REPORT 6: BLUETOOTH AND WIFI DATA INTERFACING WITH MICROCONTROLLER AND COMPUTER BASED SYSTEM

GROUP 4

MCTA 3203

SEMESTER 2 2024/2025

MECHATRONICS SYSTEM INTEGRATION

DATE OF SUBMISSION: 5TH MAY 2025

NO	NAME	MATRIC NUMBER
1.	IRDINA NABIHAH BINTI MOHD NAZRI	2214772
2.	IZZAH ZAHIRA BINTI NORAZLEE	2217696
3.	NOR MAISARAH BINTI ISMAIL	2213080

TABLE OF CONTENTS

INTRODUCTION

In this age of the Internet of Things (IoT), wireless communication has become a vital component of modern embedded systems. Hence, this experiment actually focuses on the integration of Bluetooth and Wi-Fi technologies with a microcontroller-based system for remote temperature monitoring and control. The main objective is to develop a system that reads real-time temperature data using a thermistor and transmits this data to external devices using both Wi-Fi and Bluetooth protocols. Microcontrollers with built-in Wi-Fi capabilities (such as the ESP8266 or ESP32) are used to send temperature readings to a cloud platform like ThingSpeak for real-time visualization and data logging. After that, a Bluetooth module (for example the HC-05) enables communication with a smartphone so that gadgets like a fan or heater can be controlled locally. The Arduino reads the analogue signals from temperature-sensitive resistors called thermocouples and uses the Steinhart-Hart equation to translate them into temperature measurements. In the meantime, the temperature changes are plotted in real time using Python scripting to read this data via serial transmission. Additionally, it is anticipated that this experiment will show dependable remote monitoring based on temperature thresholds, successful wireless data transmission via both Wi-Fi and Bluetooth, and discernible trends in ambient temperature over time. Therefore, the result will undoubtedly demonstrate how dual wireless communication channels can be used efficiently for data collection, processing, and control in a simple Internet of Things application.

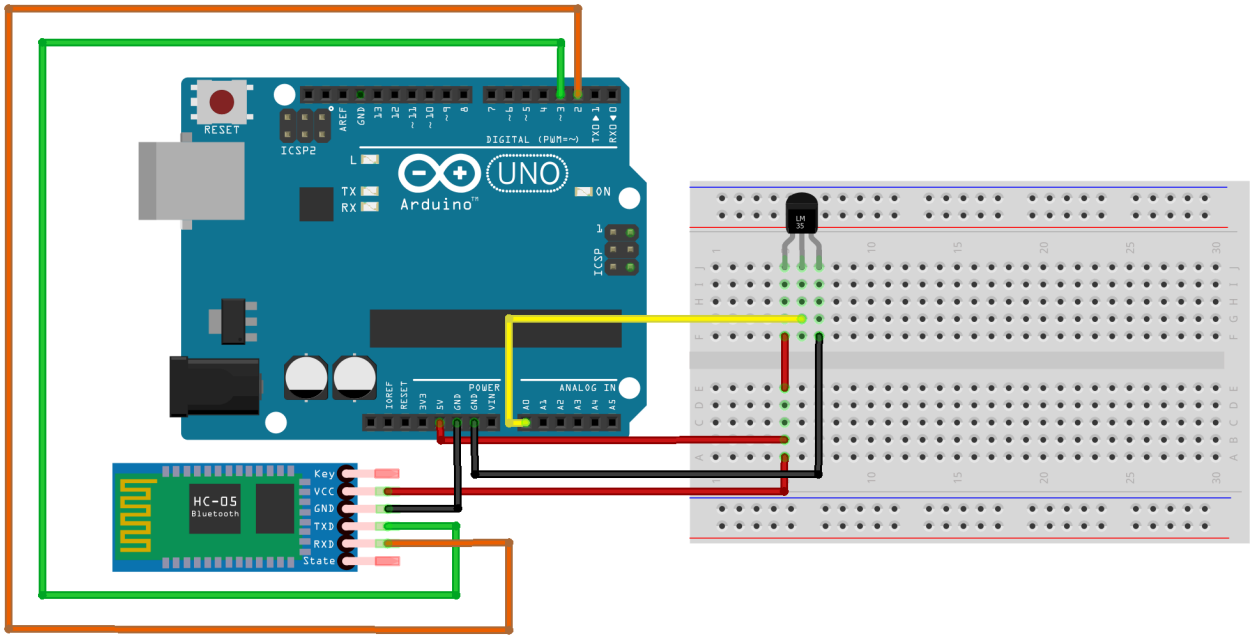
ABSTRACT

This experiment focuses on developing a wireless temperature monitoring and control system using Arduino with Wi-Fi and Bluetooth capabilities. A temperature sensor is interfaced with the Arduino to read ambient temperature data, which is then transmitted wirelessly over Wi-Fi to a cloud based dashboard for real time monitoring. Simultaneously Bluetooth communication is implemented to allow a smartphone application to send control commands to the Arduino, enabling manual or automated control of connected actuators such as fans or heaters. The system demonstrates the integration of data acquisition, wireless communication, and remote control in a microcontroller based setup, highlighting applications in IoT enabled environmental monitoring. In simple words, the objective of this experiment is 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 AND EQUIPMENT

1. Arduino board with Wi-Fi capability
2. Temperature sensor
3. Bluetooth module
4. Smartphone and Bluetooth support
5. Wi-Fi network and internet access
6. Power supply for the Arduino
7. Breadboard and jumper wires.

EXPERIMENTAL SETUP



1. Connect the LM35 Temperature Sensor to the breadboard with VOUT to A0, VIN to 5V and ground to GND on Arduino.

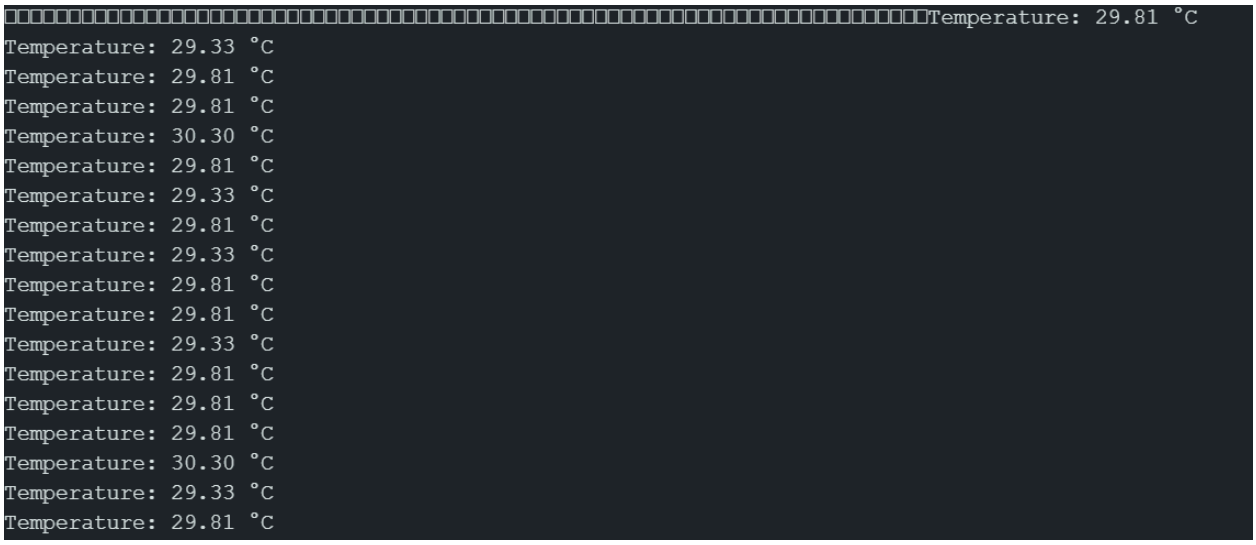
2. Connect the HC-05 Bluetooth Module with VCC to 5V, TX (transmit) to Arduino's RX (pin 2) via a voltage divider and RX (receive) to Arduino's TX (pin 3) directly.

3. Connect the Arduino to a power source using the USB cable.

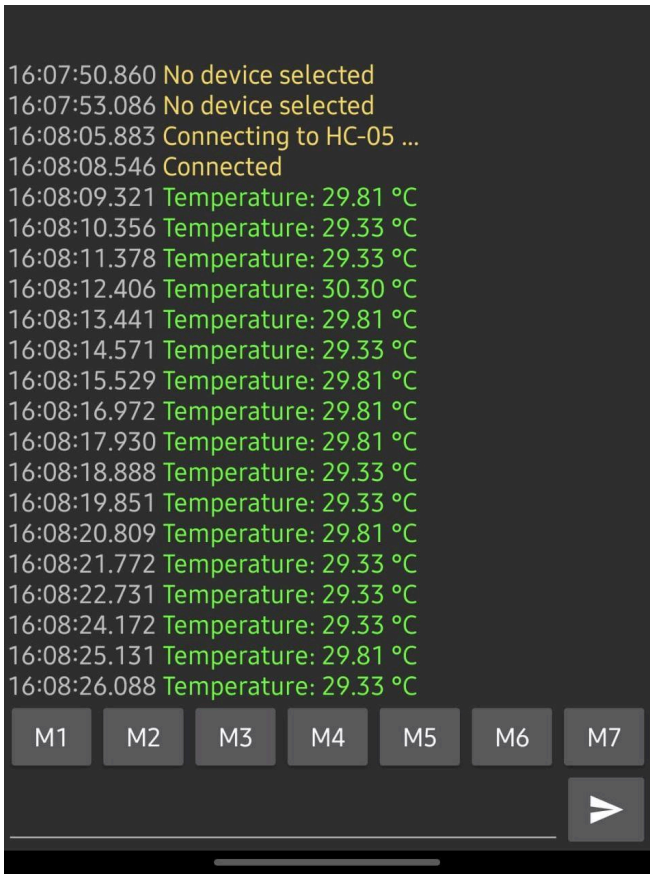
METHODOLOGY

1. **Hardware Installation:** Begin with the installation of a temperature sensor (thermistor) and a Bluetooth module on the Arduino. Additionally, install the Arduino to join a Wi-Fi network for data exchange.
2. **Arduino Programming:** Then, develop an Arduino sketch that reads temperature data from the sensor. The sketch should also enable Wi-Fi communication, such that the Arduino posts the collected temperature data to a cloud service and users can view it on a dashboard.
3. **Bluetooth Integration:** For this test, develop an Arduino sketch that enables Bluetooth communication between the Arduino and other Bluetooth-enabled devices such as smartphones.
4. **Data Collection and Analysis:** The participants will take temperature readings over time and analyze it to observe how the room temperature responds to remote control commands.
5. **Task Development:** Finally, the participants will develop a basic mobile phone application that can talk to the Arduino using Bluetooth. The application will send commands to drive connected devices such as fans or heaters from the temperature readings obtained.

RESULTS



Serial Monitor in Arduino IDE



Serial Monitor in Phone App using Bluetooth

```ino
#include <SoftwareSerial.h>

const int lm35Pin = A0; // LM35 connected to A0
SoftwareSerial bluetooth(3, 2); // RX, TX for HC-05

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

void loop() {
  int sensorValue = analogRead(lm35Pin);
  float voltage = sensorValue * (5.0 / 1023.0); // Convert to voltage
  float temperatureC = voltage * 100; // LM35 outputs 10mV/°C

  // Send temperature to Bluetooth
  // bluetooth.print("Temperature: ");
  // bluetooth.print(temperatureC);
  // bluetooth.println(" °C");
  Serial.print("Temperature: ");
  Serial.print(temperatureC);
  Serial.println(" °C");

  delay(1000); // Send every 1 second
}
```

Arduino Code

This code reads temperature data from an LM35 temperature sensor and transmits it to a Bluetooth terminal via an HC-05 module using the SoftwareSerial library. It sets up a software serial connection on pins 3 (RX) and 2 (TX) for Bluetooth communication. The LM35 sensor, connected to analog pin A0, outputs an analog voltage proportional to the temperature. In the loop(), the analog signal is read and converted into a temperature value in Celsius based on the LM35's sensitivity of 10 mV/°C. The temperature is then printed to the Serial Monitor for debugging and sent to the Bluetooth terminal for wireless monitoring. A delay of 1 seconds is added between readings to ensure periodic updates.

```python
import serial
import time

# Replace 'COMx' with your actual COM port (e.g., COM5 on Windows, /dev/ttyUSB0 on Linux)
bluetooth_port = 'COM7'
baud_rate = 9600

try:
    bt = serial.Serial(bluetooth_port, baud_rate)
    print("Connected to Bluetooth device.")
    time.sleep(2)  # Wait for connection to stabilize

    while True:
        if bt.in_waiting:
            data = bt.readline().decode('utf-8').strip()
            print(data)

except serial.SerialException as e:
    print(f"Serial error: {e}")
except KeyboardInterrupt:
    print("Exiting program.")
finally:
    if 'bt' in locals() and bt.is_open:
        bt.close()
```

Python Code

This Python script reads temperature data from an LM35 sensor connected to an Arduino via a Bluetooth module, such as the HC-05. The script uses the serial library to establish Bluetooth communication on COM7 with a baud rate of 9600.

DISCUSSION

In this experiment, the findings show that a wireless temperature monitoring system that combines an Arduino microcontroller, an LM35 temperature sensor, and an HC-01 Bluetooth module is feasible. For real-time monitoring, the device successfully recorded temperature readings and wirelessly sent the information to a smartphone app. Therefore, this demonstrates how well sensor technology, communication protocols, and embedded systems work together to provide loT-based environmental monitoring. The system's scalability for more intricate loT applications, like adding more sensors or utilising Wi-Fi to increase the communication range, is suggested by its modular and economical architecture. Additionally, its potential for automation and resource management is highlighted by the ability to remotely control appliances like fans or heaters, which is in line with the objectives of contemporary smart systems.

Despite the system's usually good performance, testing revealed numerous irregularities, such as sporadic data transmission delays and inconsistent temperature readings. Limitations in the baud rate setup or Bluetooth signal interference were probably the sources of the delays. Environmental factors, such as abrupt changes in airflow or the positioning of the sensor close to heat sources, may have caused temperature inconsistencies by skewing the measurements. These differences highlight how crucial it is to optimise system configuration and give careful thought to the operating environment in order to reduce outside influences. Numerous restrictions and error sources were found. Despite its dependability, the LM35 sensor is vulnerable to noise from the environment and changes in the surrounding conditions. Reducing inaccuracies may be possible with improved shielding or calibration. Communication dependability is impacted by the HC-01 Bluetooth module's short range and weak signal, which worsens with distance or physical obstructions. Additionally, minor errors in data transfer might have been generated by the setup's usage of resistors for voltage control. On the software side, the Arduino code's 2-second latency restricts the system's capacity to react to abrupt changes in the environment, even though it is helpful for periodic updates. It was also challenging to deal with anomalies like sudden disconnects or corrupted data packets due to the lack of strong error-checking procedures. Furthermore, the smartphone application, while functional, lacked advanced features such as visual alerts for critical temperature thresholds, which could improve user experience.

Since the experiment was carried out in a controlled setting, environmental factors also presented challenges. Additional variability in data accuracy and transmission stability may be introduced by real-world factors like temperature gradients, humidity, or physical obstacles. There are various ways to improve in order to overcome these constraints. A more sophisticated temperature sensor, such the DHT22, could provide more accurate temperature and humidity data. A Wi-Fi-capable microcontroller, like the ESP32, would enable integration with cloud-based data storage and increase the communication range. Subsequently, implementing error-handling routines in the code would enhance the system's ability to detect and resolve communication failures or incorrect data formats. Enhancements to the smartphone application, such as graphical elements, historical data tracking, and temperature alerts, would improve usability and functionality.

To sum up, our experiment showed the promise of wireless monitoring systems provided by the Internet of Things, emphasising its capacity to efficiently carry out remote control and real-time monitoring. Nonetheless, the disparities and constraints noted highlight the necessity of additional hardware, software, and environmental adaptability optimisation. Hence, by resolving these problems, the system may develop into a more durable and dependable option for a range of uses in environmental management, industrial monitoring, and smart homes.

CONCLUSION

In conclusion, this experiment successfully demonstrated the integration of Bluetooth and Wi-Fi communication technologies with a microcontroller to enable real-time temperature monitoring and remote device control. By interfacing an LM35 temperature sensor with an Arduino and using both HC-05 Bluetooth and Wi-Fi connectivity, the system achieved reliable data acquisition, wireless transmission, and remote actuation capabilities. The experiment highlights the practicality and potential of Internet of Things (IoT)-based systems in environmental monitoring and automation.

Despite minor challenges such as transmission delays and sensor sensitivity, the system proved effective in showcasing how embedded hardware and software can collaborate to build scalable and modular solutions. With appropriate enhancements in sensor selection, error-handling routines, and mobile application features, the proposed setup can evolve into a robust platform for smart home, industrial, or environmental applications. Overall, the project reinforces the relevance of wireless interfacing in modern mechatronic systems and lays a solid foundation for future IoT developments.

RECOMMENDATION

There are many recommendations to do to improve this experiment. One of the recommendations to improve this experiment is by upgrading the temperature sensor from the LM35 to a more accurate and versatile sensor. It allows for more reliable measurements and the inclusion of additional environmental parameters like humidity and pressure. For example DHT22 and BME280 which can increase the system's accuracy and applicability in broader monitoring contexts. Other than that, by replacing the Arduino with a more advanced microcontroller like ESP32 which includes built-in Wi-Fi and Bluetooth. It would streamline the hardware setup and provide better processing capabilities and more reliable wireless communication. Next is enhancing the data transmission process through higher baud rates and implementing error-handling protocols. It would reduce communication delays and minimize data loss or corruption. This is particularly important for maintaining real-time data logging. The next recommendation is by integrating the system with a cloud-based platform like ThingSpeak or Firebase. It allows for continuous remote monitoring and data access from anywhere with an internet connection. By expanding the system to include additional sensors and incorporating simple decision-making algorithms or machine learning models would increase its functionality which can enable broader applications in areas such as smart homes, environmental monitoring, and industrial automation.

ACKNOWLEDGEMENTS

A special thanks goes out to Dr. Wahju Sediono and Dr. Zulkifli Bin Zainal Abidin, my teaching assistant, and my peers for their invaluable help and support in finishing this report. Their advice, feedback, and experience have greatly influenced the level of quality and understanding of this work. Their time, patience, and commitment to supporting my academic success are greatly appreciated.

STUDENT'S DECLARATION

Certificate of Originality and Authenticity

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 persons.

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 contributors to the report.

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

Signature: 
Name: IRDINA NABIHAH BINTI MOHD NAZRI
Matric Number: 2214772

Read ☑
Understand ☑
Agree ☑

Signature: 
Name: IZZAH ZAHIRA BINTI NORAZLEE
Matric Number: 2217696

Read ☑
Understand ☑
Agree ☑

Signature: 
Name: NOR MAISARAH BINTI ISMAIL
Matric Number: 2213080

Read ☑
Understand ☑
Agree ☑