**ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL**

**FACULTAD DE INGENIERÍA EN ELECTRICIDAD Y COMPUTACIÓN**

**LABORATORIO DE MICROCONTROLADORES**

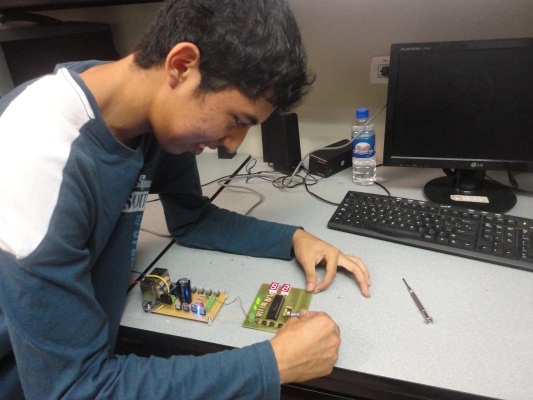
**PROYECTO # 2**

**Lectura de valor analógico de un potenciómetro y Tema Libre:**

**Camino de LEDs.**

**Nombre:**

Carlos Moisés Díaz Espinoza



**Profesor:**

Ing. Hugo Villavicencio

**Paralelo:** # 1

**Grupo:** # 5

**Fecha de Presentación:** Enero 25 del 2012

**II termino 2011 – 2012**

**Especificaciones técnicas del proyecto:**

El siguiente proyecto es desarrollado en lenguaje C y ejecuta dos programas:

* Programa #1 que consiste en la *lectura de los valores analógicos* de un potenciómetro colocado en la bornera de entrada analógica, presentando las variaciones de valores (en binario) en 8 LEDs y en dos DISPLAYs de 7 segmentos (en dos dígitos decimales) disponibles.
* Programa #2 que consiste en un *camino de LEDs* haciendo uso de los recursos de la tarjeta prototipo que consta de cuatro botoneras, 8 LEDs y dos DISPLAYs de 7 segmentos.

**Descripción del Proyecto:**

El PROGRAMA1 consiste en hacer uso del módulo ADC que posee el microcontrolador. Lee el valor analógico del potenciómetro conectado en el PORTA y muestra su equivalente digital DECIMAL: unidades y decenas en los dos DISPLAYs conectados en los puertos C y D respectivamente; y su valor BINARIO en los 8 LEDs conectados al PORTB.

El PROGRAMA2 consiste en un Camino de LEDs en el cual se tienen 2 jugadores, eligiendo el orden del que empieza al azar. Inicialmente se establece un valor aleatorio del camino para empezar, luego comienzan a jugar mostrando el número de jugador en el DISPLAY2 e incrementando el valor en el puerto b con el número que saquen los jugadores. Finalmente gana el primero en llegar al final del camino de LEDs, mostrando en los DISPLAYs la letra g de gana y el numero del jugador ganador.

**Diagrama de Bloques:**



**Diagrama de Flujo del programa principal:**



**Descripción del Algoritmo**

Se escriben los parámetros de configuración y inicializan las variables y los puertos a ser utilizados. Se muestran en los puertos C y D las iniciales del estudiante. Quedamos encerrador en un lazo infinito que pregunta por la BOTONERA1 o la BOTONERA2.

Si se presiona la BOTONERA1, se ejecuta el PROGRAMA1: Se configura el modulo ADC y se leen los Datos Analógicos desde el PIN0 del PORTA. Una vez obtenido el valor digital se lo divide para 10.23 para obtener un rango de 0 a 99. Seguido separamos el numero en unidades y decenas y Finalmente mostramos el valor binario en el PORTB y el valor Digital Decimal en los PUERTOS C y D. Usando un arreglo de códigos de 7 segmentos para poder mostrar el numero correctamente.

Si se presiona la BOTONERA2, se ejecuta el PROGRAMA2: Se inicializan las variables y puertos a usar; y quedamos dentro de un lazo infinito en el que se está generando un número aleatorio. Si se presiona la BOTONERA1 queda seleccionado el número aleatorio que se va a mostrar para mostrar el camino de LEDs, al mismo tiempo se determina que jugador va a empezar arbitrariamente.

Luego quedamos en un lazo en el que esperamos por las BOTONERAS 2 o 3 que serán el jugador 1 y jugador 2 respectivamente. Cada vez que se presionan, se genera un número aleatorio para incrementar el camino de LEDs mostrado en el PORTB, al mismo tiempo se muestra el turno de de cada jugador en los DISPLAYs. Y así sucesivamente hasta que alguno de los dos jugadores llegue al final del camino. El ganador final se muestra con una G y su número respectivo al final en los DISPLAYs.

**Programa Fuente:**

int dec,uni,decimal,dato,aux;

int const disp[10] = {0x3F, 0x06, 0x5B, 0x4F, //Se declaran como constantes

0x66, 0x6D, 0x7D, 0x07, 0x7F, 0x6F}; //Código Hexadecimal para cada número

int const id[27] = {0X77,0X7C,0X39,0X5E,0x79,0x71,0x3d,0x76,0x06,0x1e,0x76,0x38,0x4F,0x54,0x55,0x5c,0x73,0x67,0x50,0x6d,0x70,0x1c,0x3e,0x4f,0x76,0x6e,0x5b};

// A , B , C , D , E , F , G , H , I , J , K , L , M , N , Ñ , O , P , Q , R , S , T , U , V , W , X , Y , Z

// 0 , 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , 10 , 11 ,12 ,13 ,14 ,15 ,16 ,17 ,18 ,19 ,20 ,21 ,22 ,23 ,24 ,25 ,26

int const cam[9] = {0,1,2,4,8,16,32,64,128};

int espejo(int temp1) //Función Para invertir Bits de una variable

{ int temp2=0;

if ((temp1&0x01)==0x01)

temp2=temp2+128;

if ((temp1&0x02)==0x02)

temp2=temp2+64;

if ((temp1&0x04)==0x04)

temp2=temp2+32;

if ((temp1&0x08)==0x08)

temp2=temp2+16;

if ((temp1&0x10)==0x10)

temp2=temp2+8;

if ((temp1&0x20)==0x20)

temp2=temp2+4;

if ((temp1&0x40)==0x40)

temp2=temp2+2;

if ((temp1&0x80)==0x80)

temp2=temp2+1;

return temp2;

}

int j1(int cont, int temp) // función que me permite simular el juego de cada jugador

{ int alea,gana=0;

if (cont==1)

{

while(!PORTA.F2)

{

if (alea==4)

alea=0;

alea++; // se genera un numero aleatorio entre 1 y 4 para avanzar cada jugador

}

temp=temp+alea;

if(temp>8)

PORTB=0;

else

PORTB=cam[temp]; // se muestra el camino de Leds en el PORTB

Delay\_ms(100);

return temp;

}

while(!PORTA.F3)

{

if (alea==4)

alea=0;

alea++;

}

temp=temp+alea;

if(temp>8)

PORTB=0;

else

PORTB=cam[temp];

Delay\_ms(100);

return temp;

}

void p1()

{

ADC\_Init(); //inicializo el ADC

while(1)

{

dato=ADC\_Read(0); //entrada analógica 0

aux=(dato/10.23); //división perfecta para marcar rango

if(aux<=99)

{

dec=aux/10; //división para extraer decimal del dato anterior

uni=aux%10; //modulo (residuo de la división) para extraer unidades

}

if(aux==100)

{

dec=0;

uni=0;

}

PORTB=espejo(aux); // muestro el valor binario en el PORTB

PORTC=disp[uni]; // muestro valor unidades en el PORTC.

PORTD=disp[dec]; // muestro valor decenas en el PORTD

}

}

void p2()

{

int alea=0, temp=0, cont=0, gana=0, juga=0; //declaración de variables

do{ //se enceran DISPLAYs

PORTC=disp[0];

PORTD=disp[0];

}while(PORTA.F2); // se valida que suelte la Botonera

while(1)

{

while(!PORTA.F1) // lazo infinito

{ // Genera un valor aleatorio entre 1 y 4 y lo almacena en la variable alea

// se genera el numero cuando se presiona la BOTONERA1

if (alea==4)

alea=0;

alea++;

PORTD=disp[alea];

Delay\_ms(100);

}

temp=alea;

PORTB=cam[temp]; //se muestra el numero generado en el PORTB

Delay\_ms(100);

if(alea%2==0) // aprovechamos el numero aleatorio anterior para elegir que jugador empieza

juga=2; //dividiendo el numero para 2: si es par comienza el jugador 2 caso contrario comienza el jugador 1

else

juga=1;

while(1)

{

if(juga==1) // si empezó el jugador 1

{

PORTD=disp[0];

PORTC=disp[juga];

temp=j1(juga,temp); //se muestra la jugada en el camino de LEDs

if((temp>=8)&&juga==1) //alcanza o supera el tope del camino gana

{

gana=1;

break;

}

juga=2;

}

if(juga==2)

{

PORTd=disp[0]; // sino sigue jugando el jugador adversario

PORTC=disp[juga];

temp=j1(juga,temp);

if((temp>=8)&&juga==2) //si alcanza o supera el valor de 8 gana el otro jugador

{

gana=2;

break;

}

juga=1;

}

}

if(gana==1) // se muestra si gano el jugador 1

{

PORTD=id[6];

PORTC=disp[1];

Delay\_ms(1000);

break;

}

if(gana==2) // se muestra si gano el jugador 2

{

PORTD=id[6];

PORTC=disp[2];

Delay\_ms(1000);

break;

}

return;

}

}

void main()

{

ANSEL=0x01;

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

C1ON\_bit = 0; // Disable comparators

C2ON\_bit = 0;

TRISA = 0xFF; // PORTA como entrada

TRISC = 0; // PORTC como salida

TRISB = 0; // PORTB como salida

TRISD = 0; // PORTD como salida

PORTB=0; // inicializado en 0 el PORTB

while(1)

{

PORTC=id[3]; // iniciales

PORTD=id[2]; // iniciales

PORTB=0;

if(RA1\_bit=1) //prueba el bit 1 del PORTA

p1();

if(PORTA.F2=1) //prueba el bit 2 del PORTA

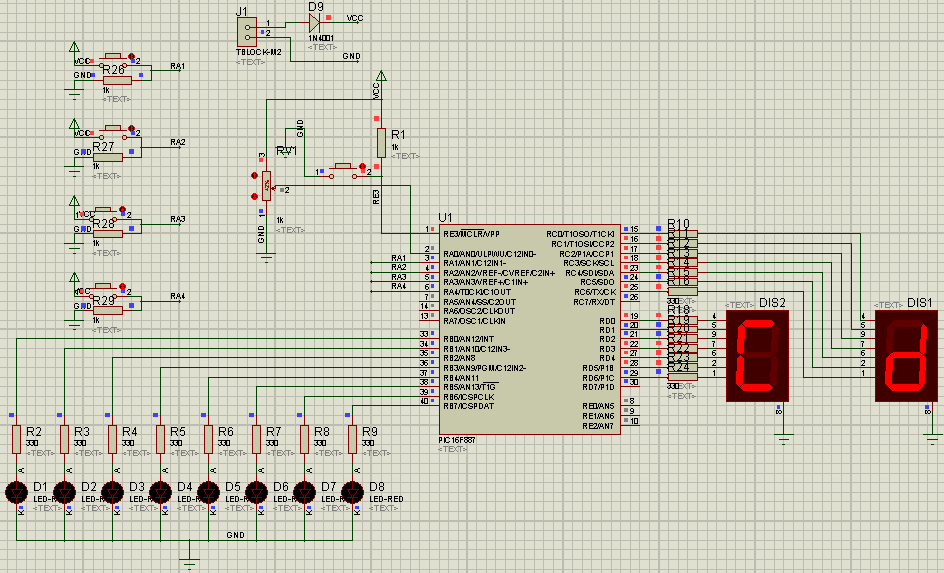
p2();

}

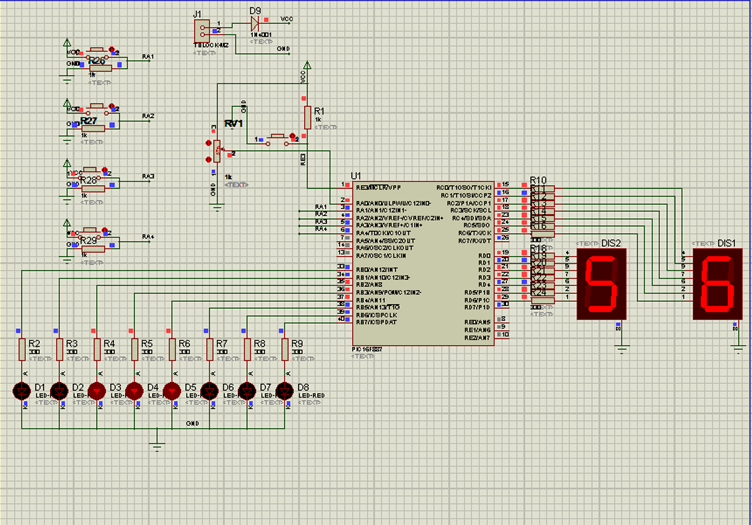
}

**Circuito armado en Proteus:**

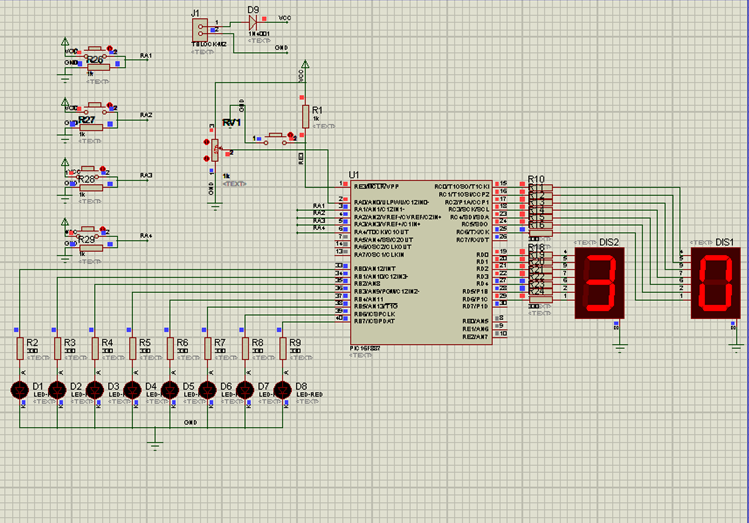
Mostrando las iniciales

****

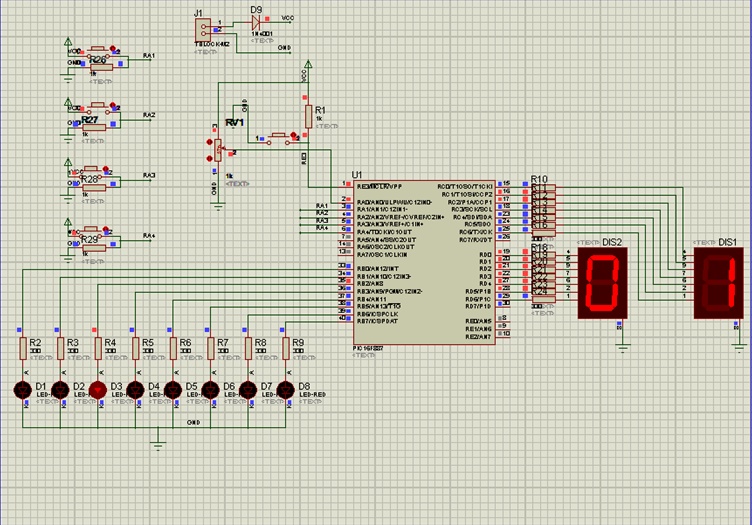
Conversión analógica digital programa 1



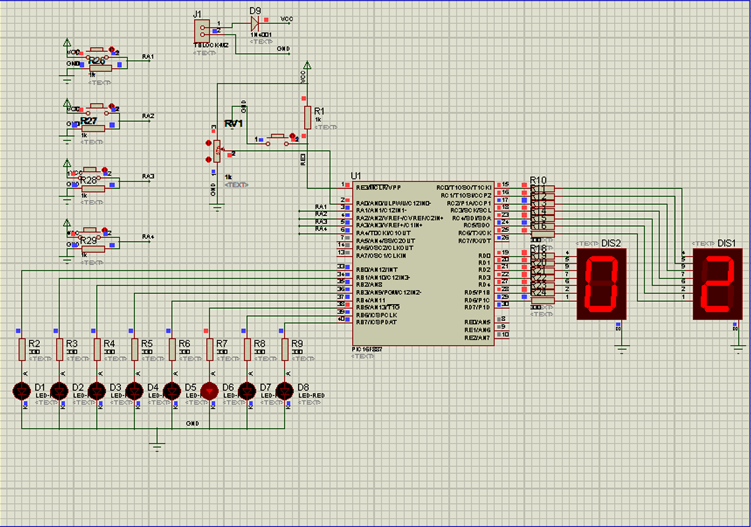
Camino de Led, generando jugador inicial aleatoriamente



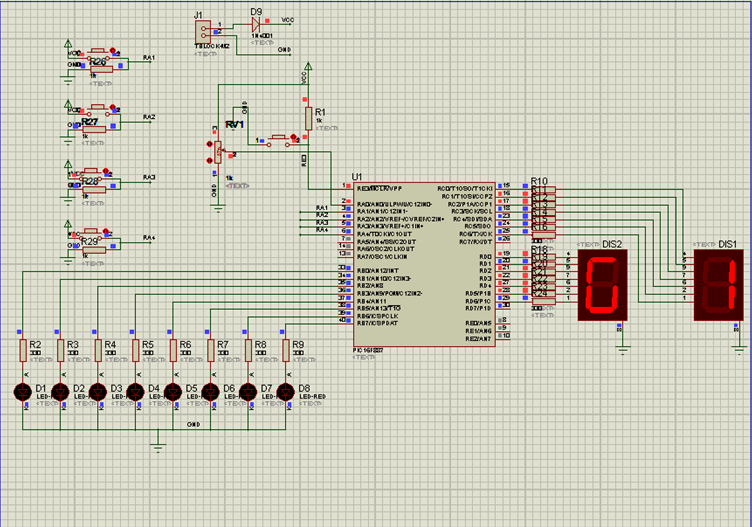
Presionando botonera 1 escogemos jugador que comienza y se muestra l valor inicial del caminito de Leds



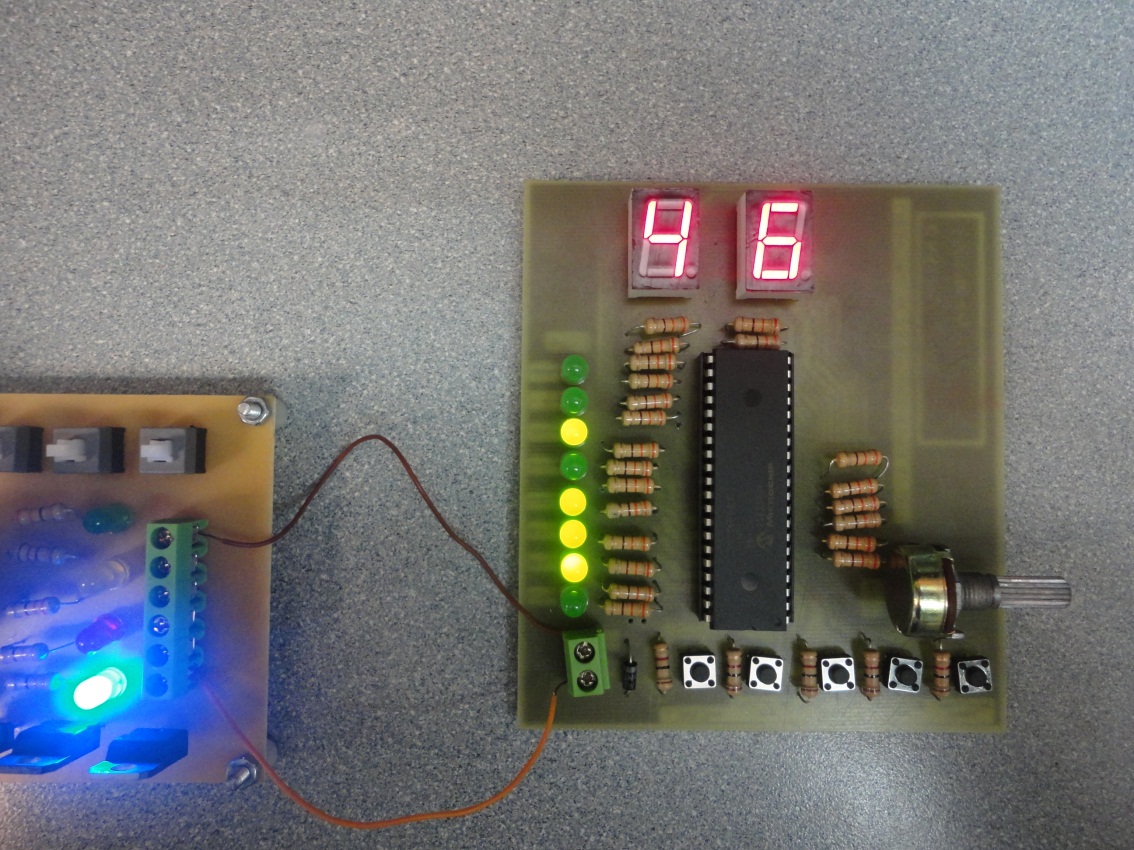
Avanzan presionando su botonera respectiva



El primero en llegar se muestra con la letra G de ganador y el número del jugador ganador



**Circuito impreso**



**Conclusiones:**

* Debemos notar que únicamente no se utilizan los programas Micro Pic C con el fin de crear el código, sino que tenemos que tener siempre a la mano un simulador de confianza (en este caso Proteus) para poder conocer los errores en nuestra programación y poder así detectar aquellos y corregirlos de una manera ordenada y pulcra.
* El convertidor ADC es util en muchas aplicaciones ya que en el mundo actual se maneja la informacion de manera digital pero la mayoria de las señales reales son analogicas asi que siempre sera importante poder manejar este tipo de convertidores.
* Cuando uno implementa un juego se puede dar cuenta de las cosas que se pueden implementar, simplemente programando el pic. Ademas de reforzar las intrucciones del lenguaje C para asi poder tener un conjunto de intrucciones utiles al momento de programar en un futuro programa.

**Recomendaciones:**

* Antes de realizar cualquier proyecto es necesario tener a la mano todos los materiales y verificar su buen estado antes de conectar para no tener problemas en la presentación final. Es útil tener más de un elemento de los necesarios.
* Estudiar del funcionamiento del ADC, para un excelente manejo de su código dentro de las sentencias del programa. Hacerlo de manera lógica y ordenada, para ello se puede guiar en los ejemplos proporcionados en Help que es la ayuda en C.
* Al terminar de soldar se recomienda medir con un multimetro la continuidad de la placa, y probar también si no existe algún corto circuito, ya que se puede dar el caso de que al soldar, se haya producido la unión de las pistas y esté haciendo corto circuito, también se recomienda limpiar la placa, para extraer causada por la pasta para soldar.