ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INNOVACIÓN PARA LA EXCELENCIA

Departamento de Eléctrica y Electrónica

Carrera de Electrónica y Automatización

MICROPROCESADORES Y MICROCONTROLADORES

Práctica 3.10

UTILIZACIÓN DE LA MEMORIA EEPROM DE DATOS INTERNA DEL PIC16F877

Autor:

Iza Tipanluisa Alex Paul

Docente:

Ing. Amparo Meythaler

NRC: 4891

1) OBJETIVOS

- Identificar la importancia de una memoria EEPROM de datos.
- Realizar una aplicación con la memoria EEPROM Interna, en lenguaje C, para elPIC16F877.

2) MARCO TEORICO

MEMORIA EEPROM DE DATOS INTERNA

El microcontrolador tiene una memoria EEPROM interna de 256 bytes.

La memoria EEPROM es útil para almacenar datos que, por su importancia, se quieren conservar almacenados en forma segura, a pesar de cortes de energía o ruido inducido en los circuitos del microcontrolador, que en un momento dado podrían destruir estos datos en caso de encontrarse almacenados en una memoria RAM.

A esta memoria se la puede leer o escribir en funcionamiento, a diferencia de las memorias de programa a las que se accede sólo el rato de programar el Microcontrolador.

3) EQUIPOS Y MATERIALES

- PC con el paquete PIC C.
- PC con el paquete PROTEUS.

4) ACTIVIDADES

1) Trabajo Preparatorio:

Realizar el diagrama de flujo, la codificación e implementación correspondiente, de un programa que solicite una contraseña por medio del teclado y se visualice en un LCD sólo como asteriscos (*). El usuario pueda modificar la contraseña de acceso, pero para que haga esto, el programa deberá validar que el usuario conozcala contraseña actual.

(revisar el link https://controlautomaticoeducacion.com/microcontroladores-pic/12-almacenamiento-en-memoria-interna-eeprom/).

1) Trabajo en el paquete que maneja lenguaje C.

- Digite el programa. (Ponga al inicio como un comentario su nombre).
- Compile el ejercicio hasta que obtenga 0 errores (presente una captura delpaquete).

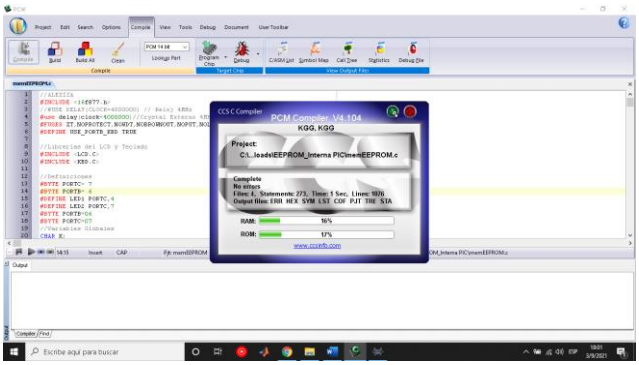

Ilustración 1, Compilación del programa

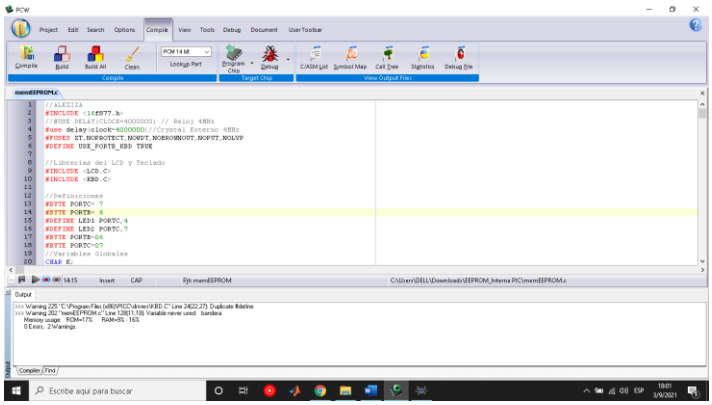

Ilustración 2, Compilación exitosa del programa

2) Trabajo en el paquete PROTEUS.

- Realice el diagrama esquemático.
- Cargue el programa compilado en el microcontrolador.

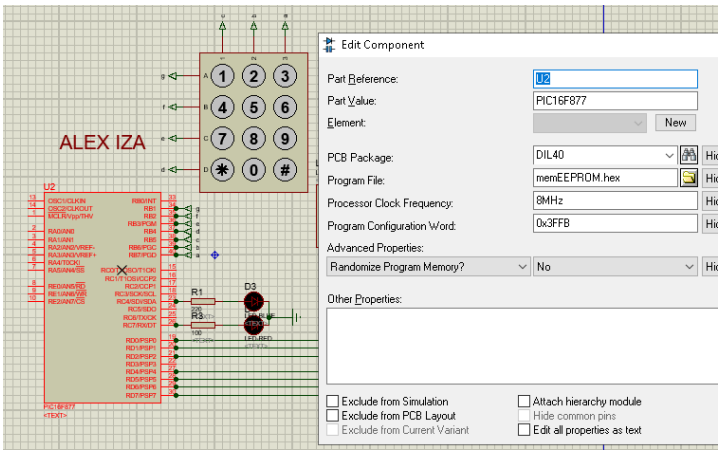

Ilustración 3, Programa cargado en el Pic

5) RESULTADOS

- Explique los errores cometidos en el ejercicio realizado (si los tuvo) y la forma decorregirlos.

  Tuve Problemas en el simulador ya que en el mismo no se estaba mostrando ningún texto en el LCD ya que un comando estaba impidiendo que se pueda visualizar, lo solucione eliminando la línea #DEFINE USE_PORTB_LCD TRUE

- Explique la forma de verificar en PROTEUS un dato guardado en la MemoriaInterna del PIC16F877.

  Podemos verificarlo realizando el guardado del dato y procediendo a reiniciar el corrido del programa en proteus, después se debe ingresar el mismo dato en el teclado y este debe verificarse siendo el mismo dando como resultado que el dato ha sido guardado.

  Otra manera de verificar ese dato es cargando el programa con la extensión ".cof" en el Pic el mismo que antes de correr el programa se debe presionar pausar, por lo que el programa desplegara 2 ventanas; un del código que se cargó y el otro es los valores de las variables que se van guardando, en esta se debe visualizar que la variable contra se cambie de defecto que es 9999 a la contraseña nueva.

6) DISEÑO

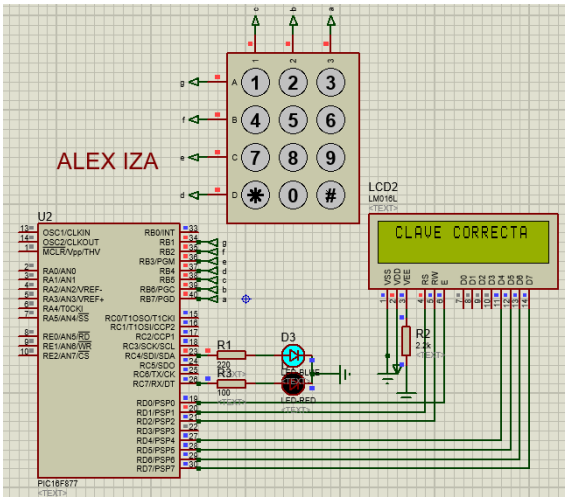

Ilustración 4, La clave ingresada es correcta

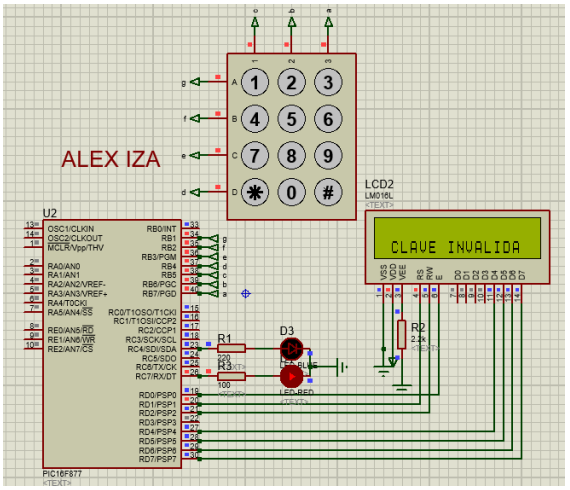

Ilustración 5, La clave ingresada es incorrecta

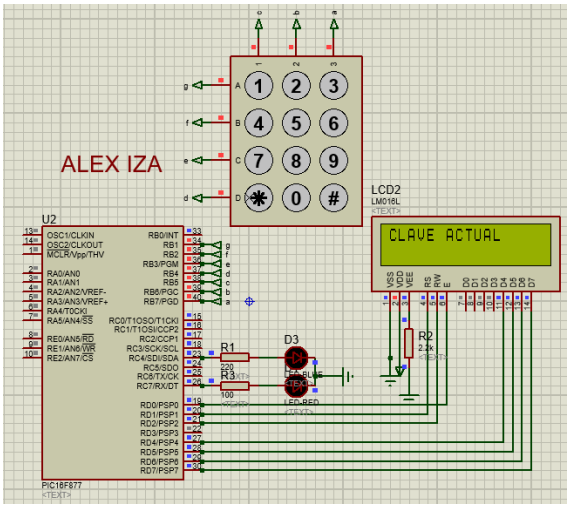

Ilustración 6, Petición de la Clave Actual

7) DIAGRAMA DE FLUJO

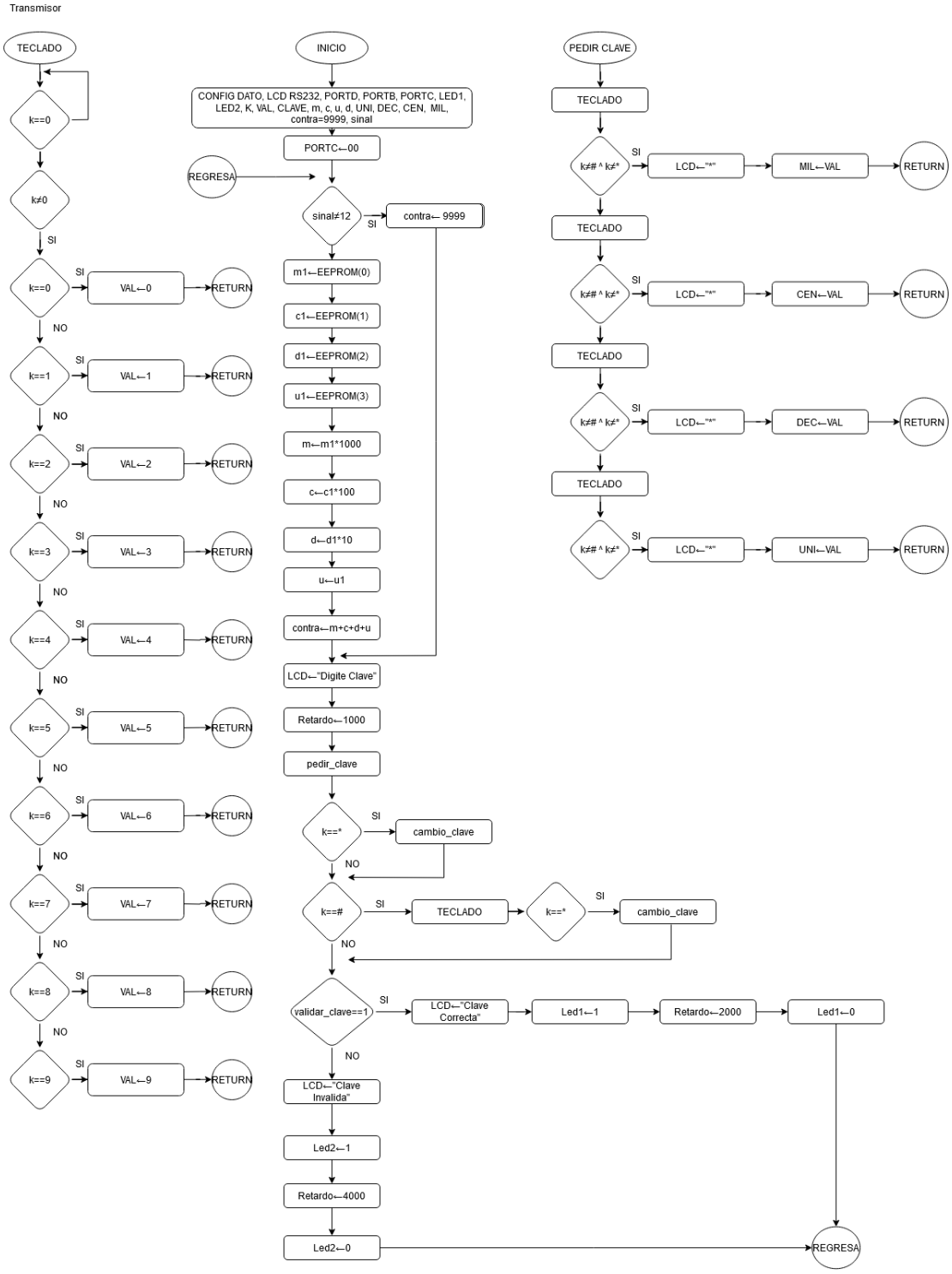

Diagrama 1

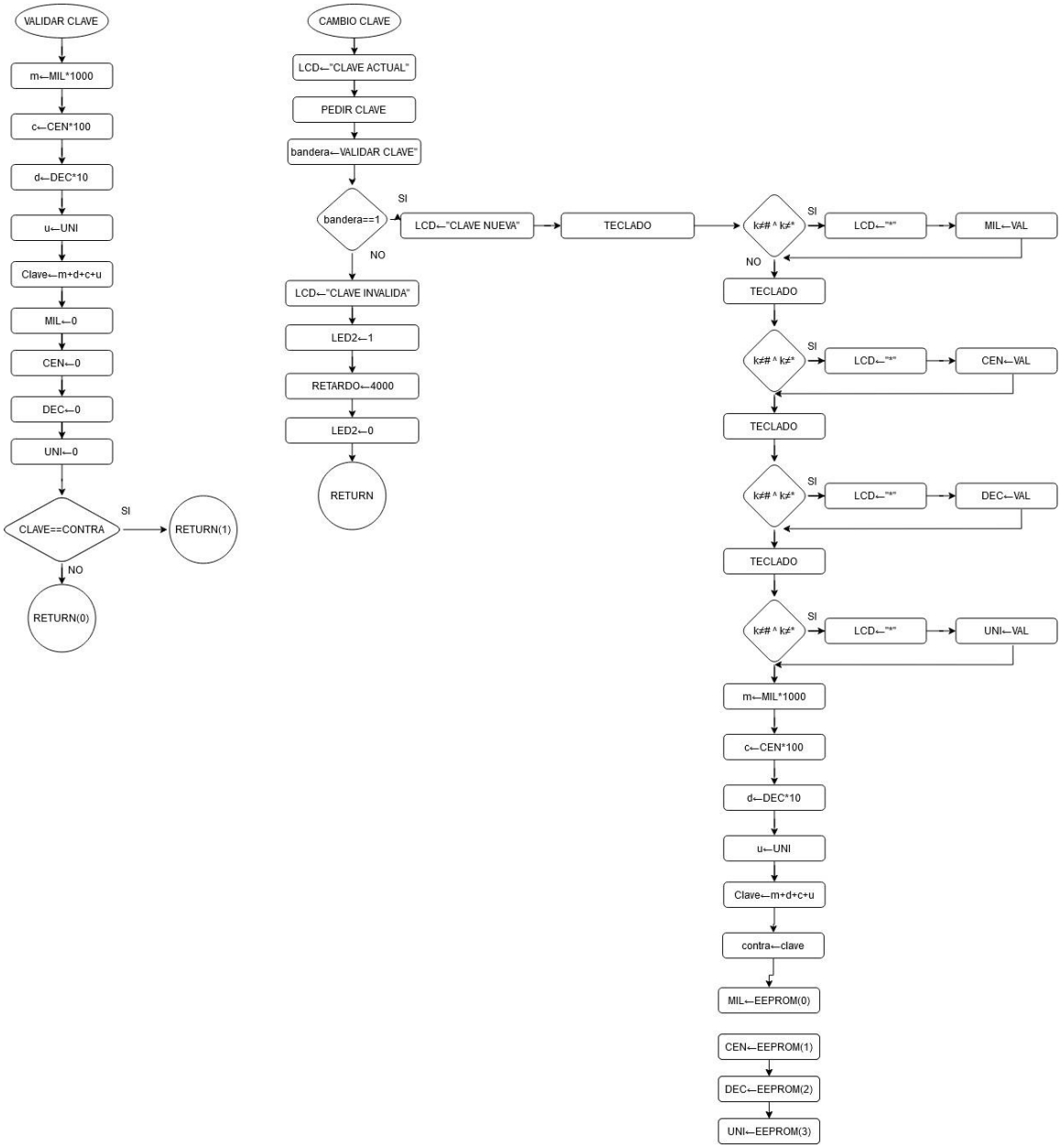

Diagrama 2

8) PROGRAMA

```
//ALEXIZA
#INCLUDE <16f877.h>
//#USE DELAY(CLOCK=4000000) // Reloj 4MHz
#use delay(clock=4000000)//Crystal Externo 4MHz
#FUSES XT,NOPROTECT,NOWDT,NOBROWNOUT,NOPUT,NOLVP
#DEFINE USE_PORTB_KBD TRUE
```

```
//Librerías del LCD y Teclado
#INCLUDE <LCD.C>
#INCLUDE <KBD.C>

//Definiciones
#BYTE PORTC= 7
#BYTE PORTB= 6
#DEFINE LED1 PORTC,4
#DEFINE LED2 PORTC,7
#BYTE PORTB=06
#BYTE PORTC=07
//Variables Globales
CHAR K;
INT8 MIL,CEN,DEC,UNI,VAL;
INT16 contra=9999; //Variable entera para almacenar la contraseña
INT8 sinal; //Variable para saber si tengo contraseña

VOID TECLADO()
{
  k=kbd_getc(); //Llamar la función del teclado y almacenar
  //el valor digitado en una variable tipo
  //carácter. Si no se oprime ninguna tecla el
  //teclado retornara el carácter nulo.*/
  WHILE(k=='\0') //si no se oprime ninguna tecla sigue llamando al teclado.
  {
    k=kbd_getc();
  }
  IF( (k!='\0'))
```

```
  {
    IF(K=='0')//Si K es igual a cero
      VAL=0;//Val es igual a cero
    IF(K=='1')
      VAL=1;
    IF(K=='2')
      VAL=2;
    IF(K=='3')
      VAL=3;
    IF(K=='4')
      VAL=4;
    IF(K=='5')
      VAL=5;
    IF(K=='6')
      VAL=6;
    IF(K=='7')
      VAL=7;
    IF(K=='8')
      VAL=8;
    IF(K=='9')
      VAL=9;
  }
}

//Rutina para pedir la clave
void pedir_clave(void)
{
    TECLADO();
```

```
IF((k!='#')&&(k!='*'))
{
  lcd_putc('*');
  MIL=VAL;
}
if(k=='*' || k=='#')
  return;
TECLADO();
IF((k!='#')&&(k!='*'))
{
  lcd_putc('*');
  CEN=VAL;
}
if(k=='*' || k=='#')
  return;
TECLADO();
IF((k!='#')&&(k!='*'))
{
  lcd_putc('*');
  DEC=VAL;
}
if(k=='*' || k=='#')
  return;
TECLADO();
IF((k!='#')&&(k!='*'))
{
  lcd_putc('*');
  UNI=VAL;
}
```

```
    if(k=='*' || k=='#')
      return;

}

int validar_clave(void)

{
  //Variables locales
  int16 clave=0,m,c,d,u;

  /* Para realizar la multiplicación cuando se tienen numeros del tipo int8,
    como por ejemplo 100 con otro tipo int8 como por ejemplo CEN, el programa
    hará un producto int8, sin embargo se desea un resultado int16, por lo
    tanto debe informarce al compilador por medio de un 'cast' (c=(int16)CEN*100)
    o utilizando la función de multiplicación de tipos del compilador (_mul)*/

  m=MIL*1000; //Convierto miles a numero
  //c=_mul(CEN,100); //Otra alternativa para multiplicar int8*int8=int16
  c=(int16)CEN*100; //Convierto centemas, haciendo un CAST para obtener un int16
  d=DEC*10; //Convierto decenas a numero
  u=UNI; //Convierto unidades a numero

  clave=m+c+d+u;
  //Borra lo último que fue digitado en el teclado
  MIL=0;CEN=0;DEC=0;UNI=0;
  if(clave==contra) //Si la clave es igual a la contraseña
    return(1);
```

```
  else
     return(0);
}

void cambio_clave(void)
{
     int bandera=0;
     int16 clave=0,m,c,d,u;

     LCD_PUTC("\f");
     LCD_GOTOXY(1,1);
     LCD_PUTC("CLAVE ACTUAL");
     LCD_GOTOXY(1,2);
     pedir_clave(); //Llama la funcion de pedir la clave
     //bandera=validar_clave(); //Compruebo si la clave actual es correcta
     if(validar_clave())
     {
       LCD_PUTC("\f");
       LCD_GOTOXY(1,1);
       LCD_PUTC("CLAVE NUEVA  ");
       LCD_GOTOXY(1,2);
       TECLADO();
       IF((k!='#')&&(k!='*'))
       {
         lcd_putc('*');
         MIL=VAL;
       }

       TECLADO();
```

```
IF((k!='#')&&(k!='*'))
{
  lcd_putc('*');
  CEN=VAL;
}

TECLADO();
IF((k!='#')&&(k!='*'))
{
  lcd_putc('*');
  DEC=VAL;
}

TECLADO();
IF((k!='#')&&(k!='*'))
{
  lcd_putc('*');
  UNI=VAL;
}

m=MIL*1000; //Convierto miles a numero
c=CEN*10; //Convierto centemas a numero y lo sumo al anterior
c=c*10;
d=DEC*10; //Convierto decenas a numero y lo sumo al anterior
u=UNI; //Convierto unidades a numero y lo sumo al anterior

clave=m+c+d+u;

contra=clave;
```

```
      WRITE_EEPROM(0,MIL); //Guarda en la eemprom posicion cero la nueva contraseña
      WRITE_EEPROM(1,CEN);
      WRITE_EEPROM(2,DEC);
      WRITE_EEPROM(3,UNI);
      WRITE_EEPROM(4,12); //Guardo un 12 en la posicion 1 de la EEPROM, para decir que tengo
                          //una contraseña guardada

    }
    else
    {
      LCD_PUTC("\f");
      LCD_GOTOXY(1,2);
      LCD_PUTC(" CLAVE INVALIDA ");
      BIT_SET(LED2);
      DELAY_MS(4000);
      BIT_CLEAR(LED2);
      LCD_PUTC("\f");
    }
}
VOID MAIN()
{
  //Variables Locales
  int16 m,c,d,u;
  INT8 m1,c1,d1,u1;

  //Puerto C como Salida
  SET_TRIS_C(0B00000000);
  PORTC=0; //Limpia Puerto C
```

```
  lcd_init(); //Inicializa el LCD
  kbd_init(); //Inicializa el Teclado

  //Configura las Resistencias PULL-UP del Puerto B
  port_b_pullups(0xFF); //PIC16F887
  //port_b_pullups(TRUE); //PIC16F877A

  sinal=READ_EEPROM(4); //Averiguo si tengo una contraseña guardada o no
  if(sinal!=12)
    contra=9999;
  else
  {
   //Lee los datos del EEPROM
    m1=READ_EEPROM(0);
    c1=READ_EEPROM(1);
    d1=READ_EEPROM(2);
    u1=READ_EEPROM(3);

  /* Para realizar la multiplicación cuando se tienen numeros del tipo int8,
     como por ejemplo 100 con otro tipo int8 como por ejemplo CEN, el programa
     hará un producto int8, sin embargo se desea un resultado int16, por lo
     tanto debe informarce al compilador por medio de un 'cast' (c=(int16)CEN*100)
     o utilizando la función de multiplicación de tipos del compilador (_mul)*/

     m=m1*1000; //Convierto miles a numero
     //c=_mul(c1,100); //Otra alternativa para multiplicar int8*int8=int16
     c=(int16)c1*100; //Convierto centemas, haciendo un CAST para obtener un int16
```

```
  d=d1*10; //Convierto decenas a numero
  u=u1; //Convierto unidades a numero
      contra=m+c+d+u;
} WHILE(TRUE)
{
  LCD_PUTC("\f");
  LCD_GOTOXY(1,1);
  LCD_PUTC(" DIGITE CLAVE ");
  LCD_GOTOXY(1,2);
  DELAY_MS(1000);
  pedir_clave(); //Llama la funcion de pedir la clave
  if(k=='*')
    cambio_clave();
  else
  {
    WHILE((k!='#'))
    {
      TECLADO();
      if(k=='*')
        cambio_clave();
    }
    IF(validar_clave()) //Aquí se compara si
    //los números digitados
    //están correctos.*/
    {
      LCD_PUTC("\f"); //Se borra LCD
      LCD_GOTOXY(1,1);//Se ubica en la posición 1,1
      LCD_PUTC(" CLAVE CORRECTA ");
      BIT_SET(LED1);
```

```
    DELAY_MS(2000);
    BIT_CLEAR(LED1);
  }      ELSE      {
    LCD_PUTC("\f");
    LCD_GOTOXY(1,2);
    LCD_PUTC(" CLAVE INVALIDA ");
    BIT_SET(LED2);
    DELAY_MS(4000);
    BIT_CLEAR(LED2);
  }    }  }}
```

9) CONCLUSIONES

- Se ha concluido que los datos almacenados en el PIC16F877 no se borraran a menos que se sobrescriba sobre ellos.
- Se ha concluido que la instrucción para almacenar información en la memoria EEPROM interna es: WRITE_EEPROM(dirección, valor)
- Se ha concluido que la instrucción para leer información en la memoria EEPROMinterna es: READ_EEPROM(dirección)

10) RECOMENDACIONES

- Se recomienda usar todas las localidades de memoria (256) para guardar los datos o sobrescribirlos ya que permite escribir al menos 100000 en una de las celdas de la memoria.
- Se recomienda que cuando tenemos un int16, se debe dividir en 2 datos de 8 bits con make8(), de esta manera dividimos los 16 bits en 8 bits en una parte alta y una parte baja.
- Se recomienda cargar en el proteus el programa con extensión ".cof", para que se pueda ver los valores que van tomando las variables, así como si se olvida la contraseña, en este mismo se puede ver cual es la contraseña.

11) BIBLIOGRAFIA

Meythaler, A. (2021). Sistemas basados en MCU. Ecuador: UFA ESP