Experiment No-1

Subject-Computer Laboratory II-Industrial Internet of Things

Class-BE AI &DS

Aim: Write an Arduino/Raspberry pi program for interfacing with PIR Sensor

Software Required: Tinkercad.

Theory:

PIR sensors allow you to sense motion. They are used to detect whether a human has moved in or out of the sensor range. They are commonly found in appliances and gadgets used at home or for businesses. **They are often referred to as PIR, "Passive Infrared", "Pyroelectric", or "IR motion" sensors.**

Following are the advantages of PIR Sensors –

- Small in size
- Wide lens range
- Easy to interface
- Inexpensive
- Low-power
- Easy to use
- Do not wear out

Procedure:

1. Assemble all the parts by following the schematics below.

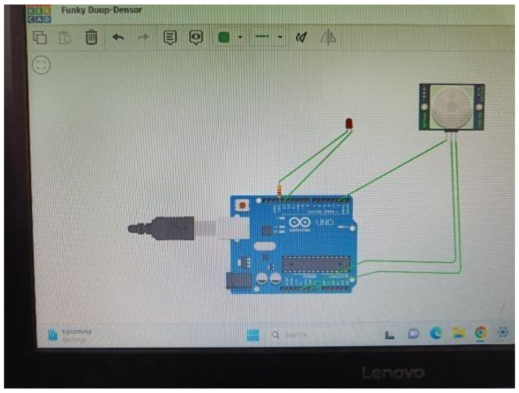

Fig.1-Interfacing of PIR sensor with Arduino

2. Connections:

1. Connect the ground pin of PIR sensor to the ground (GND) pin of Arduino.

2. Connect the power pin to PIR sensor to the 5V pin of Arduino.

3. Connect the signal pin of Arduino to pin 2 of Arduino as shown in Fig.1.

3. Copy the code by clicking on the code button on RHS and then in Blocks button there is a drop-down menu from which select the text option, click on continue and paste the code there. Then click on start simulation button.

Code Explanation:

1. First declares two integer variables: `pirPin` and `ledPin`. These variables will be used to store the pin numbers to which the PIR sensor output and LED are connected, respectively.
2. Initialize serial communication for debugging.
3. The `setup()` function is a special function in Arduino that is executed only once when the Arduino board is powered up or reset.
4. In this setup function: `pinMode(pirPin, INPUT)` sets the `pirPin` (connected to the PIR sensor) as an input pin, indicating that it will be used to read data from the sensor and `pinMode(ledPin, OUTPUT)` sets the `ledPin` (connected to an LED) as an output pin, indicating that it will be used to control the LED.
5. `Serial.begin(9600)` initializes serial communication at a baud rate of 9600 bits per second. This is used for debugging purposes to send messages to your computer for monitoring.
6. The `loop()` function is where the main code execution takes place. It runs in a continuous loop after the `setup()` function is executed.
   In this loop:
7. In this loop: `int motionState = digitalRead(pirPin)` reads the state of the PIR sensor. If the sensor detects motion, it will read `HIGH`, and if there's no motion, it will read `LOW`.
8. The code then uses an `if` statement to check whether `motionState` is `HIGH`, indicating that motion has been detected.
9. If motion is detected, `digitalWrite(ledPin, HIGH)` turns on the LED by setting the `ledPin` to `HIGH`.

10. `Serial.println("Motion detected!")` sends a message to the serial monitor indicating that motion has been detected.
11. `delay(1000)` causes the program to pause for one second before continuing. This provides a delay between LED on and off states.
12. If no motion is detected (`motionState` is `LOW`), the LED is turned off using `digitalWrite(ledPin, LOW)` and a message is printed to the serial monitor indicating that no motion was detected.

Conclusion: This code snippet demonstrates the basic operation of interfacing a PIR sensor with an Arduino. When the PIR sensor detects motion, it turns on an LED and sends a message to the serial monitor, and when no motion is detected, the LED is turned off.

CODE FOR PIR Sensor interfacing with Arduino

```
// PIR Sensor interfacing with Arduino
int pirPin = 2;  // PIR sensor output pin
int ledPin = 13; // LED pin

void setup() {
  pinMode(pirPin, INPUT);   // Set PIR pin as input
  pinMode(ledPin, OUTPUT);  // Set LED pin as output

  Serial.begin(9600);  // Initialize serial communication for debugging
}
void loop() {
  int motionState = digitalRead(pirPin);  // Read PIR sensor state

  if (motionState == HIGH) {
    digitalWrite(ledPin, HIGH);  // Turn on LED
    Serial.println("Motion detected!");
    delay(1000);  // Delay for one second
  } else {
    digitalWrite(ledPin, LOW);   // Turn off LED
    Serial.println("No motion detected.");
  }
}
```

OUTPUT

When the PIR sensor detects motion, it turns on an LED and sends a message to the serial monitor, and when no motion is detected, the LED is turned off.

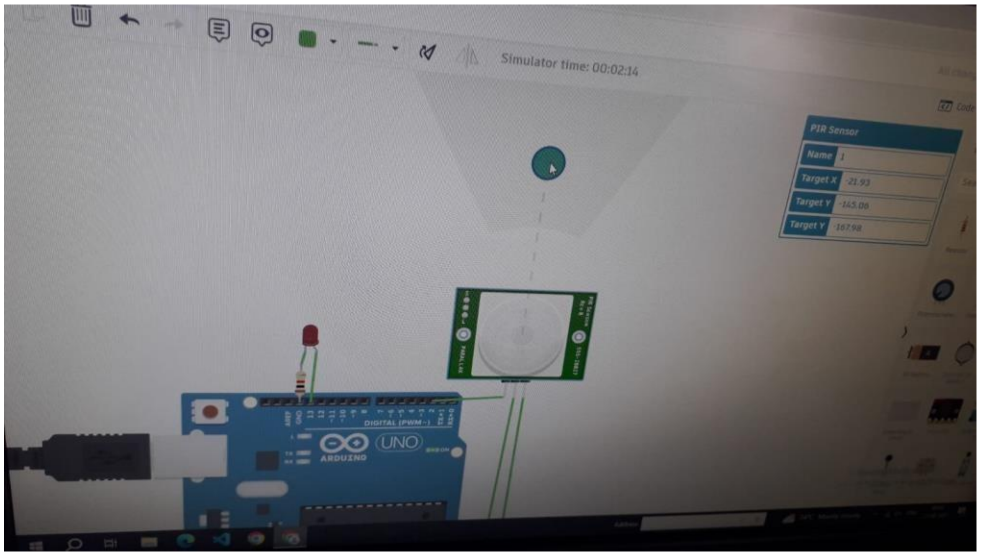

Fig.2-When there is no motion the LED is in OFF position

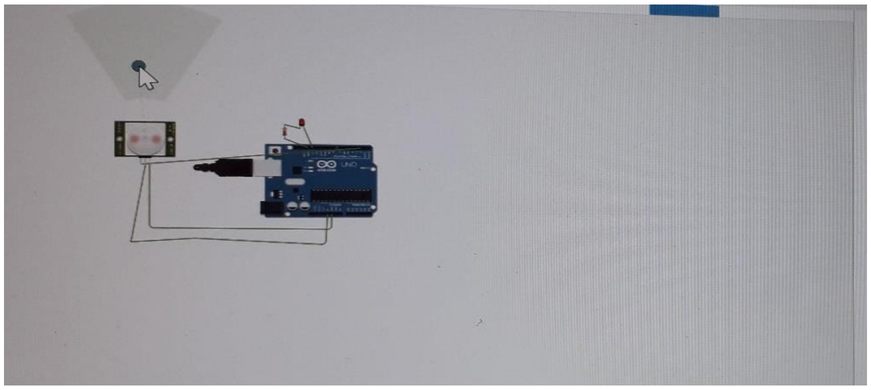

Fig.3-When there is motion the LED is in ON position