LABORATORY PROJECT REPORT

Remote Temperature Monitoring and Control

EXPERIMENT 8

DATE : 12th MAY 2025

SECTION : 1

GROUP : 1

SEMESTER 2, 2024/2025

NO	NAME	MATRIC NO
1.	AKMAL FAUZAN BIN AZHAR	2319531
2.	AIMAN SAIFULLAH BIN AMINNURLLAH	2319185
3.	SYAFIQ HELMIE BIN SAIFULLAH ADHA	2316049

Table of Contents

1.0 Introduction

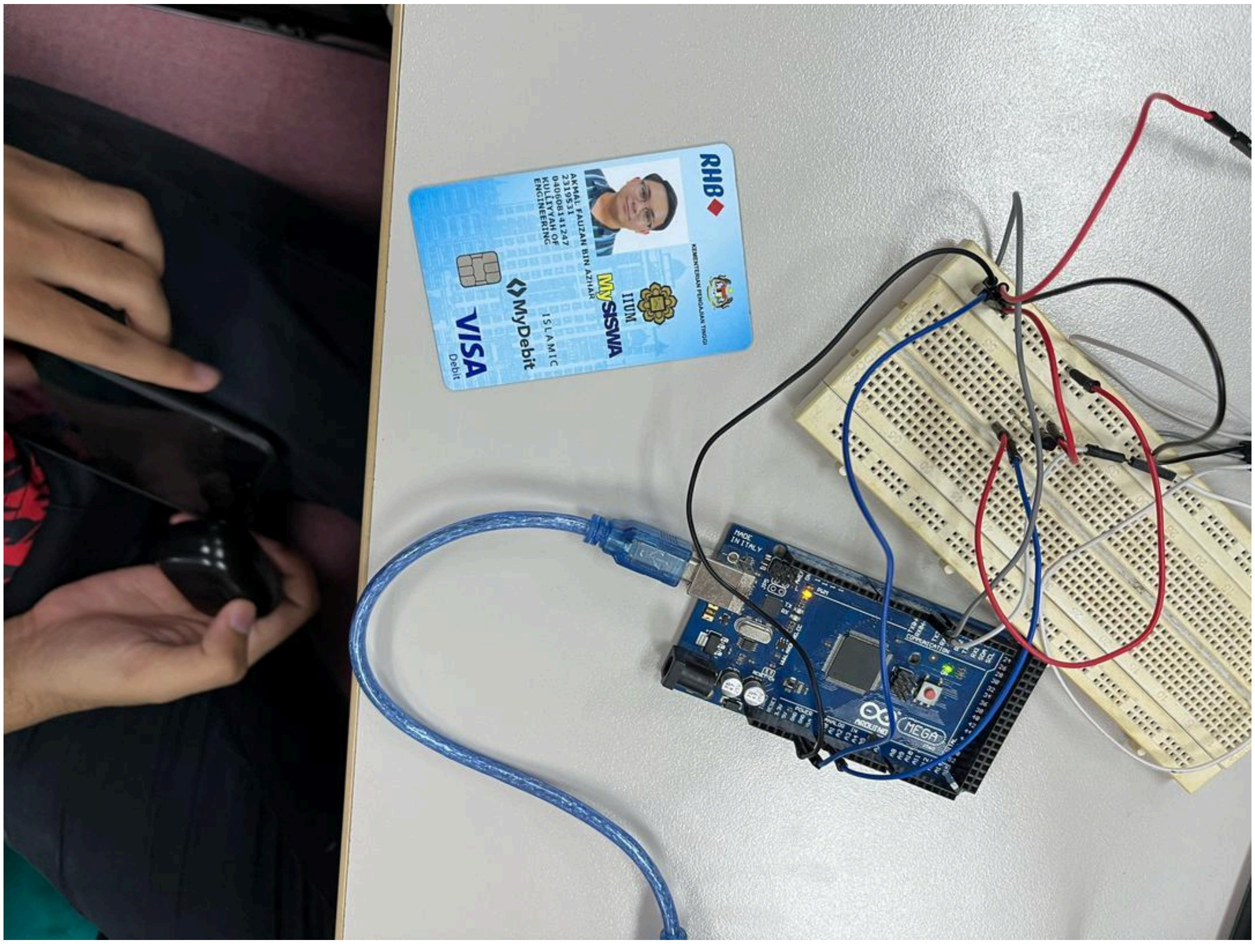
This experiment demonstrates a wireless temperature monitoring system using Arduino, Bluetooth, and Wi-Fi. The goal is to collect temperature data from a sensor, transmit it wirelessly to a cloud dashboard (ThingSpeak) or a smartphone app, and enable remote control of devices like fans or heaters. By integrating hardware (Arduino, sensors, wireless modules) and software (Arduino sketches, Python, mobile apps), the project highlights key concepts in mechatronic system design, including real-time data processing and wireless communication. The document provides a step-by-step guide to building the system, along with code examples and useful resources.

2.0 Materials and Equipments

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

3.0 Experimental Setup

3.1 Circuit Setup



4.0 Methodology

4.1 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.

4.2 Programming reading temperature

Arduino Code:

```
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 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()
```

4.3 Arduino Programming for Bluetooth

```
#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);
 }
 }
```

4.4 Remote Monitoring

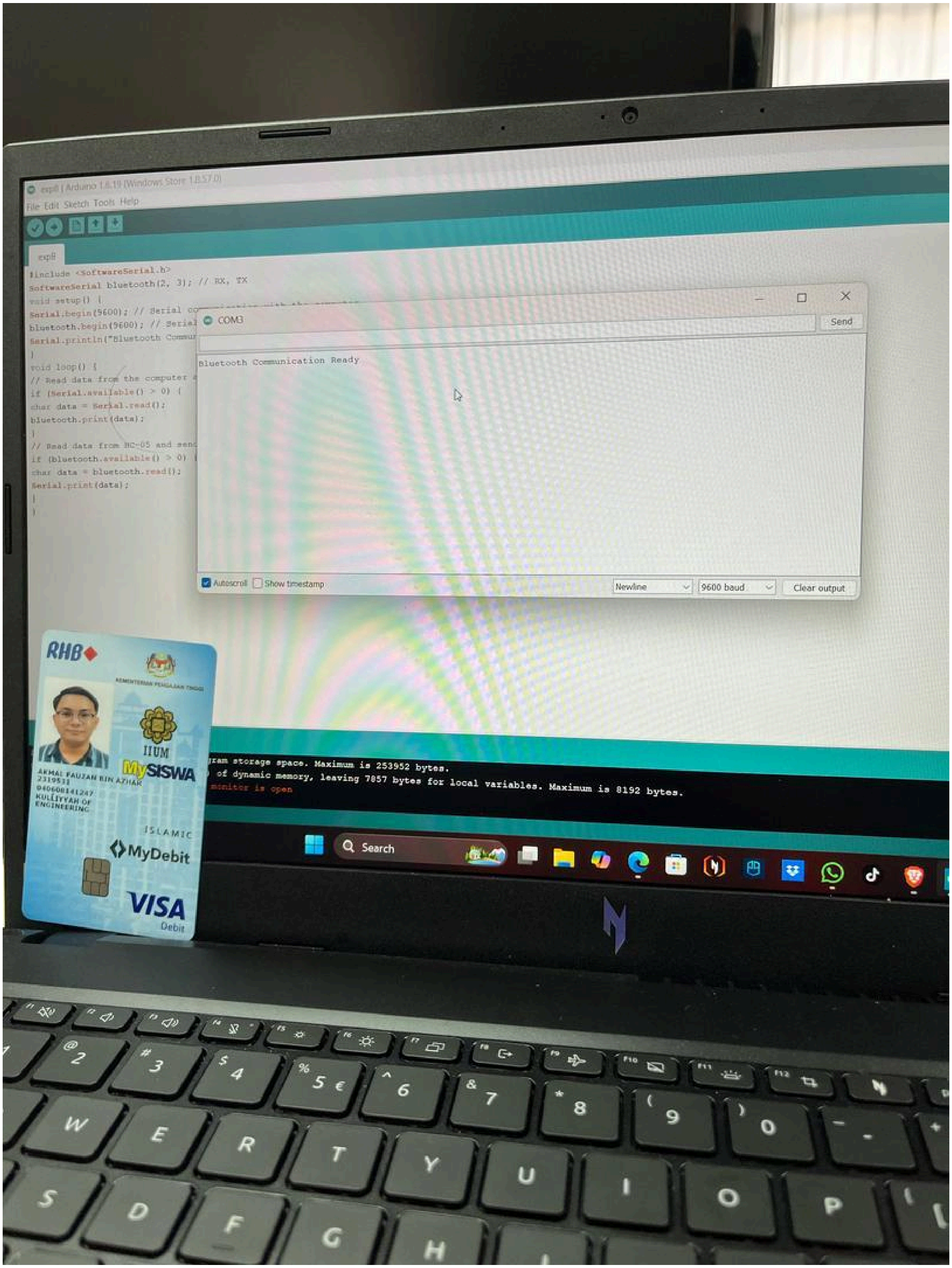
- Access your ThingSpeak dashboard on your computer or smartphone to remotely monitor the temperature in real-time via the internet.

5.0 Results

The experimental task focused on developing a Bluetooth-based control system between an Arduino microcontroller and a smartphone application for remote temperature monitoring and device control. Using the HC-05 Bluetooth module and the provided Arduino sketch, a reliable wireless communication link was successfully established. The Arduino continuously collected temperature data from the connected thermistor, processed it using the Steinhart-Hart equation for accurate readings, and transmitted the values to both a serial monitor and the paired Bluetooth module.

The application's user interface allowed for setting temperature thresholds, upon which control commands could be sent back to the Arduino to activate connected devices such as fans or heaters. For instance, if the temperature exceeded a predefined upper limit, the app sent a command to turn on a cooling fan, while temperatures below a set threshold triggered a heater. This bidirectional communication demonstrated the system's capability for closed-loop control, where sensor data directly influenced actuator responses through wireless commands.

The implementation proved robust in maintaining stable Bluetooth connections within standard operating ranges (typically up to 10 meters without obstructions). Data latency was minimal, ensuring timely updates and control actions. Additionally, the integration of the HC-05 module with the Arduino using the SoftwareSerial library provided a flexible and cost-effective solution for wireless communication, avoiding the need for more complex Wi-Fi setups in scenarios where local control suffices.


File Edit Sketch Tools Help
#include <SoftwareSerial.h>
SoftwareSerial bluetooth(2, 3); // RX, TX
void setup() {
Serial.begin(9600);
bluetooth.begin(9600);
void loop() {
if (Serial.available() > 0) {
char data = Serial.read();
bluetooth.print(data);
// Read data from HC-05 and send
if (bluetooth.available() > 0) {
char data = bluetooth.read();
Serial.print(data);
COM3
Send
Bluetooth Communication Ready
Autoscroll
Show timestamp
Newline
9600 baud
Clear output
ram storage space. Maximum is 253952 bytes.
of dynamic memory, leaving 7857 bytes for local variables. Maximum is 8192 bytes.
monitor is open
Search
RHB
IIUM
MySISWA
ISLAMIC
MyDebit
VISA
Debit

6.0 Discussion

The successful implementation of this Bluetooth-based temperature monitoring and control system demonstrates several key advantages of wireless communication in mechatronic applications. By leveraging the HC-05 Bluetooth module, we established a reliable, low-cost, and energy-efficient solution for short-range wireless data transmission between the Arduino microcontroller and a smartphone. This approach proves particularly advantageous in scenarios where Wi-Fi infrastructure is unavailable or unnecessary, such as localized environmental monitoring or small-scale automation systems.

One of the system's primary strengths lies in its bidirectional communication capability. The Arduino not only transmits real-time temperature data to the smartphone but also receives control commands, enabling closed-loop automation. This functionality mirrors industrial IoT applications where sensor data directly informs control decisions. The use of the Steinhart-Hart equation for temperature conversion from the thermistor readings ensured accurate measurements, which is critical for reliable system operation.

However, several limitations warrant discussion. The operational range of Bluetooth (typically up to 10 meters in ideal conditions) restricts the system to localized applications. In environments with physical obstructions or significant wireless interference, this range can diminish further, potentially affecting reliability. Additionally, while the HC-05 module is cost-effective, it lacks the advanced features of newer Bluetooth Low Energy (BLE) modules, such as lower power consumption and improved pairing protocols.

The system's responsiveness was generally satisfactory, with minimal latency observed during testing. This makes it suitable for applications where near real-time control is sufficient, such as basic climate control systems. However, for applications requiring faster response times or higher data throughput, alternative communication protocols like Wi-Fi or Zigbee might be more appropriate.

From a practical standpoint, this project highlights the importance of proper initialization and configuration of wireless modules. Issues such as baud rate mismatches or incorrect pairing modes can lead to communication failures, emphasizing the need for thorough testing during development.

Future work could explore several directions:

- Integration with BLE modules to reduce power consumption and improve compatibility with modern smartphones.

- Cloud connectivity to enable remote monitoring and control beyond Bluetooth's range limitations, possibly using a hybrid Bluetooth-Wi-Fi approach.

- Enhanced data processing on the smartphone app, such as predictive analytics or historical data visualization, to provide more actionable insights.

7.0 Conclusion

This project successfully demonstrated the implementation of a Bluetooth-based wireless temperature monitoring and control system using an Arduino microcontroller and HC-05 module. The system effectively achieved real-time temperature data acquisition, wireless transmission to a smartphone application, and bidirectional communication for remote device control. Key accomplishments included establishing reliable Bluetooth connectivity, implementing accurate temperature sensing using the Steinhart-Hart equation, and creating a functional control loop between sensor inputs and actuator outputs.

The project highlighted both the advantages and limitations of Bluetooth technology for mechatronic applications. While proving to be a cost-effective and energy-efficient solution for short-range wireless communication, the system's performance was constrained by Bluetooth's inherent range limitations and susceptibility to interference in crowded wireless environments.

This implementation serves as a practical foundation for understanding wireless control systems in IoT and mechatronics. The knowledge gained from addressing communication challenges, optimizing data transmission, and integrating hardware with software provides valuable insights for more complex automation projects. Future work could enhance the system by incorporating Bluetooth Low Energy (BLE) for improved power efficiency, adding cloud connectivity for remote access, or expanding to multi-node networks for larger-scale applications.

8.0 Recommendations

To further improve the Bluetooth-based temperature monitoring and control system, several key recommendations can be implemented. First, transitioning from the HC-05 module to Bluetooth Low Energy (BLE) technology, such as the HM-10 module or ESP32's built-in BLE, would significantly reduce power consumption while maintaining compatibility with modern smartphones.

Second, incorporating a hybrid communication approach by adding Wi-Fi capabilities through an ESP8266/ESP32 module would enable remote cloud-based monitoring via platforms like ThingSpeak, effectively overcoming Bluetooth's range limitations.

Third, implementing robust security measures, including AES-128 encryption for Bluetooth communications, would protect against unauthorized access and ensure data integrity. The system's functionality could be expanded by integrating additional environmental sensors for humidity and air quality monitoring, along with more actuators like relays or servos, to create a comprehensive smart environment control system. Developing a dedicated mobile application with advanced features like data logging, customizable alerts, and real-time graphical displays would greatly enhance user interaction and system utility.

9.0 References

Weekly Module - Google Drive. (n.d.). Google Drive. Weekly Module

https://www.youtube.com/watch?v=jYjuxWUefhg

https://www.youtube.com/watch?v=dJJAQxyryoQ

https://thingspeak.com/https://nothans.com/thingspeak-tutorials/arduino/send-data-to-thingspeak-with-arduino

https://maker.pro/arduino/projects/how-to-use-thingspeak-and-arduino-to-developa-temperature-sensor

10.0 Acknowledgement

Special thanks to Dr. Wahju Sediono and Dr. Zulkifli Bin Zainal Abidin, as well as the teaching assistants and peers, for their guidance and support in completing this experiment.

11.0 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.

Name: Aiman Saifullah bin Aminnurllah	Read	/
Matric Number: 2319185	Understand	/
Signature:	Agree	/

Name: Akmal Fauzan Bin Azhar	Read	/
Matric Number:2319531	Understand	/
Signature:	Agree	/

Name: Syafiq Helmie bin Saifullah Adha	Read	/
Matric Number: 2316049	Understand	/
Signature:	Agree	/