Perancangan dan Implementasi Sistem *Smart Lamp* Berbasis ESP32 dengan Integrasi WiFi dan MQTT

Diajukan untuk memenuhi Ujian Akhir Semester (UAS) Sistem Mikroprosessor.

Dosen Pengampu:

Muhammad Ikhwan Fathulloh, S.Kom.

DISUSUN OLEH:

Nama	Farhandhika Nurrohman
NPM	23552011203

TEKNIK INFORMATIKA KARYAWAN

UNIVERISTAS TEKNOLOGI BANDUNG

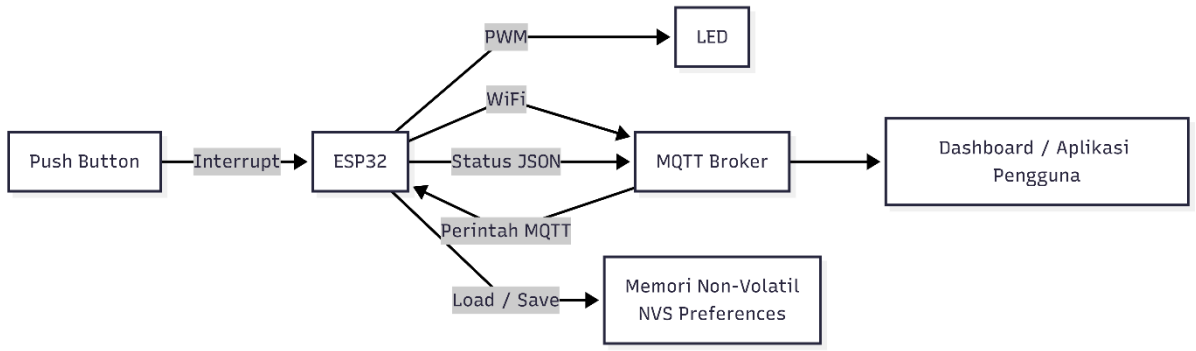
TAHUN 2026

1. Deskripsi Umum

Pada proyek ini, saya merancang dan mengimplementasikan sebuah sistem *Smart Lamp* berbasis Internet of Things (IoT) menggunakan mikroprosesor ESP32. Sistem ini memungkinkan pengendalian lampu dilakukan melalui dua metode, yaitu secara lokal menggunakan tombol fisik dan secara jarak jauh melalui aplikasi pada smartphone atau dashboard web.

Perancangan sistem memanfaatkan kemampuan manajemen mikroprosesor tingkat lanjut pada ESP32 untuk menghasilkan kinerja yang stabil, responsif, dan efisien, khususnya dalam menangani proses input, output, dan komunikasi jaringan secara bersamaan.

2. Diagram Blok Sistem



Sistem terdiri dari beberapa komponen utama sebagai berikut:

- **Input:** Push button sebagai masukan digital yang bekerja menggunakan mekanisme *external interrupt*.
- **Unit Proses:** ESP32 dual-core yang memanfaatkan *FreeRTOS* untuk manajemen tugas secara paralel.
- **Penyimpanan Data:** Memori non-volatil (NVS) menggunakan library **Preferences**.
- **Output:** LED yang dikendalikan melalui sinyal **PWM (Pulse Width Modulation)**.
- **Komunikasi:** Koneksi WiFi dalam mode *station* dan protokol MQTT dengan model *publish/subscribe*.

3. Implementasi Manajemen Mikroprosesor

a. Manajemen Proses (Multitasking dengan FreeRTOS)

Untuk menghindari terjadinya *blocking*, sistem dibagi menjadi dua tugas utama yang berjalan pada core yang berbeda. Core pertama difokuskan pada pengolahan perangkat keras, sedangkan core kedua menangani komunikasi jaringan. Dengan pendekatan ini, proses input dan output tetap berjalan secara real-time meskipun terjadi aktivitas jaringan.

b. Manajemen Input dan Output (Interrupt dan PWM)

Dalam sistem ini, saya tidak menggunakan metode *polling*, melainkan memanfaatkan *external interrupt* dengan mode **FALLING** untuk membaca input tombol. Pendekatan ini membuat penggunaan CPU lebih efisien dan respons sistem menjadi lebih cepat.

Output LED dikendalikan menggunakan PWM 8-bit dengan rentang nilai 0–255, sehingga tingkat kecerahan dapat diatur secara halus dan menghasilkan efek *soft dimming* yang nyaman secara visual.

c. Manajemen Memori (Preferences / NVS)

Sistem memanfaatkan library **Preferences** untuk menyimpan nilai kecerahan lampu ke dalam memori flash ESP32. Dengan cara ini, data tetap tersimpan meskipun perangkat dimatikan, sehingga ketika sistem dinyalakan kembali, lampu akan langsung berada pada tingkat kecerahan terakhir.

d. Komunikasi Data (MQTT dan JSON)

Pertukaran data antara ESP32 dan aplikasi dilakukan menggunakan protokol MQTT dengan format data JSON. ESP32 mengirimkan informasi status dan tingkat kecerahan lampu ke broker MQTT, serta menerima perintah pengaturan kecerahan dari aplikasi pengguna secara real-time.

4. Algoritma Sistem

Alur kerja sistem dapat dijelaskan sebagai berikut:

1. Sistem melakukan inisialisasi perangkat, memuat data dari memori non-volatil, serta mengatur PWM dan interrupt.
2. ESP32 menghubungkan diri ke jaringan WiFi dan broker MQTT.

3. Ketika tombol ditekan, interrupt akan memperbarui nilai target kecerahan.
4. Jika terdapat pesan MQTT dari perangkat pengguna, sistem akan menyesuaikan nilai kecerahan sesuai perintah.
5. Algoritma *soft dimming* mengatur perubahan kecerahan secara bertahap.
6. Sistem secara berkala mengirimkan data status terbaru ke dashboard atau aplikasi pengguna.

5. Cuplikan Kode Utama

a. Manajemen Proses (Multitasking dengan FreeRTOS)

Pada sistem yang dikembangkan, saya menerapkan mekanisme multitasking dengan memanfaatkan **FreeRTOS** serta arsitektur **dual-core** pada ESP32. Beban kerja sistem dibagi ke dalam dua *task* utama yang dijalankan secara paralel dan dipaku (*pinned*) pada core yang berbeda.

Pendekatan ini bertujuan untuk menghindari *resource contention* serta memastikan bahwa proses pengolahan perangkat keras dan komunikasi jaringan dapat berjalan secara independen tanpa saling mengganggu. Dengan demikian, sistem tetap responsif meskipun terdapat aktivitas jaringan yang intensif.

```
    xTaskCreatePinnedToCore(taskHardware, "HW_Task", 2048, NULL, 1, NULL, 1);
    xTaskCreatePinnedToCore(taskNetwork, "Net_Task", 4096, NULL, 1, NULL, 0);
```

b. Manajemen Input dan Output (Interupsi dan PWM)

Dalam menangani input tombol, sistem tidak menggunakan metode *polling*, melainkan memanfaatkan **external interrupt**. Metode ini memungkinkan CPU tetap menjalankan proses lain sambil menunggu adanya peristiwa dari perangkat input, sehingga penggunaan sumber daya menjadi lebih efisien.

Untuk pengendalian output, intensitas cahaya LED diatur menggunakan sinyal **PWM**, yang memungkinkan pengaturan kecerahan secara bertahap dan halus sesuai dengan nilai yang ditentukan oleh sistem.

1) Inisialisasi interupsi tombol

```
attachInterrupt(digitalPinToInterrupt(PIN_BUTTON), handleButton, FALLING);
```

2) Output PWM untuk pengaturan kecerahan LED

```
ledcWrite(PIN_LED, currentPWM);
```

c. Manajemen Memori (Non-Volatile Storage)

Sistem juga menerapkan manajemen memori non-volatil untuk menjaga keberlanjutan data (*persistence*). Library **Preferences** digunakan untuk menyimpan nilai kecerahan terakhir ke dalam memori flash ESP32. Dengan mekanisme ini, data tetap tersimpan meskipun terjadi pemadaman daya, dan dapat dimuat kembali saat sistem dinyalakan ulang.

1) Membuka namespace NVS

```
pref.begin("lamp_data", false);
```

2) Menyimpan data kecerahan

```
pref.putInt("last_bright", targetBrightness);
```

3) Memuat data saat boot

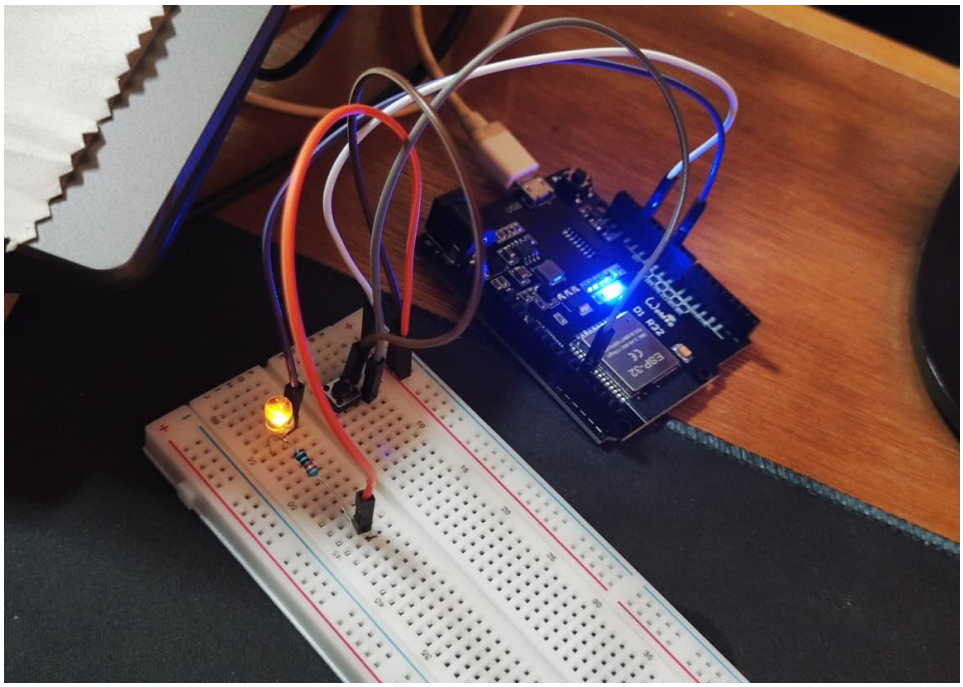
```
targetBrightness = pref.getInt("last_bright", 0);
```

6. Hasil Pengujian

Berdasarkan pengujian yang dilakukan, sistem menunjukkan kinerja yang stabil. Input tombol dapat direspon secara instan meskipun koneksi jaringan sedang tidak stabil. Data status lampu berhasil ditampilkan secara real-time pada aplikasi MQTT di smartphone, dengan latensi komunikasi rata-rata kurang dari 200 ms. Selain itu, transisi kecerahan LED berjalan halus sesuai dengan perancangan PWM.

7. Lampiran

a. Rangkaian sistem ESP32 yang digunakan



b. Tampilan status di dashboard aplikasi mymqtt

c. Hasil dari serial monitor

```
d. ============================================
e.         IOT SMART LAMP - MIKROPROSESOR
f.    DEVELOPED BY: FARHANDHIKA NURROHMAN (ESP32)
g. ============================================
h. [SYSTEM] Chip Model: ESP32-D0WD-V3
i. [SYSTEM] CPU Cores : 2
j. [SYSTEM] Free Heap : 245 KB
k. [MEMORI] Last Brightness: 0
l. [INPUT] External Interrupt Initialized.
m. [SYSTEM] Multitasking Kernel Running...
n.
o. [!] WIFI TERPUTUS
p. [WIFI] Menghubungkan kembali ke: jco&family
q. ============================================
r.
s. ........
t.
u. >>> WIFI CONNECTED INFO <<<
v. SSID      : jco&family
w. IP Address: 192.168.100.197
x. MAC Addr  : 44:1D:64:F6:19:9C
y. Signal    : -65 dBm
z. --------------------------
aa.
bb.  [MQTT] Status: Disconnected. Reconnecting...
cc.  >>> MQTT BROKER CONNECTED <<<
dd.  Broker    : broker.emqx.io
ee.  Client ID : ESP32_Farhan_4593
ff.  Subscribed: esp32/smartlamp/control
gg.  --------------------------
hh.
ii.  [BUTTON] Tombol Ditekan!
jj.  [SYSTEM] Target Brightness Baru: 153
kk.
ll.  [BUTTON] Tombol Ditekan!
mm.  [SYSTEM] Target Brightness Baru: 204
nn.
oo.  [BUTTON] Tombol Ditekan!
pp.  [SYSTEM] Target Brightness Baru: 255
qq.
rr.  [BUTTON] Tombol Ditekan!
ss.  [SYSTEM] Target Brightness Baru: 0
tt.
uu.  [MQTT] Pesan Diterima di Topik: esp32/smartlamp/control
vv.  [MQTT] Isi Pesan: 153
ww.  [SYSTEM] Brightness diatur via Remote: 153
xx.
yy.  [MQTT] Pesan Diterima di Topik: esp32/smartlamp/control
zz.  [MQTT] Isi Pesan: 0
aaa. [SYSTEM] Brightness diatur via Remote: 0
bbb.
ccc. [MQTT] Pesan Diterima di Topik: esp32/smartlamp/control
ddd. [MQTT] Isi Pesan: 255
eee. [SYSTEM] Brightness diatur via Remote: 255
fff.
ggg. [MQTT] Pesan Diterima di Topik: esp32/smartlamp/control
hhh. [MQTT] Isi Pesan: 15
iii. [SYSTEM] Brightness diatur via Remote: 15
```

8. **Link Github dan Vidio Demo**

- **Link Github:**
  https://github.com/Farhandhika-N/UAS_Sistem-Mikroprosessor_Smart-Lamp-ESP32_23552011203_Farhandhika-Nurrohman.git

- **Link Vidio Demo:**
  https://youtu.be/bhtpUIJQhmc