INSTITUTE OF ENGINEERING & MANAGEMENT
SALT LAKE, KOLKATA

LAB MANUAL

Year : 2022 - 2026

Course Name : Cloud Computing & IoT Laboratory

Course Code : PCCCS692

Semester : VI

Branch : AIML

Cloud Computing & IoT Laboratory (PCCCS692)

Name : AMAN KUMAR BATSH

University Roll No. : 12022002016030 **Class Roll** 07

Year : 3rd year **Semester :** 6th Semester

Session : 2024-2025

INDEX

LAB : Cloud Computing & IoT Laboratory

YEAR : 3rd Year **SEMESTER:** 6th

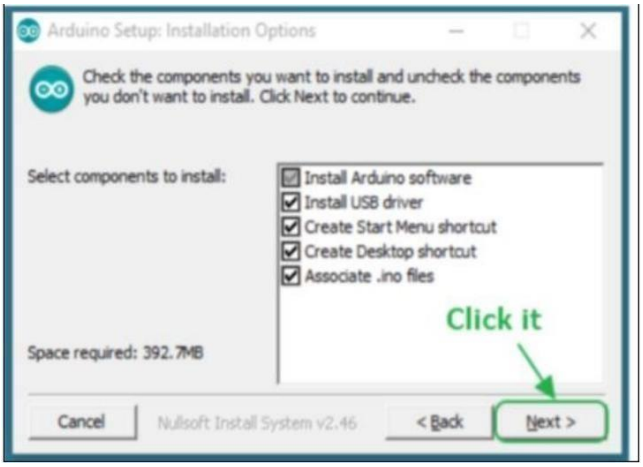
Exp. No.	Name of Experiment	Date of Experiment	Date of Submission	Page	Remarks

Experiment No: 1

Title: Study and Installation of Arduino IDE

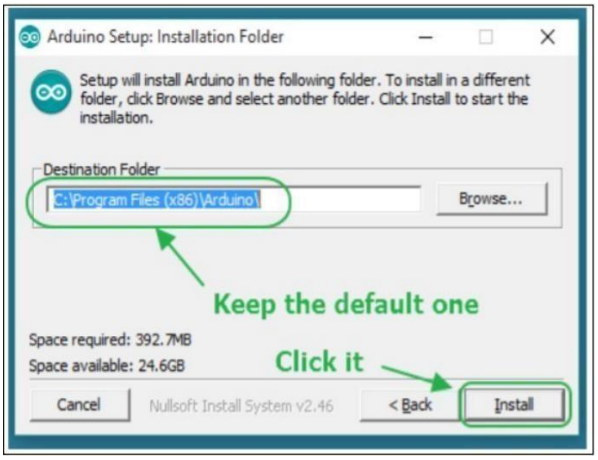
Step1: Downloading

To install the Arduino software, download this page: http://arduino.cc/en/Main/Software and proceed with the installation by allowing the driver installation process.



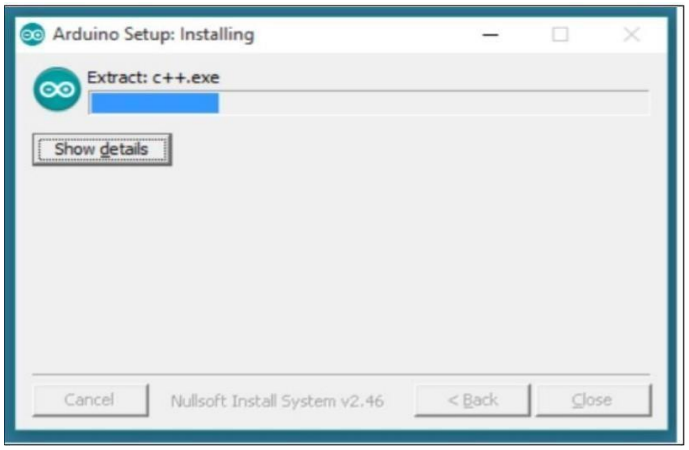
Step 2: Directory Installation

Choose the installation directory.



Step 3: Extraction of Files

The process will extract and install all the required files to execute properly the Arduino Software (IDE)

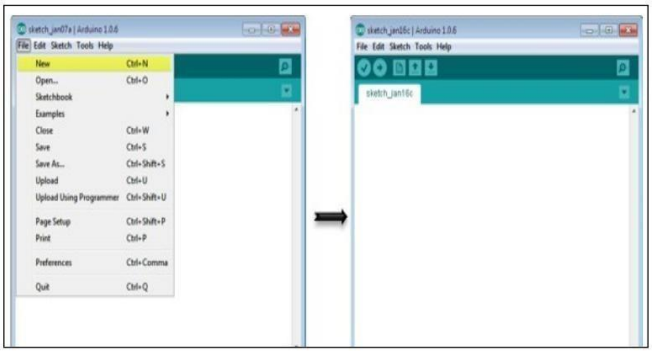


Step 4: Connecting the board

The USB connection with the PC is necessary to program the board and not just to power it up. The Uno and Mega automatically draw power from either the USB or an external power supply. Connect the board to the computer using the USB cable. The green power LED (labelled PWR) should go on.

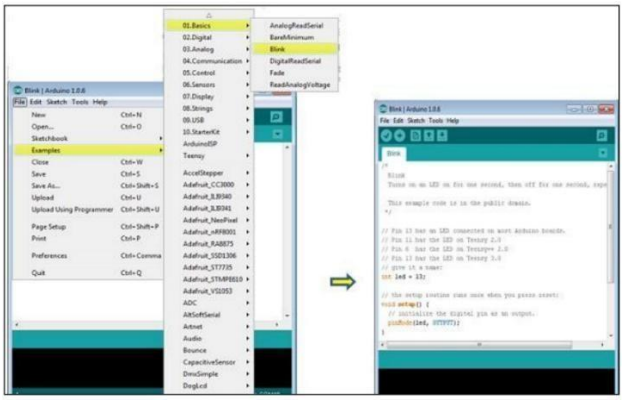
Step 5: Working on the new project

- Open the Arduino IDE software on your computer. Coding in the Arduino language will control your circuit.
- Open a new sketch File



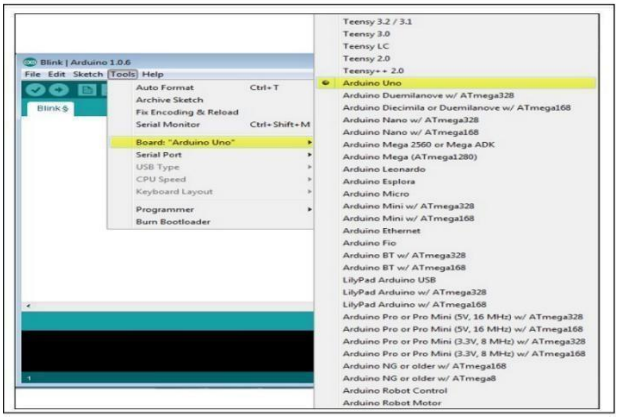
Step 6: Working on an existing project

To open an existing project example, select File →Example →Basics → Blink.



Step 7: Select your Arduino board.

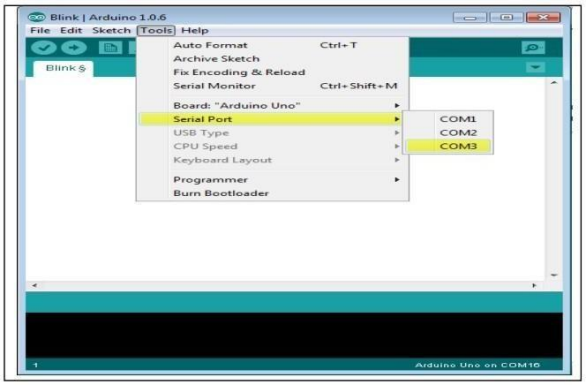
- To avoid any error while uploading your program to the board, you must select the correct Arduino board name, which matches with the board connected to your computer.
- Go to Tools → Board and select your board.



Step 8: Select your serial port

- Select the serial device of the Arduino board.
- Go to Tools → Serial Port menu. This is likely to be COM3 or higher (COM1 and COM2 are usually reserved for hardware serial ports).

- To find out, you can disconnect your Arduino board and re-open the menu, the entry that disappears should be of the Arduino board. Reconnect theboard and select that serial port.



- **Step 9: Upload the program to your board.** Click the "Upload" button in the environment.
- Wait a few seconds; you will see the RX and TX LEDs on the board, flashing.

- If the upload is successful, the message "Done uploading" will appear in the status bar.

Experiment No: 2

Title: Study and Installation of Tinkercad

Tinkercad is a web-based computer-aided design (CAD) software that allows users to create 3D models easily. Here are the steps to study and install Tinkercad:

Studying Tinkercad:

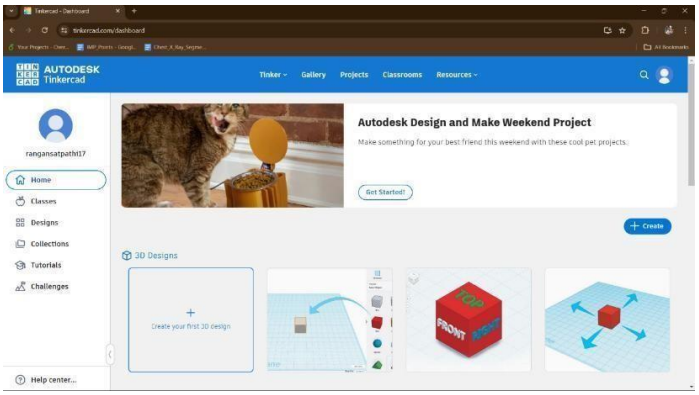
- **Explore Tinkercad's Official Website**: Visit Tinkercad's official website to get an overview of its features, capabilities, and applications.
- **Watch Tutorials:** Look for tutorials on platforms like YouTube or educational websites. Tinkercad provides its own set of tutorials for beginners.
- **Practice:** Start with simple projects to get acquainted with the interface and tools. Tinkercad offers a variety of sample projects to get started.
- **Join Communities:** Participate in forums, online communities, or social media groups related to Tinkercad. This is a great way to learn from experienced users, ask questions, and share your creations.
- **Read Documentation:** Familiarize yourself with Tinkercad's documentation, which includes guides, FAQs, and troubleshooting tips.

Installing Tinkercad:

Tinkercad is a web-based application, so there's no need to download or install any software. However, you'll need a modern web browser with WebGL support.

System Requirements: Ensure your computer meets the system requirements to run Tinkercad smoothly.

- **Login:** After creating an account, log in to Tinkercad using your credentials.
- **Start Designing**: Once logged in, you can start creating your 3D designs right away. Tinkercad provides a user-friendly interface with drag-and-drop functionality for easy designing.
- **Save Your Designs:** Remember to save your designs regularly to your Tinkercad account to avoid losing any work.
- **Export or Share**: After completing your design, you can export it in various formats or share it directly from Tinkercad.



Experiment No: 3

Title: Led control using push button in Arduino Board

Controlling an LED using a push button on an Arduino board is a common beginner project. Below are the steps to achieve this:

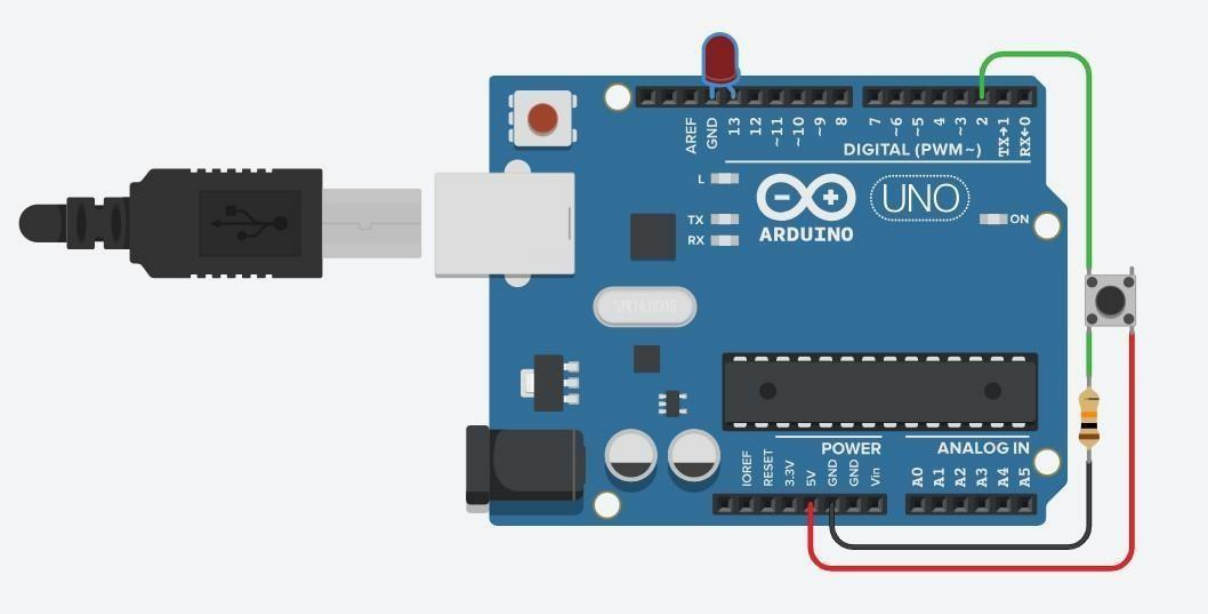
Hardware Required:

- Arduino Uno
- LED
- Push button
- Resistor (220 ohms recommended for the LED)
- Breadboard
- Jumper wires

Circuit Connections:

- Connect the longer leg (anode) of the LED to a digital pin on the Arduino (e.g., pin 13).
- Connect the shorter leg (cathode) of the LED to a current-limiting resistor (e.g., 220 ohms).
- Connect the other end of the resistor to the ground (GND) pin on the Arduino.
- Connect one leg of the push button to any digital pin on the Arduino (e.g., pin 2).
- Connect the other leg of the push button to the ground (GND) pin on the Arduino.
- Add a pull-up resistor (e.g., 10k ohms) between the digital pin connected to the push button and the 5V pin on the Arduino.

```
const int ledPin = 13; const int buttonPin =
2; int ledState = LOW; int lastButtonState =
LOW; int buttonState; unsigned long
lastDebounceTime = 0; unsigned long
debounceDelay = 50; void setup() {
pinMode(ledPin, OUTPUT); pinMode(buttonPin,
INPUT); digitalWrite(buttonPin, HIGH);
} void loop() { int reading =
digitalRead(buttonPin); if (reading !=
lastButtonState) { lastDebounceTime =
millis();
} if ((millis() - lastDebounceTime) > debounceDelay) { if
(reading != buttonState) {              buttonState =
reading; if (buttonState == LOW) {
           ledState = !ledState;
digitalWrite(ledPin, ledState);
            }
     } } lastButtonState =
reading;
}
```



Explanation: The code sets up pin modes for the LED pin and push button

pin.

It activates the internal pull-up resistor for the push button pin to ensure a stable reading when the button is not pressed.

The code implements debounce logic to ensure that a single button press is registered despite any electrical noise or bouncing that might occur when the button is pressed or released.

When the button is pressed, the LED state toggles. **Upload:**

- Connect the Arduino board to your computer using a USB cable.
- Open the Arduino IDE.
- Copy and paste the provided code into a new sketch.
- Click on the "Upload" button to compile and upload the code to the Arduino board.

Testing:

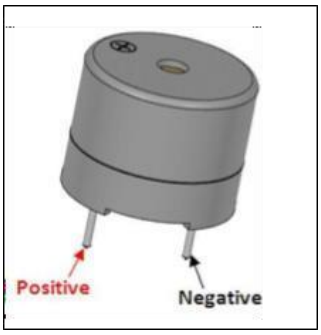
Once uploaded, you should see the LED turning on and off each time you press the push button.

Experiment No: 4

Title : Interfacing of the Active Buzzer with Arduino.

Introduction:

A piezo buzzer is a type of electronic device that's used to produce beeps and tones. The working principle of the device is piezoelectric effect. The main component of this device is a piezo crystal, which is a special material that changes shape when a voltage applied to it. The active buzzer will only generate sound when it will be electrified. It generates sound at only one frequency. This buzzer operates at an audible frequency of about 2 KHz.



Specifications:

Specification	Range
VoltageRange	3.3-5V
Frequency	2KHz

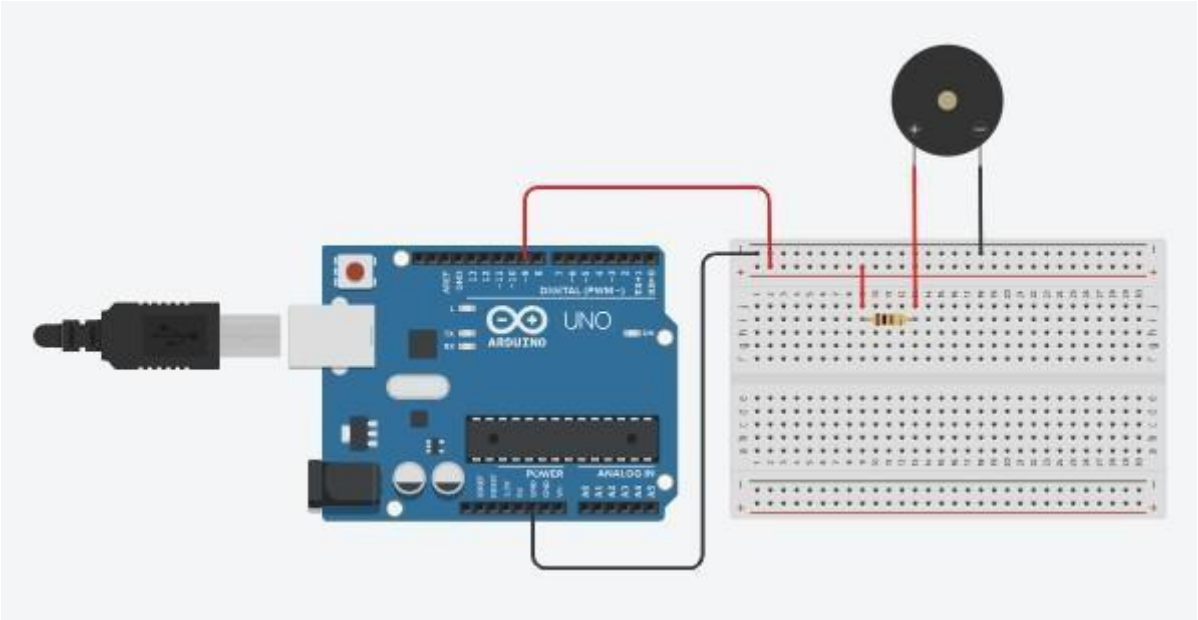
Pin Name	Description
Positive	Identified by (+) symbol or longer terminal lead. Can be powered by 6V DC
Negative	Identified by short terminal lead. Typically connected to the ground of the circuit

Hardware Required:

Component Name	Quantity
Arduino UNO	1
Buzzer / piezo speaker	1
220-ohm resistors	1

USB Cable	1
Breadboard	1
Jumper wires	several

Connection Diagram:



Steps of working:

- Connect the Supply wire (RED) of the buzzer to the Digital Pin 9 of the Arduino through a 100-ohm resistor.
- Connect the Ground wire (BLACK) of the buzzer to any Ground Pin on the Arduino.
- Upload the code
- Observe the changes in the pitch and volume of the buzzer.

Sketch:

This sketch works by setting pin D9 as for the control the buzzer. After that the run a loop that continually sends that value as voltage high or low to the D9 using the function digitalWrite(). The voltage value and the tone generated from the buzzer will vary accordingly.

```
// Define the pin for the buzzer
const int buzzerPin = 9;
// Define an array of frequencies (in Hz) to play
int frequencies[] = {500, 1000, 1500, 2000, 2500};
int numFrequencies = sizeof(frequencies) / sizeof(frequencies[0]);

// Define the duration to play each frequency (in milliseconds)
```

```
int duration = 500;

void setup() {
      // Set the buzzer pin as an output
     pinMode(buzzerPin, OUTPUT);
}

void loop() {
     // Loop through each frequency in the array
     for (int i = 0; i < numFrequencies; i++) {
            // Play the current frequency
            tone(buzzerPin, frequencies[i]);
            // Wait for the specified duration
            delay(duration);
       }
     // Turn off the buzzer
     noTone(buzzerPin);
     // Wait before repeating the loop
     delay(1000);
}
```

Observation:

Sr. no.	Change the value	Frequency of tone
1	500	500
2	1000	1000
3	1500	1500

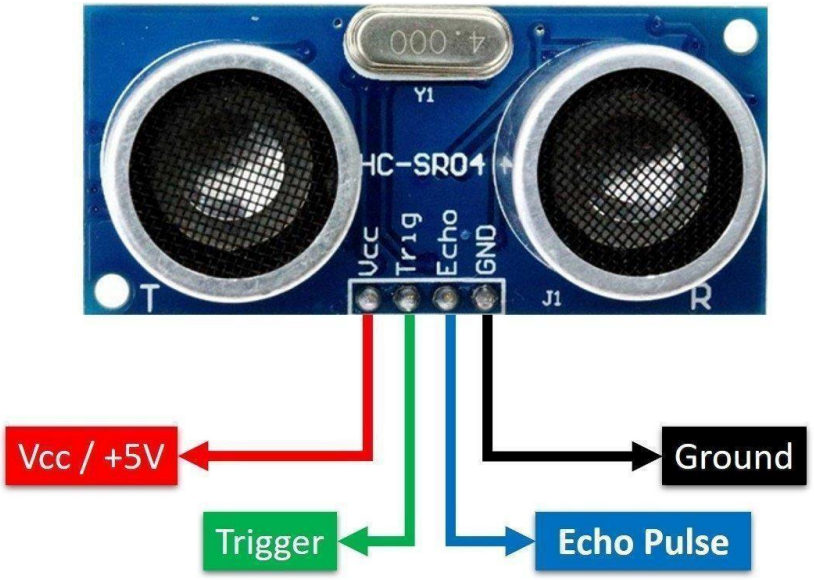
Experiment No: 5

Title : Building Intrusion Detection System with Arduino and Ultrasonic Sensor

Introduction:

An intrusion detection system (IDS) is a device or software application that monitors a network or systems for malicious activity

Ultrasonic Sensors: The HC-SR04 ultrasonic sensor uses SONAR to determine the distance of an object just like the bats do. It offers excellent non-contact range detection with high accuracy and stable readings in an easy- to-use package from 2 cm to 400 cm or 1" to 13 feet. It comes complete with ultrasonic transmitter and receiver module. The ultrasonic sensor uses the reflection of sound in obtaining the time between the wave sent and the wave received. It usually sent a wave at the transmission terminal and receives the reflected waves. The time taken is used together with the normal speed of sound in air (340ms-1) to determine the distance between the sensor and the obstacle. The Ultrasonic sensor is used here for the intruder detection. The sound via a buzzer occurs when an object comes near to the sensor. The distance to which the sensor will respond can be easily adjusted in the program.

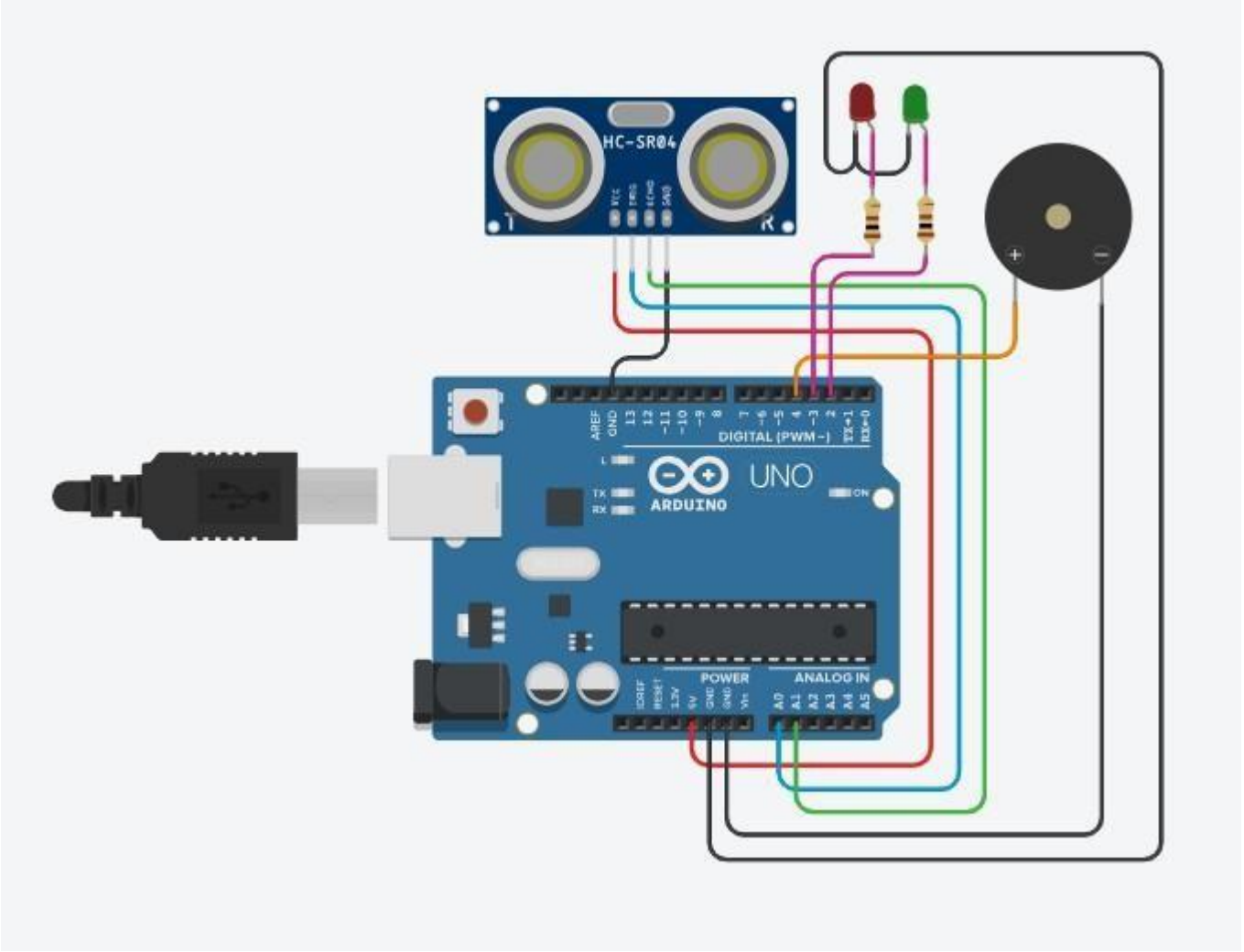


Hardware Required:

Component Name	Quantity
Arduino UNO	1
RED LED	1
Green LED	1
HC-SR04 Sensor	1
Buzzer	1

USB Cable	1
Breadboard	1
Jumper wires	several

Connection Diagram:



Steps of working

1. Insert the Ultrasonic sensor into your breadboard and connect its Echo pin to the digital pin 2 and the Trigger pin to digital pin 3 of the Arduino.
2. Insert the RED and Green LED into the breadboard. Attach the positive leg (the longer leg) of red LED to signal pin of the Buzzer via the 220- ohm resistor, and the negative leg to GND. The green LED is connected to digital pin 8 of the Arduino.
3. Upload the code.
4. Observe the LEDs and take some object in front of ultrasonic sensor.

5. Observe the changes in the LED and buzzer sound.

The Sketch

This sketch works by setting pin 2 as for the ultrasonic sensors and pin 8, pin9 & pin 10 as an OUTPUT to power the LEDs and buzzer. After that the run a loop that continually reads the value from the echo pin and sends that value as voltage to the LEDs. The color of the LED which glows will vary accordingly to the detection of object in the given range.

```
// Define the pins for ultrasonic sensor
const int trigPin = A0;
const int echoPin = A1;

// Define the pins for LEDs
const int redLED = 3;
const int greenLED = 2;

// Define the pin for the buzzer
const int buzzerPin = 4;

// Set a threshold distance (in centimeters) for triggering the buzzer
const int distanceThreshold = 50;         // Distance threshold in cm

void setup() {
     // Start the serial communication for debugging
     Serial.begin(9600);

     // Set the ultrasonic sensor pins as outputs and input
     pinMode(trigPin, OUTPUT);
     pinMode(echoPin, INPUT);

     //Set the pin for LEDs
     pinMode (greenLED, OUTPUT);
     pinMode (redLED, OUTPUT);

     // Set the buzzer pin as an output
     pinMode(buzzerPin, OUTPUT);
}
void loop() {
      // Variable to store the duration of the pulse
     long duration;

  // Trigger the ultrasonic sensor to send a pulse
     digitalWrite(trigPin, LOW);
     delayMicroseconds(2);
     digitalWrite(trigPin, HIGH);
     delayMicroseconds(10);
     digitalWrite(trigPin, LOW);
```

```
      // Read the pulse duration from the echo pin
     duration = pulseIn(echoPin, HIGH);

     // Calculate the distance in cm (Speed of sound: 343 m/s)
     long distance = (duration / 2) / 29.1;

     // Print the distance for debugging
     Serial.print("Distance: ");
     erial.print(distance);
      Serial.println(" cm");

 // Check if the distance is less than the threshold
if (distance < distanceThreshold) {
     // Activate the buzzer
     tone(buzzerPin, 2000);
     //Deactivate the greenLED
     digitalWrite(greenLED, LOW);
      // Activate the redLED
     digitalWrite(redLED, HIGH);
      }
else {
     // Deactivate the buzzer
     noTone(buzzerPin);
      //Deactivate the redLED
     digitalWrite(redLED, LOW);
     // Activate the greenLED
      digitalWrite(greenLED, HIGH);
     }

     // Add a small delay to stabilize readings
     delay(100);
}
```

Observation Table:

Sr no.	Object Detected	LED	Buzzer
1	TRUE	RED	HIGH
2	FALSE	GREEN	LOW

Experiment No: 6

Title : INTERFACING SERVO MOTOR WITH THEARDUINO

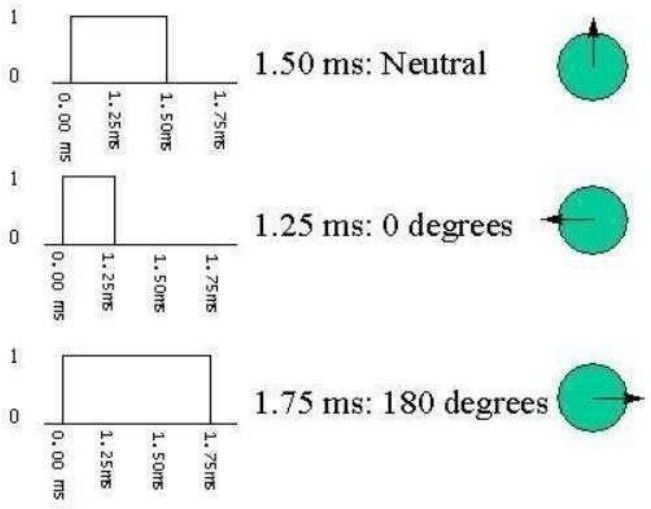
Introduction:

A Servo Motor is a small device that has an output sha . This sha can be posi oned to specific angular posi ons by sending the servo a coded signal. As long as the coded signal exists on the input line, the servo will maintain the angular posi on of the sha . If the coded signal changes, the angular posi on of the sha changes. Servo motors have three terminals – power, ground, and signal. The power wire is typically red, and should be connected to the 5V pin on the Arduino. The ground wire is typically black or brown as shown in figure

Specifications

GND	common ground for both the motor and logic
5V	posi ve voltage that powers the servo
Control	Input for the control system

The control wire is used to communicate the angle. The angle is determined by the dura on of a pulse that is applied to the control wire. This is called Pulse Coded Modula on. The servo expects to see a pulse every 20 milliseconds (.02 seconds). The length of the pulse will determine how far the motor turns. A 1.5 millisecond pulse, for example, will make the motor turn to the 90-degree position (often called as the neutral position). If the pulse is shorter than 1.5 milliseconds, then the motor will turn the sha closer to 0 degrees. If the pulse is longer than 1.5 milliseconds, the shaft turns closer to 180 degrees.



Hardware Required

- Arduino UNO,
- Breadboard
- Servo Motor
- USB Cable
- Jumper wires

Steps of working

- The servo motor has a female connector with three pins. The darkest or even black 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 to a digital pin on the Arduino.
- Upload the code
- Observe the position of the shaft.

Algorithm

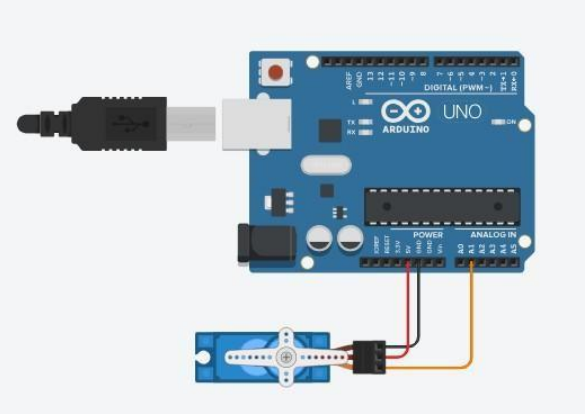
- This sketch works by setting pin D9 as for the control of servo motor
- After that run a loop that continually increment the value of the index of rota on angle and sends that value as voltage to the D9
- The voltage value is between 0–5 volts, and the rotation angle of the servo motor will vary accordingly.

Observations

Sr no.	Voltage	Position of Shaft
1		
2		
3		
4		
5		

Code & Circuit Diagram

```
#include <Servo.h>

Servo servoBase;

void setup() {
  servoBase.attach(A1);
  servoBase.write(0);
}

void loop() {
 for(int i=0; i<=180; i=i+10)
 {
  servoBase.write(i);
  delay(2000);
 }
}
```

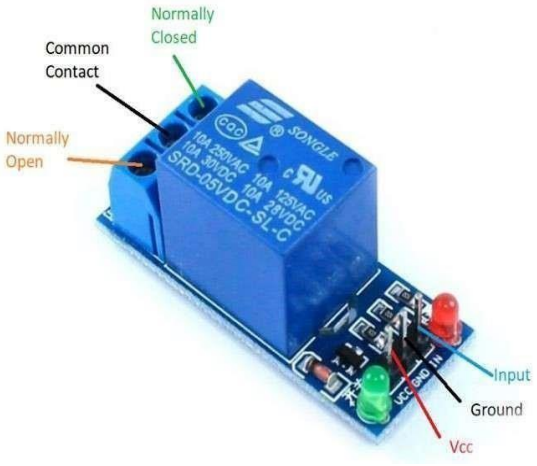


Experiment No: 7

Title : INTERFACING OF RELAYWITH THEARDUINO

Introduction:

Relay is an electromagnetic switch, which is controlled by small current, and used to switch ON and OFF relatively much larger current. Means by applying small current we can switch ON the relay which allows much larger current to flow



Hardware Required

- Arduino UNO
- Breadboard
- 5V Relay
- USB Cable
- Jumper wires

Steps of working

- The relay module connected with three pins. We will connect the relay module with Arduino in the normally open state. The black one of relay is usually the ground. Connect this to the Arduino GND.
- Connect the red wire of relay module to 5V of the Arduino.
- Connect the signal pin of relay module to a digital pin 6 of the Arduino.
- Upload the code
- Observe the clicking sound of the relay that states the ON and OFF constantly.

Algorithm

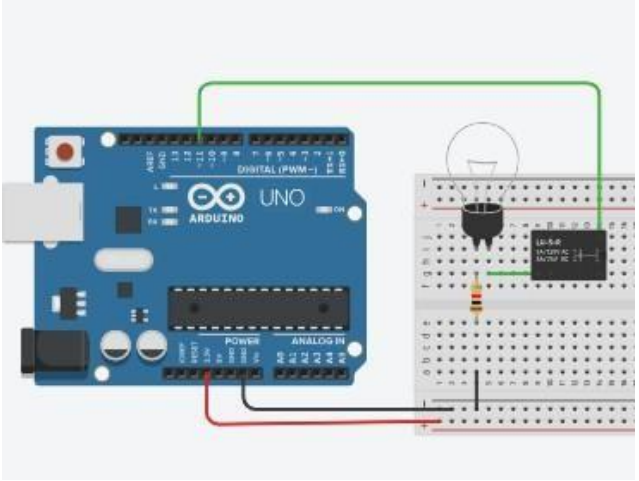
- This sketch works by setting 5V supply pin of Arduino as for the control of relay module
- After that run a loop that continually sends that value as voltage to the D6 with the delay given.

Observations

Sr no.	Delay (ms)	Relay Status
1	1000	
2	1000	

Code & Circuit Diagram

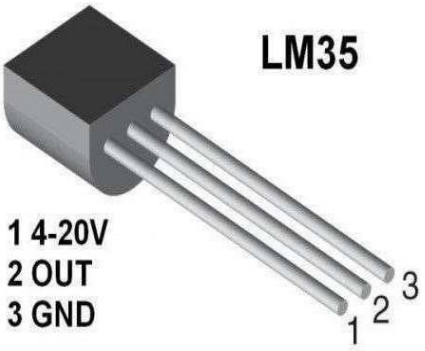
```
void setup()
{
  pinMode(11, OUTPUT);
}
void loop()
{
  digitalWrite(11, 1);
  delay(1000); // Wait for 1000 millisecond(s)
  digitalWrite(11, 0);
  delay(1000); // Wait for 1000 millisecond(s)
}
```



Experiment No: 8

Title : INTERFACING OF TEMPERATURE SENSOR WITH THE ARDUINO

Introduction:



The LM35 series are precision integrated-circuit temperature devices with an output voltage linearly proportional to the Cen grade temperature. LM35 is three terminal linear temperature sensors from National semiconductors. It can measure temperature from -55 degree Celsius to +150 degree Celsius. The voltage output of the LM35 increases 10mV per degree Celsius rise in temperature. LM35 can be operated from a 5V supply and the stand by current is less than 60uA.

Hardware Required

- Arduino UNO,
- Breadboard,
- LM35,
- USB Cable
- Jumper wires

Steps of working

- Insert the temperature sensor into your breadboard and connect its pin1 to the supply.
- Connect its center pin to the analog pin A0 and the remaining pin3 to GND on the breadboard.
- Upload the code as given below.
- Vary the temperature and read the voltage changes.
- Open the Arduino IDE's serial monitor to see the results.

Algorithm

- This sketch works by sending pin A0 as for the temperature sensor
- After that run a loop that continually reads the value from the sensor and sends that value as voltage
- The voltage value is between 0–5 V, when temperature varies accordingly

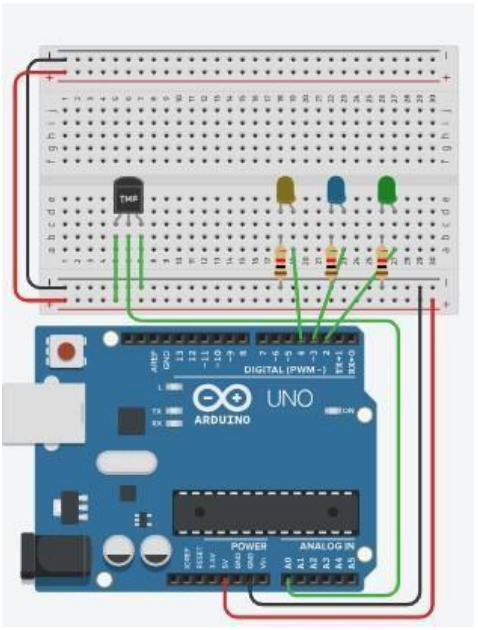
Observations

Sr no.	Voltage	Temperature
1		
2		
3		
4		
5		

Code & Circuit Diagram

```
int baselineTemp = 0;
int celsius = 0;
int fahrenheit = 0;
void setup()
{
  pinMode(A0, INPUT);
  Serial.begin(9600);

  pinMode(2, OUTPUT);
```

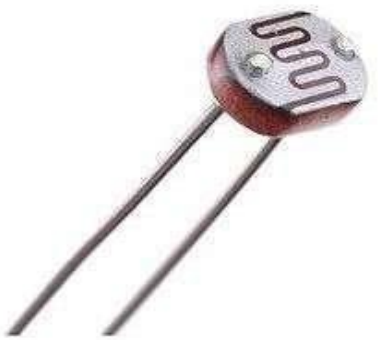
```
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
}
void loop()
{
  baselineTemp = 40;

  celsius = map(((analogRead(A0) - 20) * 3.04), 0, 1023, -40, 125);

  fahrenheit = ((celsius * 9) / 5 + 32);
  Serial.print(celsius);
  Serial.print(" C, ");
  Serial.print(fahrenheit);
  Serial.println(" F");

   if (celsius < baselineTemp) {
    digitalWrite(2, LOW);
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
  }
  if (celsius >= baselineTemp && celsius < baselineTemp + 10) {
    digitalWrite(2, HIGH);
    digitalWrite(3, LOW);
    digitalWrite(4, LOW);
  }
  if (celsius >= baselineTemp + 10 && celsius < baselineTemp + 20) {
    digitalWrite(2, HIGH);
    digitalWrite(3, HIGH);
    digitalWrite(4, LOW);
  }
  if (celsius >= baselineTemp + 20 && celsius < baselineTemp + 30) {
    digitalWrite(2, HIGH);
    digitalWrite(3, HIGH);
    digitalWrite(4, HIGH);
  }
  if (celsius >= baselineTemp + 30) {
    digitalWrite(2, HIGH);
    digitalWrite(3, HIGH);
    digitalWrite(4, HIGH);
  }
delay(1000);
```



Experiment No: 9

Title : DETECTION OF THE LIGHT USING PHOTO RESISTOR

Introduction:



A photo resistor or photocell is a light-controlled variable resistor made of a high resistance semiconductor. The resistance of a photo resistor decreases with increasing incident light intensity. A photo resistor can be applied in light-sensitive detector circuits, and light- and dark-activated switching circuits. It's also called light-dependent resistor (LDR).

Hardware Required

Arduino UNO, Breadboard, 220Ω resistor, Photo Resistor,
USB Cable, Jumper wires, 10KΩ resistor, LED

Steps of working

- Insert the photo resistor into your breadboard and connect its pin to the analog pin A0 and the remaining pin to supply on the breadboard.
- Insert the LED into the breadboard. A ach the positive leg (the longer leg) to pin 9 of the Arduino via the 220-ohm resistor, and the negative leg to GND.
- Insert the 10K-ohm resistor
- Upload the code
- Turn the photo resistor to ON the LED
- Observe the changes in the state of the LED.

Algorithm

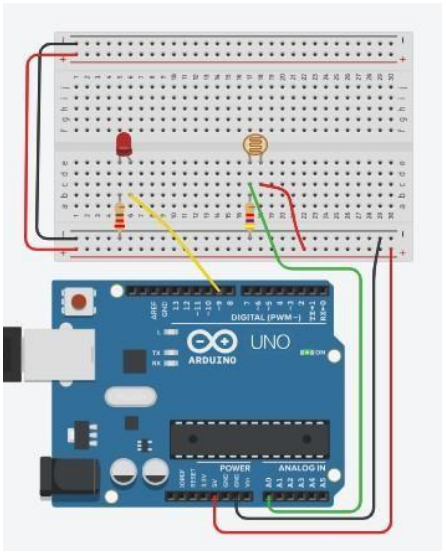
- This sketch works by setting pin A0 as for the photo sensor and pin 9 as an OUTPUT to power the LED.
- After that run a loop that continually reads value from the photo resistor and sends that value as voltage to the LED. The LED will vary accordingly.

Observations

Sr no.	Light detected	LED state
1	TRUE	TRUE
2	FALSE	FALSE

Code & Circuit Diagram

```
int sensorValue = 0;
void setup()
{
        pinMode(A0, INPUT);
         Serial.begin(9600);
         pinMode(9, OUTPUT);}
void loop()
{
        sensorValue = analogRead(A0);
        Serial.println(sensorValue);
        analogWrite(9, map(sensorValue, 0, 1023, 0, 255));
        delay(100); // Wait for 100 millisecond(s)
}
```

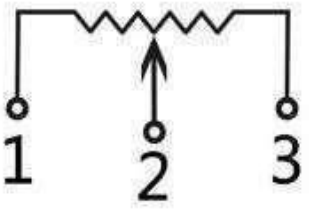


Experiment No: 10

Title : CONTROLLING THE LED BLINK RATE WITH POTENTIOMETER INTEFACING WITH ARDUINO

Introduction:

A potentiometer is a variable resistor with a knob that allows altering the resistance of the potentiometer. The potentiometer manipulates a continuous analog signal, which represents physical measurements. The potentiometer is used with Arduino to control the blink rate of the LED. Potentiometer is adjustable resistor.



Hardware Required

Arduino UNO, Breadboard, 220Ω resistor, 5mm LED, USB Cable, Jumper wires, 10KΩ potentiometer

Steps of working

- Insert the potentiometer into your breadboard and connect its centre pin to the analog pin A2 and the remaining pin to GND on the breadboard.
- Insert the LED into the breadboard. A ach the positive leg (the longer leg) to pin 13 of the Arduino via 220-ohm resistor, and the negative leg to GND. Upload the code as given below.
- Turn the potentiometer to control the brightness of the LED and move the position of pin 2 by rota ng the knob, changing the resistance value from pin 2 to both ends.
- Observe the changes in the blinking rate of the LED.

Algorithm

- This sketch works by se ng pin A2 as for the potentiometer and pin 9 as an OUTPUT to power the LED
- After that run a loop that continually reads the value from the potentiometer and sends that value as voltage to the LED
- The voltage value is between 0–5 volts, and the brightness of the LED will vary accordingly

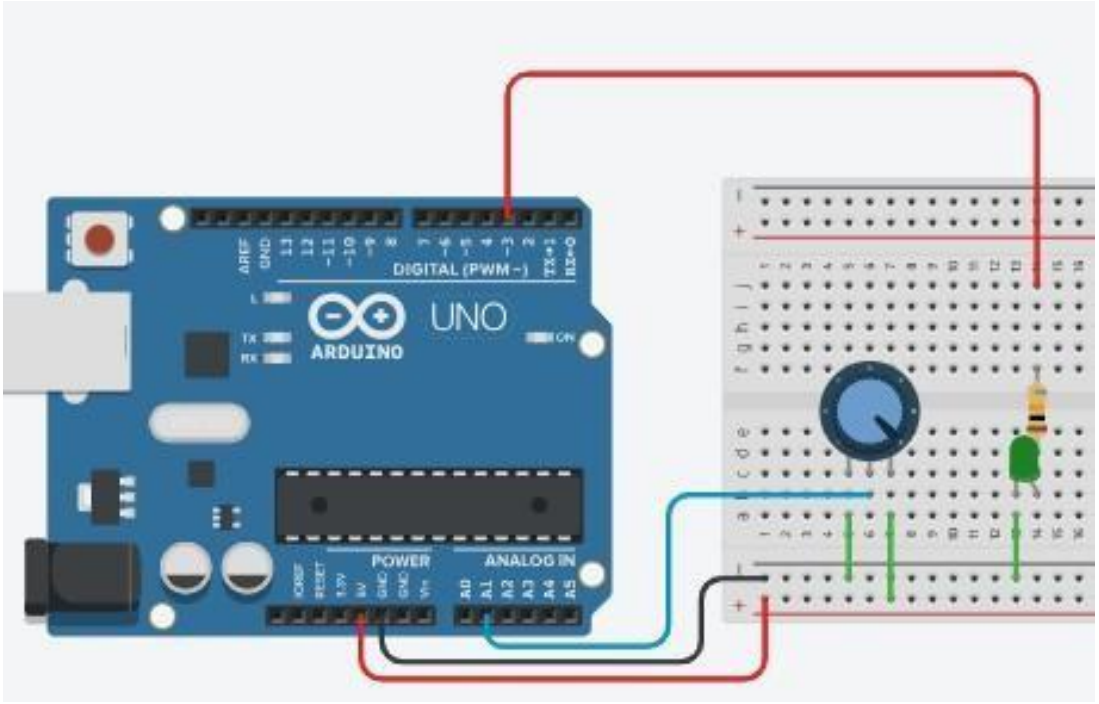
Observations

Sr no.	Voltage	Light Intensity
1	20	
2	50	
3	90	
4	170	
5	205	

Code & Circuit Diagram

```
int potPin = 1;  // select the input pin for the potentiometer
int ledPin = 3;  // select the pin for the LED
int val = 0;      // variable to store the value coming from the sensor

void setup() {
  pinMode(ledPin, OUTPUT);  // declare the ledPin as an OUTPUT
  Serial.begin(9600);       // open the serial port at 9600 bps:
}
```

```
void loop() {
   val = analogRead(potPin);    // read the value from the sensor
   digitalWrite(ledPin, HIGH);  // turn the ledPin on
   delay(val);                  // stop the program for some time
   digitalWrite(ledPin, LOW);   // turn the ledPin off
   delay(val);                  // stop the program for some time
   Serial.print(val);
    Serial.println("");
}
```

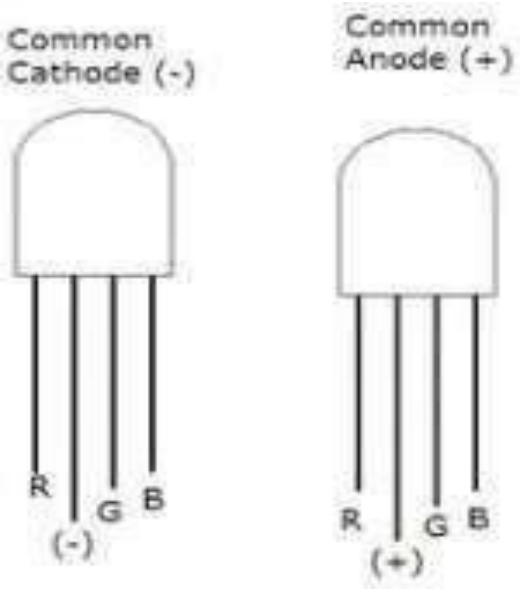


Experiment No: 11

Title : INTEFACING THE RGB LED WITHARDUINO

Introduction:

There are actually two types of RGB LED's; the common cathode one and the common anode one. In the common cathode RGB led, the cathode of all the LED's is common and we give PWM signals to the anode of LED's while in the common anode RGB led, the anode of all the LED's is common and we give PWM signals to the cathode of LED's. Inside the RGB led, there are three more LED's. So, by changing the brightness of these LED's, we can obtain many other colors. To change brightness of RGB led, we can use the PWM pins of Arduino. The PWM pins will give signal different duty cycles to the RGB led to obtain different colors.



Hardware Required

Arduino UNO, Breadboard, 3 X 220 Ω / 330 Ω resistor
USB Cable, Jumper wires, LED

Steps of working

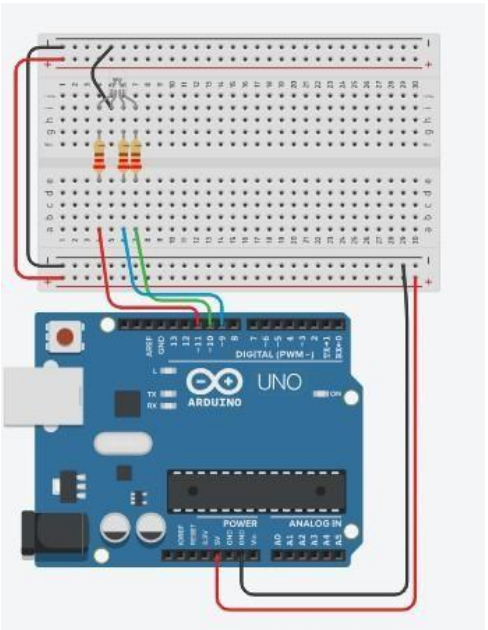
- Insert the RGB LED into your breadboard and connect its cathode pin to the GND of the Arduino.
- Insert the LED into the breadboard. A ach Red pin to pin 8, Green pin to pin 9 and Blue pin to pin 10 of the Arduino via the 220-ohm resistor, and the negative leg to GND.
- Upload the code
- Observe the changes in the color of the RGB LED.

Observations

Sr no.	Time (ms)	Color of LED
1		
2		
3		

Code & Circuit Diagram

```
void setup()
{       pinMode(11, OUTPUT);
        pinMode(10, OUTPUT);
        pinMode(9, OUTPUT);}
void loop()
{       analogWrite(11, 255);
        analogWrite(10, 0);
        analogWrite(9, 0);
        delay(1000); // Wait for 1000 millisecond(s)
        analogWrite(11, 255);
        analogWrite(10, 255);
        analogWrite(9, 102);
        delay(1000); // Wait for 1000 millisecond(s)}
```



Experiment No: 12

Title : INTRODUCTION TO INTERNET OF THINGS AND ARDUINO

Introduction:

Internet of Things (IOT)

IOT stands for "Internet of Things". The IOT is a name for the vast collection of "things" that are being networked together in the home and workplace (up to 20 billion by 2020 according to Gardner, a technology consulting firm).

Characteristics of the IOT

Actuators

IOT devices that do something. Lock doors, beep, turn lights on, or turn the TV on

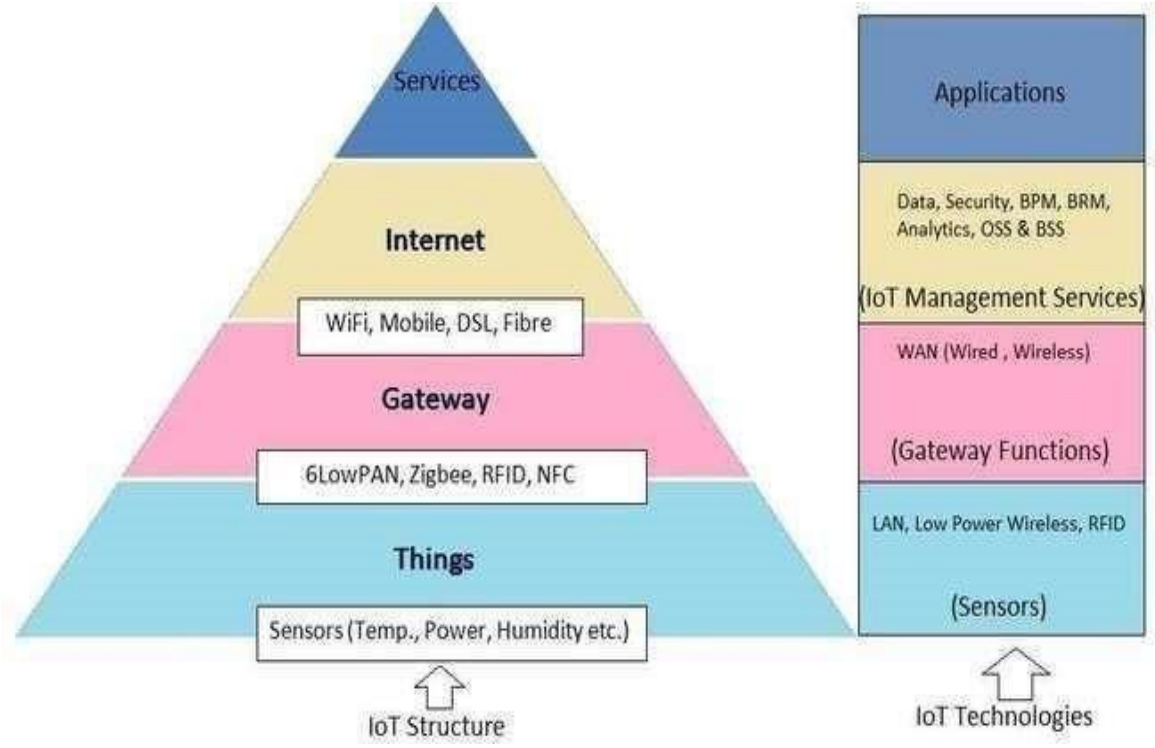
Sensing

IOT devices sense something about their environment

Networking

These IOT devices talk to one another (M2M communication) or to servers located in the local network or on the Internet. Being on the network allows the device the common ability to consume and produce data.

Communications in IoT



Communications are important to IOT projects. In fact, communications are core to the whole genre. There is a trade-off for IOT devices. The more complex the protocols and higher the data rates, the more powerful processor needed and the more electrical power the IOT device will consume.

TCP/IP base communications (think web servers; HTTP-based commutation (like REST servers); streams of data; UDP) provide the most flexibility and functionality at a cost of processor and electrical power.

Low-power Bluetooth and Zigbee types of connections allow much lower power for connections with the corresponding decrease in bandwidth and functionality. IOT projects can be all over the map with requirements for communication flexibility and data bandwidth requirements.

Arduino in IoT

In IoT applications the Arduino is used to collect the data from the sensors/devices to send it to the internet and receives data for purpose of control of actuators.

Arduino Uno

The Arduino Uno is an open-source microcontroller board based on the Microchip ATmega328P microcontroller and developed by Arduino.cc. The board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits. The board has 14 digital I/O pins (six capable of PWM output), 6 analog I/O pins, and is programmable with the Arduino IDE (Integrated Development Environment), via a type B USB cable. It can be powered by the USB cable or by an external 9-volt battery, though it accepts voltages between 7 and 20 volts. The word "uno" means "one" in Italian and was chosen to mark the initial release of Arduino So ware.

Features of the Arduino

Arduino boards are able to read analog or digital input signals from different sensors and turn it into an output such as activating a motor, turning LED on/off, connect to the cloud and many other actions.

The board functions can be controlled by sending a set of instructions to the microcontroller on the board via Arduino IDE.

Arduino IDE uses a simplified version of C++, making it easier to learn to program.

Arduino provides a standard form factor that breaks the functions of the micro- controller into a more accessible package.