

# IOT LAB MANUAL MVJ21CS71



**(Approved by AICTE | Affiliated to VTU | Recognized by  
UGC with 2(f) & 12(B) status |  
Accredited by NBA and NAAC)**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**VII Semester**

**INTERNET OF THINGS LAB – MVJ21CS71**

**ACADEMIC YEAR 2025–2026 (ODD)**

## **LABORATORY MANUAL**

**Name of the Student** :  
-----

**Branch** :  
-----

**University Seat No.** :  
-----

**Semester & Section** :  
-----

**Batch** :  
-----

# **IOT LAB MANUAL MVJ21CS71**

## **DEPARTMENT OF COMPUTERSCIENCE ANDENGINEERING**

### **VISION:**

To create an enriching learning environment where talented and industry – ready AI and ML professionals are nurtured. This is our promise.

### **MISSION:**

- To develop skilled and knowledgeable professionals in the field of Artificial Intelligence and Machine Learning.
- To impart high – quality education and career – oriented learning programs.
- To contribute towards advanced AI technologies that provide increased and better performance.
- To benefit the society through our contribution towards advancements in AI and Machine Learning.

### **PROGRAMEDUCATIONALOBJECTIVES (PEOs):**

**PEO1:** Current Industry Practices: Graduates will analyze real world problems and give solution using current industry practices in computing technology.

**PEO2:** Research and Higher Studies: Graduates with strong foundation in mathematics and engineering fundamentals that will enable graduates to pursue higher learning, R&D activities and consultancy.

**PEO3:** Social Responsibility: Graduates will be professionals with ethics, who will provide industry growth and social transformation as responsible citizens.

### **PROGRAMOUTCOMES(POs):**

**PO1:** Engineering Knowledge: Apply knowledge of mathematics, natural science, computing, engineering fundamentals and an engineering specialization as specified in WK 1 to WK 4 respectively to develop to the solution of complex engineering problems.

**PO2:** Problem Analysis: Identify, formulate, review research literature and analyse complex engineering problems reaching substantiated conclusions with consideration for sustainable development. (WK1 to WK4)

## **IOT LAB MANUAL MVJ21CS71**

**PO3:** Design/ Development of Solutions: Design creative solutions for complex engineering problems and design/ develop systems/ components/ processes to meet identified needs with consideration for the public health and safety, whole-life cost, net zero carbon, culture, society and environment as required. (WK5)

**PO4:** Conduct Investigations of Complex Problems: Conduct investigations of complex engineering problems using research-based knowledge including design of experiments, modelling, analysis & interpretation of data to provide valid conclusions. (WK8)

**PO5:** Engineering Tool Usage: Create, select and apply appropriate techniques, resources and modern engineering & IT tools, including prediction and modelling recognizing their limitations to solve complex engineering problems. (WK2andWK6)

**PO6:** The Engineer and The World: Analyse and evaluate societal and environmental aspects while solving complex engineering problems for its impact on sustainability with reference to economy, health, safety, legal framework, culture and environment. (WK1, WK5, and WK7).

**PO7:** Ethics: Apply ethical principles and commit to professional ethics, human values, diversity and inclusion; adhere to national & international laws. (WK9)

**PO8:** Individual and Collaborative Teamwork: Function effectively as an individual, and as a member or leader in diverse/multi-disciplinary teams.

**PO9:** Communication: Communicate effectively and inclusively within the engineering community and society at large, such as being able to comprehend and write effective reports and design documentation, make effective presentations considering cultural, language, and learning differences

**PO10:** Project Management and Finance: Apply knowledge and understanding of engineering management principles and economic decision-making and apply these to one's own work, as a member and leader in a team, and to manage projects and in multidisciplinary environments.

**PO11:** Life-Long Learning: Recognize the need for and have the preparation and ability for

- i) Independent and life-long learning
- ii) Adaptability to new and emerging technologies and
- iii) Critical thinking in the broadest context of technological change. (WK8)

# IOT LAB MANUAL MVJ21CS71

## **Program Educational Objectives (PEOs):**

- **IT Proficiency:** Graduates will excel as IT Experts with extensive knowledge to analyze and design solutions to Computer Engineering problems.
- **Social & moral principles:** Graduates will work in a team, showcase professionalism, ethical values expose themselves to current trends and become responsible Engineers.
- **Higher education:** Graduates will pursue higher studies with the sound knowledge of fundamental concepts and skills in basic sciences and IT disciplines.

## **PROGRAMSPECIFICOUTCOMES (PSOs):**

1. **PSO1:** Programming: Ability to understand, analyze and develop computer programs In the areas related to algorithms, system software, multimedia, web design, DBMS, and networking for efficient design of computer-based systems of varying complexity.
2. **PSO2:** Practical Solution: Ability to practically provide solutions for real world problems with a broad range of programming language and open-source platforms in various computing domains.
3. **PSO3:** Research: Ability to use innovative ideas to do research in various domains to solve societal problems.

<b>Course outcomes:</b>	
CO1	Interpret the impact and challenges posed by IoT networks leading to new architectural models.
CO2	Compare and contrast the deployment of smart objects and the technologies to connect them to network.
CO3	Appraise the role of IoT protocols for efficient network communication.
CO4	Elaborate on the need for Data Analytics and Security in IoT.
CO5	Design of Android and Rasper Beery Program for IOT Application in Industry. an Illustrate different sensor technologies for sensing real world entities

## IOT LAB MANUAL MVJ21CS71

### CO-PO Mapping

CO/PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO1	3		3	-	2	1	-	-	3	2	-	-
CO2	3	2	-	-	-	2	-	-	2	2	-	-
CO3	3	2	-	-	-	2	-	-	2	2	-	-
CO4	2	2	2	1	3	1	-	-	2	2	-	-
CO5	2	3	3	3	3	1	2	-	3	2	-	-

## IOT LAB MANUAL MVJ21CS71

### Lab Experiments:

SL NO	Experiment Name	RBT Level
1	a) Blinking of LED under different duty cycle timing with buzzer indication. b) LED ON/OFF Using Microswitch and displaying LED status on Serial Monitor. c) LED ON/OFF controlling only through Serial Monitor without Microswitch.	L3
2	a) Auto Fading of LED brightness through PWM. (b) LED Fading using Potentiometer displaying POT status value on Serial Monitor.	L3
3	a) Simulation of Traffic light. b) UP/DOWN decade counter.	L3
4	Obstacle detector using IR module with visual indication and buzzer alarm.	L3
5	Controlling lamp through relay using Light sensor using LDR module.	L3
6	Distance measurement device using ultra sonic sensor with reverse parking anticollision warning LED indicator and alarm.	L3
7	Detection of smoke with visual indication and buzzer alarm.	L3
8	Displaying of your Name USN and College name in LCD using I2C protocol.	L3
9	Displaying Humidity and Temperature using DTH-11 module in LCD using I2C protocol.	L3
10	Write an Arduino code to demonstrate home automation system to control the watering of lawn/garden through automatic water motor sprinkler using soil sensor with 3 visual conditions of soil namely dry, moist and fully wet.	L3

**IOT LAB MANUAL MVJ21CS71**

11	Write an Arduino code to demonstrate the controlling of lamp using relay module by PIR sensor.	L3
12	Voice control of lamp/relay through blue tooth module.	L3
13	Home automation system for fire alarm using flame sensor to extinguish the fire automatically through water sprinkler system.	L3

## 1. Internet of Things

The Internet of Things (IoT) describes the phenomenon of everyday devices connecting to the Internet through tiny, embedded sensors and computing power

We're entering a new era of computing technology that many are calling the Internet of Things (IoT). Machine to machine, machine to infrastructure, machine to environment, the Internet of Everything, the Internet of Intelligent Things, intelligent systems—call it what you want, but it's happening, and its potential is huge. We see the IoT as billions of smart, connected “things” that will encompass every aspect of our lives, and its foundation is the intelligence that embedded processing provides. The IoT is comprised of smart machines interacting and communicating with other machines, objects, environments and infrastructures.

### Applications of IoT

- Machine-to-machine communication
- Machine-to-infrastructure communication
- Telehealth: remote or real-time pervasive monitoring of patients, diagnosis and drug delivery
- Continuous monitoring of, and firmware upgrades for, vehicles
- Asset tracking of goods on the move
- Automatic traffic management
- Remote security and control
- Environmental monitoring and control
- Home and industrial building automation
- “Smart” applications, including cities, water, agriculture, buildings, grid, meters, broadband, cars, appliances, tags, animal farming and the environment, to name a few.

## 2. Introduction to Arduino

The Arduino board is a small microcontroller board, which is a small circuit (the board) that contains a whole computer on a small chip (the microcontroller). Arduino is composed of two



major parts: the Arduino board, which is the piece of hardware you work on when you build your objects; and the Arduino IDE, the piece of software you run on your computer. You use the IDE to create a sketch (a little computer program) that you upload to the Arduino board. The sketch tells the board what to do.

The Arduino software is easy-to-use for beginners, yet flexible enough for advanced users. It runs on Mac, Windows, and Linux. Teachers and students use it to build low cost scientific instruments, to prove chemistry and physics principles, or to get started with programming and robotics. It has the specific advantages such as

- **Inexpensive** - Arduino boards are relatively inexpensive compared to other microcontroller platforms. The least expensive version of the Arduino module can be assembled by hand, and even the pre-assembled Arduino modules cost less than \$50
- **Cross-platform** - The Arduino Software (IDE) runs on Windows, Macintosh OSX, and Linux operating systems. Most microcontroller systems are limited to Windows.
- **Simple, clear programming environment** - The Arduino Software (IDE) is easy-to-use for beginners, yet flexible enough for advanced users to take advantage of as well. For teachers, it's conveniently based on the Processing programming environment, so students learning to program in that environment will be familiar with how the Arduino IDE works.
- **Open source and extensible software** - The Arduino software is published as open source tools, available for extension by experienced programmers. The language can be expanded through C++ libraries, and people wanting to understand the technical details can make the leap from Arduino to the AVR C programming language on which it's based. Similarly, you can add AVR-C code directly into your Arduino programs if you want to.
- **Open source and extensible hardware** - The plans of the Arduino boards are published under a Creative Commons license, so experienced circuit designers can make their own version of the module, extending it and improving it. Even relatively inexperienced users can build the breadboard version of the module in order to understand how it works and save money.

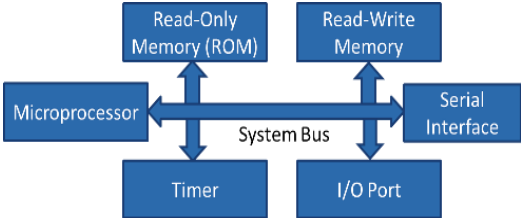
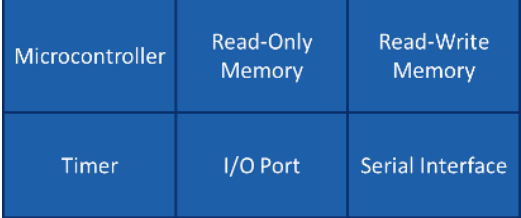
### Installing Arduino on Your Computer

To program the Arduino board, you must first download the development environment (the IDE) from here: [www.arduino.cc/en/Main/Software](http://www.arduino.cc/en/Main/Software). Choose the right version for your

## **IOT LAB MANUAL MVJ21CS71**

operating system. Download the file and double-click on it to open it; on Windows or Linux, this creates a folder named arduino-[version], such as arduino-1.0. Drag the folder to wherever you want it: your desktop, your Program Files folder (on Windows), etc. On the Mac, double-clicking it will open a disk image with an Arduino application (drag it to your Applications folder). Now whenever you want to run the Arduino IDE, you'll open up the arduino (Windows and Linux) or Applications folder (Mac), and double-click the Arduino icon.

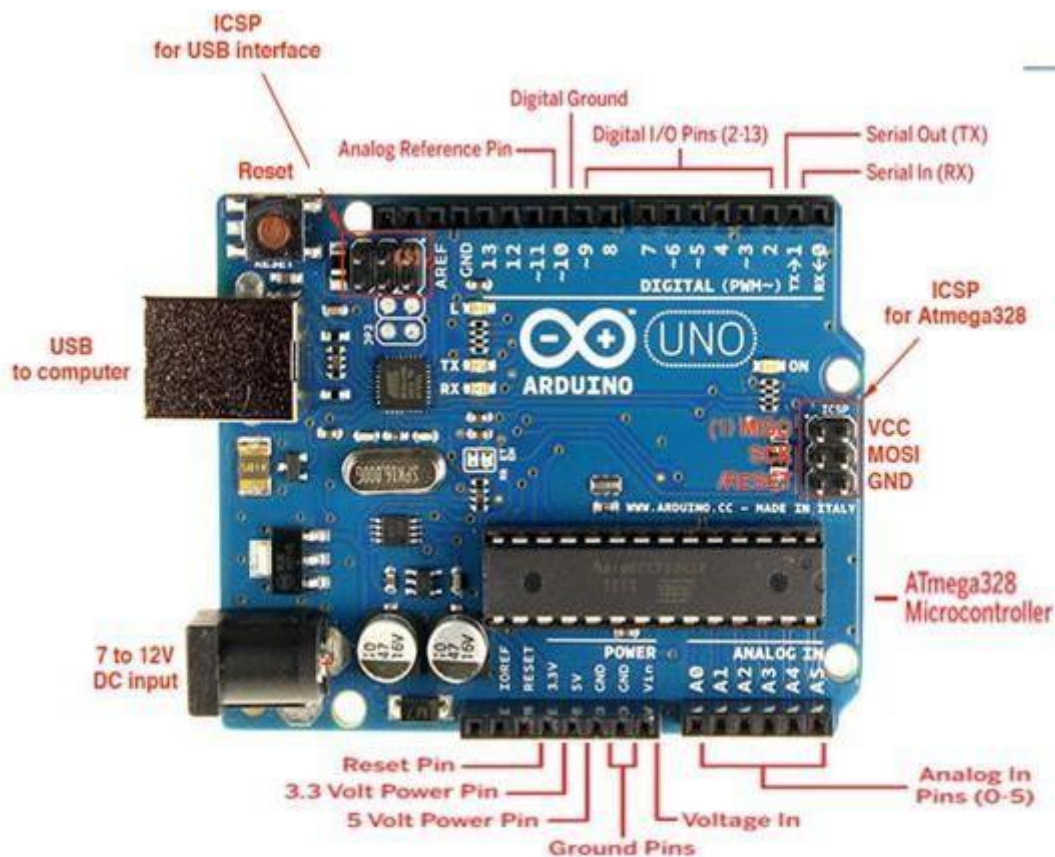
**Difference between Microprocessor and Microcontroller**

Microprocessor	Micro Controller
	
Microprocessor is heart of Computer system.	Micro Controller is a heart of embedded system.
It is just a processor. Memory and I/O components have to be connected externally	Micro controller has external processor along with internal memory and i/O components
Since memory and I/O has to be connected externally, the circuit becomes large.	Since memory and I/O are present internally, the circuit is small.
Cannot be used in compact systems and hence inefficient	Can be used in compact systems and hence it is an efficient technique
Cost of the entire system increases	Cost of the entire system is low
Due to external components, the entire power consumption is high. Hence it is not suitable to used with devices running on stored power like batteries.	Since external components are low, total power consumption is less and can be used with devices running on stored power like batteries.
Most of the microprocessors do not have power saving features.	Most of the micro controllers have power saving modes like idle mode and power saving mode. This helps to reduce power consumption even further.

## IOT LAB MANUAL MVJ21CS71

Since memory and I/O components are all external, each instruction will need external operation, hence it is relatively slower.	Since components are internal, most of the operations are internal instruction, hence speed is fast.
Microprocessor have less number of registers, hence more operations are memory based.	Micro controller have more number of registers, hence the programs are easier to write.
Microprocessors are based on von Neumann model/architecture where program and data are stored in same memory module	Micro controllers are based on Harvard architecture where program memory and Data memory are separate
Mainly used in personal computers	Used mainly in washing machine, MP3 players

### 3. The Arduino Development board



#### A. Digital Pins

The digital pins on an Arduino board can be used for general purpose input and output via the `pinMode()`, `digitalRead()`, and `digitalWrite()` commands. Each pin has an internal pull-up

resistor which can be turned on and off using `digitalWrite()` (w/ a value of HIGH or LOW, respectively) when the pin is configured as an input. The maximum current per pin is 40 mA.

Some other specific functions of pins are listed below:

- **Serial: 0 (RX) and 1 (TX).** Used to receive (RX) and transmit (TX) TTL serial data
- **External Interrupts: 2 and 3.** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value.
- **PWM: 3, 5, 6, 9, 10, and 11.** Provide 8-bit PWM output with the `analogWrite()` function. On boards with an ATmega8, PWM output is available only on pins 9, 10, and 11.
- **BT Reset: 7.** (Arduino BT-only) Connected to the reset line of the Bluetooth module.
- **SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).** These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language.
- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

## **B. Analog Pins**

In addition to the specific functions listed below, the analog input pins support 10-bit analog-to-digital conversion (ADC) using the `analogRead()` function. Most of the analog inputs can also be used as digital pins: analog input 0 as digital pin 14 through analog input 5 as digital pin 19. Analog inputs 6 and 7 (present on the Mini and BT) cannot be used as digital pins.

## **C. Power Pins**

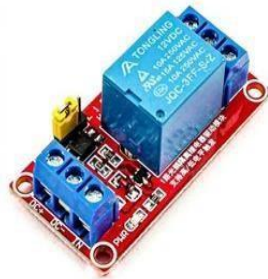
- **VIN:** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V.** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
- **3V3.** A 3.3 volt supply generated by the on-board FTDI chip. **GND.** Ground pins.

## D. Other Pins

- **AREF.** Reference voltage for the analog inputs. Used with analog Reference().
- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.



BREAD BOARD



5V RELAY MODULE Photo by ElectroPeak



3V LED



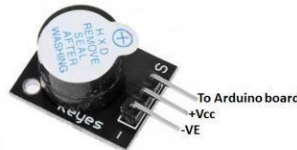
MALE TO MALE FEMALE TO FEMALE  
AND MALE TO FEMALE CONNECTING WIRES



E12 Range, Resistor 220Ω, 5% Tolerance, Carbon Film



220 Ω RESISTOR



To Arduino board  
+Vcc  
-VE

## 1 a). General Purpose Input / Output Interfacing

### LED Blinking

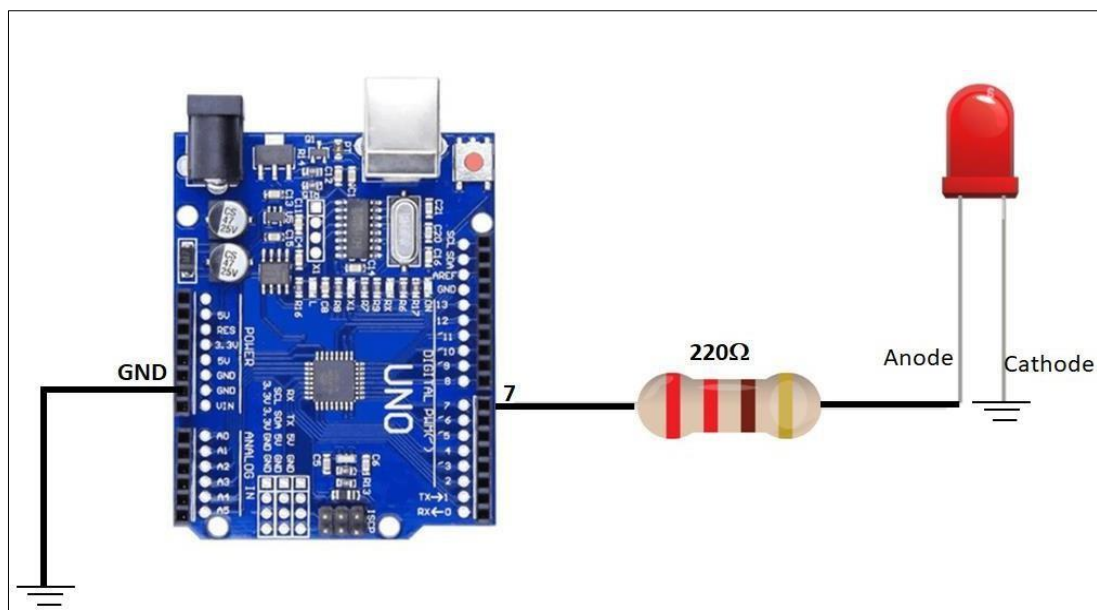
#### Aim

- Blinking of LED under different given duty cycle timing with buzzer indication.

#### Hardware Required

- Arduino Board
- LEDs

#### Circuit Diagram



#### Code:

```
void setup () {  
    // put your setup code here, to run once:  
    pinMode(7, OUTPUT); // initialize the digital pin as an output.  
}  
  
void loop() {
```

## **IOT LAB MANUAL MVJ21CS71**

// put your main code here, to run repeatedly:

digitalWrite(7,HIGH); // turn the LED on (HIGH is the voltage level)

delay(1000); // wait for a second

digitalWrite(7,LOW); // turn the LED off by making the voltage LOW

delay(1000); // wait for a second

}



## 1 b). LED ON/OFF Using Microswitch and displaying LED status on Serial Monitor

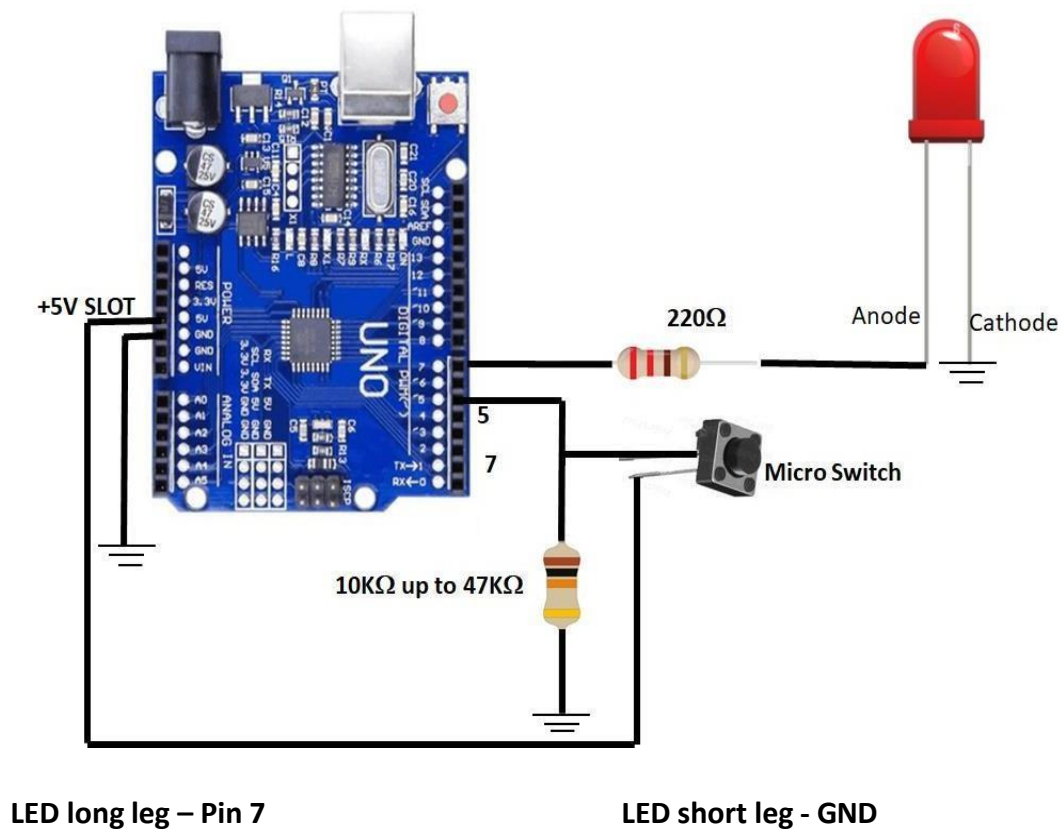
### Aim

Turn an LED ON /OFF using a Microswitch.

### Hardware Required

- Arduino Board
- LED
- Microswitch

### Circuit Diagram



**Code:**

```
#define LED_PIN 7

#define MICROSWITCH_PIN 5

void setup() {

  pinMode(LED_PIN, OUTPUT);

  pinMode(MICROSWITCH_PIN, INPUT);

  Serial.begin(9600);

}

void loop() {

  if (digitalRead(MICROSWITCH_PIN) == HIGH) {

    digitalWrite(LED_PIN, HIGH);

    Serial.println(1);

  }

  else {

    digitalWrite(LED_PIN, LOW);

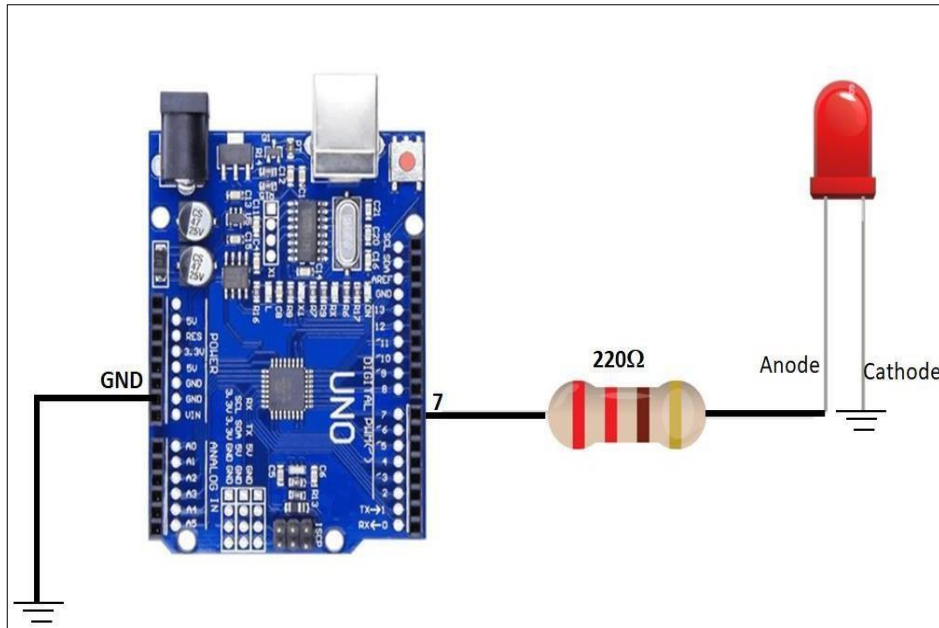
    Serial.println(0);

  }

}
```

**1 c). LED ON/OFF only through Serial Monitor without Microswitch.**

**Circuit Diagram**



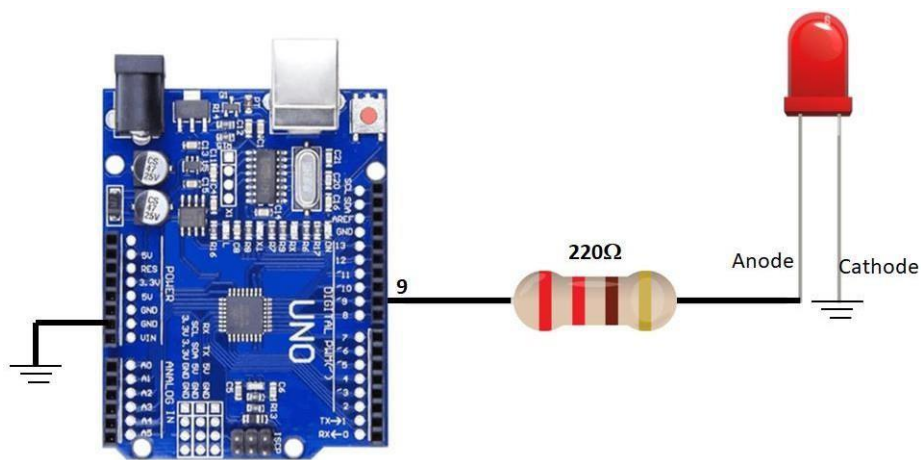
**CODE**

```
void setup()
{
  pinMode(7, OUTPUT);
  Serial.begin(9600);
  while (!Serial);
  Serial.println("Input 1 to Turn LED on and 2 to off");
}

void loop() {
  if (Serial.available())
  {
    int state = Serial.parseInt();
    if (state == 1)
```

## **IOT LAB MANUAL MVJ21CS71**

```
{  
  
digitalWrite(2, HIGH);  
  
Serial.println("Command completed LED turned ON");  
  
}  
  
if (state == 2)  
{  
  
digitalWrite(2, LOW);  
  
Serial.println("Command completed LED turned OFF");  
  
}  
  
}  
  
}
```

**2 a). Auto fading of LED brightness through PWM****CODE:**

```

int led = 9;

int brightness = 0;

int fadeAmount = 5;

void setup() {

    // put your setup code here, to run once:

    pinMode(led, OUTPUT);

} void loop() {

    // put your main code here, to run repeatedly:

    analogWrite(led, brightness);

    brightness = brightness + fadeAmount;

    if (brightness == 0 || brightness == 125) {

        fadeAmount = -fadeAmount ;

    }

    delay(200);

}

```

## 2 b. LED Fading using Potentiometer displaying POT status value on Serial Monitor

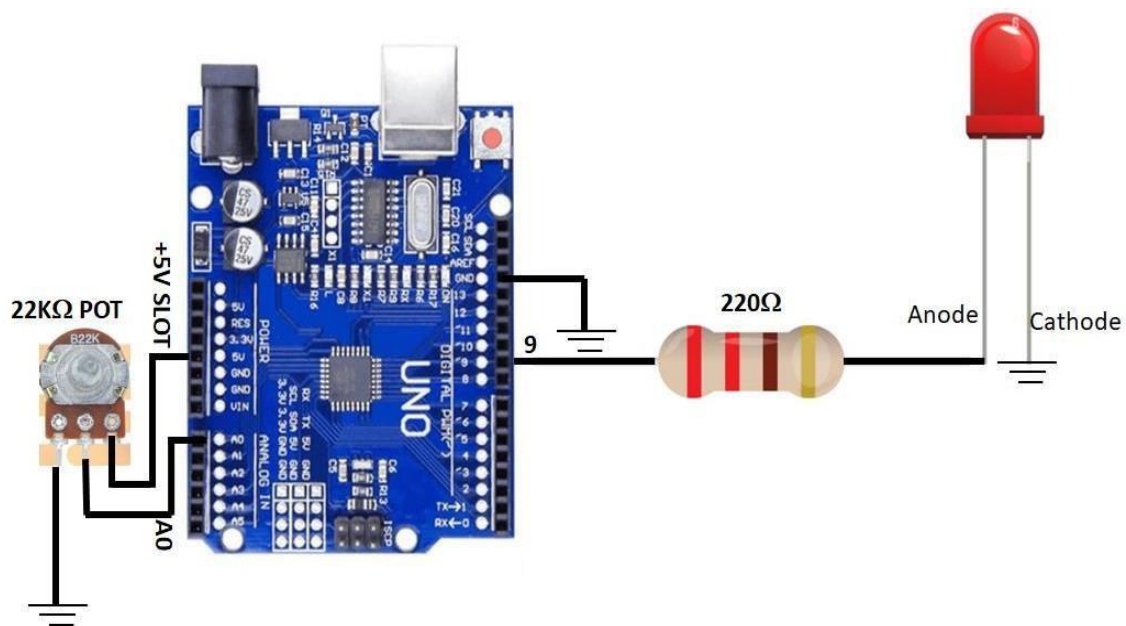
**Aim** To control the brightness of an LED using a Potentiometer.

### Hardware Required

- Arduino Board
- LED
- Potentiometer 22K
- 220 $\Omega$  Resistor.



### Circuit Diagram



### Code:

```
int potPin=A0;  
  
int gPin=9;  
  
int potVal;  
  
float LEDVal;
```

## **IOT LAB MANUAL MVJ21CS71**

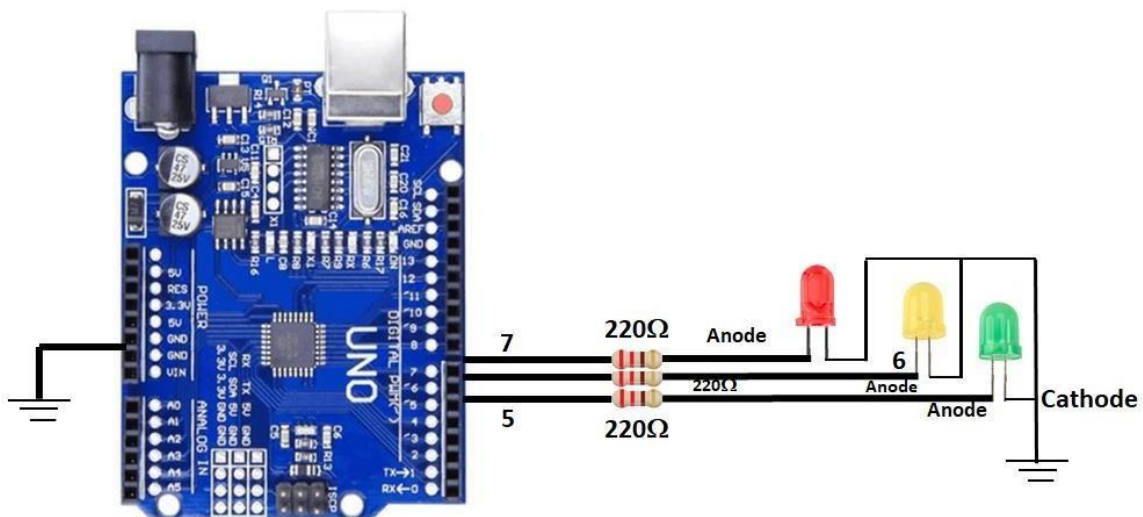
```
void setup() {  
  
    pinMode(potPin,INPUT);  
  
    pinMode(gPin, OUTPUT);  
  
    Serial.begin(9600);  
  
}  
  
void loop() {  
  
    potVal=analogRead(potPin);  
  
    LEDVal=(255./1023.)*potVal;  
  
    analogWrite(gPin,LEDVal);  
  
    delay(500);  
  
    Serial.println(LEDVal);  
  
}
```

### 3 a. Write Arduino code to simulate a Traffic Controller

**Aim** Traffic Signal Simulation

#### Hardware Required

- Arduino Board
- LEDs
- Resistors –  $220\Omega$
- Bread board



#### Circuit Diagram

#### Code

```
void setup() {  
  
    // put your setup code here, to run once:  
  
    pinMode(5, OUTPUT);  
  
    pinMode(6, OUTPUT);  
  
    pinMode(7, OUTPUT);  
  
}
```



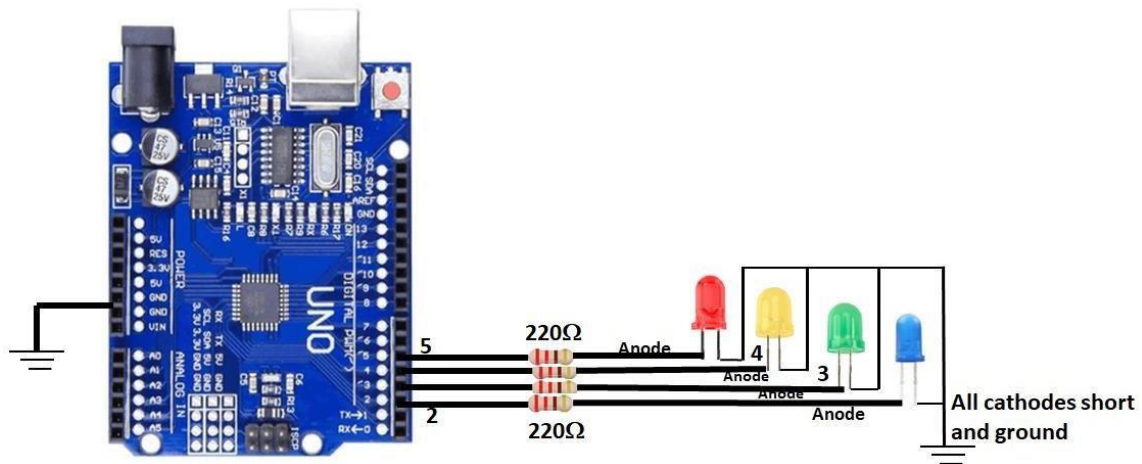
```
void loop() {  
    // put your main code here, to run repeatedly:  
  
    digitalWrite(5,HIGH);  
  
    digitalWrite(6,LOW);  
  
    digitalWrite(7,LOW);  
  
    delay(4500);  
  
    digitalWrite(5,LOW);  
  
    digitalWrite(6,HIGH);  
  
    digitalWrite(7,LOW);  
  
    delay(2000);  
  
    digitalWrite(5,LOW);  
  
    digitalWrite(6,LOW);  
  
    digitalWrite(7,HIGH);  
  
    delay(2800);  
  
}
```

**3. (b) Write an Arduino program to using 4 LED's to develop a chasing pattern back and forth.**

**Code**

```
void setup() {  
    // put your setup code here, to run once:  
    pinMode(4, OUTPUT);  
    pinMode(5, OUTPUT);  
    pinMode(6, OUTPUT);  
    pinMode(7, OUTPUT);  
}  
void loop() {  
    // put your main code here, to run repeatedly:  
    digitalWrite(4,HIGH);  
    delay(100);  
    digitalWrite(5,HIGH);  
    delay(100);  
    digitalWrite(6,HIGH);  
    delay(100);  
    digitalWrite(7,HIGH);  
    delay(100);  
    digitalWrite(7,LOW);  
    delay(100);  
    digitalWrite(6,LOW);  
    delay(100);  
    digitalWrite(5,LOW);  
    delay(100);  
    digitalWrite(4,LOW);  
    delay(100);  
}
```

3 c). Write an Arduino code to demonstrate decade UP/DOWN counter.



### CODE

```
void setup() { // put your setup code here, to run once:

pinMode(2, OUTPUT);

pinMode(3, OUTPUT);

pinMode(4, OUTPUT);

pinMode(5, OUTPUT);

}

void loop() {

// put your main code here, to run repeatedly:

digitalWrite(2,HIGH);

delay(1000);

//Number 2 digitalWrite(2,LOW);
```

```
digitalWrite(3,HIGH);  
  
delay(1000);  
  
//Number 3  
  
digitalWrite(3,HIGH);  
  
digitalWrite(2,HIGH);  
  
delay(1000);  
  
// Number 4  
  
digitalWrite(2,LOW);  
  
digitalWrite(3,LOW);  
  
digitalWrite(4,HIGH);  
  
delay(1000);  
  
//Number 5  
  
digitalWrite(2,HIGH);  
  
digitalWrite(3,LOW);  
  
digitalWrite(4,HIGH);  
  
delay(1000);  
  
//Number 6 digitalWrite(2,LOW);  
  
digitalWrite(3,HIGH);  
  
digitalWrite(4,HIGH);  
  
delay(1000);  
  
// Number 7  
  
digitalWrite(2,HIGH);  
  
digitalWrite(3,HIGH);
```

```
digitalWrite(4,HIGH);
```

```
delay(1000);
```

```
// Number 8
```

```
digitalWrite(2,LOW);
```

```
digitalWrite(3,LOW);
```

```
digitalWrite(4,LOW);
```

```
digitalWrite(5,HIGH);
```

```
delay(1000);
```

```
// Number 9
```

```
digitalWrite(2,HIGH);
```

```
digitalWrite(3,LOW);
```

```
digitalWrite(4,LOW);
```

```
digitalWrite(5,HIGH);
```

```
delay(1000);
```

```
//RESET
```

```
digitalWrite(2,LOW);
```

```
digitalWrite(3,LOW);
```

```
digitalWrite(4,LOW);
```

```
digitalWrite(5,LOW);
```

```
delay(1000);
```

```
// Number 9
```

```
digitalWrite(2,HIGH);
```

```
digitalWrite(3,LOW);
```

```
digitalWrite(4,LOW);
```

## **IOT LAB MANUAL MVJ21CS71**

```
digitalWrite(5,HIGH);
```

```
delay(1000);
```

```
// Number 8
```

```
digitalWrite(2,LOW);
```

```
digitalWrite(3,LOW);
```

```
digitalWrite(4,LOW);
```

```
digitalWrite(5,HIGH);
```

```
delay(1000);
```

```
// Number 7
```

```
digitalWrite(2,HIGH);
```

```
digitalWrite(3,HIGH);
```

```
digitalWrite(4,HIGH);
```

```
digitalWrite(5,LOW);
```

```
delay(1000);
```

```
//Number 6 digitalWrite(2,LOW);
```

```
digitalWrite(3,HIGH);
```

```
digitalWrite(4,HIGH);
```

```
delay(1000);
```

```
//Number 5
```

```
digitalWrite(2,HIGH);
```

```
digitalWrite(3,LOW);
```

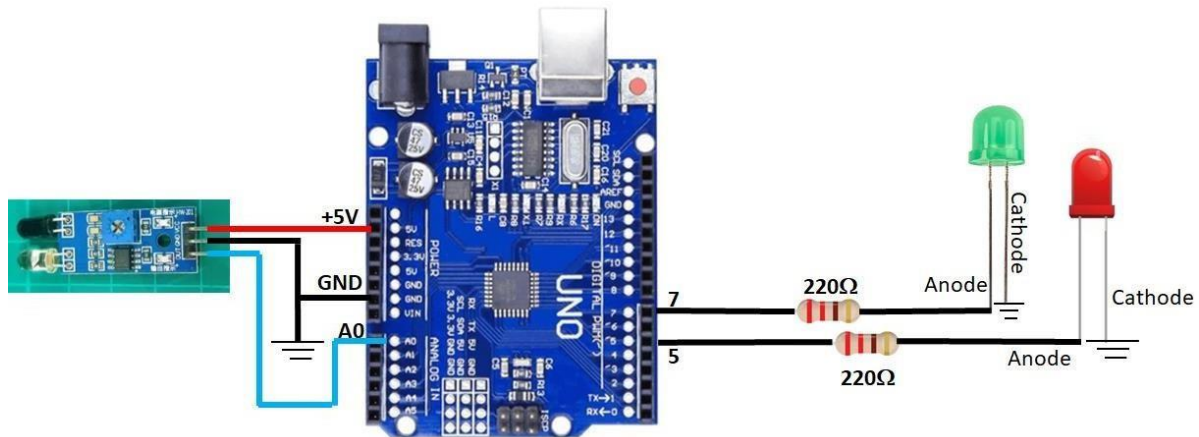
```
digitalWrite(4,HIGH);
```

```
delay(1000);
```

```
// Number 4
```

```
digitalWrite(2,LOW);  
  
digitalWrite(3,LOW);  
  
digitalWrite(4,HIGH);  
  
delay(1000);  
  
//Number 3  
  
digitalWrite(3,HIGH);  
  
digitalWrite(2,HIGH);  
  
digitalWrite(4,LOW);  
  
delay(1000);  
  
//Number 2 digitalWrite(2,LOW);  
  
digitalWrite(3,HIGH);  
  
delay(1000);  
  
//Number 1  
  
digitalWrite(2,HIGH);  
  
digitalWrite(3,LOW);  
  
delay(1000);  
  
//RESET  
  
digitalWrite(2,LOW);  
  
digitalWrite(3,LOW);  
  
digitalWrite(4,LOW);  
  
digitalWrite(5,LOW);  
  
delay(1000);  
  
}
```

4. Write an Arduino code to demonstrate an Obstacle detector using IR module with visual indication and buzzer alarm.



#### CODE:

```
void setup()
{
  Serial.begin(9600);
  pinMode(5,OUTPUT);
  pinMode(7,OUTPUT);
}

void loop()
{
  int v = analogRead(A0); // v means value
  Serial.println(v);
  delay(300);
  if(v < 260)
  {
    digitalWrite(7,LOW);
```



```
digitalWrite(5,HIGH);  
  
delay(300);  
  
Serial.println("MOTION DETECTED");
```

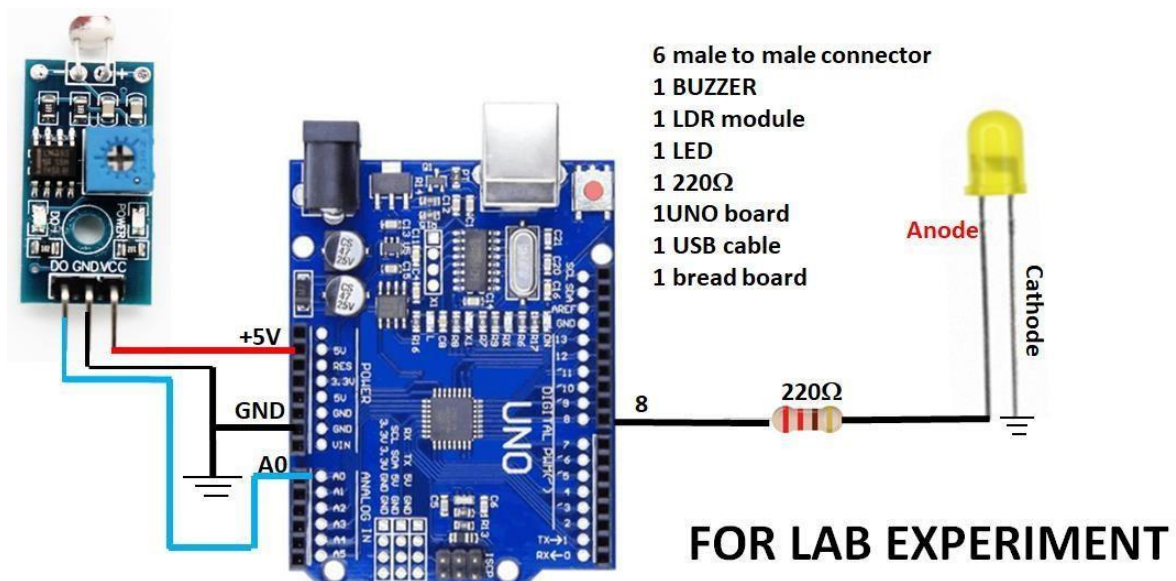
```
digitalWrite(5,LOW);  
  
}  
  
else  
  
{  
  
digitalWrite(5,LOW);  
  
digitalWrite(7,HIGH);  
  
}  
  
}
```

## 5. Controlling lamp through relay using Light sensor using LDR module.

### Aim

### Hardware Required

- 2 LED
- 1 LDR
- 10K resistor
- **Connection:**
  1. Attach one leg of LDR to 5V and another leg to Arduino Analog pin A0
  2. Attach one leg of 10K resistor with that leg of LDR connected to A0
  3. Attach another leg of resistor to the ground



4. Connect the positive leg of LED to pin 5 and another LED to pin 7 negative of both the LED's to GND.

### Code:

```
void setup()
{
  pinMode(8,OUTPUT);
  Serial.begin(9600);
}
```

## **IOT LAB MANUAL MVJ21CS71**

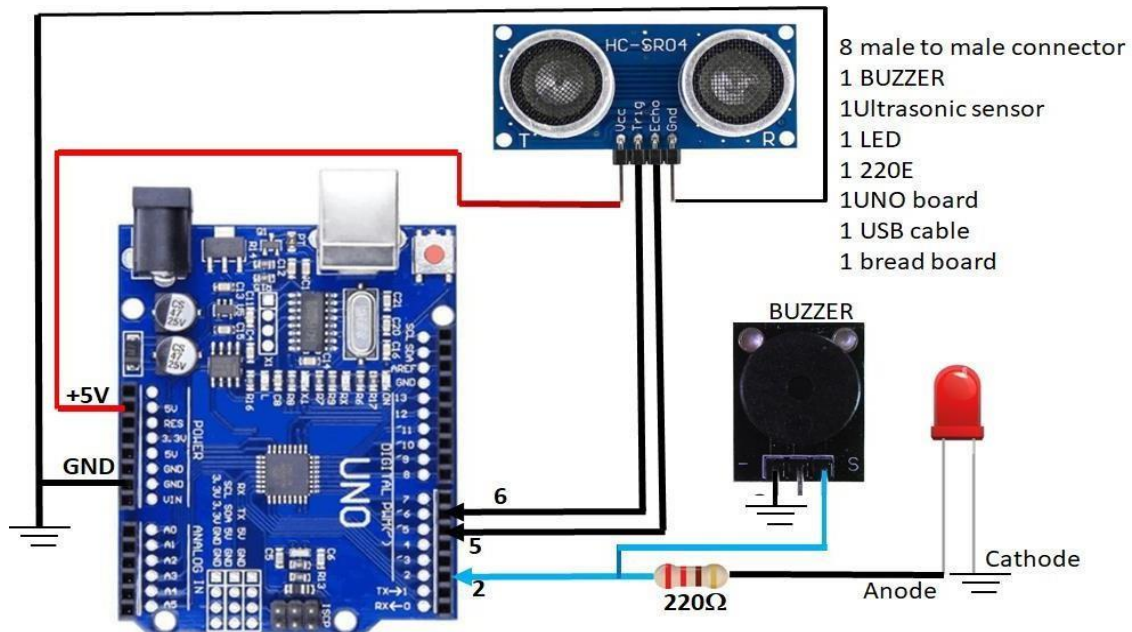
```
void loop()
{
int a=analogRead(A0);
Serial.println(a);
if(a >= 400)
{

digitalWrite(8,HIGH);
Serial.print("CLOUDY/EVENING SUN HAS SET");
}
else
{
digitalWrite(8,LOW);
Serial.print("ENOUGH LIGHT/DAY TIME ");
}
delay(1000);
}
```

### **Observation:**

While lights are switched off in the room, LED should switch ON, when lights are switched on in the room, LED should switch off immediately.

6. Write an Arduino code to demonstrate a distance measurement device using ultrasonic sensor with reverse parking anticollision warning LED indicator and alarm.



## CODE

```
const int trig Pin = 6; //configuration of ultrasonic sensor VCC TRIG ECHO GND

const int echo Pin = 5;

const int buzzer = 2;

float new delay;

long duration;

int distance;

void setup()

{

    // put your setup code here, to run once:

    Serial.begin(9600);

    pin Mode (trig Pin, OUTPUT);
```

```
pinMode(echoPin, INPUT);

pinMode(buzzer,OUTPUT);

}void loop() {

digitalWrite(trigPin,LOW); // put your main code here, to run repeatedly:

delay Microseconds(2);

digital Write(trigPin,HIGH);

delayMicroseconds(12);

digitalWrite(trigPin,LOW);

duration = pulseIn(echoPin, HIGH);

distance= duration*0.034/2;

Serial.print("Distance: ");

new_delay= (distance *3) +30;

Serial.print(distance);

Serial.println("  cm");

if (distance < 50)

{

digitalWrite(buzzer,HIGH);

delay(new_delay);

digitalWrite(buzzer,LOW);

}

else

{

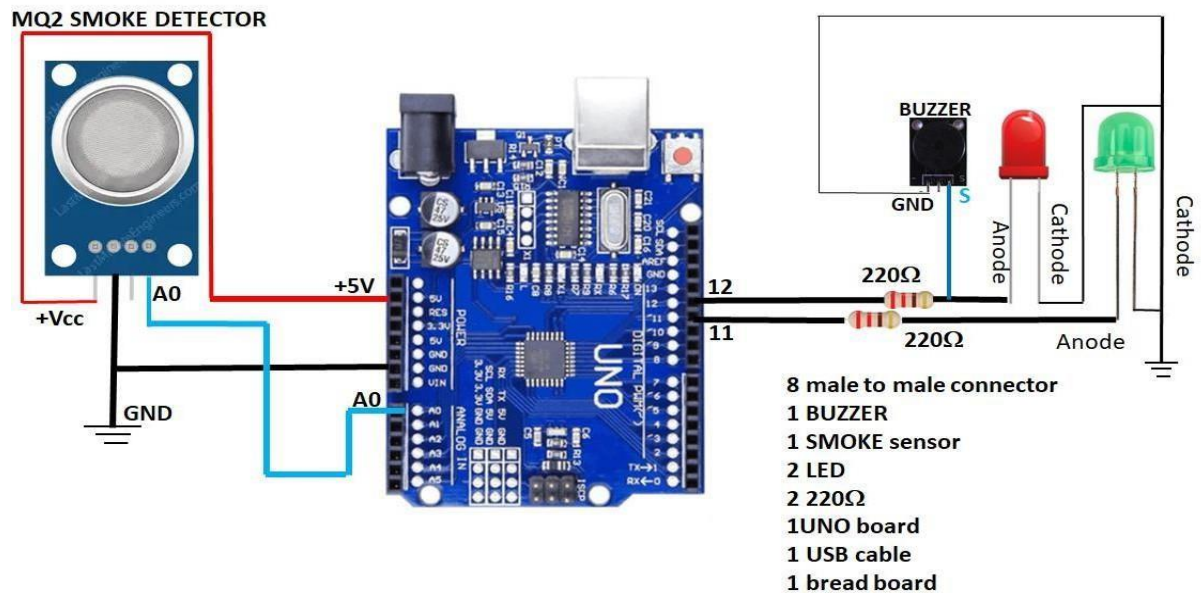
digitalWrite(buzzer,LOW);

}
```

```
delay(200);  
  
}
```

## 7. Write an Arduino code to demonstrate the detection of smoke with visual indication and buzzer alarm.

### CODE



```

int smoke;

void setup() {

  // put your setup code here, to run once:

  pinMode(11,OUTPUT);

  pinMode(12,OUTPUT);

  Serial.begin(9600);

}

void loop() {

  // put your main code here, to run repeatedly:

  smoke= analogRead(A0);

  Serial.println("Smoke Sensor");

  Serial.println(smoke);

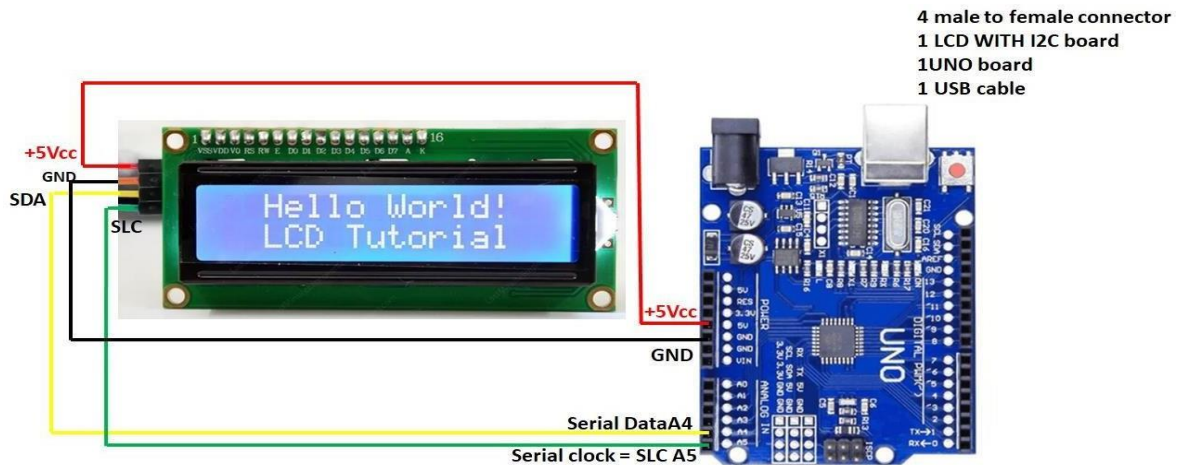
  delay(1000);

  if(smoke>250)
  
```

```
{  
    digitalWrite(11,HIGH);  
  
    digitalWrite(12,HIGH);  
    delay(1000);  
    digitalWrite(12,LOW);  
    Serial.println("SMOKE DETECTED HOGAY HAKUSKODIATHYA");  
}  
else  
{  
    digitalWrite(11,HIGH);  
    digitalWrite(12,LOW);  
    Serial.println("NO SMOKE HOGAY ILLA");  
}
```



8 a). Write an Arduino code to display your NAME USN college and department name on the I2C protocol LCD display



### CODE (LEARNING THE CONCEPT OF CODE TO DISPLAY SENTENCE ON LCD

```
#include <Wire.h>

#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27, 16, 1);

void setup() {

  Serial.begin(9600);

  lcd.init();

  lcd.backlight();

  lcd.setCursor(1,0);

  lcd.print("");

  lcd.setCursor(2,1);

} void loop() {

  lcd.print(" JACOB S E 1MJ22CS014");

  lcd.print(" MVJ CSE ");

  //Turn off the display:
```

```
//lcd.noDisplay();
```

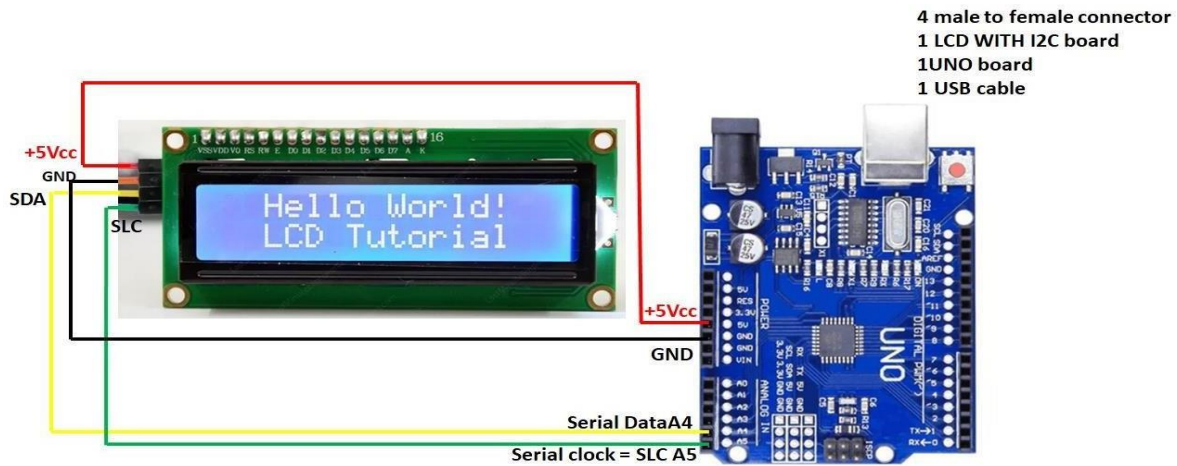
```
//delay(100);
```

```
//Turn on the display:
```

```
//lcd.display();
```

```
delay(300);
```

**8 a). Write an Arduino code to display your NAME USN college and department name on the I2C protocol LCD display**



### CODE (LEARNING THE CONCEPT OF CODE TO DISPLAY SENTENCE ON LCD

```
#include <Wire.h>

#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27, 16, 1);

void setup() {

  Serial.begin(9600);

  lcd.init();

  lcd.backlight();

  lcd.setCursor(1,0);

  lcd.print("");

  lcd.setCursor(2,1);

  }void loop() {

  lcd.print(" JACOB S E 1MJ22CS014");

  lcd.print(" MVJ CSE ");

  //Turn off the display:
```

## **IOT LAB MANUAL MVJ21CS71**

```
//lcd.noDisplay();
```

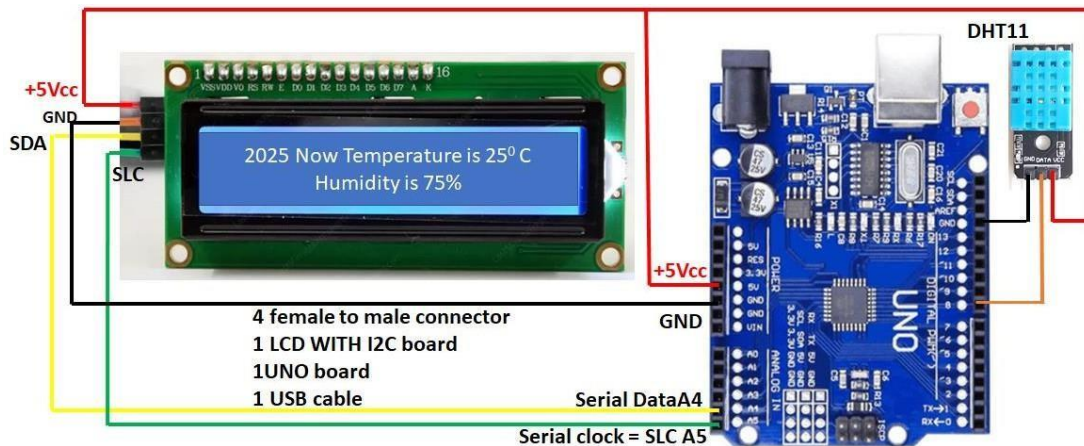
```
//delay(100);
```

```
//Turn on the display:
```

```
//lcd.display();
```

```
delay(300);
```

9. Write an Arduino code to display the temperature and humidity on the I2C protocol LCD display using DHT11.



## CODE DHT11

```
#include <DHT.h>;
//I2C LCD:
#include <LiquidCrystal_I2C.h>
#include <Wire.h>
LiquidCrystal_I2C lcd(0x27,16,2); // set the LCD address to 0x27 for a 16 chars and 2 line display
SDA = A4 SCL A5
//Constants
#define DHTPIN 8 // what pin we're connected to
#define DHTTYPE DHT11 // DHT 11
DHT dht(DHTPIN, DHTTYPE); /// Initialize DHT sensor for normal 16mhz Arduino
//Variables
//int chk;
int h; //Stores humidity value
int t; //Stores temperature value
void setup()
{
  Serial.begin(9600);
  Serial.println("Temperature and Humidity Sensor Test");
  dht.begin();
  lcd.init(); //initialize the lcd
```

```
    lcd.backlight(); //open the backlight
}

void loop()
{
    //Read data and store it to variables h (humidity) and t (temperature)
    // Reading temperature or humidity takes about 250 milliseconds!
    h = dht.readHumidity();
    t = dht.readTemperature();

    //Print temp and humidity values to serial monitor

    Serial.print("Humidity: ");
    Serial.print(h);
    Serial.print(" %, Temp: ");
    Serial.print(t);
    Serial.println(" ° Celsius");
    // set the cursor to (0,0):
    // print from 0 to 9:
    lcd.setCursor(0, 0);
    lcd.println(" Now Temperature ");

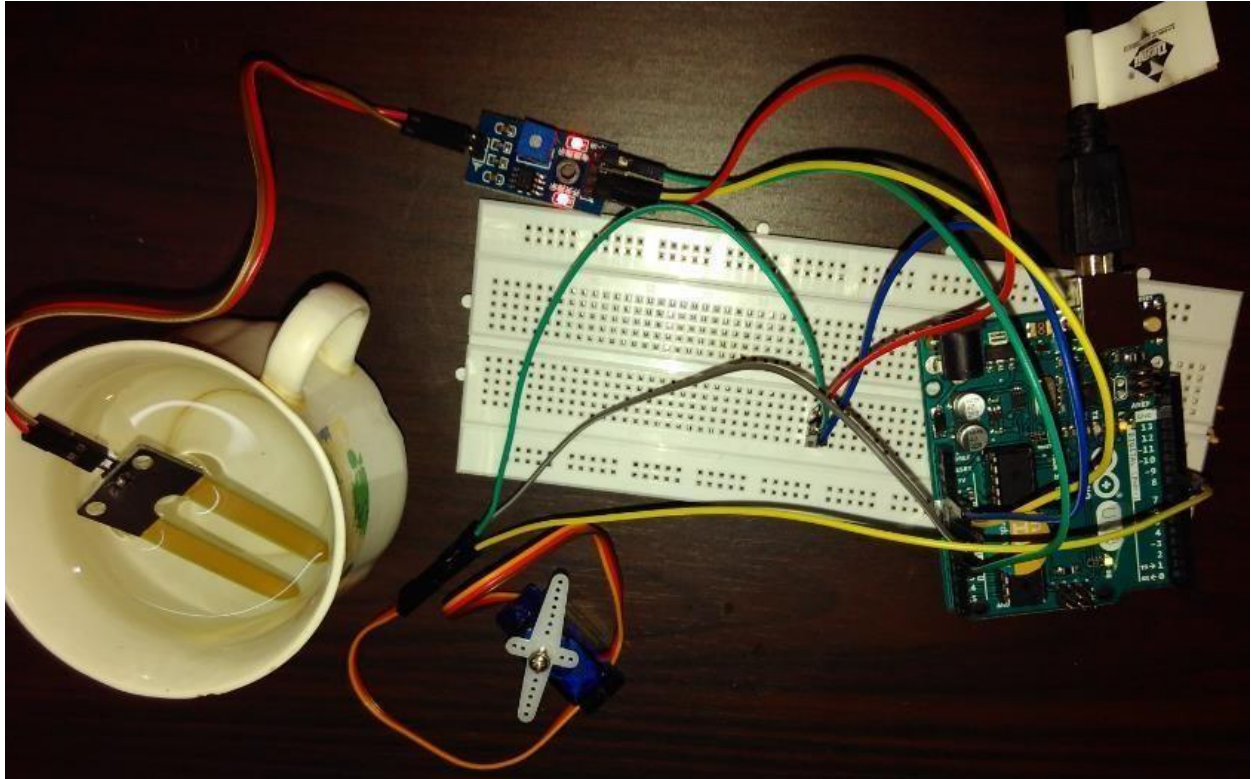
    lcd.setCursor(0, 1);
    lcd.print("T:");
    lcd.print(t);
    lcd.print("C");
    lcd.setCursor(6, 1);
    lcd.println("2024 ");

    lcd.setCursor(11, 1);
    lcd.print("H:");
    lcd.print(h);
    lcd.print("%");
    delay(1000); //Delay 1 sec.
}
```

10. Write an Arduino code to demonstrate home automation system to control the watering of lawn/garden through automatic water motor sprinkler using soil sensor with 3 visual conditions of soil namely dry, moist and fully wet.

### Aim

Sensing the soil moisture and sprinkling the Water simulation



### Hardware Required

- Arduino
- Moisture Sensor
- Breadboard
- DC/AC motor

### Circuit diagram

Moisture sensor VCC to Arduino 5V

Moisture sensor GND to Arduino GND

Moisture sensor A0 to Arduino A0

**CODE**

```
int sensor_pin = A0; // PIN CONFIGURATION OF SOIL SENSOR A0 D0 GND VCC

void setup() {

  // put your setup code here, to run once:

  Serial.begin(9600);

  pinMode(sensor_pin, INPUT);

  pinMode(2, OUTPUT);

  pinMode(3, OUTPUT);

  pinMode(4, OUTPUT);

}

void loop() {

  // put your main code here, to run repeatedly:

  int sensor_data = analogRead(sensor_pin);

  Serial.println(sensor_data);

  if(sensor_data >= 1000)

  {

    digitalWrite(2, HIGH);

    digitalWrite(3, LOW);

    digitalWrite(4, LOW);

    Serial.println("No moisture Soil is dry");

    delay(1000);

  }
```



```
else if(sensor_data <=700 && sensor_data >= 630)

{

digitalWrite(2,LOW);

digitalWrite(3,HIGH);

digitalWrite(4,LOW);

Serial.println("There is some moisture, Soil is medium");

delay(1000);

}

else if(sensor_data <=600)

{

digitalWrite(2,LOW);

digitalWrite(3,LOW);

digitalWrite(4,HIGH);

Serial.println("Soil is wet");

delay(1000);

}
```

## Home automation

11. Write an Arduino code to demonstrate the controlling of lamp using relay module by PIR sensor.

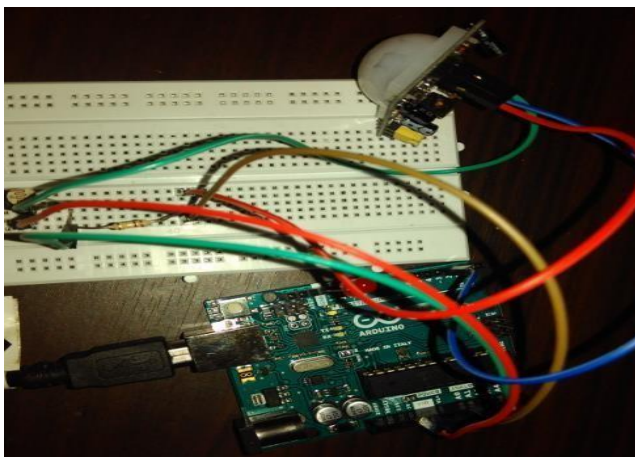
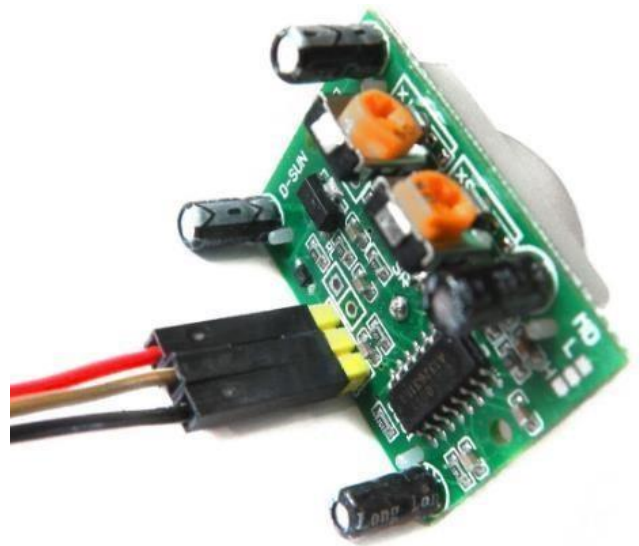
### Hardware Required

- 1 LED
- 1 PIR
- 220 $\Omega$  resistor
- 1 PIR

Understanding the configuration of PIR sensor:

#### Connection:

1. Attach one leg of LDR to 5V and another leg to Arduino Analog pin A0
2. Attach one leg of 110K register with that leg of LDR connected to A0
3. Attach another leg of register to the ground
4. Connect the positive leg of LED to pin 11 and negative to GND
5. Connect positive leg of PIR to 5V and negative leg to GND
6. Connect output pin of PIR to digital pin 3



### Steps and Observation:

1. Upload the program to Arduino and open Serial monitor
2. It will show in Serial monitor: **calibrating sensor ..... done**
3. Wait until it is shown: **SENSOR ACTIVE**
4. Caution: Avoid any movements near the sensor. **It should not show: “Motion detected at” in monitor**
5. Switch off lights avoiding motion near sensor, LED should not glow
6. Make some movement near PIR
7. LED should keep glowing for some time
8. While LED is ON, switch on the light, LED should switch off immediately.
9. Upload the program to Arduino and open Serial monitor
10. It will show in Serial monitor: **calibrating sensor ..... done**
11. Wait until it is shown: **SENSOR ACTIVE**
12. Caution: Avoid any movements near the sensor. **It should not show: “Motion detected at” in monitor**
13. Switch off lights avoiding motion near sensor, LED should not glow
14. Make some movement near PIR
15. LED should keep glowing for some time
16. While LED is ON, switch on the light, LED should switch off immediately.

### CODE

```
void setup()
{
  Serial.begin(9600);
  pinMode(5,OUTPUT);
```

## **IOT LAB MANUAL MVJ21CS71**

```
}  
  
void loop()  
  
{  
  
int v = analogRead(A0); // v means value  
  
Serial.println(v);  
  
delay(1000);  
  
if(v > 700)  
  
{  
  
digitalWrite(5,HIGH);  
  
Serial.println("PERSON PRESENT");  
  
}  
  
else  
  
{  
  
digitalWrite(5,LOW);  
  
}  
  
}
```

12. Write an Arduino code to demonstrate home automation system to control 3 different household loads by human voice through Android Mobile and blue tooth module.

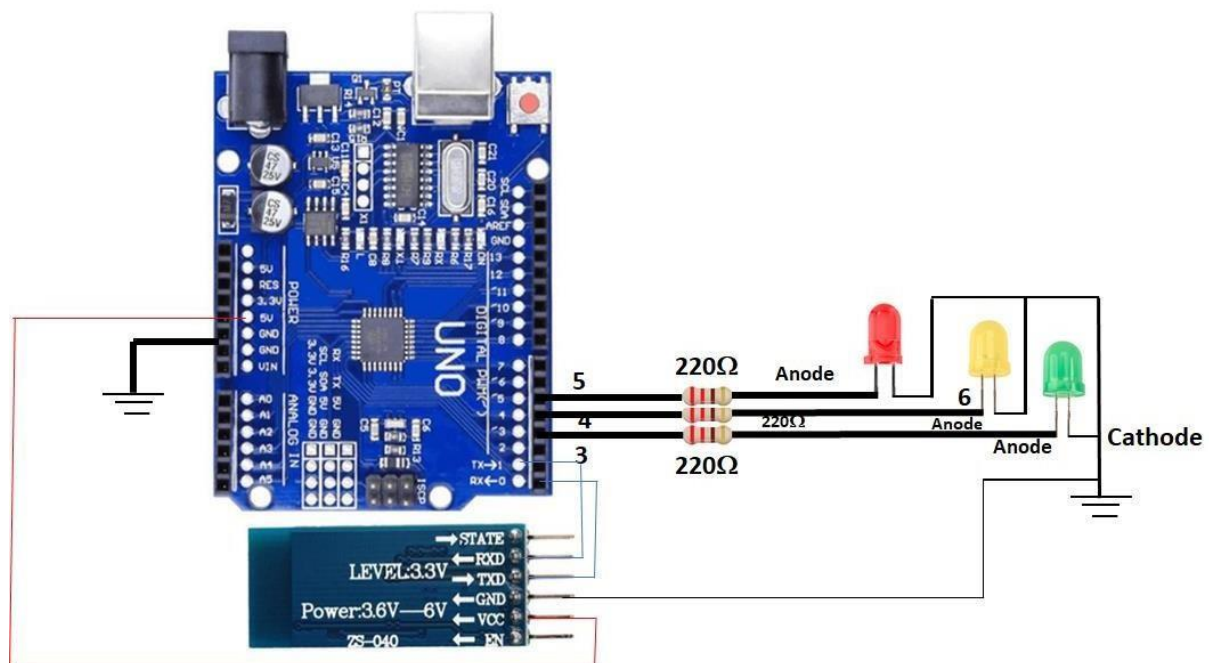
**Aim:**

To control the working of relay through Android Mobile.

**Hardware Required:**

- Arduino
- 4-Channel relay
- Bluetooth Module
- Android phone

**Circuit diagram**



Wiring Arduino Pin Diagram

Output:1 to Pin 4 ( Arduino Board )

Output 2 to Pin 5

Output 3 to Pin 6

Output 4 to Pin 7

Bluetooth Module Tx to Pin 0

Bluetooth Module Rx to Pin 1

=====

VCC of Bluetooth & relay should be connected to Arduino 5V(through breadboard)

GND of Bluetooth & relay should be connected to Arduino GND

**Important:**

- **First upload the code to Arduino and then connect Bluetooth module**
- **Install Arduino Bluetooth device App in your android mobile**

A data transmission via Bluetooth.

Device1 ON sent “a” , Device1 OFF sent “A”

Device2 ON sent “b” , Device2 OFF sent “B”

Device3 ON sent “c” , Device3 OFF sent “C”

Device4 ON sent “d” , Device4 OFF sent “D”

**Code:**

```
int bedroom = 3;
```

```
int kitchen = 4;
```

```
int bathroom = 5;
```

```
void setup() {
```

```
    Serial.begin(9600);
```

```
pinMode(bedroom,OUTPUT);

pinMode(kitchen,OUTPUT);

pinMode(bathroom,OUTPUT);
}

char c;//char c; // one byte reserved in memory to hold the ASCII code.

String voice;// The string is a data type that stores text rather than the integer values.

//It is similar to other data types such as integer, float, etc., So here in the code it is written as
String voice which means it stores voice

void loop() {

    if (Serial.available()>0)// Serial. available() returns the number of characters in the buffer
    waiting to be read, so if you don't send it anything it will always return zero/false. That means
    initially whatever it gets serially the states of the LED should not be changed and be initially
    off(0).

    {

        voice="";// here you are telling in the loop what voice command you are about to tell like
        bathroom, bedroom and kitchen.

        voice=Serial.readString();

        Serial.print(voice+'\n'); // It will print on the serial monitor what you say(voice)like
        bathroom, bedroom and kitchen.

    }

    if(voice=="bedroom")// so if microcontroller gets the voice as bedroom then.....

    {

        digitalWrite(bedroom,HIGH);// output pin 3 which is connected to bedroom will go high
        (Switch ON).

    } else if(voice=="bedroom of")// but however if it hears as bedroom of then.....

    {
```

digitalWrite(bedroom,LOW);// output pin 3 which is connected to bedroom will go low (Switch OFF).

}

if(voice=="kitchen");// so if microcontroller gets the voice as kitchen then.....

{

digitalWrite(kitchen,HIGH);// output pin 4 which is connected to kitchen will go high (Switch ON).

} else if(voice=="kitchen of");// but however if it hears as kitchen of then.....

{

digitalWrite(kitchen,LOW);// output pin 4 which is connected to kitchen will go low (Switch OFF).

}

if(voice=="bathroom");// so if microcontroller gets the voice as bathroom then.....

{

digitalWrite(bathroom,HIGH);// output pin 5 which is connected to bathroom will go high (Switch ON).

} else if(voice=="bathroom of");// but however if it hears as bathroom of then.....

{

digitalWrite(bathroom,LOW);// output pin 5 which is connected bathroom will go low (Switch OFF).

}

}

//IMPORTANT DO NOT CONNECT THE BLUE TOOTH MODULE HC05 BEFORE LOADING THE PROGRAM AFTER LOADING THE PROGRAM THEN CONNECT THE BLUE TOOTH MODULE.



Study experiments on home automation.

13. Write an Arduino code to demonstrate home automation system for fire alarm using flame sensor to extinguish the fire automatically through water sprinkler system.

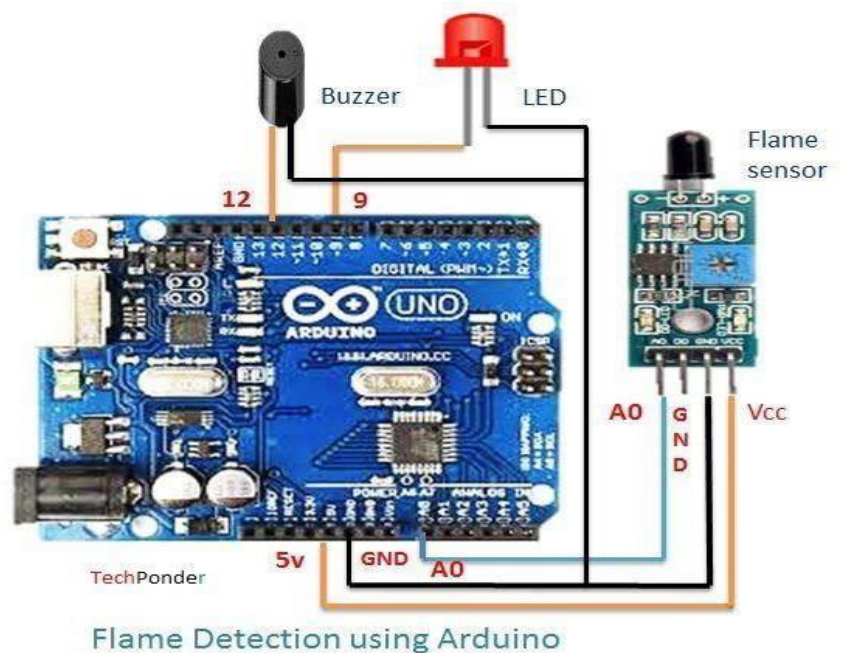
**Aim**

Fire alarm simulation

**Hardware Required**

- Flame sensor (Analogue Output)
- Arduino
- Bread board
- LED
- Buzzer
- Connecting wires

**Circuit Diagram**



**Flame sensor interfacing to Arduino**

Flame sensor to Arduino

vcc -> vcc

gnd -> gnd

A0 -> A0

**CODE**

```
// declare the ledPin and buzzer as an OUTPUT:
```

```
Serial.begin(9600);

pinMode(sensor_pin, INPUT);

pinMode(2,OUTPUT);


pinMode(3,OUTPUT);

pinMode(4,OUTPUT);

}

void loop()

{

int sensor_data = analogRead(sensor_pin);

Serial.println(sensor_data);

Serial.println("WLECOME TO NVR SUN PEARL APPTS FIRE ALRAM");

if (sensor_data <100)

{

Serial.println("Fire Detected Life in Danger");

digitalWrite(2,HIGH);

digitalWrite(4,LOW);

digitalWrite(3,HIGH);

delay(500);

digitalWrite(3,LOW);

delay(500);

}

else if(sensor_data >100)

{

Serial.println("NO Fire Detected Life is COOL");
```

**IOT LAB MANUAL MVJ21CS71**

```
Serial.println(2, LOW);
```

```
digitalWrite(2,LOW);
```

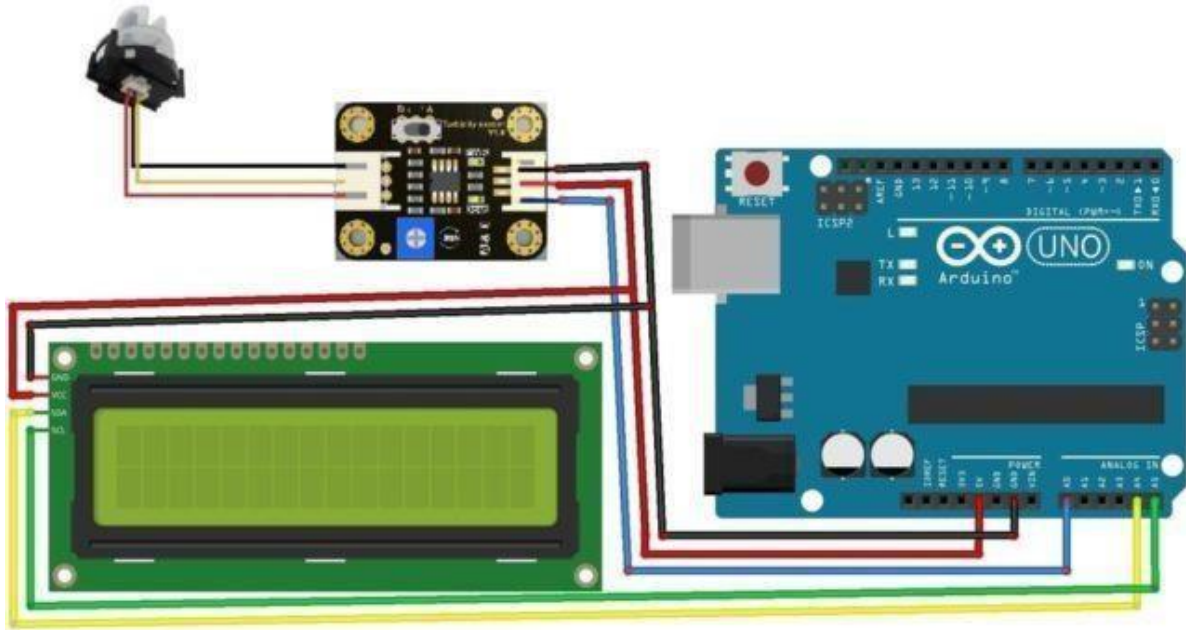
```
digitalWrite(3,LOW);
```

```
digitalWrite(4,HIGH)
```

```
delay(500);
```

```
}
```

14. write an Arduino code to demonstrate home automation system to find the fitness of daily water quality using turbidity sensor and displaying 3 conditions namely clear fit for drinking, cloudy can be used for washing and dirty unfit for drinking on I2C protocol lcd.



#### CODE

```
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27, 2, 16);

int sensorPin = A0; //A0 FOR ARDUINO/ 36 FOR ESP

void setup()
{
  Serial.begin(9600);

  // initialize LCD

  lcd.init();

  // turn on LCD backlight

  lcd.backlight();
}

void loop() {
```

```
int sensorValue = analogRead(sensorPin);

Serial.println(sensorValue);

int turbidity = map(sensorValue, 0, 750, 100, 0);

delay(100);

lcd.setCursor(0, 0);

lcd.print("Turbidity:");

lcd.print(" ");

lcd.setCursor(10, 0);

lcd.print(turbidity);

delay(100);

if (turbidity < 35) {

    lcd.setCursor(0, 1);

    lcd.print(" its clear fit for drinking ");

    Serial.print(" its clear fit for drinking ");

}

if ((turbidity > 38) && (turbidity < 50)) {

    lcd.setCursor(0, 1);

    lcd.print(" its cloudy can be used for washing ");

    Serial.print(" its cloudy can be used for washing ");

}if (turbidity > 50) {

    lcd.setCursor(0, 1);

    lcd.print(" its dirty unfit for drinking ");

    Serial.print(" its dirty unfit for drinking ");

}

}
```

**VIVA QUESTIONS**

1. What is the Internet of Things (IoT)?
2. How does IoT differ from the traditional Internet?
3. What are the key characteristics of IoT?
4. What are the benefits of IoT?
5. What is the difference between analog and digital signals?
6. How does PWM (Pulse Width Modulation) work?
7. What is the purpose of a relay module in a circuit?
8. How does an LDR (Light Dependent Resistor) work?
9. What is the difference between a potentiometer and a variable resistor?
10. How does an ultrasonic sensor measure distance?
11. What is the purpose of an IR module in obstacle detection?
12. How does a smoke detector work?
13. How can you use a soil sensor to control an automatic watering system?
14. How can you control the brightness of an LED using PWM?
15. What is the purpose of a decade counter in a circuit?
16. How can you simulate a traffic light using LEDs and a microcontroller?
17. How does a light sensor (LDR) control a lamp through a relay module?
18. What is the purpose of the I2C protocol in communication between devices?
19. How can you display humidity and temperature data on an LCD using a DHT-11 module?

What is the purpose of a soil sensor in an automatic waterin

**Do's**

Do wear ID card and follow dress code.

- Do log off the computers when you finish.
- Do ask for assistance if you need help.
- Do keep your voice low when speaking to others in the LAB.
- Do ask for assistance in downloading any software.
- Do make suggestions as to how we can improve the LAB.
- In case of any hardware related problem, ask LAB in charge for solution.
- If you are the last one leaving the LAB, make sure that the staff in charge of the LAB is informed to close the LAB.
- Be on time to LAB sessions.
- Do keep the LAB as clean as possible.

**Don'ts**

- Do not use mobile phone inside the lab.
- Don't do anything that can make the LAB dirty (like eating, throwing waste papers etc).
- Do not carry any external devices without permission.
- Don't move the chairs of the LAB.
- Don't exchange any part of one computer with another.
- Don't leave the computers of the LAB turned on while leaving the LAB.
- Do not install or download any software or modify or delete any system files on any lab computers.
- Do not damage, remove, or disconnect any labels, parts, cables, or equipment.
- Don't attempt to bypass the computer security system.
- Do not tread or modify the user's file.

If you leave the lab, do not leave your personal belonging unattended. We are not responsible for any theft.

- Do not install or download any software or modify or delete any system files on any lab computers.
- Do not damage, remove, or disconnect any labels, parts, cables, or equipment.
- Don't attempt to bypass the computer security system.
- Do not read or modify their user's file.
- If you leave the lab, do not leave your personal belongings unattended. We are not responsible for any theft.