RGB STRIP CONTROL USING MOBILE PHONE

1)Ibrahim Elsayed 19015163
2)Ibrahim Refaat 19015166
3) Abdallah Mohamed 19015994
4)Basmala El Dabaa 19015517
5)Toqa Mahmoud 19015542

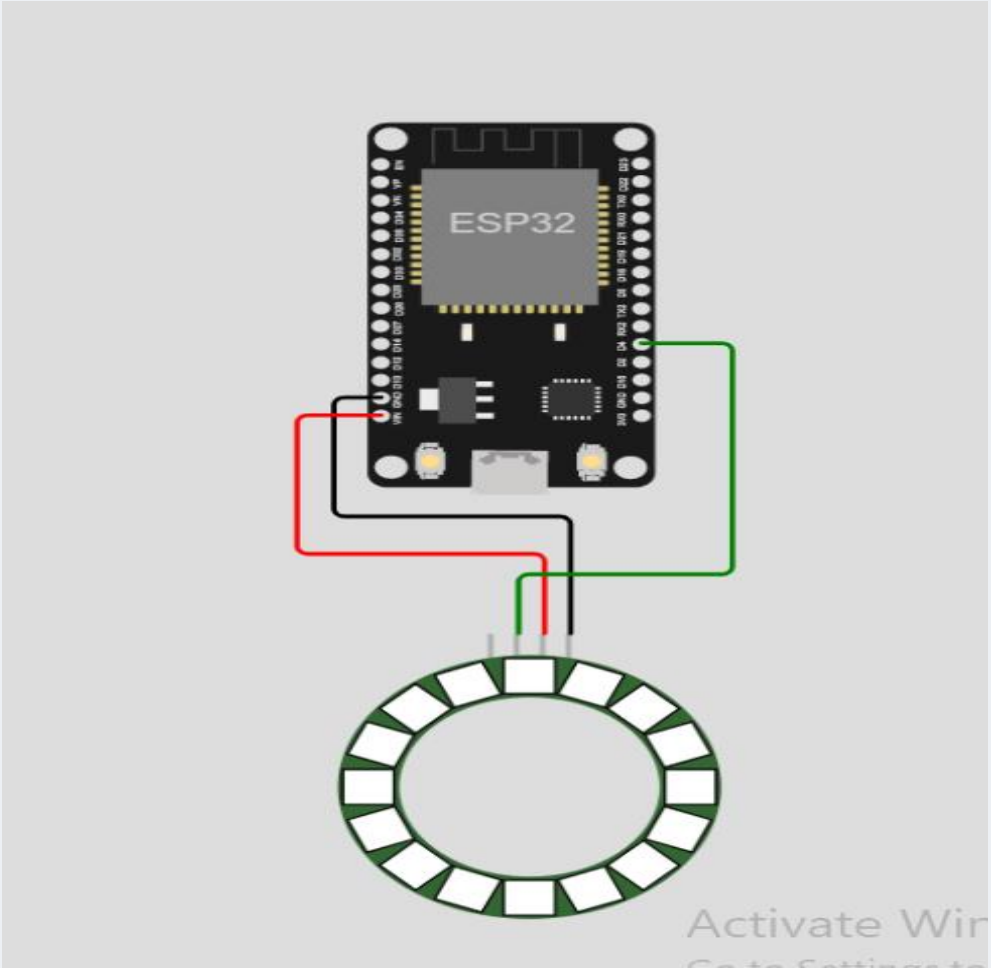
- **Aim of the hardware project.**

In this project we'll show how to remotely control an RGB LED strip (ucs1903) with an ESP32 board using a web server with a color picker and mobile application.

- **List of the used components**

1) ESP32 board
2) RGB LED strip (ucs1903)
3) Breadboard
4) Micro USB cable

- **Schematic of the circuit implemented.**



- **Procedure to use this circuit.**

For Bluetooth Mode:

1. Install (Serial Bluetooth Terminal) app on your android.
2. Open the Bluetooth and pair your phone with the esp32 with name (ESP32Test).
3. For a given RGB color ex:(255,000,000) which is red, send a message with this form (“rgb 255 000 000”) where (rgb) is the unique message which makes the esp32 understand and call the function controlling the led strip.

For Wi-Fi Mode:

1. Write your wi-fi network ssid and password in the char variable of each one in the code, then upload it to the esp32 using Arduino ide.
2. Your network must have access to the internet.
3. Download (Blynk Iot) app on your android.
4. Open the template built after logging with the given username and password.
5. Choose the color you want and pick it from the zebra.

- **Code Screenshots.**

For Bluetooth Mode:

1. Includes and variables declaration.

```
#include <string>
using namespace std;
#include "BluetoothSerial.h"

#if !defined(CONFIG_BT_ENABLED) || !defined(CONFIG_BLUEDROID_ENABLED)
#error Bluetooth is not enabled! Please run `make menuconfig` to and enable it
#endif
#include "FastLED.h"
#define NUM_LEDS 50

#include <thread>
#define DATA_PIN 4
String message = "";
int red;
int green;
int blue;
CRGB leds[NUM_LEDS];
BluetoothSerial SerialBT;
```

2. Color Change Function and Setup Function.

```
void color_mode(String color){
      red=color.substring(5,8).toInt();
      blue=color.substring(9,12).toInt();
      green=color.substring(12,15).toInt();
      for (int led_index=0;led_index<50;led_index++){
      leds[led_index] = CRGB(red,blue,green);

      //delay(500);
    }
    FastLED.show();
  }
void setup() {
  Serial.begin(115200);
  SerialBT.begin("ESP32test"); //Bluetooth device name
  Serial.println("The device started, now you can pair it with bluet
  FastLED.addLeds<UCS1903, DATA_PIN, RGB>(leds, NUM_LEDS);
}
```

3. Loop Function.

```
void loop() {
if (SerialBT.available()){
    char incomingChar = SerialBT.read();
    if (incomingChar != '\n'){
      message += String(incomingChar);
    }
    else{
      message = "";
    }
    Serial.write(incomingChar);
  }
  if (message.substring(0,3)=="rgb"){
    color_mode(message);
    }
}
```

For Wi-Fi Mode:

1. Includes and Variables declaration.

```
#define BLYNK_TEMPLATE_ID           "TMPLVDtr6Tsz"
#define BLYNK_DEVICE_NAME           "Quickstart Device"
#define BLYNK_AUTH_TOKEN            "bybNK4EJUPL4tWw_iTCINa7NPUnQ0dk


// Comment this out to disable prints and save space
#define BLYNK_PRINT Serial

#include "FastLED.h"
#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>
int data=255;
int r,g,b;
#define DATA_PIN 4
#define NUM_LEDS 50
char auth[] = BLYNK_AUTH_TOKEN;

// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "85:45";
char pass[] = "27/6/01/05@hema";
CRGB leds[NUM_LEDS];
BlynkTimer timer;
```

2. Blynk Functions

```
BLYNK_WRITE(V0)

BLYNK_WRITE(V3)
{
  r = param[0].asInt();
  g = param[1].asInt();
  b = param[2].asInt();

  static1(r, g, b,data);
}
void static1(int r, int g, int b,int brightness)
{
  FastLED.setBrightness(brightness);
  for (int i = 0; i < NUM_LEDS; i++ )
  {
    leds[i] = CRGB(r, g, b);
  }
  FastLED.show();
}

{
data = param.asInt();
static1(r, g, b,data);
}
```

3. Loop & Setup Functions

```
void setup()
{
  // Debug console
  Serial.begin(115200);

  Blynk.begin(auth, ssid, pass);

  FastLED.addLeds<UCS1903, DATA_PIN, RGB>(leds, NUM_LEDS);
}

void loop()
{
  Blynk.run();

}
```

• Budget of the project.

Component	Price	Link
UCS 1903 LED Strip	100 LE	UCS 1903-Makers
ESP32	250 LE	ESP32-Makers
USB To Micro	20 LE	USB-Makers
Total	270 LE	

• Challenges that the team had and how to overcome them.

1) *We couldn't find the led (ws2182)*
   We replaced it by (ucs1903)
   We induced that they can operate on the same code

• References.

1) *https://www.studiopieters.nl/esp32-pinout/*
2) *https://randomnerdtutorials.com/esp32-esp8266-rgb-led-strip-web-server/*
3) *https://iotdesignpro.com/projects/controlling-ws2812-neopixel-led-with-esp32-using-blynk-app*