Microcontroller-based Industrial Applications – Project

Real-Time Occupancy Detection

Varsha R

Vellore Institute of Technology, Vellore

Contact Info: +91 8531919717

mailto:varsha.r2021@vitstudent.ac.in

Project Title: Real-Time Occupancy Detection

Problem Statement

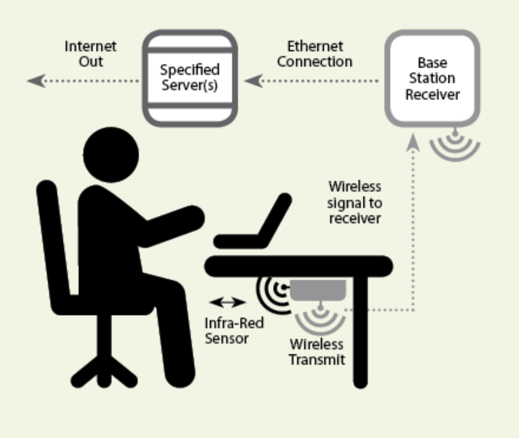
Develop a simple prototype of a real-time occupancy detection system that may be utilised in commercial and residential buildings to increase energy efficiency and save expenses. The device must be accurate, dependable, simple to set up and operate, inexpensive, and low-power. It should also be designed to minimize privacy concerns, be robust to environmental factors, and be scalable to support large buildings and multiple devices.

Scope of solution

A PIR sensor and a microcontroller might be used to build a basic prototype occupancy detection system. The PIR sensor would detect the presence of people in an area, and the microcontroller would analyse the sensor's data and switch off the lights when no one is present. A battery or an AC adaptor might power the system.

This prototype system would be quite cheap and simple to build, yet it might save a lot of energy by shutting off the lights while no one is present. The device might be used in several environments, including offices, residences, and schools.

The scope of solution for a prototype occupancy detection system can be tailored to meet the specific needs of the user. For example, A more comprehensive system may interact with current building management systems or give additional features like people counting or motion tracking.



Required components

1. Relay module
2. Arduino IDE
3. Bread board
4. Arduino Uno R3
5. PIR sensor
6. Jumper cables
7. LEDs

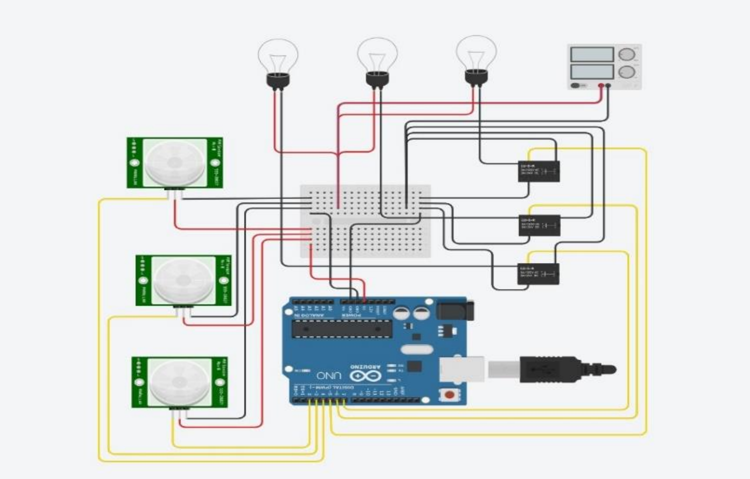
Methodology

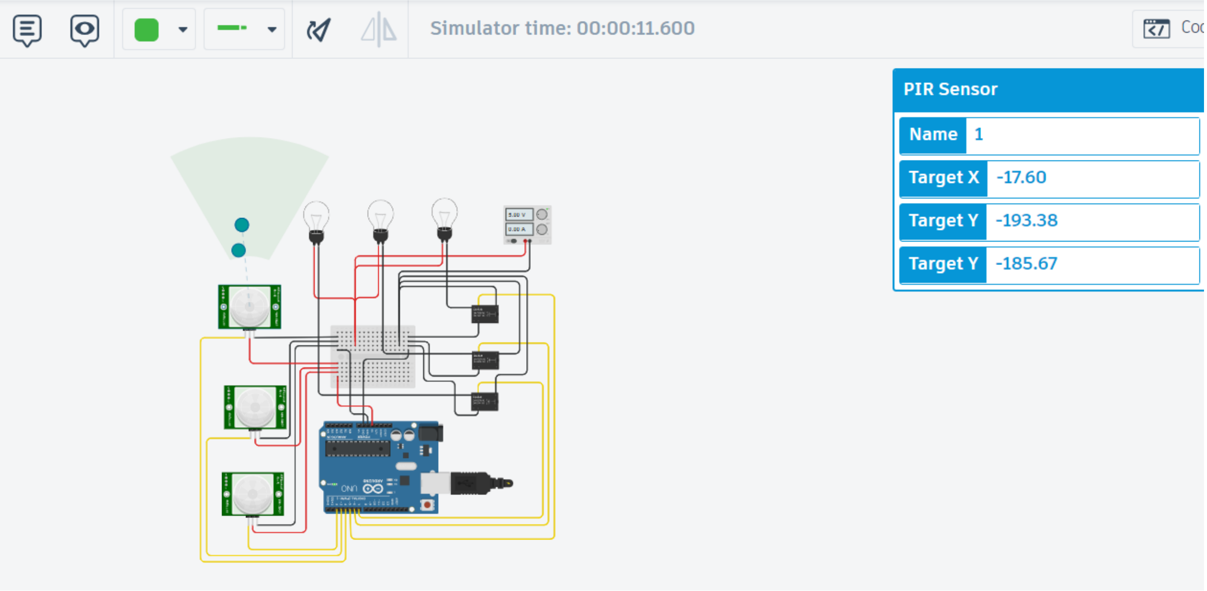
1. **Preparing Arduino board.** Make sure that you have the Arduino IDE installed on your computer. Then, connect your Arduino board to your computer using a USB cable.
2. **Connect the PIR sensors and relays to the Arduino board**. Make sure that the PIR sensors are connected to the analog pins and the relays to the digital pins.
3. **Upload the code to your Arduino board.** Open the Arduino IDE and click on the "File" menu. Then, click on "Open" and select the code file to be uploaded. Once the code file is open, click on the "Upload" button to upload the code to Arduino board.
4. **Connect the devices that you want to control to the relays**. Connect the devices that you want to control to the relays. For example, if you want to control a light bulb, you would connect the light bulb to the relay.
5. **Test your circuit**. Once everything is connected, test your circuit to make sure that it is working properly. Turn on the power to your circuit and then move in front of the PIR sensors. The relays should turn on the devices that they are connected to.

The following steps have been followed to implement the circuit

Simulated circuit

The circuit has been simulated using TinkerCad.





Simulator time: 00:00:11.600

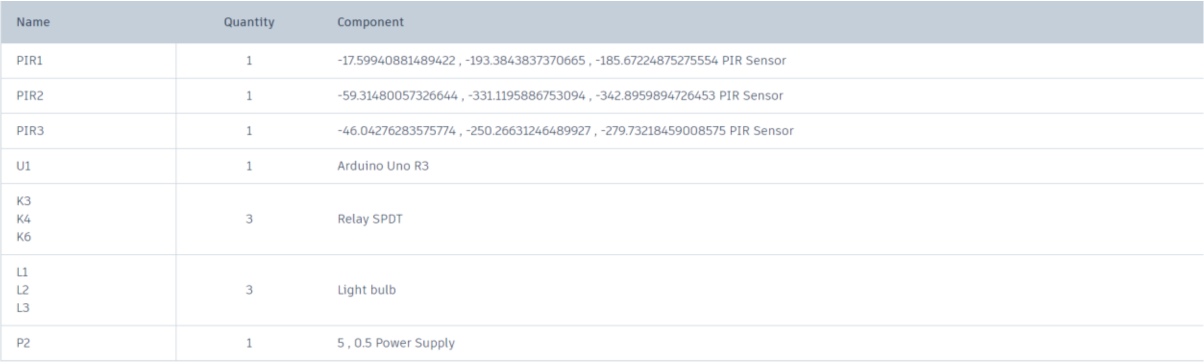
PIR Sensor

Name 1

Target X -17.60

Target Y -193.38

Target Y -185.67



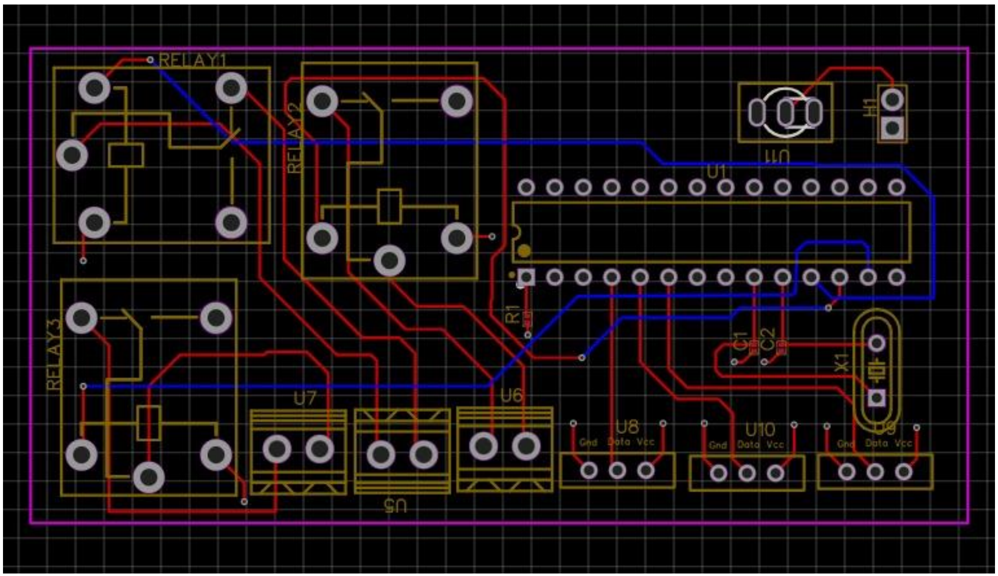
Name	Quantity	Component
PIR1	1	-17.59940881489422 , -193.3843837370665 , -185.67224875275554 PIR Sensor
PIR2	1	-59.31480057326644 , -331.1195886753094 , -342.8959894726453 PIR Sensor
PIR3	1	-46.04276283575774 , -250.26631246489927 , -279.73218459008575 PIR Sensor
U1	1	Arduino Uno R3
K3 K4 K6	3	Relay SPDT
L1 L2 L3	3	Light bulb
P2	1	5 , 0.5 Power Supply

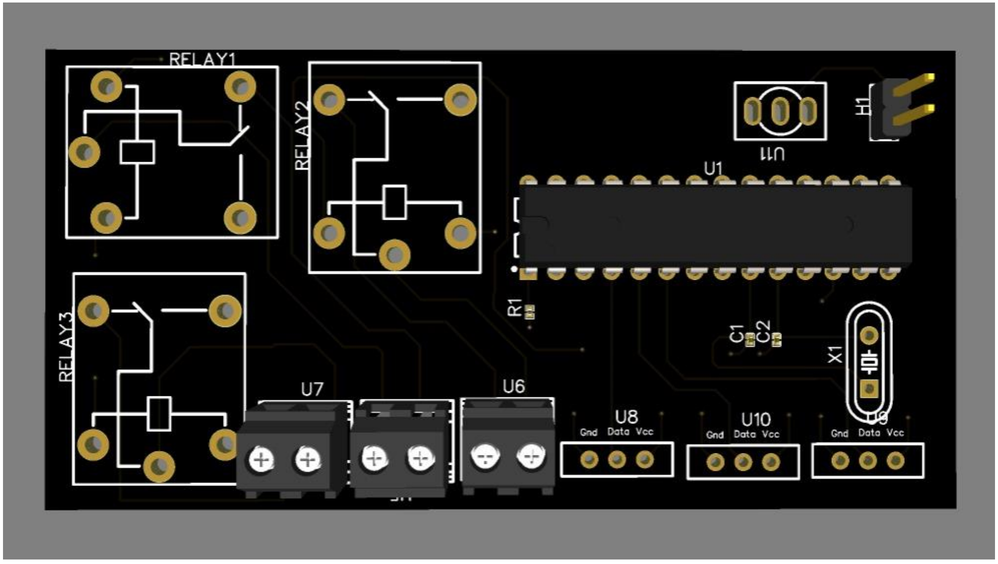
Demo video

https://drive.google.com/file/d/1nt3_bY2LIfv-w9Tz6tfnGU5IT4buez5P/view?usp=sharing

Gerber file

https://drive.google.com/file/d/1lpsjo4rUb5RfGcjN_X5tIcixYgeHbK6H/view?usp=sharing





Code for the solution

```
int pir_1 = 2;

int pir_2 = 3;

int pir_3 = 4;


int relay_1 = 5;

int relay_2 = 6;

int relay_3 = 7;


int pirState_1 = LOW;
int pirState_2 = LOW;
int pirState_3 = LOW;

int val_1 = 0;
int val_2 = 0;
int val_3 = 0;

void setup() {

  pinMode(relay_1, OUTPUT);
  pinMode(relay_2, OUTPUT);
  pinMode(relay_3, OUTPUT);

  pinMode(pir_1, INPUT);
  pinMode(pir_2, INPUT);
  pinMode(pir_3, INPUT);

  Serial.begin(9600);
}

void loop(){
  val_1 = digitalRead(pir_1);
  val_2 = digitalRead(pir_2);
  val_3 = digitalRead(pir_3);

  if (val_1 == HIGH)
  {
    digitalWrite(relay_1, HIGH);

    if (pirState_1 == LOW)
       {
      Serial.println("Motion detected!");
      pirState_1 = HIGH;
    }
  }
  else
```

```
{
  digitalWrite(relay_1, LOW);

  if (pirState_1 == HIGH)
     {
    Serial.println("Motion ended!");
    pirState_1 = LOW;
  }
}
if (val_2 == HIGH)
{
  digitalWrite(relay_2, HIGH);

  if (pirState_2 == LOW)
     {
    Serial.println("Motion detected!");
    pirState_2 = HIGH;
  }
}
else
{
  digitalWrite(relay_2, LOW);

  if (pirState_2 == HIGH)
     {
    Serial.println("Motion ended!");
    pirState_2 = LOW;
  }
}
if (val_3 == HIGH)
{
  digitalWrite(relay_3, HIGH);

  if (pirState_3 == LOW)
     {
    Serial.println("Motion detected!");
```

```
      pirState_3 = HIGH;
    }
  }
  else
  {
    digitalWrite(relay_3, LOW);

    if (pirState_3 == HIGH)
       {
      Serial.println("Motion ended!");
      pirState_3 = LOW;
    }
  }
}
```

The code uses three PIR sensors to detect motion, and three relays to control three different devices.

The code works as follows:

- In the setup() function, the code sets the relay pins as outputs, and the PIR sensor pins as inputs. It also starts serial communication at 9600 baud.
- In the loop() function, the code reads the state of each PIR sensor. If a sensor detects motion, the code turns on the corresponding relay. If a sensor no longer detects motion, the code turns off the corresponding relay.
- The code also prints a message to the serial monitor when motion is detected or ended.

Conclusion

The development of a simple prototype of a real-time occupancy detection system is a fascinating field of research that has the potential to revolutionize how we control our energy use and improve our quality of life. A system of this type is developed to be accurate, reliable, user-friendly, cost-effective, and secure, making it a feasible solution for a wide range of applications, from modest houses to huge commercial buildings.

Consider a world in which lights, heaters, fans, air conditioners, and other appliances change themselves automatically based on our presence in a location. This would not only save us money on our energy bills, but it would also have a positive impact on the environment. I believe that the development of a simple prototype of a real-time occupancy detection system is a worthwhile endeavour.

GitHub link: https://github.com/VarshaRamkumar18/varsha_48