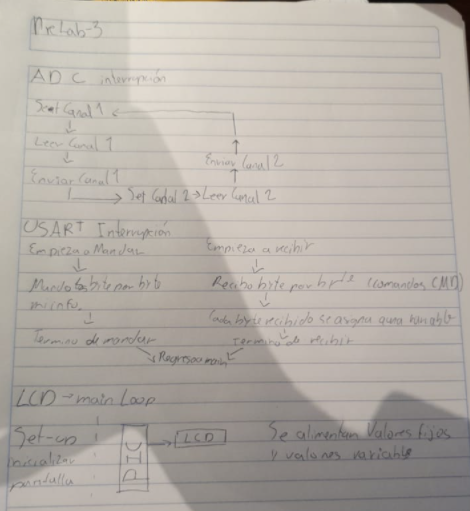
**REPORTE #1:**

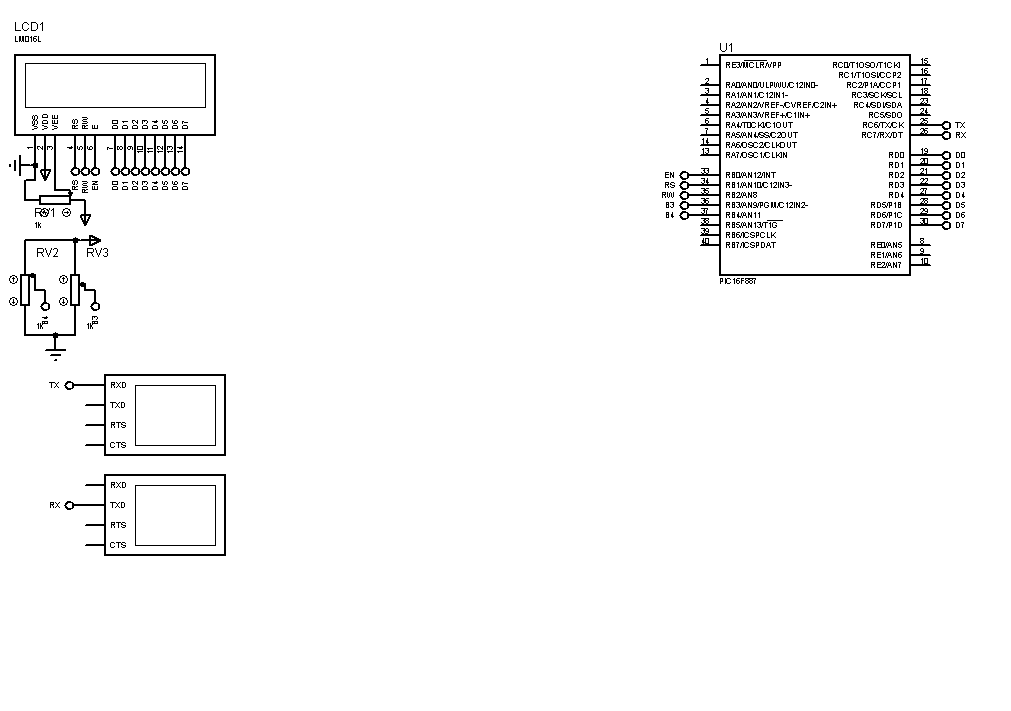
**Juego de Carreras**

Link de repositorio: <https://github.com/Cue19275/Digital2>

**Pre-lab:**



**Esquemático:**



**Código main:**

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

/\*

\* File: main.c

\* Author: Carlos Cuellar

\*

\* Fecha:

\*/

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

// Importación de Librerías

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

#include <xc.h>

#include <stdint.h>

#define \_XTAL\_FREQ 40000000

#include "Osc.h"

#include "ADCLIB.h"

#include "LCD8BIT.h"

#include "usart.h"

#include "ASCII\_NUM.h"

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

// 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 digital input, MCLR internally tied to VDD)

#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 = OFF // 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)

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

// Variables

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

uint8\_t go\_adc = 0;

uint8\_t lec\_ADC = 0;

uint8\_t toggle = 0;

uint8\_t valor\_r = 0;

uint8\_t valor\_e = 0;

uint8\_t pot1 = 0;

uint8\_t pot2 = 0;

uint8\_t terminal = 0;

uint8\_t flag\_T = 0;

uint8\_t flag\_T2 = 0;

uint8\_t estado1 = 0;

uint8\_t CONT = 0;

uint8\_t toggleTX = 0;

char CONT\_U = 0;

char CONT\_D = 0;

char CONT\_C = 0;

char POT1\_U = 0;

char POT1\_D = 0;

char POT1\_C = 0;

char POT2\_U = 0;

char POT2\_D = 0;

char POT2\_C = 0;

uint8\_t temp2 = 0;

uint8\_t temp3 = 0;

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

// Prototipos de funciones

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

void Setup(void);

void ADC\_LEER(void);

void debounce\_T(void);

void map\_cont(void);

void map\_pot1(void);

void map\_pot2(void);

void envio(void);

const char\* conver(char A, char B, char C);

const char\* conver2(char D, char E, char F);

const char\* conver3(char G, char H, char I);

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

// Vector de Interrupción

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

void \_\_interrupt() ISR(void){

if (PIR1bits.ADIF == 1){

PIR1bits.ADIF = 0;

lec\_ADC = ADRESH;

}

if (INTCONbits.TMR0IF == 1){

INTCONbits.TMR0IF = 0;

TMR0 = 236;

go\_adc++;

}

if (PIR1bits.RCIF == 1){

terminal = RCREG;

}

if (PIR1bits.TXIF == 1){

toggleTX++;

envio();

PIE1bits.TXIE = 0;

}

}

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

// Ciclo principal

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

void main(void) {

TRISD = 0x00;

initOsc(20);

Setup();

USARTconf();

Lcd\_Init();

Lcd\_Cmd(0x8A);

ADCinit(11, 20);

while (1){

//Lcd\_Clear();

Lcd\_Set\_Cursor(1,1);

Lcd\_Write\_String("S1 S2 S3");

Lcd\_Set\_Cursor(2,1);

Lcd\_Write\_String(conver(CONT\_C, CONT\_D, CONT\_U));

ADC\_LEER();

if (terminal != 43 && terminal != 13 && terminal != 45){

flag\_T = 0;

flag\_T2 = 0;

}

debounce\_T();

PORTA = POT1\_C;

map\_cont();

map\_pot1();

map\_pot2();

}

}

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

// Configuración Setup

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

void Setup(void){

//CONFIG I&0

PORTA = 0; //POT

PORTB = 0; //Pantalla

PORTC = 0; //nada

PORTD = 0; //Pantalla

ANSEL = 0;

ANSELbits.ANS0 = 1;

ANSELH = 0;

TRISA = 0; //Output

TRISB = 0b00011000; //Output Input

TRISD = 0; //Output

TRISC = 0b10000000;

INTCONbits.TMR0IF = 0;

PIR1bits.ADIF = 0;

INTCONbits.RBIF = 0;

OPTION\_REG = 0b11010111; //Pullups, prescaler, y cosas del timer

//CONFIG INTERRUPT

INTCONbits.GIE = 1;

INTCONbits.PEIE = 1;

INTCONbits.RBIE = 0;

PIE1bits.ADIE = 1;

INTCONbits.T0IE = 1;

TMR0 = 236;

PIE1bits.RCIE = 1;

PIE1bits.TXIE = 1;

}

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

// ADC\_LEER

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

void ADC\_LEER(void){

if (go\_adc > 5){

go\_adc = 0;

ADCON0bits.GO\_nDONE =1;

if (toggle == 0){

toggle = 1;

ADCinit(1, 9);

pot1 = lec\_ADC;

}

else{

toggle = 0;

ADCinit(1, 11);

pot2 = lec\_ADC;

}

}

}

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

// Debounce\_Terminal

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

void debounce\_T(void){

if (terminal == 43){

flag\_T = 1;

}

if (terminal == 13){

if (flag\_T == 1){

flag\_T = 0;

CONT++;

}

}

if (terminal == 45){

flag\_T2 = 1;

}

if(terminal ==13 && flag\_T2 == 1){

flag\_T2 = 0;

CONT--;

}

}

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

// Mapeo Contador

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

void map\_cont(void){

CONT\_C = CONT/100;

CONT\_D = (CONT-(CONT\_C\*100))/10;

CONT\_U = (CONT-(CONT\_C\*100)-(CONT\_D\*10));

CONT\_C = num\_ascii(CONT\_C);

CONT\_D = num\_ascii(CONT\_D);

CONT\_U = num\_ascii(CONT\_U);

}

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

// Conv\_Str

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

const char\* conver(char CONT\_C, char CONT\_D, char CONT\_U){

char temporal[16];

temporal[0] = CONT\_C;

temporal[1] = CONT\_D;

temporal[2] = CONT\_U;

temporal[3] = 0x20;

temporal[4] = 0x20;

temporal[5] = POT1\_C;

temporal[6] = 0x2E;

temporal[7] = POT1\_D;

temporal[8] = POT1\_U;

temporal[9] = 0x76;

temporal[10] = 0x20;

temporal[11] = POT2\_C;

temporal[12] = 0x2E;

temporal[13] = POT2\_D;

temporal[14] = POT2\_U;

temporal[15] = 0x76;

return temporal;

}

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

// Mapeo Pot1

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

void map\_pot1(void){

temp2 = pot1;

POT1\_C = ((temp2\*100)/51)/100;

POT1\_D = (((temp2\*100)/51)-(POT1\_C\*100))/10;

POT1\_U = (((temp2\*100)/51)-(POT1\_C\*100)-(POT1\_D\*10));

POT1\_C = num\_ascii(POT1\_C);

POT1\_D = num\_ascii(POT1\_D);

POT1\_U = num\_ascii(POT1\_U);

}

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

// Mapeo Pot2

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

void map\_pot2(void){

temp3 = pot2;

POT2\_C = ((temp3\*100)/51)/100;

POT2\_D = (((temp3\*100)/51)-(POT2\_C\*100))/10;

POT2\_U = (((temp3\*100)/51)-(POT2\_C\*100)-(POT2\_D\*10));

POT2\_C = num\_ascii(POT2\_C);

POT2\_D = num\_ascii(POT2\_D);

POT2\_U = num\_ascii(POT2\_U);

}

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

// Envio

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

void envio(void){

if (toggleTX == 1){ //c

TXREG = POT1\_C;

}

if (toggleTX == 2){ //.

TXREG = 0x2E;

}

if (toggleTX == 3){//d

TXREG = POT1\_D;

}

if (toggleTX == 4){ //u

TXREG = POT1\_U;

}

if (toggleTX == 5){ //,

TXREG = 0x2C;

}

if (toggleTX == 6){ //spc

TXREG = 0x20;

}

if (toggleTX == 7){ //c

TXREG = POT2\_C;

}

if (toggleTX == 8){ //.

TXREG = 0x2E;

}

if (toggleTX == 9){ //u

TXREG = POT2\_D;

}

if (toggleTX == 10){ //d

TXREG = POT2\_U;

}

if (toggleTX == 11){ //brk

TXREG = 13;

toggleTX = 0;

}

}