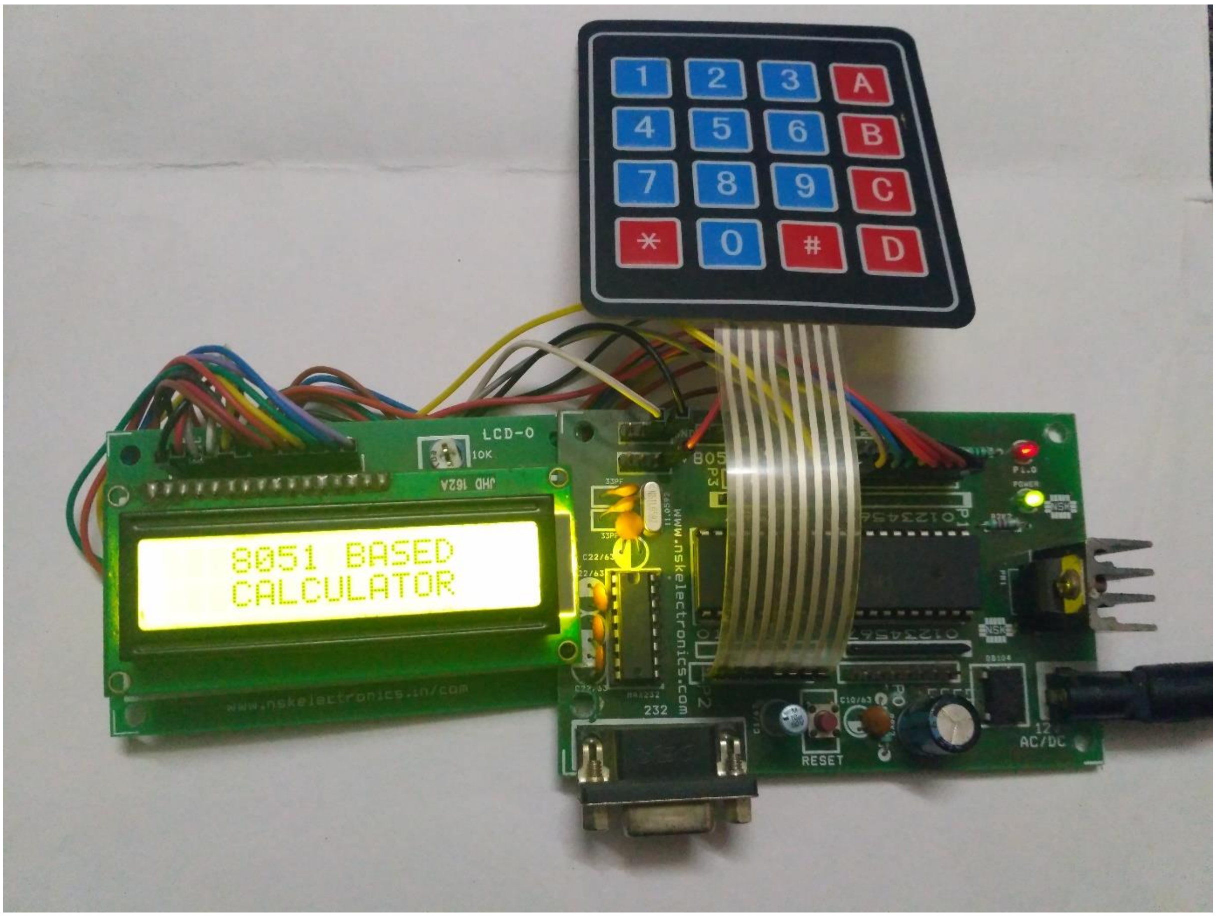
MODEL



ABSTRACT:

This project is an implementation of microcontroller based calculator which will perform simple arithmetic using a calculator keypad and a 16x2 Liquid Crystal Display (LCD). The program is limited to single digit input and double-digit results. This allows the program for the arithmetic operations to be simple, while the same principles can be extended to multi-digit calculations.

INTRODUCTION:

Advancement in technology has led to building electronic devices with simple circuit. Introduction of microcontroller has made designing of electronic devices circuit simpler. A computer on a chip is known as microcontroller. It is essential for the operation of devices such as mobile phones, video cameras, electrical appliances and most self-contained electronic systems. Microcontroller has the following elements: memory, central processing unit, ports, bus, serial communication, etc.

Memory: Program and data are stored in the memory. Central Processing Unit (CPU) has capability to multiply, divide, subtract, and move its contents from Memory locations which are called registers. Registers are memory locations whose role is to perform various mathematical operations or any other operations with data wherever data can be found. Bus is a connection between memory and CPU-the path through which data goes from one block to another. Physically, it represents a group of 8, 16, or more wires. There are two types of buses: address and data buses. The first one has many lines as the amount of memory to address, and the other one is as wide as data. First one transmits address from CPU memory, and the second connects all blocks inside the microcontroller. Ports have several memory locations whose one end is connected to the data bus, and the other has connection with the output lines on the microcontroller which can be seen as pins on the electronic component. There are several types of ports: input, output or bidirectional ports. When working with ports, first of all it is necessary to choose which port one needs to work with, and then to send data to, or take it from the port. When working with it the port acts like a memory location.

Something is being written into or read from it, and this could be noticed on the pins of the microcontroller. Watchdog is a free-run counter where a program needs to write a zero in every time it executes correctly. Analog to digital converter (ADC) is responsible for converting an information about some analog value to a binary number and follow it through to a CPU block so that CPU block can further process it. Finally, the microcontroller is completed, and all need to be done is to assemble it into an electronic component where it will access

Theory of Design:

The circuit for a calculator which will perform simple arithmetic operations in the PIC 16F887 Microcontroller, using a calculator keypad and 16X2 LCD display, is shown in Figure. The keypad has 16 keys: 10 numeric buttons, 4 arithmetic operations, equals and clear. The results obtained are displayed on the LCD display.

DESCRIPTION:

NUVOTON (8051):

The microcontroller incorporates all the features that are found in microprocessor. The microcontroller has built in ROM, RAM, Input Output ports, Serial Port, timers, interrupts and clock circuit. A microcontroller is an entire computer manufactured on a single chip. Microcontrollers are usually dedicated devices embedded within an application. For example, microcontrollers are used as engine controllers in automobiles and as exposure and focus controllers in cameras. In order to serve these applications, they have a high concentration of on-chip facilities such as serial ports, parallel input output ports, timers, counters, interrupt control, analog-to-digital converters, random access memory, read only memory, etc. The I/O, memory, and on-chip peripherals of a microcontroller are selected depending on the specifics of the target application. Since microcontrollers are powerful digital processors, the degree of control and programmability they provide significantly enhances the effectiveness of the application.

In our project we are using a 8051 based NUVOTON W78E052D microcontroller. It is an 8-bit microcontroller which can accommodate a wider frequency range with low power consumption. The W78E052D series contains 8Kbytes Flash EPROM programmable by hardware writer; a 256 bytes RAM; four 8-bit bi-directional (P0,P1,P2,P3) and bit-addressable I/O ports; an additional 4-bit I/O port P4; three 16-bit timer/counters; a hardware watchdog timer and a serial port. These peripherals are supported by 8 sources 4-level interrupt capability. To facilitate programming and verification, the Flash EPROM inside the W78E052D series allows the program memory to be programmed and read electronically.

FEATURES OF W78E052D MICROCONTROLLER

- ✓ Fully static design 8-bit CMOS microcontroller
- ✓ Pin and Instruction-sets compatiblewithMCS-51
- ✓ 256 bytes of on-chip scratchpad RAM
- ✓ 8K bytes electrically erasable/programmable Flash EPROM
- ✓ Four 8-bit bi-directional ports
- ✓ One extra 4-bit bit-addressable I/O port
- ✓ Three 16-bit Timers.

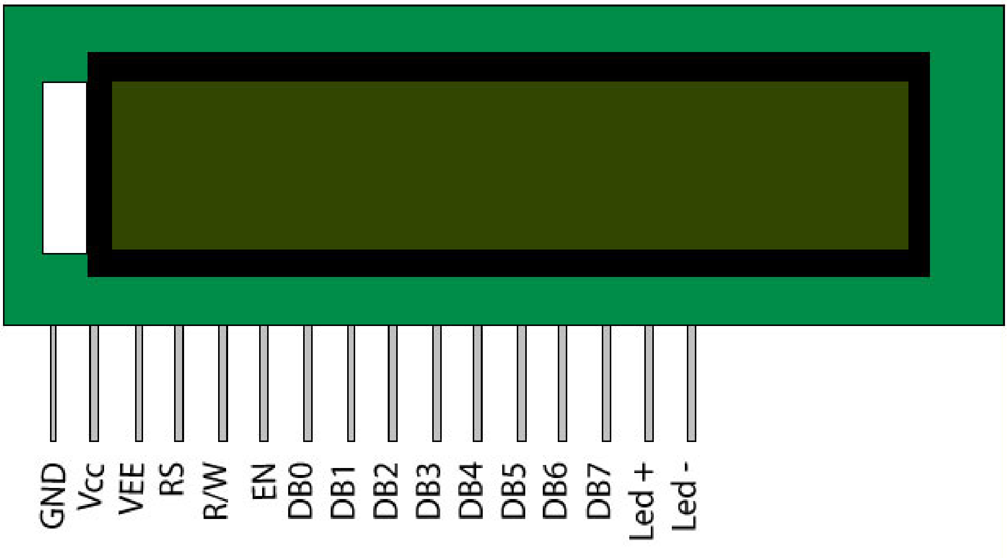
LCD DISPLAY

LCD (Liquid Crystal Display) screen is an electronic display module and find a wide range of applications. A 16x2 LCD display is very basic module and is very commonly used in various devices and circuits. These modules are preferred over seven segments and other multi segment LEDs. The reasons being: LCDs are economical; easily programmable; have no limitation of displaying special & even custom characters (unlike in seven segments), animations and so on.

A 16x2 LCD means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5x7 pixel matrix. This LCD has two registers, namely, Command and Data.

The command register stores the command instructions given to the LCD. A command is an instruction given to LCD to do a predefined task like initializing it, clearing its screen, setting the cursor position, controlling display etc. The data register stores the data to be displayed on the LCD. The data is the ASCII value of the character to be displayed on the LCD. Click to learn more about internal structure of a LCD.

Pin diagram



PIN DESCRIPTION:

Pin number	Symbol	Function
1	Vss	Ground Terminal
2	Vcc	Positive Supply
3	Vdd	Contrast adjustment
4	RS	Register Select; 0→Instruction Register, 1→Data Register
5	R/W	Read/write Signal; 1→Read, 0→ Write
6	E	Enable; Falling edge
7	DB0	Bi-directional data bus, data transfer is performed once, thru DB0 to DB7, in the case of interface data length is 8-bits; and twice, through DB4 to DB7 in the case of interface data length is 4-bits. Upper four bits first then lower four bits.
8	DB1	
9	DB2	
10	DB3	
11	DB4	
12	DB5	
13	DB6	
14	DB7	
15	LED-	Back light LED cathode terminal
16	LED+	Back Light LED anode termina

KEYPAD

A keypad is a set of buttons arranged in a block or "pad" which usually bear digits, symbols and usually a complete set of alphabetical letters. If it mostly contains numbers then it can also be called a numeric keypad. Keypads are found on many alphanumeric keyboards and on other devices such as calculators, push-button telephones, combination locks, and digital door locks, which require mainly numeric input.

This 16-button keypad provides a useful human interface component for microcontroller projects. Convenient adhesive backing provides a simple way to mount the keypad in a variety of applications.

Features

- ✓ Ultra-thin design
- ✓ Adhesive backing
- ✓ Excellent price/performance ratio
- ✓ Easy interface to any microcontroller

How it Works

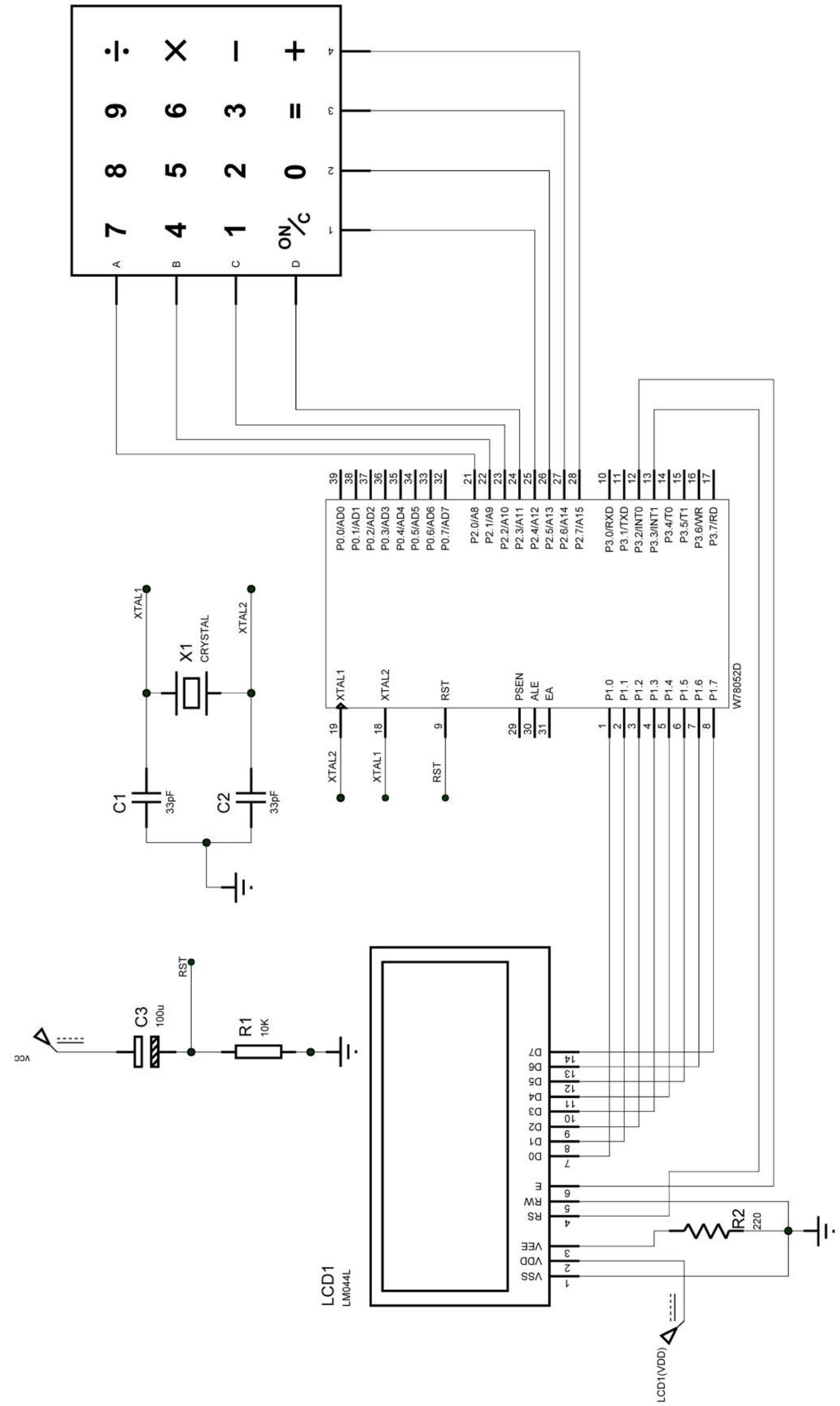
Matrix keypads use a combination of four rows and four columns to provide button states to the host device, typically a microcontroller. Underneath each key is a pushbutton, with one end connected to one row, and the other end connected to one column. In order for the microcontroller to determine which button is pressed, it first needs to pull each of the four columns (pins 1-4) either low or high one at a time, and then poll the states of the four rows (pins 5-8). Depending on the states of the columns, the microcontroller can tell which button is pressed.

For example, say your program pulls all four columns low and then pulls the first row high. It then reads the input states of each column, and reads pin 1 high. This means that a contact has been made between column 4 and row 1, so button ‘A’ has been pressed.

The calculator operates as follows:

- ✓ In order to perform a calculation, press a number key, followed by an operation key, then another number and then equals.
- ✓ The calculation and result are displayed. For the divide operation, the result is displayed as result and remainder.
- ✓ If an invalid key sequence is entered, the program should be restarted.
- ✓ The calculation routine uses the operation input code to select the required process: add, subtract, multiply or divide.
- ✓ The binary result of the calculation is passed to a routine to convert it into BCD, then ASCII, and send it to the display.
- ✓ The result of the divide, being a single digit result and remainder, is sent direct to the display.

CIRCUIT DIAGRAM



CODE:

```
#include<reg51.h>
#include<key_lcd.h>             /*Including Keypad & LCD file*/
void main()
        {
                unsigned char a,b,c,op;
                init_lcd();
                clr_lcd();
                set_cur(1,3);
                lstrng("8051 BASED");
                set_cur(2,3);
                lstrng("CALCULATOR");
                delay(300);
                clr_lcd();
                while(1)
                        {
                                a=key();                /*to get first number*/
                                clr_lcd();
                                ldata(a);               /*display it on LCD*/
                                a=conv(a);              /*converting ASCII to DECIMAL*/
                                op=key();               /*to get operator*/
                                ldata(op);
                                b=key();
                                ldata(b);
                                b=conv(b);
                                c=key();
                                if(c=='=')
                                        switch(op)
                                                {
                                                        case '+': op=a+b;ldata('=');lvar(op);break;
                                                        case '-':  op=a-b;ldata('=');lvar(op);break;
                                                        case '*': op=a*b;ldata('=');lvar(op);break;
                                                        case '/':  op=a/b;ldata('=');lvar(op);
                                                                   ldata(' ');op=a%b;lvar(op);break;
                                                }
                        }
        }
```

```
/************************KEY_LCD.H**********************************/
/**************Declaring pins of LCD & KEYPAD************************/
sbit rs=P3^3;                    /*Register Select Pin*/
sbit en=P3^2;                    /*Enable Pin*/
sbit R1=P2^0;                    /*Row 1*/
sbit R2=P2^1;                    /*Row 2*/
sbit R3=P2^2;                    /*Row 3*/
sbit R4=P2^3;                    /*Row 4*/
sbit C1=P2^4;                    /*Column 1*/
sbit C2=P2^5;                    /*Column 2*/
sbit C3=P2^6;                    /*Column 3*/
sbit C4=P2^7;                    /*Column 4*/
/*******************************************************************/

/*********************Function Declaration*****************************/
unsigned char conv(unsigned char dat);
void delay(unsigned int ms);
void init_lcd();
void lcmd(unsigned char cmd);
void ldata(unsigned char msg);
void lstrng(unsigned char *msg);
unsigned char read_key();
unsigned char key();
/*******************************************************************/

/*********************Function Definition******************************/
unsigned char conv(unsigned char dat)   /*To convert ASCII to DECIMAL*/
    {
        switch(dat)
            {
                case '0':return 0;break;
                case '1':return 1;break;
                case '2':return 2;break;
                case '3':return 3;break;
                case '4':return 4;break;
                case '5':return 5;break;
                case '6':return 6;break;
                case '7':return 7;break;
                case '8':return 8;break;
                case '9':return 9;break;
            }
    }
```

```
unsigned char read_key()   /* Scanning Pressed Key from Kepad */
        {
                C1=1;C2=1;C3=1;
                R1=0;R2=1;R3=1;R4=1;        /*first row */
                if(C1==0){delay(1);while(C1==0);return '7';}  /*first column  */
                if(C2==0){delay(1);while(C2==0);return '8';}  /*second column */
                if(C3==0){delay(1);while(C3==0);return '9';}  /*third column  */
                if(C4==0){delay(1);while(C4==0);return '/';}  /*fourth column */
                R1=1;R2=0;R3=1;R4=1;        /*second row */
                if(C1==0){delay(1);while(C1==0);return '4';}  /*first column  */
                if(C2==0){delay(1);while(C2==0);return '5';}  /*second column */
                if(C3==0){delay(1);while(C3==0);return '6';}  /*third column  */
                if(C4==0){delay(1);while(C4==0);return '*';}  /*fourth column */
                R1=1;R2=1;R3=0;R4=1;         /*third row */
                if(C1==0){delay(1);while(C1==0);return '1';}  /*first column  */
                if(C2==0){delay(1);while(C2==0);return '2';}  /*second column */
                if(C3==0){delay(1);while(C3==0);return '3';}  /*third column  */
                if(C4==0){delay(1);while(C4==0);return '-';}  /*fourth column */
                R1=1;R2=1;R3=1;R4=0;        /*fourth row */
                if(C1==0){delay(1);while(C1==0);return 'C';}  /*first column  */
                if(C2==0){delay(1);while(C2==0);return '0';}  /*second column */
                if(C3==0){delay(1);while(C3==0);return '=';}  /*third column        */
                if(C4==0){delay(1);while(C4==0);return '+';}  /*fourth column */
                return 'n';
        }
unsigned char key(void)   /* Reapeat scanning until key pressed*/
        {
                char key='n';
                while(key=='n')
                key=read_key();
                return key;
        }

void delay(unsigned int ms)    /* 1ms Delay generation */
        {
                int i,j;
                for(i=0;i<ms;i++)
                        for(j=0;j<1275;j++);
        }
```

```
void init_lcd()        /*Initializing LCD*/
    {
        lcmd(0x38);
        delay(5);
        lcmd(0x0c);
        delay(5);
        lcmd(0x01);
        delay(5);
        lcmd(0x06);
        delay(5);
    }

void lcmd(unsigned char cmd)  /*Function to send command*/
    {
        P1=cmd;
        en=1;
        rs=0;
        delay(1);
        en=0;
    }

void ldata(unsigned char msg)  /*Function to send Data*/
    {
        P1=msg;
        en=1;
        rs=1;
        delay(1);
        en=0;
    }

void lvar(unsigned char val)/*Function to display content of variable*/
    {
        unsigned char temp;
        temp = val/100;             // 123 / 100 = 1
        if(temp!=0)
            ldata(temp+0x30);
        temp = val/10;              // 123 / 10 = 12
        temp = temp%10;             // 12 % 10 = 2
        ldata(temp+0x30);
        temp = val%10;              // 123 % 10 = 3
        ldata(temp+0x30);
    }
```

```
void lstrng(unsigned char *msg)    /*Function to send String to LCD*/
        {
                int k;
                for(k=0;msg[k]!=0;msg++)
                        ldata(msg[k]);
        }
void clr_lcd()                     /*Function to clear LCD screen*/
        {
                lcmd(0x01);
                lcmd(0x80);
        }
void set_cur(char line, char position)/*Function to set cursor*/
        {
                if(line==1)
                        lcmd(0x80+position);
                if(line==2)
                        lcmd(0xC0+position);
        }


/***********************************************************************/
```

Implementation and Testing

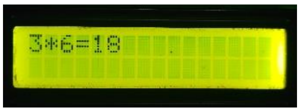
The system is successfully implemented based on the system design, Microcontroller 8051 was programmed with C-language and compiled using KEIL Compiler Series of programs were written and simulated using PROTEUS before the working program was finally achieved and then transferred to the microcontroller chip ISP- FLASH UTILITY software. The actual prototype implementation, simulations were carried out to test if the codes were working correctly.

Results:

Addition:

Subtraction:

Multiplication:



Division:

CONCLUSION:

There is need for a portable, reliable, low cost and faster means of calculation with simple design. This study designed and implemented a Microcontroller based calculator for easy and speedy calculation. This performs single digit operations only , but this can be implemented to two digits or more