LAPORAN MIKROKONTROLER

Dosen Pengampu :

Dr. Basuki Rahmat, S.Si.,M.T.

Disusun oleh :

Amelia Ananda Putri Lestari (23081010051)

PROGRAM STUDI INFORMATIKA

FAKULTAS ILMU KOMPUTER

UNIVERSITAS PEMBANGUNAN NASIONAL "VETERAN" JAWA TIMUR

2025

BAB I
PENDAHULUAN

1.1 Latar Belakang

Praktikum ini digunakan protokol Message Queuing Telemetry Transport (MQTT) sebagai media komunikasi antara perangkat ESP32 dan server broker MQTT . MQTT merupakan protokol komunikasi berbasis publish-subscribe yang dirancang khusus untuk sistem dengan keterbatasan sumber daya, baik dari sisi daya, memori, maupun bandwidth jaringan. Oleh karena itu, MQTT sangat cocok diterapkan pada perangkat Internet of Things (IoT) seperti ESP32

Pengendalian motor DC secara konvensional umumnya dilakukan menggunakan saklar mekanis, potensiometer, atau pengontrol lokal yang terhubung langsung dengan perangkat. Metode ini memiliki beberapa batasan, antara lain jarak kendali yang terbatas, ketidakpuasan yang buruk, serta tidak tersedianya fasilitas pemantauan dan pengendalian jarak jauh. Selain itu, sistem konvensional sulit diintegrasikan dengan sistem otomasi modern yang berbasis jaringan internet.Seiring dengan berkembangnya teknologi Internet of Things (IoT) , kebutuhan akan sistem kendali yang dapat diakses dan dipantau dari jarak jauh semakin meningkat. Sistem otomasi modern menuntut adanya kendali yang bersifat real-time , mudah dikonfigurasi, serta dapat diintegrasikan dengan berbagai platform dan perangkat lain melalui jaringan internet. Oleh karena itu, diperlukan sebuah solusi pengendalian motor yang tidak hanya efisien, tetapi juga adaptif terhadap perkembangan teknologi digital.

ESP32 merupakan salah satu mikrokontroler yang banyak digunakan dalam pengembangan sistem IoT karena memiliki spesifikasi yang mumpuni. ESP32 dilengkapi dengan modul WiFi dan Bluetooth terintegrasi, kecepatan pemrosesan yang tinggi, serta konsumsi daya yang relatif rendah. Hal ini menjadikan ESP32 sangat cocok digunakan sebagai kendali pusat pada sistem berbasis IoT. Dengan mengombinasikan ESP32 , protokol MQTT , dan teknik Pulse Wide Modulation (PWM) ,sistem kendali motor DC dapat dioperasikan secara jarak jauh dengan pengaturan kecepatan yang presisi. PWM memungkinkan pengaturan kecepatan motor dengan cara mengubah lebar sinyal pulsa, sehingga motor dapat dikontrol secara halus dan efisien. Sementara itu, MQTT berperan sebagai media komunikasi yang menjembatani perintah dari pengguna menuju perangkat ESP32.

Melalui sistem ini, pengguna dapat mengendalikan kecepatan dan status motor DC secara nirkabel melalui jaringan internet, sekaligus membuka peluang pengembangan lebih lanjut, seperti integrasi dengan aplikasi berbasis web atau mobile, pemantauan sistem, serta otomatisasi industri skala kecil hingga menengah.

1.2 Tujuan Praktikum

1. Merancang sistem kendali motor DC berbasis Internet of Things (IoT) yang mampu dikendalikan
2. Mengimplementasikan protokol MQTT sebagai media komunikasi antara perangkat ESP32 dan broker
3. Menerapkan teknik Pulse width Modulation (PWM) untuk mengatur kecepatan putaran motor DC
4. Menguji kinerja sistem kendali motor DC , meliputi kestabilan koneksi WiFi, keberhasilan komunikasi
5. Menganalisis efektivitas penggunaan ESP32 dan MQTT dalam sistem kendali motor DC jarak jauh

BAB II
PENJELASAN PRAKTIKUM

1.2 Penjelasan Program Praktikum

Gambar 1

ESP32_MQTT_Motor_Control.ino.ino

```
31    while (WiFi.status() != WL_CONNECTED) {
32      digitalWrite(LED_WIFI, HIGH);
33      delay(300);
34      digitalWrite(LED_WIFI, LOW);
35      delay(300);
36      Serial.print(".");
37    }
38
39    digitalWrite(LED_WIFI, HIGH);
40    Serial.println("\nWiFi Connected");
41    Serial.print("IP: ");
42    Serial.println(WiFi.localIP());
43  }
44
45  /* ================= MQTT CALLBACK ================= */
46  void callback(char* topic, byte* payload, unsigned int length) {
47    String msg = "";
48
49    for (int i = 0; i < length; i++) {
50      msg += (char)payload[i];
51    }
52
53    int speed = constrain(msg.toInt(), 0, 255);
54
55    if (speed > 0) {
56      digitalWrite(MOTOR_IN1, LOW);
57      digitalWrite(MOTOR_IN2, HIGH);
58      ledcWrite(pwmChannel, speed);
59
60      Serial.print("Motor ON | Speed: ");
```

Program ini digunakan untuk mengendalikan motor DC menggunakan ESP32 melalui jaringan internet dengan protokol MQTT. ESP32 terhubung ke WiFi, menerima pesan MQTT berupa nilai kecepatan motor, lalu mengolahnya menjadi sinyal PWM dengan jarak 0–255. Jika nilai lebih

dari nol, motor menyala dan kecepatannya diatur sesuai pesan yang diterima, sedangkan jika bernilai nol motor akan mati. Sistem ini bekerja secara real-time dan cocok untuk aplikasi IoT.

Gambar 2

ESP32_MQTT_Motor_Control.ino.ino

```
#include <WiFi.h>
#include <PubSubClient.h>

/* ================ PIN ================ */
#define LED_WIFI 2        // LED onboard ESP32
#define MOTOR_IN1 27
#define MOTOR_IN2 26
#define MOTOR_EN  12

/* ================ WIFI ================ */
const char* ssid     = "realme 10";
const char* password = "bdaazqqae";

/* ================ MQTT ================ */
const char* mqtt_server = "broker.hivemq.com";
const char* mqtt_topic  = "iot/esp32/motor";

/* ================ PWM ================ */
const int pwmChannel = 0;
const int pwmFreq    = 30000;
const int pwmRes     = 8;

WiFiClient espClient;
PubSubClient client(espClient);

/* ================ WIFI SETUP ================ */
void setupWiFi() {
  WiFi.begin(ssid, password);
  Serial.print("Connecting WiFi");
```

Kode program ini merupakan program ESP32 untuk mengontrol motor DC berbasis IoT menggunakan koneksi WiFi dan protokol MQTT. Program diawali dengan pemanggilan library `WiFi.h` untuk menghubungkan ESP32 ke jaringan WiFi dan `PubSubClient.h` untuk komunikasi MQTT. Selanjutnya dilakukan pendefinisian pin, yaitu pin LED sebagai indikator status WiFi, dua pin input motor untuk mengatur arah putaran, serta satu pin enable motor yang

digunakan sebagai PWM untuk mengontrol kecepatan motor. ESP32 dikonfigurasikan agar terhubung ke jaringan WiFi dengan SSID dan password tertentu, kemudian berkomunikasi dengan broker MQTT publik `broker.hivemq.com` melalui topik `iot/esp32/motor`. Pengaturan PWM meliputi channel, frekuensi, dan resolusi yang berfungsi untuk mengatur kecepatan motor secara halus. Objek `WiFiClient` dan `PubSubClient` digunakan sebagai media komunikasi antara ESP32 dan broker MQTT. Fungsi `setupWiFi()` berperan untuk menghubungkan ESP32 ke jaringan WiFi serta menampilkan status koneksi melalui Serial Monitor.

Gambar 3

ESP32_MQTT_Motor_Control.ino.ino

```
      delay(2000);
    }
  }
}

/* ================ SETUP ================ */
void setup() {
  Serial.begin(115200);

  pinMode(LED_WIFI, OUTPUT);
  pinMode(MOTOR_IN1, OUTPUT);
  pinMode(MOTOR_IN2, OUTPUT);
  pinMode(MOTOR_EN, OUTPUT);

  ledcSetup(pwmChannel, pwmFreq, pwmRes);
  ledcAttachPin(MOTOR_EN, pwmChannel);

  setupWiFi();

  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
}

/* ================ LOOP ================ */
void loop() {
  if (!client.connected()) {
    reconnectMQTT();
  }
  client.loop();
}
```

kode ini merupakan fungsi `setup()` dan `loop()` pada program ESP32. Pada fungsi `setup()`, komunikasi serial diinisialisasi untuk keperluan monitoring, kemudian seluruh pin seperti LED WiFi, pin arah motor, dan pin enable motor diatur sebagai output. Selanjutnya PWM dikonfigurasi menggunakan `ledcSetup()` dan dihubungkan ke pin motor enable agar kecepatan motor dapat dikontrol. Setelah itu, ESP32 dihubungkan ke jaringan WiFi melalui fungsi `setupWiFi()`, lalu dilakukan pengaturan server MQTT dan callback untuk menerima pesan. Pada fungsi `loop()`, program terus memeriksa koneksi ke broker MQTT, melakukan reconnect jika terputus, serta menjalankan `client.loop()` agar ESP32 dapat menerima dan memproses pesan MQTT secara terus-menerus.

Gambar 4

```
61      Serial.println(speed);
62    } else {
63      digitalWrite(MOTOR_IN1, LOW);
64      digitalWrite(MOTOR_IN2, LOW);
65      ledcWrite(pwmChannel, 0);
66
67      Serial.println("Motor OFF");
68    }
69  }
70
71  /* ================ MQTT RECONNECT ================ */
72  void reconnectMQTT() {
73    while (!client.connected()) {
74      Serial.print("Connecting MQTT... ");
75
76      String clientId = "ESP32-" + String(random(0xffff), HEX);
77      if (client.connect(clientId.c_str())) {
78        Serial.println("Connected");
79        client.subscribe(mqtt_topic);
80        Serial.print("Subscribe: ");
81        Serial.println(mqtt_topic);
82      } else {
83        Serial.print("Failed, retry...");
84        delay(2000);
85      }
86    }
87  }
88
89  /* ================ SETUP ================ */
90  void setup() {
```

Bagian kode ini berfungsi untuk mengontrol kondisi motor dan menangani koneksi ulang MQTT. Jika perintah yang diterima sesuai, motor dijalankan dengan kecepatan tertentu dan nilainya ditampilkan pada Serial Monitor. Sebaliknya, jika tidak ada perintah atau perintah bernilai mati, maka kedua pin motor diset LOW dan nilai PWM diatur ke nol sehingga motor berhenti, serta ditampilkan pesan "Motor OFF". Selanjutnya, fungsi `reconnectMQTT()` digunakan untuk memastikan ESP32 selalu terhubung ke broker MQTT. Fungsi ini akan mencoba menyambung kembali ke broker jika koneksi terputus, menampilkan status koneksi melalui Serial Monitor, membuat client ID secara acak, dan setelah berhasil terhubung ESP32 akan berlangganan ke topik MQTT yang telah ditentukan.

BAB 3
HASIL PEMBEHASAN

Program ESP32_MQTT_Motor_Control dirancang untuk mengendalikan motor DC secara jarak jauh menggunakan teknologi Internet of Things (IoT) melalui koneksi WiFi dan protokol MQTT. Pada tahap awal, ESP32 dikonfigurasi dengan library WiFi.h dan PubSubClient.h untuk memungkinkan koneksi ke jaringan WiFi serta komunikasi dengan broker MQTT. Penentuan pin dilakukan untuk LED indikator WiFi, pin arah motor, dan pin enable motor yang berfungsi sebagai pengatur kecepatan berbasis PWM. Pengaturan PWM menggunakan channel tertentu dengan frekuensi 30 kHz dan resolusi 8 bit, sehingga kecepatan motor dapat dikontrol secara halus.

ESP32 terhubung ke jaringan WiFi menggunakan SSID dan password yang telah ditentukan. Selama proses koneksi, LED WiFi akan berkedip sebagai indikator, dan setelah berhasil terhubung LED akan menyala stabil serta alamat IP ditampilkan pada Serial Monitor. Setelah koneksi WiFi berhasil, ESP32 melakukan koneksi ke broker MQTT publik broker.hivemq.com pada port 1883 dan berlangganan ke topik iot/esp32/motor untuk menerima pesan kontrol. Pesan MQTT yang diterima akan diproses melalui fungsi callback. Nilai pesan dikonversi menjadi bilangan bulat yang merepresentasikan kecepatan motor dalam rentang 0–255. Jika nilai kecepatan lebih dari nol, motor akan berputar dengan arah tertentu dan kecepatan sesuai nilai PWM yang diberikan. Sebaliknya, jika nilai yang diterima adalah nol atau tidak valid, maka motor akan berhenti dengan mematikan kedua pin arah dan mengatur PWM ke nol. Informasi status motor dan nilai kecepatan ditampilkan melalui Serial Monitor untuk memudahkan pemantauan.

Selain itu, sistem dilengkapi dengan mekanisme reconnect MQTT yang memastikan ESP32 tetap terhubung dengan broker. Jika koneksi terputus, ESP32 akan mencoba menyambung kembali secara otomatis hingga berhasil dan kembali berlangganan ke topik MQTT. Pada fungsi loop(), program secara terus-menerus memeriksa status koneksi MQTT dan menjalankan client.loop() agar pesan dapat diterima dan diproses secara real-time. Dengan demikian, sistem ini mampu memberikan kontrol motor DC yang stabil, responsif, dan dapat diandalkan melalui jaringan internet.