Room Vacancy Detection IoT Project

About the Project

This project is an IoT-based room vacancy detection system that uses an ultrasonic sensor and a PIR sensor to determine the occupancy and distance of objects in a room. The system provides visual indicators using red and green LEDs, and a buzzer alerts when the object is detected at a close range. The ultrasonic sensor measures the distance to an object in the room, while the PIR sensor detects human presence. Depending on the room's occupancy and distance readings, the LEDs and buzzer react accordingly, indicating the vacancy status.

Components Used

- Arduino Uno
- Ultrasonic Sensor (HC-SR04)
- PIR Sensor
- Red LED
- Green LED
- Buzzer
- Resistors (220 ohms for LEDs)
- Jumper Wires

Connections

1. **Ultrasonic Sensor (HC-SR04)**:
   - Trig → Digital Pin 3
   - Echo → Digital Pin 4
   - VCC → 5V
   - GND → GND

2. **PIR Sensor**:
   - OUT → Digital Pin 6
   - VCC → 5V
   - GND → GND

3. **Red LED**:
   - Anode (long leg) → Digital Pin 8 (via a 220-ohm resistor)
   - Cathode (short leg) → GND

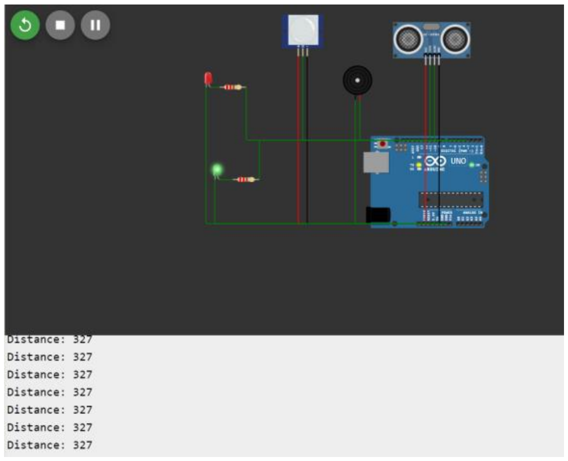
4. **Green LED**:
   - Anode (long leg) → Digital Pin 9 (via a 220-ohm resistor)
   - Cathode (short leg) → GND

5. **Buzzer**:
   - Positive lead → Digital Pin 7
   - Negative lead → GND

Code for Room Vacancy Detection

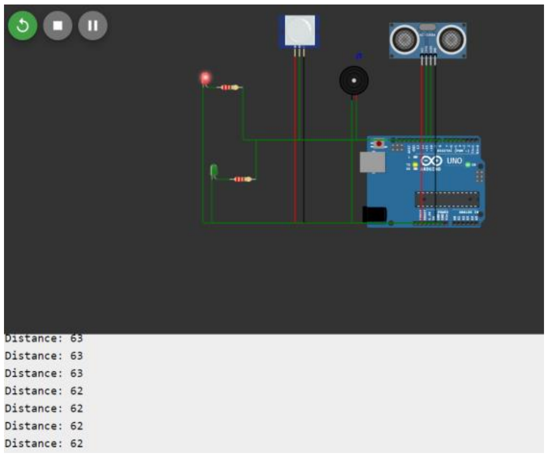
```
// Pin Definitions
const int trigPin = 3;
const int echoPin = 4;
const int pirPin = 6; // PIR sensor pin
const int redLEDPin = 8; // Red LED pin
const int greenLEDPin = 9; // Green LED pin
const int buzzerPin = 7; // Buzzer pin

// Variables
long duration;
int distance;
bool isRoomOccupied = false;

void setup() {
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(pirPin, INPUT);
  pinMode(redLEDPin, OUTPUT);
  pinMode(greenLEDPin, OUTPUT);
  pinMode(buzzerPin, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  // Reading PIR Sensor
  int pirState = digitalRead(pirPin);  // PIR output

  if (pirState == HIGH) {
    isRoomOccupied = true; // Occupancy detected
  } else {
    isRoomOccupied = false; // No occupancy
  }

  // Ultrasonic Sensor Logic
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  duration = pulseIn(echoPin, HIGH);
  distance = duration * 0.034 / 2; // Calculating the distance in cm
  Serial.print('Distance: ');
  Serial.println(distance);

  // LED and Buzzer Control Logic based on distance and PIR
  if (isRoomOccupied) {
    if (distance < 200) {
      blinkRedLED(); // Blink red LED
      tone(buzzerPin, 1000); // Turn on buzzer
    }
    else if (distance >= 200 && distance <= 300) {
      blinkGreenLED(); // Blink green LED
      noTone(buzzerPin); // Turn off buzzer
      digitalWrite(redLEDPin, LOW); // Ensure red LED is off
    }
    else if (distance > 300) {
      digitalWrite(greenLEDPin, HIGH); // Green LED ON
      noTone(buzzerPin); // Turn off buzzer
      digitalWrite(redLEDPin, LOW); // Ensure red LED is off
    }
  } else {
    digitalWrite(greenLEDPin, LOW);
    digitalWrite(redLEDPin, LOW);
    noTone(buzzerPin);  // Turn off the buzzer
  }

  delay(100);
}

void blinkRedLED() {
  digitalWrite(redLEDPin, HIGH);
  delay(250);
  digitalWrite(redLEDPin, LOW);
  delay(250);
}

void blinkGreenLED() {
  digitalWrite(greenLEDPin, HIGH);
  delay(500);
  digitalWrite(greenLEDPin, LOW);
  delay(500);
}
```

Screen Shots from “Wokwi.com (Online IoT simulator)”



```
Distance: 327
Distance: 327
Distance: 327
Distance: 327
Distance: 327
Distance: 327
Distance: 327
```

Dig: Distance above 300 & green light



```
Distance: 63
Distance: 63
Distance: 63
Distance: 62
Distance: 62
Distance: 62
Distance: 62
```

Dig: Distance below 200 and buzzer & red light