Universidad del Valle de Guatemala
Francisco Lopez 17414
Digital 2

PROYECTO NO. 3
VIDEOJUEGO PORTABLE

El proyecto se trabajará en parejas, cada grupo deberá construir un videojuego utilizando la pantalla LCD ILI9341. En dicho juego implementará todo lo aprendido a lo largo del curso, implementando el código de la pantalla y del juego desde la plataforma Tiva C, deberá tener dos controles los cuales queda a la discreción de cada uno utilizar cualquier otro microcontrolador para esto (entiéndase PIC o Tiva C). El videojuego deberá tener gráficos acordes a lo que se pueda implementar en la memoria de la plataforma.

El videojuego podrá basarse en un juego ya preexistente. Deberá utilizar el módulo de la SD como medio de almacenamiento. Deberá incorporar sonido de 8 bits, haciendo uso de un buzzer pasivo y señales pwm. Podrá hacer uso de sensores como parte de los controles.

CONSIDERACIONES IMPORTANTES

- Trabajo escrito con la información y análisis que realizaron (circuitos utilizados, datos, gráficos, explicaciones, código debidamente comentado) (10 pts.)
- Repositorio de GitHub con varios commits (10 pts.)
- Link a un video de Youtube donde expliquen el proyecto (05pts.)
- Funcionamiento de Proyecto (75 pts.)
    - Funcionamiento de juego (25 pts.)
    - Gráficos acordes (10 pts.)
    - Controles y Sonido (20 pts.)
    - Almacenamiento SD (15 pts.)

TOTAL (100 PTS.)

FECHA DE ENTREGA: El proyecto se deberá enviar (trabajo escrito) antes de medianoche del 30 de abril. SE PRESENTARÁ EN LA SEMANA DEL 27 DE ABRIL, EN EL DÍA DE CLASE (Teórica).
PUNTEO 15 puntos netos
La creatividad y presentación serán valoradas.

Trabajo Escrito:

El Proyecto se basa de un juego muy conocido llamado Space Invaders, y esta versión llamada Defend from Invaders (muy original) te permite jugar el juego para ver que tanta puntuación se puede acumular o que tanto tiempo se puede sobrevivir.

El jugador se muestra en la parte inferior de la pantalla y puede realizar 3 acciones, moverse a la derecha, moverse a la izquierda, y finalmente disparar.

Todos los objetos en el juego, la bala, los aliens, y el jugador se mueven a la misma velocidad, por lo que se requiere poder predecir los movimientos de los alienígenas para poder acertarles.

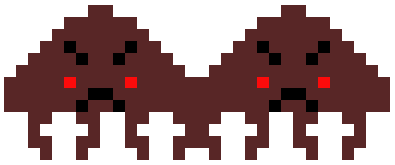
Los alienígenas aparecen a cierto tiempo de haber comenzado el juego y después de esto, siempre habrán tres en pantalla en todo momento, solo puede haber una bala a la vez en pantalla así que disparar múltiples veces no es una opción.

finalmente el jugador pierde únicamente si alguno de los aliens llega a la barrera que se encuentra frente a la nave del jugador.

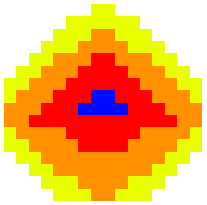
Assets:

Los Sprites usados en el juego son los siguientes:

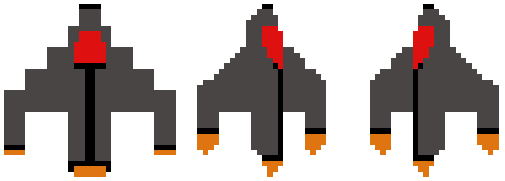
Los alienígenas tienen dos frames



La bala solo se necesita una imagen



y finalmente la nave del jugador que tiene 3 estados



Las tres imágenes fueron realizadas en https://www.piskelapp.com/ y transformadas a 8bits gracias a LCD Image Converter.

La programación del juego se encuentra casi toda dentro del Loop y funciona a base de estados para saber qué es lo que está pasando en todo momento.

```
//*************************************************************************************************************************************
// Loop Infinito
//*************************************************************************************************************************************
void loop() {
  digitalWrite(PF_2, HIGH); //5V
  START = digitalRead(PE_3);

  if (START==HIGH){
    W=0;
  }
  while(W==0){
    delay(5);
    counter++;

    for(int x = 0; x <319; x++){
    LCD_Bitmap(x, 180, 16, 5, tile);
    x += 15;
    }

    String text3 = String(score);
    LCD_Print(text3, 0, 200, 2, 0xffff, 0x0000);

    String text5 = String(counter);
    LCD_Print(text5, 250, 200, 2, 0xffff, 0x0000);

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

El primer estado que aparece es el de W, esta variable indica si el juego está corriendo, se despliegan la nave, los aliens, la bala, los puntajes, y el tiempo.
0 es para indicar que el juego está corriendo, y 1 es para cuando está detenido.

Dentro del while es donde pasa todo lo interesante, tras iniciar el juego el contador comienza a marcar, este es importante ya que es el que define cuándo aparecen los Aliens. La programación de cada Alien es relativamente sencilla pero se requiere de que este se pueda mover independientemente de los otros haciendo que cada alien requiera sus propias variables para todo.

```
//*************************************************************************************************
// Alien 1
//*************************************************************************************************
    if(V1==0){
      int anim1 = (X1/35)%2;

      if(counter >= 30){
        if(bounce1==0){
          LCD_Sprite(X1,Y1,16,16,ALIEN,2,anim1,0,1);
          V_line( X1 -1, Y1, 16, 0x0000);
          X1++;
        }
        if (bounce1==1){
          LCD_Sprite(X1,Y1,16,16,ALIEN,2,anim1,0,1);
          V_line( X1 +16, Y1, 16, 0x0000);
          X1--;
        }
        if (X1==320-32){
          bounce1=1;
          FillRect(X1-1, Y1, 16, 16, 0x0000);
          Y1=Y1+20;
        }
        if (X1==0){
          bounce1=0;
          FillRect(X1+1, Y1, 16, 16, 0x0000);
          Y1=Y1+20;
        }
      }
    }
```

Aquí se puede apreciar cómo este funciona. Lo primero es un estado para ver si está “vivo” este cambia con el contador.
Lo segundo es verificar si el tiempo es el indicado para aparecer. Esto empieza en el vector 1,0 ya que si comienza en 0,0 puede causar problemas.

Tiene un estado llamado bounce que indica la dirección a la que se mueve el Alien, 0 es derecha, y 1 es izquierda y este cambia cuando el Alien llega a cualquiera de los dos límites en x, 320-32 o 0.

Todos los aliens hacen esto así cómo cambian de la animación 0 a la 1 dependiendo de donde están en x.

Ahora tenemos al jugador, este es bastante básico pero la razón es porque la parte complicada fue la bala que se disparó.
El jugador puede desplazarse a la derecha e izquierda mientras esté dentro de los límites de movimiento (0<x<320-32)

```
//*************************************************************
// Jugador
//*************************************************************

    LCD_Sprite(XJ, 200, 32, 32, SHIP,3,animJ,1, 0);
    RIGHT = digitalRead(PF_1);
    LEFT = digitalRead(PA_5);
    SHOOT = digitalRead(PA_7);

    if (RIGHT==HIGH){
      if (XJ<=320-32){
        V_line( XJ -1, 200, 32, 0x0000);
        XJ++;
        animJ=2;
      }
    }
    if (LEFT==HIGH){
      if (XJ>=0){
        V_line( XJ + 16, 200, 32, 0x0000);
        XJ--;
        animJ=1;
      }
    }
    if (SHOOT==HIGH){
      VB=1;
      animJ=0;
    }
```

Finalmente tenemos la bala, esta es un poco complicada ya que es la que detecta la colisión de la bala y los aliens así como esta debe depender del X del jugador para dispararse pero debe mantener la X al estar en el aire, esto se logró con el uso de dos ifs separados, uno para iniciar la bala y el otro para mantener a la bala en movimiento hasta que exista una colisión.

```
//*********************************************************************************
// Bullet
//*********************************************************************************
    if (VB==1){
      YB=200;
      XB=XJ+8;
      showB = 1;
      VB=0;
    }
    if (showB==1){
      LCD_Sprite(XB, YB, 16, 16, BULLET,1,0,1, 0);
      H_line( XB,YB +16,16,0x0000);
      YB--;

      if (YB==0){
        YB=200;
        showB=0;
      }

      if ((XB>=X1-10 && XB<=X1+10) && (YB>=Y1 && YB<=Y1+16)){
        score=score+100;
        FillRect(X1-1,Y1,16, 16, 0x0000);
        YB=200;
        X1 = 1;
        Y1 =0;
        showB=0;
      }
      else if ((XB>=X2-10 && XB<=X2+10) && (YB>=Y2 && YB<=Y2+16)){
        score=score+100;
        FillRect(X2-1,Y2,16, 16, 0x0000);
        YB=200;
        X2 = 1;
        Y2 =0;
         showB=0;
       }
       else if ((XB>=X3-10 && XB<=X3+10) && (YB>=Y3 && YB<=Y3+16)){
         score=score+100;
         FillRect(X3-1,Y3,16, 16, 0x0000);
         YB=200;
         X3 = 1;
         Y3 =0;
         showB=0;
       }
     }
```

La bala está constantemente leyendo y comparando los vectores de esta junto con los vectores de cada alien por separado, si cualquiera de estos entra dentro del rango efectivo dado por el tamaño de los sprites, la bala “borra” la imagen del alien, se resetea en y para no causar colisiones accidentales, suma 100 al puntaje y regresa al alien a la posición 1.0

finalmente tenemos la condición de Game Over, este lee los vectores verticales de los aliens y los compara con la posición de la barra que hay que defender tomando en cuenta el tamaño de los Sprites. Si hay una colisión, la pantalla de Game Over aparece mostrando la puntuación y el último tiempo alcanzado.

```
//*************************************************************
// Game Over
//*************************************************************

    if (Y1 > 180-16 || Y2 > 180-16 || Y3 > 180-16){
      W=1;
      FillRect(0, 0, 319, 239, 0x0000);
      String text1 = "Game Over";
      LCD_Print(text1, 20, 100, 2, 0xffff, 0x0000);
      String text2 = "Score: ";
      LCD_Print(text2, 0, 150, 2, 0xffff, 0x0000);
      String text3 = String(score);
      LCD_Print(text3, 100, 150, 2, 0xffff, 0x0000);
      String text4 = "Time: ";
      LCD_Print(text4, 0, 200, 2, 0xffff, 0x0000);
      String text5 = String(counter);
      LCD_Print(text5, 100, 200, 2, 0xffff, 0x0000);
    }

  }
```

aquí es donde se puede presionar restar en caso se quiera volver a jugar

```
//*********************************************************
// Restart
//*********************************************************
  RESET = digitalRead(PA_6);

  if (RESET == HIGH){
    W = 0;
    counter = 0;

    V1 = 0;
    X1 = 1;
    Y1 =0;
    bounce1 = 0;
    V2 = 0;
    X2 = 50;
    Y2 =0;
    bounce2 = 0;
    V3 = 0;
    X3 = 100;
    Y3 =0;
    bounce3 = 0;

    XJ = 144;

    VB = 0;
    showB = 0;
    XB = 0;
    YB = 200;

    score=0;
  }
}
```

la imagen del juego con los Sprites en su lugar sería la siguiente.

A la derecha abajo está la puntuación, a la izquierda el tiempo, los aliens vienen desde arriba y quieren llegar a la barra que se encuentra sobre el jugador.

Finalmente aquí está el link demostrando el funcionamiento del proyecto y una breve explicación de la programación usada y el link al Github que contiene todos los registros de cambio en el proyecto.

youtube
https://youtu.be/oMHHzXr78bU
Github
https://github.com/lop7414/Proyecto-3