Mechatronics System Integration (MCTA3203)

Week **8**: Bluetooth and Wifi data interfacing with microcontroller and computer based system: Data processing, sensors and actuators.

Remote Temperature Monitoring and Control

Objective: To create a wireless temperature monitoring system using Wi-Fi, Arduino, and a temperature sensor or thermistor. The Arduino will read temperature data from the thermistor, send it to a Python script over Wi-Fi, and the Python script will display and log the temperature.

Materials Needed:

1. Arduino board with Wi-Fi capability (e.g., Arduino ESP8266, Arduino MKR1000, or an ESP32)
2. Temperature sensor (e.g., DHT11 or DHT22)
3. Bluetooth module (e.g., HC-05 or HC-06)
4. Smartphone with Bluetooth support
5. Wi-Fi network and internet access
6. Power supply for the Arduino
7. Breadboard and jumper wires

Experiment Steps:

1. Hardware Setup:
   - Connect the temperature sensor (thermistor) to the Arduino.
   - Connect the Bluetooth module to the Arduino.
   - Connect the Arduino to your Wi-Fi network using the built-in Wi-Fi capabilities.

2. Arduino Programming:
   - Write an Arduino sketch that reads temperature data from the sensor.
   - Set up Wi-Fi connectivity to send temperature data to a cloud service like *ThingSpeak*, where you can create a simple dashboard to visualize the data.

3. Bluetooth Programming:
   - Write an Arduino sketch to enable Bluetooth communication[1].
   - Complete the **task** below.

4. Remote Monitoring:
   - Access your *ThingSpeak* dashboard on your computer or smartphone to remotely monitor the temperature in real-time via the internet.

Experiment Workflow:

1. Place the temperature 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.

Data Collection and Analysis:
Collect and analyze temperature data over time to see how the room's temperature changes based on your remote control inputs.

Task

Develop a simple smartphone application (or use an existing one) that communicates with the Arduino via Bluetooth. This app should allow you to send commands to control a connected device, like a fan or heater, based on the temperature data received from the Arduino.

Source Codes

[1]Below is a simple Arduino sketch to enable Bluetooth communication using the *HC-05 Bluetooth module*. This example uses the **SoftwareSerial** library to create a software serial port for communication with the *HC-05 module*.

```
#include <SoftwareSerial.h>

SoftwareSerial bluetooth(2, 3); // RX, TX

void setup() {
  Serial.begin(9600);   // Serial communication with the computer
  bluetooth.begin(9600); // Serial communication with HC-05 module

  Serial.println("Bluetooth Communication Ready");
}

void loop() {
  // Read data from the computer and send it to HC-05
  if (Serial.available() > 0) {
    char data = Serial.read();
    bluetooth.print(data);
  }

  // Read data from HC-05 and send it to the computer
  if (bluetooth.available() > 0) {
    char data = bluetooth.read();
    Serial.print(data);
  }
}
```

Install SoftwareSerial Library:

- Open the Arduino IDE.
- Go to "Sketch" > "Include Library" > "Manage Libraries..."
- In the Library Manager, search for "SoftwareSerial" and click "Install" for the one by Paul Stoffregen.

Example of Arduino code for *reading temperature from thermistor*

```
const int analogPin = A0;  // Analog pin for thermistor
const int resistorValue = 10;  // Resistance connected to the thermistor


void setup() {
  Serial.begin(9600);
}

void loop() {
  int sensorValue = analogRead(analogPin);
  double voltage = sensorValue * (5.0 / 1023.0);  // Convert to voltage

  // Use the Steinhart-Hart equation to convert voltage to temperature
  double temperature = (1.0 / ((log(voltage / 5.0) / resistorValue) +
      (1.0 / 298.15))) - 273.15;

  // Send temperature data over serial
  Serial.println(temperature);

  delay(1000);  // Delay for 1 second
}
```

Python Side:

Install necessary Python libraries (e.g., pyserial) using pip install pyserial.
Write a Python script to read data from the Arduino over serial and display it.

```
# Python Code

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()
```

Useful Links

[1] https://www.youtube.com/watch?v=jYjuxWUefhg
[2] https://www.youtube.com/watch?v=dJJAQxyryoQ
[3] https://thingspeak.com/
[4] https://nothans.com/thingspeak-tutorials/arduino/send-data-to-thingspeak-with-arduino
[5] https://maker.pro/arduino/projects/how-to-use-thingspeak-and-arduino-to-develop-a-temperature-sensor