# AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH

## Faculty of Engineering

# LAB REPORT

| | | | |
|---|---|---|---|
| Experiment Name: | Building an Obstacle Detection System. | | |
| Experiment No: | 5 | Date of Submission: | 12 March 2022 |
| Course Title: | Microprocessor and Embedded Systems | | |
| Course Code: | Click here to enter text. | Section: | F |
| Semester: | Spring 2021-22 | Course Teacher: | **Dr. Nadia Anam** |

| No | Name | ID | Program | Signature |
|---|---|---|---|---|
| 1 | Bhuiyan Saad Bin Mobarak | 19-41059-2 | BSc [EEE] | |
| 2 | Nabil, MD. Jobair Ahmad | 18-38837-3 | BSc [CSE] | |
| 3 | Akter, Alam Morium | 19-39957-1 | BSc [CSE] | |
| 4 | Sakib,A.B.M.Nazmus | 19-41582-3 | BSc [EEE] | |
| 5 | Bhuiyan, Fahim Mahmud | 20-42970-1 | BSc [CSE] | |
| 6 | | | Choose an item. | |
| 7 | | | Choose an item. | |
| 8 | | | Choose an item. | |
| 9 | | | Choose an item. | |
| 10 | | | Choose an item. | |

*Faculty use only*

| | | |
|---|---|---|
| FACULTYCOMMENTS | **Marks Obtained** | |
| | **Total Marks** | |

## Title:

Building an Obstacle Detection System.

## Introduction:

The ease with which we may design items using a microcontroller is a factor to consider when selecting one. An assembler, a debugger, a code-efficient C language compiler, an emulator, technical assistance, and both in-house and outside knowledge are all important concerns. We picked the Arduino series because there is a lot of online community assistance. For Arduino, there are several free and efficient code samples from various projects.

## Objectives:

In this lab, we will learn –

(i) How to code a simple Obstacle Detection System in Arduino IDE.

(ii) Implement a simple Obstacle Detection System in Hardware.

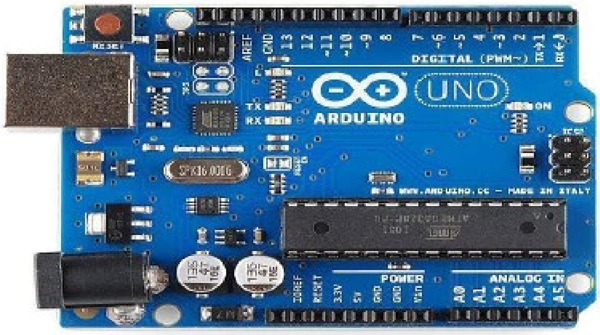
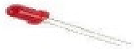
(iii) How to simulate the code in Proteus/Tinkercad and observe the results.

## Theory & Methodology:

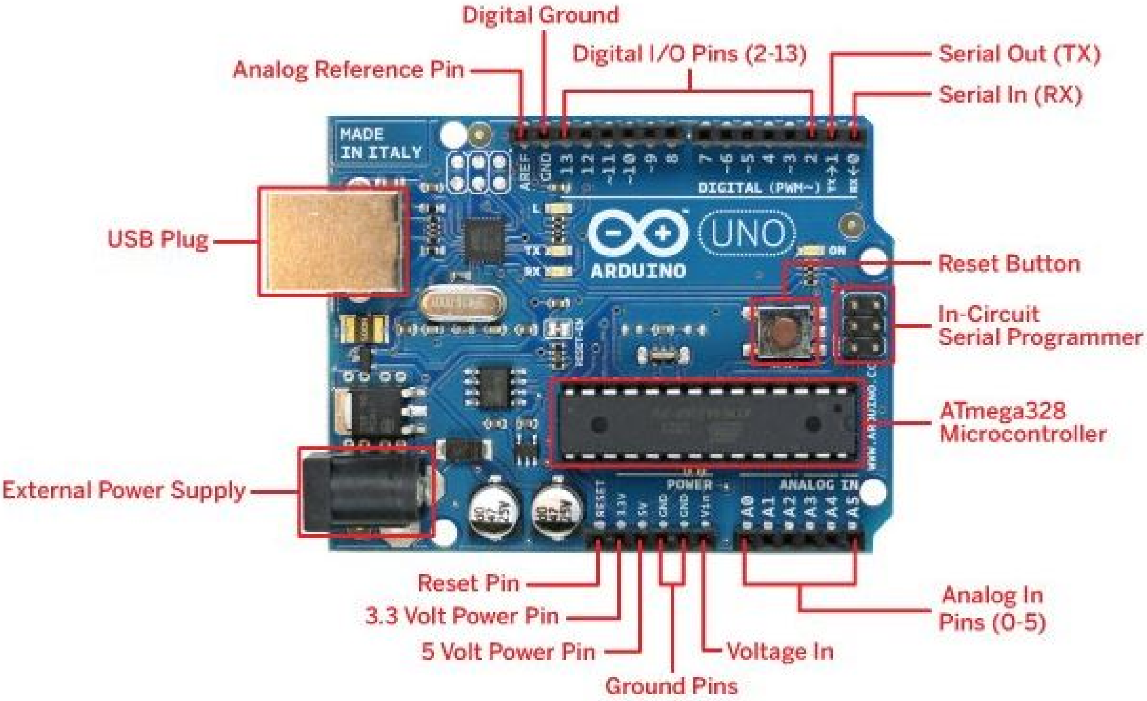
Arduino is an open-source platform that may be used to make interactive electronics projects. Arduino is made up of a programmable microcontroller and IDE (Integrated Development Environment) software that runs on your computer and is used to create and upload computer code to the microcontroller board. To load new code into the board, the Arduino Uno does not require a hardware circuit (programmer/burner). Using a USB cord and the Arduino IDE (which utilizes a simplified version of C++ to create code), we can quickly load a code into the board.

## Apparatus:

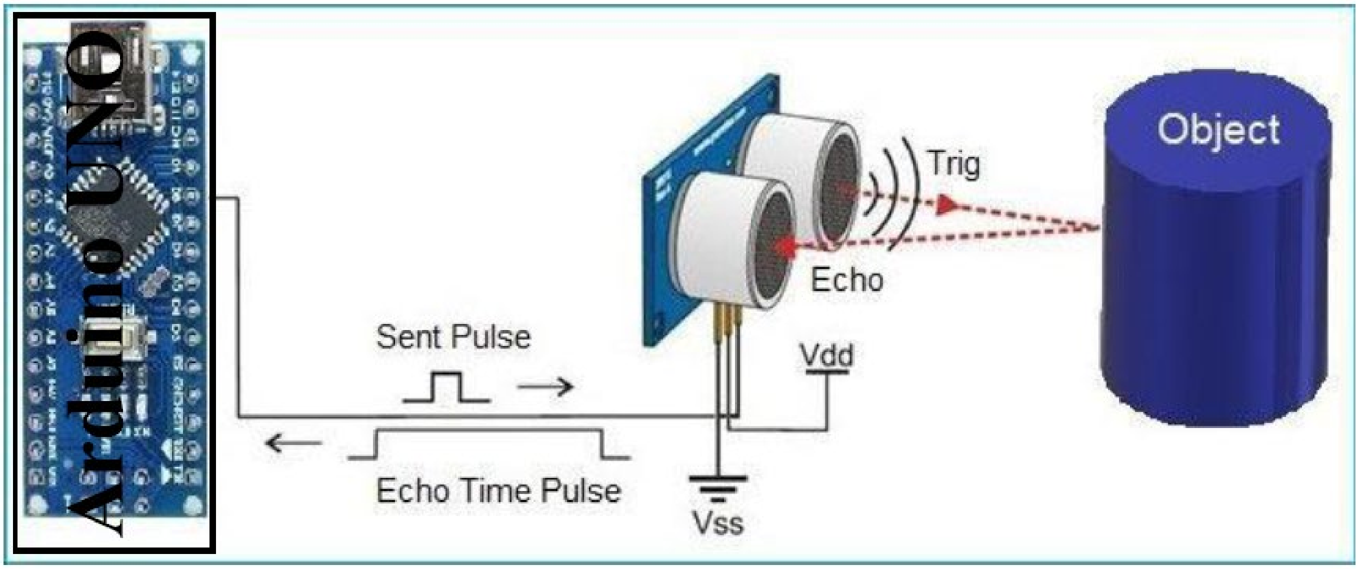
| | |
|---|---|
| 1) Arduino IDE (any version) | Software |

| | |
|---|---|
| 2) Arduino Uno (R3) Board |  |
| 3) Sonar Sensor (HCSR04) | |
| 4) LED |  |

## Overview of the Board (Arduino Uno R3):



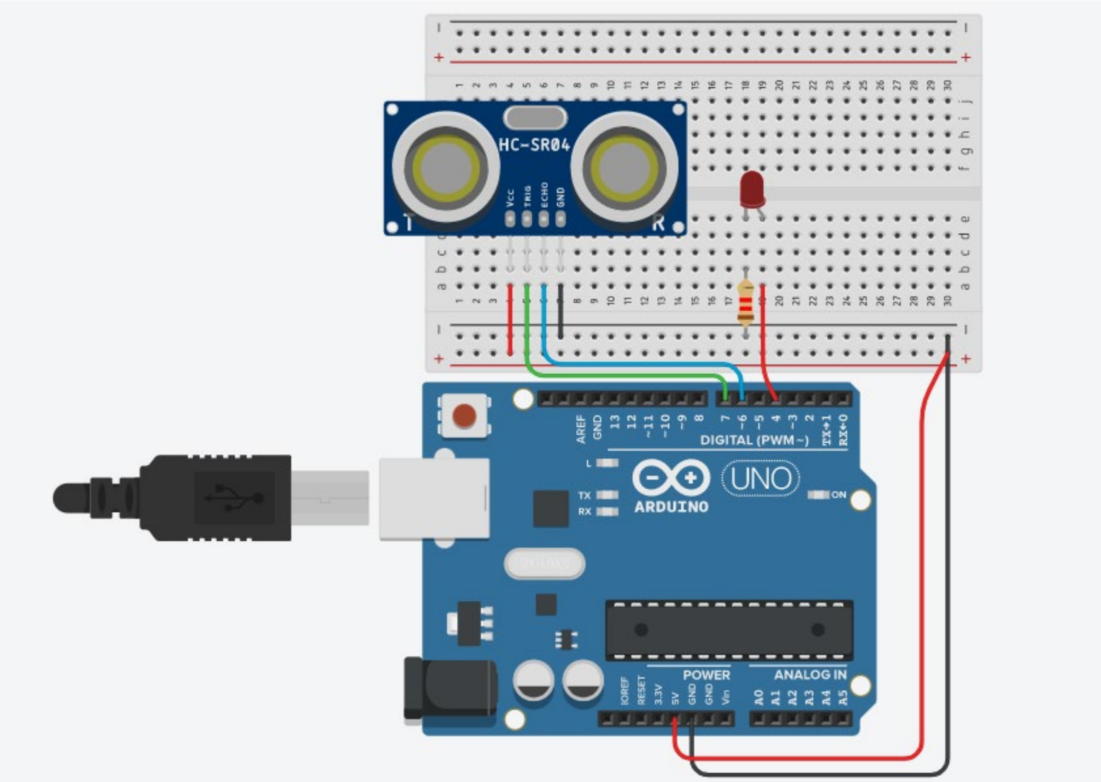
## Overview of Sonar System:



## Experimental Procedure:

### Setting up the Circuit

Our lab's main goal is to employ a sonar sensor to determine how far away an object is.



### How it works

We're utilizing a sonar sensor (HCS04) to measure the distance of an obstacle, and it'll also light up an LED when it finds it. A transmitter, receiver, and control circuit make up the HCSR04 Ultrasonic Range Module. VCC, GND, Trigger, and Echo are the four pins. It's simple to connect to Arduino boards. The Module delivers eight 40 kHz pulses and detects whether there is a pulse signal back using IO trigger for at least 10us high level signal. With connecting wire, the sensor's Trigger pin is linked to digital pin 3 and the Echo pin to digital pin 2 of the Arduino Uno R3 board. A LED is attached to pin 13 to indicate the presence of an obstruction. Because the trigger will be created by Arduino and the LED status (HIGH/LOW) will also be altered by the Arduino board, pins 3 and 13 will be used as output pins. Because the ping created by the Arduino board goes from the trigger to the echo, we divide the distance traveled by half to get the object's distance. The speed of sound is 340 m/s, or 29 microseconds per centimeter or 74 microseconds per inch, as we all know. As a result, the following calculation will be used to compute the distance traversed by the trigger in centimeters:

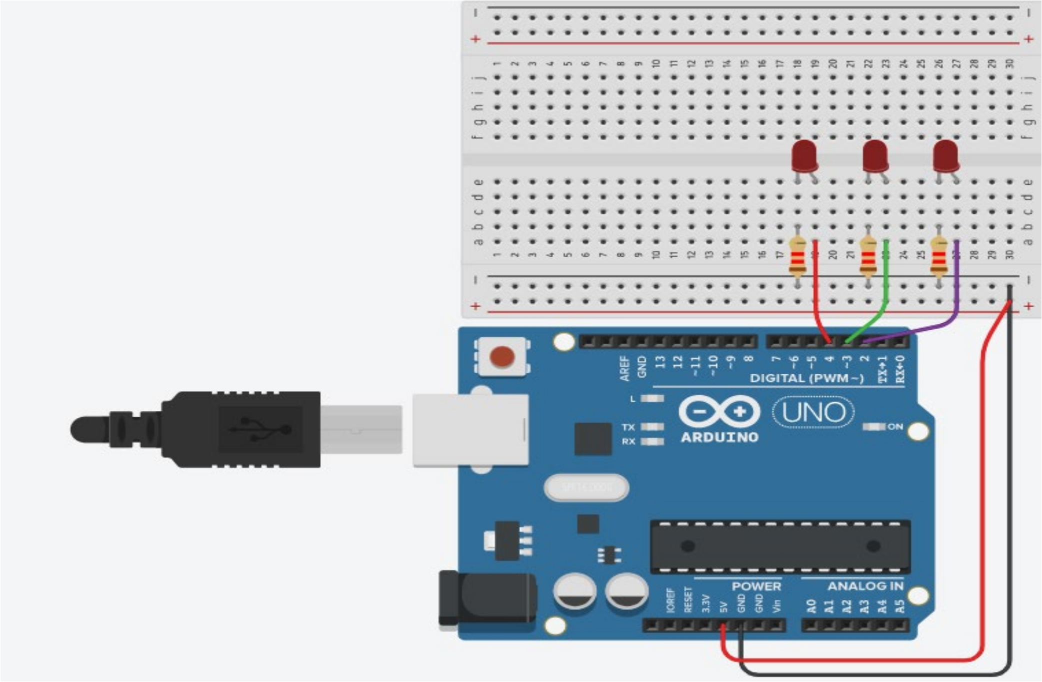
```
distance_cm = microseconds / 29 / 2;
distance_in = microseconds / 74 /2;
```

## Ultrasonic Distance Sensor in Arduino using Arduino IDE:

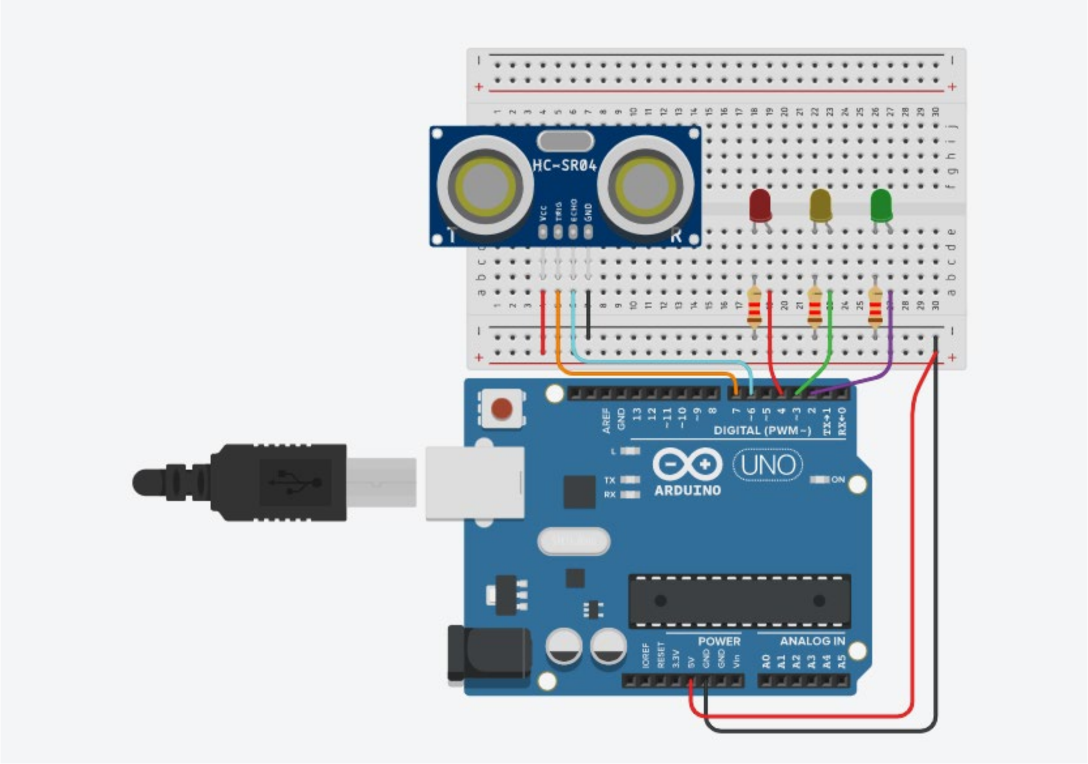
Let's use an ultrasonic rangefinder (distance sensor) and Arduino's digital input to measure distances. We'll use a breadboard to connect a circuit and some simple Arduino code to operate a single LED.

Ultrasonic rangefinders work by bouncing sound waves off things in front of them, similar to how bats use echolocation to assess their surroundings. The proximity sensor emits a signal and counts how long it takes for it to return. This information is received by the Arduino software, which calculates the distance between the sensor and the object.

## Step 1: Build the LED circuit



## Step 2: Add Proximity Sensor

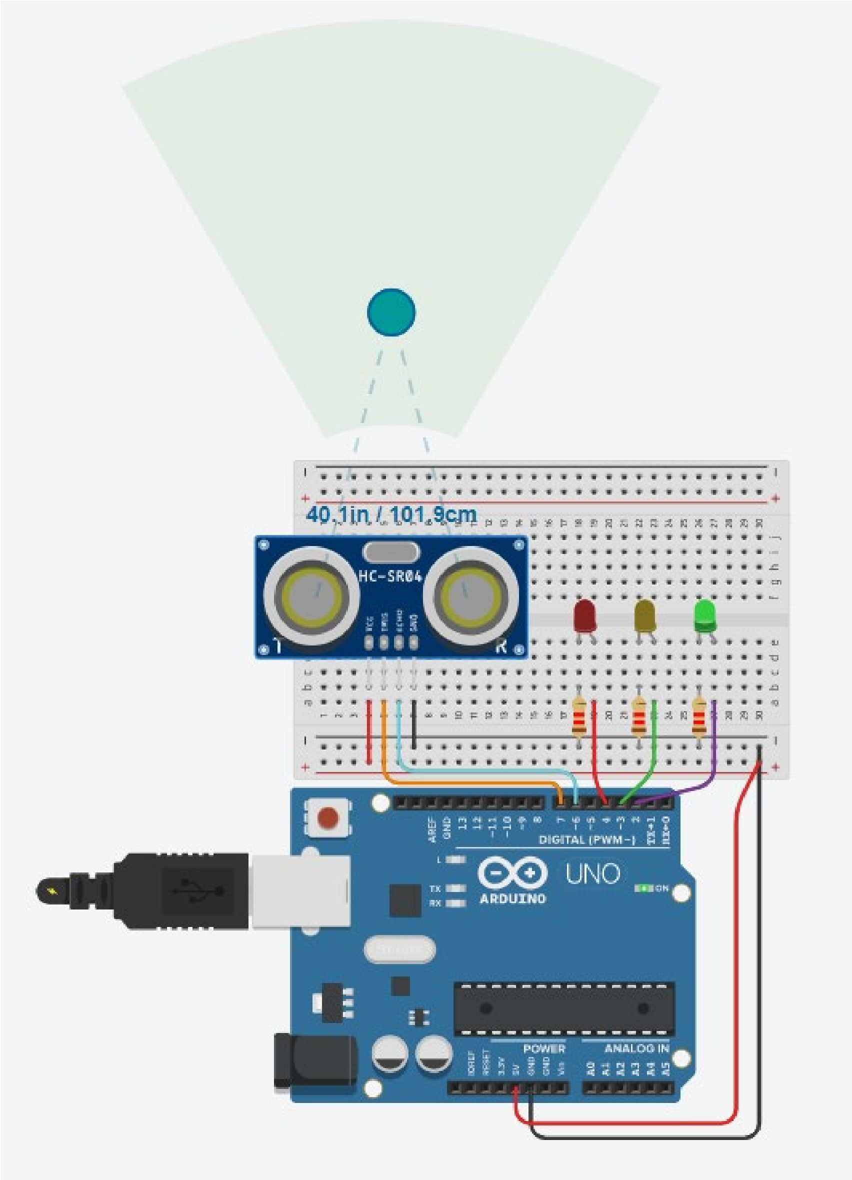


## Step 3: Ultrasonic Rangefinder Arduino Code

```
const int echoPin = 6; // Echo Pin of Ultrasonic Sensor
const int trigPin = 7; // Trigger Pin of Ultrasonic Sensor
const int R_LED = 4; // LED at Pin 4
const int Y_LED = 3; // LED at Pin 3
const int G_LED = 2; // LED at Pin 2
void setup()
{
Serial.begin(9600); // Starting Serial Communication
pinMode(trigPin, OUTPUT); // initialising pin 7 as output
pinMode(echoPin, INPUT); // initialising pin 6 as input
pinMode(Y_LED, OUTPUT); // initialising pin 4 as output
pinMode(R_LED, OUTPUT); // initialising pin 4 as output
pinMode(G_LED, OUTPUT); // initialising pin 4 as output
}


void loop()
{
long duration, inches, cm;
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
duration = pulseIn(echoPin, HIGH); // using pulsin function to determine total time
inches = microsecondsToInches(duration); // calling method
cm = microsecondsToCentimeters(duration); // calling method
if(cm<=8)
{
Serial.print(inches);
Serial.print("in, ");
Serial.print(cm);
Serial.print("cm");
Serial.println();
digitalWrite(R_LED, HIGH);
digitalWrite(Y_LED, LOW);
digitalWrite(G_LED, LOW);
}

if(cm>8 && cm<15)
{
```

```
Serial.print(inches);
Serial.print("in, ");
Serial.print(cm);
Serial.print("cm");
Serial.println();
digitalWrite(R_LED, LOW);
digitalWrite(Y_LED, HIGH);
digitalWrite(G_LED, LOW);
}

if(cm>15)
{
Serial.print(inches);
Serial.print("in, ");
Serial.print(cm);
Serial.print("cm");
Serial.println();
digitalWrite(R_LED, LOW);
digitalWrite(Y_LED, LOW);
digitalWrite(G_LED, HIGH);
}

delay(1000);
}
long microsecondsToInches(long microseconds) // method to covert microsec to inches
{
return microseconds / 74 / 2;
}
long microsecondsToCentimeters(long microseconds) // method to covert microsec to cm
{
return microseconds / 29 / 2;
}
```

## Result:



## Discussion:

The true purpose of this lab is to learn how to detect simple obstacle using Arduino.

The system that was built in this experiment, was to detect different obstacles for different distances. The system was designed in such manner that, a proxy meter was set for different distances. As, whenever an object comes within the range of 2cm or below, the Red LED will be turned on. Again, when an object is detected at above than 2cm, the Yellow LED will be turned on. Lastly, when the sensor detects an obstacle at the range of 10cm or above, it will command Arduino to turn the Green LED on.

So, it can be said that, our goal of this experiment is finally achieved.

## References:

[1] Arduino IDE, https://www.arduino.cc/en/Main/Software accessed on May 3, 2019.
[2] Arduino and Proteus Library, https://etechnophiles.com/add-simulate-ultrasonic-sensorproteus-2018-edition/ accessed on May 3, 2019.
[3] Ultrasonic Distance Sensor in Arduino With Tinkercad https://www.instructables.com/id/Ultrasonic-Distance-Sensor-Arduino-Tinkercad/ accessed on May 3, 2019.