**ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL**

**FIEC**

**Laboratorio de Microcontroladores**

**PROYECTO**

**Segundo Parcial**

**Alumna**

**Diana Decimavilla Alarcon**

**Paralelo: # 11**

**Grupo: # 3**

**Fecha de presentación:**

**26 / 01 / 2012**

**2011 – 2° TÉRMINO**

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 juego de memoria 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:**

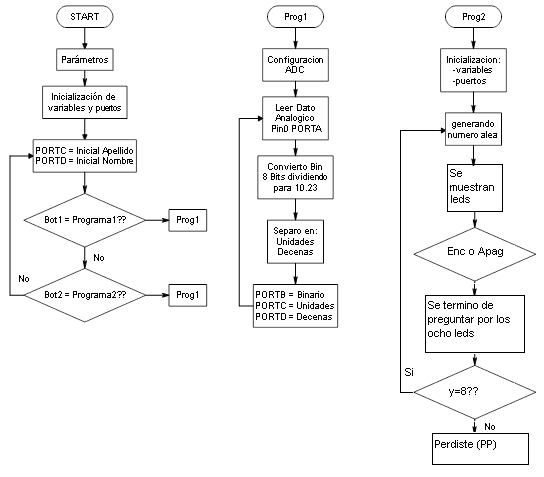
El PROGRAMA1 consiste en hacer uso del módulo ADC que posee el microcontrolador. Lee el valor analógico del potenciómetro conectado en el PORTA y muestra su equivalente digital DECIMAL: unidades y decenas en los dos DISPLAYs conectados en los puertos C y D respectivamente; y su valor BINARIO en los 8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

El PROGRAMA2 consiste en un juego de memoria LEDs en el cual se muestran 8 LED’s aleatoriamente encendidos o apagados por tres segundos, luego de este tiempo, se apagan y debes acertar led a led si este se encontraba encendido o apagado, empezamos por el bit menos significativo, si aciertas los ocho leds, se vuelve a generar otro número aleatorio pero esta vez lo muestra únicamente por dos segundos y si vuelve a acertar, se mostrara otro número aleatorio por un segundo para finalmente indicarte que ganaste (GG), y si en cualquier momento del juego no aciertas, te saldrá (PP) que significa que has perdido.

**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 número 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 número correctamente.

Si se presiona la BOTONERA2, se ejecuta el PROGRAMA2: Se inicializan las variables y puertos a usar; se genera un número y se muestra en los leds por cierto tiempo, luego recorre led por led preguntando si estaba encendido o apagado, al final de la ronda, si acertaste subes al siguiente nivel, si no acertaste todas, perdiste y te muestra en el display PP.

**Programa Fuente:**

/\*

\* Nombre del Proyecto:

DianaDecimavilla.c

\* Nombre del Autor:

Diana Decimavilla (2012)

\* Description:

Modulo AC-DC y Juego

\* Test configuration:

MCU: PIC16F887

Oscillator: HS, 08.0000 MHz

SW: mikroC PRO for PIC

\* NOTES:

\*/

unsigned short mask(unsigned short num);

unsigned long int temp\_res;

unsigned int Var1;

unsigned int unidad,decena;

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; // PORTC is output

TRISC = 0; // PORTC is output

TRISD = 0;

PORTC=0;

PORTD=0;

do{

PORTB=0;

PORTC=0x5E;// d

PORTD=0x5E; //d

}while(RA1\_bit=='0' && RA2\_bit=='0');

if(RA1\_bit=='1'){

do {

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

Var1=(temp\_res\*100/1024);

decena=Var1/10;

unidad=Var1%10;

PORTB=Var1;

PORTC=mask(unidad);

PORTD=mask(decena);

} while(1);//

}

Var1=0;

if(RA2\_bit=='1'){

PORTC=0x3F;

PORTD=0x3F;

delay\_ms(1000);

PORTB=0x76;

delay\_ms(3000);

PORTB=0;

do{

RB0\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA3\_bit=='1'){

Var1++;

}

delay\_ms(1000);

PORTB=0;

do{

RB1\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA4\_bit=='1'){

Var1++;

}

delay\_ms(1000);

PORTB=0;

do{

RB2\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA4\_bit=='1'){

Var1++;

}

delay\_ms(1000);

PORTB=0;

do{

RB3\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA3\_bit=='1'){

Var1++;

}

delay\_ms(1000);

PORTB=0;

do{

RB4\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA4\_bit=='1'){

Var1++;

}

delay\_ms(1000);

PORTB=0;

do{

RB5\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA4\_bit=='1'){

Var1++;

}

delay\_ms(1000);

PORTB=0;

do{

RB6\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA4\_bit=='1'){

Var1++;

}

delay\_ms(1000);

PORTB=0;

do{

RB7\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA3\_bit=='1'){

Var1++;

}

PORTB=Var1;

delay\_ms(3000);

if(Var1==8){

PORTB=0;

PORTC=0x1F;

PORTD=0x3D;

delay\_ms(2000);

Var1=0;

PORTC=0x3F;

PORTD=0x3F;

delay\_ms(1000);

PORTB=0xCA;

delay\_ms(2000);

PORTB=0;

do{

RB0\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA3\_bit=='1'){

Var1++;

}

delay\_ms(1000);

PORTB=0;

do{

RB1\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA4\_bit=='1'){

Var1++;

}

delay\_ms(1000);

PORTB=0;

do{

RB2\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA3\_bit=='1'){

Var1++;

}

delay\_ms(1000);

PORTB=0;

do{

RB3\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA4\_bit=='1'){

Var1++;

}

delay\_ms(1000);

PORTB=0;

do{

RB4\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA3\_bit=='1'){

Var1++;

}

delay\_ms(1000);

PORTB=0;

do{

RB5\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA3\_bit=='1'){

Var1++;

}

delay\_ms(1000);

PORTB=0;

do{

RB6\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA4\_bit=='1'){

Var1++;

}

delay\_ms(1000);

PORTB=0;

do{

RB7\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA3\_bit=='1'){

Var1++;

}

PORTB=Var1;

delay\_ms(1000);

if(Var1==8){

PORTB=0;

PORTC=0x1F;

PORTD=0x3D;

delay\_ms(3000);

PORTC=0x3F;

PORTD=0x3F;

delay\_ms(1000);

PORTB=0x65;

delay\_ms(1000);

PORTB=0;

do{

RB0\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA4\_bit=='1'){

Var1++;

}

delay\_ms(1000);

PORTB=0;

do{

RB1\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA3\_bit=='1'){

Var1++;

}

delay\_ms(1000);

PORTB=0;

do{

RB2\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA4\_bit=='1'){

Var1++;

}

delay\_ms(1000);

PORTB=0;

do{

RB3\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA3\_bit=='1'){

Var1++;

}

delay\_ms(1000);

PORTB=0;

do{

RB4\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA3\_bit=='1'){

Var1++;

}

delay\_ms(1000);

PORTB=0;

do{

RB5\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA4\_bit=='1'){

Var1++;

}

delay\_ms(1000);

PORTB=0;

do{

RB6\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA4\_bit=='1'){

Var1++;

}

delay\_ms(1000);

PORTB=0;

do{

RB7\_bit=1;

}while(RA3\_bit=='0'&&RA4\_bit=='0' );

if(RA3\_bit=='1'){

Var1++;

}

PORTB=Var1;

delay\_ms(3000);

if(Var1==8){

PORTB=0;

PORTC=0x1F;

PORTD=0x3D;

}

else{

PORTB=0;

PORTC=0x1F;

PORTD=0x73;

}

}

else{

PORTB=0;

PORTC=0x1F;

PORTD=0x73;

}

}

else{

PORTB=0;

PORTC=0x1F;

PORTD=0x73;

}

}

}

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;

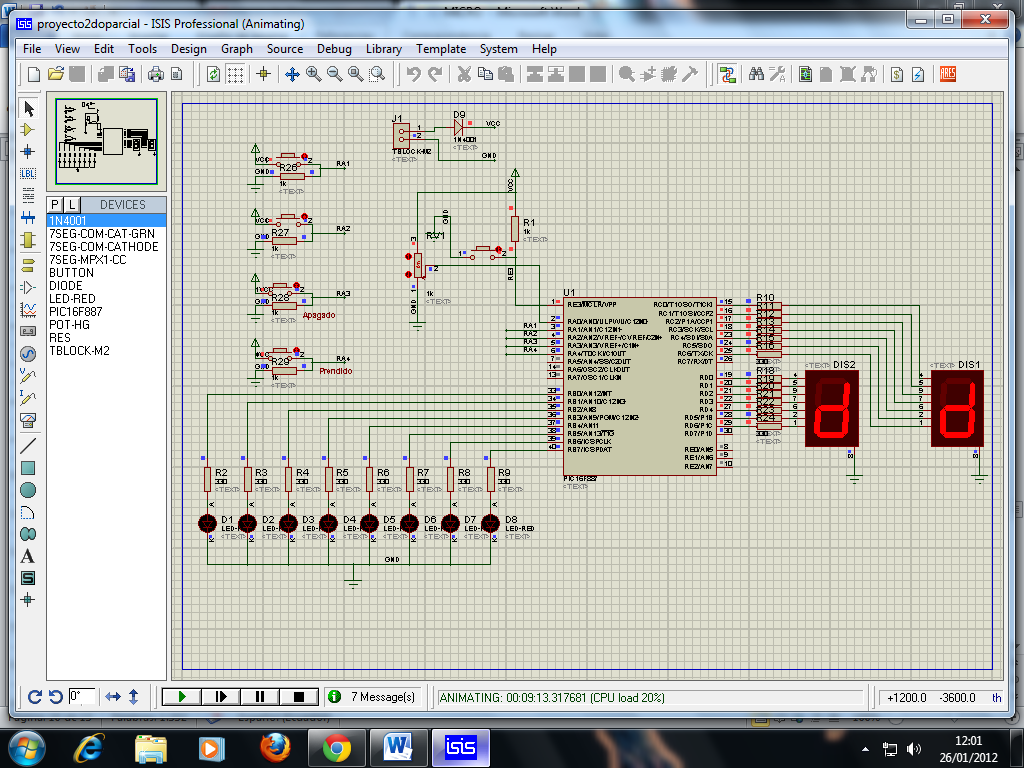
case 9 : return 0x6F;

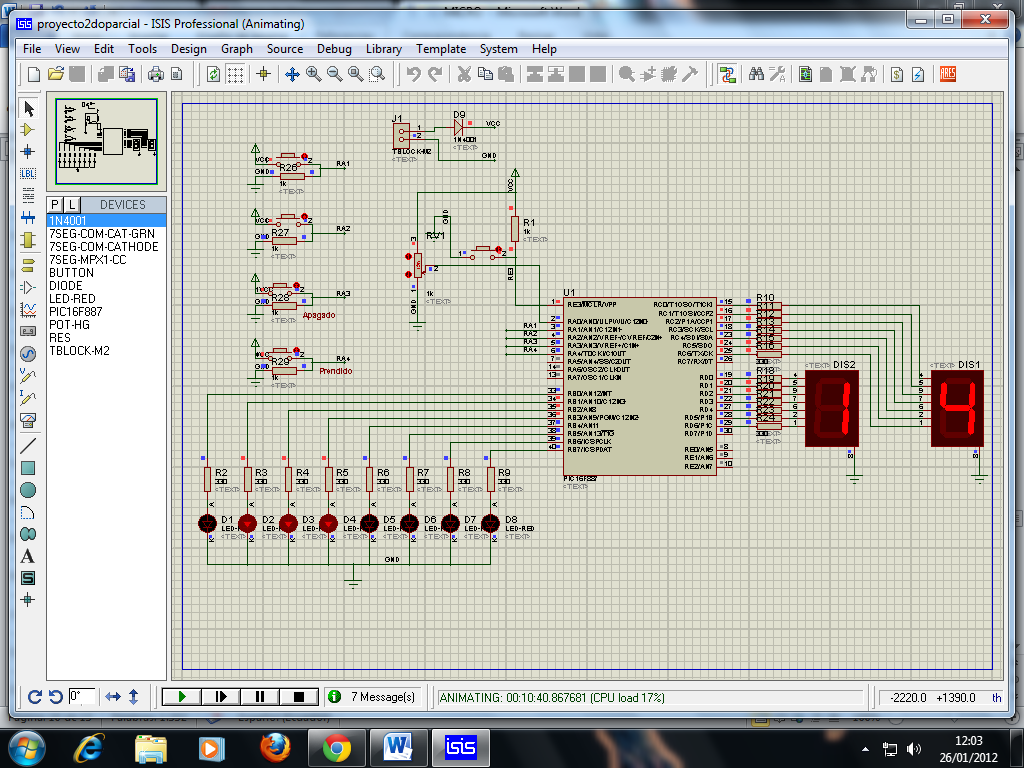
}

}

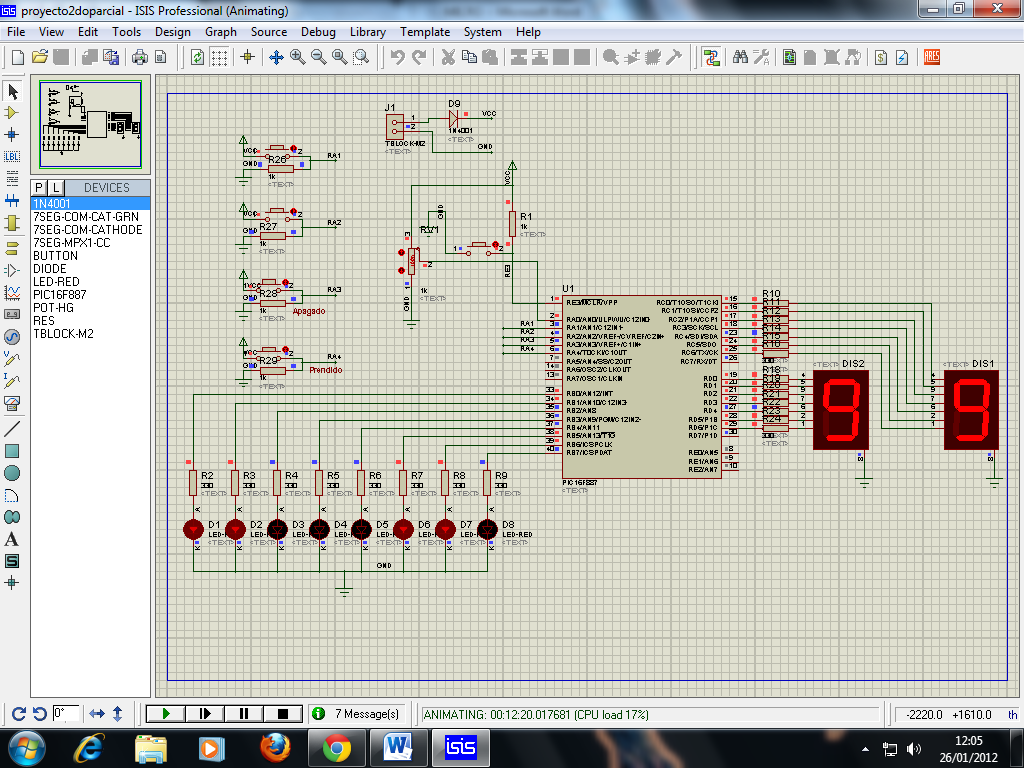
**Circuito armado en Proteus:**

Se muestran las iniciales de mi nombre y apellido A, R (Andres Romero) debido a que la letra M no se puede apreciar bien

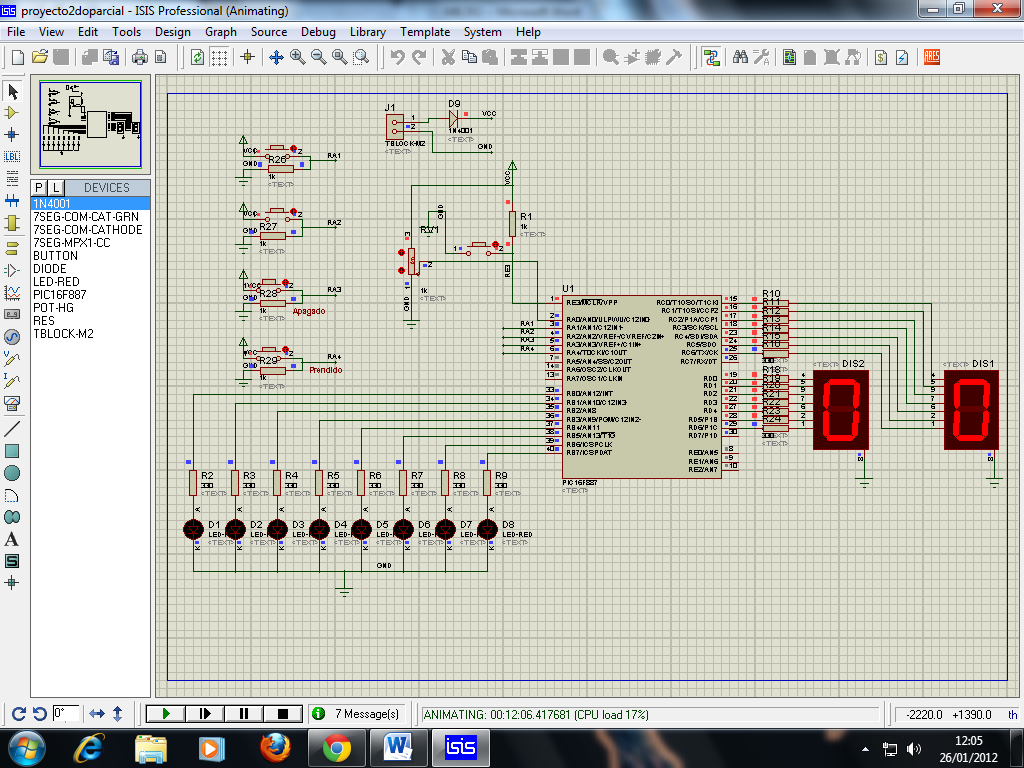




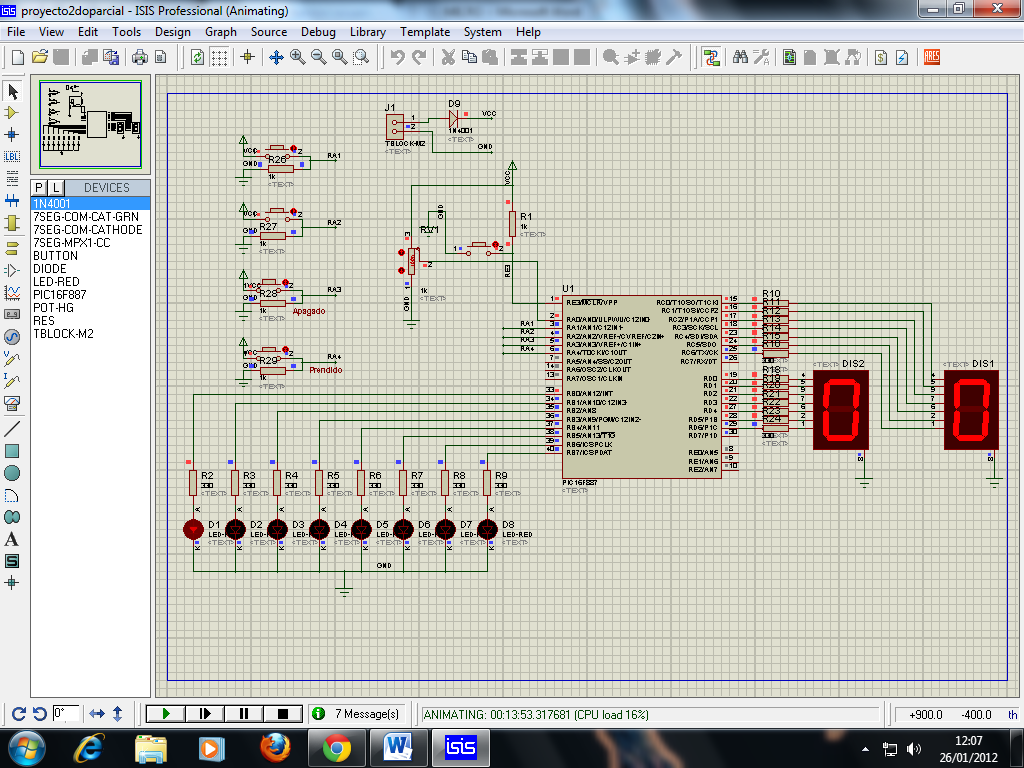
Valor máximo 99



Valor mínimo 00



Se presiona RA2



**Conclusiones:**

* Para este proyecto se pudo realizar gran cantidad de juegos con el lenguaje .C y los elementos que se nos designaron, ya que son un poco mas sencillos de programar que el lenguaje .asm y también su nivel de comprensión, se puede utilizar fácilmente la imaginación para variar cuantas veces uno quiera, aumentarle la dificultad, el tiempo o las variables.
* Para este juego que seleccione, se debe de superar tres niveles para poder ganar, pero esto no lo hace aburrido ni cansado, ya que a mayor nivel de juego, menor será el tiempo en que se muestran los leds, y también menor será el tiempo que tendrá el jugador para pensar en si estaba prendido o apagado, y uno no podrá saber si ganó o perdió hasta terminar una ronda.
* Para este proyecto se aplicaron todos los conocimientos básicos y teóricos que se pudieron aprender en las clases teóricas y las prácticas de laboratorio, ya que todo esto fue muy esencial a la hora de programar, debiéndose saber todos los comando y los tipos con las declaraciones de las variables, y siempre estando presente el reto de soldar los elementos a la placa.

**Recomendaciones:**

* Se recomienda que al momento de soldar los elementos a la placa, se utilice un buen cautín, de la punta mas fina posible, y también una tira de estaño de la mas delgada, para así evitar colocar demasiado estaño sobre las conexiones y mejorar la presentación
* También recomiendo leerse el funcionamiento de los displays de siete segmentos para así saber cual sección deben activarse y cuales no para mostrar cierto tipo de letra o número el cual nosotros quisiéremos mostrar.
* También recomiendo leerse en Internet los códigos hexadecimales, ya que en este juego se los utilizó de manera repetitiva, para mostrarlos en los leds y en los displays de siete segmento y así generar la letra o palabra deseada.