

PRAKTIK PEMBUATAN SISTEM TRAFFIC LIGHT MENGUNAKAN ESP32

Ray Nafi Salsabila Arindivah

Fakultas Vokasi, Universitas Brawijaya

Email: salsarindivah@student.ub.ac.id

ABSTRACT

Lampu lalu lintas adalah perangkat elektronik yang mengendalikan arus lalu lintas kendaraan dan pejalan kaki menggunakan serangkaian lampu berwarna. Lampu lalu lintas yang paling umum terdiri dari tiga lampu: merah, kuning, dan hijau. Lampu merah menunjukkan bahwa kendaraan dan pejalan kaki harus berhenti, lampu kuning menunjukkan bahwa lampu merah akan segera menyala, dan lampu hijau menunjukkan bahwa kendaraan dan pejalan kaki dapat melanjutkan perjalanan.

Sistem lampu lalu lintas dengan mikrokontroler ESP32 akan melibatkan penyambungan ESP32 ke serangkaian LED (merah, kuning, dan hijau) untuk mensimulasikan lampu lalu lintas. Dalam proyek ini kita akan membuat lampu lalu lintas yang dikontrol oleh kartu ESP32 di mana tombol 1 di tekan lampu merah kedip 5x, tombol 2 di tekan lampu merah dan hijau kedip bergantian, dan tombol 3 di tekan lampu merah, kuning, hijau kedip bergantian. Kesimpulan dari penelitian ini adalah bahwa penggunaan ESP32 dalam sistem traffic light memungkinkan otomatisasi yang lebih cerdas dan fleksibel.

Kata Kunci- ESP32, LED, Traffic Light

BAB I

PENDAHULUAN

1.1 Latar Belakang

Kemacetan lalu lintas jadi masalah besar di banyak kota, terutama di persimpangan jalan yang padat. Sistem traffic light konvensional umumnya menggunakan timer tetap, sehingga tidak bisa menyesuaikan kondisi lalu lintas secara real-time. Akibatnya, ada waktu tunggu yang tidak efisien—misalnya, lampu merah tetap menyala lama meskipun jalan sedang sepi.

Dengan perkembangan teknologi, sistem lalu lintas pintar mulai dikembangkan untuk mengatasi masalah ini. Salah satu solusi yang bisa diterapkan adalah menggunakan ESP32, mikrokontroler yang mendukung konektivitas Wi-Fi dan memiliki kemampuan pemrosesan yang cukup baik. Dengan bantuan sensor, sistem ini bisa mendeteksi kepadatan kendaraan dan mengatur durasi lampu lalu lintas secara dinamis, sehingga arus kendaraan bisa lebih lancar.

Penelitian ini bertujuan untuk merancang dan menguji sistem traffic light berbasis ESP32 yang mampu menyesuaikan waktu nyala lampu berdasarkan kondisi lalu lintas. Selain itu, sistem ini juga bisa dikendalikan secara nirkabel, sehingga memudahkan pemantauan dan pengelolaan dari jarak jauh. Dengan penerapan teknologi ini, diharapkan efisiensi lalu lintas bisa meningkat dan waktu tunggu pengendara bisa dikurangi.

1.2 Tujuan Eksperimen

Tujuan dari eksperimen ini adalah untuk mempelajari dan memahami cara menginput tiga tombol pada mikrokontroler ESP32 serta mengendalikan LED berdasarkan tombol yang ditekan. Dalam eksperimen ini, setiap tombol memiliki fungsi yang berbeda dalam mengontrol pola kedipan LED. Selain itu, eksperimen ini juga bertujuan untuk memahami konsep debounce tombol agar pembacaan input lebih akurat serta menerapkan pemrograman berbasis kondisi seperti if-else atau switch-case untuk mengendalikan LED sesuai dengan tombol yang ditekan. Dengan demikian, eksperimen ini dapat menjadi dasar dalam pengembangan sistem kendali sederhana berbasis ESP32 yang menggunakan tombol sebagai input dan LED sebagai indikator visual.

BAB II

METODOLOGI

2.1 Alat dan Bahan

- Visual Studio
- Wokwi Simulator
- ESP32 – Mikrokontroler utama untuk mengontrol sistem traffic light.
- LED Merah, Kuning, Hijau – Sebagai indikator lampu lalu lintas.
- Resistor – Untuk membatasi arus ke LED.
- Kabel Jumper – Menghubungkan komponen elektronik pada rangkaian.
- Adaptor atau Power Bank (5V/9V) – Sebagai sumber daya untuk ESP32.
- Pushbutton.

2.2 Langkah Implementasi

1. Perancangan Sistem

- Menentukan skema rangkaian dengan ESP32 sebagai pusat kendali.
- Susun komponen seperti LED, resistor, dan pushbutton.
- Pastikan koneksi kabel jumper sesuai dengan kebutuhan sistem.

2. Pengkodean Program

- Gunakan Arduino IDE untuk menulis kode ESP32.
- Membuat program untuk:
 - a. Mengontrol nyala LED sesuai tombol yang ditekan.
 - b. Mengatur tombol 1 di tekan lampu merah kedip 5x, tombol 2 di tekan lampu merah dan hijau kedip bergantian, dan tombol 3 di tekan lampu merah, kuning, hijau kedip bergantian.
- Unggah kode ke ESP32 dan lakukan debugging jika ada kesalahan.

3. Pengujian Sistem

- Uji apakah lampu lalu lintas menyala sesuai tombol yang diinginkan.
- Pastikan tombol dan nyala lampu sesuai dengan yang sudah ditentukan.
- Lakukan penyempurnaan jika ada kendala atau kesalahan dalam sistem.

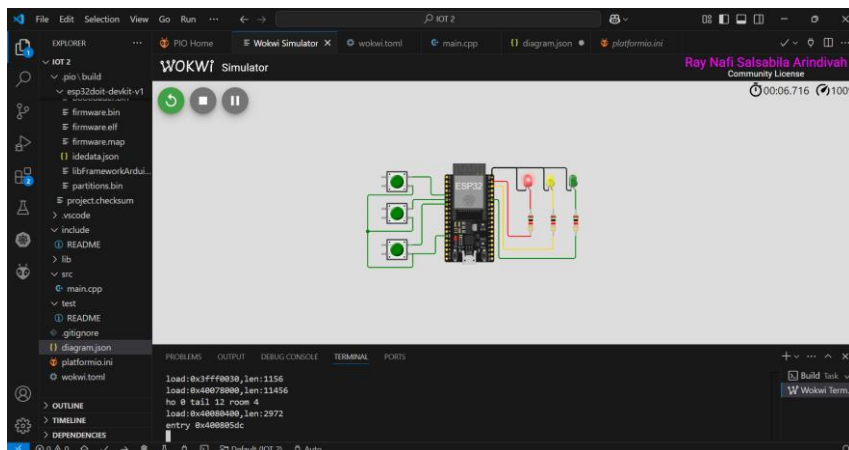
BAB III

Hasil dan Pembahasan

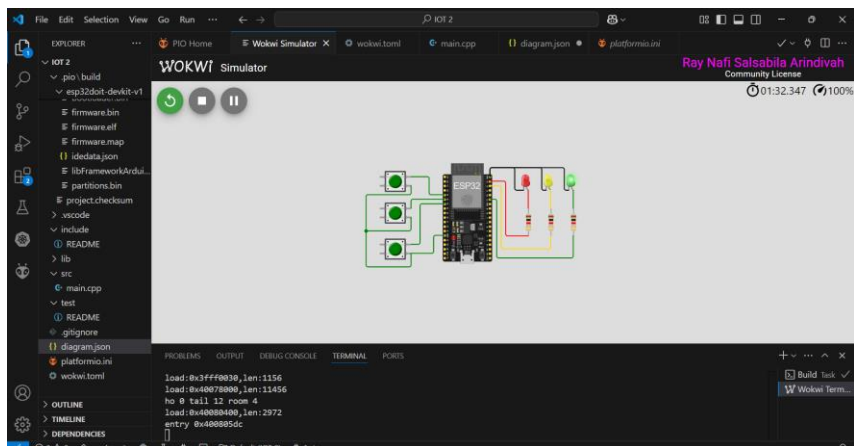
3.1 Hasil Eksperimen

No.	Aspek Pengujian	Hasil	Keterangan
1.	Penyesuaian Tombol Lampu	Berhasil	Lampu menyala sesuai tombol yang ditentukan. (tombol 1 di tekan lampu merah kedip 5x, tombol 2 di tekan lampu merah dan hijau kedip bergantian, dan tombol 3 di tekan lampu merah, kuning, hijau kedip bergantian.)

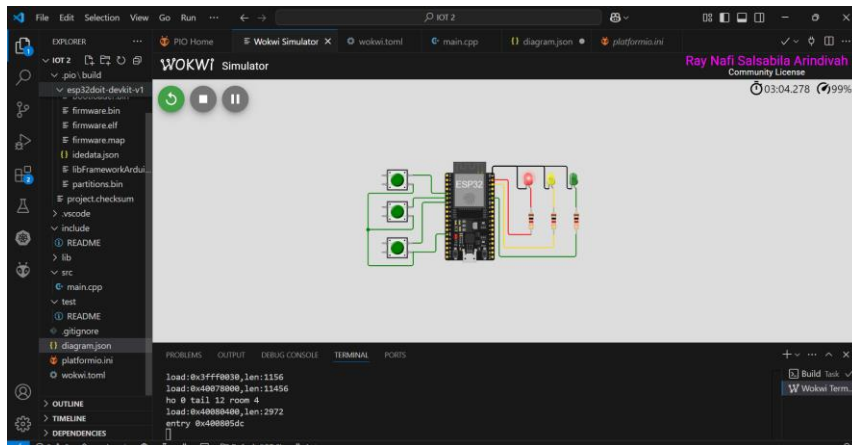
• Screenshoot ESP32 tombol 1 ditekan



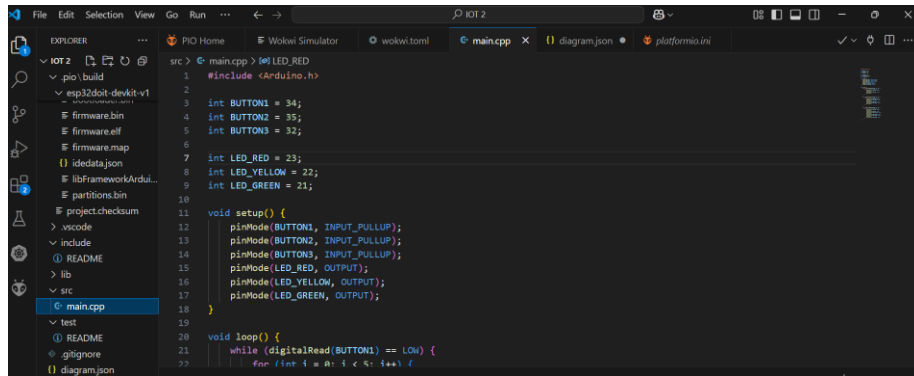
• Screenshoot ESP32 tombol 2 ditekan



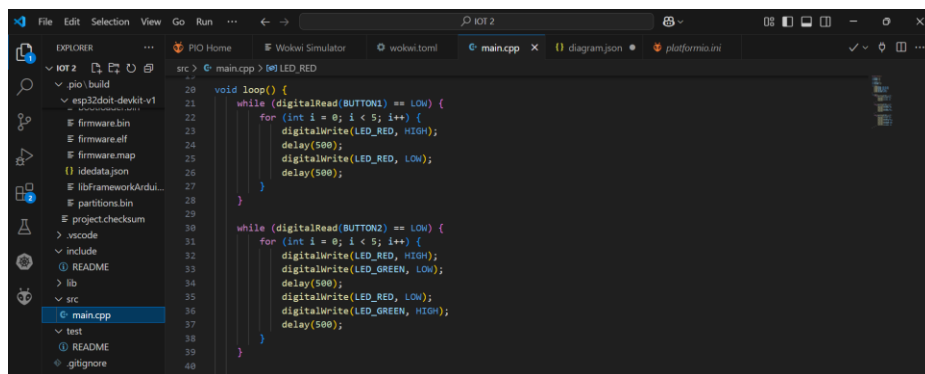
- Screenshoot ESP32 tombol 3 ditekan



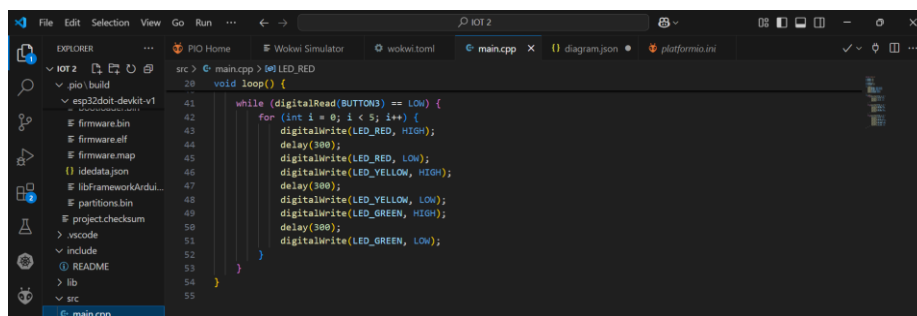
BAB IV LAMPIRAN



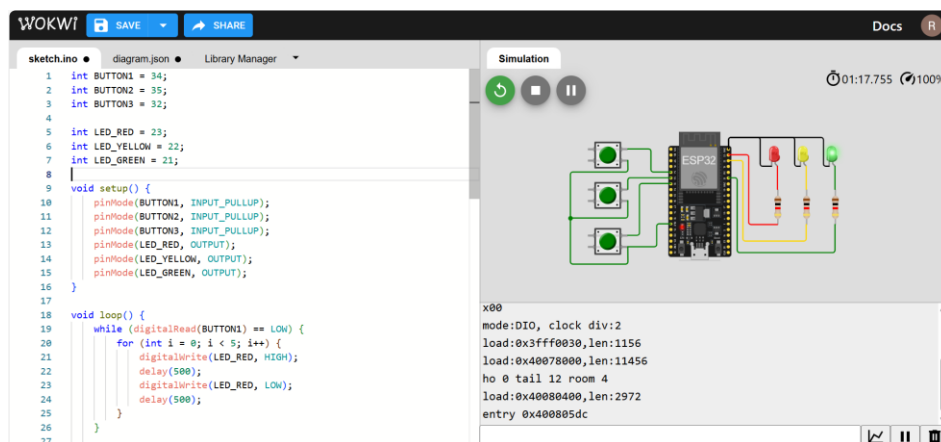
```
src > E: main.cpp > [0] LED_RED
1 #include <Arduino.h>
2
3 int BUTTON1 = 34;
4 int BUTTON2 = 35;
5 int BUTTON3 = 32;
6
7 int LED_RED = 23;
8 int LED_YELLOW = 22;
9 int LED_GREEN = 21;
10
11 void setup() {
12   pinMode(BUTTON1, INPUT_PULLUP);
13   pinMode(BUTTON2, INPUT_PULLUP);
14   pinMode(BUTTON3, INPUT_PULLUP);
15   pinMode(LED_RED, OUTPUT);
16   pinMode(LED_YELLOW, OUTPUT);
17   pinMode(LED_GREEN, OUTPUT);
18 }
19
20 void loop() {
21
22 }
```



```
20 void loop() {
21   while (digitalRead(BUTTON1) == LOW) {
22     for (int i = 0; i < 5; i++) {
23       digitalWrite(LED_RED, HIGH);
24       delay(500);
25       digitalWrite(LED_RED, LOW);
26       delay(500);
27     }
28   }
29
30   while (digitalRead(BUTTON2) == LOW) {
31     for (int i = 0; i < 5; i++) {
32       digitalWrite(LED_RED, HIGH);
33       digitalWrite(LED_GREEN, LOW);
34       delay(500);
35       digitalWrite(LED_RED, LOW);
36       digitalWrite(LED_GREEN, HIGH);
37       delay(500);
38     }
39   }
40 }
```



```
41 void loop() {
42   while (digitalRead(BUTTON3) == LOW) {
43     for (int i = 0; i < 5; i++) {
44       digitalWrite(LED_RED, HIGH);
45       delay(300);
46       digitalWrite(LED_RED, LOW);
47       digitalWrite(LED_YELLOW, HIGH);
48       delay(300);
49       digitalWrite(LED_YELLOW, LOW);
50       digitalWrite(LED_GREEN, HIGH);
51       delay(300);
52       digitalWrite(LED_GREEN, LOW);
53     }
54   }
55 }
```



WOKWI

SAVE SHARE Docs

sketch.ino diagram.json Library Manager

```
1 int BUTTON1 = 34;
2 int BUTTON2 = 35;
3 int BUTTON3 = 32;
4
5 int LED_RED = 23;
6 int LED_YELLOW = 22;
7 int LED_GREEN = 21;
8
9 void setup() {
10   pinMode(BUTTON1, INPUT_PULLUP);
11   pinMode(BUTTON2, INPUT_PULLUP);
12   pinMode(BUTTON3, INPUT_PULLUP);
13   pinMode(LED_RED, OUTPUT);
14   pinMode(LED_YELLOW, OUTPUT);
15   pinMode(LED_GREEN, OUTPUT);
16 }
17
18 void loop() {
19   while (digitalRead(BUTTON1) == LOW) {
20     for (int i = 0; i < 5; i++) {
21       digitalWrite(LED_RED, HIGH);
22       delay(500);
23       digitalWrite(LED_RED, LOW);
24       delay(500);
25     }
26   }
27 }
```

Simulation

01:17.755 100%

ESP32

x00

mode:DIO, clock div:2

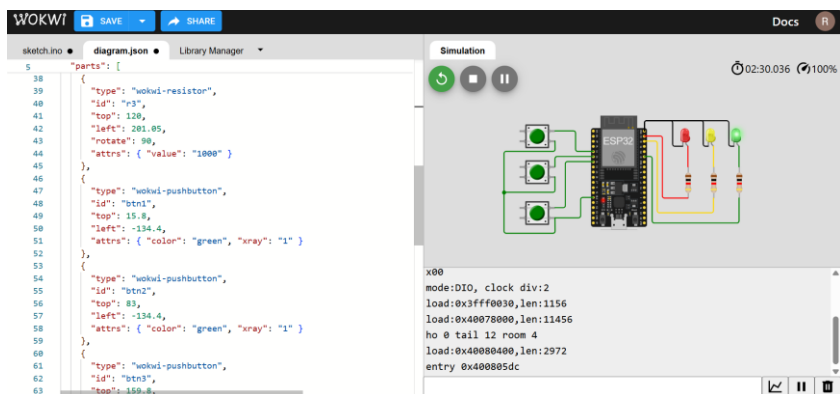
load:0x3fff0030,len:1156

load:0x40078000,len:11456

ho 0 tail 12 room 4

load:0x40080400,len:2972

entry 0x400805dc



WOKWI

SAVE

SHARE

Docs

sketch.ino

diagram.json

Library Manager

5

60

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

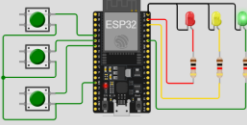
87

88

```
"parts": [
  {
    "left": -134.4,
    "attr": { "color": "green", "xray": "1" }
  },
  { "type": "wokwi-junction", "id": "j1", "top": 139.2, "left": -168,
  },
  "connections": [
    [ "esp:TX", "$serialMonitor:RX", "", [] ],
    [ "esp:RX", "$serialMonitor:TX", "", [] ],
    [ "esp:GND.2", "led2:C", "black", [ "v-19.2", "h52.84", "v48" ] ],
    [ "led2:A", "r2:1", "red", [ "v0" ] ],
    [ "led1:A", "r3:1", "gold", [ "v38.4" ] ],
    [ "led3:A", "r1:1", "green", [ "v0" ] ],
    [ "r2:2", "esp:123", "red", [ "h-48", "v-116.4" ] ],
    [ "r3:2", "esp:22", "gold", [ "v27.6", "h-105.6", "v-134.4" ] ],
    [ "r1:2", "esp:21", "green", [ "v46.8", "h-163.2", "v-124.8" ] ],
    [ "btn1:1.n", "esp:34", "green", [ "v0", "h38.6", "v38.4" ] ],
    [ "btn2:1.n", "esp:35", "green", [ "v0" ] ],
    [ "btn3:1.n", "esp:32", "green", [ "v0", "h39.4", "v-86.4" ] ],
    [ "btn1:2.n", "btn2:2.n", "green", [ "h0.2", "v19.4", "h-96", "v76.8" ] ],
    [ "btn3:2.n", "esp:GND.1", "green", [ "h48.2", "v-38.2" ] ],
    [ "btn3:2.n", "j1:1", "green", [ "h0.2", "v29", "h-96" ] ],
    [ "led1:C", "esp:GND.2", "black", [ "h-9.2", "v-48", "h-118.44" ] ],
    [ "led3:C", "esp:GND.2", "black", [ "h-9.2", "v-48", "h-158.44" ] ]
  ],
  "dependencies": {}
}
```

Simulation

02:46:951 99%



x00
mode:DIO, clock div:2
load:0x3ffff030,len:1156
load:0x40078000,len:11456
ho 0 tail 12 room 4
load:0x40080400,len:2972
entry 0x400805dc

BAB V

PENUTUP

5.1 Kesimpulan

Kesimpulannya, eksperimen ini berhasil menunjukkan bahwa ESP32 dapat digunakan untuk membaca input dari tiga tombol dan mengendalikan LED sesuai dengan logika yang telah ditentukan. Setiap tombol dapat menjalankan fungsinya dengan baik, di mana tombol pertama menyebabkan LED merah berkedip lima kali, tombol kedua membuat LED merah dan hijau berkedip bergantian, serta tombol ketiga menghasilkan kedipan bergantian antara LED merah, kuning, dan hijau. Selain itu, penerapan debounce tombol terbukti efektif dalam mencegah pembacaan input yang tidak akurat. Pemrograman berbasis kondisi seperti if-else atau switch-case juga berfungsi dengan baik dalam mengatur alur logika kendali. Dengan demikian, eksperimen ini dapat menjadi dasar dalam pengembangan sistem kontrol sederhana yang menggunakan tombol sebagai input dan LED sebagai indikator visual.