UNIVERSIDAD NACIONAL EXPERIMENTAL POLITECNICA

“ANTONIO JOSE DE SUCRE”

VICE-RECTORADO DE PUERTO ORDAZ

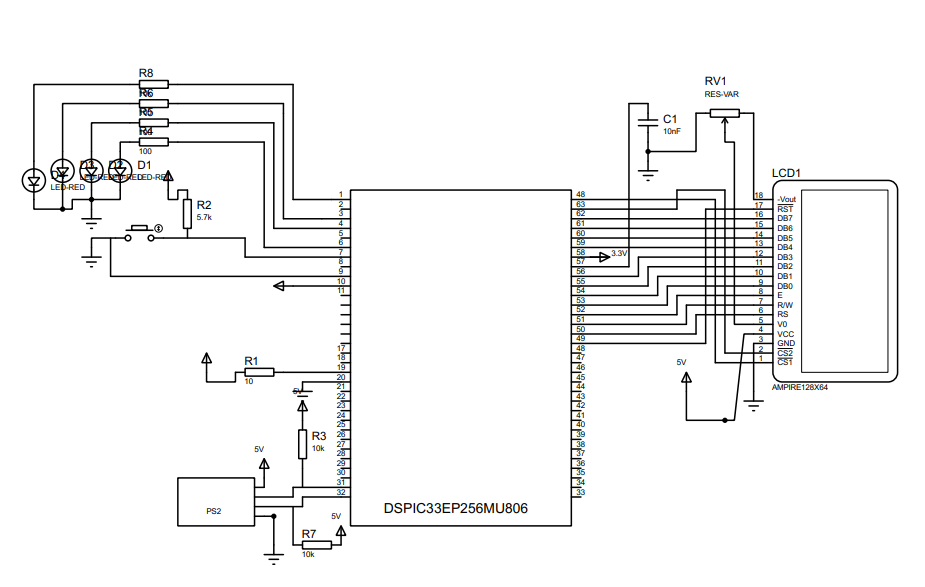
DEPARTAMENTO DE ELECTRONICA

**MICROPROCESADORES II**

Laboratorio 5

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

Ciudad Guayana. Junio, 2018



Codigo TX

#include "hid.h"

#include "adc.h"

#include "config\_TX.h"

#define ESC\_key 254

#define PARTE\_ALTA 0x3FC

#include "pwm\_reloj.h"

#include "config\_maestro.h"

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

void buffer\_caso1();void caso2();void caso3();void caso1();void SPI\_ESC();void caso5();void caso4();

unsigned leer\_24lc256(unsigned char adres);

void escribir\_24lc256(unsigned char valor,unsigned char adres);

void programar\_Alarma();void programar\_RTCC();void caso2\_check();

//~~~~~~~~~~~~~~~~~~~~~~~~Variables del sistema~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

int i=0,j=0,escape=0,cnt,mot1=0,mot2=0,bandera1=0,bandera2=0,bandera4=0,bandera5=0,rpm1=0,rpm2=0;

unsigned char henhol=0; // varuable de posicion en la memoria

unsigned valor=0,adres=0;

unsigned potval[5]={0,0,0,0,0};

float pantalla=0,voltaje=0;

char readbuff[64];

char writebuff[64];

unsigned short enviado=0;

unsigned caso1\_val[6]={0,0,0,0,0,0},adc\_value1=0,adc\_value2=0,adc\_value=0;

unsigned short pote1=0,pote2=0,pote3=0;

char txt [30];

int value;

char txt4[5];char txt3[4];

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

void timer1 () org 0x1A {IFS0bits.T1IF=0;

if (bandera2==1)

write("Voltaje en limite superior");

else if (bandera2==2)

write("Voltaje en limite inferior");

else if (bandera2==3)

write("Voltaje normal");

adc\_value = ADC2\_Get\_Sample(14);

pantalla=adc\_value\*0.0034;

sprintf(txt,"Voltaje=%.3f",pantalla);

write(txt);

space2();

space2();

space2();

space2();

enviado=bandera2;

SPI1\_Write(enviado);

}

void USB1Interrupt() iv IVT\_ADDR\_USB1INTERRUPT{USB\_Interrupt\_Proc();}

void INT\_Inicio\_Conversion\_T3 () org 0x24 {IFS0bits.T3IF=0;

adc\_value1 = ADC1\_Get\_Sample(0);

delay\_ms(10);

adc\_value2 = ADC1\_Get\_Sample(3);

pote1=adc\_value1>>2;

pote2=adc\_value2>>2;

delay\_ms(10);

enviado=pote2;

SPI1\_Write(enviado); delay\_ms(10);

enviado=pote1;

SPI1\_Write(enviado); delay\_ms(10);

bandera1++;

}//inicializa conversion del adc

void INT\_ADC() org 0x2E {

IFS0bits.AD1IF=0;

adc\_value1=ADC1BUF0;

adc\_value2=ADC1BUF1;

pote1=adc\_value1>>2;

pote2=adc\_value2>>2;

delay\_ms(5);

enviado=pote2;

SPI1\_Write(enviado); delay\_ms(10);

enviado=pote1;

SPI1\_Write(enviado); delay\_ms(10);

bandera1++;

}

void interupcion\_ext1() org 0x3C {IFS1bits.INT1IF=0;rpm1++;}

void interupcion\_ext2() org 0x4E {IFS1bits.INT2IF=0;rpm2++;}

void timer7 () org 0x74 {

IFS3bits.T7IF=0;

caso1\_val[0]=rpm1\*75;

caso1\_val[1]=rpm2\*75;

rpm1=0;

rpm2=0;

adc\_value = ADC2\_Get\_Sample(14);

logd(adc\_value);

}

void Error\_SPI() org 0x26{

IFS0bits.SPI1EIF=0;

write("Error");

delay\_ms(3000);

}

void SPI() org 0x28{

IFS0bits.SPI1IF=0;

write("ALGUN DATO SE TRANSMITIO");

delay\_ms(3000);

}

//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~MENU~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

void main() {

config\_pin();

config\_INT();

I2C1\_Init(100000); //inicializa i2c a 100Khz

config\_timer1();

config\_Pwm\_reloj();

InitMCU();

config\_timer3();

// config\_adc();

ADC1\_Init\_Advanced(\_ADC\_10bit, \_ADC\_INTERNAL\_REF);

ADC2\_Init\_Advanced(\_ADC\_10bit, \_ADC\_INTERNAL\_REF);

HID\_Enable(&readbuff,&writebuff); //inicializamos en módulo usb hid

SPI1\_Init\_Advanced(\_SPI\_MASTER,\_SPI\_8\_BIT,\_SPI\_PRESCALE\_SEC\_8,

\_SPI\_PRESCALE\_PRI\_64,\_SPI\_SS\_DISABLE,\_SPI\_DATA\_SAMPLE\_MIDDLE,

\_SPI\_CLK\_IDLE\_LOW,\_SPI\_ACTIVE\_2\_IDLE);

RPOR0bits.RP64R=5; //SDO1

RPOR8bits.RP99R=6; //SCK1

RPINR20bits.SDI1R=72; //SDI1

while(1){

menu2();

while(!HID\_Read());

for(cnt=0;cnt<64;cnt++) {writebuff[cnt]=readbuff[cnt];}

if(strcmp(readbuff,caso\_1)==0){//CASE 1

write("Bienvenido al Caso 1");

enviado=1;

SPI1\_Write(enviado);

caso1();

}

else if(strcmp(readbuff,caso\_2)==0){//CASE 2

write("Bienvenido al Caso 2");

enviado=2;

SPI1\_Write(enviado);

caso2();

T1CONbits.TON=0;

}

else if(strcmp(readbuff,caso\_3)==0){//CASE 3

write("Bienvenido al Caso 3");

enviado=3;

SPI1\_Write(enviado);

caso3();

}

else if(strcmp(readbuff,caso\_4)==0){//CASE D

enviado=4;

SPI1\_Write(enviado);

bandera4=0;

caso4();

}

else if(strcmp(readbuff,caso\_5)==0){//CASE D

enviado=5;

SPI1\_Write(enviado);

bandera5=0;

caso5();

}

else if(strcmp(readbuff,"P")==0){//CASE D

unsigned char as;

as=leer\_24lc256(1);

sprintf(txt,"se leyo %u",as);

write(txt);

escribir\_24lc256(42,1);

as=0;

as=leer\_24lc256(1);

sprintf(txt,"se leyo %u",as);

write(txt);

}

Delay\_ms(1000);

escape=0;

}

}

void caso1(){

bandera1=0;

// AD1CON1bits.ADON=1;// Se act el modulo

T3CONbits.TON=1; // activa timer 3 para inicio de conver.

//hid\_caso\_1(caso1\_val[0],caso1\_val[1],caso1\_val[2],caso1\_val[3],caso1\_val[4],caso1\_val[5],pote1,pote2);

delay\_ms(150);

// config\_velocidad ();

while(escape==0){

if (bandera1==3) {

buffer\_caso1();

hid\_caso\_1(caso1\_val[0],caso1\_val[1],

caso1\_val[2],caso1\_val[3],

caso1\_val[4],caso1\_val[5],

pote1,pote2);

bandera1=0;

}

if(HID\_Read()){

escape++;SPI\_ESC();}

}

AD1CON1bits.ADON=0;// Se act el modulo

T3CONbits.TON=0;

T7CONbits.TON=0;

IPC5BITS.INT1IP=0;

IPC7bits.INT2IP=0;

}

void buffer\_caso1(){

if(pote1<128){caso1\_val[2]=IZQ;}

else {caso1\_val[2]=DER;}

if(pote2<128){caso1\_val[3]=IZQ;}

else {caso1\_val[3]=DER;}

caso1\_val[4]=FALLA;

caso1\_val[5]=FALLA;

}

void caso2(){

T1CONbits.TON=1;

config\_CM();

delay\_ms(100);

escape=0;

while(escape==0){

if(HID\_Read()){

escape++;SPI\_ESC();}

else{

caso2\_check();

}

}

T1CONbits.TON=0;

}

void caso3(){

hid\_caso\_3();

delay\_ms(100);

while(escape==0){

while(!HID\_Read());

for(cnt=0;cnt<64;cnt++) {writebuff[cnt]=readbuff[cnt]; }

switch(readbuff[0]){

case '0':enviado=0; caso3\_1[28]=readbuff[0]; hid\_caso\_3(); SPI1\_Write(enviado);break;

case '1':enviado=1; caso3\_1[28]=readbuff[0]; hid\_caso\_3(); SPI1\_Write(enviado);break;

case '2':enviado=2; caso3\_1[28]=readbuff[0]; hid\_caso\_3(); SPI1\_Write(enviado);break;

case '3':enviado=3; caso3\_1[28]=readbuff[0]; hid\_caso\_3(); SPI1\_Write(enviado);break;

case '4':enviado=4; caso3\_1[28]=readbuff[0]; hid\_caso\_3(); SPI1\_Write(enviado);break;

case '5':enviado=5; caso3\_1[28]=readbuff[0]; hid\_caso\_3(); SPI1\_Write(enviado);break;

case '6':enviado=6; caso3\_1[28]=readbuff[0]; hid\_caso\_3(); SPI1\_Write(enviado);break;

case '7':enviado=7; caso3\_1[28]=readbuff[0]; hid\_caso\_3(); SPI1\_Write(enviado);break;

case '8':enviado=8; caso3\_1[28]=readbuff[0]; hid\_caso\_3(); SPI1\_Write(enviado);break;

case '9':enviado=9; caso3\_1[28]=readbuff[0]; hid\_caso\_3(); SPI1\_Write(enviado);break;

case 'A':case 'a':enviado=10; caso3\_1[28]='A'; hid\_caso\_3(); SPI1\_Write(enviado);break;

case 'B':case 'b':enviado=11; caso3\_1[28]='B'; hid\_caso\_3(); SPI1\_Write(enviado);break;

case 'C':case 'c':enviado=12; caso3\_1[28]='C'; hid\_caso\_3(); SPI1\_Write(enviado);break;

case 'D':case 'd':enviado=13; caso3\_1[28]='D'; hid\_caso\_3(); SPI1\_Write(enviado);break;

case 'E':case 'e':enviado=14; caso3\_1[28]='E'; hid\_caso\_3(); SPI1\_Write(enviado);break;

case 'F':case 'f':enviado=15; caso3\_1[28]='F'; hid\_caso\_3(); SPI1\_Write(enviado);break;

default:escape++;SPI\_ESC();break;

}

}

}

void SPI\_ESC(){

delay\_ms(10);

enviado=ESC\_key;

SPI1\_Write(enviado);

}

void caso2\_check(){

adc\_value = ADC2\_Get\_Sample(14);

if(CM3CONbits.COUT==0){

bandera2=1;

}

else if(CM1CONbits.COUT==0){

bandera2=2;

}

else if(CM1CONbits.COUT==1&&CM3CONbits.COUT==1) {

bandera2=3;

}

}

void escribir\_24lc256(unsigned char valor,unsigned char adres){

I2C1\_Start(); // emitir señal de inicio I2C

I2C1\_Write(0xA0); // enviar byte a través de (dirección del dispositivo + W)

I2C1\_Write(123); // direccion sin usar

I2C1\_Write(adres); // enviar byte (dirección de ubicación EEPROM)

I2C1\_Write(valor); // enviar datos (los datos se escriban)

I2C1\_Stop();

Delay\_ms(100);

}

unsigned leer\_24lc256(unsigned char adres){

unsigned char valor;

I2C1\_Start(); // emitir señal de inicio I2C

I2C1\_Write(0xA0); // enviar byte a través de (dirección del dispositivo + W)

I2C1\_Write(123); // enviar byte (dirección de datos)

I2C1\_Write(adres); // enviar byte (dirección de datos)

I2C1\_Restart(); // emitir señal repetida para iniciar I2C

I2C1\_Write(0xA1); // enviar byte (dirección del dispositivo + R)

valor = I2C1\_Read(1); // Leer los datos

I2C1\_Stop();

Delay\_ms(50);

return valor;

}

void guardado\_int1( unsigned adc\_valor, unsigned char adre){

unsigned char tmp;

if(adre<10){

inttostr(adre,txt);

// write(txt);

inttostr(adc\_valor,txt);

//write("adc\_valor ");

//write(txt);

tmp=(adc\_valor & 0x0300)>>8;

escribir\_24lc256(tmp,adre); //guardando parte alta en posicion 0

tmp=leer\_24lc256( adre);

inttostr(tmp,txt);

//write("adc\_valor>>8 ");

//write(txt);

delay\_ms(50) ;

adre++;

tmp=(adc\_valor & 0x00FF);

escribir\_24lc256(tmp,adre); //guardando parte alta en posicion 0

tmp=leer\_24lc256( adre);

//write("adc\_valor&0xFF ");

inttostr(tmp,txt);

//write(txt);

}else{adre=0;}

}

void valores (){

//Paso los valores de la memoria en 16 bits al array potval[5]

unsigned char parte\_baja,parte\_alta,k;

adres=0;

while(adres<10){

parte\_alta=leer\_24lc256(adres);

adres++;

parte\_baja=leer\_24lc256(adres);

adres++;

//sprintf(txt\_caso4,"Alta:%u, Baja:%u",parte\_alta,parte\_baja); write(txt\_caso4);

potval[adres-2]=(parte\_alta<<8)+parte\_baja;

//sprintf(txt\_caso4,"potval:%u",potval[adres-2]); write(txt\_caso4);

}

delay\_ms(80);

write("sali de valores");

}

unsigned prueba=0,prueba2=0;

//~~~~~~~~~~~~~~~~~~~~~~~Caso 4~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

float fpot[5];

void caso4() {

henhol=0;

//T1CONbits.TON=1;

while (escape==0){

adc\_value = ADC2\_Get\_Sample(14);

voltaje=adc\_value\*0.0034;

valores();

for(i=0;i<10;i++){

fpot[i]=potval[i]\*0.0034;

}

sprintf(txt\_caso41,"V1:%.3f, V2:%.3f, V3:%.3f",fpot[0],fpot[2],fpot[4]);

delay\_ms(50);

sprintf(txt\_caso42,"V4:%.3f,V5:%.3f",fpot[6],fpot[8]);

delay\_ms(50);

hid\_caso\_4(voltaje,henhol);

// bandera4++;

escape=0;

while(!HID\_Read());

for(cnt=0;cnt<64;cnt++) { writebuff[cnt]=readbuff[cnt];}

if(strcmp(readbuff,"b")==0){//CASE 4

write("nada");

}

else if(strcmp(readbuff,"T")==0){//CASE 2

write("actualizar");

adc\_value = ADC2\_Get\_Sample(14);

escape=0;

}

else if(strcmp(readbuff,"K")==0){//CASE 4

if (henhol==10){henhol=0;}

adc\_value = ADC2\_Get\_Sample(14);

guardado\_int1(adc\_value,henhol);

henhol+=2;

escape=0;

}

else {escape++;}

delay\_ms(300);

}

}

//~~~~~~~~~~~~~~~~~~~~~~~Caso 5~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

void update\_hid(){

sprintf(buffer,"%u:%u:%u del dia %u del mes %u,del ano%u",u\_hora[HOR],u\_hora[MIN],u\_hora[SEG],u\_hora[DIA],u\_hora[MES],u\_hora[ANO]);

logd(buffer);

}

void caso5(){

while (escape==0){

hid\_caso\_5();

while(!HID\_Read());

for(cnt=0;cnt<64;cnt++) { writebuff[cnt]=readbuff[cnt];}

if(strcmp(readbuff,"X")==0){

//Iniciar Reloj

enviado="X";

SPI1\_Write(enviado);

logd("Ver glcd pantalla");

//update\_hid()

//Delay\_ms(3000);

escape=0;

}else if(strcmp(readbuff,"Y")==0){

enviado='Y';

SPI1\_Write(enviado);

programar\_RTCC();

escape=0;

}else if(strcmp(readbuff,"U")==0){

enviado='U';

SPI1\_Write(enviado);

logd("RELOJ ACTUALIZADO");

escape=0;

}

else if(strcmp(readbuff,"Z")==0){

enviado='Z';

SPI1\_Write(enviado);

programar\_Alarma();

}else {escape++;

enviado=254;

SPI1\_Write(enviado);

}

delay\_ms(300);

}

}

//Configutacion Hora

//usar programar\_RTCC, programar Alarma;

int update\_hid\_hora(){

int i,j;

while(!HID\_Read()); for(cnt=0;cnt<64;cnt++) {writebuff[cnt]=readbuff[cnt];}

i=readbuff[0]-'0';

j=readbuff[1]-'0';

return (j+i\*10);

}

void hid\_configuracion\_hora(){

sprintf(buffer,"1) SEGUNDO (0-59) %u",u\_hora[SEG]);write(buffer);

sprintf(buffer,"2) MINUTO (0-59) %u",u\_hora[MIN]);write(buffer);

sprintf(buffer,"3) HORA (0-23) %u",u\_hora[HOR]);write(buffer);

sprintf(buffer,"4) DIA (1-28/29/30/31) %u",u\_hora[DIA]);write(buffer);

sprintf(buffer,"5) MES (1-12) %u",u\_hora[MES]);write(buffer);

sprintf(buffer,"6) YEAR (0-99) %u",u\_hora[ANO]);write(buffer);

}

void programar\_RTCC(){

int i;

for(i=0;i<6;i++){u\_hora[i]=0;}

hid\_configuracion\_hora();

for(i=0;i<6;i++){

sprintf(buffer,"Modificando %d",i+1);write(buffer);

u\_hora[i]=update\_hid\_hora();

hid\_configuracion\_hora();

enviado=u\_hora[i];

SPI1\_Write(enviado);

}

write("Mande la vaina");

}

//Alarma

void programar\_Alarma(){

hid\_config\_al();

RTCBandera=0;

while(!HID\_Read());

for(cnt=0;cnt<64;cnt++) {writebuff[cnt]=readbuff[cnt];}

if(strcmp(readbuff,caso\_1)==0){//CASE 1

write("Alarma 30");

enviado=1;

SPI1\_Write(enviado);

}

else if(strcmp(readbuff,caso\_2)==0){//CASE 2

CASO\_ALARMA=RTCC\_60;

write("Alarma 60");

enviado=2;

SPI1\_Write(enviado);

}

else if(strcmp(readbuff,caso\_3)==0){//CASE 3

CASO\_ALARMA=RTCC\_90;

write("Alarma 90");

enviado=3;

SPI1\_Write(enviado);

}

else if(strcmp(readbuff,caso\_4)==0){//CASE 3

CASO\_ALARMA=RTCC\_120;

write("Alarma 120");

enviado=4;

SPI1\_Write(enviado);

}

}

void config\_timer3 () {

TMR3=0;

T3CON=0X0030; //Prescaler 64:1, modo timer

PR3=65000; //1000ms

IEC0bits.T3IE=1;

IFS0bits.T3IF=0;

}

void config\_adc() {

AD1CON1bits.AD12B=0;// ADC de 10 bits

AD1CON1bits.FORM=0; // ENTERO SIN SIGNO

AD1CON1bits.SSRCG=0;

AD1CON1bits.SSRC=2;//TIMER3 inicia la convercion

AD1CON1bits.SIMSAM=1;// Muestreo simultaneo

AD1CON1bits.ASAM=1;

AD1CON1bits.SAMP=0;

AD1CON2bits.VCFG=0;// AVdd y AVss como referencias

AD1CON2bits.CSCNA=0;

AD1CON2bits.CHPS=1;//CH0 y CH1 Act.

AD1CON2bits.SMPI=0;// interrupcion a la primera conversion

AD1CON2bits.BUFM=0; //llenado desde la primera direccion

AD1CON2bits.ALTS=0;

AD1CON3bits.ADRC=0; //reloj interno Tcy

AD1CON3bits.SAMC=11; //10TAD del tiempo de muestreo

AD1CON3bits.ADCS=0; //TAD=Tcy

AD1CON4bits.ADDMAEN=0; //SIN DMA

AD1CHS123bits.CH123NA=0; //entrada negativa es AVss

AD1CHS123bits.CH123SA=0;// CH1 entrada positiva AN0(RB0)

AD1CHS0bits.CH0NA=0; //entrada negativa es AVss

AD1CHS0bits.CH0SA=4;//ENTRADA positiva AN4(4)

IFS0bits.AD1IF=0;//bandera=0

IEC0bits.AD1IE=1;//hab interrupcion

IPC3bits.AD1IP=6;

delay\_ms(10);

} #define FP 15000000

#define BAUDRATE 9600

#define BRGVAL ((FP/BAUDRATE)/16)-1

unsigned char UENVIAR;

void CONFIG\_CM(){

ANSELDbits.ANSD7=1;//ENTRADA ANALOGICA c3in1+ 3V pin 55

ANSELCbits.ANSC14=1;//ENTRADA ANALOGICA c3in1- POT pin 48

ANSELBbits.ANSB5=1;//ENTRADA ANALOGICA c1in+ POT RB5 pin 11

CM3CONbits.COE=0;

CM1CONbits.COE=0;

CM3CONbits.CON=1;

CM1CONbits.CON=1;

CM3CONbits.CPOL=0;

CM1CONbits.CPOL=0;

CM3CONBITS.CREF=0;//CONECTADO A C3IN1+

CM3CONBITS.CCH=1;//CONECTADO A C3IN1-

CM1CONBITS.CREF=0;//CONECTADO A C1IN1+

CM1CONBITS.CCH=11;////CONECTADO A INTREF

CVRCONbits.BGSEL=1;

CVRCONbits.CVREN=1;

CM1CONbits.CEN=1;

CM3CONbits.CEN=1;

}

void config\_timer1() {

TMR1=0;

PR1=65000;

T1CON=0x0030;

IFS0BITS.T1IF=0;

IEC0BITS.T1IE=1;

IPC0BITS.T1IP=5;

}

void InitMCU(){

ANSELC=0x0000; //Configuracion de E/S digitales

ANSELD=0x0000; //Configuracion de E/S digitales

ANSELE=0x0040; //RE6 como entrada analogica

ANSELB=0x0000; //Configuracion de E/S digitales

//CONFIGURACION DEL PLL PARA ALCANZAR UNA VELOCIDAD DE 30MHZ

PLLFBD = 58; //M = 60

CLKDIVbits.PLLPOST = 0; // N1 = 2

CLKDIVbits.PLLPRE = 0; // N2 = 2

OSCTUN = 0;

OSCCON=0x0301;

while (OSCCONbits.COSC != 0x3);

//CONFIGURACION DEL PLL AUXILIAR PARA EL USB

//SE REQUIEREN 48MHZ

ACLKCON3 = 0x24C0;

ACLKDIV3 = 0x7;

ACLKCON3bits.ENAPLL = 1;

while(ACLKCON3bits.APLLCK != 1);

}

void config\_TX(){

U1MODEbits.STSEL=0;

U1MODEbits.PDSEL=0;

U1MODEbits.ABAUD=0;

U1MODEbits.BRGH=0;

U1BRG=BRGVAL;

U1STAbits.UTXISEL1=0;

IEC0BITS.U1TXIE=1;

U1MODEbits.UARTEN=1;

delay\_us(105);

IFS0bits.U1TXIF=0;

IEC0bits.U1TXIE=1;

IPC3bits.U1TXIP=6;

}

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

IPC14bits.QEI1IP=2; // interrupcion del modulo cuadratura 2

IPC2bits.T3IP=5;

IPC6bits.T4IP=3;

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

}

void config\_pin () {

ANSELBbits.ANSB0=1; // RBO analogico

ANSELBbits.ANSB3=1; // RB4 analogico

TRISBbits.TRISB0=1;

TRISBbits.TRISB3=1;

//TRISDbits.TRISD9=1; // entrada opto para motor 1

TRISDbits.TRISD8=1; // entrada opto para motor 2

// RPINR0bits.INT1R=73; // RPI73 en INT1 motor 1

RPINR1bits.INT2R=72; // RPI72 en INT2 motor 2

TRISDbits.TRISD10=0;

TRISFBITS.TRISF5=0;

}

/\*void config\_velocidad () {

IFS1bits.INT2IF=0;

INTCON2bits.INT1EP=0; //flanco positivo

INTCON2bits.INT2EP=0; //flanco positivo

IEC1bits.INT1IE=1;

IEC1bits.INT2IE=1;

IPC5BITS.INT1IP=4;

IPC7bits.INT2IP =4;

}\*/ #define ESC 27

#define DER 0

#define IZQ 1

#define FALLA 1

#define NOFALLA 0

char menu[] = "Menu";

char Bien1[] = "\r Bienvenido al caso 1 \x0a \x0d";

char Bien2[] = "\n Bienvenido al caso 2 \x0a \x0d";

char Space[]="\n \x0a \x0d" ;

char Bien3[] = " Bienvenido al caso 3";

char caso\_1[] = "A",caso\_2[] = "B",caso\_3[] = "C",caso\_4[]="D",caso\_5[]="E";

char POT1[] = " POT 1: \r",POT2[] = " POT 2: \r";

char MOTO1[] = " MOT 1 ACT ",MOTO2[] = " MOT 2 ACT ";

char error1[]= "\r POT 1 HA SUPERADO\n \r";

char error2[]= "\r POT 2 HA SUPERADO \n \r";

char Bien5[] = "\r Bienvenido al caso 5 \r";

char caso1\_1[]="VEL(RPN) ###### | ####";

char caso1\_2[]="SENT ### | ###";

char caso1\_3[]="FALLA ## | ##";

char caso1\_4[]="POT ## | ##";

char caso2\_nor[]="Todo esta funcionando bien | ";

char caso2\_sup[]="Limite Superior | ";

char caso2\_inf[]="Limite Inferior | ";

char caso2\_pot[]= "POT ## | ";

char caso3\_1[]="Salida por el PIN AN18 | OP=#";

char caso3\_2[]="V |0.83 0.93 1.03 1.13 1.24 1.34 1.44 1.55|";

char caso3\_3[]="OP| 0 1 2 3 4 5 6 7|";

char caso3\_4[]="V |1.65 1.75 1.86 1.96 2.06 2.17 2.27 2.37|";

char caso3\_5[]="OP| 8 9 A B C D E F|";

int cont = 0,it=0;

char txt7[30];

void space1(){

while(!HID\_Write(space,64));

}

void space2(){

while(!HID\_Write(space,64));

while(!HID\_Write(space,64));

}

void write(char \*txt){

space2();

while(!HID\_Write(txt,64));

}

//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~HID MENU~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

void menu2(){

write(menu);

write("--------------------------------------------------");

write("CASO A MOTORES");

write("CASO B COMPARACION");

write("CASO C DAC");

write("CASO D I2C");

write("CASO E RTCC");

space1();

space1();

space1();

space1();

}

//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~HID CASO1~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

void update\_caso1(unsigned rpn1,unsigned rpn2){

inttostr(rpn1,txt7);

caso1\_1[9] =txt7[0];

caso1\_1[10]=txt7[1];

caso1\_1[11]=txt7[2];

caso1\_1[12]=txt7[3];

caso1\_1[13]=txt7[4];

caso1\_1[14]=txt7[5];

inttostr(rpn2,txt7);

caso1\_1[45]=txt7[1];

caso1\_1[46]=txt7[2];

caso1\_1[47]=txt7[3];

caso1\_1[48]=txt7[4];

caso1\_1[49]=txt7[5];

}

void update\_caso3(unsigned falla1,unsigned falla2){

//falla=1 si & falla=0 no

if(falla1){

caso1\_3[13]='S';

caso1\_3[14]='I';

}else{

caso1\_3[13]='N';

caso1\_3[14]='O';

}

if(falla2){

caso1\_3[48]='S';

caso1\_3[49]='I';

}else{

caso1\_3[48]='N';

caso1\_3[49]='O';

}

}

void update\_caso4(unsigned pot1,unsigned pot2){

inttostr(pot1,txt7);

caso1\_4[9] =txt7[0];

caso1\_4[10]=txt7[1];

caso1\_4[11]=txt7[2];

caso1\_4[12]=txt7[3];

caso1\_4[13]=txt7[4];

caso1\_4[14]=txt7[5];

inttostr(pot2,txt7);

caso1\_4[45]=txt7[1];

caso1\_4[46]=txt7[2];

caso1\_4[47]=txt7[3];

caso1\_4[48]=txt7[4];

caso1\_4[49]=txt7[5];

}

void update\_caso2(unsigned sent1,unsigned sent2){

//sent=1 izquierda & sent=0 derecha

if(sent1){

caso1\_2[12]='I';

caso1\_2[13]='Z';

caso1\_2[14]='Q';

}else{

caso1\_2[12]='D';

caso1\_2[13]='E';

caso1\_2[14]='R';

}

if(sent2){

caso1\_2[47]='I';

caso1\_2[48]='Z';

caso1\_2[49]='Q';

}else{

caso1\_2[47]='D';

caso1\_2[48]='E';

caso1\_2[49]='R';

}

}

void logd(char \*txt){

write(txt);

space1();

space1();

space1();

space1();

space1();

space1();

space1();

space1();

space1();

}

void hid\_caso\_1(unsigned rpn1,unsigned rpn2,

unsigned sent1,unsigned sent2,

unsigned falla1,unsigned falla2,

unsigned pot1,unsigned pot2){

update\_caso1(rpn1,rpn2);

update\_caso2(sent1,sent2);

update\_caso3(falla1,falla2);

update\_caso4(pot1,pot2);

write(" MOTOR 1 | MOTOR 2");

write("----------------------------|---------------------");

write(caso1\_1);

write(caso1\_2);

write(caso1\_3);

write(caso1\_4);

space1();

space1();

space1();

space1();

space1();

}

void hid\_caso\_3(){

write(Bien3);

write("----------------------------|---------------------");

write(caso3\_1);

write(caso3\_2);

write(caso3\_3);

write(caso3\_4);

write(caso3\_5);

space1();

space1();

space1();

space1();

}

void hid\_caso\_2(unsigned pot){

write(Bien2);

if(pot<1000){

write(caso2\_inf);

}else if(pot>1000){

write(caso2\_sup);

}else{

write(caso2\_nor);

}

inttostr(pot,txt7);

caso2\_pot[9] =txt7[0];

caso2\_pot[10]=txt7[1];

caso2\_pot[11]=txt7[2];

caso2\_pot[12]=txt7[3];

caso2\_pot[13]=txt7[4];

caso2\_pot[14]=txt7[5];

write(caso2\_pot);

space1();

space1();

space1();

space1();

space1();

space1();

space1();

space1();

space1();

}

char txt\_caso4[64]="No hay valores guardados";

char txt\_caso41[64]="No hay valores guardados";

char txt\_caso42[64]="No hay valores guardados";

void hid\_caso\_4(float pot, int henhol){

write("Bienvenido a Caso 4");

write("----------------------------|---------------------");

sprintf(txt7,"Presione K Guardar Valor en V%u",henhol/2);write(txt7);

sprintf(txt7,"Valor de Voltaje Actual %.3f",pot);write(txt7);

write(txt\_caso41);

space1();

write(txt\_caso42);

space1();

space1();

space1();

}

void hid\_caso\_5(){

write("Caso 5");

write("--------------------------------------------------");

write("CASO X INICIAR RELOJ");

write("CASO Y PROGRAMAR RELOJ");

write("CASO Z PROGRAMAR ALARMA");

space2();space1();

space2();

}

void hid\_config\_al(){

write("Configuracion Alarma");

write("--------------------------------------------------");

write("CASO A Cada 30s");

write("CASO B Cada 60s");

write("CASO C Cada 90s");

write("CASO D Cada 120s");

space2();space2();space1();

} // 2.8.- Se debe activar y visualizar un reloj digital,

//usando el RTCC del DsPIC,

//con la opción de programar la fecha y

// hora inicial y una alarma repetitiva

//cada 30 , 60 , 90 o 120 segundos, y

// que una vez activada, encienda y apague un led externo.

// Nota: Estudiantes que logre activar el RTCC

//usando el cristal externo de 32 Khz obtendrá una nota extra.

#define RTCC\_NONE 0

#define RTCC\_30 1

#define RTCC\_60 2

#define RTCC\_90 3

#define RTCC\_120 4

#define SEG 0

#define MIN 1

#define HOR 2

#define DIA 3

#define MES 4

#define ANO 5

#define SEM 6

#define SEGMIN 0

#define HORSEM 1

#define DIAMES 2

#define ANONAD 3

unsigned short rtcc\_VAL=0,RTCBandera=0,CASO\_ALARMA=RTCC\_NONE;

unsigned short u\_hora [7]={0,0,0,0,0,0,0};

char buffer[30];

void config\_Pwm\_reloj() {

PTPER = 458; /\* Periodo del PWM en tiempo base primario \*/

/\* Desplazamiento de fase \*/

PHASE1 = 0;

/\* Ciclo útil\*/

PDC1 = 227;

/\* Tiempo muerto\*/

DTR1 = 10;

ALTDTR1 = 10;

/\* Modo Push-Pull \*/

IOCON1 = 0xC800;

/\* Tiempo base primario, Modo Flanco Alineado y ciclo útil independiente \*/

PWMCON1 = 0x0000;

/\* prescaler 1:1 \*/

PTCON2 = 0x0000;

/\* habilitación de módulo PWM\*/

PTCON = 0x8000;

}

RX

#include "config\_RX.h"

#include "motores.h"

#include "sprites.h"

#include "RTCC.h"

#define ESC\_key 254

//~~~~~~~~~~~~~~~~~~~~~~~~Variables del sistema~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

unsigned short dato=0, dato2=0;

char tecla, buff;

int pantalla=0;

char txt\_hora[2];

char txt[7]={'#','#','#','#','#','#','#'};

int cnt,rpm1=0,rpm2=0;

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

void caso2();void caso3();void caso1();void caso5(); void update\_hid(); void caso4();

void encender\_led(){LATFBITS.LATF4=1;delay\_ms(92);LATFBITS.LATF4=0;}

//void update\_hid(){encender\_led();}

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

/\*void timer8 () org 0x7A {

IFS3bits.T8IF=0;

rpm=0;

rpm2=0;

}\*/

void selector\_sprite(int caso,int cnt){

if (caso==0)

glcd\_write\_text("Error CASE\_NULL",64,4,1);

if (caso==1)

{ encender\_led();

if(cnt==1){animate\_64b(mayorquea);}

else if(cnt==2){animate\_64b(mayorqueb);}

else if(cnt==3){animate\_64b(mayorquec);}

}

if (caso==2)

{ encender\_led(); if(cnt==1){animate\_64b(menorqueb);}

else if(cnt==2){animate\_64b(menorquec);}

else if(cnt==3){animate\_64b(menorquea);}

}

if (caso==3)

{ encender\_led(); if(cnt==1){animate\_64b(pwm3a);}

else if(cnt==2){animate\_64b(pwm3b);}

else if(cnt==3){animate\_64b(pwm4c);}

}

if (caso==4)

{ encender\_led(); if(cnt==1){animate\_64b(pwm4a);}

else if(cnt==2){animate\_64b(pwm4b);}

else if(cnt==3){animate\_64b(pwm4c);}

}

}

//------------------------------------

void INT\_T05s\_T2()org 0x22{

T05s++;

inttostr(T05s,txt);glcd\_write\_text(txt,0,3,1);

if(T05s%2==0){T1s++;}

if(T05s>6){T2CONbits.TON=0;T1s=0;T05s=0;}

if(caso!=CASE\_NULL&&T1s==3){T1s=0;}

else if(caso==CASE\_NULL&&T1s==2){T1s=0;}

selector\_sprite(caso,T1s);

IFS0bits.T2IF=0;

}

void INT\_RELOJ()org 0x000090{

LATFBITS.LATF4=0;

switch(CASO\_ALARMA){

case RTCC\_30:

if(RTCBandera%3==0){encender\_led();glcd\_write\_text("ALRM 60s",64,6,1);}

break;

case RTCC\_60:

if(RTCBandera%6==0){encender\_led();glcd\_write\_text("ALRM 60s",64,6,1);}

break;

case RTCC\_90:

if(RTCBandera%9==0){encender\_led();glcd\_write\_text("ALRM 90s",64,6,1);}

break;

case RTCC\_120:

if(RTCBandera%12==0){encender\_led();glcd\_write\_text("ALRM 120s",64,6,1);}

break;

default:

case RTCC\_NONE:

RTCBandera=0;

break;

}

sprintf(buffer,"ALRM #%u",RTCBandera+1);

glcd\_write\_text(buffer,64,7,1);

RTCBandera++;

IFS3bits.RTCIF=0;

}

void interupcion\_ext1() org 0x3C {IFS1bits.INT1IF=0; rpm1++;}

void interupcion\_ext2() org 0x4E {IFS1bits.INT2IF=0; rpm2++;}

void timer7 () org 0x74 {

IFS3bits.T7IF=0;

rpm1=rpm1\*75;

rpm2=rpm2\*75;

inttostr(rpm1,txt);

glcd\_write\_text(txt,64,0,1);

inttostr(rpm2,txt);

glcd\_write\_text(txt,64,1,1);

rpm1=0;

rpm2=0;

}

void PWM4() org 0xD6{

IFS6bits.PWM4IF=0;

PWMCON3bits.FLTSTAT=0;

if(PORTDbits.RD11==0){

glcd\_write\_text("Falla Motor 4",64,7,1);

IOCON4bits.FLTDAT=0;

PWMCON4bits.FLTSTAT=0;

caso=CASE\_PWM4; T2CONbits.TON=1;

}

}

void PWM3() org 0xD4{

IFS6bits.PWM3IF=0;

PWMCON3bits.FLTSTAT=0;

if(PORTDbits.RD0==0){

IOCON4bits.FLTDAT=0;

PWMCON4bits.FLTSTAT=0;

caso=CASE\_PWM3; T2CONbits.TON=1;

glcd\_write\_text("Falla Motor 3",64,6,1);

}

}

void update\_hid(){

rtcc\_read();rtc2int();

//sprintf(buffer,"%02.2u|%02.2u",minyseg,diasyhora);

//glcd\_write\_text(buffer,64,4,1);

//sprintf(buffer,"%02.2u|%02.2u",mesydia,year);

//glcd\_write\_text(buffer,64,5,1);

sprintf(buffer,"%02.2u:%02.2u:%02.2u",hora,minn,seg);

glcd\_write\_text(buffer,64,1,1);

sprintf(buffer,"%02.2u/%02.2u/%02.2u",dia,mes,anio);

glcd\_write\_text(buffer,64,2,1);

}

//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~MAIN~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

void Error\_SPI() org 0x26{IFS0bits.SPI1EIF=0;}

void SPI() org 0x28{

Glcd\_Write\_Text("si interrupmpe", 60, 5, 1);

dato=SPI1BUF;

Glcd\_Write\_Text("algo esta en dato", 60, 2, 1);

delay\_ms(1000) ;

IFS0bits.SPI1IF=0;

}

void main (){

ANSELC=0;ANSELD=0;ANSELE=0;ANSELB=0;

TRISFBITS.TRISF5=0;

SRbits.IPL=0;

CORCONbits.IPL3=0;

INTCON1bits.NSTDIS=0;

INTCON2bits.GIE=1;

IPC2bits.SPI1EIP=3;

IPC2bits.SPI1IP=3;

IEC0bits.SPI1EIE=0;

//CONFIGURACION DEL PLL PARA ALCANZAR UNA VELOCIDAD DE 30MHZ

PLLFBD = 58; //M = 60

CLKDIVbits.PLLPOST = 0; // N1 = 2

CLKDIVbits.PLLPRE = 0; // N2 = 2

OSCTUN = 0;

OSCCON=0x0301;

while (OSCCONbits.COSC != 0x3);

//CONFIGURACION DEL PLL AUXILIAR PARA EL USB

//SE REQUIEREN 48MHZ

ACLKCON3 = 0x24C0;

ACLKDIV3 = 0x7;

ACLKCON3bits.ENAPLL = 1;

while(ACLKCON3bits.APLLCK != 1);

SPI1\_Init\_Advanced(\_SPI\_SLAVE,\_SPI\_8\_BIT,\_SPI\_PRESCALE\_SEC\_8,

\_SPI\_PRESCALE\_PRI\_64,\_SPI\_SS\_DISABLE,\_SPI\_DATA\_SAMPLE\_MIDDLE

,\_SPI\_CLK\_IDLE\_LOW, \_SPI\_ACTIVE\_2\_IDLE);

RPINR20bits.SDI1R=46; //SDI1

RPINR20bits.SCK1R=47; //SCK1

RPOR9bits.RP101R=5; //SDO

config\_IO();config\_vref();

config\_LCD(); config\_timer2();

while(1){

glcd\_fill(0);

Glcd\_write\_text("Esperando...",64,0,1);

dato=0;

while(!SPI1STATbits.SPIRBF); //Esperando que llegue el dato

dato=0;

dato=SPI1\_Read(buff);

if (dato==1) {

Glcd\_write\_text("P1",64,0,1);

caso1();

}

else if (dato==2) {

Glcd\_write\_text("P2",64,0,1);

caso2();

}

else if (dato==3) {

Glcd\_write\_text("P3",64,0,1);

caso3();

}

else if (dato==4) {

caso4();

}

else if (dato==5) {

Glcd\_write\_text("P5",64,0,1);

caso5();

}

}

}

//-------------------------------------------------CASO 1------------------------------------------------------

void caso1(){

config\_motor();config\_timer7();//config\_velocidad ();

//T8CONbits.TON=1;

glcd\_write\_text("Caso 1",64,2,1);

delay\_ms(100);

// T7CONbits.TON=1; // activa conteo para RPM de motores

while (dato!=ESC\_key){

encender\_led();

while(!SPI1STATbits.SPIRBF); //Esperando que llegue el dato

dato=SPI1\_Read(buff);

if (dato==ESC\_key){continue;}

PDC3=ajuste(dato);

while(!SPI1STATbits.SPIRBF); //Esperando que llegue el dato

dato2=SPI1\_Read(buff);

if (dato2==ESC\_key){continue;}

PDC4=ajuste(dato2);

pantalla=PDC3;

inttostr(pantalla,txt);

glcd\_write\_text(txt,64,6,1);

pantalla=PDC4;

floattostr(pantalla,txt);

glcd\_write\_text(txt,64,7,1);

}

T8CONbits.TON=0;

dato=0;

}

//-------------------------------------------------CASO 2------------------------------------------------------

void caso2(){

GLCD\_fill(0);

glcd\_write\_text("Caso 2",64,0,1);

while (dato!=ESC\_key){

dato=SPI1\_Read(buff);

if (dato==1) {

caso=CASE\_MAY;

T2CONbits.TON=1;

}

if (dato==2) {

caso=CASE\_MEN;

T2CONbits.TON=1;

}

if (dato==3) {

GLCD\_fill(0);

glcd\_write\_text("NORMAL",64,0,1);

}

}

}

//-------------------------------------------------CASO 3------------------------------------------------------

void caso3(){

GLCD\_FILL(0);

dato=0;

glcd\_write\_text("Caso 3",64,4,1);

delay\_ms(50);

while (dato!=ESC\_key){

while(!SPI1STATbits.SPIRBF); //Espera que reciba un dato

dato=SPI1\_Read(buff);

if(dato<16&&dato>=0){

pantalla=dato;

CVRCONbits.CVR=dato;

//glcd\_write\_text(space,65,7,1);

floattostr(pantalla,txt);

glcd\_write\_text(txt,65,7,1);

dato=0;

}

}

dato=0;

}

//---------------------------caso 4--------------------------------

void caso4() {

Glcd\_fill(0);

glcd\_write\_text("CASO 4",65,0,1);

delay\_ms(2000);

}

//-------------------------------------------------CASO 5------------------------------------------------------

void caso5(){

update\_hid();

config\_RTCC();

GLCD\_fill(0);

glcd\_write\_text("Caso 5",64,0,1);

while (dato!=ESC\_key){

Update\_hid();

dato=SPI1\_Read(buff);

if (dato=='X') {

glcd\_write\_text("RelojEN",64,0,1);

RCFGCALbits.RTCEN = 0;// Limpia bit RTCEN , desactiva modulo

}

else if (dato=='Y') {

glcd\_write\_text("Config Reloj",64,0,1);

while(!SPI1STATbits.SPIRBF); dato=SPI1\_Read(buff);

seg=dato;encender\_led();

while(!SPI1STATbits.SPIRBF); dato=SPI1\_Read(buff);

minn=dato;encender\_led();

while(!SPI1STATbits.SPIRBF); dato=SPI1\_Read(buff);

hora=dato;encender\_led();

while(!SPI1STATbits.SPIRBF); dato=SPI1\_Read(buff);

dia=dato;encender\_led();

while(!SPI1STATbits.SPIRBF); dato=SPI1\_Read(buff);

mes=dato;encender\_led();

while(!SPI1STATbits.SPIRBF); dato=SPI1\_Read(buff);

anio=dato;encender\_led();

RTCC\_assembler(anio,mes,dia,hora,minn,seg);

PADCFG1bits.RTSECSEL=1; // habilita salida de reloj a un segundo

RCFGCALbits.RTCOE=1; // habilita el pin RTCC ( se puede usar para verificar que el RTCC está funcionando )

delay\_ms(50);

}

else if (dato=='Z') {

RTCBandera=0;

glcd\_write\_text("Config Alarma",64,0,1);

while(!SPI1STATbits.SPIRBF);

dato=SPI1\_Read(buff);

switch (dato){

case 1:

glcd\_write\_text("CALRM 30s",64,6,1);

CASO\_ALARMA=RTCC\_30;

break;

case 2:

glcd\_write\_text("CALRM 60s",64,6,1);

CASO\_ALARMA=RTCC\_60;

break;

case 3:

glcd\_write\_text("CALRM 90s",64,6,1);

CASO\_ALARMA=RTCC\_90;

break;

case 4:

glcd\_write\_text("CALRM 120s",64,6,1);

CASO\_ALARMA=RTCC\_120;

break;

default:

glcd\_write\_text("CALRM NONE",64,6,1);

CASO\_ALARMA=RTCC\_NONE;

break;

}

ALCFGRPTbits.ALRMEN = 0;

ALCFGRPTbits.ALRMPTR=0;

ALRMVAL = 0x0000;

ALCFGRPTbits.AMASK=0b0010;

ALCFGRPTbits.CHIME = 1;

ALCFGRPTbits.ALRMEN = 1;

}

}

}

/\*void main() {

InitMCU();config\_IO();config\_vref();encender\_led();

delay\_ms(50);

config\_INT(); UART1\_Init(9600);

// config\_timer8();

<<<<<<< HEAD

config\_TMR2\_ANIM ();

=======

config\_TMR2\_ANIM (); config\_LCD();

>>>>>>> 0a11a144dccf812f1381596135ce4779763da330

//animate\_charmander\_2s(); Glcd\_Fill(0);

//encender\_led();

while(1){

encender\_led();

glcd\_write\_text("Laboratorio 4",64,0,1);

glcd\_write\_text("Esperando Comando",64,1,1);

delay\_ms(100);

while(!SPI1STATbits.SPIRBF); //Espera que reciba un dato

dato=UART1\_Read();

pantalla=dato;

floattostr(pantalla,txt);

glcd\_fill(0);

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

glcd\_write\_text("recibio algo",60,0,1);

if (dato>0&&dato<4) {

switch(dato){

case 1:

caso1();

PDC4=15000;

PDC3=15000;

T7CONbits.TON=0;

IPC5BITS.INT1IP=0;

IPC7bits.INT2IP=0;

Glcd\_Fill(0);

break;

case 2:

caso2();Glcd\_Fill(0);

break;

case 3:

caso3();Glcd\_Fill(0);

break;

default:

glcd\_write\_text("Error",64,4,1);

delay\_ms(100);

break;

}

dato=0;

}

}

}\*/

void config\_PWM();void config\_PWM3();

void config\_motor() {

config\_PWM(); config\_PWM3();

}

void config\_PWM(){

IEC6bits.PWM4IE=1; // interrupcion pwm4

IFS6bits.PWM4IF=0; // limpia bandera pwm4

IPC24bits.PWM4IP=6; // nivel de interrupcion pwm4

PTCONbits.PTEN=1;

PWMCON4bits.FLTSTAT=0; //bandera de pin de falla

PWMCON4bits.FLTIEN=1;

PWMCON4bits.ITB=1;

PWMCON4bits.CAM=1;

PWMCON4bits.IUE=1;

IOCON4bits.PENH=1;

IOCON4bits.PENL=1;

IOCON4bits.POLH=0;

IOCON4bits.POLL=0;

DTR4=1000;

ALTDTR4=1000;

PHASE4=30000; // periodo de 2ms

PDC4=20000; //100%

IOCON4bits.FLTDAT=0; // falla en pin pone bajo todo

FCLCON4=0x0015;//falla indep. act,pin falla 3,act en baj0

//FCLCON4bits.FLTSRC=3; // pin de falla 4

}

void config\_PWM3(){

IEC6bits.PWM3IE=1; // interrupcion pwm4

IFS6bits.PWM3IF=0; // limpia bandera pwm4

IPC24bits.PWM3IP=6; // nivel de interrupcion pwm4

PTCONbits.PTEN=1;

PWMCON3bits.FLTSTAT=0; //bandera de pin de falla

PWMCON3bits.FLTIEN=1;

PWMCON3bits.ITB=1;

PWMCON3bits.CAM=1;

PWMCON3bits.IUE=1;

IOCON3bits.PENH=1;

IOCON3bits.PENL=1;

IOCON3bits.POLH=0;

IOCON3bits.POLL=0;

DTR3=1000;

ALTDTR3=1000;

PHASE3=30000; // periodo de 2ms

PDC3=20000; //100%

IOCON3bits.FLTDAT=0; // falla en pin pone bajo todo

FCLCON3=0x0015;//falla indep. act,pin falla 3,act en baj0

//FCLCON3bits.FLTSRC=2; // pin de falla 3

}

//Variables de GLCD

sbit GLCD\_D0 at RD4\_bit;

sbit GLCD\_D1 at RD5\_bit;

sbit GLCD\_D2 at RD6\_bit;

sbit GLCD\_D3 at RD7\_bit;

sbit GLCD\_D4 at RF0\_bit;

sbit GLCD\_D5 at RF1\_bit;

sbit GLCD\_D6 at RE0\_bit;

sbit GLCD\_D7 at RE1\_bit;

sbit GLCD\_CS1 at LATE2\_bit;

sbit GLCD\_CS2 at LATE3\_bit;

sbit GLCD\_RS at LATD1\_bit;

sbit GLCD\_RW at LATD2\_bit;

sbit GLCD\_EN at LATD3\_bit;

sbit GLCD\_RST at LATB5\_bit;

sbit GLCD\_D0\_Direction at TRISD4\_bit;

sbit GLCD\_D1\_Direction at TRISD5\_bit;

sbit GLCD\_D2\_Direction at TRISD6\_bit;

sbit GLCD\_D3\_Direction at TRISD7\_bit;

sbit GLCD\_D4\_Direction at TRISF0\_bit;

sbit GLCD\_D5\_Direction at TRISF1\_bit;

sbit GLCD\_D6\_Direction at TRISE0\_bit;

sbit GLCD\_D7\_Direction at TRISE1\_bit;

sbit GLCD\_CS1\_Direction at TRISE2\_bit;

sbit GLCD\_CS2\_Direction at TRISE3\_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 TRISB5\_bit;

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

void config\_LCD(){

Glcd\_Init();

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

Glcd\_Fill(0);

}

void config\_IO(){

ANSELB=ANSELC=ANSELD=ANSELE=0;

TRISDbits.TRISD0=1; //Pines falla

TRISDbits.TRISD11=1; //Pines falla

TRISDbits.TRISD9=1; //entrada opto para motor 1

TRISDbits.TRISD8=1; //entrada opto para motor 2

TRISFbits.TRISF4=0; //LED de prueba;

RPOR9bits.RP101R=1; //U1TX

RPINR0bits.INT1R=73; // RPI73 en INT1 motor 1 RPM

RPINR1bits.INT2R=72; // RPI72 en INT2 motor 2 RPM

RPINR18bits.U1RXR=46; //U1RX

// RPINR13bits.FLT4R=75; // pin de falla pwm4

// RPINR13bits.FLT3R=64; //pin de falla pwm3

}

void config\_TMR2\_ANIM () {

TMR2=0;

T2CON=0x0030; //Prescaler 256:1, modo timer

PR2=65350; //1000ms

IEC0bits.T2IE=1;

IFS0bits.T2IF=0;

IPC1bits.T2IP=7;

}

void InitMCU(){

//CONFIGURACION DEL PLL PARA ALCANZAR UNA VELOCIDAD DE 30MHZ

PLLFBD = 58; //M = 60

CLKDIVbits.PLLPOST = 0; // N1 = 2

CLKDIVbits.PLLPRE = 0; // N2 = 2

OSCTUN = 0;

OSCCON=0x0301;

while (OSCCONbits.COSC != 0x3);

//CONFIGURACION DEL PLL AUXILIAR PARA EL USB

//SE REQUIEREN 48MHZ

ACLKCON3 = 0x24C0;

ACLKDIV3 = 0x7;

ACLKCON3bits.ENAPLL = 1;

while(ACLKCON3bits.APLLCK != 1);

}

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

// IEC3bits.T8IE=1;

IFS3bits.T7IF=0;

IEC3bits.T7IE=1;

// IPC12bits.T8IP=7;

IPC12bits.T7IP=5;

}

void config\_timer8() {

TMR8=0;

PR8=58594;

T8CON=0x0030;

}

void config\_timer2() {

TMR2=0;

PR2=58594;

T2CON=0x0030;

IFS0bits.T2IF=0;

IEC0bits.T2IE=1;

IPC1bits.T2IP=5;

}

void config\_timer7() {

TMR7=0;

PR7=23438;

T7CON=0x0030;

}

void config\_vref () {

//CM1CONbits.CON=1;

CVRCONbits.CVREN=1;

CVRCONbits.CVROE=1;

CVRCONbits.BGSEL=1;

CVRCONbits.CVRR=0;

CVRCONbits.CVR=2;

CVRCONbits.VREFSEL=0;

}

void config\_velocidad () {

IFS1bits.INT2IF=0;

INTCON2bits.INT1EP=0; //flanco positivo

INTCON2bits.INT2EP=0; //flanco positivo

IEC1bits.INT1IE=1;

IEC1bits.INT2IE=1;

IPC5BITS.INT1IP=4;

IPC7bits.INT2IP =4;

}

unsigned ajuste (unsigned control) {

if (control<=20) {

return 6500;

}else

if (control>20&&control<=60) {

return 10000;

} else

if (control>60&&control<=80) {

return 11000;

} else

if (control>80&&control<=90) {

return 12000;

} else

if (control>90&&control<=140) {

return 15000;

} else

if (control>140&&control<=160) {

return 19000;

} else

if (control>160&&control<=180) {

return 20000 ;

} else

if (control>180&&control<=210) {

return 21500;

} else

if (control>210&&control<=255) {

return 23800;

}

}

// 2.8.- Se debe activar y visualizar un reloj digital,

//usando el RTCC del DsPIC,

//con la opción de programar la fecha y

// hora inicial y una alarma repetitiva

//cada 30 , 60 , 90 o 120 segundos, y

// que una vez activada, encienda y apague un led externo.

// Nota: Estudiantes que logre activar el RTCC

//usando el cristal externo de 32 Khz obtendrá una nota extra.

#define RTCC\_NONE 0

#define RTCC\_30 1

#define RTCC\_60 2

#define RTCC\_90 3

#define RTCC\_120 4

unsigned short rtcc\_VAL=0,RTCBandera=0,CASO\_ALARMA=RTCC\_NONE;

char buffer[30];

unsigned short int anio=0,mes=0,dia=0,hora=0,seg=0,minn=0 ;

/\*fecha tiene {Unidad\_DIA,Decena\_Dia,Unidad\_A~o,Decena A~o}

rtc\_hora es una copia de RTCVAL en su config cochina de microchip

rtc\_hora[6]{seg,min,hora,dia,mes,a~o}\*/

int year=0x0000, mesydia=0x0000, diasyhora=0x0000, minyseg=0x0000;

// \*\*\*\_hora[4]{SEGMIN,HORAdiadelasemana,DIAMES,ANONAD} se usan las mayusculas

//~~~~~~~~~~~~~~~~~~~~~~~Capa Hardware~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

void rtcc\_read(){

//while(RCFGCALbits.RTCSYNC==0){

RCFGCALbits.RTCPTR=3;

year=RTCVAL;

mesydia=RTCVAL;

diasyhora=RTCVAL;

minyseg=RTCVAL;

//}

}

void RTCC\_assembler (short yea,short mon,short day,short hor,short min,short seg){

asm {MOV # 0x55, W0}

asm {MOV W0, NVMKEY }

asm {MOV # 0xAA, W0 }

asm {MOV W0, NVMKEY }

asm {BSET RCFGCAL, # RTCWREN;}// Setea el bit RTCWREN

RCFGCALbits.RTCEN = 0;// Limpia bit RTCEN , desactiva modulo

ALCFGRPTbits.ALRMEN=0; //desactivar alarma

// Escribe a Timer RTCC

RCFGCALbits.RTCOE = 1; // habilita el pulso de salida RTCC

PADCFG1bits.RTSECSEL = 0; // salida de la alarma

//Configura la fecha y hora al RTC

RCFGCALbits.RTCPTR = 3;

RCFGCALbits.RTCPTR=3;

RTCVAL = Dec2Bcd(yea);

RTCVAL = (Dec2Bcd(day)|Dec2Bcd(mon)<<8);

RTCVAL = Dec2Bcd(hor);

RTCVAL = (Dec2Bcd(seg)|Dec2Bcd(min)<<8);

RCFGCALbits.RTCEN = 1;

RCFGCALbits.RTCWREN = 0; // deshabilita escritura en RTCC

delay\_ms(1000);

}

void config\_RTCC(){

OSCCONbits.LPOSCEN=1; //activa el oscilador secundario

RTCC\_assembler(96,04,01,05,15,05);

PADCFG1bits.RTSECSEL=1; // habilita salida de reloj a un segundo

RCFGCALbits.CAL=1; //ajusta a cero

RCFGCALbits.RTCOE=1; // habilita el pin RTCC ( se puede usar para verificar que el RTCC está funcionando )

IFS3bits.RTCIF=0;//Habilitacion de Interrupcion

IEC3bits.RTCIE=1;

IPC15bits.RTCIP=6;

}

//~~~~~~~~~~~~~~~~~~~~Movimiento entre capas~~~~~~~~~~~~~~~~~~~~

void rtc2int(){

anio= Bcd2Dec( year & 0x00FF );

mes= Bcd2Dec((mesydia & 0xFF00)>>8 );

dia= Bcd2Dec( mesydia & 0x00FF );

hora= Bcd2Dec( diasyhora & 0x00FF );

minn= Bcd2Dec((minyseg & 0xFF00)>>8 );

seg= Bcd2Dec( minyseg & 0x00FF );

//minn=minn-64;

//hora=hora-20

//dia=dia-159;

}

void int2rtc(){

year=(Dec2Bcd(anio));

mesydia=Dec2Bcd(dia)|(Dec2Bcd(mes)<<8);

diasyhora=Dec2Bcd(hora);

minyseg=(Dec2Bcd(seg)|(minn)<<8);

}

//hid