**ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL**



**FIEC**

**Laboratorio de Microcontroladores**

**Proyecto del Primer Parcial**

**Lectura de valor analógico de un potenciómetro y juego de adivinar un número.**

**Estudiante**

**Villacis Solis Milton Saul**

**Paralelo: 07**

**Grupo: 05**

**Fecha de presentación:**

**26/01/2012**

**2011 – 2° TÉRMINO**

****

**1.- Enunciado del proyecto**

Este proyecto ejecuta dos ejercicios:

1. 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. Para este caso se consideran en la salida digital valores que representan el porcentaje de voltaje presente en la entrada analógica considerando su máximo voltaje de referencia.
2. Consiste en un juego propuesto haciendo uso de los recursos de la tarjeta prototipo que consta de cuatro botoneras, 8 LEDs y dos displays de 7 segmentos. El juego trata de adivinar un número entre 0 y 10 que será ajustado en los displays por el usuario mientras las luces de los leds rotan interactivamente, en el momento que él cree haber adivinado el número, con un botón detiene la rotación de los leds y el sistema le muestra la respuesta correcta en los leds y en decimal en los displays por un tiempo de 1 segundo, volverá al inicio del sistema total.

**2.- Diagrama de Bloques**



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

*Programa Principal*

INICIO

DECLARACION DE FUNCIONES

-mask

-convertidor\_bit

DECLARACION DE VARIABLES

valor, ban, apuesta, ram,i, num, record;

digit\_no= 0, digit10, digit1, digit;

CONFIGURACION DEL ADC Y DE LAS RESPECTIVAS ENTRADAS Y SALIDAS

MOSTRAR INICIALES:

PORTC = 0x6D; A

PORTD = 0x3E; b

***Subrutinas*:**

Subrutina **Ejercicio1**



Subrutina **mask**



Subrutina **Ejercicio2**



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

1. Se declaran las funciones y variables necesarias para el del programa.
2. Se configura una de las entradas de PORTA para que sea entrada analógica que será leída por el ADC y también los pines que van a funcionar como entrada y salida digital.
3. Se muestran las iniciales del autor, nombre en el PORTD y apellido en el PORTC, en este caso será PORTD=”P” y PORTC=”L”, pero considerando la correcta codificación que manejarán dos displays de salida.
4. Si el botón uno (bit RA1) se presiona entonces se procederá a realizar el ejercicio 1 descrito en el enunciado anteriormente, si es el botón dos (bit RA2) se realizará el ejercicio 2.
5. Mientras no sucedan ninguno de los dos eventos anteriores se continuará mostrando las iniciales y esperando aquellos eventos, esto se logra regresando al paso 3.
6. Si el usuario elige la opción del ejercicio 1 una variable tomará el valor digital leído de la entrada analógica configurada en RA0.
7. Matemáticamente se transforma aquel numero obtenido en porcentaje, es decir, se tendrán número entre 0-99% (decimos 99 porque el display solo puede representar dos dígitos).
8. Se dividen los dígitos de aquel número en unidades y decenas.
9. Se obtienen los códigos correspondientes para unidades y decenas a representar en los displays de 7 segmentos de cátodo común.
10. Se muestran los códigos en los displays y será posible visualizar el valor de la conversión.
11. Continuaremos en el ejercicio 1 mientras no pasemos al ejercicio 2 o reiniciemos el sistema con el botón de reset asincrónico.
12. Si el usuario elige la opción del ejercicio 2 el sistema calcula un número aleatorio entre 0 y 10.
13. Inicialmente en los displays se muestra “00” y en los leds del PORTB se rota constantemente un bit en alto.
14. Mientras el paso anterior sucede el usuario puede ajustar el valor de los displays para colocar el número que cree estar adivinando, con el botón en RA3 ajusta decenas y con RA4 ajusta unidades. Este ajuste permite ajustes para números entre 0 y 10.
15. Para realizar los ajustes se enmascara el número que se va ajustando para que pueda ser mostrado en los displays.
16. Cuando el usuario elije el número presiona el botón RA2 para confirmarlo y en seguida el sistema muestra el número correcto tanto en binario en los leds del PORTB como en decimal en los displays por 2 segundos.
17. Luego de lo anterior regresa al estado inicial del sistema mostrando las iniciales.
18. El usuario tiene la opción de hacer trampa viendo el número si mantiene presionado el botón en RA1.

**5.- Listado del programa fuente en lenguaje ensamblador.**

unsigned long temp\_res;

unsigned int valor, ban, apuesta, ram,i, num, record;

unsigned short digit\_no= 0, decena, unidad, x;

unsigned short mask(unsigned short n){

switch (n) {

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;

case 10 : return 0x31;

case 11 : return 0x7D;

default : return 0x7f;

}

}

unsigned short convertidor\_bit(unsigned short n){

switch (n) {

case 1 : return 0x01;

case 2 : return 0x02;

case 3 : return 0x04;

case 4 : return 0x08;

case 5 : return 0x10;

default : return 0x1F;

}

}

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; // PORTB is output

TRISC = 0; // PORTC is output

TRISD = 0; // PORTD is output

PORTE = 0; // Turn off both displays

TRISE = 0; // All port A pins are configured as outputsPORTD = 0;

// PORTA = 255;

PORTB = 0;

PORTC = 0; // Turn off all display segments

PORTD = 0;

do {

PORTC = 0x6D;

PORTD = 0x3E;

if (RA1\_bit) {

while(1){

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

valor = (temp\_res \* 99) / 1023;

PORTB = (temp\_res \* 255) / 1023; // Send lower 8 bits to PORTD

x = (char)valor % 10u;

unidad = mask(x); // Prepare mask for displaying ones

x = (char)(valor / 10u) % 10u;

decena = mask(x); // Prepare mask for displaying tens

PORTC = unidad; // Set mask for displaying ones on PORTD

PORTD = decena; // Set mask for displaying tens on PORTD

Delay\_ms(15);

}

}

if (RA2\_bit) {

ram=0;

num=4,record=0;

while(1){

if (RA1\_bit){

delay\_ms(250);

num=num+1;

if(num>5)

num=1;

}

if (RA2\_bit){

delay\_ms(250);

num=num-1;

if(num<1)

num=5;

}

if (RA3\_bit){

delay\_ms(25);

while(RA3\_bit){

PORTB = convertidor\_bit(ram);

delay\_ms(1000);

if(num==ram){

record++;

PORTC=1;PORTD=1;

for(i=1;i<24;i++){

delay\_ms(100);

PORTC=PORTC\*2;

PORTD=PORTD\*2;

if(PORTC\*2==128){

//delay\_ms(100); //Retardo de 200 mseg

PORTC = 1;

PORTD = 1;

}

}

PORTC = mask(record);

delay\_ms(2000);

PORTC = 0;

}

else

record=0;

ram=0;

}

}

ram=ram+1;

delay\_ms(5);

if(ram>5)

ram=1;

x = (char)num;

PORTD = mask(x); // Prepare mask for displaying ones

x = (char)record;//aleat;

PORTC = mask(x); // Prepare mask for displaying tens

// = unidad; // Set mask for displaying ones on PORTD

// = decena; // Set mask for displaying tens on PORTC

}

}

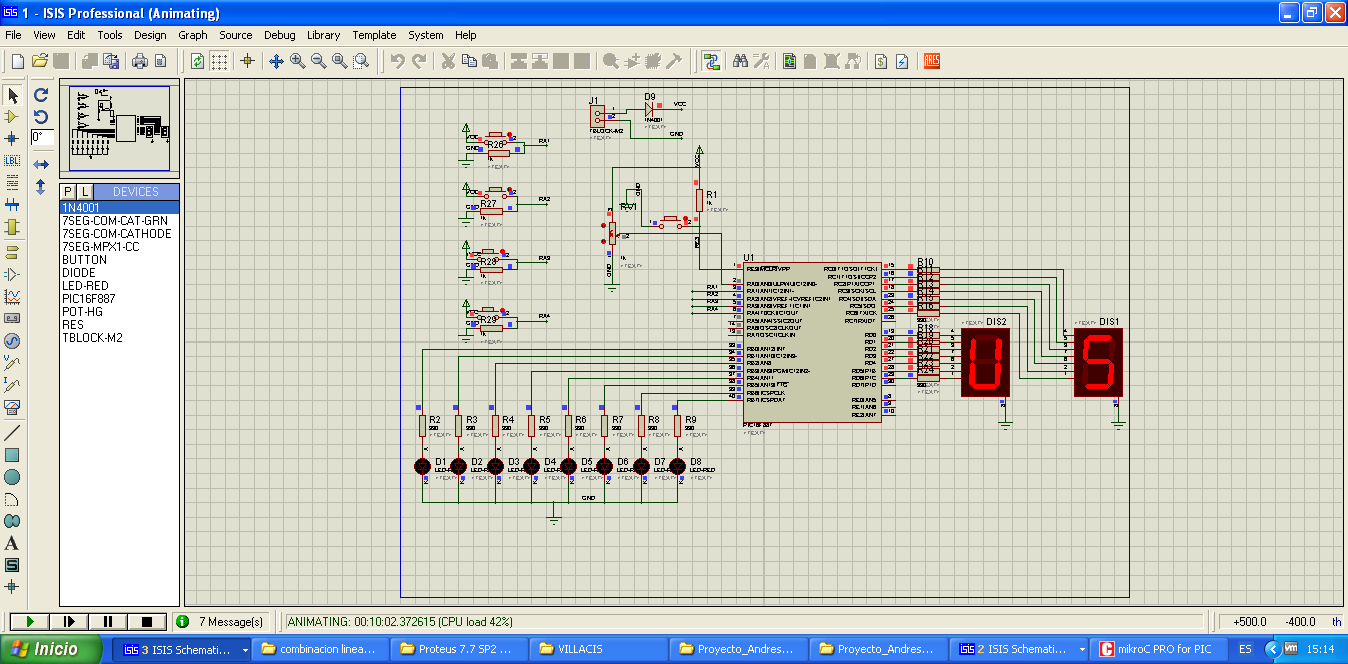
Delay\_ms(5);

} while(1);

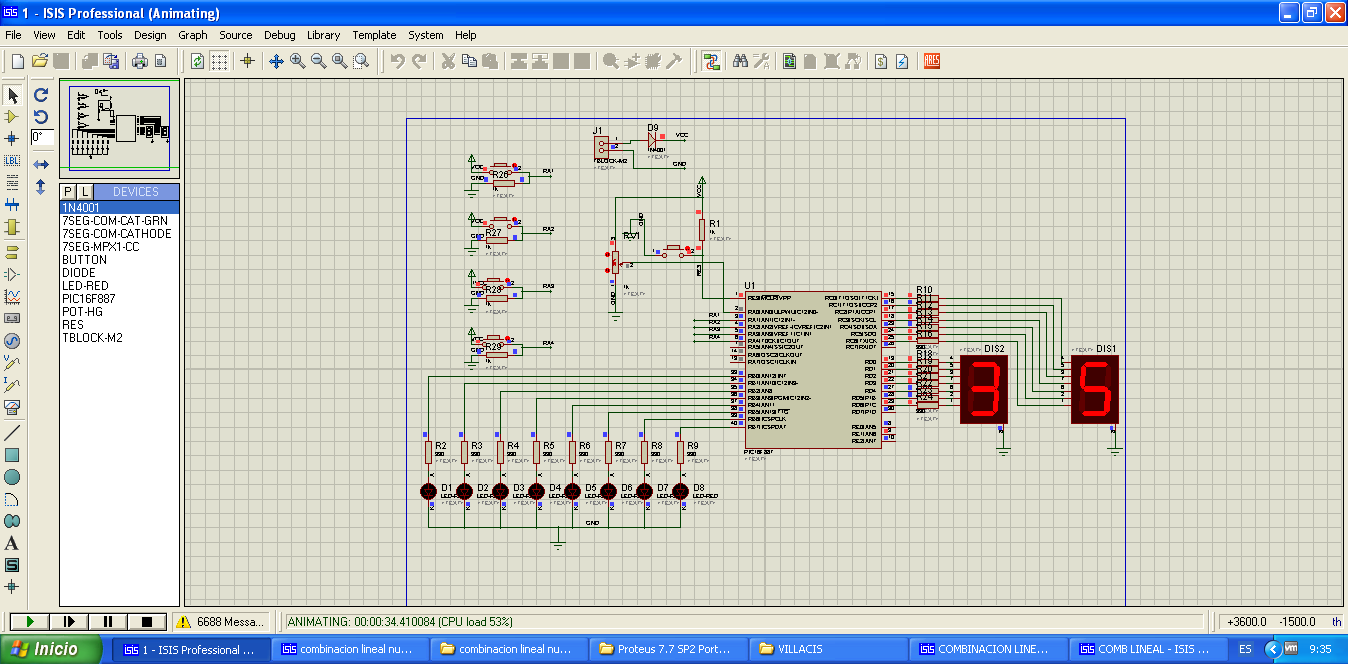
}

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

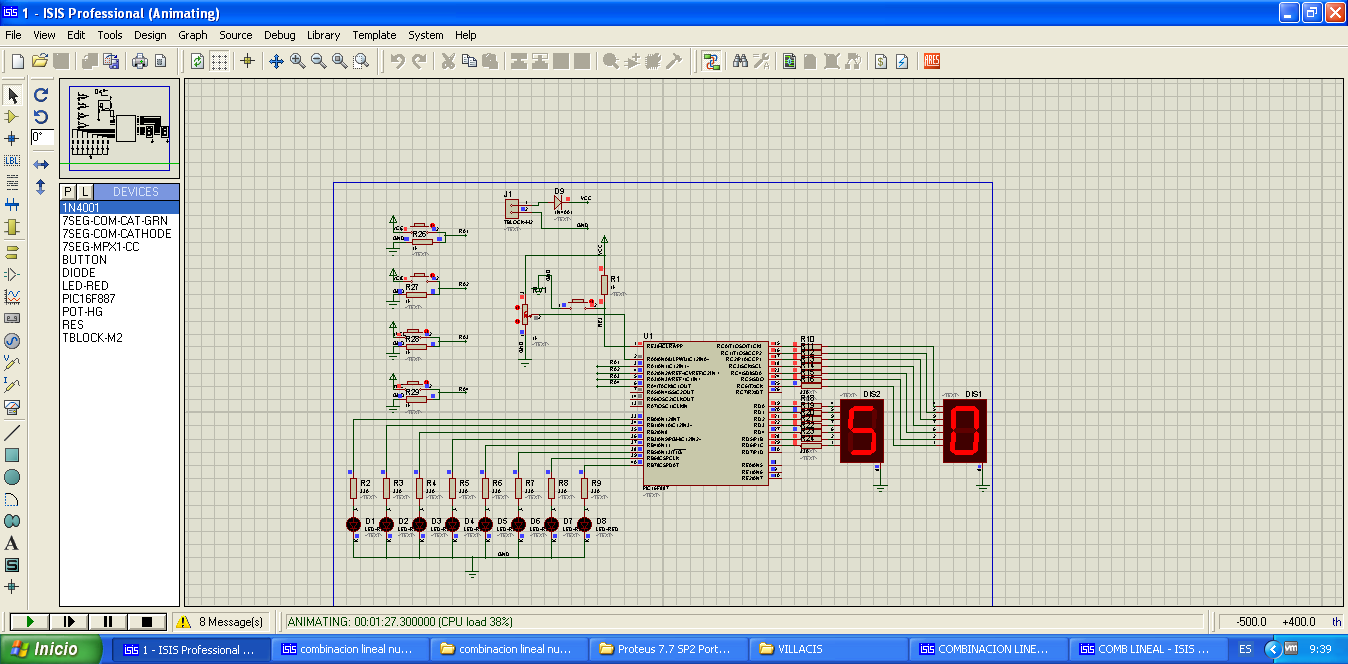
Mostrando las iniciales



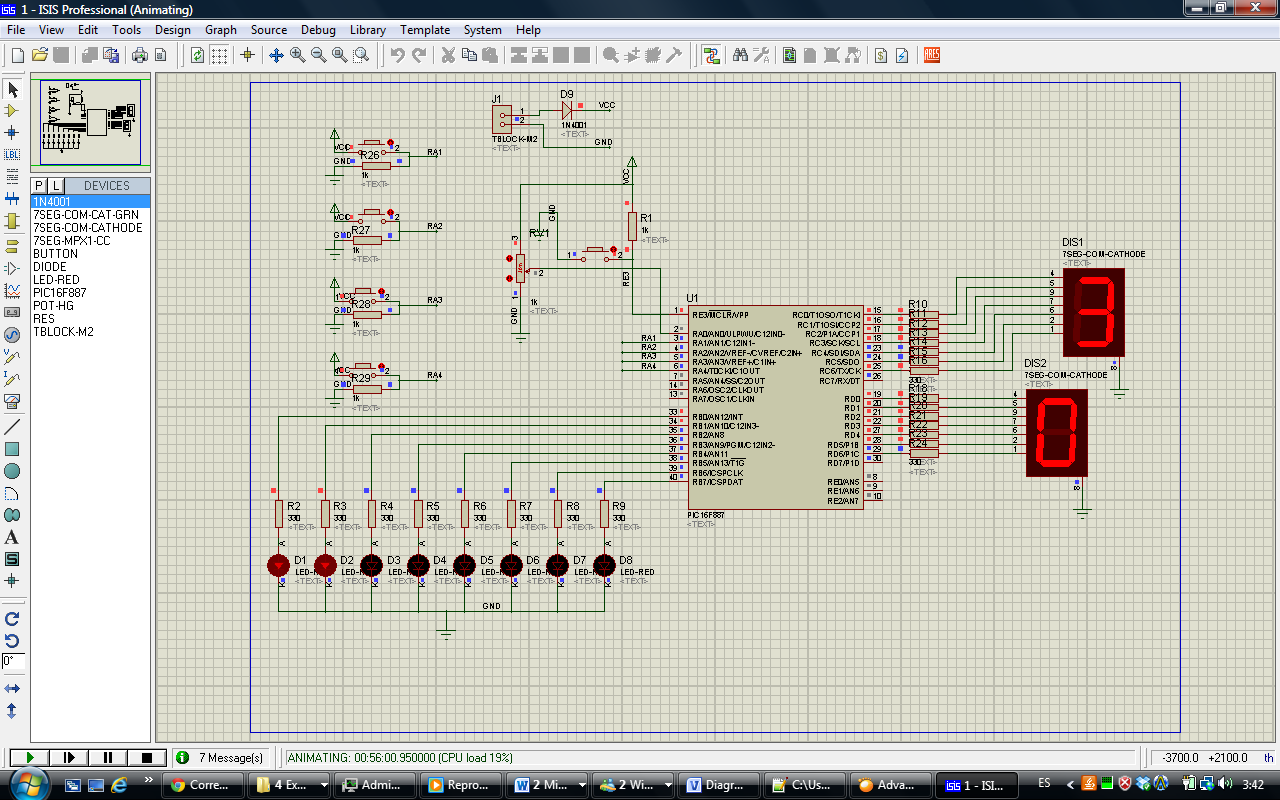
Colocando el Ejercicio 1 con un valor del potenciómetro al 35%.



Número a adivinar en el ejercicio 2, en este caso se ajustó el 5.



Cuando se muestra el número correcto. En este caso el 3.



**7.- Conclusiones**

* Para obtener un valor de lectura de una señal analógica se utiliza el convertidor ADC que mediante la manipulación del modelo matemático de la conversión se puede obtener un valor en binario que está en función del valor que el ADC decodifica, para esto es necesario configurar una de los pines ANx como entrada analógica para recibir la señal por aquel canal.
* La utilización de un lenguaje de alto nivel para programar Microcontroladores reduce considerablemente el esfuerzo de codificación, escatima detalles y permite un manejo eficiente de recursos de la arquitectura ya que las funciones que se proveen se encargan de la manipulación interna. Además ofrece una visión más abstracta que permite una mayor creatividad a la hora de construir una aplicación como por ejemplo el juego realizado en este proyecto.
* Para generar números aleatorios se utilizan las funciones rand() y srand() que aplicadas correctamente pueden darnos buenos resultados, aunque rand() no se ajusta al estándar de C ya que en la versión de MicroC el resultado es un entero entre 0 y 32727, además para ajustar la aleatoriedad se utiliza srand() que en este caso (es decir el presente proyecto) el parámetro enviado es un contador que constantemente se incrementa y que en cierto punto se desborda y vuelve a contar, de esta manera nos aseguramos que los números sean lo más aleatorios posible.

**8.- Recomendaciones**

* Para la conversión con el ADC no olvidar configurar como entrada analógica el pin receptor de la señal, ya que no se obtendrán los resultados deseados y la lectura del dato no será congruente con lo que verdaderamente debió ser leído.
* Para generar números aleatorios tratar de generarlos lo más randómico posible (esto se logra con la correcta manipulación de srand() ) , ya que de no ser así es posible que siempre nos arroje un mismo número bajo las mismas condiciones de ejecución.
* Para calcular un número aleatorio entre un límite inferior y uno superior, tener en cuenta el funcionamiento de rand() para manipularlo matemáticamente de una manera adecuada, quizás existen otros casos en que rand() tiene una diferente implementación y obligue a usar otro modelo matemático .
* Tener cuidado con la interacción de la interfaz de entrada (botoneras) con el usuario para no causar confusión a la hora de la utilización de la aplicación.