Occupancy-Based Smart Fan Control System

G V Krishna Kumar
CH.EN.U4AIE22012

Guruprasath M R
CH.EN.U4AIE22015

Shree Prasad M
CH.EN.U4AIE22050

Sudeesh Kumar V
CH.EN.U4AIE22059

Tharun Kaarthik G K
CH.EN.U4AIE22062

***Abstract*—This project presents an Internet of Things (IoT) based smart fan control system. The system utilizes a combination of infrared (IR) sensors to detect human presence and dynamically adjust fan operation in a room. Traditionally, fans operate on manual controls which often result in energy consumption when a room is unoccupied. Addressing this, we use a device with an Infrared Sensor (IR) that would be placed in a specific manner. Considering the scenario where there are only a few people occupying the room, IR sensors will be positioned strategically under each fan's coverage area. When an occupant is sensed within a particular fan's zone, a dedicated (IR) sensor detects the signal which transmits to the corresponding relay, activating the fan. This personalized approach ensures cool air circulation only in areas where it is actively needed. This project demonstrates a practical application of IoT principles for developing a smart fan control system. By utilizing the combination of IR sensors, an 8-Relay Module, and with the help of Arduino Uno for control & communication, the system controls fans according to the occupant's position and manages to save some energy when there are no occupants in the room.**

Git Hub Link

I. AIM

The goal of this project is to create a smart fan control system based on the Internet of Things that allows for optimal energy consumption in a commercial or residential room. At present, fans tend to operate based on user instructions, which means they are being operated when the room is unoccupied. This project aims to introduce a system that would address the described inefficiency. The offered system contains six infrared sensors - four of them will be placed in the room on four distinct sides, and the remaining two will measure the fan zones. If they detect people in a certain amount of time, the fan would be turned on through the relays and Arduino Uno microcontroller. This way, the fans would operate exclusively at necessary times and locations.

II. COMPONENTS

A. *Arduino Uno*

The microcontroller board Arduino Uno is an example of tools utilized in the domain of technology innovation that support simplicity and availability. The core of this board is the ATmega328P microcontroller. However, this board complements a robust platform rich in digital and analog input-output pins as well as supporting a smooth interaction with a wide range of sensors, actuators, and other peripheral devices. It is based on an easy-to-understand programming language Wiring with a user-friendly integrated development environment. Thus, allowing people of all specialties to dive into electronics and embedded systems.

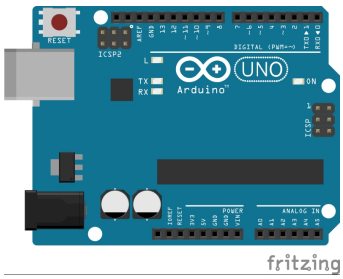

Fig. 1. Arduino

B. *IR Sensor*

An infrared sensor is an electronic device that detects infrared radiation, which is an electromagnetic wave with a longer wavelength than visible light. IR sensors can be passive and active. PIR sensors detect infrared radiation from objects around them. Active IR sensors detect objects based on the amount of infrared radiation they reflect. IR sensors are commonly used for many applications, including motion detection, object counting, and night vision because they can operate in low-light or no-light conditions.

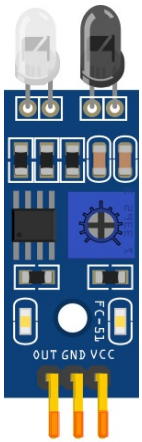

Fig. 2. IR sensor

C. 8 Relay Module

An 8-channel relay module is a small unit designed to manage eight individual circuits with a low-power control signal. Each of the eight channels has a relay, which is essentially a mechanical switch that can be turned on or off by the control signal. Usually, the control circuit is isolated from the high-power loads, providing additional safety, and enabling microcontrollers or other low-power units to control more significant appliances or systems.

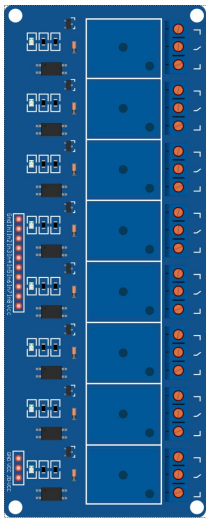

Fig. 3. Relay

D. USB Type-A to Type-B cable

The USB Type-A to Type-B cable is a basic element used to link a computer with a range of peripheral apparatuses. The former is a flat rectangular slot used in computers and laptops . A Type-B connector is a square-shaped orifice often found in side printers, scanners, and other devices, with two of its corners having an angle. The USB Type-A to Type-B cable transmits data and power, enabling it to be used to accomplish several purposes while using a device.

E. Breadboard

Breadboards refer to perforated plastic boards used to make temporary electronic circuits. The components' legs, which follow spacing rules for compatibility with different breadboard types, are placed into the holes. Connections are formed via metal strips underneath the board, which lack soldering, hence making it easy to insert components and remove and adjust them, especially for electronics beginners for learning and experimenting.

F. Connecting Wires

The connecting wires are basic requirements to make up an electrical circuit. They allow a live connection of the current to flow from one end to the other using the various components. Generally, connecting wires are made of copper due to their receptivity but may be coated to prevent unnecessary short and provide safety. The gauge of the wire should be chosen carefully enough to allow the anticipated current flow without developing too much resistance or heat.

G. Light-Emitting Diode

A light-emitting diode is a semiconductor-based device working according to the phenomenon of electroluminescence. The light is produced when a current passes through the LED, which causes electrons to recombine with the holes in the semiconductor material, releasing a quantum of energy as photons. The color of the light emitted depends on the band gap of the semiconductor materials. Due to high energy efficiency, long life span, and small size, the LED is widely used as a light source in modern electronic, opto-electronic, and solid-state lightning.

H. Resistor

A resistor refers to a primary passive electronic component used to add electrical resistance to a circuit. The latter limitation impedes the flow of current and is quantified in ohms. Resistors are used to stabilize the flow of current, protector or 'voltage divider,' encompass additional active elements, and terminate transmission lines in a given electronic circuit. There are various fixed resistances, and numerous compositions are obtained for distinct power levels.

III. BLOCK DIAGRAM

The circuit shown in Fig. 4 involves two Arduino Uno boards, seven sensors, and a 9V battery. One Arduino acts as the data collection hub. All eight sensors are connected to this board, feeding the information to it. Since one Arduino is linked to all the sensors and gathers sensor data, this data is then transmitted (through serial asynchronous communication) to the other Arduino, which is connected to the battery. The second Arduino is responsible for showing output based on the received sensor data.

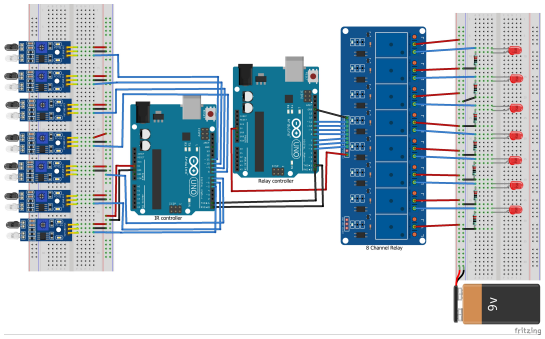

Fig. 4. Circuit Design

IV. PROTOTYPE

Figure 5 depicts the system prototype utilizing two Arduino Uno microcontrollers interconnected with breadboards and an 8-channel relay module. One breadboard houses a set of infrared (IR) sensors, while the other breadboard

accommodates light-emitting diodes (LEDs) and a power source (battery). The relay module, which connects with one Arduino Uno, is connected to the breadboard with LEDs. The IR sensors are wired to the other Arduino Uno.

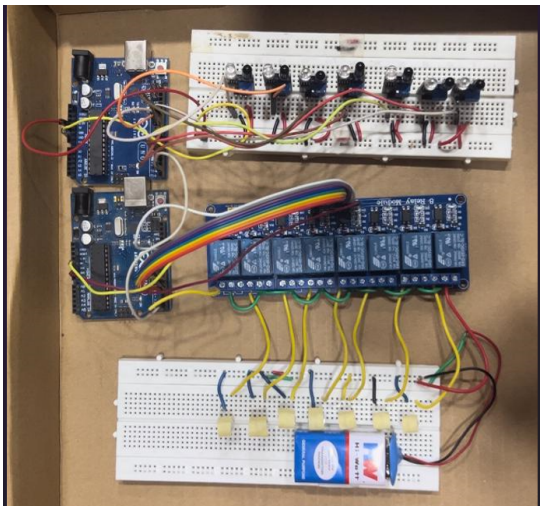

Fig. 5. Layout of the structure - Prototype

V. WORKING INSTANCES

*1) **Perception Layer**:* The IR-controller Arduino utilizes seven infrared (IR) sensors connected to digital pins 5 to 11. and PinMode is set as INPUT, These sensors detect incoming IR signals.

*2) **Signal Processing**:* The IR-controller Arduino continuously scans the sensor inputs. Every 125 milliseconds, it combines the current status of all seven relays into a single-byte signal. where, Each bit in the byte corresponds to a specific relay (bit 0 for relay connected to pin 5, and so on).

*3) **Serial Communication**:* The IR-controller Arduino transmits the generated byte signal to the relay-controller Arduino using serial asynchronous communication. here Tx and RX pins of the IR-Controller Arduino are connected to the RX and TX of Relay-Controller Arduino

*4) **Relay Control Unit**:* The relay-controller Arduino remains inactive until communication is established with the IR-controller Arduino. Once communication is initiated, it waits for 125 milliseconds to receive a byte signal.

*5) **Relay Actuation**:* The received byte from the IR-controller Arduino contains the control information for the relays. The first seven bits of the byte represent the desired state (on or off) of the relays connected to digital pins 5 to 11 on the relay-controller Arduino. If a specific bit is set to 1, the corresponding relay driver on the relay-controller Arduino activates the associated relay.

This IR controller system, comprised of two collaborating Arduino boards, bridges the gap between an IR remote control and various controllable devices. The IR-controller Arduino acts as a perceptive unit, continuously monitoring seven infrared sensors for incoming signals. It simultaneously tracks the state (on or off) of connected relays. Every 125 milliseconds, this information is cleverly combined. The state of each relay is encoded as a single bit within a byte signal, with a specific bit position corresponding to a particular relay. This byte signal is then transmitted serially to the relay-controller Arduino. The relay-controller remains inactive until communication is established. Once connected, it waits for the synchronized byte signal and interprets it. The first seven bits of this byte dictate the desired on/off state for the relays managed by the relay-controller Arduino. If a specific bit is set to 1, it triggers the corresponding relay driver to activate its associated relay. In essence, this system transforms IR commands from a remote into targeted relay control, enabling you to automate various devices based on simple remote control interactions.

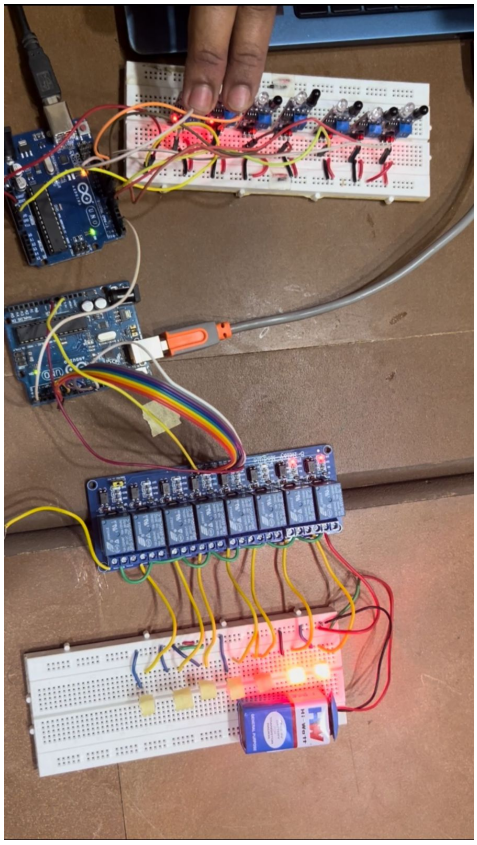

Fig. 6. Working Instance

VI. CODING

The below code controls the functionality of relays. Digital pins that the relays are connected to are listed. To extract a single bit from the byte value, a *getBit* function is defined. The baud rate is set to 9600. *serial.read()* reads the signal from the serial port and stores it in *out*. Now each bit is extracted and checked whether the value is 1, and the corresponding the relay is set HIGH, else the relay is set to LOW. A slight delay is set before iterating the next loop for synchronising the adruino . The delay is 125 ms. In addition, the existing queue in the serial stream is flushed out.

```
int irSensors[] = {5, 7, 9, 8, 10, 11, 12};
int size = 7;
char out = 0;
void setBit(int pos, int value) {
  if (value)
    out |= (1 << pos);
  else
    out &= ~(1 << pos);
}
void setup(){
  Serial.begin(9600);
  Serial.flush();
  for(int i = 0;i < size; i++){
    pinMode(irSensors[i], INPUT);
  }
  out = 0;
}
void loop(){
  for(int i = 0; i < size; i++){
    int sensorStatus = digitalRead(
                          irSensors[i]);
    if (sensorStatus == 0){
        setBit(i, 1);
    }else{
        setBit(i, 0);
    }
  }
  Serial.write(out);
  delay(125);
  Serial.flush();
  out = 0;
}
```

The below code controls the infrared sensors. Digital pins that the sensors are connected to are listed. To set the byte according to value, *setBit* is defined such that if the value is 1, the corresponding bit position is set to *out*. Iterated over, *digitalRead* reads voltage level on the current sensor where a high voltage indicates presence of activation. If the voltage level is high, the *out* is set to 1.

```
int relay_pins[] = {6, 7, 8, 9, 10, 11, 5};
int SIZE = 7;
char out;
int getBit(int pos) {
  return (out >> pos) & 1;
}
void setup(){
  Serial.begin(9600);
  Serial.flush();
  out = 0;
  for(int i = 0; i < SIZE; i++){
    pinMode(relay_pins[i], OUTPUT);
    digitalWrite(relay_pins[i], HIGH);
  }
}
void loop() {
  if (Serial.available() >= 1) {
    out = Serial.read();
    Serial.println(out, DEC);
    for(int i = SIZE; i >= 0; i--){
        if(getBit(i) == 1){
            digitalWrite(relay_pins[i], LOW);
        }else{
            digitalWrite(relay_pins[i], HIGH);
        }
    }
    out =0;
  }
  delay(125);
  Serial.flush();
}
```

VII. RESULTS

The required system was successfully built was tested upon. The IR sensors could accurately sense the presence and send a signal to the relays that are received in the Arduino. The two Arduino communicate with each other through serial communication and according to the activated sensors, the corresponding LED (representing a fan in this case) is turned on. For testing the device, each sensor was first activated separately, for which results were promising with the output displayed in short intervals of time. As the next step of testing, different combinations of activation were performed which yielded correct results. Fig. 7 shows the output when only one IR sensor gets activated and the corresponding fan (i.e. LED) is turned on. The output for a combination of IR sensor activation and receiving the output is displayed in Fig. 8.

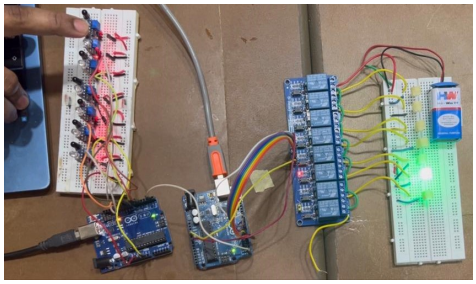

Fig. 7. Output for single activation

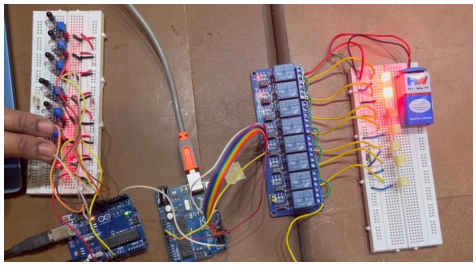

Fig. 8. Output for multiple activation