LABORATORY REPORT

LAB 6: Bluetooth and Wi-Fi data interfacing with microcontroller and computer-based system

GROUP B

PROGRAMME: MECHATRONIC ENGINEERING

GROUP MEMBERS:	MATRIC NO:
1. MUHAMMAD AFIQ BIN AHMAD	2115203
2. HANIS BINTI MOHD IZANI	2020590
3. SARAH AISHAH BINTI SA'AID HAZLEY	2117600
4. NUR AYU NAZIERA BINTI ROSLI	2119202
5. NOORUL GHOUSIAH BINTI NOORDEEN SAHIB	2118298

DATE OF SUBMISSION:
Wednesday, 13th December 2023

ABSTRACT

This lab consists of two parts, that is part A and part B. The objective of part A is to create a wireless temperature monitoring system using Wi-Fi, Arduino, and a sensor. The objective of part B is to use an app to communicate with Arduino via Bluetooth using a smartphone. For part A, the circuit connection is set up, an Arduino sketch is built, and a cloud service, in this case, ThinkSpeak, is set up for real-time monitoring of sensor values. As for part B, after the circuit connection is set up, the Arduino Bluetooth Controller app is used to send commands to Arduino Uno board via Bluetooth connectivity. The results of both parts are satisfactory as we successfully established communication with Arduino board through Wi-Fi and Bluetooth connectivity.

TABLE OF CONTENTS

Table of Contents

ABSTRACT 2
TABLE OF CONTENTS 2
INTRODUCTION 3
PART A 3
  PROCEDURE 3
  RESULTS 6
  DISCUSSION 8
PART B 8
  PROCEDURE 8
  RESULTS 11
  DISCUSSION 14
CONCLUSION 15
RECOMMENDATIONS 15
REFERENCES 16
APPENDICES 16
ACKNOWLEDGEMENT 16
STUDENT'S DECLARATION 16

INTRODUCTION

Lab 6 is a laboratory experiment on the topic of Bluetooth and Wi-Fi data interfacing with microcontroller and computer-based system. The purpose of this lab is to study the connectivity of Arduino board to Wi-Fi and Bluetooth for data transmission. This experiment also exposes us to cloud services available online for remote monitoring purposes. This lab has two parts, that is part A: Wi-Fi data interfacing, and part B: Bluetooth data interfacing. The objective of part A is to create a wireless temperature monitoring system using Wi-Fi, Arduino, and a sensor. A Wi-Fi module is used to add Wi-Fi capabilities to Arduino board. The objective of part B is to use an app to communicate with Arduino via Bluetooth using a smartphone. A Bluetooth module, HC-05 is connected to Arduino board to provide the board with Bluetooth connection.

HC-05 is a Bluetooth module that uses serial communication to communicate with electronics. It uses the 2.45GHz frequency band. The rate of data transfer can vary up to 1Mbps and is in a range of 10 meters. The Bluetooth module can be operated in Master-Slave mode which means that it will not send or receive data from external sources. There are two modes of operation for this module: command mode and data mode. In command mode, the programmer can communicate with the module through AT Commands for configuration of various settings and parameters. In command mode, the module is used for communicating with other Bluetooth devices, thus data transfer happens in this mode.

ESP32 is a development board. It is a successor to the popular ESP8266. ESP 32 has built-in Wi-Fi and Bluetooth connectivity. The ESP32 is based on a Tensilica Xtensa LX6 dual-core microprocessor, with an operating frequency of up to 240 MHz. This board comes with touch-sensitive pins that can be used to “wake up” the ESP32 from deep sleep mode and a built-in Hall effect sensor.

PART A: Wi-Fi data interfacing

OBJECTIVE

To create a wireless temperature monitoring system using Wi-Fi, Arduino, and a sensor.

PROCEDURE

Materials And Equipment

- ESP32
- Temperature Sensor, BME280
- Wi-Fi network and internet access
- ThingSpeak
- Power supply for the ESP32
- Breadboard and jumper wires

Experimental Setup

1. Hardware Setup

    a. Connect the Bme280 sensor to the ESP32 using the GPIO pins 22(SCL) and 22(SDL).
    b. Connect the ESP32 to Wi-Fi network using the built-in Wi-Fi capabilities.
2. ESP32 Programming
    a. Write an Arduino sketch that reads temperature intensity data from the sensor.
    b. Set up Wi-Fi connectivity to send temperature data to a ThingSpeak cloud service.

    The following is the Arduino sketch used:

```
#include <WiFi.h>
#include "secrets.h"
#include "ThingSpeak.h" // always include thingspeak header file after other header files and custom macros
#include <SPI.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BME280.h>

#define BME_SCK 13
#define BME_MISO 12
#define BME_MOSI 11
#define BME_CS 10

#define SEALEVELPRESSURE_HPA (1013.25)

Adafruit_BME280 bme; // I2C
char ssid[] = "Xiaomi 11 Lite 5G NE";   // your network SSID (name)
char pass[] = "entahlah";   // your network password
int keyIndex = 0;            // your network key Index number (needed only for WEP)
WiFiClient  client;

unsigned long myChannelNumber = 2383522;
const char * myWriteAPIKey = "CYIXWVA4505XSHQ6" ;

int number = 0;


unsigned long delayTime;

void printValues() {
   Serial.print("Temperature = ");
   Serial.print(bme.readTemperature());
   Serial.println(" °C");


   Serial.println();
}


void setup() {
   Serial.begin(9600);
   Serial.println("BME280 test");
```

```
    while(!Serial);    // time to get serial running
    Serial.println(F("BME280 test"));

    unsigned status;

    // default settings
    status = bme.begin(0X76);
    // You can also pass in a Wire library object like &Wire2
    // status = bme.begin(0x76, &Wire2)
    if (!status) {
        Serial.println("Could not find a valid BME280 sensor, check wiring,
address, sensor ID!");
        Serial.print("SensorID was: 0x"); Serial.println(bme.sensorID(),16);
        Serial.print("        ID of 0xFF probably means a bad address, a BMP 180 or
BMP 085\n");
        Serial.print("   ID of 0x56-0x58 represents a BMP 280,\n");
        Serial.print("        ID of 0x60 represents a BME 280.\n");
        Serial.print("        ID of 0x61 represents a BME 680.\n");
        while (1) delay(10);
    }

    Serial.println("-- Default Test --");
    delayTime = 1000;

    Serial.println();

    while (!Serial) {
      ; // wait for serial port to connect. Needed for Leonardo native USB port only
    }

    WiFi.mode(WIFI_STA);
    ThingSpeak.begin(client);  // Initialize ThingSpeak
}


void loop() {
    printValues();
    delay(delayTime);

      if(WiFi.status() != WL_CONNECTED){
      Serial.print("Attempting to connect to SSID: ");
      Serial.println(SECRET_SSID);
      while(WiFi.status() != WL_CONNECTED){
        WiFi.begin(ssid, pass); // Connect to WPA/WPA2 network. Change this line
if using open or WEP network
        Serial.print(".");
        delay(5000);
      }
      Serial.println("\nConnected.");
    }

    number =  bme.readTemperature();
    int x = ThingSpeak.writeField(myChannelNumber, 1, number,
```

•

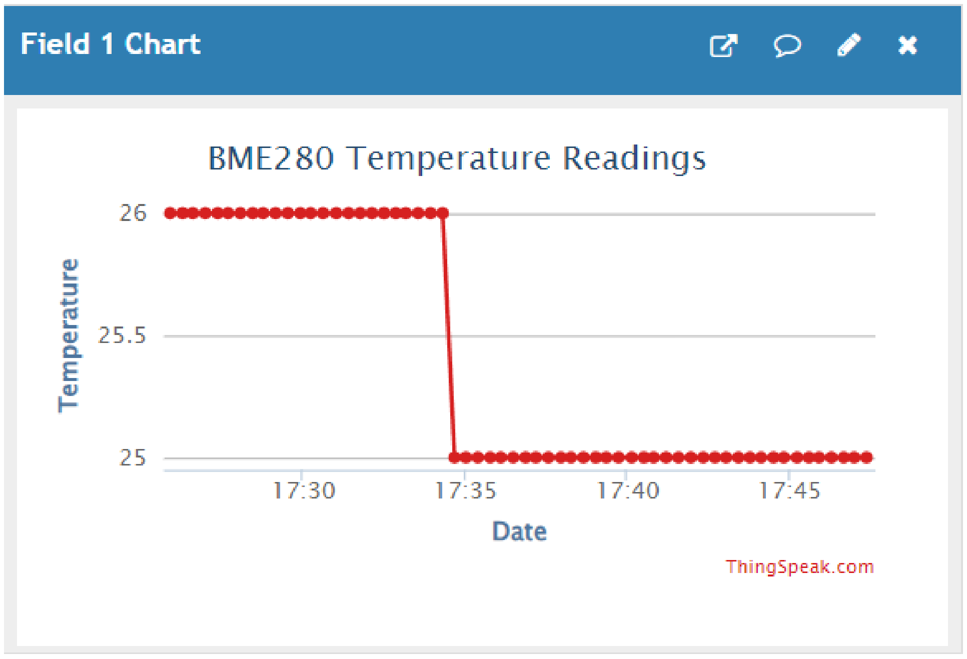
```
        myWriteAPIKey);
          if(x == 200){
            Serial.println("Channel update successful.");
          }
          else{
            Serial.println("Problem updating channel. HTTP error code " + String(x));
          }


          delay(20000); // Wait 20 seconds to update the channel again
        }
```

3. Remote Monitoring:
    a. Access ThingSpeak dashboard on computer to remotely monitor the temperature intensity in real-time via the internet.

Methodology

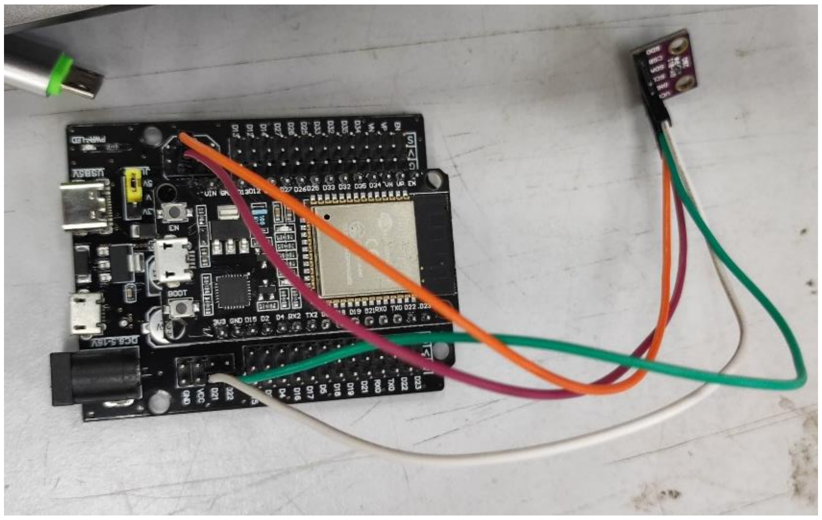
1. Place the Bme280 sensor in a room or area you want to monitor and control the temperature.
2. Connect the Arduino to a power source and ensure it's connected to your Wi-Fi network.
3. Monitor the room's temperature in real-time using the ThingSpeak dashboard.

RESULTS



•

Connection:



Analysis:

From the ThingSpeak dashboard, we managed to monitor our room temperature for about 20 minutes. The first half of the monitoring session, we put the sensor near a heated source (our laptops' exhaust fan) and thus the temperature raised to 26 degrees Celsius. However, the resolution of the BME280 temperature is 0.01°C, so the changes in the temperature are too small for us to see in the graph. After removing the sensor form the exhaust fan, we can see that the temperature drops by 1-degree Celcius to the room's temperature (with Airconditioning).

DISCUSSION

For this particular Lab Session, we used ESP32's built in Wi-Fi capabilities, we coded in using the Arduino Sketch. From the code, to use ThingSpeak, we had to download the ThingSpeak library from Adafruit. And from there we created a channel on the ThingSpeak in order for us to write data to the graph every 3 seconds. Using the Channels address and API write address, and our Wi-Fi, we were able to successfully write to our Channel. Furthermore, In order to utilize the BME280 sensor, we had to include the BME280 library. In the sketch above, we included programming so that the we can check if we are connected to the SSID (our network). In the loop, it reads the temperature from the BME280 sensor, connects to Wi-Fi if not connected, updates the ThingSpeak channel with the temperature data, and then waits for a specified time before repeating.

PART B: Bluetooth data interfacing

OBJECTIVE

To communicate with Arduino via Bluetooth using a smartphone.

PROCEDURE

<u>Materials And Equipment</u>

- ESP32 board with Wi-Fi capability
- Temperature sensor, BME280
- Smartphone and Bluetooth support
- Power supply for the ESP32
- Breadboard and jumper wires

<u>Experimental Setup</u>

1. Connect VCC with 3.3V of ESP32, please do not connect it with 5V as that can cook the module

2. Connect GND with any GND of Arduino, .

3. Connect the long end of LED with 220 ohms to 1K ohm resistor, connect the other end of resistor to the pin 13 of Arduino, connect the short leg of LED to GND of Arduino.

4. Send Serial Commands to Arduino using Serial Monitor (or Any Terminal).

5. Connect the Rx and Tx pins of HC-05 back to Arduino.

6. Pair the device with your mobile phone, download "Arduino Bluetooth Controller" app from Google Play Store.

```
char junk;
String inputString="";

void setup()
{
Serial.begin(9600);            // set the baud rate to 9600, same should be of your Serial Monitor
pinMode(13, OUTPUT);
}

void loop()
{
  if(Serial.available()){
  while(Serial.available())
    {
      char inChar = (char)Serial.read(); //read the input
      inputString += inChar;        //make a string of the characters coming on serial
    }
    Serial.println(inputString);
    while (Serial.available() > 0)
    { junk = Serial.read() ; }      // clear the serial buffer
    if(inputString == "A"){         //in case of 'A' turn the LED on
      digitalWrite(13, HIGH);
    }else if(inputString == "B"){   //incase of 'B' turn the LED off
      digitalWrite(13, LOW);
    }
    inputString = "";
  }
}
```

<u>Bluetooth control with motor and temperature sensor Bme280:</u>

```
#include <Wire.h>
#include <SPI.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BME280.h>
#include <BluetoothSerial.h>
#include <ESP32_Servo.h>

#define BME_SCK 13
#define BME_MISO 12
#define BME_MOSI 11
#define BME_CS 10

#define SEALEVELPRESSURE_HPA (1013.25)
```

```
Adafruit_BME280 bme; // I2C
BluetoothSerial SerialBT;

unsigned long delayTime;

void setup() {
  Serial.begin(9600);
  while (!Serial); // time to get serial running

  SerialBT.begin("ESP32_BT_Classic"); // Bluetooth device name
  Serial.println(F("BME280 test"));

  unsigned status;

  // default settings
  status = bme.begin(0X76);
  if (!status) {
    Serial.println("Could not find a valid BME280 sensor, check wiring, address, sensor ID!");
    Serial.print("SensorID was: 0x");
    Serial.println(bme.sensorID(), 16);
    Serial.print("        ID of 0xFF probably means a bad address, a BMP 180 or BMP 085\n");
    Serial.print("   ID of 0x56-0x58 represents a BMP 280,\n");
    Serial.print("        ID of 0x60 represents a BME 280.\n");
    Serial.print("        ID of 0x61 represents a BME 680.\n");
    while (1)
      delay(10);
  }

  Serial.println("-- Default Test --");
  delayTime = 1000;

  Serial.println();
}

void loop() {
  if (SerialBT.available()) {
    processBluetoothCommand();
  }

  printTemperatureOverBluetooth();
  delay(delayTime);
}

void printTemperatureOverBluetooth() {
  SerialBT.print("Temperature = ");
  SerialBT.print(bme.readTemperature());
  SerialBT.println(" °C");
}

void processBluetoothCommand() {
  String command = SerialBT.readStringUntil('\n');
  command.trim();
```

```
    if (command == "ON") {
      // Replace PIN_NUMBER with the actual pin number you want to turn on
      analogWrite(5, 255);
    }
    if (command == "OFF") {
      // Replace PIN_NUMBER with the actual pin number you want to turn on
      analogWrite(5, 0);
    }
}
```

Materials And Equipment and Setup (LED)

1. Connecting HC-05 Bluetooth Module with Arduino
2. Connect the LED and Control it using Arduino Serial Monitor
3. Send Serial Commands to Arduino using Serial Monitor
4. Getting HC-05 to Work with Arduino and testing the communication

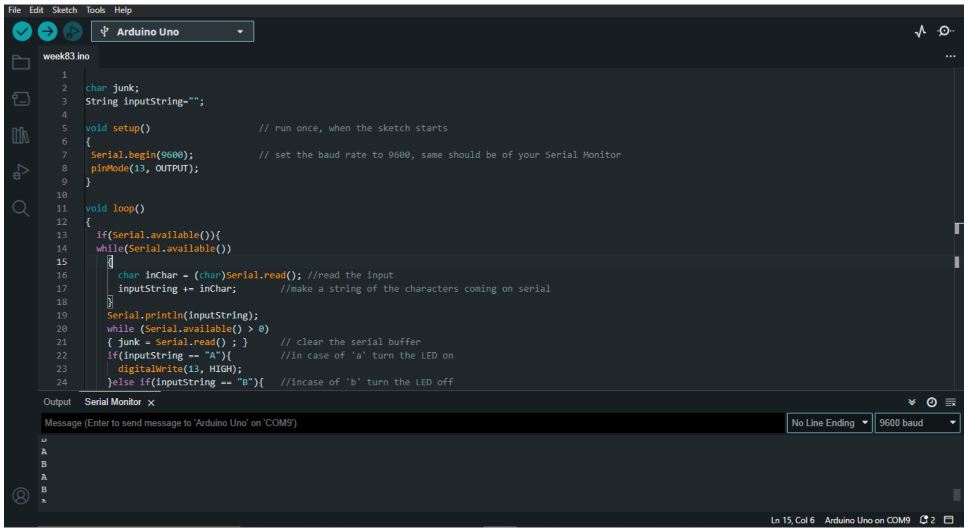
Materials And Equipment and Setup (Motor and Sensor)

5. Connect BME280 Sensor to ESP32 pin22 and 21
6. Connect motor to single channel relay and relay to pin 5

Ways to use Arduino Bluetooth Controller app (LED):

1. Open the Arduino Bluetooth Controller app.
2. From the menu, tap on "Connect a device".
3. there will be a pop-up of "Paired Devices", tap on "HC-05", after a second, we will get a toast notifying "Connected to HC-05".
4. Now type "A" and send, the LED will turn on, similarly, send "B" to turn the LED off.

RESULTS



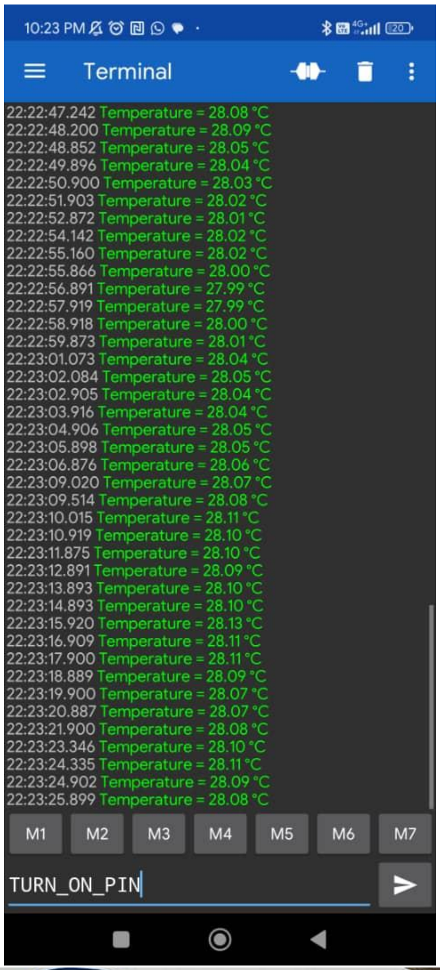
```

char junk;
String inputString="";

void setup()                    // run once, when the sketch starts
{
 Serial.begin(9600);            // set the baud rate to 9600, same should be of your Serial Monitor
 pinMode(13, OUTPUT);
}

void loop()
{
  if(Serial.available()){
  while(Serial.available())
    {
      char inChar = (char)Serial.read(); //read the input
      inputString += inChar;        //make a string of the characters coming on serial
    }
    Serial.println(inputString);
    while (Serial.available() > 0)
    { junk = Serial.read() ; }      // clear the serial buffer
    if(inputString == "A"){         //in case of 'a' turn the LED on
      digitalWrite(13, HIGH);
    }else if(inputString == "B"){   //incase of 'b' turn the LED off
```

Message (Enter to send message to 'Arduino Uno' on 'COM9')

```
A
B
A
B
```

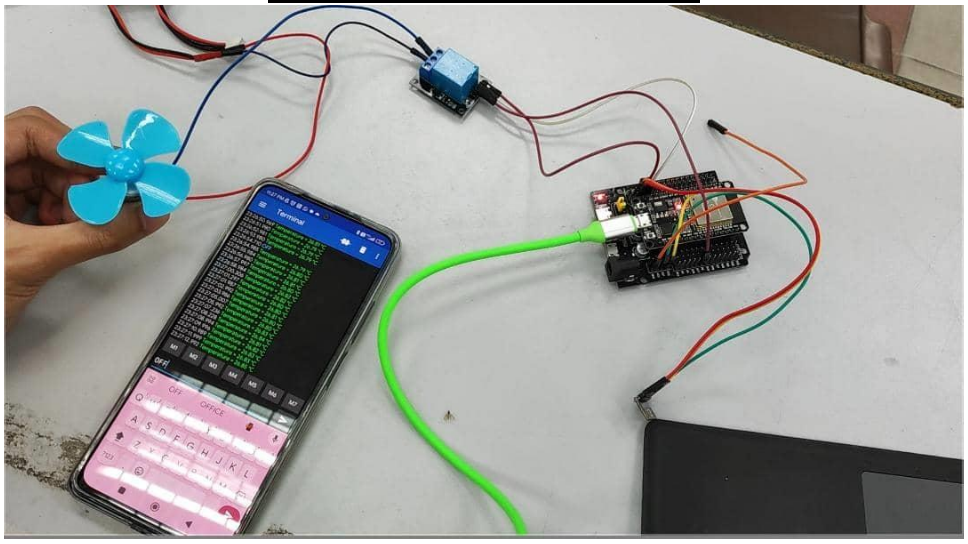
The video link: https://youtu.be/rzkJDNdaOc4?feature=shared

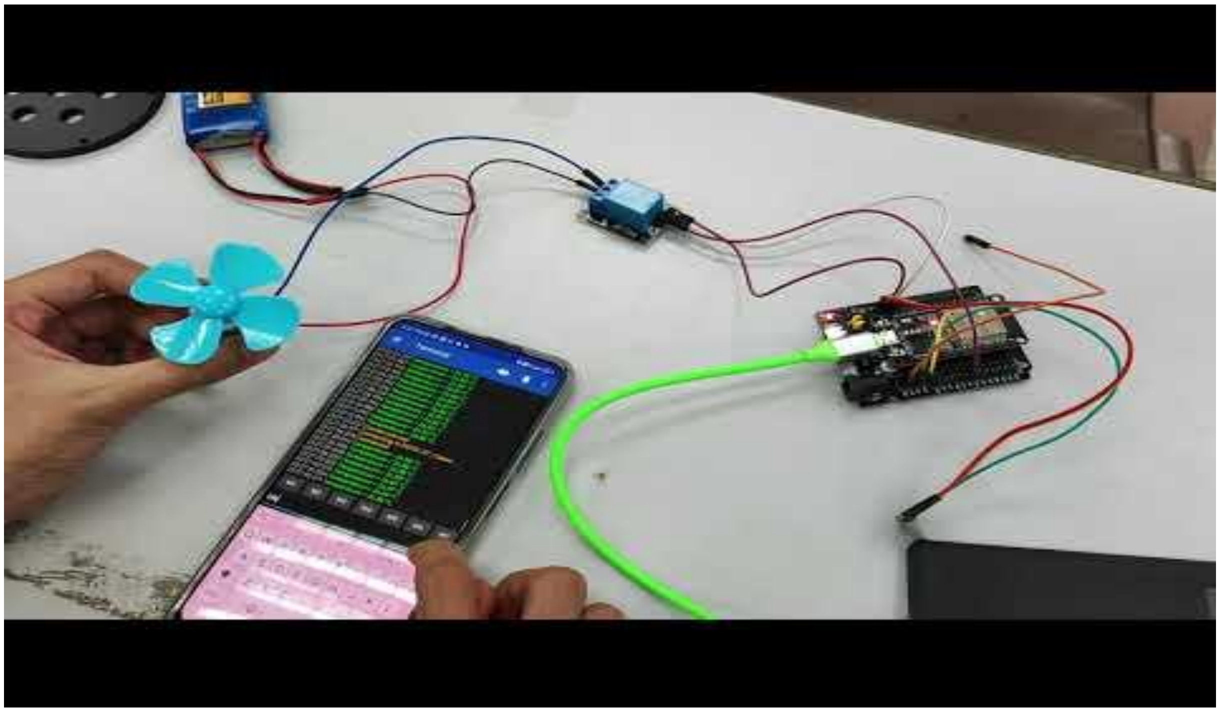
Bluetooth with motor and Bme280 sensor:



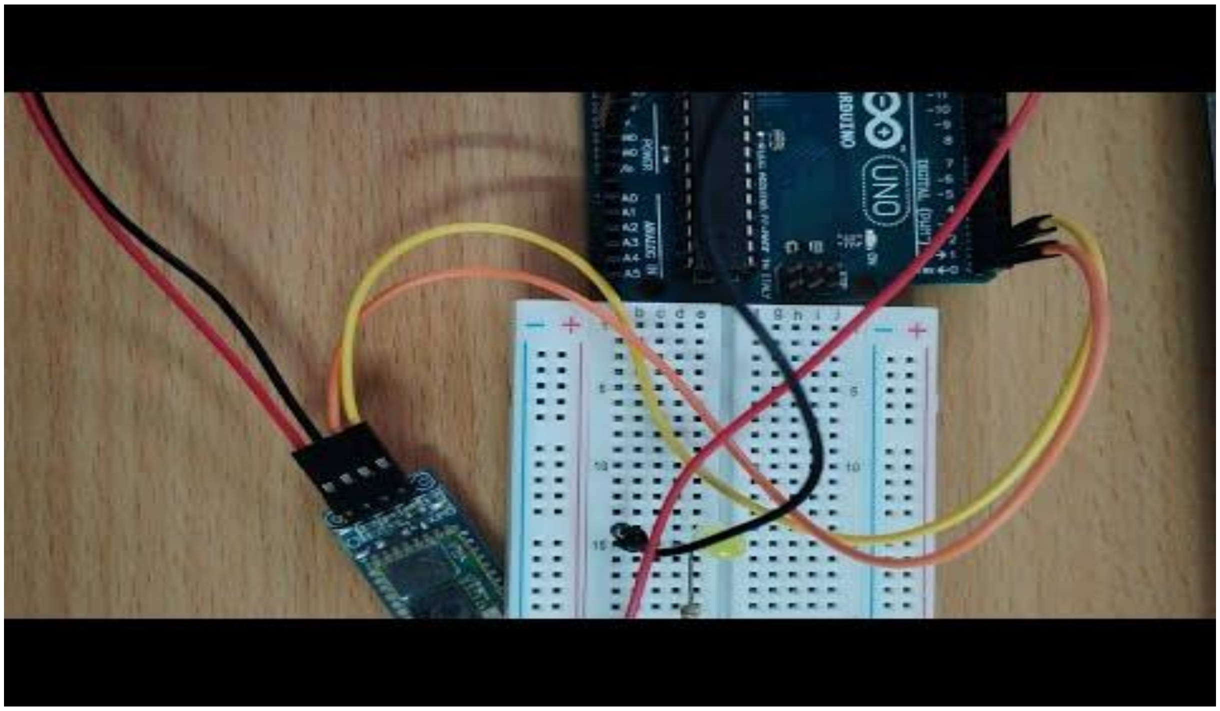
```
22:22:47.242 Temperature = 28.08 °C
22:22:48.200 Temperature = 28.09 °C
22:22:48.852 Temperature = 28.05 °C
22:22:49.896 Temperature = 28.04 °C
22:22:50.900 Temperature = 28.03 °C
22:22:51.903 Temperature = 28.02 °C
22:22:52.872 Temperature = 28.01 °C
22:22:54.142 Temperature = 28.02 °C
22:22:55.160 Temperature = 28.02 °C
22:22:55.866 Temperature = 28.00 °C
22:22:56.891 Temperature = 27.99 °C
22:22:57.919 Temperature = 27.99 °C
22:22:58.918 Temperature = 28.00 °C
22:22:59.873 Temperature = 28.01 °C
22:23:01.073 Temperature = 28.04 °C
22:23:02.084 Temperature = 28.05 °C
22:23:02.905 Temperature = 28.04 °C
22:23:03.916 Temperature = 28.04 °C
22:23:04.906 Temperature = 28.05 °C
22:23:05.898 Temperature = 28.05 °C
22:23:06.876 Temperature = 28.06 °C
22:23:09.020 Temperature = 28.07 °C
22:23:09.514 Temperature = 28.08 °C
22:23:10.015 Temperature = 28.11 °C
22:23:10.919 Temperature = 28.10 °C
22:23:11.875 Temperature = 28.10 °C
22:23:12.891 Temperature = 28.09 °C
22:23:13.893 Temperature = 28.10 °C
22:23:14.893 Temperature = 28.10 °C
22:23:15.920 Temperature = 28.13 °C
22:23:16.909 Temperature = 28.11 °C
22:23:17.900 Temperature = 28.11 °C
22:23:18.889 Temperature = 28.09 °C
22:23:19.900 Temperature = 28.07 °C
22:23:20.887 Temperature = 28.07 °C
22:23:21.900 Temperature = 28.08 °C
22:23:23.346 Temperature = 28.10 °C
22:23:24.335 Temperature = 28.11 °C
22:23:24.902 Temperature = 28.09 °C
22:23:25.899 Temperature = 28.08 °C
```

```
TURN_ON_PIN
```





https://youtu.be/WafsT79lFho?si=WyvbL45rlz_UJmiA



DISCUSSION

Bluetooth data interfacing has become the linchpin in an array of real-world applications, reshaping the landscape of home automation, industrial monitoring, and beyond. In homes, Bluetooth-enabled devices seamlessly communicate, allowing for smart homes where lights, thermostats, and security systems interconnect. In industries, Bluetooth empowers monitoring systems, ensuring efficient and real-time data transfer. Bluetooth provides the freedom for devices to communicate effortlessly, unburdened by physical constraints. Through this task, we were able to understand the role and the versatility of HC-05. The HC-05 module is a compact and powerful component that acts as the bridge between Arduino and the wireless world. Its primary functions include establishing Bluetooth connections, handling data transmission, and ensuring a reliable link between the Arduino and external devices. The HC-05's versatility is a cornerstone of its popularity. Compatible with a range of Arduino projects, its adaptability spans from simple LED control experiments to complex sensor integrations. Its capability to operate in both Master and Slave modes adds a layer of sophistication, making it a go-to choose for diverse applications.

For this Bluetooth part, we did a remote-controlled LED using HC-05 Bluetooth, Arduino and a mobile app. Through Bluetooth commands, we can remotely turn an LED on or off using a smartphone or another Bluetooth-enabled device. For our task, we use Arduino Bluetooth Controller. At first, we want to use Bluetooino and Arduino Bluetooth Control but ours cannot effectively function in these two apps. To check whether the connection and the code was functioning as we want, firstly we send Serial Commands to Arduino Using Serial Monitor. We enter A to turn the LED on and B to turn off the LED. Then, we use the app with the same method by entering A to turn the LED on and B to turn off the LED.

Other than that, we explore the integration of Bluetooth technology with a motor and a BME280 temperature sensor. The primary objective for this is to establish wireless control over the motor and simultaneously monitor environmental temperature using the BME280 sensor. In this experiment, we focus on monitoring temperature data that is sent by BME280 Temperature Sensor. For the motor used, we can control the speed and direction through Bluetooth commands.

Setting up an HC-05 Bluetooth module with Arduino involves a combination of hardware and software configurations. Through this process, we establish a wireless communication link that opens possibilities for our future Arduino projects, such as remote-control applications or data exchange with other Bluetooth-enabled devices.

CONCLUSION

In conclusion, our experiment focused on the integration of Bluetooth technology with Arduino, employing an ESP32 board and an HC-05 Bluetooth module. The goal was to establish wireless communication between Arduino and a smart phone.
The initial phase of the experiment involved a remote-controlled LED using the Arduino Bluetooth Controller app. Through serial commands and the app interface, we were able to demonstrate the capability to turn an LED on and off wirelessly, providing a foundation for understanding basic Bluetooth communication.
Next, to test the applications by integrating Bluetooth control with a motor and a BME280 temperature sensor, the motor was wirelessly controlled using Bluetooth commands ('ON' and 'OFF'). Simultaneously, we monitored environmental temperature wirelessly through the BME280 sensor.

RECOMMENDATIONS

To improve the experiment, to make it more efficient, we should be looking into the Bluetooth Module Configuration. This is so that Bluetooth module (HC-05) can be connected to a phone much faster for optimal performance. By looking into the Bluetooth Module configuration, we can find information on pairing modes, security settings, and ensuring compatibility with the Bluetooth version supported by the smartphone.

REFERENCES

[Makers Group]. (2022, August 24). *Arduino with ThinkSpeak | Monitoring IoT data over ThingSpeak | ThingSpeak with esp8266 | 2022* [Video]. YouTube. https://www.youtube.com/watch?v=764bRMvGZR4

TariqMore, hammadtqHammad. (n.d.). *Remote Controlled LED Using HC-05 Bluetooth, Arduino and Mobile Phone App*. Instructables. https://www.instructables.com/Remotely-Control-LED-using-HC-05-Bluetooth-Arduino/

GeeksforGeeks (2021, October 28). *All about HC-05 Bluetooth Module | Connection with Android*. Retrieved December 16, 2023, from https://www.geeksforgeeks.org/all-about-hc-05-bluetooth-module-connection-with-android/

CircuitSchools (2022, January 11). *What is ESP32, how it works and what you can do with ESP32?* Circuit Schools. Retrieved December 16, 2023, from https://www.circuitschools.com/what-is-esp32-how-it-works-and-what-you-can-do-with-esp32/

APPENDICES

ACKNOWLEDGEMENT

We would like to express our gratitude to Dr. Wahju Sediono, one of the lecturers of this course, mechatronic system integration 1 (MCTA 3203) for his guidance while the experiment is being carried out.

STUDENT'S DECLARATION

This is to certify that we are responsible for the work submitted in this report, that the original work is our own except as specified in the references and acknowledgement, and that the original work contained herein have not been untaken or done by unspecified sources or person.

We hereby certify that this report has not been done by only one individual and all of us have contributed to the report. The length of contribution to the reports by each individual is noted within this certificate.

We also hereby certify that we have read and understand the content of the total report and no further improvement on the reports is needed from any of the individual’s contributor to the report.

We, therefore, agreed unanimously that this report shall be submitted for marking and this final printed report have been verified by us.

Signature: Ghousiah Name: Noorul Ghousiah Binti Noordeen Sahib Matric Number: 2118298 Contribution: abstract, introduction, objective, procedure part A	Read	✓
	Understand	✓
	Agree	✓
Signature: hanis Name: hanis Matric Number: 2020590 Contribution:	Read	✓
	Understand	✓
	Agree	✓
Signature: Afiq Name: Muhammad Afiq bin Ahmad Matric Number: 2115203 Contribution: procedure part B,methodology, results	Read	✓
	Understand	✓
	Agree	✓
Signature: ayunaziera Name: Nur Ayu Naziera Binti Rosli Matric Number: 2119202 Contribution: discussion part b, result part b, procedure of using the app	Read	✓
	Understand	✓
	Agree	✓
Signature:Sarah Name: Sarah Aishah binti Sa'aid Hazley Matric Number: 2117600 Contribution: Recommendation, Conclusion, result part b, discussion part b	Read	✓
	Understand	✓
	Agree	✓