Interfacing PIR Sensor with Arduino

Aim

1.To interface a Passive Infrared (PIR) sensor with an Arduino and detect motion.

2. To simulate a **PIR sensor with Arduino** in **Tinkercad** and detect motion to turn on an LED.

Components Required

1. Arduino Uno
2. PIR Sensor (HC-SR501)
3. LED or Buzzer (for output indication)
4. Resistors (330Ω if using LED)
5. Connecting Wires
6. Breadboard
7. USB cable

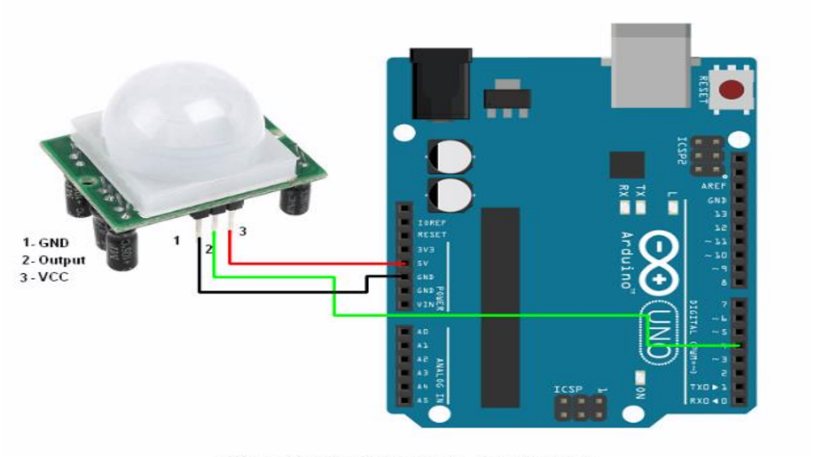
Background Theory

A **PIR (Passive Infrared) sensor** detects motion by sensing infrared radiation changes in its field of view. It has a pyroelectric sensor that detects heat (infrared radiation) emitted by objects like humans and animals. When a motion is detected, the sensor outputs a **HIGH** signal, which can be used to trigger an alarm, light, or other actions.

Hardware Connection

- **PIR Sensor Pins:**
  - VCC → 5V (Arduino)
  - GND → GND (Arduino)
  - OUT → 4th digital pin (e.g., D2 on Arduino)
- **LED/Buzzer Connection:**
  - Connect the positive leg of the LED (Anode) to **D3** (or any digital pin).
  - Connect the negative leg (Cathode) to GND via a **330Ω resistor**.
  - If using a buzzer, connect it similarly.

Connection Diagram of PIR Sensor with Arduino



Interfacing PIR Sensor with Arduino UNO

Code:

```
const  int  PIR_SENSOR_OUTPUT_PIN  =  4; /* PIR sensor O/P pin */
int  warm_up;

void setup() {
  pinMode(PIR_SENSOR_OUTPUT_PIN, INPUT);
  Serial.begin(9600);    /* Define baud rate for serial communication */
  delay(20000);    /* Power On Warm Up Delay */
}

void loop() {
  int  sensor_output;
  sensor_output = digitalRead(PIR_SENSOR_OUTPUT_PIN);
  if( sensor_output == LOW )
  {
    if( warm_up == 1 )
     {
      Serial.print("Warming Up\n\n");
      warm_up = 0;
      delay(2000);
    }
    Serial.print("No object in sight\n\n");
    delay(1000);
  }
  else
  {
    Serial.print("Object detected\n\n");
    warm_up = 1;
    delay(1000);
  }

}
```

Steps

1. **Connect the PIR sensor to Arduino as per the circuit diagram.**
2. **Upload the following code to Arduino.**
3. **Wave your hand in front of the PIR sensor.**
4. **If motion is detected, the LED or buzzer should activate.**
5. **Observe the output in the Serial Monitor.**

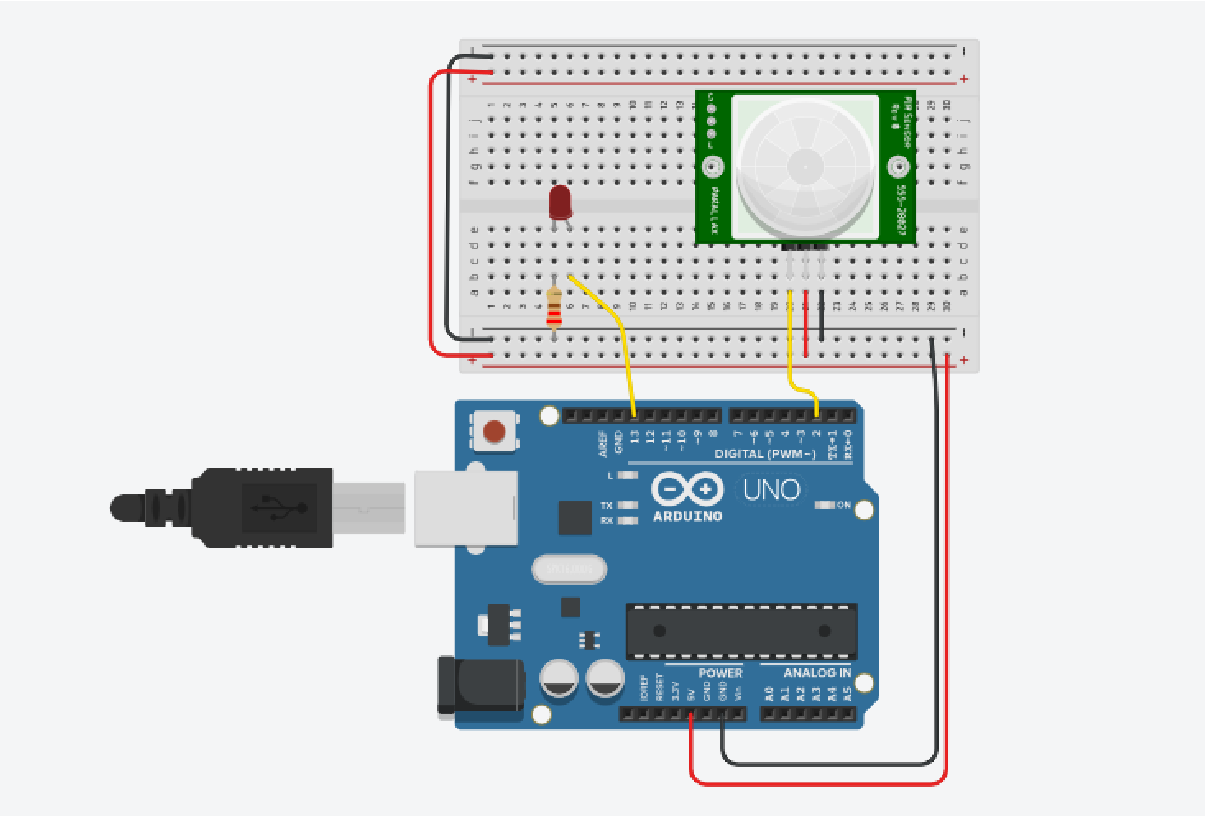
Hardware Connection in Tinkercad

- **PIR Sensor Pins:**
  - **VCC → 5V** (Arduino)
  - **GND → GND** (Arduino)
  - **OUT → Digital Pin 2** (Arduino)
- **LED Connection:**
  - **Anode (+) → Digital Pin 3** (Arduino) (via 330Ω resistor)
  - **Cathode (-) → GND**

Steps to Simulate in Tinkercad

1. **Open Tinkercad Circuits** and create a new circuit.
2. **Drag and drop** the required components (Arduino Uno, PIR sensor, LED, resistor, and wires).
3. **Make the connections** as per the circuit diagram.
4. **Write and upload the Arduino code** in Tinkercad's code editor.
5. **Start the simulation** and observe the LED turning ON when motion is detected

Hardware Connection



Code:

```
// C++ code
//
int sensorState = 0;

void setup()
{
  pinMode(2, INPUT);
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop()
{
  // read the state of the sensor/digital input
  sensorState = digitalRead(2);
  // check if sensor pin is HIGH. if it is, set the
  // LED on.
  if (sensorState == HIGH) {
    digitalWrite(LED_BUILTIN, HIGH);
  } else {
    digitalWrite(LED_BUILTIN, LOW);
  }
  delay(10); // Delay a little bit to improve simulation performance
}
```

Conclusion

The PIR sensor was successfully interfaced with Arduino, and motion was detected, triggering an LED or buzzer. This setup can be used for security systems, automatic lighting, and smart home applications.

The PIR sensor was successfully simulated in **Tinkercad**, detecting motion and turning on an LED. This setup can be further modified for smart security systems, automatic lighting, and IoT applications.