Experiment – 6

Aim: Design a circuit that uses an LDR to interface with an Arduino board and other required parts.

Components

- Arduino Uno R3
- Small Breadboard
- Jumper cable
- LED
- Power supply
- Photoresistor
- Relay SPDT

Theory

1. Arduino Uno R3

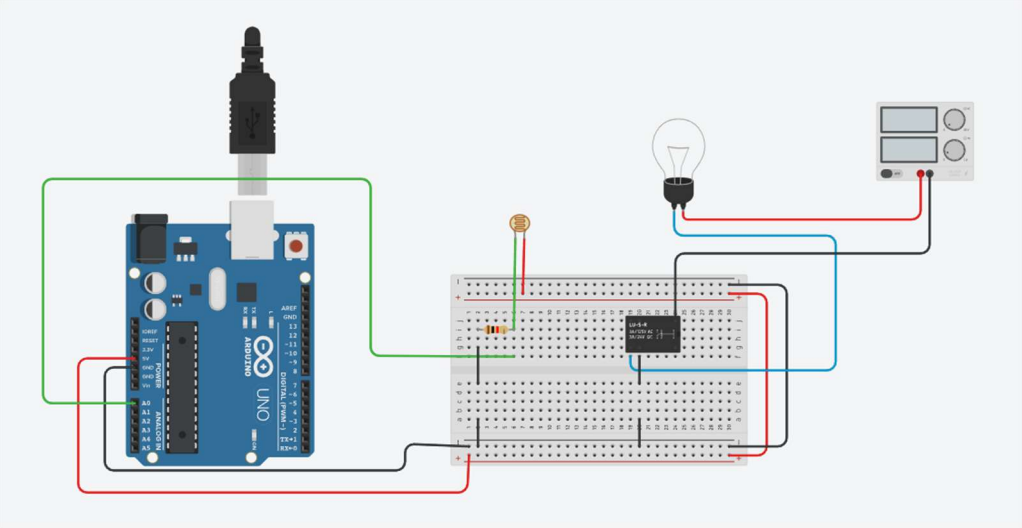
   The Arduino Uno R3 is the main microcontroller in this circuit. It acts as the brain of the setup, executing the programmed code to control the LED. It provides both power and signal to the circuit, making it an essential component for automation and control projects.
2. Small Breadboard

   The small breadboard is used for easy and temporary circuit connections without soldering. It allows the LED and other components to be connected securely while maintaining flexibility for modifications.
3. Jumper Cable:

   Used to make electrical connections between the Arduino, breadboard, and components.
4. The LED

   The LED (Light Emitting Diode) is the primary output component in this circuit. It emits light when current flows through it, and its blinking pattern is controlled by the Arduino using programmed delays.
5. **Power Supply**: Provides power to the Arduino and external components.
6. **Photoresistor (LDR)**: Senses light intensity and sends an analog signal to the Arduino.
7. **Relay (SPDT)**: Acts as a switch to control high-power devices like bulbs, fans, or motors based on light levels.

Circuit Design



Procedure to Design a Light-Controlled Circuit using LDR and Arduino on Tinkercad

Step 1: Add Components to the Workspace

1. Open Tinkercad Circuits and create a new project.
2. Search for Arduino Uno R3 and add it to the workspace.
3. Search for a small breadboard and place it next to the Arduino.
4. Search for a photoresistor (LDR) and position it on the breadboard.
5. Search for an LED and place it on the breadboard.
6. Search for a relay (SPDT) to control external power devices.
7. Use jumper cables to connect the components properly.

Step 2: Build the Circuit Connections

LDR (Photoresistor) Connections:

1. One terminal of the LDR → Connect to 5V on Arduino.
2. Other terminal of the LDR → Connect to Analog Pin A0 on Arduino.
3. 10kΩ resistor → Connect between the LDR's second terminal and GND (pull-down resistor).

LED Connections:

1. Anode (+) of LED → Connect to digital pin 9 on the Arduino.
2. Cathode (-) of LED → Connect to one side of a 220Ω resistor.
3. Other side of the resistor → Connect to GND.

Relay (SPDT) Connections:

1. Coil terminal 1 → Connect to digital pin 8 on Arduino.
2. Coil terminal 2 → Connect to GND.
3. Common (COM) terminal → Connect to external power source (+12V or AC phase, depending on the appliance used).
4. Normally Open (NO) terminal → Connect to the positive terminal of the external device (e.g., light bulb, fan, etc.).
5. Negative terminal of the external device → Connect to external power ground (or AC neutral).

Step 3: Code for Light-Dependent Operation

```
int photoResistorSensor = 0;
int relayPinOutput = 4;
void setup(){
        pinMode(A0,INPUT);
```

```
        pinMode(relayPinOutput,OUTPUT);
        Serial.begin(9600);
}
void loop(){
        int photoResistorSensor = analogRead(A0);
        Serial.println(photoResistorSensor);
        if(photoResistorSensor > 400){
                Serial.println("Sufficient Light Outside");
        digitalWrite(relayPinOutput,LOW);
   } else {
        Serial.println("In Sufficient Light Outside");
        digitalWrite(relayPinOutput,HIGH);
 }
}
```

Step 4: Simulate the Circuit

1. Click "Start Simulation" in Tinkercad.
2. Adjust the light intensity on the LDR (drag the light source in simulation).
3. Observe how the LED and relay switch ON in darkness and OFF in bright light.

Conclusion

This project demonstrates how to use an LDR (photoresistor) with an Arduino to detect light intensity and control a relay and LED accordingly. The circuit functions as an automatic light system, where the LED and relay turn ON in darkness and OFF in bright light. This is useful for street lights, automatic room lighting, and security systems. By simulating this in Tinkercad, we can test and optimize the circuit before implementing it in real hardware.