# Level crossing with barriers


## 1. Introducción

En el proyecto que presentamos vamos a realizar la simulación de un paso a nivel con barreras en una carretera, el cual será activado al detectar la proximidad de un tren que circula por las vías.

En este proyecto vamos a combinar la programación del robot Ozobot con el que simularemos el tren, y por otro lado, la programación de una placa arduino para simular la parte física de las barreras.

El principal motivo de la realización del presente Proyecto radica en el profundo interés en las capacidades de Arduino como herramienta de control electrónico. El manejo de Arduino implica una gran reducción de tiempo en diseño electrónico y en la programación de su microcontrolador.

Sus características hacen de él una oportunidad de negocio en cuanto a su implementación en los sectores de la automatización y las comunicaciones industriales, a fin de una futura dedicación laboral en el marco de estos sectores industriales.

# 2. Objetivos

Los objetivos de este proyecto los agrupamos en:

Objetivos Generales:

- Fomentar las habilidades sociales.
- Realizar bocetos y utilizar herramientas adecuadas.
- Familiarización con Arduino
- Familiarización con el Robot Ozobot

Objetivos Específicos:

- Saber realizar conexiones físicas.
- Realizar diagramas de circuitos.
- Medir distancias a través del efecto rebote del sonido.
- Operar en sistema sexagesimal.
- Programar autómatas digitales

# 3. Componentes

Para la realización del proyecto necesitaremos:

- Placa de Arduino Uno
- Robots Ozobots
- Rotuladores diseño Ozobots
- BreadBoard PCB.
- 2 servo Motores
- 1 Sensor ultrasonido
- 12 cables dupont

# 4. Conocer Ozobot

Ozobot es un pequeño robot inteligente diseñado para el aprendizaje del lenguaje STEM. Este pequeño robot es capaz de leer su programa de las líneas de color dibujadas en un papel.

En nuestro caso usaremos el código Rojo-Negro-Rojo para conseguir reducir la velocidad, y por otro lado usaremos los códigos Azul-Negro-Azul para acelerar y así aumentar la velocidad.

# 5. Conocer Arduino

En este proyecto vamos a trabajar con la placa Arduino UNO, concretamente el Modelo R3.

***Hardaware***:

Las características principales de esta placa es que tiene tres partes claramente diferenciadas. La primera parte hace referencia a la energía o power. Dicha energía tendrá valores de 5V, 3.3V además de su correspondiente conexión GND o Masa. La segunda parte la componen 13 pines digitales, que como su nombre indica, hacen la simulación de código binario (0 ó 1) y finalmente tendremos 5 pines analógicos que funcionan de forma similar a las digitales por lo que su montaje es similar pero estas pueden tomar valores de tensión entre 0V y 5V.

**Software:**

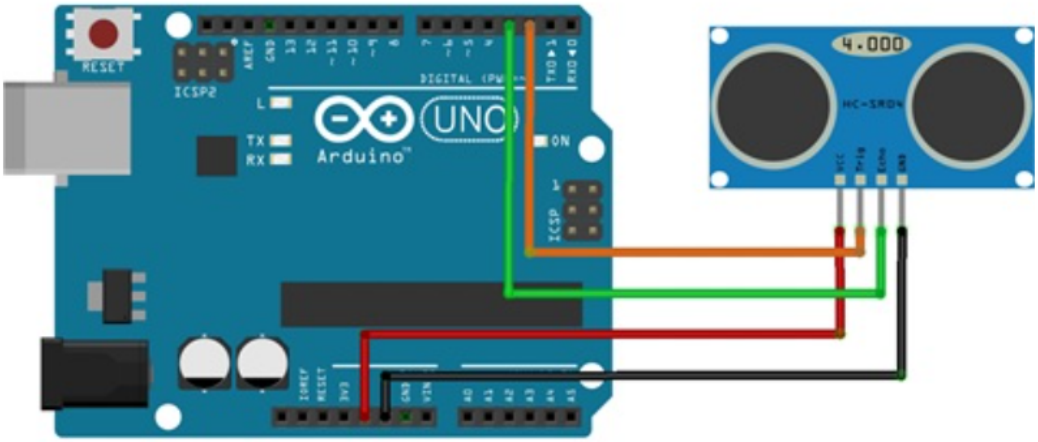
Para configurar Arduino trabajaremos desde el entorno de programación o IDE de Arduino. Se componen de dos funciones principales que son Setup() y Loop().

La función **Setup** será la encargada de inicializar los parámetros y pines con la configuración predeterminada. La función **Loop** se repetirá indefinidamente. Aquí configuraremos y programaremos lo que necesitemos realizar.

# 6. Fase mecánica

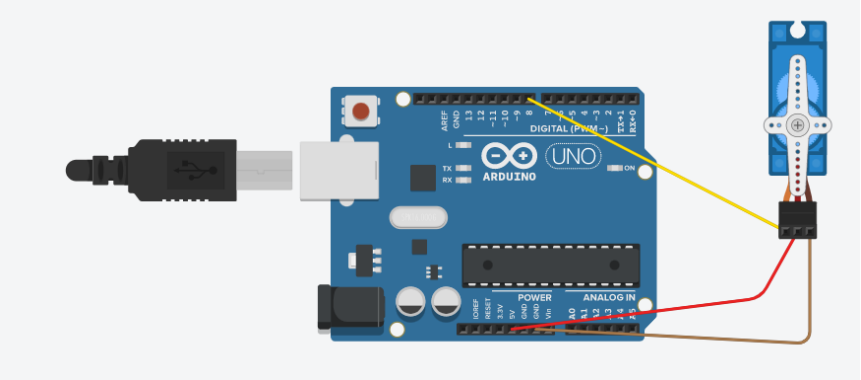
Creación de Grafcet como autoayuda para clarificar los pasos de montaje del proyecto.

Comenzamos nuestro proyecto con la conexión del sensor de ultrasonidos.




Como podemos comprobar necesitamos 4 conexiones. Los extremos serán Power, concretamente el polo positivo irá a 5V y el negativo conectado a GND. Por otro lado necesitamos el pin Trigger que será de salida ya que nos mandará impulsos de corriente y por otro lado Echo será el pin de entrada con el que podemos obtener el pulso.

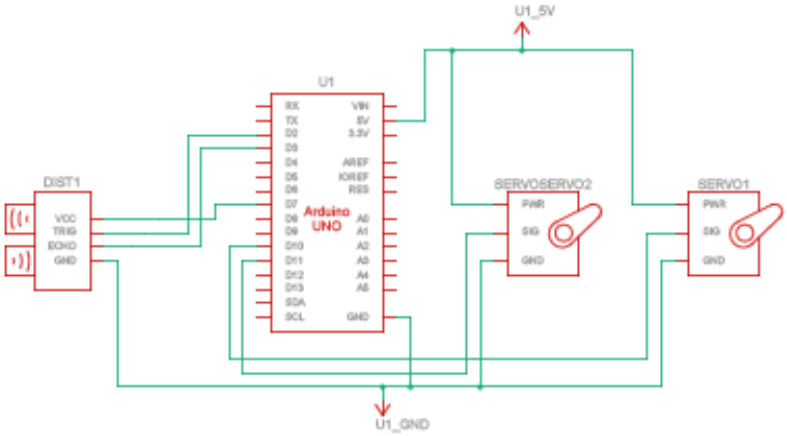
A continuación vamos a explicar el montaje de los *servos*



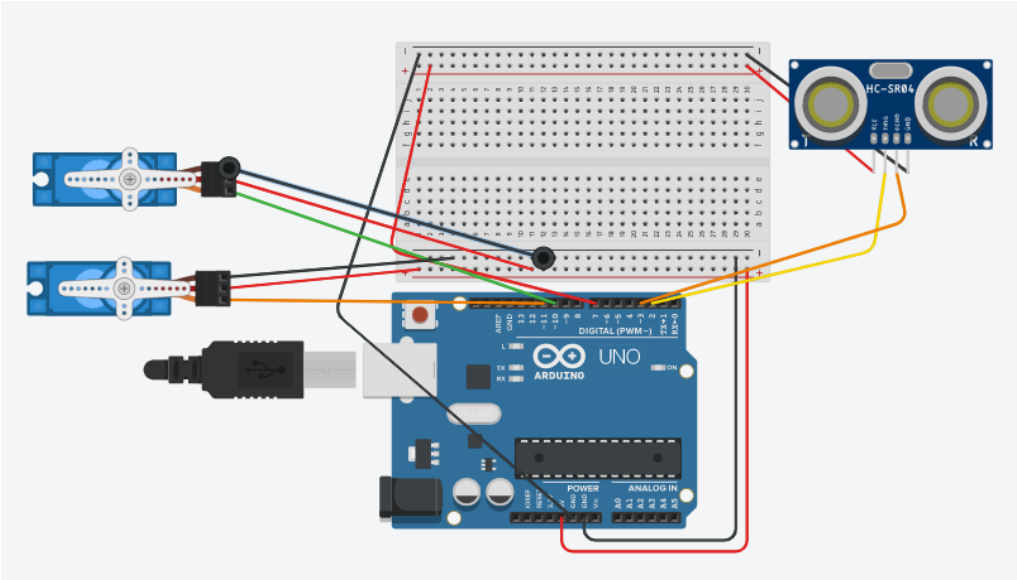

Como podemos ver necesitaremos tres cables para el funcionamiento del servo. Dos de ellos se usarán para energía (un extremo a 5V y otro a GND), mientras que el cable restante irá conectado a un pin digital para poder programar el ángulo de giro.

Recordamos que un servo no es más que un motor DC (de corriente continua o direct current) con reductora que sólo puede girar 180 grados. Se controla mediante el envío de impulsos eléctricos de Arduino. Estos pulsos le dicen al servo a qué posición se deben mover.

Nuestro proyecto mezclará ambos conceptos para conseguir hacer funcionar los servos a partir de lo que detectemos desde el sensor ultrasonido. El esquema de montaje quedaría de la siguientes forma:



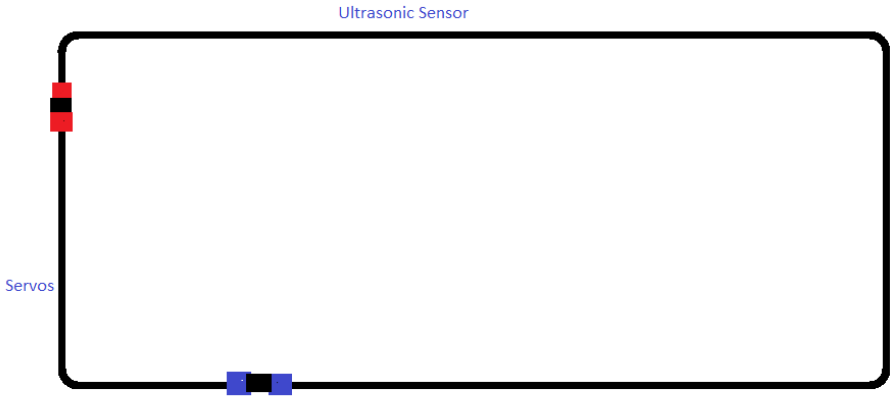

Si lo vemos como componente quedaría:




# 7. Fase Lógica: Programación

Como se ha comentado en secciones anteriores, vamos a construir un circuito para un Ozobot que será controlado con sensores de Arduino para poder simular un paso a nivel

con barreras en un cruce. Cuando el Ozobot pase por un zona determinada (que tendremos colocado el sensor de ultrasonido) activaremos los servos para que las barreras se cierren, seguidamente el Ozobot al acercarse al paso a nivel reducirá la velocidad (utilizando los códigos de colores comentados anteriormente). Al pasar el Ozobot las barreras, éstas se volverán a levantar y seguidamente el Ozobot podrá acelerar.




Para la programación del sensor de **ultrasonido**, necesitamos polo positivo y GND. El pin Trigger será la salida y nos mandará impulsos de corriente y por otro lado Echo será el pin de entrada con el que obtenemos el pulso.

Primero configuramos los pines y la comunicación serial a 9800 baudios para poder imprimir por pantalla en el momento deseado los valores que necesitemos ( en nuestro caso la distancia)

```
const int Trigger = 2;   //Pin digital 2 para el Trigger del sensor
const int Echo = 3;   //Pin digital 3 para el echo del sensor

void setup() {
  Serial.begin(9600);//iniciailzamos la comunicación
  pinMode(Trigger, OUTPUT); //pin como salida
  pinMode(Echo, INPUT);  //pin como entrada
  digitalWrite(Trigger, LOW);//Inicializamos el pin con 0
}
```

Como podemos observar el Trigger será pin de salida ya que enviará impulsos, por otro lado el Echo será pin de entrada ya que recogerá el impulso de rebote. Además será lógico comenzar el proyecto con Trigger apagado para que no esté mandando impulsos.

Ahora tenemos que programar la parte del Loop() . Nuestro objetivo en esta parte es enviar impulsos desde el pin Trigger y recogerlos en el pin Echo para calcular el tiempo que tarda en hacer en rebote.

Vamos a envíar un impulso de 10 microsengundos al Trigger del sensor

```
digitalWrite(Trigger, HIGH);
  delayMicroseconds(10);          //Enviamos un pulso de 10us
  digitalWrite(Trigger, LOW);
```

Seguidamente vamos a recoger el pulso de respuesta del Echo usando la función pulseIN(pin, valor).

```
  t = pulseIn(Echo, HIGH); //obtenemos el ancho del pulso
```

Se manda pulso de 10 microsegundos ya que está determinado empíricamente que funciona a partir de esa cantidad. De esta forma almacenamos en t el tiempo que tarda en llegar el eco del ultrasonido, cuanto más tiempo tarde significará que el eco se ha producido más lejos.

Para calcular la distancia partimos de la fórmula $Velocidad = \frac{espacio}{tiempo}$

La velocidad la conocemos ya que será la del sonido 340m/s, nos interesa medir unidades pequeñas por tanto debemos trabajar en cm y como el impulso se manda en µs será la unidad de tiempo que usaremos.

$v = \frac{d}{t}$ Como sabemos al ser la distancia de rebote, consideramos la distancia doble(ida y vuelta) para un tiempo determinado.

La fórmula será $340\frac{m}{s} = \frac{2\,d}{t}$ por tanto si despejamos la distancia ( que será el valor que nos interesa calcular con el sensor será $d = \frac{340\frac{m}{s} \cdot t}{2}$.

Es importante tener en cuenta que el tiempo va expresado en µs y la distancia nos interesa que esté expresada en cm ( ya que vamos a medir distancias pequeñas).

$$170\ \frac{m}{s} \cdot \frac{100\ cm}{1m} \cdot \frac{1s}{10^6 \mu s} = \frac{17}{1000}\ \frac{cm}{\mu s} = \frac{1}{59}\ \frac{cm}{\mu s}$$

Tras realizar el factor de conversión la distancia expresada en centímetros dependiendo del valor tiempo expresado en microsegundos se calculará :

$$d = \frac{1}{59} \cdot t\ \ cm$$

Finalmente enviamos al puerto serie (para poder imprimir) el valor de la distancia y hacemos una pausa de 100 ms que será superior a los 60 ms que los fabricantes de ultrasonido recomiendan que se haga entre medida y medida.

El código de arduino quedaría :

```
const int Trigger = 2;   //Pin digital 2 para el Trigger del sensor
const int Echo = 3;   //Pin digital 3 para el Echo del sensor

void setup() {
  Serial.begin(9600);//iniciailzamos la comunicación
  pinMode(Trigger, OUTPUT); //pin como salida
  pinMode(Echo, INPUT);  //pin como entrada
  digitalWrite(Trigger, LOW);//Inicializamos el pin con 0
}

void loop()
{

  long t; //timepo que demora en llegar el eco
  long d; //distancia en centimetros

  digitalWrite(Trigger, HIGH);
  delayMicroseconds(10);          //Enviamos un pulso de 10us
  digitalWrite(Trigger, LOW);

  t = pulseIn(Echo, HIGH); //obtenemos el ancho del pulso
  d = t/59;             //escalamos el tiempo a una distancia en cm

  Serial.print("Distancia: ");
  Serial.print(d);      //Enviamos serialmente el valor de la distancia
  Serial.print("cm");
  Serial.println();
  delay(100);          //Hacemos una pausa de 100ms
}
```

Con esto tenemos controlado las distancias. A continuación debemos programar los servos, los cuales serán activados al detectar a través del ultrasonido una distancia inferior a la medida estipulada.

Una vez programado el sensor de ultrasonido vamos a programar el servo.

Como se ha comentado anteriormente los pines de conexión los conectaremos a 5V y a GND respectivamente mientras que el otro pin irá a uno digital. Hay que destacar que para que funcione el servo debemos incluir la librería servo.h.

#include <Servo.h>

Servo servo1;

En nuestra función Setup() pondremos servo1.attach(“puerto conectado”);

Para activar el servo: En el momento que queremos modificar los parámetros del servo para que gire debemos usar servo1.write(º*Grados* ). Recuerda que solo puede girar de 0º a 180º.

Nuestro proyecto controlará dos servos cuando la distancia en el sensor ultrasonido es menor a la considerada según el tamaño de la maqueta.

El código final sería el siguiente:

Hay que destacar que usamos el puerto 7 como fuente de alimentación para el sensor ultrasonido ya que si conectamos todos los elementos a la misma fuente estamos perdiendo diferencia de potencial.

```
#include <Servo.h>

Servo servo1;
Servo servo2;
int puerto = 7;
const int Trigger = 2;   //Pin digital 2 para el Trigger del sensor
const int Echo = 3;   //Pin digital 3 para el Echo del sensor



void setup() {
  Serial.begin(9600);//iniciailzamos la comunicación
  pinMode(Trigger, OUTPUT); //pin como salida
  pinMode(Echo, INPUT);  //pin como entrada
digitalWrite(Trigger, LOW);//Inicializamos el pin con 0
servo1.attach(10);
 servo1.write(90);
 servo2.attach(11);
 servo2.write(90);
 pinMode(puerto, OUTPUT);
 digitalWrite(puerto, HIGH);

}

void loop()
{

  long t; //timepo que demora en llegar el eco
  long d; //distancia en centimetros

  digitalWrite(Trigger, HIGH);
  delayMicroseconds(10);          //Enviamos un pulso de 10us

  digitalWrite(Trigger, LOW);

  t = pulseIn(Echo, HIGH); //obtenemos el ancho del pulso
  d = t/59;             //escalamos el tiempo a una distancia en cm

  Serial.print("Distancia: ");
  Serial.print(d);      //Enviamos serialmente el valor de la distancia
  Serial.print("cm");
```

```
  Serial.println();
delay(100);          //Hacemos una pausa de 100ms
  if(d <= 10){
    for(int i = 90; i >= 0 ; i--){
      servo1.write(i);
      servo2.write(i);
      delay(10);
    }

    delay(8000);
    servo1.write(90);
    servo2.write(90);
  }
}
```