Experiment 2. Develop simple application - testing infrared sensor - IoT Applications - using Arduino.

Aim:

To design and test an application for detecting infrared signals using an IR sensor and Arduino, and to simulate the experiment online.

Apparatus Required

1. **Online Simulator**

   Components:

   - Arduino Uno
   - IR Receiver (TSOP1838 or similar)
   - Push Buttons (optional, to mimic remote control buttons)
   - LED (optional, for output indication)
   - Resistors (1 kΩ and 220 Ω)
   - Virtual Serial Monitor (for viewing results)
   - Breadboard (virtual)
   - Connecting wires (virtual)

Background Theory

An Infrared (IR) sensor detects infrared signals emitted by sources like an IR remote control. The IR receiver decodes the signal based on its frequency and converts it into electrical signals. These signals can be processed using an Arduino to perform specific tasks.
In IoT applications, IR sensors can be used for automation, remote control, and object detection.

Algorithm

1. **Initialize** the IR sensor and Serial Monitor in the Arduino program.
2. **Setup** the hardware connections:

   1. Connect the IR receiver to the Arduino.
   2. Connect an optional LED for visual feedback.

3. **Detect Signals**:
   1. Monitor the IR sensor for incoming signals.
   2. Decode and process the signal.
4. **Display Results**:
   1. Send the decoded signal data to the Serial Monitor.
   2. Optionally, blink an LED for specific signals.
5. **Terminate** the program.

Procedure

1. Hardware Setup in Wokwi:

1. Open **Wokwi.com** and create a new Arduino Uno project.
2. Add the following components from the library:

    - ✓ Arduino Uno
    - ✓ IR Receiver (e.g., TSOP1838)
    - ✓ Optional: LED and Resistors.

3. Connect the components:
    - IR receiver:

        - ✓ Signal pin → Arduino digital pin 2
        - ✓ VCC pin → 5V
        - ✓ GND pin → GND

    - Optional LED:

        - ✓ Anode (long leg) → Arduino digital pin 13 (with a 220 Ω resistor)
        - ✓ Cathode (short leg) → GND.

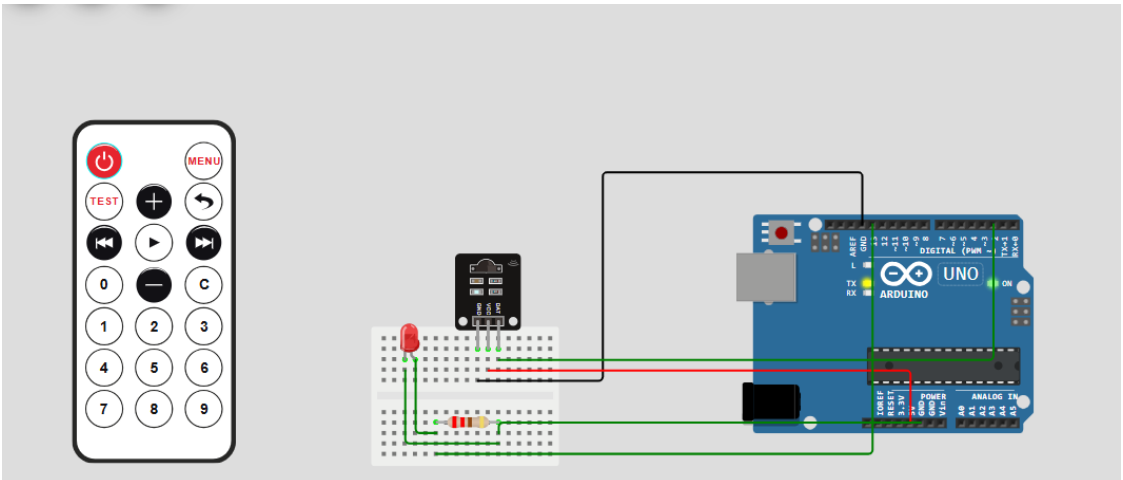
4. Save the project.

2. Software Setup:

1. Write the Arduino code to process IR signals.
2. Use the `IRremote` library for decoding signals.

3. Simulation:

1. Upload the program to the virtual Arduino in Wokwi.
2. Start the simulation by clicking the green **"Play"** button.
3. Open the Serial Monitor to observe the decoded signal data.
4. Use the virtual IR remote to send signals to the IR receiver.



Program

```
#include <IRremote.h> // Define pin for IR receiver

const int RECV_PIN = 2;

IRrecv irrecv(RECV_PIN);

decode_results results;

void setup() {

Serial.begin(9600);  // Initialize Serial Monitor

  irrecv.enableIRIn(); // Start IR receiver

  pinMode(13, OUTPUT); // Optional: LED pin

  Serial.println("IR Receiver is ready...");

}

void loop() {

  if (irrecv.decode(&results)) {

    // Print the received signal value

    Serial.print("IR Signal Received: ");

    Serial.println(results.value, HEX);         // Optional: Blink LED for visual feedback

    digitalWrite(13, HIGH);

    delay(200);

    digitalWrite(13, LOW);

    irrecv.resume(); // Receive the next value

  }

}
```

Result

1. Successfully detected and decoded IR signals using an IR sensor and Arduino in Wokwi.
2. The Serial Monitor displayed the signal value when an IR remote was used.
3. The LED blinked whenever a signal was detected.

Pre-Lab Questions

1. What is an Infrared (IR) sensor? Explain its working principle.
2. What are the typical applications of IR sensors in IoT?
3. Why do we use an IR receiver module like TSOP1838 with Arduino?
4. What is the function of the IRremote library in the Arduino program?
5. How does the Arduino process signals received from an IR sensor?
6. What is the role of the Serial Monitor in this experiment?
7. What are the advantages of simulating an IR sensor experiment in Wokwi compared to using physical hardware?

Post-Lab Questions

1. What signal format is used by the IR remote control, and how was it decoded in this experiment?
2. What are the main components required to interface an IR sensor with Arduino, and how are they connected?
3. Explain how the irrecv.decode() function works in the program.
4. What modifications would you make to the program to perform specific tasks based on the IR signal received?
5. What challenges might arise when using IR sensors in real-world IoT applications, and how can they be addressed?
6. Why is the LED used as a visual indicator in this experiment, and how does it help in debugging?
7. How can this experiment be extended to control devices like fans, lights, or other appliances using an IR remote?
8. What are some limitations of IR-based communication compared to other wireless communication methods (e.g., Bluetooth, Wi-Fi)?
9. Discuss how IR technology can be integrated into a complete IoT ecosystem.