AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH

Faculty of Engineering

Lab Report

Experiment # 7

Experiment Title: Interfacing the Arduino with an external sensor using serial communication protocol for implementing an obstacle detection system.

Date of Perform:	Date Month 2023	**Date of Submission:**	24 July 2023
Course Title:	MICROPROCESSOR AND EMBEDDED SYSTEM LAB		
Course Code:	COE3104	**Section:**	N/A
Semester:	Summer 2022-23	**Degree Program:**	BSc in CSE/EEE
Course Teacher:	**Prof. Dr. Engr. Muhibul Haque Bhuyan**		

Declaration and Statement of Authorship:

1. I/we hold a copy of this Assignment/Case-Study, which can be produced if the original is lost/damaged.
2. This Assignment/Case-Study is my/our original work and no part of it has been copied from any other student's work or from any other source except where due acknowledgment is made.
3. No part of this Assignment/Case-Study has been written for me/us by any other person except where such collaboration has been authorized by the concerned teacher and is clearly acknowledged in the assignment.
4. I/we have not previously submitted or currently submitting this work for any other course/unit.
5. This work may be reproduced, communicated, compared, and archived for the purpose of detecting plagiarism.
6. I/we give permission for a copy of my/our marked work to be retained by the Faculty Member for review by any internal/external examiners.
7. I/we understand that Plagiarism is the presentation of the work, idea, or creation of another person as though it is your own. It is a form of cheating and is a very serious academic offense that may lead to expulsion from the University. Plagiarized material can be drawn from, and presented in, written, graphic and visual forms, including electronic data, and oral presentations. Plagiarism occurs when the origin of the source is not appropriately cited.
8. I/we also understand that enabling plagiarism is the act of assisting or allowing another person to plagiarize or copy my/our work.

** Student(s) must complete all details except the faculty use part.*
** Please submit all assignments to your course teacher or the office of the concerned teacher.

Group #

Sl No	Name	ID	PROGRAM	SIGNATURE

Faculty use only

FACULTY COMMENTS	**Marks Obtained**	
	Total Marks	

Table of Contents

Experiment Title: Interfacing the Arduino with an external sensor using serial communication protocol for implementing an obstacle detection system.

Objectives:
The objectives of this experiment are to-

- Implement a simple obstacle detection system in Arduino IDE
- Implement a simple obstacle detection system using an Arduino microcontroller.

Equipment List:

1) Arduino IDE (2.0.1 or any recent version)
2) Arduino UNO (R3) board
3) Sonar Sensor (HCSR04)
4) Breadboard
5) LEDs (red, green and yellow)
6) Resistor of 100 Ω
7) Jumper Wires.

Circuit Diagram:

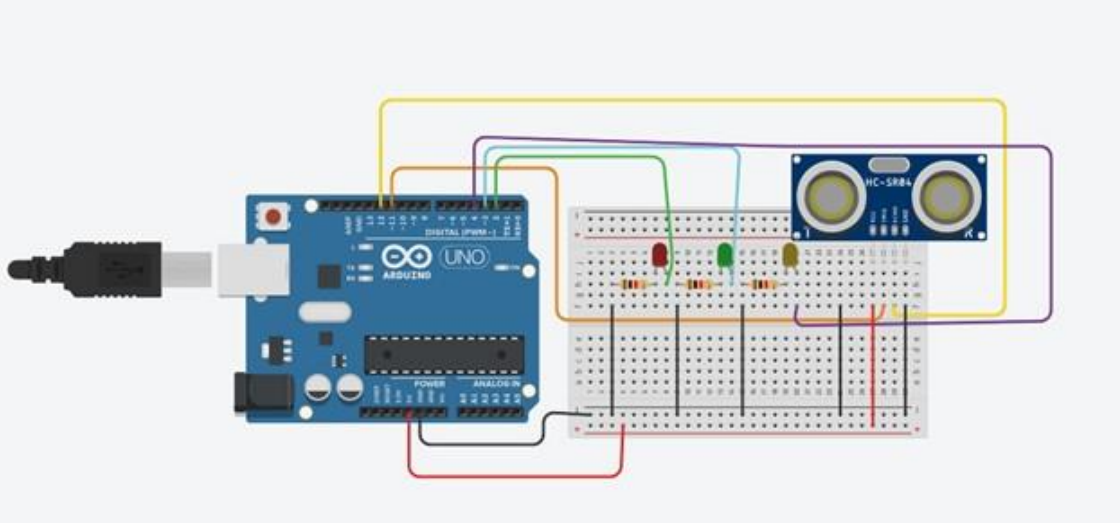

Fig. 1 Experimental setup of an obstacle detection system using Arduino

Code/Program:
The following is the code for the implementation of an obstacle detection system using Arduino with the necessary code explanation:

```
// define the pin numbers
const int trigPin = 11;
const int echoPin = 12;
```

```
// define variables
long duration;
float distance, distanceinches, distanceThreshold;

void setup() {
 Serial.begin(9600); // Starts the serial communication
 pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input
pinMode(2, OUTPUT); // Sets pins 2, 3, and 4 as the Output pin
pinMode(3, OUTPUT);
pinMode(4, OUTPUT);
}

void loop() {
 // Clears the trigPin
 digitalWrite(trigPin, LOW);
 delayMicroseconds(2);
 // Sets the trigPin on HIGH state for 10 microseconds
 digitalWrite(trigPin, HIGH);
 delayMicroseconds(10);
 digitalWrite(trigPin, LOW);
 // Reads the echoPin, returns the sound wave travel time in microseconds
 duration = pulseIn(echoPin, HIGH);
 // Calculating the distance
 distance = (duration/2)*1e-6*340*100;
 distanceinches = (distance/2.54);
 // Prints the distance on the Serial Monitor
 Serial.print("Distance = ");
 Serial.print(distance);
 Serial.print(" cm; ");
 Serial.print("Distance = ");
 Serial.print(distanceinches);
 Serial.println(" inches");
 // set threshold distance to activate LEDs
distanceThreshold = 80;
if (distance > distanceThreshold) {
 digitalWrite(2, LOW);
 digitalWrite(3, LOW);
 digitalWrite(4, LOW);
}
if (distance < distanceThreshold && distance > distanceThreshold-30) {
 digitalWrite(2, HIGH);
 digitalWrite(3, LOW);
 digitalWrite(4, LOW);
}
if (distance < distanceThreshold-30 && distance > distanceThreshold-50) {
```

```
 digitalWrite(2, HIGH);
 digitalWrite(3, HIGH);
 digitalWrite(4, LOW);
}
 if (distance < distanceThreshold-50 && distance > distanceThreshold-70 ) {
 digitalWrite(2, HIGH);
 digitalWrite(3, HIGH);
 digitalWrite(4, HIGH);
}
delay(200); // Wait for 200 millisecond(s)
}
```

Hardware Output Results:

Here is the hardware implementation of an obstacle detection system using Arduino and the necessary explanation of the implementation:

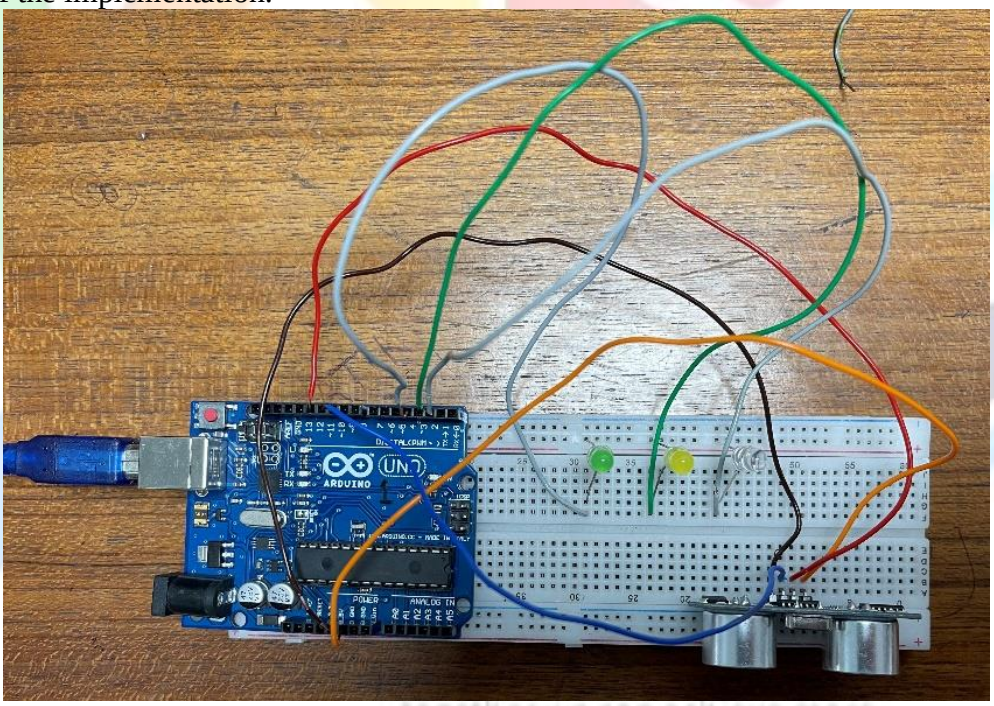

Fig. 2 Hardware implementation of an obstacle detection system using Arduino

Explanation:

In this experiment, a object detection system was created using a sonar system and some LEDs. An Arduino Uno board was taken. The sonar was connected with the Arduino Uno board. The VCC of the sonar was connected with the 5V pin of the Arduino Uno Board, The GND of the sonar was connected with the GND pin of the Arduino Uno board. The Trig of the sonar was connected with pin 11 of the Arduino Uno. At the end, the Echo of the sonar was connected with pin 12 of the microcontroller. To express the distance of the object from the sonar, three lights were set with the Arduino Uno board at the following pin 2, 3 and 4. Each of these LEDs indicated if an object was a far distance, medium distance or close distance if it was within the range of the sonar. All of these LEDs were grounded using 100 Ω resistors.

Experimental Output Results:

Here are the results of an obstacle detection system using Arduino and the necessary explanation of the results:

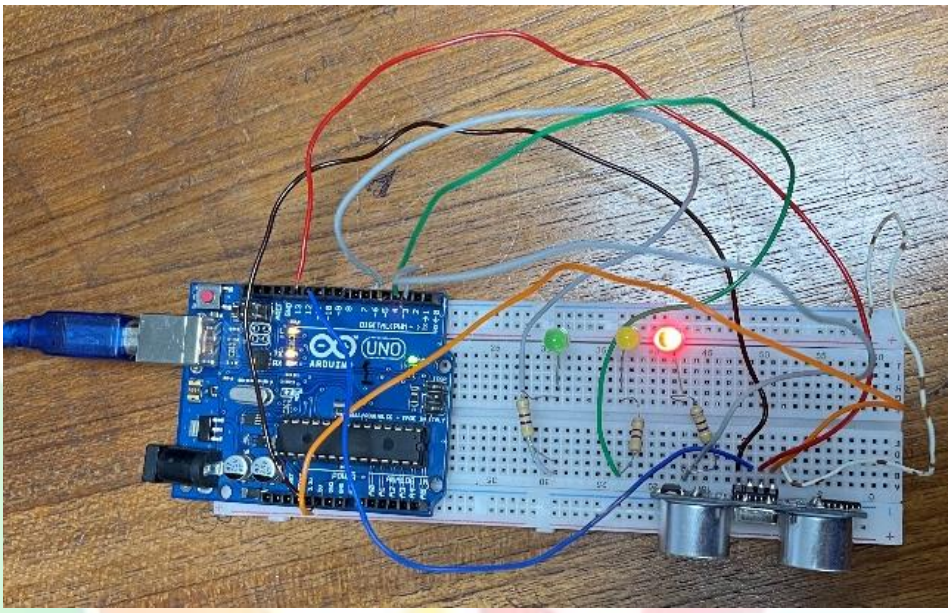

Fig.3 RED LED turned ON when a distance as at a far distance

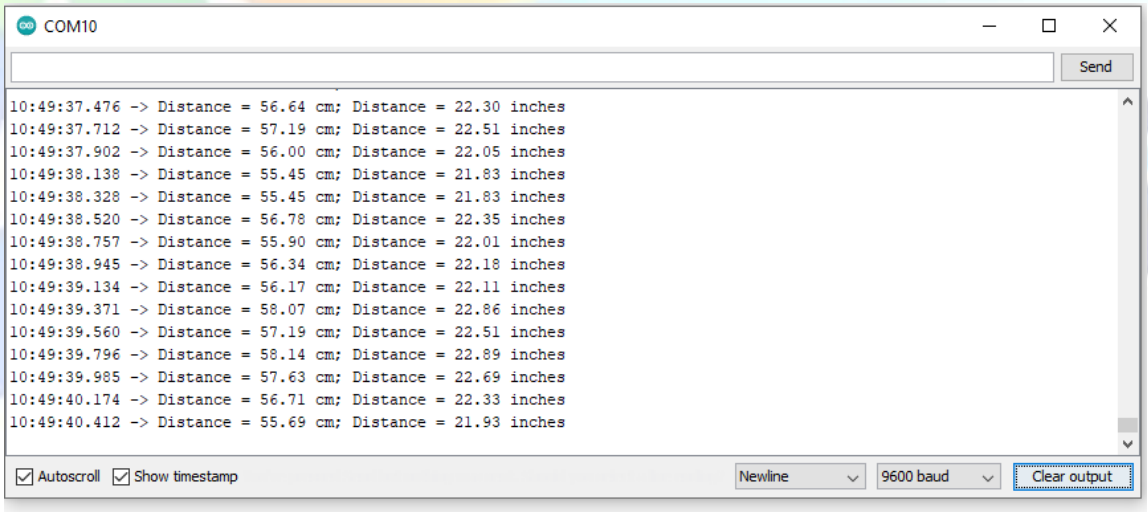

Fig.4 Serial Monitor output when the RED LED turns ON

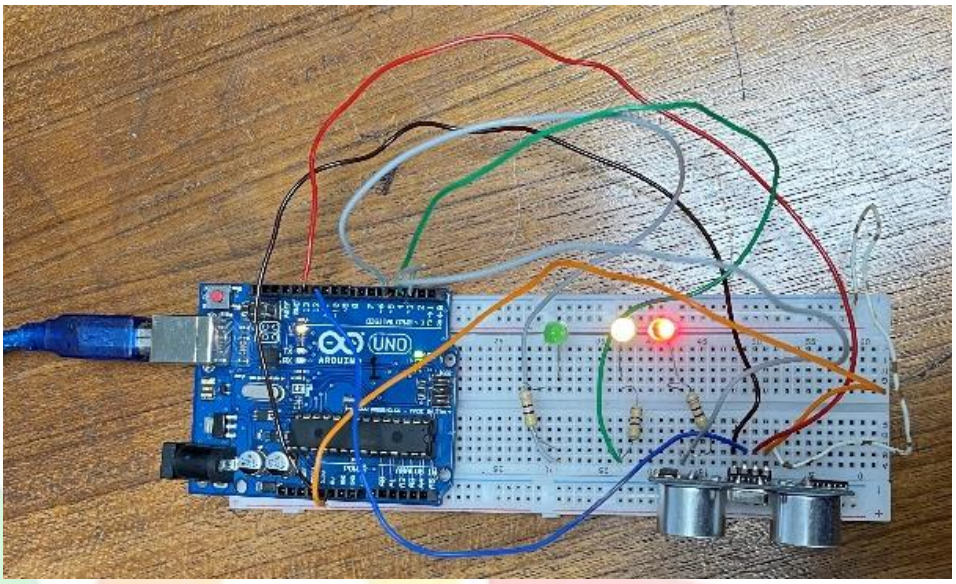

Fig.5 YELLOW LED turned ON when a distance as at a medium distance

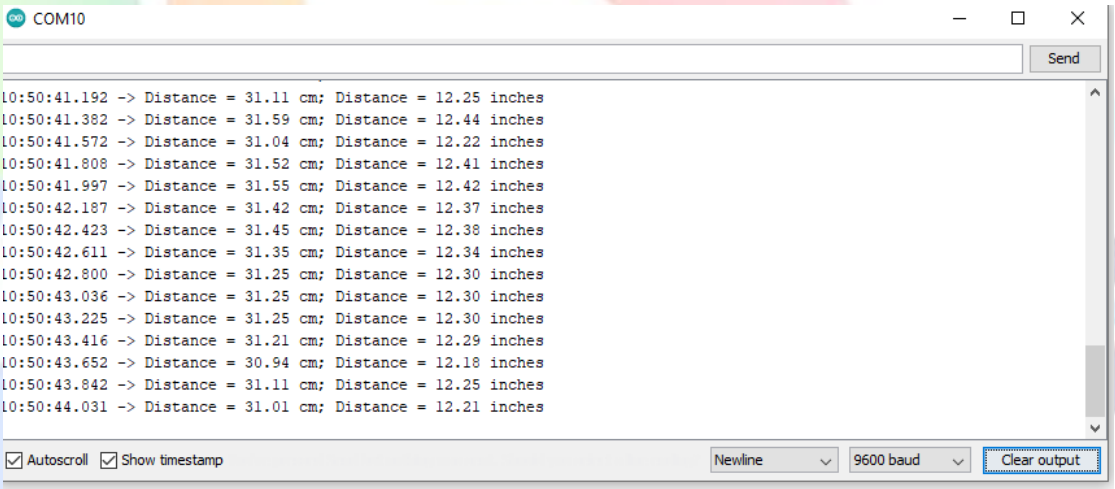

Fig.6 Serial Monitor output when the YELLOW LED turns ON

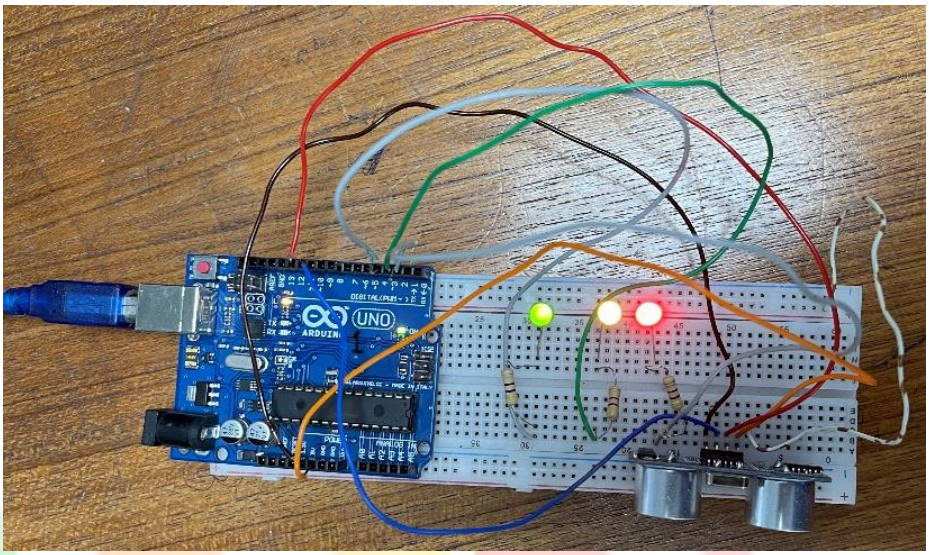

Fig.7 GREEN LED turned ON when a distance as at a close distance

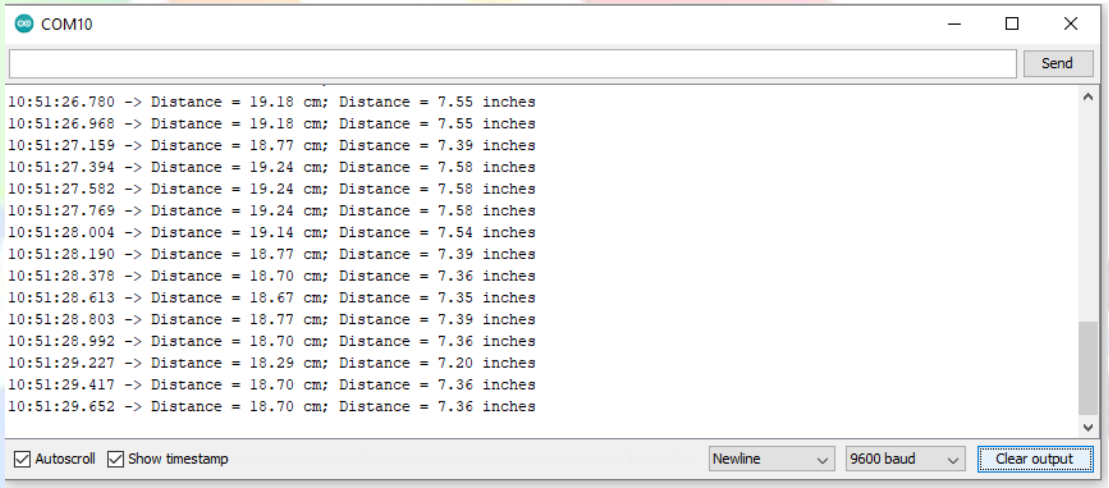

Fig.8 Serial Monitor output when the GREEN LED turns ON

Explanation:

In this experiment, while the circuit was running an object was brought in front of the sonar. As the object was within the range of the sonar and far away from the sonar, only the RED LED turned ON. In the serial monitor, it showed the distance in centimeter and inches. The distance was > 50 cm and > 20 inches. When the object was brought to medium range, the YELLOW LED also turned ON. The serial monitor showed the distance > 30 cm ~ < 50 cm and > 7 inches ~ < 20 inches. When the object brought more closer to the sonar, the third LED turned ON, in this case it was the GREEN LED. The distance was shown on the serial monitor.

Simulation Output Results:

Here is the simulation implementation of an obstacle detection system using Arduino and the necessary explanation of the implementation:

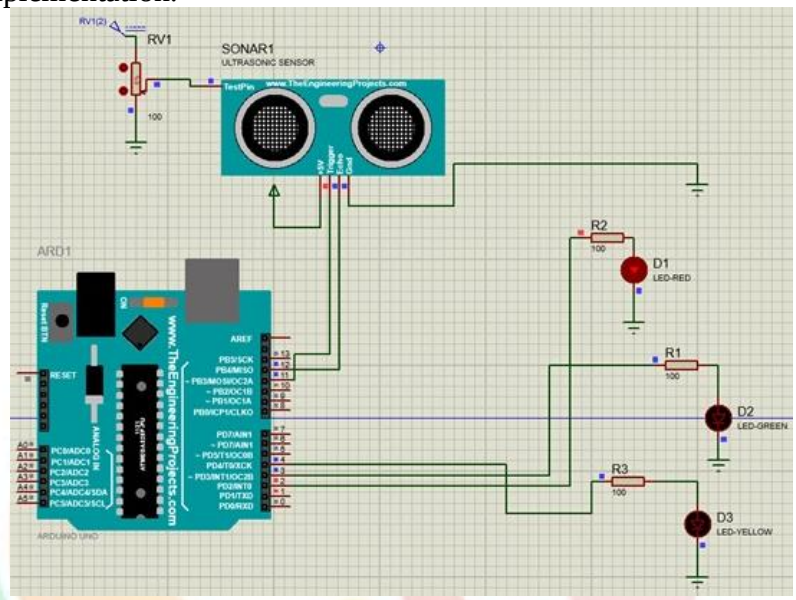

Fig.9 First LED ON in Proteus Simulation when the object was far away

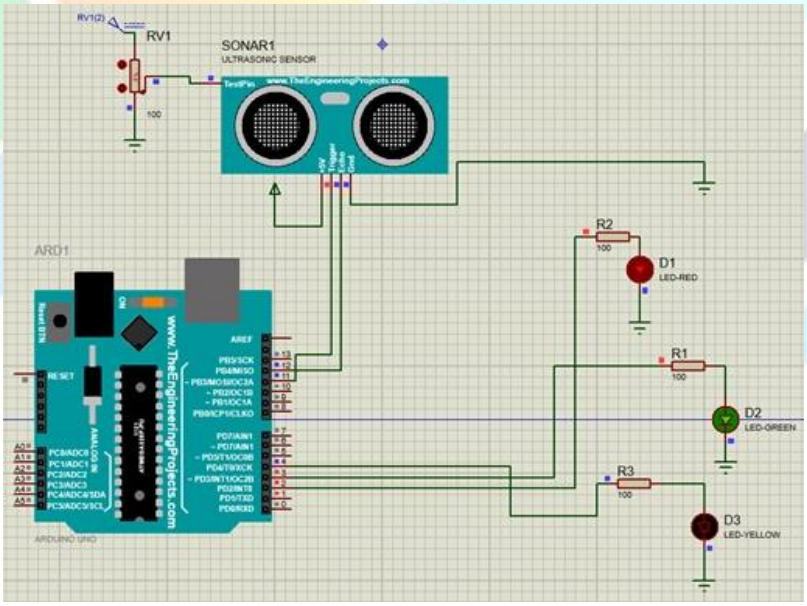

Fig.10 Second LED ON in Proteus Simulation when the object was in Medium Distance

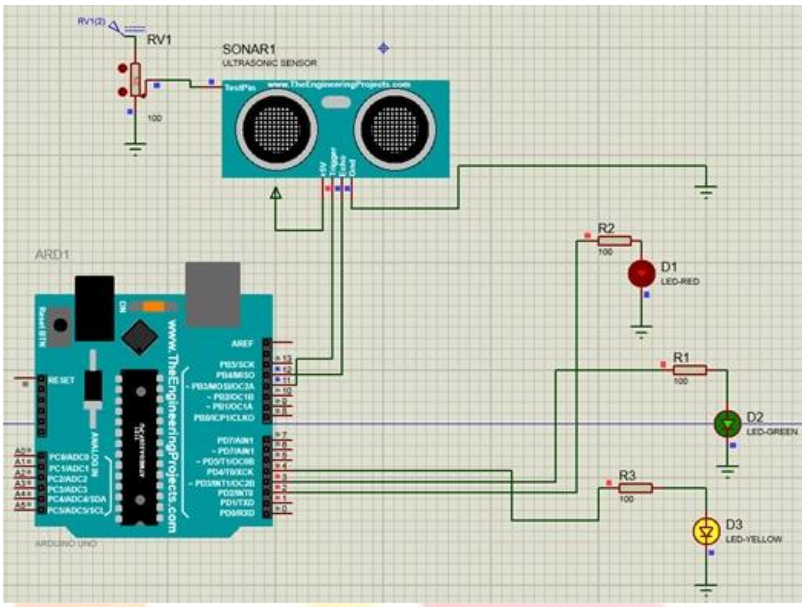

Fig.11 Third LED ON in Proteus Simulation when the object was in Close Distance

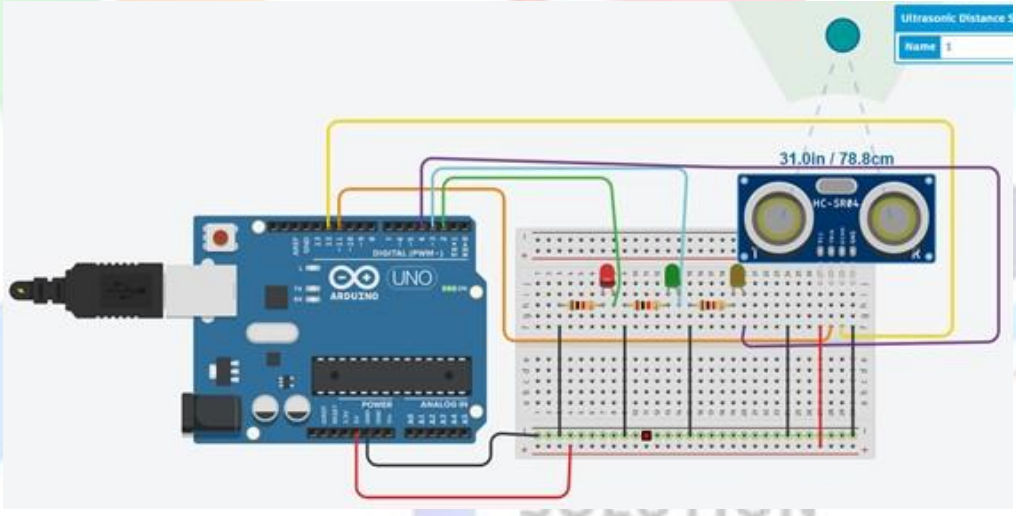

Fig.12 First LED ON in TinkerCad Simulation when the object was far away

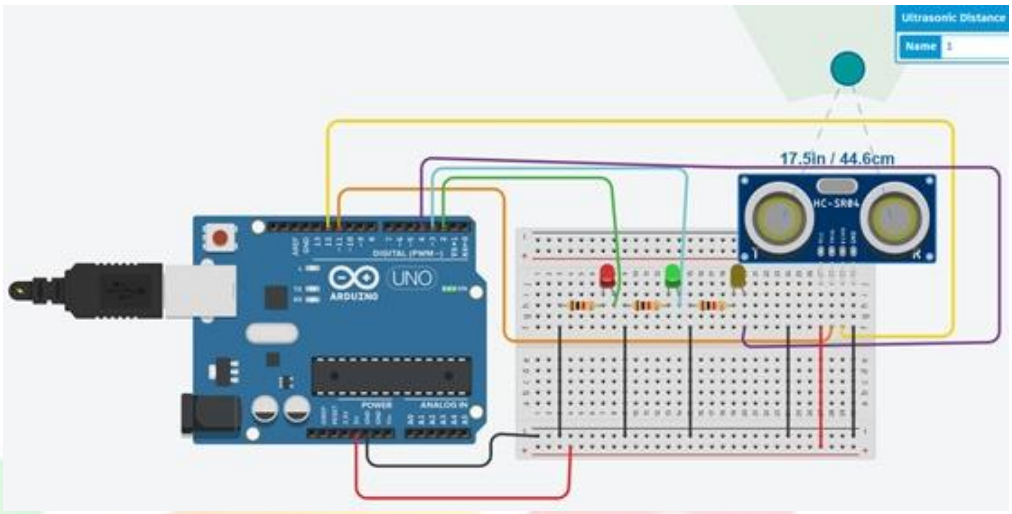

Fig.13 Second LED ON in TinkerCad Simulation when the object was in Medium Distance

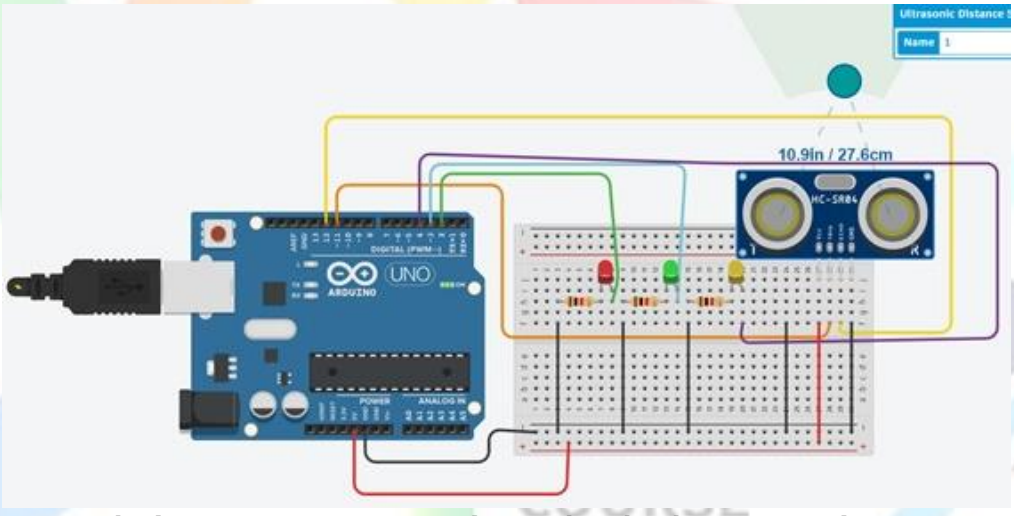

Fig.14 Third LED ON in Proteus Simulation when the object was in Close Distance

Explanation:

In this experiment, two different simulation software were used. Proteus Simulation Software and TinkerCad simulation software both were used to verify the results that were obtained in the experiment. At first, the circuit was made in the Proteus simulation software accordingly. After developing it, the Arduino HEX file was generated from the Arduino IDE and was imported in the proteus simulation software. The results were observed by bringing the cursor close to the sonar. The results were observed by the operation of the LEDs. The same was performed on the TinkerCad simulation software. The circuit was developed on the software. The object was set graphically on the simulation software after importing the Arduino code on the simulation software. The results were observed accordingly was well by observing the LEDs and serial monitor of the Arduino Uno microcontroller that was used on the TinkerCad software.

Answers to the Questions in the Lab Manual:

Configure the port numbers for outputs and inputs according to your ID. Consider the middle five digits from your ID (if your ID is XY-PQABC-Z then consider input port as P and output ports as QABC of your ID). Include all the programs and results within your lab report.

Solution: The following university ID was used:

2	0	-	4	2	5	5	7	-	1
X	Y	-	P	Q	A	B	C	-	Z

Port of Input pin would be accordingly:
echoPin will be PIN 2

Port of Output pin would be accordingly:
trigPin will be PIN 4
RED LED will be PIN 5
GREEN LED will be PIN 6 (As same pins cannot express two different distances)
YELLOW LED will be PIN 7

The following is the code for a LED light control using a switch from Master and slave with the necessary code explanation:

The Modified Code:

```
// define the pin numbers
const int trigPin = 4;
const int echoPin = 2;
// define variables
long duration;
float distance, distanceinches, distanceThreshold;
void setup() {
 Serial.begin(9600); // Starts the serial communication
 pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input
pinMode(5, OUTPUT); // Sets pins 2, 3, and 4 as the Output pin
pinMode(6, OUTPUT);
pinMode(7, OUTPUT);
}
void loop() {
 // Clears the trigPin
 digitalWrite(trigPin, LOW);
 delayMicroseconds(2);
 // Sets the trigPin on HIGH state for 10 microseconds
 digitalWrite(trigPin, HIGH);
 delayMicroseconds(10);
 digitalWrite(trigPin, LOW);
 // Reads the echoPin, returns the sound wave travel time in microseconds
 duration = pulseIn(echoPin, HIGH);
 // Calculating the distance
 distance = (duration/2)*1e-6*340*100;
 distanceinches = (distance/2.54);
 // Prints the distance on the Serial Monitor
```

```
 Serial.print("Distance = ");
 Serial.print(distance);
 Serial.print(" cm; ");
 Serial.print("Distance = ");
 Serial.print(distanceinches);
 Serial.println(" inches");
 // set threshold distance to activate LEDs
distanceThreshold = 80;
if (distance > distanceThreshold) {
 digitalWrite(5, LOW);
 digitalWrite(6, LOW);
 digitalWrite(7, LOW);
}
if (distance < distanceThreshold && distance > distanceThreshold-30) {
 digitalWrite(5, HIGH);
 digitalWrite(6, LOW);
 digitalWrite(7, LOW);
}
if (distance < distanceThreshold-30 && distance > distanceThreshold-50) {
 digitalWrite(5, HIGH);
 digitalWrite(6, HIGH);
 digitalWrite(7, LOW);
}
 if (distance < distanceThreshold-50 && distance > distanceThreshold-70 ) {
 digitalWrite(5, HIGH);
 digitalWrite(6, HIGH);
 digitalWrite(7, HIGH);
}
delay(200); // Wait for 200 millisecond(s)
}
```

Here is the simulation implementation of a LED light control using a switch from Master and slave and the necessary explanation of the implementation:

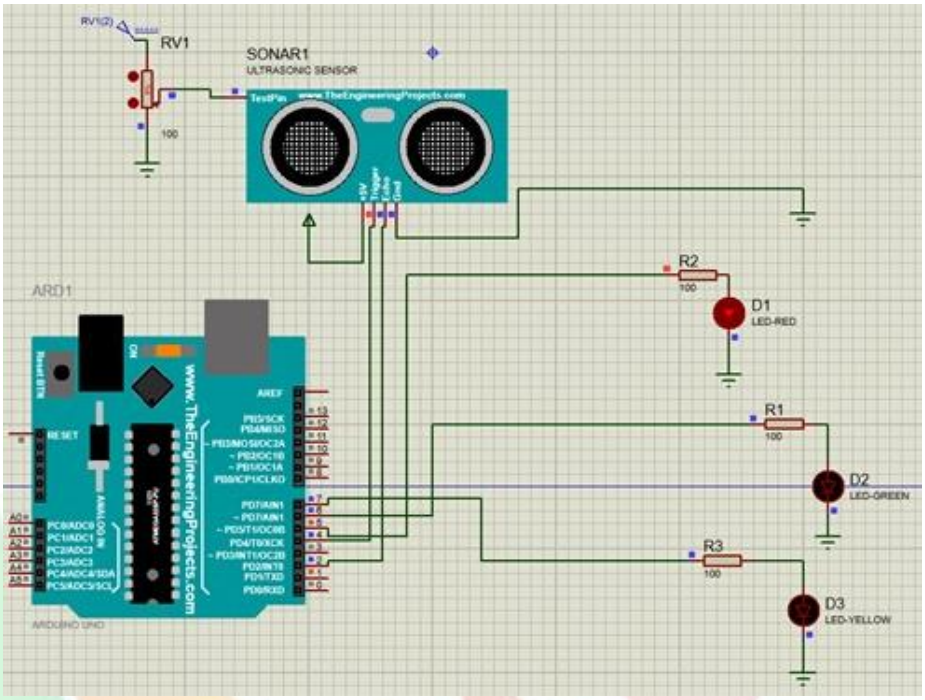

Fig.15 First LED ON in Proteus Simulation when the object was far away on the modified code

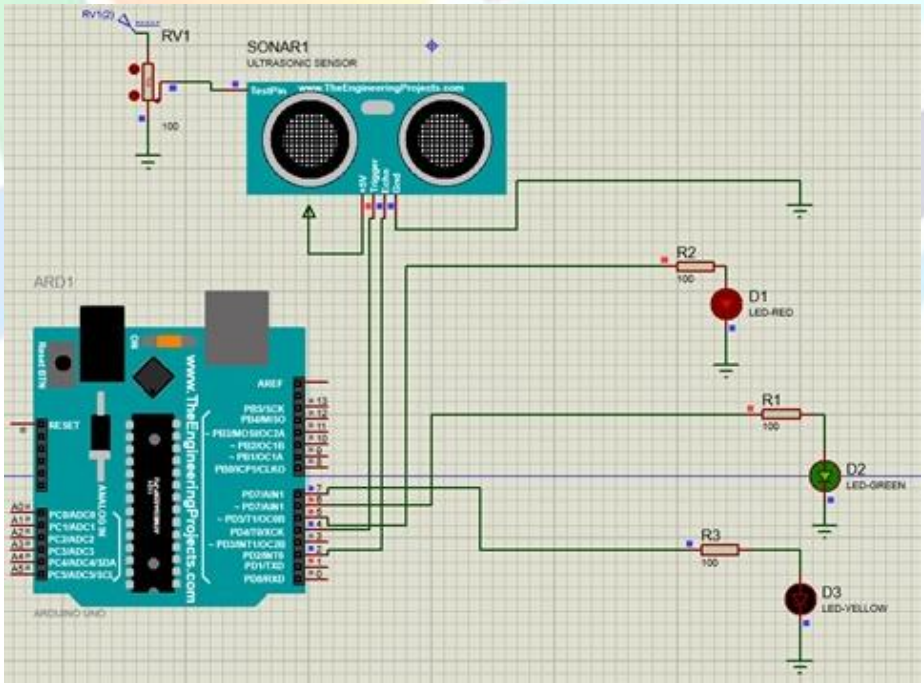

Fig.16 Second LED ON in Proteus Simulation when the object was in medium distance on the modified code

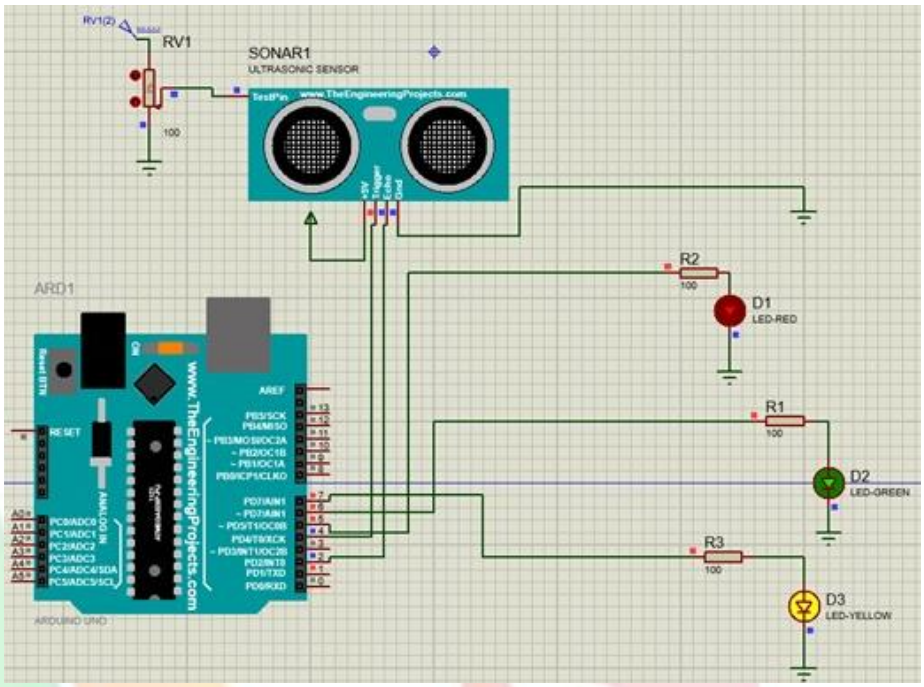

Fig.17 Third LED ON in Proteus Simulation when the object was in Close Distance on the modified code

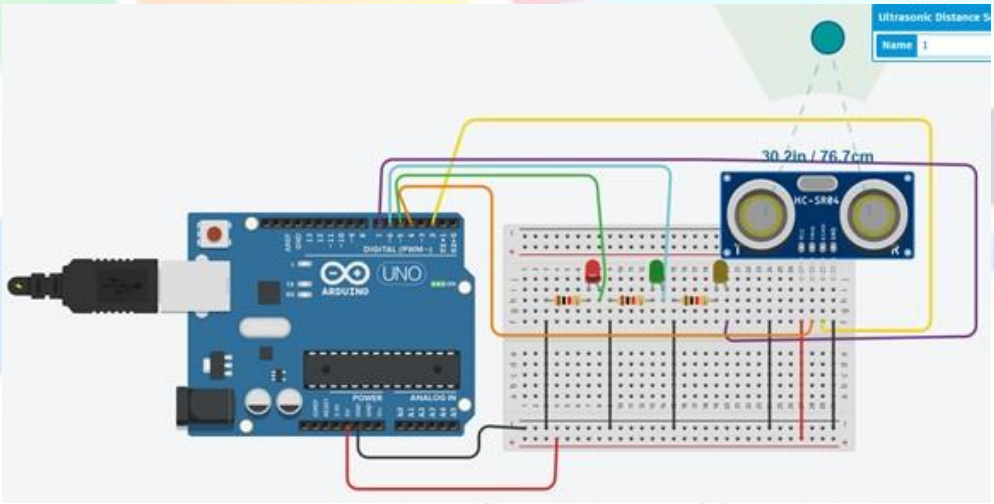

Fig.18 First LED ON in TinkerCad Simulation when the object was far away on the modified code

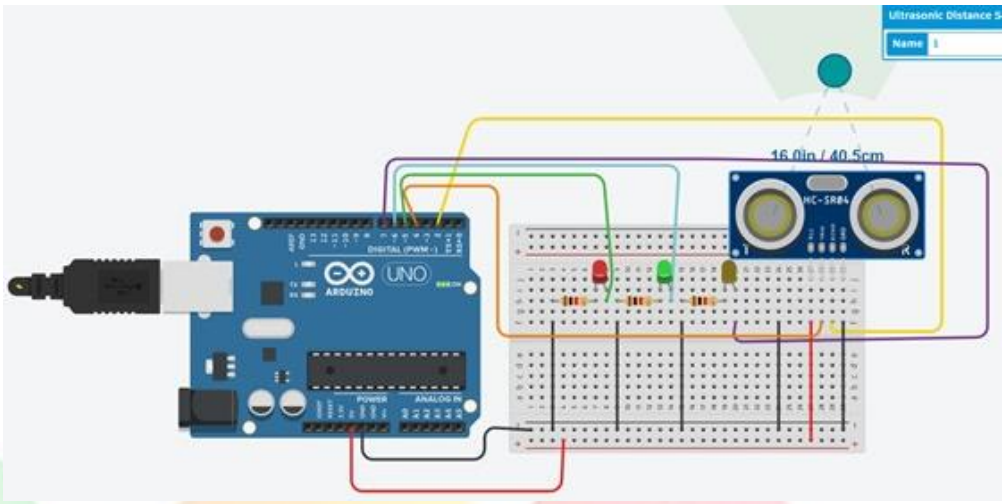

Fig.19 Second LED ON in TinkerCad Simulation when the object was in medium distance on the modified code

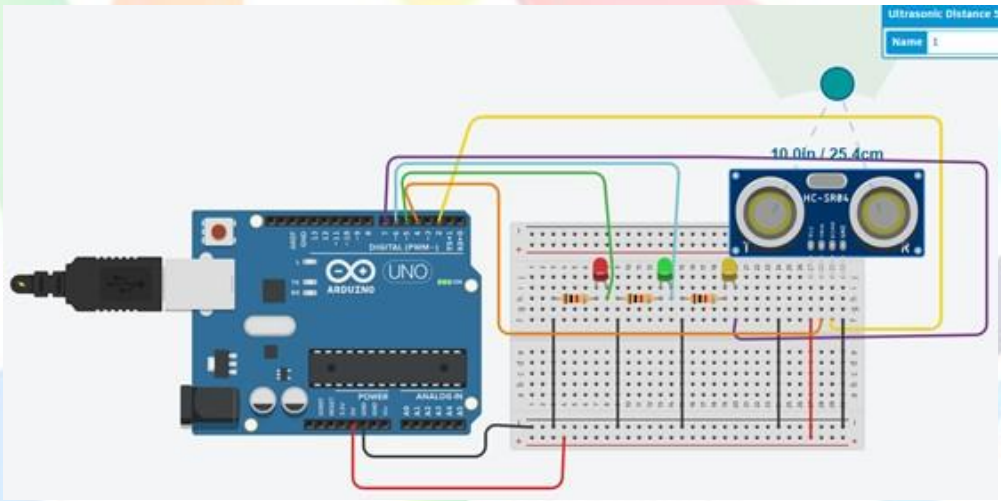

Fig.20 Third LED ON in TinkerCad Simulation when the object was in close distance on the modified code

Discussion:

In this experiment, a sonar sensor was used to detect the distance of an object or obstacle and the distance between the object and sonar was displayed using the LEDs that were connected. The sonar sensor that was used in this experiment were studied carefully before using it. The pin operations and how the sonar works were observed and carefully understood. After that, the sonar sensor was set accordingly with the Arduino Uno board accordingly. The LEDs were set as per necessity. After that, the circuit was operated and the systems operations were observed. The method of how the system was detecting the distance between the object and sonar were observed. The method of how it was performed using Trigger and Echo was observed accordingly as well and co-related with the experiment's working methodology. The LEDs were turning ON accordingly based on the distance and formulas that were set on the code. Changes in the detection mechanism was also observed by changing the detecting parameters accordingly. All the results that were observed were carefully noted down for further evaluation. The similar system was developed on the simulation softwares like Proteus and TinkerCad. The results that were obtained on the physical operation were evaluated with the simulated outcomes. There were some minor discrepancies that were observed. The distance that were generated on the simulation's serial monitor were a bit different

compared to the ones that were observed on the physical testing. This might be caused due to minor system and human errors. This caused the inconsistencies in the values of the serial monitor. Moreover, the detecting rate of the system in the physical environment and simulation virtual environment were a bit different. This was ruled as normal as human error was general in the physical world. From the observation it can be said that after both hardware and software implementation showed the expected outcomes and the experimental objectives was achieved.

References:

1) *How HC-SR04 Ultrasonic Sensor Works & Interface It With Arduino*, Last Minute Engineers, [Cited: July 22, 2023] Available: https://lastminuteengineers.com/arduino-sr04-ultrasonic-sensor-tutorial/

2) Arduino CC Website, [Online] [Cited: July 22, 2023] Available: https://docs.arduino.cc/hardware/uno-rev3

3) Sonar System Using Arduino and Altair Embed, [Online] [Cited: July 22, 2023] Available: https://altairuniversity.com/45471-sonar-system-using-arduino-block-diagram-model/

4) AIUB Microprocessor and Embedded Systems Lab Manual 7