Practica 5: Semáforo y Proyecto Libre

David Martínez Díaz GII-ADE

- **Proyecto: Semáforo.**

Código final:

```
// Declaro las variables
int verde = 2;
int amarillo = 3;
int rojo = 4;

// Declaro los tiempos de espera de cada led
int tiempo_verde = 5000;
int tiempo_amarillo = 300;
int tiempo_rojo = 5000;

// Seleccion el modo de cada pin, en este caso, con cada leds

void setup()
{
  pinMode(verde, OUTPUT);
  pinMode(amarillo, OUTPUT);
  pinMode(rojo, OUTPUT);
}

// En el bucle vamos a llamar a los distintos metodos y sus tiempos
void loop()
{
  luz_verde();
  delay(tiempo_verde);

  luz_amarilla();

  luz_roja();
  delay(tiempo_rojo);
}

// En el metodo apagamos los leds y encedemos el verde

void luz_verde(){

  digitalWrite(verde, HIGH);
  digitalWrite(amarillo, LOW);
  digitalWrite(rojo, LOW);
}

// En el metodo apagamos los leds y encedemos el amarillo
void luz_amarilla()
{
  digitalWrite(verde, LOW);
  digitalWrite(rojo, LOW);

  // Con un bucle for, vamos haciendo que el led amarillo se ponga
  for(int i=0; i<5; i++){

    digitalWrite(amarillo, HIGH);
    delay(tiempo_amarillo);
    digitalWrite(amarillo, LOW);
    delay(tiempo_amarillo);

  }

}

// En el metodo apagamos los leds y encedemos el rojo
void luz_roja()
{
  digitalWrite(verde, LOW);
  digitalWrite(amarillo, LOW);
  digitalWrite(rojo, HIGH);
}
```

Explicación del código:

Primero declaras las variables “amarillo, verde, rojo”, que son los dispositivos leds igualándolos a sus respectivos pines a los cuales van enchufados.

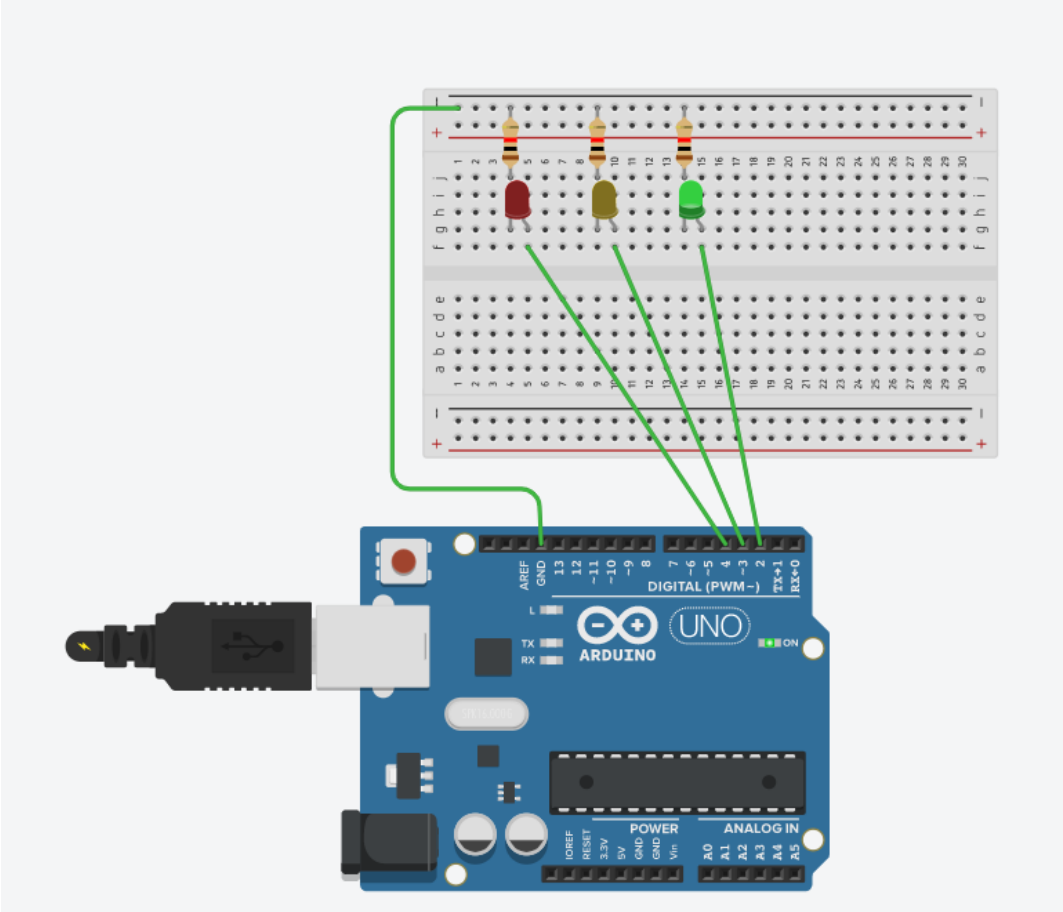
Por otro lado, declaramos las variables de tiempos de espera para cada uno de los leds, por ellos son llamados “tiempo_verde, tiempo_rojo, tiempo_amarillo”.

Ahora empezamos con la función setup(), este es la función que se llama nada más comenzar la simulación, ahí definimos que, en los pines de los leds, se forman en modo OUTPUT, para que cuando reciban cierta señal actúen y se iluminen entre otras más.

Después en la función loop(), la cual es la que se ejecuta en un bucle infinito, llamamos a distintas funciones externas, como son luz_verde, luz_roja, luz_amarillo, con sus respectivos tiempos de espera a través de la función delay().

Una vez dentro de los métodos comentemos la función “luz_verde” y la función “luz_roja”, donde simplemente escriben el modo en el que se encuentran el pin, haciendo que se enciendan su respectivo led y se apaguen los otros dos.

En la función luz_amarilla se presenta una variante, en esta se realiza un bucle for donde se van encendiendo y apagando la luz, para que de ese efecto intermitente característicos de los semáforos.



- **Proyecto: Piano.**

Código final:

```
// Defino la palabra ACTIVADO a LOW para que cuando un boton se
// pulse se compare con este.
#define ACTIVADO LOW
#include <LiquidCrystal.h>

#define COLS 16 // Columnas del LCD
#define ROWS 2 // Filas del LCD

// Declaro el megafono con su correspondiente pin
const int megafono = 3;

LiquidCrystal lcd(12, 13, 7, 5, 4, 2);

// Declaro los botones con uss respectivos pines

const int boton_B =6;
const int boton_C =8;
const int boton_D =9;
const int boton_E =10;
const int boton_F =11;

// Inicio cada una de las variables con sus outputs e inputs.

void setup(){

  pinMode(megafono,OUTPUT);

  // INDICAMOS QUE TENEMOS CONECTADA UNA PANTALLA DE 16X2
  lcd.begin(COLS, ROWS);
  // MOVER EL CURSOR A LA PRIMERA POSICION DE LA PANTALLA (0, 0)
  lcd.home();
  // IMPRIMIR "Hola Mundo" EN LA PRIMERA LINEA
  lcd.print("Pulse un boton");



  pinMode(boton_B,INPUT);
  digitalWrite(boton_B,HIGH);

  pinMode(boton_C,INPUT);
  digitalWrite(boton_C,HIGH);

  pinMode(boton_D,INPUT);
  digitalWrite(boton_D,HIGH);

  pinMode(boton_E,INPUT);
  digitalWrite(boton_E,HIGH);

  pinMode(boton_F,INPUT);
  digitalWrite(boton_F,HIGH);

}


// Luego en el loop hacemos las distintas comparaciones para
// saber si un boton esta siendo pulsado o no.

void loop(){

  // Si se cumple que algun boton esta siendo pulsado se le
  // llama al megafono, con su respectiva frecuencia de sonido.


  while(digitalRead(boton_B) == ACTIVADO){
    tone(megafono,420,500);
    lcd.clear();
    lcd.print("Nota: (420, 500).");
  }

  while(digitalRead(boton_C) == ACTIVADO){
    tone(megafono,490,500);
    lcd.clear();
    lcd.print("Nota: (490, 500).");
  }

  while(digitalRead(boton_D) == ACTIVADO){
    tone(megafono,530,500);
    lcd.clear();
    lcd.print("Nota: (530, 500).");
  }

  while(digitalRead(boton_E) == ACTIVADO){
    tone(megafono,590,500);
    lcd.clear();
    lcd.print("Nota: (590, 500).");
  }

  while(digitalRead(boton_F) == ACTIVADO){
    tone(megafono,620,500);
    lcd.clear();
    lcd.print("Nota: (620, 500).");
  }

  // Por ultimo, apagamos el megafono y volvemos a reiniciar
  // el bucle.
  noTone(megafono);

}
```

Explicación del código:

En primer lugar, para este proyecto nos va a hacer falta la libre LiquidCrystal.h, la cual nos permitirá utilizar la pantalla LCD a nuestro gusto.

Además, he definido una constante “ACTIVADO = LOW”, porque más adelante lo utilizare para comprobar ciertas condiciones, “COLS = 16”, una constante que nos diga el número de columnas que tendrá la pantalla LCD y “ROWS = 2” otra constante para las filas de la pantalla LCD.

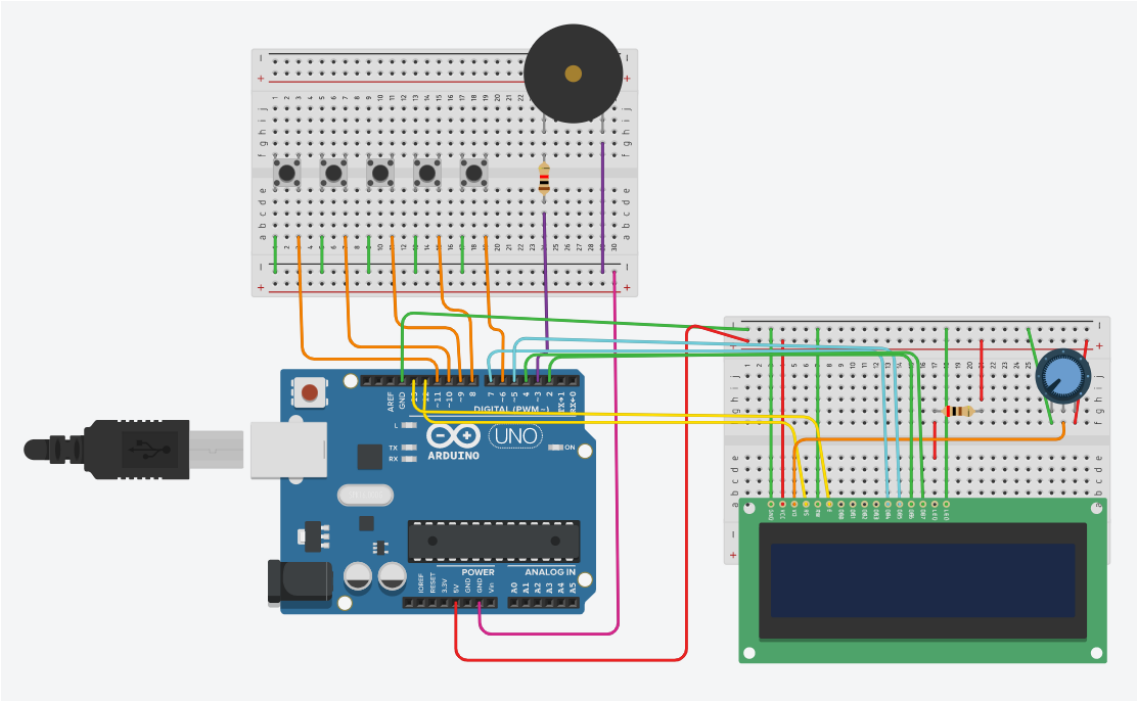
Luego definimos, los dispositivos que utilizaremos, en primer lugar, el megáfono, asignándole el pin 3, luego la pantalla LCD con respectivos pines (12, 13, 7, 5, 4, 2) y por último los botones que vamos a utilizar (B = 6, C = 8, D = 9, E = 10, F = 11).

En la función de inicio setup(), vamos a poner en modo OUTPUT el megáfono o timbre, iniciamos el LCD con sus respectivas filas y columnas, ponemos el cursor en el (0,0) y escribimos “Pulsa un botón” para explicar que se debe hacer en el circuito.

Entonces declaramos los botones a modo INPUT, y los definimos a HIGH, para observar que los botones no están pulsados al principio.

Luego en el bucle loop(), realizamos una serie de bucles while(), donde cada vez que este presionando un botón ya sea (B,C,D,E,F) con la condición [digitalRead(boton_B) == ACTIVADO], utilizando la función tone(), con distintos valores y poniendo por pantalla la nota que se está tocando, por ejemplo, ("Nota: (420, 500).").

Por último, reseteamos el megáfono a través de la función noTone() y volvemos a empezar el bucle.



Enlaces:

https://www.tinkercad.com/things/hrmChBVkJTL

https://www.tinkercad.com/things/k007i1vDsAB