University Of Engineering And Technology Taxila

MICROPROCESSOR AND MICROCONTROLLER

Lab Project Report

Topic:

Calculator with display on 40x4 LCD module

Submitted To:

Mam. Komal Munir
Lab Engineer, EED, UET Taxila.

Submitted By:

Talha Ahmad	(18-EE-13)
Azib Farooq	(18-EE-43)
Zarqa Sana	(18-EE-61)

ABSTRACT:

In this project we focused on design and implementation of a microcontroller PIC16F877a based calculator that will perform simple arithmetic as per the operators like for addition (+), subtraction (-), multiplication(x) and division (/) using a calculator keypad and 40x4 Liquid Crystal Display(LCD).The circuit was built in Proteus software and mikroC was used as compiler.

CONTENT TABLE:

INTRODUCTION:

Advancement in technology has led to building electronic devices with simple circuit. Introduction of microcontroller has made designing of electronic devices circuit simpler. 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. PIC16F877a is microcontroller and is very convenient to use, the coding or programming of this controller is also easier. One of the main advantages is that it can be write-erase as many times as possible because it uses FLASH memory technology. It is used in Embedded Projects like Home Automation System, Bank Security System etc. The reason LCD is more popular than LED, Seven Segment displays. Because we can display characters, numbers and custom characters with ease (Just by easily programming a module) and thus for designing a calculator we used 40x4 lcd for our ease.

THEORETICAL BACKGROUND:

MICROCONTROLLER(PIC16F877a): It has a total number of 40 pins and there are 33 pins for input and output and two 8 bit and one 16 bit timer. Capture and compare modules, serial ports, parallel ports and five input/output ports are also present in it.
It has a smaller **35** instructions set**.** It can operate up to 20MHz frequency**.** The operating voltage is between 4.2 volts to 5.5 volts. If you provide it voltage more than 5.5 volts, it may get damaged permanently. It does not have an internal oscillator.

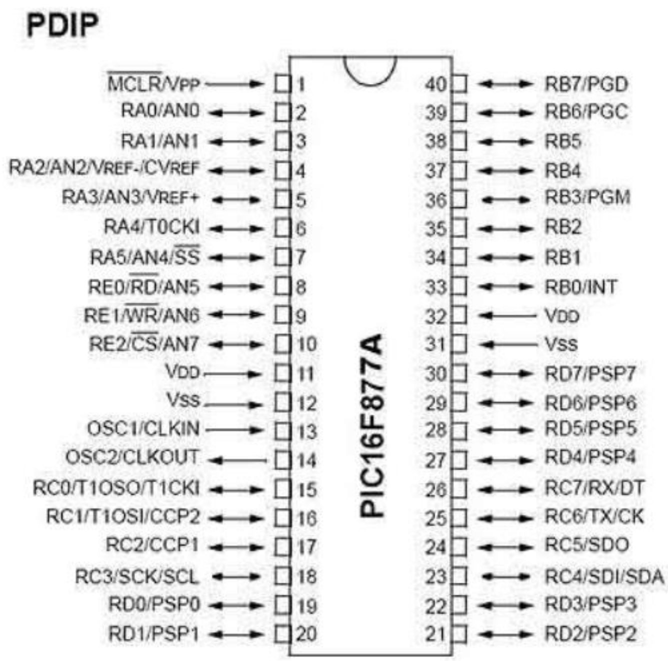
Pin Configuration:

PIN1:MCLR: he first pin is the master clear pin of this IC. It resets the microcontroller and is active low, meaning that it should constantly be given a voltage of 5V and if 0 V are given then the controller is reset. Resetting the controller will bring it back to the first line of the program that has been burned into the IC.

PIN2:RA0/AN0: PORTA consists of 6 pins, from pin 2 to pin 7, all of these are bidirectional input/output pins. Pin 2 is the first pin of this port. This pin can also be used as an analog pin AN0. It is built in analog to digital converter.

PIN3:RA1/AN1: This can be the analog input 1.

PIN4:RA2/AN2/Vref-: It can also act as the analog input2. Or negative analog reference voltage can be given to it.



PIN5:RA3/AN3/Vref+: It can act as the analog input 3. Or can act as the analog positive reference voltage

PIN6:RA0/T0CKI: To timer0 this pin can act as the clock input pin, the type of output is open drain.

PIN7:RA5/SS/AN4: This can be the analog input 4. There is synchronous serial port in the controller also and this pin can be used as the slave select for that port.

PIN 8: RE0/RD/AN5: PORTE starts from pin 8 to pin 10 and this is also a bidirectional input output port. It can be the analog input 5 or for parallel slave port it can act as a ‘read control’ pin which will be active low.

PIN 9: RE1/WR/AN6: It can be the analog input 6. And for the parallel slave port it can act as the ‘write control’ which will be active low.

PIN 10: RE2/CS/A7: It can be the analog input 7, or for the parallel slave port it can act as the ‘control select’ which will also be active low just like read and write control pins.

PIN 11 and 32: VDD: These two pins are the positive supply for the input/output and logic pins. Both of them should be connected to 5V.

PIN 12 and 31: VSS: These pins are the ground reference for input/output and logic pins. They should be connected to 0 potential.

PIN 13: OSC1/CLKIN: This is the oscillator input or the external clock input pin.

PIN 14: OSC2/CLKOUT: This is the oscillator output pin. A crystal resonator is connected between pin 13 and 14 to provide external clock to the microcontroller. ¼ of the frequency of OSC1 is outputted by OSC2 in case of RC mode. This indicates the instruction cycle rate.

PIN 15: RC0/T1OCO/T1CKI: PORTC consists of 8 pins. It is also a bidirectional input output port. Of them, pin 15 is the first. It can be the clock input of timer 1 or the oscillator output of timer 2.

PIN 16: RC1/T1OSI/CCP2: It can be the oscillator input of timer 1 or the capture 2 input/compare 2 output/ PWM 2 output.

PIN 17: RC2/CCP1: It can be the capture 1 input/ compare 1 output/ PWM 1 output.

PIN 18: RC3/SCK/SCL: It can be the output for SPI or I2C modes and can be the input/output for synchronous serial clock.

PIN 23: RC4/SDI/SDA: It can be the SPI data in pin. Or in I2C mode it can be data input/output pin.

PIN 24: RC5/SDO: It can be the data out of SPI in the SPI mode.

PIN 25: RC6/TX/CK: It can be the synchronous clock or USART Asynchronous transmit pin.

PIN 26: RC7/RX/DT: It can be the synchronous data pin or the USART receive pin.

PIN 19,20,21,22,27,28,29,30: All of these pins belong to PORTD which is again a bidirectional input and output port. When the microprocessor bus is to be interfaced, it can act as the parallel slave port.

PIN 33-40: PORT B: All these pins belong to PORTB. Out of which RB0 can be used as the external interrupt pin and RB6 and RB7 can be used as in-circuit debugger pins.

Main Features:

Like all other microcontroller, PIC16F877A also provide built-in useful features as mentioned in this list:

Analog to digital converter module : It has 8 bit ADC module which consists of 8 channels. We can use 8 analog sensors with this microcontroller.

Timers: It provides three timers timer0, timer1 and timer2. All these timers can be used either in timer mode or in counter mode. These timers are used to generate delays, pulse width modulation, counting external events and timer interrupts. TIMER0 is a 8 bit timer and it can operate with internal or external clock frequency. When we use Timer0 in timer mode, we usually operate it with internal frequency and in counter mode, we trigger it with external clock source. Similarly, TIMER1 is a 16-bit timer and it can also operate in both modes. TIMER2 is also of 8-bit. It is used with PWM as a time base for CCP module.

EEPROM : It also has built-in Electrically erasable read only memory 256 x 8 bytes which can used to store data permanently even if the microcontroller is switched off, data will remain there.

PWM modules : It also provide 2 CCP modules. CCP stands for capture compare PWM modules. We can easily generate two PWM signals with this microcontroller. The maximum resolution it supports is 10 bits.

Serial or UART communication pins : It support one UART channel. UART pins are used for serial communication between digital devices. RC7 pin is a transmitter or RX pin which is pin number 26. RC6 is a receiver or Tx pin which is pin number 25

I2C Communication : PIC16F877A also support I2C communication and its has one module for I2C communication. Pin#18/RC3 and 23/RC4 are **SCL** and **SDA** pins respectively. SCL is a serial clock line and SDA is serial data line.

Interrupts : Interrupts have wonderful applications in embedded systems field. If you don’t know about interrupts, I suggest you to get complete understanding about them, you will not get command on embedded programming them. PIC16F877A microcontroller provides 8 types of interrupts namley; External interrupts, timer interrupts, PORT state change interrupts, UART interrupt, I2C, PWM interrupts

Comparator module : It has a comparator module which composed of two comparators. They are used for comparison of analog signal similar to comparators in electronics circuits. Input pins for these comparators are RA0, RA1, RA2 and RA3 and output can measured through RA4 and RA5.

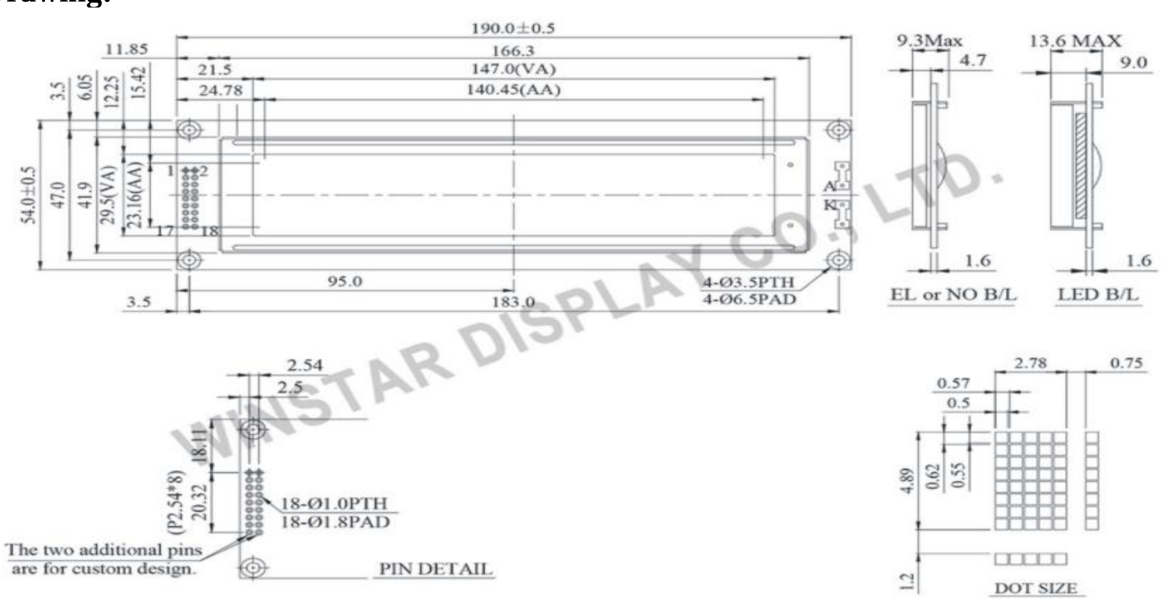
Watchdog timer : WDT is a on chip separate oscillator which runs freely. It is a separate oscillator from OSC1/CLKI. WDT will also work even if the device is in sleep mode. It is used to wake up device from sleep mode and also used to generate watchdog timer reset.

Sleep mode : PIC16F877A also provide sleep mode operation. In this mode, device operates at very low power. All peripherals draws minimum amount of current. Wake up from sleep mode from interrupts resources like timer1 interrupt, uart interrupt, EEPROM write completion operation and many others.

Brown out reset : This option reset the device upon detection of brown out interrupt signal from BODEN signal. if supply voltage goes below threshold for more than 100 micro seconds,

<u>40X4 LCD:</u> 40x4 character LCD display run by 40 characters x 4 lines. It is available in different models . it has normally 5V or 3.3V power supply option.

Drawing:



Datasheet:

Interface pin function:

Pin No.	Symbol	Description
1~8	DB7~DB0	Data bus line
9	E1	Enable signal
10	R/W	Read/Write select signal
11	RS	Data/ Instruction select signal
12	V_O	Contrast Adjustment
13	V_{SS}	Ground
14	V_{DD}	Power supply for logic
15	E2	Enable signal
16	NC	No connection
17	A	Power supply for B/L +
18	K	Power supply for B/L -

Mechanical Data:

Item	Standard Value	Unit
Module Dimension	190.0 × 54.0	mm
Viewing Area	147.0 × 29.5	mm
Mounting Hole	183.0 × 47.0	mm
Character Size	2.78 × 4.89	mm

Display character Address code:

Display Position	1	2	3	4	5	6	---	---	40	
DD RAM Address	00	1							27	Lin1
DD RAM Address	40	41							67	Lin2
DD RAM Address	00	01							27	Lin3
DD RAM Address	40	41							67	Lin4

4X4 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. Here we are using **4 X 4 matrix keypad**.

Interfacing Keypad with PIC16F877a:

In case of 4X4 matrix Keypad both the ends of switches are connected to the port pin i.e. four rows and four columns. So in all sixteen switches have been interfaced using just eight lines.Keypads arranged by matrix format, each row and column section pulled by high or low by selection J15, all row lines(PORTB.0 – PORTB.3) and column lines(PORTB.4 to PORTB.7) connected directly by the port pins

Pin Assignment with PIC16F877a:

	4X4 MATRIX LINES	PIC16F LINES	4X4 MATRIX KEYPAD
ROW	ROW-0 ROW-1 ROW-2 ROW-4	PORTB-0 PORTB-1 PORTB-2 PORTB-3	Connect PortB with JP15 Turn on TXD and RXD pins Connect serial cables b/w USART in the board and PC and open Hyper terminal
COLUMN	COLUMN-0 COLUMN-1 COLUMN-2 COLUMN-3	PORTB-4 PORTB-5 PORT-6 PORT-7	Press reset once Outputs Press key and number will display in hyper terminal.

METHODOLOGY:

In the main code, keypad and LCD are initialized at first. Then the code waits for the first number from the keypad. After getting this number LCD screen is cleared. And this number is displayed on the LCD. After that, code waits for the function key from the user. After getting the function key, code waits for the second number and then the equal sign. After getting the equal sign, according to the desired function the result is calculated and displayed on the screen.The clear key erases the current display, and a new calculation can be entered. 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. The clear operation sends a command to the display to clear the last set of characters.

SOFTWARE DESIGN:

PROGRAM(mikroC):

```
#include<htc.h>
#define_XTAL_FREQ 20e6
#define rs RD6
#define en RD7
#define r0 RC0
#define r1 RC1
#define r2 RC2
#define r3 RC3
#define r4 RC4
#define r5 RC5
#define r6 RC6
#define r7 RC7

void lcdcmd (unsigned char);
void lcddata (unsigned char);
void disp_num (float num); int
get_num(char ch);
void lcdinit ();
char scan_key(void);

unsigned char s[]={"ENTER 1 NO.= "};
unsigned char s1[]={"ENTER 2 NO.= "};
unsigned char s2[]={"Operator= "};
void lcdinit ()
{
      delay_ms(400);   lcdcmd(0x30);
      delay_ms(400);   lcdmd(0x30);
      delay_ms(400);   lcdcmd(0x30);
      delay_ms(400);

      lcdmd(0x38);
      lcdmd(0x0F);    lcdmd(0x01);
      lcdmd(0x06);     lcdmd(0x80);
}

void main (void)
{
  TRISC=0xF0;
  TRISB=0x00;
  TRISD6=0;
  TRISD7=0;
  delay_ms(400);
```

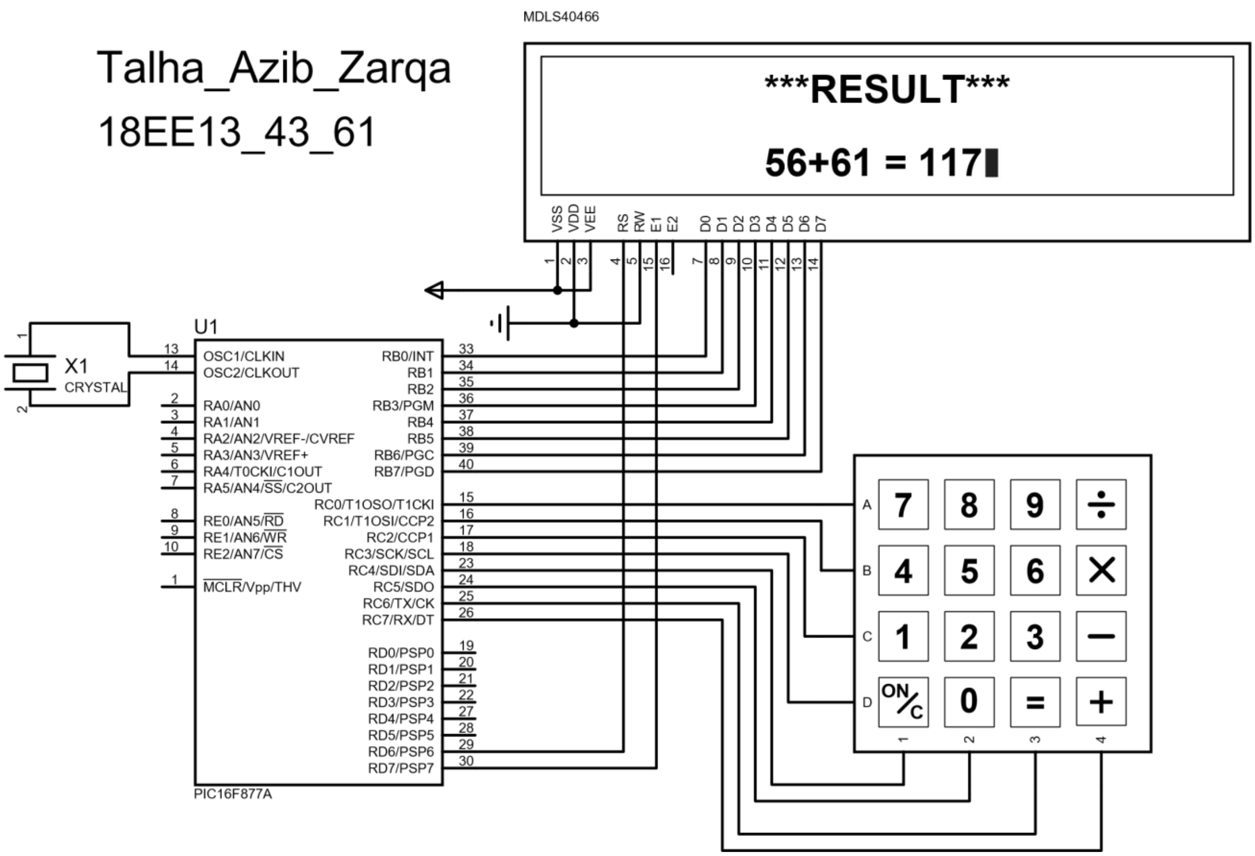
```
    unsigned incount=0;
int k2,k1;
      char ke,key,key1;
      lcdinit();
      While(1)
       {
        while(s[count] !=' \ 0')
                 {
           lcddata(s[count]);
count++;
 }
 Ke = scan_key();
 k2 = get_num(ke);
lcdcmd(0x01);
count=0;

while(s2[count] !=' \ 0')
   {
lcddata(s2[count]);
count++;
 }
Key = Scan_key();
lcdcmd(0x01);
 count=0;

while(s3[count] != '\ 0')
      {
lcddata(s3[count]);
count++;
      }
count=0;
lcdcmd(0xC0);
lcddata(ke);
lcddata(key);
lcddata(key2);
lcddata(' ');
lcddata('=');

 switch(key)
{
 case'+' ; disp_num(k1+k2);
break;
case '-'; disp_num(k1-k2);
break;
case '*'; disp_num(k1*k2);
break;
case '/'; disp_num(k1/k2);
break;
              }
        }
}
```

```
void lcdcmd(unsigned char value)
{
PORTB = value;
rs = 0;       en =
1;
delay_ms(200);
en = 0;
delay_ms(100);
}

void lcddata(unsigned char value)
{
  PORTB = value;
rs = 0;      en =
1;
delay_ms(200);
en = 0;
delay_ms(100);
}
char scan_key()
{
     unsigned char c= 's';
while(c!='a');           {
           r0 = 0; r1 = 1; r2 = 1; r3 = 1;
            if(c0 == 1 && r0 == 0) { (lcddata('7');   delay_ms(500);
return '7'; c='a'; }
            if(c0 == 1 && r0 == 0) { (lcddata('8');   delay_ms(500);
return '8'; c='a'; }
            if(c0 == 1 && r0 == 0) { (lcddata('9');   delay_ms(500);
return '9'; c='a'; }
            if(c0 == 1 && r0 == 0) { (lcddata('/');   delay_ms(500);
return ' '; c='a'; }
           r0 = 0; r1 = 1; r2 = 1; r3 = 1;
            if(c0 == 1 && r0 == 0) { (lcddata('4');   delay_ms(500);
return '4'; c='a'; }
            if(c0 == 1 && r0 == 0) { (lcddata('5');   delay_ms(500);
return '5'; c='a'; }
            if(c0 == 1 && r0 == 0) { (lcddata('6');   delay_ms(500);
return '6'; c='a'; }
            if(c0 == 1 && r0 == 0) { (lcddata('*');   delay_ms(500);
return '*'; c='a'; }
           r0 = 0; r1 = 1; r2 = 1; r3 = 1;
            if(c0 == 1 && r0 == 0) { (lcddata('1');   delay_ms(500);
return '1'; c='a'; }
            if(c0 == 1 && r0 == 0) { (lcddata('2');   delay_ms(500);
return '2'; c='a'; }
            if(c0 == 1 && r0 == 0) { (lcddata('3');   delay_ms(500);
return '3'; c='a'; }
            if(c0 == 1 && r0 == 0) { (lcddata('-');   delay_ms(500);
return '-'; c='a'; }
           r0 = 0; r1 = 1; r2 = 1; r3 = 1;
```

```
            if(c1 == 1 && r3 == 0) { (lcddata('0');   delay_ms(500);
return '0'; c='a'; }
            if(c3 == 1 && r3 == 0) { (lcddata('+');   delay_ms(500);
return '+'; c='a'; }
   }
return 0;
}

int get_num(char ch)
{
switch(ch)
{
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;
          }
return 0;
}

void disp num(float num)
{
unsigned char Unitdigit =0;
unsigned char Tenthdigit =0;
unsigned char decimal =0;

int j,numb;
int J = num*10;
int num = numb;

if(numb < 0)
 {        numb = 1*numb;
lcddata('-');
     }

   Tenthdigit = (numb/10);

   if(Tenthdigit !=0)
{
       lcddata(Tenthdigit+0x30);
      }
   Unitdigit = numb - (Tenthdigit*10);
Numb=(int)num;

 if(numb < 0)
```

```
{
numb = -1*numb;
 }
lcddata('-');
 Tenthdigit = (numb/10);

   if( Tenthdigit != 0)
     {
      lcddata(Tenthdigit+0x30);
     }

Unitdigit = numb - (Tenthdigit*10);
lcddata(Unitdigit+0x30);
lcddata('.');    decimal=(j%10)+0x30);
lcddata(decimal);    delay_ms(4000);
lcdcmd(0x01);
}
```

PROTEUS SIMULATION:



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(Pic16f877A) based calculator in which display is shown on 40x4 LCD for easy and speedy calculation. Results of the calculator were found to agree with the other calculators.

APPENDIX:

Some other momentous features of **PIC 16F877a** are listed below :

1. The maximum current each PORT can sink or source is around 100mA. Therefore, the current limit for each GPIO pin of PIC16F877A is 10 mili ampere.
2. It is available in four IC packaging such as 40-pin PDIP 44-pin PLCC, 44-pin TQFP, 44-pin QFN
3. Power on Reset
4. Multiple oscillator group
5. In-Circuit Debugger
6. In-Circuit Serial programming
7. Low voltage ICSP programming

REFRENCES:

- https://microcontrollerslab.com/pic16f877a-introduction-features/
- https://www.researchgate.net/publication/326493983_Design_and_Implementation_of_Microcontroller_Based_Calculator
- https://www.winstar.com.tw/products/character-lcd-display-module/40x4-lcd.html
- https://www.pantechsolutions.net/blog/how-to-interface-keypad-with-pic16f877a-pic-advanced-development-board/