LAB REPORT

Course Title: **Embedded Systems and IoT Lab**

Course Code: **CSE-234**

Topic Name: Ultrasonic & PIR Sensors interfacing with Arduino.

Submitted To:

Avizit Nandi
Lecturer
Department of CSE
Daffodil International University

Submitted By:

Name: Rita Faria Richi

ID: 221-15-5025

Section: 61-T2

Department of CSE

Daffodil International University

Date of Submission: 07-08-2025

Lab Report No: 08

Lab Report Name: Ultrasonic & PIR Sensors interfacing with Arduino.

Objective:

- To interface PIR and Ultrasonic sensors with Arduino Uno for motion detection and distance measurement.
- To understand the working principles of PIR (Passive Infrared) and Ultrasonic sensors.
- To acquire real-time sensor data and display it through LED indicators and the Serial Monitor.
- To develop basic skills in digital sensor integration, data processing, and conditional control using Arduino.
- To build a foundational understanding for designing smart automation and IoT-based monitoring systems.

Introduction:

In the era of smart automation and intelligent surveillance, motion detection and distance measurement are essential components of various real-world applications. From security systems, robotics, and smart lighting, to automated parking and obstacle detection, sensors such as the PIR (Passive Infrared) and Ultrasonic sensors play a crucial role. These sensors help detect human presence and measure the distance between objects, allowing devices to interact more intelligently with their surroundings.

✓ The PIR sensor detects motion by sensing infrared radiation changes caused by moving objects (typically humans or animals).

✓ The Ultrasonic sensor measures distance by emitting ultrasonic waves and calculating the time taken for the echo to return.

Both sensors are low-cost, easy to interface with microcontrollers like Arduino Uno, and widely used in educational and prototype-level projects. In this lab, we interface these two sensors with Arduino to detect motion and measure distance, displaying results through the Serial Monitor and indicating activity using an LED. This hands-on experience introduces practical skills in sensor integration, serial communication, and real-time monitoring essential for automation and IoT projects. These sensors serve as the building blocks for developing intelligent systems capable of interacting with their environment. Mastering their interfacing strengthens core skills required for advanced embedded and IoT applications.

Theory:

Ultrasonic sensors use sound waves to measure distances, detect objects, and provide precise range measurements for short to medium distances, while PIR (Passive Infrared) sensors detect movement by sensing changes in infrared radiation from heat sources like humans. Ultrasonic sensors offer accurate distance data, whereas PIR sensors excel at detecting the general presence of moving occupants.

✓ **PIR Sensor Working Principle-**

The PIR sensor detects infrared (IR) light radiated from objects in its field of view. It uses a pair of pyroelectric sensors to detect temperature changes. When a warm body (like a human) passes by, it causes a sudden change in infrared levels, which the sensor detects and sends as a digital HIGH signal to the Arduino.

✓ **PIR Sensor Pinout:**

VCC – Power Supply (usually
5V) OUT – Output Signal
GND – Ground

Figure-1: PIR sensor

✓ **Ultrasonic Sensor Working Principle-**

The Ultrasonic sensor (HC-SR04) operates by emitting an ultrasonic sound wave (usually 40 kHz) through the Trigger pin and receiving the echo through the Echo pin. The time taken for the echo to return is used to calculate the distance to the object.

✓ **Ultrasonic Sensor Pinout-**

VCC – Power Supply (5V)
Trig – Trigger Pulse Input
Echo – Echo Pulse
Output GND – Ground

Figure-2: Ultrasonic Sensor

Equipment:

- Arduino Uno
- USB Cable
- PIR Motion Sensor
- Ultrasonic Sensor (HC-SR04)
- LCD Display
- Resistors
- Breadboard
- Jumper Wires

Circuit Diagram:

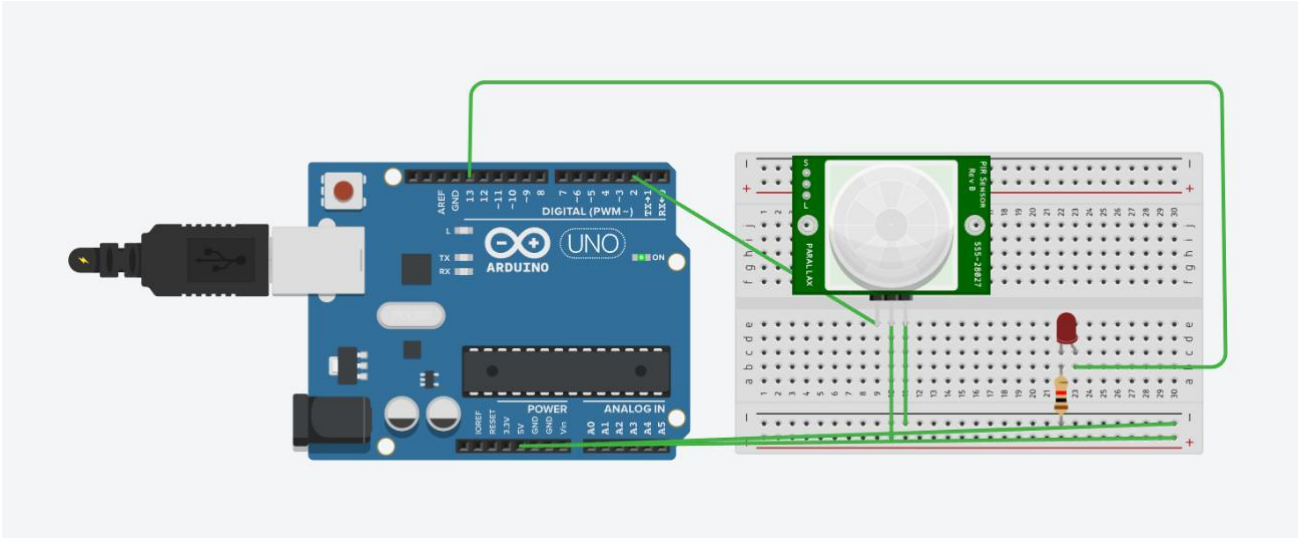

Figure-3: Simulated Output from Tinkercad (using PIR sensor, LED, Register)

Code:

```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);

const int sensorPin = A0;

void setup() {
  lcd.begin(16, 2, 0);
  lcd.backlight();
}

void loop() {
  int reading = analogRead(sensorPin);
  float voltage = reading * (5.0 / 1023.0);
  float temperatureC = (voltage - 0.5) * 100.0;

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Temp: ");
  lcd.print(temperatureC);
  lcd.print(" C");

  delay(1000);
}
```

Figure-4: Arduino Code for PIR sensor (using PIR sensor, LED, Register)

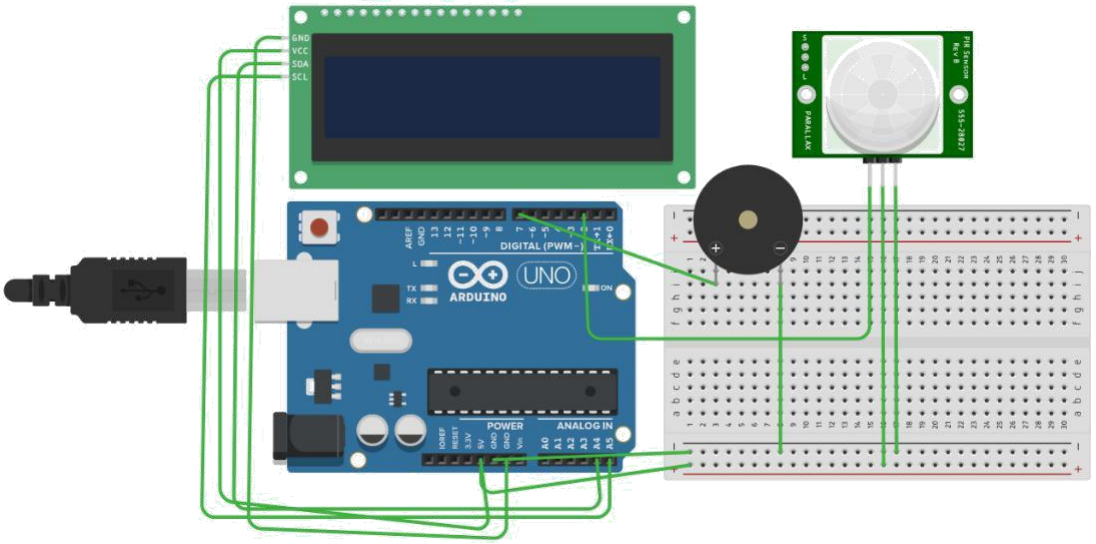

Figure-5: Simulated Output from Tinkercad (using PIR sensor & Buzzer)

Code:

```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);
int pirPin = 2;
int buzzerPin = 7;
void setup() {
  pinMode(pirPin, INPUT);
  pinMode(buzzerPin, OUTPUT);
  lcd.begin();
  lcd.backlight();
  lcd.setCursor(0, 0);
  lcd.print("Motion Detector");
  delay(2000);
}

void loop() {
  int motion = digitalRead(pirPin);

  if (motion == HIGH) {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Intruder Alert!");
    tone(buzzerPin, 1000);
    delay(1000);
    noTone(buzzerPin);
  }
  else{
    lcd.clear();
    lcd.print("Monitoring...");
    delay(500);
  }
}

```

Figure-6: Arduino Code for PIR sensor (using PIR sensor & Buzzer)

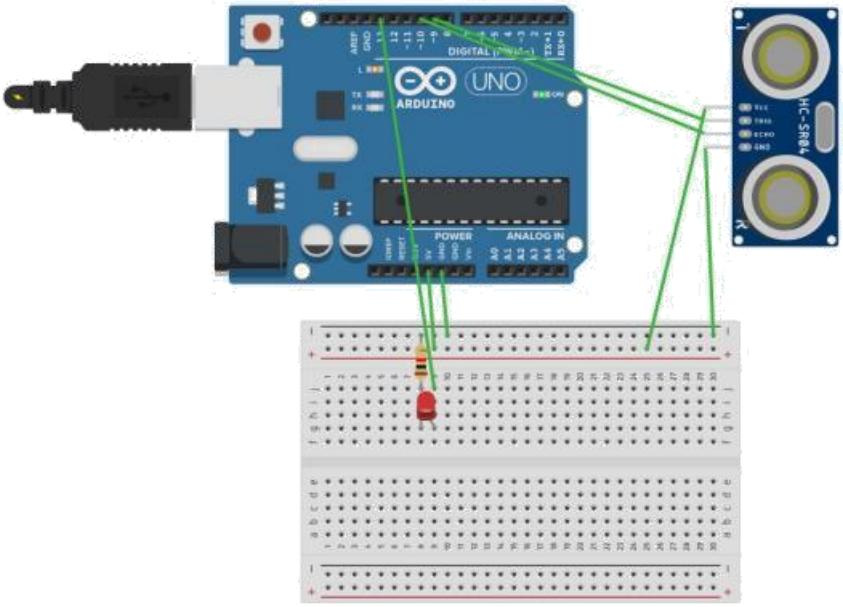

Figure-7: Simulated Output from Tinkercad (using Ultrasonic sensor, LED, Register)

Code:

```
const int trigPin = 10;
const int echoPin = 9;
const int ledPin = 13;
long duration;
float distance;
void setup() {
pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);
pinMode(ledPin, OUTPUT);
Serial.begin(9600);
}
void loop() {
 digitalWrite(trigPin, LOW);
 delayMicroseconds(2);
 digitalWrite(trigPin, HIGH);
 delayMicroseconds(10);
 digitalWrite(trigPin, LOW);
 duration = pulseIn(echoPin, HIGH);
 distance = (duration * 0.0343) / 2;
 Serial.print("Distance: ");
 Serial.print(distance);
 Serial.println(" cm");
 digitalWrite(ledPin, HIGH);
 delay(100);
 digitalWrite(ledPin, LOW);
 delay(400);
```

Figure-8: Arduino Code for Ultrasonic sensor (using Ultrasonic sensor, LED, Register)

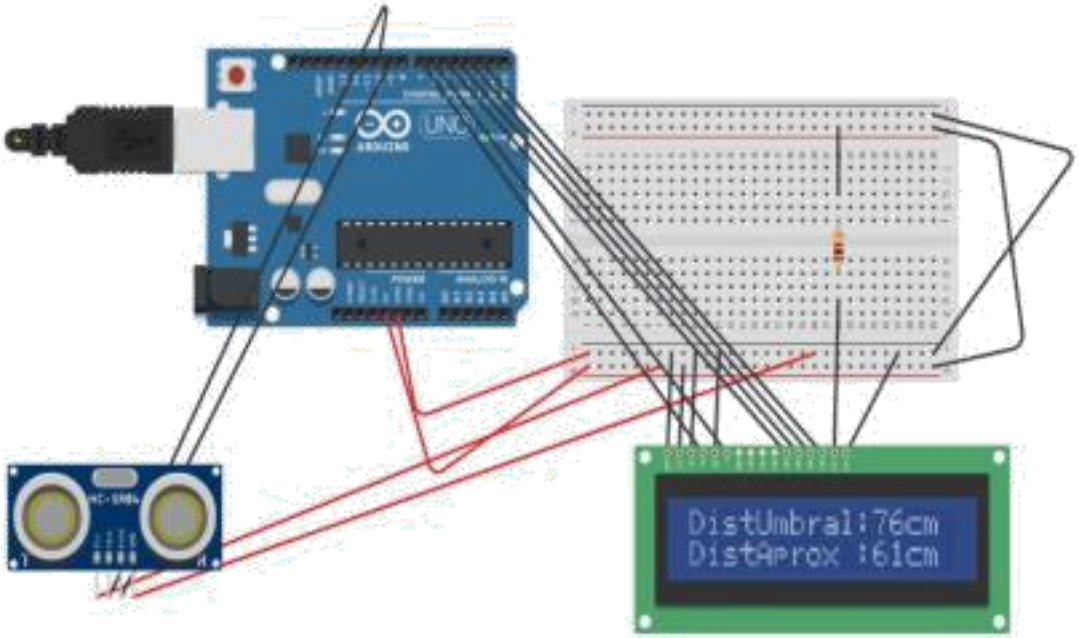

Figure-9: Simulated Output from Tinkercad (using Ultrasonic Sensor)

Code:

```
#include <LiquidCrystal.h>
#define ECHO 9
#define TRIGGER 10
#define TIEMPO_MUESTREO 1000
#define PULSO_TRIGGER 10
int DURACION;
float DISTANCIA;
float DISTUmb;
LiquidCrystal lcd(7, 6, 5, 4, 3, 2);
void setup() {
  pinMode(TRIGGER, OUTPUT);
  pinMode(ECHO, INPUT);
  Serial.begin(9600);
  lcd.begin(16, 2);
  lcd.setCursor(0, 0);
  lcd.print("DistUmbral:");
  lcd.setCursor(0, 1);
  lcd.print("DistAprox :");
}
```

Figure-10(A) : Arduino Code for Ultrasonic sensor

```
void loop() {
  digitalWrite(TRIGGER, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIGGER, HIGH);
  delayMicroseconds(PULSO_TRIGGER);
  digitalWrite(TRIGGER, LOW);

  DURACION = pulseIn(ECHO, HIGH);
  DISTANCIA = 0.01716 * DURACION;
  DISTUmb = 1.25 * DISTANCIA;
  Serial.print("Distancia: ");
  Serial.print(DISTANCIA);
  Serial.println(" cm");
  lcd.setCursor(11, 0);
  lcd.print("     ");
  lcd.setCursor(11, 1);
  lcd.print("     ");
  lcd.setCursor(11, 0);
  lcd.print(int(DISTUmb));
  lcd.print("cm");
  lcd.setCursor(11, 1);
  lcd.print(int(DISTANCIA));
  lcd.print("cm");


  delay(TIEMPO_MUESTREO);
}
```

Figure-10(B): Arduino Code for Ultrasonic sensor

Discussion:

In this lab, we interfaced two commonly used sensors—PIR and Ultrasonic—with an Arduino Uno. The goal was to detect motion and measure distance accurately.

For the PIR sensor, the Arduino code continuously monitors the sensor's output. When motion is detected (HIGH signal), an LED is turned ON to indicate presence. The digitalRead() function is used to detect the output from the sensor. For the Ultrasonic sensor, the digitalWrite() and pulseIn() functions are used to send a trigger pulse and measure the time taken for the echo. The measured time is converted to distance and printed to the Serial Monitor. A delay is used to prevent overwhelming the serial output. From the coding and interfacing experience, we learned to:

- ✓ Read digital input from sensors.
- ✓ Calculate real-time distance using timing functions.
- ✓ Display sensor data using Serial Monitor.
- ✓ Use conditional statements to perform actions based on sensor output.

This experiment introduced fundamental principles of motion detection and distance sensing, paving the way for security systems, robotics, and IoT automation.

Conclusion:

The integration of PIR and Ultrasonic sensors with Arduino Uno successfully demonstrated basic concepts in motion detection and real-time distance measurement.

- Successfully interfaced both PIR and Ultrasonic sensors with Arduino.
- Detected human motion using PIR sensor and responded with LED indication.
- Accurately measured distance using Ultrasonic sensor and displayed it via Serial Monitor.
- Gained practical experience in sensor interfacing, signal processing, and timing control.
- Built foundational skills essential for smart automation, robotics, and IoT systems.

Overall, this lab helped reinforce essential embedded system skills, preparing us for more advanced projects in motion sensing and intelligent systems.