Lab Report: **Sensor Interfacing with Raspberry Pi and Arduino**

Rohit Purushottam Sorte
12504420
MSS-M-1: Advanced Programming (SS25)
Instructor: Prof. Tobias Schaffer

23 June 2025

1 Introduction

The objective is to demonstrate embedded sensor interfacing using the Arduino Nano 33 BLE Sense Rev2 for sensor data acquisition and processing. The Arduino Nano 33 BLE Sense Rev2 is packed with sensors, and two of the most powerful onboard are the Inertial Measurement Unit(BMI270+BMM150) and the temperature sensor(HS3003). The task is to utilize the aforementioned sensors to obtain data and demonstrate sensor interfacing capabilities.

2 Methodology

2.1 Software and Hardware Used

- Programming language: C++
- Libraries: Arduino_BMI270_BMM150.h, <Arduino_HS300x.h>
- Hardware: Arduino Nano 33 BLE Sense Rev2 (CPU: 64 MHz Arm® Cortex®-M4F)

2.2 Code Repository

The full source code for this project is available on GitHub at:

`https://github.com/RobotaBull/Lab-Report-2-Final.git`

This repository includes:

- Source Code Files
- Installation Instructions (Readme)
- Arduino® Nano 33 BLE Sense Rev2 Datasheet
- Sample output from Serial Monitor
- Final Lab Report

2.3 Summery of Experiment

The first experiment reads and displays acceleration data from an IMU (Inertial Measurement Unit) sensor, specifically the BMI270 and BMM150. The setup() function initializes serial communication at 9600 baud and attempts to connect to the IMU. If the IMU does not initialize, the program is stopped. The loop() function continuously checks for available acceleration data. When the data is ready, it reads the acceleration values of X, Y, and Z and prints them to the serial monitor. There is a 500-millisecond delay between each reading, resulting in approximately two updates per second. This code provides a basic framework for monitoring motion and orientation with the specified IMU sensor.

The second experiment reads and displays data from two different sensors: an IMU (Inertial Measurement Unit) for acceleration and a HS3003 sensor for temperature and humidity. The setup() function first initializes serial communication at 9600 baud. Then it attempts to initialize the IMU (BMI270/BMM150). If that fails, it proceeds to try initializing the HS3003 temperature and humidity sensor. If either sensor initialization fails, the program halts. In the loop() function, the code continuously checks if acceleration data is available from the IMU. When it is, it reads the acceleration values for X, Y, and Z. During the same time, it reads the temperature and humidity of the HS3003 sensor. All of these sensor readings (acceleration, temperature, and humidity) are then printed on the serial monitor. To control the update rate, there is a delay(2000) (2 seconds) after printing the sensor data when acceleration is available. There is also a delay(500) (0.5 seconds) at the end of the loop, which will apply if acceleration data is not immediately available, or as a minimum delay between checks. This setup allows for periodic monitoring of both motion and environmental conditions.

2.4 Code Implementation

The following code snippets demonstrate important parts of the implementation.

```
# C++ code for reading IMU Sensor Data

# include "Arduino_BMI270_BMM150.h"
void setup() {
  Serial.begin(9600);
  while (!Serial);
  if (!IMU.begin()) {
    Serial.println("IMU␣initialization␣failed!");
    while (1);
  }
}
void loop() {
  float x, y, z;
  if (IMU.accelerationAvailable()) {
    IMU.readAcceleration(x, y, z);
    Serial.print("Acceleration:␣X=");
    Serial.print(x);
    Serial.print("␣Y=");
    Serial.print(y);
    Serial.print("␣Z=");
    Serial.println(z);
  }
```

```
  delay(500);
}
```

```
# C++ code for reading IMU Sensor, Temperature and Humidity Data

#include <Arduino_HS300x.h>
#include "Arduino_BMI270_BMM150.h"
void setup() {
  Serial.begin(9600);
  while (!Serial);
  if (!IMU.begin()) {
    Serial.println("␣IMU␣initialization␣␣failed␣!");
    if (!HS300x.begin()) {
      Serial.println("␣HS300␣␣sensor␣␣initialization␣␣failed␣!");
      while (1);
    }
  }
}
void loop() {
  float x, y, z;
  float temp = HS300x.readTemperature();
  float humidity = HS300x.readHumidity();
  if (IMU.accelerationAvailable()) {
    IMU.readAcceleration(x, y, z);
    Serial.print("␣Acceleration␣:␣X=");
    Serial.print(x);
    Serial.print("␣Y=");
    Serial.print(y);
    Serial.print("␣Z=");
    Serial.println(z);
    Serial.print("␣Temp␣:");
    Serial.print(temp);
    Serial.print("C");
    Serial.print("␣Humidity␣:");
    Serial.print(humidity);
    Serial.println("%");
    delay(2000);
  }
  delay(500);
}
```

3 Results

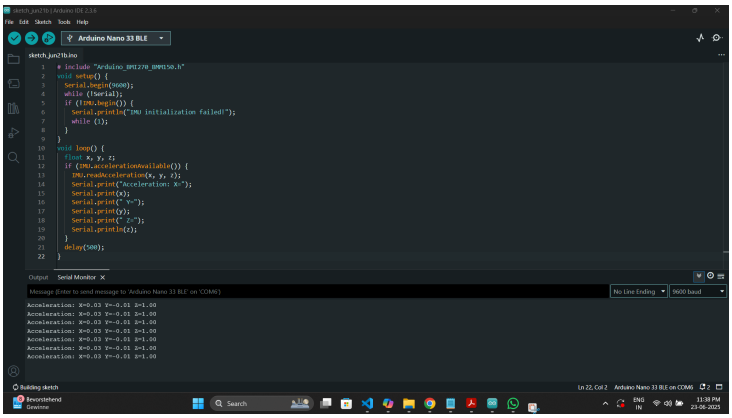

Figure 1: Reading IMU Sensor Data.

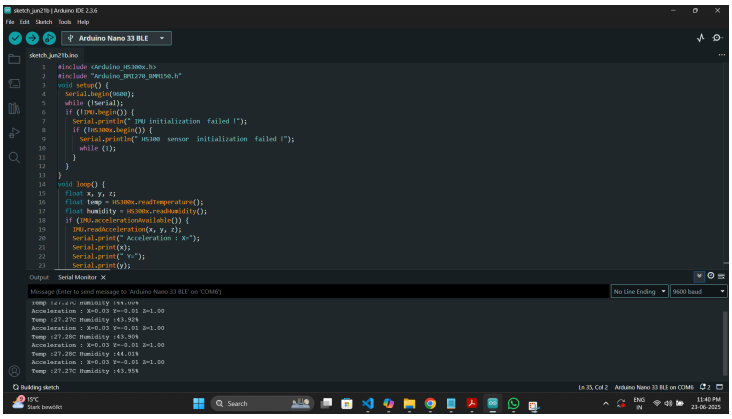

Figure 2: Reading IMU Sensor, Temperature and Humidity Data.

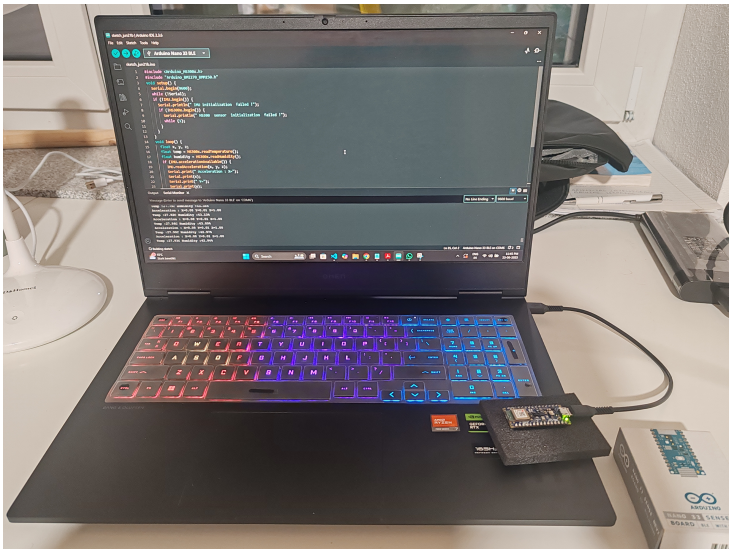

Figure 3: Physical Experimental Setup.

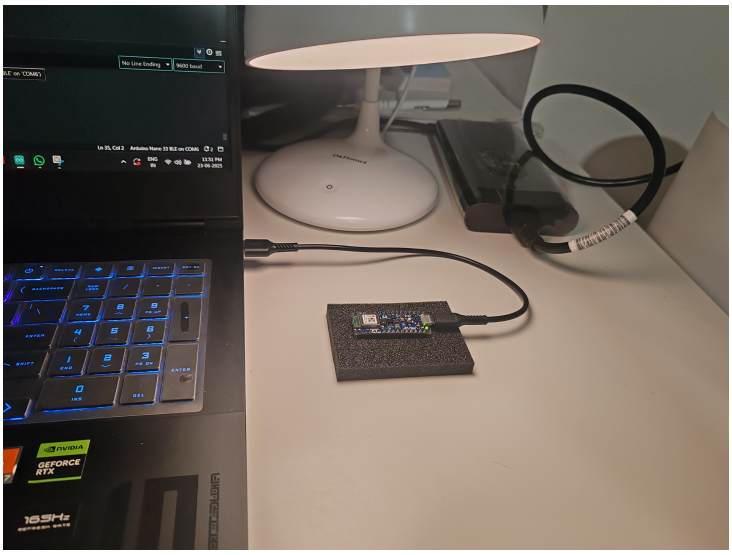

Figure 4: Sensors subject to external light source

4 Challenges, Limitations, and Error Analysis

During the implementation of this project, several challenges and errors were encountered. Below are some key points:

4.1 Challenges Faced

- The IMU sensor didn't work at first, it showed an error during compile time. This was fixed by installing the missing library.
- Temperature values jumped when the board is touched because the sensor is sensitive to heat.
- Combining both sensors in one sketch caused some issues, which were fixed by organizing the code better.
- No plots were observed on serial plotter although values were observed on serial monitor.
- It was a challenge to figure out geometric orientation of X,Y and Z acceleration direction for IMU unit. Axes are figured out by shaking arduino along different planes in 3 dimensions.

4.2 Error Analysis

Some common errors that occurred during the development:

- Sometimes the Arduino Nano didn't show up in the port list. Restarting the Arduino IDE fixed it.
- An error appeared during uploading because the wrong board type was selected in the Tools menu. Selecting "Arduino Nano 33 BLE Sense" solved it.

- When printing both IMU and humidity data together, the Serial Monitor looked messy. Adding clear labels and spacing in the program code lines improved readability.

4.3 Limitations of the Implementation

- The data was printed in plain text without formatting or color, which made it harder to quickly understand trends or changes.
- All measurements were done indoors, so we don't know how the sensors would behave in outdoor or extreme conditions.
- The sensors were tested for a short time; long-term performance or stability was not checked.

5 Discussion

This lab showed how to use the sensors on the Arduino Nano 33 BLE Sense board. The motion sensor (IMU) gave good values when the board was still or moved. The temperature and humidity sensor also worked well and gave results close to a normal room thermometer. The readings changed when conditions changed, which shows the sensors were working. In the future, we could power up the arduino board with reliable DC power source and hook up 16x2 LCD Module to display readings directly on the screen to make it practically useful.
Moreover, it was observed that when sensors were subjected to an external light source, this lead to steady increase in the temperature reading and reduction in humidity readings which are as expected.

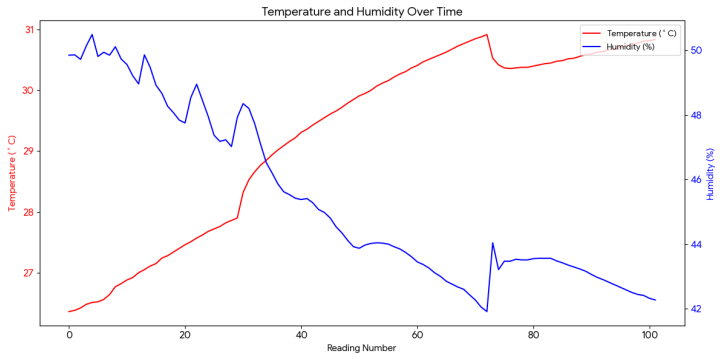

Figure 5: Temperature and Humidity change when subjected to external light source.

The distorted spike (near about reading no. 72) in temperature and humidity reading is observed when light source is slightly disturbed.

6 Conclusion

Through this hands-on session, we validated the ability of the Arduino Nano 33 BLE Sense Rev2 to gather physical motion and environmental data effectively. Sensor initialization, data retrieval, and interpretation were successfully demonstrated using the Arduino IDE

and serial monitoring. The setup and coding helped us understand how sensors work and how to read data from them. Next steps could include improving how the data is shown or saved, and using the board in a real project like a weather monitor or step counter.

7 References

- Website: `https://docs.arduino.cc/hardware/nano-33-ble-sense/`
- Website: `https://docs.arduino.cc/resources/datasheets/ABX00069-datasheet.pdf`
- Website: `https://deepmind.google/models/gemini/` (for plotting fig. 5 based on input values to model)