

---

# **Topview Trainer**

**Embedded System Trainer for 8031/8051 Microcontrollers**

## **Hardware User Guide**

***FRONTLINE***  
E L E C T R O N I C S

---

---

Copyright © 2002 Frontline Electronics Pvt Ltd. All Rights Reserved.

Information in this document is subject to change without notice. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose without the express written permission of Frontline Electronics Pvt Ltd.

---

# Contents

## **Chapter 1: Introduction**

1.1	Welcome .....	1
1.2	For Technical or Customer Support.....	2
1.3	Packaging .....	3

## **Chapter 2: Specifications**

2.1	Trainer Specifications .....	5
2.2	Software Specifications .....	6

## **Chapter 3: Hardware Description**

3.1	Introduction .....	9
3.2	Microcontroller .....	11
3.3	RAM .....	12
3.4	EPROM/Microcontroller Flash Memory .....	13
3.5	Keyboard & Seven Segment Display .....	14
3.6	LED Indicators .....	15
3.7	Switches .....	15
3.8	Serial Port .....	16
3.9	Bus Connector .....	16
	Trainer PCB Layouts .....	17
	Schematic Diagram of Topview Trainer .....	21

## **Chapter 4: Keyboard Operations**

4.1	Introduction .....	31
4.2	Enter / View a Program .....	36
4.3	Execute a Program .....	39
4.4	Examine Register .....	42
4.5	Single Step Execution and Debugging .....	48
4.6	Examine/Modify Internal Data Memory .....	50
4.7	Data Fill ( External Memory ) .....	52
4.8	Block Copy ( External Memory ) .....	54
4.9	Data Fill ( Internal Memory ) .....	56

---

4.10	Block Copy ( Internal Memory ) .....	58
4.11	Block Copy ( From Internal Memory to External Memory ) .....	60
4.12	Block Copy ( From External Memory to Internal Memory ) .....	62
4.13	Load Examples .....	64
 <b>Chapter 5: Keyboard and Display</b>		
5.1	Introduction .....	67
5.2	Seven Segment Display .....	67
5.3	Keyboard .....	69
 <b>Chapter 6: Connector and Jumper Details</b>		
6.1	Introduction .....	71
 <b>Chapter 7 Add-on Board</b>		
7.1	Introduction .....	73
7.2	I <sup>2</sup> C Bus Based EEPROM and RTC .....	74
7.3	LCD Module Interfacing .....	75
7.4	Parallel Printer Port .....	76
7.5	High Performance ADC Circuit.....	77
7.6	Stepping Motors .....	78
7.7	Input/output Port lines .....	78
7.8	Digital to Analog Converter .....	79

### 1.1 Welcome

Thank you for purchasing Topview Trainer ( Model : MAGNA 31C ) using Atmel's AT89C52 microcontroller. This trainer aids you learn about 8031/8051 by providing you with the following:

- A field proven Hardware trainer based on Atmel's AT89C52 microcontroller.
- On-board Facility to study the interrupts and the timers of the controller.
- A GUI based development software Topview Debugger with support for editing and debugging target programs.
- All relevant information, detailed manual, plenty of examples and routines are provided to enable even the first time users get a flying start on 8031/8051.
- Facility is also provided for the experienced user to develop his/her real life projects using this trainer.

## 1.2 For Technical or Customer Support

You can reach Frontline Electronics Pvt, Ltd for the technical support and application assistance in following ways:

Email questions to: **feplslm@frontlinemail.com**

Send questions by mail to:

Frontline Electronics Pvt Ltd.,  
2/30 - Paraivattam,  
Alagapuram,  
Salem - 636 016,  
Tamilnadu,  
India.

Phone : 0427 2449238 / 2431312.

Fax : 0427 2449010.

Web site : **www.Frontline-Electronics.com**

### 1.3 Packaging

Following documents / hardware are supplied:

1. Topview Trainer Hardware User Guide.
2. Software Examples Manual.
3. Topview Debugger Software User Guide.
4. Hardware :
  - Magna31C Trainer
  - Serial Port Communication Cable.
  - CDROM with Topview Debugger.
  - DC Regulated Power Supply meant for the trainer.





### 2.1 Trainer Specifications

- AT89C52 operating at 12 MHz. AT89CXX family of 8031 devices from Atmel is the most popular 8031 derivative having same instruction set and architecture of the original 8031/8051 microcontrollers introduced by Intel. Also AT89CXX devices are freely available in open market everywhere. These devices are comparatively cheaper than many others.
- AT89C52's On-chip Flash memory (8K X 8 Capacity) is programmed with the monitor program that is responsible for the trainer's operation.
- Optional EPROM with 32KB capacity. A separate blank socket is provided for this purpose.
- RAM with 32KB capacity. Battery backup for the RAM is provided.
- Keyboard with 24 high performance keys.
- 6 digits of seven segment green colour LED displays.
- Serial Communication port with PC that enables the debugging facilities in the host personal computer.
- Four push button switches are available on board for the study of Interrupts (INT0 and INT1) and Timers 0 and 1.
- 6 LED indicators and 2 Toggle switches are also available on board for experimental purpose.
- All standard trainer debugging facilities are available.
- Bus lines and I/O lines are terminated for user's convenience.

## **2.2 Software Specifications**

MAGNA 31C sports a powerful and high performance system monitor responsible for the Trainer's operation. This monitor stays in the AT89C52's internal Flash Program Memory space of 8K byte capacity. This system software enables the trainer to operate in two different modes:

- Stand-alone Mode.
- PC based Host controlled mode.

When operated as the stand-alone mode, you can interact with this trainer using the keypad available in the trainer. This keyboard and seven segment LED displays are meant for user's interaction.

The PC based host controlled mode gives a more sophisticated and versatile development environment suitable for fast target program development.

- To enter / view program
- Execution with full speed
- Single step execution for debugging purpose

Monitor also provides other useful functions, which improves the performance of the trainer and makes the trainer user friendly.

- Block copy function (in Internal/External RAM)
- Block fill function (in Internal/External RAM)
- Edit Internal/External Data Memory
- Edit/View Special Function Registers.
- Block copy from external RAM to internal RAM
- Block copy from internal RAM to external RAM
- Load Examples

### Software Specifications

Topview Debugger, a GUI based development environment enables you to gain confidence in using 8031 microcontrollers. If you are a beginner, you can find it as a most handy tool that makes your learning as fun and you get confidence in few hours using the debugger. If you are an experienced user, Debugger gives you the required development power when understanding real life challenging projects.

**Note:**

The RAM locations FD00H to FEFFH is used internally by the system monitor. Hence take care while using RAM, otherwise you will get unexpected errors.



### 3.1 Introduction

The Topview Trainer is designed using Atmel's 8031 derivative device, AT89C52 operating at 12MHz. This device is the improved 8031 derivative of original 8051 architecture introduced by INTEL.

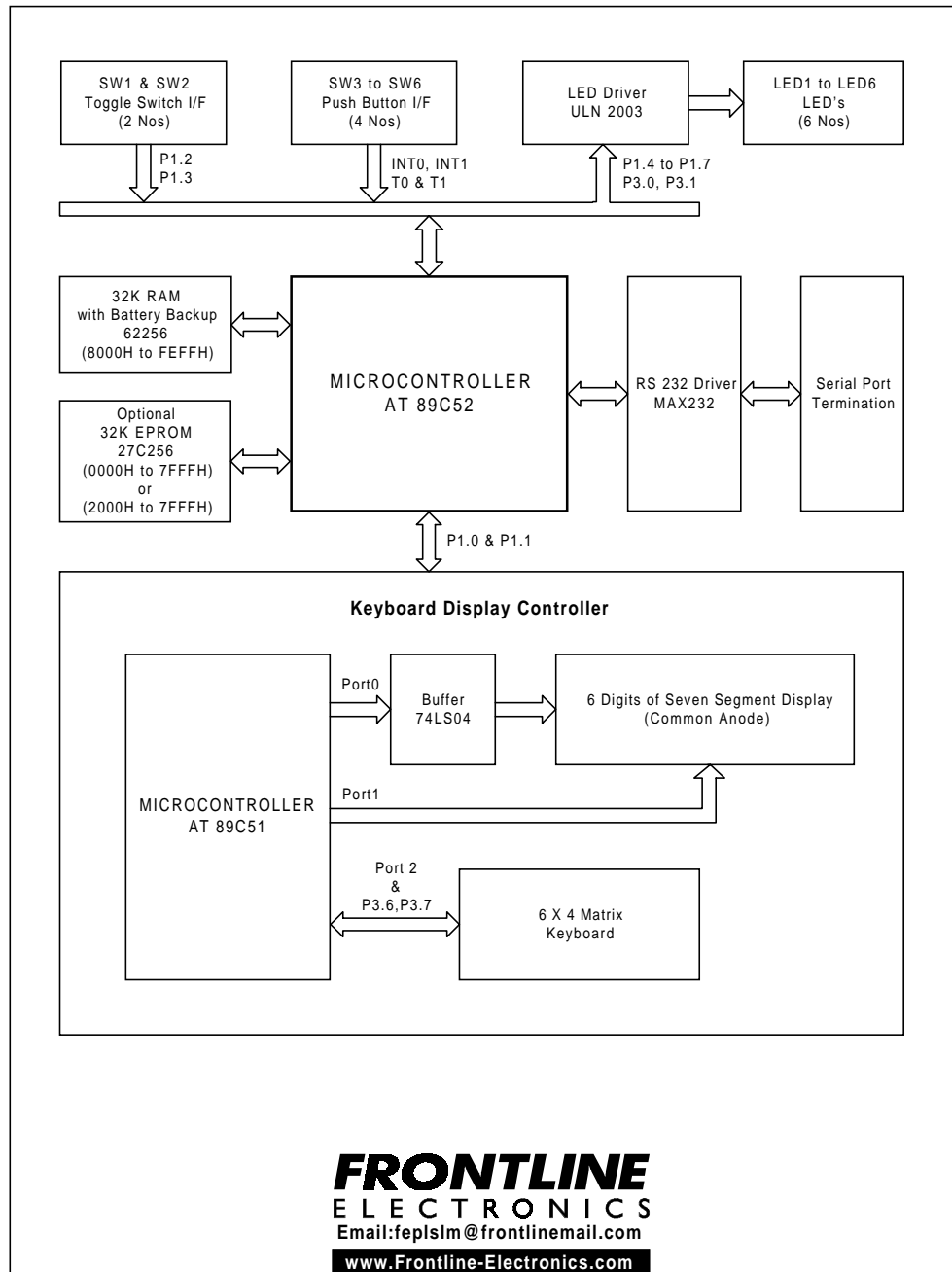
AT89C52 sports an internal Flash Program Memory of 8K byte in the device itself. This 8KB memory is mapped in the lower part of the Program Memory with address: 0000H - 1FFFFH. Rest of the memory space can be obtained using external devices. When the  $\overline{EA}$  pin of the device is kept at the active low level, then the controller accesses the entire Program Memory from outside and the internal flash memory will not come into picture.

The Keyboard/Display PCB holds the 24 matrix keys, 6 digits of seven segment displays and a dedicated controller which is also an AT89C51 for controlling the keyboard and display.

The different blocks of the trainer are

- Microcontroller
- RAM
- EPROM (optional)
- Keyboard and 7 segment displays
- LED Indicators
- Switches and keys
- Serial Port
- Bus Connector
- Power Supply

The block diagram of the trainer is shown below:



### **3.2 Microcontroller**

In the 8 bit Microcontrollers, the most popular and used architecture is 8031. The 8031 architecture supports a Harvard architecture, which contains two separate buses for both program and data. So, it has two distinct memory spaces of 64K X 8 size for both program and data. It is based on an 8 bit central processing unit with an 8 bit Accumulator and another 8 bit B register as main processing blocks. Other portions of the architecture include few 8 bit and 16 bit registers and 8 bit memory locations.

Each microcontroller has some amount of data RAM built-in the device for processing. Usually this memory space is mapped to the lower end of the data memory. This base architecture is supported with on-chip peripheral functions like I/O ports, timers/counters and a versatile serial communication port.

The following list gives the features of the 8031 architecture:

- 8 bit Harvard architecture with separate Program Memory and Data Memory.
- Single supply 5V operation.
- 8K x 8 size On-chip Program Memory - Flash (Atmel's AT89C52).
- 256 bytes of On-chip Data Memory.
- 64K x 8 bytes of Program Memory and 64K x 8 bytes of Data Memory when expanded.
- One microsecond instruction cycle using 12 MHz clock.
- 32 Bi-directional I/O lines organized as four 8 bit ports.
- Multiple mode, high speed programmable serial port.
- Two multiple mode 16 bit Timers / Counters.
- Two level prioritized interrupt structure.
- Full depth stack for sub routine linkage and data storage.
- Direct byte and bit addressable.
- Binary and decimal arithmetic.
- Signed overflow detection and parity computation.
- Hardware multiply and divide in 4 micro seconds.
- Integrated Boolean processor for control applications.

### Microcontroller

An instruction set comprising of 140 one byte instructions, 91 two byte instructions, and 24 three byte instructions. Virtually all instructions execute in either one or two instruction cycles - one or two microseconds with a 12 MHz crystal. Only exceptions are multiply and divide instructions that take four instruction cycles.

The instruction set is designed to make programs efficient both in terms of code size and execution speed.

### 3.3 RAM

A RAM chip of 32K capacity, IC 62256 is used.

To edit and execute a program we require a memory area with both code and data memory accesses. Hence overlapping of code and data memory is provided for this RAM area.

The address of the RAM is 8000H to FEFFH.

The remaining area FF00H to FFFFH is left free for interfacing I/O devices.

The monitor uses some RAM area FD00H to FEFFH (2 pages) for its internal operation. Hence the available user RAM area is from 8000H to FCFFH.

Battery backup facility is provided for the RAM to retain the data during power failure.



### 3.4 EPROM/Microcontroller Flash Memory

The microcontroller's  $\overline{EA}$  pin defines the type of internal or external memory that comes into action during program executions.

When the  $\overline{EA}$  pin is connected to high level, the first 8K byte (0000H - 1FFFFH) accesses are directed to the internal Flash memory of the microcontroller and the remaining memory space, 2000H to 7FFFFH are mapped into the external EPROM. As you can notice, in this arrangement first 8K byte of EPROM stays unused.

When the  $\overline{EA}$  pin is connected to the ground, the internal Flash memory doesn't come into picture and the entire access will be directed to the external EPROM memory with address range, 0000H to 7FFFFH.

To make this selection a convenient one, a jumper JP1 is provided. Using this jumper, you can connect the  $\overline{EA}$  either to high level, 5V or ground level. The details are given in the chapter 6, "Connectors and Jumpers".

As you know, AT89C52 contains 8K X 8 size internal Flash memory to keep program instructions. Topview Trainer keeps the monitor program in this Flash memory.

Also there is a provision made in the trainer to accommodate an external EPROM with size 32K X 8. You can fuse this EPROM with your target program when you are using this trainer in real life applications.

When you use this external EPROM, you can make use of Topview Debugger for application developing.

For your information, AT89C52's Program and Data Memory spaces are overlapped and a total of 64K X 8 memory space is available ( instead of total 64K + 64K memory space possible using a 8031 microcontroller ) for your applications.

### EPROM/Microcontroller Flash Memory

This 64K byte space is divided into two parts. First 32K byte ( Address : 0000H to 7FFFH ) is occupied by the external EPROM and balance 32K byte ( 8000H to FFFFH ) is occupied by the static RAM of 32K byte.

### 3.5 Keyboard & Seven Segment Display

This is an interesting part of the Topview Trainer. We have used a dedicated Microcontroller, AT89C51 to implement the required keymatrix and the 7 segment displays. This is a simple example of how AT89CXX devices can be used in unexpected places.

Normally, Microcontroller/microprocessor trainers use the popular keyboard/display controller, 8279 of Intel origin to implement the required user interaction facility. These days Intel has almost stopped manufacturing the device and whatever devices we could purchase from the local market come from the used equipments. Apart from this, the cost of 8279 is comparatively on the higher side.

So to avoid these problems, we decided to use AT89C51 to get keyboard and display facility.

The secondary Microcontroller acts as a slave to the main AT89C52 controller. It monitors the keymatrix and drives the multiplexed seven segment displays as per its master's commands.

The main controller communicates with the secondary controller through the serial port. Since the serial port of the main controller is kept aside for the PC host communication purpose, another serial port is simulated in two I/O lines ( P1.0 and P1.1 ) of its Port1. This simulated serial port communicates with secondary controller's regular serial port. The baud rate for this communication is about 2400 bits/second.

The seven segment displays are connected in multiplexed form and connected with ports 0 and 1 of the controller. Port 0 drives the segments and port1

### Keyboard & Seven Segment Display

manages the display digits. The 24 keys are connected in the keymatrix and are connected with the ports, port 2 and port 3.

More information on how to interface these multiplexed seven segment displays and the keyboard matrix is given in the following chapters.

### 3.6 LED Indicators

LED indicators are connected to port lines using a driver IC ULN2003. The 4 lines of Port 1 (P1.4 to P1.7) and 2 lines of port 3 (P3.0 and P3.1) are used for this purpose. These LED's are provided on board for experimental purpose.

To switch ON an LED, send '1' level through the respective port line.

Refer to the Software Examples Manual, where plenty of examples are provided.

### 3.7 Switches

Four numbers of push button switches are connected to port lines P3.2, P3.3, P3.4 and P3.5.

These switches are connected to the port lines for experimental purpose. These switches enable you to conduct experiments using timers (T0 and T1) and interrupts (INT0 and INT1).

Similarly 2 numbers of toggle switches are made available onboard for experimental purpose. Port lines P1.2 and P1.3 are used for this purpose.

The signal levels for switch positions:

(for both toggle and push button switches)

ON (Pressed) - '0' level

OFF(Not Pressed) - '1' level

### 3.8 Serial Port

The serial port of the trainer is driven from the serial I/O of the AT89C52. This serial port is a full duplex one available in two port lines P3.0 (RXD) and P3.1 (TXD).

The serial port is used mainly to communicate with the IBM PC. The Topview Debugger from the PC communicates with the trainer through the serial port.

The connector details are given in the chapter 6.

### 3.9 Bus Connector

The Bus lines (address bus and data bus), all lines of port1 and port3, I/O enable line which helps to interface I/O devices (with address space FF00H to FFFFH), Vcc and Ground lines are terminated in a separate connector called CON1.

This facilitates user to interface I/O devices externally for any project development. The I/O Enable line will give a low level when the address fetched by microcontroller is FFXXH (FF00H to FFFFH). Based on this address decoding, I/O devices can be connected externally.

The connector details are given in the chapter 6.

### 3.10 Power Supply

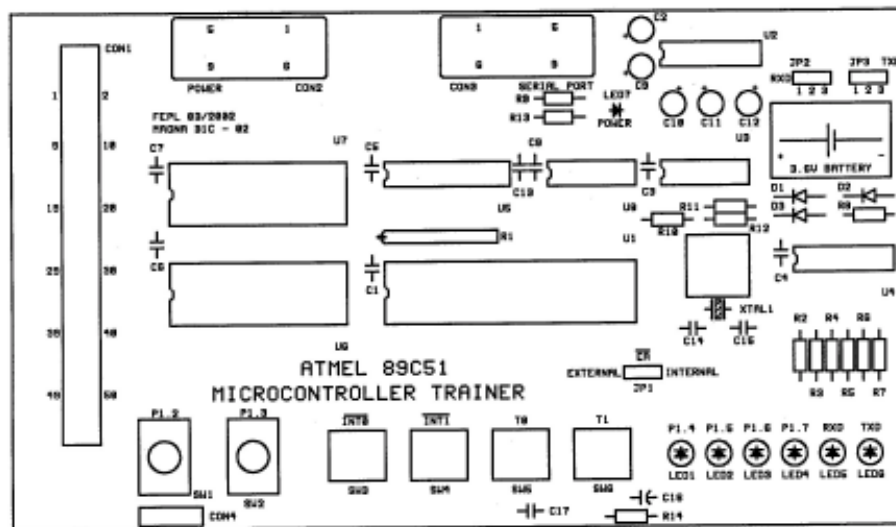
A regulated power supply of 5V, 500mA capacity is sufficient for the operation of the trainer.

The details of the power supply connector is presented in the chapter 6.

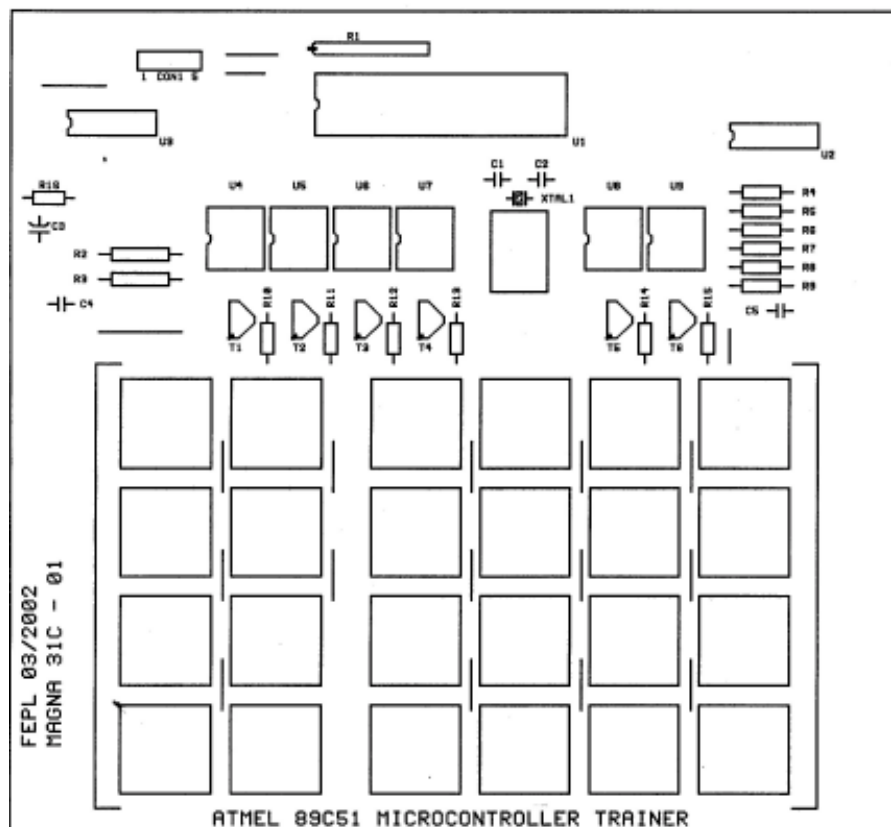
---

## Trainer PCB Layouts





Main Board Layout



Keyboard Controller Layout

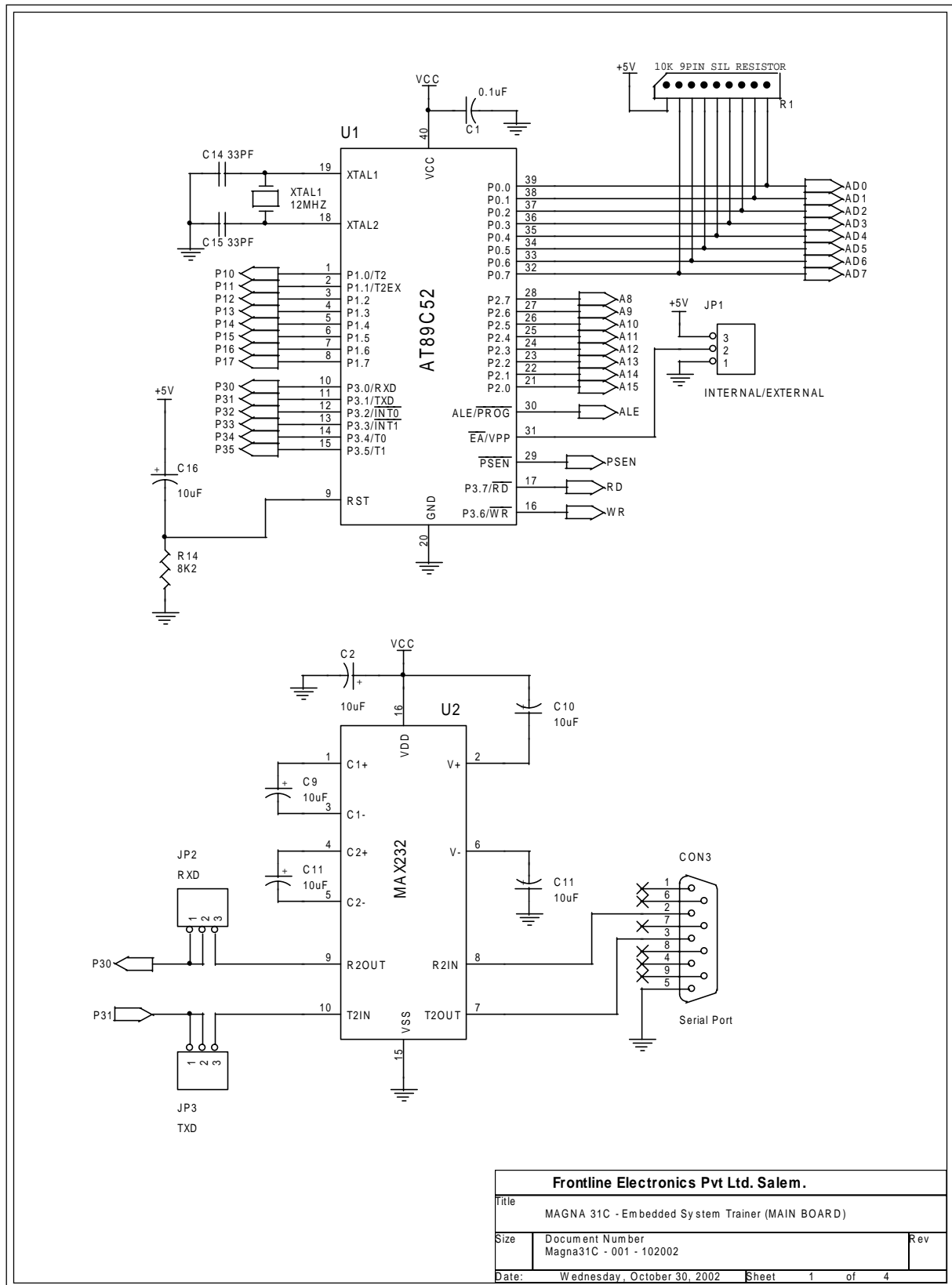


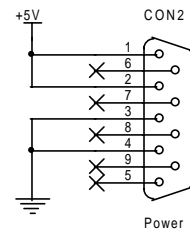
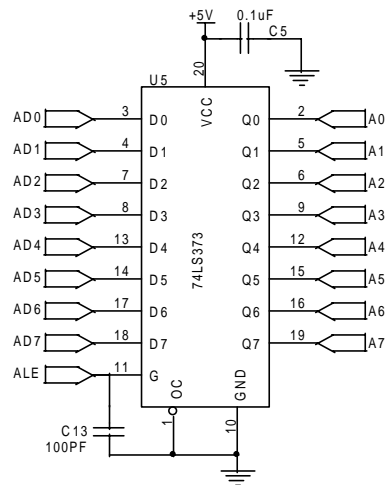
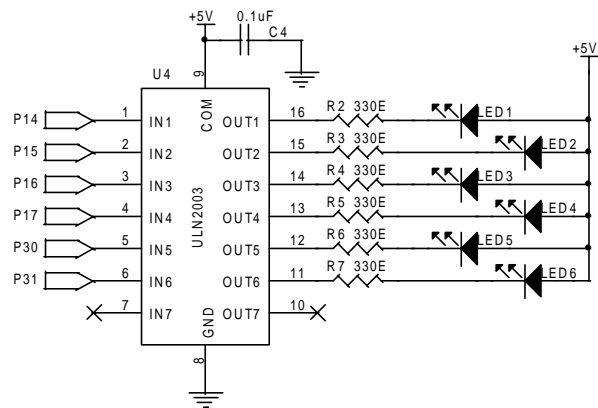
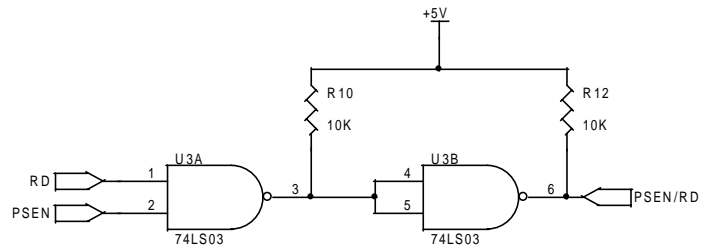


---

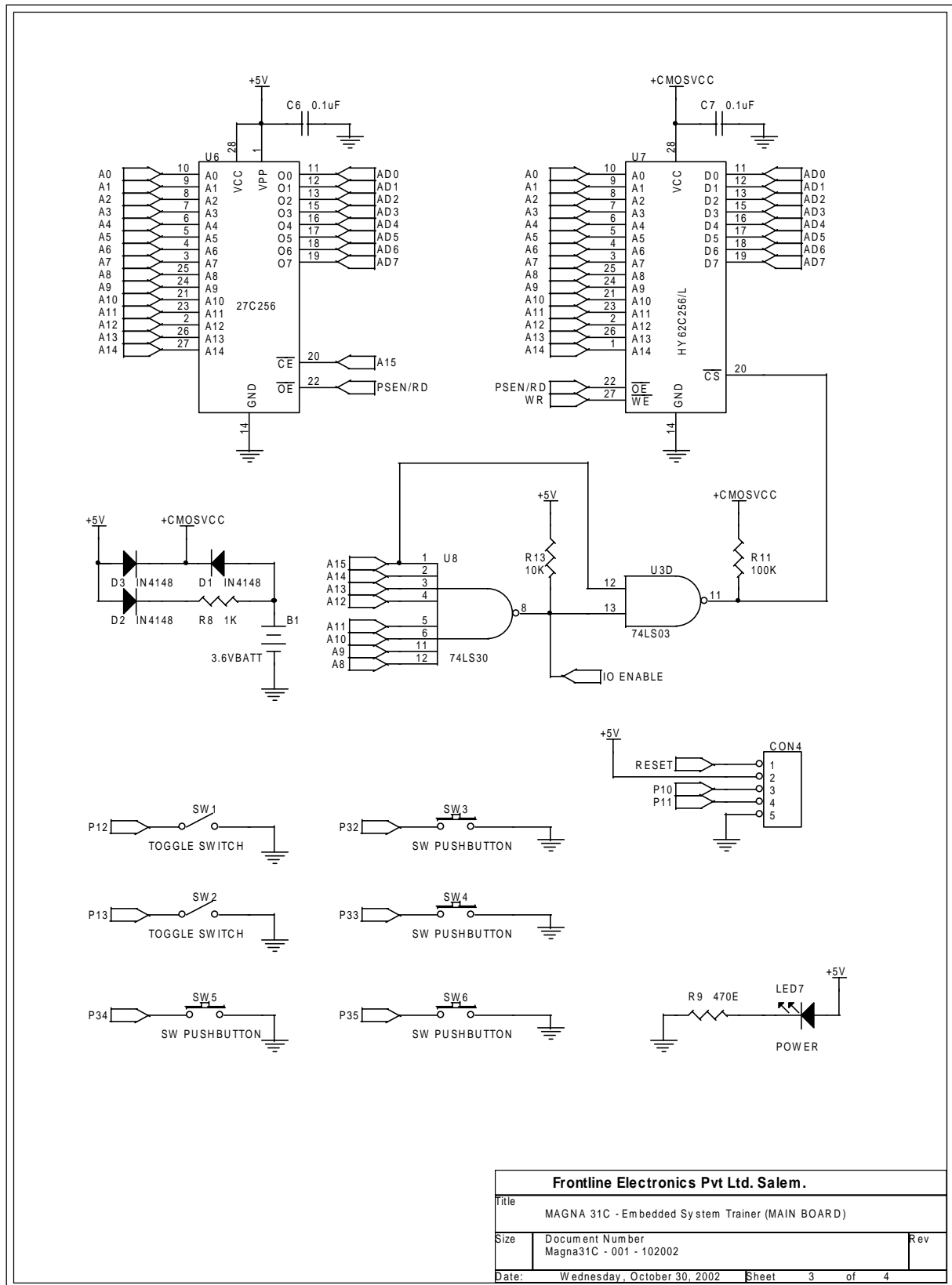
## Schematic Diagram of Topview Trainer

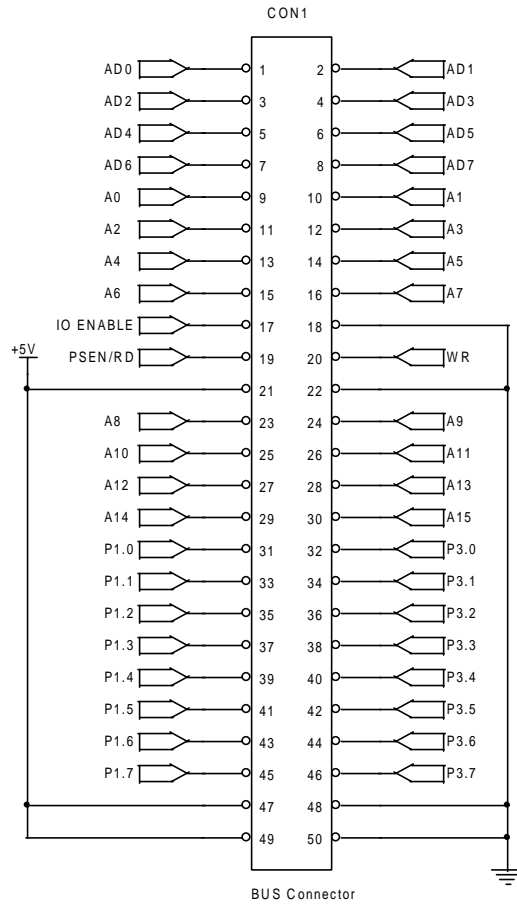




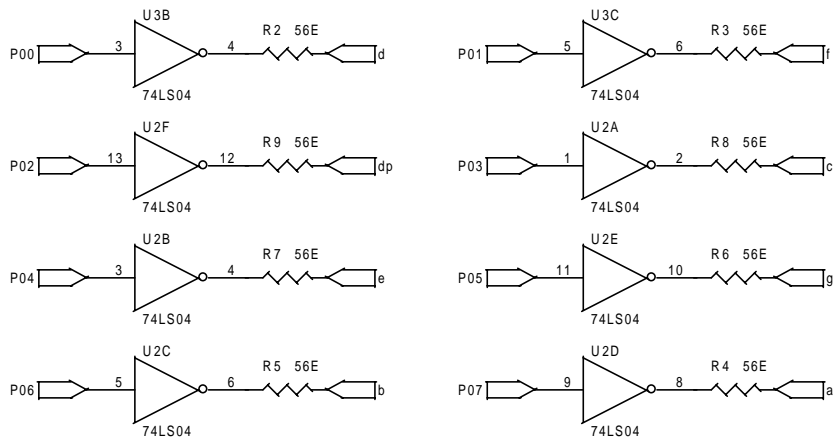
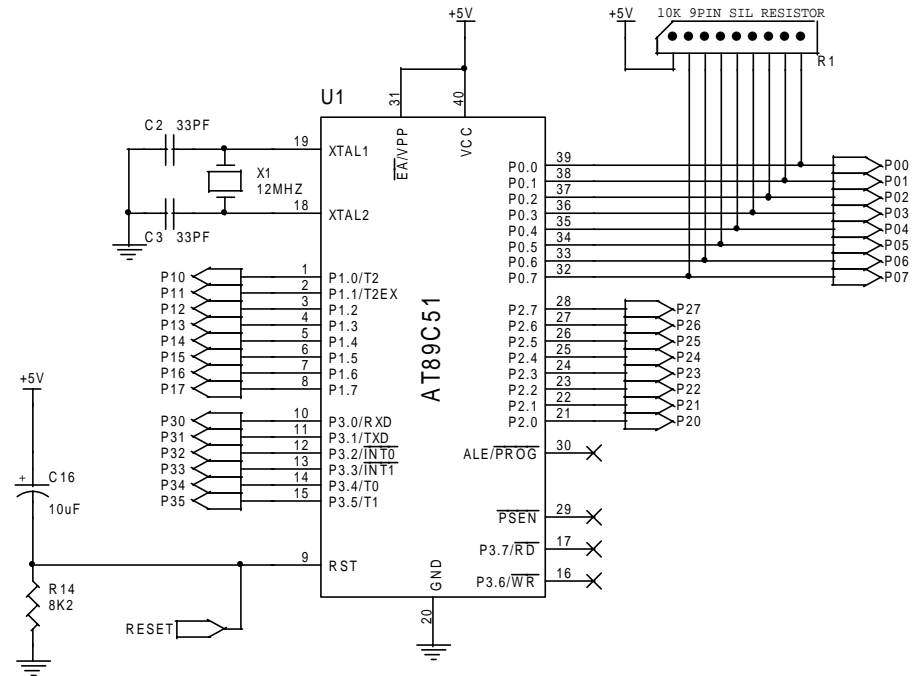


Frontline Electronics Pvt Ltd. Salem.		
Title	MAGNA 31C - Embedded System Trainer (MAIN BOARD)	
Size	Document Number Magna31C - 001 - 102002	Rev
Date:	Wednesday, October 30, 2002	Sheet 2 of 4

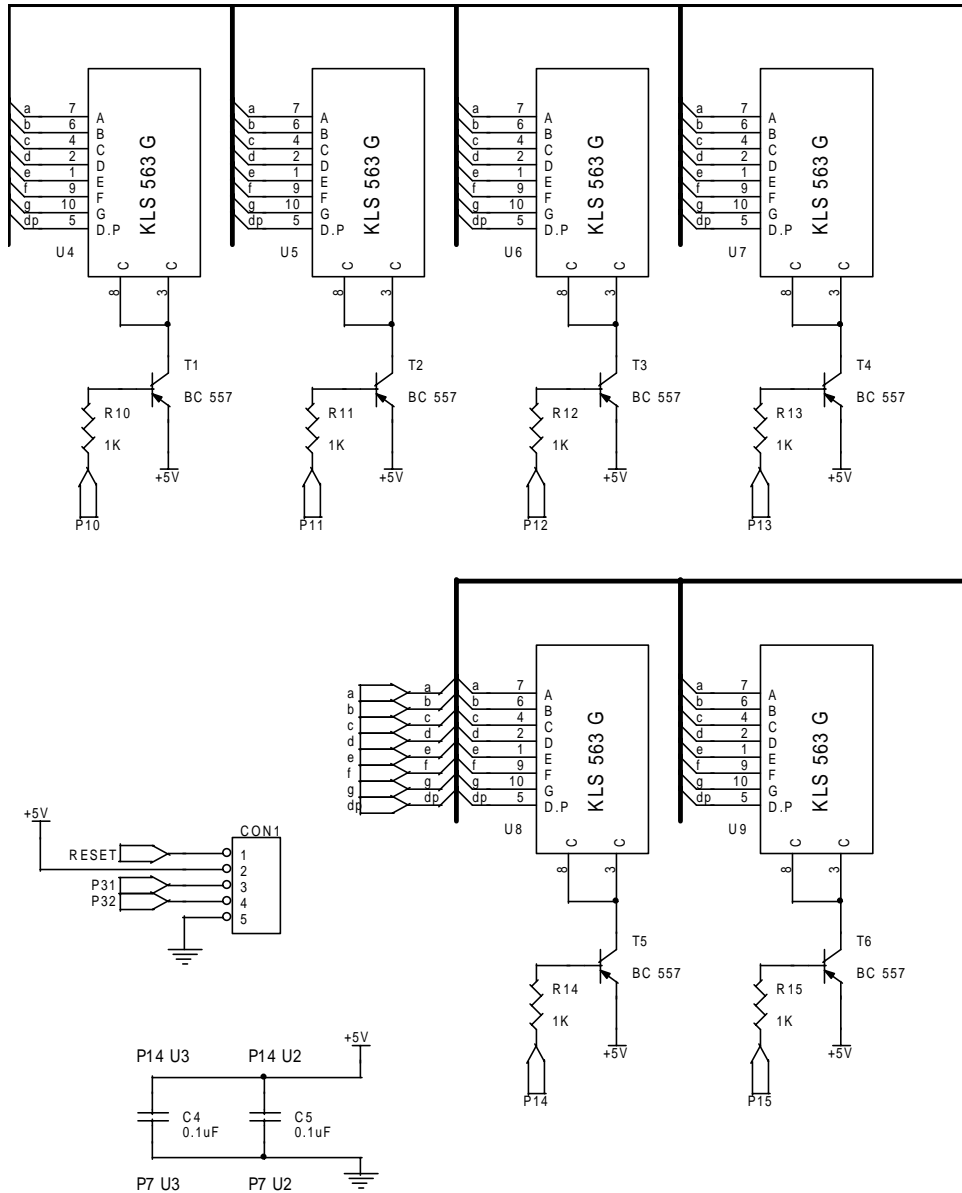




Frontline Electronics Pvt Ltd. Salem.			
Title		MAGNA 31C - Embedded System Trainer (MAIN BOARD)	
Size	Document Number		Rev
	Magna31C - 001 - 102002		
Date:	Wednesday, October 30, 2002	Sheet	4 of 4

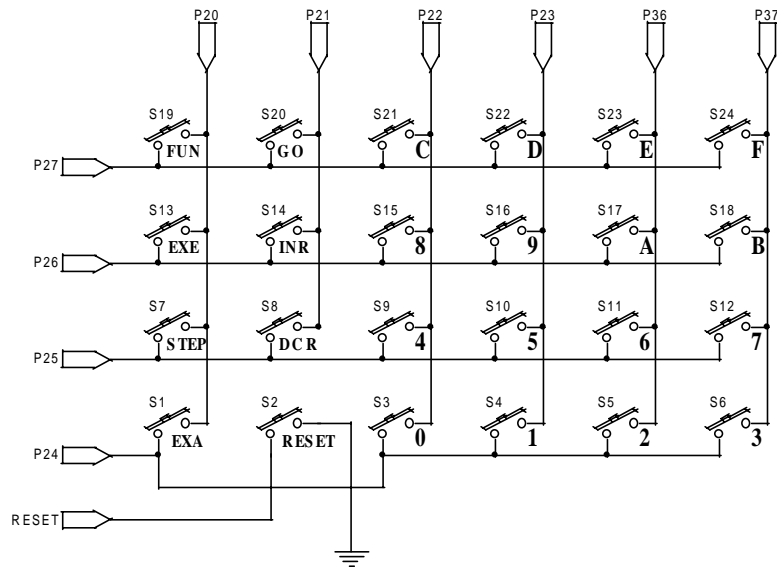


Frontline Electronics Pvt Ltd. Salem.		
Title		
MAGNA 31C - Embedded System Trainer (KEYBOARD CONTROLLER)		
Size	Document Number	Rev
	Magna31C - 003 - 102002	
Date: Wednesday, October 30, 2002 Sheet 1 of 3		



Frontline Electronics Pvt Ltd. Salem.		
Title	MAGNA 31C - Embedded System Trainer (KEYBOARD CONTROLLER)	
Size	Document Number Magna31C - 003 - 102002	Rev
Date:	Wednesday, October 30, 2002	Sheet 2 of 3





Frontline Electronics Pvt Ltd. Salem.		
Title	MAGNA 31C - Embedded System Trainer (KEYBOARD CONTROLLER)	
Size	Document Number Magna31C - 003 - 102002	Rev
Date:	Wednesday, October 30, 2002	Sheet 3 of 3



## 4.1 Introduction

This part gives information on using the Topview Trainer in stand-alone configuration which uses onboard keyboard and seven segment displays for user interaction.

This chapter explains the various functions available with the trainer. The monitor program resides in the flash area of the controller. The monitor program controls all the operations of the trainer. It is designed to respond to the user through the keypad.

The simple layout of the keyboard is shown below:

<b>EXAM INT</b> key 21	<b>GO</b> key 17	<b>PCH C</b> key 13	<b>PCL D</b> key 14	<b>DPH E</b> key 15	<b>DPL F</b> key 16
<b>STEP</b> key 22	<b>NEXT</b> key 18	<b>SP 8</b> key 9	<b>PSW 9</b> key 10	<b>ACC A</b> key 11	<b>B B</b> key 12
<b>EXE</b> key 23	<b>BACK</b> key 19	<b>TCON 4</b> FILL Internal key 5	<b>TMOD 5</b> Copy Int to Ext key 6	<b>SCON 6</b> Copy Ext to Int key 7	<b>IE 7</b> Load Examples key 8
<b>EXAM EXT</b> key 24	<b>RST</b> key 20	<b>Bank0 0</b> Examine Register key 1	<b>Bank1 1</b> Copy External key 2	<b>Bank2 2</b> Fill External key 3	<b>Bank3 3</b> Copy Internal key 4

The descriptions of the keys are:

Key 1 - Hex key value 0 / Examine Register Function / Bank 0

Key 2 - Hex key value 1 / Copy External Function / Bank 1

Key 3 - Hex key value 2 / Fill External Memory Function / Bank 2

Key 4 - Hex key value 3 / Copy Internal Memory Function / Bank 3

Key 5 - Hex key value 4 / Fill Internal Memory Function / TCON

### Introduction

- Key 6 - Hex key value 5 / Copy Internal to External Memory Function / TMOD
- Key 7 - Hex key value 6 / Copy External to Internal Memory Function / SCON
- Key 8 - Hex key value 7 / IE / Load Examples
- Key 9 - Hex key value 8 / SP
- Key 10 - Hex key value 9 / PSW
- Key 11 - Hex key value 0A / ACC
- Key 12 - Hex key value 0B / B
- Key 13 - Hex key value 0C / PCH
- Key 14 - Hex key value 0D / PCL
- Key 15 - Hex key value 0E / DPH
- Key 16 - Hex key value 0F / DPL
- Key 17 - Go Function (Execute program from given address)
- Key 18 - Next key
- Key 19 - Back key
- Key 20 - Reset key
- Key 21 - Examine Internal Memory Function
- Key 22 - Single Step Function
- Key 23 - Execute Function
- Key 24 - Examine External Memory Function

### Introduction

Some of the keys in the keyboard will have multiple functions and all the functions are represented clearly on the top surface of each key in the trainer for your convenience.

#### Condition 1:

When the sign on message/command prompt appears on the display, the hex keys will act as function keys.

For example, if you press **0** key under reset condition, the examine register function will become active and the control enters into that function.

The keys and their functions are listed below.

- <0> - Examine Register Function
- <1> - Copy External Function
- <2> - Fill External Memory Function
- <3> - Copy Internal Memory Function
- <4> - Fill Internal Memory Function
- <5> - Copy Internal to External Memory Function
- <6> - Copy External to Internal Memory Function
- <7> - Load Examples

#### Condition 2:

When the monitor waits for data or address entry, the hex key pad will act as number keys and the value from 0-F can be keyed in.

#### Condition 3:

The third option will come into action when the monitor is in the examine register function. After entering examine register function, the hex key pad will act as register selection keys and any register can be selected for viewing/editing.

### Introduction

The keys and their functions are listed below.

<0>	-	Bank 0
<1>	-	Bank 1
<2>	-	Bank 2
<3>	-	Bank 3
<4>	-	TCON
<5>	-	TMOD
<6>	-	SCON
<7>	-	IE
<8>	-	SP
<9>	-	PSW
<A>	-	ACC
<B>	-	B
<C>	-	PCH
<D>	-	PCL
<E>	-	DPH
<F>	-	DPL

During Power on reset the monitor initializes the complete system and displays a sign on message '- - 80 31' on the seven segment displays. Further commands can be entered from the keyboard when the sign on message appears.

During any operation, when **EXE** key is pressed a prompt '-' appears on the display. This symbol also indicates that the trainer is ready to accept any command.

Hence under the reset condition of the trainer (during sign on message display) and also when a '-' prompt appears on the display, the monitor waits for the user's response.

The address is displayed in the first 4 digits of the display and the data is displayed in the next 2 digits of the display as shown below.

## Introduction

Digits

<u>1 2 3 4</u>	<u>5 6</u>
Address	Data
field	field

Various operations can be carried out using the available function keys. Hence care is taken here to explain the operations that can be carried out with this trainer. The flow of the operation and the key strokes to be used are given in detail in step by step format for every operation separately. Various possible operations are listed below:

1. Enter/view a program.
2. Execute a program.
3. Examine Register function.
4. Single Step Execution and debugging.
5. Examine Internal Data Memory function.
6. Data Fill function (External Memory).
7. Block Copy function (External Memory).
8. Data Fill function (Internal Memory).
9. Block Copy function (Internal Memory).
10. Block Copy function (From Internal Memory to External Memory).
11. Block Copy function (From External Memory to Internal Memory).
12. Load Examples function.

**Note:**

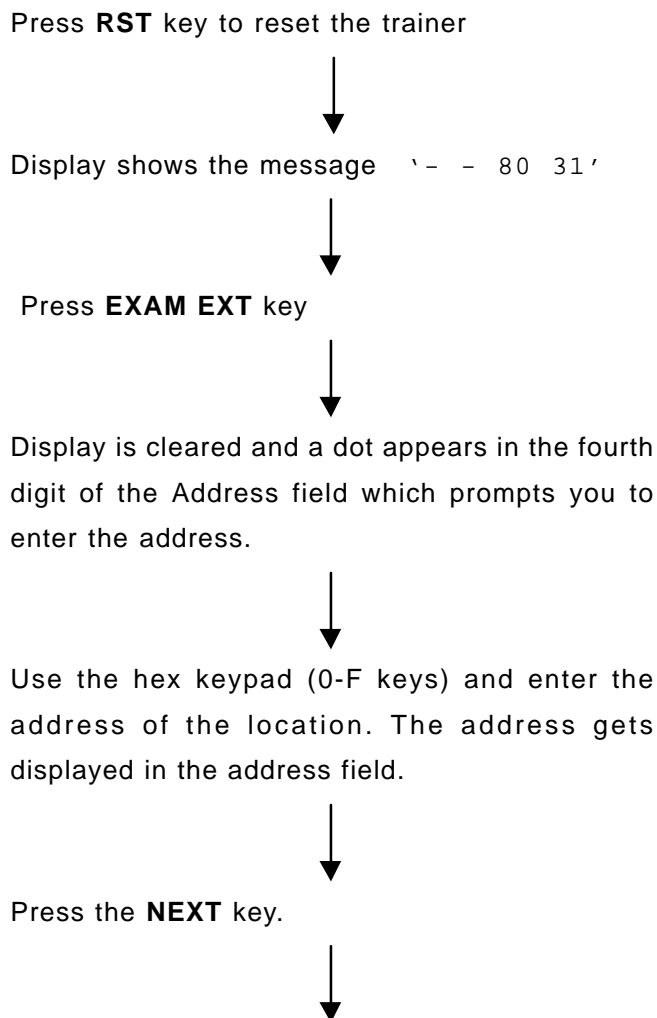
When the dot point(.) appears in the fourth digit of the display, it indicates that the monitor waits for the entry of the address. Similarly when the dot point (.) appears in the sixth digit of the display, the monitor prompts for the entry of data.

### 4.2 Enter / View a Program

The first step in using the trainer is to enter a program into the RAM area of the trainer. This can be carried out using the keys, **EXAM EXT**, **NEXT**, **BACK** keys.

The available RAM area is 8000H to FCFFH. Use this area to store both program and data.

The flow of the program entry is shown below.





### Enter / View a Program

The content of that memory location is displayed in the data field and a dot appears in the right edge of the data field.



Enter a new data and press **NEXT** key. Only then the data will be stored in the specified address of the RAM. The next address then appears in the address field of the display.



Press the **BACK** key to view/edit the previous Location. After editing press **NEXT** to store the contents



Repeat the above steps to enter program line by line in the required locations.



Press **EXE** key to terminate this command. A command prompt '-' appears on the display.

## Enter / View a Program

## Example:

KEY PRESSED	DISPLAY		COMMENT
	ADDRESS	DATA	
	FIELD	FIELD	
RST	- - 8 0	3 1	;System reset.
EXAM			
EXT	.		;Examine memory command.
8	0 0 0 8.		;First memory location
0	0 0 8 0.		;to be examined is 8000H.
0	0 8 0 0.		
0	8 0 0 0.		
NEXT	8 0 0 0	X X.	;Content of this location.
NEXT	8 0 0 1	X X.	;Content of 8001H.
1	8 0 0 1	0 1.	;New data-17H to be entered
7	8 0 0 1	1 7.	
NEXT	8 0 0 2	X X.	;Data is entered.
EXE	-		;Command termination
			;prompt

### 4.3 Execute a Program

To execute a program stored in the RAM in full speed, use 'GO' command followed by the starting address.

Once the execution command is activated, the monitor transfers control to the user program and the execution starts from the user specified address. To stop execution at any time, reset the trainer with the help of the reset key. Then the control will be transferred again to the monitor program.

The following sequence will give an idea of how to execute a program.

Check for the reset condition of the trainer. If not press the reset key



Display shows the message ' - - 80 31 '



Press **GO** key



Already stored execution address will be displayed in the address field, a dot point appears in the fourth digit of the address field which prompts you to enter a new address.



Use the hex keypad (0-F keys) and enter the starting address of the program. The address gets displayed in the address field.



Press the **EXE** key.

## Execute a Program



The display field gets cleared and a character 'E' will be displayed in the first digit of the display. This indicates that the control is transferred to the user program.



In order to break execution press the **RST** key. This resets the trainer and transfers control to the monitor program.

The following example gives an actual picture of the execution process.

**Example:**

Consider a simple example program.

ADDR	OBJECT	CODE	MNEMONIC	COMMENT
8100	B2 94		CPL P1.4	;Complement the Level at ;portline P1.4
8102	12 F6 60		LCALL F660	;Delay for one second ;(approx)
8105	80 F9		SJMP 8100	;Repeat

## Execute a Program

To run above mentioned program,

KEY PRESSED	DISPLAY		COMMENT
	ADDRESS	DATA	
	FIELD	FIELD	
RST	- - 8 0	3 1	;System reset.
GO	X X X X.	X X	;Go command.
8	0 0 0 8.		;Enter the address of the ;program.
1	0 0 8 1.		
0	0 8 0 0.		
0	8 0 0 0.		
EXE	E		;Program is under execution

### 4.4 Examine Register

The general purpose registers (register banks) and the special function registers of the 8031 microcontroller can be viewed/edited by using this function.

This function also helps to check the register contents during single step operation.

Check for the reset condition of the trainer. If not press the reset key



Display shows the message ' - - 80 31 '



Press **Examine Register** key



Display is cleared and a dot appears in the address field. Press the **NEXT** key to view the first register (ACC) contents. Otherwise, to view a particular register (SFR), press the respective key. Refer to keyboard layout which shows the keys used for viewing/editing registers.



Press the **NEXT** key to view the next register.



Press the **BACK** key to view the previous register contents



### Examine Register

Use the hex keypad (0-F keys) to change the register contents and press **NEXT** key to update that register with new data.



In order to terminate the function press the **EXE** key. The display gives a '-' prompt and now the other commands can be executed.

#### Note:

To view the registers of respective bank, press the respective bank number key when a dot point appears in the display.

The bank number will be displayed in the first digit. To select a particular register in that bank, press any one of the keys from 0 to 7. The contents of the selected register will be displayed. Then use **NEXT** or **BACK** key to view the next/previous registers.

The following table defines the processor's register's name, the display abbreviation and the sequence in which the registers are examined.

Display	Name of the register
1. ACC	ACCUMULATOR.
2. b	B Register
3. dph	Data pointer register higher byte.
4. dpl	Data pointer Register lower byte.
5. pch	Program counter - higher byte.
6. pcl	Program counter - lower byte.
7. psw	Program status word.
8. sp	Stack pointer.
9. lp	Interrupt priority control.
10. IE	Interrupt enable control.

### Examine Register

11. tmod	Timer / Counter mode control.
12. tcon	Timer / Counter control.
13. 2mod	Timer / Counter 2 mode control register (T2MOD) .
14. 2con	Timer / Counter 2 mode control register (T2CON) .
15. tH0	Timer / Counter 0 higher byte.
16. tL0	Timer / Counter 0 lower byte.
17. tH1	Timer / Counter 1 higher byte.
18. tL1	Timer / Counter 1 lower byte.
19. tH2	Timer / Counter 2 higher byte.
20. tL2	Timer / Counter 2 lower byte.
21. CAPH	Timer 2 Capture Register higher byte (RCAP2H).
22. CAPL	Timer 2 Capture Register Lower byte (RCAP2L).
23. Scon	Serial control.
24. Sbuf	Serial data buffer.
25. pcon	Power control.
26. 0 r0	Register Bank 0. Register 0.
27. 0 r1	Register Bank 0. Register 1.
28. 0 r2	Register Bank 0. Register 2.
29. 0 r3	Register Bank 0. Register 3.
30. 0 r4	Register Bank 0. Register 4.
31. 0 r5	Register Bank 0. Register 5.
32. 0 r6	Register Bank 0. Register 6.
33. 0 r7	Register Bank 0. Register 7.
34. 1 r0	Register Bank 1. Register 0.
35. 1 r1	Register Bank 1. Register 1.
36. 1 r2	Register Bank 1. Register 2.
37. 1 r3	Register Bank 1. Register 3.
38. 1 r4	Register Bank 1. Register 4.
39. 1 r5	Register Bank 1. Register 5.
40. 1 r6	Register Bank 1. Register 6.
41. 1 r7	Register Bank 1. Register 7.
42. 2 r0	Register Bank 2. Register 0.
43. 2 r1	Register Bank 2. Register 1.
44. 2 r2	Register Bank 2. Register 2.



### Examine Register

45. 2 r3	Register Bank 2. Register 3.
46. 2 r4	Register Bank 2. Register 4.
47. 2 r5	Register Bank 2. Register 5.
48. 2 r6	Register Bank 2. Register 6.
49. 2 r7	Register Bank 2. Register 7.
50. 3 r0	Register Bank 3. Register 0.
51. 3 r1	Register Bank 3. Register 1.
52. 3 r2	Register Bank 3. Register 2.
53. 3 r3	Register Bank 3. Register 3.
54. 3 r4	Register Bank 3. Register 4.
55. 3 r5	Register Bank 3. Register 5.
56. 3 r6	Register Bank 3. Register 6.
57. 3 r7	Register Bank 3. Register 7.

To terminate this command, press the key **EXE** at any point of the above mentioned operation.

#### Note:

At any time, if you reset the trainer, the special function registers (SFR's) and R0 of Bank 0 in internal RAM will get affected. Consider the following conditions:

#### Case 1:

To debug your program, if you want to set some values in the SFR before execution and these values should be taken into consideration during execution, follow the steps shown below:

1. Set the SFR's using **Examine Register** function and quit the function by pressing **EXE** key. You should not reset the trainer to come out of the function, because resetting will change the SFR contents.
2. Use **GO** function to execute the program when the '-' prompt appears on the display. During execution the set register values will be taken into consideration.

## Examine Register

## Case 2:

If you try to store the result of any operation in SFRs during execution and verify it after completing the execution. The only way to come out of execution is to reset the trainer. Resetting will change the SFR contents. Hence store the result in external RAM/internal RAM for viewing or use single step execution, which facilitates viewing register contents after executing each instruction.

## Examples:

Examine and modify register.

KEY PRESSED	DISPLAY		COMMENT
	ADDRESS FIELD	DATA FIELD	
RST	- - 8 0	3 1	;System Reset.
Examine Register	.		;Examine register command
NEXT	A c c	X X.	;contents of 1st register
NEXT	b	X X.	;Contents of 2nd register
1	b	0 1.	;modify the contents
5	b	1 5.	;of b
NEXT	d p H	X X.	
EXE	-		;Command termination prompt

## Examine Register

Examine and modify register bank register

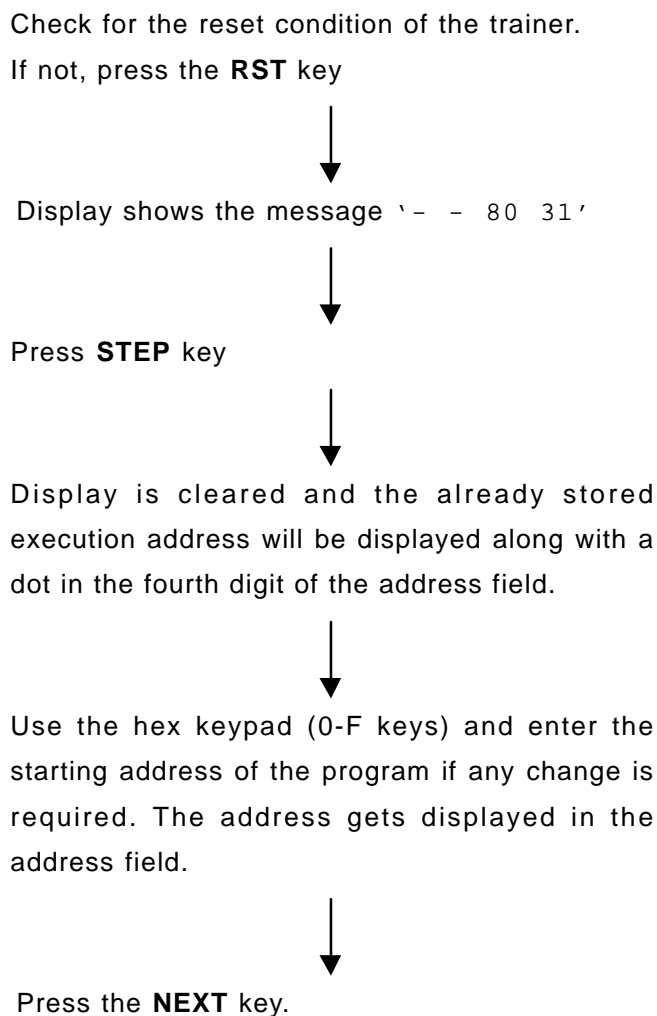
KEY PRESSED	DISPLAY		COMMENT
	ADDRESS FIELD	DATA FIELD	
RST	- - 8 0	3 1	;System Reset.
Examine Register	.		;Examine register command
Bank0	0		;Select bank 0
3	0 r 3	X X.	;select register r3
1	0 r 3	0 1.	;modify the contents
6	0 r 3	1 6.	;Of r3 of bank 0
NEXT	0 r 4	X X.	;contents of next register
EXE	-		;Command termination prompt

#### 4.5 Single Step Execution and Debugging

Single step execution as the name implies is used to execute the total program instruction by instruction.

This feature is very useful to debug a program. After executing an instruction in your program, the control is returned to the monitor program if necessary and this permits using other functions like **Examine Register**, **EXAM INT**, **EXAM EXT** to verify the result of executing an instruction.

The step by step execution process along with using other functions is explained below.



### Single Step Execution and Debugging

↓

The first instruction gets executed and the next instruction address gets displayed in the display.

↓

At this juncture, the result of executing the first instruction can be verified. Press **EXE** key to halt the single step execution temporarily.

↓

A prompt '-' appears on the display. Now use any other functions like Examine Internal Memory or Examine Register to view the data in the memory and the registers respectively.

↓

After viewing either register/memory contents press **EXE** key. Once again the '-' prompt appears on the display.

↓

Press **STEP** key to enter again into single step execution. The current instruction to be executed will be displayed

↓

Press **NEXT** key for executing one more instruction and proceed with single step further.

↓

Press **EXE** key or Reset the trainer to terminate the command.

#### 4.6 Examine/Modify Internal Data Memory

This function as the name implies helps to examine or modify the contents of the internal data memory.

Check for '-' prompt or the sign on message.



Press **EXAM INT** key



Display is cleared and a dot appears in the fourth digit of the Address field which prompts for the starting address of the internal memory.



Use the hex keypad (0-F keys) and enter the required starting address or just press the **NEXT** key.



When **NEXT** key is pressed without entering the starting address, the RAM address starts at 00H, otherwise at the user specified address.



The relevant data will be displayed along with the address. Modify the data using hex keypad (0 - F) keys if necessary.



Press **NEXT** key after any modification in the data, otherwise the change will not be stored in the RAM.

## Examine/Modify Internal Data Memory



Press **NEXT** key to view the next location. Similarly press **BACK** key to view the previous location contents.



After viewing memory contents press **EXE** key. The command prompt '-' appears on the display.

## Example.

KEY PRESSED	DISPLAY		COMMENT
	ADDRESS FIELD	DATA FIELD	
RST	- - 8 0	3 1	;System Reset.
EXAM			
INT	.		;Examine Internal memory
NEXT	0 0	X X.	
NEXT	0 1	X X.	
2	0 1	0 2.	
5	0 1	2 5.	;Enter the data 25.
NEXT	0 2	X X.	
EXE	-		;Command termination.

#### 4.7 Data Fill ( External Memory )

This function is useful when a block of external memory is required to be filled with the same data.

Reset the trainer by pressing **RST** key



Display shows the message ' - - 80 31 '



Press **Fill External** key



A message 'SA' appears on the data field along with the already stored starting address in the address field.



Enter the starting address and press the **NEXT** key, if necessary



The monitor will then prompt for the ending address entry with a message 'EA' in the data field. Already stored ending address will be displayed in the address field.



If necessary alter the address and press the **NEXT** key. The next entry will be the data, which is indicated by a message 'dAtA'.





**Data Fill ( External Memory )**

Enter the new data to be used for filling the block  
and press **EXE** key.



The monitor completes the filling operation and  
displays the Command prompt.

**Note:**

Error occurs under the following conditions.

1. When the ending address is less than the starting address of the block
2. When the block or part of the block is nonexistent.

**Example:**

KEY PRESSED	DISPLAY		COMMENT
	ADDRESS FIELD	DATA FIELD	
RST	- - 8 0	3 1	;System Reset.
Fill			
External	X X X X.	S A	;Fill External memory
8	0 0 0 8.	S A	
0	0 0 8 0.	S A	
0	0 8 0 0.	S A	;Starting address of the
0	8 0 0 0.	S A	;block 8000H
NEXT	X X X X.	E A	
8	0 0 0 8.	E A	
0	0 0 8 0.	E A	
5	0 8 0 5.	E A	;Ending address of the
0	8 0 5 0.	E A	;block 8050H
NEXT	d A t A	X X.	;filling data
1	d A t A	0 1.	
EXE	-		;Command prompt appears ;after completing fill ;operation

#### 4.8 Block Copy ( External Memory )

This function is useful when a block of data is copied from one area of external memory to another area in the external memory.

Reset the trainer by pressing **RST** key



Display shows the message ' - - 80 31 '



Press **Copy External** key



A message 'SA' appears in the data field along with the already stored starting address in the address field.



Enter the starting address and press the **NEXT** key



The monitor will then prompt for the ending address entry with a message 'EA' in the data field. Already stored ending address will be displayed in the address field.



If necessary alter the address and press the **NEXT** key. The next entry will be the destination address, mentioned in the display as 'dA'.



### Block Copy ( External Memory )

Enter the destination address and press **EXE** key.



The monitor completes the block copy operation and displays the Command prompt.

#### Note:

Error occurs under the following conditions.

1. When the ending address is less than the starting address of the block
2. If any one of the blocks are out of range (8000H to FEFFH)

#### Example:

KEY PRESSED	DISPLAY		COMMENT
	ADDRESS FIELD	DATA FIELD	
RST	- - 8 0	3 1	;System Reset.
Copy			
External	X X X X.	S A	;Copy command.
8	0 0 0 8.	S A	
0	0 0 8 0.	S A	
0	0 8 0 0.	S A	;Starting address of the
0	8 0 0 0.	S A	;block 8000H
NEXT	X X X X.	E A	
8	0 0 0 8.	E A	
1	0 0 8 1.	E A	
0	0 8 1 0.	E A	;Ending address of the
0	8 1 0 0.	E A	;block 8100H
NEXT	X X X X.	d A	
9	0 0 0 9.	d A	
0	0 0 9 0.	d A	
0	0 9 0 0.	d A	;Destination address
0	9 0 0 0.	d A	
EXE	-		;Command prompt appears.

#### 4.9 Data Fill ( Internal Memory )

This function is used to fill a block of memory with the same data.

Reset the trainer by pressing **RST** key



Display shows the message ' - - 80 31 '



Press **Fill Internal** key



A message 'SA' appears in the data field along with the already stored starting address of the internal memory in the address field.



Enter the starting address and press the **NEXT** key, if a change is required



The monitor will then prompt for the ending address entry with a message 'EA' in the data field. Already stored ending address will be displayed in the address field.



If necessary alter the address and press the **NEXT** key. The next entry will be the data, which is indicated by a message 'dAtA' and the already stored data will be displayed.



**Data Fill ( Internal Memory )**

Enter the new data to be used for filling the block  
and press **EXE** key.



The monitor completes the filling operation of the  
internal memory and displays the Command  
prompt.

**Note:**

Error occurs under the following conditions.

1. When the ending address is less than the starting address of the block
2. When the ending address is greater than FFH.

**Example:**

KEY PRESSED	DISPLAY		COMMENT
	ADDRESS FIELD	DATA FIELD	
RST	- - 8 0	3 1	;System Reset.
Fill			
Internal	X X.	S A	;Fill Internal memory
5	0 5.	S A	;Starting address of
0	5 0.	S A	;the block
NEXT	X X.	E A	;Ending address of
5	0 5.	E A	;the block
F	5 F.	E A	
NEXT	d A t A	X X.	;fill data
1	d A t A	0 1.	
EXE	-		;Command prompt appears

#### 4.10 Block Copy ( Internal Memory )

Use this function to copy block of data from one area of internal memory to another area in the internal memory.

Reset the trainer by pressing **RST** key



Display shows the message ' - - 80 31 '



Press **Copy Internal** key



A message 'SA' appears in the data field along with the already stored starting address in the address field.



Enter the starting address and press the **NEXT** key, if necessary



The monitor will then prompt for the ending address entry with a message 'EA' in the data field. Already stored ending address will be displayed in the address field.



The next entry will be the destination address, mentioned in the display as 'dA'.



### Block Copy ( Internal Memory )

Enter the destination address and press **EXE** key.



The monitor completes the block move operation and displays the command prompt.

#### Note:

Error occurs under the following conditions.

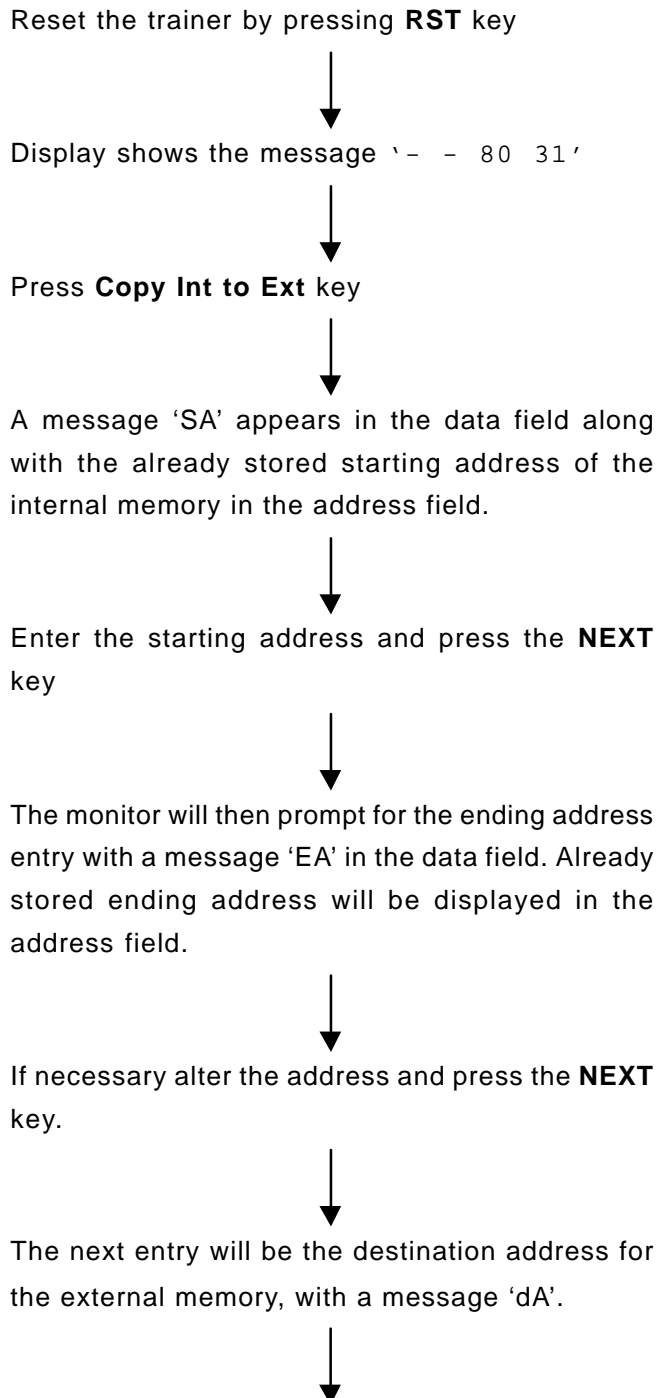
1. When the ending address is less than the starting address of the block
2. When any one of the addresses (starting, ending, destination) is greater than FFH.

#### Example:

KEY PRESSED	DISPLAY		COMMENT
	ADDRESS FIELD	DATA FIELD	
RST	- - 8 0	3 1	;System Reset.
Copy			
Internal	X X.	S A	;Copy Internal memory.
4	0 4.	S A	;Starting address of
0	4 0.	S A	;the block
NEXT	X X.	E A	;Ending address of
4	0 4.	E A	;the block
F	4 F.	E A	
NEXT	X X.	d A	;Destination address
5	0 5.	d A	
0	5 0.	d A	
EXE	-		;Command prompt appears.

#### 4.11 Block Copy ( From Internal Memory to External Memory )

This function helps to copy a block of data from internal memory to another block in external memory.





**Block Copy ( From Internal Memory to External Memory )**

Enter the destination address and press **EXE** key.



The monitor completes the block copy operation from the internal memory to the External memory and displays the Command prompt.

**Note:**

Error occurs under the following conditions.

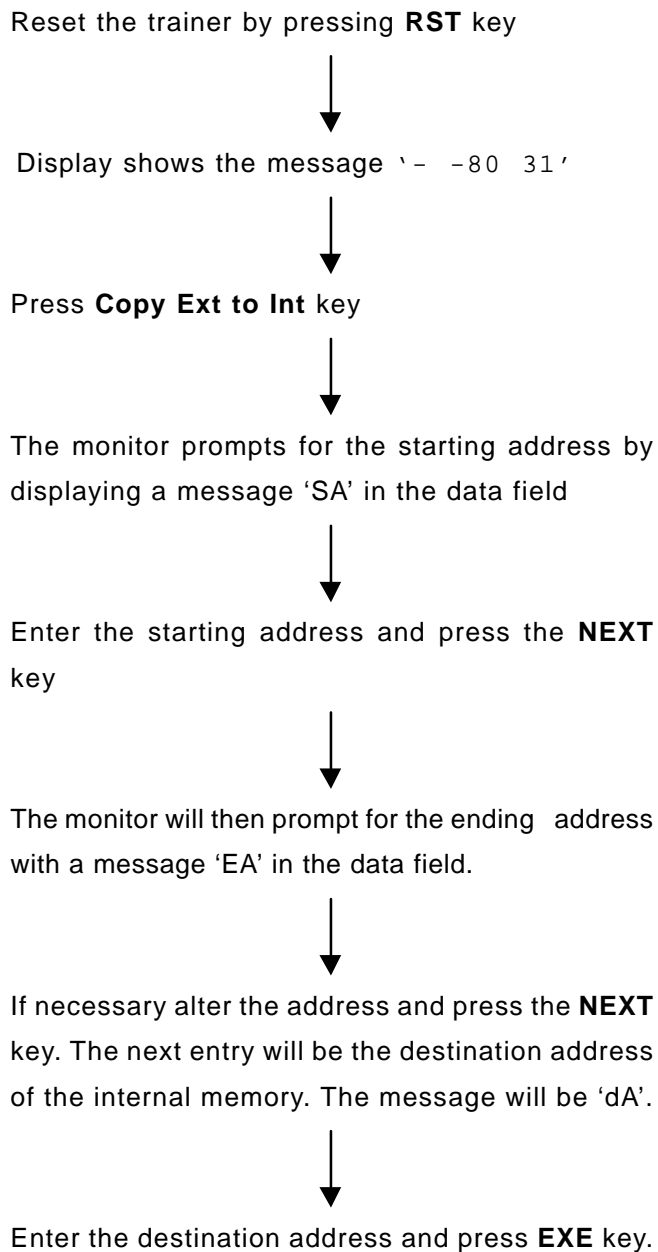
1. When the ending address is less than the starting address of the block
2. When the starting address and ending address is greater than FFH.
3. When destination block appears in the nonexistent RAM area.

**Example:**

KEY PRESSED	DISPLAY		COMMENT
	ADDRESS FIELD	DATA FIELD	
RST	- - 8 0	3 1	;System Reset.
Copy			
Int to Ext	X X.	S A	;Copy from internal mem ;to external mem.
5	0 5.	S A	;Starting address of the
0	5 0.	S A	;block internal memory
NEXT	X X.	E A	;Ending address of
5	0 5.	E A	;the block
F	5 F.	E A	
NEXT	X X X X.	d A	;destination address of
9	0 0 0 9.	d A	;external memory
0	0 0 9 0.	d A	
0	0 9 0 0.	d A	
0	9 0 0 0.	d A	
EXE	-		;Command prompt appears.

#### 4.12 Block Copy ( From External Memory to Internal Memory )

Use this function to copy a block of data from external memory to another block in internal memory.



**Block Copy ( From External Memory to Internal Memory )**

The monitor completes the block copy operation and displays the command prompt.

**Note:**

Error occurs under the following conditions.

1. When the ending address is less than the starting address of the block
2. When the destination address is greater than FFH.
3. When the source block appears in the nonexistent RAM area.

**Example:**

KEY PRESSED	DISPLAY		COMMENT
	ADDRESS FIELD	DATA FIELD	
RST	- - 8 0	3 1	;System Reset.
Copy			
Ext to Int	X X X X.	S A	;Copy from external mem ;to internal mem.
8	0 0 0 8.	S A	
0	0 0 8 0.	S A	
0	0 8 0 0.	S A	;Starting address of
0	8 0 0 0.	S A	;the block 8000H
NEXT	X X X X.	E A	
8	0 0 0 8.	E A	
0	0 0 8 0.	E A	
0	0 8 0 0.	E A	;Ending address of
F	8 0 0 F.	E A	;the block 800FH
NEXT	X X.	d A	
5	0 5.	d A	
0	5 0.	d A	
EXE	-		;Command prompt appears.

### 4.13 Load Examples

All the example programs and other useful routines supplied in the examples part of the manual are stored in the flash memory of the Microcontroller along with the monitor program.

Use this function to load the example programs and routines from the flash to the RAM locations (F600H to FCFFH).

This function saves your time and avoids the tedious process of entering the opcodes for the example programs and the relevant routines.

Reset the trainer by pressing **RST** key.



Display shows the message “- - 8031 -”



Press **Load Examples** key



Monitor displays a message ‘ LoAd E ’ and prompts the user for a confirmation.



Press **EXE** to confirm the loading operation.

The monitor copies the program from the flash area to the RAM locations F600H to FCFFH and displays the command prompt.

You can execute the programs using **Go** command. Refer to the Software Examples Manual for more details on example programs and routines.

### Load Examples

**Note:**

When you execute this function the RAM locations F600H to FCFFH will be loaded with example programs and useful routines.

See that you don't have your own programs in these locations while using this function.



### 5.1 Introduction

Six digits of 7 segment displays, a keyboard with 24 high performance keys and a dedicated Microcontroller forms keyboard and display section. A dedicated Microcontroller Atmel's AT89C51 is used here to control the operations of the keyboard and the seven segment displays. Necessary control program is stored in the flash of this dedicated controller.

The master controller AT89C52 which is in the main board communicates through the serial port of this slave controller for interaction.

But in the case of master the only available serial port is used for communicating with the PC and hence the second serial port is simulated using port lines.

The baud rate used for communication between the controllers is 2400.

Master controller sends necessary commands to the slave controller which in turn decodes the commands and sends necessary control signals to the keyboard and the display for proper operation.

Let us discuss the keyboard and the display separately.

### 5.2 Seven Segment Display

The seven segment displays are connected in multiplexed form to the slave controller to Port 0. The segments of all digits are connected and Port 1 is used to drive the displays.

The following table gives the bit position of the segments.

Bit position	D7	D6	D5	D4	D3	D2	D1	D0
Segment	a	b	g	e	c	.	f	d

### Seven Segment Display

Make the bit position of the segment as '1' when it is to be enabled. For hexadecimal numbers the required display data values is as shown below.

Number	display data
0	0DBH
1	048H
2	0F1H
3	0E9H
4	06AH
5	0ABH
6	0BBH
7	0C8H
8	0FBH
9	0EBH
A	0FAH
B	03BH
C	093H
D	079H
E	0B3H
F	0B2H

The slave expects a command word for selecting any digit in the display. Send the command word followed by the seven segment data to be displayed. The command words and the relevant digits are shown here.

Digit number	Command word
1	90H
2	91H
3	92H
4	93H
5	94H
6	95H



### Seven Segment Display

Necessary routines are given in the Software Examples Manual. The routines are also stored in the flash of the master controller and the relevant addresses are supplied there. Simply use LCALL instruction to call those routines.

### 5.3 Keyboard

The keyboard consists of 24 high performance keys controlled by the slave controller. Port 2 and Port 3 lines of the dedicated slave controller are used for interfacing the keyboard.

The high performance keys are scanned and checked by the slave for any key press. The slave controller also sends back the pressed key value to the master whenever requested.

Two command words are used for this purpose.

Command word 0A0H is issued by the master to make the slave scan the keyboard for any key press. If the return value from the slave controller is a non zero value then a key has been pressed.

Another command word 0A1H, which makes the slave read the value of the pressed key and return it to the master.



### 6.1 Introduction

This chapter gives information on various signals and connectors available in the Topview Trainer. You can refer previous chapter 3 to locate the signal/connector place.

In total, there are three connectors and one jumper.

CON1 - Bus Connector

CON2 - Power Connector

CON3 - Serial Port Connector

#### CON1 - Bus Connector

1. AD0	2. AD1
3. AD2	4. AD3
5. AD4	6. AD5
7. AD6	8. AD7
9. A0	10. A1
11. A2	12. A3
13. A4	14. A5
15. A6	16. A7
17. I/O ENABLE	18. GND
19. $\overline{\text{RD}}$	20. $\overline{\text{WR}}$
21. +5V	22. GND
23. A8	24. A9
25. A10	26. A11
27. A12	28. A13
29. A14	30. A15
31. P1.0	32. P3.0
33. P1.1	34. P3.1
35. P1.2	36. P3.2
37. P1.3	38. P3.3
39. P1.4	40. P3.4
41. P1.5	42. P3.5

### Connector and Jumper Details

43.	P1.6	44.	P3.6
45.	P1.7	46.	P3.7
47.	+5V	48.	GND
49.	+5V	50.	GND

#### CON 2 - Power Supply Connector

1 & 2 - +5V  
3 & 4 - GND

#### CON 3 - Serial Port Connector

2. RXD  
3. TXD  
5. GND

#### JUMPER - J1

Short pins 1 & 2 for selecting external code memory ( EPROM ).  
Short pins 2 & 3 for internal memory selection.

#### JUMPER - J2

Short pins 1 & 2 for Bus Connector.  
Short pins 2 & 3 for Serial Port

#### JUMPER - J3

Short pins 1 & 2 for Bus Connector.  
Short pins 2 & 3 for Serial Port

## 7.1 Introduction

The Topview Trainer is supported with an add-on board that contains many external interfacing options to implement many real time applications. Because of this add-on board, the utility of the trainer is enhanced in many ways. Facilities like Real Time Clock, 12 bit ADC, 8 bit DAC, Serial EEPROM, LCD module are made available in that add-on board to equip you to face real life challenges.

The following list gives the features of the add-on board :

- I<sup>2</sup>C RTC with battery backup.
- I<sup>2</sup>C EEPROM.
- 12 Bit Analog to Digital Converter with 8 input multiplexer.
- LCD Module with 2 line X 16 characters.
- Parallel Printer Port.
- Drivers for 2 Stepping Motors.
- 16 output port lines and 16 input port lines.
- An 8 bit Digital to Analog converter.

This chapter gives complete description about all these interfacing options and also the schematic circuit diagram meant for the board.

Address Decoder for the Add-on Board:

The address of all these peripheral modules are generated using IOENABLE control line coming out of the trainer. This control line is available in the Bus connector. This IOENABLE line becomes active (active low) between the address range, FF00H to FFFFH.

8 numbers of Chip Enable signals are generated using a 3 to 8 decoder. IOENABLE controls this decoder. The address lines A2, A3 and A4 are connected at decoder inputs. With this arrangement you get 8 different distinctive Chip Enable Signals are given here:

## Introduction

Chip Enable Signals	Address Range	Peripheral
CE0	FF00H to FF03H	LCD
CE1	FF04H to FF07H	Printer Port
CE2	FF08H to FF0BH	Printer Port
CE3	FF0CH to FF0FH	ADC
CE4	FF10H to FF13H	Stepping Motors
CE5	FF14H to FF17H	I/O Lines
CE6	FF18H to FF1BH	I/O Lines
CE7	FF1CH to FFFFH	DAC

7.2 I<sup>2</sup>C Bus Based EEPROM and RTC

The add-on board sports I<sup>2</sup>C bus based EEPROM and RTC devices. Reference information can be stored in the EEPROM and the RTC takes care of timing requirements of real life applications. These two devices have common bus lines and are connected with the port lines P1.2 and P1.3 through jumpers. As you can notice, these port lines are also connected to the Toggle switches in the trainer board itself. So, to get trouble free operation from these I<sup>2</sup>C devices, these toggle switches should be kept in OFF position. Otherwise these I<sup>2</sup>C devices may not behave properly.

The RTC device used in the add-on board is one of the most popular timing device from Philips, PCF8583. Apart from timing operations the device also carries 256 bytes of RAM.

A 32.768KHz crystal is connected between pins 1 and 2 to generate the required system clock. Also a 10pf capacitor is connected between pins 1 and 8 for fine-tuning.

The A0 line of the device is connected to low level. The data line, SDA is connected to the port line P1.3 and the clock, SCL is connected to the port line P1.2 of the microcontroller.

### I<sup>2</sup>C Bus Based EEPROM and RTC

A simple battery 3.6V is available on-board to retain the timing information in case of power failure.

One of the most popular EEPROM from the Atmel's table, AT24C04 of 512 bytes capacity is used in the add-on board. You can keep reference information, calibration tables, Set value variables in this EEPROM in real life applications.

The A0 line of the AT24C04 is pulled upto high level and other address lines A1 and A2 are connected to the low level. The Data line, SDA is connected to the port line P1.3 of the microcontroller and the clock line SCL is connected with the port line P1.2.

In case of requirement, you can add three more EEPROM devices with this I<sup>2</sup>C bus with different combination of address lines.

### 7.3 LCD Module Interfacing

A LCD module of configuration 2 lines X 16 character is available in the add-on board. The module is connected with the microcontroller's bus in 8 bit interfacing format. The data lines of the bus, AD0 to AD7 are connected to pins 7 to 14 of the LCD module.

The address lines A0 and A1 of the microcontroller is connected to R/W (pin 5) and RS (pin 4) of LCM.

**LCD Module Interfacing**

The addresses for different operations are tabulated here:

Write Command	FF00H.
Read Status	FF01H
Write Data	FF02H
Read Data	FF03H

A variable resistor is available to adjust the contrast of LCM.

**7.4 Parallel Printer Port**

The parallel printer port demands 10 output port lines and 5 input port lines.

This printer interface is designed using two latches, 74LS273, one buffer, 74LS244 and one open collector buffer 74LS07. The remaining unused lines of printer interface are used for ADC part to create more utility.

The 10 output lines are available using two output ports, 74LS273 and 5 input lines are generated using 74LS244. The Chip Enable Signals CE1 and CE2 are used to address two output ports and CE1 line is also used address the input port lines.

The input port lines D0 to D4 are used for the printer interface and the D7 line is used for ADC status checking.

The first output port using CE1 is used as data lines. The D6 and D7 bit of the second port is used as strobe and Initialize signals to the printer. These output lines are connected to the printer using an open collector non inverting buffer, 74LS07.

The D0 to D4 bits of the second output port using CE2 used in the ADC interface.



### **7.5 High Performance ADC Circuit**

The ADC part of the add-on board is designed using Intersil's most popular and industry standard 12 bit Analog to Digital converter ICL 7109. In fact this converter is capable of giving results in 12 bit absolute resolution along with 2 more bits for over range and polarity indications.

This ADC is supported with an 8 channel single ended input multiplexer using CD 4051.

The AD converter is capable of managing low speed input signals upto 30 samples per second. Ideally this is suitable for measuring temperature, pressure, load cell variables. The polarity bit indicates the polarity of the input signal. The over range bit indicates the pressure of analog input voltages exceeding maximum input limits.

The reference voltage for this AD converter is given between pins 39 and 36. Maximum allowable input range is twice of this reference voltage. You can even adjust this reference voltage using a variable resistor in the board.

The output of ADC is coming in a 14 bit format and the microcontroller reads this in two read cycles.

If you read the address FF0CH, the data bits B0 to B7 of ADC are read into controller. When reading the next location, FF0DH you can get remaining 4 data bits, polarity and Over Range bits.

The D0 to D4 bits of output port at address (FF08H to FF0BH) are used to select analog input channel and also enabling the multiplexer.

## **7.6 Stepping Motors**

Incorporating drivers for the stepping motors is a relatively an easy affair. The add-on board contains drivers to control two motors. Anywhere, this document doesn't give any detailed information on these motors. You can refer any decent text book to get the correct picture on these motors.

Basically to drive a stepping motor, you need 4 output lines and these lines control 4 phases (corresponding to 4 windings of the stepping motor).

So, as you can see, a facility to drive a byte of data is sufficient to control these two stepping motors. The lower nibble drives one motor and upper nibble takes care of second motor.

Power transistors are used to source required high current into motor windings. The most popular transistors BC147, SL100 and 2N 3055 are used for this purpose.

## **7.7 Input/output Port lines**

No trainer is complete without input and output port lines. So this add-on board comes with 32 I/O lines to take care of operations. You can use these lines to manage relays, solenoids, input sensing, output driving requirements. There are two output latches ( 16 output lines) using 74LS273 and two input buffers (16 input lines) using 74LS244.

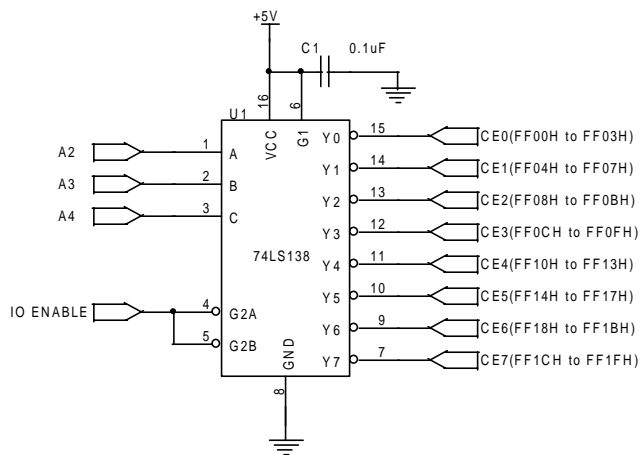
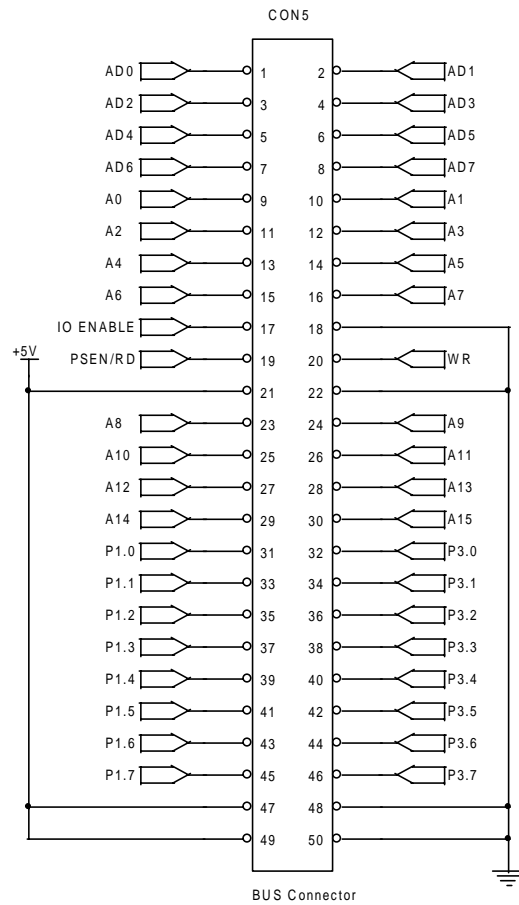
The Chip Enable Signals CE5 and CE6 used to control these I/O devices. The access addresses are FF14H and FF18H. The same address is used for both input and output operations. If you read that address input buffers are accessed and when you write into a latch, that latch becomes output port.

## 7.8 Digital to Analog Converter

This 8 bit DAC is an after thought in this add-on board. It is designed using DAC 0808 or AD1408.

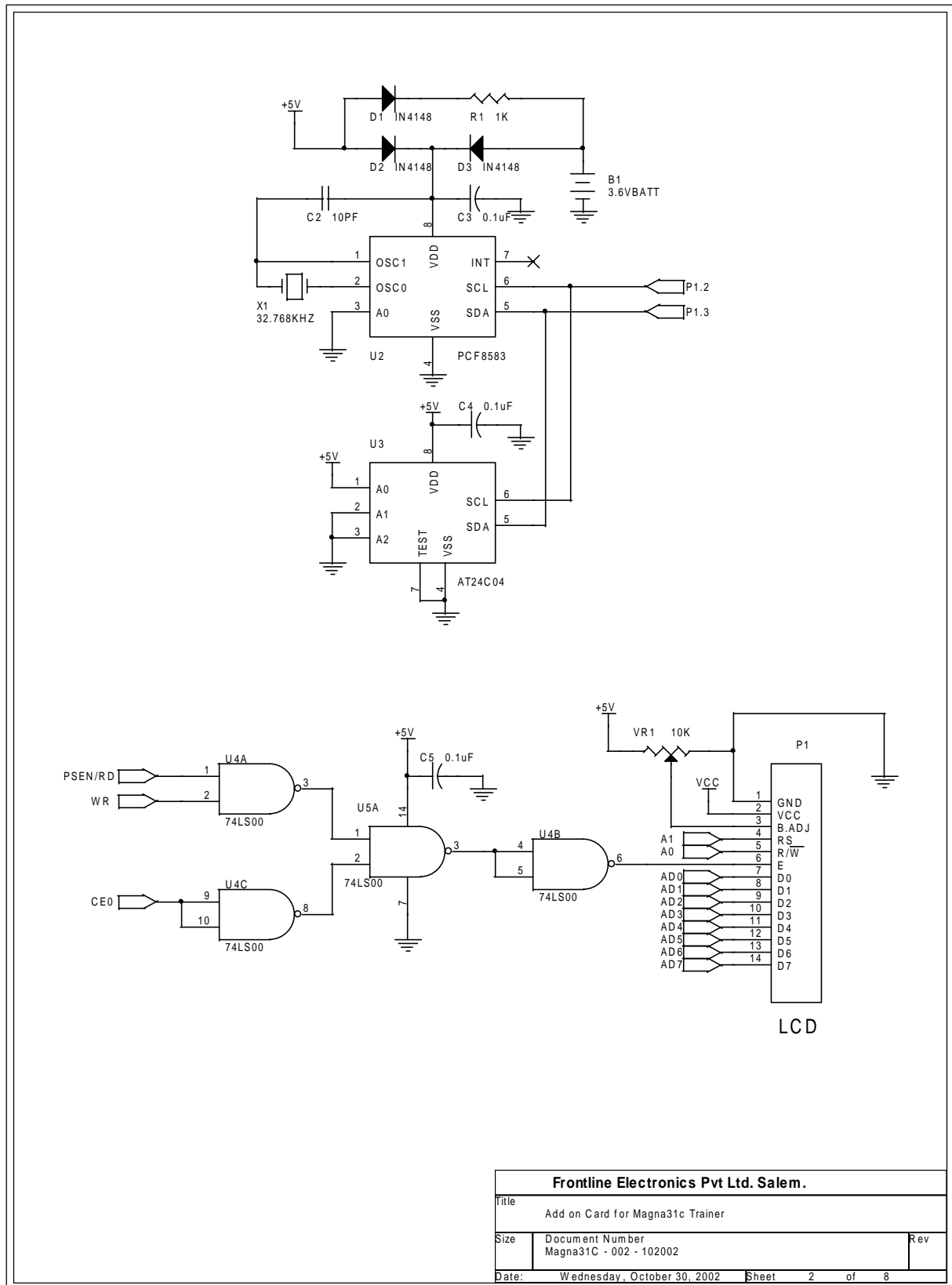
One 8 bit output latch, 74LS273 is used to drive out the digital data and the OP-AMP, LM741 sitting at the output of DAC device generates the required output analog voltage. The Chip Enable Signal CE7 is meant for this DAC. A variable resistor is provided to adjust the output range.

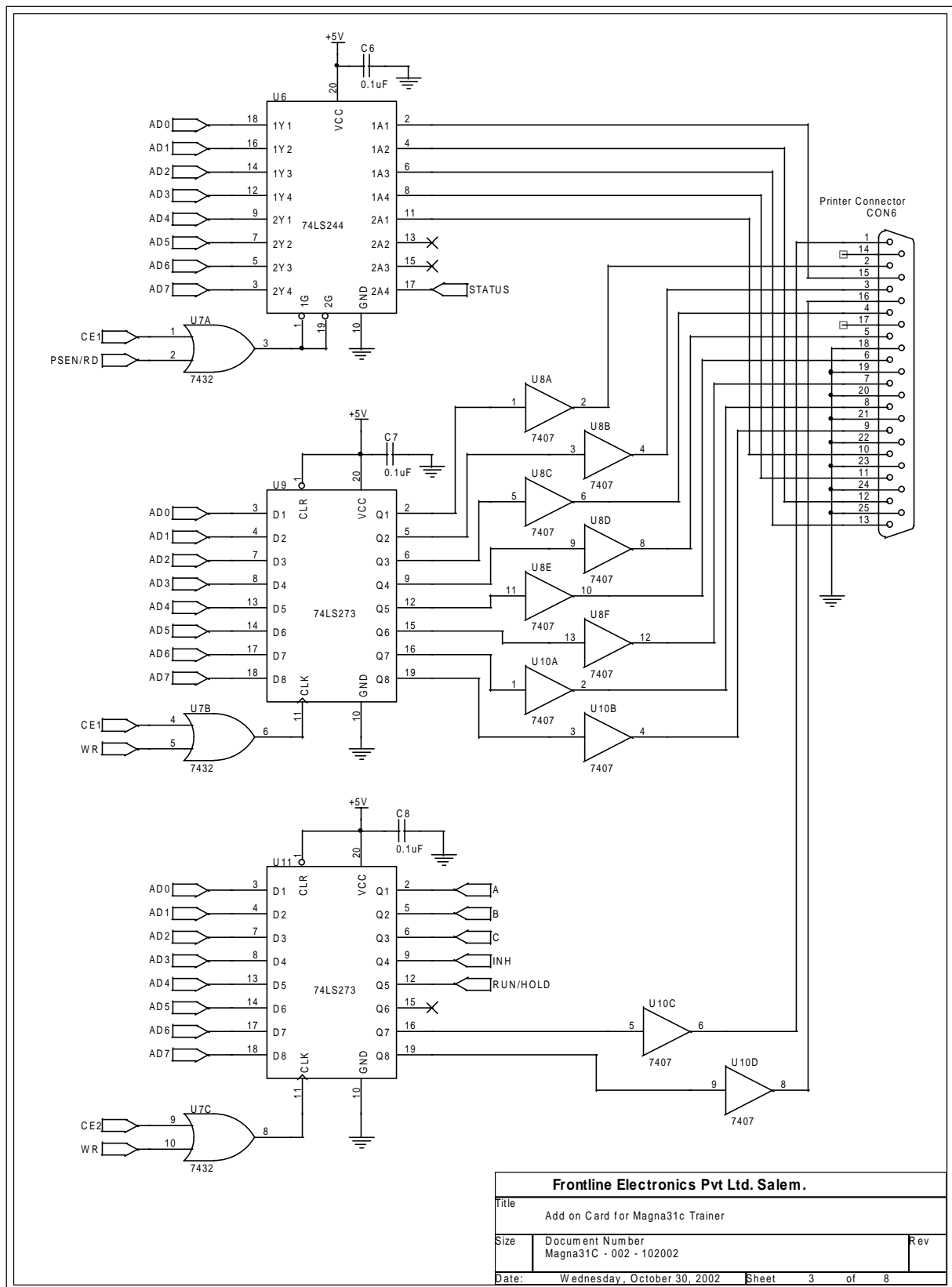
You can have the choice of getting unipolar or bipolar analog output from the DAC.

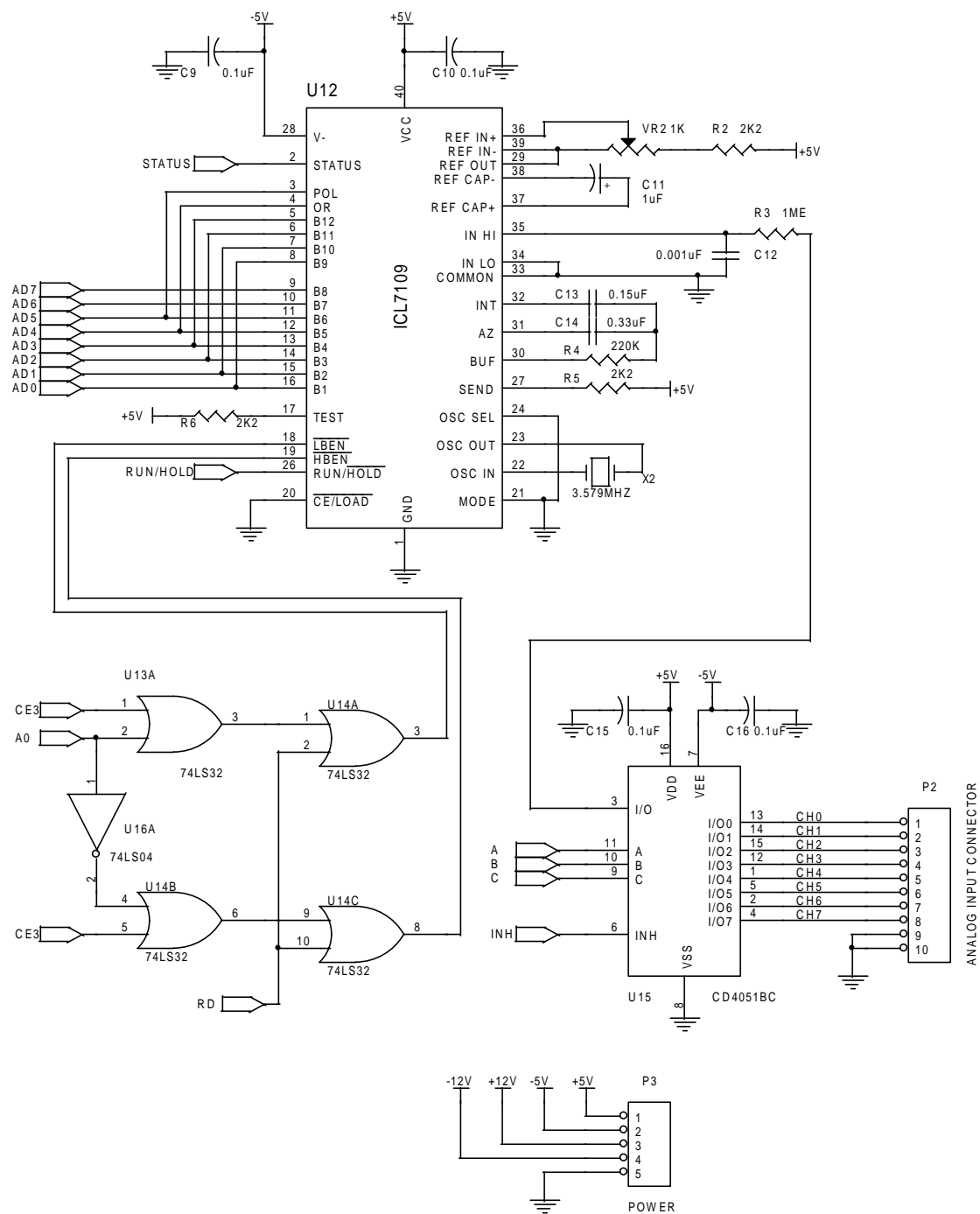


**Frontline Electronics Pvt Ltd. Salem.**

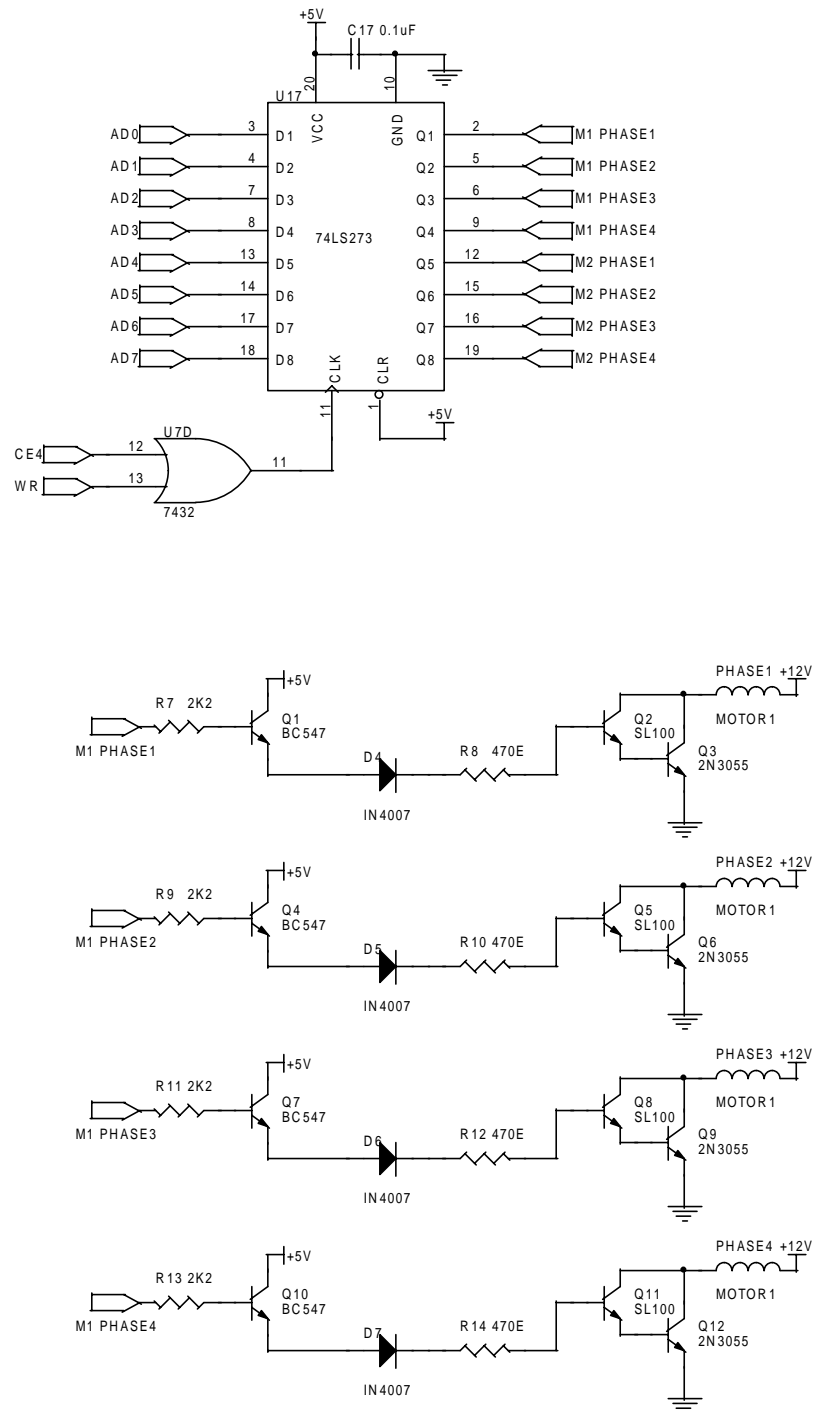
Title		
Add on Card for Magna31c Trainer		
Size	Document Number	Rev
	Magna31C - 002 - 102002	
Date: Wednesday, October 30, 2002 Sheet 1 of 8		







Frontline Electronics Pvt Ltd. Salem.		
Title		
Add on Card for Magna31c Trainer		
Size	Document Number	Rev
	Magna31C - 002 - 102002	
Date:	Wednesday, October 30, 2002	Sheet 4 of 8

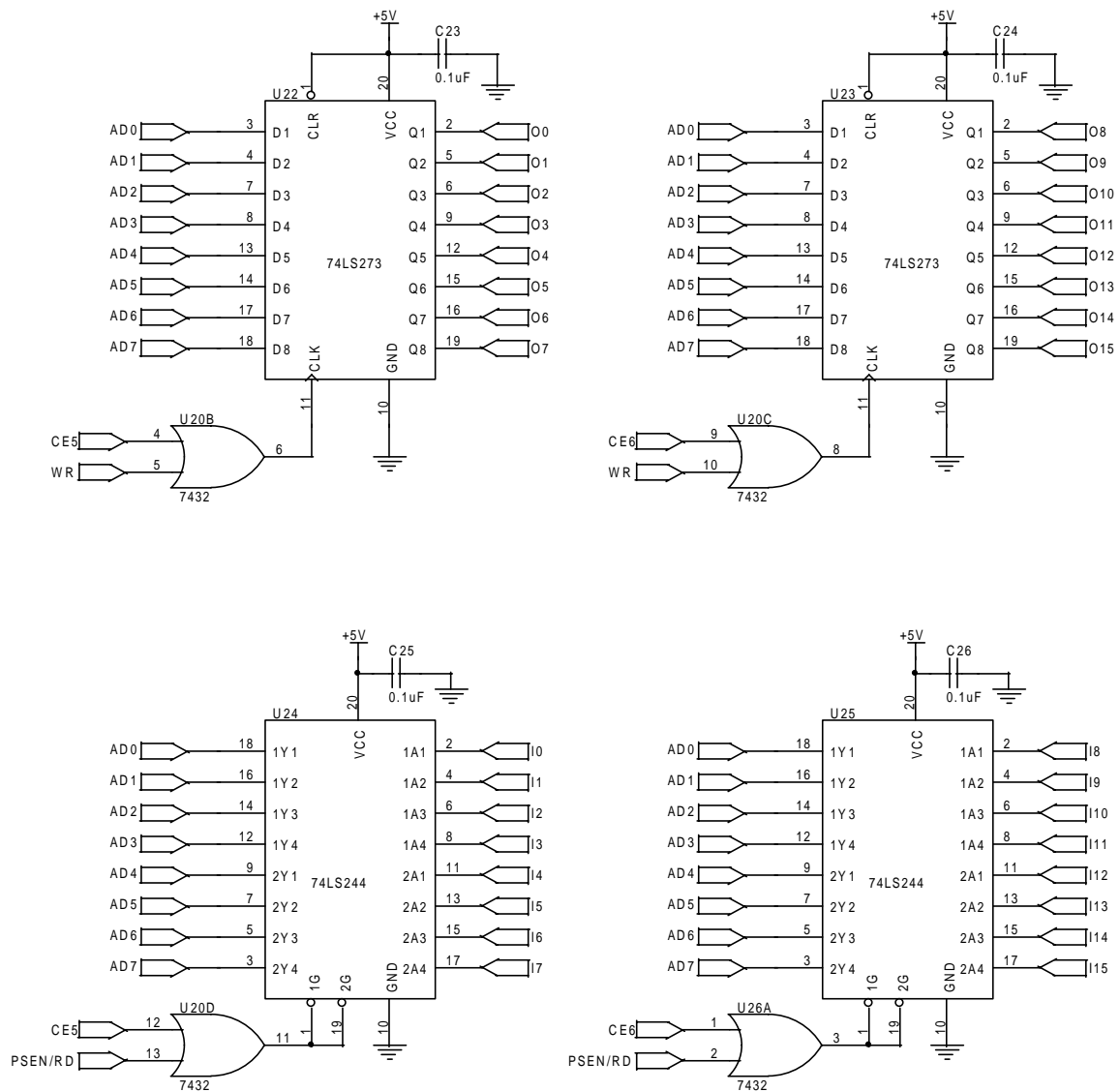


# Frontline Electronics Pvt Ltd. Salem.

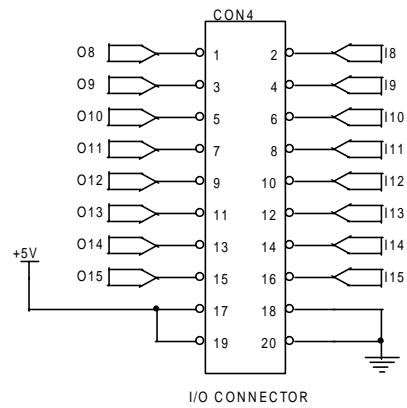
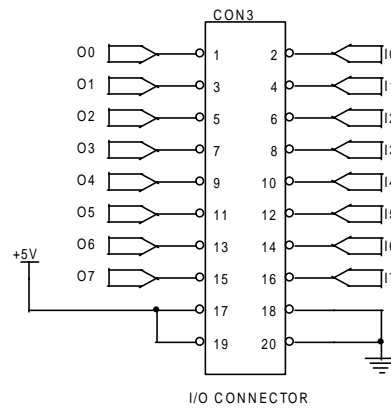
Title		
Add on Card for Magna31c Trainer		
Size	Document Number	Rev
	Magna31C - 002 - 102002	
Date: Wednesday, October 30, 2002		
Sheet 5 of 8		







Frontline Electronics Pvt Ltd. Salem.		
Title	Add on Card for Magna31c Trainer	
Size	Document Number Magna31C - 002 - 102002	Rev
Date:	Wednesday, October 30, 2002	Sheet 7 of 8



Frontline Electronics Pvt Ltd. Salem.		
Title		
Add on Card for Magna31c Trainer		
Size	Document Number	Rev
	Magna31C - 002 - 102002	
Date: Wednesday, October 30, 2002 Sheet 8 of 8		



**FRONTLINE ELECTRONICS PVT LTD**

1/255C - Thatha Gounder St, Kumaran Nagar, Alagapuram,  
Salem - 636 016, Tamilnadu. India.

Phone : 0091 427 - 244 9238 / 243 1312. Fax : 0091 427 - 244 9010.

Email : [feplslm@frontlinemail.com](mailto:feplslm@frontlinemail.com)

**[www.Frontline-Electronics.com](http://www.Frontline-Electronics.com)**