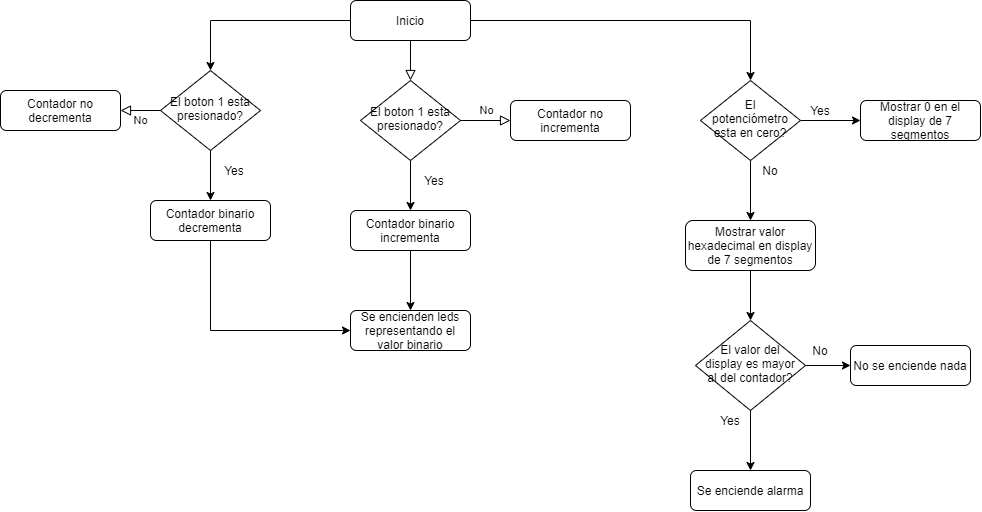
Laboratorio No. 2

Interrupciones y uso de librerías

**Link de GitHub:** <https://github.com/mon19379/DIGITAL2.git>

**Pseudocódigo:**



**Descripción:**

El laboratorio consiste en realizar un contador de 8 bits utilizando el interrupt on change en botones para incrementar y decrementar el valor del contador. También consistía en implementar en Analog to Digital converter(ADC) y display de 7 segmentos para mostrar el valor de un potenciómetro en hexadecimal. El contador era una referencia respecto al display ya que una vez el valor del display fuera mayor al del contador, se enciende una alarma.

**Código:**

**Main**:

/\*FRANCISCO MONTÚFAR

\*CARNET 19379

\*ELECTRÓNICA DIGITAL 2

\*LABORATORIO #2

\*LIBRERÍAS E INTERRUPCIONES

\*

\*/

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

// Importación de librerias

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

#include <xc.h>

#include <stdint.h>

#include "Osc.h"

#include "adc.h"

#include "SIETESEG.h"

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

// Palabra de configuración

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

// CONFIG1

#pragma config FOSC = INTRC\_NOCLKOUT // 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 B1 = 0;

uint8\_t B2 = 0;

uint8\_t NH = 0;

uint8\_t NL = 0;

uint8\_t pot = 0;

uint8\_t FLAG = 0;

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

// Prototipos de funciones

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

void Setup(void);

void split(void);

void displays(void);

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

//Interrupción

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

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

if (T0IF == 1){ //SE REVISA LA BANDERA DE INTERRUPCION DEL TIMER0

TMR0 = 236; //SE CARGA UN VALOR AL TIMER CERO PARA QUE SEA DE 20ms

T0IF = 0; //SE APAGA LA BANDERA DE INTERRUPCION

displays(); //SE LLAMA LA RUTINA DE MUTIPLEXACION

}

if (ADIF == 1){ //SE REVISA LA BANDERA DE INTERRUPCION DEL ADC

pot = ADRESH; //SE INGRESA EL VALOR DE ADRESH A UNA VARIABLE

ADIF = 0; //SE APAGA LA BANDERA DE INTERRUPCION

ADCON0bits.GO\_nDONE = 1;

}

if (RBIF == 1){ //SE REVISA LA BANDERA DE INTERRUPCION DEL PUERTO B

if (PORTBbits.RB0 == 0){ //ANTIREBOTE, SI NO SE PRESIONA EL BOTON

B1 = 1; // SE ENCIENDE LA BANDERA DEL BOTON DE INCREMENTO

}

else{

if (B1 == 1 && PORTBbits.RB0 == 1 ){ //SE PRESIONA EL BOTON

B1 = 0; //SE APAGA LA BANDERA DE BOTON DE INCREMENTO

PORTD ++;// SE INCREMENTA EL PUERTOD

}

}

if (PORTBbits.RB1 == 0){ //ANTIREBOTE, SI NO SE PRESIONA EL BOTON

B2 = 1; // SE ENCIENDE LA BANDERA DEL BOTON DE DECREMENTO

}

else{

if (B2 == 1 && PORTBbits.RB1 == 1 ){ //SE PRESIONA EL BOTON

B2 = 0; //SE APAGA LA BANDERA DE BOTON DE DECREMENTO

PORTD --;// SE DECREMENTA UN EL PUERTOD

}

}

INTCONbits.RBIF = 0; //SE APAGA LA BANDERA DE INTERRUPION DEL PUERTO B

}

}

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

//Ciclo pincipal

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

void main(void) {

Setup();

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

// Loop principal

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

while(1){

if(pot > PORTD){ //RUTINA DE ALARMA EN DONDE SE COMPARAN EL ADC Y

PORTAbits.RA1 = 1;//EL CONTADOR

}

else{

PORTAbits.RA1 = 0;

}

}

}

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

//Configuracion

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

void Setup(void) {

initOsc(10); //SE LLAMA LA CONFIG DEL OSCILADOR

configADC(2,0); //SE LLAMA LA CONFIG DEL ADC

ANSEL = 0; //SE LIMPIA EL ANSEL

ANSEL = 0b00000001; // ENTRADAS DIGITALES Y BIT 0 ANALÓGICA

ANSELH = 0;

PORTA = 0; //PUERTO A EN 0

PORTB = 0; //PUERTO B EN 0

PORTC = 0; //PUERTO C EN 0

PORTD = 0; //PUERTO D EN 0

PORTE = 0; //PUERTO E EN 0

//PINES RA0 Y RA2 COMO ENTRADAS, LOS DEMAS COMO SALIDAS

TRISA = 0b00000101;

TRISB = 0b00000011; //PUERTO B

TRISC = 0; //PUERTO C SALIDAS

TRISD = 0; //PUERTO D SALIDAS

TRISE = 0; //PUERTO E SALIDAS

OPTION\_REG = 0b00000111; //SE APAGAN LAS PULLUPS DEL PUERTO B

INTCONbits.GIE = 1; //SE HABILITAN LAS INTERRUPCIONES GLOBALES

INTCONbits.T0IE = 1; //SE HABILITA LA INTERRUPCION DEL TIMER0

INTCONbits.PEIE = 1; //SE HABILITAN LAS INTERRUPCIONES PERIFERICAS

PIE1bits.ADIE = 1; //SE HABILITA LA INTERRUPCION DEL ADC

INTCONbits.T0IF = 0; // SE LIMPIA LA BANDERA DE INTERRUPCION DEL TIMER 0

PIR1bits.ADIF = 0; //SE LIMPIOA LA BANDERA DE INTERRUPCION DEL ADC

INTCONbits.RBIE = 1; //SE HABILITA LA INTERRUPCION DEL PUERTO B

INTCONbits.RBIF = 0; //SE LIMPIA LA BANDERA DEL INTERRUPCION DEL PUERTO B

IOCB = 3; //SE HABILITA EL INTERRUPT ON CHANGE

}

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

// Subrutinas

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

void displays(void){

PORTAbits.RA3 = 0;

PORTAbits.RA4 = 0; //SE LIMPIA EL PIN DE LOS TRANSISTORES

if (FLAG == 0){ //MULTIPLEXACIÓN

NL = pot & 0b00001111; //SE HACE UN AND

PORTAbits.RA3 = 1; //SE ENCIENTE EL TRANSISTOR DE ESE DISPLAY

display(NL); //SE LLAMA LA TABLA

FLAG = 1; //TOGGLE

}

else{

NH = pot; //SE IGUALA EL NH A LA VARIABLE DEL ADC

NH = NH & 0b11110000; //SE HACE UN AND PARA MANTENER LOS MSB

NH = NH>>4; //SE HACE EL SHIFT A LA DERECHA

PORTAbits.RA4 = 1; //SE ENCIENDE EL TRANSISTOR DE ESE DISPLAY

display(NH); //SE LLAMA LA TABLA

FLAG =0; //TOGGLE

}

}

**ADC.H**

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

// more than once.

#ifndef ADC\_H

#define ADC\_H

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

#include <stdint.h>

void configADC(uint8\_t fosc, uint8\_t chan);

#endif /\* ADC\_H\*/

**ADC.C**

**#include <pic16f887.h>**

**#include <xc.h>**

**#include "adc.h"**

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

**// CONFIGURACION DEL ADC**

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

**void configADC(uint8\_t fosc, uint8\_t chan){**

**switch (fosc) {**

**case 0:**

**ADCON0bits.ADCS = 0b00;**

**break;**

**case 1:**

**ADCON0bits.ADCS = 0b01;**

**break;**

**case 2:**

**ADCON0bits.ADCS = 0b10;**

**break;**

**case 3:**

**ADCON0bits.ADCS = 0b11;**

**break;**

**default:**

**ADCON0bits.ADCS = 0b00;**

**break;**

**}**

**switch (chan) {**

**case 0:**

**ADCON0bits.CHS = 0b0000;**

**break;**

**case 1:**

**ADCON0bits.CHS = 0b0001;**

**break;**

**case 2:**

**ADCON0bits.CHS = 0b0010;**

**break;**

**case 3:**

**ADCON0bits.CHS = 0b0011;**

**break;**

**case 4:**

**ADCON0bits.CHS = 0b0100;**

**break;**

**case 5:**

**ADCON0bits.CHS = 0b0101;**

**break;**

**case 6:**

**ADCON0bits.CHS = 0b0110;**

**break;**

**case 7:**

**ADCON0bits.CHS = 0b0111;**

**break;**

**case 8:**

**ADCON0bits.CHS = 0b1000;**

**break;**

**case 9:**

**ADCON0bits.CHS = 0b1001;**

**break;**

**case 10:**

**ADCON0bits.CHS = 0b1010;**

**break;**

**case 11:**

**ADCON0bits.CHS = 0b1011;**

**break;**

**case 12:**

**ADCON0bits.CHS = 0b1100;**

**break;**

**case 13:**

**ADCON0bits.CHS = 0b1101;**

**break;**

**case 14:**

**ADCON0bits.CHS = 0b1110;**

**break;**

**case 15:**

**ADCON0bits.CHS = 0b1111;**

**break;**

**default:**

**ADCON0bits.CHS = 0b0000;**

**break;**

**}**

**ADCON0bits.GO\_nDONE = 1;**

**ADCON0bits.ADON = 1;**

**ADCON1 = 0;**

**}**

**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

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

#include <stdint.h>

void initOsc(uint8\_t IRCF);

#endif /\* Osc\_H \*/

**OSC.C**

**#include <pic16f887.h>**

**#include <xc.h>**

**#include "Osc.h"**

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

**//Inicialización del oscilador interno pg. 62**

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

**void initOsc(uint8\_t IRCF){**

**switch (IRCF){**

**case 0: //OSCILADOR DE 31 kHz**

**OSCCONbits.IRCF2 = 0;**

**OSCCONbits.IRCF1 = 0;**

**OSCCONbits.IRCF0 = 0;**

**break;**

**case 1: //OSCILADOR DE 125 kHz**

**OSCCONbits.IRCF2 = 0;**

**OSCCONbits.IRCF1 = 0;**

**OSCCONbits.IRCF0 = 1;**

**break;**

**case 2: //OSCILADOR DE 250 kHz**

**OSCCONbits.IRCF2 = 0;**

**OSCCONbits.IRCF1 = 1;**

**OSCCONbits.IRCF0 = 0;**

**break;**

**case 3: //OSCILADOR DE 500kHz**

**OSCCONbits.IRCF2 = 0;**

**OSCCONbits.IRCF1 = 1;**

**OSCCONbits.IRCF0 = 1;**

**break;**

**case 4: //OSCILADOR DE 1MHz**

**OSCCONbits.IRCF2 = 1;**

**OSCCONbits.IRCF1 = 0;**

**OSCCONbits.IRCF0 = 0;**

**break;**

**case 5: //OSCILADOR DE 2MHz**

**OSCCONbits.IRCF2 = 1;**

**OSCCONbits.IRCF1 = 0;**

**OSCCONbits.IRCF0 = 1;**

**break;**

**case 6: //OSCILADOR DE 4MHz**

**OSCCONbits.IRCF2 = 1;**

**OSCCONbits.IRCF1 = 1;**

**OSCCONbits.IRCF0 = 0;**

**break;**

**case 7: //OSCILADOR DE 8MHz**

**OSCCONbits.IRCF2 = 1;**

**OSCCONbits.IRCF1 = 1;**

**OSCCONbits.IRCF0 = 1;**

**break;**

**default: //OSCILADOR DE 4MHz**

**OSCCONbits.IRCF2 = 1;**

**OSCCONbits.IRCF1 = 0;**

**OSCCONbits.IRCF0 = 0;**

**break;**

**}**

**OSCCONbits.SCS = 1; //SE VA A USAR EL OSCILADOR INTERNO**

**}**

**SEGMENT.H**

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

// more than once.

#ifndef SIETESEG\_H

#define SIETESEG\_H

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

#include <stdint.h>

void display(uint8\_t segment);

#endif /\* SIETESEG\_H \*/

**SEGMENT.C**

**#include <pic16f887.h>**

**#include <xc.h>**

**#include "SIETESEG.h"**

**void display(uint8\_t segment){**

**switch (segment){**

**case 0: //0**

**PORTCbits.RC0 = 0;**

**PORTCbits.RC1 = 0;**

**PORTCbits.RC2 = 0;**

**PORTCbits.RC3 = 0;**

**PORTCbits.RC4 = 0;**

**PORTCbits.RC5 = 0;**

**PORTCbits.RC6 = 1;**

**PORTCbits.RC7 = 1;**

**break;**

**case 1: //1**

**PORTCbits.RC0 = 1;**

**PORTCbits.RC1 = 0;**

**PORTCbits.RC2 = 0;**

**PORTCbits.RC3 = 1;**

**PORTCbits.RC4 = 1;**

**PORTCbits.RC5 = 1;**

**PORTCbits.RC6 = 1;**

**PORTCbits.RC7 = 1;**

**break;**

**case 2: //2**

**PORTCbits.RC0 = 0;**

**PORTCbits.RC1 = 0;**

**PORTCbits.RC2 = 1;**

**PORTCbits.RC3 = 0;**

**PORTCbits.RC4 = 0;**

**PORTCbits.RC5 = 1;**

**PORTCbits.RC6 = 0;**

**PORTCbits.RC7 = 1;**

**break;**

**case 3: //3**

**PORTCbits.RC0 = 0;**

**PORTCbits.RC1 = 0;**

**PORTCbits.RC2 = 0;**

**PORTCbits.RC3 = 0;**

**PORTCbits.RC4 = 1;**

**PORTCbits.RC5 = 1;**

**PORTCbits.RC6 = 0;**

**PORTCbits.RC7 = 1;**

**break;**

**case 4: //4**

**PORTCbits.RC0 = 1;**

**PORTCbits.RC1 = 0;**

**PORTCbits.RC2 = 0;**

**PORTCbits.RC3 = 1;**

**PORTCbits.RC4 = 1;**

**PORTCbits.RC5 = 0;**

**PORTCbits.RC6 = 0;**

**PORTCbits.RC7 = 1;**

**break;**

**case 5: //5**

**PORTCbits.RC0 = 0;**

**PORTCbits.RC1 = 1;**

**PORTCbits.RC2 = 0;**

**PORTCbits.RC3 = 0;**

**PORTCbits.RC4 = 1;**

**PORTCbits.RC5 = 0;**

**PORTCbits.RC6 = 0;**

**PORTCbits.RC7 = 1;**

**break;**

**case 6: //6**

**PORTCbits.RC0 = 0;**

**PORTCbits.RC1 = 1;**

**PORTCbits.RC2 = 0;**

**PORTCbits.RC3 = 0;**

**PORTCbits.RC4 = 0;**

**PORTCbits.RC5 = 0;**

**PORTCbits.RC6 = 0;**

**PORTCbits.RC7 = 1;**

**break;**

**case 7: // 7**

**PORTCbits.RC0 = 0;**

**PORTCbits.RC1 = 0;**

**PORTCbits.RC2 = 0;**

**PORTCbits.RC3 = 1;**

**PORTCbits.RC4 = 1;**

**PORTCbits.RC5 = 1;**

**PORTCbits.RC6 = 1;**

**PORTCbits.RC7 = 1;**

**break;**

**case 8: //8**

**PORTCbits.RC0 = 0;**

**PORTCbits.RC1 = 0;**

**PORTCbits.RC2 = 0;**

**PORTCbits.RC3 = 0;**

**PORTCbits.RC4 = 0;**

**PORTCbits.RC5 = 0;**

**PORTCbits.RC6 = 0;**

**PORTCbits.RC7 = 1;**

**break;**

**case 9: //9**

**PORTCbits.RC0 = 0;**

**PORTCbits.RC1 = 0;**

**PORTCbits.RC2 = 0;**

**PORTCbits.RC3 = 0;**

**PORTCbits.RC4 = 1;**

**PORTCbits.RC5 = 0;**

**PORTCbits.RC6 = 0;**

**PORTCbits.RC7 = 1;**

**break;**

**case 10: //A**

**PORTCbits.RC0 = 0;**

**PORTCbits.RC1 = 0;**

**PORTCbits.RC2 = 0;**

**PORTCbits.RC3 = 1;**

**PORTCbits.RC4 = 0;**

**PORTCbits.RC5 = 0;**

**PORTCbits.RC6 = 0;**

**PORTCbits.RC7 = 1;**

**break;**

**case 11: //b**

**PORTCbits.RC0 = 1;**

**PORTCbits.RC1 = 1;**

**PORTCbits.RC2 = 0;**

**PORTCbits.RC3 = 0;**

**PORTCbits.RC4 = 0;**

**PORTCbits.RC5 = 0;**

**PORTCbits.RC6 = 0;**

**PORTCbits.RC7 = 1;**

**break;**

**case 12: //C**

**PORTCbits.RC0 = 0;**

**PORTCbits.RC1 = 1;**

**PORTCbits.RC2 = 1;**

**PORTCbits.RC3 = 0;**

**PORTCbits.RC4 = 0;**

**PORTCbits.RC5 = 0;**

**PORTCbits.RC6 = 1;**

**PORTCbits.RC7 = 1;**

**break;**

**case 13: //d**

**PORTCbits.RC0 = 1;**

**PORTCbits.RC1 = 0;**

**PORTCbits.RC2 = 0;**

**PORTCbits.RC3 = 0;**

**PORTCbits.RC4 = 0;**

**PORTCbits.RC5 = 1;**

**PORTCbits.RC6 = 0;**

**PORTCbits.RC7 = 1;**

**break;**

**case 14: //E**

**PORTCbits.RC0 = 0;**

**PORTCbits.RC1 = 1;**

**PORTCbits.RC2 = 1;**

**PORTCbits.RC3 = 0;**

**PORTCbits.RC4 = 0;**

**PORTCbits.RC5 = 0;**

**PORTCbits.RC6 = 0;**

**PORTCbits.RC7 = 1;**

**break;**

**case 15: //F**

**PORTCbits.RC0 = 0;**

**PORTCbits.RC1 = 1;**

**PORTCbits.RC2 = 1;**

**PORTCbits.RC3 = 1;**

**PORTCbits.RC4 = 0;**

**PORTCbits.RC5 = 0;**

**PORTCbits.RC6 = 0;**

**PORTCbits.RC7 = 1;**

**break;**

**default:**

**PORTCbits.RC0 = 0;**

**PORTCbits.RC1 = 0;**

**PORTCbits.RC2 = 0;**

**PORTCbits.RC3 = 0;**

**PORTCbits.RC4 = 0;**

**PORTCbits.RC5 = 0;**

**PORTCbits.RC6 = 1;**

**PORTCbits.RC7 = 1;**

**break;**

**}**

**}**