NAME: ANANYA PILLAI

Q1) LED & BUZZER INTERFACING WITH ARDUINO

1. The wire connect to ground, then the same wire attach to board.

2. The wire connect to buzzer short leg, then the same wire attach to GND on the board.

3. The wire attach to LED short leg, then the same wire connect to GND on the board.

4. The wire connect to 10k resistor empty leg, then the same wire connect to GND on the board.

5. The wire connect to +5V, then the same wire attach to LDR empty leg.

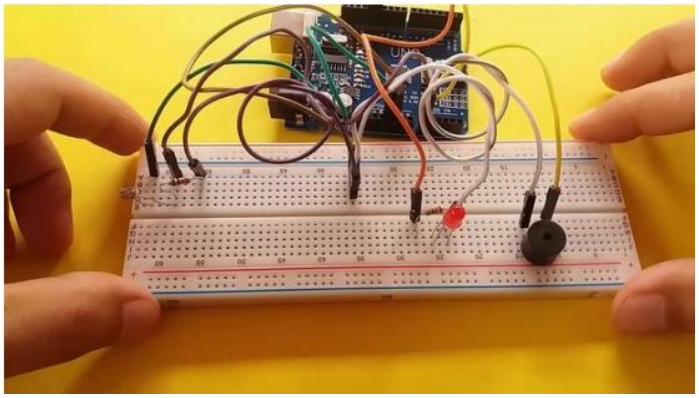
6. The wire connect to digital 12, then attach to buzzer long leg.

7. The wire connect to digital 13, then attach to 220 resistor empty leg.

8. The wire connect to A0, then attach to LDR's - resistor's same column.

COMPONENTS REQUIRED:

- Arduino Uno
- Buzzer
- LED
- LDR (photoresistor)
- 220 and 10k ohm resistor
- Wires
- Breadboard



CODE:

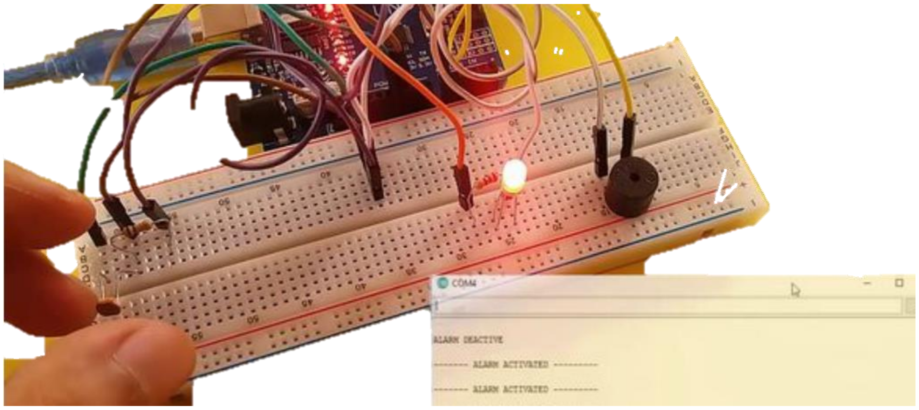
```
const int ledPin = 13;

const int buzzerPin = 12;

const int ldrPin = A0;

void setup () {
```

```
Serial.begin(9600);

pinMode(ledPin, OUTPUT);

pinMode(buzzerPin, OUTPUT);

pinMode(ldrPin, INPUT); }

void loop() {

int ldrStatus = analogRead(ldrPin);

if (ldrStatus >= 400) {

tone(buzzerPin, 100);

http://www.instructables.com/id/Arduino-Buzzer-With-LDR-and-LED/

digitalWrite(ledPin, HIGH);

delay(100);

noTone(buzzerPin);

digitalWrite(ledPin, LOW);

delay(100);

Serial.println("----------- ALARM ACTIVATED -----------");}

else {

noTone(buzzerPin);

digitalWrite(ledPin, LOW);

Serial.println("ALARM DEACTIVATED"); } }
```

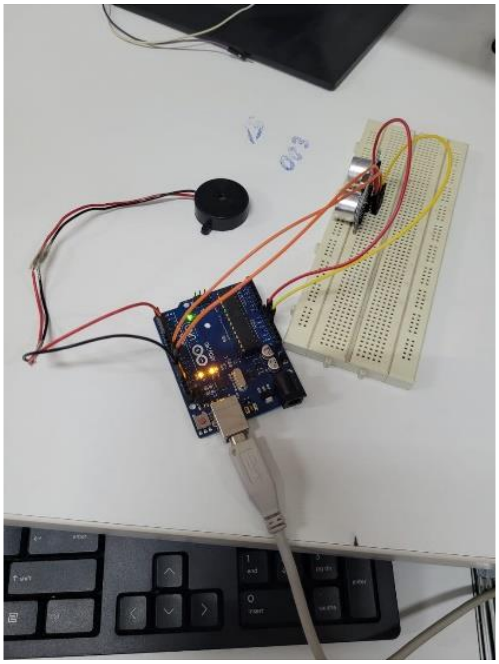
OR

```
const int buzzer = 9;
  void setup(){

    pinMode(buzzer, OUTPUT);  }
  void loop(){

    tone(buzzer, 1000);


    delay(1000);

    noTone(buzzer);

    sound...delay(1000);

  }
```

UPLOADING AND TESTING :



```
ALARM DEACTIVE

------- ALARM ACTIVATED ---------

------- ALARM ACTIVATED ---------
```



Q2) INTERFACING ULTRASONIC SENSOR(Smart Dustbin,Buzzer) WITH ARDUINO

An Ultrasonic Sensor is a device that measures distance to an object using Sound Waves. It works by sending out a sound wave at ultrasonic frequency and waits for it to bounce back from the object. Then, the time delay between transmission of sound and receiving of the sound is used to calculate the distance. We divide the distance formula by 2 because the sound waves travel a round trip i.e from the sensor and back to the sensor which doubles the actual distance. The HC-SR04 is a typical ultrasonic sensor which is used in many projects such as obstacle detector and electronic distance measurement tapes. In this Instructable I'll teach you how to interface the HC-SC04 with an Arduino Uno.

FORMULA: Distance = (Speed of sound * Time delay) / 2

The HC-SR04 is an ultrasonic ranging module. This economical sensor provides 2cm to 400cm of non-contact measurement functionality with a ranging accuracy that can reach up to 3mm. Each HC-SR04 module includes an ultrasonic transmitter, a receiver and a control circuit. There are Four Pins on the HC-SR04. They are :

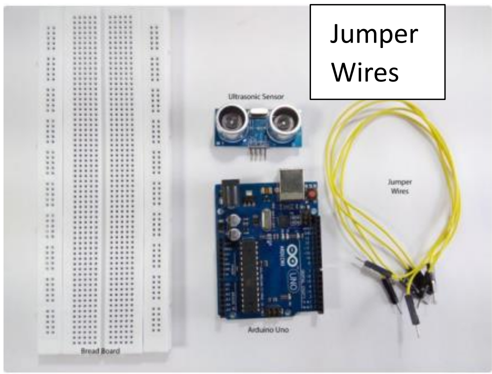
- Vcc (5V supply)
- Gnd (Ground)
- Trig (Trigger)
- Echo (Receive) .

The key features to be noted are:

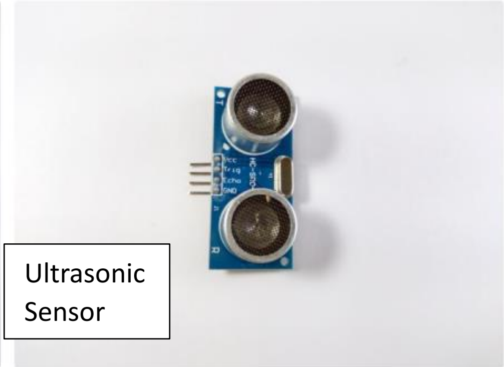
- Operating Voltage: 5V DC   Measure Angle: 15°
- Ranging Distance: 2cm - 4m   Operating Current: 15mA

COMPONENTS REQUIRED:

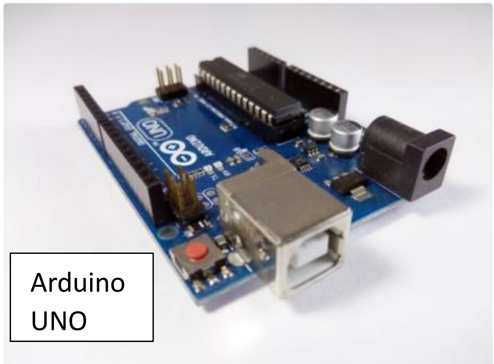
To interface an Ultrasonic Sensor with an Arduino and view the distance on the serial monitor you'll need: Arduino Uno HC-SR04 Module BreadBorad Jumper wires You'll need a laptop or a PC to upload code to the Arduino and view the Distance on the Serial Monitor.



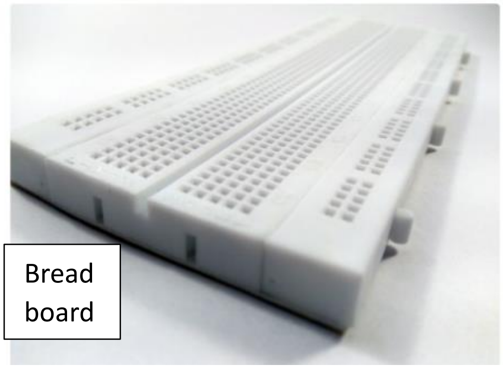
Jumper Wires



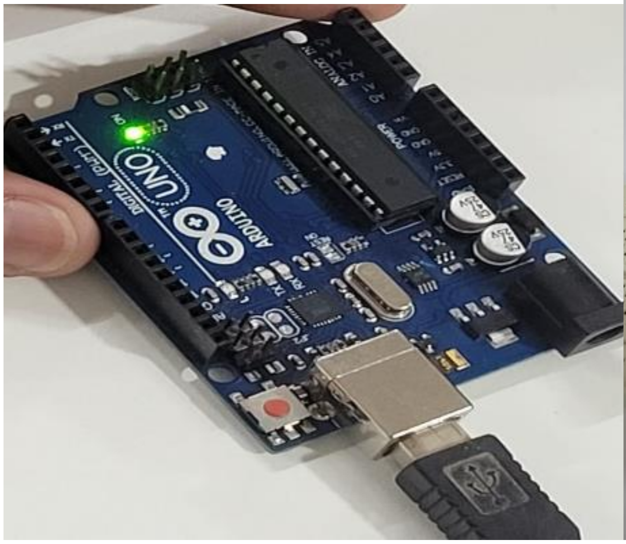
Ultrasonic Sensor

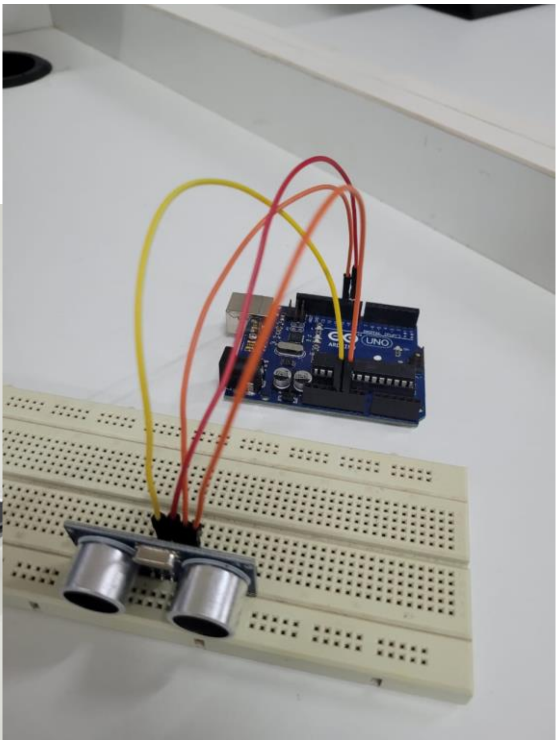


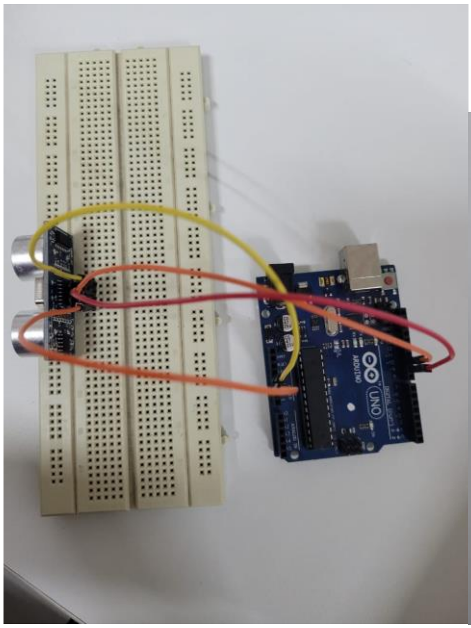
Arduino UNO

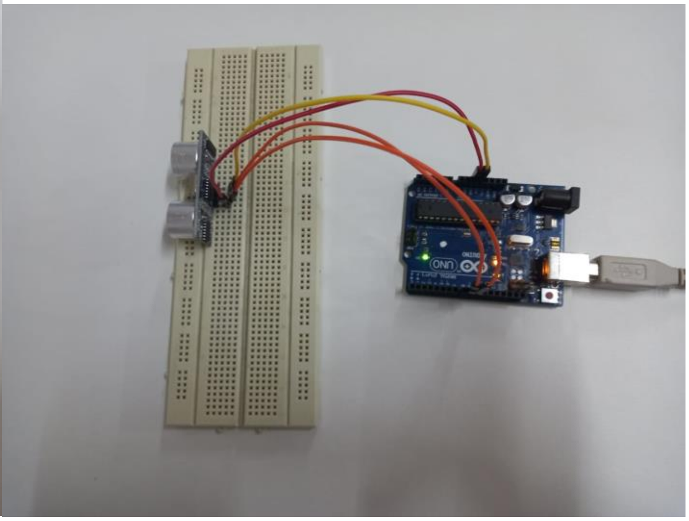


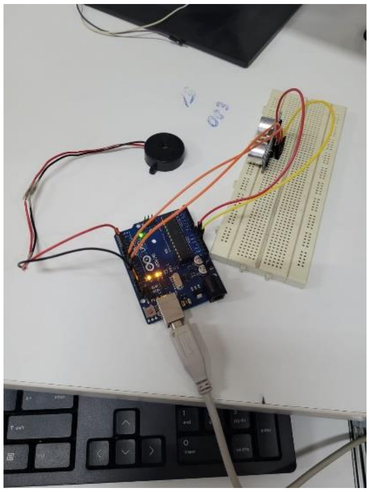
Bread board











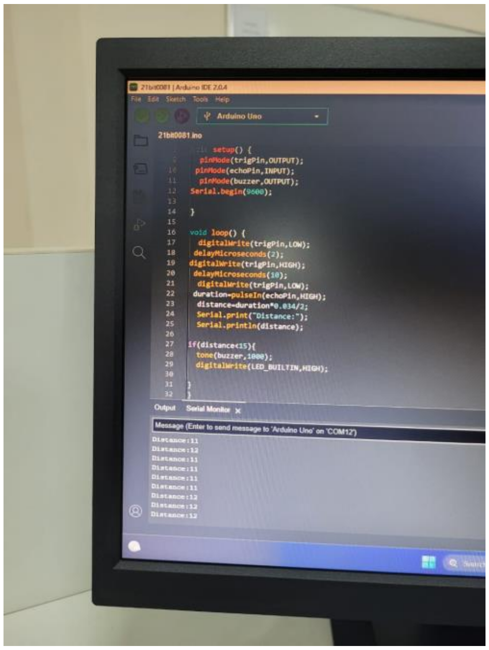
CODE:

```
// Ultrasonic Sensor HC-SR04 interfacing with Arduino.

// defining the pins

const int trigPin = 9;

const int echoPin = 10;

// defining variables

long duration;

int distance;

void setup() {

pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output

pinMode(echoPin, INPUT); // Sets the echoPin as an Input

Serial.begin(9600); // Starts the serial communication }
```

```
void loop() {
// Clears the trigPin
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
// Sets the trigPin on HIGH state for 10 micro seconds
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*0.034/2;
// Prints the distance on the Serial Monitor
Serial.print("Distance: ");
Serial.println(distance);}
If(distance<15){
tone(buzzer,1000);
digitalWrite(LED_BUILTIN,HIGH);
}}
```

UPLOADING AND TESTING :

Once you've uploaded the code, the board will begin to transmit data to the computer. You will know this when you see the Tx LED on the Arduino blinking each time it transmits a data. Now if you open the Serial Monitor, you'll see the distance being displayed.


21bit0081 | Arduino IDE 2.0.4
File Edit Sketch Tools Help
Arduino Uno
21bit0081.ino
Output
Serial Monitor
Message (Enter to send message to 'Arduino Uno' on 'COM12')

```
  setup() {
   pinMode(trigPin,OUTPUT);
  pinMode(echoPin,INPUT);
   pinMode(buzzer,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*0.034/2;
  Serial.print("Distance:");
  Serial.println(distance);

if(distance<15){
  tone(buzzer,1000);
  digitalWrite(LED_BUILTIN,HIGH);

}
}
```

```
Distance:11
Distance:12
Distance:11
Distance:11
Distance:11
Distance:11
Distance:12
Distance:12
Distance:12
```

a) SMART DUSTBIN:

Smart Dustbin as its name represents its work smartly or we can say that it is an **automatic dustbin**. Smart Dustbin is a very good project from the Arduino. We can say that It is a decent gadget to make your home clean and attractive. kids spread trash to a great extent by paper, rappers and numerous different things at home. They will have fun with this dustbin they play with the dustbin and in the play of them, they clean your home as well because every time they use the smart Dustbin and it attracts the kids. They generally will be utilized to throw all trash and waste into this smart dustbin. if your hands are full of trash or junk and you're unable to open it manually this can be used. As it **opens automatically without touching**, this can help to control the spread of Coronavirus and even mosquitoes will not move around it so that it helps from preventing the spread of new diseases.

COMPONENTS REQUIRED:

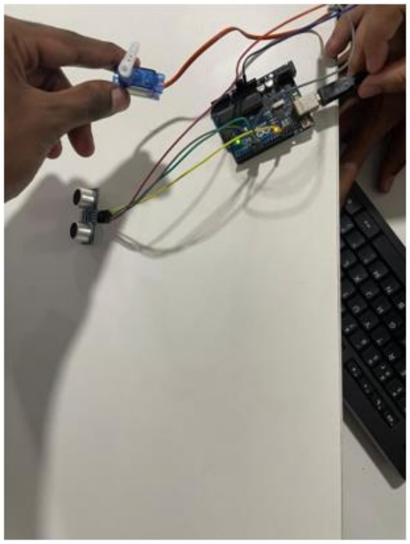
- Arduino UNO,
- Jumper Wires,
- Servo Motor,
- Ultrasonic Sensor

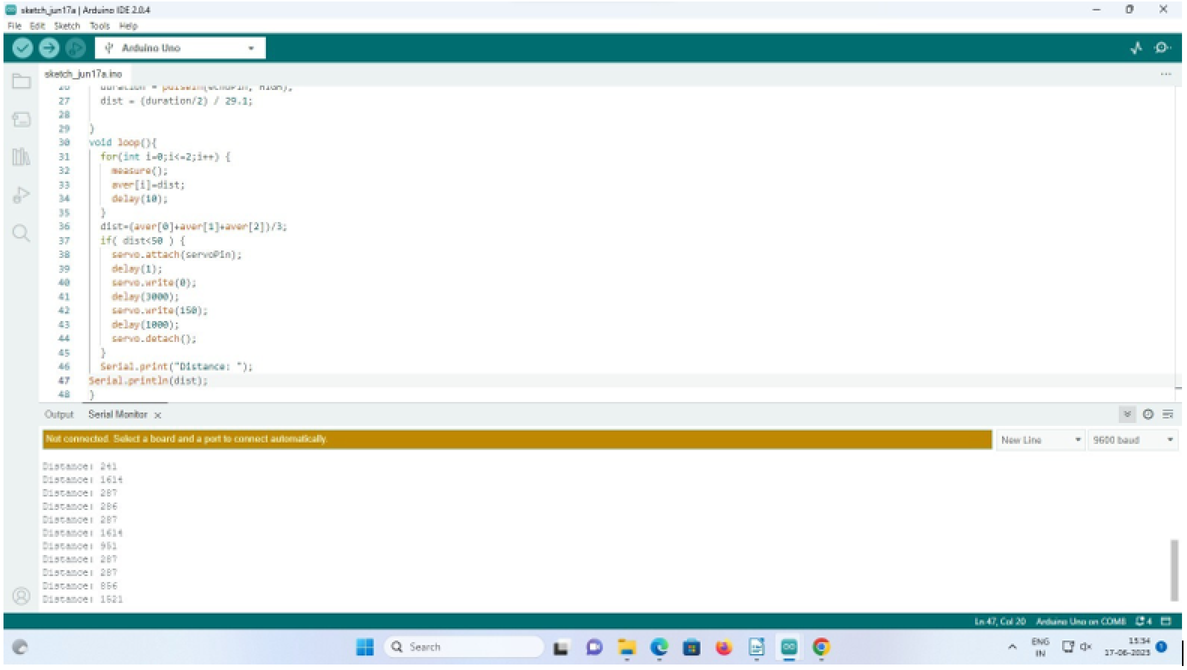
CODE:

```
// defining the pins
#include <Servo.h>
Servo servo1; // Create object for Servo motor 1int
position = 0;
const int trigPin = 9;
const int echoPin = 10;
// defining variables
long duration;
int distance;
```

```
void setup() {
pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
pinMode(echoPin, INPUT); // Sets the echoPin as an Input
Serial.begin(9600); // Starts the serial communication
servo1.attach(3);
}
void loop() {
// Clears the trigPin
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
// Sets the trigPin on HIGH state for 10 micro seconds
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*0.034/2;
// Prints the distance on the Serial Monitor
Serial.print("Distance: ");
Serial.println(distance);
if (distance<20)
{
for (position = 0; position <= 180; position++)
{
servo1.write(position); // Set position of Servo motor 1
delay(3);
}
// Rotating Servo motor 1 in clockwise from 180 degree to 0 degreefor
(position = 180; position >= 0; position--)
{
servo1.write(position); // Set position of Servo motor 1delay(15); // Short delay to control the speed }
} }
```

UPLOADING AND TESTING :




sketch_jun17a | Arduino IDE 2.0.4
File Edit Sketch Tools Help
Arduino Uno
sketch_jun17a.ino
Output
Serial Monitor
New Line
9600 baud
Ln 47, Col 20 Arduino Uno on COM8
Search
ENG IN
15:34
17-06-2023

```
  duration = pulseIn(echoPin, HIGH);
  dist = (duration/2) / 29.1;

}
void loop(){
  for(int i=0;i<=2;i++) {
    measure();
    aver[i]=dist;
    delay(10);
  }
  dist=(aver[0]+aver[1]+aver[2])/3;
  if( dist<50 ) {
    servo.attach(servoPin);
    delay(1);
    servo.write(0);
    delay(3000);
    servo.write(150);
    delay(1000);
    servo.detach();
  }
  Serial.print("Distance: ");
Serial.println(dist);
}
```

Not connected. Select a board and a port to connect automatically.

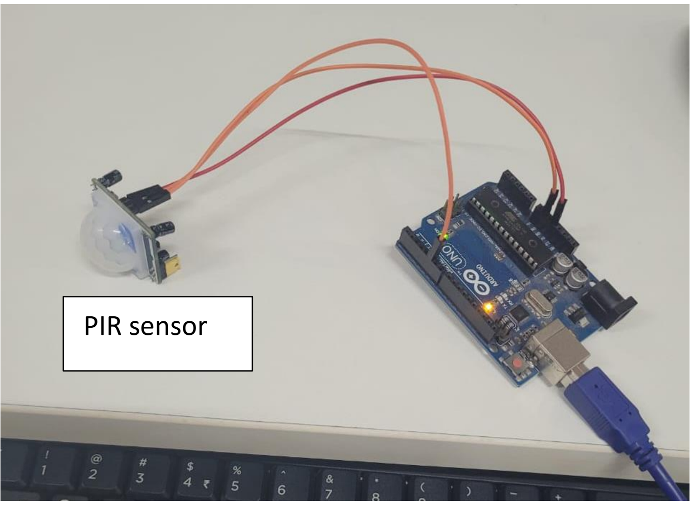
```
Distance: 241
Distance: 1614
Distance: 287
Distance: 286
Distance: 287
Distance: 1614
Distance: 951
Distance: 287
Distance: 287
Distance: 856
Distance: 1521
```

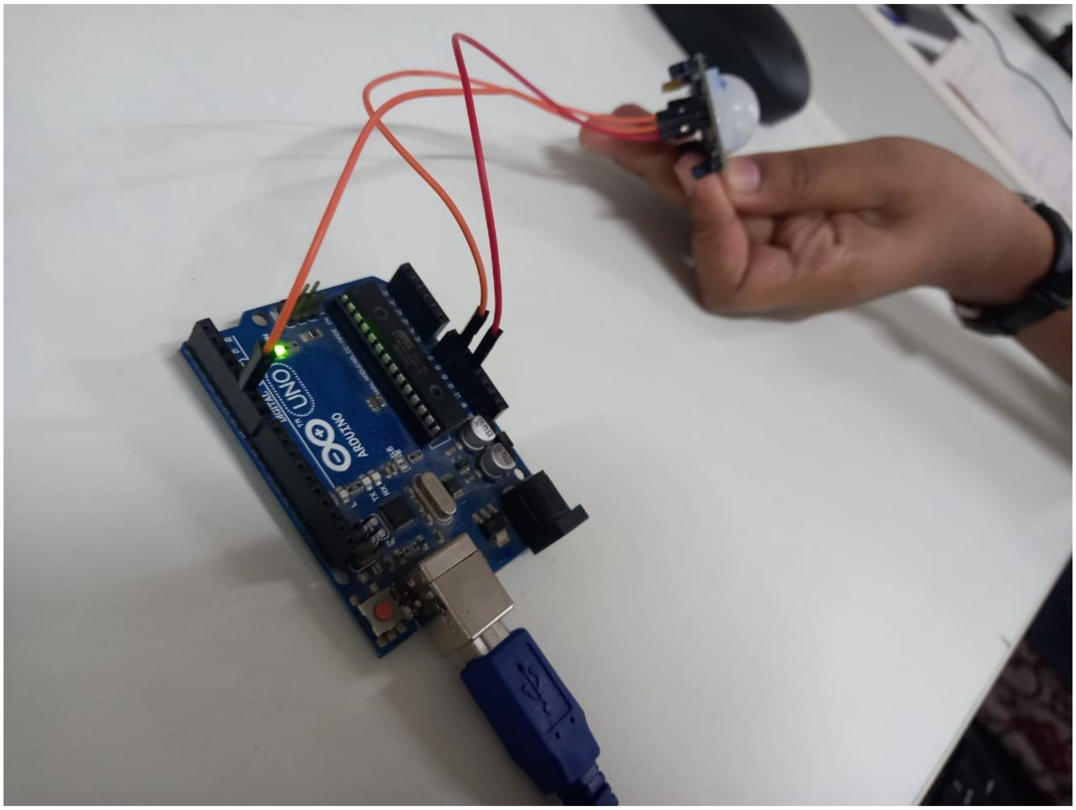
Q3) PIR MOTION SENSOR WITH ARDUINO.

A passive infrared sensor (PIR) is an electronic sensor which is used to detect motion. When a person in the field of the sensor moves, it detects a sudden change in infrared energy and the sensor is triggered (activated). It acts as a switch. They don't detect or measure Heat, but they detect infrared radiation emitted from objects. These sensors commonly used in security, lighting, and alarm systems. The range of PIR sensors is approximately 6 meters, depending on conditions. When PIR detects the motion of any object the output voltage is high because of its sense infrared radiation whereas it is low when there is no motion.

COMPONENTS REQUIRED:

Arduino UNO. PIR Sensor. Jumper Wires.





PIN CONNECTIONS OF PIR WITH ARDUINO

Arduino UNO	PIR Sensor
5v	Vcc
GND	GND
D5	Out

CODE:

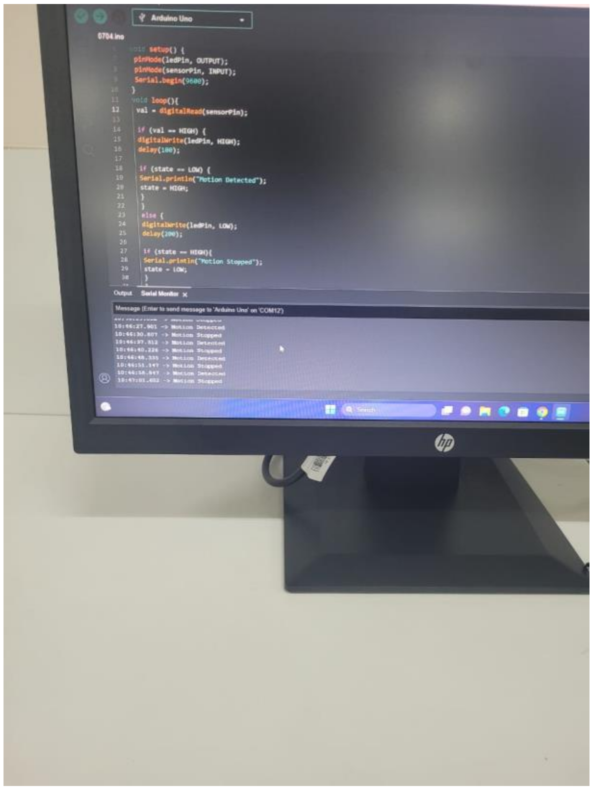
```
int ledPin = 13; // Attach LED Pin
int sensorPin = 5; // Attach sensor out pin to Arduino D5
int state = LOW; // Motion State
int val = 0; // sensorPin value
void setup() {
 pinMode(ledPin, OUTPUT);
 pinMode(sensorPin, INPUT);
 Serial.begin(9600);
}
void loop(){
 val = digitalRead(sensorPin);
```

PIR Motion Sensor with Arduino Code2.ino hosted with by GitHub view raw

Code 3

```
 if (val == HIGH) {
 digitalWrite(ledPin, HIGH);
 delay(100);
 if (state == LOW) {
 Serial.println("Motion Detected");
 state = HIGH;
 }
 }
 else {
 digitalWrite(ledPin, LOW);
delay(200);
 if (state == HIGH){
 Serial.println("Motion Stopped");
 state = LOW;
 } } }
```

UPLOADING AND TESTING :



```
void setup() {
 pinMode(ledPin, OUTPUT);
 pinMode(sensorPin, INPUT);
 Serial.begin(9600);
}
void loop(){
 val = digitalRead(sensorPin);

 if (val == HIGH) {
 digitalWrite(ledPin, HIGH);
 delay(100);

  if (state == LOW) {
  Serial.println("Motion Detected");
  state = HIGH;
  }
 }
  else {
  digitalWrite(ledPin, LOW);
  delay(200);

   if (state == HIGH){
   Serial.println("Motion Stopped");
   state = LOW;
   }
```

Message (Enter to send message to 'Arduino Uno' on 'COM12')

```
10:46:27.901 -> Motion Detected
10:46:30.807 -> Motion Stopped
10:46:37.312 -> Motion Detected
10:46:40.226 -> Motion Stopped
10:46:48.335 -> Motion Detected
10:46:51.147 -> Motion Stopped
10:46:58.847 -> Motion Detected
10:47:01.652 -> Motion Stopped
```

USES OF PIR SENSOR:

Automatic Doors which open and close on Motion Detection. All Outdoor Lights. Lift Lobby. Motion Cameras. Lights on/off by Motion Detection. Basement Parking Areas. Escalator Lights(Staircase). Security Alarm based on Motion Detection.

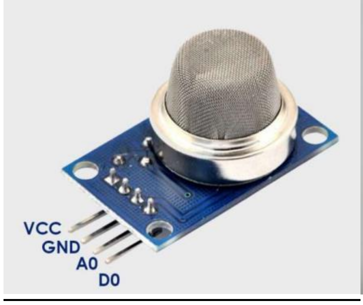
Q4) SMOKE SENSOR

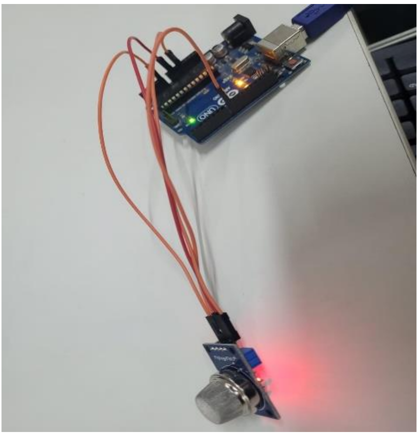
- VCC provides power to the sensor comparator board and needs to be connected to the 5V of the Arduino
- GND is the ground pin and needs to be connected to the GND pin of the Arduino
- D0 is the digital output pin which shows the digital representation of the detected gas

- A0 is the analog output pin from which we can detect the gas type by analysing analogue values
- Connect the digital pin of the sensor D0 to the digital pin number 8 of the Arduino
- connect the analog pin of the sensor to the analog pin A0 of the Arduino

COMPONENTS REQUIRED:

- Arduino Uno
- connecting cable
- breadboard
- MQ2 gas sensor
- DuPont cable



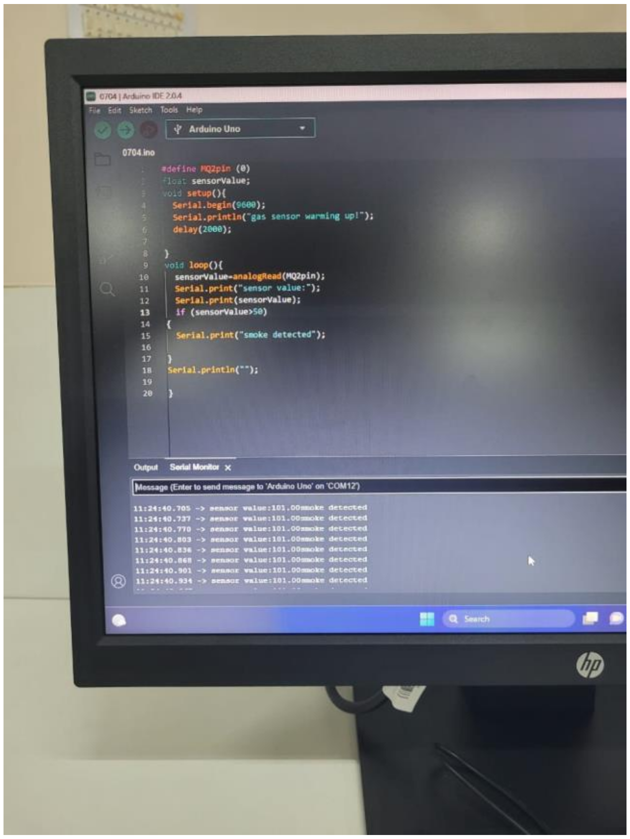


CODE:

```
#define MQ2pin (0)

float sensorValue;

void setup(){

Serial.begin(9600);

Serial.println("Gas sensor warming up!");

Delay(20000);}

void loop(){

sensorValue=analogRead(MQ2pin);

Serial.print("Sensor value:");

Serial.print(sensorValue);

If(sensorValue>200)

{
```

```
Serial.print("Smoke detected!");
}
Serial.println("");
```

UPLOADING AND TESTING :



0704 | Arduino IDE 2.0.4
File Edit Sketch Tools Help
Arduino Uno
0704.ino
Output
Serial Monitor
Message (Enter to send message to 'Arduino Uno' on 'COM12')
Search

```
#define MQ2pin (0)
float sensorValue;
void setup(){
  Serial.begin(9600);
  Serial.println("gas sensor warming up!");
  delay(2000);

 }
 void loop(){
  sensorValue=analogRead(MQ2pin);
  Serial.print("sensor value:");
  Serial.print(sensorValue);
  if (sensorValue>50)
 {
  Serial.print("smoke detected");

 }
 Serial.println("");

 }
```

```
11:24:40.705 -> sensor value:101.00smoke detected
11:24:40.737 -> sensor value:101.00smoke detected
11:24:40.770 -> sensor value:101.00smoke detected
11:24:40.803 -> sensor value:101.00smoke detected
11:24:40.836 -> sensor value:101.00smoke detected
11:24:40.868 -> sensor value:101.00smoke detected
11:24:40.901 -> sensor value:101.00smoke detected
11:24:40.934 -> sensor value:101.00smoke detected
```

Q5) TEMPERATURE

The Temperature Sensor LM35 series are precision integrated-circuit temperature devices with an output voltage linearly proportional to the Centigrade temperature.

The LM35 device has an advantage over linear temperature sensors calibrated in Kelvin, as the user is not required to subtract a large constant voltage from the output to obtain convenient Centigrade scaling. The LM35 device does not require any external calibration or trimming to provide typical accuracies of ±¼°C at room temperature and ±¾°C over a full −55°C to 150°C temperature range.

LM35 sensor has three terminals - V_s, V_{out} and GND. We will connect the sensor as follows −

Connect the +V_S to +5v on your Arduino board.
Connect V_{out} to Analog0 or A0 on Arduino board.
Connect GND with GND on Arduino.

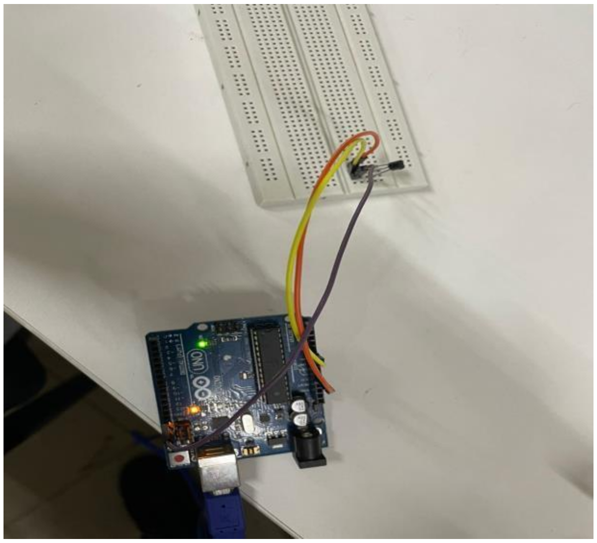
The Analog to Digital Converter (ADC) converts analog values into a digital approximation based on the formula ADC Value = sample * 1024 / reference voltage (+5v). So with a +5 volt reference, the digital approximation will be equal to input voltage * 205.

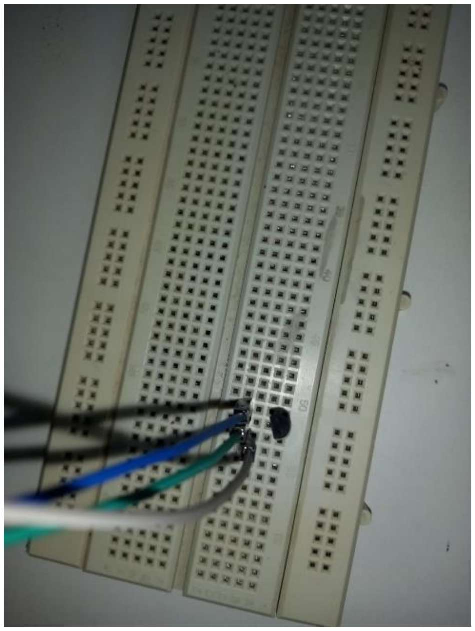
TECHNICAL SPECIFICATIONS:

Calibrated directly in Celsius (Centigrade)
Linear + 10-mV/°C scale factor
0.5°C ensured accuracy (at 25°C)
Rated for full −55°C to 150°C range
Suitable for remote applications

COMPONENTS REQUIRED:

1 × Breadboard
1 × Arduino Uno R3
1 × LM35 sensor

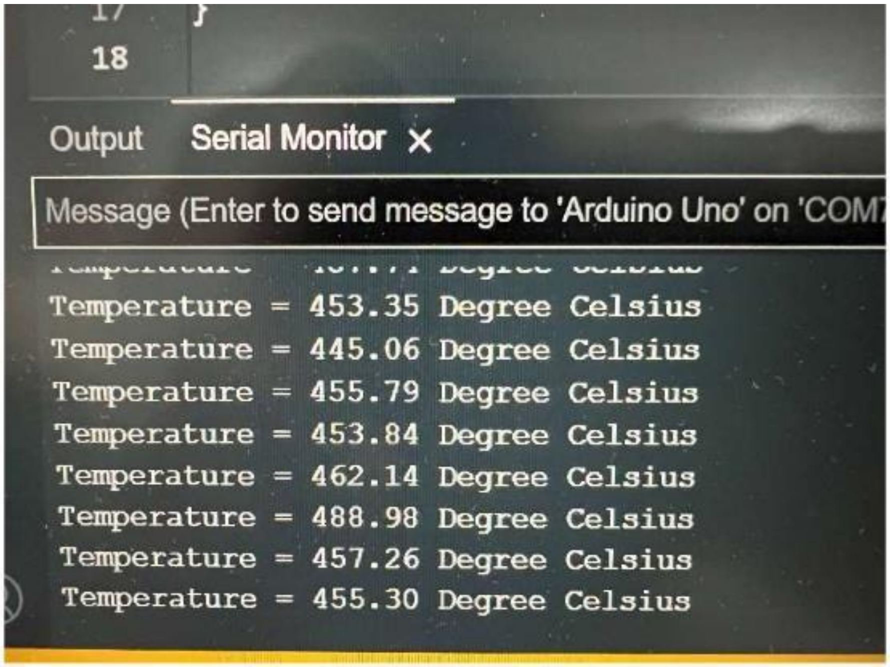




CODE:

```
Float temp
Int tempPin = 0;
Void setup(){
Serial.begin(9600);
}
Void loop(){
Temp = analogRead(tempPin);
Temp = temp*0.48828195;
Serial.print(“TEMPERATURE = “);
Serial.print(temp);
Serial.print(“Degree Celsius\n”);
Serial.println();
Delay(1000);
```

UPLOADING AND TESTING :

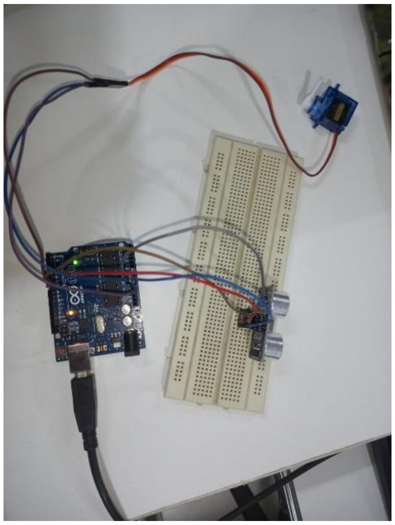


```
}
```

```
Temperature = 453.35 Degree Celsius
Temperature = 445.06 Degree Celsius
Temperature = 455.79 Degree Celsius
Temperature = 453.84 Degree Celsius
Temperature = 462.14 Degree Celsius
Temperature = 488.98 Degree Celsius
Temperature = 457.26 Degree Celsius
Temperature = 455.30 Degree Celsius
```

Q6) SERVO MOTOR INTERFACING WITH ARDUINO

COMPONENTS REQUIRED:

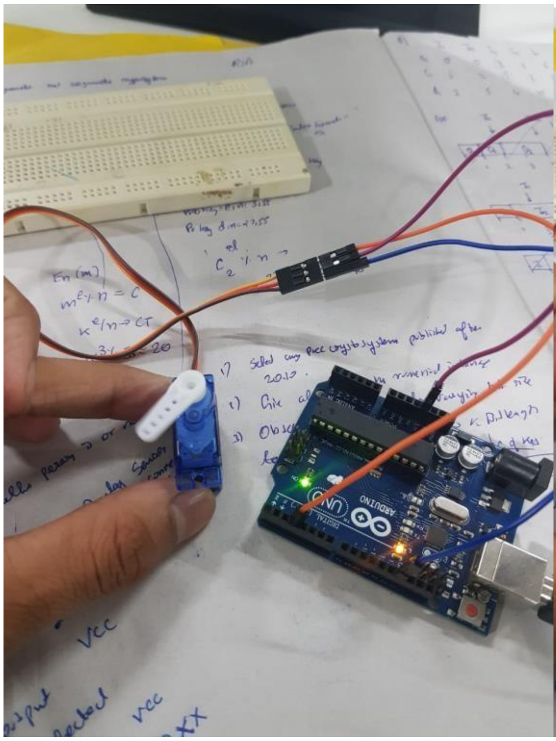


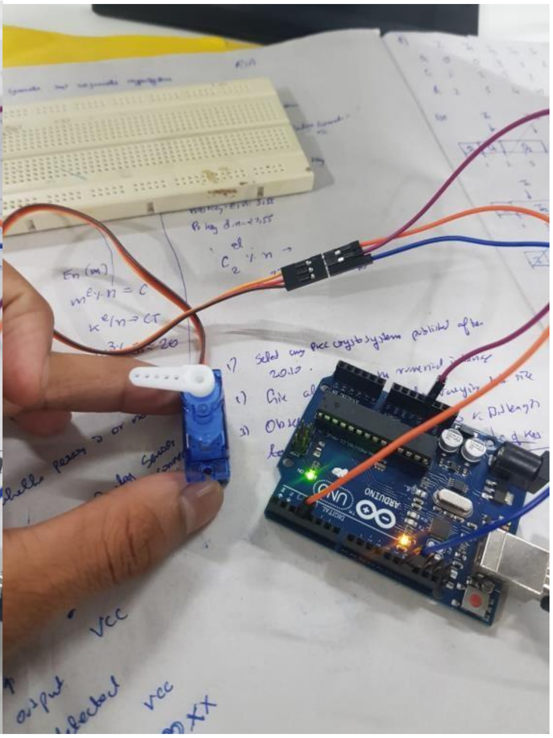
CODE:

```
// defining the pins

#include <Servo.h>

Servo servo1; // Create object for Servo motor 1

int position = 0;

const int trigPin = 9;

const int echoPin = 10;
```

```
// defining variables
long duration;
int distance;
void setup() {
pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
pinMode(echoPin, INPUT); // Sets the echoPin as an Input
Serial.begin(9600); // Starts the serial communication
servo1.attach(3);
}
void loop() {
// Clears the trigPin
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
// Sets the trigPin on HIGH state for 10 micro seconds
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*0.034/2;
// Prints the distance on the Serial Monitor
Serial.print("Distance: ");
Serial.println(distance);
if (distance<20)
{
for (position = 0; position <= 180; position++)
{
servo1.write(position); // Set position of Servo motor 1
delay(3);
}
// Rotating Servo motor 1 in clockwise from 180 degree to 0 degree

for (position = 180; position >= 0; position--)

{

servo1.write(position); // Set position of Servo motor 1

delay(15); // Short delay to control the speed

}

}

}
```

UPLOADING AND TESTING :





THANK YOU !