Mechatronics System Integration (MCTA3203)

Week **8**: Bluetooth Data Interfacing

Remote Temperature Monitoring and Control via Bluetooth

Objective:
To create a wireless temperature monitoring system using Bluetooth communication between an ESP32 (or Arduino with HC-05) and a computer/smartphone. The Arduino will read temperature data from a DHT22 sensor, send it over Bluetooth, and receive simple control commands from the paired device.

Required Hardware:

1. ESP32 development board (or Arduino + HC-05 Bluetooth module)
2. Temperature sensor (DHT22)
3. Smartphone or computer with Bluetooth support
4. Power supply for the ESP32/Arduino
5. Breadboard and jumper wires

Experiment Steps:

1. Hardware Setup:
   - Connect the DHT22 sensor to the ESP32/Arduino.
   - Connect the HC-05 Bluetooth module (if using Arduino).
   - Power up the system.

2. Arduino Programming:
   Write an Arduino sketch to:
   - Read temperature data from the DHT22 sensor.
   - Transmit the data over Bluetooth serial connection.
   - Receive simple commands (e.g., "FAN ON", "FAN OFF") via Bluetooth.

3. Bluetooth Programming:
   - Pair the HC-05/ESP32 Bluetooth with your smartphone or PC.
   - Use a serial terminal app (e.g., Serial Bluetooth Terminal on Android, or a Python script on PC) to view temperature data and send commands.

4. Remote Monitoring:
   - Observe real-time temperature readings on your paired device.
   - Send control commands (e.g., to toggle an LED or simulate a fan/heater).

Experiment Workflow:

1. Place the DHT22 sensor in the test environment.
2. Power up the ESP32/Arduino.
3. Pair with your computer/smartphone via Bluetooth.
4. View temperature readings and send remote commands.

Data Collection and Analysis:

- Record temperature changes over time.
- Observe how commands (e.g., "FAN ON") could be used to simulate environment control.

Task

Develop or use a simple Bluetooth terminal app that communicates with the Arduino/ESP32. The app should:

- Display received temperature data.
- Allow sending control commands (e.g., "FAN ON", "FAN OFF").

Example of Arduino Sketch (Bluetooth Communication)

```
#include "DHT.h"
#include <SoftwareSerial.h>

#define DHTPIN 4                    // Data pin connected to DHT22
#define DHTTYPE DHT22               // DHT 22 (AM2302)

DHT dht(DHTPIN, DHTTYPE);

// For Arduino UNO + HC-05 (use pins 2 and 3).
SoftwareSerial bluetooth(2, 3);     // RX, TX
// For ESP32, replace:
// SoftwareSerial bluetooth(2, 3);
// with:
// #define bluetooth Serial2


void setup() {
  Serial.begin(9600);               // For debug via USB
  bluetooth.begin(9600);            // For HC-05
  dht.begin();

  Serial.println("DHT22 + Bluetooth Monitoring Ready");
}

void loop() {
  float temp = dht.readTemperature();   // Default °C
  float hum  = dht.readHumidity();


// --
// Control commands (optional extension)
// In the Arduino code, you can prepare for receiving commands by
// adding something like:

// if (bluetooth.available()) {
//   String cmd = bluetooth.readStringUntil('\n');
//   cmd.trim();
//   if (cmd == "FAN ON") {
//     digitalWrite(LED_BUILTIN, HIGH);
//   } else if (cmd == "FAN OFF") {
//     digitalWrite(LED_BUILTIN, LOW);
```

```
//   }
// }
// --


  if (!isnan(temp) && !isnan(hum)) {
    // Send only numeric temperature for easy parsing in Python
    bluetooth.println(temp);

    // Debug output
    Serial.print("Temp: ");
    Serial.print(temp);
    Serial.print(" °C | Hum: ");
    Serial.print(hum);
    Serial.println(" %");
  }
  else {
    Serial.println("Failed to read from DHT22 sensor!");
  }

  delay(2000);  // Read every 2 seconds
}
```

Python Side:

Write a Python script to read data from the Arduino over serial and display it.

```
import serial
import matplotlib.pyplot as plt

ser = serial.Serial('COMx', 9600)  # adjust as needed

temperatures = []

try:
    while True:
        data = ser.readline().decode('utf-8').strip()
        temperature = float(data)
        temperatures.append(temperature)

        # Display real-time temperature
        print(f"Temperature: {temperature} °C")

except KeyboardInterrupt:
    # Plot the recorded temperatures when the user interrupts the script
    plt.plot(temperatures, marker='o')
    plt.title('Temperature Monitoring')
    plt.xlabel('Time (s)')
    plt.ylabel('Temperature (°C)')
    plt.show()

finally:
    ser.close()
```

References

[1] https://projecthub.arduino.cc/NeilChaudhary/arduino-bluetooth-basic-tutorial-9cff12

Arduino Bluetooth Basic Tutorial

[2] https://howtomechatronics.com/tutorials/arduino/arduino-and-hc-05-bluetooth-module-tutorial/
Arduino and HC-05 Bluetooth Module Complete Tutorial

[3] https://projecthub.arduino.cc/superturis/basic-bluetooth-communication-with-arduino-hc-05-3a431c
Basic Bluetooth communication with Arduino & HC-05