Universidad Del Valle De Guatemala 01/02/2021

Digital 2 Helder Ovalle Barrios

Sección: 18349

**Labrotorio #3**

**Link de github:**

<https://github.com/Helder1121/Labsdigitaldos>

**Link del video:**

<https://www.youtube.com/watch?v=3-n3GE8pubo>

**Diagrama de flujo:**

Lab 3(LCD) Pre-lab

Crear la librería LCD DE 8 bits para la comunicación con USART también crear su librería. Incorporando la librería del ADC (lab2).

Config del ADC para leer los potenciómetros. En los puertos RA0 y RA1.

(Con case…)

Guardar el valor recibido en el USART

Mandar los valores al USART

Mnadar los datos la consolar (RC6 y RC7)

Convertir los valores obtenidos a voltajes

Verificar el valor si es “+” o “-”

Enviar los resultados a la LCD. La LCD estará en el puerto D.

If “+” Incrementa else “-” decrementa

Desplegar los valores en dados en la LCD.

**Progra comentada:**

/\*

\* File: lab03.c

\* Author: Helder Ovalle

\*

\* Created on 8 de febrero de 2021, 10:35 AM

\*/

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Palabra de configuración

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// CONFIG1

#pragma config FOSC = XT // Oscillator Selection bits (XT oscillator: Crystal/resonator on RA6/OSC2/CLKOUT and RA7/OSC1/CLKIN)

#pragma config WDTE = OFF // Watchdog Timer Enable bit (WDT disabled and can be enabled by SWDTEN bit of the WDTCON register)

#pragma config PWRTE = OFF // Power-up Timer Enable bit (PWRT disabled)

#pragma config MCLRE = OFF // RE3/MCLR pin function select bit (RE3/MCLR pin function is MCLR)

#pragma config CP = OFF // Code Protection bit (Program memory code protection is disabled)

#pragma config CPD = OFF // Data Code Protection bit (Data memory code protection is disabled)

#pragma config BOREN = ON // Brown Out Reset Selection bits (BOR disabled)

#pragma config IESO = OFF // Internal External Switchover bit (Internal/External Switchover mode is disabled)

#pragma config FCMEN = OFF // Fail-Safe Clock Monitor Enabled bit (Fail-Safe Clock Monitor is disabled)

#pragma config LVP = OFF // Low Voltage Programming Enable bit (RB3 pin has digital I/O, HV on MCLR must be used for programming)

// CONFIG2

#pragma config BOR4V = BOR40V // Brown-out Reset Selection bit (Brown-out Reset set to 4.0V)

#pragma config WRT = OFF // Flash Program Memory Self Write Enable bits (Write protection off)

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Importación de librerías

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include <xc.h>

#include <stdint.h>

//Permite realizar los prints

#include <stdio.h>

#include <pic16f887.h>

#include "LCD.h"

#include "ADC.h"

#include "USART.h"

#define \_XTAL\_FREQ 8000000

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Variables

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

char data[16];//Variable mostrara los valos en la lcd

float volt, volt2;//variable para los voltajes en los pots

char LecUSART = 0;

char entregado = 0;

uint8\_t contador = 0;

//uint8\_t valorADC1;

//uint8\_t valorADC2;

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Prototipos de funciones

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void config\_P();

float ADC\_1(void);

float ADC\_2(void);

void Enviar\_1(void);

void Enviar\_2(void);

//float Volts\_Bina(uint8\_t b);

//Interrupcion del RCIF

void \_\_interrupt() ISR(){

if (RCIF == 1){

RCIF = 0;

LecUSART = Read\_USART();

if(LecUSART=='+'){

contador++;}

else if(LecUSART=='-'){

contador--;}

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Ciclo principal

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void main(void){

config\_P();

config\_ADC();

\_baudios();

config\_txsta();

config\_rcsta();

Lcd\_Init();

LCD\_Limpia();

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Loop principal

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

while(1){

LCD\_Limpia();

Lcd\_Set\_Cursor(1, 1);

Lcd\_Write\_String("S1 S2 CONT ");

//Lo que se muestra en la LCD en la primera fila

ADC\_1();

ADC\_2();

sprintf(data, "%1.2f %1.2f %d" ,volt,volt2,contador);

//Despliega en dos decimales el voltaje de 0V-5V

Lcd\_Set\_Cursor(2, 1);

Lcd\_Write\_String(data);//Mostrara el valor en la LCD

Write\_USART\_String("S1 S2 CONT");

//Mensaje que se muestra en la terminal en la segunda linea

Write\_USART(13);

Write\_USART(10);

//Saltar lineas

Write\_USART\_String(data);

Write\_USART(13);

Write\_USART(10);

//Saltar lineas

if (RCIF == 1){

entregado = RCREG;

if(entregado == '+'){contador = contador +1;}

if(entregado == '-'){contador = contador -1;}

}

\_\_delay\_ms(500);

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Configuración

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void config\_P(){

//Configuracion de los puertos

TRISD = 0;

TRISE = 0;

TRISA = 3;

//TRISCbits.TRISC7 = 1;

//TRISCbits.TRISC6 = 0;

ANSEL = 3;//Para los potenciometros

ANSELH = 0;

//Steo los puertos

PORTD = 0;

PORTE = 0;

PORTC = 0;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Funciones

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

float ADC\_1(void){

Canal\_ADC(0);//canal 0

//Configuracion bits ADCON0

ADCON0bits.ADCS0 = 1;//Clock ADC conversion

ADCON0bits.ADCS1 = 0;

ADCON0bits.ADON = 1;//Habilitamos el ADC

\_\_delay\_ms(0.25);//Para la conversion

ADCON0bits.GO = 1;//Inicia la conversion

while (ADCON0bits.GO == 1){

volt = ((ADRESH \* 5.0)/255);//Conversion de 0V-5V

}

}

float ADC\_2(void){

Canal\_ADC(1);//Canal 1

//Configuracion bits ADCON0

ADCON0bits.ADCS0 = 1;//Clock ADC conversion

ADCON0bits.ADCS1 = 0;

ADCON0bits.ADON = 1;//Habilitamos el ADC

\_\_delay\_ms(0.25);//Para la conversion

ADCON0bits.GO = 1;//Inicia la conversion

while (ADCON0bits.GO == 1){

volt2 = ((ADRESH \* 5.0)/255); //Conversion de 0V-5V

}

}

void Enviar\_1(void){//Envio de datos

TXREG = volt;

while (TXSTAbits.TRMT == 1){//Retorna y envia el voltaje a ADC1

return;

}

}

void Enviar\_2(void){//Envio de datos

TXREG = volt2;

while (TXSTAbits.TRMT == 1){//Retorna y envia el voltaje a ADC2

return;

}

}

Librería USART

#include <xc.h>

#include <pic16f887.h>

#include "USART.h"

void \_baudios(void){

SPBRG = 12; //9600 baudios para 8MHZ

}

//Configuracion dada en el datasheet

void config\_txsta(void){

TXSTAbits.CSRC = 0;//Clock terminal

TXSTAbits.TX9 = 0;//8 bits de transmicion

TXSTAbits.TXEN = 1;//Transmicion habilitada

TXSTAbits.SYNC = 0;//modo asincrono

TXSTAbits.BRGH = 0;//low speed

TXSTAbits.TRMT = 0;//Tsr full

TXSTAbits.TX9D = 0;

}

//Configuracion dada en el datasheet

void config\_rcsta(void){

RCSTAbits.SPEN = 1;//Se habilita el puerto serial

RCSTAbits.RX9 = 0;

RCSTAbits.SREN = 0;

RCSTAbits.CREN = 1;//Recibir habilitadp

RCREG = 0;

}

//Extraido de https://electrosome.com/uart-pic-microcontroller-mplab-xc8/

void Write\_USART(uint8\_t a){

while(!TRMT);

TXREG=a;

}

void Write\_USART\_String(char \*a){

uint8\_t i;

for(i=0;a[i]!='\0';i++){

Write\_USART(a[i]);

}

}

uint8\_t Read\_USART(){

while(!RCIF);

return RCREG;

}

Librería LCD

//Libreria de Pablo Mazariegos en clase de 4 bits modificada a unos de 8bits

#include <xc.h>

#include <stdint.h>

#include "LCD.h"

#define \_XTAL\_FREQ 8000000

//Funcion para indicar el caracter segun sea el tamaño del mismo.

void Puerto(uint8\_t x){

if(x & 1){D0 = 1;}else{D0 = 0;}

if(x & 2){D1 = 1;}else{D1 = 0;}

if(x & 4){D2 = 1;}else{D2 = 0;}

if(x & 8){D3 = 1;}else{D3 = 0;}

if(x & 16){D4 = 1;}else{D4 = 0;}

if(x & 32){D5 = 1;}else{D5 = 0;}

if(x & 64){D6 = 1;}else{D6 = 0;}

if(x & 128){D7 = 1;}else{D7 = 0;}

}

//Funcion para imprimir el caracter

void LCD\_CMD(char a){

RS = 1;//Las direcciones a los caracteres

Puerto(a);

EN = 1;//Mandar el valor

\_\_delay\_us(5);

EN = 0;//Verificar si el valor de carac llego

\_\_delay\_us(5);

\_\_delay\_us(50);

}

//Funcion para mandar los datos a la LCD

void datosLCD(uint8\_t x){

RS = 0;//Modifica el contraste de la patalla

Puerto(x);

EN = 1;//Mandar el valor

\_\_delay\_us(5);

EN = 0;//Verificar si el valor de carac llego

\_\_delay\_us(5);

\_\_delay\_ms(2);

}

//Funcion para limpiar la LCD

void LCD\_Limpia(void){

datosLCD(0);

datosLCD(1);

}

//Funcion para iniciar la LCD

//En base de la presentacion de clase.

void Lcd\_Init(){

\_\_delay\_ms(20);

datosLCD (0x30);

\_\_delay\_ms(5);

datosLCD (0x30);

\_\_delay\_us(100);

datosLCD (0x30);

\_\_delay\_us(100);

datosLCD (0x38);

\_\_delay\_us(60);

datosLCD (0x08);

\_\_delay\_us(60);

datosLCD (0x01);

\_\_delay\_ms(5);

datosLCD (0x06);

\_\_delay\_us(60);

datosLCD (0x0C);

\_\_delay\_us(60);

}

//Funcion para configurar el cursor

void Lcd\_Set\_Cursor(uint8\_t x, uint8\_t y){

uint8\_t a;

if(x == 1){//Linea que se coloca arriba

a = 0x80 + y;//direccion(hexadecimal) y posicion para colocarlo en la fila

//adecuada para ire leyendo adecuadamente

datosLCD(a);

}

else if(x == 2){//Linea que se coloca abajo

a = 0xC0 + y;//direccion(hexadecimal) y posicion para colocarlo en la fila

//adecuada para ire leyendo adecuadamente

datosLCD(a);

}

}

//Funcion para mandar un string

void Lcd\_Write\_String(char \*a){

//funcion para poder imprimir texto usando el puntero

//para guardar la direccion del registro o valor de a

int i;

for(i=0;a[i]!='\0';i++)

LCD\_CMD(a[i]);

}

Librería ADC

#include <xc.h>

#include <stdint.h>

#include "ADC.h"

#define \_XTAL\_FREQ 8000000

void config\_ADC(void){

ADCON1 = 0b00000000;//Justificado a la izquierda

}

unsigned Canal\_ADC(unsigned short x){ //Fosc/8,datasheet

switch(x){

//Canal analogico

case 0:

ADCON0bits.CHS3 = 0;

ADCON0bits.CHS2 = 0;

ADCON0bits.CHS1 = 0;

ADCON0bits.CHS0 = 0;//Canal00

break;

case 1:

ADCON0bits.CHS3 = 0;

ADCON0bits.CHS2 = 0;

ADCON0bits.CHS1 = 0;

ADCON0bits.CHS0 = 1;//Canal1

break;

case 2:

ADCON0bits.CHS3 = 0;

ADCON0bits.CHS2 = 0;

ADCON0bits.CHS1 = 1;

ADCON0bits.CHS0 = 0;//Canal2

break;

case 3:

ADCON0bits.CHS3 = 0;

ADCON0bits.CHS2 = 0;

ADCON0bits.CHS1 = 1;

ADCON0bits.CHS0 = 1;//Canal3

break;

case 4:

ADCON0bits.CHS3 = 0;

ADCON0bits.CHS2 = 1;

ADCON0bits.CHS1 = 0;

ADCON0bits.CHS0 = 0;//Canal4

break;

case 5:

ADCON0bits.CHS3 = 0;

ADCON0bits.CHS2 = 1;

ADCON0bits.CHS1 = 0;

ADCON0bits.CHS0 = 1;//Canal5

break;

case 6:

ADCON0bits.CHS3 = 0;

ADCON0bits.CHS2 = 1;

ADCON0bits.CHS1 = 1;

ADCON0bits.CHS0 = 0;//Canal6

break;

case 7:

ADCON0bits.CHS3 = 0;

ADCON0bits.CHS2 = 1;

ADCON0bits.CHS1 = 1;

ADCON0bits.CHS0 = 1;//Canal7

break;

case 8:

ADCON0bits.CHS3 = 1;

ADCON0bits.CHS2 = 0;

ADCON0bits.CHS1 = 0;

ADCON0bits.CHS0 = 0;//Canal8

break;

case 9:

ADCON0bits.CHS3 = 1;

ADCON0bits.CHS2 = 0;

ADCON0bits.CHS1 = 0;

ADCON0bits.CHS0 = 1;//Canal9

break;

case 10:

ADCON0bits.CHS3 = 1;

ADCON0bits.CHS2 = 0;

ADCON0bits.CHS1 = 1;

ADCON0bits.CHS0 = 0;//Canal10

break;

case 11:

ADCON0bits.CHS3 = 1;

ADCON0bits.CHS2 = 0;

ADCON0bits.CHS1 = 1;

ADCON0bits.CHS0 = 1;//Canal11

break;

case 12:

ADCON0bits.CHS3 = 1;

ADCON0bits.CHS2 = 1;

ADCON0bits.CHS1 = 0;

ADCON0bits.CHS0 = 0;//Canal12

break;

case 13:

ADCON0bits.CHS3 = 1;

ADCON0bits.CHS2 = 1;

ADCON0bits.CHS1 = 0;

ADCON0bits.CHS0 = 1;//Canal13

break;

case 14:

ADCON0bits.CHS3 = 1;

ADCON0bits.CHS2 = 1;

ADCON0bits.CHS1 = 1;

ADCON0bits.CHS0 = 0;//CVref

break;

case 15:

ADCON0bits.CHS3 = 1;

ADCON0bits.CHS2 = 1;

ADCON0bits.CHS1 = 1;

ADCON0bits.CHS0 = 1;//Fixed Ref

break;

default:

ADCON0bits.CHS3 = 0;

ADCON0bits.CHS2 = 0;

ADCON0bits.CHS1 = 0;

ADCON0bits.CHS0 = 0;//Canal 0

break;

}

}