

ECE LABORATORY

DREXEL UNIVERSITY

To: Dr. Peters

From: Ehi Simon

Re: ECE 304 Lab 3 - Clients, Servers, and HTML

PURPOSE:

The purpose of this week's lab is to perform rudimentary remote monitoring and control using a smart phone and/or laptop through the use of the Bluetooth and Bluetooth Low Energy (BLE) protocols. Using Bluetooth and BLE will make it easy to perform on-demand monitoring and control.

Discussion:

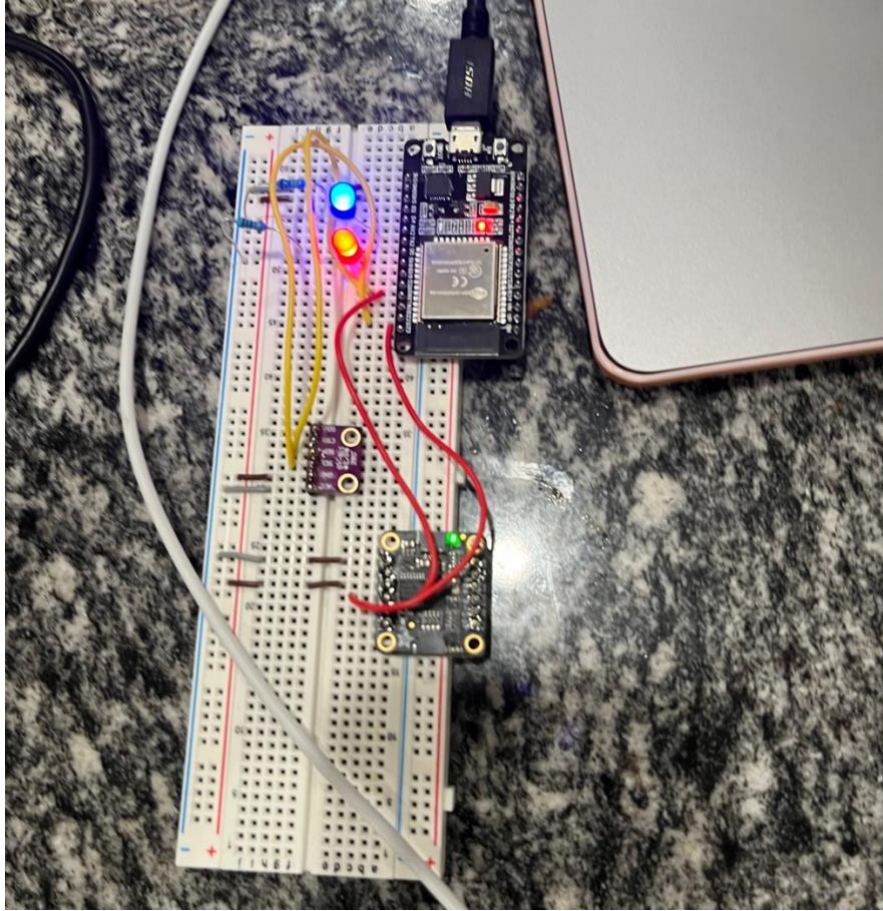


Fig. 1. Circuit Connection for Project 2

The circuit for the lab was built like the one above. It consists of 2 330Ω resistors, a red LED, a blue LED, a BME280 Environmental Sensor, an Adafruit BNO085 IMU, and an ESP32S microcontroller.

Exp3.cpp

In my main.cpp file, I initialized the multiple libraries that were needed for the sensors to work and provide readings. I also included libraries to get the ESP32 to connect to AP, STA, or both for the Wi-Fi. The LEDs are defined, the sea level reference pressure is defined and the BNO08X chip is reset. The BME object is created, the soft AP SSID & password are set, as well as the network SSID and password. IP Address details are put in for AP mode, and then the web server is opened on port 80. This can all be found in the figure below:

```
// Import Libraries
#include <WiFi.h> // Wifi Library
#include <WebServer.h>
#include <Wire.h>
#include <SPI.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BME280.h>
#include <Arduino.h>
#include <Adafruit_BNO08x.h>

// Define constants
const int RED_LED = 5;
const int BLUE_LED = 4;
const int freq = 5000;
const int ledChannel = 0;
const int resolution = 8;
#define BNO08X_RESET -1

// Define Sea level reference pressure (hPa)
#define SEALEVELPRESSURE_HPA (1013.25)

// Pin Request Status
// bool LED_request = LOW;
// bool dht_request = LOW;

// Create object for BME280
Adafruit_BME280 bme;

/* Put your Soft AP SSID & Password*/
const char* AP_ssid ="PetersESP32";
const char* AP_password = "12345678";

/* Put your Network SSID & Password */
const char* network_ssid = "SimonESP32";
const char* network_password = "12345678";

/* Put IP Address details for AP mode */
IPAddress local_ip(192, 168, 1, 200);
IPAddress gateway(192, 168, 1, 200);
IPAddress subnet(255, 255, 255, 0);

WebServer server(80); // Web Server open on port 80
```

Fig. 2. Figure Showing Initialization of Libraries and Variable Definitions

```

String SendHTML(String t_string, String h_string,
                String p_string, String a_string, String yaw, String pitch, String roll) {
    String ptr = "<!DOCTYPE html>"
    "<html>"
    "<head>"
    "<meta name=\"viewport\" content=\"width=device-width, initial-scale=1.0, user-scalable=no\">"
    "<title>LED Control</title>"
    "<style>"
    "html { font-family: Helvetica; display: inline-block; margin: 0px auto; text-align: center; }"
    "input.largerCheckbox { width: 20px; height: 20px; }"
    "body { margin-top: 50px; }"
    "div { text-align: center; }"
    "h1 { color: #444444; margin: 50px auto 30px; }"
    "h3 { color: #444444; margin-bottom: 50px; }"
    "p { font-size: 14px; color: #888; margin-bottom: 10px; }"
    "table, th, td { border: 1px solid black; border-collapse: collapse; }"
    "</style>"
    "</head>"
    "<body style=\"background-color:aquamarine\";>"
    "<div>"
    "<h1>LED Commander</h1>"
    "<form action=\"\" method=\"post\">"
    "<label for=\"bluetoggle\">Blue LED Toggle </label>"
    "<input type=\"checkbox\" class=\"largerCheckbox\" id=\"blue\" name=\"blueled\"><br><br>"
    "<label for=\"blue\">Red LED Value</label>"
    "<input type=\"text\" id=\"redtoggle\" name=\"redled\" value=\"\" maxlength=\"3\" size=\"2\"><br>"
    "<input type=\"submit\" value=\"Submit\">"
    "</form>"
    // ptr += "<p>" + t_string + "</p>"
    // "<p>" + h_string + "</p>"
    // "<p>" + p_string + "</p>"
    // "<p>" + a_string + "</p>"
    "<table>"
    "<tr>"
    "<th>Characteristic</th>"
    "<th>BME280 Readings</th>"
    "</tr>"
    "<tr>"
    "<td>Temperature</td>"
    "<td>" + t_string + "</td>"
    "</tr>"
    "<tr>"
    "<td>Humidity</td>"

```

Fig. 3. Figure Showing SendHTML Function

The figure above shows the HTML code that was inserted into my exp3.cpp file in the code. I create the webpage, give it a background color, give it a title, a heading, and I create two tables that contain the readings from the BME 280 sensor and the readings from the BNO. It also has a checkbox to toggle the blue LED, as well as a form to choose a value for the red LED.

```

    struct euler_t {
        float yaw;
        float pitch;
        float roll;
    } ypr;

    Adafruit_BNO08x bno08x(BNO08X_RESET);
    sh2_SensorValue_t sensorValue;
    sh2_SensorId_t reportType = SH2_ARVR_STABILIZED_RV;
    long reportIntervalUs = 5000;

    void setReports(sh2_SensorId_t reportType, long report_interval) {
        Serial.println("Setting desired reports");
        if (! bno08x.enableReport(reportType, report_interval)) {
            Serial.println("Could not enable stabilized remote vector");
        }
    }

    void quaternionToEuler(float qr, float qi, float qj, float qk, euler_t* ypr, bool degrees = false) {
        float sqr = sq(qr);
        float sqi = sq(qi);
        float sqj = sq(qj);
        float sqk = sq(qk);

        ypr->yaw = atan2(2.0 * (qi * qj + qk * qr), (sqi - sqj - sqk + sqr));
        ypr->pitch = asin(-2.0 * (qi * qk - qj * qr) / (sqi + sqj + sqk + sqr));
        ypr->roll = atan2(2.0 * (qj * qk + qi * qr), (-sqi - sqj + sqk + sqr));

        if (degrees) {
            ypr->yaw *= RAD_TO_DEG;
            ypr->pitch *= RAD_TO_DEG;
            ypr->roll *= RAD_TO_DEG;
        }
    }

    void quaternionToEulerRV(sh2_RotationVectorWAcc_t* rotational_vector, euler_t* ypr, bool degrees = false) {
        quaternionToEuler(rotational_vector->real, rotational_vector->i, rotational_vector->j, rotational_vector->k, ypr, degrees);
    }

```

Fig. 4. Figure Showing Functions to Calculate Yaw, Pitch, and Roll

The figure above shows functions that were needed, and implemented to calculate the yaw, pitch, and roll.

The figure below shows the `handle_OnConnect()` function. This function assigns readings from the sensors to variables that are used in the HTML code to be displayed on the webpage. It also prints them out to the serial terminal based on the LED values that the user enters on the webpage. This function toggles the blue LED after receiving info from the webpage, changes the intensity of the red LED. It calls the `SendHTML` function that was seen earlier in figure 3 with arguments converted to strings from the sensor readings. This is how the html webpage is created.

```

void handle_OnConnect() {
    String blueLED = server.arg("blueled");
    digitalWrite(BLUE_LED, blueLED.equals("on") ? HIGH : LOW);

    String redLED = server.arg("redled");
    int rledval = redLED.toInt();
    if (rledval >= 0 && rledval < 256) {
        digitalWrite(ledChannel, redLED.toInt());
    }

    float temp = bme.readTemperature();
    float humid = bme.readHumidity();
    float press = bme.readPressure();
    float alt = bme.readAltitude(SEALEVELPRESSURE_HPA);
    String tempStr = "Current Temperature: " + String(temp, 2) + " C";
    String humidStr = "Current Humidity: " + String(humid, 2) + "%";
    String pressStr = "Current Pressure: " + String(press/1000, 2) + " kPa";
    String altStr = "Current Altitude: " + String(alt, 2) + " m";

    //Serial.println("LED Status: LOW");
    Serial.println(tempStr);
    Serial.println(humidStr);
    Serial.println(pressStr);
    Serial.println(altStr);

    if (bno08x.getSensorEvent(&sensorValue)) {
        quaternionToEulerRV(&sensorValue.un.arvrStabilizedRV, &ypr, true);

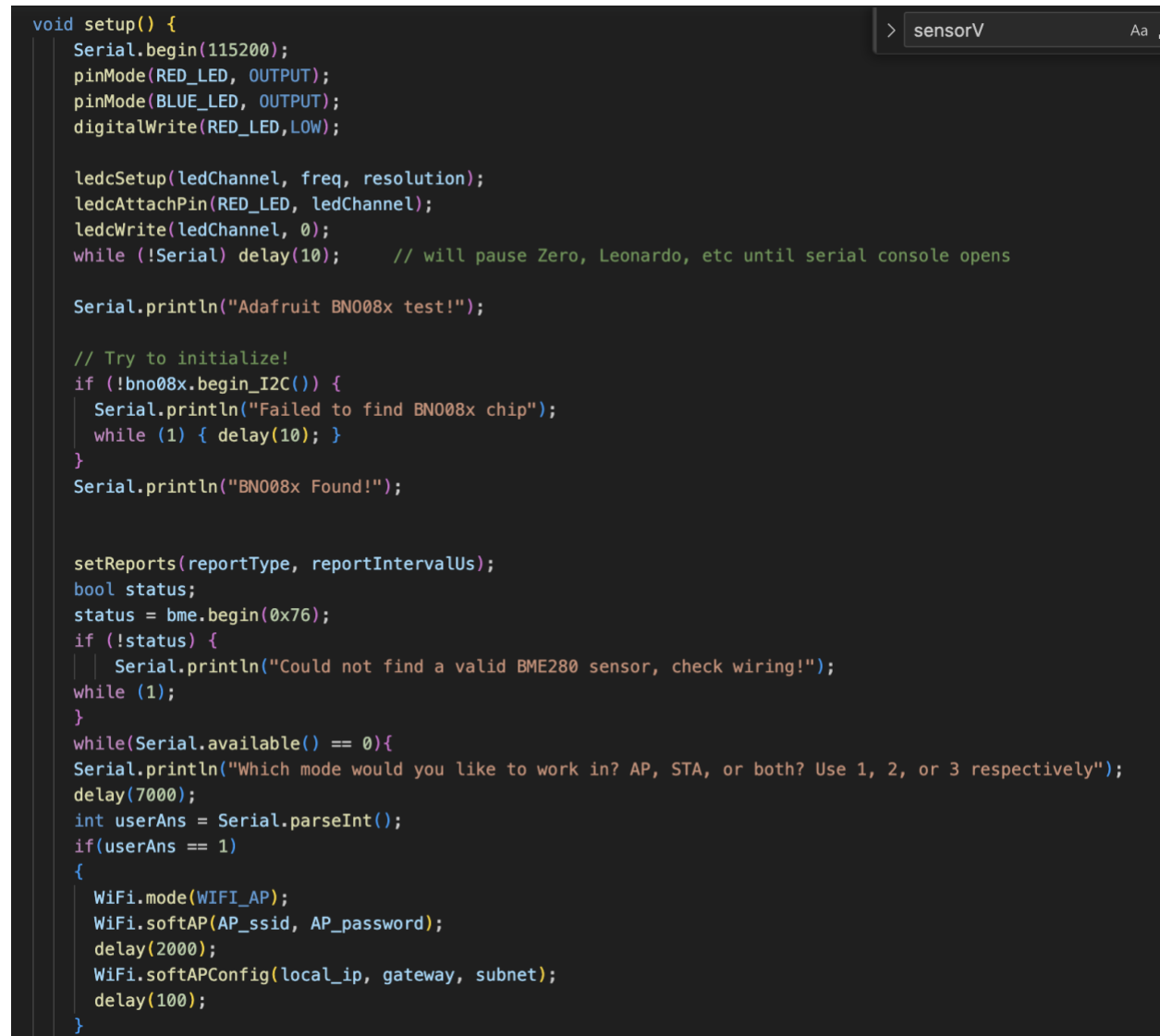
        static long last = 0;
        long now = micros();
        Serial.print(now - last);          Serial.print("\t");
        last = now;
        Serial.print(sensorValue.status);   Serial.print("\t"); // This is accuracy in the range of 0 to 3
        Serial.print(ypr.yaw);             Serial.print("\t");
        Serial.print(ypr.pitch);           Serial.print("\t");
        Serial.println(ypr.roll);

        String yawStr = String(ypr.yaw, 4);
        String pitchStr = String(ypr.pitch, 4);
        String rollStr = String(ypr.roll, 4);
        server.send(200, "text/html", SendHTML(tempStr,
            humidStr, pressStr, altStr, yawStr, pitchStr, rollStr));
    }
}

```

Fig. 5. Figure Showing handle_OnConnect Function

The figure below shows the setup function of the file. This setup function does more than those in previous projects. It sets the LED pin modes. It also initializes the BME280 sensor and the BNO08X chip. It then prompts the user to answer which mode it wants to work with. After the user enters a value, it uses if statements to choose the Wi-Fi mode based on the answer and initialized that Wi-Fi mode.



```
void setup() {
  Serial.begin(115200);
  pinMode(RED_LED, OUTPUT);
  pinMode(BLUE_LED, OUTPUT);
  digitalWrite(RED_LED, LOW);

  ledcSetup(ledChannel, freq, resolution);
  ledcAttachPin(RED_LED, ledChannel);
  ledcWrite(ledChannel, 0);
  while (!Serial) delay(10); // will pause Zero, Leonardo, etc until serial console opens

  Serial.println("Adafruit BNO08x test!");

  // Try to initialize!
  if (!bno08x.begin_I2C()) {
    Serial.println("Failed to find BNO08x chip");
    while (1) { delay(10); }
  }
  Serial.println("BNO08x Found!");

  setReports(reportType, reportIntervalUs);
  bool status;
  status = bme.begin(0x76);
  if (!status) {
    Serial.println("Could not find a valid BME280 sensor, check wiring!");
    while (1);
  }
  while(Serial.available() == 0){
    Serial.println("Which mode would you like to work in? AP, STA, or both? Use 1, 2, or 3 respectively");
    delay(7000);
  }
  int userAns = Serial.parseInt();
  if(userAns == 1)
  {
    WiFi.mode(WIFI_AP);
    WiFi.softAP(AP_ssid, AP_password);
    delay(2000);
    WiFi.softAPConfig(local_ip, gateway, subnet);
    delay(100);
  }
}
```

Fig. 6. Figure Showing Setup Function

After the if statements, the function turns on the server using our handle_OnConnect function and begins it. It then prints the HTTP Address and the local IP address.

```
if (userAns == 2)
{
  WiFi.mode(WIFI_STA);
  WiFi.begin(network_ssid, network_password);
}
if (userAns == 3)
{
  WiFi.mode(WIFI_AP_STA);
  WiFi.softAP(AP_ssid, AP_password);
  delay(2000);
  WiFi.softAPConfig(local_ip, gateway, subnet);
  delay(100);
  WiFi.begin(network_ssid, network_password);
}
}

// while (WiFi.status() != WL_CONNECTED){
//   delay(500);
//   Serial.println("Connecting to Wifi...");
// }
delay(100);

server.on("/", handle_OnConnect);
server.onNotFound(handle_NotFound);
server.begin();

Serial.print("HTTP server started at AP Address: ");
Serial.println(WiFi.softAPIP());
Serial.print("HTTP server started at Network Address: ");
Serial.println(WiFi.localIP());
```

Fig. 7. Figure Showing Continuation of Setup Function

The loop function is made up of one line only and that can be seen in the figure below.

```
void loop() {
  server.handleClient();
}
```

Fig. 8. Figure Showing Loop Function

The final result was a webpage as well as output on the serial monitor. It can be seen in the figures below:

```
PROBLEMS  OUTPUT  TERMINAL

Adafruit BN008x test!
BN008x Found!
Setting desired reports
Which mode would you like to work in? AP, STA, or both? Use 1, 2, or 3 respectively
1
HTTP server started at AP Address: 192.168.1.200
HTTP server started at Network Address: 0.0.0.0
Current Temperature: 18.55 C
Current Humidity: 56.74%
Current Pressure: 100.55 kPa
Current Altitude: 64.71 m
Current Temperature: 18.54 C
Current Humidity: 56.72%
Current Pressure: 100.55 kPa
Current Altitude: 64.90 m
```

Fig. 9. Figure Showing Output on Serial Monitor

LED Commander

Blue LED Toggle ☐

Red LED Value

Submit

Characteristic	BME280 Readings
Temperature	Current Temperature: 18.54 C
Humidity	Current Humidity: 56.72%
Pressure	Current Pressure: 100.55 kPa
Altitude	Current Altitude: 64.90 m

Characteristic	BNO08X Readings
Yaw	-144.3603
Pitch	-5.8742
Roll	-8.8226

Fig. 10. Figure Showing Output on Webpage

Conclusion

In this experiment, I learned about the different Wi-Fi modes and how ESP32 connects to them. I learnt how to incorporate HTML into my cpp file to build a webpage, and I polished my HTML skills.