Logbook 1
Penerapan Kontrol Motor DC Pada iMCLab

Di logbook kedua ini, saya mempelajari penerapan kontrol motor DC pada iMCLab menggunakan ESP32, dengan tujuan untuk mempelajari cara mengatur arah dan kecepatan motor DC melalui PWM (Pulse Width Modulation) dan pengendalian sinyal digital.

Contoh kode:

```
int motor1Pin1 = 27;
int motor1Pin2 = 26;
int enable1Pin = 12;

// Setting PWM properties
const int freq = 30000;
const int pwmChannel = 0;
const int resolution = 8;
int dutyCycle = 200;

void setup () {
    // sets the pins as outputs:
    pinMode (motor1Pin1, OUTPUT) ;
    pinMode (motor1Pin2, OUTPUT);
    pinMode (enable1Pin, OUTPUT);

    // configure LED PWM functionalitites
    ledcSetup (pwmChannel, freq, resolution) ;

    // attach the channel to the GPIO to be controlled
    ledcAttachPin (enable1Pin, pwmChannel) ;

    Serial.begin(115200);

    // testing
    Serial.print ("Testing DC Motor ... ");
}

void loop() {
    // Move the DC motor forward at maximum speed
    Serial.println ("Moving Forward") ;
    digitalWrite(motor1Pin1, LOW) ;
    digitalWrite(motor1Pin2, HIGH);
    delay (2000) ;

    // stop the DC motor
    Serial.println("Motor stopped") ;
```

```
    digitalWrite(motor1Pin1, LOW);
    digitalWrite(motor1Pin2, LOW);
    delay (1000) ;

    // Move DC motor backwards at maximum speed
    Serial.println ("Moving Backwards");
    digitalWrite(motor1Pin1, HIGH);
    digitalWrite(motor1Pin2, LOW);
    delay (2000) ;

    // Stop the DC motor
    Serial.println("Motor stopped");
    digitalWrite(motor1Pin1, LOW);
    digitalWrite(motor1Pin2, LOW);
    delay (1000) ;

    // Move DC motor forward with increasing speed
    digitalWrite(motor1Pin1, HIGH);
    digitalWrite(motor1Pin2, LOW);

    while (dutyCycle <= 255) {
        ledcWrite (pwmChannel, dutyCycle) ;
        Serial.print ("Forward with duty cycle: ");
        Serial.println (dutyCycle);
        dutyCycle = dutyCycle + 5;
        delay (500) ;
    }

    dutyCycle = 200;
}
```

Analisis

Kode ini mengatur tiga pin utama pada ESP32: dua pin digital untuk mengontrol arah putaran motor (motor1Pin1 dan motor1Pin2) serta satu pin PWM (enable1Pin) untuk mengatur kecepatannya. Ketiga pin diinisialisasi sebagai output dalam setup(), dan konfigurasi PWM dilakukan dengan frekuensi 30 kHz dan resolusi 8-bit menggunakan ledcAttachPin(). Pada loop(), motor dijalankan dalam beberapa mode: maju dengan kecepatan maksimum, berhenti, mundur, berhenti lagi, lalu maju dengan kecepatan bertahap menggunakan penyesuaian duty cycle melalui ledcWrite(). Selama proses berjalan, Serial.println() digunakan untuk mencetak status motor ke Serial Monitor, yang sangat membantu dalam pemantauan dan debugging. Implementasi ini menunjukkan cara efektif mengontrol arah dan kecepatan motor DC menggunakan PWM pada ESP32.