Teknoloji Fakültesi

MRM4002 – Endüstriyel Haberleşme ve SCADA

DENEY 4 – ESP8266 ile Motor veya Araç

Kontrolü deneyisi

Laboratuvar Raporu

Öğrenci Abdelwahab-Rayis 170219922

1. GİRİŞ

Deneyin Amacı

ESP8266 ile Motor veya Araç Kontrolü yapmak

Giriş

Bu rapor, bir DC motoru web istekleri aracılığıyla kontrol etmek için tasarlanmış bir ESP32 tabanlı projenin uygulamasını ve test edilmesini sunar. Proje, kablosuz ağ üzerinden motor kontrol komutlarını alıp işleyen iki ESP32 Geliştirme Modülü kullanır. Birinci ESP32 modülü bir istemci olarak işlev görür ve HTTP GET istekleri gönderirken, ikinci ESP32 modülü bir sunucu olarak işlev görür ve bu istekleri işleyerek motoru kontrol eder.

Kullanılan Bileşenler

- İki adet ESP32
- Breadboard
- DC Motor
- L298N H-Köprüsü Motor Sürücü Modülü
- İki adet Buton
- Jumper kabloları
- Arduino IDE 2.3.2 yüklü dizüstü bilgisayar

Amaçlar

- ESP32 modüllerini bir Wi-Fi ağına bağlamak.
- Bir ESP32 modülünü DC motoru kontrol etmek için HTTP sunucusu olarak yapılandırmak.
- Bir butona basıldığında motor kontrol komutları gönderen istemci modülü uygulamak.

2. DENEY YÖNTEMİ

Donanım Kurulumu

Donanım kurulumu, ekli fotoğraflarda gösterilmektedir. Bağlantılar şu şekildedir:

- Butonlar İstemci (emir veren) ESP32 modülünün 13 ve 14 numaralı pinlerine bağlanmıştır.
- Modülü, ESP32 modülü , L298N H-Köprüsü Motor Sürücüye (en = 14 , pin1 = 27, pin2 =26) ve DC motoruna bağlanmıştır.
- Breadboard üzerindeki bağlantılar jumper kabloları ile sağlanmıştır.

Kütüphaneler

Projede kullanılan kütüphaneler şunlardır:

- WiFi.h: ESP32 modülünü Wi-Fi ağına bağlamak için kullanılır.
- HTTPClient.h: HTTP istemci işlevselliği sağlar ve ESP32'nin HTTP GET ve POST istekleri göndermesine olanak tanır.
- ESPAsyncWebServer.h: ESP32 üzerinde asenkron web sunucusu oluşturmak için kullanılır. Bu, HTTP isteklerini asenkron olarak işleyerek daha hızlı ve verimli bir sunucu uygulaması sağlar.

KODLAR

1. ESP32 inin İstemci Kodu (Kontrol Modülü) :-

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

const char* ssid = "HUAWEI P30 lite";
const char* password = "00000000";

int buttonPin1 = 13;
int buttonPin2 = 14;

// Server IP address
const char* serverIP = "192.168.43.245"; // استخدم عنوان IP الفعلي للوحدة المتصلة بالمحرك

void setup() {
  Serial.begin(115200);

  pinMode(buttonPin1, INPUT_PULLUP);
  pinMode(buttonPin2, INPUT_PULLUP);
```

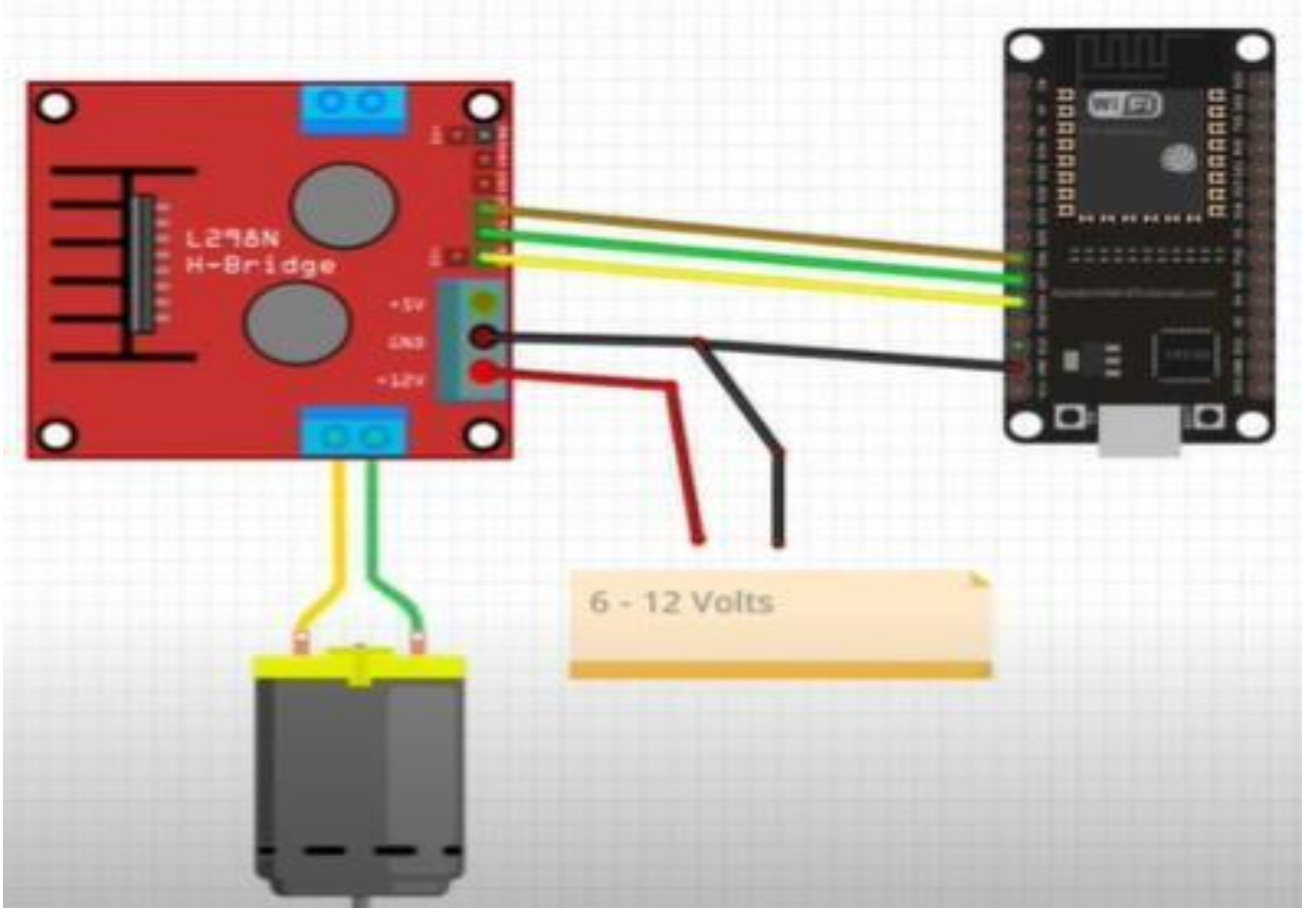
```
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi...");
  }

  Serial.println("Connected to WiFi");
  Serial.println(WiFi.localIP());
}

void loop() {
  bool button1State = digitalRead(buttonPin1) == LOW;
  bool button2State = digitalRead(buttonPin2) == LOW;
  String command = "S";

  if (button1State && button2State) {
    command = "S";
  } else if (button1State) {
    command = "F";
  } else if (button2State) {
    command = "B";
  }

  sendCommand(command);
  delay(200);
}

void sendCommand(String command) {
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;
    String url = String("http://") + serverIP + "/control?cmd=" + command;
    http.begin(url);
    int httpResponseCode = http.GET();
    if (httpResponseCode > 0) {
      String response = http.getString();
      Serial.println(response);
    } else {
      Serial.print("Error on sending GET: ");
      Serial.println(httpResponseCode);
    }
    http.end();
  }
}
```

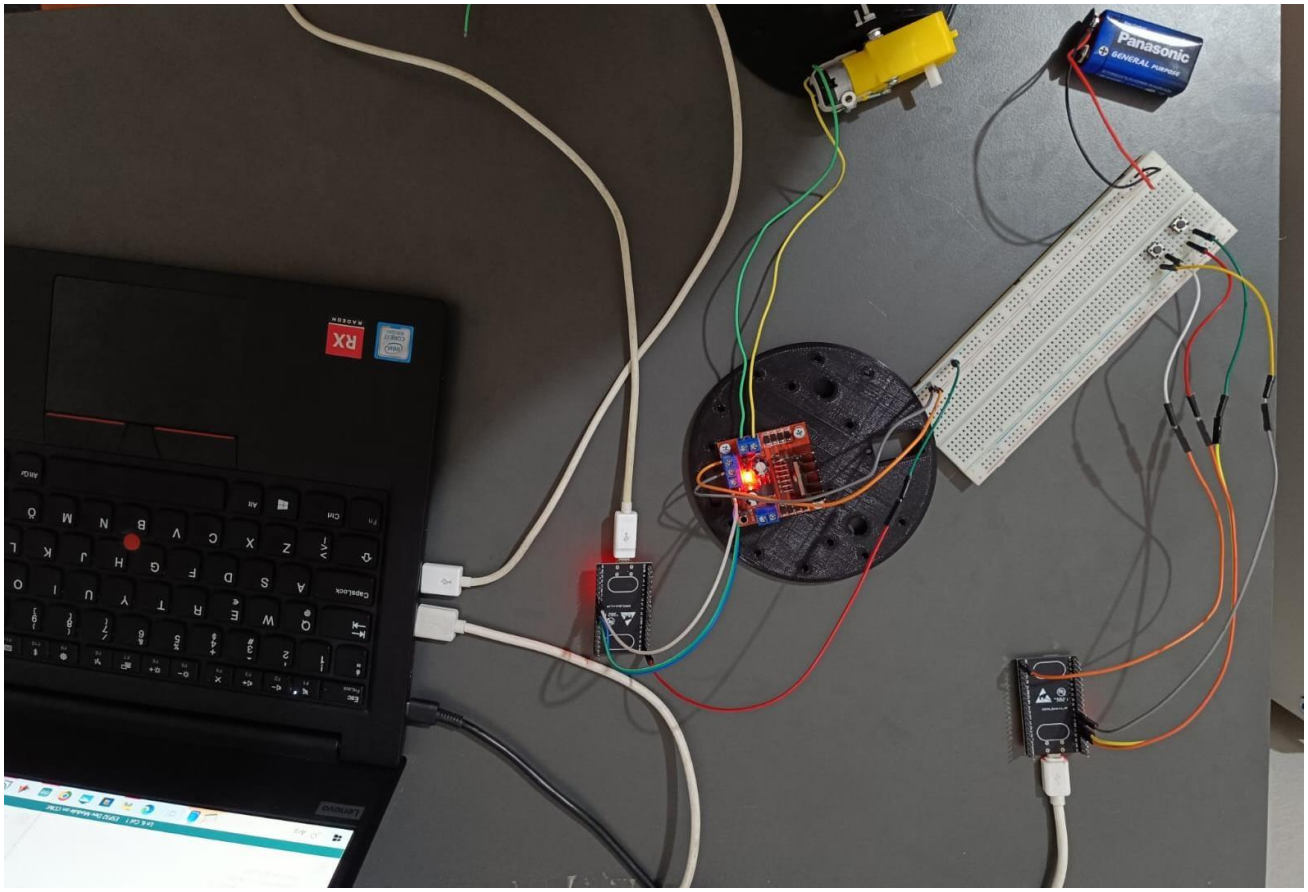
2. ESP32 inin Sürücü Kodu (Motor Kontrol Modülü) :-

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

const char* ssid = "HUAWEI P30 lite";
const char* password = "00000000";

int motor1Pin1 = 27;
int motor1Pin2 = 26;
int enable1Pin = 14;

const int freq = 30000;
const int pwmChannel = 0;
const int resolution = 8;
int dutyCycle = 0;

AsyncWebServer server(80);

void setup() {
  Serial.begin(115200);

  pinMode(motor1Pin1, OUTPUT);
  pinMode(motor1Pin2, OUTPUT);
  pinMode(enable1Pin, OUTPUT);

  ledcSetup(pwmChannel, freq, resolution);
  ledcAttachPin(enable1Pin, pwmChannel);
  ledcWrite(pwmChannel, dutyCycle);

  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi...");
  }

  Serial.println("Connected to WiFi");
  Serial.println(WiFi.localIP());

  server.on("/control", HTTP_GET, [](AsyncWebServerRequest *request){
    String command = request->getParam("cmd")->value();
    if (command == "F") {
      rotateClockwise();
    } else if (command == "B") {
      rotateCounterclockwise();
    } else if (command == "S") {
      stopMotor();
    }
    request->send(200, "text/plain", "OK");
```

```
  });

  server.begin();
}

void loop() {
  // Nothing to do here
}

void rotateClockwise() {
  digitalWrite(motor1Pin1, HIGH);
  digitalWrite(motor1Pin2, LOW);
  increaseSpeed();
}

void rotateCounterclockwise() {
  digitalWrite(motor1Pin1, LOW);
  digitalWrite(motor1Pin2, HIGH);
  increaseSpeed();
}

void stopMotor() {
  decreaseSpeed();
  digitalWrite(motor1Pin1, LOW);
  digitalWrite(motor1Pin2, LOW);
}

void increaseSpeed() {
  for (dutyCycle = 0; dutyCycle < 255; dutyCycle++) {
    ledcWrite(pwmChannel, dutyCycle);
    delay(10);
  }
}

void decreaseSpeed() {
  for (dutyCycle = 255; dutyCycle > 0; dutyCycle--) {
    ledcWrite(pwmChannel, dutyCycle);
    delay(10);
  }
}
```

3. DENEY SONUÇLARI





dc_motor_o | Arduino IDE 2.3.2

File Edit Sketch Tools Help

ESP32 Dev Module

dc_motor_o.ino

```
60    increaseSpeed();
61  }
62
63  void rotateCounterclockwise() {
64    digitalWrite(motor1Pin1, LOW);
65    digitalWrite(motor1Pin2, HIGH);
66    increaseSpeed();
67  }
68
69  void stopMotor() {
70    decreaseSpeed();
71    digitalWrite(motor1Pin1, LOW);
72    digitalWrite(motor1Pin2, LOW);
73  }
74
75  void increaseSpeed() {
76    for (dutyCycle = 0; dutyCycle < 255; dutyCycle++) {
```

Output Serial Monitor

```
Writing at 0x000c2112... (93 %)
Writing at 0x000c785b... (96 %)
Writing at 0x000cd2b6... (100 %)
Wrote 785008 bytes (498817 compressed) at 0x00010000 in 6.6 seconds (
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
```

Ln 6, Col 1 ESP32 Dev Module on COM:

controler | Arduino IDE 2.3.2

File Edit Sketch Tools Help

ESP32 Dev Module

controler.ino

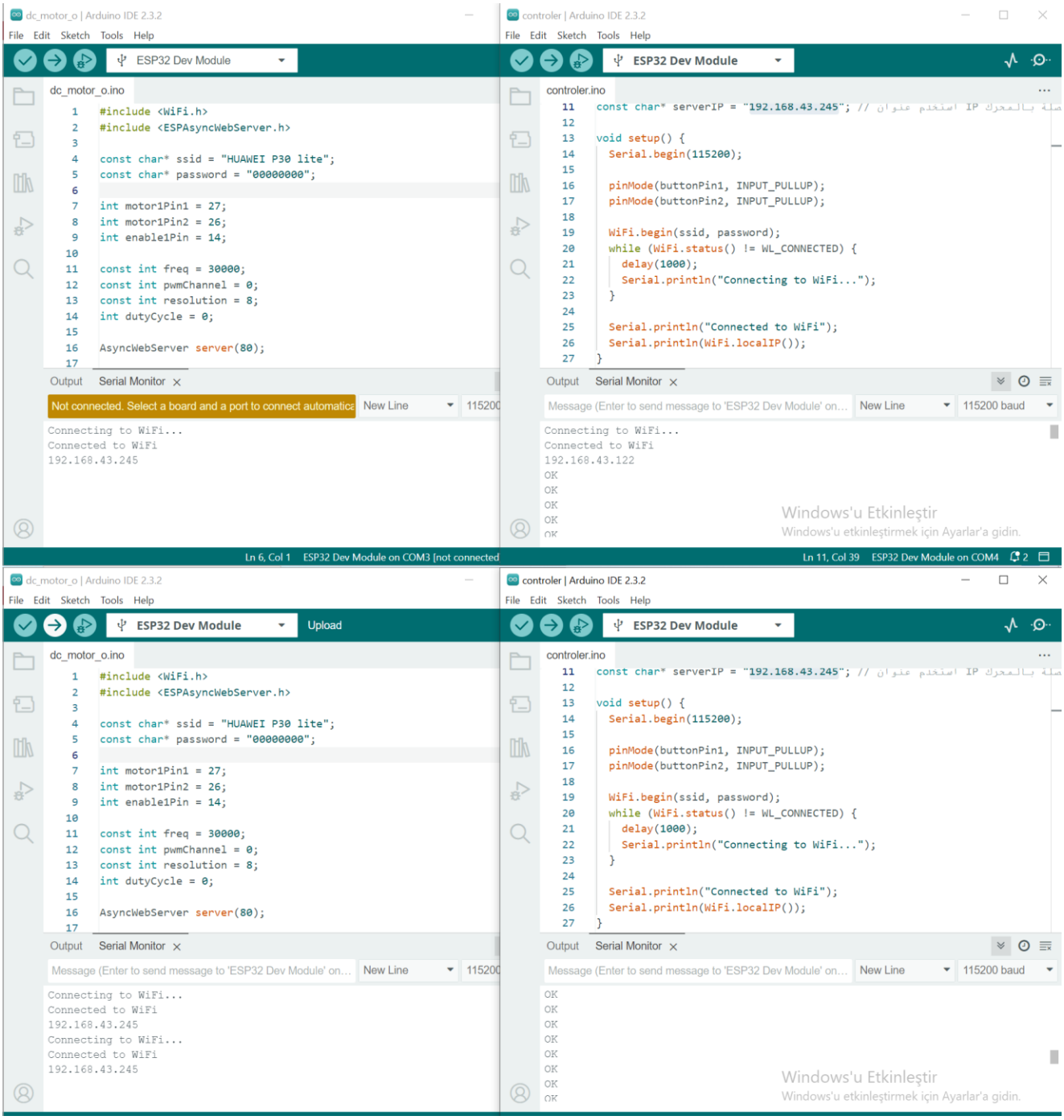
```
11  const char* serverIP = "192.168.43.245"; // استخدم عنوان IP الخاص بالمحرك
12
13  void setup() {
14    Serial.begin(115200);
15
16    pinMode(buttonPin1, INPUT_PULLUP);
17    pinMode(buttonPin2, INPUT_PULLUP);
18
19    WiFi.begin(ssid, password);
20    while (WiFi.status() != WL_CONNECTED) {
21      delay(1000);
22      Serial.println("Connecting to WiFi...");
23    }
24
25    Serial.println("Connected to WiFi");
26    Serial.println(WiFi.localIP());
27  }
```

Output Serial Monitor

```
Writing at 0x000de375... (94 %)
Writing at 0x000e3a01... (97 %)
Writing at 0x000e9094... (100 %)
Wrote 902112 bytes (582439 compressed) at 0x00010000 in 7.6 seconds (effective
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
```

Windows'u Etkinleştir
Windows'u etkinleştirmek için Ayarlar'a gidin.

Ln 11, Col 39 ESP32 Dev Module on COM4 2



https://youtube.com/shorts/ECScYEonM-E?feature=share

,

4. DEĞERLENDİRME

Test ve Sonuçlar

Donanım ve yazılım kurulumları, doğru çalıştığından emin olmak için test edilmiştir. Sağlanan görüntülerde görüldüğü üzere:

- ESP32 modülleri Wi-Fi ağına başarıyla bağlanmıştır.
- İstemci modülündeki butonlara basıldığında sunucu modülüne HTTP GET istekleri gönderilmiştir.
- Sunucu modülü, alınan isteklere göre motorun yönünü kontrol etmiştir.

Sonuç

Bu proje, ESP32 modüllerinin HTTP istekleri kullanarak bir Wi-Fi ağı üzerinden DC motoru kontrol etmek için nasıl kullanılacağını başarılı bir şekilde göstermektedir. Kurulum, daha karmaşık IoT uygulamaları için genişletilebilir, böylece birden fazla cihaz, uzaktan çeşitli çıktıları kontrol edebilir.