Universidad del Valle de Guatemala
Proyecto 2 - Digital 2
Estudiantes:
- Pablo Andres Figueroa Gámez 21775
- Sebastián Mayen Dávila 21215

Link Video Funcionamiento Youtube: https://www.youtube.com/watch?v=wMObD6cpBQc&t=1s
Link Github: https://github.com/PabloFigue/Proyecto2_Digital2

Explicación del Código

```
#include <stdint.h>
#include <Energia.h>
#include <stdbool.h>
#include <TM4C123GH6PM.h>
#include <SPI.h>
#include <SD.h>

#include "inc/hw_ints.h"
#include "inc/hw_memmap.h"
#include "inc/hw_types.h"
#include "driverlib/debug.h"
#include "driverlib/gpio.h"
#include "driverlib/interrupt.h"
#include "driverlib/rom_map.h"
#include "driverlib/rom.h"
#include "driverlib/sysctl.h"
#include "driverlib/timer.h"

#include "bitmaps.h"
#include "font.h"
#include "lcd_registers.h"
```

En esta parte lo que se realiza es incluir las librerías que se utilizarán para el control de la LCD, SD, TIVA C e incluir las librerías donde se tendrá parte del juego como lo son los Bitmaps, font.

```
#define LCD_RST PD_0
#define LCD_DC PD_1
#define LCD_CS PA_3
#define SW1 PF_4
#define SW2 PF_0

//pin para la sd
#define scard PA_7

//pines para la musica
#define sound PC_6
#define TX2 PD_7

//Pines para los controles Ralph
#define Rizquierda PB_5
#define Rataque PE_1
#define Rderecha PA_6

//Pines para los controles Felix
#define Farriba PE_3
#define Fabajo PE_2
#define Fderecha PD_2
#define Fizquierda PD_3
```

En esta parte del código lo que se hace es asignar un nombre para los pines de la TIVA C con el objetivo de facilitar la programación. De esta manera, a través de nombrar el pin en cada función, se hace referencia al nombre puesto antes el cual cumple la misma función que nombrar el pin directamente.

```
void LCD_Init(void);
void LCD_CMD(uint8_t cmd);
void LCD_DATA(uint8_t data);
void SetWindows(unsigned int x1, unsigned int y1, unsigned int x2, unsigned int y2);
void LCD_Clear(unsigned int c);
void H_line(unsigned int x, unsigned int y, unsigned int l, unsigned int c);
void V_line(unsigned int x, unsigned int y, unsigned int l, unsigned int c);
void Rect(unsigned int x, unsigned int y, unsigned int w, unsigned int h, unsigned int c);
void FillRect(unsigned int x, unsigned int y, unsigned int w, unsigned int h, unsigned int c);
void LCD_Print(String text, int x, int y, int fontSize, int color, int background);
void LCD_Bitmap(unsigned int x, unsigned int y, unsigned int width, unsigned int height, unsigned char bitmap[]);
void LCD_Sprite(int x, int y, int width, int height, unsigned char bitmap[], int columns, int index, char flip, char offset);
```

Las funciones anteriores son los prototipos para cada una de las funciones de la librería de la LCD SPI.

```
//PROTOTIPOS NUEVOS
void movfelix(int x, int y, int sprintedelay, unsigned char pos1[], unsigned char pos2[], unsigned char pos3[]);
void movralph(int x, int sprintedelay, unsigned char pos1[], unsigned char pos2[], unsigned char pos3[]);
void felixmenu(int sprintedelay);
void movselector(int y);
void reiniciovar(void);

//Prototipos Funciones Interrupcion Movimiento Jugadores.
void Felix_arriba(void);
void Felix_abajo(void);
void Felix_derecha(void);
void Felix_izquierda(void);
void Ralph_ataque(void);
void Ralph_derecha(void);
void Ralph_izquierda(void);
void conteoventana(void);
```

Las funciones anteriores son los prototipos para el movimiento de los personajes y las interrupciones que producen el movimiento de estos.

```
// NUEVAS VARIABLES

extern uint8_t inicio[];
//extern uint8_t fondo[];
extern uint8_t fondoinicio[];

extern uint8_t ralph1nvl3[];
extern uint8_t ralph2nvl3[];
extern uint8_t ralph3nvl3[];
extern uint8_t ralph4nvl3[];
extern uint8_t ralph5dnvl3[];
extern uint8_t ralph5invl3[];
extern uint8_t rstfondo3ralph[];

char conteofelix [19];
unsigned int nivel;
unsigned int disparo;
unsigned int posicionLadrilloX;
unsigned int posicionLadrilloY;
unsigned int vermax;
unsigned int vermin;
unsigned int bandera;

unsigned int posicionralphX;
unsigned int posicionralphY;

unsigned int posicionfelixX;
unsigned int posicionfelixY;
unsigned int orientationfelix;

unsigned int posicionSelector;
```

Esta parte del código consiste en la declaración de las variables que se utilizarán y las variables externas las cuales contienen los gráficos para mostrar en la lcd.

Funciones de interrupción del movimiento de Felix y control de las distintas pantallas del juego.

Control Izquierdo Felix:

```
void Felix_izquierda(void) { //IzquierdaFelix
  if (nivel == 3) { //JUEGO
    orientationfelix = 0;
    if (posicionfelixX == 36) {
      movfelix(0, 0, 100, felix2i, felix3i, felix1i);
    }
    else {
      movfelix(-33, 0, 100, felix2i, felix3i, felix1i);
    }
    conteoventana();
  } else if (nivel == 2) { //MENU
    movfelixmenu(100);
  } else if (nivel == 1) { //INICIO
    nivel = 2;
    bandera = 0;
  } else if (nivel == 4) {
    nivel = 1;
    bandera = 1;
  } else if (nivel == 5) {
    nivel = 1;
    bandera = 1;
  }
}
```

Control Derecho Felix.

```
void Felix_derecha(void) { //DerechaFelix
  if (nivel == 3) {     //JUEGO
    orientationfelix = 1;
    if (posicionfelixX == 168) {
      movfelix(0, 0, 100, felix2d, felix3d, felix1d);
    }
    else {
      movfelix(33, 0, 100, felix2d, felix3d, felix1d);
    }
    conteoventana();
  } else if (nivel == 2) { //MENU
    if (posicionSelector == 168) { //INICIAR
      nivel = 3;
      bandera = 0;
    } else if (posicionSelector == 198) { //MUTEAR
      FillRect(0, 0, 240, 320, 0x0000);
      Serial2.write('6');
      digitalWrite(sound, HIGH);
      digitalWrite(sound, LOW);
      nivel = 2;
      bandera = 0;
    } else if (posicionSelector == 228) { // CREDITOS
      Serial2.write('7');
      digitalWrite(sound, HIGH);
      digitalWrite(sound, LOW);
      FillRect(0, 0, 240, 320, 0x0000);
      if (!SD.begin(scard)) {
        Serial.println("initialization failed!");
        return;
      }
      Serial.println("initialization done.");
      myFile = SD.open("creditos.txt");
      if (myFile) {
        Serial.println("creditos.txt:");
        // read from the file until there's nothing else in it:
        int yc = 10;
        while (myFile.available()) {
          String datasd = myFile.readStringUntil(' ');
          Serial.println(datasd);
          LCD_Print(datasd, 50, yc, 1, 0xffff, 0x0000);
          delay(1500);
          yc = yc + 20;
          if (yc == 330){
            break;
          }
        }

        // close the file:
        myFile.close();
      } else {
        // if the file didn't open, print an error:
        Serial.println("error opening creditos.txt");
      }
      nivel = 2;
      bandera = 0;
    }
  } else if (nivel == 1) { //INICIO
    nivel = 2;
    bandera = 0;
  } else if (nivel == 4) {
    nivel = 1;
    bandera = 1;
  } else if (nivel == 5) {
    nivel = 1;
    bandera = 1;
  }
}
```

En esta función de interrupción se puede observar que si se está en el MENÚ, y se presiona, se estaría escogiendo una opción y por lo tanto realiza diferentes acciones dependiendo de la posición en la que se encuentra el selector.

Control Abajo Felix:

```
void Felix_abajo(void) { //AbajoFelix
  if (nivel == 3) { //JUEGO
    if (orientationfelix == 1) {
      if (posicionfelixY == 264) {
        movfelix(0, 0, 100, felix2d, felix3d, felix1d);
      }
      else {
        movfelix(0, 50, 100, felix2d, felix3d, felix1d);
      }
    } else {
      if (posicionfelixY == 264) {
        movfelix(0, 0, 100, felix2i, felix3i, felix1i);
      }
      else {
        movfelix(0, 50, 100, felix2i, felix3i, felix1i);
      }
    }
    conteoventana();
  } else if (nivel == 2) { //MENU
    if (posicionSelector == 228) {
      movselector(0);
    }
    else {
      movselector(30);
    }
    movfelixmenu(100);
  } else if (nivel == 1) { // INICIO
    nivel = 2;
    bandera = 0;
  } else if (nivel == 4) {
    nivel = 1;
    bandera = 1;
  } else if (nivel == 5) {
    nivel = 1;
    bandera = 1;
  }
}
```

Control Arriba Felix:

```
void Felix_arriba(void) {  //ArribaFelix
  if (nivel == 3) { //JUEGO
    if (orientationfelix == 1) {
      if (posicionfelixY == 114) {
        movfelix(0, 0, 100, felix2d, felix3d, felix1d);
      }
      else {
        movfelix(0, -50, 100, felix2d, felix3d, felix1d);
      }
    } else {
      if (posicionfelixY == 114) {
        movfelix(0, 0, 100, felix2i, felix3i, felix1i);
      }
      else {
        movfelix(0, -50, 100, felix2i, felix3i, felix1i);
      }
    }
    conteoventana();
  } else if (nivel == 2) { //MENU
    if (posicionSelector == 168) {
      movselector(0);
    }
    else {
      movselector(-30);
    }
    movfelixmenu(100);
  } else if (nivel == 1) { //INICIO
    nivel = 2;
    bandera = 0;
  } else if (nivel == 4) {
    nivel = 1;
    bandera = 1;
  } else if (nivel == 5) {
    nivel = 1;
    bandera = 1;
  }
}
```

Funciones de interrupción del movimiento de Ralph y control de las distintas pantallas del juego.

Control Derecha Ralph:

```
void Ralph_derecha(void) { // DerechaRalph
  if (nivel == 3) {  //JUEGO
    if (posicionralphX == 136) {
      movralph(0, 100, ralph5dnvl3, ralph2nvl3, ralph1nvl3);
    }
    else {
      movralph(33, 100, ralph5dnvl3, ralph2nvl3, ralph1nvl3);
    }
  } else if (nivel == 2) { //MENU
    if (posicionSelector == 168) { //INICIAR
      nivel = 3;
      bandera = 0;
    } else if (posicionSelector == 198) { //MUTEAR
      FillRect(0, 0, 240, 320, 0x0000);
      Serial2.write('6');
      digitalWrite(sound, HIGH);
      digitalWrite(sound, LOW);
      nivel = 2;
      bandera = 0;
    } else if (posicionSelector == 228) { // CREDITOS
      FillRect(0, 0, 240, 320, 0x0000);
      Serial2.write('7');
      digitalWrite(sound, HIGH);
      digitalWrite(sound, LOW);
      if (!SD.begin(scard)) {
        Serial.println("initialization failed!");
        return;
      }
      Serial.println("initialization done.");
      myFile = SD.open("creditos.txt");
      if (myFile) {
        Serial.println("creditos.txt:");

        // read from the file until there's nothing else in it:
        int yc = 10;
        while (myFile.available()) {
          String datasd = myFile.readStringUntil(' ');
          Serial.println(datasd);
          LCD_Print(datasd, 50, yc, 1, 0xffff, 0x0000);
          delay(1500);
          yc = yc + 20;
          if (yc == 330){
            break;
          }
        }
        // close the file:
        myFile.close();
      } else {
        // if the file didn't open, print an error:
        Serial.println("error opening creditos.txt");
      }
      nivel = 2;
      bandera = 0;
    }
  } else if (nivel == 1) { //INICIO
    nivel = 2;
    bandera = 0;
  } else if (nivel == 4) {
    nivel = 1;
    bandera = 1;
  } else if (nivel == 5) {
    nivel = 1;
    bandera = 1;
  }
}
```

Control Izquierdo Ralph:

```
void Ralph_izquierda(void) {    //IzquierdaRalph
  if (nivel == 3) {        //JUEGO
    if (posicionralphX == 37) {
      movralph(0, 100, ralph5invl3, ralph2nvl3, ralph1nvl3);
    }
    else {
      movralph(-33, 100, ralph5invl3, ralph2nvl3, ralph1nvl3);
    }
  } else if (nivel == 2) { //MENU
    if (posicionSelector == 168) {
      movselector(0);
    }
    else {
      movselector(-30);
    }
    movralph(0, 100, ralph2nvl3, ralph3nvl3, ralph1nvl3);
  } else if (nivel == 1) { //INICIO
    nivel = 2;
    bandera = 0;
  } else if (nivel == 4) {
    nivel = 1;
    bandera = 1;
  } else if (nivel == 5) {
    nivel = 1;
    bandera = 1;
  }
}
```

Control Abajo/Ataque Ralph:

```
void Ralph_ataque(void) { //AtaqueRalph
  if (nivel == 3) { // JUEGO
    disparo = 1;
    movralph(0, 100, ralph3nvl3, ralph4nvl3, ralph1nvl3);
  } else if (nivel == 2) { //MENU
    if (posicionSelector == 228) {
      movselector(0);
    } else {
      movselector(30);
    }
    movralph(0, 100, ralph2nvl3, ralph3nvl3, ralph1nvl3);
  } else if (nivel == 1) { //INICIO
    nivel = 2;
    bandera = 0;
  } else if (nivel == 4) {
    nivel = 1;
    bandera = 1;
  } else if (nivel == 5) {
    nivel = 1;
    bandera = 1;
  }

}
```

```
void setup() {

  SysCtlClockSet(SYSCTL_SYSDIV_2_5 | SYSCTL_USE_PLL | SYSCTL_OSC_MAIN | SYSCTL_XTAL_16MHZ);
  Serial.begin(115200);
  Serial2.begin(115200);
  SPI.setModule(0);

  pinMode(scard, OUTPUT);
  pinMode(sound, OUTPUT);
  pinMode(Farriba, INPUT);
  pinMode(Fabajo, INPUT);
  pinMode(Fderecha, INPUT);
  pinMode(Fizquierda, INPUT);

  pinMode(Rizquierda, INPUT);
  pinMode(Rderecha, INPUT);
  pinMode(Rataque, INPUT);


  //Anclar Interrupciones a los pines
  attachInterrupt(digitalPinToInterrupt(Farriba), Felix_arriba, FALLING);
  attachInterrupt(digitalPinToInterrupt(Fabajo), Felix_abajo, FALLING);
  attachInterrupt(digitalPinToInterrupt(Fderecha), Felix_derecha, FALLING);
  attachInterrupt(digitalPinToInterrupt(Fizquierda), Felix_izquierda, FALLING);

  attachInterrupt(digitalPinToInterrupt(Rizquierda), Ralph_izquierda, FALLING);
  attachInterrupt(digitalPinToInterrupt(Rderecha), Ralph_derecha, FALLING);
  attachInterrupt(digitalPinToInterrupt(Rataque), Ralph_ataque, FALLING);
  interrupts();

  LCD_Init();
  LCD_Clear(0x00);

  //  INICIO
  nivel = 1;
  disparo = 0;
  //LCD_Bitmap(0, 0, 240, 320, fondo);
  //LCD_Bitmap(87, 5, 65, 53, inicio);

  orientationfelix = 1;

}
```

En el Setup se ejecutará la inicialización de la pantalla, se setean las condiciones iniciales, se declaran los pines de entrada y salida y se Anclan las interrupciones a las funciones de servicio.

A continuación se explicará el LOOP principal del programa.

```
void loop() {
  if (nivel == 1) {

    digitalWrite(sound, HIGH);
    digitalWrite(sound, LOW);
    Serial2.write('1');
    reiniciovar();
    bandera = 1;
    LCD_Bitmap(0, 0, 240, 320, fondoinicio);
    while (bandera == 1) {
      String start = "START";
      LCD_Print(start, 73, 180, 2, 0xffff, 0x0000);
      delay(300);
      FillRect(73, 180, 80, 15, 0x0000);
      delay(300);
    }
  }
```

En esta primera parte se ejecuta la pantalla de inicio, en la cual se muestra el fondo inició y se pone la canción 1. Además se reinician las variables por sí ya es el segundo juego y con él mientras se muestra parpadeando en la pantalla la palabra START.

```
else if (nivel == 2) {

  digitalWrite(sound, HIGH);
  digitalWrite(sound, LOW);
  Serial2.write('2');
  bandera = 1;
  FillRect(0, 0, 240, 320, 0x0000);
  posicionfelixX = 140;
  posicionfelixY = 97;
  posicionralphX = 65;
  posicionralphY  = 64;
  posicionSelector = 168;

  String text1 = "Menu";
  LCD_Print(text1, 87, 30, 2, 0xffff, 0x0000);

  String text2 = "Iniciar";
  LCD_Print(text2, 70, 170, 2, 0xffff, 0x0000);
  String text3 = "Mutear";
  LCD_Print(text3, 70, 200, 2, 0xffff, 0x0000);
  String text4 = "Creditos";
  LCD_Print(text4, 70, 230, 2, 0xffff, 0x0000);

  LCD_Bitmap(posicionralphX, posicionralphY, 66, 66, ralph1nvl3);
  LCD_Bitmap(posicionfelixX, posicionfelixY, 30, 33, felixmenu2);
  LCD_Bitmap(50, posicionSelector, 16, 20, selector);
```

```
    while (bandera == 1) {
      if (posicionSelector == 168) {
        LCD_Print(text3, 70, 200, 2, 0xffff, 0x0000);
        LCD_Print(text4, 70, 230, 2, 0xffff, 0x0000);
        //INICIAR
        LCD_Bitmap(50, posicionSelector, 16, 20, selector);
        LCD_Print(text2, 70, 170, 2, 0xffff, 0x7a7a7a);
        delay(500);
        FillRect(50, posicionSelector, 16, 20, 0x0000);
        LCD_Print(text2, 70, 170, 2, 0xffff, 0x0000);
        delay(500);

      } else if (posicionSelector == 198) {
        LCD_Print(text2, 70, 170, 2, 0xffff, 0x0000);
        LCD_Print(text4, 70, 230, 2, 0xffff, 0x0000);
        //MUTEAR
        LCD_Bitmap(50, posicionSelector, 16, 20, selector);
        LCD_Print(text3, 70, 200, 2, 0xffff, 0x7a7a7a);
        delay(500);
        FillRect(50, posicionSelector, 16, 20, 0x0000);
        LCD_Print(text3, 70, 200, 2, 0xffff, 0x0000);
        delay(500);

      } else if (posicionSelector == 228) {
        LCD_Print(text2, 70, 170, 2, 0xffff, 0x0000);
        LCD_Print(text3, 70, 200, 2, 0xffff, 0x0000);
        // CREDITOS
        LCD_Bitmap(50, posicionSelector, 16, 20, selector);
        LCD_Print(text4, 70, 230, 2, 0xffff, 0x7a7a7a);
        delay(500);
        FillRect(50, posicionSelector, 16, 20, 0x0000);
        LCD_Print(text4, 70, 230, 2, 0xffff, 0x0000);
        delay(500);

      }
    }
  }
```

En el nivel dos lo que se hace es presentar el menú, se pone la música 2, se imprimen en pantalla los 3 textos los cuales son las posiciones y en el while lo que se hace es variar la posición del selector y la animación al cambiar de posición en el selector. La posición del selector incrementa en las rutinas de servicio de interrupción. Por lo que aquí solo se verifica la posición actual la cual se puede modificar en cualquier momento por medio de las interrupciones.

```
else if (nivel == 3) {

  digitalWrite(sound, HIGH);
  digitalWrite(sound, LOW);
  Serial2.write('3');
  bandera = 1;
  FillRect(0, 0, 240, 320, 0x0000);
  posicionfelixX = 36;
  posicionfelixY = 264;
  posicionralphX = 37;
  posicionralphY  = 3;
  orientationfelix = 1;
  LCD_Bitmap(0, 0, 240, 320, fondoinicio);
  LCD_Bitmap(87, 3, 66, 66, rstfondo3ralph);
  LCD_Bitmap(posicionfelixX, posicionfelixY, 31, 39, felixld);
  LCD_Bitmap(posicionralphX, posicionralphY, 66, 66, ralphlnvl3);
  while (bandera == 1) {
```

En esta parte del código consiste en el nivel 3 el cual es el juego, se carga el fondo y se inicializa la posición de cada jugador.

En esta parte se describe el funcionamiento del juego. Cuando el felix pasa sobre las ventanas, automáticamente las repara, dejando el último rastro que es una ventana reconstruida. Y la forma de ganar es pasando por todas las posiciones de todas las ventanas por lo menos 1 vez. Ahora para ganar Ralph lo que debe de hacer es tirar el ladrillo y que Félix esté a la derecha o izquierda inmediata del ladrillo.

```
if (conteofelix[1] == 1 && conteofelix[2] == 1
  nivel = 4;
  bandera = 0;
  disparo = 0;
  break;
}
```

En este if lo que se hace es verificar si félix ya pasó por todas las posiciones de las ventanas. Si sí, entonces félix gana.

```
    if (disparo == 1) {
      posicionLadrilloX = posicionralphX;
      for (int y = 115; y < 300; y = y + 10) {
        LCD_Bitmap(posicionLadrilloX + 26, y, 10, 10, ladrillo);
        LCD_Bitmap(posicionLadrilloX + 26, y - 10, 10, 10, tapladrillo);
        LCD_Bitmap(posicionLadrilloX + 26, 295, 10, 10, tapladrillo);
        posicionLadrilloY = y + 10;
        delay(10);
        if (posicionLadrilloY >= posicionfelixY + 20) {
          if (posicionLadrilloX >= posicionfelixX && posicionLadrilloX <= posicionfelixX + 33) {
            nivel = 5;
            disparo = 0;
            bandera = 0;
            break;
          }
        }
      }
      disparo = 0;
    } else {
      delay(10);
    }
  }
```

En esta parte del código se hace la segunda verificación la cual es para saber si gana Ralph, la cual es verificar la posición del ladrillo con la posición del felix. Si ambas posiciones en Y coinciden, entonces se verifica la posición X. Si Felix está a la izquierda inmediata o derecha inmediata del ladrillo, entonces pierde y gana Ralph.

```
} else if (nivel == 4) {
  Serial2.write('4');
  digitalWrite(sound, HIGH);
  digitalWrite(sound, LOW);
  FillRect(0, 0, 240, 320, 0x0000);
  String win1 = "J1 WINNER";
  LCD_Print(win1, 50, 180, 2, 0xffff, 0x0000);
  while (bandera == 0) {
    posicionfelixX = 110;
    posicionfelixY = 105;
    LCD_Bitmap(posicionfelixX, posicionfelixY, 30, 33, felixmenu1);
    delay(100);
    LCD_Bitmap(posicionfelixX, posicionfelixY, 30, 33, felixmenu2);
    delay(100);
    LCD_Bitmap(posicionfelixX, posicionfelixY, 30, 33, felixmenu3);
  }
```

El nivel 4 es cuando gana Felix. Se muestra en pantalla que ganó el jugador 1 y se muestra una animación de Félix en la pantalla LCD.

```
  } else if (nivel == 5) {
    Serial2.write('5');
    digitalWrite(sound, HIGH);
    digitalWrite(sound, LOW);
    FillRect(0, 0, 240, 320, 0x0000);
    String win2 = "J2 WINNER";
    LCD_Print(win2, 50, 180, 2, 0xffff, 0x0000);
    while (bandera == 0) {
      LCD_Bitmap(85, 80, 66, 66, ralph2nvl3);
      delay(100);
      LCD_Bitmap(85, 80, 66, 66, ralph3nvl3);
      delay(100);
      LCD_Bitmap(85, 80, 66, 66, ralph1nvl3);

    }
  }
}
```

En el nivel 5 se muestra en pantalla que ganó el jugador 2 y se muestra una animación de Ralph.

OTRAS FUNCIONES CLAVE:

```
void reiniciovar(void) {
  for (int i = 0; i < 20; i++) {
    conteofelix[i] = 0;
  }
}
```

En esta función lo que se hace es en el registro conteo félix se recorre cada una de sus posiciones donde previamente se pudo haber guardado 1 si pasó por la posición de 1 ventana y entonces se setea en 0 para volver a jugar.

```
void conteoventana(void) {
  if (posicionfelixY == 264) { //sumarle 50 o restarle
    if (posicionfelixX == 36) { //sumarle 33 o restarle
      conteofelix[0] = 1;
    } else if (posicionfelixX == 69) {
      conteofelix[1] = 1;
    } else if (posicionfelixX == 102) {
      conteofelix[2] = 1;
    } else if (posicionfelixX == 135) {
      conteofelix[3] = 1;
    } else if (posicionfelixX == 168) {
      conteofelix[4] = 1;
    }

  } else if (posicionfelixY == 214) {
    if (posicionfelixX == 36) { //sumarle 33 o restarle
      conteofelix[5] = 1;
    } else if (posicionfelixX == 69) {
      conteofelix[6] = 1;
    } else if (posicionfelixX == 102) {
      conteofelix[7] = 1;
    } else if (posicionfelixX == 135) {
      conteofelix[8] = 1;
    } else if (posicionfelixX == 168) {
      conteofelix[9] = 1;
    }
  } else if (posicionfelixY == 164) {
    if (posicionfelixX == 36) { //sumarle 33 o restarle
      conteofelix[10] = 1;
    } else if (posicionfelixX == 69) {
      conteofelix[11] = 1;
    } else if (posicionfelixX == 102) {
      conteofelix[12] = 1;
    } else if (posicionfelixX == 135) {
      conteofelix[13] = 1;
    } else if (posicionfelixX == 168) {
      conteofelix[14] = 1;
    }

  } else if (posicionfelixY == 114) {
    if (posicionfelixX == 36) { //sumarle 33 o restarle
      conteofelix[15] = 1;
    } else if (posicionfelixX == 69) {
      conteofelix[16] = 1;
    } else if (posicionfelixX == 102) {
      conteofelix[17] = 1;
    } else if (posicionfelixX == 135) {
      conteofelix[18] = 1;
    } else if (posicionfelixX == 168) {
      conteofelix[19] = 1;
    }
  }
}
```

En la función conteo ventana lo que se hace es verificar la posición del félix para ver si coincide con la posición de alguna ventana y si es asi asigna un 1 en una posición específica del registro conteo félix con el objetivo de verificar que haya pasado por todas las ventanas.

```
//FUNCIONES NUEVAS
void movfelixmenu(int sprintedelay) {
  LCD_Bitmap(140, 97, 30, 33, felixmenu1);
  delay(sprintedelay);
  LCD_Bitmap(140, 97, 30, 33, felixmenu2);
  delay(sprintedelay);
  LCD_Bitmap(140, 97, 30, 33, felixmenu3);
}
void movfelix(int x, int y, int sprintedelay, unsigned char pos1[], unsigned char pos2[], unsigned char pos3[]) {
  LCD_Bitmap(posicionfelixX, posicionfelixY, 31, 39, rstfondofelix);
  posicionfelixX = posicionfelixX + x;
  posicionfelixY = posicionfelixY + y;
  LCD_Bitmap(posicionfelixX, posicionfelixY, 31, 39, pos1);
  delay(sprintedelay);
  LCD_Bitmap(posicionfelixX, posicionfelixY, 31, 39, pos2);
  delay(sprintedelay);
  LCD_Bitmap(posicionfelixX, posicionfelixY, 31, 39, pos3);
}
void movralph(int x, int sprintedelay, unsigned char pos1[], unsigned char pos2[], unsigned char pos3[]) {
  LCD_Bitmap(posicionralphX, posicionralphY, 66, 66, rstfondo3ralph);
  posicionralphX = posicionralphX + x;
  LCD_Bitmap(posicionralphX, posicionralphY, 66, 66, pos1);
  delay(sprintedelay);
  LCD_Bitmap(posicionralphX, posicionralphY, 66, 66, pos2);
  delay(sprintedelay);
  LCD_Bitmap(posicionralphX, posicionralphY, 66, 66, pos3);
}
void movselector(int y) {
  FillRect(50, posicionSelector, 16, 20, 0x0000);
  posicionSelector = posicionSelector + y;
  LCD_Bitmap(50, posicionSelector, 16, 20, selector);
}
```

Estas 4 funciones tienen como objetivo crear animaciones de bitmaps. Por lo que solo se mandan a llamar y se puede visualizar una animación en la pantalla LCD. Mandandole los parámetros específicos de cada función para que se ejecute correctamente.

……

Las demás funciones que se pueden observar en el código corresponden a la librería para la LCD.

……