UNIVERSIDAD DE EXTREMADURA

PROYECTO CON ARDUINO

JUEGO DEL DINOSAURIO

INFORME

Melany Vinagre León

ÍNDICE:

1.MOTIVACIÓN:

En la actualidad uno de los temas más importantes a tratar y saber manejar es el de la programación .Entender qué hace un determinado código da cierta satisfacción a uno mismo. Pero saber programarlo para que haga una cosa determinada y comprobar que funcione es aún mejor .

Por esta razón he elegido este proyecto , Juego del dinosaurio con Arduino , en el que necesitamos un código muy extenso debido a que hay una gran variedad de posibilidades dentro del juego (saltos, objetos, perder, ganar….) pero a la vez es muy entretenido el estar aprendiendo constantemente qué hace que funcione cada parte del código .

Además hoy en día la mayoría de niños se pasan horas jugando a diversos juegos , entre ellos este , por lo que al tener la oportunidad de poder hacerlo con Arduino y ver cómo funciona por dentro a la vez que saber que se necesita para programar cada parte del código me parecía muy curioso e interesante de hacer .

Por último decir que es un juego muy bueno para poder profundizar los conocimientos de programar con Arduino y saber para qué sirve cada uno de los módulos esenciales con los que se suele programar en él .

2. MATERIAL UTILIZADO Y CIRCUITO EMPLEADO

Los materiales necesarios para este proyecto son muy pocos y fáciles de conseguir en cualquier página web :

- **Arduino Uno:** es una placa de desarrollo la cual es para la que vamos a programar y gracias a ella se produce el funcionamiento del juego . De una manera más sencilla de decir : es una pequeña computadora que puedes programar para que haga lo que tú quieras (en nuestro caso el juego del dinosaurio).

- **LCD keypad shield :** es la pantalla que hemos utilizado en la cual se va a reproducir el juego programado gracias al Arduino.
  Tiene unos botones que puedes presionar para darle órdenes a tu programa. Por lo tanto no hay que usar botones secundarios ni cables ni plotoboard… ya que todo lo necesario viene ya incluido en este dispositivo .
  En conclusión es una herramienta muy útil en general para muchos proyectos ya que puedes imprimir por pantalla cualquier cosa necesaria y también para hacer comprobaciones de tu código en un instante determinado .

  También es verdad que en un primer momento para este proyecto se iba a utilizar otra pantalla diferente sin botones , por lo tanto se necesitaban muchos cables,

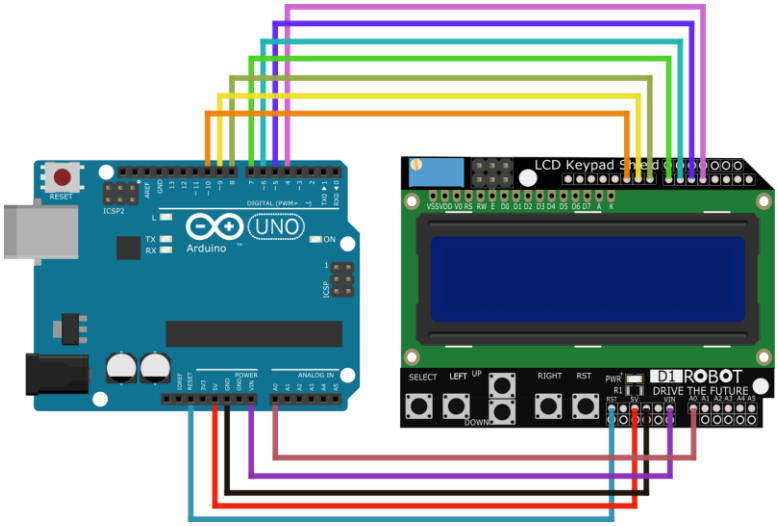
botones etc. Es otra manera de hacerlo pero considero que como se ha realizado con este tipo de pantalla se entiende todo mucho mejor.

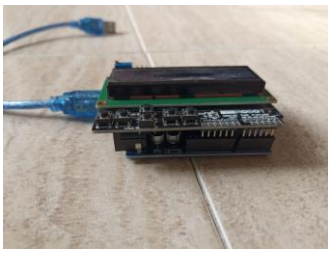
- **<u>Portapilas de 6 pilas :</u>** esto no es estrictamente necesario pero si es muy útil si quieres jugar con tus amigos en algún sitio y no quieres llevarte el ordenador , es una opción muy cómoda si quieres sacar tu juego fuera de casa .

Para realizar el montaje solo es necesario poner la placa de la pantalla sobre la del arduino , en este caso la pantalla tiene diversos pines , se meten todos sobre la placa de arduino y se realizan una serie de uniones para activar lo que se necesita. En este caso son las siguientes:

Diagrama de Conexión (Shield LCD)

- Pin D4 del LCD conectado al pin digital 4 del Arduino.
- Pin D5 del LCD conectado al pin digital 5 del Arduino.
- Pin D6 del LCD conectado al pin digital 6 del Arduino.
- Pin D7 del LCD conectado al pin digital 7 del Arduino.
- Pin A0 del LCD conectado al pin A0 del Arduino (teclado del shield LCD).
- Pin VIN del LCD conectado al pin VIN del Arduino.
- Pin GND del LCD conectado al pin GND del Arduino.
- Pin 5V del LCD conectado al pin 5V del Arduino.
- Pin RST del LCD conectado al pin RST del Arduino.
- Pin RS del LCD conectado al pin digital 8 del Arduino.
- Pin Enable del LCD conectado al pin digital 9 del Arduino.
- LED de retroiluminación del LCD conectado al pin 10 del Arduino.





También adjunto : Diagrama de Conexión (Versión Tinkercad, otra forma de hacerlo):

- Pin D4 del LCD conectado al pin digital 5 del Arduino.
- Pin D5 del LCD conectado al pin digital 4 del Arduino.
- Pin D6 del LCD conectado al pin digital 3 del Arduino.
- Pin D7 del LCD conectado al pin digital 2 del Arduino.
- Pin R/W del LCD conectado a tierra del Arduino.
- Pin VSS del LCD conectado a tierra del Arduino.
- Pin VCC del LCD conectado a 5V del Arduino.
- Pin RS del LCD conectado al pin digital 12 del Arduino.
- Pin Enable del LCD conectado al pin digital 11 del Arduino.
- Terminal 1a del botón pulsador conectado al pin 7 del Arduino.
- Terminal 2a del botón pulsador conectado a tierra del Arduino.

Importante: en esta página no hay exactamente ese modelo de LCD por lo tanto no se pueden realizar igual las conexiones (ya que se usa otro tipo de LCD , sin los mismos pines).

3.CÓDIGO EMPLEADO

Para poder realizar este juego necesitamos un código muy extenso ya que hay muchos casos(y muchas posibilidades en cada uno) , adjunto el código empleado con la explicación de cada línea de código implementada y el por qué:

```
-----------,--------IMPORTAMOS BIBLIOTECAS NECESARIAS PARA EL JUEGO -------------------------------
  #include <LiquidCrystal.h>// para manejar una pantalla LCD. (La pantalla LCD es donde veremos el juego. )
  #include <Keypad.h>// nos enseña cómo detectar cuando presionamos un botón (en este juego no se usa mucho)
  #include <EEPROM.h>//nos ayuda a guardar la puntuación más alta en un lugar seguro (memoria externa), incluso si apagamos el Arduino.

// Comenta esta sección si estás ejecutando el código con un solo botón(en mi caso yo siempre uso dos  botones de la pantalla : up y reset )
  #define CMK_HARDWARE
  const int eepromAddress = 0; // direccion de la eeprom para guardar el valor del record sin que se pierda
  int valorrecord = 0; // valor inicial que vamos a meter en la eeprom para luego modificarlo(todas las veces que se actualice )
  #ifdef CMK_HARDWARE
    // LCD pins
    #define RS 8    // LCD reset pin
    #define En 9    // LCD enable pin
    #define D4 4    // LCD data pin 4
    #define D5 5    // LCD data pin 5
    #define D6 6    // LCD data pin 6
    #define D7 7    // LCD data pin 7
  #else
    // LCD pins
    #define RS 12   // LCD reset pin
    #define En 11   // LCD enable pin
    #define D4 5    // LCD data pin 4
    #define D5 4    // LCD data pin 5
    #define D6 3    // LCD data pin 6
    #define D7 2    // LCD data pin 7
  #endif
  //inicializamos la pantalla LCD
  LiquidCrystal lcd(RS, En, D4, D5, D6, D7);

  // puntos del juego
  int cuadrados = 0; // numero de cuadraditos totales que consigues pasar en una partida en el juego( en la pantalla lo tenemos puesto con el nombre "puntos")

  // Limitamos la frecuencia de salto( porque si no ponemos un limite el personaje podria mantenerse arriba todo el tiempo que quisiese y no tendria complicacion el juego)
  bool salto = false;
```

```
//-------PERSONAJES NECESARIAS EN EL JUEGO----------
// ----- Vamos a configurar cada dibujo que aparece en el juego para que pueda aparecer bien en la pantalla LCD
  // dinosaurio con pierna derecha
  byte dino_r[8] = {
    B00000111,
    B00000101,
    B00000111,
    B00010110,
    B00011111,
    B00011110,
    B00001110,
    B00000010
  };
  // dinosaurio con pierna izquierda
  byte dino_l[8] = {
    B00000111,
    B00000101,
    B00000111,
    B00010110,
    B00011111,
    B00011110,
    B00001110,
    B00000100
  };
   //cactus grande
  byte cactus_big[8] = {
    B00000000,
    B00000100,
    B00000101,
    B00010101,
    B00010110,
    B00001100,
    B00000100,
    B00000100
  };
  // Cactus pequeño
  byte cactus_small[8] = {
    B00000000,
    B00000000,
    B00000100,
    B00000101,
    B00010101,
    B00010110,
    B00001100,
    B00000100
  };
 // configuramos el mundo para crear una pequeña parte el  juego.(CHAR WORLD )

//Cada número dentro del arreglo representa un elemento o carácter que se mostrará en la pantalla

   //  112, 117, 110, 116, 111, 115 es la palabra puntos  que utilizaos para imprimir por pantalla la variable cuadrados
  // el numero 32 es espacio en blanco
  //el numero 0 es un dinosaurio y el 3 es un cactus
  char world[] = {
    32, 32, 32, 32, 32, 32, 32, 112, 117, 110, 116, 111, 115, 32, 32, 32,
    32, 0, 32, 32, 32, 32, 32, 32, 32, 32, 32, 32, 32, 32, 32, 32,
};
  // bucle de juego infinito
  uint8_t scroll_world() {
    // en el hardware real necesitamos un retraso (delay 250)
    #ifdef CMK_HARDWARE// comprobamos si esta definida la macro ( para un uso determinado)
      delay(250);// el ptograma se pausa durante 0.25 segundos
    #endif
     //Esta línea de código decide si el siguiente objeto que aparecerá en la pantalla será un cactus pequeño, un cactus grande o un espacio vacío.
    //-----------POSIBLES CASOS Y QUE AARECERA EN PANTALLA -------------
    // Si el número es muy pequeño (cerca del 2), aparece un cactus grande.
    //Si el número es un poco más grande, aparece un cactus pequeño.
    //Si el número es muy grande (cerca del 34), aparece un espacio vacío.
    // el numero que sale se almacena en la variable que esta definida como objeto_aleatorio
    char objeto_aleatorio = random(2, 35);
    // lugar del cactus en el mapa
     if (objeto_aleatorio< 4) world[31] = objeto_aleatorio;//si aparece esto se pondra un cactus en la posicion 31 ( aparecera en la pantalla )
    // si no lo que nos indica este codigo es que en esa posicion (si el numero aleatorio es >4 )aparecera un espacio vacio
    else world[31] = 32;
     //Esta parte del código se encarga de hacer que los cactus se muevan hacia el dinosaurio, simulando que el dinosaurio está corriendo.
```

```
   // Recorre la segunda fila del mapa del juego
    for (int i = 16; i < 32; i++) {// con esto revivimos constantemente la segunda fila de la pantalla (16-31, ya sabemos que cada numero es un hueco en la pantalla LCD)

      // vemos si hay un cactus ( es 2 o es 3 ) para que se ejecute la siguiente parte del codigo
      if (world[i] == 2 or world[i] == 3) {

         // vemos si la celda anterior era cactus o estaba vacia ( se guarda en anterior , hemos utilizado esta variable para que se entienda mejor )
        char anterior = (i < 31) ? world[i + 1] : 32;

        //vemos si el espacio justo antes del cactus está vacío. Si no está vacío, significa que el dinosaurio está ahí y ha chocado con el cactus.
        //Si hay una colisión, el juego termina.
        if (world[i - 1] < 2) return 1;
        world[i - 1] = world[i];//Esta parte hace que el cactus se mueva un espacio hacia la izquierda.
        world[i] = anterior;//Aquí se llena el espacio que quedó vacío detrás del cactus con lo que había antes(cactus o vacio )
      }
    }
    world[15] = 32;//Esto asegura que la esquina superior derecha del mundo del juego siempre esté vacía.(si aparece algo , mal)

    // este if verifica si hay un cactus en la posición 16 del mundo del juego.
    //Si hay un cactus ahí, significa que el dinosaurio acaba de saltar y dejar un espacio vacío. Entonces, se borra el cactus de esa posición(para que no haya fallos)
    if (world[16] < 2) world[16] = 32;

    return 0;// el juego continua ya que si se cumple la condicion significa que  no habria chocado el dinosaurio
  }
  void update_world() {  // hace que el cactus venga hacia el dinosaurio

    int juego_terminado = scroll_world();// vemos si detecta colisiones o no (1 , seria jueo terminado ) se almacena en juego_terminado y si pasa eso salta a la linea de codigo de juego_terminado

    // en el caso en el que se gane el juego aparecera la siguiente linea de codigo
    if (cuadrados ==250) {// se pone a 250 ya que es un valor alto y dificil pero que a la vez que algina vez se pueda ganar , se ha comprobado que se imprime bien en pantalla
      lcd.setCursor(0, 0);
      lcd.print("   HAS GANADO!    ");
      lcd.setCursor(0, 1);
      lcd.write(byte(0));
      lcd.write(byte(32));
      lcd.write(byte(2));
      lcd.write(byte(2));
      lcd.write(byte(2))
      lcd.write(byte(3));
      lcd.write(byte(3));
      lcd.write(byte(3));
      lcd.write(byte(3));
      lcd.write(byte(3));
      lcd.write(byte(3));
      lcd.write(byte(2));
      lcd.write(byte(2));
      lcd.write(byte(2));
      lcd.write(byte(32));
      lcd.write(byte(1));
      while(1);
    }
    // se ha acabado el juego porque hemos chocado contra un cactus ( aparecera esto en pantalla )
    if (juego_terminado) {
      lcd.setCursor(0, 1);
      lcd.write(byte(0));// ya sabemos que el 0 representara un dinosaurio
      lcd.write(byte(3));// este reresentara un cactus
      lcd.print("HAS PERDIDO!");
      lcd.write(byte(3));
      lcd.write(byte(0));
      while(1);
    }
    // esta linea de codigo se utiliza para ir incrementando el numero de cuadrados que salen en pantalla
     //Así  mostramos  en la pantalla cuánto cuadrados  ha recorrido(que son nuestros puntos ) .
    cuadrados++;

    // actualizamos los puntos ( el numero de cuadrados recorridos en esa partida )
    //lo mostramos en pantalla (puntos es el nombre que se muestra ,indica el maximo de esa partida )
    lcd.setCursor(13, 0);
    lcd.print(cuadrados);

    lcd.setCursor(0, 0);// ponemos el cursor para empezar a poner cosas a partir de esa posicion
    //------Bucle para ir posicionando el mundo ------------
```

```
    for (int i = 0; i < 32; i++) {// recorremos todas las posiciones de la pantalla con este bucle for
       if (world[i] < 2) world[i] ^= 1;// con esto hacemos que el dinosaurio simule que camina levantando pierna izquierda y derecha simultaneamente
      if (i == 16) lcd.setCursor(0, 1);// aqui actualizamos fila ya que al llegar al hueco 16 en la pantalla ya seria la segunda fila que tenemos
      if (i < 13 || i > 15)// imprimimos en la pantalla lo que nos toque en esa posicion
        lcd.write(byte(world[i]));
    }
  }
  // para el uso de los botones que implementamos en el juego
  bool get_button() {
    // uso del boton up de la pantalla
    #ifdef CMK_HARDWARE
      int entrada_pantalla;
     entrada_pantalla = analogRead (0);//Aquí se lee el valor analógico del pin 0 del Arduino. Esto se utiliza para leer los botones del shield LCD.
      if (entrada_pantalla < 200) return LOW;  // button up (vemos que si es menor , el boton esta siendo presionado )
      else return HIGH;
    #else
      return digitalRead(7);// esta linea " no hace falta" es para por si se ejecuta con otro hardware
    #endif
  }
  //------------- arduino setup----------
  void setup() {
       Serial.begin(9600); // Inicia la comunicación serial
    delay(1000); // Espera un segundo para asegurarse de que el Monitor Serial esté listo
    // Inicializa el LCD
    #ifndef CMK_HARDWARE
      EEPROM.put(eepromAddress, valorrecord);// ponemos el valorrecord en la posicion correspondiente de la eeprom
      pinMode(7, INPUT_PULLUP);// por si no esta conectada la pantalla  LCD
    #endif
    // -------asociamos numeros a objetos------------
    lcd.createChar(0, dino_l);// el numero 0 se asocia al dinosaurio con la pierna izquierda
    lcd.createChar(1, dino_r);// el numero 1 se asocia al dinosaurio con la pierna derecha
    lcd.createChar(2, cactus_small);// numero 2 con cactus pequeño
    lcd.createChar(3, cactus_big);// numero 3 con el cactus grande
    lcd.begin(16, 2);// inicializamos la pantalla con su tamaño correspondiente (16x2)
  }
  // -------------arduino loop-----------------
  void loop() {
       lcd.setCursor(0, 0); // para empezar a poner cosas en la pantalla desde esa posicion (inicial)
    int readValue; // inicializamos esta variable para leer el valor en la EEPROM del valor que le hemos puesto en esa casilla (RECORD DE CUADRADOS , puntos )
    readValue=EEPROM.read(eepromAddress);//leemos ese valor
    lcd.print(" ESTAS LISTO? ");
    lcd.setCursor(1, 0);
    lcd.setCursor(0, 1);
    lcd.write(byte(0));
    lcd.write(byte(2));
    lcd.print("RECORD:");
    lcd.print(readValue);//imprimimos ese valor de record de cuadrados(máximo de puntos) en pantalla inicial
    lcd.write(byte(3));
    lcd.write(byte(0));
    while(get_button() == HIGH);
    // ----------------loop juego-------------
    // el while true inicia un bucle infinito.
    // Esto significa que el código dentro de este bucle se repetirá constantemente hasta que se apague el dispositivo o se reinicie el programa.
    while(true) {
      salto ^= 1; // Para evitar que el dinosaurio se quede colgado en la fila superior todo el tiempo
      // cuando presionamos el boton
      if (get_button() == LOW && salto == true) {// si se encuentra  saltando se realiza lo de dentro del if
        // actualizamos la posicion del dinosaurio en la pantalla
        lcd.setCursor(1, 1);
        lcd.write(byte(32));
        lcd.setCursor(1, 0);
        lcd.write(byte(0));
        // actualizamos la posicion del dinosaurio en el juego
        world[1] = byte(0);
        world[17] = byte(32);
        // mientras el dinosaurio se "actualiza" para ver que tiene que salir despues
        for (int i = 0; i < 4; i++) update_world();

        // actualizamos la posicion del dinosaurio en el juego
        world[1] = byte(32);
        world[17] = byte(0);

        // actualizamos la posicion del dinosaurio en la pantalla
        lcd.setCursor(1, 0);
        lcd.write(byte(32));
```

```
        lcd.setCursor(1, 1);
        lcd.write(byte(0));
      }

      //------  para almacenar el valor del RECORD de cuadrados en la eeprom-----------------
       if (cuadrados > readValue) {// este if se hace para actualizar el valor en el caso de que sea mayor
       readValue = cuadrados;// si es mayor pues ponemos ese valor
       EEPROM.update(eepromAddress , readValue);// y actualizamos la eeprom con el nuevo valor  del record
       }

      //actualizamos el mundo para volver a iniciar una nueva partida en el juego , la variable record de la pantalla
siempre se mantendrá pase lo que pase
      update_world();
    }
  }
```

4 Resultado y futuro desarrollo

El resultado que se ha obtenido una vez hecho el código y montado la pantalla con el Arduino es poder ver la simulación de un juego conocido como el juego del dinosaurio de Google que consta de un dinosaurio que tiene que ir saltando los obstáculos (cactus) que aparecen .

Además se ha realizado una carcasa para que recree mejor una consola con su videojuego que es el fin de este trabajo .

En el juego se guardará el record máximo de puntos que aparecerá en la pantalla inicial , antes de empezar la partida .No obstante aparece la puntuación de cada partida una vez pierdes y si has superado tu record anterior , este se actualizará . Adjunto unas imágenes para que se vea mejor dicho resultado:

Imagen 1

Imagen 2

Imagen 3

Aquí vemos en primer lugar el inicio del juego, pulsamos el botón up y comenzamos a jugar (imagen 2) , en el momento que choquemos aparecerá algo similar a la imagen 3 pero con los puntos obtenidos en esa partida en concreto .Si queremos volver a jugar le damos al botón de reset y nos vuelve a aparecer el inicio del juego .

En el caso de que ganemos , se acabará el juego y aparecerá esto en pantalla:

Enhorabuena , te habrás pasado el juego, has conseguido el máximo número de puntos posibles (que en este juego está programado para que sean 250, que es un número difícil de llegar pero a veces con un poco de práctica es posible)

A continuación voy a explicar un futuro desarrollo que sería posible para este juego :

- Poner un altavoz para que programándolo y haciendo su correspondiente configuración hagamos que en el momento en el que se encienda el juego, este empiece a sonar con una canción (típica de cualquier videojuego) para conseguir un mejor ambiente a la vez de que sea más entretenido bajo mi punto de vista .
- Hacer diferentes niveles en el juego , es una forma más entretenida al tener que ir pasando de niveles más fáciles a más difíciles , aunque eso ya es un desarrollo más complicado y habría que dedicarle mucho tiempo al código es una innovación que en un futuro estaría muy bien .

5.CONCLUSIÓN

Este proyecto ha sido una gran opción ya que gracias a él he aprendido mucho a cerca de la programación con Arduino a la vez de cómo sería poder ver y hacer un videojuego desde dentro que es algo por lo que siempre he tenido mucha curiosidad por lo que este proyecto a sido una gran oportunidad para mi, ya que he podido profundizar y adquirir muchos conocimientos de algo que era completamente nuevo para mi.

6.BIBIOGRAFÍA

Para poder entender un poco mejor la manera en la que había que hacer el código (funcionamiento de la pantalla, manejo de la eeprom, crear la forma del dinosaurio… y todas las opciones y dudas que han aparecido mientras lo he programado) he visto algunos videos explicativos y páginas web para adquirir algo más de información .

https://www.youtube.com/watch?v=8rZ6LyCiRwY

https://www.prometec.net/lcd-keypad-shield/

https://www.youtube.com/watch?v=ubUx5pf8rp0

https://programarfacil.com/blog/arduino-blog/eeprom-arduino/