Infrared Sensor triggered I2C communication between Arduino Mega and Arduino Uno along with Ultrasonic Sensor

Shrey Kothavade
Master of Engineering in Information Technology SRH Hochschule Heidelberg
Matriculation Number: 11018044
Heidelberg, Germany
11018044@stud.hochschule-heidelberg.de

Abstract: Inter-Integrated Circuit(I2C) Protocol is used to establish communication between multiple peripheral digital integrated circuits or controller chips. I2C communication is widely used in industries since 1990's. It is commonly used when a single controller board cannot process the excessive amount of data being processed and starts lagging. Therefore, to achieve higher processing speeds or to fulfil the need of new controller in the circuit, I2C protocol is preferred. This project focuses on the I2C protocol communication between the Atmega 2560 and Atmega 328P based development boards. Where, Atmega 2560 based board is used in the master position and Atmega 328P based board is used in the Slave position. Information is generated from the two Ultrasonic sensors based on the Slave side. The exchange of information is triggered by the Infrared sensor on the Master side. Information exchanged is displayed on the Serial monitor of the Arduino.

Keywords: I2C, Arduino Mega, Arduino Uno, IR Sensor, Ultrasonic Sensor

I. INTRODUCTION:

In recent times, many applications, e.g. Automobile, Laptops, embedded systems, etc, are need to be connected to multiple systems at same time. To establish a two-way connection between these peripherals is very important. There are multiple options to do this, but one of the most efficient and easy to implement is Inter-Integrated Circuit(I2C). I2C protocol was developed by Philips semiconductor in early 80's. It is multiple slave and multiple master bus which is commonly used in various control architectures such as power management bus (PMBus), Advanced Telecom Computing Architecture (ATCA), System Management Bus (SMBus), etc. [1]

I2C is a bidirectional wire interface, commonly known as Two-Wire serial interface. serial clock (SCL) and Serial data (SDA) are carried on this bidirectional line. To begin with, the SDA line is where data is conveyed when it is sent from one device to another. Second, the master device generates the SCL line, which controls when data is delivered and when it is read. Both wires, SDA and SCL, transport messages between the master and slave connected to the bus. The devices linked to the bus are known as the master or slave. Each device on the bus is represented by a unique address, and each device has the capability of acting as a transmitter or receiver.[2][3]

Devices on the I2C bus can function both as master or slave. When a node on the bus generates clock, it becomes the master. Aside from that, the master node should establish communication with the slave. Whereas Slave mode receives clock generated by master. In addition, when addressed by the master, this node will respond. A bus device can operate in four modes. A bus device can function as both a Master and a Slave. Each Master and Slave has the ability to serve as Transmit and Slave. Thus, there will be four modes of operation: Slave Receiver, Master Receiver, Slave Transmitter and Master Transmitter. [2][4][5]

This project emphasizes on I2C protocol between the Arduino Mega (Atmel 2560 based development board) and Arduino Uno (Atmel 328P based development board) for transmitting and receiving data from multiple Ultrasonic sensors (HC 04) with infrared sensor triggered circuit at master.

This report is structured as follows: section II contains the Objective and Aim of Work. The Section III explores the Working of the project. Whereas Section IV will contain description of the components used and section V will depict results of the project. Section VI will conclude the paper and Section VII will be references. Code of the project will be available in Section VIII.

II. OBJECTIVE AND AIM OF WORK:

I2C communication is widely used in industries to read certain memory IC's, reading hardware sensors, Transmitting and controlling user related actions, communicating with multiple microcontrollers etc. This project implements, I2C protocol between the Atmega IC's for transmitting and receiving data from Ultrasonic sensors

with infrared sensor triggered circuit. This circuit can be implemented in Mechanical industries for distance measurement, in home automation, in robotics to have multiple slave IC's to a Multiple masters, etc.

III. Working:

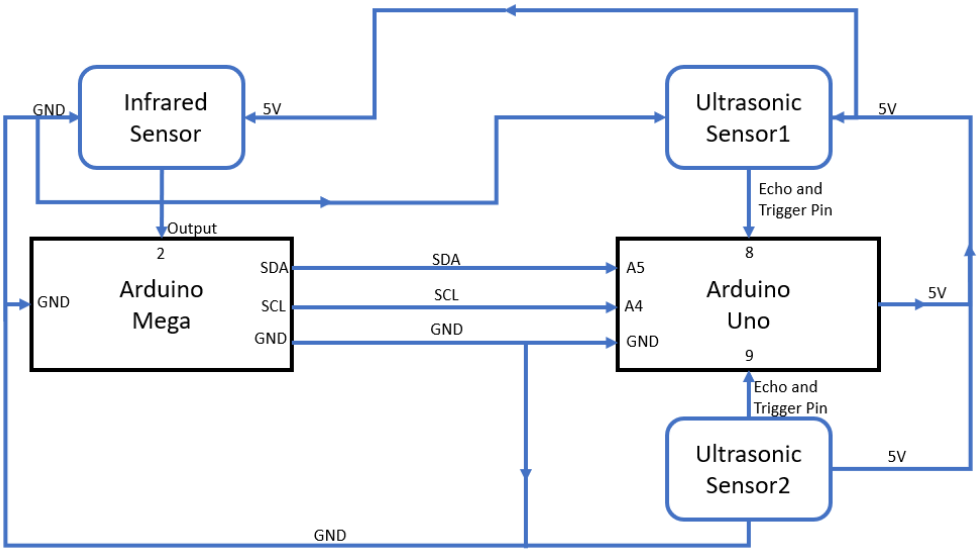

Fig1: Circuit Diagram

In this project, Arduino Mega is the master in the bus and Arduino Uno is the slave. Arduino Mega will continuously check reading from Infrared Sensor. Infrared Sensor emits the Infrared light and the reflected light detected using photodiode, if the light reflects it is considered as Logic 1, whereas if reflected light is not detected, it is considered as Logic 0. In our case, if the IR Sensor logic turns 0, master will start fetching data from slave i.e. Arduino Uno.

Arduino Uno is connected to multiple Ultrasonic Sensors which gives the distance measurement. Ultrasonic sensor works on the principle of reflection. It emits Ultrasonic sound waves, and the reflected waves gives us the distance measurement. As we are using multiple Ultrasonic Sensors, we stored this data in an array and that array is forwarded to Master. (Refer Fig3)

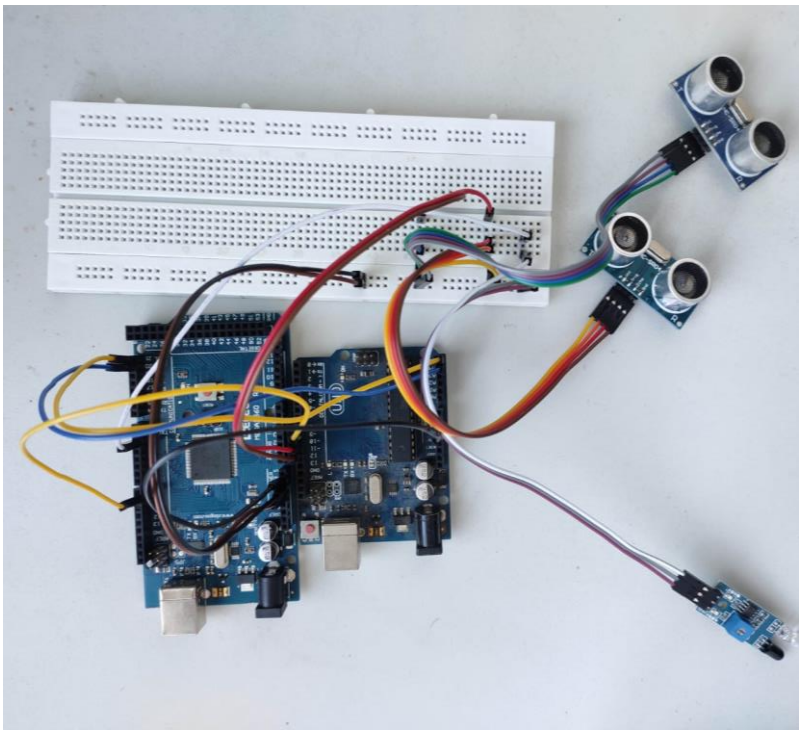

Fig2: Actual Snippet of the Circuit

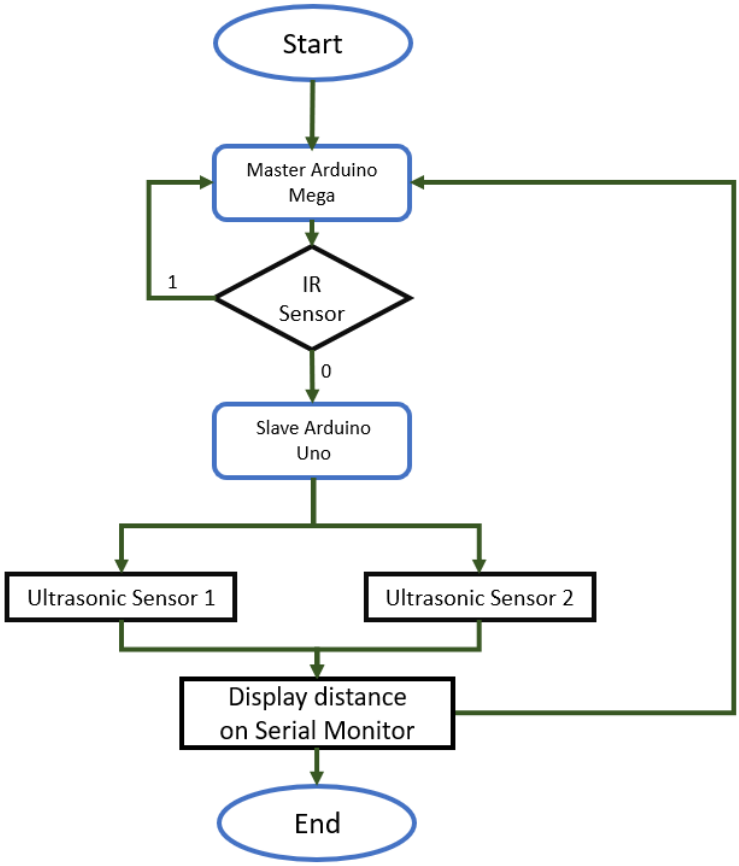

Fig3: Flowchart of the Circuit

IV. SYSTEM DESIGN:

The designing of system consists of following parts:

- Arduino Mega
- Arduino Uno
- Ultrasonic Sensor *4
- Infrared Sensor
- M-M Jumper Wires
- M-F Jumper Wires
- Arduino IDE

Arduino Mega:

Arduino Mega is an Atmel 2560 microcontroller based development board. It operates at 16MHz frequency. The Arduino microcontroller is a sophisticated single-board computer that has achieved popularity in both the Professional and hobby markets. The Arduino is an open-source platform for developing interactive objects that can take input from a variety of switches or sensors and control a variety of lights, motors, and other physical outputs.

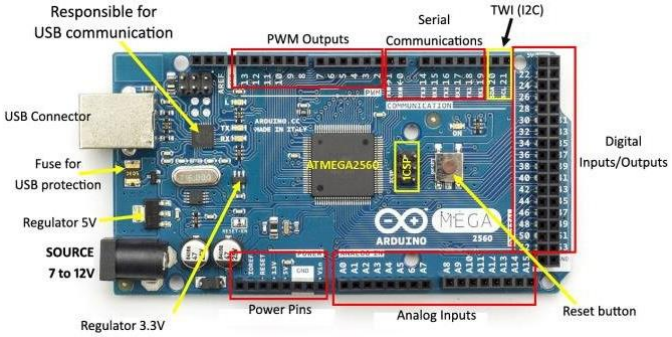

Fig4: Arduino Mega

Arduino Uno:

The Arduino UNO is a flexible, simple-to-use open-source microcontroller board, and low-cost that can be integrated into a wide range of electronic projects. This development board is based on Atmel 328P microcontroller IC.

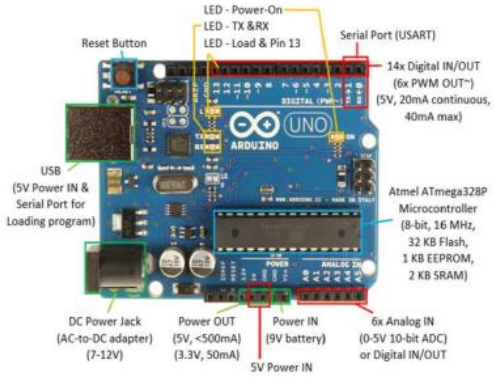

Fig5: Arduino Uno

Ultrasonic Sensor:

An ultrasonic sensor is a device that uses ultrasonic waves to determine the distance between two objects. An ultrasonic sensor employs a transducer to transmit and receive ultrasonic signals that relay information about the proximity of an item.

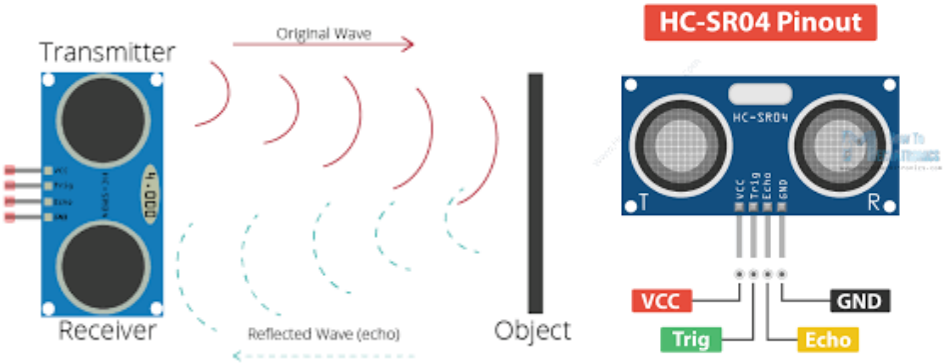

Fig6: Ultrasonic sensor working and pinouts

Infrared Sensor:

An infrared (IR) sensor is an electronic device that measures and detects infrared radiation in its surrounding environment. An IR sensor can detect motion as well as measure the heat of an item. Typically, all objects in the infrared range emit some degree of thermal radiation. These radiations are imperceptible to human eyes, but an infrared sensor can detect them.

Fig7: IR sensor

Arduino IDE:

Arduino IDE is an open-source platform provided to program Arduino microcontrollers. It consists of Integrated Development Environment (IDE) to program µc devices, uploading and writing codes. It is compatible with most of the Operating Systems. It uses java as its programming language, which is based on open-source software and processing.

Ultrasonic_side | Arduino 1.8.15

File Edit Sketch Tools Help

Ultrasonic_side §

```
// Include NewPing Library for HC-SR04 sensor
#include <NewPing.h>

// Include Arduino Wire library for I2C
#include <Wire.h>

// Define Slave I2C Address
#define SLAVE_ADDR 9

// Hook up 4 HC-SR04 sensors in 1-pin mode
// Sensor 0
#define TRIGGER_PIN_0  8
#define ECHO_PIN_0     8

// Sensor 1
#define TRIGGER_PIN_1  13
#define ECHO_PIN_1     13

// Sensor 2
#define TRIGGER_PIN_2  10
#define ECHO_PIN_2     10

// Sensor 3
#define TRIGGER_PIN_3  11
#define ECHO_PIN_3     11

// Maximum Distance is 260 cm
#define MAX_DISTANCE 260

// Create objects for ultrasonic sensors
NewPing sensor0(TRIGGER_PIN_0, ECHO_PIN_0, MAX_DISTANCE);
NewPing sensor1(TRIGGER_PIN_1, ECHO_PIN_1, MAX_DISTANCE);
NewPing sensor2(TRIGGER_PIN_2, ECHO_PIN_2, MAX_DISTANCE);
NewPing sensor3(TRIGGER_PIN_3, ECHO_PIN_3, MAX_DISTANCE);
```

1 Arduino/Genuino Uno on COM9

Fig8: Arduino IDE snippet

V. RESULTS:

The circuit worked and the output on serial monitor is shown in the images below:

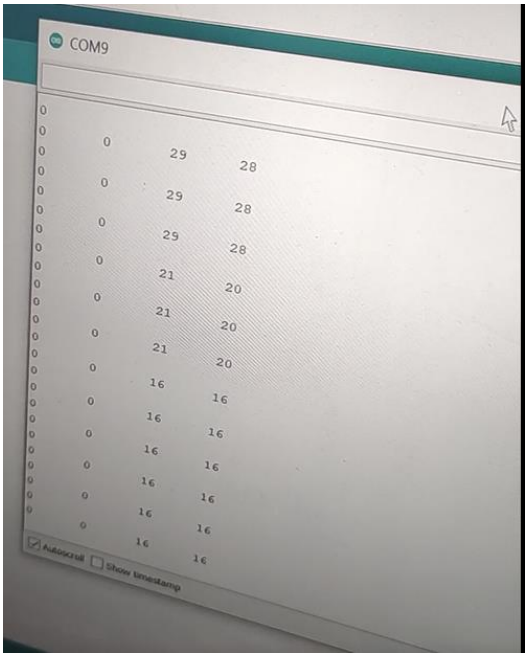

Fig9: Output 1

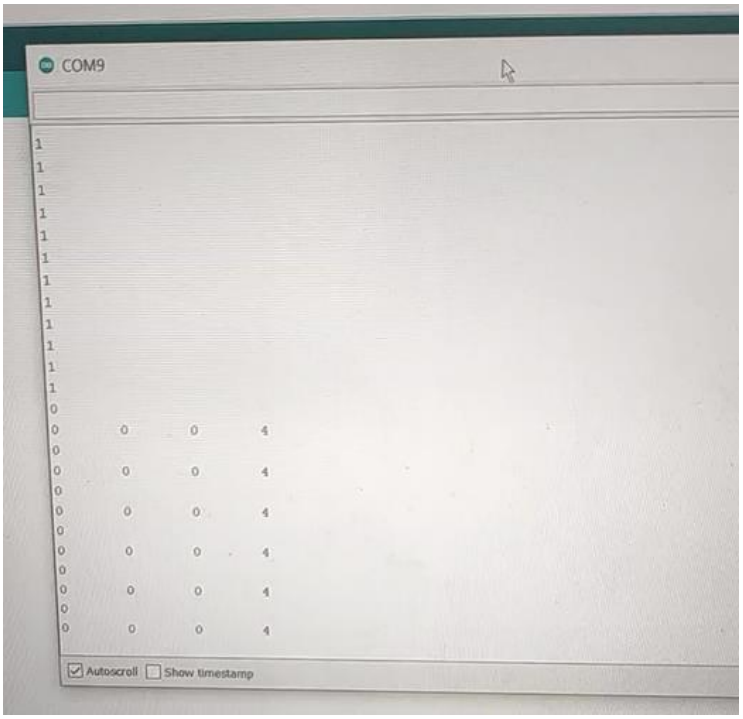

Figure10: Output 2

VI. CONCLUSION:

It this project, I2C based communication was successfully established between the Arduino Mega and Arduino Uno. Where, Arduino Mega was connected to Infrared Sensor and was the Master in the bus. Whereas Arduino Uno was connected to Ultrasonic Sensors was the slave in the bus. Infrared sensor gave Boolean logic of 1 and 0 to the master, i.e., if the transmitted IR light does not reach back on its photodiode (Logic 0), it will start fetching from the slave in this case Arduino Uno and Ultrasonic Sensors. Ultrasonic sensors will emit ultrasonic sound waves and measure the distance which will be sent back to the master Arduino mega.

VII. FUTURE SCOPE

As mentioned earlier, I2C communication has numerous applications in the industry. Stating few of these application as followed:

1. This project with minor changes can be implemented in CNC industry for probes.
2. Currently, processing of data locally is gaining more importance. For this, I2C communication can be use to improve the processing speeds of the embedded systems locally.
3. In the automotive sector, I2C can be used between controller and several peripheral IC's. e.g. reading the data from sensor and saving it in EEPROM.
4. I2C can be used to read certain memory IC's.

VIII. REFERENCES

1. V K Pandey, S Kumar, V Kumar, P Goel, "A Review Paper on I2C Communication Protocol" ISSN: 2454-132X (Volume 4, Issue 2)
2. M P Kumar, "Design and Implementation of I2C BUS Protocol on Xilinx FPGA" oai:generic.eprints.org:39653/core423
3. Radha R C , R A Kumar "Design and Implementation of I2C Communication Protocol on FPGA for EEPROM" International Journal of Scientific & Engineering Research, Volume 5, Issue 3, March-2014 ISSN 2229-5518
4. Prinkle Talan, Anshul Tyagi," I2C BUS PROTOCOL USING VERILOG" 2019 JETIR June 2019, Volume 6, Issue 6
5. Patel, Sagarkumar & Talati, Prachi & Gandhi, Saniya. (2019). Design of I2C Protocol.

IX. CODE OF THE PROJECT

Master Side:

```
// Include Arduino Wire library for I2C
#include <Wire.h>

// Define Slave I2C Address
#define SLAVE_PIN 9
int IRSENSOR = 2;

// Define counter to count bytes in response
int bcount;

// Define array for return data
byte distance[4];

void setup()
{
  pinMode (IRSENSOR, INPUT);
  Wire.begin();
  Serial.begin(9600);
}

byte readI2C(int address) {
  // Define a variable to hold byte of data
  byte bval ;
  long entry = millis();
  // Read one byte at a time
  Wire.requestFrom(address, 1);
  // Wait 100 ms for data to stabilize
  while (Wire.available() == 0 && (millis() - entry) < 100)  Serial.print("Waiting");
  // Place data into byte
  if  (millis() - entry < 100) bval = Wire.read();
  return bval;
}
```

```
void loop()
{
  int statusSensor = digitalRead (IRSENSOR);
  Serial.println(statusSensor);

  if (statusSensor == 0)
 {
   while (readI2C(SLAVE_PIN) < 255) {
 // Until first byte has been received print a waiting message
   Serial.println("Waiting");
   }
 for (bcount = 0; bcount < 4; bcount++) {
   distance[bcount] = readI2C(SLAVE_PIN);
 }
 for (int i = 0; i < 4; i++) {
   Serial.print(distance[i]);
   Serial.print("\t");
 }
 Serial.println();
 delay(200);
 }
 else {
 Serial.println(statusSensor);
 }
}
```

Slave Side:

```
// Include NewPing Library for HC-SR04 sensor
#include <NewPing.h>

// Include Arduino Wire library for I2C
#include <Wire.h>

// Define Slave I2C Address
#define SLAVE_ADDR 9

// Hook up 4 HC-SR04 sensors in 1-pin mode
// Sensor 0
#define TRIGGER_PIN_0  8
#define ECHO_PIN_0     8

// Sensor 1
#define TRIGGER_PIN_1  13
#define ECHO_PIN_1     13

// Sensor 2
#define TRIGGER_PIN_2  10
#define ECHO_PIN_2     10

// Sensor 3
#define TRIGGER_PIN_3  11
#define ECHO_PIN_3     11

// Maximum Distance is 260 cm
#define MAX_DISTANCE 260

// Create objects for ultrasonic sensors
NewPing sensor0(TRIGGER_PIN_0, ECHO_PIN_0, MAX_DISTANCE);
NewPing sensor1(TRIGGER_PIN_1, ECHO_PIN_1, MAX_DISTANCE);
NewPing sensor2(TRIGGER_PIN_2, ECHO_PIN_2, MAX_DISTANCE);
```

```
NewPing sensor3(TRIGGER_PIN_3, ECHO_PIN_3, MAX_DISTANCE);

// Define return data array, one element per sensor
int distance[4];

// Define counter to count bytes in response
int bcount = 0;

void setup() {

  // Initialize I2C communications as Slave
  Wire.begin(SLAVE_ADDR);

   // Function to run when data requested from master
  Wire.onRequest(requestEvent);

}

void requestEvent() {

  // Define a byte to hold data
  byte bval;

  // Cycle through data
  // First response is always 255 to mark beginning
  switch (bcount) {
    case 0:
      bval = 255;
      break;
    case 1:
      bval = distance[0];
      break;
    case 2:
      bval = distance[1];
```

```
      break;
    case 3:
      bval = distance[2];
      break;
    case 4:
      bval = distance[3];
      break;
  }

  // Send response back to Master
  Wire.write(bval);

  // Increment byte counter
  bcount = bcount + 1;
  if (bcount > 4) bcount = 0;

}

void readDistance()
{

  distance[0] = sensor0.ping_cm();
  if (distance[0] > 254 ) {
    distance[0] = 254;
  }
  delay(20);

  distance[1] = sensor1.ping_cm();
  if (distance[1] > 254 ) {
    distance[1] = 254;
  }
  delay(20);

  distance[2] = sensor2.ping_cm();
```

```
  if (distance[2] > 254 ) {
    distance[2] = 254;
  }
  delay(20);

  distance[3] = sensor3.ping_cm();
  if (distance[3] > 254 ) {
    distance[3] = 254;
  }
  delay(20);



}

void loop()
{
  // Refresh readings every half second
  readDistance();
  delay(500);
}
```