Universidad del Valle de Guatemala

Departamento de Electrónica

Electrónica Digital 2

Pablo Mazariegos

Proyecto # 3

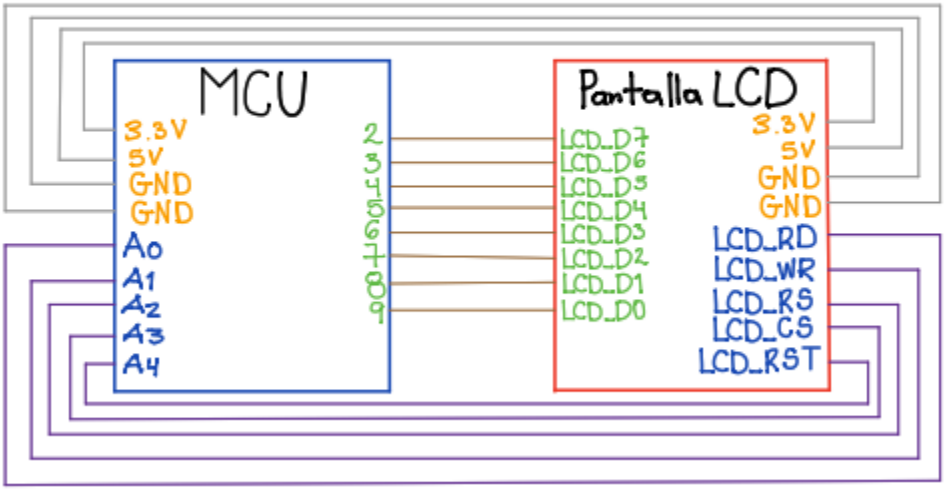
Dino Run

Mariandrée Rivera 18178

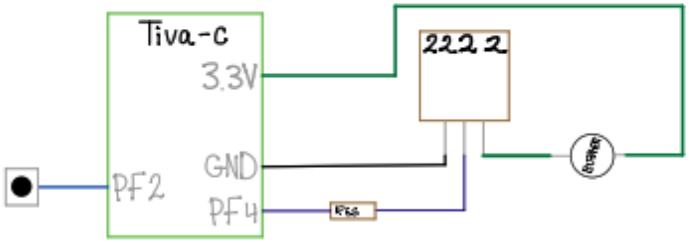
1 de mayo de 2021, Ciudad de Guatemala, Guatemala.

Circuitos Utilizados:

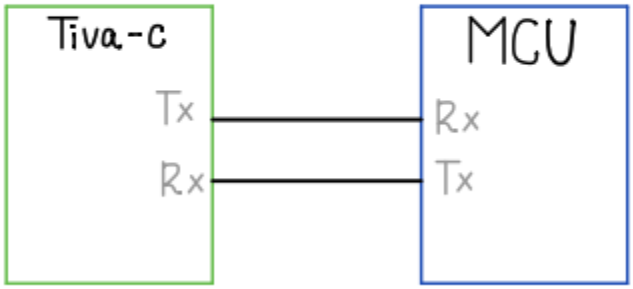
Circuito para implementación de la Pantalla LCD



Circuito para implementación de la musica y el pushbutton para el control



Circuito para implementación de la comunicación Usart

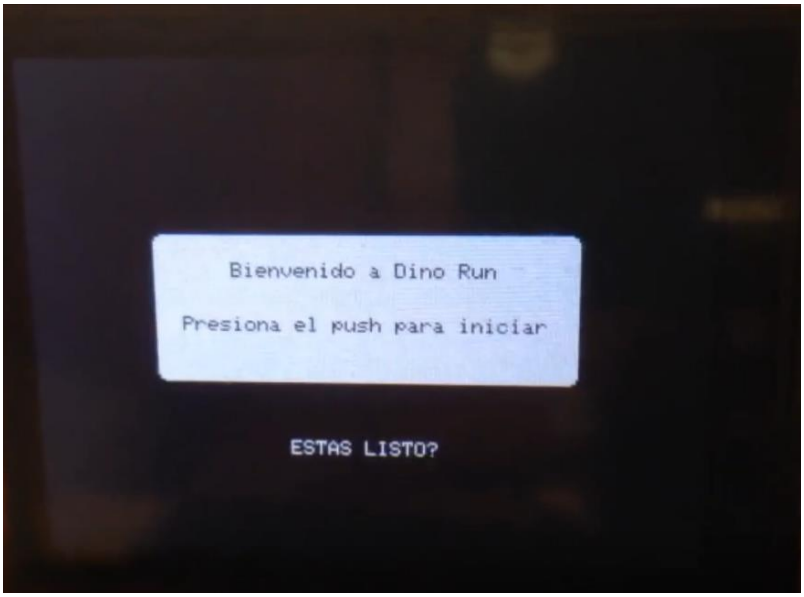


Datos:

- PushT, guarda el dato que viene de la comunicación.
- Run, posición inicial del personaje.
- Bloque, posición inicial del obstáculo.
- Action, variable para entrar a una parte del funcionamiento.

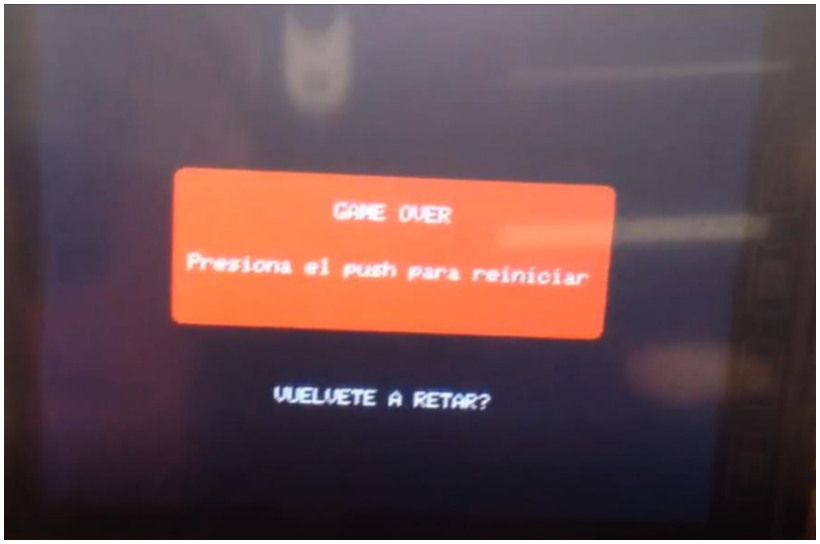
Gráficos:

Pantalla de inicio:



Pantalla del juego:

Pantalla Game Over:



Explicación:

Este juego consistía en que un personaje, inicialmente un dinosaurio, debía esquivar un obstáculo, para ello la forma de hacerlo es brincar sobre él para no golpearse con el obstáculo, ya que este viene en movimiento directo hacia él. De no evitarlo perderá el juego, si logra evitar el bloque, después de un tiempo aleatorio vendrá un nuevo obstáculo que también tendrá que buscar la manera de evitarlo para continuar en el juego.

Para hacer realidad el juego se necesito de una pantalla LCD, dos microcontroladores, un pushbutton y el Buzzer para el sonido. La música que se utilizo para amenizar el juego fue la introducción del programa de televisión Game of Thrones.

Códigos:

Código para el uso de la pantalla:

```
/*******************************************************************************
 * Alumno: Mariandree Rivera
 * Carnet: 18178
 * Proyecto #3, Dino_Run
 * ******************************************************************************
 */
//la libreria fue tomada de :
// UTFT_Demo_320x240
// Copyright (C)2015 Rinky-Dink Electronics, Henning Karlsen. All right reserved
// web: http://www.RinkyDinkElectronics.com/
// This demo was made for modules with a screen resolution
// of 320x240 pixels.
//
// This program requires the UTFT library.
//
// Bitmap info for dinojuego_mono
#include <UTFTGLUE.h>              //libreria de GLUE para usar la pantalla
UTFTGLUE myGLCD(0,A2,A1,A3,A4,A0); //variables utilizadas para la transmision con la pantalla.
 int pushT;
 int pushTE = 10;
 int Run = 0;                 //Variables declaradas
 int Bloque = 0;
 int Action = 1;

// ------------------------------------------------------------------------------
```

```
// ------------------------------------------------------------------------------
// Arduino Uno & ESP32 / 2009:
// -------------------
// Standard Arduino Uno & ESP32 /2009 shield        : <display model>,A5,A4,A3,A2
// DisplayModule Arduino Uno & ESP32 TFT shield     : <display model>,A5,A4,A3,A2
// ------------------------------------------------------------------------------

void setup()
{
  randomSeed(analogRead(0));


  myGLCD.InitLCD();                                 //inicializar la pantalla.
  myGLCD.setFont(BigFont);                          //Seteo de la letra a utilizar.
  Serial.begin(9600);                               //Baute rate para la comunicacion.
  myGLCD.fillScr(0,0,0);                            //limpieza de la pantalla.
}
```

```
void loop(){
   Serial.write(1);                                    //Inicializacion de la comunicacion.
   pushT = Serial.read();                              //lectura del puerto de la comunicacion.
   Home();                                             //manda a llamar la funcion de la inicio.
   delay (1000);
   if (digitalRead(pushTE) == 0){                      //verifica que si el push esta presionado.
    myGLCD.fillScr(0,0,0);                             //coloca el fondo en negro
    Action = 1;
    while (Action == 1){
      Juego();                                         //manda a llamar la funcion principal del juego.
      Runner();
      MoveB ();
      if (digitalRead(pushTE) == 0){                  //verifica que si el push esta presionado.
        SaltoRunner();                                //manda a llamar la funcion del brinco del personaje.
        MoveB ();
      }
      Perder ();
    }
    return;
}
}
```

```
void Home(){
  myGLCD.setColor(255,255,255);                        //color blanco para el cuadrado
  myGLCD.fillRoundRect(60, 80, 260,150);               //parametros donde se crea el rectangulo
  myGLCD.setColor(0, 0, 0);                            //color de la letra
  myGLCD.setBackColor(255,255,255);                    //fondo de la letra
  myGLCD.print("Bienvenido a Dino Run", CENTER, 93);
  myGLCD.print("Presiona el push para iniciar", CENTER, 119);
  Action = 1;
  myGLCD.setColor(255, 255, 255);                      //color de la letra
  myGLCD.setBackColor(0,0,0);                          //fondo de la letra
  myGLCD.print("ESTAS LISTO?", CENTER, 180);
}

void Perder (){

  if (((Bloque<=-120)&&(Bloque)>=-220) && Run == 0){

    myGLCD.fillScr(0,0,0);                             //Coloca la pantalla en negro.
    Gameover();                                        //manda a llamar la funcion del brinco del personaje.
    delay(2000);
    myGLCD.fillScr(0,0,0);

    }
  return;

  }
```

```
void Juego(){
  myGLCD.setColor(14, 95, 166);                        //color del rectangulo del titulo.
  myGLCD.fillRect(0, 0, 319, 13);                      //coordenadas donde se debe imprimir el rectangulo superior.
  myGLCD.fillRect(0, 226, 319, 239);                   //coordenadas donde se debe imprimir el rectangulo interior.
  myGLCD.setColor(255, 255, 255);                      //color de la letra del titulo.
  myGLCD.setBackColor(14, 95, 166);                    //color del desaltado de texto del titulo.
  myGLCD.print("*** Juego Dino Run ***", CENTER, 1);
  myGLCD.setBackColor(14, 95, 166);
  myGLCD.setColor(255,255,255);
  myGLCD.print("By Mariandree Rivera", CENTER, 227);
  myGLCD.setColor(255, 255, 255);
  myGLCD.drawRect(0, 14, 319, 225);
  myGLCD.setColor(255, 255, 255);
  myGLCD.fillRect(0, 196, 319, 201);


}
```

```
void Gameover(){                                                    //funcion que se despliega al perder en el juego.
  myGLCD.fillScr(0,0,0);
  myGLCD.setColor(255,0,0);
  myGLCD.fillRoundRect(60, 80, 260,150);

  myGLCD.setColor(255, 255, 255);
  myGLCD.setBackColor(255,0,0);
  myGLCD.print("GAME OVER", CENTER, 93);
  myGLCD.print("Presiona el push para reiniciar", CENTER, 119);
  myGLCD.setColor(255, 255, 255);
  myGLCD.setBackColor(255,255,255);
  myGLCD.print("VUELVETE A RETAR?", CENTER, 180);
  Action = 0;
  Run = 0;
  Bloque = 0;

  return;

}

void Runner(){
   myGLCD.setColor(255, 255, 255);
   myGLCD.fillRoundRect(110, 130+Run, 120,195+Run);                  //Funcion del personaje.
   myGLCD.fillRoundRect(110, 185+Run, 155,195+Run);
   myGLCD.fillRoundRect(100, 160+Run, 140,170+Run);
   myGLCD.fillRoundRect(100, 140+Run, 140,150+Run);
}
void LimpiarRunner(){
   myGLCD.setColor(0,0,0);
   myGLCD.fillRoundRect(110, 130+Run, 120,195+Run);                  //Funcion de limpieza del personaje.
   myGLCD.fillRoundRect(110, 185+Run, 155,195+Run);
   myGLCD.fillRoundRect(100, 160+Run, 140,170+Run);
   myGLCD.fillRoundRect(100, 140+Run, 140,150+Run);

}

void SaltoRunner(){                                                 //funcion del salto del personaje.

      Run = 0;
      Runner();
      delay(250);
      MoveB ();
      LimpiarRunner();
      Run = -50;
      Runner();
      delay(500);
      MoveB ();
      //Perder ();
      LimpiarRunner();
      Run = -100;
      Runner();
      delay(500);
      MoveB ();
      LimpiarRunner();
      Run = -50;
      //Perder ();
      Runner();
      MoveB ();
      MoveB ();
      LimpiarRunner();
      Run = 0;
      Runner();
}
```

```
void Block (){
  myGLCD.setColor(0, 255, 255);                                          //Funcion del obstaculo.
  myGLCD.fillRoundRect(280+Bloque, 160, 320+Bloque,195);
  }

void LimpiarBlock (){
  myGLCD.setColor(0,0,0);                                               //Funcion de limpieza del obstaculo.
  myGLCD.fillRoundRect(280+Bloque, 160, 320+Bloque,195);
  }

void MoveB (){                                                          //Funcion del movimiento del obstaculo.
      LimpiarBlock();
      Bloque = Bloque -30;
      if(Bloque <= -270){
        Bloque = 0;
        }
      Block();
      delay(450);
}
```

Código para la implementación de música:

```
/*************************************************************************************
 * Alumno: Mariandree Rivera
 * Carnet: 18178
 * Proyecto #3, Dino_Run
 * ***********************************************************************************
 */
/*
  Game of Thrones
  Connect a piezo buzzer or speaker to pin 11 or select a new pin.
  More songs available at https://github.com/robsoncouto/arduino-songs

                                            Robson Couto, 2019
*/
```

```
#define NOTE_B0  31                          //definicion de las notas
#define NOTE_C1  33
#define NOTE_CS1 35
#define NOTE_D1  37
#define NOTE_DS1 39
#define NOTE_E1  41
#define NOTE_F1  44
#define NOTE_FS1 46
#define NOTE_G1  49
#define NOTE_GS1 52
#define NOTE_A1  55
#define NOTE_AS1 58
#define NOTE_B1  62
#define NOTE_C2  65
#define NOTE_CS2 69
#define NOTE_D2  73
#define NOTE_DS2 78
#define NOTE_E2  82
#define NOTE_F2  87
#define NOTE_FS2 93
#define NOTE_G2  98
#define NOTE_GS2 104
#define NOTE_A2  110
#define NOTE_AS2 117
```

```
#define NOTE_B2  123
#define NOTE_C3  131
#define NOTE_CS3 139
#define NOTE_D3  147
#define NOTE_DS3 156
#define NOTE_E3  165
#define NOTE_F3  175
#define NOTE_FS3 185
#define NOTE_G3  196
#define NOTE_GS3 208
#define NOTE_A3  220
#define NOTE_AS3 233
#define NOTE_B3  247
#define NOTE_C4  262
#define NOTE_CS4 277
#define NOTE_D4  294
#define NOTE_DS4 311
#define NOTE_E4  330
#define NOTE_F4  349
#define NOTE_FS4 370
#define NOTE_G4  392
#define NOTE_GS4 415
#define NOTE_A4  440
#define NOTE_AS4 466
#define NOTE_B4  494
#define NOTE_C5  523
#define NOTE_CS5 554
#define NOTE_D5  587
#define NOTE_DS5 622
#define NOTE_E5  659
#define NOTE_F5  698
#define NOTE_FS5 740
#define NOTE_G5  784
#define NOTE_GS5 831
#define NOTE_A5  880
#define NOTE_AS5 932
#define NOTE_B5  988
#define NOTE_C6  1047
#define NOTE_CS6 1109
#define NOTE_D6  1175
#define NOTE_DS6 1245
#define NOTE_E6  1319
#define NOTE_F6  1397
#define NOTE_FS6 1480
#define NOTE_G6  1568
#define NOTE_GS6 1661
#define NOTE_A6  1760
#define NOTE_AS6 1865
#define NOTE_B6  1976
#define NOTE_C7  2093
#define NOTE_CS7 2217
#define NOTE_D7  2349
#define NOTE_DS7 2489
#define NOTE_E7  2637
#define NOTE_F7  2794
#define NOTE_FS7 2960
#define NOTE_G7  3136
#define NOTE_GS7 3322
#define NOTE_A7  3520
#define NOTE_AS7 3729
#define NOTE_B7  3951
#define NOTE_C8  4186
#define NOTE_CS8 4435
#define NOTE_D8  4699
#define NOTE_DS8 4978
#define REST      0
```

```
int tempo = 85;
int buzzer = PF_4;
int push = PF_2;

int melody[] = {

  // Game of Thrones
  // Score available at https://musescore.com/user/8407786/scores/2156716

  NOTE_G4,8, NOTE_C4,8, NOTE_DS4,16, NOTE_F4,16, NOTE_G4,8, NOTE_C4,8, NOTE_DS4,16, NOTE_F4,16, //1
  NOTE_G4,8, NOTE_C4,8, NOTE_DS4,16, NOTE_F4,16, NOTE_G4,8, NOTE_C4,8, NOTE_DS4,16, NOTE_F4,16,
  NOTE_G4,8, NOTE_C4,8, NOTE_E4,16, NOTE_F4,16, NOTE_G4,8, NOTE_C4,8, NOTE_E4,16, NOTE_F4,16,
  NOTE_G4,8, NOTE_C4,8, NOTE_E4,16, NOTE_F4,16, NOTE_G4,8, NOTE_C4,8, NOTE_E4,16, NOTE_F4,16,
  NOTE_G4,-4, NOTE_C4,-4,//5

  NOTE_DS4,16, NOTE_F4,16, NOTE_G4,4, NOTE_C4,4, NOTE_DS4,16, NOTE_F4,16, //6
  NOTE_D4,-1, //7 and 8
  NOTE_F4,-4, NOTE_AS3,-4,
  NOTE_DS4,16, NOTE_D4,16, NOTE_F4,4, NOTE_AS3,-4,
  NOTE_DS4,16, NOTE_D4,16, NOTE_C4,-1, //11 and 12
```

```
 //repeats from 5
 NOTE_G4,-4, NOTE_C4,-4,//5

 NOTE_DS4,16, NOTE_F4,16, NOTE_G4,4, NOTE_C4,4, NOTE_DS4,16, NOTE_F4,16, //6
 NOTE_D4,-1, //7 and 8
 NOTE_F4,-4, NOTE_AS3,-4,
 NOTE_DS4,16, NOTE_D4,16, NOTE_F4,4, NOTE_AS3,-4,
 NOTE_DS4,16, NOTE_D4,16, NOTE_C4,-1, //11 and 12
 NOTE_G4,-4, NOTE_C4,-4,
 NOTE_DS4,16, NOTE_F4,16, NOTE_G4,4,  NOTE_C4,4, NOTE_DS4,16, NOTE_F4,16,

 NOTE_D4,-2,//15
 NOTE_F4,-4, NOTE_AS3,-4,
 NOTE_D4,-8, NOTE_DS4,-8, NOTE_D4,-8, NOTE_AS3,-8,
 NOTE_C4,-1,
 NOTE_C5,-2,
 NOTE_AS4,-2,
 NOTE_C4,-2,
 NOTE_G4,-2,
 NOTE_DS4,-2,
 NOTE_DS4,-4, NOTE_F4,-4,
 NOTE_G4,-1,


  NOTE_C5,-2,//28
  NOTE_AS4,-2,
  NOTE_C4,-2,
  NOTE_G4,-2,
  NOTE_DS4,-2,
  NOTE_DS4,-4, NOTE_D4,-4,
  NOTE_C5,8, NOTE_G4,8, NOTE_GS4,16, NOTE_AS4,16, NOTE_C5,8, NOTE_G4,8, NOTE_GS4,16, NOTE_AS4,16,
  NOTE_C5,8, NOTE_G4,8, NOTE_GS4,16, NOTE_AS4,16, NOTE_C5,8, NOTE_G4,8, NOTE_GS4,16, NOTE_AS4,16,

  REST,4, NOTE_GS5,16, NOTE_AS5,16, NOTE_C6,8, NOTE_G5,8, NOTE_GS5,16, NOTE_AS5,16,
  NOTE_C6,8, NOTE_G5,16, NOTE_GS5,16, NOTE_AS5,16, NOTE_C6,8, NOTE_G5,8, NOTE_GS5,16, NOTE_AS5,16,
};

int notes = sizeof(melody) / sizeof(melody[0]) / 2;
int wholenote = (60000 * 4) / tempo; // calculo de la nota en  ms
int divider = 0, noteDuration = 0;
int HSPI = 0;
int SendD = 0;
```

```
void setup() {

  Serial2.begin(9600);

  pinMode(push,INPUT_PULLUP);             //declaracion del push para el juego
  attachInterrupt(digitalPinToInterrupt(push),pushFunction1,FALLING);

}
void pushFunction1 (){                    //Funcion que nos habla como se envia
  byte info=1;                            //la informacion en UART.
  bitWrite(SendD,1,info);
  delay(10);
  if (Serial2.available()>0){
    HSPI=Serial.read();
    Serial2.write(SendD);
  }
  Serial.println(info);
  delay(10);
  info = 0;
  bitWrite(SendD,1,info);

  }
```

```
void loop() {           // funcion de que envia la informacion para sonar la musica.
    for (int thisNote = 0; thisNote < notes * 2; thisNote = thisNote + 2) {
    divider = melody[thisNote + 1];    // calculo de las notas
    if (divider > 0) {
      noteDuration = (wholenote) / divider;       // nota normal suena.
    }
    else if (divider < 0) {
      noteDuration = (wholenote) / abs(divider);
      noteDuration *= 1.5; // incrementar la duracion de la nota un 50% mas.
    }

    tone(buzzer, melody[thisNote], noteDuration * 0.9);

    delay(noteDuration); // espera el tiempo de la nota sonando.

    noTone(buzzer); // calla el sonido antes de cada nota
  }
}
```

Link:

https://youtu.be/fgenHa0SRs4