Lab Report: ESP32 Interfacing Experiments

V.L.S. Bhargav ; Lanka Kushwanth
Table Number: 3
Room Number: 114
Roll Number: 2025102061

October 15, 2025

PART A: RGB LED Control Using Push Button

Aim:

To control the color of an RGB LED using a push button on the ESP32. Each button press cycles the LED through Red, Green, Blue, and OFF states, demonstrating basic digital I/O control.

Components Required:

- ESP32 Development Board
- Common Cathode RGB LED
- Push Button
- $3 \times 220\Omega$ resistors
- Breadboard and Jumper Wires

Theory:

- The RGB LED combines red, green, and blue LEDs in one package.
- Each LED color is controlled via a GPIO pin using digitalWrite().
- The push button provides user input, read using digitalRead().
- Debouncing and cycling logic allow changing LED color on each button press.

Procedure:

1. Connect RGB LED pins (R, G, B) to ESP32 GPIO pins through 220Ω resistors.
2. Connect button between GPIO pin and GND (use internal pull-up if needed).
3. Write and upload code using Arduino IDE.
4. Each press should cycle LED through different colors.

Physical Circuit:

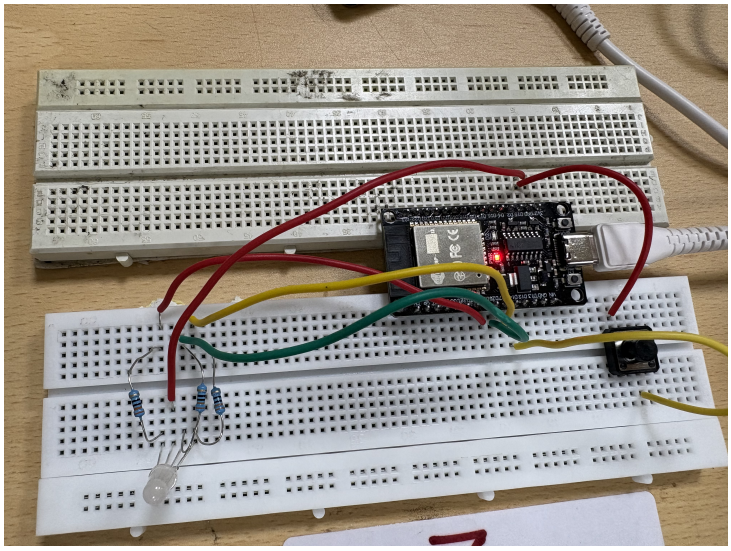
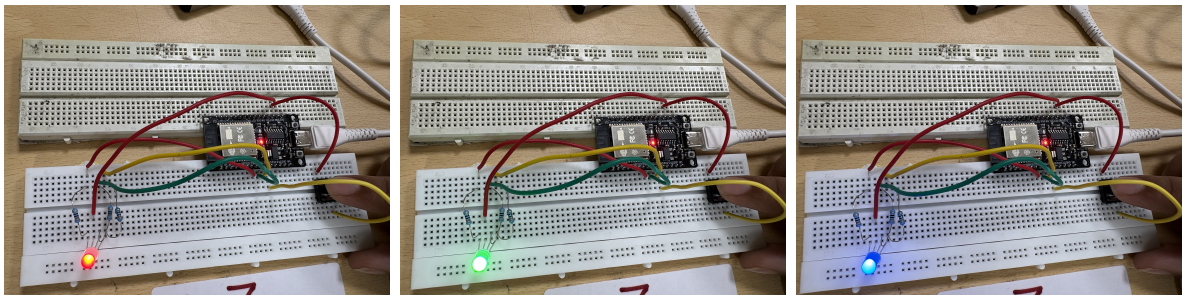


Figure 1: Working setup for RGB LED and Push Button on ESP32.



(a) Red Color (b) Green Color (c) Blue Color

Figure 2: RGB LED displaying Red, Green, and Blue colors respectively.

Code:

```
int redPin = 25;
int greenPin = 26;
int bluePin = 27;
int buttonPin = 14;
int buttonState = 0;
int lastState = 0;
int colorState = 0;

void setup() {
  pinMode(redPin, OUTPUT);
  pinMode(greenPin, OUTPUT);
  pinMode(bluePin, OUTPUT);
  pinMode(buttonPin, INPUT_PULLUP);
  Serial.begin(115200);
  Serial.println("RGB LED Control Started");
}

void loop() {
  buttonState = digitalRead(buttonPin);

  if (buttonState == LOW && lastState == HIGH) {
    colorState = (colorState + 1) % 4;
    delay(200); // debounce delay
  }
  lastState = buttonState;

  switch (colorState) {
    case 0: digitalWrite(redPin, HIGH);
            digitalWrite(greenPin, LOW);
            digitalWrite(bluePin, LOW);
            break;
    case 1: digitalWrite(redPin, LOW);
            digitalWrite(greenPin, HIGH);
            digitalWrite(bluePin, LOW);
            break;
    case 2: digitalWrite(redPin, LOW);
            digitalWrite(greenPin, LOW);
            digitalWrite(bluePin, HIGH);
            break;
    default:digitalWrite(redPin, LOW);
            digitalWrite(greenPin, LOW);
            digitalWrite(bluePin, LOW);
  }
}
```

Listing 1: RGB LED Control Using Push Button (ESP32)

Observations:

- Button press cycles through Red → Green → Blue → OFF.
- The LED color changes immediately upon button press.
- Debouncing ensures single-color transition per press.

Conclusion:

- RGB LED successfully controlled using push button with ESP32.
- Demonstrates basic GPIO input/output operations.

PART B: Servo Motor Control Using IR Sensor

Aim:

To use an IR sensor to detect an object and control a servo motor's rotation using the ESP32.

Components Required:

- ESP32 Development Board
- IR Obstacle Sensor (HW201)
- SG90 Servo Motor
- Breadboard and Jumper Wires

Theory:

- The IR sensor outputs LOW when an object is detected.
- The servo motor angle is controlled using PWM (Pulse Width Modulation).
- The ESP32Servo library provides easy servo control functions.

Procedure:

1. Connect the IR sensor output pin to a GPIO input pin of ESP32.
2. Connect servo control pin to a PWM-capable GPIO pin.
3. Power both modules from ESP32 (use 3.3V for sensor, Vin for servo).
4. Upload the code and observe servo movement when object is detected.

Code:

```
#include <ESP32Servo.h>
Servo gateServo;

int irPin = 15;
int servoPin = 18;
int irState = 0;

void setup() {
  Serial.begin(115200);
  gateServo.attach(servoPin);
  pinMode(irPin, INPUT);
  gateServo.write(0);
  Serial.println("IR Sensor Servo Control Started");
}

void loop() {
  irState = digitalRead(irPin);

  if (irState == LOW) {   // Object detected
    gateServo.write(90);
    Serial.println("Object detected: Servo rotated to 90 degrees");
    delay(1000);
  } else {
    gateServo.write(0);
    Serial.println("No object: Servo at 0 degrees");
  }
  delay(200);
}
```

Listing 2: Servo Control Using IR Sensor (ESP32)

Physical Circuit:

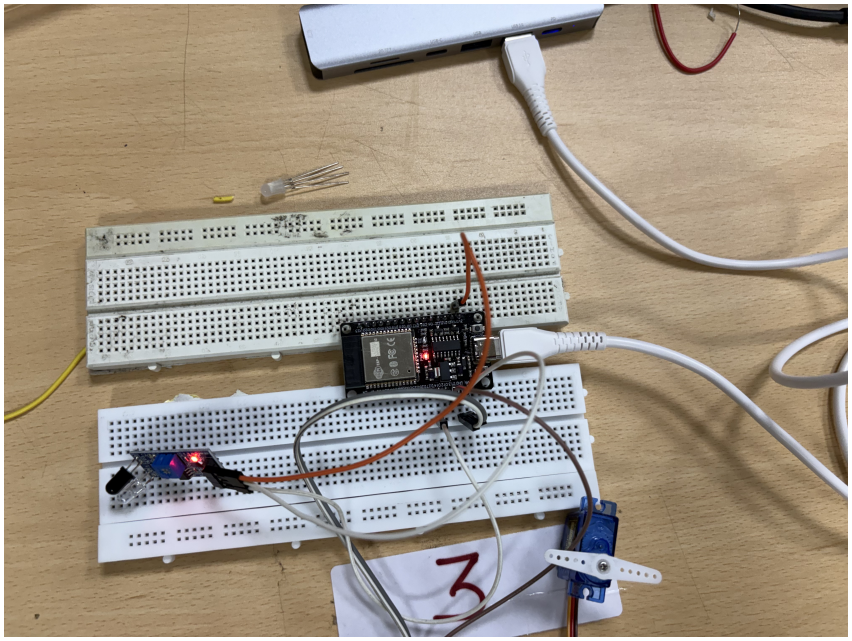


Figure 3: Working setup of ESP32 with IR Sensor and Servo Motor.

Observations:

- Servo motor rotates to 90° when IR sensor detects an obstacle.
- Servo returns to 0° when the obstacle is removed.
- System response was smooth and reliable.

Conclusion:

- IR sensor successfully detected objects and controlled servo motor rotation.
- Demonstrated effective use of sensors and actuators using ESP32 GPIO and PWM.

References:

- Lab Manual
- Arduino and ESP32 Documentation