**ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL**



**LABORATORIO DE MICROCONTROLADORES**

**Proyecto Final:**

**LECTURA ADC**

**ADIVINADOR DE CONTRASEÑA**

**Nombre:**

* **Roberto Noboa Gárate**

**Paralelo: 5 Grupo: 5**

**2012 – TÉRMINO I**

**Fecha de presentación: Agosto 23 / 2012**

**1.- Enunciado del Proyecto**

Este programa hemos desarrollado un Controlador de acceso, para validar contraseña, el proyecto dará inicio al presionar en el teclado el botón ON/C para acceder al menú, luego si presionamos el botón l del teclado, entra al convertidor ADC donde dará lectura de los valores analógicos de un potenciómetro colocado en la patita RE0/AN5 del PIC16F887.

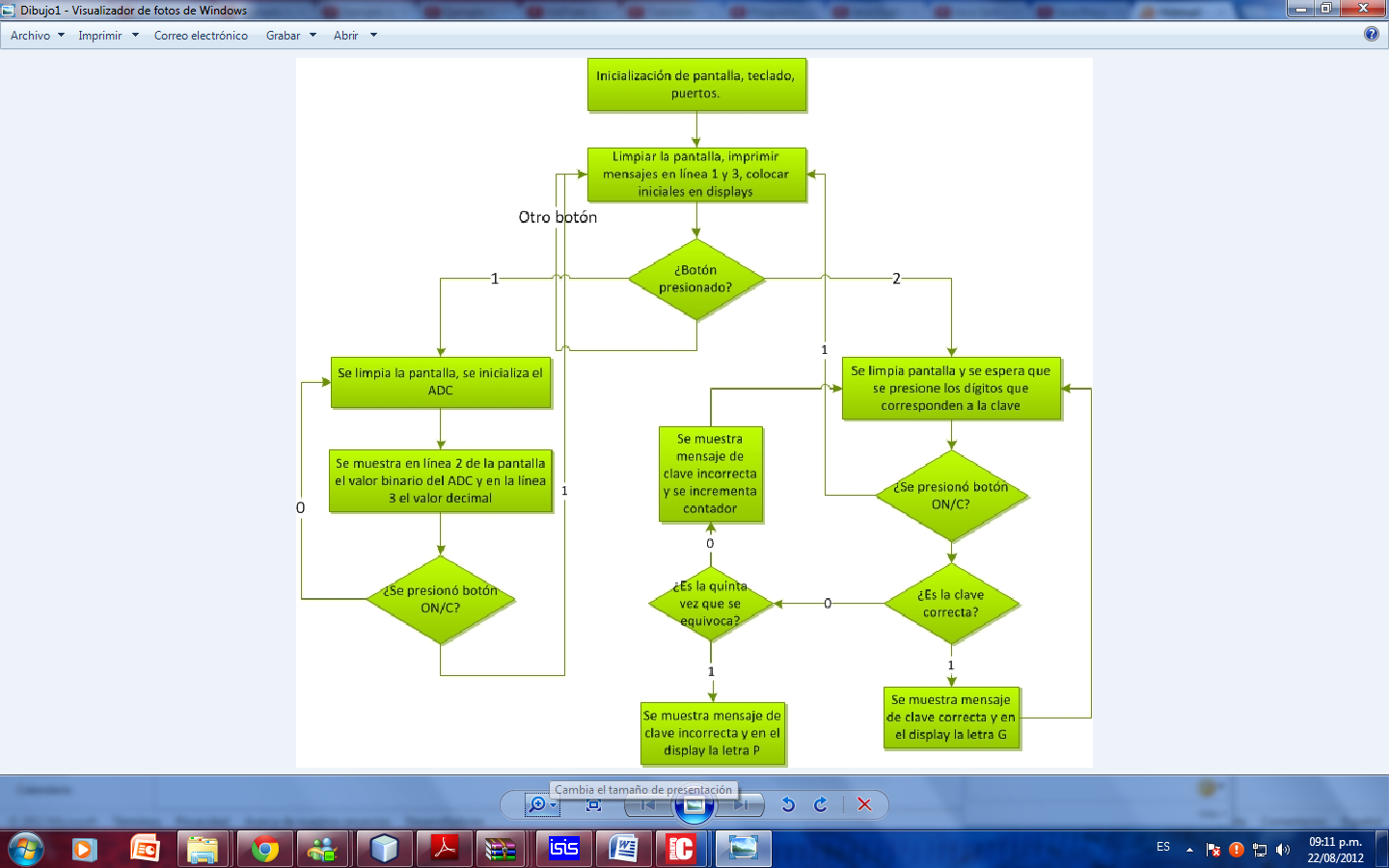
Luego al presionar la tecla del botón 2 ingresamos al juego adivinador de clave (Control de acceso), la clave correcta es 357, donde el juego permite 5 intentos para adivinar al 5 intento le aparece en el display una P (Perdió) y sí acierta en el display aparece una G (Ganó).

Para regresar al menú presionamos el botón ON/C del teclado.

**2.- Diagrama de Bloques**



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



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

1. Se inicializan la LCD el teclado y la comunicación usar con una velocidad de 9600 baudios, esto es importante para poder usar las funciones de las librerías de los dispositivos que vamos a usar.
2. Se inicializa en teclado y se le asigna un puerto el cual requerimos como entrada para poder obtener datos del usuario.
3. Se espera a que se presione la tecla ON/C para el menú principal del teclado 4x4.
4. Luego se espera a que selección una de la opciones 1 para el convertidor ADC que usando el Potenciómetro.
5. Se inicia el juego en presionando 2 en el teclado donde en 5 turnos tiene la posibilidad de acertar y ganar como máximo.

**6.- Listado del programa fuente en lenguaje c con comentarios en las líneas de código que considere fundamentales**

/\*

\* Nombre del Proyecto:

Controlador de acceso.c

\* Nombre del Autor:

Roberto Noboa

\* Descripción:

1. Es el convertidor ADC que que consiste en la lectura de los valores analógicos de un potenciómetro

colocado en patita RE0/AN5 del PIC16F887,

presentando las variaciones de valores en binario y su equivalente decimal en una pantalla LCD 4x20.

2. Se tiene que ingresar 3 dígitos y esto se comparará con

un arreglo que contiene la clave y de ser correcto lo que se

ingresó por aparece ne le display la G que indica que ganó si despues de 5 intentos

no acierta aparece en el display la P que indicá perdio...

\* Test configuration:

MCU: PIC16F887

Oscillator: HS, 08.0000 MHz

SW: mikroC PRO for PIC

unsigned short mask(unsigned short num);

unsigned short kyp(unsigned short kp);

unsigned short clave(char txt[],char txt2[],short size);

unsigned long a,tmp;

unsigned int q;

unsigned short u,kp,m,n,o,az,cnt=1,n2,p1=0,p2;

char cente[5],cst2[6]="357",txt2[6];

// Keypad module connections

char keypadPort at PORTD;

// End Keypad module connections

// LCD module connections

sbit LCD\_RS at RB4\_bit;

sbit LCD\_EN at RB5\_bit;

sbit LCD\_D4 at RB0\_bit;

sbit LCD\_D5 at RB1\_bit;

sbit LCD\_D6 at RB2\_bit;

sbit LCD\_D7 at RB3\_bit;

sbit LCD\_RS\_Direction at TRISB4\_bit;

sbit LCD\_EN\_Direction at TRISB5\_bit;

sbit LCD\_D4\_Direction at TRISB0\_bit;

sbit LCD\_D5\_Direction at TRISB1\_bit;

sbit LCD\_D6\_Direction at TRISB2\_bit;

sbit LCD\_D7\_Direction at TRISB3\_bit;

// End LCD module connections

void main() {

ANSEL = 0x20; // Configure AN2 pin as analog

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

C1ON\_bit = 0; // Disable comparators

C2ON\_bit = 0;

TRISA = 0; // PORTA is input

TRISC = 0; // PORTC is output

Keypad\_Init(); // Initialize Keypad

Lcd\_Init(); // Initialize Lcd

men:

Lcd\_Cmd(\_LCD\_CLEAR); // Clear display

Lcd\_Cmd(\_LCD\_CURSOR\_OFF); // Cursor off

do{

PORTA=0X77;

PORTC=0X37;

Lcd\_Out(1, 1, "1. PARTE A");

Lcd\_Out(3, 1, "2. PARTE B");

kp = 0;

do

kp = Keypad\_Key\_Click(); // Store key code in kp variable

while (!kp);

az=kyp(kp);

if(az==49){

Lcd\_Cmd(\_LCD\_CLEAR);

ADC\_Init();

while(1){

PORTA=0;

PORTC=0;

Lcd\_Out(1, 4, "CONVERSION ADC");

tmp=ADC\_Read(5);

a=(tmp\*489);

m=a/100000;

n=(a%100000)/10000;

q=(a%100000)%10000;

o=q/1000;

u=(q%1000)/100;

p2=tmp;

p2&=0x01;

ShortToStr(p2,cente);

Lcd\_Chr(2, 15, cente[3]);

p2=tmp>>1;

p2&=0x01;

ShortToStr(p2,cente);

Lcd\_Chr(2, 14, cente[3]);

p2=tmp>>2;

p2&=0x01;

ShortToStr(p2,cente);

Lcd\_Chr(2, 13, cente[3]);

p2=tmp>>3;

p2&=0x01;

ShortToStr(p2,cente);

Lcd\_Chr(2, 12, cente[3]);

p2=tmp>>4;

p2&=0x01;

ShortToStr(p2,cente);

Lcd\_Chr(2, 11, cente[3]);

p2=tmp>>5;

p2&=0x01;

ShortToStr(p2,cente);

Lcd\_Chr(2, 10, cente[3]);

p2=tmp>>6;

p2&=0x01;

ShortToStr(p2,cente);

Lcd\_Chr(2, 9, cente[3]);

p2=tmp>>7;

p2&=0x01;

ShortToStr(p2,cente);

Lcd\_Chr(2, 8, cente[3]);

p2=tmp>>8;

p2&=0x01;

ShortToStr(p2,cente);

Lcd\_Chr(2, 7, cente[3]);

p2=tmp>>9;

p2&=0x01;

ShortToStr(p2,cente);

Lcd\_Chr(2, 6, cente[3]);

ShortToStr(m,cente);

Lcd\_Chr(3, 9, cente[3]);

Lcd\_Chr(3, 10,'.');

ShortToStr(n,cente);

Lcd\_Chr(3, 11, cente[3]);

ShortToStr(o,cente);

Lcd\_Chr(3, 12, cente[3]);

ShortToStr(u,cente);

Lcd\_Chr(3, 13, cente[3]);

kp = Keypad\_Key\_Press(); // Store key code in kp variable

az=kyp(kp);

if (az==42)

goto men;

}

}

if(az==50){

Lcd\_Cmd(\_LCD\_CLEAR); // Clear display

do{

PORTA=0;

PORTC=mask(p1%10);

Lcd\_Out(1, 1, "Ingrese clave:");

kp = 0;

do

kp = Keypad\_Key\_Click(); // Store key code in kp variable

while (!kp);

az=kyp(kp);

if (az==42)

goto men;

Lcd\_Out(2, cnt, "\*"); //imprime en pantalla la contraseña

txt2[cnt-1]=az;

cnt=cnt++;

if(cnt==4){

n2=clave(txt2,cst2,6);

if(n2==1)

{

if (p1<5){

PORTA=0x7D;

p1=0;

}

Lcd\_Cmd(\_LCD\_CLEAR);

Lcd\_Out(2,1,"CLAVE CORRECTA");

Lcd\_Out(4,1,"ACCESO");

Delay\_ms(2000);

}

else{

p1++;

PORTC=mask(p1%10);

if (p1==5){

PORTA=0x73;

p1=0;

}

Lcd\_Cmd(\_LCD\_CLEAR);

Lcd\_Out(1,1,"CLAVE INCORRECTA");

Lcd\_Out(3,1,"ACCESO DENEGADO");

Delay\_ms(2000);

}

Lcd\_Cmd(\_LCD\_CLEAR);

cnt=1;

}

} while (1);

}

} while (1);

}

unsigned short clave(char txt[],char txt2[],short size){

unsigned short i,x=0;

for(i=0;i<size;i++)

{

if(txt[i]==txt2[i]){

x++;

}

}

if(x==size){

return 1;

}

else return 0;

}

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 kyp(unsigned short kp) {

switch (kp) {

case 1: return 55; // 7 // Uncomment this block for keypad4x4

case 2: return 52; // 4

case 3: return 49; // 1

case 4: return 42; // \*

case 5: return 56; // 8

case 6: return 53; // 5

case 7: return 50; // 2

case 8: return 48; // 0

case 9: return 57; // 9

case 10: return 54; // 6

case 11: return 51; // 3

case 12: return 35; // #

case 13: return 65; // A

case 14: return 66; // B

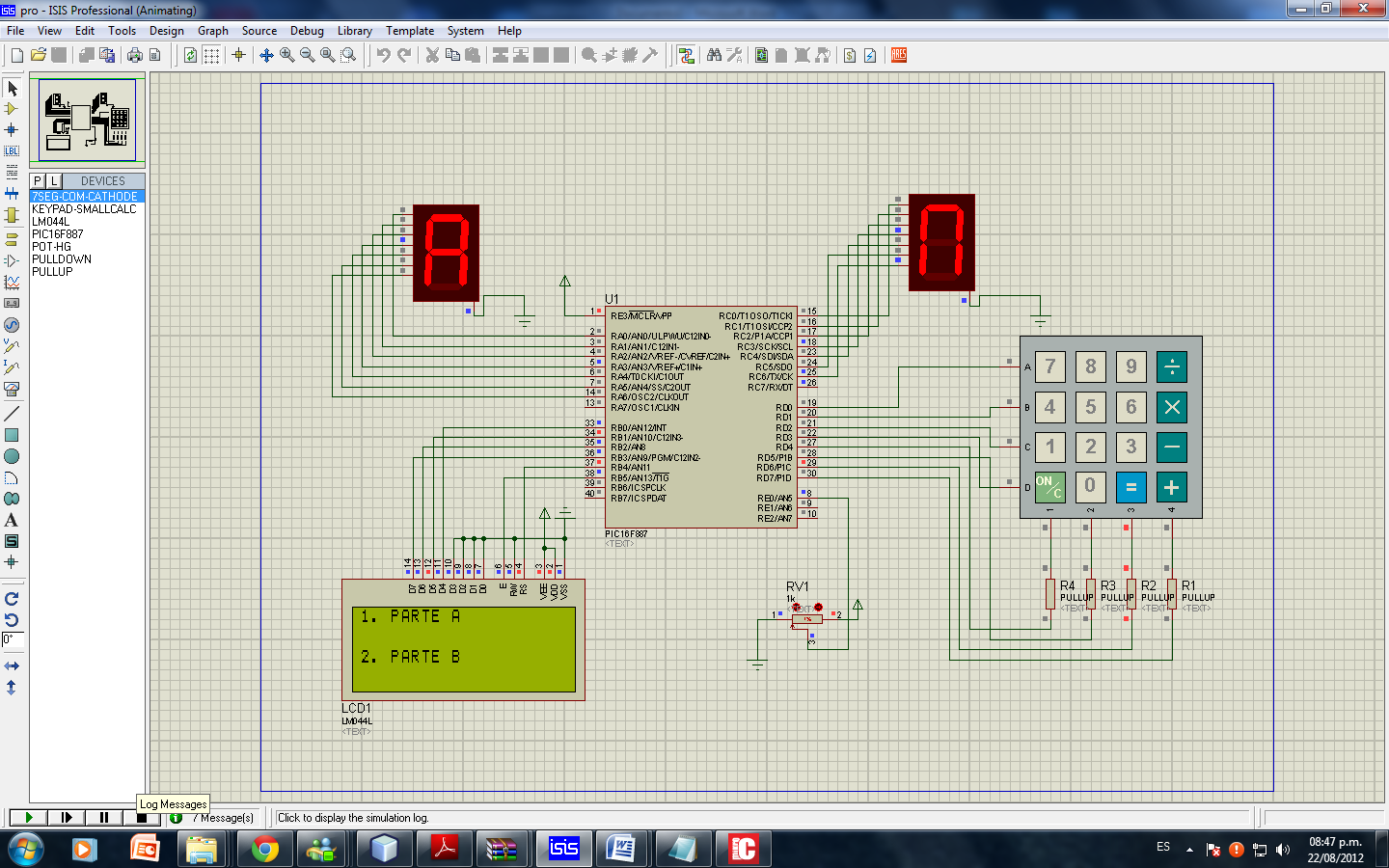
case 15: return 67; // C

case 16: return 68; // D

}

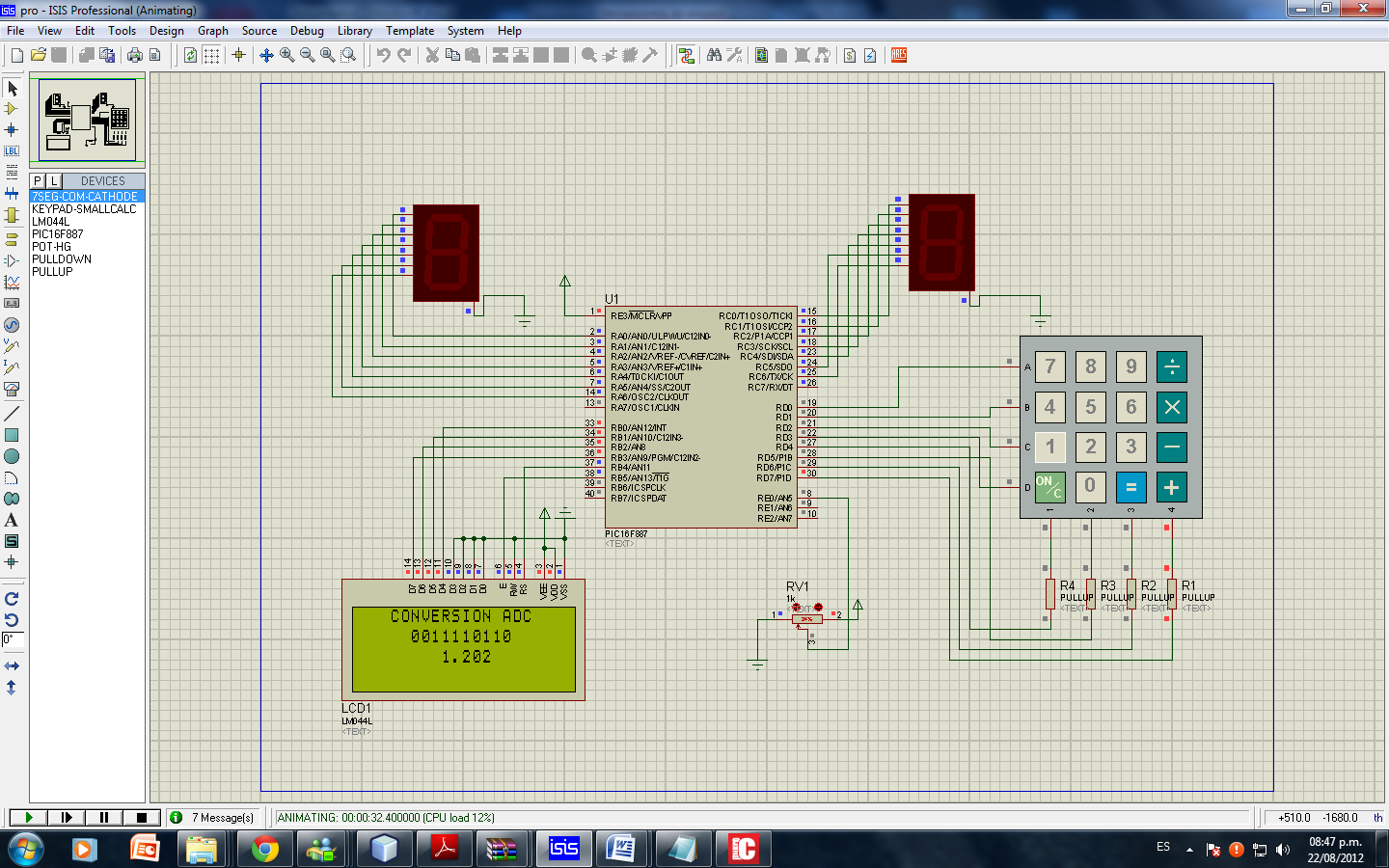
}

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



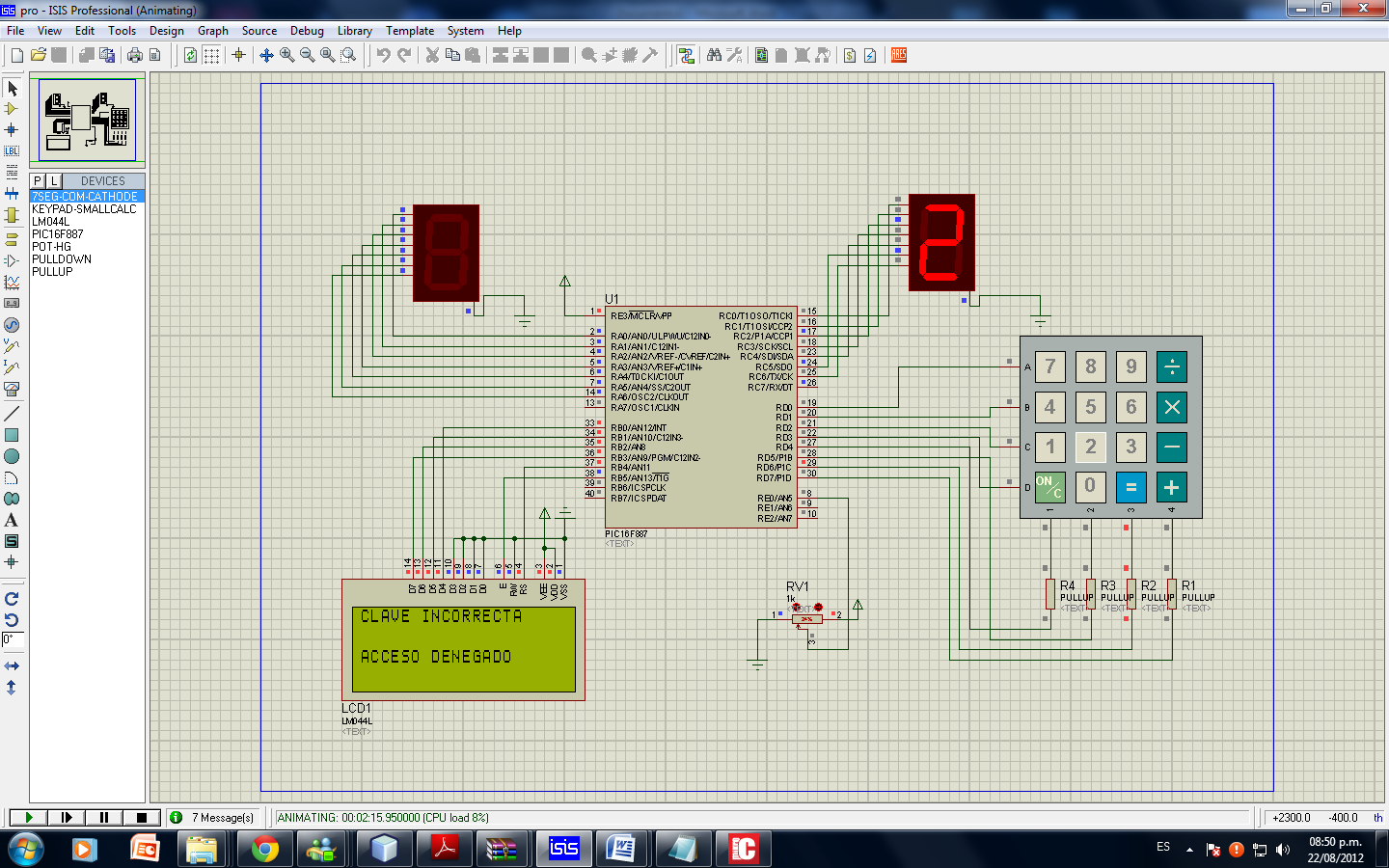
***Sección 1***

PARTE A

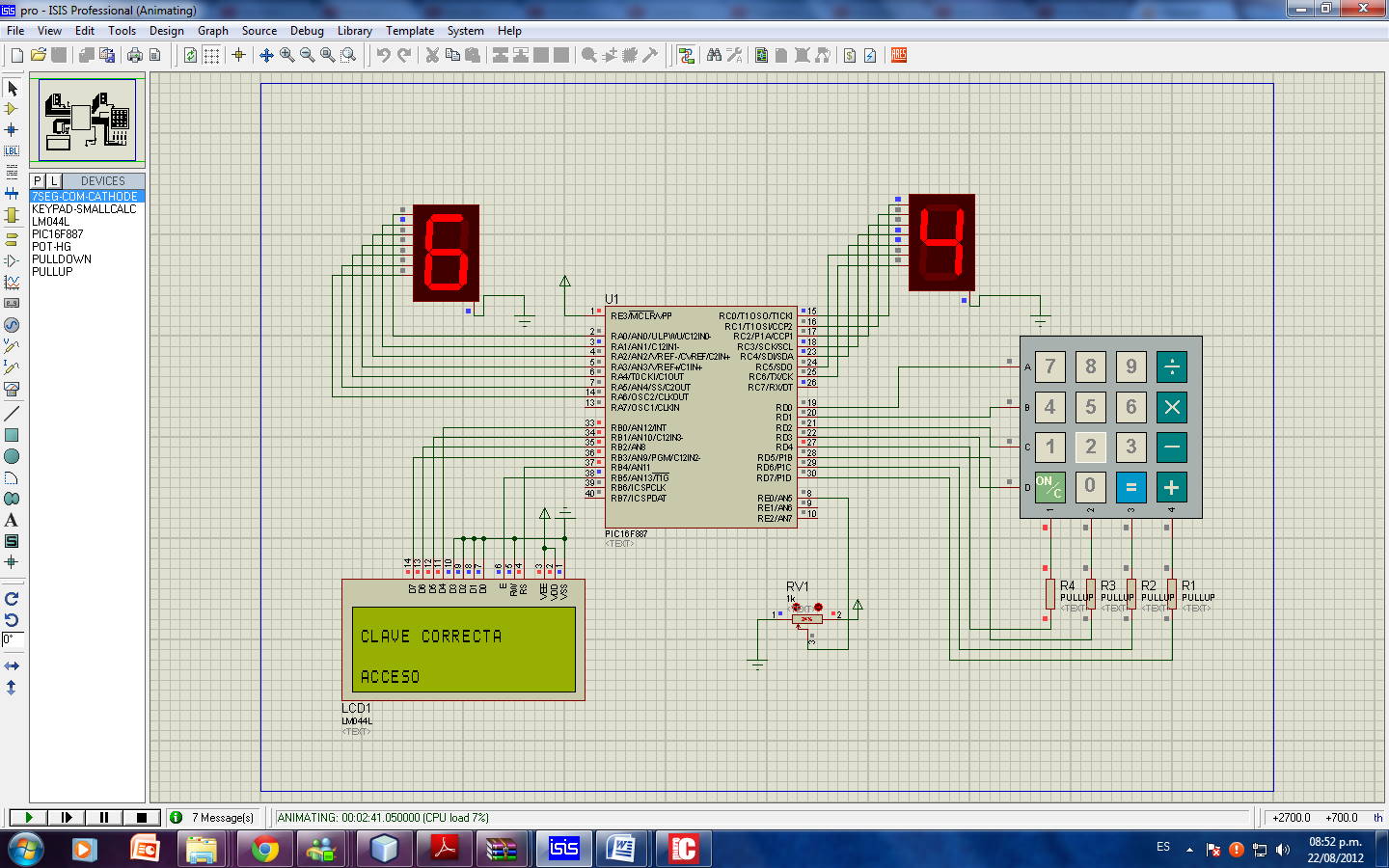


***Sección 2***

PARTE B



Clave correcta Display1 G (Ganó) y el Display2 (Indica los turnos “4”)



**8.- CONCLUSIONES**

* Se logró aprender las sentencias que se usan en lenguaje C y sus diferentes formas de uso, lo cual me permitió realizar el proyecto de forma secuencial y entendible para cualquier usuario, sin importar las veces que se usen estas funciones lo importante es saber si solucionan el problema que se nos presenta en cada línea de programación.
* Se usó de las librerías de LCD, UART y teclado para la realización de este proyecto ya que se debía cumplir al mismo tiempo, por un lado se preguntada por la botonera, y se debía obtener una respuesta dependiendo de cómo el usuario avanzar dentro del juego para lo cual se necesita presentar información para que el usuario comprendiera y avanzara de forma correcta.
* Se puede concluir como punto final que el lenguaje C debe de poseer un orden estructurado lo que quiere decir si se hace muchas funciones estas deben de ser explicadas y tener valores de retorno al main en momentos críticos como finales de contadores o preguntar por cambios de valor en los puertos para así poder evitar problemas uso excesivo de la ram del microprocesador ya que no es muy grande o problemas de creación del punto ejecutable.

**9.- RECOMENDACIONES**

* Se recomienda no confiar en un 100% en el simulador de PROTEUS ya que este presenta falencia sobre todo en lo que tiene que ver con frecuencia en el simulador.
* Es recomendable siempre verificar que todos los componentes se encuentren ajustados a la misma velocidad, en el caso de la comunicación UART deben estar a la misma frecuencia de bit caso contrario solo obtenemos basura.
* 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.