is used as an output terminal for working with ESA 51 Trainer. The display is initialized as follows:

20 Digits, 4 Lines, Left entry display.

LCD Module comprises two register. Instruction and Data register. Three control signals RS,R/W, and E determine the operating status of LCD.

E	=	1	:	For any operation with the LCD.
RS	=	1	:	Operation with Data register
R/W	=	0	:	Operation with instruction register
R/W	=	1	:	Read from LCD

The control and data lines to the LCD are provided by 8255 at U30. These lines are brought to the flow-strip at J8 where the LCD is inserted.

5.7 PROGRAMMABLE INTERVAL TIMER

ESA 51 has on-board programmable interval timer 8253-5 at socket position U8. Its I/O address can be found in Table 5.2 in section 5.4.
8253 has one command/status port and three data ports called Timer0, Timer1 and Timer2 to provide three programmable timers. All the timers are available for user. The signal related to timer 0,1 and 2 are available on system connector J1. Connector details are provided in the last section of this chapter.

5.8 SERIAL INTERFACE

Serial communication in ESA 51 trainer is implemented using SCN 2681. Dual Universal Asynchronous Receiver Transmitter (DUART). The trainer is capable of communicating with serial terminal using either RS-232-C standards or RS 485 standards.

SCN 2681 DUART provides two independent full duplex synchronous receiver / transmitter in a single package. It interfaces directly with microprocessors and may be used in a polled or interrupt driven system. The operating mode and data format of each channel can be programmed independently, and each receiver and transmitter can select its operating speed as one of eighteen fixed baud rates, a 16 X clock derived from a programmable counter/timer, or an external 1 X or 16 X clock.

ESA 51 provides a 3.6864 MHz crystal as a clock input to the DUART for its operation. channel A of the DUART is programmed for serial communication using RS-232-C standards while channel B is programmed for RS 485 communication. The interfacing requirements and connector details for either mode of operation have been discussed earlier in the manual. The baud rate selection for serial communication is made by DIP switch settings and different standards baud rates can be established. Refer the chapter on configuration and installation for the look up table for this selection.

The following table gives the addressing information with respect to the various control registers of SCN 2681 DUART. The user may program the serial controller interfacing using this information to suit his needs.

SCN 2681 Register Description & Addressing

Address	Read Operation (RDN=0)	Write Operation (WRN =0)
EB00	Mode Register A (MR1A,MR2A)	Mode register A (MR1A, MR2A)
EB01	Status Register A (SRA)	Clock select register A (CSRA)
EB02	Reserved	Command register A (CR)
EB03	Rx holding register A (RHRA)	Tx holding register A (THRA)
EB04	Input Port change register (IPCR)	Auxiliary control register (ACR)
EB05	Interrupt status register (ISR)	Interrupt Mask register (IMR)
EB06	Counter/Timer Upper (CTU)	C/T upper register (CTUR)
EB07	Counter/Timer Lower (CTL)	C/T Upper lower register (CTUL)
EB08	Mode register B (MR1B, MR2B)	Mode register B (MR1B, MR2B)
EB09	Status register B	Clock select register B (CSRB)
EB0A	Reserved	Command register B (CBR)
EB0B	Rx holding register B (RHRB)	Tx holding register B (THRB)
EB0C	Reserved	Reserved
EB0D	Input Port Register	Output Port configuration register (OPCR)
EB0E	Start counter command	Set Output port Bits commands
EB0F	Start counter command	Reset Output Port Bits commands

Separate serial connections are provided on the interfacing for interfacing with RS 232C and RS 485 ports.

J4 : 9-pin female type connector for RS 232C communication.

J5 : 9-pin male type connector for RS 485 communication.

To establish serial communication using ESA 51 the user has to interface the serial terminal with either of these connector using appropriate serial cable.

5.9 PROGRAMMABLE PERIPHERAL INTERFACE DEVICES

ESA 51 has two numbers of 8255As (Programmable Peripheral Interface Devices). Each 8255A consists of a command port and three 8-bit programmable input/output ports called Port A, Port B and Port C. The port addresses for these devices can be found in table 5.2 in section 5.4.

The two 8255As at U18 & U42 are completely available to user. The port signals are available on connectors J7 & J10.

5.10 PROGRAMMABLE INPUT/OUTPUT AND TIMER

ESA 51 has one 8155 (programmable I/O and timer). The 8155 has 256 bytes of RAM, two 8-bit and one 6-bit programmable input/output ports, command port and two 8-bit registers to load counter for timer.
The Port A of the 8155 is configured as input to read the data from ADC output. The Port B is configured as output to provide data for the on-board DAC. The Port C is configured as input port to read the most significant bits of ADC data.

VECTOR ADDRESS LOOK-UP TABLE FOR INTERRUPTS

Function	Interrupt source	Vector address	Trainer Address
External Interrupt 0	IE0	0003H	276CH
Timer Interrupt 0	TF0	000BH	FFF0H
External Interrupt 1	IE1	0013H	FFF3H
Timer Interrupt 1	TF1	001BH	FFF6H
Serial Interrupt	RI & TI	0023H	FFF9H

5.11 8-CHANNEL 12 – BIT A/D CONVERTOR

ESA 51 features an optional onboard 8 channel, 12 bit ADC. The interfacing circuit consists of analog multiplexer enables data from up to eight different analog sources to be acquired. This circuit can accept either unipolar signals in the range 0 to +10V, or bipolar signals of –5V to +5V or in the range of –10V to +10V. The voltage span can be selected by placing suitable jumpers. This circuit is built around the industry standard. Fast ADC, AD1674 has built in clock and sample - hold circuit which completes a conversion in 10usec. This interfacing finds extensive use in the fields of analog measurements, transducer interfacing. Industrial monitoring etc. For making use of the ADC, the user has to connect +12V and –12V at connector J3.

DESCRITION OF THE CIRCUITS

Please refer to the schematic diagram presented in appendix.

Single Channel / Multi channel Operation

Jumper JP15 decides whether the circuit is intended for single channel or 8- channel operation. When single channel operation is intended, no multiplexer is used and JP15 is OPEN. Input is applied to screw terminal TP. When the ADC is in multi-channel mode, the multiplexer (ADG508) is populated and eight channels are available as selected by channel select lines connected to ADG508. Analog signals are applied to any desired channel at screw terminals provided at J11. The jumpers JP11, JP12, JP13, JP14, JP15 & JP24 are closed. When the ADC is in multi-channel mode no signal is applied at terminal TP. The multiplexed signal can however be monitored at terminal TP. The channel selection is as shown in the table below.

P 1.5 (E)	P1.4 (A2)	P 1.3 (A1)	P 1.2 (A0)	Channel selected
0	*	*	*	None
1	0	0	0	0
1	0	0	1	1
1	0	1	0	2
1	0	1	1	3
1	1	0	0	4
1	1	0	1	5
1	1	1	0	6
1	1	1	1	7

(* don't care.)

The interfacing of ADC to 8155 is as follows :

The converted data is latched into the latches 74LS374 at U27 and U28 P1.1 of 8031 control of these latches. When P1.1 is high the output of the latches are tri-stated. When P1.1 is low, data is available at port bit P1.0 is used to initiate the conversion process in the ADC. A high to low transition initiates a conversion and high STS of ADC indicates that ADC is busy. At end of conversion, the STS line goes on low. This transition is used to strobe the converted data into the latches U27 and U28.

Input voltage range

This circuit can be operated with inputs of 0 to +10V,-5V to +5V or –10V to +10V. This is determined by the location of shorting plugs at jumpers JP9 and JP8 details are shown below.

Input Range	JP9	JP8
Unipolar 0 to +10V	2-3	1-2
Bipolar –5 to +5V	2-3	2-3
Bipolar –10V to +10V	1-2	2-3

5.12 ONBOARD 8 BIT D/A CONVERTOR

ESA 51 features onboard Digital to analog converter using DAC 0800 IC at U32. The user can program the DAC to suit his needs using the interfacing information described here. For making use of the DAC, the user has to connect +12V and –12V at connector J3 . The Analog output may be obtained from terminal-point J12 on the trainer. Interesting waveforms may be observed at this point by programming the DAC suitably.

The digital inputs to the DAC are provided through port B of 8155 (U29) after initializing it as an output port. The Analog output from the DAC is given to an operational amplifier (LM741 at U40) which serves the purpose of current to voltage conversion. A 10K POT (VR1) is provided for offset balancing of the op-amp. The reference voltage needed for the DAC is obtained from an onboard voltage regulator LM723. The voltage obtained at the output of this regulator is about 8 volts. The output from the DAC varies between –5V and +5V depending on the input digital pattern fed to the DAC.

5.13 BUS EXPANSION

ESA 51 permits easy expansion of the system by providing all the necessary signals on two connectors J1 and J2. The signals are STD bus compatible and thus user can easily expand the capabilities of ESA 51.

5.14 CONNECTOR DETAILS

The connector details for interfacing peripherals is described in this section. A brief summary of the connectors available on the trainers is described below. Refer the component layout diagram in Appendix G to locate these connectors. The signal definition of these connectors are available in Appendix G.

J1 & J2	:	50 pin Bus connectors connected to system bus.
J3	:	4 – terminal power connector
J4	:	9 pin, D type female connector for RS-232-C compatible serial communication
J5	:	9 pin D type, male connector for RS 485 compatible serial communication.
J6	:	Keyboard DIN connector.
J7	:	26 pin FRC Rt. Male connector for 8255 port lines.
J8	:	Flow strip for LCD interface.
J9	:	25 pin D-type female for printer interface.
J10	:	26 pin FRC Rt. Male connector for 8255 port lines.
J11	:	Terminal strip for ADC Analog input.
J12	:	Terminal point for DAC Analog O/P.
J13	:	Terminal point for Analog GND for ADC circuitry.

J1 ADDRESS & DATA

Pin No.	Signals	Pin No.	Signals
1	P1.0	2	P1.1
3	P1.2	4	P1.3
5	P1.4	6	P1.5
7	P1.6	8	P1.7
9	P3.0	10	P3.1
11	P3.3	12	P3.4
13	P3.5	14	STS
15	BA0	16	BA1
17	BA2	18	BA2
19	BA4	20	BA3
21	BA6	22	BA4
23	BA8	24	BA5
25	BA10	26	BA6
27	CD0	28	CD1
29	CD2	30	CD3
31	CD4	32	CD5
33	CD6	34	CD7
35	CLK0	36	CLK1
37	CLK2	38	GATE0
39	GATE1	40	GATE2
41	OUT0	42	OUT1
43	OUT2	44	OFFBOARDSEL*
45	BRD*	46	BWR*
47	+5V	48	+5V
49	GND	50	GND

J2 MCU PORT & COMMAND

Pin No.	Signals	Pin No.	Signals
1	P0.0	2	PO.1
3	P0.2	4	PO.3
5	P0.4	6	PO.5
7	P0.6	8	PO.7
9	P0.0	10	P1.1
11	P1.2	12	P1.3
13	P1.4	14	P1.5
15	P1.6	16	P1.7
17	P3.0	18	P3.1
19	INT	20	P3.3
21	P3.4	22	P3.5
23	BWR*	24	BRD
25	P2.0	26	P2.1

27	P2.2	28	P2.3
29	P2.4	30	P2.5
31	P2.6	32	P2.7
33	ALE	34	PSEN*
35	TIN	36	TOUT
37	NC	38	NC
39	NC	40	NC
41	NC	42	NC
43	NC	44	NC
45	NC	46	NC
47	+5V	48	+5V
49	GND	50	GND

J7 PORTS CONNECTOR

PIN NO	SIGNALS	PIN NO	SIGNALS
1	P2C4	14	P2B1
2	P2C5	15	P2A6
3	P2C2	16	P2A7
4	P2C3	17	P2A4
5	P2C0	18	P2A5
6	P2C1	19	P2A2
7	P2B6	20	P2A3
8	P2B7	21	P2A0
9	P2B4	22	P2A1
10	P2B5	23	P2C6
11	P2B2	24	P2C7
12	P2B3	25	+5V
13	P2B0	26	GND

J10 PORTS CONNECTOR

PIN NO	SIGNALS	PIN NO	SIGNALS
1	P1C4	14	P1B1
2	P1C5	15	P1A6
3	P1C2	16	P1A7
4	P1C3	17	P1A4
5	P1C0	18	P1A5
6	P1C1	19	P1A2
7	P1B6	20	P1A3
8	P1B7	21	P1A0
9	P1B4	22	P1A1
10	P1B5	23	P1C6
11	P1B2	24	P1C7
12	P1B3	25	+5V
13	P1B0	26	GND

J9 PRINTER CONNECTOR

PIN NO	SIGNALS
1	STROBE*
2	DATA 0
3	DATA 1
4	DATA 2
5	DATA 3
6	DATA 4
7	DATA 5
8	DATA 6
9	DATA 7
10	NC
11	BUSY*
12	NC
13	NC

PIN NO	SIGNALS
14	NC
15	NC
16	NC
17	NC
18	GND
19	GND
20	GND
21	GND
22	GND
23	GND
24	GND
25	GND

CHAPTER 6

MONITOR ROUTINES ACCESSIBLE TO USER

ESA51 monitor offers several user– callable routines both in the stand-alone and serial modes of operation, details of which are given below. These routines can be used to considerably simplify the program development work.

NOTE : Users should, as general rule, save the registers of interest before calling the monitor routines and restore them after returning form the monitor routines.

6.1 STAND-ALONE MODE

Calling Address	Mnemonic	Function/Description
0200H	LCDINIT	To initialize the LCD module. All registers are affected.
035DH	LCDWR	Displays one character to the LCD display module. Character to be displayed should be in accumulator. All registers are affected.
03A1H	LCDOUT	Displays a string of character on LCD. The string should end with a zero which is not output. The parameter for this routine are as follows : Reg DPTR = Starting address of string of characters. The string characters will be displayed from program memory area or data memory area depending on the status of flag register bit F0. If flag register bit F0 = 0 program memory will be selected. If flag register bit F0 = 1 data memory will be selected.

03BBH	C LRLCD	Clear the display. The routine blanks the entire display fields.
0161H	GETKB	Reads keyboard. This routines wait until a character is entered from the keyboard and upon return, it places the character in the A register. All registers are affected.
0303H	NEXTLINE	Moves the LCD cursor to the next line.

6.2 SERIAL MONITOR ROUTINES ACCESSIBLE TO USER

Calling Address	Mnemonic	Function/Description
12BBH	GETCH	Gets one character from the USART input parameters None. Output : A = Character (ASCII) received from the USART. Reg. A, and flags are affected.
11A8H	OUTCHR	Outputs one character to the USART. Input : A = character (ASCII) to be output to USART. Reg. A, and flags are affected.
1200H	OUTSTG	Displays a string of characters. The string should be terminated by character Zero which is not output. Inputs : DPTR = Starting address of the string of character. The string of character will be displayed from program memory or data memory area depending on status of flag register bit F0. If flag register bit F0 = 0 program memory will be selected. If flag register bit F0 = 1 data memory will be selected.

6.3 USER ACCESSIBLE ROUTINES COMMON TO BOTH STAND-ALONE MODE AND SERIAL MONITOR

Calling Address	Mnemonic	Function/Description
0404H	OUTPUT	Displays a string of characters both in LCD and PC Console. The string should be terminated by character Zero which is not output. Inputs : DPTR = Starting address of the string of character. The string of character will be displayed from program memory or data memory area depending on status of flag register bit F0. If flag register bit F0 = 0 program memory will be selected. If flag register bit F0 = 1 data memory will be selected.
13D2H	PUTBYTE	Outputs one character to the USART. Input : 71H = character to be output to USART. Reg. A, and flags are affected.
03E6H	DSPCHR	Displays a character on both LCD and PC console. Accumulator should contain the character to be displayed.

CHAPTER 7

PARALLEL PRINTER INTERFACE

7.1 INTRODUCTION

ESA 51 trainer support centronics compatible parallel printer interface. The interface makes use of BUSY signal for hand - shaking and strobe pulses for synchronization. Using this facility the user can obtain hard copy on any centronics compatible printer. However to get properly formatted listing it is advisable to use 80 / 132 column printer. The on - board 8255 (U30) is made use of, to implement this interface.

7.2 INSTALLATION

To install the printer interface

a) Switch OFF the power supply.

b) Connect one end of the printer cable to J9 (25 pin "D" type connector) of ESA 51 (Refer the component layout diagram in Appendix A to locate the connector J9).

c) Connect the other end of connector to the printer.

d) Configure the system for serial mode of operation by setting SW4 of the on - board DIP switch to ON position.
e) Enable the printer interface by setting the SW6 of the on - board DIP switch to ON position.
f) Switch on power to the printer and ESA 51.

NOTE : The necessary printer cable could be obtained from Electro Systems Associates (P) Ltd., as an optional accessory. However, the connector details are given in section 7.5 and the user can make use of these details to make a suitable cable if desired. Please note that cable must be short enough to be driven by the on - board 8255. We suggest a maximum length of 3 feet for reliable operation.

7.3 OPERATION

When the printer interface is installed and enabled as described above, any character sent to the console is sent to the printer also. For example, to obtain a hard copy of the contents of a block of memory locations, user can issue the M (Display Memory) command from the serial monitor. The contents of the specified memory block are printed exactly as they appear on the screen. Note that the M command itself is also printed.

NOTES :

1) All control and non - printable ASCII characters are printed as "." (ASCII code 2EH)
2) If any errors occurs during printing (for e.g. : Printer is not in ON - LINE, paper out error etc.,), the system will be looping indefinitely in the print character routine. To recover, user may have to press the RESET key.

7.4 DIRECT OUTPUT TO PRINTER

As already described, when the printer interface is enabled, any character sent to the console is sent to the printer also. This facility is available in the serial mode of operation only. However, user can directly access a routine "print character" to print a single character. This routine can be called from the user's program when the system is operating in either of the two modes – stand-alone or serial. Further, this routine prints a character independent of the setting of SW6. Thus this routine can be used to print the desired information when the system is running in the stand-alone mode. Even in a serial mode of operation, this routine can be used to print information which may not be sent to the console. The details of this routine are given below :

Name of the Routine : PRINT

Function : Print a character (Non - printable one is printed as ".").

Calling address : 1235H

Input : Register A = ASCII code of the character to be printed.

Destroys : Registers A, B, DPTR, R0, Flags.

Returns : Reg C = 1 if error otherwise C = 0.

Example : The following program, if entered and executed from the stand-alone mode, prints 'ABCDEFGHIJKL'.

Address	Opcode	Mnemonic	Comment
8000	90 80 50	MOV DPTR, # MSG	; point to msg
8003	E4	L1: CLR A	
8004	93	MOVC A,@A+DPTR	; get the chr
8005	60 0E	JZ L2	; end of msg ?
8007	C0 82	PUSH DPH	; no, save pointer
8009	C0 83	PUSH DPL	
800B	12 12 35	LCALL PRINT	; print chr
800E	D0 83	POP DPL	; restore pointer
8010	D0 82	POP DPH	
8012	A3	INC DPTR	; point to next chr
8013	80 EE	SJMP L1	; repeat the process
8015	02 00 03	L2 : LJMP 0003	; save status and
			; return to monitor
8050	41, 42, 43, 44	DB: 41H, 42H, 43H, 44H	
8054	45, 46, 47, 48	45H, 46H, 47H, 48H	
8058	49, 4A, 4B, 4C	49H, 4AH, 4BH, 4CH	
805C	0D, 0A, 00	0DH, 0AH, 00H	

7.5 CONNECTOR DETAILS

The signal definitions on the 25 pin, female D type connector used for parallel printer interface are given below :

PIN NO ON J9	SIGNAL	DIRECTION FROM ESA 51	DESCRIPTION	PIN. NO. ON CENTRONICS CONNECTOR
1	STROBE	O / P	STROBE * pulse to the printer	1
2	Data 0	O / P	These signals represent 8	2
3	Data 1	O / P	bits of parallel data	3
4	Data 2	O / P	High – 1	4
5	Data 3	O / P	Low – 0	5
6	Data 4	O / P		6
7	Data 5	O / P		7
8	Data 6	O / P		8
9	Data 7	O / P		9
10	BUSY	O / P		11
11	BUSY	I / P	A high indicates that printer cannot receive data. The signal becomes high in following cases a) During the data entry. b) During printing operation. c) In the OFF – LINE state. d) During printer error status.	11
18-25	GND		Signal Ground	19

CHAPTER 8

EPROM PROGRAMMER SYSTEM

8.1 INTRODUCTION

This chapter describes the use of the EPROM Programmer system. The ESA 51 trainer and EPROM programmer interface module with a 26 core flat ribbon cable together form the EPROM programmer system.

The system permits the user to program, verify, blank check and read any of the popular EPROMs 2716 through 27512. The system consists of the necessary hardware and software. The software can be invoked either from the stand-alone mode monitor or from the serial monitor. A 28 pin ZIF socket is provided for placing the EPROMs. When 24 pin EPROM is replaced, it must be aligned with the bottom rows i . e top two rows of ZIF are to be left blank.

The system uses intelligent programming algorithm whenever possible which reduces the programming time significantly.

The devices supported by the system and the type number to be entered by the user are listed below :

Device		Type number to be entered by the user
2716	(@ 25 V)	2716
27C16	(@ 25 V)	2716
2732A	(@ 21 V)	732A
27C32A	(@ 21 V)	732A
27C32	(@ 25 V)	2732
2732	(@ 25 V)	2732
2764A	(@ 12.5 V)	764A
27C64D	(@ 12.5 V)	764A
27C64	(@ 21 V)	2764
2764	(@ 21 V)	2764
27128A	(@ 12.5 V)	128A
27C128	(@ 12.5 V)	128A
27128	(@ 21 V)	0128
27256	(@ 12.5 V)	0256
27C256	(@ 12.5 V)	0256
27C256	(@ 21 V)	2256
27512	(@ 12.5 V)	0512

The device selection is totally software - controlled and no hardware changes or jumper settings are necessary for selecting any of the above listed devices.

8.2 INSTALLATION PROCEDURE

a) Turn OFF power to ESA 51 trainer.

b) Attach the hardware module (EPROM Programmer Interface) to ESA 51 over connector J7 using 26 core ribbon cable supplied with the module.

c) Connect black, yellow and blue wires coming from 4 pin polarized connector on the programmer module to corresponding power supplies as shown below :

colour of the wire	supply to be connected
BLACK	GND
YELLOW	+ 12V
BLUE	+ 30V

d) Power ON the system

8.3 OPERATION FROM BOTH SERIAL AND STAND-ALONE MODE MONITORS:

Enter P when prompted for command entry. Then system enters EPROM Programmer menu and displays

R : Read B : Blank check P : Program V : Verify E : Exit

Enter Option :

The monitor provides the following 4 commands to support the EPROM Programmer System :

- P - Program command
- V - Verify command
- B - Blank check command
- R - Read command

Enter the appropriate command, when prompted for command entry by monitor.

Aborting a command :

With ‘ E’ it exits from the EPROM Programmer Menu to the monitor prompt.

Once a specific command is issued, further prompts will depend on command itself. However, if the user enters <Esc> whenever the system is looking for an entry from the user, the current operation is aborted and control returns to the warm start of the monitor. So in this case the user has to re – enter the EPROM Programmer menu by entering ‘ P ’ before issuing any command.

8.3.1 P COMMAND :

This command is used to program an EPROM. This command requires the following 4 parameters:

EPROM type = EPROM Type should be one of the types listed above in section 8.1

Buffer Start = Starting address of the source of data

Buffer End = Ending address of the source of data

EPROM start = Absolute starting address of the EPROM
(from where programming is to begin)

NOTE : Buffer always means data memory area. If the program to be programmed on to an EPROM is in program memory area, the user has to transfer it to the data memory area before attempting programming. Similarly the user must note that when Read and Verify operations are performed, buffer means data memory area.

As soon as P is typed, the system displays each parameter value and prompts for new value. User can enter the new value followed by <CR> or simply enter <CR> if the displayed value is not to be changed. Note that the parameters must satisfy certain conditions as listed below.

1) EPROM type can only be one of the valid types listed in section 8.1
2) Buffer end address must be greater than or equal to the Buffer start address.
3) The EPROM must have enough space to accommodate all the bytes specified by the Buffer start address and Buffer end address. In other words, the following relation must be satisfied. EPROM Start + (Buffer end address – Buffer start address)< = Highest absolute address of the EPROM.

For example, suppose EPROM type is 2764. Then its highest absolute address is 1FFFH. Suppose the other parameters are as follows :

Buffer Start = 8000
Buffer End = 9FFF
EPROM Start = 100

Then 100 + (9FFF – 8000) 20FF > 1FFF. So this combination of parameters is invalid.

- After user enters the parameter values, the above mentioned constraints are checked and if any of the constraint is violated, it displays the message “invalid parameter(s) entered” and returns to the sub menu. However if the EPROM type entered is invalid, it immediately flashes a message “invalid EPROM type” and reprompts user to enter valid EPROM type.

After optional modification of the parameter values by the user, the system checks the EPROM for blank values (0FFH) in the required zone. It displays the message,

Blank check in progress…

If the EPROM is not blank, the following prompt appears :

EPROM is not blank @ XXXX – YY

Continue programming (Y / N)

If user types N, the command is aborted and control returns to command prompt of the monitor.
If the user enters Y, the system proceeds further. Any other character results in error message and repetition of the same prompt.

Now the following message appears :

Programming in progress …

The system proceeds with programming and verification on a byte by byte basis. Intelligent Programming Algorithm is used if the EPROM support it. This results in considerable reduction in programming time required for some Devices.

If the complete programming is successful, the system will display a 16 – bit checksum and control will return to the sub menu.

If the programming is unsuccessful, the following information is displayed.

Programming failed @XXXX – YY

When XXXX is the EPROM address where programming failed and YY is the Data at that location on EPROM.

NOTE: During programming and verification the location in EPROM and the corresponding data that is being programmed are displayed in the display field continuously.

After programming the specified range, the system display 16 bit checksum and returns to sub menu.

Checksum = NNNN

8.3.2 V COMMAND

This command is used to verify the contents of an EPROM against a source. The parameter and their interpretation is completely similar to that of the P command. If the verification is successful, the 16 – bit checksum is displayed and the control returns to EPROM programmer sub menu. It the verification fails, a message and parameter at the point of failure are displayed as shown below.

Verification fails AAAA – BB CC – DDDD

Where AAAA is the EPROM address where verification has failed, BB is the data at that location Similarly it also displays the corresponding buffer address (DDDD) and the data (CC) at that location. If there are multiple locations where verification has failed it will list out them in the same format as above. The control returns to the EPROM programmer sub menu. The user can abort to main menu by pressing <Esc>.

Thus the operation of this command is quite similar to the operation of the P command, except that here the EPROM is just verified, not programmed.

8.3.3 B COMMAND

This command is used to check if a specified range in the EPROM is blank (contains 0FFH).

This command requires the following three parameters:

Type	: same as for P command
EPROM Start	: The absolute starting address of the EPROM.
EPROM End	: The absolute ending address of the EPROM.

The parameter must satisfy the following relations :

i. EPROM start < = Absolute last address of the EPROM.

ii. EPROM End < = Absolute last address of the EPROM and

iii. EPROM End > = EPROM Start.

The parameter display and modification procedures are menu - driven and are similar to those of the P command.

The EPROM is checked for blank values in the specified range. The EPROM address and data read are displayed in the display field. If it is blank then the following message is displayed.

EPROM is blank

Then the control returns to the sub-menu.
If a location in the specified range is not blank, the following message is displayed.

EPROM is not blank @ XXXX – YY

Where XXXX is the absolute EPROM address of the first non – blank location and YY is its contents. Then it display the address and data of subsequent non blank locations form next line onwards. If <Esc> is pressed control returns to the monitor.

8.3.4 R COMMAND

This command is used to transfer contents of the EPROM into the ESA 51 memory space.

This command requires the following four parameter.

EPROM Type	: same as for P command
EPROM Start	: same as for B command
EPROM End	: same as for B command
Buffer Start	: starting address in ESA 51 memory space.

The parameter display and modification procedures are menu driven and are completely similar to the ones described for P command.

The starting and ending address of the EPROM must satisfy the relations described for the B (Blank Check) command.

After the optional modification of the parameter, the contents of the EPROM, specified range are transferred into ESA 51 memory, starting at the specified Buffer starting address. The EPROM address and data read are displayed in the display fields. During the transfer, as each byte is written into memory, it is read back and verified. If the write is successful for all locations, a 16 – bit checksum is displayed and control returns to the sub menu.

If an error occurs during transfer (i.e. unsuccessful write into location), the following message is displayed.

Read fails @ XXXX

Where XXXX is the address of location where write failure occurred. Then control returns to the monitor.

8.3.5 EXAMPLE:

From monitor, program the contents of locations 8000H to 8FFFH into a 2764, starting at 1000H.

.P

R: Read B: Blank Check P: Program V: Verify E: Exit

Enter option: P

EPROM Type = 2732 – 2764 <CR>

Buffer start = 0000 – 8000 <CR>

Buffer End = 0000 – 8FFF <CR>

EPROM Start = 0000 – 1000 <CR>

PROGRAMMING IN PROGRESS…..

Checksum 1724

R: Read B: Blank Check P: Program V: Verify E: Exit

Enter option: E

>

CHAPTER 9

COMMUNICATION WITH A HOST COMPUTER SYSTEM

9.1 INTRODUCTION

As already noted, ESA 51 operating in the serial mode, can be connected to a host computer system. When a computer system is the controlling element, it must be executing a driver software to communication with ESA 51.

XT51 is such an optional communication package which allows the user to establish a communication link between asynchronous serial ports of the computer (COM1/COM2), and ESA 51.

XT51 is supplied as a ".EXE" file on a 3½ ″ 2SHD diskette and can be executed on a PC under PC-DOS/MS-DOS operating system. A suitable RS-232-C/RS 485 cable has to be used for connecting ESA 51 to a PC.

XT51 fully supports all the command of ESA 51. Further, it allows the contents of a disk file to be downloaded from the computer system into memory of ESA 51. User can develop assembly language program on the PC, cross-assemble them using a suitable cross-assembler to generate object code files and then use XT51 to download these object code files into ESA 51 for execution. Thus the extensive development facilities available on the PC can be used to supplement the facilities available on ESA 51.

Further XT51 allows uploading of data from memory of ESA 51 to the computer. The data so uploaded is saved in a disk file. Thus this facility can conveniently be used to save user program.

9.2 INSTALLATION

NOTE : Make sure that you have made a back up copy of XT51 .EXE before proceeding with the installation.

The detailed installation procedure is as follows :

a) Configures ESA 51 for serial mode of operation and set the serial port of ESA 51 for 9600 Baud and No parity (Refer sections 2.1.1 and 2.1.3)

b) Connect the computer system to ESA 51 over the COM1/COM2 serial port (Refer to Technical Manual of your system for details regarding the signal definition on COM ports. The signal definition of the RS-232-C port of ESA 51 can be found in Appendix C)

c) Insert the diskette containing the file XT51.EXE into the available drive and run the program by typing XT51 or XT51 /B to select black and white mode if computer system has a CGA monitor.

d) Now the following message appears on the screen.

XT51 Version 1.0

ELECTRO SYSTEM ASSOCIATES PVT LTD

BANGALORE

ALT+S	- Set communication Parameters
CTRL+F1	- Help
ALT+F1	- Command Help
<Esc>	- Clear Command

<F1>	- Previous Command Character
<F3>	- Command Recall
CTRL+U	- Uploaded Command
CTRL+D	- Download Command
!Command	- Dos Shell/command
ALT+X	- Exit

Press any key to continue

e) XT51 checks for the presence of communication ports COM1 & COM2. If both ports are not available it will display the message NO serial port present as reported by BIOS and exits to DOS. Otherwise XT51 will read the communication parameters from file "XT51.INS " and initializes the communication port. XT51 searches current directory for file "XT51 .INS" . If search fails, it will search the path given by the DOS environment variable INIT. In the file is not present following message is displayed.

XT51.INS is not found !

Serial parameters are set to COM1, 9600, 8, 2, None

Do you want to change?

Yes No

If options "No" is selected the communication parameters, Serial Port COM1, Baud 9600, Data bits 8, Stop bits 2, parity none are set. If option "Yes" is selected the communication parameters can be interactively modified as described in section 9.4.6. Now XT51 attempts to establish communication between the computer and ESA 51. If successful the command prompt ">" appears on the screen. Subsequently during the power – on or reset of the trainer, the sign – on message "ESA 8051 SERIAL MONITOR V x.y appears on the screen followed by command prompt ">", otherwise it will display the message

Unable to transmit data

Retry or Ignore

If ESA 51 is not powered on power it on and press <R> to retry to establish the communication. The above mentioned message appears on the host side again. Pressing <I> will exit XT51 to DOS. Now check for the following.

a) Ensure that ESA 51 is connected to the correct COM and that the COM port is functioning properly.

b) Ensure that ESA 51 is functioning properly and configured correctly.

c) Check the RS-232-C/RS 485 cable and its connections.

Since the communication package utilizes the hardware handshake signal RTS while communicating with ESA 51, the interfacing cable must support this signal also.

NOTE : XT51 utilizes an interrupt driven routine for reading character from the COM port. Thus it is possible for XT51 to miss some character if the system has any resident program which are interrupt driven (For example, many system include a CLOCK program in the AUTOEXEC file, to display the time on the upper right corner of the screen). Hence it is desirable not to run any such resident program while XT51 is running.

If the problem persists, please contact the manufacturer.

9.3 RETURNS TO DOS

User can terminate XT51 and return control to DOS by typing ALT+X when the program is waiting for a keyboard input.

9.4 OPERATIONAL DETAILS

The complete command set of the ESA 51 is transparent and is fully supported by XT51 (Refer to chapter 4 for serial monitor mode command) Press F1 for the help command.

In addition, XT51 supports the file download, file upload and other commands which are explained below.

NOTE : During parameter entry, the system expects the alphabetic character to be in uppercase. Thus it is convenient to use the keyboard with the CAPS LOCK on.

9.4.1 DOWNLOAD OPERATION :

This feature allows downloading of the contents of an object code file into the memory of ESA 51.

NOTE : The object code file must be a "HEX" file with records in INTEL 8-bit HEX format. Please refer to the relevant INTEL manuals for the definition of INTEL 8-bit HEX format. Most of the cross assemblers for 8031 do produce object code files with records in INTEL 8-bit HEX format.

To perform download operations, type CTRL+D in response to the command prompt (">").
The system will now prompt for the name of the disk file, from which the information is to be downloaded. The prompt is as follows :

Download filename [.HEX] :

Enter the file name with extension, terminated by <CR>. If the filename is invalid, it display a message " FILE NOT FOUND" and prompts again for the filename. If the path specified is invalid, it displays the message "PATH NOT FOUND" and prompt again for the filename. If none of the above errors occur, the system will prompt for the memory type as follows.

Memory type {P|D} :

The user has to enter P or D depending on whether program has to be downloaded to program or data memory.

Now the system will prompt for the staring address of the program as follow.

Start Address:

Enter the starting address followed by <CR>. A maximum of four hex digits are allowed for the starting address. Now the system will prompt for the ending address as follows.

End Address:

Enter the ending address followed by <CR>. A maximum of four hex digits are allowed for the ending address. Now the system will prompt for the load offset value as follows.

Load offset Address:

If the user wishes to download the file to an address range different from the actual address range of the file, then a suitable offset value can be entered to enable the file to be downloaded to the desired address range. For example if the user wishes to download it to an address range starting at C000H, the user has to enter an offset value of C000H –8000H=4000H. In case the user wishes to download to the same address range an offset value of 0000 has to be entered or simply press <CR>. After obtaining the filename, starting address and the ending address, the system

will read the file, gather the data in specified address range, reformat the data for compatibility with the protocol required by ESA 51 and send the data to ESA 51.

Downloading in progress ……xxx

After downloading is over, the system returns to the command prompt of ESA 51. It also displays the starting address of each record being download.

9.4.2 UPLOAD OPERATION :

This feature allow uploading of the data from the memory of ESA 51 to the computer system and save the data in the specified disk file in INTEL 8 bit HEX format.

To perform upload operations, type Ctrl +U in response to the command prompt (“>”).

The system will now prompt for the name of the disk file, into which information is to be uploaded. The prompt is as follows :

UPLOAD FILENAME (.HEX)

Enter the file name with extension, terminated by <CR>. If the file already exists, then the system will display

File already exists

Overwrite?

Yes No

Select the first option by pressing <Y> to overwrite the contents of the existing file. Pressing <N> will let the user specify another file name. Select the third option <A> to append to the contents of the existing file.

If no error occurs, the system will prompt for the starting address of the program as follow :

Memory Type {P|D}

The user has to enter P or D depending on whether program has to be uploaded from program or data memory.

Now the system will prompt for the starting address of the program as follows:

Start Address

Enter the starting address terminated by <CR>. A maximum of four hex digits are allowed for the starting address.

Now the system will prompt for the ending address as follows

End Address

Enter the ending address terminated by <CR>. A maximum of four hex digits are allowed for the ending address file.

After obtaining the filename, starting address and the ending address, the system will gather the data in the specified address range of the ESA 51, reformat the data into INTEL 8 bits HEX record and store the data in specified file.

Uploading in progress (XXXX)

While the uploading is going on the system will display the starting address (XXXX) of each record being uploaded. Once the uploading is complete XT51 will let user specify another address range. User may specify a new address range or enter <Esc> to terminate uploading operation.

The following error message may appear during upload and download operations

1. Invalid function number !
   This is XT51 internal error, therefore contact ESA technical support for assistance.
2. File not found !
3. Path not found !
4. No more files !
5. Access denied !

6. Invalid file handled !
7. Insufficient disk space !
8. Unable to continue upload !
9. Colon is not present at the start of the record
10. Invalid data in (source file) the following record !
11. Check sum error in the following record !

9.4.3 DOS COMMANDS

At the command prompt ">" any valid DOS command can be entered preceded by "!" XT51 environment is saved and the DOS command is executed. Then XT51 environment is restored and XT51 command prompt be displayed again.

9.4.4 BOTTOM LINE :

During the session XT51 display many of the XT51 commands at the bottom line in reverse video for convenience of user. The bottom line is displayed as

Ctrl+F1 Help, Alt+F1CmdHlp, Alt+S Commn, <Esc> ClrCmd, Alt+X Exit, F1,F3,

9.4.5 COMMAND RECALL

This feature facilitates recall of the command already entered by the user, upto a maximum of 16 commands. Press <F3> to recall the previous command. All the command are kept in a circular buffer. User may use Up-arrow and Down- arrow keys to traverse through the sequence of command is backward or forward direction in circular fashion. User may recall just the previous command being entered can be cleared by using <Esc> key anytime before pressing <CR>.

9.4.6 COMMUNICATION

Communication parameters can be set during the session by pressing ALT+S. List of parameters and their current values will appear on the screen. Select the desired parameter with the help of arrow keys and keep the space bar pressed till the desired value appears. The parameters allowed to be set are communication Port (COM1/COM2), Baud Rate (110/150/300/600/1200/2400/4800/9600), Number of Data bits (7/8), Parity (NONE /ODD/EVEN) and Number of stops bits (1/2), After selecting the desired values press <CR> to set the parameters or press <Esc> to ignore the values.

Communication parameters can also be modified, while user is in DOS by editing the file XT51.INS. This file contains single data line, having five integer separated by blanks, representing various communication parameter. This five integer represent serial communication port, baud rate, no. of data bits, no. of stop bits and parity, in sequence.

Table 9.1 shows details of the integer values and corresponding parameters.

Commn. Port	**int1**	**Baud Rate**	**int2**	**Data Bits**	**int 3**	**Stop Bits**	**int 4**	**Parity**	**int 5**
COM 1	0	110	0	7	0	1	0	Odd	0
COM 2	1	150	1	8	1	2	1	None	1
		300	2					Even	2

	600	3			
	1200	4			
	2400	5			
	4800	6			
	9600	7			

9.4.7 HELP :

On-line help is available on all ESA 51 monitor commands specific to XT51. Help facility can be selected by CTRL+F1. A menu of commands is displayed from which desired commands can be selected by using arrow keys and help information about that command is displayed in the remaining part of the screen. Context sensitive help is available using ALT+F1. This facility can be used if help information is desired about the command being entered against command prompt.

CHAPTER 10

PROGRAMMING EXAMPLES

10.1 Program to display ESA PVT LTD on the LCD. Execute the program in stand-alone mode only.

```
          0348              LCDOUT EQU     0348H
          03BB              CLRLCD EQU     03BBH

ADDRESS  OBJECT              MNEMONICS                COMMENTS

 8000                         ORG     8000H
 8000     12 03 BB            LCALL   CLRLCD
 8003     C2 D5               CLR     F0          ;To select
                                                   ;program memory
 8005     90 80 0E            MOV     DPTR,#STG
 8008     12 03 48            LCALL   LCDOUT   ;Output routine to
                                                  ;display string
                                                  ; of characters
 800B     02 80 0B            LJMP    800BH

 800E     45 53 41 20 50 STG: DB      'ESA PVT LTD.',0
 8013     56 54 20 4C 54
 8018     44 2E 00
```

10.2 Program to perform multiplication of two numbers, present at data memory Location 9000 & 9001 and storing the result in 9002 & 9003 Data memory.

```
ADDRESS  OBJECT          MNEMONICS          COMMENTS

8000                     ORG      8000H
8000     90 90 01        MOV      DPTR,#9001H  ;Keep data in
                                               ;9000h & 9001h
                                               ;data memory
                                               ;location.
8003     E0              MOVX     A,@DPTR
8004     F5 F0           MOV      0F0H,A
8006     90 90 00        MOV      DPTR,#9000H
8009     E0              MOVX     A,@DPTR
800A     A4              MUL      AB         ;Perform
                                            ;multiplication
                                            ;operation
800B     90 90 02        MOV      DPTR,#9002H ;Store the
                                              ;result in 9002h
                                              ;& 9003h of data
                                              ;memory
800E     F0              MOVX     @DPTR,A
800F     A3              INC      DPTR
8010     E5 F0           MOV      A,0F0H
8012     F0              MOVX     @DPTR,A
8013     02 00 00        LJMP     0
```

10.3 Program to perform Division of 2 numbers. Execute the program either in stand-alone mode or in serial mode. 2 numbers available at 9000 & 9001 of Data memory. After division operation store the result in 9002 & 9003 Data memory.

```
ADDRESS  OBJECT          MNEMONICS              COMMENTS

8000                     ORG        8000H
8000     90 90 01        MOV        DPTR,#9001H;Keep data in
                                               ;9000h & 9001h
                                               ;data memory
                                               ;location
8003     E0              MOVX       A,@DPTR
8004     F5 F0           MOV        B,A
8006     15 82           DEC        DPL
8008     E0              MOVX       A,@DPTR
8009     84              DIV        AB         ;Perform
                                               ;division operation
```

```
800A   90 90 02       MOV         DPTR,#9002H;Store the result
                                              ;in 9002h&
                                              ;9003h data memory
800D   F0            MOVX        @DPTR,A
800E   A3            INC         DPTR
800F   E5 F0         MOV         A,B
8011   F0            MOVX        @DPTR,A
8012   02 00 03      LJMP        3
```

10.4 Program to display
ELECTRO SYSTEMS ASSOCIATES
BANGALORE.
Execute this program in serial mode only.

```
            1200            OUTSTG  EQU     1200H
```

ADDRESS	OBJECT	MNEMONICS	COMMENTS

```
8000                        ORG     8000H
8000    90 80 09            MOV     DPTR,#STG  ;Keeping the data
                                               ;in the program
                                               ;memory
8003    12 12 00            LCALL   OUTSTG     ;Routine to
                                               ;display string
                                               ;of characters on
                                               ;the console
8006    02 00 00            LJMP    0
8009    20 20 20 20 20 STG:
                            DB    '       ELECTRO SYSTEMS
                                    ASSOCIATES PVT. LTD.'
800E    20 20 45 4C 45
8013    43 54 52 4F 20
8018    53 59 53 54 45
801D    4D 53 20 41 53
8022    53 4F 43 49 41
8027    54 45 53 20 50
802C    56 54 2E 20 4C
8031    54 44 2E
8034    0D 0A 20 20 20      DB    0dh,0ah,' BANGALORE.',0
8039    20 20 20 20 20
803E    20 20 20 20 20
8043    20 20 20 20 20
8048    20 42 41 4E 47
804D    41 4C 4F 52 45
8052    2E 00
```

10.5 Program to display ASCII character on the console. Execute this program in serial mode only.

```
          11A8              OUTCHR  EQU  11A8H
```

ADDRESS	OBJECT	MNEMONICS		COMMENTS
8000		ORG	8000H	
8000	90 90 00	MOV	DPTR,#9000H	;Keep the hex ;equivalent of ;the ASCII ;character in ;9000H DATA ;memory
8003	E0	MOVX	A,@DPTR	
8004	12 11 A8	LCALL	OUTCHR	
8007	02 00 03	LJMP	3	

Keep the hex equivalent of the ASCII character to be displayed in 9000h in data memory.

10.6 Program to convert HEX equivalent to ASCII on the trainer display. If the code is less than 40, then 30 is subtracted from the code to get its binary equivalent. If the code is greater than 40, then the equivalent no. lies between A & F. The program can be executed both in stand-alone and serial mode.

```
          13D2              PUTBYTE   EQU   13D2H
```

ADDRESS	OBJECT		MNEMONICS		COMMENTS
8000			ORG	8000H	
8000	90 90 00	LOOP1:	MOV	DPTR,#9000H	;Keep the HEX eq ;in 9000h data ;memory
8003	E0		MOVX	A,@DPTR	
8004	F9		MOV	R1,A	
8005	C3		CLR	C	
8006	94 0A		SUBB	A,#0AH	
8008	40 0B		JC	LOOP2	;Check for carry
800A	E9		MOV	A,R1	
800B	24 37		ADD	A,#37H	
800D	F5 71	LOOP3:	MOV	71H,A	
800F	12 13 D2		LCALL	PUTBYTE	
8012	02 80 00		LJMP	3	
8015	E9	LOOP2:	MOV	A,R1	
8016	C3		CLR	C	

```
8017   24 30              ADD     A,#30H        ;Add ACC.
                                                ;content with 30
8019   02 80 0D           LJMP    LOOP3
```

10.7 In the given byte checking the 5th bit is '1' or '0'. If the 5th bit is '1' 00 should be stored in 8901 data memory or if it is '0' FF should be stored in 8901h data memory

```
ADDRESS  OBJECT            MNEMONICS          COMMENTS

8000                       ORG     8000H
8000     90 89 00          MOV     DPTR,#8900H  ;Store the
                                                ;given byte in
                                                ;8900h data
                                                ;memory
8003     E0                MOVX    A,@DPTR
8004     A3                INC     DPTR
8005     33                RLC     A            ;Rotate left
8006     33                RLC     A            ;three times
8007     33                RLC     A
8008     40 06             JC      LOOP         ;Check for carry
800A     74 FF             MOV     A,#0FFH      ;Move FFh into
                                                ;ACC.
800C     F0                MOVX    @DPTR,A
800D     02 00 00          LJMP    3
8010     74 00     LOOP:   MOV     A,#00H       ;Move 00h into
                                                ;ACC.
8012     F0                MOVX    @DPTR,A
8013     02 00 00          LJMP    3
```

10.8 Program to display largest number among 'N' numbers. Execute the program either in Stand-alone mode or in serial mode.

```
              13D2              PUTBYTE  EQU  13D2H

ADDRESS  OBJECT            MNEMONICS          COMMENTS

8000                       ORG     8000H
8000     90 89 00  LOOP4:  MOV     DPTR,#8900H  ;The total No.
                                                ;'N' of data
                                                ;bytes is
                                                ;stored in
                                                ;8900h data
                                                ;memory
8003     E0                MOVX    A,@DPTR
8004     FA                MOV     R2,A         ;Reg R2 is used
                                                ;as counter
```

```
8005    90 89 01          MOV     DPTR,#8901H   ;Data will be
                                                ;put from
                                                ;8901h onwards
8008    E0                MOVX    A,@DPTR
8009    1A                DEC     R2            ;Decrement
                                                ;counter
800A    F9                MOV     R1,A
800B    A3        LOOP2:  INC     DPTR          ;Increment data
                                                ;memory
800C    E0                MOVX    A,@DPTR
800D    FB                MOV     R3,A
800E    99                SUBB    A,R1          ;Comparing 2
                                                ;Nos.
800F    50 0B             JNC     LOOP1
8011    DA F8      LOOP3: DJNZ    R2,LOOP2
8013    E9                MOV     A,R1
8014    F5 71             MOV     71H,A
8016    12 13 D2          LCALL   PUTBYTE       ;Display the
                                                ;largest no.
8019    02 00 03          LJMP    3
801C    EB         LOOP1: MOV     A,R3
801D    F9                MOV     R1,A
801E    02 80 11          LJMP    LOOP3
8021    12 13 D2          LCALL   PUTBYTE        ;Display the
                                                 ;largest no.
8024    02 80 00          LJMP    LOOP4
```

10.9 Program to display decimal count 0 to 20. Execute the program either in stand-alone mode or in serial mode.

```
        03E6            DSPCHR          EQU     03E6H
        13D2            PUTBYTE         EQU     13D2H
        03BB            CLRLCD          EQU     03BBH
```

ADDRESS	OBJECT	MNEMONICS		COMMENTS
8000		ORG	8000H	
8000	7A 00	MOV	R2,#00H	;Store count 00 ;in R2
8002	90 E9 04 RPT:	MOV	DPTR,#0E904H	
8005	E0	MOVX	A,@DPTR	;Read the ;dipswitch
8006	30 E3 0B	JNB	ACC.3,L1	
8009	12 03 BB	LCALL	CLRLCD	;Clear the LCD ;display
800C	EA	MOV	A,R2	
800D	F5 71	MOV	71H,A	
800F	12 13 D2	LCALL	PUTBYTE	;Display the

```
                                                    ;count on LCD
8012    80 0F             SJMP    L2
8014    EA        L1:     MOV     A,R2
8015    F5 71             MOV     71H,A
8017    12 13 D2          LCALL   PUTBYTE     ;Display the
                                              ;count on PC
                                              ;console
801A    7B 03             MOV     R3,#03H
801C    74 08     LOOP:   MOV     A,#08H
801E    12 03 E6          LCALL   DSPCHR
8021    DB F9             DJNZ    R3,LOOP
8023    12 80 3A  L2:     LCALL   DELAY
8026    12 80 3A          LCALL   DELAY
8029    12 80 3A          LCALL   DELAY
802C    12 80 3A          LCALL   DELAY
802F    EA                MOV     A,R2
8030    24 01             ADD     A,#01H
8032    D4                DA      A             ;Perform decimal
                                                ;adjust ACC
8033    FA                MOV     R2,A
8034    BA 21 CB          CJNE    R2,#21H,RPT   ;Repeat the
                                                ;process till
                                                ;the count is
                                                ;equal to 20
8037    02 00 00          LJMP    0000H
803A    7B FF     DELAY:  MOV     R3,#0FFH      ;Delay routine
803C    7C FF             MOV     R4,#0FFH
803E    1B        BACK1:  DEC     R3
803F    BB 00 FC          CJNE    R3,#00H,BACK1
8042    1C        BACK2:  DEC     R4
8043    BC 00 F8          CJNE    R4,#00H,BACK1
8046    22                RET
```

10.10 Program to display 24 hours digital clock in serial mode. Execute the program from 8000H. To change the min, hours change the data in location 8009H & 8007H respectively.

```
0404            OUTPUT    EQU  0404H
13D2            PUTBYTE   EQU  13D2H
03E6            DSPCHR    EQU  03E6H
```

ADDRESS	OBJECT	MNEMONICS		COMMENTS
8000		ORG	8000H	
8000	90 80 F	MOV	DPTR,#STG	;Keep data from ;80F3H

```
                                                ;program memory.
8003    12 04 04          LCALL    OUTPUT    ;Display routine
8006    7D 23   START:    MOV      R5,#23H   ;Store hrs in R5
8008    7F 59             MOV      R7,#58H   ;Store minutes in
                                                ;R7 reg.
800A    ED      LOOP:     MOV      A,R5
800B    F5 71             MOV      71H,A
800D    C0 05             PUSH     5
800F    12 13 D2          LCALL    PUTBYTE   ;Routine to put
                                                ;character on
                                                ;console.
8012    74 20             MOV      A,#20H    ;Provide space.
8014    12 03 E6          LCALL    DSPCHR
8017    74 20             MOV      A,#20H
8019    12 03 E6          LCALL    DSPCHR
801C    EF                MOV      A,R7
801D    F5 71             MOV      71H,A
801F    12 13 D2          LCALL    PUTBYTE
8022    74 20             MOV      A,#20H
8024    12 03 E6          LCALL    DSPCHR
8027    74 20             MOV      A,#20H
8029    12 03 E6          LCALL    DSPCHR
802C    D0 05             POP      5
802E    C0 05   BEGIN:    PUSH     5
8030    79 50             MOV      R1,#00H   ;Keep seconds 00
8032    E9      SEC:      MOV      A,R1      ;In reg R2
8033    F5 71             MOV      71H,A
8035    12 13 D2          LCALL    PUTBYTE
8038    7B 02             MOV      R3,#02H
803A    74 08   LOOP1:    MOV      A,#08H
803C    12 03 E6          LCALL    DSPCHR
803F    DB F9             DJNZ     R3,LOOP1
8041    12 80 92          LCALL    DELAY     ;Delay Routine
8044    E9                MOV      A,R1
8045    24 01             ADD      A,#01H
8047    D4                DA       A
8048    F9                MOV      R1,A
8049    B9 60 E6          CJNE     R1,#60H,SEC;Check for 60 sec
                                                ;over or not
804C    EF                MOV      A,R7
804D    24 01             ADD      A,#01H
804F    D4                DA       A
8050    FF                MOV      R7,A
8051    BF 60 51          CJNE     R7,#60H,MIN ;Check for 60
                                                ;min over or not
8054    7B 07             MOV      R3,#07H
8056    74 08   LOOP2:    MOV      A,#08H
```

```
8058    12 03 E6          LCALL    DSPCHR
805B    DB F9             DJNZ     R3,LOOP2
805D    74 01             MOV      A,#01H
805F    FF                MOV      R7,A
8060    F5 71             MOV      71H,A
8062    12 13 D2          LCALL    PUTBYTE
8065    7B 07             MOV      R3,#07H
8067    74 08   LOOP3:    MOV      A,#08H
8069    12 03 E6          LCALL    DSPCHR
806C    DB F9             DJNZ     R3,LOOP3
806E    D0 05             POP      5
8070    ED                MOV      A,R5
8071    24 01             ADD      A,#01H
8073    D4                DA       A
8074    FD                MOV      R5,A
8075    BD 24 53          CJNE     R5,#24H,HRS;Check for 24
                                                ;hrs. over or not
8078    7B 04             MOV      R3,#04H
807A    74 08   LOOP4:    MOV      A,#08H
807C    12 03 E6          LCALL    DSPCHR
807F    DB F9             DJNZ     R3,LOOP4
8081    74 00             MOV      A,#00H
8083    F5 71             MOV      71H,A
8085    12 13 D2          LCALL    PUTBYTE
8088    7D 00             MOV      R5,#00H
808A    7F 00             MOV      R7,#00H
808C    12 80 C1          LCALL    CURSOR
808F    02 80 0A          LJMP     LOOP
8092    7A 05   DELAY:    MOV      R2,#05H    ;Delay routine
8094    7C FF   BACK3:    MOV      R4,#0FFH
8096    7B FF   BACK2:    MOV      R3,#0FFH
8098    1B      BACK1:    DEC      R3
8099    BB 00 FC          CJNE     R3,#00H,BACK1
809C    1C                DEC      R4
809D    BC 00 F6          CJNE     R4,#00H,BACK2
80A0    1A                DEC      R2
80A1    BA 00 F0          CJNE     R2,#00H,BACK3
80A4    22                RET
Subroutine to display minutes.
80A5    7B 04    MIN:     MOV      R3,#04H
80A7    74 08   RPT4:     MOV      A,#08H
80A9    12 03 E6          LCALL    DSPCHR
80AC    DB F9             DJNZ     R3,RPT4
80AE    EF                MOV      A,R7
80AF    F5 71             MOV      71H,A
80B1    12 13 D2          LCALL    PUTBYTE
80B4    74 20             MOV      A,#20H
```

```
80B6    12 03 E6          LCALL    DSPCHR
80B9    74 20             MOV      A,#20H
80BB    12 03 E6          LCALL    DSPCHR
80BE    02 80 2E          LJMP     BEGIN
Subroutine to move the cursor three times back.
80C1    7B 03  CURSOR:    MOV      R3,#03H
80C3    74 08    RPT5:    MOV      A,#08H
80C5    12 03 E6          LCALL    DSPCHR
80C8    DB F9             DJNZ     R3,RPT5
80CA    22                RET
Subroutine to display hours.
80CB    ED        HRS:    MOV      A,R5
80CC    F5 71             MOV      71H,A
80CE    12 13 D2          LCALL    PUTBYTE
80D1    74 20             MOV      A,#20H
80D3    12 03 E6          LCALL    DSPCHR
80D6    74 20             MOV      A,#20H
80D8    12 03 E6          LCALL    DSPCHR
80DB    74 00             MOV      A,#00H
80DD    F5 71             MOV      71H,A
80DF    12 13 D2          LCALL    PUTBYTE
80E2    7F 00             MOV      R7,#00H
80E4    7E 00             MOV      R6,#00H
80E6    74 20             MOV      A,#20H
80E8    12 03 E6          LCALL    DSPCHR
80EB    74 20             MOV      A,#20H
80ED    12 03 E6          LCALL    DSPCHR
80F0    02 80 2E          LJMP     BEGIN
Subroutine to display HRS, MIN, SEC on the console.
                  STG:    DB      'HRS MIN SEC',0DH,0AH,0
80F3    48 52 53 20 4D
80F8    49 4E 20 53 45
80FD    43 0D 0A 00
8101    02 80 06          LJMP     START
```

10.11 Program to perform addition of two numbers. This program can be executed in serial mode or in stand-alone mode. Two numbers are taken from locations 9000H & 9001H of Data memory. They are added and the result is stored in the location 9002H of data memory.

```
        13D2            PUTBYTE    EQU    13D2H

ADDRESS  OBJECT             MNEMONICS                COMMENTS

 8000                       ORG     8000H
 8000    90 90 00           MOV     DPTR,#9000H  ;Keep the data
```

```
                                                    ;in 9000h&
                                                    ;9001h locations
                                                    ;of data memory
8003    E0                  MOVX    A,@DPTR
8004    F5 F0               MOV     0F0H,A
8006    90 90 01            MOV     DPTR,#9001H
8009    E0                  MOVX    A,@DPTR
800A    25 F0               ADD     A,0F0H      ;Add them store
800C    90 90 02            MOV     DPTR,#9002H  ;the result in
                                                ;9002h location
                                                ; of data memory
800F    F0                  MOVX    @DPTR,A
8010    F5 71               MOV     71H,A
8012    12 13 D2            LCALL   PUTBYTE   ;Display the result
8015    02 00 00            LJMP    0
```

10.12 Program to perform subtraction of two numbers. This program can be executed either in stand-alone mode or in serial mode. Two numbers are taken from locations 9000H & 9001H of data memory. They are subtracted and the result is stored in the location 9002 of data memory.

```
    13D2          PUTBYTE  EQU 13D2H
```

```
ADDRESS  OBJECT             MNEMONICS                COMMENTS

8000                        ORG     8000H
8000    90 90 01            MOV     DPTR,#9001H ;Keep data in
8003    E0                  MOVX    A,@DPTR     ;9000h and 9001H
8004    F5 F0               MOV     0F0H,A      ;location of Data
                                                ;Memory
8006    90 90 00            MOV     DPTR,#9000H
8009    E0                  MOVX    A,@DPTR
800A    95 F0               SUBB    A,0F0H     ;Subtract them
800C    90 90 02            MOV     DPTR,#9002H;Store the result
800F    F0                  MOVX    @DPTR,A    ;in 9002H location
8010    F5 71               MOV     71H,A      ;of Data Memory
8012    12 13 D2            LCALL   PUTBYTE
8015    02 00 00            LJMP    0
```

10.13 Demonstration program for on-board DAC. The program will generate a square wave output at DAC O/P(J12).

```
ADDRESS  OBJECT             MNEMONICS                COMMENTS

8000                        ORG     8000H
8000    75 A0 E1            MOV     P2,#0E1H
8003    78 00               MOV     R0,#00H   ;8155 CONTROL PORT
```

```
8005    74 02              MOV     A,#02H    ;CONFIGURE 8155
                                             ;PORTB AS O/P PORT
8007    F2                 MOVX    @R0,A
8008    08                 INC     R0
8009    08                 INC     R0
800A    74 FF      AGAIN:  MOV     A,#FFH
800C    F2                 MOVX    @R0,A     ;OUTPUT FFH TO PORTB
                                             ;FOR 5V AT DAC O/P
800D    12 80 17           CALL    DELAY     ;CALL DELAY ROUTINE
8010    E4                 CLR     A
8011    F2                 MOVX    @R0,A     ;OUTPUT 00H TO PORTB
                                             ;FOR 0V AT DAC O/P
8012    12 80 17           CALL    DELAY     ;CALL DELAY ROUTINE
8015    80 F3              SJMP    AGAIN     ;REPEAT THE PROCESS
8017    7E FF      DELAY:  MOV     R6,#FFH   ;THE DELAY ROUTINE
8019    DE FE      BACK:   DJNZ    R6,BACK
801B    22                 RET
```

10.14 Demonstration program for 12bit ADC for both multi & single channel operation. When the ADC is operated in single channel mode, the MUX at U47 need not be populated and JP15 & JP24 are open. Analog input is applied at screw terminal TP. When the ADC is in multi channel mode, the MUX is populated and eight channels are available as selected by the channel select lines. The jumpers JP11 to JP15 are closed. The analog signals are applied at the screw terminals provided at J11.When the ADC is in multi channel mode no signal is applied at TP.

```
        13D2            PUTBYTE EQU     13D2H
        12BB            GETCH   EQU     12BBH
        0404            OUTPUT  EQU     404H
```

```
ADDRESS  OBJECT            MNEMONICS                COMMENTS

  8000                     ORG     8000H
  8000   E4                CLR     A
  8001   90 90 00          MOV     DPTR,#9000H
  8004   F0                MOVX    @DPTR,A ;CLEAR THE DATA
                                           ;MEMORY LOCATION 9000H
                                           ;TO STORE THE
                                           ;CHANNEL NO:
  8008   78 00              MOV     R0,#00H ;MAKE THE 8155 PORTA
  800A   74 02              MOV     A,#02H  ;&PORT B AS OUTPUT
  800C   F2                 MOVX    @R0,A   ;PORTS
  800D   75 90 03           MOV     90H,#03H;MUX&LATCHES ARE
                                           ;DISABLED&ADC IS IN
```

```
                                              ;READ MODE
8010   E0          CONVERT:MOVX    A,@DPTR
8011   23                  RL      A
8012   23                  RL      A       ;TO SELECT THE CHANNEL
8013   F5 F0               MOV     B,A     ;SAVE THE CHANNEL
                                              ;VALUE IN B REG
8015   44 23               ORL     A,#23H ;THE CONTROL SIGNALS-
                                              ;MUX ENABLED LATCHES
                                              ;DISABLED ADC IS IN
                                              ;READ MODE
8017   F5 90               MOV     90H,A
8019   02 80 B2            LJMP    SETLTIME;AFTER ANALOG SIGNAL
                                              ;IS APPLIED THE MUX IS
                                              ;GIVEN THE REQUIRED
                                              ;SETTLING TIME
801C   E5 F0         BACK: MOV     A,B
801E   44 23                ORL     A,#23H
8020   F5 90                MOV     90H,A   ;GIVE THE HIGH TO LOW
                                              ;FOR CONVERSIONSTART
8022   54 FC                ANL     A,#0FCH ;& THE LATCHES
8024   F5 90                MOV     90H,A  ;ARE ALSO ENABLED
8026   00                   NOP             ;THE CONVERSION
8027   00                   NOP             ;TIME
8028   00                   NOP
8029   D2 90                SETB    P1.0   ;ENABLE CONVERSION
802B   78 01                MOV     R0,#01H;ACCEPT THE LOWER 8
802D   E2                   MOVX    A,@R0  ;BIT DATA THROUGH PORT
                                              ;A OF 8155
802E   90 90 01             MOV     DPTR,#9001H;AND STORE IT
                                                  ;IN DATA
8031   F0                   MOVX    @DPTR,A   ;MEMORY 9001
                                              ;LOCATION
8032   08                   INC     R0         ;ACCEPT THE
8033   08                   INC     R0         ;UPPER 4 BIT DATA
8034   E2                   MOVX    A,@R0    ;FROM PORT C OF 8155
8035   54 0F                ANL     A,#0FH
8037   90 90 02             MOV     DPTR,#9002H ;AND STORE IT IN
803A   F0                   MOVX    @DPTR,A  ;DATA MEMORY 9002
                                              ;LOCATION
803B   90 80 8F    SRL:     MOV     DPTR,#CHANNEL
803E   C2 D5                CLR     F0      ;TO SELECT PROGRAM
                                              ;MEMORY
8040   12 04 04             LCALL   OUTPUT  ;DISPLAY THE MESSAGE
                                              ;"CHANNEL ="
8043   90 90 00             MOV     DPTR,#9000H ;DISPLAY
8046   E0                   MOVX    A,@DPTR    ;THE
8047   F5 71                MOV     71H,A      ;CHANNEL
```

```
8049   12 13 D2           LCALL   PUTBYTE    ;NO:
804C   90 80 9C           MOV     DPTR,#DIGITAL   ;
804F   C2 D5              CLR     F0  ;TO SELECT PROGRAM MEMORY
8051   12 04 04           LCALL   OUTPUT   ;DISPLAY THE MESSAGE
                                           ;"DIGITAL VALUE ="
8054   90 90 02           MOV     DPTR,#9002H  ;DISPLAY
8057   E0                 MOVX    A,@DPTR     ;THE UPPER 4 BITS
8058   F5 71              MOV     71H,A      ;OF THE
805A   12 13 D2           LCALL   PUTBYTE    ;CONVERTED DATA
805D   90 90 01           MOV     DPTR,#9001H ;DISPALY THE
8060   E0                 MOVX    A,@DPTR     ;LOWER 8 BITS
8061   F5 71              MOV     71H,A       ;OF THE
8063   12 13 D2           LCALL   PUTBYTE     ;CONVERTED DATA
8066   12 12 BB   S2:     LCALL   GETCH       ;GET A CHARACTER
                                           ;FROM KEYBOARD
8069   B4 2C 02           CJNE    A,#2CH,S1   ; CHECK WHETHER
806C   80 13              SJMP    CHINR       ;IF YES INCREMENT
                                           ;THE CHANNEL NO:
806E   B4 2D 02   S1:      CJNE    A,#2DH,S6  ;IF NOT ',' CHECK
                                           ;FOR '-'
8071   80 07              SJMP    CHDCR      ;IF YES  DECREMENT
                                           ;THE CHANNEL NO:
8073   64 0D      S6:     XRL     A,#0DH     ;IF IT IS <CR>?
8075   60 EF              JZ      S2         ;YES,WAIT FOR THE
                                           ;NEXT KEY TO BE
                                           ;PRESSED
8077   02 00 03           LJMP    3       ;ELSE, GO TO INT0
                                          ;ROUTINE.

807A   90 90 00   CHDCR:  MOV     DPTR,#9000H ;SUBROUTINE
807D   E0                 MOVX    A,@DPTR     ;TO
807E   14                 DEC     A           ;DECREMENT THE
807F   80 05              SJMP    ST          ;CHANNEL NO:
8081   90 90 00   CHINR:  MOV     DPTR,#9000H ;SUBROUTINE
8084   E0                 MOVX    A,@DPTR     ;TO
8085   04                 INC     A           ;INCREMENT THE
                                           ;CHANNEL NO:
8086   54 07      ST:     ANL     A,#07H          ;
8088   F0                 MOVX    @DPTR,A  ;STORE THE CHANNEL
                                          ;NO.IN 9000 LOCATION
                                          ; OF DATA MEMORY
8089   75 A0 E1           MOV     P2,#0E1H
808C   02 80 10           LJMP    CONVERT  ;REPEAT THE PROCESS
808F   0D 0A 43 48 41  CHANNEL:   DB     0DH,0AH,'CHANNEL = ',0
8094   4E 4E 45 4C 20
8099   3D 20 00
809C   20 20 20 20 20   DIGITAL:  DB  '     DIGITAL VALUE = ',0
```

```
80A1   44 49 47 49 54
80A6   41 4C 20 56 41
80AB   4C 55 45 20 3D
80B0   20 00
80B2                    SETLTIME:
80B2   00                       NOP
80B3   00                       NOP
80B4   00                       NOP
80B5   00                       NOP
80B6   00                       NOP
80B7   00                       NOP
80B8   00                       NOP
80B9   00                       NOP
80BA   00                       NOP
80BB   00                       NOP
80BC   02 80 1C                 LJMP    BACK
```

APPENDIX A

COMPONENT PLACEMENT DIAGRAM

APPENDIX B

ASCII CODES

APPENDIX C

RS 232C/RS 485
CABLE REQUIREMENTS

APPENDIX D

PRODUCT LIST

APPENDIX E

INSTRUCTION SET

APPENDIX F

CONNECTOR DETAILS

RS 232 C CABLE DETAILS

THE ESA 85-2 REQUIRES A NULL MODEM CABLE IN ORDER TO COMMUNICATE WITH OTHER SYSTEMS.

THESE ARE THE CONNECTIONS REQURIED FOR THE NULL-MODEM CABLE :

TXD (Pin 2) > ..... < RXD (pin 3)

RXD (Pin 3) > ..... < TXD (pin 2)

RTS (Pin 4) > ..... < CTS (pin 5)

CTS (Pin 5) > ..... < RTS (pin 4)

DSR (Pin 6) > ..... < DTR (pin 2)

DTR (Pin 20) > ..... < DSR (pin 6)

GND (Pin 7) > ..... < GND (pin 7)

NOTE :

1) Use male of 25 PIN 'D' Connector on ESA85-2 side and appropriate on other side.

2) If hardware handshaking is not required interconnect RTS and CTS (PIN 4 and 5) and DSR and DTR (PIN 6 and 20).

APPENDIX B-1

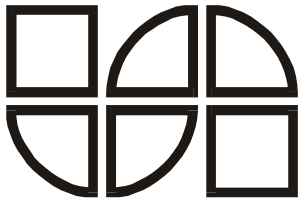
Hexadecimal	Decimal	Character	Hexadecimal	Decimal	Character
00	0	NUL	20	32	SP
01	1	SOH	21	33	!
02	2	STX	22	34	"
03	3	ETX	23	35	#
04	4	EOT	24	36	$
05	5	ENQ	25	37	%
06	6	ACK	26	38	&
07	7	BEL	27	39	,
08	8	BS	28	40	(
09	9	HT	29	41	)
0A	10	LF	2A	42	*
0B	11	VT	2B	43	+
0C	12	FF	2C	44	,
0D	13	CR	2D	45	-
0E	14	S0	2E	46	.
0F	15	SI	2F	47	/
10	16	DLE	30	48	0
11	17	DCI	31	49	1
12	18	DC2	32	50	2
13	19	DC3	33	51	3
14	20	DC4	34	52	4
15	21	NAK	35	53	5
16	22	SYN	36	54	6
17	23	ETB	37	55	7
18	24	CAN	38	56	8
19	25	EM	39	57	9
1A	26	SUB	3A	58	:
1B	27	ESC	3B	59	;
1C	28	FS	3C	60	<
1D	29	GS	3D	61	=
1E	30	RS	3E	62	>
1F	31	US	3F	63	?

APPENDIX B-2

Hexadecimal	Decimal	Character	Hexadecimal	Decimal	Character
40	64	@	60	96	,
41	65	A	61	97	a
42	66	B	62	98	b
43	67	C	63	99	c
44	68	D	64	100	d
45	69	E	65	101	e
46	70	F	66	102	f
47	71	G	67	103	g
48	72	H	68	104	h
49	73	I	69	105	j
4A	74	J	6A	106	i
4B	75	K	6B	107	k
4C	76	L	6C	108	l
4D	77	M	6D	109	m
4E	78	N	6E	110	n
4F	79	O	6F	111	o
50	80	P	70	112	p
51	81	Q	71	113	q
52	82	R	72	114	r
53	83	S	73	115	s
54	84	T	74	116	t
55	85	U	75	117	u
56	86	V	76	118	v
57	87	W	77	119	w
58	88	X	78	120	x
59	89	Y	79	121	y
5A	90	Z	7A	122	z
5B	91	[	7B	123	{
5C	92	/	7C	124	|
5D	93	[	7D	125	}
5E	94	^	7E	126	~
5F	95	-	7F	127	DEL

SCHEMATICS

ESA 51 USER'S MANUAL



ELECTRO SYSTEMS ASSOCIATES PVT LTD.,

4215 J.K. Complex, First Main Road, Subramanyanagar
P.O. Box No. 2139 BANGALORE - 560 021 INDIA
Fax : 91-80-3325615 Phone : 3323029 3322924
email : esaindia@vsnl.com,
www.esaindia.com