Bluetooth Controlled Smart Home LED Lighting System(Controlled via app)

A Mini Project Report
Of
Microprocessor Lab (20CP202)

Department of Computer Engineering
School of Technology
Pandit Deendayal Energy University
Gandhinagar, Gujarat, India

Sem - 3rd
Div-2, G4

Submitted by:
Dhruv Jain (23BCP135)
Darshit Vadher (23BCP134)
CSE, 3rd Sem

Submitted To:
Dr. Deepak Sahu
Assistant Professor,
Department of ICT,
SOT, PDEU

Abstract / Objective:

This project creates a smartphone-controlled smart lighting system using Arduino and Bluetooth. It enables remote, convenient control of home lighting via an Android app, allowing users to manage individual or all lights with ease. The setup demonstrates a practical approach to integrating IoT concepts into household devices, enhancing everyday convenience.

Apparatus and Software Requirements:

- 3 x 5mm LED
- 3 x 220Ω Resistor
- Arduino-Compatible UNO R3
- 5.5cm x 8.5cm Breadboard
- 1 x HC-05 Bluetooth Module
- 9 x Jump Wire (M-M)
- 1 x Bluetooth-Enabled Android Phone
- Arduino IDE Software (for coding)
- Arduino Bluetooth Controller (Application)

Complete Running Code:

```
#include <Arduino.h>
#include <Wire.h>
#include <SoftwareSerial.h>

// Pin assignments for each room's light
int livingroom = 5;
int bedroom = 6;
int diningroom = 7;

// Define Bluetooth serial on default TX/RX pins
SoftwareSerial Bluetooth(0, 1);
char Data;  // Variable to hold incoming Bluetooth data

// Function to send data back to Bluetooth device
void sendData(String transmitData) {
    Bluetooth.println(transmitData);
```

```
}

void setup() {
    // Initialize Bluetooth communication
    Bluetooth.begin(9600);

    // Set each room's light pin as an output
    pinMode(livingroom, OUTPUT);
    pinMode(bedroom, OUTPUT);
    pinMode(diningroom, OUTPUT);
}

void loop() {
    // Check if Bluetooth data is available
    if (Bluetooth.available()) {
        Data = Bluetooth.read();  // Read received data

        // Control lights based on received data
        if (Data == '4') {
            digitalWrite(livingroom, HIGH);
            sendData("Living Room Light ON");
        }
        if (Data == '1') {
            digitalWrite(livingroom, LOW);
            sendData("Living Room Light OFF");
        }
        if (Data == '5') {
            digitalWrite(bedroom, HIGH);
            sendData("Bedroom Light ON");
        }
        if (Data == '2') {
            digitalWrite(bedroom, LOW);
            sendData("Bedroom Light OFF");
        }
        if (Data == '6') {
            digitalWrite(diningroom, HIGH);
            sendData("Dining Room Light ON");
        }
        if (Data == '3') {
            digitalWrite(diningroom, LOW);
            sendData("Dining Room Light OFF");
        }
        if (Data == '9') {
            // Turn on all lights
            digitalWrite(livingroom, HIGH);
            digitalWrite(bedroom, HIGH);
            digitalWrite(diningroom, HIGH);
            sendData("ALL LIGHTS ON");
```

```
        }
        if (Data == '0') {
            // Turn off all lights
            digitalWrite(livingroom, LOW);
            digitalWrite(bedroom, LOW);
            digitalWrite(diningroom, LOW);
            sendData("ALL LIGHTS OFF");
        }
    }
}
```

Algorithm and Implementation Procedure:

Step-by-Step Explanation:

1. ***Component Setup and Wiring***:
    - *Connect each LED (representing room lights) to specific pins on the Arduino: living room light to pin 5, bedroom light to pin 6, and dining room light to pin 7.*
    - *Attach the Bluetooth module to the Arduino using RX/TX pins for wireless communication.*
2. ***Initialize Code and Libraries***:
    - *Import the necessary libraries: SoftwareSerial for Bluetooth communication.*
    - *Define pins for each room's light and configure them as outputs in the setup() function.*
3. ***Bluetooth Communication Configuration***:
    - *Set the Bluetooth module's communication rate (9600 baud) and assign it to pins 0 (RX) and 1 (TX).*
4. ***Command Handling via Bluetooth***:
    - *Continuously check for incoming data from the Bluetooth module.*
    - *When a command is received, execute the corresponding action:*
        - *Turn on or off a specific light based on received data (characters '4', '5', '6' to turn on lights; '1', '2', '3' to turn them off).*
        - *Handle global commands, like '9' for turning on all lights and '0' for turning them all off.*
    - *Send confirmation messages back to the Bluetooth device to indicate the action taken.*
5. ***Testing and Adjustment***:
    - *Test each command with the Bluetooth app to ensure accurate control and confirmation feedback.*
    - *Adjust any responses or troubleshoot connectivity issues as needed.*

This procedure enables a robust, mobile-controlled lighting system suitable for various room configurations.

Connections and Setup for the Smart Home Lighting System:

1. ***Arduino Board**: Acts as the main controller, executing the code to control lighting through Bluetooth commands.*
2. ***Bluetooth Module (HC-05)**: Connected to Arduino via TX (pin 1) and RX (pin 0) for wireless communication with a smartphone. The module receives commands from the phone and transmits them to the Arduino.*
3. ***LED Connections**: Each room's light (represented by an LED) connects to digital pins on the Arduino—pin 5 for the living room, pin 6 for the bedroom, and pin 7 for the dining room. These LEDs toggle based on commands.*
4. ***Power Supply**: The Arduino and Bluetooth module receive power from an external supply or USB connection.*
5. ***Software Setup**: Arduino IDE for coding, and a Bluetooth control app on the phone to send control signals to the Arduino for testing and operation.*

This configuration enables Bluetooth-based remote control of each LED, simulating smart home lighting controls.

Photos:

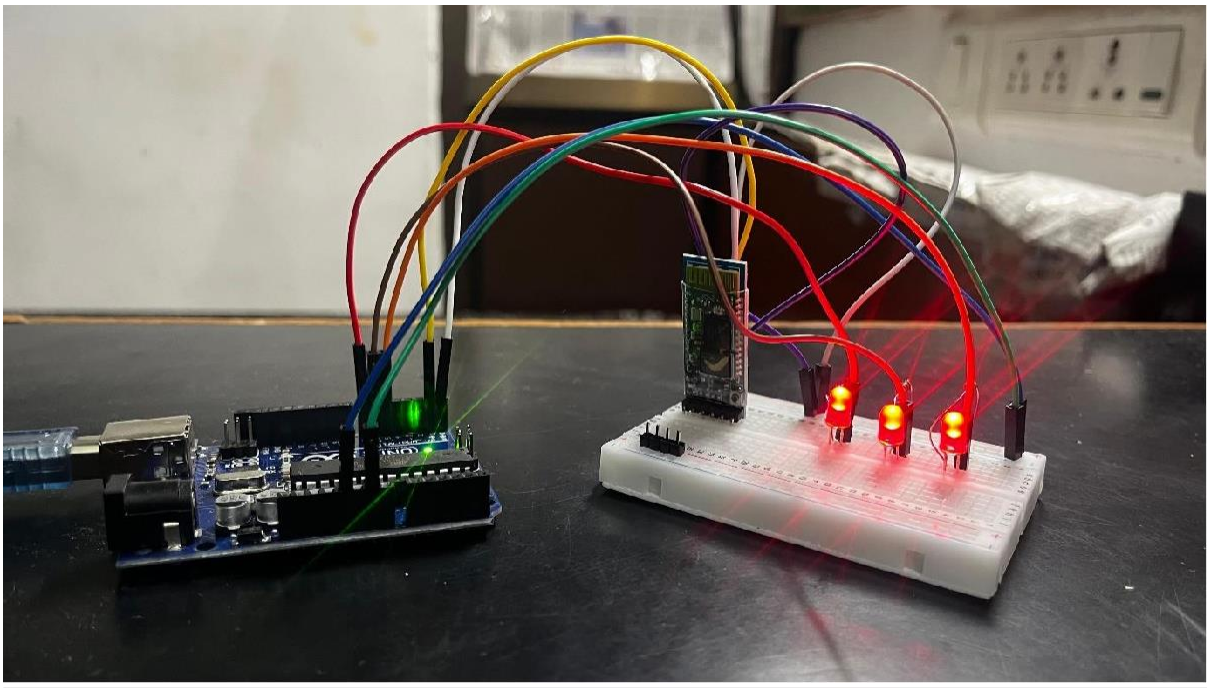

Figure 1: Running Project Photo

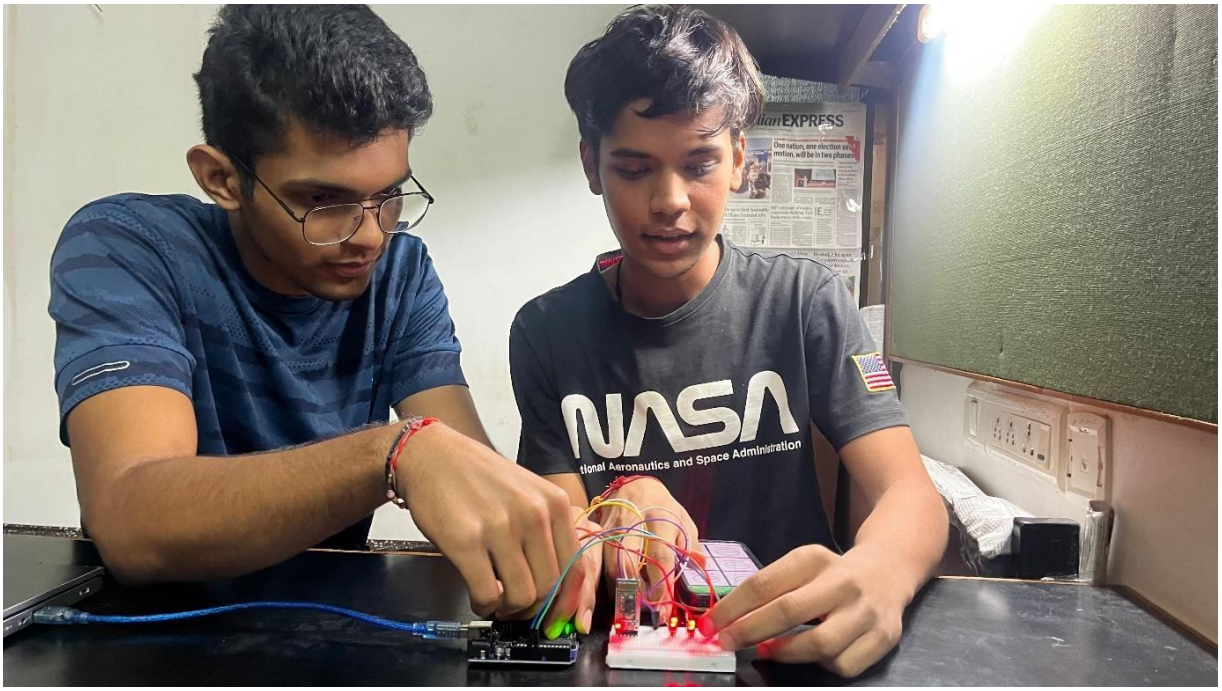

Figure 2: Team Working On Project

Summary Of Experience:

Working on this smart home lighting project was both challenging and exciting. We got to dive deep into Arduino programming, experiment with Bluetooth communication, and connect it all through an Android app. This experience has given me a real sense of how technology can make everyday tasks easier, and it's boosted our confidence in tackling similar projects in the future.