**OBJETIVOS:**

* Uso de la función ADC\_Read() para realizar la conversión de analógico a digital.
* Uso de retardos para evitar los rebotes naturales que se producen al presionar una botonera..
* Uso de funciones para convertir de binario a 7 segmentos.
* Uso de acumuladores para almacenar los puntajes de cada jugador.
* Manejo de lazos para realizar las tareas principales del programa

**1.- ENUNCIADO DEL PROYECTO**

**Convertidor ADC y juego de dados de 5 turnos**

El PIC16F887 se lo usa para programar dos juegos que el usuario puede elegir mediante botoneras, (RA1) para la conversión ADC y (RA2) para el juego de dados. Con la misma botonera el usuario puede ingresar o salir del juego en el que está. El primer juego consiste en un convertidor analógico-digital, resulta sencillo realizar la conversión mediante la programación en c ya que con una función se realiza dicha tarea, luego se realizan operaciones aritméticas para poder mostrar dicha conversión en dos display, estos valores van desde 00 cuando el potenciómetro esta al mínimo y 99 cuando el potenciómetro está al máximo. El segundo juego consiste en elegir un número del 1 al 6 que se genera a alta frecuencia y se presenta mediante un display. Para seleccionar el número el jugador A debe presionar la botonera 3 (RA3), el puntaje acumulado se va mostrando en el juego de 8 leds. Luego de que el jugador A elija un número se da paso para que el jugador 2 elija un número presionando la botonera (RA4), el número se presenta en otro display. Al final de los 5 turnos se muestra que jugador ganó la partida, mediante los display y mediante los 4 leds menos significativos si ganó el jugador A y los 4 leds más significativos si ganó el jugador B.

**2.- DIAGRAMA DE BLOQUES**

Display 1****

But4

8

8

8

But1

1

1

Display 2****

**Microcontrolador**

**PIC: 16F887**

But2

1

Juego de

8 Leds

1

1

But3

1

Potenciómetro

 ****

But5

**3.- DIAGRAMA DE FLUJO**

Mostrar nombre

Delay

Seleccionar juego

No

 RA1=1

Si

No

 RA2=1

Convertidor ADC

Delay

temp\_res=ADC\_Read(0)

var = (temp\_res/4);

Conversión de 10 bits a 8 bits

tmp=(var\*100)/255

Separar decenas

tmp2=tmp/10

Separar unidades

tmp3=tmp-(tmp2\*10)

PORTC=display(tmp2)

PORTC=display(tmp2)

No

 RA1=1

Si

No

Delay

var1+=0x01

PORTC=var1

Si

 RA3=1

No

No

var1==0x07

Si

var1=0x01

w uni\_seg-w

Z=1

Z=1

w dec\_seg-w

w dec\_alarma

Si

No

PORTD=var2

 RA4=1

Var2=0x01

Var2+=0x01

Var2==0x07

Si

No

cont++

Si

Cont==5

No

**ALGORITMO:**

Para comprender el funcionamiento del programa se describen los siguientes pasos del algoritmo.

En la parte inicial del programa, se muestran las iniciales del nombre de la persona que presenta el proyecto

Mediante dos botoneras se selecciona que tipo de juego se quiere, si se presiona RA1 se elije el juego 1 que consiste en un convertidor ADC, la señal analógica ingresa al pic a través de la patita RA0. Para poder presentar la conversión se guarda el valor que retorna la función ADC\_ Read en una variable, luego se realizan operaciones aritméticas para separar la unidades y decenas y así poder mostrar el resultado de la conversión en dos displays y en 8 leds se retorna el valor en binario. Para salir de el juego se debe presionar nuevamente la botonera que va al pin RA1.

 Para elegir el juego 2 se debe presionar la botonera RA2 donde se ingresa a un juego de dados, se genera a alta frecuencia un número del 1 al 6 que corresponde al dado y se muestra mediante un display, el jugador A debe presionar la botonera RA3 para cargar el valor del dado es un registro e ir acumulando el puntaje, para que al final de las 5 partidas se muestre que jugador ganó. Para mostrar el ganador se usan los dos displays y se prenden los bits menos significativos si ganó el jugador A y los más significativos si ganó el jugador B.

Una vez que el jugador A escogió un número, le corresponde al jugador B elegir, para ello debe presionar la botonera RA4. Y sucede la misma secuencia que el jugador A. Si se quiere salir del juego 2 se debe presionar nuevamente la botonera RA2.

Una vez terminado los 5 turnos se muestra al ganador, luego de aquello se regresa al estado donde se espera hasta que el usuario elija que juego quiere ejecutar.

**4.- PROGRAMA FUENTE**

/\*

 \* Nombre del Proyecto:

 P9b\_adc.c

 \* Nombre del Autor:

 (c) Mikroelektronika, 2009.

 \* Description:

 (Explicación del ejercicio)

 \* Test configuration:

 MCU: PIC16F887

 Oscillator: HS, 08.0000 MHz

 SW: mikroC PRO for PIC

 \* NOTES:

\*/

unsigned int temp\_res;

unsigned short kp1=0,kp2=0,var,var1=0,var1dis,var2=0,var2dis,n,jugA=0,jugB=0,cont=10,disp\_uni=0,disp\_dec=0;

unsigned short tmp=0,tmp2=0,tmp3=0;

unsigned short display(unsigned short num);

void main() {

 ANSEL = 0x01; // Configure AN2 pin as analog

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

 C1ON\_bit = 0; // Disable comparators

 C2ON\_bit = 0;

 TRISA = 0xFF; // PORTA is input

 TRISB = 0;

 TRISC = 0; // PORTC is output

 TRISD = 0; // PORTD is output

 PORTC= 14;

 PORTD= 63;

 Delay\_ms(2000);

 do {

 PORTB= 0;

 PORTC= 0;

 PORTD= 0;

 if (RA1\_bit){// El jugador elige la conversion ADC

 Delay\_ms(200); //Retardo para la botonera

 kp1= 1;

 do {

 temp\_res = ADC\_Read(0); // Get 10-bit results of AD conversion

 var = (temp\_res/4); //Conversion de 10 bits a 8 bits

 tmp=(var\*100)/255;

 tmp2=tmp/10; //Separacion de las decenas

 tmp3=tmp-(tmp2\*10);//Separacion de las unidades

 if(tmp==100)

 {

 PORTC=display(9); //Mostrar en el display

 PORTD=display(9); //Mostrar en el display

 }else{

 PORTC=display(tmp2); //Llama a la funcion que convierte de binario a 7 segmentos

 PORTD=display(tmp3);

 }

 PORTB = var;

 if (RA1\_bit){// El usuario elige salir del juego 1

 Delay\_ms(50);

 kp1= 0;

 }

 }while(kp1);

 }

 if(RA2\_bit){ // El usuario elige el juego 2

 Delay\_ms(200);

 kp2= 1;// Inicializacion de variables y acumuladores

 cont=0;

 jugA=0;

 jugB=0;

 n=1;

 do{

 if((n%2)!=0){

 var1+=0x01; //Se genera el numero para el jugador A

 if(var1==0x07){

 var1=0x01;

 }

 var1dis=display(var1); //Mostrar el valor del numero

 PORTC=var1dis;//Puerto de salida

 Delay\_ms(40);

 if(RA3\_bit){ //Jugador A elige un numero

 Delay\_ms(50);

 n++;

 jugA+=var1;//Acumulacion de puntaje jugador a

 PORTB=jugA;// Mostrar puntaje acumulado

 }

 }

 if((n%2)==0)

 { var2+=0x01;

 if(var2==0x07){

 var2=0x01;

 }

 var2dis=display(var2); //Mostrar el valor del numero

 PORTD=var2dis; //Puerto de salida

 Delay\_ms(40);

 if(RA4\_bit){ //Jugador B elige un numero

 Delay\_ms(50);

 n++;

 jugB+=var2; //Acumulacion de puntaje jugador b

 PORTB=jugB; // Mostrar puntaje acumulado

 cont++;

 }

 }

 if(RA2\_bit){

 Delay\_ms(200);

 kp2=0;

 break;

 }

 if(cont==5){ // Se cumplieron los 5 turnos

 kp2=0;

 if(jugA>jugB){ //Muestra si gana el jugador A

 PORTC=0x0E;

 PORTD=0x06;

 PORTB=15;

 Delay\_ms(2000);}

 if(jugA<jugB){ //Muestra si gana el jugador B

 PORTC=0x0E;

 PORTD=0x5B;

 PORTB=240;

 Delay\_ms(2000);}

 if(jugA==jugB){ //Muestra si hay empate entre A y B

 PORTC=0x79;

 PORTD=0x73;

 PORTB=255;

 Delay\_ms(2000);}

 }

 } while(kp2);

 }

 } while(1);

}

 unsigned short display(unsigned short num) { //Funcion que realiza la conversion a 7 segmentos

 switch (num) {

 case 0 : return 0x3F;

 case 1 : return 0x06;

 case 2 : return 0x5B;

 case 3 : return 0x4F;

 case 4 : return 0x66;

 case 5 : return 0x6D;

 case 6 : return 0x7D;

 case 7 : return 0x07;

 case 8 : return 0x7F;

 case 9 : return 0x6F;

 }

 }

**5.- CIRCUITO EN PROTEUS**

Al inicio se muestran las iniciales del estudiante



Conversión Analógica digital



Juego de Dados



**6.- CONCLUSIONES**

* La programación en lenguaje de alto nivel como es C simplifica mucho la realización de distintas tareas, esto lo hace mucho más eficiente que el lenguaje ensamblador. Se ahorra mucho tiempo al no tener que configurar ciertos parámetros ya que el programa tiene funciones que realizan aquello.
* Para el convertidor analógico digital se utiliza la función ADC\_read, dependiendo de la variación de voltaje (0 a 5V) y la resolución que para el pic 16f887 es de 10 bits, devuelve un valor que va de 0 a 1024. Para poder mostrarlo en los displays se lo convirtió a 8 bits, luego se separaron las unidades de las decenas.
* Los lazos cíclicos permiten programar secuencias, los mismos que permiten ejecutar de forma continua el programa hasta que se cumpla la condición de salida de dicho lazo. Mediante operadores lógicos se estableció un orden para el turno de cada jugador y el número de veces que podrá escoger un número.

**7.- RECOMENDACIONES**

* Aunque C tiene funciones predeterminadas es importante utilizar la ayuda del programa mikroC para conocer con más detalle que realiza la función, o que tipos de dato devuelve. Usar sub-funciones en el programa ayudarán a reducir la programación y se logrará una mejor comprensión del código.
* Soldar las pistas con cuidado. Revisar que haya continuidad en cada una de las pistas. Antes de soldar los leds revisar que estos funcionen, de pronto vayan a estar quemados. Antes de energizar el micro controlador revisar que no haya corto entre vcc y gnd, además verificar de no sobrepasarse el voltaje de polarización del PIC que es de 5v.
* Elegir el oscilador interno en el momento de programar el PIC. Elegir el oscilador interno de 4MHz, cuando se carga el programa el pic del simulador de Proteus colocar también la frecuencia de 4MHz y comprobar el paso del reloj con el tiempo del simulador.

**8.- PCB DEL CONVERTIDOR ADC Y JUEGO DE DADOS**

