MULTIPURPOSE MICROCONTROLLER BOARD

JAPINDER SINGH CHUADHRY (071036)

Under the supervision of

Mr. JITENDRA MOHAN

juit
विद्या तत्व ज्योतिसम

May 2011

Submitted in partial fulfillment for the Degree of

Bachelor of Technology

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY,

WAKNAGHAT

CERTIFICATE

This is to certify that the work titled **MULTIPURPOSE MICROCONTROLLER BOARD** submitted by **JAPINDER SINGH CHAUDHRY** in partial fulfillment for the award of degree of **Bachelor of Technology in Electronics & Communication Engineering** of Jaypee University of Information Technology, Waknaghat has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.

Signature of Supervisor ……………………

Name of Supervisor JITENDRA MOHAN

Designation Senior Lecturer

Date 21/05/11

ACKNOWLEDGEMENT

"The aim of making the major project is to correlate the theoretical course contents with practical applications of the components and to get to know the degree of one's own capabilities when contributing to the conversion of scientific knowledge into practical results."

Availing the facility of making the final year's major project provide us an outlook towards the practical electronics world I came to know about the components, their versions, their versatility among their choice to implement in the project help us to a great deal to make this project successful.

I express our heartfelt thanks to Sr. Lecturer JITENDRA MOHAN, who through his expert guidance helped us throughout the course of this project. If it were not his motivation and encouragement, I would not have seen through this project in an honest course to the splendor of success.

Finally, I wish to convey our gratitude to all the faculty members of the Electronics & Communication department of our college for providing necessary help and encouragement in the course of completion of this project.

Signature of the student ....Japinder Singh....

Name of Student JAPINDER SINGH.

Date ....21/05/11....

Table of Contents

ABSTACT

Multipurpose Microcontroller board in general sense is define as a device which is capable to perform many kind of microcontroller based operation ranging from simple LED interfacing to other complex peripheral interfacing such as Analog to Digital converter, motors , LCD, serial communication etc.

Multipurpose microcontroller board serves the purpose of explaining the students about different application of microcontroller. It can be used for generation of various waves like square, triangular waves & is much useful for embedded application and different kind of robotic projects. This project can help the beginners in field of embedded system and robotics by giving them practical knowledge in comparison to theoretical knowledge.

Any beginner having no idea about circuitry can test there code by running it practically on this board. It is project which can be used to perform any number of applications depending upon the modules which are interfaced & the devices for which the code has been prepared.

The main aim of the project is to expose the student to the industrial technical problems to which he is to be exposed in the future life where Making Things Right in the first instance is the driving motto, perfection and accuracy are inevitable.

In conclusion, I must say that the project has helped me to enhance my working skills & stamina and to further enlighten me to enter a new phase of life after completion of this project

LIST OF FIGURES

CHAPTER – 1 INTRODUCTION

1.1 OVERVIEW

- Multipurpose Microcontroller Board in general sense is define as a device which is capable to perform many kind of microcontroller based operation ranging from simple led interfacing to other complex peripheral interfacing such as Analog to Digital converter, stepper motor , LCD, serial communication etc.
- It can be used for generation of various waves like square, triangular fed to the Controller chip.
- Useful for embedded application and robotics project.
- Helpful for beginner in field of embedded system and robotics.

1.2 OBJECTIVE

- Our approch to gain knowledge of interfacing different device with microcontroller for Useful application and fullfill the hardware module as per our requirement and College.
- Microcontroiller Board is such a project which can be used to perform any kind of application depend on the module which are interfaced.
- These leads us to design microcontroller board.
- Once we have developed a microcontroller kit we can able to perfrom practise of any project.

1.3 SYSTEM SPECIFICATIONS

The following are the list of interfacing devices and components we have used in our project.

- PHILIPS Microcontroller Chip P89C51RD2BN.
- LEDs.
- LCD (2X16) and (4x20) Is Optional.
- Serial Port (9 pin DB connector).
- ADC (8-bit, single channel).
- Tactile Switches for input operations.
- Relays driver.
- Matrix keypad (4x4).
- Programmer (ISP).
- +5V supply.
- Single sided designed PCB.
- Connectors.
- Motor driver circuit.
- 2 Seven segment with its driver 7447.
- Software: flash magic, keil compiler, diptrace, tina .

1.4 TOOL USED

- PCB layout designing software diptrace 1.5
- KATANA (adobe postscript driver) for converting file generated from diptrace to corel draw format (.prn).
- Corel draw x4 for designing PCB blue print.
- VISUAL BASIC for designing parallel port interfacing software.
- Parallel port driver for parallel port interfacing.

1.5 FEATURE

1. Single sided designed PCB is used & no jumper is involved.
2. Lower power consumption & long life since peripheral got power only after connecter is inserted.
3. Low cost.
4. External more peripheral can be interfaced.
5. Easily inserted connector.
6. Single connector for each peripheral.
7. No external eliminator / power supply needed.

CHAPTER -2 PERIPHERAL INTERFACING

2.1 BLOCK DIAGRAM

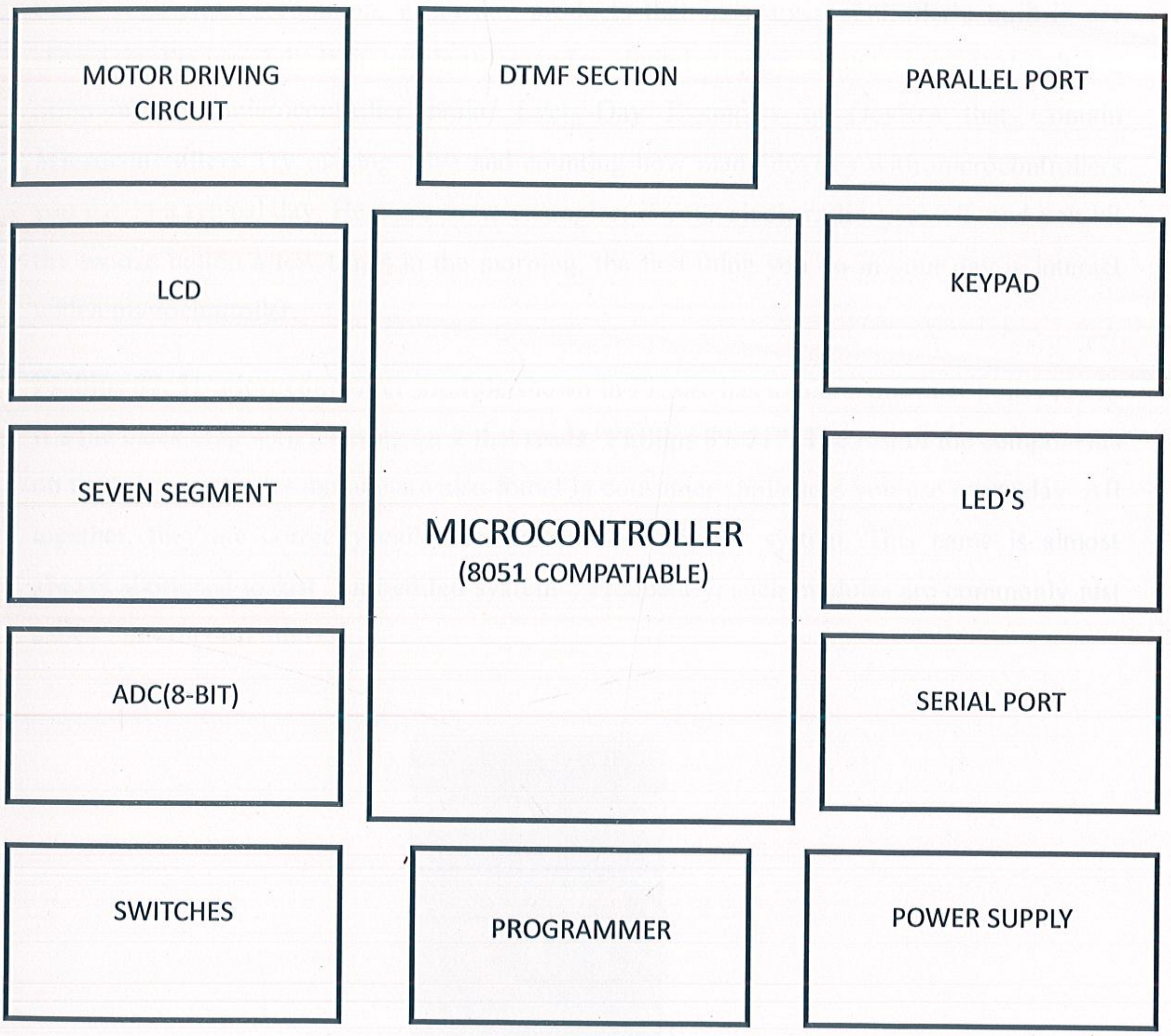


FIG.-2.1- BLOCK DIAGRAM OF PROJECT

2.2 MICROCONTROLLERS 8051 SERIES

A microcontroller is a kind of miniature computer that you can find in all kinds of gizmos. Some examples of common, every-day products that have **microcontroller's built-in** are shown in Fig.ure 2-1. If it has buttons and a digital display, chances are it also has a programmable microcontroller brain. Every-Day Examples of Devices that Contain **Microcontrollers** Try making a list and counting how many devices with microcontrollers you use in a typical day. Here are some examples: if your **clock radio** goes off, and you hit the snooze button a few times in the morning, the first thing you do in your day is interact with a microcontroller.

Phillips 89c51 microcontroller module shown in Fig.ure has a microcontroller built onto it. It's the black chip with lettering on it that reads "Phillips 89c51". The rest of the components on the microcontroller module are also found in consumer appliances you use every day. All together, they are correctly called an embedded computer system. This name is almost always shortened to just "**embedded system**". Frequently, such modules are commonly just called "**microcontrollers.**"

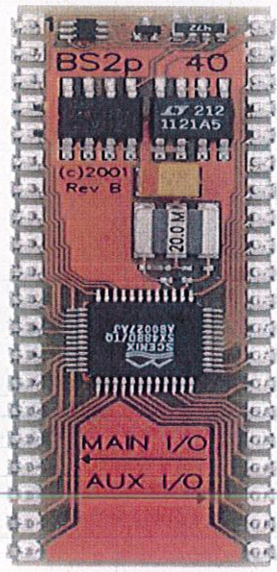


FIG. 2.2: 40 PINS PHILLIPS 89C51 MICROCONTROLLER

2.3 TYPES OF MICROCONTROLLER AND ITS FAMILY

(A) INTEL 8051 AND ITS FAMILY

In 1981, Intel Corporation introduces an 8-bit microcontroller called 8051. The Intel 8051 became widely popular and allowed other companies to produce any flavor of 8051 but with condition that 'code remains compatible with 8051'.

Other two members in 8051 family of microcontroller are 8052 & 8031.

Some other companies producing member of 8051 family are:

1. Intel
2. Phillips
3. Dallas Semiconductors
4. Philips/ Signetics
5. Siemens

(B) PIC MICROCONTROLLER (MICROCHIP TECHNOLOGY)

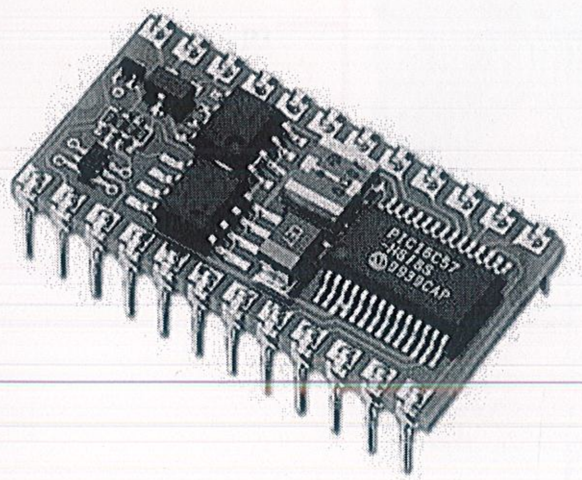


FIG. 2.3: PIC MICROCONTROLLER

2.4 INSIDE 8051:

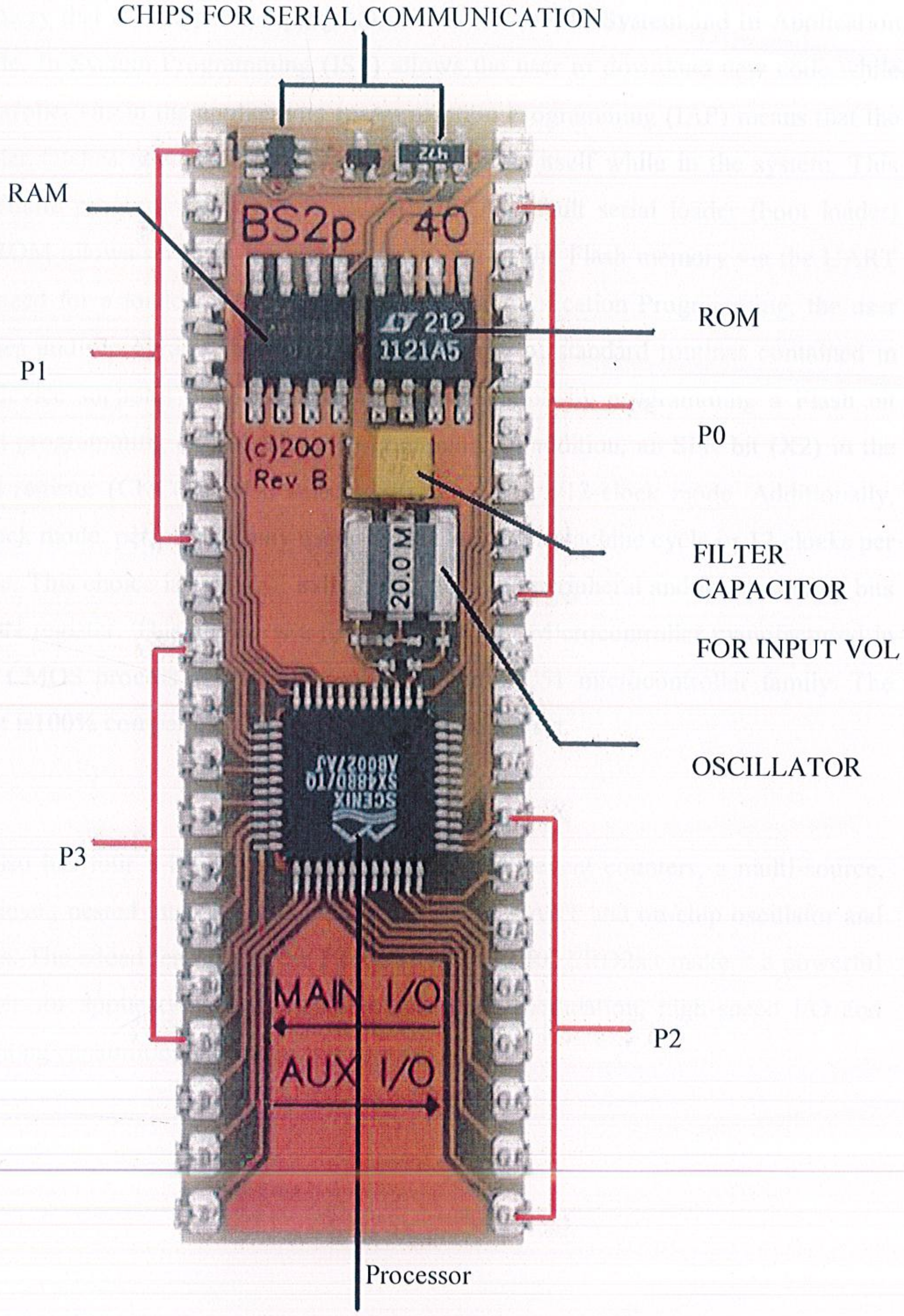


FIG. 2.4: INSIDE MICROCONTROLLER

2.5 DESCRIPTION OF PHILLIPS P89C51RD2BN

The P89C51RA2/RB2/RC2/RD2xx contains a non-volatile8KB/16KB/32KB/64KB Flash program memory that is both parallel programmable and serial In-System and In-Application Programmable. In-System Programming (ISP) allows the user to download new code while the microcontroller sits in the application. In-Application Programming (IAP) means that the microcontroller fetches new program code and reprograms itself while in the system. This allows for remote programming over a modem link. A default serial loader (boot loader) program in ROM allows serial In-System programming of the Flash memory via the UART without the need for a loader in the Flash code. For In-Application Programming, the user program erases and reprograms the Flash memory by use of standard routines contained in ROM. The device supports 6-clock/12-clock mode selection by programming a Flash bit using parallel programming or In-System Programming. In addition, an SFR bit (X2) in the clock control register (CKCON) also selects between 6-clock/12-clock mode. Additionally, when in 6-clock mode, peripherals may use either 6 clocks per machine cycle or 12 clocks per machine cycle. This choice is available individually for each peripheral and is selected by bits in the CKCON register. This device is a Single-Chip 8-Bit Microcontroller manufactured in an advanced CMOS process and is a derivative of the 80C51 microcontroller family. The instruction set is100% compatible with the 80C51 instruction set.

The device also has four 8-bit I/O ports, three 16-bit timer/event counters, a multi-source, four-priority-level, nested interrupt structure, an enhanced UART and on-chip oscillator and timing circuits. The added features of the P89C51RA2/RB2/RC2/RD2xx make it a powerful microcontroller for applications that require pulse width modulation, high-speed I/O and up/down counting capabilities such as motor control.

(A) FEATURES

- 80C51 Central Processing Unit
- On-chip Flash Program Memory with In-System Programming (ISP) and In-Application Programming (IAP) capability
- Boot ROM contains low level Flash programming routines for downloading via the UART.
- Can be programmed by the end-user application (IAP)
- Parallel programming with 87C51 compatible hardware interface to programmer
- Supports 6-clock/12-clock mode via parallel programmer (default clock mode after Chip Erase is 12-clock)
- 6-clock/12-clock mode Flash bit erasable and programmable via ISP
- 6-clock/12-clock mode programmable "on-the-fly" by SFR bit
- Peripherals (PCA, timers, UART) may use either 6-clock or 12-clock mode while the CPU is in 6-clock mode
- Speed up to 20 MHz with 6-clock cycles per machine cycle (40 MHz equivalent Performance); up to 33 MHz with 12 clocks per machine cycle
- Fully static operation
- RAM expandable externally to 64 Kbytes.
- Four interrupt priority levels
- Seven interrupt sources
- Four 8-bit I/O ports
- Full-duplex enhanced UART

– Framing error detection

– Automatic address recognition

• Power control modes

– Clock can be stopped and resumed

– Idle mode

– Power down mode

• Programmable clock-out pin

• Second DPTR register

(B) PIN DIAGRAM-

Pin description of 8051 series of microcontroller is show below in Fig. 2.5.

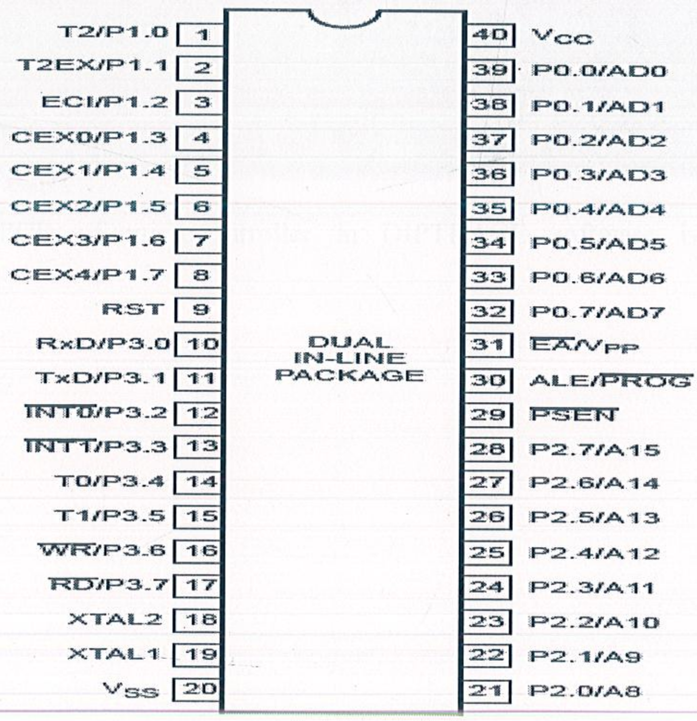
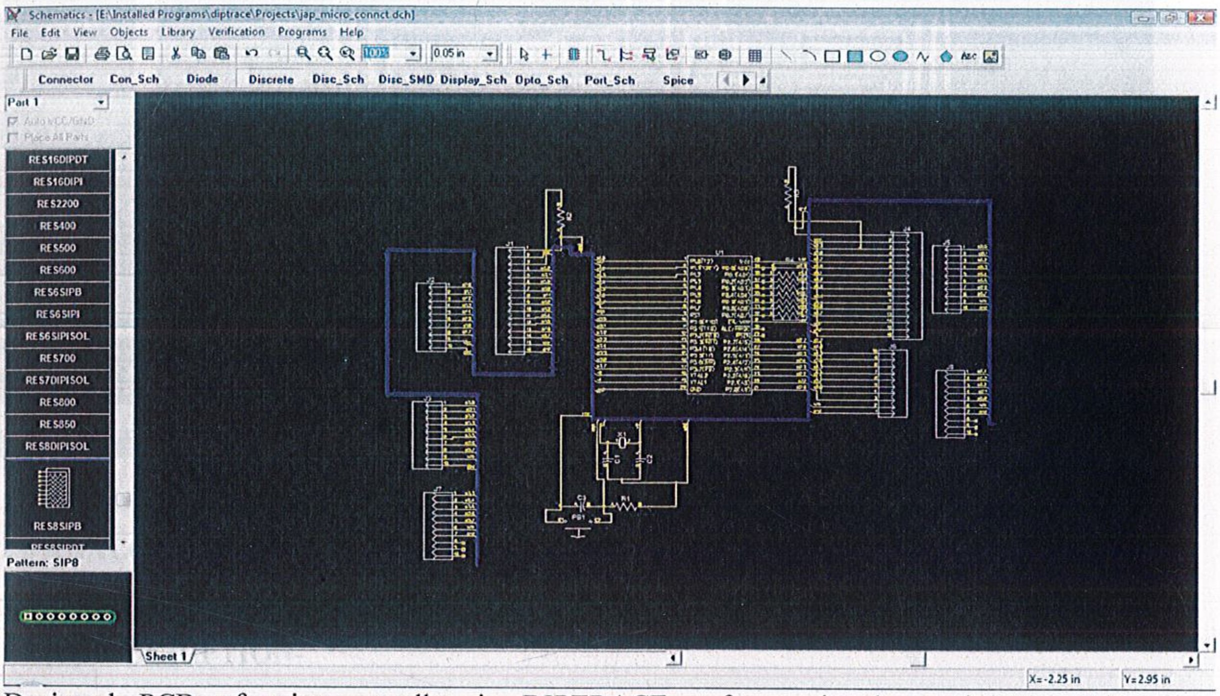


FIG. 2.5: PIN DESCRIPTION

(C) PCB DESIGN OF MICROCONTROLLER-

Designed circuit diagram in DIPTRACE software



Designed PCB of microcontroller in DIPTRACE software is shown in Fig. 2.6.

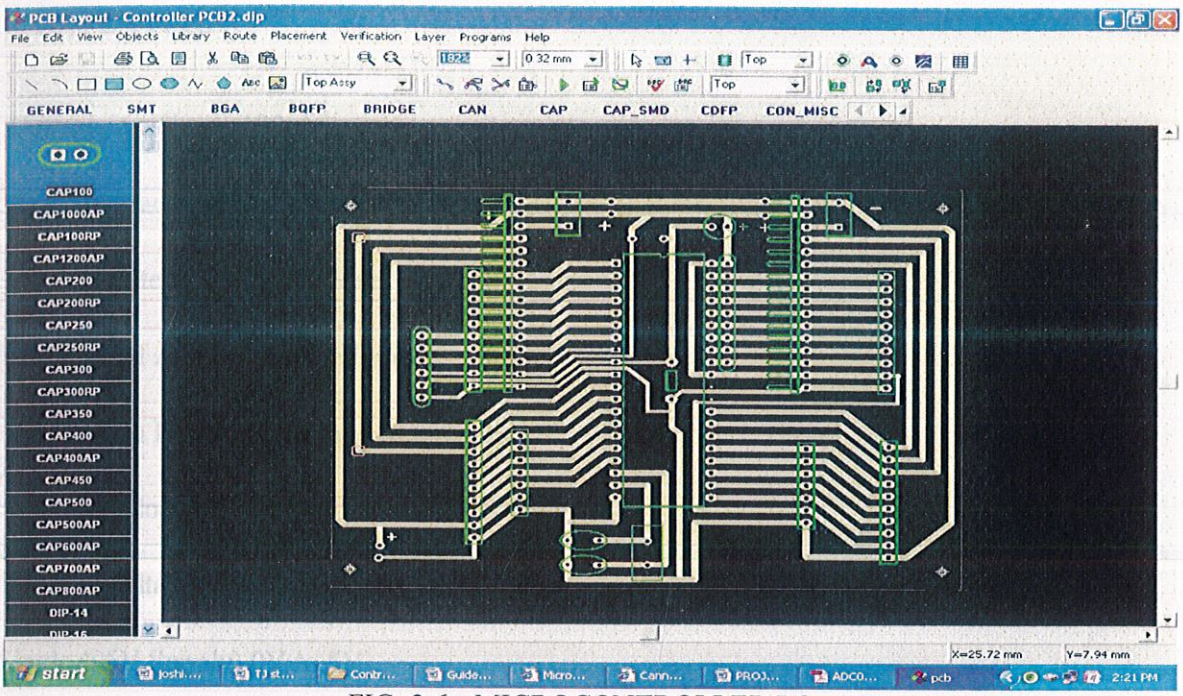


FIG. 2.6:- MICROCONTROLLER PC

2.6 ANALOG TO DIGITAL CONVERTOR

(A) DESCRIPTION-

The ADC080X family is CMOS 8-Bit, successive approximation A/D converters which use a modified potentiometric ladder and are designed to operate with the 8080A control bus via three-state outputs. These converters appear to the processor as memory locations or I/O ports, and hence no interfacing logic is required. The differential analog voltage input has good common mode- rejection and permits offsetting the analog zero-input voltage value. In addition, the voltage reference input can be adjusted to allow encoding any smaller analog voltage span to the full 8 bits of resolution.

(B) FEATURES-

- Conversion Time <100μs.
- Easy Interface to Most Microprocessors.
- Will Operate in a "Stand Alone" Mode.
- Differential Analog Voltage Inputs.
- Works with Band gap Voltage References.
- On-Chip Clock Generator.
- Analog Voltage Input Range.

(Single + 5V Supply) 0V to 5V.

- No Zero-Adjust Required.

(C) PIN DIAGRAM

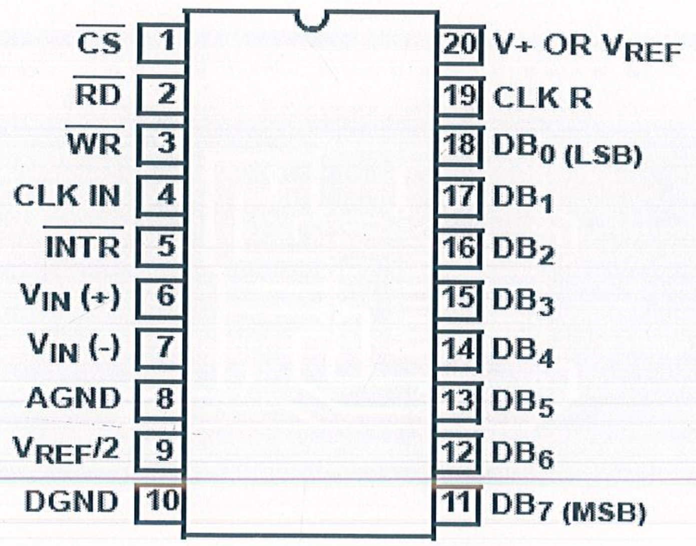


FIG. 2.7: ADC0804 PIN DIAGRAM

(D) INTERFACING WITH MICROCONTROLLER

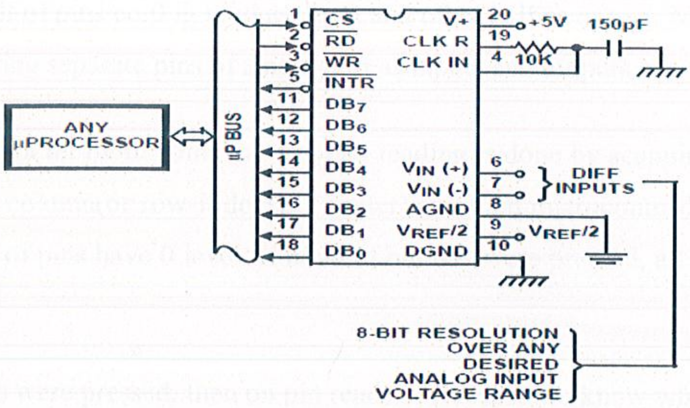


FIG. 2.8: INTERFACING CIRCUIT DIAGRAM OF ADC0804

(E) PCB DESIGN OF A/D-

Designed PCB Layout of ADC converter chip ADC0804 in DIPTRACE software is shown in Fig.2.9 below.

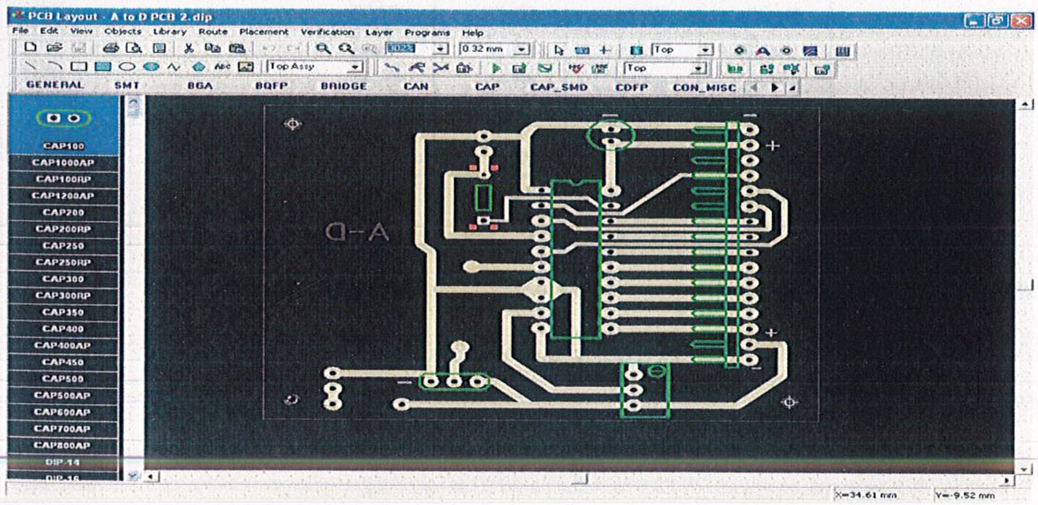


FIG. 2.9: ADC0804 PCB DESIGN

2.7 MATRIX KEYPAD

(A) INTRODUCTION:-

Keypad usually is organized as crossed rows and columns if button matrix. Such keypad may be read with half of pins conFig.ured as input and other half as output. Most microcontrollers allow conFig.uring separate pins of single port as inputs and outputs.

Matrix keypad is a bit more complex. Keypad reading is done by scanning columns. So loop sends 0 to each column(or row-it doesn't matter) and then microcontroller reads all rows by checking if any of pins have 0 level. If none of buttons were pressed, all input pins will be at 1level.

If one of buttons were pressed, then on pin reads 0. Because we know which column were set to 0. We can define which button was pressed by crossing row and column. This way it is possible to interface more buttons that there are free microcontroller pins. With one 8 bit port you can interface 16 buttons.

(B) GRAM OF 4X4 MATRIX KEYPAD

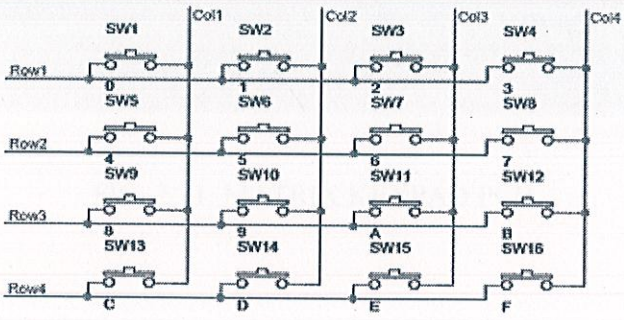


FIG. 2.10:- 4X4 MATRIX KEYPAD

(C) PCB DESIGN-

Designed single sided PCB layout of Matrix keypad is show below in Fig. 2.11.

The designed keypad consist of 4 Row and 4 Colum thus making it a 4x4 matrix.

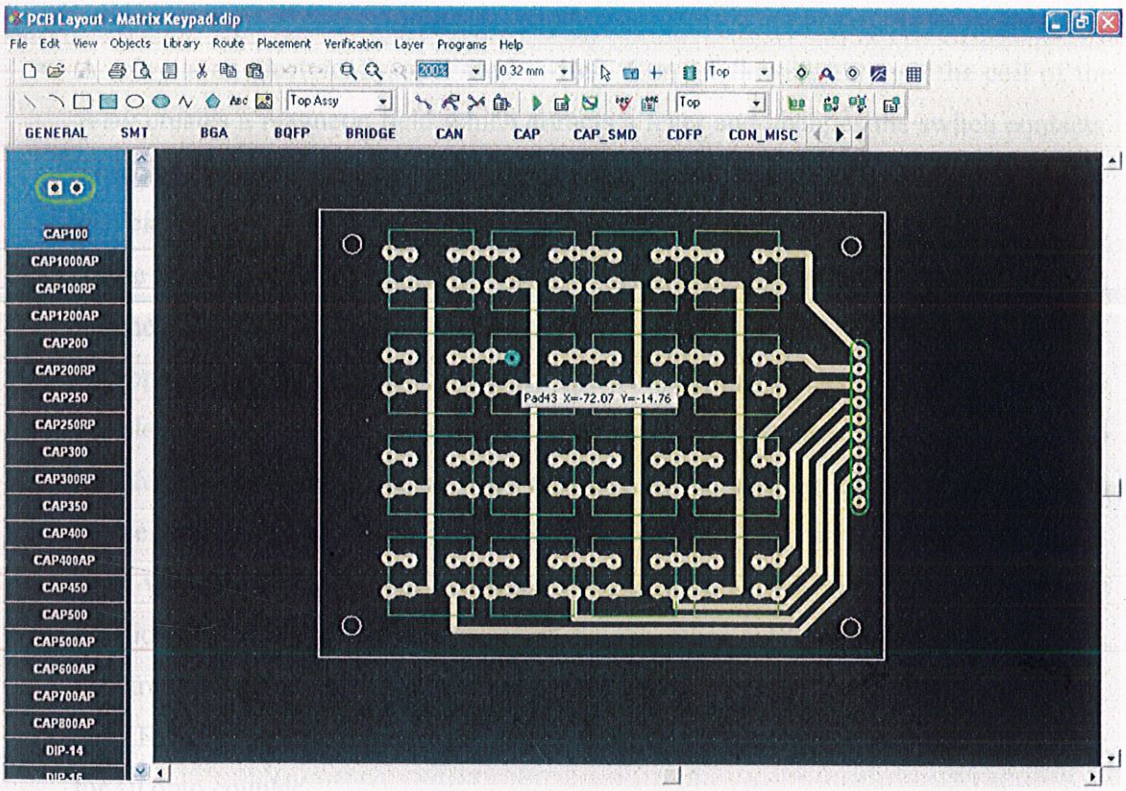
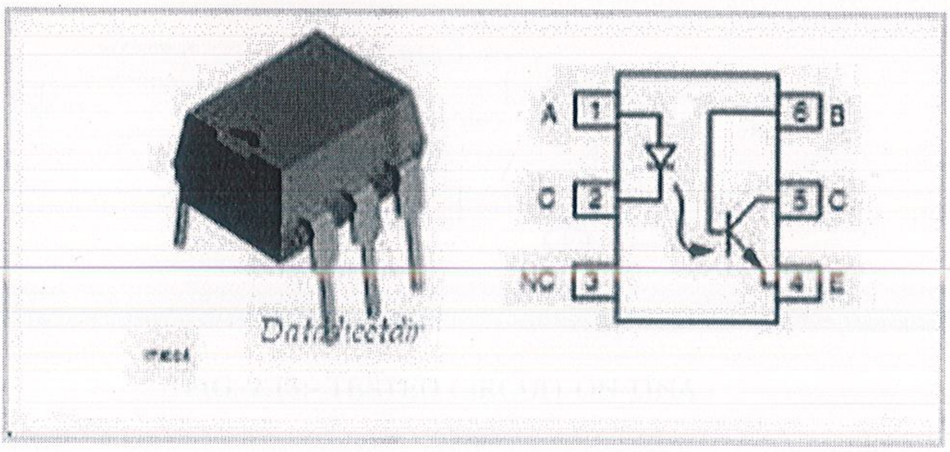


FIG. 2.11: MATRIX KEYPAD PCB

2.8 RELAY DRIVER & OPTOCOUPLER-

(A) DESCRIPTION-

- A relay is an **electrically operated switch**. Current flowing through the coil of the relay creates a magnetic field which attracts a lever and changes the switch contacts. The coil current can be on or off so relays have two switch positions. Relays allow one circuit to switch a second circuit which can be completely separate from the first. There is no electrical connection inside the relay between the two circuits, the link is magnetic and mechanical.
- An opto-isolator (or optical isolator, optical coupling device, optocoupler, photo coupler, or photo MOS) is a device that uses a short optical transmission path to transfer an electronic signal between elements of a circuit, typically a transmitter and a receiver, while keeping them electrically isolated. The Main Purpose is to prevent high voltages or fluctuating voltages on one side of the circuit from damaging components or distorting transmissions on the other side. Opto couplers are often constructed with light emitting diodes (LEDs) as part of the optical transmitter side. An LED produces light when voltage is added to it, making it a perfect light source for an opto coupler



(B) CIRCUIT DIAGRAM -

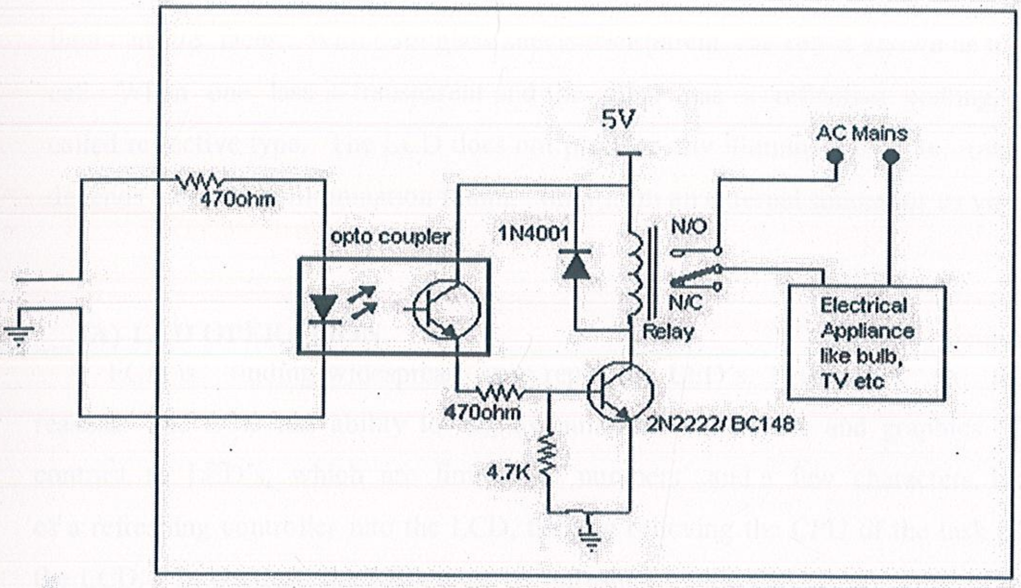


FIG. 2.12: RELAY USING OPTOCOUPLER CIRCUIT

(C) CIRCUIT TESTED-

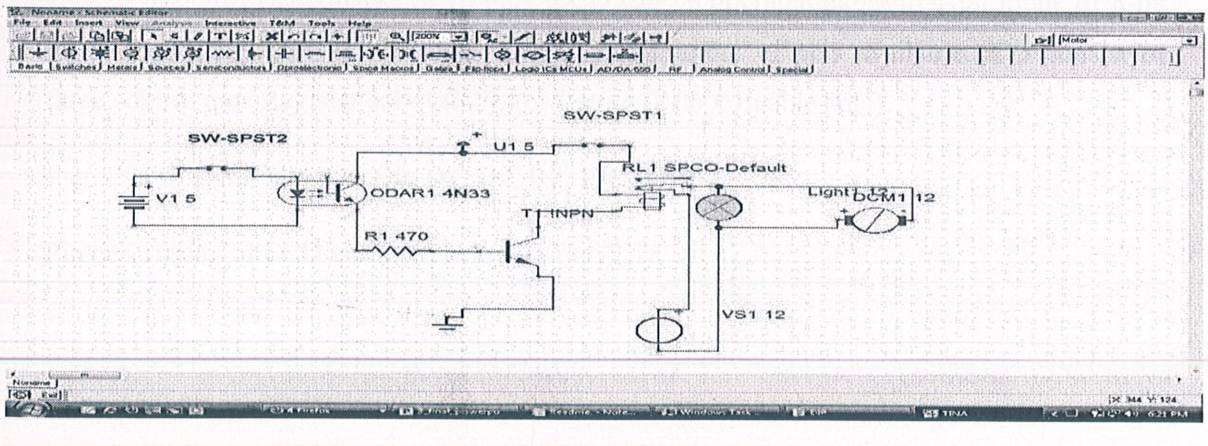


FIG. 2.13:- TESTED CIRCUIT ON TINA

2.9 LCD

LCD A liquid crystal cell consists of a thin layer (about 10 μ m) of a liquid crystal sandwiched between two glass sheets with transparent electrodes deposited on their inside faces. With both glass sheets transparent, the cell is known as transitive type cell. When one lass is transparent and the other has a reflective coating, the cell is called reflective type. The LCD does not produce any illumination of its own. It, in fact, depends entirely on illumination falling on it from an external source for its visual effects.

(A) LCD OPERATION

LCD is finding widespread use replacing LED's. This is due to the following reasons. The LCD has ability to display numbers, characters, and graphics. This is in contrast to LED's, which are limited to numbers and a few characters. Incorporation of a refreshing controller into the LCD, thereby relieving the CPU of the task of refreshing the LCD. In contrast, the LED must be refreshed by the CPU (or in some other way) to keep displaying the data.

(B) DESCRIPTIONS OF LCD

Liquid LCD consists of 16 pins. LCD can be operated either in 4-bit mode or 8-bit mode depending upon the requirement and requirement of pins available.LCD consist of three control pins Register select (RS), read/write (R/W), Enable (EN).

It consist of eight data pins D0, D1, D2, D3, D4, D5, D6, D7

Pin description of LCD is shown below in Fig. 2.18

PIN	SYMBOL	INPUT/OUPUT	DESCRIPTION
1	V_{SS}	--	Ground
2	V_{CC}	--	+5V power supply
3	V_{EE}	--	Power supply To control contrast
4	RS	Input	RS=0 to select Command register, RS=1 to select Data register
5	R/W	Input	R/W=0 for write, R/W=0 for read
6	E	Input/output	Enable
7	DB0	Input/output	The 8-bit data bus
8	DB1	Input/output	The 8-bit data bus
9	DB2	Input/output	The 8-bit data bus
10	DB3	Input/output	The 8-bit data bus
11	DB4	Input/output	The 8-bit data bus
12	DB5	Input/output	The 8-bit data bus
13	DB6	Input/output	The 8-bit data bus
14	DB7	Input/output	The 8-bit data bus

FIG. 2.18: LCD PIN DESCRIPTION

(C) TABLE FOR -- LCD COMMAND CODES

CODES COMMAND TO LCD INSTRUCTION	
(HEX)	**Register**
1	Clear display screen
2	Return home
4	Decrement cursor (shift cursor to left)
6	Increment cursor (shift cursor to right)
5	Shift display right
7	Shift display left
8	Display off, cursor off
A	Display off, cursor on
C	Display on, cursor off
E	Display on, cursor blinking
F	Display on, cursor blinking
10	Shift cursor position to left
14	Shift cursor position to right
18	Shift the entire display to the left
1C	Shift the entire display to the right
80	Force cursor to beginning of 1st line
C0	Force cursor to beginning of 2nd line
38	2 line and 5x 7 matrix

Fig. 2.19:- LCD commands

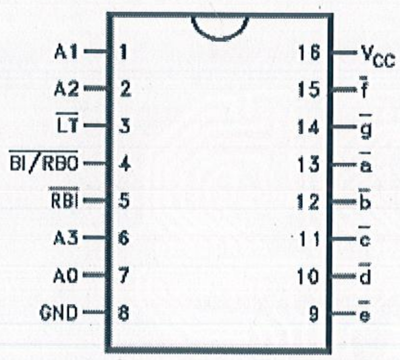
2.10 SEVEN SEGMENT

The DM74LS47 accepts four lines of BCD (8421) input data, generates their complements internally and decodes the data with seven AND/OR gates having open-collector outputs to drive indicator segments directly. Each segment output is guaranteed to sink 24 mA in the ON (LOW) state and withstand 15V in the OFF (HIGH) state with a maximum leakage current of 250 mA. Auxiliary inputs provided blanking, lamp test and cascadable zero-suppression functions. Its pin and logic diagram is shown in Fig. 2.20

(A) FEATURES

- Open-collector outputs
- Drive indicator segments directly
- Cascadable zero-suppression capability
- Lamp test input

(B) CONNECTION DIAGRAM



(C) LOGIC SYMBOL

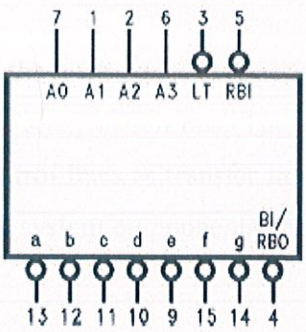


FIG. 2.20:- SEVEN SEGMENT PIN & LOGIC DIAGRAM

(D) PCB DESIGN

Liquid Single sided PCB design layout of Seven segment is shown in Fig. 2.21 below

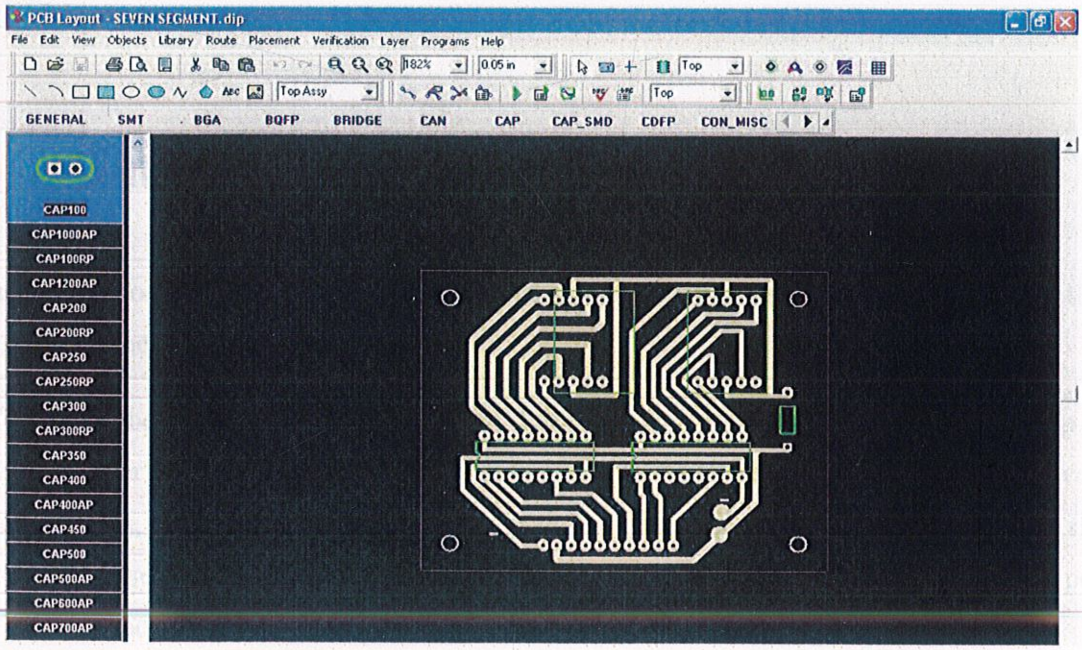


FIG. 2.21:- SEVEN SEGMENT PCB LAYOUT

2.11 PARALLEL PORT

(A) INTRODUCTION

The parallel port's hardware includes the back-panel connector and the circuits and cabling between the connector and the system's expansion bus. The PC's microprocessor uses the expansion bus's data, address, and control lines to transfer information between the parallel port and the CPU, memory, and other system components. A sample picture of DB-25 pin connector is shown in Fig. 2.22

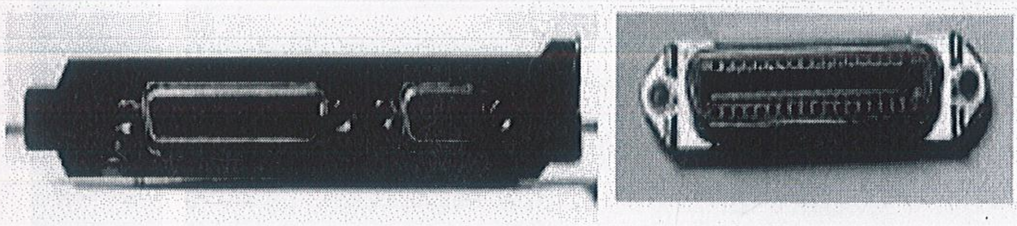


FIG. 2.22:- DB-25 PIN CONNECTOR

(B) INSIDE CIRCUITS

Inside the computer, the parallel-port circuits may be on the motherboard or on a card that plugs into the expansion bus.

The motherboard is the main circuit board that holds the computer's microprocessor chip as well as other circuits and slots for expansion cards. Because just about all computers have a parallel port, the port circuits are often right on the motherboard, freeing the expansion slot for other uses. Notebook and laptop computers don't have expansion slots, so the port circuits in these computers must reside on the system's main circuit board.

The port circuits connect to address, data, and control lines on the expansion bus, and these in turn interface to the microprocessor and other system.

(C)PCB DESIGN

Single sided PCB design layout of parallel port connector is shown in Fig. 2.3 below

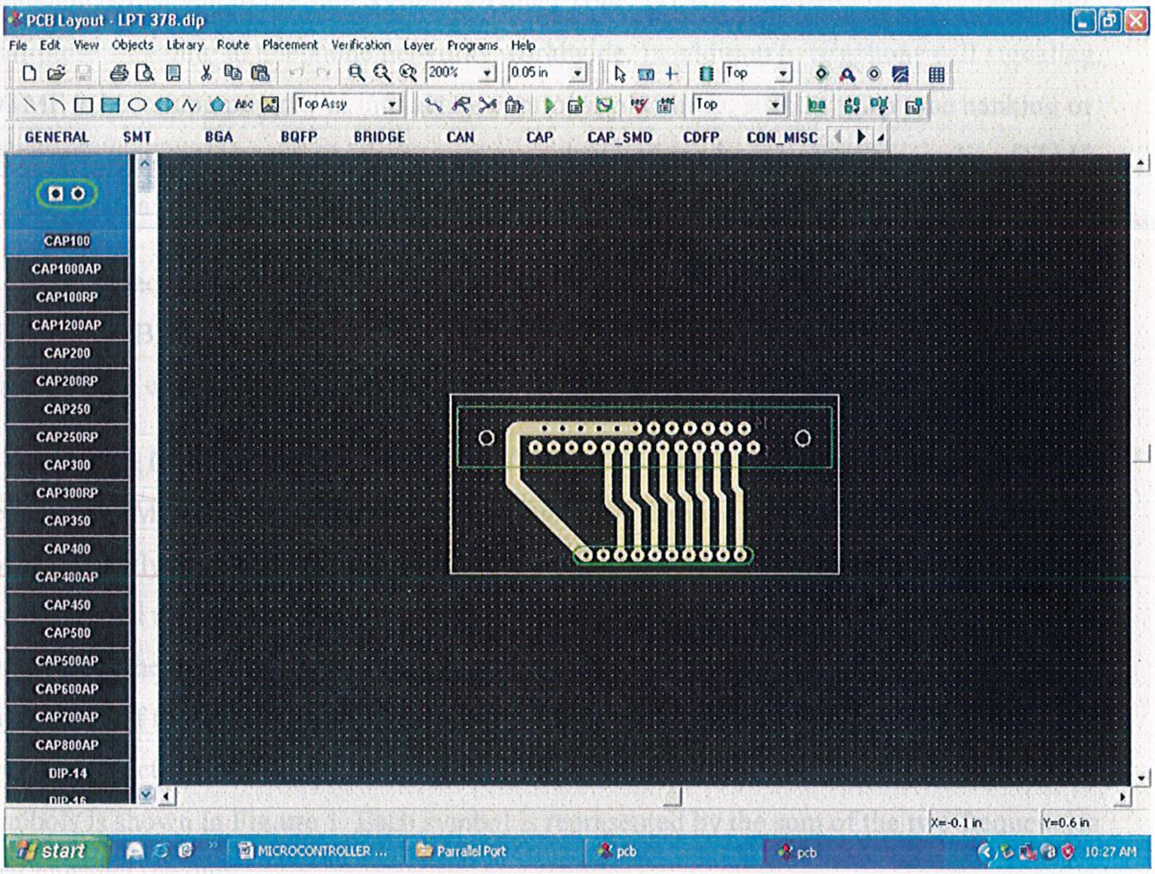


FIG. 2.23:- DB-25 PCB LAYOUT

2.12 DTMF

(A) DESCRIPTION

Dual-Tone Multi-Frequency (DTMF) is the generic name for pushbutton telephone signaling equivalent to the Bell System's Touch Tone. DTMF signaling is quickly replacing dial-pulse signaling in telephone networks worldwide. In addition to telephone call signaling, DTMF is becoming popular in interactive control applications, such as telephone banking or electronic mail systems, in which the user can select options from a menu by sending DTMF signals from a telephone.

This application note describes a DTMF coding and decoding implementation based on Hypersignal Block Diagram. The DTMF coding is based on two tones used to generate a digit. Two of eight tones can be combined so as to generate sixteen different DTMF digits.

In a DTMF signal generation, a DTMF keypad could be used for digit entry, the resultant DTMF tones are generated mathematically and added together. The values are logarithmically compressed and passed to the receiver. In a DTMF scheme, pairs of tones are used to signal the digits 0 through 9, pound(#), star(*), and the digits A, B, C and D. For each pair, one of the tones is selected from a low group of four frequencies, and the other from a high group of four frequencies. The correct detection of a digit requires both a valid tone pair and the correct timing intervals. The matrix of frequencies used to encode the 16 DTMF symbols is shown in Fig.ure 1. Each symbol is represented by the sum of the two frequencies that intersect the digit.

The row frequencies are in a low band, below 1 kHz, and the column frequencies are in a high band, between 1 kHz and 2 kHz. The digits are displayed as they would appear on a telephone's 4x4 matrix keypad (on standard telephone sets, the fourth column is omitted). The user should note that there are a number of different algorithms possible for generation and detection of DTMF tones

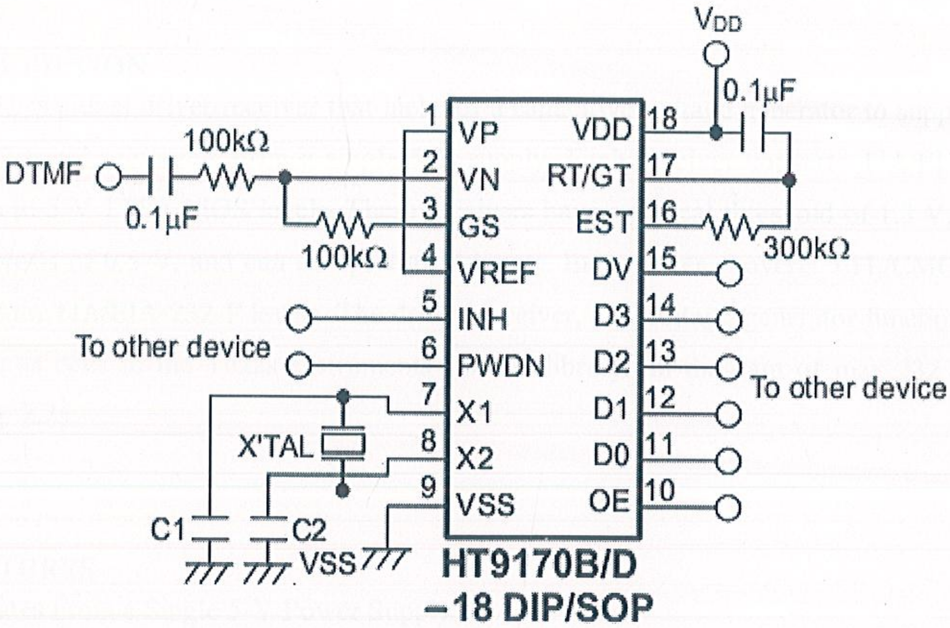
	HIGH GROUP FREQUENCY				
LOW GROUOP FREQUENCY		1209 HZ	1336 HZ	1477 HZ	1633 HZ
	697 HZ	1	2	3	A
	770HZ	4	5	6	B
	852 HZ	7	8	9	C
	941 HZ	*	0	#	D

FIG. 2.24:- HCT541 PIN DIAGRAM

When you press the buttons on the keypad, a connection is made that generates two tones at the same time. A "Row" tone and a "Column" tone. These two tones identify the key you pressed to any equipment you are controlling. If the keypad is on your phone, the telephone company's "Central Office" equipment knows what numbers you are dialing by these tones, and will switch your call accordingly. If you are using a DTMF keypad to remotely control equipment, the tones can identify what unit you want to control, as well as which unique function you want it to perform.

When you press the digit 1 on the keypad, you generate the tones 1209 Hz and 697 Hz. Pressing the digit 2 will generate the tones 1336 Hz and 697 Hz. Sure, the tone 697 is the same for both digits, but it take two tones to make a digit and the equipment knows the difference between the 1209 Hz that would complete the digit 1, and a 1336 Hz that completes a digit 2.

(B) CIRCUIT DIAGRAM



2.13 SERIAL PORT

(A) DESCRIPTION

The MAX232 is a dual driver/receiver that includes a capacitive voltage generator to supply TIA/EIA-232-F voltage levels from a single 5-V supply. Each receiver converts TIA/EIA-232-F inputs to 5-V TTL/CMOS levels. These receivers have a typical threshold of 1.3 V, a typical hysteresis of 0.5 V, and can accept ±30-V inputs. Each driver converts TTL/CMOS input levels into TIA/EIA-232-F levels. The driver, receiver, and voltage-generator functions are available as cells in the Texas Instruments Linasic library.Pin diagram of max 232 is shown in Fig. 2.25

(B) FEATURES

- Operates From a Single 5-V Power Supply
- With 1.0-_F Charge-Pump Capacitors
- Operates Up To 120 kbit/s
- Two Drivers and Two Receivers
- ±30-V Input Levels
- Low Supply Current . . . 8 mA Typical
-

(C) PIN DESCRIPTION

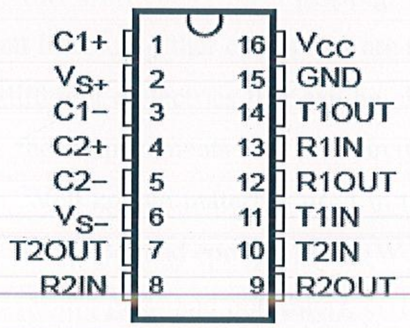


FIG. 2.25:- MAX232 PIN DIAGRAM

CHAPTER-3 PRINTED CIRCUIT BOARD

3.1 INTRODUCTION

A printed circuit board, or PCB, is used to mechanically support and electrically connect electronic components using conductive pathways, or traces, etched from copper sheets laminated onto a non-conductive substrate. It is also referred to as printed wiring board (PWB) or etched wiring board. A PCB populated with electronic components is a printed circuit assembly (PCA), also known as a printed circuit board assembly (PCBA).

PCBs are rugged, inexpensive, and can be highly reliable. They require much more layout effort and higher initial cost than either wire-wrapped or point-to-point constructed circuits, but are much cheaper and faster for high-volume production. Much of the electronics industry's PCB design, assembly, and quality control needs are set by standards that are published by the IPC organization.

3.2 MANUFACTURING

(A) MATERIALS

Conducting layers are typically made of thin copper foil. Insulating layers dielectric are typically laminated together with epoxy resin prepreg. The board is typically coated with a solder mask that is green in color. Other colors that are normally available are blue, and red. There are quite a few different dielectrics that can be chosen to provide different insulating values depending on the requirements of the circuit. Some of these dielectrics are polytetrafluoroethylene. Well known materials used in the PCB industry are FR-2 (Phenolic cotton paper), FR-3 (Cotton paper and epoxy), FR-4 (Woven glass and epoxy), FR-5 (Woven glass and epoxy), FR-6 (Matte glass and polyester), G-10 (Woven glass and epoxy), CEM-1 (Cotton paper and epoxy), CEM-2 (Cotton paper and epoxy), CEM-3 (Woven glass and epoxy), CEM-4 (Woven glass and epoxy), CEM-5 (Woven glass and polyester).

A PCB as a design on a computer (left) and realized as a board assembly with populated components (right). The board is double sided, with through-hole plating, green solder resist, and white silkscreen printing. Both surface mount and through-hole components have been used.

Typical density of a raw PCB (an average amount of traces, holes, and via's, with no components) is 2.15g / cm3

(B) PATTERNING (ETCHING)

The vast majority of printed circuit boards are made by bonding a layer of copper over the entire substrate, sometimes on both sides, (creating a "blank PCB") then removing unwanted copper after applying a temporary mask (eg. by etching), leaving only the desired copper traces. A few PCBs are made by adding traces to the bare substrate (or a substrate with a very thin layer of copper) usually by a complex process of multiple electroplating steps.

There are three common "subtractive" methods (methods that remove copper) used for the production of printed circuit boards:

1. Silk screen printing uses etch-resistant inks to protect the copper foil. Subsequent etching removes the unwanted copper. Alternatively, the ink may be conductive, printed on a blank (non-conductive) board. The latter technique is also used in the manufacture of hybrid circuits.
2. Photoengraving uses a photo mask and chemical etching to remove the copper foil from the substrate. The photo mask is usually prepared with a photo plotter from data produced by a technician using CAM, or computer-aided manufacturing software. Laser-printed transparencies are typically employed for photo tools; however, direct laser imaging techniques are being employed to replace photo tools for high-resolution requirements.
3. PCB milling uses a two or three-axis mechanical milling system to mill away the copper foil from the substrate. A PCB milling machine (referred to as a 'PCB Prototype') operates in a similar way to a plotter, receiving commands from the host software that control the position of the milling head in the x, y, and (if relevant) z

axis. Data to drive the Prototyper is extracted from files generated in PCB design software and stored in HPGL or Gerber file format.

"Additive" processes also exist. The most common is the "semi-additive" process. In this version, the unpatterned board has a thin layer of copper already on it. A reverse mask is then applied. (Unlike a subtractive process mask, this mask exposes those parts of the substrate that will eventually become the traces.) Additional copper is then plated onto the board in the unmasked areas; copper may be plated to any desired weight. Tin-lead or other surface platings are then applied. The mask is stripped away and a brief etching step removes the now-exposed original copper laminate from the board, isolating the individual traces.

The additive process is commonly used for multi-layer boards as it facilitates the plating-through of the holes (to produce conductive via) in the circuit board.

(C) LAMINATION

Some PCBs have trace layers inside the PCB and are called multi-layer PCBs. These are formed by bonding together separately etched thin boards.

(D) DRILLING

Holes through a PCB are typically drilled with tiny drill bits made of solid tungsten carbide. The drilling is performed by automated drilling machines with placement controlled by a drill tape or drill file. These computer-generated files are also called numerically controlled drill (NCD) files or "Excellon files". The drill file describes the location and size of each drilled hole. These holes are often filled with annular rings to create via. Vias allow the electrical and thermal connection of conductors on opposite sides of the PCB.

When very small vias are required, drilling with mechanical bits is costly because of high rates of wear and breakage. In this case, the vias may be evaporated by lasers. Laser-drilled vias typically have an inferior surface finish inside the hole. These holes are called micro vias.

It is also possible with controlled-depth drilling, laser drilling, or by pre-drilling the individual sheets of the PCB before lamination, to produce holes that connect only some of the copper layers, rather than passing through the entire board. These holes are called blind vias when they connect an internal copper layer to an outer layer, or buried vias when they connect two or more internal copper layers and no outer layers.

The walls of the holes, for boards with 2 or more layers, are plated with copper to form plated-through holes that electrically connect the conducting layers of the PCB. For multilayer boards, those with 4 layers or more, drilling typically produces a smear comprised of the bonding agent in the laminate system. Before the holes can be plated through, this smear must be removed by a chemical de-smear process, or by plasma-etch.

(E) EXPOSED CONDUCTOR PLATING AND COATING

The places to which components will be mounted are typically plated, because bare copper oxidizes quickly, and therefore is not readily solder able. Traditionally, any exposed copper was plated with solder by hot air solder levelling (HASL). This solder was a tin-lead alloy, however new solder compounds are now used to achieve compliance with the RoHS directive in the EU, which restricts the use of lead. One of these lead-free compounds is SN100CL, made up of 99.3% tin, 0.7% copper, 0.05% nickel, and a nominal of 60ppm germanium.

Other platings used are OSP (organic surface protectant), immersion silver (IAg), immersion tin, electroless nickel with immersion gold coating (ENIG), and direct gold. Edge connectors, placed along one edge of some boards, are often gold plated.

Electrochemical migration (ECM) is the growth of conductive metal filaments on or in a printed circuit board (PCB) under the influence of a DC voltage bias.

(F) SOLDER RESIST

Areas that should not be soldered to may be covered with a polymer solder resist (solder mask) coating. The solder resist prevents solder from bridging between conductors and thereby creating short circuits. Solder resist also provides some protection from the environment

(G)SCREEN PRINTING

Line art and text may be printed onto the outer surfaces of a PCB by screen printing. When space permits, the screen print text can indicate component designators, switch setting requirements, test points, and other features helpful in assembling, testing, and servicing the circuit board.

Screen print is also known as the silk screen, or, in one sided PCBs, the red print.

Lately some digital printing solutions have been developed to substitute the traditional screen printing process. This technology allows printing variable data onto the PCB, including serialization and barcode information for traceability purposes.

(H)PRINTED CIRCUIT ASSEMBLY

After the printed circuit board (PCB) is completed, electronic components must be attached to form a functional printed circuit assembly[3][4], or PCA (sometimes called a "printed circuit board assembly" PCBA). In through-hole construction, component leads are inserted in holes. In surface-mount construction, the components are placed on pads or lands on the outer surfaces of the PCB. In both kinds of construction, component leads are electrically and mechanically fixed to the board with a molten metal solder.

There are a variety of soldering techniques used to attach components to a PCB. High volume production is usually done with machine placement and bulk wave soldering or reflow ovens, but skilled technicians are able to solder very tiny parts (for instance 0201 packages which are 0.02" by 0.01") by hand under a microscope, using tweezers and a fine tip soldering iron for small volume prototypes. Some parts are impossible to solder by hand, such as ball grid array (BGA) packages.

Often, through-hole and surface-mount construction must be combined in a single PCA because some required components are available only in surface-mount packages, while others are available only in through-hole packages. Another reason to use both methods is that through-hole mounting can provide needed strength for components likely to endure physical stress, while components that are expected to go untouched will take up less space using surface-mount techniques.

After the board has been populated it may be tested in a variety of ways:

- While the power is off, visual inspection, automated optical inspection. JEDEC guidelines for PCB component placement, soldering, and inspection are commonly used to maintain quality control in this stage of PCB manufacturing.
- While the power is off, analog signature analysis, power-off testing.
- While the power is on, in-circuit test, where physical measurements (i.e. voltage, frequency) can be done.
- While the power is on, functional test, just checking if the PCB does what it had been designed for.

To facilitate these tests, PCBs may be designed with extra pads to make temporary connections. Sometimes these pads must be isolated with resistors. The in-circuit test may also exercise boundary scan test features of some components. In-circuit test systems may also be used to program nonvolatile memory components on the board.

CHAPTER- 4 CODING AND TESTING

4.1 ADC TESTING CODE

```
; --------------------------ADC TESTING----------------------------------------------------------
; lcd connected at  port 1
; adc connected at  port 3
org 0000h
; -------------------------LCD INTERFACING---------------------------------
mov a,#38h
acall command        ;sub routine
mov a,#38h          ;initialise two line 5x7 matrix
acall command        ;sub routine
mov a,#0ch          ;display on,cursor blinking
acall command        ;sub routine
mov a,#01h          ;clear lcd
acall command        ;sub routine
mov a,#80h          ;shift cursor TO 1st line
acall command                  ;
; ------------------------RESISTANCE MEASUREMENT----------------------------
mov a,#'A'
acall data1
mov a,#'D'
acall data1
mov a,#'C'
acall data1
; --------------------------ADC-----------------------------------------------------
mov p0,#0ffh
  go:
```

```
    setb p2.5
    clr p2.7          ; INTR=p2.6  ; start conversion
    setb p2.7          ; WR = p2.7
                  ; RD = p2.5 active low
    hee:jb p2.6,hee
          acall delay2
    clr p2.5
    mov a,p0                ; a contain temp in hex
; ---------------------hex to BCD conversion-----------------------------------
Lop: cjne a, #30d, next                         ; if DESTINATION smaller carry=0
          sjmp nex
    next: jnc gom
      setb p2.0
          setb p2.0
          sjmp hoi
gom: clr p2.0
hoi:
    mov b,#10d
          div ab
          mov r6,b                    ; 0ne
          mov b,#10d
          div ab
          mov r7,b          ; tens
          mov r2,a                    ;hundred

    mov a,#89h              ;shift cursor TO 1st line
    acall command           ;command subroutine
          mov a, r2
          orl a,#30h
    acall data1
          mov a, r7
```

```
        orl a,#30h
    acall data1
        mov a, r6
        orl a,#30h
    acall data1
        mov a,#20h
    acall data1
        sjmp go
;-----------------------------delay1---------------------------------------------------------
delay1:
   mov r3,#150d
h130: mov r4,#150d
h230: djnz r4,h230
   djnz r3,h130
   ret
;-------------------------delay2-----------------------------------
delay2:
   mov r3,#255d
h1300: mov r4,#255d
h2300: djnz r4,h2300
   djnz r3,h1300
       ret
;-----------------------------------delay-----------------

delay:
   mov r3,#60d
h13: mov r4,#40d
h23: djnz r4,h23
   djnz r3,h13
   ret
; ----------------------------command--------------------------
```

```
command:
mov p1,a
clr p3.2
clr p3.1
setb p3.0
clr p3.0
acall delay1
ret
; ----------------------------data1----------------------------------------------------
data1:
mov p1,a
setb p3.2
clr p3.1
setb p3.0
clr p3.0
acall delay1
ret
END
```

4.2 KEYPAD & RELAY TESTING CODE

DOUBBLE PASSWORD CONTROL DEVCICE COONTROL USING EXTERNAL SWITCHES AND KEYPAD

```
; Double password lock with logout
;-------------------------------------------------------------------------------
; keypad connected to port p3
;LCD data pins to P0
;-------------------------------------------------------------------------------

;relay   AT P1.0
;buzzer at         p1.5
```

```
;logout switch at p2.0
;-------------------------------------------------------------------------------------


;      DOUBLE PASSWORD CONTROL DEVICE
;    ENTER PASSWORD
;                                                     **********
;    LOCK OPEN                                        WRONG PASSWORD
;   SELECT DEVICE                                        RETRY
;  1.FAN 2.LIGHT
;  PRESS 2.0 tO 2.4

;------------------------------------------------------------------------------------

org 0000h
regain :
clr p1.0
buzzer EQU p1.5
clr buzzer

mov a,#38h          ;initialise two line 5x7 matrix
acall command         ;sub routine
mov a,#38h          ;initialise two line 5x7 matrix
acall command        ;sub routine
mov a,#0eh          ;display off,cursor o  ff
acall command        ;sub routine
mov a,#01h           ;clear lcd
acall command        ;sub routine
mov a,#6h            ;increment cursor right
acall command        ;command subroutine
mov a,#80h           ;shift cursor TO 1st line
```

```
acall command          ;command subroutine

;--------------------------------------------------------------------------------

mov a, #'D'
acall data1            ; subroutine
mov a, #'O'
acall data1            ; subroutine
mov a, #'U'
acall data1            ; subroutine
mov a, #'B'
acall data1            ; subroutine
mov a, #'L'
acall data1            ; subroutine
mov a, #'E'
acall data1
mov a, #20h
acall data1
mov a, #'P'
acall data1            ; subroutine
mov a, #'A'
acall data1            ; subroutine
mov a, #'S'
acall data1            ; subroutine
mov a, #'S'
acall data1            ; subroutine
mov a, #'W'
acall data1            ; subroutine
mov a, #'O'
acall data1            ; subroutine
mov a, #'R'
```

```
acall data1          ; subroutine
mov a, #'D'
acall data1          ; subroutine
mov a,#0c0h           ;shift cursor TO 1st line
acall command        ;command subroutine
mov a, #'C'
acall data1          ; subroutine
mov a, #'O'
acall data1          ; subroutine
mov a, #'N'
acall data1          ; subroutine
mov a, #'T'
acall data1          ; subroutine
mov a, #'R'
acall data1          ; subroutine
mov a, #'O'
acall data1          ; subroutine
mov a, #'L'
acall data1          ; subroutine
mov a, #20h
acall data1
mov a, #'L'
acall data1          ; subroutine
mov a, #'O'
acall data1          ; subroutine
mov a, #'C'
acall data1          ; subroutine
mov a, #'K'
acall data1          ; subroutine
mov a, #'.'
acall data1          ; subroutine
```

```
acall delay2
mov a, #'.'
acall data1          ; subroutine
acall delay2
mov a, #'.'
acall data1          ; subroutine
acall delay2

;-------------------------------------------------------------------------------------------

ag:mov a,#01h           ;clear lcd
acall command        ;sub routine
mov a,#80h           ;shift cursor TO 1st line
acall command        ;command subroutine
mov a, #'E'
acall data1          ; subroutine
mov a, #'n'
acall data1          ; subroutine
mov a, #'t'
acall data1          ; subroutine
mov a, #'e'
acall data1          ; subroutine
mov a, #'r'
acall data1          ; subroutine
mov a,#86h           ;shift cursor TO 1st line
acall command        ;command subroutine
mov a, #'P'
acall data1          ; subroutine
mov a, #'a'
acall data1          ; subroutine
mov a, #'s'
```

```
acall data1            ; subroutine
mov a, #'s'
acall data1            ; subroutine
mov a, #'w'
acall data1            ; subroutine
mov a, #'o'
acall data1            ; subroutine
mov a, #'r'
acall data1            ; subroutine
mov a, #'d'
acall data1            ; subroutine
mov a, #':'
acall data1            ; subroutine
mov a, #'-'
acall data1            ; subroutine
mov a,#0c0h              ;shift cursor TO 2ND line
acall command          ;command subroutine
;--------------------------STORAGE OF PASSWORD----------------------------------

mov 30h,#'1'  ; stored password at ram location 30h onward
mov 31h,#'2'  ;
mov 32h,#'3'
mov 33h,#'4'
mov 34h,#'5'
mov 35h,#'6'
mov 36h,#'7'
mov 37h,#'8'
mov 38h,#'9
mov r0,#30h    ;--------------------datda pointer ---------------------------------
mov r1,#50h
;----------------------------KEYPAD SCANNING-----------------------------------------
```

```
    mov p3,#00001111b           ; high all colloum   row|coloum
k1:  mov p3,#00001111b            ; groung all row
    mov a,p3              ; read collum

    cjne a,#00001111b,k1   ; check till all keys are released
k2:  acall delay        ;
    mov a,p3
    cjne a,#00001111b,over
    sjmp k2
over: acall delay  ;
      acall delay
      mov a,p3

    cjne a,#00001111b,over1  ; key pressed find row
     sjmp k2

over1:mov p3,#11111111b
     clr p3.7      ; groung row 0
     mov a,p3
     anl a,#00001111b
     cjne a,#00001111b,row_0 ; key row 0, find coloum


     mov p3,#11111111b
     clr  p3.6      ; groung row 1
     mov a,p3
     anl a,#00001111b
     cjne a,#00001111b,row_1 ; key row 1, find coloum

     mov p3,#11111111b
     clr p3.5      ; groung row 2
```

```
      mov a,p3
      anl a,#00001111b
      cjne a,#00001111b,row_2 ; key row 2, find coloum

      mov p3,#11111111b
      clr p3.4      ; groung row 3
      mov a,p3
      anl a,#00001111b
      cjne a,#00001111b,row_3 ; key row 3, find coloum2
      ljmp k2

row_0: mov dptr ,#kcode0
      sjmp find

row_1:mov dptr ,#kcode1
      sjmp find

row_2:mov dptr ,#kcode2
      sjmp find

row_3:mov dptr ,#kcode3
      sjmp find

find: rrc a
      jnc match
      inc dptr
      sjmp find

match: clr a
      movc a,@a+dptr
```

```
        cjne a,#75h,yo
        sjmp dd

yo:   cjne a,#76h,go
        lcall del
        sjmp ho

go:   mov @r1,a                         ; STORE DATA IN LOCATION 50H LINE
        inc r1
              ;mov a,#'*'                        ;display on screen '*' password
        acall data1
        acall delay1
ho:   ljmp k1

dd:clr a

; ----------------------------COUNTER SET FOR MATCHING OF CODE ------------
mov r6,#8d
mov r0,#30h
mov r1,#50h

; --------------------------------MATCHING STARTED----------------------------------------

loop: mov a,@r1
cpl a
add a,@r0
cjne a,#0ffh,wrong
inc r0
inc r1
djnz r6,loop
; ----------------------------------CORRECT PASSWORD--------------------------------------
```

```
; -------------------------------------------------------lock open-------------------------------------------
-----
mov a,#01h          ;clear lcd
acall command       ;sub routine
mov a,#80h          ;clear lcd
acall command       ;sub routine
clr buzzer
mov a, #'C'
acall data1         ; subroutine
mov a, #'o'
acall data1         ; subroutine
mov a, #'d'
acall data1         ; subroutine
mov a, #'e'
acall data1         ; subroutine

mov a,#85h          ;clear lcd
acall command       ;sub routine

mov a, #'M'
acall data1         ; subroutine
mov a, #'a'
acall data1         ; subroutine
mov a, #'t'
acall data1         ; subroutine
mov a, #'c'
acall data1         ; subroutine
mov a, #'h'
acall data1         ; subroutine
mov a, #'e'
acall data1
```

```
mov a, #'d'
acall data1
mov a, #'.'
acall data1            ; subroutine
acall delay2
mov a, #'.'
acall data1            ; subroutine
acall delay2
mov a, #'.'
acall data1            ; subroutine
;---------------------------------------------------------------------------
mov 50h,#'0'  ; stored password at ram location 50h onward
mov 51h,#'0'  ;
mov 52h,#'0'
mov 53h,#'0'
mov 54h,#'0'
mov 55h,#'0'
mov 56h,#'0'
mov 57h,#'0'
mov 58h,#'0'
;---------------------------------------------------------------------------
clr buzzer
ljmp exit
;--------------------WRONG PASSWORD-----------------------------
wrong:
mov a,#80h
acall command
mov a,#1h
acall command
setb buzzer
mov a, #'I'
```

```
acall data1             ; subroutine
mov a, #'n'
acall data1             ; subroutine
mov a, #'v'
acall data1             ; subroutine
mov a, #'a'
acall data1             ; subroutine
mov a, #'l'
acall data1             ; subroutine
setb buzzer
mov a, #'i'
acall data1             ; subroutine
mov a, #'d'
acall data1             ; subroutine

clr buzzer
mov a,#88h
acall command
mov a, #'C'
acall data1             ; subroutine
mov a, #'o'
acall data1             ; subroutine
mov a, #'d'
acall data1             ; subroutine
mov a, #'e'
acall data1
mov a, #'!'
acall data1
setb buzzer
mov a,#0c5h
acall command
```

```
mov a, #'R'
acall data1              ; subroutine
mov a, #'e'
acall data1              ; subroutine
mov a, #'t'
acall data1              ; subroutine
mov a, #'r'
acall data1
mov a, #'y'
acall data1
acall delay2
acall delay2
clr buzzer
ljmp ag
;-----------------------------------------------------------------------------------------------------------------
---
org 300h

kcode0: db '0','1','2','3'

kcode1: db '4','5','6','7'

kcode2: db '8','9','a','b'

kcode3: db 'c','d',75h,76h
;-----------------------------------------------------------------------------------------------------------------
-
delay1:
    mov r4,#120d
h11:  mov r5,#100d
h21:   djnz r5,h21
```

```
djnz r4,h11
ret
;-----------------------------------------------------------------------------------------------------------
-----command:
mov p0,a
clr p2.5
clr p2.6
setb p2.7
clr p2.7
acall delay1
ret
;-----------------------------------------------------------------------------------------------------------
---
data1:
mov p0,a
setb p2.5
clr p2.6
setb p2.7
clr p2.7
acall delay1
ret
;-----------------------------------------------------------------------------------------------------------
----
del:
    mov a,#10h          ;shift cursor left
    acall command       ;command subroutine
    mov a,#20h          ;shift cursor left
    acall data1         ;command subroutin
    mov a,#10h          ;shift cursor left
    acall command       ;command subroutine
    dec r1
```

```
    ret
;----------------------------------------------------------------------------------------------------------------------
--
delay2:
     mov r4,#15d
h3:   mov r5,#255d
h1:   mov r3,#255d
h2:   djnz r3,h2
     djnz r5,h1
     djnz r4,h3
     ret
;----------------------------------------------------------------------------------------------------------------------
----
delay:
mov r5,#30d
h13:  mov r4,#45d
h23:   djnz r4,h23
     djnz r5,h13
     ret
;----------------------------------------------------------------------------------------------------------------------
-
close:
clr p1.0
clr p1.0
clr buzzer
mov a,#01h             ;clear lcd
acall command          ;sub routine
mov a,#80h             ;clear lcd
acall command
setb buzzer
mov a, #'S'
```

```
acall data1        ; subroutine
mov a, #'y'
acall data1        ; subroutine
mov a, #'s'
acall data1        ; subroutine
mov a, #'t'
acall data1        ; subroutine
mov a, #'e'
clr buzzer
acall data1        ; subroutine
mov a, #'m'
acall data1

mov a,#87h          ;clear lcd
acall command
setb buzzer
mov a, #'L'
acall data1        ; subroutine
mov a, #'o'
acall data1        ; subroutine
mov a, #'c'
acall data1        ; subroutine
mov a, #'k'
acall data1        ; subroutine
mov a, #'e'
acall data1
setb buzzer
mov a, #'d'
acall data1
mov a,#0c0h         ;clear lcd
acall command
```

```
mov a, #'B'
acall data1            ; subroutine
mov a, #'y'
acall data1            ; subroutine
mov a, #'e'
acall data1
mov a,#0c4h             ;clear lcd
acall command
mov a, #'B'
acall data1            ; subroutine
mov a, #'y'
acall data1            ; subroutine
mov a, #'e'
acall data1
mov a, #'.'
acall data1            ; subroutine
mov a, #'.'
acall data1            ; subroutine
mov a, #'.'
acall data1
setb buzzer
acall delay2
acall delay2
acall delay2
  clr buzzer
ljmp regain

ret
;------------------------------------slect device to contorl-----------------------------
exit:
ag5:
```

```
mov a,#01h          ;clear lcd
acall command       ;sub routine
mov a,#80h          ;clear lcd
acall command       ;sub routine
mov a, #'E'
acall data1         ; subroutine
mov a, #'N'
acall data1         ; subroutine
mov a, #'T'
acall data1         ; subroutine
mov a, #'E'
acall data1         ; subroutine
mov a, #'R'
acall data1         ; subroutine
mov a,#87h          ;clear lcd
acall command       ;sub routine
mov a, #'C'
acall data1         ; subroutine
mov a, #'O'
acall data1         ; subroutine
mov a, #'D'
acall data1         ; subroutine
mov a, #'E'
acall data1         ; subroutine
mov a, #'-'
acall data1         ; subroutine
mov a, #'2'
acall data1         ; subroutine
clr buzzer
mov a,#0c0h         ;clear lcd
acall command       ;sub routine
```

```
;-----------------------------STORAGE OF PASSWORD----------------------------------

mov 40h,#'9'  ; stored password at ram location 40h onward
mov 41h,#'8'  ;
mov 42h,#'1'
mov 43h,#'6'
mov 44h,#'0'
mov 45h,#'5'
mov 46h,#'6'
mov 47h,#'5'
mov 48h,#'8'
mov r0,#40h    ;---------------------data pointer----------------------------------
mov r1,#60h    ;
;---------------------------------------------------------------------------------
; ----------------------------KEYPAD SCANNING-------------------------------------------------
--
   mov p3,#00001111b          ; high all colloum   row|coloum
k11:  mov p3,#00001111b          ; groung all row
   mov a,p3                ; read collum

   cjne a,#00001111b,k11   ; check till all keys are released
k21:  acall delay            ; 20msec
   mov a,p3
   cjne a,#00001111b,overP
   sjmp k21
overP:
   acall delay          ;debounce time
   acall delay
   mov a,p3
   cjne a,#00001111b,over11  ; key pressed find row
    sjmp k21
```

```
overl1:
      mov p3,#11111111b
      clr p3.7      ; groung row 0
      mov a,p3
      anl a,#00001111b
      cjne a,#00001111b,row_01 ; key row 0, find coloum
      mov p3,#11111111b
      clr  p3.6      ; groung row 1
      mov a,p3
      anl a,#00001111b
      cjne a,#00001111b,row_11 ; key row 1, find coloum
      mov p3,#11111111b
      clr p3.5      ; groung row 2
      mov a,p3
      anl a,#00001111b
      cjne a,#00001111b,row_21 ; key row 2, find coloum
      mov p3,#11111111b
      clr p3.4      ; groung row 3
      mov a,p3
      anl a,#00001111b
      cjne a,#00001111b,row_31 ; key row 3, find coloum2
      ljmp k21
      row_01:mov dptr ,#kcode0
      sjmp find1
row_11:mov dptr ,#kcode1
     sjmp find1
row_21:mov dptr ,#kcode2
      sjmp find1
row_31:mov dptr ,#kcode3
      sjmp find1
find1: rrc a
```

```
        jnc match1
        inc dptr
        sjmp find1
match1: clr a
        movc a,@a+dptr
        cjne a,#75h,yo1
        sjmp dd1
yo1:    cjne a,#76h,go1
        lcall del
        sjmp ho1
go1:    mov @r1,a
        inc r1
            mov a,#'*'                  ;display on screen '*' password
        acall data1
        acall delay1
ho1:    ljmp k11
dd1:clr a
;-----------------------------------------------------------------------------------------------------------
---
;----------------------------COUNTER SET FOR MATCHING OF CODE ----------------------
mov r6,#8d
mov r0,#40h
mov r1,#60h
;-------------------------------MATCHING STARTED------------------------------------------------
--
loop1: mov a,@r1
cpl a
add a,@r0
cjne a,#0ffh,wrong1
inc r0
inc r1
```

```
djnz r6,loop1
; ----------------------------------CORRECT PASSWORD-------------------------------------------
; ----------------------------------------lock open-----------------------------------------------
mov a,#01h          ;clear lcd
acall command       ;sub routine
mov a,#80h          ;clear lcd
acall command       ;sub routine
clr buzzer
mov a, #'L'
acall data1         ; subroutine
mov a, #'o'
acall data1         ; subroutine
mov a, #'c'
acall data1         ; subroutine
mov a, #'k'
acall data1         ; subroutine
mov a, #'e'
acall data1         ; subroutine
mov a, #'r'
acall data1
mov a,#87h          ;clear lcd
acall command       ;sub routine
mov a, #'i'
acall data1         ; subroutine
mov a, #'s'
acall data1         ; subroutine
mov a,#8ah          ;clear lcd
acall command
mov a, #'o'
acall data1         ; subroutine
mov a, #'p'
```

```
acall data1             ; subroutine
mov a, #'e'
acall data1             ; subroutine
mov a, #'n'
acall data1
;------------------------------------------------------------------------
mov 60h,#'0'   ; stored password at ram location 30h onward
mov 61h,#'0'   ;
mov 62h,#'0'
mov 63h,#'0'
mov 64h,#'0'
mov 65h,#'0'
mov 66h,#'0'
mov 67h,#'0'
mov 68h,#'0'
clr buzzer
;------------------------------------------------------------------------
ljmp now
;--------------------WRONG PASSWORD------------------------------------------------------------
wrong1:
setb buzzer
mov a,#01h
acall command
mov a,#80h
acall command
mov a, #'B'
acall data1             ; subroutine
mov a, #'a'
acall data1             ; subroutine
mov a, #'d'
acall data1             ; subroutine
```

```
mov a,#84h
acall command
mov a, #'C'
acall data1            ; subroutine
mov a, #'o'
acall data1            ; subroutine
mov a, #'d'
acall data1            ; subroutine
mov a, #'e'
acall data1
mov a,#0c0h
acall command
mov a, #'R'
acall data1            ; subroutine
mov a, #'e'
acall data1            ; subroutine
mov a, #'t'
acall data1            ; subroutine
mov a, #'r'
acall data1
mov a, #'y'
acall data1
acall delay2
acall delay2
clr buzzer
ljmp ag5
;------------------------------------------------------------------
; SWITCHES CONTROL PANNEL              |
;-----------------------------select device-------------------------
now:
    setb  p1.0                 ; relay
```

```
mom1: Setb p2.0                ;log off
   jb p2.0,mom1
       ljmp close
       end
```

4.3 DTMF TESTING

```
; ---------------------------------------DTMF TESTING--------------------------------
; DTMF + PASSWOD
; LCD AT PORT1
; DTMF DATA PINS AT P2.0 TO P2.3
; DTMF VD PIN AT P2.4
;-----------------------------------------------------------------------------------------------
org 0000h
clr p0.1
clr p0.0
mov a,#38h
acall command
mov a,#38h
acall command
mov a,#0ch
acall command
mov a,#01h
acall command
mov a,#80h
acall command
;-----------------------------------------------
mov a,#'d'
acall data1
mov a,#'t'
acall data1
mov a,#'m'
acall data1
```

```
mov a,#'f'
acall data1
mov a,#20h
acall data1
mov a,#'t'
acall data1
mov a,#'e'
acall data1
mov a,#'s'
acall data1
mov a,#'t'
acall data1
mov a,#'i'
acall data1
mov a,#'n'
acall data1
mov a,#'g'
acall data1
mov a,#0c0h
acall command
mov a,#20h
acall data1
mov a,#'s'
acall data1
mov a,#'y'
acall data1
mov a,#'s'
acall data1
mov a,#'t'
acall data1
mov a,#'e'
```

```
acall data1
mov a,#'m'
acall data1
acall delay2
;------------------------------------------
ag:
clr p0.1
mov a,#01h
acall command
mov a,#80h
acall command
;----------------------------------------------------------
mov a, #'e'
acall data1
mov a, #'n'
acall data1
mov a, #'t'
acall data1
mov a, #'e'
acall data1
mov a, #'r'
acall data1
mov a,#86h
acall command
mov a, #'p'
acall data1
mov a, #'a'
acall data1
mov a, #'s'
acall data1
mov a, #'s'
```

```
acall data1
mov a, #'w'
acall data1
mov a, #'o'
acall data1
mov a, #'r'
acall data1
mov a, #'d'
acall data1
mov a, #':'
acall data1
mov a, #'-'
acall data1
mov a,#0c0h
acall command
;-----------------------------------------------------------------
mov 30h,#'9'
mov 31h,#'8'
mov 32h,#'1'
mov 33h,#'6'
mov 34h,#'0'
mov 35h,#'5'
mov 36h,#'6'
mov 37h,#'5'
mov 38h,#'8'
mov 39h,#'5'
mov r0,#30h
mov r1,#50h
;-----------------------------------------------------------------------
ho:clr p2.4
  jb p2.4,gop
```

```
sjmp ho
gop:
mov p2,#00h
mov a, p2
acall delay3
anl a ,#0fh
cjne a,#01h,next2
mov a,#'1'
lcall store
next2:
mov p2,#00h
mov a, p2
acall delay3
anl a ,#0fh
cjne a,#02h,next3
mov a,#'2'
lcall store
next3:
mov p2,#00h
mov a, p2
acall delay3
anl a ,#0fh
cjne a,#03h,next4
mov a,#'3'
lcall store
next4:
mov p2,#00h
mov a, p2
acall delay3
anl a ,#0fh
cjne a,#04h,next5
```

```
mov a,#'4'
lcall store
next5:
mov p2,#00h
mov a, p2
acall delay3
anl a ,#0fh
cjne a,#05h,next6
mov a,#'5'
lcall store
next6:
mov p2,#00h
mov a, p2
acall delay3
anl a ,#0fh
cjne a,#06h,next7
mov a,#'6'
lcall store
next7:
mov p2,#00h
mov a, p2
acall delay3
anl a ,#0fh
cjne a,#07h,next8
mov a,#'7'
lcall store
next8:
mov p2,#00h
mov a, p2
acall delay3
anl a ,#0fh
```

```
cjne a,#08h,next9
mov a,#'8'
lcall store
next9:
mov p2,#00h
mov a, p2
acall delay3
anl a ,#0fh
cjne a,#09h,next0
mov a,#'9'
lcall store
next0:
mov p2,#00h
mov a, p2
acall delay3
anl a ,#0fh
cjne a,#0ah,nextb
mov a,#'0'
lcall store
nextb:
mov p2,#00h
mov a, p2
acall delay3
anl a ,#0fh
cjne a,#0bh,nextc
mov a,#76h
lcall store
nextc:
mov p2,#00h
mov a, p2
acall delay3
```

```
anl a ,#0fh
cjne a,#0ch,hoo
mov a,#75h
lcall store
hoo:ljmp ho
dd:clr a
;------------------------------------------------------------------------------------
mov r6,#10d
mov r0,#30h
mov r1,#50h
loop: mov a,@r1
cpl a
add a,@r0
cjne a,#0ffh,wrong
inc r0
inc r1
djnz r6,loop
sjmp cont
wrong:ljmp invalid
;-------------------------------------------------------------------
cont:
mov a,#01h
acall command
mov a,#80h
acall command
mov a, #'c'
acall data1
mov a, #'o'
acall data1
mov a, #'d'
acall data1
```

```
mov a, #'e'
acall data1
mov a,#85h
acall command
mov a, #'m'
acall data1
mov a, #'a'
acall data1
mov a, #'t'
acall data1
mov a, #'c'
acall data1
mov a, #'h'
acall data1
mov a, #'e'
acall data1
mov a, #'d'
acall data1
acall delay3
acall delay2
;-----------------------------------------------------------------------------
mov 50h,#'0'
mov 51h,#'0'
mov 52h,#'0'
mov 53h,#'0'
mov 54h,#'0'
mov 55h,#'0'
mov 56h,#'0'
mov 57h,#'0'
mov 58h,#'0'
mov 59h,#'0'
```

```
;--------------------------------------------------------------------------------
invalid:
mov a,#01h
acall command
mov a,#80h
acall command
mov a, #'r'
acall data1
mov a, #'e'
acall data1
mov a, #'t'
acall data1
mov a, #'r'
acall data1
mov a, #'y'
acall data1
mov a, #'!'
acall data1
acall delay2
ljmp ag
;--------------------------------------------------------------------------------------------------------
-
store:
    cjne a,#75h,yo
    ljmp dd
yo:  cjne a,#76h,go
    lcall del
    ljmp ho
go:  mov @r1,a
    inc r1
    ;mov a,#'*'
```

```
      acall data1
      acall delay1
      ret
;---------------------------------------------------------------
delay1:
      mov r4,#120d
h11:  mov r5,#100d
h21:  djnz r5,h21
      djnz r4,h11
      ret
delay3:
      mov r4,#200d
h114: mov r5,#130d
h214: djnz r5,h214
      djnz r4,h114
      ret
command:
mov p1,a
clr p3.2
clr p3.1
setb p3.0
clr p3.0
acall delay1
ret
data1:
mov p1,a
setb p3.2
clr p3.1
setb p3.0
clr p3.0
acall delay1
```

```
ret
;----------------------------------------------------------------------
del:
      mov a,#10h
      acall command
      mov a,#20h
      acall data1
      mov a,#10h
      acall command
      dec r1
      ret
delay2:
      mov r4,#15d
h3:   mov r5,#255d
h1:   mov r3,#255d
h2:   djnz r3,h2
      djnz r5,h1
      djnz r4,h3
      ret
delay:
      mov r5,#30d
h13:  mov r4,#45d
h23:  djnz r4,h23
      djnz r5,h13
      ret
end
```

CHAPTER-5 RESULT, MODIFICATION AND FUTURE SCOPE

5.1 RESULT

After complete designing of our Multipurpose Microcontroller board we have made number of experiment on it. We have also designed coding for various project on the microcontroller kit. Various projects such as

(a) Double password based security system
(b) Mobile controlling of devices
(c) Analog to digital binary and hexadecimal conversion
And more projects can be implemented

We found that microcontroller kit was properly working and each peripheral interfaced is properly working, and many operations on this kit can be performed very easily without any difficulty.

5.2 MODIFICATION

Microcontroller kit we have designed can be further modified by designing more peripheral in same size of PCB, but such a process make PCB design more complicated and difficult for new user in the field of microcontroller, but this will be useful for advanced user.

Various other features that can be embed on board are:

1. RTC (real time clock)
2. DAC (digital to analog converter)
3. RF, FM, DTMF etc.
4. Optical fiber transmitter and receiver
5. AVR , PIC etc such microcontroller can be used

5.3 FUTURE SCOPE

Various future scope of this microcontroller kit are:-

1. Multipurpose Microcontroller Board serves the purpose of explaining the practicality of any microcontroller based system.
2. Useful for embedded application and robotics.
3. Helpful for beginner in field of embedded system and robotics.

REFERENCES

Websites

1. *433MHz RF modules , Online , 3 August 2008*
< **http://www.triindia.co.in** >

2. **Robokits** *World* **RF Modules** *433 MHz ASK [RKI-1064] - The RX ASK is an ASK Hybrid receiver* **module,** *Online, 15 August 2008*

< **http://www.robokitsindia.com** >

3. *Wireless Made Simple - RF module, antenna and connector solutions offer easily applied remote control functionality to your product, Online, 18 August 2008*
< **http://linxtechnology.com** >

4. **Sunrom** *Technologies Online , 28 August 2008*
< **http://www.sunrom.com** >

5. *Compact size & low cost robotics , Online , 30 August 2008*
< **http://www.summitek.com** >

6. *Datasheet search site,* **datasheets**, *Datasheet search site for Electronic Components and Semiconductors, integrated circuits, diodes and other semiconductors. Online, 31 August 2008*

< **http://www.alldatasheet.com** >

7. **How Stuff Works** *explains hundreds of subjects, from car engines to lock-picking to ESP, using clear language and tons of illustrations Online, 11 October 2008*
< **www.howstuffworks.com** >

Books

1. Author: Muhammad Ali Mazidi & Janice Gillispie Mazidi
The 8051 Microcontroller and Embedded system

Publisher: PEARSON, Prentice Hall, 1990

PUBLICATIONS

- An Embedded Platform for Intelligent Traffic Control, 2010 Fourth UKSim European Symposium on Computer Modeling and Simulation Pisa, Italy

BRIEF BIO DATA

JAPINDER SINGH CHAUDHRY- Department of Electronics and Communication Engineering, Jaypee University Of Information Technology, Waknaghat, Solan (H.P)

Pursuing	**Name Of School/University**	**Year**	**CGPA**
B.Tech (ECE)	Jaypee University of Information Technology, Solan (H.P.)	2011	7.6 (80%) (Up Till 7^{th}sem)
XII	St.Francis Convent School Bareilly	2006	78.6%
X	St.Francis Convent School Bareilly	2004	79.3%

Currently working on the MULTIPURPOSE MICROCONTROLLER BOARD