Development Tools
Arduino Car Speed Detector

Thursday 14th January, 2021

Shijagurumayum Nganaremba Sharma

***Abstract*—In this project, I will show you how to design and build a simple Car Speed Detector circuit using Arduino UNO and IR Sensors. This Arduino Car Speed Detector project can be used to detect speed of a moving car.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

There are definite rules laid out by authorities about driving cars on roads. The most common rule in any country is speed limit in certain roads i.e. you will be in violation of the law if your car speed exceeds this limit.

In order to detect the speed of a moving car, the patrolling officers usually depend on a handheld gun that works on Radar Technology or Lidar Technology. This is a tedious process as the officer has to manually check for over speeding for each vehicle.

What if the Car Speed Detection is made automatic? A simple automatic detection of speed of a vehicle is designed in Arduino Car Speed Detector project, where you can place the system in one place and view the results instantly without any human intervention.

II. Principle of Project

IR Sensors are the main part of the project that detect the speed of a car. Practically, you can implement the setup of IR Sensors in many ways but in this project, I have used two reflective type IR Sensors and placed them 10cm apart.

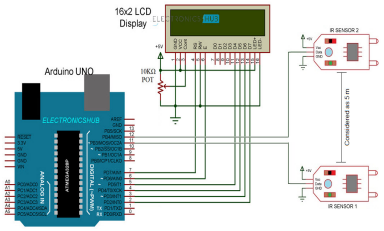
When a car travelling reaches the first sensor, the IR Sensor gets activated. From this moment onward, a timer is initiated and will continue to keep time until the car reaches the second IR Sensor.

By simulating the distance between the two sensors to be 5 meters, you can calculate the speed at which the car travelled from IR Sensor 1 to IR Sensor 2 as you already know the time of travel.

All the calculations and data gathering are done by Arduino and the final result is displayed on a 16X2 LCD Module.

III. Circuit Diagram of Arduino Car Speed Detector

The following image shows the circuit diagram of the Arduino car speed detector project.



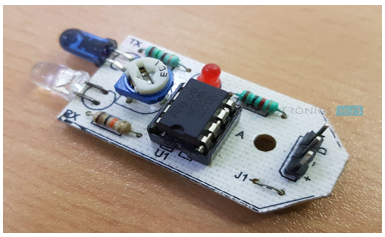
IV. Components Required

- Arduino UNO
- IR Sensors x 2
- 16X2 LCD Display Module
- Breadboard
- Connecting Wires
- Power Supply

V. A Brief Note on IR Sensor

First of all, I have used two digital IR Sensors, which consists of an IR Transmitter (IR LED), an IR Receiver (Photo Diode), a Comparator IC and a few supporting components. The IR Transmitter and Receiver Pair are placed side-by-side so that they form a Reflective Type IR Sensor.

In this type, the IR Transmitter continuously emits Infrared radiations and if there is no object in front of the sensor, none of the Infrared radiation gets reflected back to the IR Receiver.



Interfacing IR Sensor with Raspberry Pi IR Sensor

But if there is an object in front of the sensor, some of the infrared radiation hits the object and gets reflected back. This reflected radiation falls on the IR Receiver, which means that the sensor has detected the object.

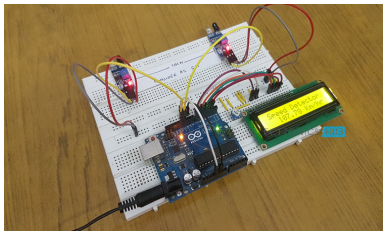
Some IR Sensors has the option to produce both Analog and Digital Outputs but the module I have used has only Digital Output i.e. the output is HIGH when an object is detected and LOW when there is no object.

[2][3][4][1]

VI. Circuit Design

The Digital OUT of the first IR Sensor is connected to Pin 11 of Arduino and the Digital OUT of the second IR Sensor is connected to Pin 12 of Arduino. Both the IR Sensors are provided with necessary power supply connections.

In order to view the car speed details, I have used a 16×2 LCD. Its data pins i.e. D4 – D7 are connected to Digital I/O pins 5 – 2. The RS and E pins of LCD are connected to pins 7 and 6 of Arduino. Rest of the connections are mentioned in the circuit diagram.



Code

```
#include <LiquidCrystal.h>
const int rs = 7, en = 6, d4 = 5,
d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
int sen1=11;
int sen2=12;
unsigned long t1=0;
unsigned long t2=0;
float velocity;
void setup()
{
  lcd.begin(16, 2);
  pinMode(sen1,INPUT);
  pinMode(sen2,INPUT);
  Serial.begin(9600);
  lcd.setCursor(0,0);
  lcd.print(" Speed Detector ");
}

void loop()
{
  while(digitalRead(sen1));
  while(digitalRead(sen1)==0);
  t1=millis();
  while(digitalRead(sen2));
  t2=millis();
  velocity=t2-t1;
  velocity=velocity/1000;
  //convert millisecond to second
  velocity=(5.0/velocity);
  //v=d/t
  velocity=velocity*3600;
  //multiply by seconds per hr
  velocity=velocity/1000;
  //division by meters per Km
  for(int i=5;i>0;i--)
  {
   lcd.setCursor(3,1);
   lcd.print(velocity);
   lcd.print(" Km/hr    ");
   delay(500);
   lcd.setCursor(3,1);
   lcd.print("              ");
   delay(500);
  }

}
}
```

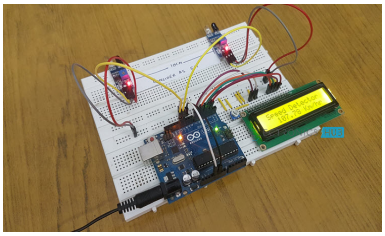
VII. How to operate Arduino Car Speed Detector Project?

Make all the necessary connections with respect to the circuit diagram and upload the code to Arduino. Place the two IR Sensors on the edge of the breadboard so that the distance between them is approximately 10 centimeters. Simulate a car movement in front of the sensors either by using your hands or a toy car. Arduino calculates the speed and displays the result on the 16×2 LCD.

WORKING

The working of the Arduino based car speed detector project is very simple. Arduino continuously reads the inputs from the IR Sensors. When a car moving in front of the setup reaches the first sensor, Arduino becomes alert and capture a time stamp the moment the car leaves the first IR Sensor.

Another time stamp is recorded when the car reaches the second IR Sensor. Millis() function of Arduino used for capturing the time stamps.



Arduino Car Speed Detector Image 1

Arduino then calculates the velocity by assuming the distance as 5 meters between the two IR Sensor and displays the result in kilometers per hour on the 16×2 LCD Display

VIII. Applications

- Helps in capturing speed of vehicles without any human involvement.
- This project can also be used as traffic logger, traffic counter and few other traffic related applications.

REFERENCES

[1] M. S. Afaqui, E. Garcia-Villegas, and E. Lopez-Aguilera. Ieee 802.11ax: Challenges and requirements for future high efficiency wifi. *IEEE Wireless Communications*, 24(3):130–137, 2017.

[2] B. Bellalta. Ieee 802.11ax: High-efficiency wlans. *IEEE Wireless Communications*, 23(1):38–46, 2016.

[3] R. Billinton, M. Fotuhi-Firuzabad, and L. Bertling. Bibliography on the application of probability methods in power system reliability evaluation 1996-1999. *IEEE Transactions on Power Systems*, 16(4):595–602, 2001.

[4] H. Shin, H. R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R. M. Summers. Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning. *IEEE Transactions on Medical Imaging*, 35(5):1285–1298, 2016.