

Escuela Superior Politecnica del Litoral

Facultad de Ingeniería en Electricidad y Computación

**Laboratorio de Microcontroladores**

**PROYECTO PARCIAL**

Alumno:

Bosco Alcivar

Paralelo: 12

Grupo: xx

Fecha de Presentación:

25/01/2012

2011 II TÉRMINO

**1.- Enunciado del proyecto**

**PARTE 1: LECTURA ANALOGICA**

El proyecto 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.

**PARTE 2: JUEGO DE ADIVINANZA**

El juego consiste en un juego de adivinanza, es decir antes de generar los números aleatorios pregunta si el número a generarse es par o impar, si se presiona RA3 entonces el numero aleatorio es par, pero si al mostrar el numero generado es impar entonces pierdes, caso contrario ganas así mismo si presionas RA4 entonces el numero aleatorio es impar, pero si al mostrar el numero generado es par entonces pierde, caso contrario ganas.

**2.- Diagrama de Bloques**

****

**3.- Diagrama de Flujo funcional del Programa principal**









**4.- Diagrama de Flujo funcional de las Subrutinas**



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

1. Se declaran las variables temp, temp\_res, n1, n2, oldstate, oldstate2, mostc, mostc1, mostd, mostd2, por y tiempo y además se setea los puertos B, C, D como salidas y el puerto A como entrada.
2. Al encender el circuito aparece las iniciales B y A, luego si presiona RA1 se va a la etiqueta adc, y si se presiona RA2 se va a la etiqueta juego.
3. En la etiqueta adc está la lectura analógica del potenciómetro que está conectado en RA0 y la representa en los dos display y en los leds que está conectado en el puerto b y si presiona RA3 regresara al inicio del programa.
4. En la etiqueta juego esta lo relacionado con el juego y consiste en que en generar dos números aleatorios y preguntar si dicho número es par o impar.
5. Si se presiona RA3 se dice que el numero aleatorio a generarse es par y al no ser así, se presenta en los display la letra i, pero al ser par se presenta la letra p y ganas el juego y regresas al paso 1.
6. Si se presiona RA4 se dice que el numero aleatorio a generarse es impar y al no ser, asi se presenta en los display la letra p, pero al ser impar presenta la letra i y ganas el juego y regresas al paso

**6.- Listado del programa fuente en lenguaje C**

unsigned short mostc,mostc1,mostd,mostd1,oldstate,oldstate2,i;

unsigned long temp,temp\_res,n1,n2,mostd2,mostc12;

float por;

unsigned short mask(unsigned short num) {

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;

}

}

unsigned short aleatorio(unsigned short tiempo){

 unsigned short in=0;

 unsigned short y=0;

 unsigned short y1=0;

 unsigned short r=0;

 do{

 y=rand();

 in=y%10u;

 y=y/10;

 y1=y%10u;

 r = min(in,y1);

 tiempo--;

 }while(!(r>0&&r<=5)||tiempo!=0);

 return r;

}

void main() {

 ANSEL = 0x01; // Configure AN0 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 = 0X00;

 TRISC = 0X00;

 TRISD = 0X00;

 main1:

 PORTB = 0X00;

 PORTC = 0X00;

 PORTD = 0X00;

 PORTD = 0X7C;

 PORTC = 0X77;

 oldstate=0;

 oldstate2=0;

 mostd=0;

 mostd1=0;

 mostc=0;

 mostc1=0;

 n1=0;

 n2=0;

 por=0.098;

 do {

 if (Button(&PORTA, 1, 1, 1)) { // Detect logical one

 oldstate = 1; // Update flag

 }

 if (oldstate && Button(&PORTA, 1, 1, 0)) { // Detect one-to-zero transition

 oldstate = 0; // Update flag

 goto adc;

 }

 if (Button(&PORTA, 2, 1, 1)) { // Detect logical one

 oldstate2 = 1; // Update flag

 }

 if (oldstate2 && Button(&PORTA, 2, 1, 0)) { // Detect one-to-zero transition

 oldstate2 = 0; // Update flag

 goto juego;

 }

 } while(1);

 adc:

 ADC\_Init();

 do {

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

 temp=temp\_res\*por;

 PORTB=temp;

 mostc= temp%10;

 mostc1=mask(mostc);

 PORTC=mostc1;

 temp=temp/10;

 mostd= temp%10;

 mostd1=mask(mostd);

 PORTD=mostd1;

 if (Button(&PORTA, 3, 1, 1)) { // Detect logical one

 oldstate = 1; // Update flag

 }

 if (oldstate && Button(&PORTA, 3, 1, 0)) { // Detect one-to-zero transition

 oldstate = 0; // Update flag

 goto main1;

 }

 } while(1);

 juego:

 PORTD=0X00;

 PORTC=0X00;

 PORTB=0X00;

 PORTC=0X3F;

 PORTD=0X3F;

 Delay\_ms(1000);

 oldstate=0;

 oldstate2=0;

 mostc=0;

 mostc1=0;

 mostd=0;

 mostd1=0;

 n1=0;

 n2=0;

 i=0;

menu:

mostc=0;

mostd=0;

 mostd1=0;

 n1=0;

 n2=0;

 i=0;

do {

 if (Button(&PORTA, 3, 1, 1)) { // Detect logical one

 mostc=1;

 oldstate = 1; // Update flag

 }

 if (oldstate && Button(&PORTA, 3, 1, 0)) { // Detect one-to-zero transition

 oldstate = 0; // Update flag

 goto listo;

 }

 if (Button(&PORTA, 4, 1, 1)) { // Detect logical one

 mostd=1;

 oldstate2 = 1; // Update flag

 }

 if (oldstate2 && Button(&PORTA, 4, 1, 0)) { // Detect one-to-zero transition

 oldstate2 = 0; // Update flag

 goto listo;

 }

 } while(1);

listo:

PORTB=0X00;

PORTD=0X3F;

PORTC=0X3F;

Delay\_ms(500);

do{

if(mostc1==2)

{goto perdio

}

n1=aleatorio(10);

n2=aleatorio(5);

PORTD=mask(n1);

PORTC=mask(n2);

i=n1+n2;

Delay\_ms(1000);

mostc1++;

goto adivinansa;

}while(1);

adivinansa:

if(mostc==1)

{ if((i%2)==0)

{ PORTD=0X3F;

 PORTC=0X3F;

 PORTB=0XFF;

 Delay\_ms(500);

 PORTD=mask(n1);

 PORTC=mask(n2);

 Delay\_ms(500);

 PORTD=0X73;

 PORTC=0X73;

 PORTB=0XFF;

 Delay\_ms(1000);

 goto main1;

}else{PORTC=mask(mostc1);

 PORTD=0x3F;

 PORTB=0XF0;

 Delay\_ms(1000);

 goto menu;

}

}

if(mostd==1)

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

{ PORTD=0X3F;

 PORTC=0X3F;

 PORTB=0XFF;

 Delay\_ms(500);

 PORTD=mask(n1);

 PORTC=mask(n2);

 Delay\_ms(500);

 PORTD=0x06;

 PORTC=0x06;

 PORTB=0XFF;

 Delay\_ms(1000);

 goto main1;

}else{PORTC=mask(mostc1);

 PORTD=0x3F;

 PORTB=0X0F;

 Delay\_ms(1000);

 goto menu;

}

}

perdio:

PORTB=0XFF;

PORTD=0X3F;

PORTC=0X3F;

Delay\_ms(500);

PORTB=0X00;

PORTD=0X00;

PORTC=0X00;

Delay\_ms(500);

PORTB=0XFF;

PORTD=0X3F;

PORTC=0X3F;

Delay\_ms(500);

PORTB=0X00;

PORTD=0X00;

PORTC=0X00;

Delay\_ms(1000);

goto main1;

}

1. **- Simulación en PROTEUS**

****

**ADC**

****

**Juego**

****

**8.- Conclusiones**

* Para poder leer valores analógicos se debe configurar los pines como entrada analógica en este caso utilizamos el pin RA0 del puerto A, además para poder utilizar el adc se debe inicializar la librería también hay otra forma que es a nivel de bits del registro adcon, adc del pic tiene una resolución de 10 0 12 bits, esto se lo configura seteando los bits del registro adcon.
* Debes tener muy en cuenta al momento de crear variables, es decir saber de qué tipo de variables quieres trabajar para ser más óptima la programación y así no gastar memoria de manera innecesaria, además evita utilizar nombres para la variables que sea igual al nombre de una variable o registro predeterminado del compilador es decir propia de micro c ya que no te aceptara dicha variable y al momento de compilar te botara error
* El pic puede trabajar como un oscilador de cuarzo o simplemente como el oscilador interno que tiene el pic, el cual va de 31khz hasta 8Mhz esta frecuencia se la puede seleccionar programando el registro OSCON o simplemente con el software MICRO C el cual automáticamente te lo selecciona según tus requerimientos, en el proyecto estamos utilizando una frecuencia de oscilación de 8Mhz el cual generara ciclos de 0.5 us
* Micro c es un lenguaje de alto nivel el cual tiene diferentes librerías con funciones predeterminadas que te pueden ser muy útil al momento de hacer algún proyecto además para evitar el anti reboté se puede utilizar la función button el cual te ayudara para eliminar este efecto, además existe la librería delays el cual te ayudara para generar retardos.

.

**9.- Recomendaciones**

* Reducir el número de líneas es bueno para un mejor uso en la memoria del PIC y evitar un desperdicio de la misma, además se evita redundancias que podrían confundir a otro programador que vea el informe del proyecto.
* Al momento de probar el proyecto ya soldado en la placa y verificando que no haiga ni un corto que produzca un mal funcionamiento se debe verificar que la fuente que vas a utilizar sea de 5 voltio y que cuyo voltaje no se caiga ya que el pic al no detectar un voltaje igual a 5 se resetea, ya que el diodo que protege al pic de que se queme puede reducir el voltaje de la fuente.
* Se recomienda eliminar lo antirebotes para que no haiga problemas al momento de presionar las botoneras, al no ser asi puede producir errores en la ejecución del programa.
* Si vas a utilizar sentencias en lenguaje ensamblador en micro c no olvides de anteponer la palabra asm, para que el compilador lo reconosca ya que alno ponerla puede botar errores al momento de compilar.