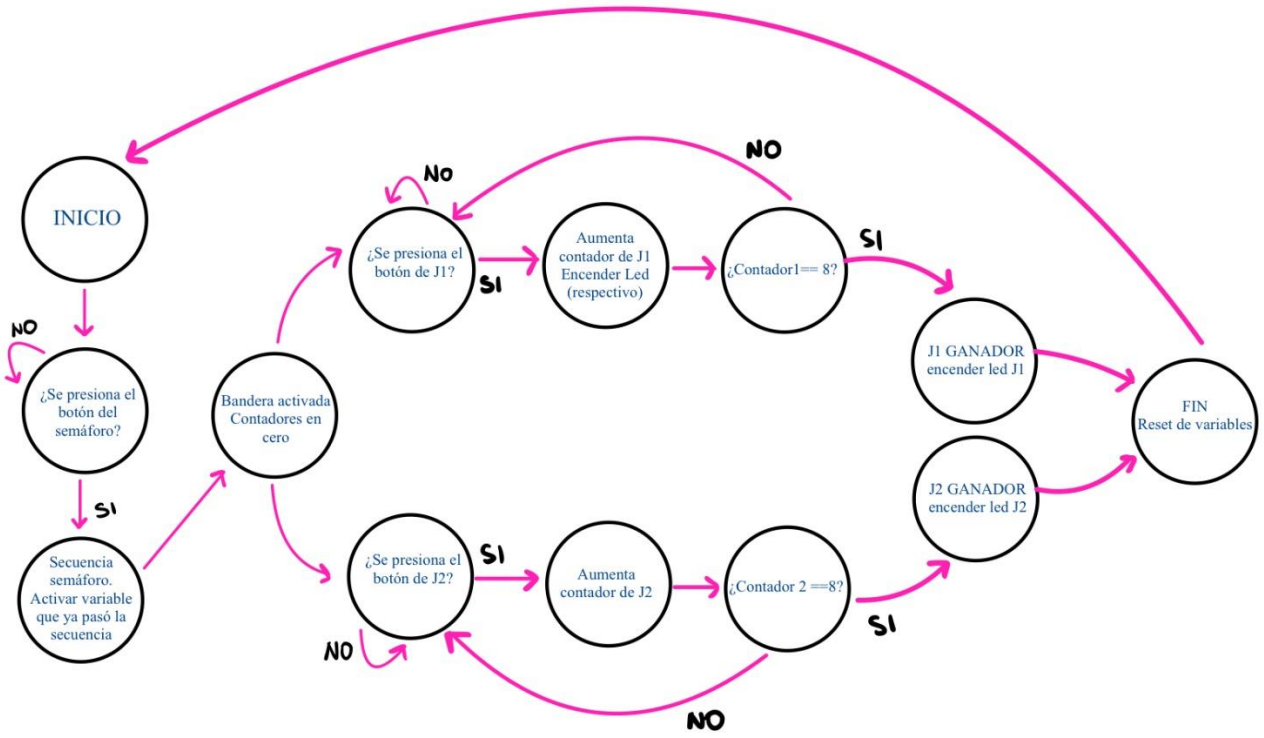
Universidad del Valle de Guatemala
Natalia de León Bercián
Electrónica Digital 2
carne: 18193
enero 2021
sección: 20

Laboratorio # 1

Juego de carreras

Pseudocódigo



Código

```
/*
 * File:  Main_Lab1_Digital2.c
 * Autor: Natalia de Leon Bercian
 * Carne: 18193
 * Seccion: 20
* Titulo: LABORATORIO 1 -- JUEGO DE CARRERAS
* Created on 25 de enero de 2021
```

```
 */

//*****************************************************************************//
//IMPORTAR LIBRERIAS                                    //
//*****************************************************************************//
#include <xc.h>

//*****************************************************************************//
//CONFIGURACION BITS                                    //
//*****************************************************************************//
// 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)

//*****************************************************************************//
//DEFINE                                                //
```

```
//*****************************************************************************//
#define _XTAL_FREQ 8000000

//*****************************************************************************//
//VARIABLES                                                                    //
//*****************************************************************************//
//char contador = 0;
unsigned char contador1 = 0;
unsigned char contador2 = 0;
unsigned char seguro_semaforo = 0; //Asegura que no pueden empezar antes del semaforo
unsigned char J1_GANADOR = 0; //indica cuando ya llega a 8
unsigned char J2_GANADOR = 0;

//*****************************************************************************//
//PROTOTIPOS DE FUNCIONES                                                      //
//*****************************************************************************//
void setup(void);
void semaforo(void);
void conteoJ1(void);
void conteoJ2(void);
void J1_WIN(void);
void J2_WIN(void);

//*****************************************************************************//
//PROGRAMACION PRINCIPAL                                                       //
//*****************************************************************************//

void main(void) {
  setup();

  //*****************************************************************************//
```

```
//LOOP PRINCIPAL                                        //
//*************************************************************************//
while (1) {
   if (PORTBbits.RB0 == 1){ //push del semaforo
      while (PORTBbits.RB0 == 1){//DEBOUNCING
         seguro_semaforo = 0;
      }
      semaforo();
   }
   if (seguro_semaforo == 1){//No avanzan hasta que haya pasado el semaforo

      //Programa para jugador 1
      if (J1_GANADOR != 1){//Si no ha terminado conteo, revisa push
         if (PORTBbits.RB1 == 1){ //por cada botonazo aumenta el contador
            while (PORTBbits.RB1 == 1){ //DEBOUNCING
               contador1 = contador1;
               }
            contador1 = contador1 + 1;
            if (contador1 >= 0  && contador1 <= 8){//se asegura que solo
               conteoJ1();                    //cuente hasta 8
            }
         }
      }
      else {       //cuando ya termino de contar y se activa la variable
         J1_WIN();}//que manda a encender el led de GANAR

      //Programa para jugador (practicamente lo mismo que en J1)
      if (J2_GANADOR != 1){
         if (PORTBbits.RB2 == 1){
            while (PORTBbits.RB2 == 1){//DEBOUNCING
               contador2 = contador2;
```

```
                }
            contador2 = contador2 + 1;
            if (contador2 >= 0 && contador2 <= 8){
                conteoJ2();
            }
          }
        }
        else{
        J2_WIN();}
      }
  }
}
//*****************************************************************************//
//CONFIGURACION  (puertos, bits...)                                            //
//*****************************************************************************//

void setup(void){
   ANSEL = 0;
   ANSELH = 0;
   TRISA = 0b11111000;
   PORTA = 0;
   TRISB = 0b00000111;
   PORTB = 0;
   TRISC = 0;
   PORTC = 0;
   TRISD = 0;
   PORTD = 0;
   TRISE = 0;
   PORTE = 0;

   }
```

```
//*****************************************************************************//
//FUNCIONES                                                    //
//*****************************************************************************//
void semaforo(void){ //enciende con delays las tres luces del semaforo
   PORTE = 0;
   PORTAbits.RA0 = 1; //led rojo
   PORTAbits.RA1 = 0; //led amarillo
   PORTAbits.RA2 = 0; //led verde
   __delay_ms(800);
   PORTAbits.RA0 = 0;
   PORTAbits.RA1 = 1;
   PORTAbits.RA2 = 0;
   __delay_ms(600);
   PORTAbits.RA0 = 0;
   PORTAbits.RA1 = 0;
   PORTAbits.RA2 = 1;
   __delay_ms(400);
   PORTAbits.RA0 = 0;
   PORTAbits.RA1 = 0;
   PORTAbits.RA2 = 0;
   seguro_semaforo = 1; //se activa, para que ya se puedan presionar los push
   }              // de los jugadores

//FUNCION DEL BOTON DEL PRIMER JUGADOR
void conteoJ1(void){     //al presionar el push, se aumenta el contador y segun
   if (contador1 == 1){ //sea el valor, se enciende el led que es
   PORTC = 0b00000001;
   }
   else if (contador1 == 2){
   PORTC = 0b00000010;
```

```
    }
    else if (contador1 == 3){
    PORTC = 0b00000100;
    }
    else if (contador1 == 4){
    PORTC = 0b00001000;
    }
    else if (contador1 == 5){
    PORTC = 0b00010000;
    }
    else if (contador1 == 6){
    PORTC = 0b00100000;
    }
    else if (contador1 == 7){
    PORTC = 0b01000000;
    }
    else {
    PORTC = 0b10000000;
    __delay_ms(400);
    PORTC = 0b00000000;
    J1_GANADOR = 1; //cuando ya llega a los 8, se activa esta variable para que
    }           //indique quien gana
}

//FUNCION DEL BOTON DEL SEGUNDO JUGADOR (lo mismo practicamente que J1)
void conteoJ2(void){
    if (contador2 == 1){
    PORTD = 0b00000001;
    }
    else if (contador2 == 2){
    PORTD = 0b00000010;
```

```
    }
    else if (contador2 == 3){
    PORTD = 0b00000100;
    }
    else if (contador2 == 4){
    PORTD = 0b00001000;
    }
    else if (contador2 == 5){
    PORTD = 0b00010000;
    }
    else if (contador2 == 6){
    PORTD = 0b00100000;
    }
    else if (contador2 == 7){
    PORTD = 0b01000000;
    }
    else {
    PORTD = 0b10000000;
    __delay_ms(400);
    PORTD = 0b00000000;
    J2_GANADOR = 1;
    }
}

void J1_WIN(void){ //cuando gana J1 se enciende el led RE0
    PORTEbits.RE0 = 1;
    PORTEbits.RE1 = 0;
    seguro_semaforo = 0;//se resetean todas las variables para que se pueda
    contador1 = 0;      //jugar de nuevo sin tener que reiniciar el programa
    contador2 = 0;
    J1_GANADOR = 0;
```

```
    J2_GANADOR = 0;
    PORTC = 0;
    PORTD = 0;
}

void J2_WIN(void){ //cuando gana J2 se enciende el led RE1
    PORTEbits.RE0 = 0;
    PORTEbits.RE1 = 1;
    seguro_semaforo = 0;//reset a toda las variables
    contador1 = 0;
    contador2 = 0;
    J1_GANADOR = 0;
    J2_GANADOR = 0;
    PORTC = 0;
    PORTD = 0;
}
```

Link de github

https://github.com/nataliadlb/LABS_REPOSITORIO.git