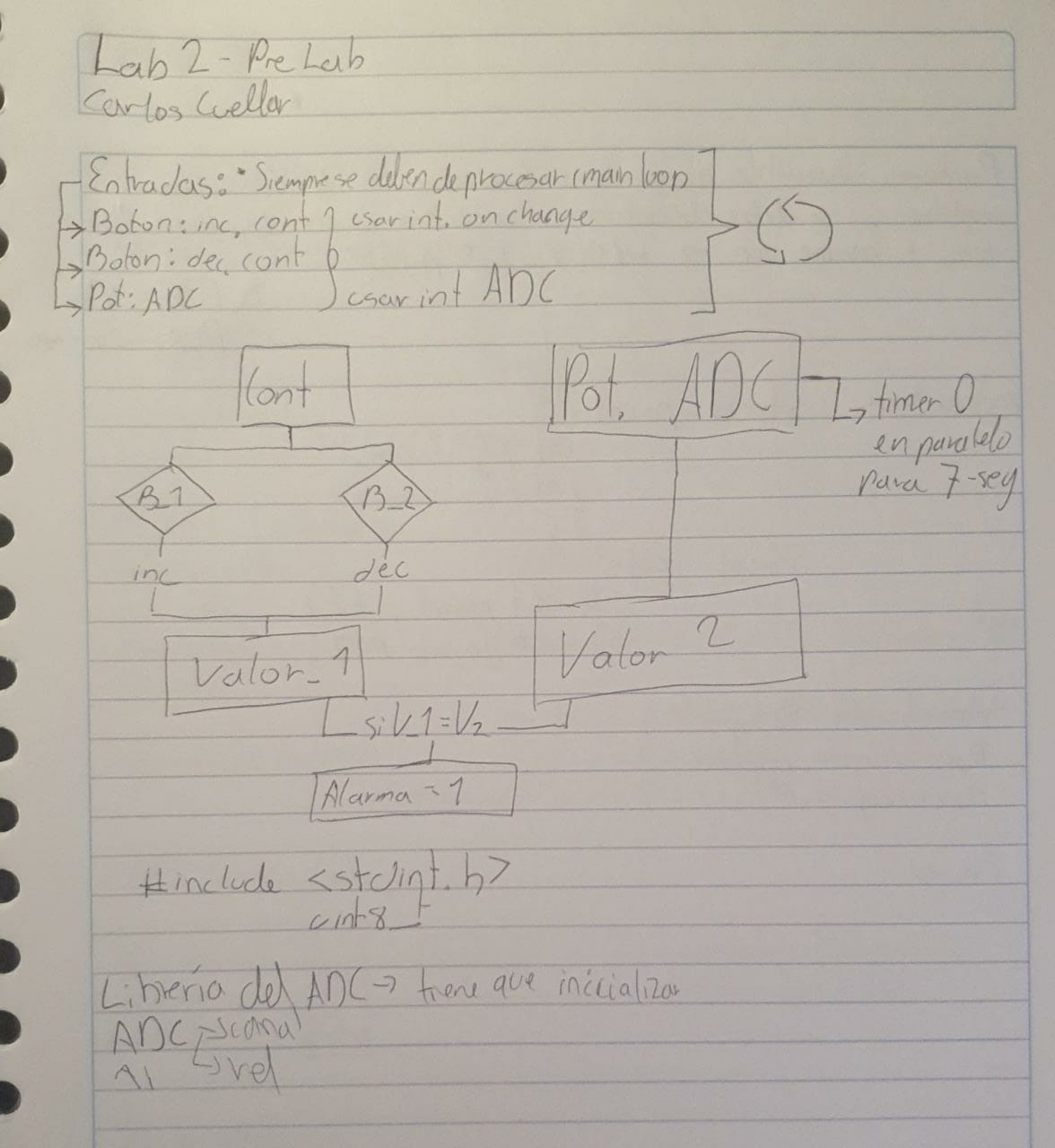
**REPORTE #1:**

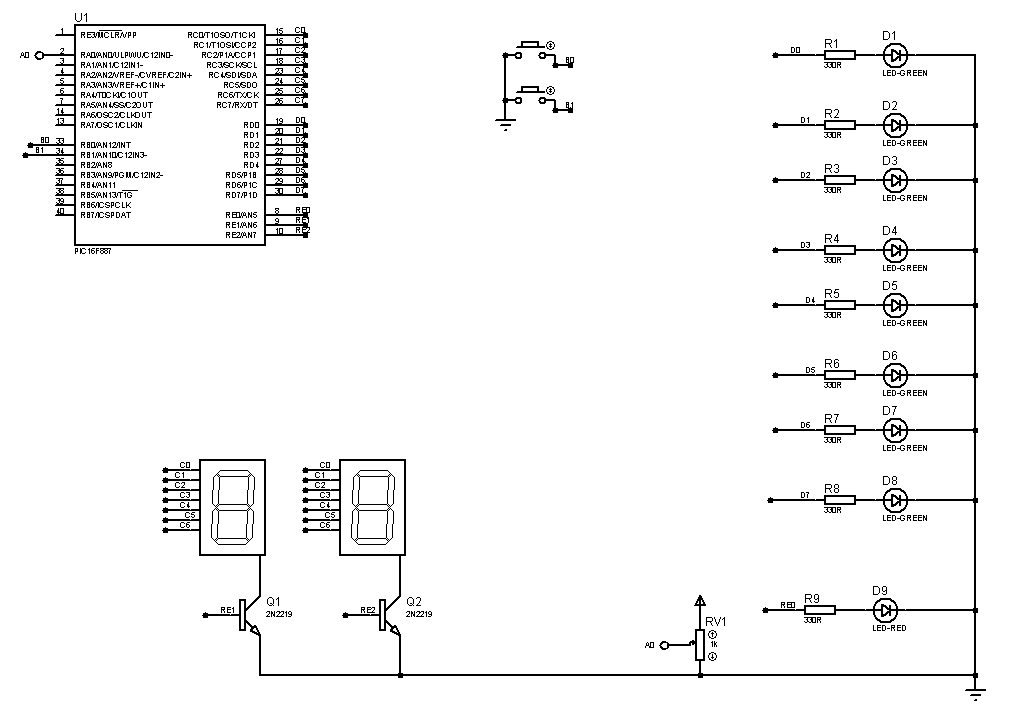
**Juego de Carreras**

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>

#include "Osc.h"

#include "ADCLIB.h"

#include "DECODESSD.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 tiempo\_mux = 0;

uint8\_t sel\_mux = 0;

uint8\_t lec\_ADC = 0;

uint8\_t go\_adc = 0;

uint8\_t v = 1;

uint8\_t can = 30;

uint8\_t estadoSalida;

uint8\_t estado;

uint8\_t estadoSalidaC2;

uint8\_t estadoC2;

uint8\_t nibbleH = 0;

uint8\_t nibbleL = 0;

#define BTN\_1 PORTBbits.RB0

#define BTN\_2 PORTBbits.RB1

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

// Prototipos de funciones

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

void Setup(void);

void ADC\_LEER(void);

void Alarma(void);

void nibbles(void);

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

// Vector de Interrupción

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

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

if (INTCONbits.TMR0IF == 1){

INTCONbits.TMR0IF = 0;

TMR0 = 236;

go\_adc++;

PORTEbits.RE1 = 0;

PORTEbits.RE2 = 0;

if (sel\_mux == 0){

sel\_mux = 1;

PORTEbits.RE1 = 1;

nibbleH = (lec\_ADC & 0b11110000) >> 4;

tabla(nibbleH);

}

else{

sel\_mux = 0;

PORTEbits.RE2 = 1;

nibbleL = lec\_ADC & 0b00001111;

tabla(nibbleL);

}

}

if (PIR1bits.ADIF == 1){

PIR1bits.ADIF = 0;

lec\_ADC = ADRESH;

}

if (INTCONbits.RBIF == 1) {

INTCONbits.RBIF = 0;

//Aquí van los debounce de los dos botones

estado = BTN\_1 ;

if (estado == 1){

estadoSalida=1;

}

if (estadoSalida==1){

if (estado == 0){

PORTD++;

estadoSalida = 0;

}

}

estadoC2 = BTN\_2 ;

if (estadoC2 == 1){

estadoSalidaC2=1;

}

if (estadoSalidaC2==1){

if (estadoC2 == 0){

PORTD--;

estadoSalidaC2 =0;

}

}

}

}

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

// Ciclo principal

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

void main(void) {

initOsc(20);

Setup();

ADCinit(1, 20);

while (1){

ADC\_LEER();

Alarma();

}

}

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

// Configuración

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

void Setup(void){

//CONFIG I&0

PORTA = 0; //POT

PORTB = 0; //BOTONES

PORTC = 0; //7SEG

PORTD = 0; //CONT

PORTE = 0; //Alarma & 2 Trans

ANSEL = 0;

ANSELbits.ANS0 = 1;

ANSELH = 0;

TRISA = 1; //Input

TRISB = 0b11111111; //Intput

TRISC = 0; //Output

TRISD = 0; //Output

TRISE = 0; //Output

INTCONbits.TMR0IF = 0;

PIR1bits.ADIF = 0;

INTCONbits.RBIF = 0;

OPTION\_REG = 0b01010111; //Pullups, prescaler, y cosas del timer

//CONFIG INTERRUPT

INTCONbits.GIE = 1;

INTCONbits.PEIE = 1;

INTCONbits.RBIE = 1;

WPUBbits.WPUB0 = 1;

WPUBbits.WPUB1 = 1;

IOCBbits.IOCB0 = 1;

IOCBbits.IOCB1 = 1;

PIE1bits.ADIE = 1;

INTCONbits.T0IE = 1;

TMR0 = 236;

//TRISBbits.TRISB0 = 0;

//PORTBbits.RB0 = 0;

}

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

// ADC\_LEER

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

void ADC\_LEER(void){

if (go\_adc > 5){

go\_adc = 0;

ADCON0bits.GO\_nDONE =1;

}

}

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

// Alarma

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

void Alarma(void){

if(lec\_ADC > PORTD){

PORTEbits.RE0 = 1;

}

else{

PORTEbits.RE0 = 0;

}

}

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

// nibbles

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

**Código Osc.c:**

#include <stdint.h>

#include <pic16f887.h>

#include "Osc.h"

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

// Función para inicializar Oscilador Interno

// Parametros: Opción de frecuencia a utilizar ver pág. 62 Manual

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

void initOsc(uint8\_t frec){

switch(frec){

case 0: // 31 KHz

OSCCONbits.IRCF0 = 0;

OSCCONbits.IRCF1 = 0;

OSCCONbits.IRCF2 = 0;

break;

case 1: // 125 KHz

OSCCONbits.IRCF0 = 1;

OSCCONbits.IRCF1 = 0;

OSCCONbits.IRCF2 = 0;

break;

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

\* Acá se debería de programar para todas las demás

\* frecuencias, colocando un caso por cada una de

\* las opciones que tiene el microcontrolador

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

case 7: // 8 MHz

OSCCONbits.IRCF0 = 1;

OSCCONbits.IRCF1 = 1;

OSCCONbits.IRCF2 = 1;

break;

default: // 4 MHz

OSCCONbits.IRCF0 = 0;

OSCCONbits.IRCF1 = 1;

OSCCONbits.IRCF2 = 1;

break;

}

//En adc configurarlo, velocidad y canal

//Tabla hacer cases: Cae 0 =

OSCCONbits.SCS = 1; // Se utilizará el reloj interno para el sistema

}

**Código Osc.h:**

// This is a guard condition so that contents of this file are not included

// more than once.

#ifndef OSC\_H

#define OSC\_H

#pragma config FOSC = INTRC\_NOCLKOUT// Oscillator Selection bits (INTOSCIO oscillator: I/O function on RA6/OSC2/CLKOUT pin, I/O function on RA7/OSC1/CLKIN)

#include <xc.h> // include processor files - each processor file is guarded.

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

// Prototipo de la función para inicializar Oscilador Interno

// Parametros: Opción de frecuencia a utilizar ver pág. 62 Manual

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

void initOsc(uint8\_t frec);

//ADC

//TABLA

#endif /\* XC\_HEADER\_TEMPLATE\_H \*/

**Código ADCLIB.h:**

/\*

\* File: ADCLIB.h

\* Author: Carlo

\*

\* Created on 2 de febrero de 2021, 12:23

\*/

#ifndef ADCLIB\_H

#define ADCLIB\_H

#include <stdint.h>

#include <xc.h>

void ADCinit(uint8\_t vel, uint8\_t chan);

#endif /\* ADCLIB\_H \*/

**Código ADCLIB.c:**

/\*

\* File: ADCLIB.c

\* Author: Carlo

\*

\* Created on 2 de febrero de 2021, 12:22

\*/

#include "ADCLIB.h"

void ADCinit(uint8\_t vel, uint8\_t chan){

switch(vel){

case 1: //Este es el que usamos siempre

ADCON0bits.ADCS0 = 1;

ADCON0bits.ADCS1 = 0;

break;

case 2:

ADCON0bits.ADCS0 = 0;

ADCON0bits.ADCS1 = 1;

break;

case 3:

ADCON0bits.ADCS0 = 1;

ADCON0bits.ADCS1 = 1;

break;

default:

ADCON0bits.ADCS0 = 0;

ADCON0bits.ADCS1 = 0;

break;

}

switch(chan){

default:

ADCON0bits.CHS0 = 0;

ADCON0bits.CHS1 = 0;

ADCON0bits.CHS2 = 0;

ADCON0bits.CHS3 = 0;

break;

case 1:

ADCON0bits.CHS0 = 1;

ADCON0bits.CHS1 = 0;

ADCON0bits.CHS2 = 0;

ADCON0bits.CHS3 = 0;

break;

case 2:

ADCON0bits.CHS0 = 0;

ADCON0bits.CHS1 = 1;

ADCON0bits.CHS2 = 0;

ADCON0bits.CHS3 = 0;

break;

case 3:

ADCON0bits.CHS0 = 1;

ADCON0bits.CHS1 = 1;

ADCON0bits.CHS2 = 0;

ADCON0bits.CHS3 = 0;

break;

case 4:

ADCON0bits.CHS0 = 0;

ADCON0bits.CHS1 = 0;

ADCON0bits.CHS2 = 1;

ADCON0bits.CHS3 = 0;

break;

case 5:

ADCON0bits.CHS0 = 1;

ADCON0bits.CHS1 = 0;

ADCON0bits.CHS2 = 1;

ADCON0bits.CHS3 = 0;

break;

case 6:

ADCON0bits.CHS0 = 0;

ADCON0bits.CHS1 = 1;

ADCON0bits.CHS2 = 1;

ADCON0bits.CHS3 = 0;

break;

case 7:

ADCON0bits.CHS0 = 1;

ADCON0bits.CHS1 = 1;

ADCON0bits.CHS2 = 1;

ADCON0bits.CHS3 = 0;

break;

case 8:

ADCON0bits.CHS0 = 0;

ADCON0bits.CHS1 = 0;

ADCON0bits.CHS2 = 0;

ADCON0bits.CHS3 = 1;

break;

case 9:

ADCON0bits.CHS0 = 1;

ADCON0bits.CHS1 = 0;

ADCON0bits.CHS2 = 0;

ADCON0bits.CHS3 = 1;

break;

case 10:

ADCON0bits.CHS0 = 0;

ADCON0bits.CHS1 = 1;

ADCON0bits.CHS2 = 0;

ADCON0bits.CHS3 = 1;

break;

case 11:

ADCON0bits.CHS0 = 1;

ADCON0bits.CHS1 = 1;

ADCON0bits.CHS2 = 0;

ADCON0bits.CHS3 = 1;

break;

case 12:

ADCON0bits.CHS0 = 0;

ADCON0bits.CHS1 = 0;

ADCON0bits.CHS2 = 1;

ADCON0bits.CHS3 = 1;

break;

case 13:

ADCON0bits.CHS0 = 1;

ADCON0bits.CHS1 = 0;

ADCON0bits.CHS2 = 1;

ADCON0bits.CHS3 = 1;

break;

}

ADCON0bits.GO\_nDONE =0;

ADCON0bits.ADON = 1;

ADCON1 = 0;

}

**Código DECODESSD.h:**

#ifndef DECODESSD\_H

#define DECODESSD\_H

#include <stdint.h>

#include <xc.h>

void tabla(uint8\_t dec);

#endif /\* DECODESSD\_H \*/

**Código DECODESSD.c:**

#include <xc.h>

#include <stdint.h>

#include <pic16f887.h>

#include "DECODESSD.h"

void tabla(uint8\_t dec) {

switch (dec){

case 0:

PORTC = 0b00111111;

break;

case 1:

PORTC = 0b00000110;

break;

case 2:

PORTC = 0b001011011;

break;

case 3:

PORTC = 0b001001111;

break;

case 4:

PORTC = 0b01100110;

break;

case 5:

PORTC = 0b01101101;

break;

case 6:

PORTC = 0b01111101;

break;

case 7:

PORTC = 0b00000111;

break;

case 8:

PORTC = 0b01111111;

break;

case 9:

PORTC = 0b01100111;

break;

case 10:

PORTC = 0b01110111;

break;

case 11:

PORTC = 0b01111100;

break;

case 12:

PORTC = 0b00111001;

break;

case 13:

PORTC = 0b01011110;

break;

case 14:

PORTC = 0b01111001;

break;

case 15:

PORTC = 0b01110001;

break;

}

}