

**EE4216 Hardware for Internet of Things**

Lab3

**Men Jiying**

**A0239631A**

### 1. Introduction

* Background: Provide context about the ESP32-S3 and its networking capabilities.
* Objectives: Clearly state the objectives of the lab as listed:
  1. Program ESP32-S3 as a Wi-Fi station.
  2. Program ESP32-S3 as a client to connect to a web server.
  3. Program ESP32-S3 as a web server and sensor controller.

### 2. Materials

* List the materials and equipment used, as mentioned in the lab manual:
  + ESP32-S3 devkit board
  + Breadboard and Dupont wires
  + USB 2.0 cable
  + DHT22 temperature and humidity sensor
  + Computer

### 3. Procedure

* Task 1: Programming ESP32-S3 as a Wi-Fi Station

#include <WiFi.h>

const char\* ssid = "ESP32Server3"; // Wi-Fi SSID

const char\* password = "12345678"; // Wi-Fi Password

void setup() {

Serial.begin(115200);

// Start connecting to Wi-Fi

Serial.println("Connecting to Wi-Fi...");

WiFi.begin(ssid, password);

// Wait until connected to Wi-Fi

while (WiFi.status() != WL\_CONNECTED) {

delay(1000);

Serial.println("Connecting...");

}

// When connected, print IP address

Serial.println("Connected to Wi-Fi");

Serial.print("IP Address: ");

Serial.println(WiFi.localIP());

}

void loop() {

// Keep the connection alive

}

* Task 2: Programming ESP32-S3 as a HTTP Client

#include <WiFi.h>

#include <HTTPClient.h>

const char\* ssid = "ESP32Server";

const char\* password = "12345678";

const char\* serverURL = "http://192.168.4.1"; // Base URL

int matricNumber = 239631; // Replace this with your actual matric number

void setup() {

Serial.begin(115200);

// Connect to Wi-Fi

Serial.println("Connecting to Wi-Fi...");

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(1000);

Serial.println("Connecting...");

}

Serial.println("Connected to Wi-Fi");

// First HTTP request

if(WiFi.status() == WL\_CONNECTED) {

HTTPClient http;

http.begin(serverURL);

int httpCode = http.GET();

if(httpCode > 0) {

String payload = http.getString();

Serial.println("HTTP Response: " + payload);

// Get the last digit of the matric number and perform modulo 3

int lastDigit = matricNumber % 10;

int k = lastDigit % 3;

// Perform the second GET request by converting serverURL to String and concatenate

String endpoint = String(serverURL) + "/" + String(k);

Serial.println("Fetching from endpoint: " + endpoint);

http.begin(endpoint);

int secondHttpCode = http.GET();

if(secondHttpCode > 0) {

String secondPayload = http.getString();

Serial.println("Second HTTP Response: " + secondPayload);

} else {

Serial.println("Error in second HTTP request");

}

} else {

Serial.println("Error in first HTTP request");

}

http.end(); // Close connection

}

}

void loop() {

// No continuous tasks in the loop

}

* Task 3: Programming ESP32-S3 as a Webserver and Sensor Controller

#include <WiFi.h>

#include <WebServer.h>

#include <DHT.h>

#define DHTPIN 6 // Pin connected to DHT22 sensor

#define DHTTYPE DHT22 // DHT22 sensor type

#define LED\_PIN 16 // Onboard LED pin

const char\* ssid = "notyouriphone";

const char\* password = "hidejy123";

DHT dht(DHTPIN, DHTTYPE);

WebServer server(80);

void handleRoot() {

float temp = dht.readTemperature();

float hum = dht.readHumidity();

String html = "<h1>ESP32 Web Server</h1>";

html += "<p>Temperature: " + String(temp) + " °C</p>";

html += "<p>Humidity: " + String(hum) + " %</p>";

html += "<a href=\"/LED\_ON\">Turn LED ON</a><br>";

html += "<a href=\"/LED\_OFF\">Turn LED OFF</a><br>";

server.send(200, "text/html", html);

}

void handleLEDOn() {

digitalWrite(LED\_PIN, HIGH);

server.send(200, "text/html", "LED is ON");

}

void handleLEDOff() {

digitalWrite(LED\_PIN, LOW);

server.send(200, "text/html", "LED is OFF");

}

void setup() {

Serial.begin(115200);

dht.begin();

pinMode(LED\_PIN, OUTPUT);

// Connect to Wi-Fi

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(1000);

Serial.println("Connecting to Wi-Fi...");

}

Serial.println("Connected to Wi-Fi");

Serial.println(WiFi.localIP());

// Start Web Server

server.on("/", handleRoot);

server.on("/LED\_ON", handleLEDOn);

server.on("/LED\_OFF", handleLEDOff);

server.begin();

Serial.println("Web server started.");

}

void loop() {

server.handleClient(); // Listen for HTTP requests

}

* Lab test:

#include <WiFi.h>

#include <HTTPClient.h>

const char\* ssid = "ESP32Server3";

const char\* password = "12345678";

const char\* serverURL = "http://192.168.4.1"; // Webserver IP

void setup() {

Serial.begin(115200);

// Connect to the Wi-Fi Access Point (AP)

Serial.println("Connecting to Wi-Fi...");

WiFi.begin(ssid, password);

// Wait until the ESP32 is connected to Wi-Fi

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

Serial.println("\nConnected to Wi-Fi");

// Print the IP address assigned to the ESP32

Serial.print("IP Address: ");

Serial.println(WiFi.localIP());

// Make sure the connection is active

if(WiFi.status() == WL\_CONNECTED) {

HTTPClient http;

// Send an HTTP GET request to the root of the web server

http.begin(serverURL);

int httpCode = http.GET();

// Check the response code

if(httpCode > 0) {

String payload = http.getString();

Serial.println("HTTP Response: " + payload);

// The server asks for a POST request to "/secret"

String secretEndpoint = String(serverURL) + "/secret";

Serial.println("Sending POST request to: " + secretEndpoint);

http.begin(secretEndpoint);

http.addHeader("Content-Type", "application/x-www-form-urlencoded");

// Send the POST request

int postHttpCode = http.POST("");

if(postHttpCode > 0) {

String secretMessage = http.getString();

Serial.println("Secret Message: " + secretMessage);

} else {

Serial.println("Error in POST request to /secret");

}

} else {

Serial.println("Error in HTTP request to the root");

}

http.end();

} else {

Serial.println("Wi-Fi connection failed");

}

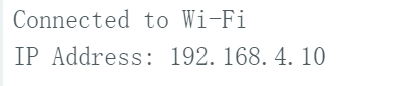
}

void loop() {

}

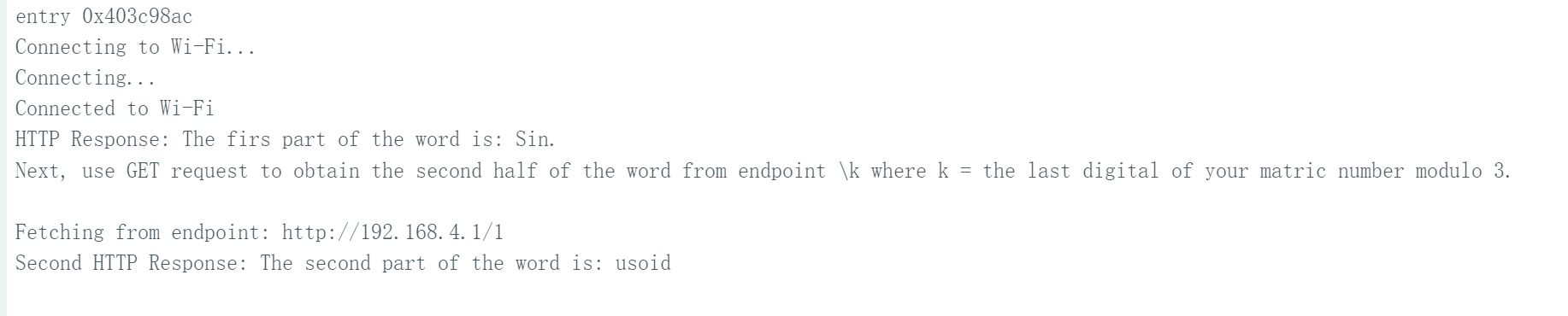
### 4. Results

* Task 1 Results:



* Task 2 Results:

The result is: Sinusoid



* Task 3 Results:



* Labtest Result:  
  + The secret is EE4216