Universidad Politécnica de Madrid

Grado en Ingeniería Electrónica y Automática

Asignatura

INFORMÁTICA

Curso 2018-2019

Universidad Politécnica de Madrid

Grado en Ingeniería Electrónica y Automática

Datos del Grupo

Mark Epelbaum Garcia 54578
Patrick Castillo Mogollon 54939

Introducción

El proyecto consiste en un programa que anticipe y advierta a los conductores de automóviles en una calle unidireccional cuando parar o deban tener precaución al cruzar a través de las vías del tren. Es decir, es un semáforo y una barra que funcionan a costa de sensores de movimiento que captan la trayectoria del tren.

El Arduino, que fue la base para el desarrollo del proyecto, es una herramienta que permite y facilita la configuración de sistemas electrónicos a través de la circulación de corriente eléctrica mediante puertos, y esto siendo contralado por programas y códigos. Este estará acompañado de componentes como sensores, que son aparatos capaces de adquirir y medir ciertos datos, un motor, que emplea movimiento generados por una fuente de energía para emplear ciertas acciones, y luces led que advertirán al usuario algún dato en concreto para avisarles de cómo deben actuar.

Para el desarrollo del trabajo se aplicó contantemente y únicamente el lenguaje C aplicado en el Arduino. Se requirieron conceptos aprendidos en clase, como la creación de funciones para facilitar la lectura del código, la determinación y uso de variables para un uso favorable, y condiciones acompañada de condiciones con sentencias y bucles. Aparte de eso, se investigó una gran gama de códigos y conceptos para poder cumplir los objetivos del programa. Entre ellos, la activación y desactivación de sensores a través del código, el paso de la corriente a través de un circuito, librerías y uso de motores a nuestro antojo, etc.

Desarrollo

La estructura del código está clasificada en 4 etapas:

- Luz verde
- Luces de precaución (parte 1)
- Luz roja
- Luces de precaución (parte 2)
- Luz verde nuevamente

La estructura del código esta clasificada de dicha manera para facilitar la explicación. Cada etapa se refiere al encendimiento de alguna luz en particular, y mientras eso ocurre un motor o sensor estará en uso.

Luz verde:

La primera etapa consiste en un bucle que no tiene fin, debido a que el código está en una función loop, en que se repite lo que está dentro indeterminadamente. El sensor de movimiento ultrasonido 1 estará dando mediciones en cm en números enteros del solido mas cercano que se le encuentra frente del mismo, esto se realizara a partir las funciones ***sensor_ultra(int a , int b)*** y ***Distancia(int b,int *c)*** que después se explicara detalladamente como funcionan. Todas las mediciones obtenidas por dichas funciones aparecerán en el monitor, esto ocurre por los siguientes códigos:

```
Serial.print("Distancia del sensor de movimiento 1 : "); // Indicara cual es de los dos sensores de movimiento está captando el movimiento
Serial.println(distancia_1); //significa el valor obtenido por el sensor
```

En esta etapa se declaró una condición que consiste en que si la medición obtenida por los sensores es menor a 8cm, entonces comenzara la parte de ***Luces de precaución (parte 1)*** del código.

Luces de precaución (parte 1)

En este segmento del programa se apagará la luz verde y comenzará una función void que tiene la capacidad de mover la barra de seguridad, por parte del "ServoMotor", y encender las luces amarillas de forma intermitente. Esta función tiene como beneficio dar tiempo a los conductores para que terminen de cruzar las vías del tren, si ya están en medio camino o detenerse con un tiempo de antelación antes de que cierre por completo la barra de seguridad. La función inventada lleva el nombre de ***bajar_barra*** y está compuesto por un bucle "for" que permite el control de la velocidad de la barra a nuestra disposición y depende de un parámetro denominado velocidad grado.

Luz roja

Al finalizar la función anterior se prendera la luz roja, luego se activara el segundo sensor dentro de un bucle do while que tiene como condición que la distancia medida por el sensor tiene que ser menor de 9cm para salir de un bucle. Nuevamente entrará en otro bucle do while en donde el sensor tendrá que captar una distancia mayor a 8cm de movimiento para salir del bucle. Cada uno de sus datos obtenidos de el aparecerá en monitor. Al salir del segundo bucle iniciara la siguiente etapa y se apagara la luz roja. El primer bucle representa la detección del tren por parte de los sensores y el segundo la ausencia del tren en frente al captador de movimiento.

Luces de precaución (parte 2)

Después de que se terminaron de cumplir las condiciones necesarias para poder llegar a esta etapa, se dará lugar esta parte del código. Durante este fragmento de programa ocurre la función ***subir_barra*** que al igual que la función ***bajar_barra*** funcionan de la misma manera, con la única diferencia que en esta función se sube la barra de seguridad a su estado inicial.

Luz verde nuevamente

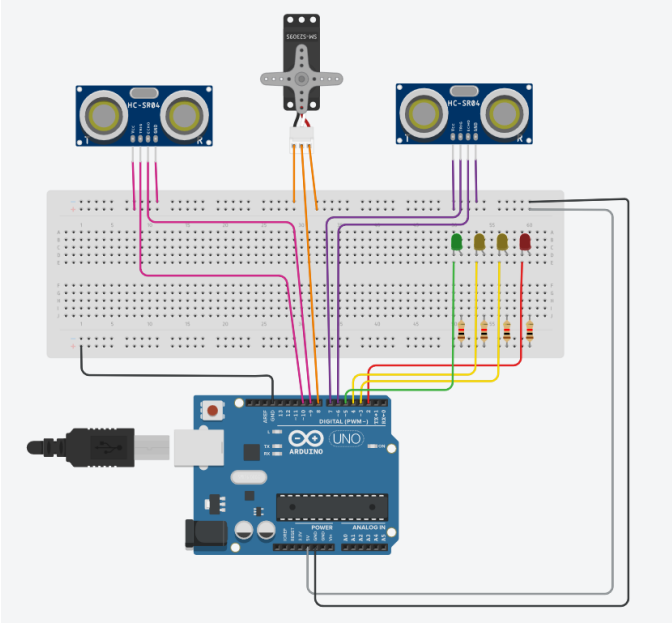
La LED verde se prendera nuevamente. Es decir, el programa empezará desde el principio y se repetirá el código que se explicó cuando la luz estaba en verde.

Arduino y componentes

Para la construcción del proyecto se necesitaron los siguientes componentes:

- 1 Arduino ELEGOO UNO 3
- 2 sensores de movimiento ultrasónicos (HC-SR04)
- 1 micro Servomotor (SG90)
- 1 Protoboard o placa de pruebas
- 4 resistencias de 220 Ω
- 4 luces LED (2 amarillas, 1 roja y 1 verde)
- Varios cables que permiten la conectividad entre los dispositivos.

En la siguiente imagen se puede observar un esquema simplificado del proyecto para obtener una mejor idea de cómo funciona y como está diseñado:



Las luces en el circuito actuaran como semáforo, el primer sensor de movimiento estará a 3 kilómetros de la calle unidireccional, en donde estará el semáforo y la barra de seguridad. Por último, después de la barra, en los costados de la vía del tren estará el segundo sensor de movimiento.

Las luces LED estuvieron conectados en serie junto a resistencias de 220 Ω debido a que se necesitaba una baja intensidad de corriente para que no se quemen las luces y dejasen de funcionar. Todas las luces estuvieron conectados a la misma tierra(ground) ya que tenían corrientes similares para su funcionamiento.

Los sensores y motores fueron conectados a la misma fuente de voltaje, que era el Arduino, debido a que todos necesitaban como mínimo 5 voltios para funcionar. Están conectados a la misma tierra porque circula el mismo voltaje entre ellos. Se decidió que se elegiría una tierra distinta a la de las luces LED por precaución, ya que existía la posibilidad de dañar el Arduino por la emisión de diferentes diferencias de potencial.

Inicialización de variables

```
//inicialización del servo motor
Servo servo;

//Variables de pines arduino
int Yellow1 = 12;
int Yellow2 = 13;
int red = 11;
int green = 10;
int trigPin_1 = 9;
int echoPin_1 = 4;
int trigPin_2 = 6;
int echoPin_2 = 7;


//Variables numericas
int pos=45;
long duracion_1;
int distancia_1;
long duracion_2;
int distancia_2;
const int grados_velocidad=4;
```

- El motor de rotación en dicho programa fue denominado servo
- Las variables de pin corresponden a donde están posicionados los LED, los sensores y motores en los puertos del Arduino
- Las duraciones de tiempo se adquieren directamente de los sensores
- Las distancias se les asignara un valor en la función distancia
- El grado de velocidad es una constante y se refiere a la velocidad del motor por bucle en las funciones en donde se sube y baja la barra.
- Pos se refiere a el valor inicial en la que comenzara el Servo motor y se eligió tal valor ya que facilita su uso.

La mayoría de las variables que se usaron fueron asignadas como int ya que facilita el manejo de dato a lo largo del programa

Función setup:

```
void setup()
{
  //output
 pinMode(Yellow1, OUTPUT);
 pinMode(Yellow2, OUTPUT);
 pinMode(green, OUTPUT);
 pinMode(red, OUTPUT);
 pinMode(trigPin_1, OUTPUT);
 pinMode(trigPin_2, OUTPUT);

 //input
 pinMode(echoPin_1, INPUT);
 pinMode(echoPin_2, INPUT);

 //Servo
 servo.attach(5);

 //Monitor de computador
 Serial.begin(9600);
}
```

La función “setup” consiste en establecer como están situados los componentes electrónicos en el Arduino. Los pinMode acompañados de “output” son aquellos que reciben distintas señales de parte del Arduino, mientras que los “input” son aquellos que entregaran información o

alguna variable al equipo. El serial.begin(9600) permite que en el monitor salga la información que se desee que aparezca cuando se requiera.

Las luces LED

Las luces LED son componentes electrónicos capaces de crear luz, se caracterizan porque en él solo puede pasar la intensidad de corriente a una única dirección. En este programa tienen el papel de informar al conductor de cómo deben actuar respecto a las distintas situaciones, luz verde es avanzar, luz amarilla que deba tener precaución y rojo que debería frenar por completo. El código más común que se usa en estos equipos es:

digitalwrite(a, LOW o HIGH);

En donde "a" sería el número del puerto del pin en el que fue conectado dicha luz en el Arduino. Se usa LOW para detener el paso de corriente y HIGH para emitir corriente al puerto que se encuentra dicha LED. Comúnmente también está acompañada por líneas de código como:

delay(b);
En donde b tendría como significado el numero de milisegundos que haya entre el funcionamiento de líneas de código. Se pudo observar como se utiliza en la función de subir_barra para que las luces amarillas sean intermitentes.

Sensores de movimiento ultrasónicos (HC-SR04)

Los sensores de ultra sonidos son herramientas que tienen como propósito medir longitudes. Este aparato posee la capacidad de enviar una onda de alta frecuencia y detectar la onda cuando se devuelva, de esta manera mide la distancia. Mide el tiempo en el que la onda es emitida y tarda en venir, y mediante la velocidad del sonido se puede aproximar con alta precisión la distancia que recorrió dicha onda.

Para usar estos sensores se utilizaron las siguientes funciones por referencia junto a sus parámetros:

- int sensor_ultra(int trigger , int echo)

```
int sensor_ultra(int trigger,int echo)
{
  int tiempo;
 digitalWrite(trigger, LOW);
 delay(10);
 digitalWrite(trigger, HIGH);
 delayMicroseconds(10);
 digitalWrite(trigger, LOW);
 tiempo = pulseIn(echo, HIGH);
 return(tiempo);
}
```

La siguiente función int consiste en que en la línea del digitalwrite(trigger, HIGH) se transmite la onda de baja frecuencia y después de 10 microsegundos se detiene el paso de corriente de la herramienta que genero la onda y se activa la parte del dispositivo que detecta la onda a través del digitalwrite(echo, HIGH) y regresa un valor entero de tiempo.

Los parámetros que se les asignan a la función son valores constantes, que es donde están conectado dichas partes del Sensor Ultrasonido en el Arduino.

- void Distancia(int duración , int *distancia)

```
void Distancia(int duracion,int *distancia)
 {
*c=b*0.034/2;
 }
```

Después de adquirir el valor de tiempo obtenido por la función anterior, el valor del tiempo será usado como parámetro en esta función para obtener la distancia. A partir de una operación simple que toma en cuenta la velocidad constante en la que viaja el sonido se obtendrá a que longitud quedará el objeto del sensor.

Micro Servomotor (SG90)

Un Micro Servo motor es un motor que tiene la capacidad hacer girar objetos muy pequeños. Comúnmente estos dispositivos están limitados a funcionar de 0 a 180 grados sin poder hacer giros, se suelen usar en casos muy concretos. No poseen una gran velocidad y tampoco aplican gran fuerza.

El motor dentro del programa funciona como la barra de seguridad que advierte y detiene a los conductores que intentan atravesar las vías de tren. El Servo motor se controla a partir de **ángulos** sexagesimales, se puede controlar para que el movimiento de la barra sea instantáneo o controlado. En este caso se utilizo el controlado en las siguientes funciones:

```
void subir_barra(int velocidad)
{
   for(pos = 45; pos <= 135; pos += velocidad)
  {
    servo.write(pos);
    delay(60);
    digitalWrite(Yellow1, HIGH);
    delay(250);
    digitalWrite(Yellow1, LOW);
    digitalWrite(Yellow2, HIGH);
    delay(250);
    digitalWrite(Yellow2, LOW);
  }
}
void bajar_barra(int velocidad)
{
for(pos = 135; pos>=45; pos-=velocidad)
  {
    servo.write(pos);
    delay(30);
    digitalWrite(Yellow1, HIGH);
    delay(275);
    digitalWrite(Yellow1, LOW);
    digitalWrite(Yellow2, HIGH);
    delay(275);
    digitalWrite(Yellow2, LOW);
  }
}
```

La línea de código de servo.writer(pos); tiene como intención mover instantemente la barra dependiendo del valor en grados sexagesimales que se coloque dentro de los paréntesis del código. Este código se introdujo en un bucle “for” para que la barra de seguridad baje con lentitud y que simultáneamente puedan titilar las luces de advertencia, y dar el comunicado que se quiere al usuario. El parámetro de velocidad dentro de la función presentada tiene como significado la rapidez de rotación de grados por bucle del for.

Conclusiones

El programa desarrollado en muchas circunstancias funciona de manera correcta, pero no en todas. Los sensores ultrasonido no son sensores perfectos debido a que pueden captar información errónea si el objeto se encuentra a una alta velocidad, ya que la onda producida por el sensor se puede desviar al entrar en contacto con el vehículo y no dar la información correcta.

Después de la ejecución del proyecto se llegó a la conclusión que es mejor la realización de la misma idea con la única diferencia de que no sean sensores de movimiento ultrasonidos, sino sensores de presión. Esto se debe a que no solo sería necesario la presencia del móvil, pero que también tendría que cumplir que el cuerpo que interactúe con los sensores tenga gran cantidad de masa y sea grande como un tren.

Al fin de cuentas después de probar una y otra vez el programa gran parte de los intentos fueron un éxito, pero en una pequeña cantidad de intentos los sensores no captaban correctamente la información a su alrededor. Finalmente concluimos que el sensor no es del todo funcional contantemente en la práctica, pero apartando dicho problema este programa se puede emplear en la realidad como solución de distintos problemas en el entorno del transito y vías de transporte.

Bibliografía

Autor Desconocido. (2018, December 26). Medir distancia con Arduino y sensor de ultrasonidos HC-SR04. Obtenido de https://www.luisllamas.es/medir-distancia-con-arduino-y-sensor-de-ultrasonidos-hc-sr04/

Autor Desconocido. (2018, April 21). Controlar un servo con Arduino. Obtenido de https://www.luisllamas.es/controlar-un-servo-con-arduino/

Luis. (2018, April 15). Encender un LED con Arduino. Obtenido de https://www.luisllamas.es/encender-un-led-con-arduino/

Arduino. (2019, 3 de mayo). *Wikipedia, La enciclopedia libre*. desde https://es.wikipedia.org/w/index.php?title=Arduino&oldid=115691850.

Autor desconocido. (2016, 29 de junio) Obtenido de https://aprendiendoarduino.wordpress.com/tag/constantes/

Código del programa

```
#include <Servo.h>

//inicialización del servo motor
Servo servo;

//Variables de pines arduino
int Yellow1 = 12;
int Yellow2 = 13;
int red = 11;
int green = 10;
int trigPin_1 = 9;
int echoPin_1 = 4;
int trigPin_2 = 6;
int echoPin_2 = 7;


//Variables numericas
int pos=45;
long duracion_1;
int distancia_1;
long duracion_2;
int distancia_2;
const int grados_velocidad=4;

//funciones prototipos

void subir_barra(int);
void bajar_barra(int);
void Distancia(int,int*);
int sensor_ultra(int,int);
int a=0;

void setup()
{
  //output
 pinMode(Yellow1, OUTPUT);
 pinMode(Yellow2, OUTPUT);
 pinMode(green, OUTPUT);
 pinMode(red, OUTPUT);
 pinMode(trigPin_1, OUTPUT);
 pinMode(trigPin_2, OUTPUT);

 //input
```

```
 pinMode(echoPin_1, INPUT);
 pinMode(echoPin_2, INPUT);

 //Servo
 servo.attach(5);

 //Monitor de computador
 Serial.begin(9600);
}

void loop()
{
 digitalWrite(green, HIGH);//Conectado junto a una resistencia de 220ohms
 duracion_1=sensor_ultra(trigPin_1,echoPin_1);
 Distancia(duracion_1,&distancia_1);
 Serial.print("Distancia del sensor de movimiento 1 : ");
 Serial.println(distancia_1);
 if (distancia_1 <= 8)
  {
 digitalWrite(green, LOW);
 duracion_1 = pulseIn(echoPin_1, LOW);
 bajar_barra(grados_velocidad);
 digitalWrite(red, HIGH);
  do
 {
duracion_2=sensor_ultra(trigPin_2,echoPin_2);
Distancia(duracion_2,&distancia_2);
Serial.print("Distancia del sensor de movimiento 2 : ");
Serial.println(distancia_2);//cm
 }
while(distancia_2>=9);
 do
 {
duracion_2=sensor_ultra(trigPin_2,echoPin_2);
Distancia(duracion_2,&distancia_2);
Serial.print("Distancia del sensor de movimiento 2 : ");
Serial.println(distancia_2);//cm
 }
while(distancia_2<=8);

digitalWrite(red, LOW);
subir_barra(grados_velocidad);
  }
  }
```

```
void subir_barra(int velocidad)
{
  for(pos = 45; pos <= 135; pos += velocidad)
 {
   servo.write(pos);
   delay(60);
   digitalWrite(Yellow1, HIGH);
   delay(250);
   digitalWrite(Yellow1, LOW);
   digitalWrite(Yellow2, HIGH);
   delay(250);
   digitalWrite(Yellow2, LOW);
 }
}
void bajar_barra(int velocidad)
{
for(pos = 135; pos>=45; pos-=velocidad)
 {
   servo.write(pos);
   delay(30);
   digitalWrite(Yellow1, HIGH);
   delay(275);
   digitalWrite(Yellow1, LOW);
   digitalWrite(Yellow2, HIGH);
   delay(275);
   digitalWrite(Yellow2, LOW);
 }

}

void Distancia(int duracion,int *distancia)
 {
*distancia=duracion*0.034/2;
 }

int sensor_ultra(int trigger,int echo)
{
  int tiempo;
 digitalWrite(trigger, LOW);
 delay(10);
 digitalWrite(trigger, HIGH);
 delayMicroseconds(10);
 digitalWrite(trigger, LOW);
 tiempo = pulseIn(echo, HIGH);
```

```
 return(tiempo);
}
```