IOT LAB MANUAL

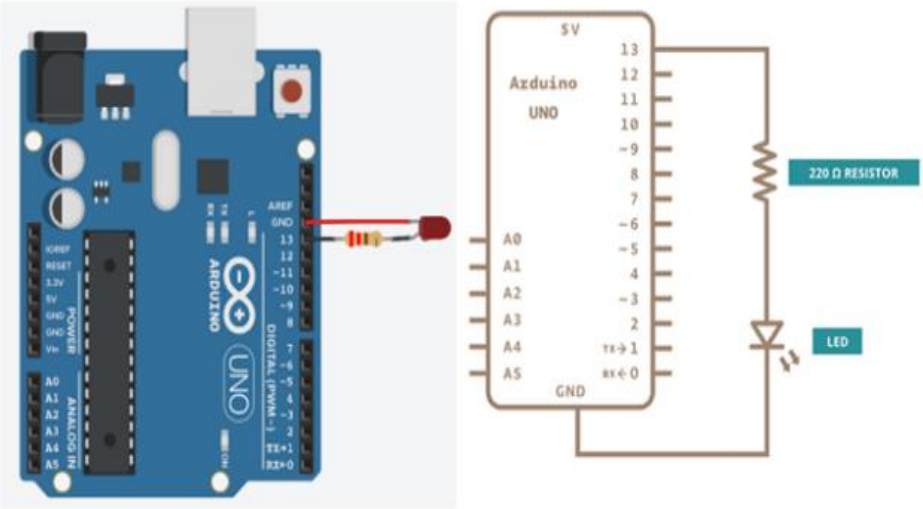
1. Write an Arduino program to accomplish the following tasks.
a. Blink a led connected to any digital pin of the Arduino board.

Aim: To control the state of an LED using a microcontroller (Arduino).

Application:

Traffic Light Simulation: The Blink concept can be expanded to simulate a traffic light system. By connecting LEDs to different digital pins, one can create a program that mimics the changing states of a traffic light.

Circuit Diagram:



Components Required:

1. Arduino UNO
2. USB Cable
3. 220Ω Resistor
4. LED
5. Jumper Wires

Procedure:

- Connect the positive/longer terminal (Anode) of the LED to the Resistor and Resistor to the digital pin of the UNO board, i.e., PIN **13**.
- Connect the negative/shorter terminal (Cathode) of the LED to the GND pin of the UNO board using the jumper wire.
- Connect the USB cable.
- Select the board and serial port in the Arduino IDE.
- Upload the sketch or code on the board.
- The LED will dim and light for the specified duration.

Code:

```
void setup()
{
pinMode(LED_BUILTIN, OUTPUT);
}
 void loop()
{
digitalWrite(LED_BUILTIN, HIGH);
delay(1000);
digitalWrite(LED_BUILTIN, LOW);
delay(1000);
 }
```

Result:

- LED_BUILTIN turns ON for 1 second.
- LED_BUILTIN turns OFF for 1 second.
- The cycle repeats continuously.

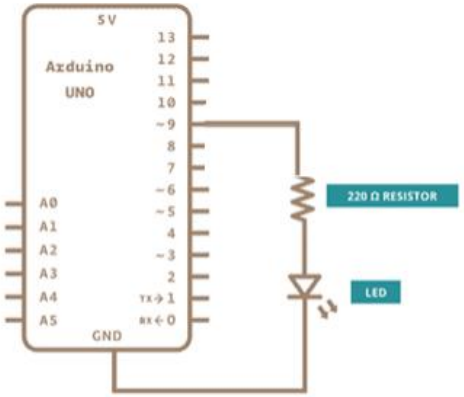
b. Fade an LED connected to any digital pin of the Arduino board. Include the necessary steps to set up the circuit.

Aim: To control the brightness of an LED using a microcontroller (Arduino) through the Fade operation.

Application:

Dimmable Lighting System: The Fade concept can be extended to simulate a dimmable lighting system. By connecting LEDs to different digital pins and applying Fade logic, one can create a program that mimics the smooth transition of light intensity, providing a customizable ambient lighting experience.

Circuit Diagram:



Components Required:

1. Arduino Uno
2. USB cable
3. 220Ω Resistor
4. LED
5. Jumper Wires

Procedure:

- Connect the positive/longer terminal (Anode) of the LED to the Resistor and Resistor to the digital pin of the UNO board, i.e., PIN **9. (PWM pin 9)**

- Connect the negative/shorter terminal (Cathode) of the LED to the GND pin of the UNO board using the jumper wire.
- Connect the USB cable.
- Select the board and serial port in the Arduino IDE.
- Upload the sketch or code on the board.
- The LED will dim and light for the specified duration.

Code:

```
int led = 9;
int brightness = 0;
int fadeAmount = 5;
 void setup()
 {
pinMode(led, OUTPUT);
 }
 void loop()
 {
 analogWrite(led, brightness);
 brightness = brightness + fadeAmount;
 if (brightness <= 0 || brightness >= 255)
{
fadeAmount = -fadeAmount;
}
delay(10);
 }
```

Result:

• The LED connected to pin 9 fades in and out smoothly, creating a visually pleasing transition of brightness.
• The fadeAmount variable controls the rate of brightness change, and the delay(10) introduces a short delay between each intensity adjustment, contributing to the smooth fading effect.
• The LED's brightness transitions from 0 to 255 and vice versa in a continuous loop.

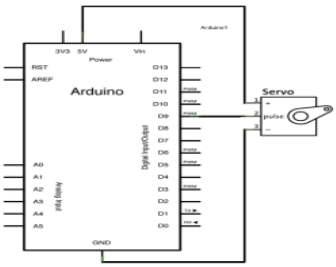
2. Develop an Arduino program capable of controlling the position of the servo motor.

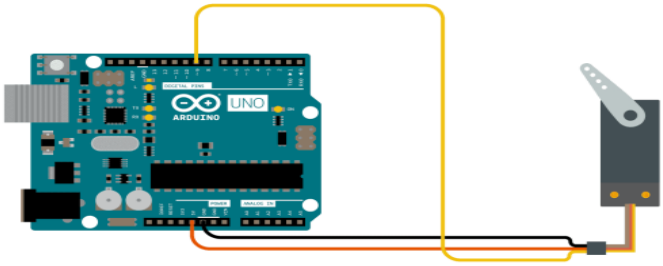
Aim: To control the position of a servo motor using a microcontroller (Arduino) through the Servo Motor operation.

Application:

Door/Gate Control System: The Servo Motor concept can be extended to control doors or gates in home automation systems. By connecting the servo to a door or gate mechanism, the Arduino can precisely control the position of the servo motor to open or close the door.

Circuit Diagram:





Components Required:

1. Arduino Uno Board
2. USB cable
3. Servo Motor
4. Jumper Wires

Procedure:

- The servo motor has a female connector with three pins. The darkest or even black(brown) one is usually the ground. Connect this to the Arduino GND.
- Connect the power cable that in all standards should be red to 5V on the Arduino.
- Connect the remaining line on the servo connector(Orange) to a digital pin on the Arduino. (PIN 9) using jumper wires.
- Connect the USB cable.
- Select the board and serial port in the Arduino IDE.
- Upload the sketch or code on the board.

Code:

```
#include <Servo.h>
Servo myservo;
int pos = 0;
void setup() {
myservo.attach(9);
}
void loop()
{
for (pos = 0; pos<= 180; pos += 1)
{
myservo.write(pos);
delay(15);
 }
 for (pos = 180; pos>= 0; pos -= 1)
{
myservo.write(pos);
delay(15);
}
}
```

Result:

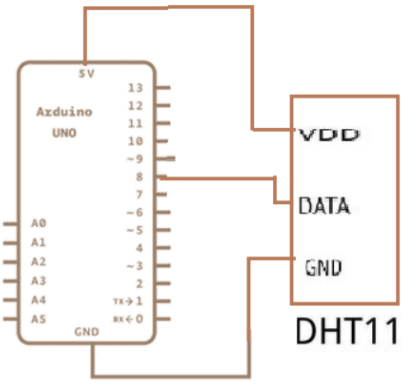
- The servo motor connected to pin 9 smoothly sweeps from 0 to 180 degrees in one direction.
- After reaching 180 degrees, the servo motor smoothly sweeps back from 180 to 0 degrees.
- This sweeping motion repeats continuously, creating a visually perceivable back-and-forth movement.

3. Set up the DHT 11 Humidity sensor on an Arduino.

Aim: To utilize a microcontroller (Arduino) to interface with a DHT (Digital Humidity and Temperature) sensor, and obtain accurate readings of humidity, temperature, and heat index.

Application:
Climate Monitoring System: DHT sensors are commonly used in climate monitoring systems. The experiment can be extended to create a device that continuously monitors and reports changes in temperature, humidity, and heat index, providing valuable information for environmental control.

Circuit Diagram:



Components Required:

1. Arduino Uno Board
2. USB cable
3. DHT11 Sensor
4. Jumper Wires

Procedure:

- DHT11 pin-1(VDD) is connected to the 5V power supply pin on the Arduino Uno Board.
- DHT11 pin -2(DATA) is connected to the digital pin 8 on the Arduino Uno.
- DHT11 pin-3(GND) is connected to the Ground (GND) pin on the Arduino Uno board using jumper wires.
- Connect the USB cable.
- Select the board and serial port in the Arduino IDE.
- Upload the sketch or code on the board

Code:

```
#include <dht.h> // Include library
#define outPin 8 // Defines pin number to which the sensor is connected
dht DHT; // Creates a DHT object
void setup()
{
Serial.begin(9600);
}
void loop()
{
int readData = DHT.read11(outPin);
float t = DHT.temperature; // Read temperature
float h = DHT.humidity; // Read humidity
Serial.print("Temperature = ");
Serial.print(t);
Serial.print("°C | ");
Serial.print((t*9.0)/5.0+32.0); // Convert celsius to fahrenheit
Serial.println("°F ");
Serial.print("Humidity = ");
```

```
Serial.print(h);
Serial.println("% ");
Serial.println("");
delay(2000); // wait two seconds
}
```

Result:

• Serial monitor displays humidity, temperature in Celsius and Fahrenheit, and the calculated heat index every 2 seconds.
• The readings provide real-time information about the environmental conditions sensed by the DHT sensor.

4. Design an Arduino program to detect obstacles using IR (Infrared) sensor. The task involves creating a system where the Arduino microcontroller interprets data from the IR sensor to identify the presence or absence of an obstacle.

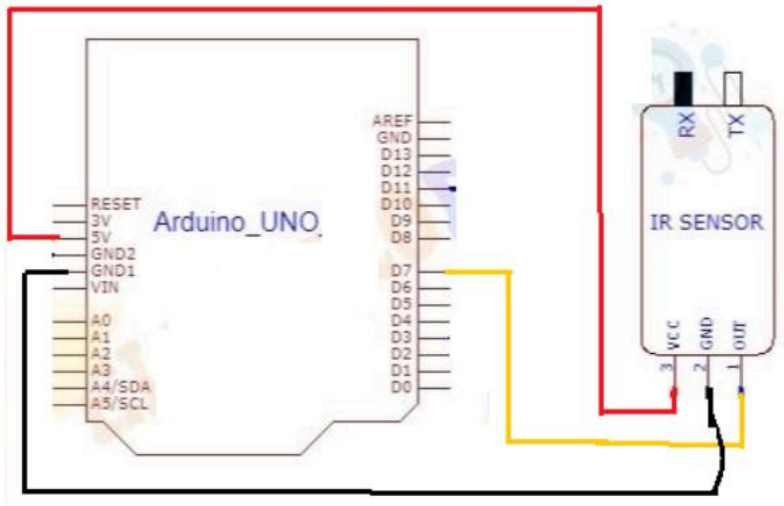
Aim: To interface an Infrared (IR) sensor with a microcontroller (Arduino) and detect the obstacle with it.

Application:

Distance Measurement: The IR sensor's analog output can be utilized for distance measurement. As an object approaches or moves away from the sensor, the analog voltage changes, allowing for proximity or distance detection.

Obstacle Avoidance: In robotics, IR sensors are often employed for obstacle avoidance. The analog readings can be used to identify the presence and proximity of obstacles, enabling the robot to navigate around them.

Circuit Diagram:



Components Required:

1. Arduino Uno Board
2. USB cable
3. IR Sensor
4. Jumper Wires

Procedure:

- Connect the VCC pin of IR Sensor to the 5V power supply pin on the Arduino Uno Board.
- Connect the OUT pin to the digital pin 7 on the Arduino Uno.
- Connect the ground pin (GND) to the Ground (GND) pin on the Arduino Uno board using jumper wires.
- Connect the USB cable.
- Select the board and serial port in the Arduino IDE.
- Upload the sketch or code on the board.

Code:

```
int IRSensor=7;
void setup()
{
pinMode(7,INPUT);
Serial.begin(9600);
}
void loop()
{
int sensordata=digitalRead(IRSensor);
if(sensordata==0)
{
Serial.println("Stop..Something is ahead");
Serial.println(sensordata);
}
else
{
Serial.println("Path is Clear");
Serial.println(sensordata);
delay(2000);
}
}
```

Result:

Whenever an obstacle is present in front of the IR sensor it will detect and display that"Stop something is ahed" on the serial monitor otherwise it will display that " Path is clear".

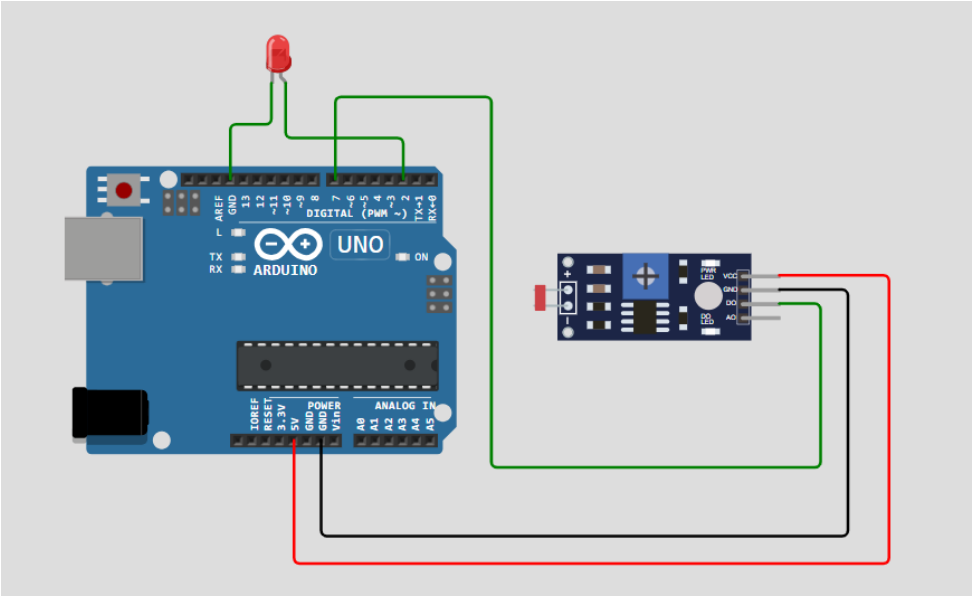
5. Design an Arduino program to control an LED based on ambient light conditions using an LDR sensor and Arduino microcontroller.

Aim: To interface a Light-Dependent Resistor (LDR) sensor with a microcontroller (Arduino), with the aim of measuring light intensity and categorizing it into different levels.

Application:

Ambient Light Control: This experiment serves as the foundation for creating systems that adjust ambient lighting based on the surrounding light intensity. For example, it can be applied to automatically control indoor lighting based on natural light conditions.

Circuit Diagram:



Components Required:

1. Arduino Uno Board
2. USB cable
3. LDR Sensor
4. LED
5. 220Ω Resistor
6. Jumper Wires

Procedure:

- Connect the VCC pin of LDR Sensor to the 5V power supply pin on the Arduino Uno Board.
- Connect the OUT pin to the digital pin 7 on the Arduino Uno.
- Connect the ground pin (GND) of the LDR sensor to the Ground (GND) pin on the Arduino Uno board using jumper wires.
- Connect the positive/longer terminal (Anode) of the LED to the Resistor and Resistor to the digital output pin of the UNO board, i.e., PIN **2**.
- Connect the negative/shorter terminal (Cathode) of the LED to the GND pin of the UNO board using the jumper wire.
- Connect the USB cable.
- Select the board and serial port in the Arduino IDE.
- Upload the sketch or code on the board.

Code:

```
void setup()
{
pinMode(2,OUTPUT); //LED
pinMode(7, INPUT); //LDR
Serial.begin(9600);
}
void loop()
{
int r =digitalRead(7);
if(r==HIGH)
{
Serial.println("It's dark.LED is turned on");
digitalWrite(2, HIGH);
}
else
{
Serial.println("It's day.LED is turned off");
digitalWrite(2, Low);
}
```

```
delay(1000);
}
```

Result:

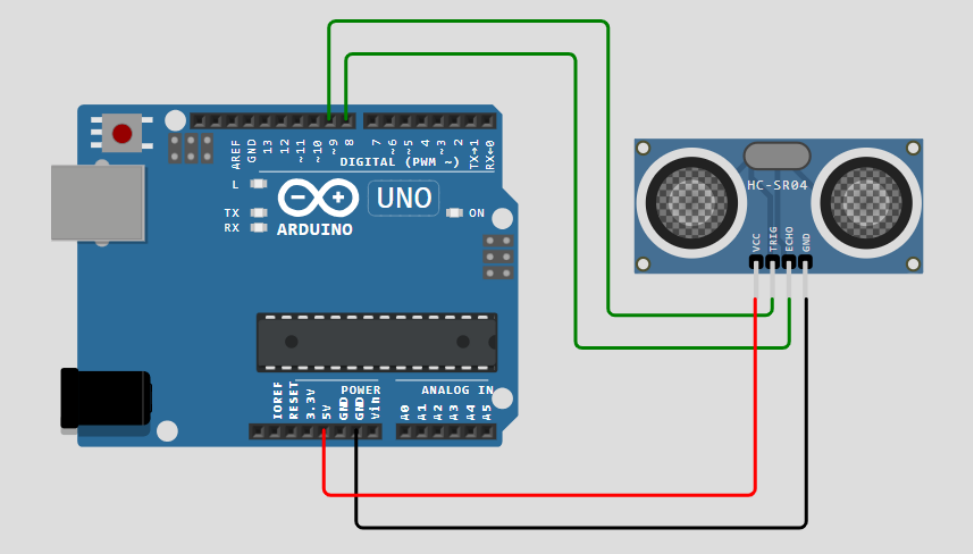
- Whenever a light is detected from the LDR sensor it will and display that “It's day. LED is turned off” on the serial monitor and the LED should be turned off.
- When the light is not detected from the LDR sensor it will display that “It's dark. LED is turned on” and the LED should be turned off.
- The system continuously monitors the LDR sensor and categorizes light intensity based on predefined thresholds.

6. Write a Arduino program to measure and display the distance of an object from the sensor using ultrasonic sensor.

Aim:
To interface an ultrasonic sensor with a microcontroller (Arduino), aiming to measure and display the distance of an object from the sensor.

Application:
Distance Measurement System: This experiment serves as the foundation for creating distance measurement systems. Ultrasonic sensors are commonly used in robotics, parking assistance systems, and object detection applications.

Circuit Diagram:



Components Required:

1. Arduino Uno Board
2. USB Cable
3. Ultrasonic Sensor
4. Jumper Wires

Procedure:

- Connect the VCC pin of Ultrasonic Sensor to the 5V power supply pin on the Arduino Uno Board.
- Connect the trig pin to the digital pin 9 on the Arduino Uno.
- Connect the echo pin to the digital pin 8 on the Arduino Uno.
- Connect the ground pin (GND) of the Ultrasonic sensor to the Ground (GND) pin on the Arduino Uno board using jumper wires.
- Connect the USB cable.
- Select the board and serial port in the Arduino IDE.
- Upload the sketch or code on the board.

Code:

```
#define echoPin 8
#define trigPin 9
long duration;
int distance;
void setup()
{
 pinMode(trigPin, OUTPUT);
 pinMode(echoPin, INPUT);
 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.print(distance);
Serial.println(" cm");
delay(500);
}
```

Result:

• Serial monitor displays "Distance: [Distance in cm]" every 500 milliseconds.
• The ultrasonic sensor continuously measures the distance, updating the value based on the object's proximity.

7. Write a program to interface a Smoke Sensor (MQ2) with a microcontroller (Arduino), with the aim of monitoring and responding to smoke levels.

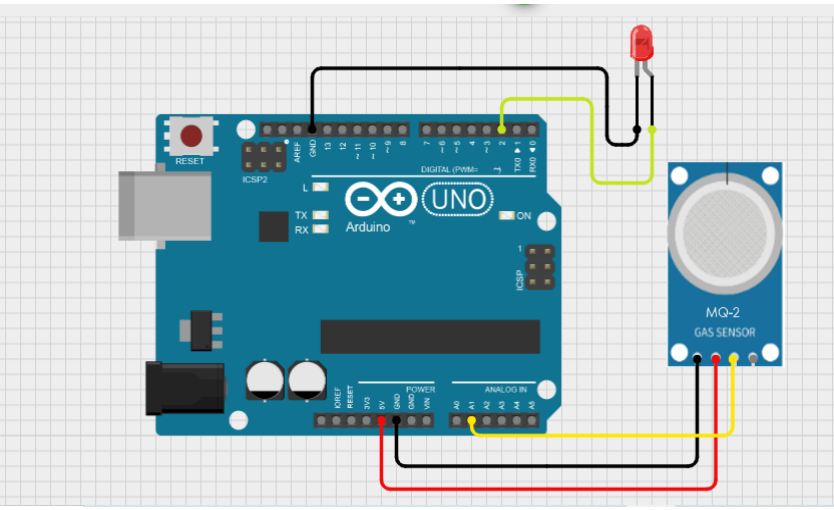
Aim:

To interface a Smoke Sensor (MQ2) with a microcontroller (Arduino), with the aim of monitoring and responding to smoke levels.

Application:

Smoke Detection System: This experiment serves as the basis for creating a smoke detection system. The Smoke Sensor (MQ2) is commonly used in applications where the detection of smoke or harmful gases is crucial, such as in fire alarm systems.

Circuit Diagram:



Components Required

1. Arduino Uno Board
2. USB Cable
3. MQ2 Smoke Sensor
4. 220Ω Resistor
5. LED
6. Jumper Wires

Procedure:

- Connect the VCC pin of MQ2 Sensor to the 5V power supply pin on the Arduino Uno Board.
- Connect the A0 pin to the analog pin A1 on the Arduino Uno.
- Connect the ground pin (GND) of the MQ2 sensor to the Ground (GND) pin on the Arduino Uno board using jumper wires.
- Connect the positive/longer terminal (Anode) of the LED to the Resistor and Resistor to the digital output pin of the UNO board, i.e., PIN **2**.
- Connect the negative/shorter terminal (Cathode) of the LED to the GND pin of the UNO board using the jumper wire.
- Connect the USB cable.
- Select the board and serial port in the Arduino IDE.
- Upload the sketch or code on the board.

Code:

```
int smoke=A1;
void setup()
{
 Serial.begin(9600);
 pinMode(A1,INPUT);
 pinMode(2,OUTPUT);
}
void loop()
{
 int ss=analogRead(A1);
 Serial.println("smoke value is");
```

```
 Serial.println(ss);
 if(ss<50)
 digitalWrite(2,HIGH);
 else
 digitalWrite(2,LOW);
delay(1000);
}
```

Result:

• Serial monitor displays "smoke value is [Analog Reading]" continuously.
• The external LED turns on if the smoke concentration falls below the set threshold (condition: ss < 50), indicating a potential smoke presence.