

# Cuadernillo de Prácticas

## Laboratorio de Sistemas Electrónicos Digitales

ÁREA DE TECNOLOGÍA ELECTRÓNICA - URJC

## Parte 1: Probando el hardware

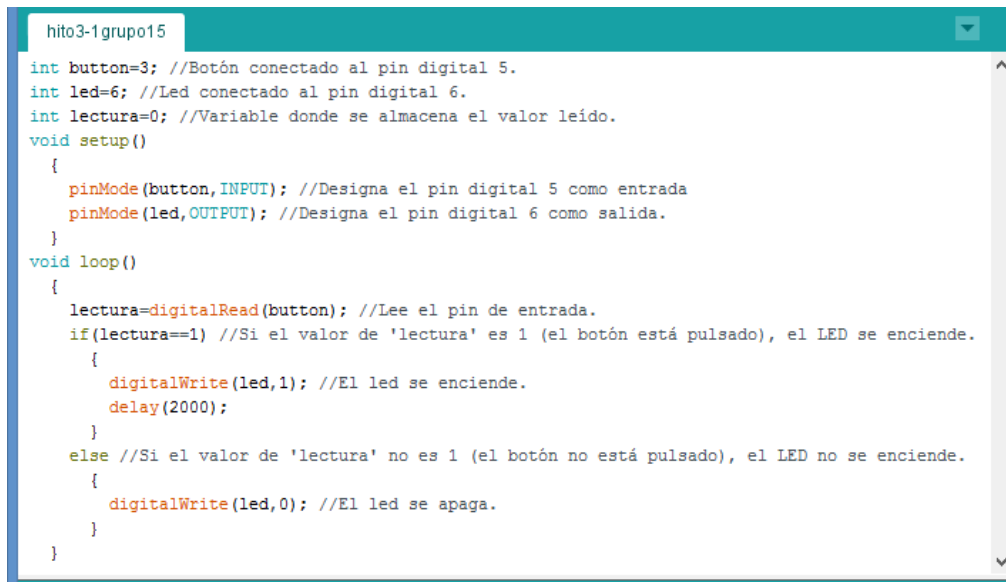
En este capítulo, empezaremos a trabajar con algunos elementos hardware que puedes conectar a Arduino. Después, en la segunda parte te ponemos ejemplos más específicos del kit de las prácticas.

Al finalizar el módulo podrás llegar a desarrollar prototipos electrónicos integrando diferentes elementos de entrada (sensores, entrada analógica, digital, botones, interruptores) y de salida (leds, servos, pantallas). Te recomendamos seguir el libro que viene con el kit para el montaje de los diferentes elementos. Verás que es una placa que se superpone a tu placa Arduino, pero si tienes una Arduino Uno también puedes trabajar sin problemas con este guión.

Esta primera parte sólo debes seguirla, leerla y hacer los bloques. No hay que entregar nada más. La parte 2 es el semáforo digital que es el que tendrás que entregar en clase al profesor y una pequeña entrega a través del campus virtual.

### 2. HITO 2.1

El siguiente es un ejemplo para probar el uso del módulo "button" - botón e interrupción. Este programa enciende el LED cuando se pulsa el botón, si el botón no está pulsado, el LED está apagado.



```
hito3-1grupo15
int button=3; //Botón conectado al pin digital 5.
int led=6; //Led conectado al pin digital 6.
int lectura=0; //Variable donde se almacena el valor leído.
void setup()
{
  pinMode(button,INPUT); //Designa el pin digital 5 como entrada
  pinMode(led,OUTPUT); //Designa el pin digital 6 como salida.
}
void loop()
{
  lectura=digitalRead(button); //Lee el pin de entrada.
  if(lectura==1) //Si el valor de 'lectura' es 1 (el botón está pulsado), el LED se enciende.
  {
    digitalWrite(led,HIGH); //El led se enciende.
    delay(2000);
  }
  else //Si el valor de 'lectura' no es 1 (el botón no está pulsado), el LED no se enciende.
  {
    digitalWrite(led,LOW); //El led se apaga.
  }
}
```

### 2. HITO 2.2

El siguiente es un ejemplo para controlar el brillo de 2 leds diferentes. Una vez realizado cambia el código para que te funcione con 3 leds.

```
hito3-2grupo15
int brightness=0;
int brightness2=0;
int fadeAmount=5;

void setup()
{
  pinMode(6,OUTPUT); //Led 1.
  pinMode(4,OUTPUT); //Led 2.
}

void loop()
{
  analogWrite(6,brightness);
  analogWrite(4,brightness2);

  brightness= brightness + fadeAmount;
  brightness2= brightness2 + fadeAmount;

  if(brightness==0 || brightness==255) //Condición: Si es 0 o 255.
  {
    fadeAmount = -fadeAmount; //Cambia de sentido.
  }
  if(brightness2==0 || brightness2==255)
  {
    fadeAmount = -fadeAmount;
  }

  delay(30);
}
```

3.

## HITO 2.3

El siguiente es un ejemplo para que permita usar el módulo "buzzer" – que básicamente permite emitir sonido desde tu placa Arduino (p.e. tipo alarma o música) conectando el buzzer a la placa de desarrollo.

```
int speakerPin = 9;

int length = 15; // the number of notes
char notes[] = "ccggaagffeeddc "; // a space represents a rest
int beats[] = { 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 2, 4 };
int tempo = 300;

void playTone(int tone, int duration) {
  for (long i = 0; i < duration * 1000L; i += tone * 2) {
    digitalWrite(speakerPin, HIGH);
    delayMicroseconds(tone);
    digitalWrite(speakerPin, LOW);
    delayMicroseconds(tone);
  }
}

void playNote(char note, int duration) {
  char names[] = { 'c', 'd', 'e', 'f', 'g', 'a', 'b', 'C' };
  int tones[] = { 1915, 1700, 1519, 1432, 1275, 1136, 1014, 956 };

  // play the tone corresponding to the note name
  for (int i = 0; i < 8; i++) {
    if (names[i] == note) {
      playTone(tones[i], duration);
    }
  }
}

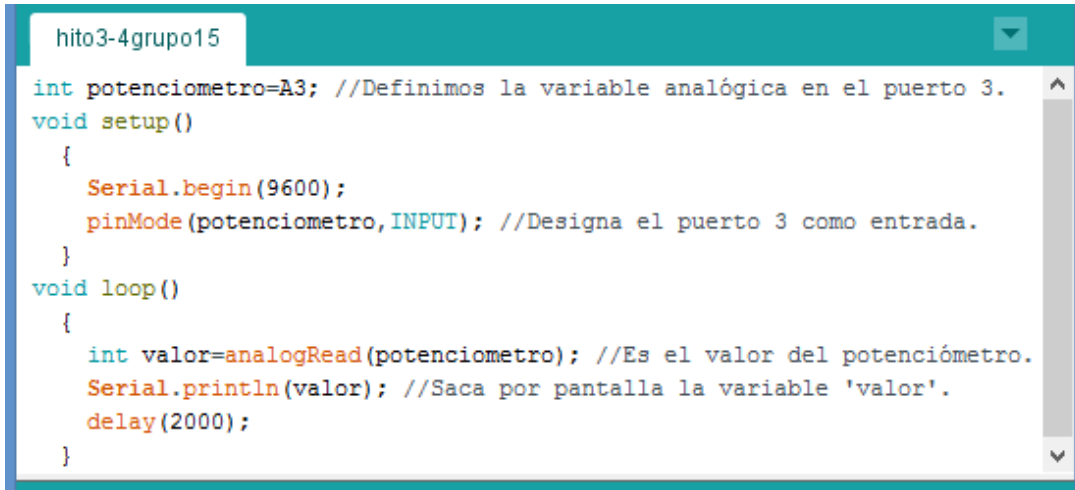
void setup() {
  pinMode(speakerPin, OUTPUT);
}
```

```
void loop() {  
  for (int i = 0; i < length; i++) {  
    if (notes[i] == ' ') {  
      delay(beats[i] * tempo); // rest  
    } else {  
      playNote(notes[i], beats[i] * tempo);  
    }  
  
    // pause between notes  
    delay(tempo / 2);  
  }  
}
```

## 4. HITO 2.4

---

El siguiente es un ejemplo para conectar el sensor "Rotary angle sensor" que permite darte información sobre los grados de giro escalados al potenciómetro. Una vez que consigas que te funcione este programa, puedes incluir un sensor cómo de apertura de puertas o similar, y con el mismo programa conectando en el mismo pin de entrada tendrás el valor del ángulo de la apertura de la puerta



```
hito3-4grupo15  
  
int potenciometro=A3; //Definimos la variable analógica en el puerto 3.  
void setup()  
{  
  Serial.begin(9600);  
  pinMode(potenciometro, INPUT); //Designa el puerto 3 como entrada.  
}  
void loop()  
{  
  int valor=analogRead(potenciometro); //Es el valor del potenciómetro.  
  Serial.println(valor); //Saca por pantalla la variable 'valor'.  
  delay(2000);  
}
```

## 5. HITO 2.5

---

El siguiente es un ejemplo para usar el Sensor de sonido - "Sound Sensor". Prueba a emitir diferentes tonos de voz, más agudos, graves, finos, e incluso música desde tu móvil para que vayas viendo que los datos que aparecen en la pantalla van variando en función del tipo de sonido.

```
hito3-5grupo15
int ledazul=4; //Led conectado al pin digital 4.
int microfono=600; //El micrófono tiene un valor de 60 a partir
//del cual funcionará el led.

void setup()
{
  pinMode(ledazul,OUTPUT); //Designa el pin digital 4 como salida.
}
void loop()
{
  int valorsensor=analogRead(A3); //Lee el valor que le llega al sensor.
  if(valorsensor>microfono) //Condición: Si el valor del sensor es mayor
    //que el valor del micrófono.
  {
    digitalWrite(ledazul,HIGH); //El led azul se enciende.
    delay(3000);
    digitalWrite(ledazul,LOW); //El led se apaga.
  }
  else //Si el valor del sensor es menor que el valor del micrófono.
  {
    digitalWrite(ledazul,LOW); //El led permanece apagado.
  }
}
```

## 6. HITO 2.6

---

El siguiente es un ejemplo para utilizar el Sensor de Luz - "Light Sensor". En función de la luz ambiente que haya en la habitación variará el valor que salga por la pantalla, puedes probar a encender la luz, bajar las persianas, etc., para ver los diferentes valores. Verás que en función de la luz el valor es distinto. Realiza un programa, que en función del rango que visualice de valores te indique si hay baja luz, media luz o mucha luz.

```
hito3-6grupo15
int ledazul=4; //Led conectado al pin digital 5.
int valorluz=50; //La luz tiene un valor de 500 hasta el cual el
//led funcionará.

void setup()
{
  pinMode(ledazul,OUTPUT); //Designa el pin digital 5 como salida.
}
void loop()
{
  int valorsensor=analogRead(5); //Lee el valor del sensor.
  if(valorsensor<valorluz) //Condición: Si el valor del sensor es
    //menor que el valor de la luz.
  {
    digitalWrite(ledazul,HIGH); //El led se enciende.
    delay(3000);
    digitalWrite(ledazul,LOW);
  }
  else //Si el valor del sensor es mayor que el valor de la luz.
  {
    digitalWrite(ledazul,LOW); //El led se apaga.
  }
}
```

## 7. HITO 2.7

El siguiente es un ejemplo para utilizar el Sensor de temperatura - "Temperature Sensor". Deberías comprobar cuál es la fórmula para visualizar los grados en centígrados o Fahrenheit. Cambia el programa para indicar el valor de la temperatura en las dos escalas.

```
hito3-7grupo15

int a; //Parámetro del sensor
int b=3975; //Parámetro del sensor.
float temperatura;
float resistencia;
void setup()
{
    Serial.begin(9600);
}
void loop()
{
    a=analogRead(A1); //Lee el valor del sensor (medición en función de los parámetros)
    resistencia=(float) (1023-a)*10000/a;
    temperatura=1/(log(resistencia/10000)/b+1/298.15)-273.15; //Convierte la temperatura del sensor en grados Celsius
    delay(1000);
    Serial.println(temperatura);
}
```

## 8. HITO 2.8

El siguiente es un ejemplo para el uso de un servo - "Servo". Básicamente este ejemplo permitirá que se vaya moviendo muy poco a poco el dispositivo que conectes que viene en el kit. Te recomendamos que como pone en el ejemplo conectes a la placa de este ejemplo el potenciómetro para que en función del valor registrado gire lo que indiques con el potenciómetro. Puedes también cambiar el programa para que haya distintos rangos en la operación de giro en función de lo que leas desde este sensor.

Otro ejemplo que puedes hacer es que además de girar, con el dispositivo de sonido, puedas emitir sonidos más o menos atenuados en función de los valores registrados en este hito.

```
hito3-8grupo15
#include <Servo.h>
Servo groveServo;

int potenciometro=0;
int shaft;

void setup()
{
  groveServo.attach(3);
  pinMode(potenciometro, INPUT);
}

void loop()
{
  shaft = analogRead(potenciometro);
  shaft = map(shaft, 0, 500, 0, 179);
  groveServo.write(shaft);
  delay(10);
}
```

## Parte 2

### 3.1. NOMBRE DEL SEMÁFORO

Semáforo inteligente.

A continuación, se indica el diagrama de bloques del sistema que deberás implementar para que haga las veces de un Semáforo de peatones. Deberás implementar el código y la parte hardware para que cumpla con los requisitos de cada uno de los estados. Lee atentamente cada apartado

El apartado 3.2 describe los diferentes estados que debe implementar tu semáforo inteligente.

Después, en el apartado 3.3 (DISEÑO DEL SISTEMA), se describirán los componentes hardware que necesitarás y detalles de la implementación de cada estado. El apartado 3.4 describe el código a alto nivel que debes realizar. El apartado 3.5 el montaje hardware de ejemplo. Y el apartado 6 lo que debes entregar.

### 3.2 . DIAGRAMA DE BLOQUES

El semáforo empezará con el estado 1 y acabará realizando los pasos que se describen a continuación.

#### ESTADO 1



#### ESTADO 2

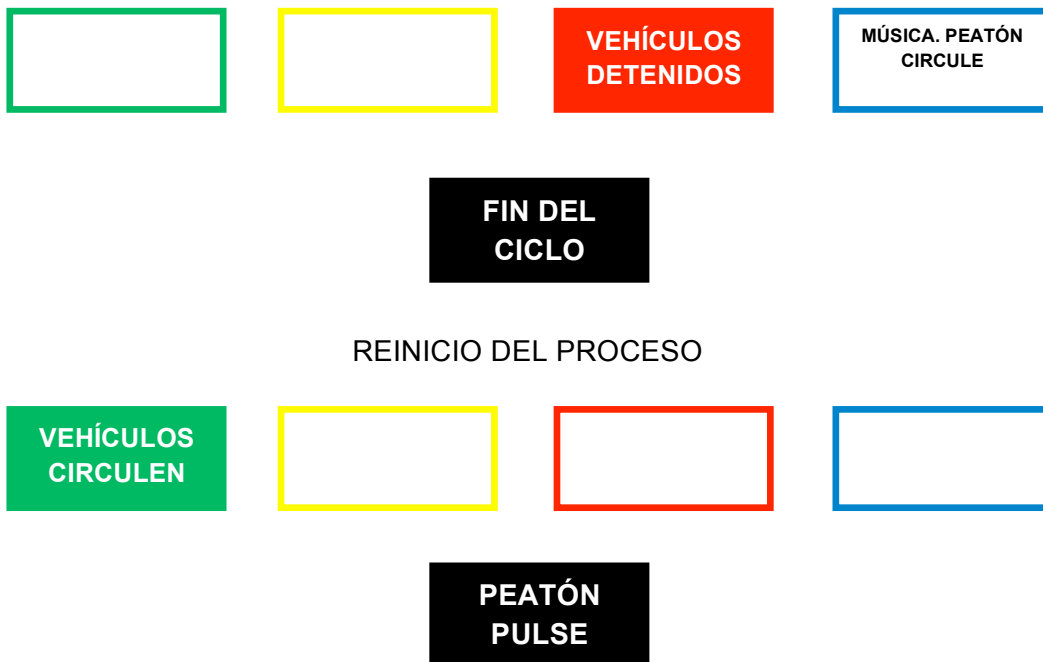


#### ESTADO 3





#### ESTADO 4



### **3.3. DISEÑO DEL SISTEMA**

---

El semáforo inteligente consta de un botón, un buzzer y cuatro leds (rojo, amarillo, verde y azul).

El semáforo diseñado sigue la siguiente secuencia. Inicialmente, sólo se encuentra encendido el led verde, dando paso a los coches (ESTADO 1).

Al pulsar el botón el sistema se activará, tras dos segundos se apagará el verde y se encenderá el azul indicando que el peatón debe esperar. Simultáneamente, se enciende el led amarillo parpadeando (intermitente) que indica a los vehículos que reduzcan su velocidad (ESTADO 2).

Posteriormente, el azul permanecerá encendido indicando 'Peatón espere' mientras que el amarillo dejará de parpadear y se quedará 'Amarillo fijo' durante cuatro segundos. Esto último servirá para indicar a los vehículos que deben detenerse porque está a punto de ponerse en rojo (ESTADO 3).

Finalmente, los leds de color amarillo y azul se apagarán, entonces el led rojo se enciende indicando que los vehículos deben estar detenidos y suena la melodía indicando que los peatones pueden cruzar. Por último, se apagará el led rojo (ESTADO 4).

Sobre el tema de la melodía. Un sonido no es más que una vibración del aire que nuestros oídos pueden captar. Un sonido que tiene un determinado tono, depende de la frecuencia a la cual vibra el aire. Las notas musicales son vibraciones de frecuencias determinadas. Por supuesto, en la creación de música intervienen muchos otros factores complejos, como por ejemplo, el timbre.

Para crear música con Arduino debemos definir una serie de notas, que en inglés se definen mediante letras, así como el tiempo que queremos que dure cada nota. También podemos incluir espacios entre las notas, para meter silencios en nuestra melodía.

Por ello, como verás en lo que tienes que implementar en la siguiente sección, la función suena melodía simplemente recorre todas las notas y llama a PlayNote para tocarlas con la duración adecuada. La función PlayNote asigna la frecuencia adecuada a la nota de entrada que queremos tocar, y llama a PlayTone. La función PlayNote toca la nota adecuada, haciendo vibrar el Buzzer durante el tiempo que dura la nota, con la frecuencia de entrada que define la nota concreta.

Cuando acaba este proceso vuelve a su estado inicial, donde únicamente estará encendido el led verde esperando a que el usuario pulse el botón para que se vuelve a iniciar el sistema.

### 3.4. PSEUDOCÓDIGO

---

A continuación, te indicamos las variables a modo de ejemplo y el pseudocódigo que deberías seguir para la elaboración del semáforo. Si vas a realizar algún cambio adicional deberás documentarlo en la entrega de este bloque.

#### VARIABLES

**Tipo Entero** ledrojo, ledamarillo, ledverde, ledazul, boton, sonido, length, beats[], tempo, posicionboton.

**Carácter** notes[].

#### FUNCIÓN CONFIGURACIÓN

##### Función setup

Inicio

```
Espacio Serial ← 9600;  
Pin digital boton ← INPUT;  
Pin digital ledrojo ← OUTPUT;  
Pin digital ledamarillo ← OUTPUT;  
Pin digital ledverde ← OUTPUT;  
Pin digital ledazul ← OUTPUT;  
Pin digital sonido ← OUTPUT;
```

Fin

#### PROCESOS Y FUNCIÓN CÍCLICA (LOOP)

**Procedimiento** playTone (**entero** tone, duration)

Inicio

**Para** i ← 0 hasta i < duration \* 1000L con paso tone \* 2 hacer

```
sonido ← HIGH;  
retrasarmicrosegundos (tone);  
sonido ← LOW;  
retrasarmicrosegundos (tone);
```

**Fin para**;

Fin

**Procedimiento** playNote (**caracter** note, **entero** duration)

Inicio

```
names[] ← { 'c', 'd', 'e', 'f', 'g', 'a', 'b', 'C' };
tones[] ← { 1915, 1700, 1519, 1432, 1275, 1136, 1014, 956 };

Para i ← 0 hasta i < 8 con paso 1 hacer
    Si (names[i] = note) entonces
        playTone (tones[i], duration);
    Fin si;
Fin para;
```

Fin

**Procedimiento** suenamelodia

Inicio

```
Para i ← 0 hasta i < length con paso 1 hacer
    Si (notes[i] = ' ') entonces
        espera (beats[i] * tempo);
    Si no
        playNote (notes[i], beats[i] * tempo);
    Fin si;
    espera (tempo / 2);
Fin para;
```

Fin

**Función** loop

Inicio

```
ledverde ← HIGH;
```

```

posicionboton ← leer boton;

Si posicionboton = 1 entonces

    espera (2000);

    ledverde ← LOW;

    ledazul ← HIGH;

Para i ← 0 hasta i<11 con paso 1 hacer

    ledamarillo ← HIGH;

    espera (1000);

    ledamarillo ← LOW;

    espera (1000);

Fin para;

ledamarillo ← HIGH;

espera (4000);

ledamarillo ← LOW;

    ledazul ← LOW;

    ledrojo ← HIGH;

    suenamelodia ();

    ledrojo ← LOW;

Fin si;

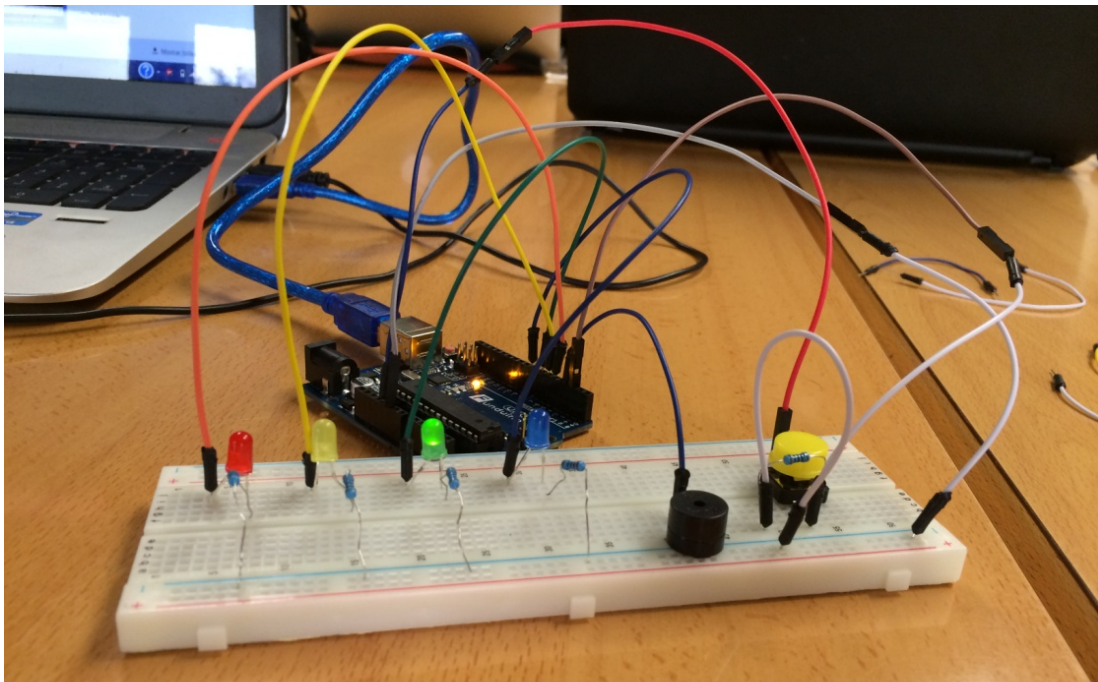
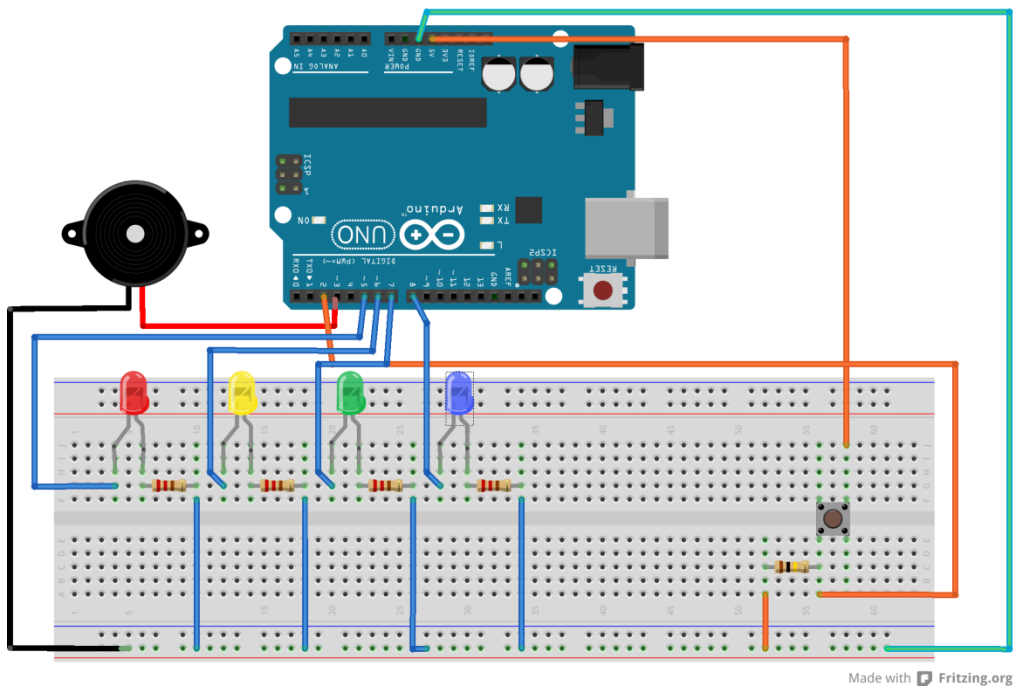
Fin

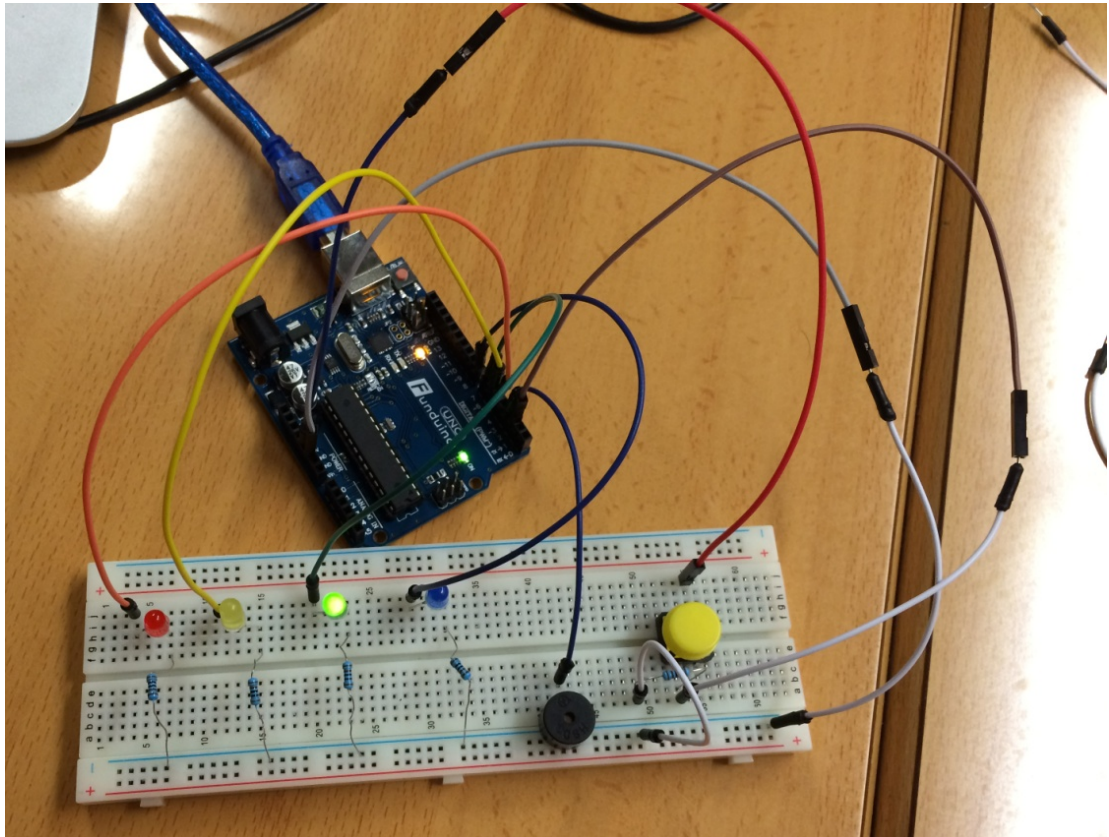
```

### 3.5. EMPEZAMOS EL MONTAJE

Lo primero que deberás hacer después de haber entendido los pasos anteriores, es ponerte a realizar el montaje hardware.

A continuación, te exponemos un ejemplo del montaje del Semáforo Digital acorde a las especificaciones planteadas hasta el 3.4. Si realizas cambios para hacer un semáforo diferente, deberás documentarlo también con imágenes del montaje en la entrega del campus (ver punto 3.6 te explicamos la entrega). Aunque en la imagen aparece una Arduino Uno puedes adaptarlo a tu placa de prácticas sin problemas.





### 3.6. ENTREGA y CÓDIGO FINAL

---

Para que la entrega del semáforo esté aprobada hay que mostrar el funcionamiento del semáforo digital siguiendo las mismas instrucciones que te hemos planteado, o planteando tu propio semáforo digital al profesor en la fecha establecida por el mismo (VER NORMATIVA DE PRÁCTICAS – campus virtual).

Puedes incorporar nuevas funcionalidades como que sólo se ponga en verde si se detecta presencia de peatones (sensor magnético), que dure más o menos el semáforo en función del tiempo, etc.

A parte, se habilitará una entrega en el campus virtual de los resultados de este cuadernillo en el que tienes que respetar la fecha de entrega de lo siguiente.

Si realizas las mismas instrucciones de esta memoria en la entrega sólo debes hacer un .zip con el siguiente contenido y formato:

- Entrega2-NumeroGrupoX-sed.zip,
- Contenido: Debes entregar un fichero .docx que tenga las siguientes secciones denominadas del siguiente modo y que tengan lo siguiente:
  - o Sección 1-Semáforo: La primera página con el nombre y apellidos de los miembros del grupo.
  - o Sección 2-Semáforo: Tabla de Verdad que represente el funcionamiento de tu código y puertas lógicas asociadas.
  - o Sección 3-Semáforo : El código de tu fichero de Arduino copiado dentro del mismo documento.

Si realizas un semáforo diferente al de esta memoria en la entrega debes hacer un .zip con el siguiente contenido y formato:

- NumeroGrupoX-sed.zip,
- Contenido: Debes entregar un fichero .docx que tenga las siguientes secciones:
  - o Sección 1-Semáforo: La primera página con el nombre y apellidos de los miembros del grupo.
  - o Sección 2-Semáforo: Tabla de Verdad que represente el funcionamiento de tu código.
  - o Sección 3-Semáforo: Diagrama de estados / bloque para explicar el funcionamiento de tu bloque y la descripción que consideres para explicar cómo funciona.
  - o Sección 4-Semáforo: El código de tu fichero de Arduino copiado dentro del mismo documento.
  - o [optativo] Sección 5-Semáforo: Un enlace a un video que demuestre su funcionamiento (o bien, fotos de su funcionamiento en ese fichero).

(NOTA: DESPUÉS DE LA SESIÓN 2 ESTE ZIP TENDRÁ NUEVOS APARTADOS, SU ENTREGA SE HABILITARÁ DESPUÉS DE QUE FINALICE LA SESIÓN 2 DEL



LABORATORIO TENIENDO HASTA LA SESIÓN 3 PARA SU ENTREGA POR EL CAMPUS VIRTUAL).

## ANEXO – EJEMPLO PRÁCTICO

---

ESTE ES SÓLO UN EJEMPLO, puedes realizarlo o no. **NO hay que entregar nada al profesor.** En este ejemplo, vamos a hacer una “Atracción Mario Bros” con Arduino.

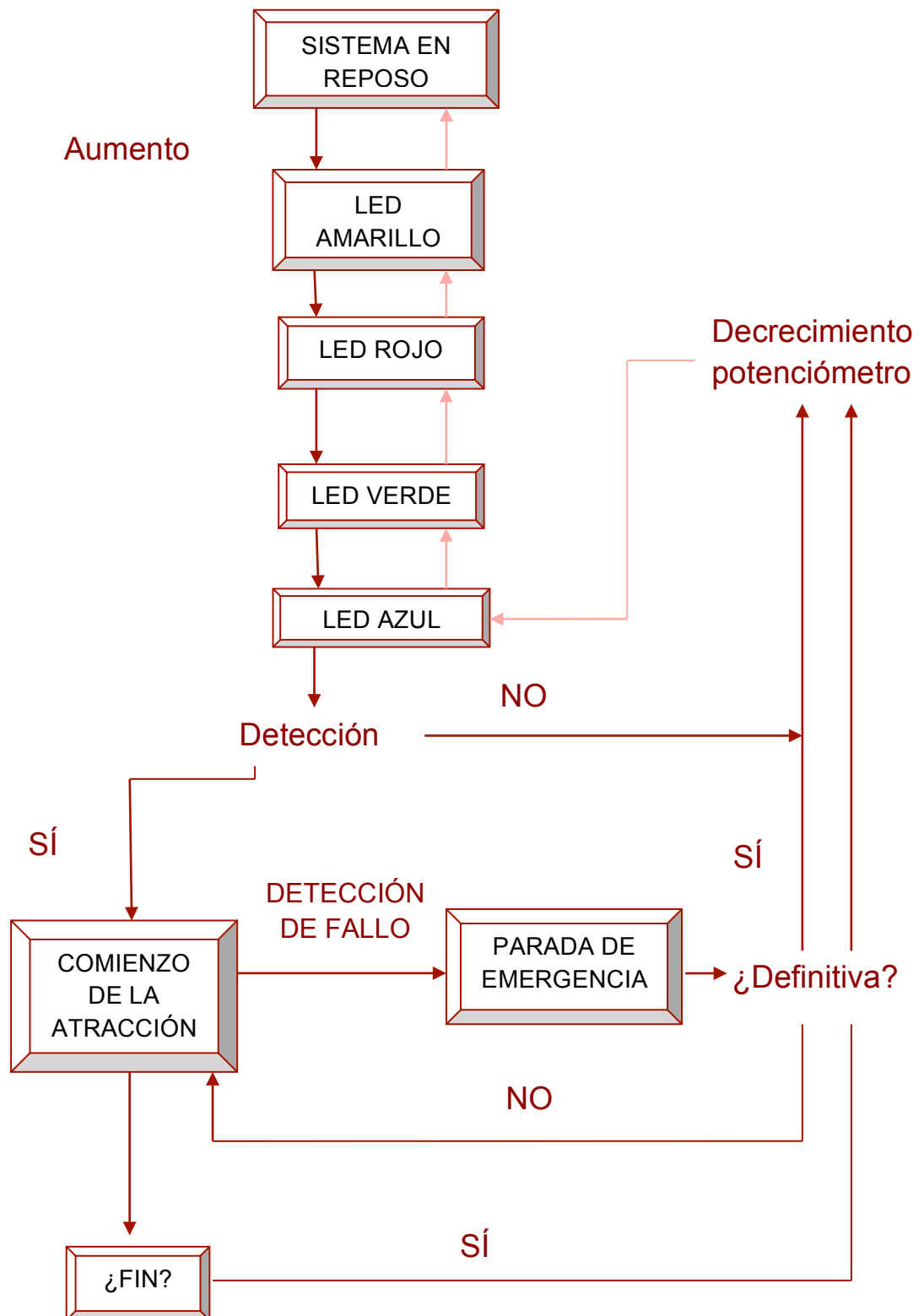
### Diseño del sistema

En La Atracción Mario Bros está compuesta por un buzzer, un potenciómetro, un sensor de luz, un botón, un led amarillo, un led verde, un led rojo y un led azul.

Hemos diseñado una atracción cuyo funcionamiento es el siguiente: Se activa girando la rueda del potenciómetro hasta que se enciendan todas las luces. Una vez que se enciende la última luz, se activa el sensor de luz y empieza a sonar la canción del videojuego de Mario Bros y da comienzo a la atracción.

Por si ocurre algún percance, hemos diseñado un sistema de emergencia en el que pulsando un botón se detiene la atracción. Si se desea que la atracción para por completo, mientras se pulsa el botón debe girarse en sentido contrario el potenciómetro. Si lo que se desea es una simple parada, se vuelve a poner en marcha dejando de pulsar el botón.

## EJEMPLO DE POSIBLE Diagrama de bloques sobre el funcionamiento



## Pseudocódigo

### VARIABLES

**A continuación, te ponemos las diferentes variables y constantes que debes configurar para llevar a cabo este hito:**

**Entero** led1, led2, led3, led4, pinLDR, valorLDR, boton, melody[], tempo[], valor, song, size, noteDuration, pauseBetweenNotes, thisNote

definir NOTE\_B0 31

definir NOTE\_C1 33

definir NOTE\_CS1 35

definir NOTE\_D1 37

definir NOTE\_DS1 39

definir NOTE\_E1 41

definir NOTE\_F1 44

definir NOTE\_FS1 46

definir NOTE\_G1 49

definir NOTE\_GS1 52

definir NOTE\_A1 55

definir NOTE\_AS1 58

definir NOTE\_B1 62

definir NOTE\_C2 65

definir NOTE\_CS2 69

definir NOTE\_D2 73

definir NOTE\_DS2 78

definir NOTE\_E2 82

definir NOTE\_F2 87

definir NOTE\_FS2 93

definir NOTE\_G2 98

definir NOTE\_GS2 104

definir NOTE\_A2 110

definir NOTE\_AS2 117  
definir NOTE\_B2 123  
definir NOTE\_C3 131  
definir NOTE\_CS3 139  
definir NOTE\_D3 147  
definir NOTE\_DS3 156  
definir NOTE\_E3 165  
definir NOTE\_F3 175  
definir NOTE\_FS3 185  
definir NOTE\_G3 196  
definir NOTE\_GS3 208  
definir NOTE\_A3 220  
definir NOTE\_AS3 233  
definir NOTE\_B3 247  
definir NOTE\_C4 262  
definir NOTE\_CS4 277  
definir NOTE\_D4 294  
definir NOTE\_DS4 311  
definir NOTE\_E4 330  
definir NOTE\_F4 349  
definir NOTE\_FS4 370  
definir NOTE\_G4 392  
definir NOTE\_GS4 415  
definir NOTE\_A4 440  
definir NOTE\_AS4 466  
definir NOTE\_B4 494  
definir NOTE\_C5 523  
definir NOTE\_CS5 554  
definir NOTE\_D5 587

definir NOTE\_DS5 622  
definir NOTE\_E5 659  
definir NOTE\_F5 698  
definir NOTE\_FS5 740  
definir NOTE\_G5 784  
definir NOTE\_GS5 831  
definir NOTE\_A5 880  
definir NOTE\_AS5 932  
definir NOTE\_B5 988  
definir NOTE\_C6 1047  
definir NOTE\_CS6 1109  
definir NOTE\_D6 1175  
definir NOTE\_DS6 1245  
definir NOTE\_E6 1319  
definir NOTE\_F6 1397  
definir NOTE\_FS6 1480  
definir NOTE\_G6 1568  
definir NOTE\_GS6 1661  
definir NOTE\_A6 1760  
definir NOTE\_AS6 1865  
definir NOTE\_B6 1976  
definir NOTE\_C7 2093  
definir NOTE\_CS7 2217  
definir NOTE\_D7 2349  
definir NOTE\_DS7 2489  
definir NOTE\_E7 2637  
definir NOTE\_F7 2794  
definir NOTE\_FS7 2960

```

definir NOTE_G7 3136
definir NOTE_GS7 3322
definir NOTE_A7 3520
definir NOTE_AS7 3729
definir NOTE_B7 3951
definir NOTE_C8 4186
definir NOTE_CS8 4435
definir NOTE_D8 4699
definir NOTE_DS8 4978
definir melodyPin 7

```

```

entero melody[] =

```

```

    Inicio

```

```

        NOTE_E7, NOTE_E7, 0, NOTE_E7
        0, NOTE_C7, NOTE_E7, 0,
        NOTE_G7, 0, 0, 0,
        NOTE_G6, 0, 0, 0,

```

```

        NOTE_C7, 0, 0, NOTE_G6,
        0, 0, NOTE_E6, 0,
        0, NOTE_A6, 0, NOTE_B6,
        0, NOTE_AS6, NOTE_A6, 0,

```

```

        NOTE_G6, NOTE_E7, NOTE_G7,
        NOTE_A7, 0, NOTE_F7, NOTE_G7,
        0, NOTE_E7, 0, NOTE_C7,
        NOTE_D7, NOTE_B6, 0, 0,

```

```

        NOTE_C7, 0, 0, NOTE_G6,
        0, 0, NOTE_E6, 0,

```

0, NOTE\_A6, 0, NOTE\_B6,  
0, NOTE\_AS6, NOTE\_A6, 0,

NOTE\_G6, NOTE\_E7, NOTE\_G7,  
NOTE\_A7, 0, NOTE\_F7, NOTE\_G7,  
0, NOTE\_E7, 0, NOTE\_C7,  
NOTE\_D7, NOTE\_B6, 0, 0

Fin

**entero** tempo[] =

Inicio

12, 12, 12, 12,  
12, 12, 12, 12,  
12, 12, 12, 12,  
12, 12, 12, 12,

12, 12, 12, 12,  
12, 12, 12, 12,  
12, 12, 12, 12,  
12, 12, 12, 12,

9, 9, 9,  
12, 12, 12, 12,  
12, 12, 12, 12,  
12, 12, 12, 12,

12, 12, 12, 12,  
12, 12, 12, 12,  
12, 12, 12, 12,



12, 12, 12, 12,

9, 9, 9,

12, 12, 12, 12,

12, 12, 12, 12,

12, 12, 12, 12,

Fin

## **FUNCIÓN CONFIGURACIÓN**

### **Función setup**

Inicio

Espacio Serial  $\leftarrow$  9600;

Pin digital 7  $\leftarrow$  OUTPUT;

Pin digital led1  $\leftarrow$  OUTPUT;

Pin digital led2  $\leftarrow$  OUTPUT;

Pin digital led3  $\leftarrow$  OUTPUT;

Pin digital led4  $\leftarrow$  OUTPUT;

Interrupción  $\leftarrow$  RISING;

Fin

## **FUNCIÓN CÍCLICA**

### **Función loop**

Inicio

**entero** valor  $\leftarrow$  leer A0;

Imprimir pantalla  $\leftarrow$  valor;

valorLDR  $\leftarrow$  leer pinLDR;

Imprimir pantalla  $\leftarrow$  valorLDR;

**Si** valor  $\geq$  250 entonces

led1 ← HIGH;

Si no

led1 ← LOW;

Fin si;

Si valor ≥ 500 entonces

led2 ← HIGH;

Si no

led2 ← LOW;

Fin si;

Si valor ≥ 750 entonces

led3 ← HIGH;

Si no

led3 ← LOW;

Fin si;

Si valor ≥ 900 entonces

led4 ← HIGH;

Si no

led4 ← LOW;

Fin si;

espera (100);

Si valorLDR > 256 entonces

song(1);

Fin si;

espera (200);

Fin

**Procedimiento song (entero s)**

Inicio

song ← s;

**entero** size  $\leftarrow$  sizeof melody;

**Para** thisNote  $\leftarrow$  0 hasta thisNote < size con paso 1 hacer

**entero** noteDuration  $\leftarrow$  1000 / tempo [thisNote];

    buzz (melodyPin, melody [thisNote], noteDuration);

**entero** pauseBetweenNotes  $\leftarrow$  noteDuration \* 1.30;

    espera (pauseBetweenNotes);

    buzz (melodyPin, 0, noteDuration);

**Fin para**;

Fin

**Procedimiento** buzz (**entero** targetPin, longitud frequency, longitud length)

Inicio

    13  $\leftarrow$  HIGH;

    longitud delayValue  $\leftarrow$  1000000 / frequency / 2;

    longitud numCycles  $\leftarrow$  frequency \* length / 1000;

**Para** i  $\leftarrow$  0 hasta i < numCycles con paso 1 hacer

        targetPin  $\leftarrow$  HIGH;

        espera (delayValue);

        targetPin  $\leftarrow$  LOW;

        espera (delayValue);

**Fin para**;

    13  $\leftarrow$  LOW;

Fin

**Procedimiento** interr

Inicio

    entero thisNote  $\leftarrow$  0;

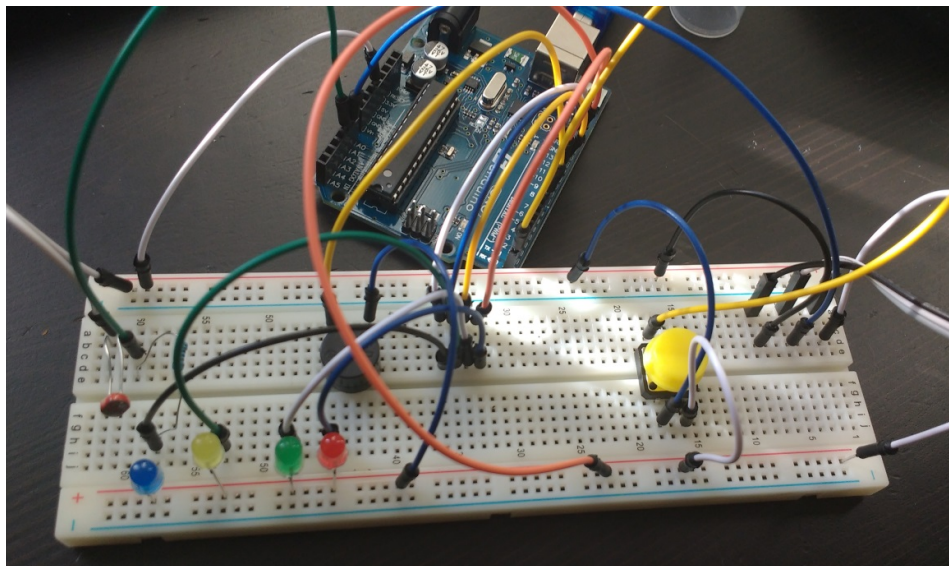
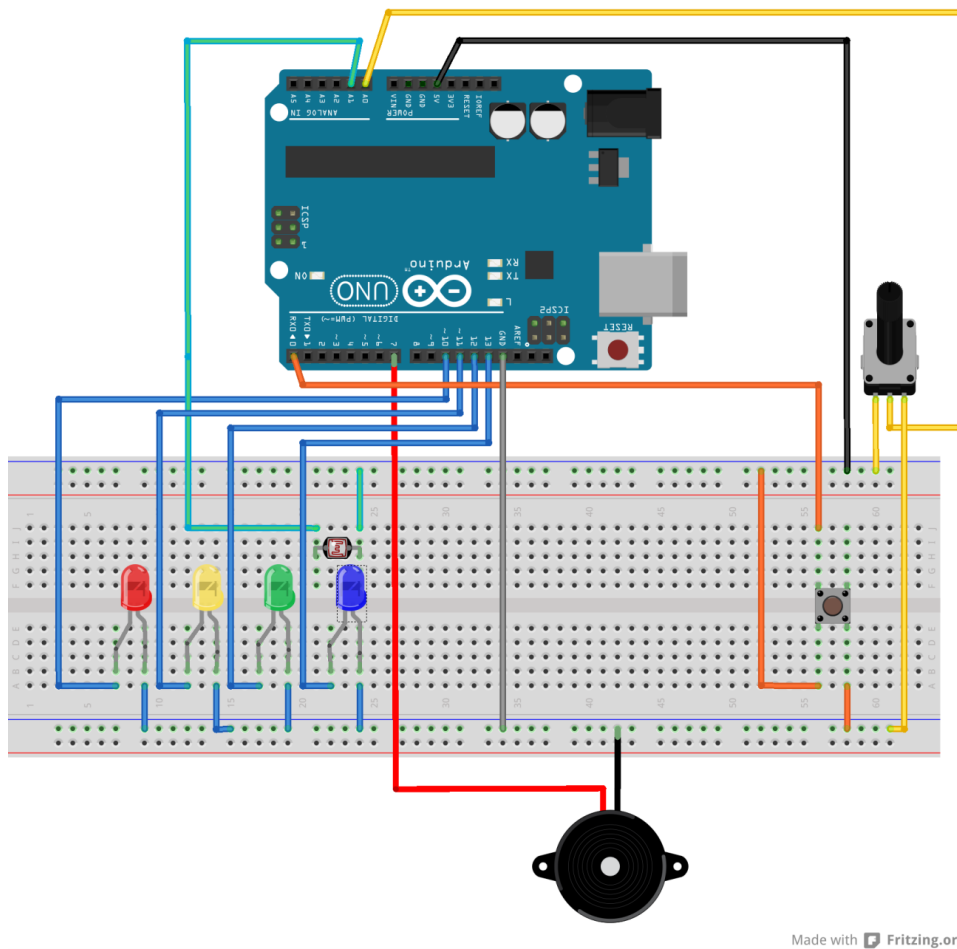
    entero noteDuration  $\leftarrow$  100 / tempo [thisNote];

    buzz(melodyPin, melody[thisNote], noteDuration);

    buzz(melodyPin, 0, noteDuration);

Fin

## Ejemplo de la disposición hardware final



**Video del sistema final (al final de los bloques de prácticas el profesor colgará la solución).**

[https://www.youtube.com/watch?v=a\\_-rJ-Lwg-M](https://www.youtube.com/watch?v=a_-rJ-Lwg-M)

