Universidad Del Valle De Guatemala 08/02/2021

Digital 2 Helder Ovalle Barrios

Sección: 18349

**Labrotorio #2**

**Link de github:**

[**https://github.com/Helder1121/Labsdigitaldos/tree/main/lab2**](https://github.com/Helder1121/Labsdigitaldos/tree/main/lab2)

**Progra comentada:**

**Principal:**

/\*

\* File: lab2.c

\* Author: Helder Ovalle

\*

\* Created on 31 de enero de 2021, 10:24 PM

\*/

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

// Importación de librerías

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

#include <xc.h>

//#include <stdint.h>

#include "ADC.h"

#include "Multiplexado.h"

#define \_XTAL\_FREQ 4000000 //Delay

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

// 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 = 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

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

int Contador = 0; //Incremento

int conta = 0;

int banders = 0;

//valor asignando

unsigned int valor\_adc;

//Nibbles

unsigned int val\_high;

unsigned int val\_low;

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

// Prototipos de funciones

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

void Config\_P(void);

void Config\_Int(void);

void Banderas(void);

void Sep\_Nb(void);

void Display(void);

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

// Interrupcion

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

void \_\_interrupt() isr(void){

//Interrupccion puerto B

if (INTCONbits.RBIF){

INTCONbits.RBIF = 0;

if (PORTBbits.RB0 == 1){

//Utilice delay en vez de antirebote

\_\_delay\_ms(200);

}

//Delay para el puerto RB0

if (PORTBbits.RB0 == 0){

Contador++; //Incrementa el contador

PORTD = Contador;

\_\_delay\_ms(200);

}

if (PORTBbits.RB1 == 1){

\_\_delay\_ms(200);

}

if (PORTBbits.RB1 == 0){

Contador--; //Decrementa el contador

PORTD = Contador;

\_\_delay\_ms(200);

}

}

if (PIR1bits.ADIF) {

PIR1bits.ADIF = 0;

ADC\_READ(8); //Canal 8

//Comienzo de la conversion del ADC

\_\_delay\_ms(2);

ADCON0bits.GO = 1;

while (ADCON0bits.GO !=0){

//Mientras Go sea diferente de 0 se deplegara en Adresh

//el valor del ADC

valor\_adc = ADRESH;

Display();

}

}

//Configuracion del TMR0

if (TMR0IF){

TMR0IF = 0;

TMR0 = 4; //0.5 MS

conta++;

}

}

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

// Ciclo principal

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

void main(void) {

Config\_P();

Config\_Int();

ADC\_INIT();

ADC\_INTERRUPCION();

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

// Loop principal

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

while (1) {

if (conta >= 1){

conta = 0;

//Limpiara la var si en dado caso es >=1

Banderas();

}

//Función de la alarma

Sep\_Nb();//Separacion de los nibbles

if (ADRESH >= PORTD) { //Se prendera el led cuando se mayor

PORTEbits.RE0 = 1;

} else { // de lo contrario estara apagado

PORTEbits.RE0 = 0;

}

}

//return;

}

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

// Configuración

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

void Config\_P(void){

TRISD = 0;

TRISB = 0b00000111;

TRISA = 0;

TRISE = 0;

//Todos estanran como salida menos loas tres de RB0.RB1 Y RB2

ANSEL = 0;

ANSELH = 0b00000000;

//Unicamente el canal8

PORTA = 0;

PORTB = 0;

PORTC = 0;

PORTD = 0;

PORTE = 0;

//Steo los puertos

}

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

// Funciones

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

void Config\_Int(void){

TMR0 = 4;

OPTION\_REG = 0b10000001;//TMR0 1:4

INTCON = 0b10101001; //Habilitos los GIE

IOCB = 0b00000011;

}

void Display(void){

PORTE = 0;//steo el puerto E

if (banders == 0){

Multiplexado(val\_low);

PORTEbits.RE1 = 1;

} else if (banders == 1){

Multiplexado(val\_high);

PORTEbits.RE2 = 1;

//Se asignan el valor a cada uno de los puertoE

}

}

void Banderas(void){

if (banders == 1){

banders = 0;

} else if (banders == 0){

banders = 1;

}

//Apagamos y encedemos las banderas para el multiplexado

}

void Sep\_Nb(void){

val\_low = (0b11110000 & valor\_adc) >> 4;

val\_high = (0b00001111 & valor\_adc) << 4;

//https://www.geeksforgeeks.org/swap-two-nibbles-byte/ utilice esta pagina

//de ejemplo para realizar la separacion de nibbles

}

**Librerías:**

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

// more than once.

#ifndef \_\_ADC\_H\_

#define \_\_ADC\_H\_

#include <stdint.h>

#define \_XTAL\_FREQ 4000000

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

unsigned ADC\_READ(unsigned short canal);

void ADC\_INTERRUPCION();

void ADC\_INIT();

void ADC\_INIT(){

ADCON1 = 0b00000000; //Justificado a la izquierda

}

unsigned ADC\_READ(unsigned short canal){

switch (canal){//canal 8

case 8:

ADCON0 = 0b01100001;

break;

}

}

void ADC\_INTERRUPCION(){

PIE1bits.ADIE = 1;

PIR1bits.ADIF = 1;

}

#endif /\* \_\_ADC\_H\_ \*/

#ifndef \_\_Multiplexado\_H\_

#define \_\_Multiplexado\_H\_

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

void Multiplexado(uint8\_t num);

void Multiplexado(uint8\_t num){

TRISC = 0;

PORTC = 0;

switch (num){

//Menos significativo

case 0:

PORTC = 0x3F;//0

break;

case 1:

PORTC = 0x06;//1

break;

case 2:

PORTC = 0x5B;//2

break;

case 3:

PORTC = 0x4F;//3

break;

case 4:

PORTC = 0x66;//4

break;

case 5:

PORTC = 0x6D;//5

break;

case 6:

PORTC = 0x7D;//6

break;

case 7:

PORTC = 0x07;//7

break;

case 8:

PORTC = 0x7F;//8

break;

case 9:

PORTC = 0x6F;//9

break;

case 10:

PORTC = 0x77;//A

break;

case 11:

PORTC = 0x1F;//B

break;

case 12:

PORTC = 0x4E;//C

break;

case 13:

PORTC = 0x3D;//D

break;

case 14:

PORTC = 0x4F;//E

break;

case 15:

PORTC = 0x47;//F

break;

//Siguiente display

case 128:

PORTC = 0x06;//1

break;

case 64:

PORTC = 0x5B;//2

break;

case 192:

PORTC = 0x4F;//3

break;

case 32:

PORTC = 0x66;//4

break;

case 160:

PORTC = 0x6D;//5

break;

case 96:

PORTC = 0x7D;//6

break;

case 224:

PORTC = 0x07;//7

break;

case 16:

PORTC = 0x7F;//8

break;

case 144:

PORTC = 0x6F;//9

break;

case 80:

PORTC = 0x77;//A

break;

case 208:

PORTC = 0x1F;//B

break;

case 48:

PORTC = 0x4E;//C

break;

case 176:

PORTC = 0x3D;//D

break;

case 112:

PORTC = 0x4F;//E

break;

case 240:

PORTC = 0x47;//F

break;

default:

PORTC = 0xFF;

break;

}

}

#endif /\* \_\_Multiplexado\_H\_ \*/

**Diagrama de flujo:**

Librerias

Botón de incremente y decremento

Verificar botones ejecutar el adc y guardarlo en una variable declarada

Configurar los puertos del adc

Encender alarmar si el adc es mayor al del contador

Para prender un display y apagar el otro, se realizará lo contrario a lo interrupción desplegándolo en el puerto asignado

Para la multiplexación usaremos el timer0 cambiando bit x interrupción

Configurar puertos de entrada y salida

Inicio