Mini Project
on

# HEXADECIMAL CALCULATOR

(Sem : VI, Third Year of Engineering)

BY

**KARAN SHAH** **60002120095**

**PROF. AARTI G. AMBEKAR**

**ELECTRONICS AND TELECOMMUNICATION**

**DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING**
**2014-2015**

# CERTIFICATE

This is to certify that the mini project entitled **“HEXADECIMAL CALCULATOR”** is a work of

**KARAN SHAH 60002120095**

submitted to the EXTC department, D.J. Sanghvi C.O.E., during sixth semester of the academic year 2014-2015,**for the“Third Year Of Engineering”** course.

PROF. AARTI G. AMBEKAR
GUIDE

(External Examiner)

PROF. AMIT DESHMUKH
(Head Of the Deparment)

## Acknowledgement

Any task requires effort from many people and this is no different. First and foremost we feel immense pleasure and heartfelt gratitude towards our project guide Ms. Aarti G. Ambekar for her constant inspiration, guidance, constructive criticism and painstaking efforts to help us during this study and would thank her for the same.

We would like to thank God for giving us physical and mental strength to get through the report. We would also like to thank H.O.D of Electronics and Telecommunication, Mr. Amit Deshmukh for allowing us to avail the facilities of the college for conducting our research.Last but not the least we would like to thank Mr.Tushar Sawant and Miss. Swarali Narvekar for their constant support. We also acknowledge the contribution of our staff members, both teaching and non-teaching, and our friends who have directly or indirectly helped us in compiling of our report.

Last but not the least we would like to thank our entire team whose effort made this report possible.

## Abstract

The basic idea of this report is to describe the components used for making a basic calculator. This report will have in understanding the process of interfacing of LCD and Keypad with the microcontrolller.

In this report we have discussed about in brief the circuit being implemented and the specifications of certain components. It will also cover in brief the softwares being used as well the process of Etching and PCB design.

After implementing the ciruit, we also discuss the ways to improve the output of the circuit.

## TABLE OF CONTENTS

Acknowledgement........................................i

Abstract........................................ii

1. Introduction........................................1

2. Theory........................................2

    2.1. 8051........................................2

    2.2. LCD........................................8

    2.3. KEYPAD........................................14

3. Implementation........................................1.7

    3.1.Software implementation........................................17

    3.2. Hardware implementation........................................18

4. Conclusion........................................19

References........................................iii

Logbook........................................iv

# 1. Introduction

We have made a calculator using 8051 microcontroller which performs basic arithmatic operations namely addition, subtration, multiplication and division. Here, we accept the input in the hexadecimal form via a 4*4 keypad. This hexadecimal input is then converted to binary number system in the microcontroller. Then, the required operation is performed on the inputs. Finally, the binary result is then converted to the decimal number system and displayed on the LCD that is interfaced with the microcontroller.

# 2. Components

For implementing the hexadecimal calculator, we have used:

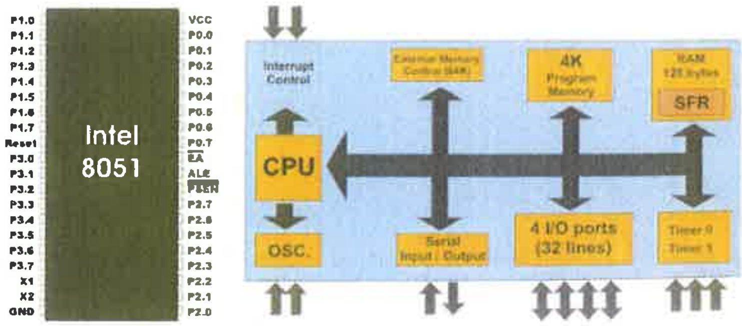
i) 8051 microcontroller
ii) 4*4 keypad
iii) 16*2 LCD

## 2.1.8051 Microcontroller

The Intel 8051 microcontroller is one of the most popular general purpose microcontrollers in use today. The success of the Intel 8051 spawned a number of clones which are collectively referred to as the MCS-51 family of microcontrollers, which includes chips from vendors such as Atmel, Philips, Infineon, and Texas Instruments.

Some of the features that have made the 8051 popular are:

- 4 KB on chip program memory.
- 128 bytes on chip data memory(RAM)
- 4 register banks.
- 128 user defined software flags.
- 8-bit data bus
- 16-bit address bus
- 16 bit timers (usually 2, but may have more, or less).
- 3 internal and 2 external interrupts.
- Bit as well as byte addressable RAM area of 16 bytes.
- Four 8-bit ports, (short models have two 8-bit ports).
- 16-bit program counter and data pointer.
- 1 Microsecond instruction cycle with 12 MHz Crystal.



### 2.1.1. Pins Description:

Pins 1-8: Port 1 Each of these pins can be configured as an input or an output.

Pin 9: RS A logic one on this pin disables the microcontroller and clears the contents of most registers. In other words, the positive voltage on this pin resets the microcontroller. By applying logic zero to this pin, the program starts execution from the beginning.

Pins10-17: Port 3 Similar to port 1, each of these pins can serve as general input or output. Besides, all of them have alternative functions:

Pin 10: RXD Serial asynchronous communication input or Serial synchronous communication output.

Pin 11: TXD Serial asynchronous communication output or Serial synchronous communication clock output.

Pin 12: INT0 Interrupt 0 input.

Pin 13: INT1 Interrupt 1 input.

Pin 14: T0 Counter 0 clock input.

Pin 15: T1 Counter 1 clock input.

Pin 16: WR Write to external (additional) RAM.

Pin 17: RD Read from external RAM.

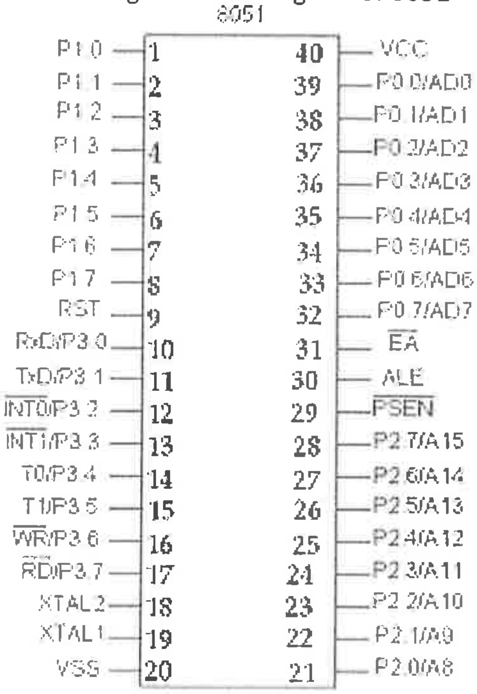
Pin 18, 19: X2, X1 Internal oscillator input and output. A quartz crystal which specifies operating frequency is usually connected to these pins. Instead of it, miniature ceramics resonators can also be used for frequency stability. Later versions of microcontrollers operate at a frequency of 0 Hz up to over 50 Hz.

Pin 20: GND Ground.

Pin 21-28: Port 2 If there is no intention to use external memory then these port pins are configured as general inputs/outputs. In case external memory is used, the higher address byte, i.e. addresses A8-A15 will appear on this port. Even though memory with capacity of 64Kb is not used, which means that not all eight port bits are used for its addressing, the rest of them are not available as inputs/outputs.

Pin 29: PSEN If external ROM is used for storing program then a logic zero (0) appears on it every time the microcontroller reads a byte from memory.

Figure 2: Pin diagram of 8051




Pin 30: ALE Prior to reading from external memory, the microcontroller puts the lower address byte (A0-A7) on P0 and activates the ALE output. After receiving signal from the ALE pin, the external register (usually 74HCT373 or 74HCT375 add-on chip) memorizes the state of P0 and uses it as a memory chip address. Immediately after that, the ALU pin is returned its previous logic state and P0 is now used as a Data Bus. As seen, port data multiplexing is performed by means of only one additional (and cheap) integrated circuit. In other words, this port is used for both data and address transmission.

Pin 31: EA By applying logic zero to this pin, P2 and P3 are used for data and address transmission with no regard to whether there is internal memory or not. It means that even there is a program written to the microcontroller, it will not be executed. Instead, the program written to external ROM will be executed. By applying logic one to the EA pin, the microcontroller will use both memories, first internal then external (if exists).

Pin 32-39: Port 0 Similar to P2, if external memory is not used, these pins can be used as general inputs/outputs. Otherwise, P0 is configured as address output (A0-A7) when the ALE pin is driven high (1) or as data output (Data Bus) when the ALE pin is driven low (0).

Pin 40: VCC +5V power supply

### 2.1.2. Input/Output Ports (I/O Ports)

All 8051 microcontrollers have 4 I/O ports each comprising 8 bits which can be configured as inputs or outputs. Accordingly, in total of 32 input/output pins enabling the microcontroller to be connected to peripheral devices are available for use.

Pin configuration, i.e. whether it is to be configured as an input (1) or an output (0), depends on its logic state. In order to configure a microcontroller pin as an output, it is necessary to apply a logic zero (0) to appropriate I/O port bit. In this case, voltage level on appropriate pin will be 0.

Similarly, in order to configure a microcontroller pin as an input, it is necessary to apply a logic one (1) to appropriate port. In this case, voltage level on appropriate pin will be 5V (as is the case with any TTL input). This may seem confusing but don't loose your patience. It all becomes clear after studying simple electronic circuits connected to an I/O pin.

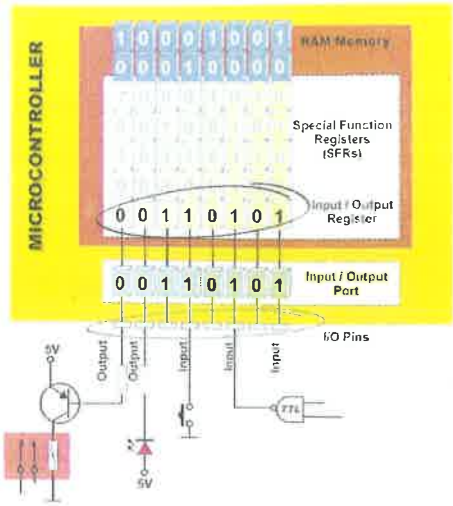



Figure 3: Input Output Port

### 2.1.2.1.Input/Output(I/O)pin

a) Outputpin

A logic zero (0) is applied to a bit of the P register. The output FE transistor is turned on, thus connecting the appropriate pin to ground.

b) Inputpin

A logic one (1) is applied to a bit of the P register. The output FE transistor is turned off and the appropriate pin remains connected to the power supply voltage over a pull-up resistor of high resistance.

### 2.1.3. Program Memory

The first models of the 8051 microcontroller family did not have internal program memory. It was added as an external separate chip. These models are recognizable by their label beginning with 803(for example 8031 or 8032). All later models have a few Kbyte ROM embedded. Even though such an amount of memory is sufficient for writing most of the programs, there are situations when it is necessary to use additional memory as well. A typical example are so called

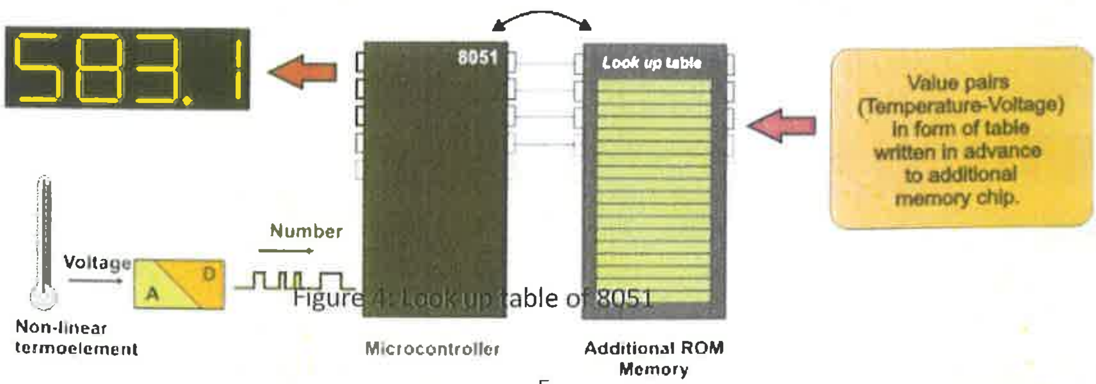



Figure 4: Look up table of 8051

lookup tables.

They are used in cases when equations describing some processes are too complicated or when there is no time for solving them. In such cases all necessary estimates and approximates are executed in advance and the final results are put in the tables (similar to logarithmic tables).

How does the microcontroller handle external memory depends on the EA pin logic state:
EA=0 In this case, the microcontroller completely ignores internal program memory and executes only the program stored in external memory.
EA=1 In this case, the microcontroller executes first the program from built-in ROM, then theprogram stored in external memory.In both cases, P0 and P2 are not available for use since being used for data and address transmission. Besides, the ALE and PSEN pins are also used.

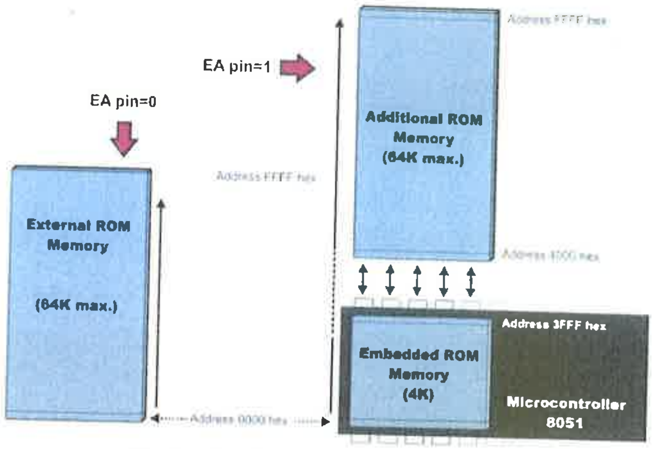


Figure 5: Memory management of 8051

### 2.1.4. Data Memory

As already mentioned, Data Memory is used for temporarily storing data and intermediate results created and used during the operation of the microcontroller. Besides, RAM memory built in the 8051 family includes many registers such as hardware counters and timers, input/output ports, serial data buffers etc. The previous models had 256 RAM locations, while for the later models this number was incremented by additional 128 registers. However, the first 256 memory locations (addresses 0-FFh) are the heart of memory common to all the models belonging to the 8051 family. Locations available to the user occupy memory space with addresses 0-7Fh, i.e. first 128 registers. This part of RAM is divided in several blocks.

The first block consists of 4 banks each including 8 registers denoted by R0-R7. Prior to accessing any of these registers, it is necessary to select the bank containing it. The next memory block (address 20h-2Fh) is bit- addressable, which means that each bit has its own address (0-7Fh). Since there are 16 such registers, this block contains in total of 128 bits with separate addresses (address of bit 0 of the 20h byte is 0, while address of bit 7 of the 2Fh byte is 7Fh). The third group of registers occupy addresses 2Fh-7Fh, i.e. 80 locations, and does not have any special functions or features.

### 2.1.5. Additional RAM

In order to satisfy the programmers' constant hunger for Data Memory, the manufacturers decided to embed an additional memory block of 128 locations into the latest versions of the 8051 microcontrollers. However, it's not as simple as it seems to be... The problem is that electronics performing addressing has 1 byte (8 bits) on disposal and is capable of reaching only the first 256 locations, therefore. In order to keep already existing 8-bit architecture and compatibility with other existing models a small trick was done.

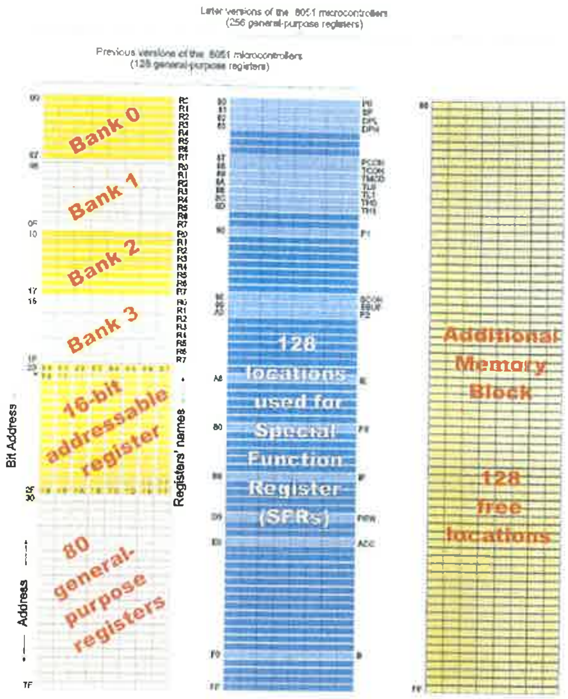



Figure 6: Register Bank

What does it mean? It means that additional memory block shares the same addresses with locations intended for the SFRs (80h- FFh). In order to differentiate between these two physically separated memory spaces, different ways of addressing are used. The SFRs memory locations are accessed by direct addressing, while additional RAM memory locations are accessed by indirect addressing.

### 2.1.6. Memory expansion

In case memory (RAM or ROM) built in the microcontroller is not sufficient, it is possible to add two external memory chips with capacity of 64Kb each. P2 and P3 I/O ports are used for their addressing and data transmission.From the user's point of view, everything works quite simply when properly connected because most operations are performed by the microcontroller itself. The 8051 microcontroller has two pins for data read RD#(P3.7) and PSEN#. The first one is used for reading data from external data memory (RAM), while the other is used for reading data from external program memory (ROM). Both pins are active low. A typical example of memory expansion by adding RAM and

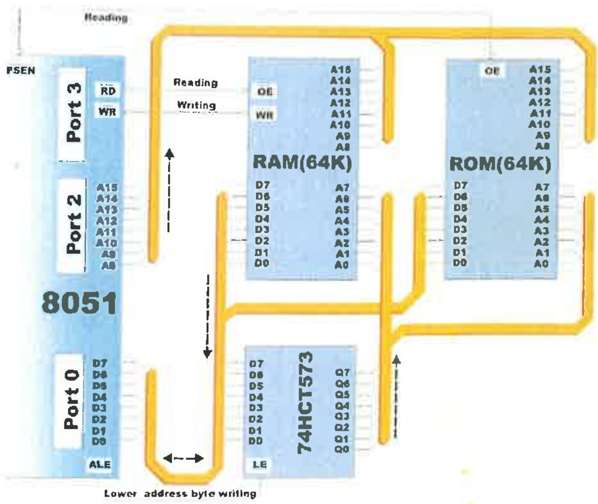



Figure 7: External Memory Management

ROM chips (Hardward architecture), is shown in figure above.
Even though additional memory is rarely used with the latest versions of the microcontrollers, we will describe in short what happens when memory chips are connected according to the previous schematic. The whole process described below is performed automatically.

When the program during execution encounters an instruction which resides in external memory (ROM), the microcontroller will activate its control output ALE and set the first 8 bits of address (A0-A7) on P0. IC circuit 74HCT573 passes the first 8 bits to memory address pins.
A signal on the ALE pin latches the IC circuit 74HCT573 and immediately afterwards 8 higher bits of address (A8-A15) appear on the port. In this way, a desired location of additional program memory is addressed. It is left over to read its content.

Port P0 pins are configured as inputs, the PSEN pin is activated and the microcontroller reads from memory chip.Similarly, it occurs when it is necessary to read location from external RAM. Addressing is performed in the same way, while read and write are performed via signals appearing on the control outputs RD (is short for read) or WR (is short for write).

### 2.2.Liquid Crystal Displays (LCD)

An LCD display is specifically manufactured to be used with microcontrollers, which means that it cannot be activated by standard IC circuits. It is used for displaying different messages on a miniature liquid crysal display.
The model described here is for its low price and great capabilities most frequently used in practice. It is based on the HD44780 microcontroller (Hitachi) and can display messages in two lines with 16 characters each. It displays all the letters of alphabet, Greek letters, punctuation marks, mathematical symbols etc. In addition, it is possible to display symbols made up by the user. Other useful features include automatic message shift (left and right), cursor appearance, LED backlight etc.

Figure 8: LCD Display

#### 2.2.1. LCD Pins

There are pins along one side of a small printed board. These are used forconnecting to the microcontroller. There are in total of 14 pins marked with numbers (16 if it has backlight). Their function is described in the table (1)

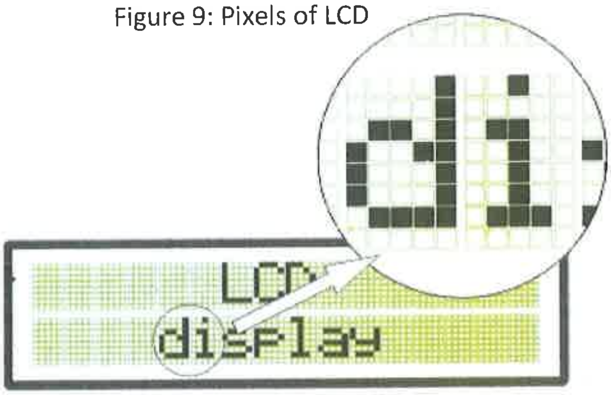
| FUNCTION | PIN NUMBER | NAME | LOGIC STATE | DESCRIPTION |
|---|---|---|---|---|
| Ground | 1 | Vss | - | 0V |
| Power supply | 2 | Vdd | - | +5V |
| Contrast | 3 | Vee | - | 0 - Vdd |
| Control of operating | 4 | RS | 0<br>1 | D0 – D7 are interpreted as commands<br>D0 – D7 are interpreted as data |
| | 5 | R/W | 0<br>1 | Write data (from controller to LCD)<br>Read data (from LCD to controller) |
| | 6 | E | 0<br>1<br>From 1 to 0 | Access to LCD disabled<br>Normal operating<br>Data/commands are transferred to LCD |
| Data / commands | 7 | D0 | 0/1 | Bit 0 LSB |
| | 8 | D1 | 0/1 | Bit 1 |
| | 9 | D2 | 0/1 | Bit 2 |
| | 10 | D3 | 0/1 | Bit 3 |
| | 11 | D4 | 0/1 | Bit 4 |
| | 12 | D5 | 0/1 | Bit 5 |
| | 13 | D6 | 0/1 | Bit 6 |
| | 14 | D7 | 0/1 | Bit 7 MSB |

Table 1: LCD pins

### 2.2.2. LCD screen

An LCD screen consists of two lines each containing 16 characters. Each character consists of 5x8 or 5x11 dot matrix. This book covers the most commonly used display, i.e. the 5x8 character display.

Figure 9: Pixels of LCD




Display contrast depends on the power supply voltage and whether messages are displayed in one or two lines. For this reason, varying voltage 0-Vdd is applied on the pin marked as Vee. Trimmer potentiometer is usually used for that purpose. Some LCD displays have built-in backlight (blue or green LEDs). When used during operation, a current limiting resistor should be serially connected to one of the pins for backlight power supply (similar to LEDs).

If there are no characters displayed or if all of them are dimmed when the display is on, the first thing that should be done is to check the potentiometer for contrast regulation. Is it properly adjusted? The same applies if the mode of operation has been changed (writing in one or two lines).

### 2.2.3. LCD Basic Commands

All data transferred to LCD through the outputs D0-D7 will be interpreted as a command or a data, which depends on the pin RS logic state:

RS = 1 - Bits D0-D7 are addresses of the characters to be displayed. LCD processor addresses one character from the character map and displays it. The DDRAM address specifies the location on which the character is to be displayed. This address is defined before the character is transferred or the address of previously transferred character is automatically incremented.

RS = 0 - Bits D0 - D7 are commands which determine the display mode. The commands recognized by the LCD are given in the table (2).

Table 2: LCD commands

| COMMAND | RS | RW | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | EXECUTION TIME |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Clear display | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1.64mS |
| Cursor home | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | x | 1.64mS |
| Entry mode set | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | I/D | S | 40uS |
| Display on/off control | 0 | 0 | 0 | 0 | 0 | 0 | 1 | D | U | B | 40uS |
| Cursor/Display Shift | 0 | 0 | 0 | 0 | 0 | 1 | D/C | R/L | x | x | 40uS |
| Function set | 0 | 0 | 0 | 0 | 1 | DL | N | F | x | x | 40uS |
| Set CGRAM address | 0 | 0 | 0 | 1 | CGRAM address | | | | | | 40uS |
| Set DDRAM address | 0 | 0 | 1 | DDRAM address | | | | | | | 40uS |
| Read "BUSY" flag (BF) | 0 | 1 | BF | DDRAM address | | | | | | | - |
| Write to CGRAM or DDRAM | 1 | 0 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | 40uS |
| Read from CGRAM or DDRAM | 1 | 1 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | 40uS |

I/D 1 = Increment (by 1)
0 = Decrement (by 1)

R/L 1 = Shift right
0 = Shift left

S 1 = Display shift on

DL 1 = 8-bit interface

0 = Display shift off

D 1 = Display on
0 = Display off

U 1 = Cursor on
0 = Cursor off

B 1 = Cursor blink on
0 = Cursor blink off

0 = 4-bit interface

N 1 = Display in two lines
0 = Display in one line

F 1 = Character format 5x10 dots
0 = Character format 5x7 dots

D/C 1 = Display shift
0 = Cursor shift

### 2.2.4. LCD Connection

Depending on how many lines are used for connecting the LCD to the microcontroller, there are 8-bit and 4-bit LCD modes. The appropriate mode is selected at the beginning of the operation. This process is called "initialization". 8-bit LCD mode uses outputs D0-D7 to transfer data in the way explained on the previous page. The main purpose of 4-bit LED mode is to save valuable I/O pins of the microcontroller. Only 4 higher bits (D4-D7) are used for communication, while other may be left unconnected. Each data is sent to the LCD in two steps: four higher bits are sent first (normally through the lines D4-D7), then four lower bits. Initialization enables the LCD to link and interpret received bits correctly. Data is rarely read from the LCD (it is mainly transferred from the microcontroller to LCD) so that it is often possible to save an extra I/O pin by simple connecting R/W pin to ground. Such saving has its price. Messages will be normally displayed, but it will not be possible to read the busy flag since it is not possible to read the display either.

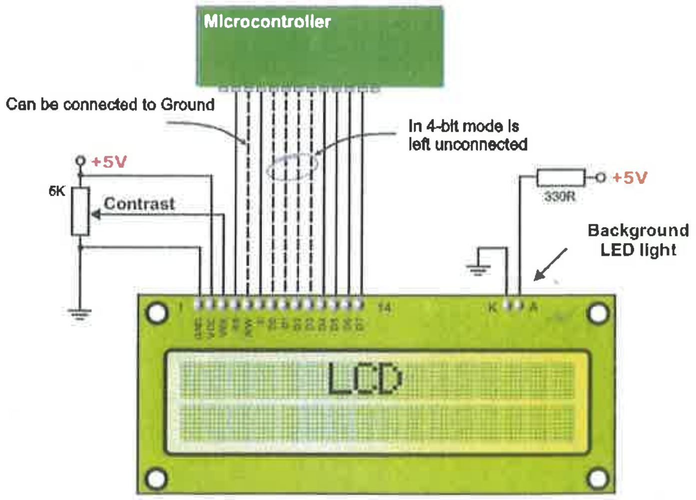



Figure 10: LCD connection to 8051

Fortunately, there is a simple solution. After sending a character or a command it is important to give the LCD enough time to do its job. Owing to the fact that execution of the slowest command lasts for approximately 1.64mS, it will be sufficient to wait approximately 2mS for LCD.

LCD Initialization

The LCD is automatically cleared when powered up. It lasts for approximately 15mS. After that, the display is ready for operation. The mode of operation is set by default. It means that:

Display is cleared
Mode
DL = 1 Communication through 8-bit interface
N = 0 Messages are displayed in one line
F = 0 Character font 5 x 8 dots
Display/Cursor on/off
D = 0 Display off
U = 0 Cursor off
B = 0 Cursor blink off
Character entry
ID = 1 Displayed addresses are automatically incremented by 1
S = 0 Display shift off

Automatic reset is in most cases performed without any problems. In most cases, but not always! If for any reason the power supply voltage does not reach ful value within 10mS, the display will start to perform completely unpredictably. If the voltage supply unit is not able to meet this condition or if it is needed to provide completely safe operation, the process of initialization is applied. Initialization, among other things, causes a new reset enabling display to operate normally.

Refer to the figure below for the procedure on 8-bit initialization:

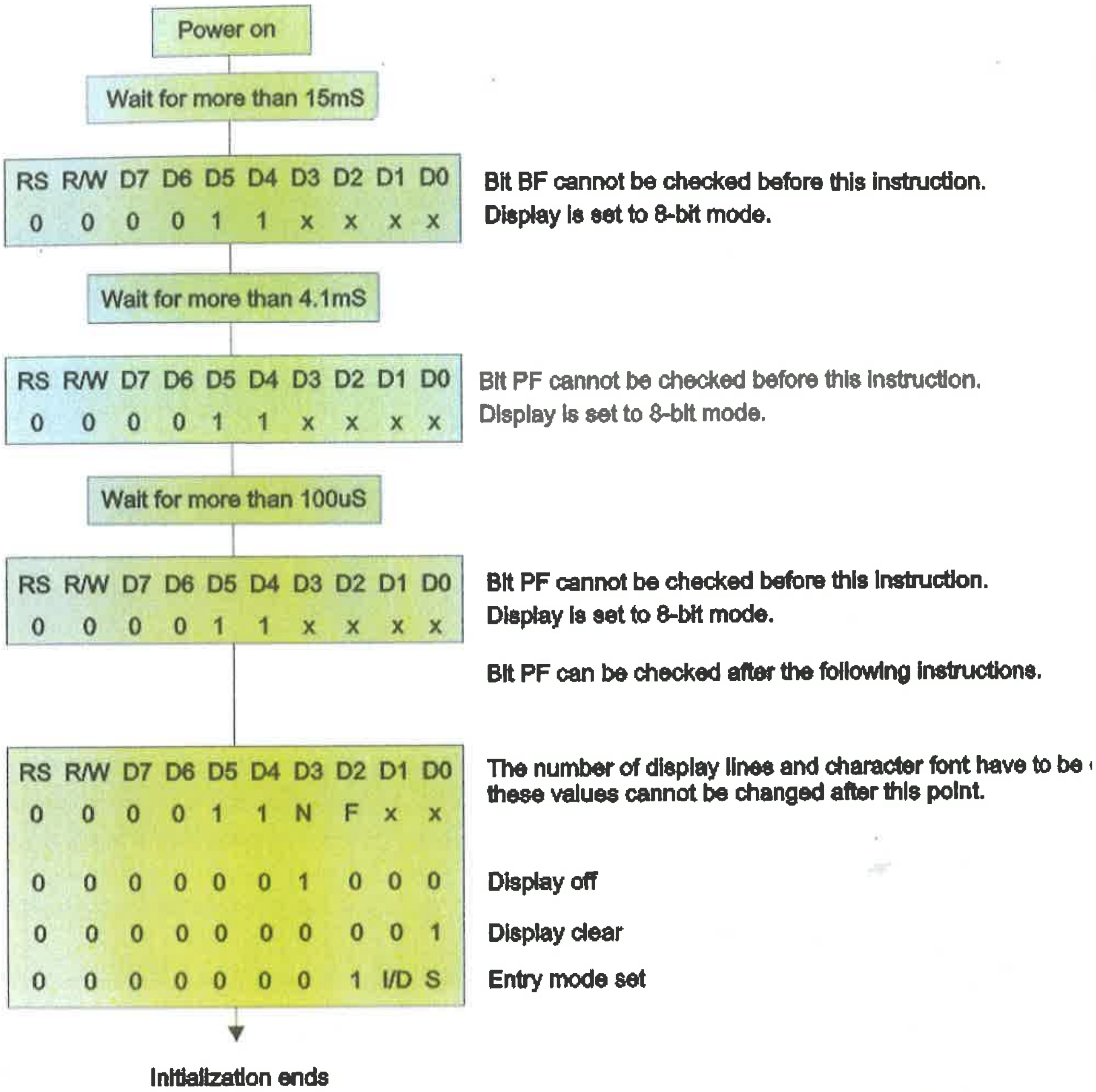


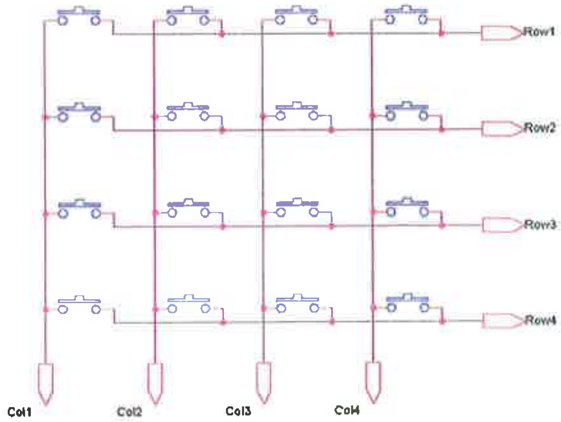

Figure 11: LCD commands

## 2.3. KEYPAD

Matrix Keypads are commonly used in calculators, telephones etc where a number of input switches are required. We know that matrix keypad is made by arranging push button switches in row and columns. In the straight forward way to connect a 4×4 keypad (16 switches) to a microcontroller we need 16 inputs pins. But by connecting switches in the following way we can read the status of each switch using 8 pins of the microcontroller.

### 2.3.1. 4×4-Matrix-Keypad

Figure 12: 4x4 Matrix Keypad




The key board here we are interfacing is a matrix keyboard. This key board is designed with a particular rows and columns. These rows and columns are connected to the microcontroller through its ports of the micro controller 8051. We normally use 8*8 matrix key board. So only two ports of 8051 can be easily connected to the rows and columns of the key board.

When ever a key is pressed, a row and a column gets shorted through that pressed key and all the other keys are left open. When a key is pressed only a bit in the port goes high. Which indicates microcontroller that the key is pressed. By this high on the bit key in the corresponding column is identified.

Once we are sure that one of key in the key board is pressed next our aim is to identify that key. To do this we firstly check for particular row and then we check the corresponding column the key board.To check the row of the pressed key in the keyboard, one of the row is made high by making one of bit in the output port of 8051 high . This is done until the row is found out. Once we get the row next out job is to find out the column of the pressed key. The column is detected by contents in the input ports with the help of a counter. The content of the input port is rotated with carry until the carry bit is set.

The contents of the counter is then compared and displayed in the display. This display is designed using a seven segment display and a BCD to seven segment decoder IC 7447.The BCD equivalent number of counter is sent through output part of 8051 displays the number of pressed key.

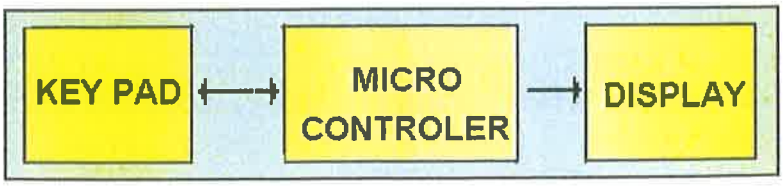



Figure 13: Interfacing of keypad and Lcd with 8051

### 2.3.2. Circuit diagram of INTERFACING KEYPAD TO 8051.

The programming algorithm, program and thė circuit diagram is as follows. Here program is explained with comments.

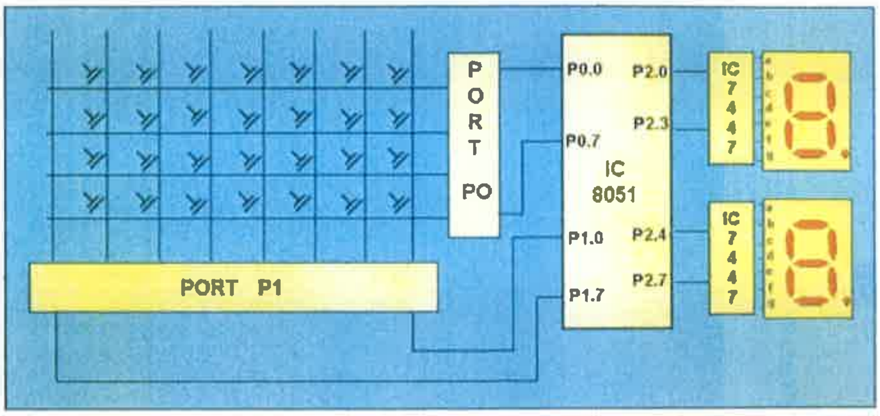



Figure 14: Interfacing of 8051

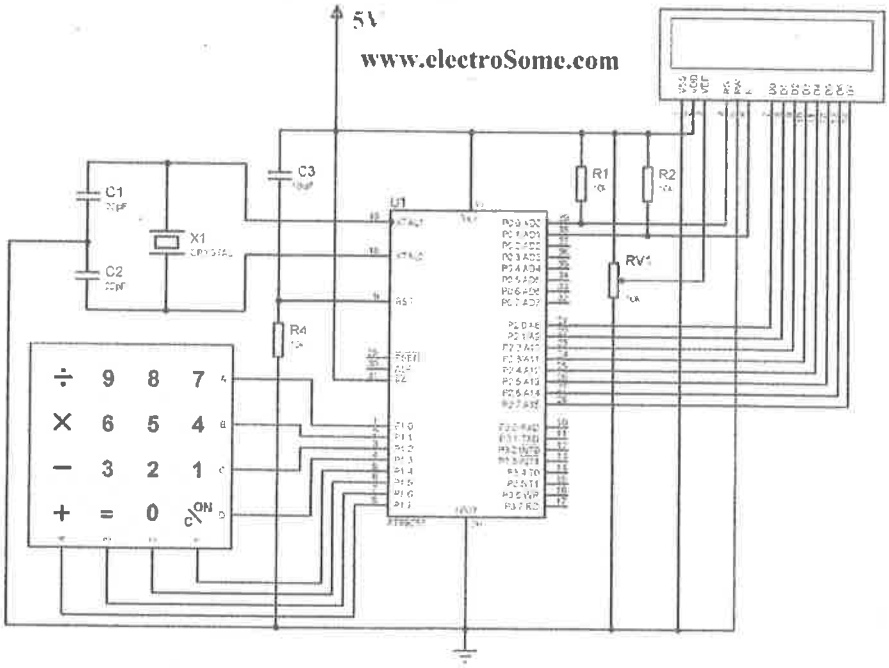
Keyboard is organized in a matrix of rows and columns as shown in the figure. The microcontroller accesses both rows and columns through the port.

The 8051 has 4 I/O ports P0 to P3 each with 8 I/O pins, P0.0 to P0.7,P1.0 to P1.7, P2.0 to P2.7, P3.0 to P3.7. The one of the port P1 (it understood that P1 means P1.0 to P1.7) as an I/P port for microcontroller 8051, port P0 as an O/P port of microcontroller 8051 and port P2 is used for displaying the number of pressed key.

1) Make all rows of port P0 high so that it gives high signal when key is pressed.
2) See if any key is pressed by scanning the port P1 by checking all columns for non zero condition.
3) If any key is pressed, to identify which key is pressed make one row high at a time.
4) Initiate a counter to hold the count so that each key is counted.
5) Check port P1 for nonzero condition. If any nonzero number is there in [accumulator], start column scanning by following step 9.
6) Otherwise make next row high in port P1.
7) Add a count of 08h to the counter to move to the next row by repeating steps from step 6.
8) If any key pressed is found, the [accumulator] content is rotated right through the carry until carry bit sets, while doing this increment the count in the counter till carry is found.
9) Move the content in the counter to display in data field or to memory location
10) To repeat the procedures go to step 2.

### 2.3.3. Interfacing Keypad with 8051 Microcontroller using Keil C

Figure 14: Circuit Diagram




10KΩ resistor and 10μF will provide the required Power On Reset (POR) signal to the 8051 microcontroller. 12MHz crystal is used to provide required clock for the microcontroller and 22pF capacitors will stabilize the oscillations of the crystal. AT89C51 can works upto 24MHz. We can choose the required frequency by changing the crystal and clock frequency in the project settings of Keil C. Keypad is connected to the Port P1 and column inputs pins are pulled up internally. 16×2 LCD is connected to Port P2 and P0. P0.0 and P0.1 pins are pulled up externally using 10KΩ resistors since Port P0 has no internal pull up.

# 3. Implementation

## 3.1. Software Implementation

We have simulated our circuit on Proteus. Then, we have used Eagle software to make the layout.

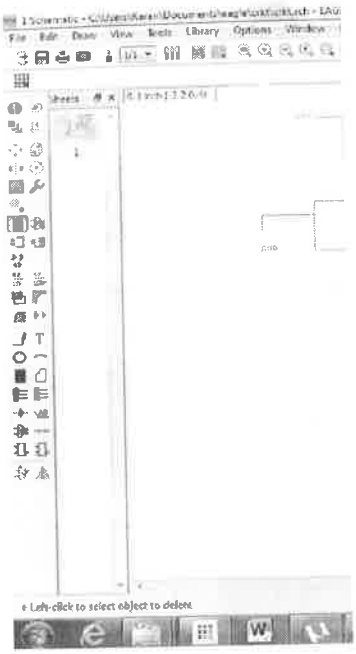



Figure 15: circuit Diagram

### 3.2. Hardware Implementation

We then printed the layout and carried out ironing, etching, drilling and soldering. But, it did not give us the required results, hence we shifted to the general circuit board.

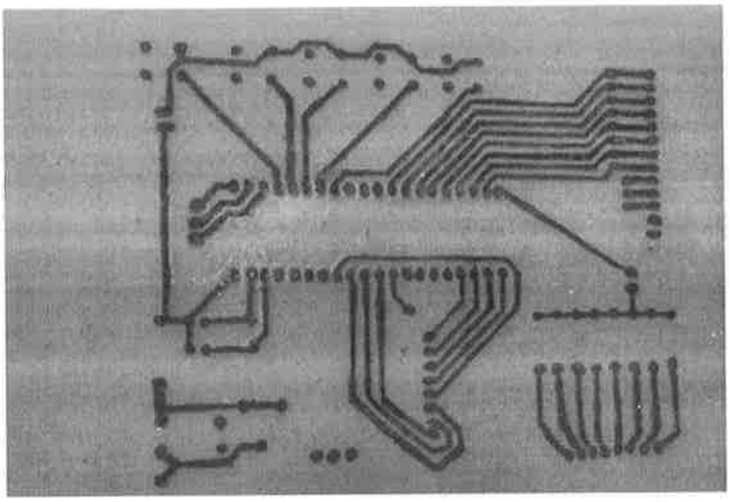


Figure 17: PCB layout

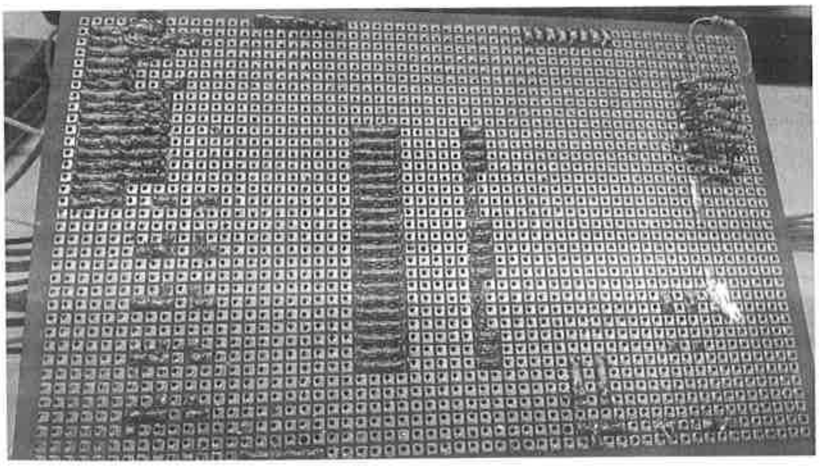


Figure 18: General Purpose Board

## Conclusion

During the designing, many problems were encountered. To minimize the problems encountered, we can use C++ coding instead of Asm code. Asm coding is very complicated and hard to understand. The C coding can also be used as it is easier to implement it. For future development, we can use a Arduino board or any other microcontroller for making an advanced calculator with many other features.

## References

[1] www.microe.com

[2]www.electrosome.com

[3]8051 Microcontrollers and Embedded systems, Muhammad Ali Mazidi

Karan Shah
60002120095

Department of Electronics and Telecommunication Engineering

T. E.(Project) LogBook

| Date | Activity | Outcome | Comment on outcome | Resources utilized | Next meeting date | Target | Guide's Remark & signature |
|---|---|---|---|---|---|---|---|
| 19/1/15 | Research for deciding the Project topic | Choose 'health check upsystem | not pratically possible for mini project | Internet | 2/2/15 | to finalize topic. | [signature] |
| 23/2/15 | found the circuit diagram & implemented on proteus | Got o/p as intended. | Perfect o/p not obtained for 8051 pins misalignment | Internet proteus s/w | 2/3/15 | How to use eagle s/w now. | [signature] |

## Department of Electronics and Telecommunication Engineering

### T. E.(Project) LogBook

| Date | Activity | Outcome | Comment on outcome | Resources utilized | Next meeting date | Target | Guide's Remark & signature |
|---|---|---|---|---|---|---|---|
| 2/3/15 | obtain circuit & PCB layout using Eagle s/w | got perfect layout | The layout was as needed | Eagle S/W | 9/3/15 | How to implement on bread board | [signature] |
| 9/3/15 | cut PCB of required size & traced the layout on it by ironing sketching cleaning | base for soldering was ready | Now we could start soldering | $FeCl_3$ solution for etching iron, xerox of layout, pcb, acetone, cotton etc | 16/3/15 | to solder the pcb as required. | [signature] |

| Date | Activity | Outcome | Comment on outcome | Resources utilized | Next meeting date | Target | Guide's Remark & signature |
|---|---|---|---|---|---|---|---|
| 16/3/15 | Soldering the components & burning the 8051uc IC with hexcode & connections | PCB ready for final testing. | Entire PCB hardware connected & mounted with s/w installed successfully | components, IC burner, solderiron, & wire. | 30/3/15 | final testing | [signature] |
| 30/3/15 | making report & final testing of the project completed (did on dotted PCB as well) | final project done. | Project is ready | Supply internet | 13/4/15 | Project should work properly | [signature] |

# Department of Electronics and Telecommunication Engineering
## T. E.(Project) LogBook

| Date | Activity | Outcome | Comment on outcome | Resources utilized | Next meeting date | Target | Guide's Remark & signature |
|---|---|---|---|---|---|---|---|
| 16/3/15 | soldering the components & burning the 8051uc IC with hexcode & connections | PCB ready for final testing. | Entire PCB hardware connected & mounted with s/w installed successfully | components, IC burner, solderiron & wire. | 30/3/15 | final testing | [signature] |
| 30/3/15 | making report & final testing of the project completed (did on dotted PCB as well) | final project done. | Project is ready | supply internet | 13/4/15 | Project should work properly | [signature] |