UNIVERSIDAD NACIONAL EXPERIMENTAL POLITECNICA

“ANTONIO JOSE DE SUCRE”

VICE-RECTORADO DE PUERTO ORDAZ

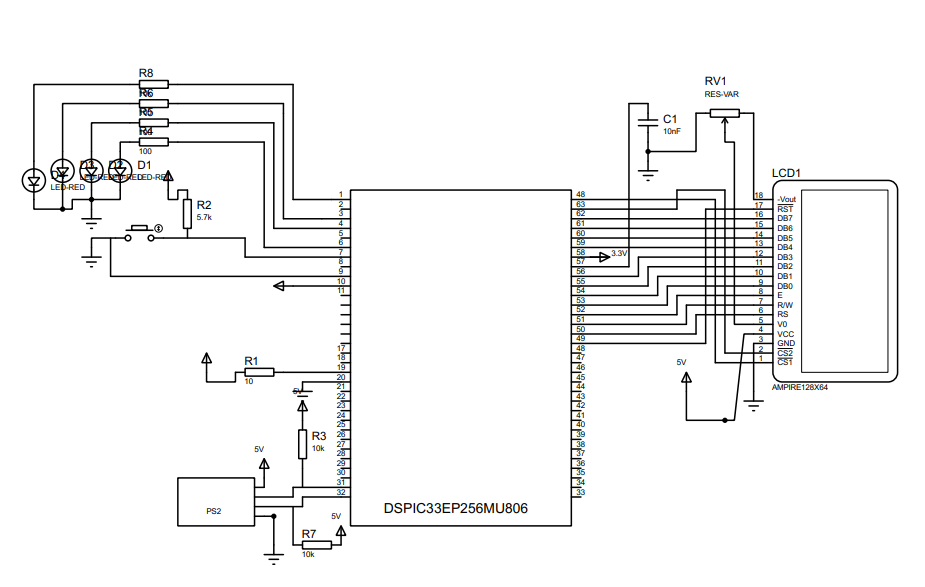
DEPARTAMENTO DE ELECTRONICA

**MICROPROCESADORES II**

Laboratorio 2

|  |  |  |
| --- | --- | --- |
| Profesor: |  | Bachilleres: |
| Rafael Surga Luvo |  | Brito Eduardo C.I:24560857 |
|  |  | Duran Alejandro C.I: 25395510 |
|  |  | Zorrilla Paola C.I:24848197 |

Ciudad Guayana. Abril, 2018



Codigo

#include "config.h"

#include "ui.h"

#include "extras.h"

#include "sprites.h"

unsigned short unidad\_segundo=0, decena\_segundo=0, unidad\_minuto=0,decena\_minuto=0, unidad\_hora=0, decena\_hora=0,decena\_milisegundo=0,unidad\_milisegundo=0;

char hora[12]={'0','0',':','0','0',':','0','0',':','0','0','\0'};

char alarma[6]={'0','0',':','0','0','\0'};

char texta=1+'0';

int ENALARM=1,conta1=0,selected=0;

float T1,T2,T3,T4;

int pulso=0, pulso2=0,pulso3=0,pulso4=0;

float frecuencia=0,frecuencia2=0,frecuencia3=0,frecuencia4=0;

void cron\_write(); void cron\_inttostr(); void caso\_1();

//~~~~~~~~~~Interrupciones~~

void cron(){

unidad\_milisegundo=unidad\_milisegundo+5;

if(unidad\_milisegundo>9){

unidad\_milisegundo=0;

decena\_milisegundo++;

if(decena\_milisegundo==10){

decena\_milisegundo=0;

unidad\_segundo++;

if(unidad\_segundo==10){

unidad\_segundo=0;

decena\_segundo++;

if(decena\_segundo==6){

decena\_segundo=0;

unidad\_minuto++;

if(unidad\_minuto==10){

unidad\_minuto=0;

decena\_minuto++;

if(decena\_minuto==6){

decena\_minuto=0;

unidad\_hora++;

if(unidad\_hora==10 && decena\_hora==0){

unidad\_hora=0;

decena\_hora++;

}

else if(unidad\_hora==1 && decena\_hora==3){

unidad\_hora=0;

decena\_hora=0;

}

}

}

}

}

}

}

cron\_inttostr();

if(ENALARM==0 &&

alarma[0]==HORA[0] &&

alarma[1]==HORA[1] &&

alarma[3]==HORA[3] &&

alarma[4]==HORA[4]){

Glcd\_fill(0);

ENALARM=2;

IFS0Bits.T1IF=0;

}

Glcd\_Write\_Text(hora, 30, 7, 1);

IFS0bits.T1IF=0

}

void tee() org 0x1A{

cron();

}

void captura\_onda\_ic1() org 0x16{

pulso++;

IFS0bits.IC1IF=0;

}

void captura\_onda\_ic3() org 0x5E{

pulso2++;

IFS2bits.IC3IF=0;

}

void captura\_onda\_ic5() org 0x62{

pulso3++;

IFS2bits.IC5IF=0;

}

void captura\_onda\_ic11() org 0x112{

pulso4++;

IFS7bits.IC11IF=0;

}

void int\_timer2 () org 0x22{

IC1CON1bits.ICM=0;

IC3CON1bits.ICM=0;

IC5CON1bits.ICM=0;

IC11CON1bits.ICM=0;

T2CONbits.TON=0;

IFS0bits.T2IF=0; // limpia bandera de interrupcion

frecuencia=pulso\*4\*2;

frecuencia2=pulso2\*4\*2;

frecuencia3=pulso3\*4\*2;

frecuencia4=pulso4\*4\*2;

T1=(1/frecuencia)\*1000000;

T2=(1/frecuencia2)\*1000000;

T3=(1/frecuencia3)\*1000000;

T4=(1/frecuencia4)\*1000000;

Glcd\_Write\_Text("Entrada", 0, 1, 1);

Glcd\_Write\_Text("2)", 5, 3, 1);

Glcd\_Write\_Text("3)", 5, 4, 1);

Glcd\_Write\_Text("4)", 5, 5, 1);

Glcd\_Write\_Text("Periodo (us)", 50, 1, 1);

Glcd\_Write\_Text("1)", 5, 2, 1);

floattostr(T1,txt);

Glcd\_Write\_Text(txt,65, 2, 1);

floattostr(T2,txt);

Glcd\_Write\_Text(txt,65, 3, 1);

floattostr(T3,txt);

Glcd\_Write\_Text(txt,65, 4, 1);

floattostr(T4,txt);

Glcd\_Write\_Text(txt,65, 5, 1);

IFS0bits.IC2IF=0;

}

// IFS0bits.IC2IF=0;

//}

void Timer4() org 0x4A{

IFS1bits.T4IF=0;

}

void Timer5() org 0x4C{

T4CONbits.TON=1

IFS1bits.T5IF=0;

}

//~~~~~~~~~~~~~~~~Declaraciones de Funciones~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

void cron\_write();

void clean\_PS2();

void cron\_inttostr(){

hora[0]=decena\_hora+'0';

hora[1]=unidad\_hora+'0';

hora[3]=decena\_minuto+'0';

hora[4]=unidad\_minuto+'0';

hora[6]=decena\_segundo+'0';

hora[7]=unidad\_segundo+'0';

hora[9]=decena\_milisegundo+'0';

hora[10]=unidad\_milisegundo+'0';

hora[11]= '\0';

}

int i=0;

void cron\_write(){

// Glcd\_Write\_TEXT(" ",60,7,1);

i=0;

for(i=0;i<5;i++){

if(i!=2||i!=5||i!=8){

Glcd\_Write\_Char(HORA[i],50+i\*5,7,1);

}

}

}

void num\_update(int it,int x\_pos,int page){

Glcd\_Write\_Char(it+'0',x\_pos,page,1);

}

int it=0;

int arreglo\_hora\_militar(int indice,int valor\_nuevo){

if(indice==1){

//DECENA\_HORA

if(valor\_nuevo>2){return 0;}//Decena de hora solo puede ser 0,1 o 2 2->0

else if(valor\_nuevo<0){return 2;} //Decena de hora solo puede ser 0,1 o 2 2->0;

}

else if(indice==2){

//UNIDAD\_HORA

if(valor\_nuevo>9){return 0;}//Arreglo Unidad Hora && Unidad Minuto 9->0

else if(valor\_nuevo<0){return 9;}//Arreglo Unidad Hora && Unidad Minuto 0->9

else if(HORA[0]=='2'&&valor\_nuevo>3){//Arreglo Hora 23->00

HORA[0]='0';num\_update(0,50,7);return 0;}

else if(HORA[0]=='2'&&valor\_nuevo<0){//Arreglo Hora 00->23

HORA[0]='2';num\_update(2,50,7);return 3;}

}

else if(indice==3){

//DECENA\_MIN

if(valor\_nuevo>5){ return 0;}//Decena minuto llega hasta 5->0

if(valor\_nuevo<0){return 5;}//Decena minuto llega hasta 5

}

else if(indice==4){

//UNIDAD\_MIN

if(valor\_nuevo>9){return 0;}//Arreglo Unidad Hora && Unidad Minuto 9->0

if(valor\_nuevo<0){return 9;}//Arreglo Unidad Hora && Unidad Minuto 9->0

}

else {

return valor\_nuevo;}

}

int num\_selector(int x\_pos,int is){

it=0;

num\_update(it,x\_pos,7);

clean\_PS2();

while(keydata!=ENTER){

if(Ps2\_Key\_Read(&keydata, &special, &down)){

if(down){

if(keydata==DOWN\_ARROW||keydata==UP\_ARROW){

if(keydata==UP\_ARROW){

inttostr(it,txt) ;

glcd\_write\_text(txt,0,0,1);

inttostr(is,txt) ;

glcd\_write\_text(txt,60,0,1);

it=arreglo\_hora\_militar(is,it+1);

clean\_PS2();

}

if(keydata==DOWN\_ARROW){

it=it-1;

inttostr(it,txt) ;

glcd\_write\_text(txt,0,0,1);

inttostr(is,txt) ;

glcd\_write\_text(txt,60,0,1);

it=arreglo\_hora\_militar(is,it-1);

clean\_PS2();

}

num\_update(it,x\_pos,7);

clean\_PS2();

}

clean\_PS2();

}

}

}

return it;

}

void cron\_cursor(){

//hay 5 variables de hora y alarma

num\_selector(0,0);

glcd\_write\_text("Hora:",0,7,1);

decena\_hora=num\_selector(50,0);

unidad\_hora=num\_selector(56,1);

glcd\_write\_char(':',62,7,1);

decena\_minuto=num\_selector(68,3);

unidad\_minuto=num\_selector(74,4);

decena\_segundo=0;

unidad\_segundo=0;

decena\_milisegundo=0;

unidad\_milisegundo=0;

cron\_inttostr();

}

void cron\_alarm(){

num\_selector(0,0);

glcd\_write\_text("Alarma:",0,7,1);

alarma[0]=num\_selector(50,0)+'0';

alarma[1]=num\_selector(56,0)+'0';

glcd\_write\_char(':',62,7,1);

alarma[3]=num\_selector(68,0)+'0';

alarma[4]=num\_selector(74,0)+'0';

ENALARM=0;

}

int set=0;

void caso\_1(){

clean\_PS2();

set=0;

while(set!=ESC){

set=cursor\_menu2(5);

switch(set){

case 1:

cron\_cursor();

clean\_PS2();

Glcd\_write\_text(" ",0,7,1);

break;

case 2:

T1CONbits.TON=1;

Glcd\_write\_text(" ",0,7,1);

break;

case 3:

T1CONBits.TON=0;

Glcd\_write\_text(" ",0,7,1);

break;

case 4:

decena\_hora=0; unidad\_hora=0;

decena\_minuto=0; unidad\_minuto=0;

decena\_segundo=0; unidad\_segundo=0;

decena\_milisegundo=0; unidad\_milisegundo=0;

cron\_inttostr();

Glcd\_Write\_Text(hora, 30, 7, 1);

Glcd\_write\_text(" ",0,7,1);

break;

case 5:

ENALARM=0;

cron\_alarm();

Glcd\_write\_text(" ",0,7,1);

break;

default:

Glcd\_write\_text(" ",0,7,1);

break;

}

}

}

int glcd\_write\_float(float f,unsigned short pos,unsigned short page,unsigned short color){

if(floattostr(f,txt)==0){

Glcd\_Write\_Text(txt,pos,page,color);

return 0;} else {

return 1;

}

}

void frecuencia\_pantalla (){

Glcd\_Write\_Text("Frecu.(hz)", 0, 1, 1);

Glcd\_Write\_Text("Periodo(ms)", 65, 1, 1);

inttostr(pulso,txt);

glcd\_write\_text(txt,5,2,1);

inttostr(pulso2,txt);

glcd\_write\_text(txt,5,3,1);

inttostr(pulso3,txt);

glcd\_write\_text(txt,5,4,1);

inttostr(pulso4,txt);

glcd\_write\_text(txt,5,5,1);

}

void caso\_2(){

pulso=0;

pulso2=0;

pulso3=0;

pulso4=0;

Glcd\_Write\_Text("Caso 2",65, 0, 1);

config\_pin();

config\_captura();

T2CONbits.TON=1; //enciende timer 2

Glcd\_Fill(0);

clean\_PS2();

while(keydata!=ESC){

Ps2\_Key\_Read(&keydata, &special, &down);

}

}

void caso\_3();

void main(){

config\_IO(); config\_LCD();

config\_INT();

config\_cron();

// config\_OC();

// config\_TMR\_45();

config\_pin();

PS2\_Config(); Glcd\_Fill(0);

// glcd\_write\_text("a",1,1,1);

//T1conbits.Ton=1;

while(1){

texto\_menu();

selected=cursor\_menu(3);

switch(selected){

case 1:

clean\_PS2();

texto\_caso\_1();

caso\_1();

break;

case 2:

clean\_PS2();

// texto\_caso\_2();

caso\_2();

break;

case 3:

clean\_PS2();

texto\_caso\_3();

caso\_3();

break;

}

}

}

void caso\_3(){

texto\_caso\_3();

ANSELE=0;

ANSELG=0;

ANSELD=0;

TRISEbits.TRISE5=0;

TRISEbits.TRISE7=0;

TRISGbits.TRISG6=0;

TRISGbits.TRISG8=0;

TRISGbits.TRISG8=0;

RPOR6bits.RP85R =0b010000; //pin comparador 1 RE5

RPOR6bits.RP87R =0b010001; //pin comparador 2 RE7

RPOR13bits.RP118R=0b010010; //pin comparador 3 RG6

RPOR14bits.RP120R=0b010011;

RPOR0bits.RP64R=0b010100;// pin comparador 4 RG8

IEC1bits.T4IE=1;

IEC1bits.T5IE=1;

IFS1bits.T4IF=0;

IFS1bits.T5IF=0;

T4CON=0; //preescaler 1:1

TMR4=0;

PR4=2000; // timer a 1ms

T5CON=0x8030; //prescaler 256:1

TMR5=0;

PR5=39062; // timer a 5s

OC5CON1bits.OCTSEL=3;

OC1CON1bits.OCTSEL=2; // T4CLK fuente de reloj

OC2CON1bits.OCTSEL=2;

OC3CON1bits.OCTSEL=2;

OC4CON1bits.OCTSEL=2;

OC5CON1bits.OCM=5;

OC1CON1bits.OCM=5; //MODO COMPARADOR DUAL DE PULSOS CONTINUOS

OC2CON1bits.OCM=5;

OC3CON1bits.OCM=5;

OC4CON1bits.OCM=5;

OC1CON2bits.OCTRIG=1;

OC2CON2bits.OCTRIG=1;

OC3CON2bits.OCTRIG=1;

OC4CON2bits.OCTRIG=1;

OC5CON2bits.OCTRIG=0;

OC1CON1bits.TRIGMODE=0;

OC2CON1bits.TRIGMODE=0;

OC3CON1bits.TRIGMODE=0;

OC4CON1bits.TRIGMODE=0;

OC5R=8000; OC5RS=8200;

OC1R=306; OC1RS=2047; //CICLO UTIL 15%

OC2R=921; OC2RS=2047; //CICLO UTIL 45%

OC3R=1227; OC3RS=2047; //CICLO UTIL 60%

OC4R=1637; OC4RS=2047; //CICLO UTIL 80%

OC1CON2bits.SYNCSEL=0b00101; //disparo con el OC 5

OC2CON2bits.SYNCSEL=0b00101;

OC3CON2bits.SYNCSEL=0b00101;

OC4CON2bits.SYNCSEL=0b00101;

OC5CON2bits.SYNCSEL=0b01111;

while(keydata!=ESC){

Ps2\_Key\_Read(&keydata, &special, &down);

}

T5CONbits.TON=0;

T4CONbits.TON=0;

}

Config.h

//\_\_\_\_\_//~~~~~~~~~~~~~~~~~~Constantes del dsPIC~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

//Variables de PS2

sbit PS2\_Data at RF4\_bit;

sbit PS2\_Clock at RF5\_bit;

sbit PS2\_Data\_Direction at TRISF4\_bit;

sbit PS2\_Clock\_Direction at TRISF5\_bit;

//Variables de GLCD

sbit GLCD\_D7 at RE1\_bit;

sbit GLCD\_D6 at RE0\_bit;

sbit GLCD\_D5 at RF1\_bit;

sbit GLCD\_D4 at RF0\_bit;

sbit GLCD\_D3 at RD7\_bit;

sbit GLCD\_D2 at RD6\_bit;

sbit GLCD\_D1 at RD5\_bit;

sbit GLCD\_D0 at RD4\_bit;

sbit GLCD\_D7\_Direction at TRISE1\_bit;

sbit GLCD\_D6\_Direction at TRISE0\_bit;

sbit GLCD\_D5\_Direction at TRISF1\_bit;

sbit GLCD\_D4\_Direction at TRISF0\_bit;

sbit GLCD\_D3\_Direction at TRISD7\_bit;

sbit GLCD\_D2\_Direction at TRISD6\_bit;

sbit GLCD\_D1\_Direction at TRISD5\_bit;

sbit GLCD\_D0\_Direction at TRISD4\_bit;

sbit GLCD\_CS1 at LATE3\_bit;

sbit GLCD\_CS2 at LATE2\_bit;

sbit GLCD\_RS at LATD1\_bit;

sbit GLCD\_RW at LATD2\_bit;

sbit GLCD\_EN at LATD3\_bit;

sbit GLCD\_RST at LATE4\_bit;

sbit GLCD\_CS1\_Direction at TRISE3\_bit;

sbit GLCD\_CS2\_Direction at TRISE2\_bit;

sbit GLCD\_RS\_Direction at TRISD1\_bit;

sbit GLCD\_RW\_Direction at TRISD2\_bit;

sbit GLCD\_EN\_Direction at TRISD3\_bit;

sbit GLCD\_RST\_Direction at TRISE4\_bit;

//~~~~~~~~~~~~~~~~~~~~~~Configuraciones Iniciales~~~~~~~~~~~~~~~~~~~~~~~~~~

void config\_IO(){

ANSELB=0;

ANSELC=0;

ANSELD=0;

ANSELE=0;

ANSELG=0; //ANA//logiCO SON B Y F

TRISB=0xffff;

TRISE=0;

TRISG=0;

TRISDbits.TRISD11=1; // desde rd11 a rd8 son entradas del modulo captura

TRISDbits.TRISD10=1;

TRISDbits.TRISD9=1;

TRISDbits.TRISD8=1;

TRISFbits.TRISF3=0; // salida paola

TRISEbits.TRISE5=0;

TRISEbits.TRISE7=0;

TRISGbits.TRISG6=0;

TRISGbits.TRISG8=0;

RPOR6bits.RP85R =0b010000; //pin comparador 1 RE5

RPOR6bits.RP87R =0b010001; //pin comparador 2 RE7

RPOR13bits.RP118R=0b010010; //pin comparador 3 RG6

RPOR14bits.RP120R=0b010011;

RPOR0bits.RP64R=0b010100;// pin comparador 4 RG8

}

void config\_LCD(){

Glcd\_Init();

Glcd\_set\_Font(font5x7 , 5, 7, 32);

Glcd\_Fill(0);

}

void config\_pin () {

TRISDbits.TRISD11=1; // pin D11 como entrada

RPINR7bits.IC1R=75; //captura 1 por el D11

TRISDbits.TRISD10=1;// pin D10 como entrada

RPINR8bits.IC3R=74; // captura 3 por el d10

TRISDbits.TRISD9=1; // pin d09 como entrada

RPINR9bits.IC5R=73; // captura 5 por el d09

TRISDbits.TRISD8=1; // pin D08 como entrada

RPINR34bits.IC11R=72; //captura 7 por el d08

}

void config\_captura (){

Glcd\_Write\_Text("Captura",65, 4, 1);

delay\_ms(250);

//-------------timer2-----------------------

TMR2=0;

T2CONbits.TCKPS=2; // Prescaler 64:1, modo timer

T2CONbits.TGATE=0;

T2CONbits.TCS=0;

PR2=31250; //1000ms

IEC0bits.T2IE=1;//enable interrupcion

IFS0bits.T2IF=0;//limpia bandera

//--------------captura 2----------------------

IC2CON1bits.ICTSEL=7; //usa como fuente de reloj el timer1

IC2CON1bits.ICM=5; // captura cada 4 flancos positivos

//---------------configuracion captura 2 como 32 bits-------------

IC2CON2bits.IC32=1; //configuracion 32 bits

//----------------captura 1------------------

IC1CON1bits.ICTSEL=7; //usa como fuente de reloj el timer1

// T1CONbits.TON=1; //enciende timer 1

IEC0bits.IC1IE=1; //habilita interrupciÃ³n del IC1

IFS0bits.IC1IF=0; //limpia la bandera de interrupciÃ³n

//------------configuracion captura 1 como 32 bits------------

IC1CON2bits.IC32=1; //configuracion 32 bits;

//--------------captura 4----------------------

IC4CON1bits.ICTSEL=7; //usa como fuente de reloj el timer1

IC4CON1bits.ICM=5; // captura cada 4 flancos positivos

//---------------configuracion captura 4 como 32 bits-------------

IC4CON2bits.IC32=1; //configuracion 32 bits

//----------------captura 3------------------

IC3CON1bits.ICTSEL=7; //usa como fuente de reloj el timer1

// T1CONbits.TON=1; //enciende timer 1

IEC2bits.IC3IE=1; //habilita interrupciÃ³n del IC1

IFS2bits.IC3IF=0; //limpia la bandera de interrupciÃ³n

//------------configuracion captura 3 como 32 bits------------

IC3CON2bits.IC32=1; //configuracion 32 bits;

//--------------captura 6----------------------

IC6CON1bits.ICTSEL=7; //usa como fuente de reloj el timer1

IC6CON1bits.ICM=5; // captura cada 4 flancos positivos

//---------------configuracion captura 6 como 32 bits-------------

IC6CON2bits.IC32=1; //configuracion 32 bits

//----------------captura 5------------------

IC5CON1bits.ICTSEL=7; //usa como fuente de reloj el timer1

// T1CONbits.TON=1; //enciende timer 1

IEC2bits.IC5IE=1; //habilita interrupciÃ³n del IC1

IFS2bits.IC5IF=0; //limpia la bandera de interrupciÃ³n

//------------configuracion captura 1 como 32 bits------------

IC5CON2bits.IC32=1; //configuracion 32 bits

//--------------captura 12----------------------

IC12CON1bits.ICTSEL=7; //usa como fuente de reloj el timer1

IC12CON1bits.ICM=5; // captura cada 4 flancos positivos

//---------------configuracion captura 12 como 32 bits-------------

IC12CON2bits.IC32=1; //configuracion 32 bits

//----------------captura 11------------------

IC11CON1bits.ICTSEL=7; //usa como fuente de reloj el timer

// T1CONbits.TON=1; //enciende timer 1

IEC7bits.IC11IE=1; //habilita interrupciÃ³n del IC1

IFS7bits.IC11IF=0; //limpia la bandera de interrupciÃ³n

//------------configuracion captura 1 como 32 bits------------

IC5CON2bits.IC32=1; //configuracion 32 bits;

//-------------activacion captura--------------------

IC1CON1bits.ICM=5; // captura cada 4 flancos positivos

IC3CON1bits.ICM=5; // captura cada 4 flancos positivos

IC5CON1bits.ICM=5; // captura cada 4 flancos positivos

IC11CON1bits.ICM=5; // captura cada 4 flancos positivos

}

void config\_cron(){

//PR1=22938; //1000ms

// TMR1=0;

IEC0bits.T1IE=1;

IFS0bits.T1IF=0;

IPC0bits.T1IP=3;

// T1CONBits.TGATE=0;

// T1CONBits.TSYNC=0;

// T1CONBits.TCS=0;//TGATE, TSYNC Y TCS Habilitan modo TEMP

// T1CONBits.TCKPS=1;

// T1conbits.TON=0;

TMR1=0;

T1CON=0X0010; //Prescaler 16:1, modo timer

PR1=6250;

}

/\*void config\_TMR\_45(){

IEC1bits.T4IE=1;

IEC1bits.T5IE=1;

IFS1bits.T4IF=0;

IFS1bits.T5IF=0;

T4CONbits.TCKPS=0; //preescaler 1:1

T4CONbits.TCS=0;

TMR4=0;

PR4=2000; // timer a 1ms

T5CONbits.TCKPS=5; //prescaler 256:1

T5CONbits.TCS=0;

TMR5=0;

PR5=39062; // timer a 5s

}\*/

void config\_OC(){

OC5CON1bits.OCTSEL=7;

OC1CON1bits.OCTSEL=2; // T4CLK fuente de reloj

OC2CON1bits.OCTSEL=2;

OC3CON1bits.OCTSEL=2;

OC4CON1bits.OCTSEL=2;

OC1CON2bits.OCTRIS=1;

OC2CON2bits.OCTRIS=1;

OC3CON2bits.OCTRIS=1;

OC4CON2bits.OCTRIS=1;

OC5CON2bits.OCTRIS=1;

OC5CON1bits.OCM=5;

OC1CON1bits.OCM=5; //MODO COMPARADOR DUAL DE PULSOS CONTINUOS

OC2CON1bits.OCM=5;

OC3CON1bits.OCM=5;

OC4CON1bits.OCM=5;

OC5R=306; OC5RS=2047;

OC1R=306; OC1RS=2047; //CICLO UTIL 15%

OC2R=921; OC2RS=2047; //CICLO UTIL 45%

OC3R=1227; OC3RS=2047; //CICLO UTIL 60%

OC4R=1637; OC4RS=2047; //CICLO UTIL 80%

OC1CON2bits.SYNCSEL=0b01111; //disparo con el OC 5

OC2CON2bits.SYNCSEL=0b00101;

OC3CON2bits.SYNCSEL=0b01111; OC4CON2bits.SYNCSEL=0b01111;

OC5CON2bits.SYNCSEL=0b01111;

}

void config\_INT(){

SRbits.IPL =0;// iNTERRUPCION DE CPU ES DE NIVEL 0

INTCON1bits.NSTDIS =0;// INTERRUPCION ANIDADAS ACTIVADAS

INTCON2bits.GIE=1; //interrupciones habilitadas

CORCONbits.IPL3 = 0; // El nivel del cpu es de nivel 0, las interrupciones por perifericos habilitadas

//------------------------- habilitacion de interrupcion

IPC0bits.T1IP=2;

IPC1bits.T2IP=4;

IPC6bits.T4IP=5;

IPC7bits.T5IP=5;

IPC0bits.IC1IP=2;

IPC9bits.IC3IP=2;

IPC9bits.IC5IP=2;

IPC31bits.IC11IP=2;

// IPC1bits.IC2IP=2;

}

Extras.h

#define ENTER 13

#define ESC 118

#define LEFT\_ARROW 30

#define RIGHT\_ARROW 31

#define UP\_ARROW 111

#define DOWN\_ARROW 108

#define SANGRIA 6

unsigned short keydata = 0, special = 0, down = 0,y;

char txt[15];

char t[3],c1;

void clean\_PS2(){ keydata = 0, special = 0, down = 0;}

void cursor\_animation(int it){

int i;

for(i=1; i < 6; i++ ){

if(i==it){

Glcd\_Write\_Text("\*",0,i,1);

}else{

Glcd\_Write\_Text(" ",0,i,1);

}

}

}

int cursor\_menu(int max\_n\_options){

//aqui se va a estar utilizando el it como iterator, revisando si el teclado

//ha seleccionado las flechas de arriba o abajo, aumentando o disminuyendo al operador

int it=1;

clean\_PS2();

while(keydata!=ENTER){

cursor\_animation(it);

if(Ps2\_Key\_Read(&keydata, &special, &down)){

if(down){

if(keydata==DOWN\_ARROW||keydata==UP\_ARROW){

if(keydata==DOWN\_ARROW){

it=it+1;

if(it>max\_n\_options){it=1;}

clean\_PS2();

}

if(keydata==UP\_ARROW){

it=it-1;

if(it==0){it=max\_n\_options;}

clean\_PS2();

}

cursor\_animation(it);

}

}

}

}

clean\_PS2();

return it;

}

int cursor\_menu2(int max\_n\_options){

int it=0;

clean\_PS2();

while(keydata!=ENTER){

cursor\_animation(it);

if(Ps2\_Key\_Read(&keydata, &special, &down)){

if(down){

if(keydata==DOWN\_ARROW||keydata==UP\_ARROW){

if(keydata==DOWN\_ARROW){

it=it+1;

if(it>max\_n\_options){it=0;}

clean\_PS2();

}

if(keydata==UP\_ARROW){

it=it-1;

if(it==-1){it=max\_n\_options;}

clean\_PS2();

}

cursor\_animation(it);

}

}

}

}

clean\_PS2();

return it;

}

//Cursor Numero

//Texto del laboratorio

void texto\_menu(){

glcd\_fill(0);

Glcd\_Write\_TEXT("Laboratorio 2",60,0,1);

Glcd\_Write\_TEXT("1) Cronometro",SANGRIA,1,1);

Glcd\_Write\_TEXT("2) Captura" ,SANGRIA,2,1);

Glcd\_Write\_TEXT("3) PWM " ,SANGRIA,3,1);

}

void texto\_caso\_1(){

GLCD\_FILL(0);

Glcd\_Write\_Text("Cronometro",60,0,1);

Glcd\_Write\_Text("2) PLAY" ,SANGRIA,2,1);

Glcd\_Write\_Text("1) Programar Hora",SANGRIA,1,1);

Glcd\_Write\_Text("3) PAUSE" ,SANGRIA,3,1);

Glcd\_Write\_Text("4) RESET " ,SANGRIA,4,1);

Glcd\_Write\_Text("5) Alarma",SANGRIA,5,1);

}

void texto\_caso\_2(){

GLCD\_FILL(0);

Glcd\_Write\_Text("Puerto de Entrada",60,0,1);

Glcd\_Write\_Text("RP75/RD11" ,0,1,1);

Glcd\_Write\_Text("RP74/RD10" ,0,2,1);

Glcd\_Write\_Text("RP73/RD09" ,0,3,1);

Glcd\_Write\_Text("RP72/RD8" ,0,4,1);

}

void texto\_caso\_3(){

GLCD\_FILL(0);

Glcd\_Write\_Text("Puerto de Entrada",60,0,1);

Glcd\_Write\_Text("RP101/RG8 80%" ,0,4,1);

Glcd\_Write\_Text("RP85 /RE5 15%" ,0,1,1);

Glcd\_Write\_Text("RP87 /RE7 45%" ,0,2,1);

Glcd\_Write\_Text("RP87 /RG6 60%" ,0,3,1);

}