**ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL**

**FIEC**

**Laboratorio de Microcontroladores**

**Proyecto # 2**

Medidor de potenciómetro y   
  
Adivinador de contraseña

**Alumno**:

Geovanny Arguello A.

**Paralelo #** **9**

**Grupo: # 3**

**Fecha de presentación:**

22 Julio 2012

**2012 – 1° TÉRMINO**

**PROYECTO 2**

**OBJETIVO:**

* Lograr la lectura de los valores analógicos de un potenciómetro, presentando sus variaciones de valores en binario y su equivalente decimal en una pantalla LCD 4x20.
* Adivinar la secuencia o contraseña de una entrada mediante el uso de un teclado 4x4 y una pantalla LCD para observar los valores entrantes.

**1.- Enunciado del proyecto**

En este proyecto se utilizará el PIC 16F887, para realizar 2 acciones, las cuales serán seleccionadas por 2 pulsadores; para la primera opción se pretende observar los valores que marca un potenciómetro en una pantalla LCD, tanto el valor en decimal como en binario, y también si se cambia el valor del potenciómetro, se podrán observar las variaciones en la misma pantalla, adicionalmente se podrá observar las iniciales del alumno en displays de 7 segmentos. Para la segunda parte consiste en un adivinador de contraseña la cual es escrita por un teclado 4x4, y si se acierta en la adivinación, se enciende un LED verde, caso contrario uno rojo.

**2.- Diagrama de Bloques**



**3.- Diagrama de Flujo funcional del Programa principal y de las subrutinas**

***Programa Principal:***



***Subrutinas***

**4.- Descripción del algoritmo o estrategia utilizada.**

1. En este programa utilizaremos el oscilador de 8MHz.
2. Se debe configurar el PIC16F887 de tal manera que todos los bits del puerto A sean salidas digitales, mientras que los de los puertos B sean acoplados a los pines del LCD, los bits del puerto C y D sean entradas digitales y finalmente el puerto E como entrada digitales.
3. Luego se programa el menú que será presentado en la pantalla LCD.
4. Luego se configura las diferentes opciones que se llevan a cabo luego de la pulsación del botón correspondiente.
5. Si se pulsa el botón 1 se realiza la opción del potenciómetro el cual lee el valor analógico del potenciómetro y lo lleva a un valor digital para ser mostrado en la pantalla LCD, tanto su equivalente decimal, como su equivalente en binario; si existiesen variaciones en el potenciómetro, los valores mostrados también cambian en tiempo real.
6. Si se pulsa el botón 2 se realiza la opción del juego, en la cual se pretende adivinar una contraseña de 4 números inscritos por teclado.
7. Si se acierta se encenderá un led verde, caso contrario un led rojo.
8. Luego de esto se regresa al menú principal para volver a elegir alguna opción.
9. El proceso se repite hasta acabar la simulación.

**5.- Listado del programa fuente en lenguaje ensamblador con comentarios en las líneas de código que considere fundamentales**

/\*

\* Nombre del Proyecto:

Proyecto\_2.c

\* Nombre del Autor:

(c) Mikroelektronika, 2009.

\* Description:

(Explicación del ejercicio)

1. Ejercicio en lenguaje C propuesto por el profesor

que consiste en la lectura de los valores analógicos

de un potenciómetro colocado en patita RE0/AN5 del

PIC16F887, presentando las variaciones de valores en

binario y su equivalente decimal en una pantalla LCD 4x20.

Para la parte del juego se pretende adivinar una contraseña con

la ayuda de un teclado 4x4 y 2 leds de aviso de acierto o fallo.

\* Test configuration:

MCU: PIC16F887

Oscillator: HS, 08.0000 MHz

SW: mikroC PRO for PIC

\* NOTES:

\*/

unsigned int temp\_res, sub, digit1 , digit10, pot, temp;

unsigned int p0, p1, p2, p3, p4, p5, p6, p7,cnt=0;

unsigned short digit,digit\_no=0;

char i;

//ldc module connections

sbit LCD\_RS at RB4\_bit;

sbit LCD\_EN at RB5\_bit;

sbit LCD\_D4 at RB0\_bit;

sbit LCD\_D5 at RB1\_bit;

sbit LCD\_D6 at RB2\_bit;

sbit LCD\_D7 at RB3\_bit;

sbit LCD\_RS\_Direction at TRISB4\_bit;

sbit LCD\_EN\_Direction at TRISB5\_bit;

sbit LCD\_D4\_Direction at TRISB0\_bit;

sbit LCD\_D5\_Direction at TRISB1\_bit;

sbit LCD\_D6\_Direction at TRISB2\_bit;

sbit LCD\_D7\_Direction at TRISB3\_bit;

// End LCD module connections

char txt[20];

char txt1[] = "PROYECTO 2";

char txt2[] = "MENU INICIAL";

char txt3[] = "1.- POTENCIOMETRO";

char txt4[] = "2.- JUEGO";

char temp2[20];

char bina[20];

unsigned short kp;

// Keypad module connections

char keypadPort at PORTD;

// End Keypad module connections

void main() {

abc:

ANSEL = 0x20; // Configure AN5 pin as analog

ANSELH = 0; // Configure other AN pins as digital I/O

C1ON\_bit = 0; // Disable comparators

C2ON\_bit = 0;

TRISA = 0x00; // PORTA is output

TRISB = 0xFF; // PORTB is input

TRISC = 0x00; // PORTC is output

TRISE = 0x07; // PORTE is input

Lcd\_Init(); // Initialize LCD

PORTC = 0;

Lcd\_Cmd(\_LCD\_CLEAR); // Clear display

Lcd\_Cmd(\_LCD\_CURSOR\_OFF); // Cursor off

i=0;

do {

Lcd\_Out(1,6,txt1); // Write text in first row

Lcd\_Out(2,6,txt2); // Write text in second row

Lcd\_Out(3,1,txt3); // Write text in third row

Lcd\_Out(4,1,txt4); // Write text in fourth row

Delay\_ms(200);

} while((RE1\_bit) && (RE2\_bit)); // Endless loop

Lcd\_Cmd(\_LCD\_CLEAR);

Delay\_ms(200);

if(!(RE2\_bit))

{

cnt=0;

Keypad\_Init(); // Initialize Keypad

Lcd\_Init(); // Initialize Lcd

Lcd\_Cmd(\_LCD\_CLEAR); // Clear display

Lcd\_Cmd(\_LCD\_CURSOR\_OFF); // Cursor off

Lcd\_Out(1, 1, "PASSWORD :"); // Write message text on Lcd

// Wait for key to be pressed and released

for(i=0;i<4;i++)

{

kp = 0;

do

{

kp = Keypad\_Key\_Click(); // Store key code in kp variable

}while (!kp);

// Prepare value for output, transform key to it's ASCII value

switch (kp)

{

case 1: kp = 55; break; //49 es el código ASCII del número 1.

case 2: kp = 56; break; //50 es el código ASCII del número 2.

case 3: kp = 57; break; //51 es el código ASCII del número 3.

case 4: kp = 65; break; // A

case 5: kp = 52; break; // 4

case 6: kp = 53; break; // 5

case 7: kp = 54; break; // 6

case 8: kp = 66; break; // B

case 9: kp = 49; break; // 7

case 10: kp = 50; break; // 8

case 11: kp = 51; break; // 9

case 12: kp = 67; break; // C

case 13: kp = 42; break; // \*

case 14: kp = 48; break; // 0

case 15: kp = 35; break; // #

case 16: kp = 68; break; // D

}

if(i==0)

p0=kp;

if(i==1)

p1=kp;

if(i==2)

p2=kp;

if(i==3)

p3=kp;

Lcd\_Chr(2, 2+i, kp); // Print key ASCII value on Lcd

}

Delay\_ms(200);

Lcd\_Cmd(\_LCD\_CLEAR); // Clear display

Lcd\_Cmd(\_LCD\_CURSOR\_OFF); // Cursor off

Lcd\_Out(1, 1, "PASSWORD :"); // Write message text on Lcd

for(i=0;i<4;i++)

{

kp = 0;

do{

kp = Keypad\_Key\_Click(); // Store key code in kp variable

}while (!kp);

// Prepare value for output, transform key to it's ASCII value

switch (kp)

{

case 1: kp = 55; break; //49 es el código ASCII del número 1.

case 2: kp = 56; break; //50 es el código ASCII del número 2.

case 3: kp = 57; break; //51 es el código ASCII del número 3.

case 4: kp = 65; break; // A

case 5: kp = 52; break; // 4

case 6: kp = 53; break; // 5

case 7: kp = 54; break; // 6

case 8: kp = 66; break; // B

case 9: kp = 49; break; // 7

case 10: kp = 50; break; // 8

case 11: kp = 51; break; // 9

case 12: kp = 67; break; // C

case 13: kp = 42; break; // \*

case 14: kp = 48; break; // 0

case 15: kp = 35; break; // #

case 16: kp = 68; break; // D

}

if(i==0)

{

p4=kp;

if(p0==p4)

cnt=cnt+1;

}

if(i==1)

{

p5=kp;

if(p1==p5)

cnt=cnt+1;

}

if(i==2)

{

p6=kp;

if(p2==p6)

cnt=cnt+1;

}

if(i==3)

{

p7=kp;

if(p3==p7)

cnt=cnt+1;

}

Lcd\_Chr(2, 2+i, kp); // Print key ASCII value on Lcd

}

do{

if(cnt==4)

RC6\_bit=1;

else

RC7\_bit=1;

Delay\_ms(200);

}while(!(RE2\_bit));

goto abc;

}

if (!(RE1\_bit)) {

do{

temp\_res = ADC\_Read(5); // Get 10-bit results of AD conversion

temp=temp\_res;

for(i=0;i<10;i++)

{

pot=(temp%2);

IntTostr(pot,bina);

Lcd\_Out(4,(10-i),bina);

temp=temp/2;

}

IntTostr(temp\_res,temp2);

Lcd\_Out(1,1,txt3); // Write text in first row

Lcd\_Out(3,1,temp2); // Write text in second row

Delay\_ms(200);

if (digit\_no==0) {

PORTC = 0; // Turn off both displays

PORTA = 0X88; // Set mask for displaying ones on PORTD

PORTC = 1; // Turn on display for ones (LSD)

digit\_no = 1;

} else {

PORTC = 0; // Turn off both displays

PORTA = 0x82; // Set mask for displaying tens on PORTD

PORTC = 2; // Turn on display for tens (MSD)

digit\_no = 0;

}

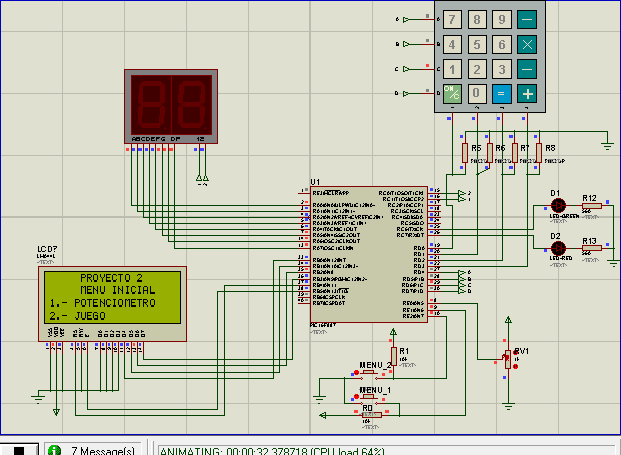
}while(!(RE1\_bit));

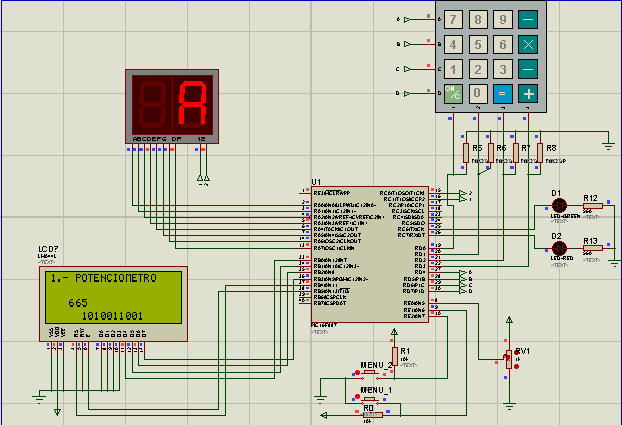
goto abc;

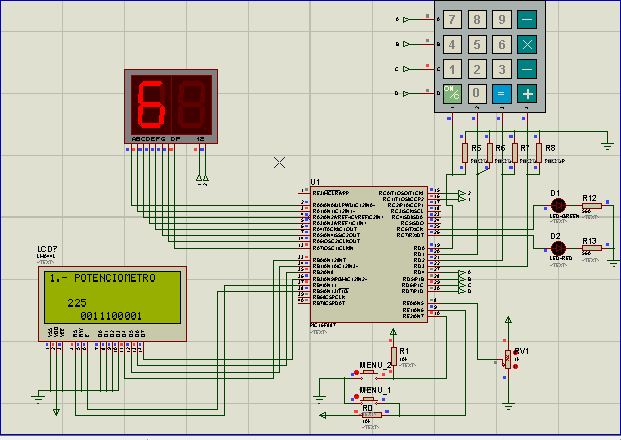
} // Endless loop

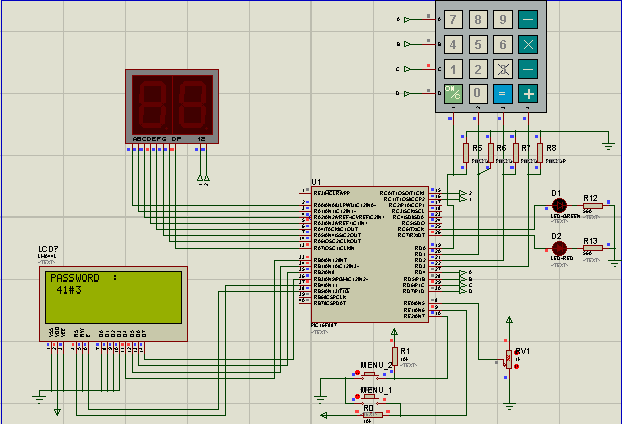
}

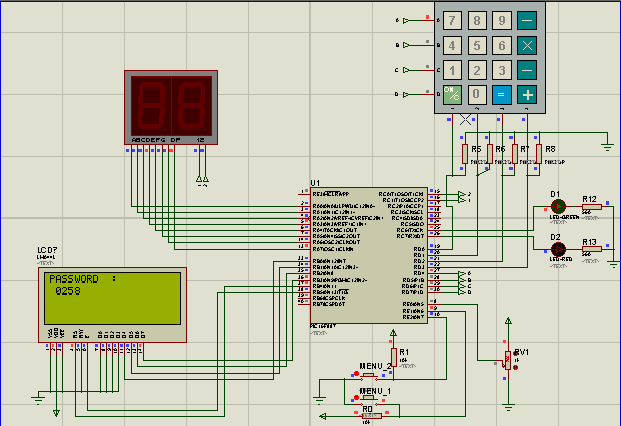
**6.- Copia impresa del circuito armado en PROTEUS para la simulación en el momento de su ejecución.**

****

****

****

****

****

**7.- Conclusiones**

* Se pudo observar la facilidades que ofrece la programación en lenguaje C, con respecto al ensamblador, sin embargo la memoria que se utiliza es mucho mayor en este lenguaje que en el ensamblador; ya que se utilizan funciones ya escritas, las mismas que ocupan cierta cantidad de memoria, además se pudo comprobar los diferentes conocimientos adquiridos a lo largo del curso como el uso de pantalla LCD y teclado 4x4 y entradas analógicas.
* Para la parte del potenciómetro se recogió los valores analógicos del potenciómetro y luego se los llevo a digital con la ayuda de una función, además de esto se presentaron dichos valores en binario y decimal en la pantalla con la ayuda de un algoritmo para convertirlo en los mismos y poder proyectarlos en la pantalla. Para elegir en el menú se necesitó de 2 botoneras que aplican para la selección de cada programa, los mismos que seguirán activos siempre que sigan presionados.

* Para el actual proyecto se utilizó una frecuencia de 8 MHz con la misma se podía realizar todas las funciones de la mejor manera, aunque para la multiplexación se debió colocar las funciones que dan retardos, logrando con esto la proyección de las iniciales, sin embargo se nota el cambio entre displays, pero este efecto se lo deja para que no afecte el resto del programa. El correcto código ASCII para cada tecla debe ser colocado para poder proyectarlos de manera correcta.

**8.- Recomendaciones**

* Se debe tener cuidado para el momento de la simulación de colocar las resistencias correctas para el uso del teclado, para este proyecto se utilizó las pullups debido a que gracias a las mismas no ocurren errores en la simulación, mientras que con las resistencias normales se producen errores con el simulador.
* Se debe tener cuidado al momento de programar el pic16f887, recordando que señales funcionan como entrada y cuales como salida, así mismo deben de tener en cuenta cuales son analógicas y cuales son digitales, ya que si no en el simulador se producirán errores, mientras que en el lenguaje c no se encuentren, ya que no hay error de sintaxis.
* Debido a que no existe botón de encendido ni apagado, el programa se ejecuta constantemente durante todo el tiempo, recordando que el menú se elige para realizar alguna de las dos acciones que existen, pudiendo siempre regresar al menú luego de dejar de presionar el botón que ejecuta al mismo.