

Dokumentasi Praktikum ESP32

Untuk memenuhi Ujian Akhir Semester mata kuliah Sistem Mikroprosessor
Dosen pengampu Bapak Muhammad Ikhwan Fathulloh, S.Kom.



disusun oleh :

Nama	:	Nur Anisa
NIM	:	23552011171
Kelas	:	TIF K 23A

**Universitas Teknologi Bandung
Teknik Informatika
Bandung
2026**

1. Deskripsi Sistem

Sistem ini merupakan implementasi embedded system berbasis ESP32 yang mengintegrasikan kontrol perangkat keras, komunikasi jaringan, dan antarmuka web. Sistem membaca input tombol menggunakan mekanisme interupsi, mengontrol LED menggunakan teknik PWM, menyimpan konfigurasi WiFi pada memori non-volatile, serta mengirim data melalui protokol MQTT. Seluruh proses dijalankan secara multitasking menggunakan FreeRTOS. Tujuan utama sistem adalah menunjukkan penerapan konsep manajemen mikroprosesor yang mencakup manajemen proses, memori, komunikasi data, dan interupsi.

2. Alur Kerja Sistem

1. ESP32 melakukan booting dan membaca konfigurasi WiFi dari Preferences.
2. Jika WiFi belum dikonfigurasi, sistem masuk mode Access Point.
3. Jika WiFi tersedia, ESP32 terhubung ke jaringan.
4. Web server lokal menyediakan dashboard kontrol LED.
5. Tombol fisik memicu interrupt untuk mengubah brightness LED.
6. Task MQTT berjalan di background untuk mengirim status perangkat.
7. Software PWM mengatur intensitas LED secara real-time.

3. Diagram Blok Sistem

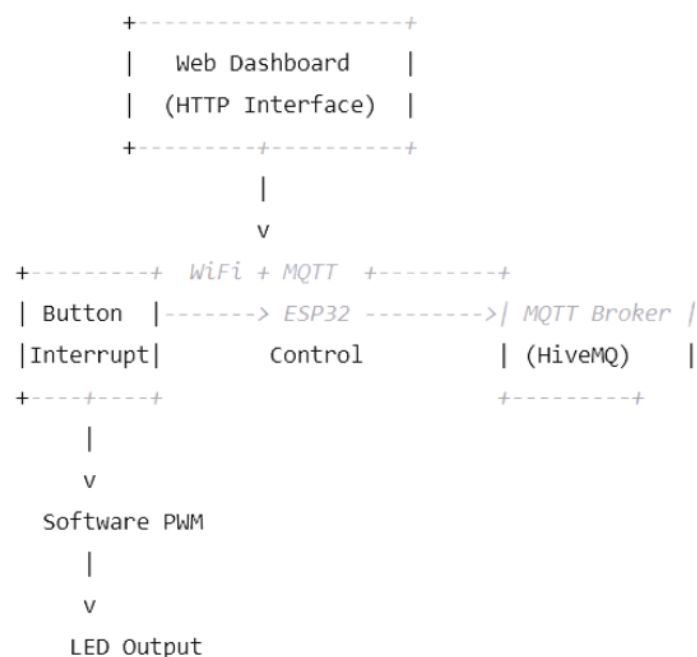


Diagram menunjukkan hubungan antara input (tombol), proses internal ESP32, komunikasi jaringan, dan output LED.

4. Manajemen Proses

Manajemen proses diterapkan menggunakan FreeRTOS task pada ESP32:

- Task utama menangani web server dan PWM
- Task MQTT berjalan paralel untuk komunikasi jaringan
- Interrupt handler menangani input tombol tanpa polling

Dengan multitasking, sistem tetap responsif meskipun menjalankan beberapa proses secara bersamaan.

Contoh potongan kode task:

```
xTaskCreatePinnedToCore(mqttTask, "mqtt", 4096, NULL, 1, NULL, 1);
```

Pendekatan ini menunjukkan pengelolaan CPU secara efisien melalui penjadwalan task.

5. Manajemen Memori

Konfigurasi WiFi disimpan menggunakan Preferences, yaitu memori non-volatile pada ESP32.

```
prefs.putString("ssid", ssid);  
prefs.putString("password", password);
```

Data tetap tersimpan meskipun perangkat dimatikan. Penggunaan memori persistent ini merupakan contoh manajemen memori dalam sistem embedded.

Variabel global digunakan secara terbatas untuk menjaga efisiensi RAM.

6. Manajemen Komunikasi

Sistem menggunakan dua metode komunikasi:

a. HTTP (Web Dashboard)

- Dashboard berbasis web untuk kontrol LED
- Komunikasi client-server lokal

b. MQTT

- Protokol publish/subscribe

- Pengiriman data status perangkat secara real-time
- Ringan dan efisien untuk IoT

Contoh publish data:

```
mqtt.publish(mqtt_topic, msg.c_str());
```

Komunikasi ini memungkinkan integrasi dengan sistem IoT eksternal.

7. Penggunaan Interupsi

Tombol fisik menggunakan mekanisme interrupt hardware:

```
attachInterrupt(BUTTON_PIN, handleButton, FALLING);
```

Interrupt memungkinkan sistem merespons input tanpa polling, sehingga lebih efisien dalam penggunaan CPU.

8. Kontrol Output dengan PWM

LED dikontrol menggunakan software PWM untuk mengatur brightness. Teknik ini mensimulasikan sinyal analog melalui switching digital.

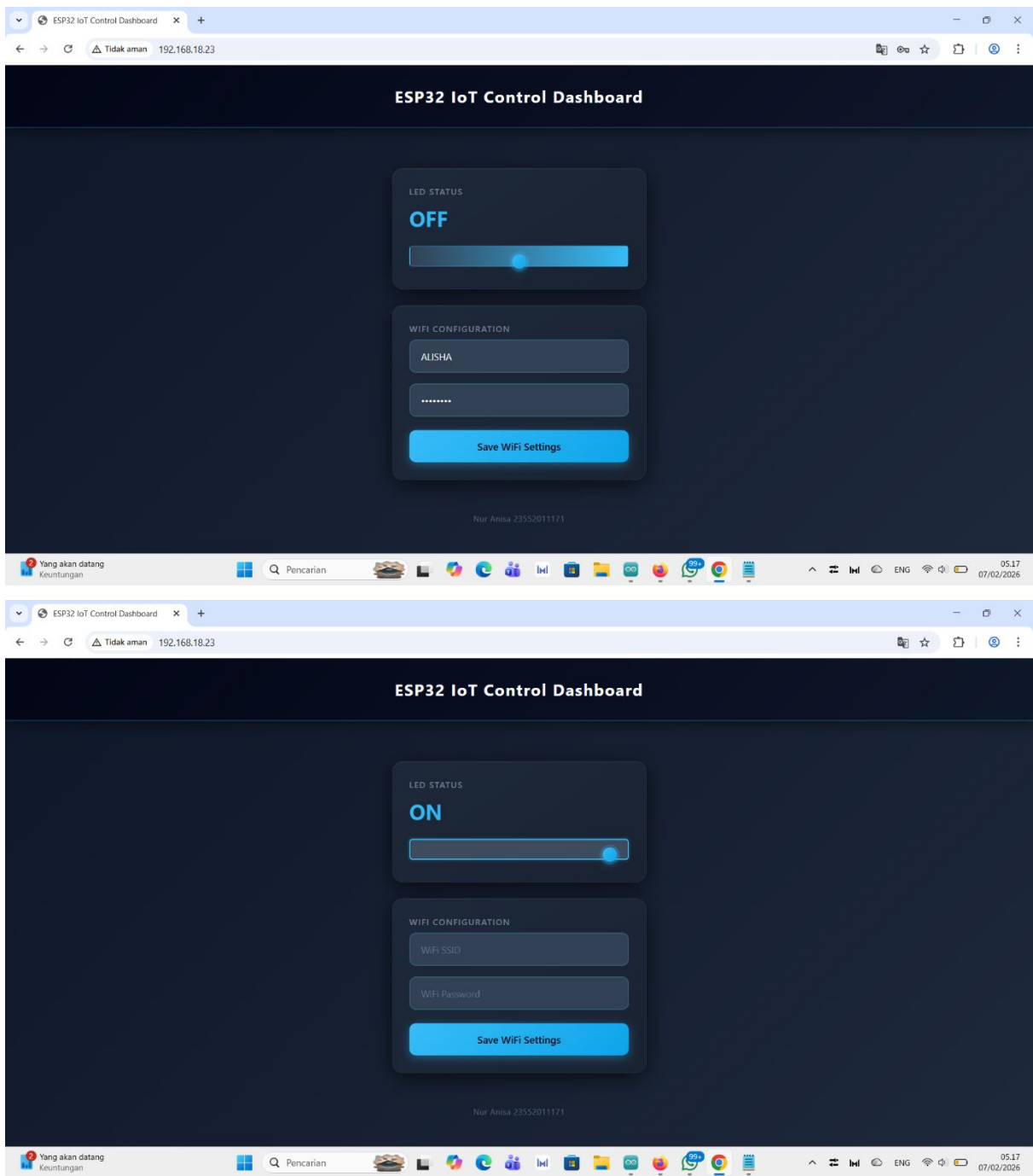
```
digitalWrite(LED_PIN, counter < brightness ? HIGH : LOW);
```

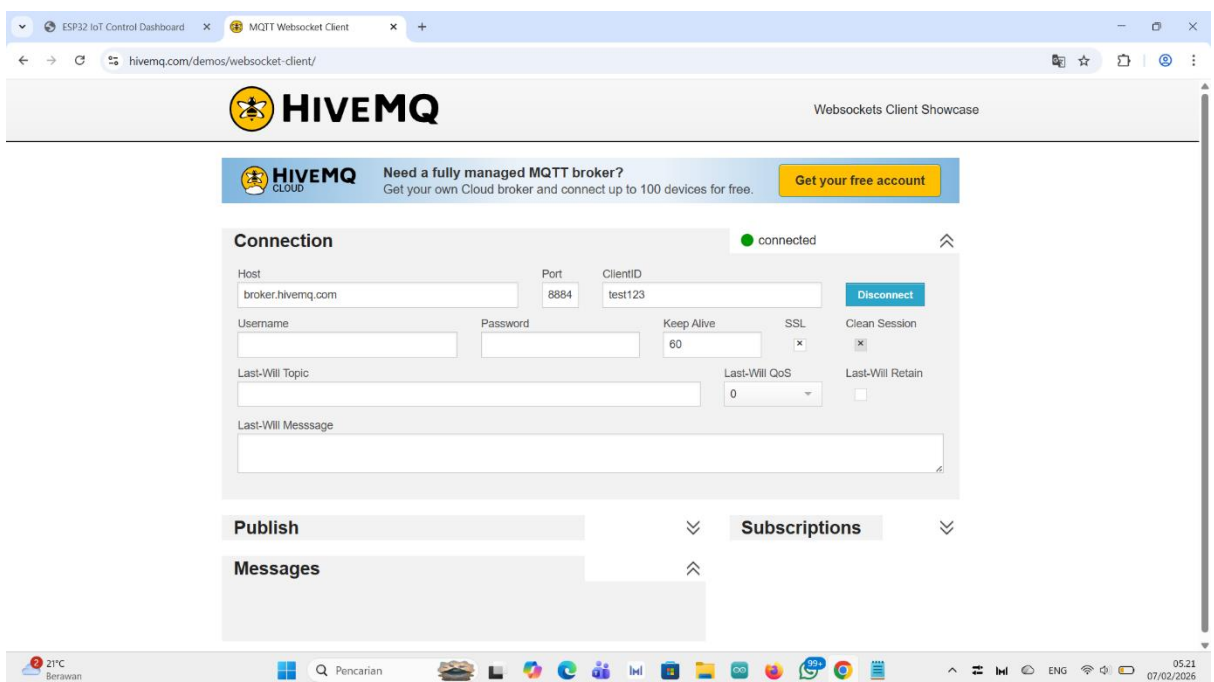
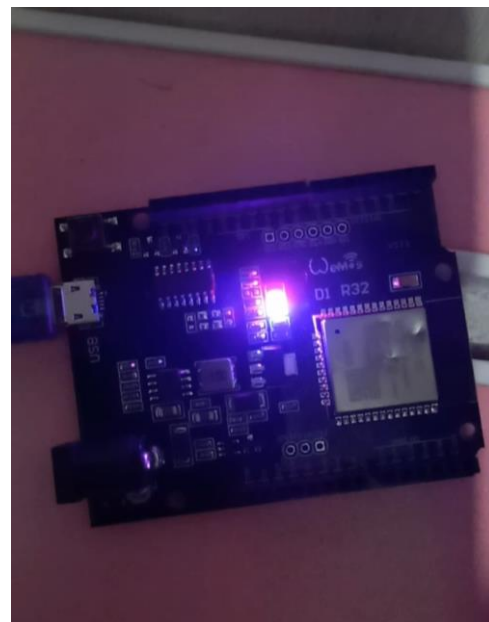
PWM menunjukkan kontrol presisi terhadap aktuator dalam sistem embedded.

9. Kesimpulan

Kesimpulan dari pengembangan sistem ini menunjukkan bahwa perangkat berhasil mengimplementasikan konsep utama manajemen mikroprosesor secara terpadu. Mekanisme interupsi digunakan untuk meningkatkan efisiensi pembacaan input, sementara multitasking berbasis FreeRTOS memungkinkan beberapa proses berjalan paralel tanpa mengganggu kinerja sistem. Penyimpanan konfigurasi menggunakan memori non-volatile memastikan data tetap tersimpan setelah perangkat dimatikan. Komunikasi jaringan dilakukan melalui protokol HTTP dan MQTT untuk mendukung kontrol dan monitoring jarak jauh, serta aktuator dikendalikan menggunakan teknik PWM untuk menghasilkan output yang presisi. Secara keseluruhan, sistem berjalan stabil, responsif, dan telah memenuhi ketentuan proyek embedded berbasis ESP32.

10. Lampiran





ESP32 IoT Control DashboardMQTT Websocket Client

hivemq.com/demos/websocket-client/

Publish

Messages

2026-02-07 05:22:02	Topic: esp32/dashboard	Qos: 0
{"led":false,"brightness":0}		
2026-02-07 05:22:00	Topic: esp32/dashboard	Qos: 0
{"led":false,"brightness":0}		
2026-02-07 05:21:58	Topic: esp32/dashboard	Qos: 0
{"led":false,"brightness":0}		
2026-02-07 05:21:56	Topic: esp32/dashboard	Qos: 0
{"led":false,"brightness":0}		
2026-02-07 05:21:54	Topic: esp32/dashboard	Qos: 0
{"led":false,"brightness":0}		
2026-02-07 05:21:52	Topic: esp32/dashboard	Qos: 0
{"led":true,"brightness":23}		
2026-02-07 05:21:50	Topic: esp32/dashboard	Qos: 0
{"led":true,"brightness":68}		
2026-02-07 05:21:48	Topic: esp32/dashboard	Qos: 0
{"led":true,"brightness":68}		
2026-02-07 05:21:46	Topic: esp32/dashboard	Qos: 0
{"led":true,"brightness":68}		
2026-02-07 05:21:44	Topic: esp32/dashboard	Qos: 0
{"led":true,"brightness":100}		

Subscriptions

Add New Topic Subscription

Qos: 2

esp32/dashboard

X

21°C
Berawan

Pencarian

05.22
07/02/2026