EE 3330 - Weather Station Project

Design Documenent

Laith Al Sairafi and Luke Post

Department of Electrical and Computer Engineering

Iowa State University

December 17th 2025

Contents

List of Figures

List of Tables

1 Introduction

1.1 Project Overview

This is a design of a weather station device that provides multiple essential metrics used in real-world weather scenarios. This device can be used in various settings, including homes, farms, and offices, allowing users to gain insight into the weather without leaving the comfort of their own place. Additionally, the device provides a simple web dashboard that can be accessed locally to display the sensors' readings on the user's computer or phone, as well as a small OLED display on the device for easy local monitoring.

1.2 Main Components List

Table 1 lists the main components used in this design (e.g., sensors, microcontroller, modules, integrated circuits (ICs)). There are many other passive components, such as inductors, capacitors, transistors, and resistors, with various values as predefined in the respective datasheets, which were not listed here for brevity.

Description	**Manufacturer Part Number**	**Quantity**
Temperature, Humidity, Pressure Sensor	BME280 [1]	1
USB-to-UART Bridge IC	CP2102N-A02-GQFN28 [2]	1
Buck Boost 1.5A Regulator IC	MC33063AD [3]	1
GPS Module	TESEO-LIV3R [4]	1
1.3″ OLED Display Breakout Board	938 [5]	1
UV light Sensor	SI1146-M01-GMR [6]	1
ESP32-C3	ESP32-C3-WROOM-02-H4 [7]	1

Table 1: List of the main components used.

1.3 Ports & Switches

This design features two main ports and a switch, which include a micro USB port, a barrel jack, and a power Switch. The micro USB port is used **ONLY** to program the microcontroller (ESP32) on the board, meaning that it does not provide power to the device. The barrel jack is used to provide power to the system. Please refer to the Power Requirements section for more information on the required power supply to prevent damage to internal components. Lastly, the power switch controls the main power to the board. When switched **ON**, the system receives power from the barrel jack; when switched **OFF**, all power to the device is cut.

1.4 Power Supply Requirements

To provide the necessary power to this device through the barrel jack, please use a power supply that can supply a minimum of 1 amp of current at 12 volts to ensure system stability.

2 Design Block Diagram

2.1 Overview

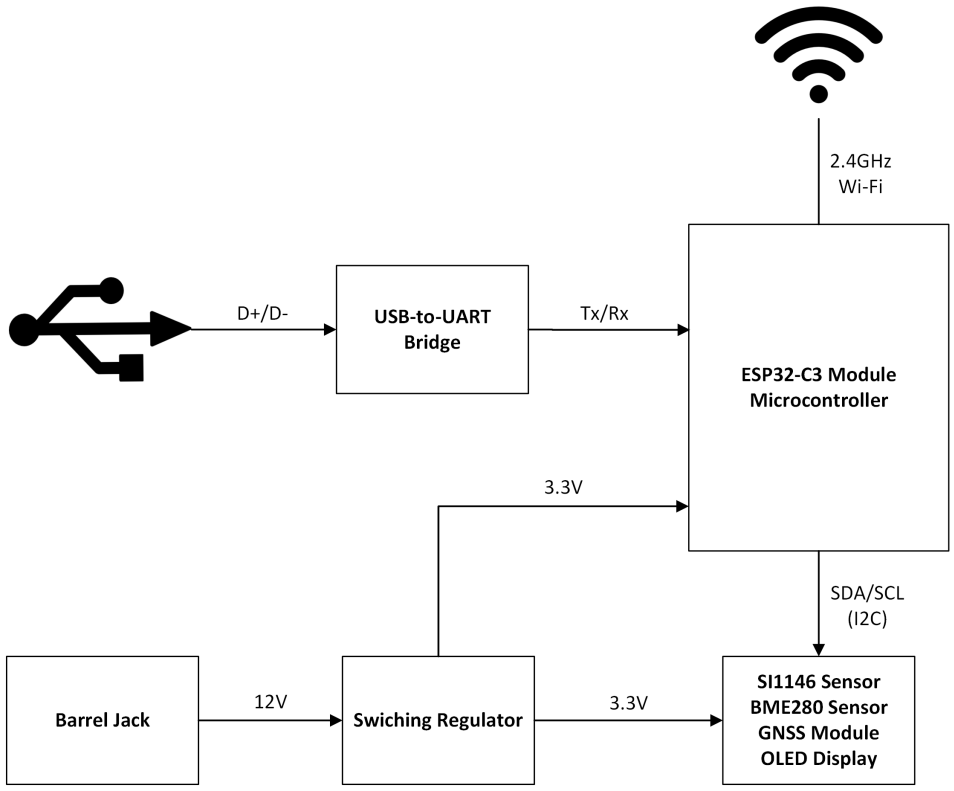


Figure 1: High-level design block diagram.

2.2 Description

The high-level system block diagram, as seen in Fig. 1, contains six primary blocks, which are the following: a **USB** input, for uploading the firmware, a **Barrel Jack** that supplies 12V, a **Switching Regulator** that steps down the voltage to 3.3V to power all components used, a **USB-to-UART Bridge** that receives USB data and sends it through UART to the **Microcontroller**. Moreover, all sensors and modules, which are the **Si1146 sensor, BME280 sensor, OLED display, and GNSS module**, communicate with the microcontroller using the I^2C protocol to provide measurement data and are powered with 3.3V.

3 Schematic

3.1 Power Supply Port

The device is powered through a 12V barrel jack input, **PJ-002A**. It is designed to be switched on and off using a slide switch, **SLW-1276864-4A-D**, and features test points at both the output and ground, as shown in Fig. 2. An LED is used at the output to ensure that power is being supplied to the printed circuit board (PCB).

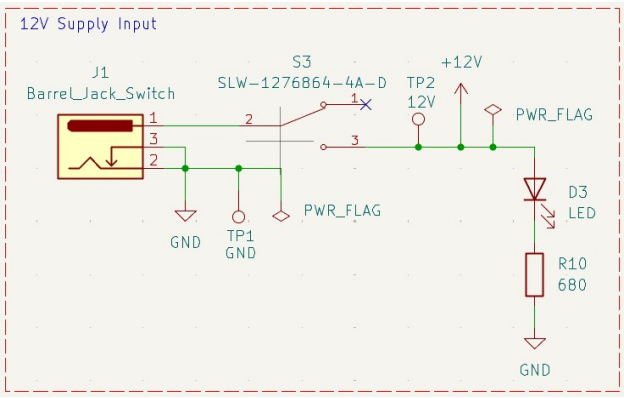


Figure 2: Power supply port schematic.

3.2 Micro USB Port

The device is programmed through a micro USB input, **47346-0001**, which connects the data lines, D+/D-, to the USB-to-UART Bridge chip, as shown in Fig. 3. TVS diodes are used to protect the device from accidental electrostatic discharges.

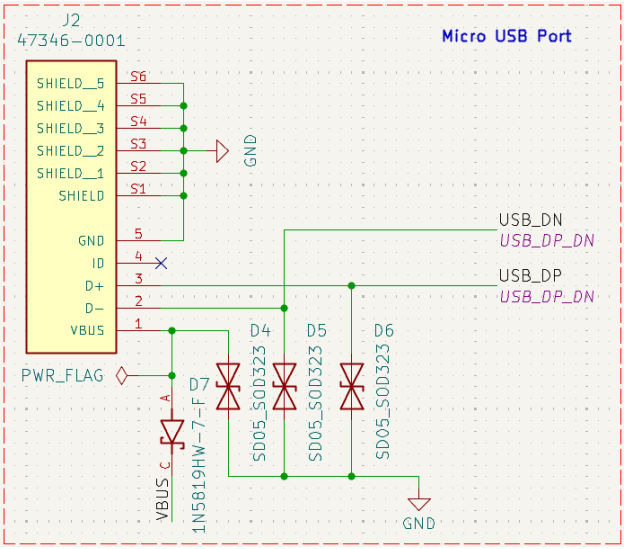


Figure 3: Micro USB port schematic.

3.3 Buck Converter

All the components chosen for this device can be powered by a 3.3V input, so a linear regulator, **MC33063AD**, is used to reduce the 12V input to 3.3V, as shown in Fig. 4. A buck converter was chosen due to its higher efficiency compared to a linear regulator, and this one was specifically selected because it is manufactured by a reputable company and can provide a varying range of outputs from a 12V input. A resistor network is used at the output to attain the desired voltage, where the values were calculated using Eq. 1. The value of the feedback resistor was also calculated using Eq. 2, where the maximum current from was found to be about 1A from summing the maximum currents of all the components.

$$V_{out} = 1.25 * (1 + \frac{R_1 + R_2}{R_3}) \tag{1}$$

$$R_{SC} = \frac{0.3}{I_{pk(switch)}} \tag{2}$$

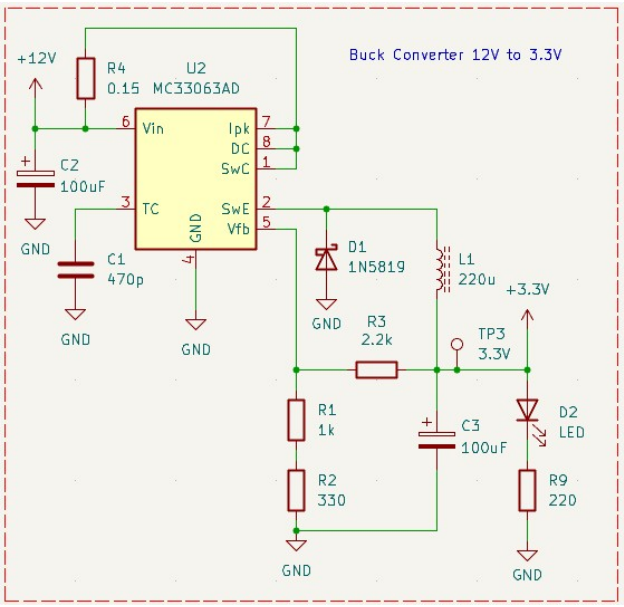
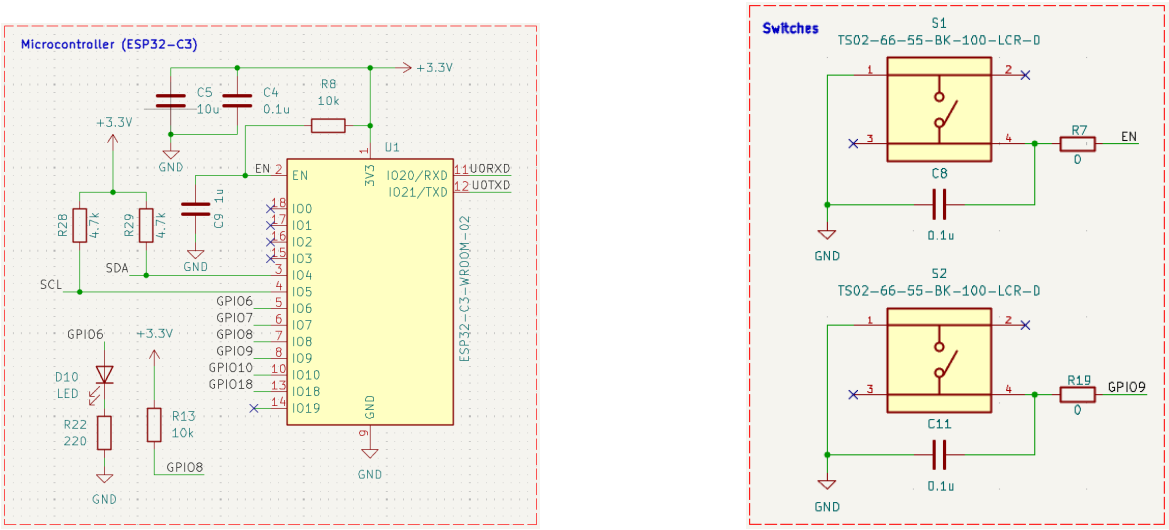


Figure 4: Buck converter schematic.

3.4 Microcontroller

The microcontroller used in this design is the ESP32, specifically the **ESP32-C3-WROOM-02-H4**, as seen in Fig. 5a. It has the ability to communicate through the I²C protocol, which is used in this design, and built-in Wi-Fi capabilities. A pull-up resistor is used on all the I²C data and clock lines, as well as on GPIO pin 8. A microcontroller reset and boot can be performed manually using the switches, **TS02-66-70-BK-160-LCR-D**, as shown in Fig. 5b. Additionally, an LED connected to GPIO pin 6 is added to perform a test, ensuring

good communication between the USB-to-UART bridge and the microcontroller. All other design decisions, such as component types and values, were predefined in the datasheet.

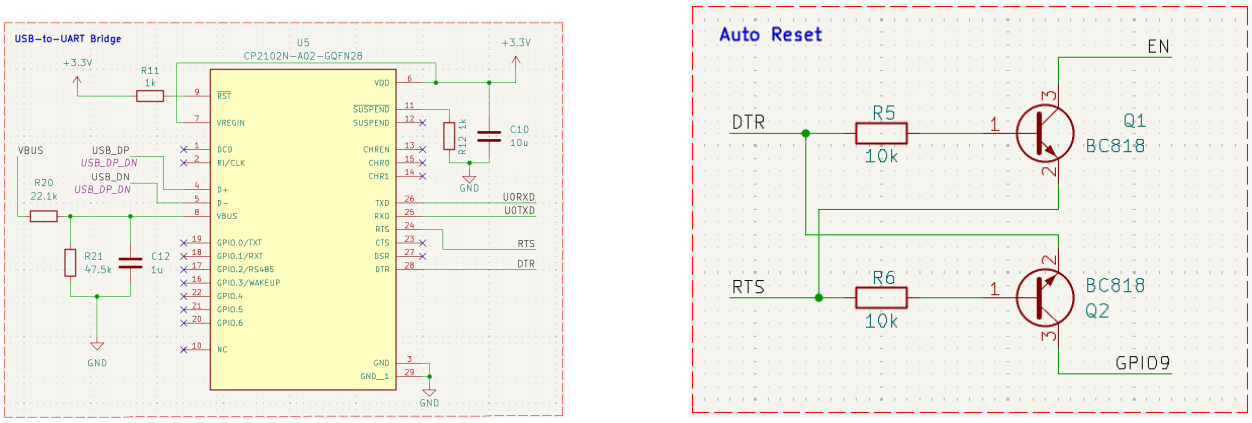


(a) (b)

Figure 5: Schematics of ESP32 Microcontroller and Switch. a) Microcontroller (ESP32) Schematic. b) Boot and reset switches schematic

3.5 USB-to-UART Bridge (Bootloader)

To upload the firmware to the microcontroller, a USB-to-UART bridge chip is used, specifically **CP2102N-A02-GQFN28**, as shown in Fig. 6a. This chip receives data input from the micro USB port and transmits it through the UART to the microcontroller. Additionally, an automatic reset circuitry was also used for this component, as indicated in the schematic, which connects to pin 2 of the ESP32 to put the microcontroller in boot mode and reset it after the new software is successfully uploaded, as shown in Fig. 6b.



(a) (b)

Figure 6: Bootloader and Automatic Reset Circuitry. a) Bootloader Schematic. b) Automatic Reset Schematic.

3.6 Temperature, Humidity, and Barometric Pressure Sensor

This design utilizes a single component for measuring temperature, humidity, and barometric pressure, namely the **BME280**, as shown in Fig. 7. It was chosen due to its ability to measure all of the desired quantities. It can be programmed through I^2C and connects directly to the ESP32 through the SDA and SCL lines. All components and values used were specified in the device's datasheet.

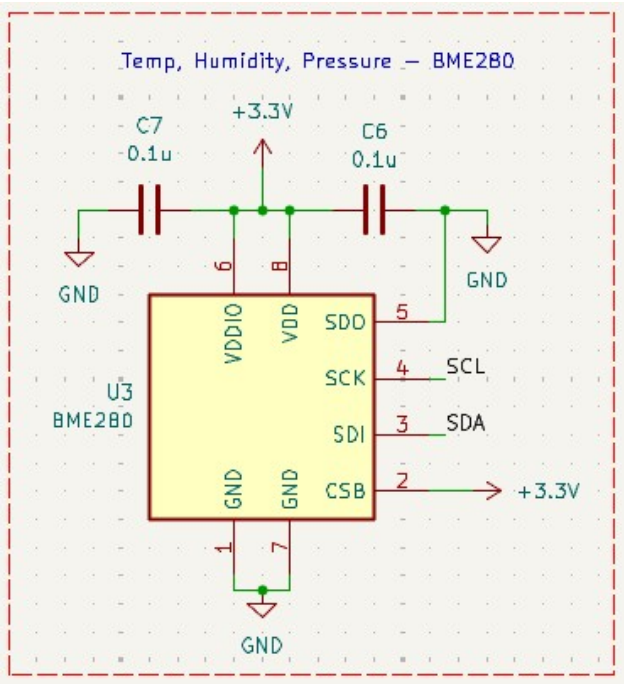


Figure 7: Temperature, Humidity, and Barometric Pressure sensor schematic.

3.7 UV Light Sensor

The UV sensor chosen for this project was the **SI1146-M01-GMR**, seen in Fig. 8 for its integrated UV index sensor and low power consumption. It is important to note that in order to get the most accurate UV index measurements, the device should be calibrated according to the datasheet. All resistor and capacitor values in the schematic were also chosen according to the datasheet. Tolerance levels of 5% were required for these components, so the parts were chosen to meet that criterion.

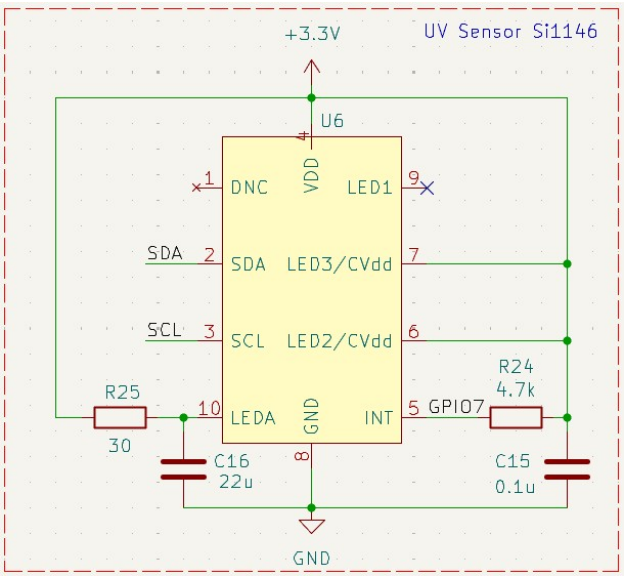


Figure 8: UV Sensor Schematic.

3.8 GNSS Module & Antenna

To get the GPS location of the device, four main components are needed, which are a Global Navigation Satellite System (GNSS) Module, a low-noise amplifier (LNA), a SAW filter, and an antenna. The **TESEO-LIV3R**, seen in Fig. 9, was chosen for this application due to its small size, and it is an easy-to-use standalone module. The reset pin on the module was connected to the same reset circuitry as the ESP32, and the values for the RLC components were specified in the datasheet.

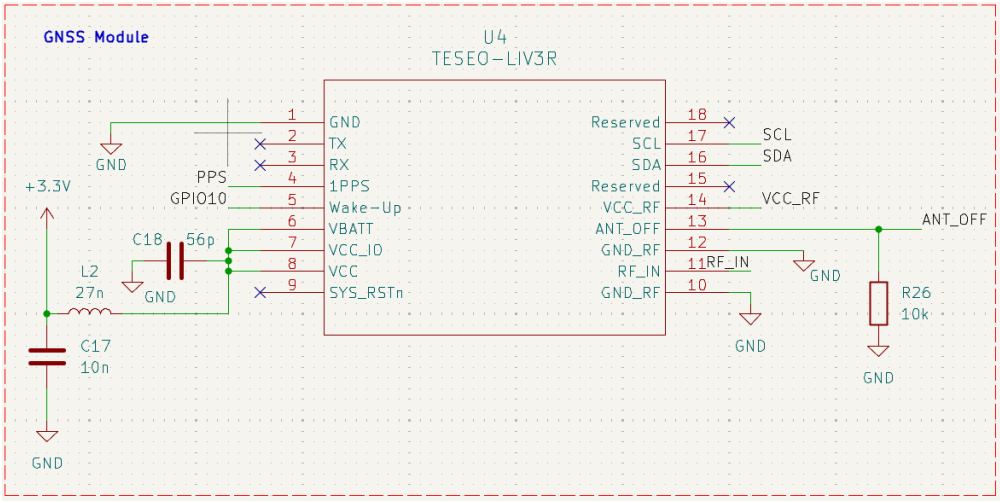


Figure 9: GNSS Module Schematic.

To connect the RF antenna to the GNSS module, the input signal must be amplified, as the received signal is very small. The LNA chosen was the **BGA725L6E6327FTSA1**, seen in Fig. 10. It is designed specifically for a GNSS module and provides 20dB of gain. The output of the LNA is then fed into a SAW filter, **B39162B4327P810**, which is designed for low-loss RF GPS applications, as seen in Fig. 10. The antenna chosen for this project was the **YEGT001AA** for its low cost and frequency range of 1.555GHz to 1.605GHz, which includes the GPS frequency signal. To connect the antenna to the PCB, a side-mount SMA connector was chosen.

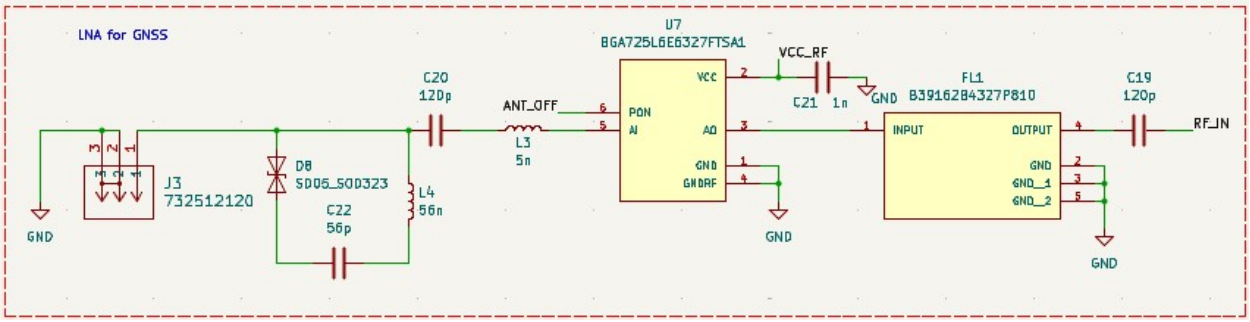


Figure 10: LNA and SAW Schematic

An LED is also used to indicate how often the GNSS module is receiving data, as shown in Fig. 11.

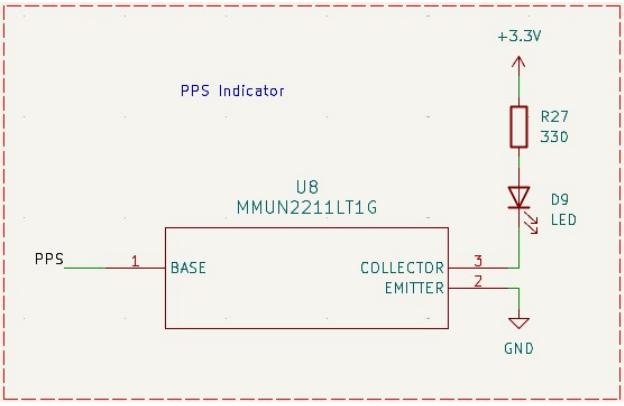


Figure 11: LED Showing when Data is Received from the GNSS Module.

3.9 OLED Display

To display the outputs of all the sensors, an OLED display will be used. When communicating over I^2C, the display only needs four wires: power, ground, data, and clock. Therefore, four pins were added to the schematic for that connection as seen in Fig. 12

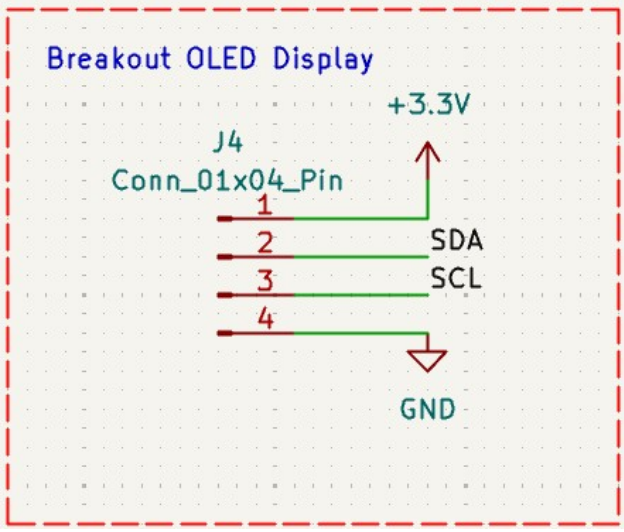


Figure 12: Schematic to connect PCB to external OLED display.

3.10 Mounting Holes

Finally, mounting holes with 3.2mm diameter were added to the schematic, which also serve as grounds, as shown in Fig. 13.

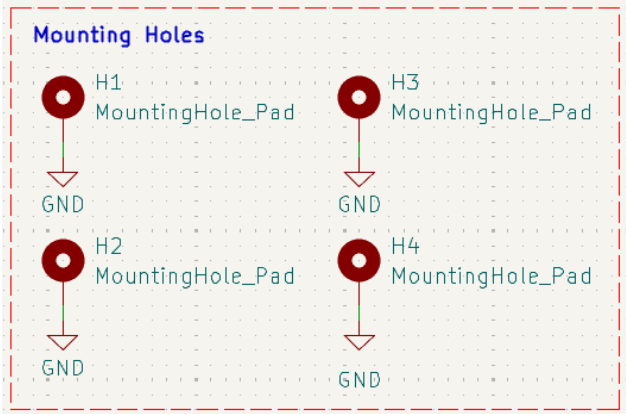


Figure 13: Mounting holes schematic.

3.11 Overall Schematic

Fig. 14 shows the full PCB schematic that contains all parts in the previous sections 3.1 – 3.10.

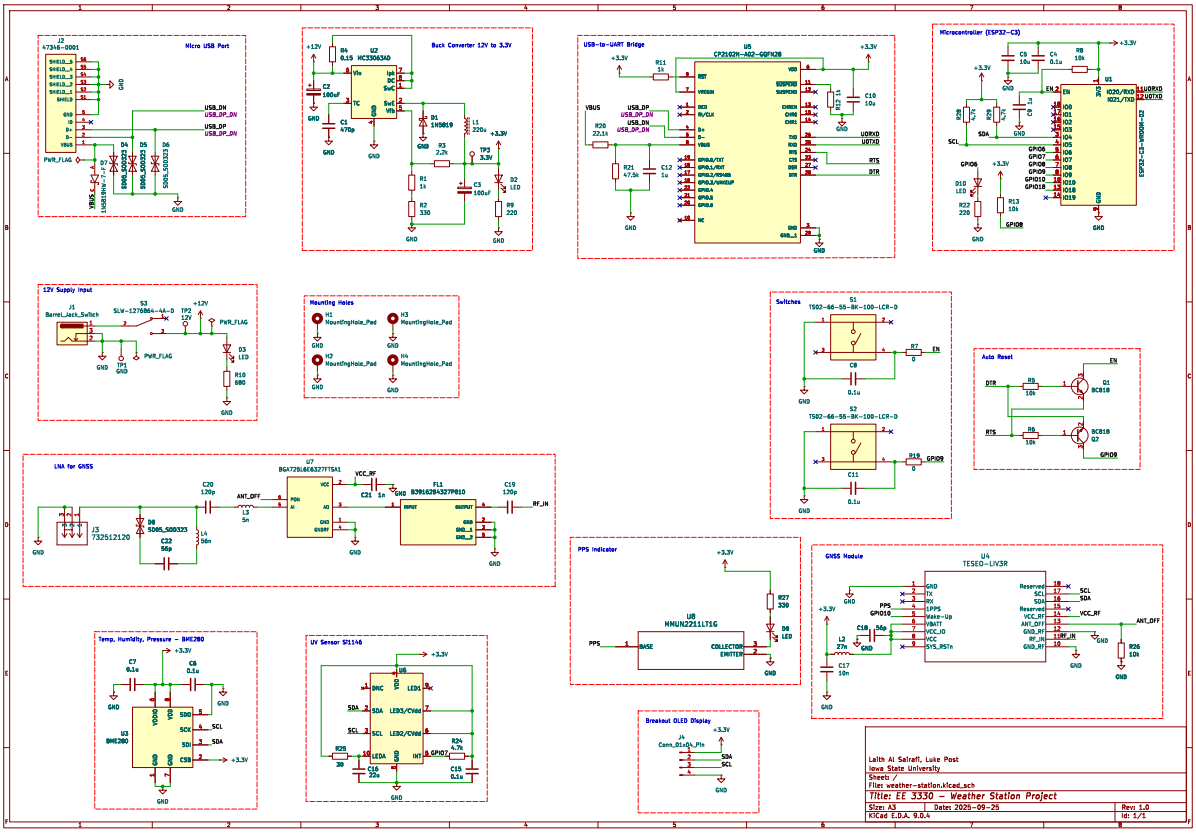


Figure 14: Overall schematic of the Weather Station Project.

4 PCB Layout

4.1 PCB Overview

Fig. 15 shows the initial PCB layout. All components that belong to the same chip (e.g., power regulator, sensor, microcontroller, or port) were placed close to each other to simplify the trace drawing process and minimize the length of the traces, thereby reducing inductance. Additionally, all ports are positioned along one side of the PCB to enhance usability.

During the initial layout process, particular attention was given to signal integrity and power distribution. Components were placed close to each other to minimize the length of the traces. Decoupling capacitors were placed as close as possible to the power pins of ICs, as recommended in the datasheets, to ensure optimal performance. Finally, the layout was designed to facilitate easy debugging and future testing by maintaining adequate spacing and clear labeling around key test points.

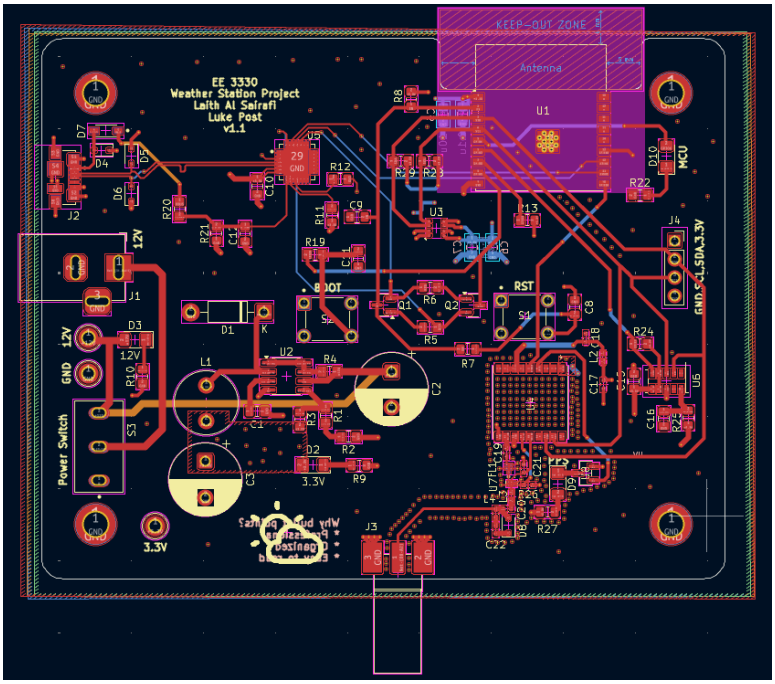


Figure 15: PCB layout and routing

This PCB is a four-layer board with the top layer being signal/ground, the second being ground, the third being the 3.3V plane, and the bottom layer being another signal/ground plane. Originally, the plan was to have all of the components on one layer and just use the bottom signal plane for routing purposes, but after finding issues in the initial layout, some new considerations needed to be made. Initially, the ESP32 was not positioned properly and needed to be closer to the edge of the board. The SCL and SDA lines were also routed too close together and would be plagued by crosstalk. It was also noted that the routing around the ESP32 was considerably messy, so some of the layout also needed to be changed.

It is also necessary for this design to have via-fencing around the RF signal from the GPS antenna to ensure no noise can interfere with the signal received. Additionally, vias were placed all around the GNSS module, as recommended in the user manual. A 3D view of the initial PCB layout and routing is shown in Fig. 16.

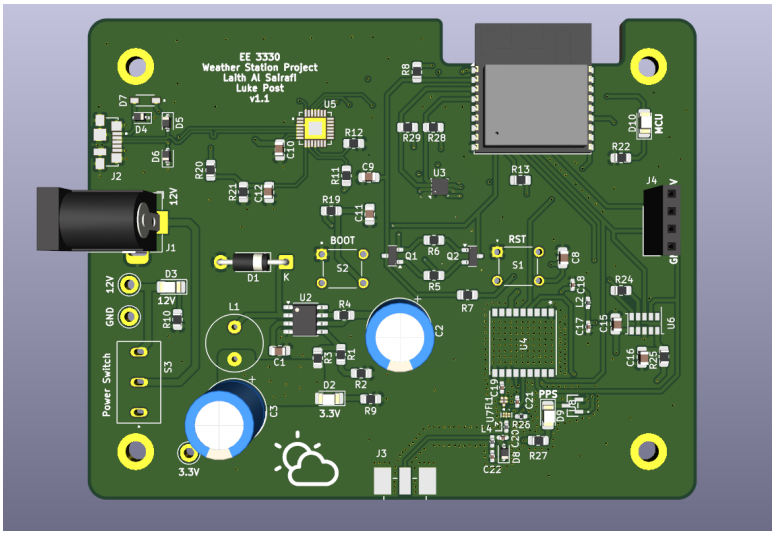
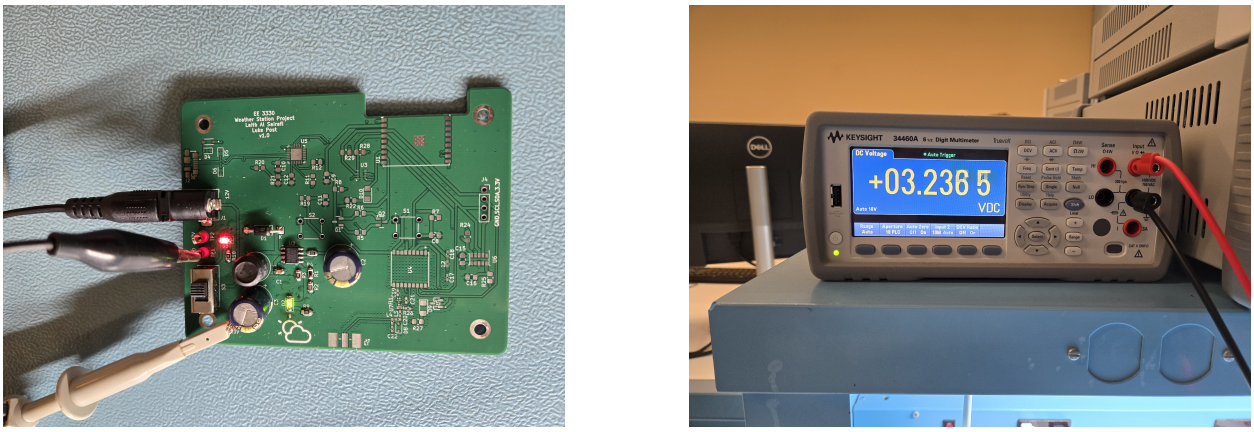


Figure 16: Front 3D view of the PCB layout and routing.

5 Assembly & Testing

5.1 PCB Assembly

The assembly of the PCB began with the buck converter to ensure the correct voltage was being output before any of the sensitive ICs were added. This was an important step in the testing because a faulty or incorrectly designed buck regulator could cause significant damage to the ICs. All of the components were hand-soldered onto the PCB, and test points were used to verify the output voltage of the buck converter. The measured output voltage was a stable 3.24V, as seen in Fig. 17b, which is close enough to our goal voltage of 3.3V.



(a) (b)

Figure 17: a) Soldered buck converter circuitry. b) Measured output voltage from the buck converter.

Once the voltage was verified, the ICs were soldered, along with the low noise amplifier and the surface acoustic wave (SAW) filter in the RF circuitry for the GPS module. In order to solder these components, small amounts of solder were placed on all of the pads, then a heating bed and a heat gun were used to melt the solder and attach the components. Testing the connections for these is slightly more difficult than testing passive components because all of the pads are underneath the IC, so they cannot be probed with a multimeter. Instead, the pads of components connected to each pin of the ICs were probed using the diode setting of a multimeter because most ICs have a diode on the input of each pin. During this testing, several components flagged issues, especially the BME280, which had to be removed and replaced several times. Once all these components passed testing, the remaining passive components, switches, and connectors were hand-soldered into place, and overall PCB testing commenced.

5.2 PCB Testing

To start testing, the Arduino IDE was used to program the ESP32 microcontroller. An issue was immediately discovered because communication could not be established with the device. After investigating the schematic, it was discovered that the Rx and Tx lines coming from the bootloader entered the Rx and Tx pins of the microcontroller, respectively, which is incorrect. The Rx pin of the bootloader must be wired to the Tx pin of the microcontroller, and vice versa. To resolve this issue, the Rx and Tx lines on the PCB were cut, and jumper wires were used to reconnect the correct lines as seen in Fig. 18. Kapton tape was then placed over the exposed wires to prevent them from shorting. This fix was successful, enabling communication with the ESP32.

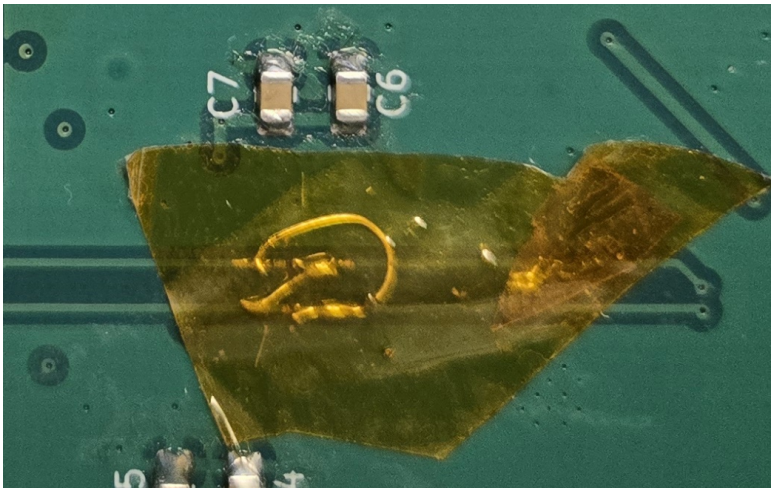


Figure 18: View of the cut Rx and Tx lines and the jumper wires

The next issue involved programming the ICs. Although communication with the ESP32 was successful, the ICs were not responding to any of the I^2C messages being sent. The initial assumption was that the microcontroller or ICs were not soldered correctly, but after performing the diode test on all pins again, it was decided that the issue was arising from elsewhere. Next, the documentation was reviewed again to ensure that nothing was missed, either in the circuitry or in the code being used to program the ICs. During this process, it was discovered that pins 3 and 4 on the ESP32, which correspond to the SDA and SCL lines, were also labeled as IO4 and IO5. This caused confusion and resulted in the pins being labeled as 4 and 5 in the code instead of 3 and 4. Another key detail discovered was that a pull-up resistor was needed on GPIO pin 8 to put the microcontroller into programming mode, which was missing. To solve this, a 10 kΩ resistor was added from pin 8 to the 3.3V pin on the microcontroller, as seen in Fig. 19. After making these changes, the ESP32 could communicate with all of the ICs. The schematic and layout were updated to correct for the previous errors (Rx and Tx lines and the missing pull-up resistor) as shown in Fig. 20 and Fig. 21. Once it was clear that the PCB and components were working correctly, the firmware was developed to accurately program and read the data from the ICs.

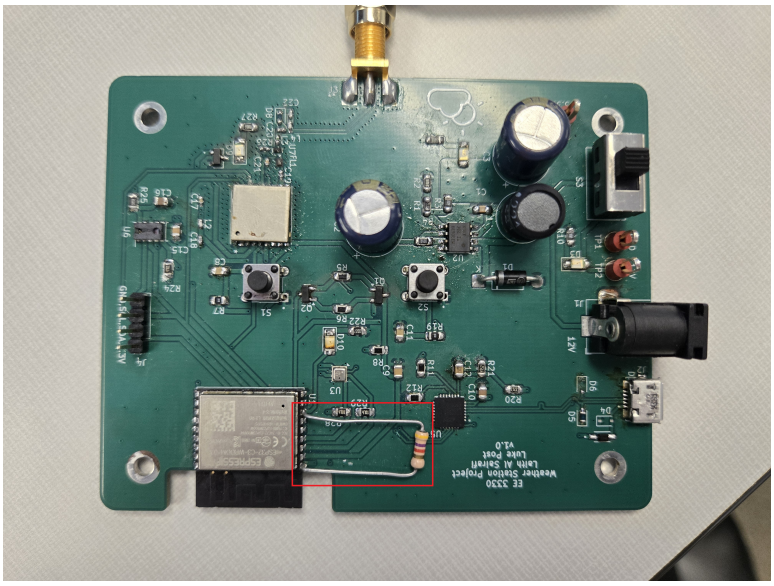


Figure 19: View of the pull-up resistor added to the GPIO pin 8 of the microcontroller to the 3.3V pin.

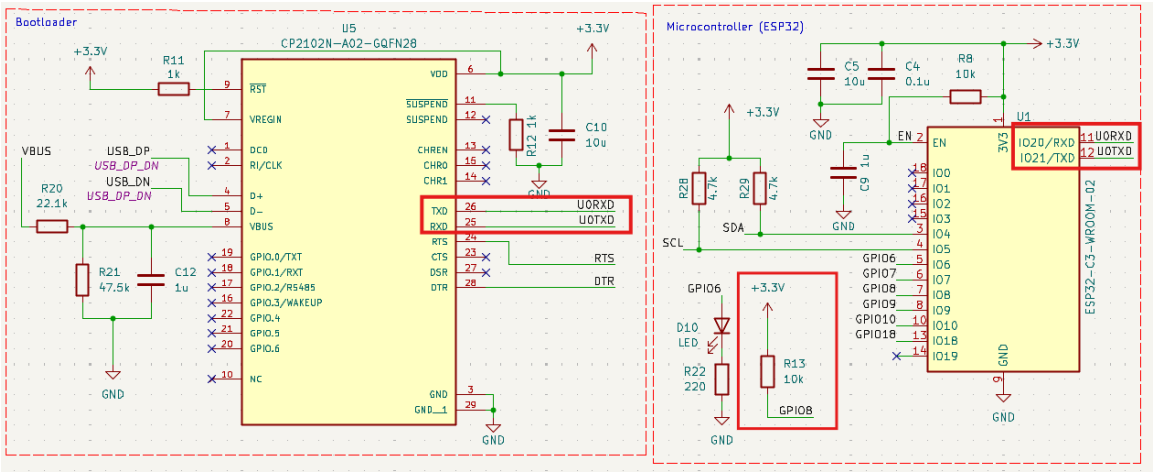


Figure 20: Schematic with the changes marked by the red squares.

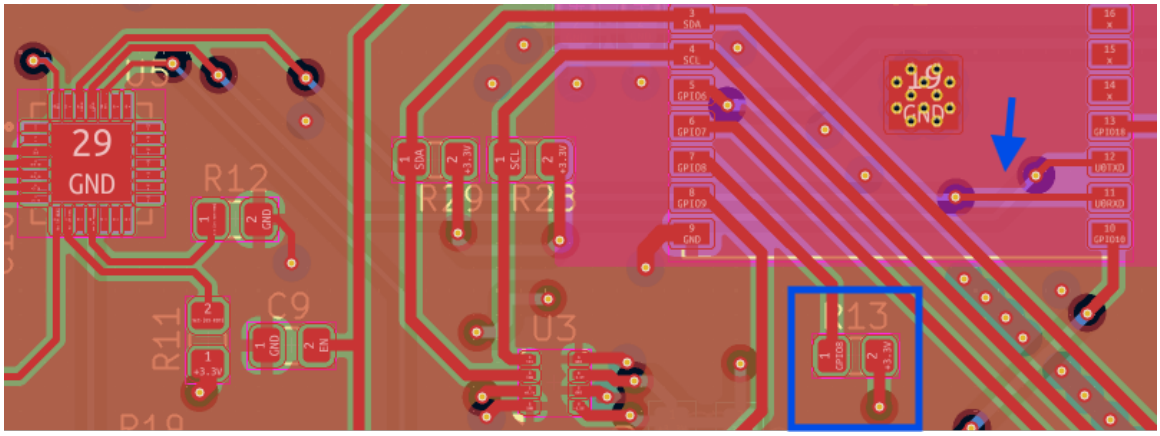


Figure 21: Layout with the pull-up resistor marked with a blue square and the Rx, Tx line switch shown at the blue arrow.

5.3 Firmware Stability Testing

The firmware was developed using Arduino IDE and the available open-source libraries for each sensor. Testing started with each sensor's library and displaying the read data to ensure that the library works as intended. Then, after all sensor libraries were tested, all libraries and code were combined into a single file, which contained functions to read the sensors, set up Wi-Fi connections, and configure the web server. Later, the web dashboard was developed using HTML, CSS, and JavaScript to show the data on a locally accessible web dashboard. The web dashboard files are stored on the microcontroller and served through the web server. Fig. 22 shows the final interface design of the web dashboard. Finally, the device was left running continuously for around 48 hours to ensure the stability of the device over a long period of time.

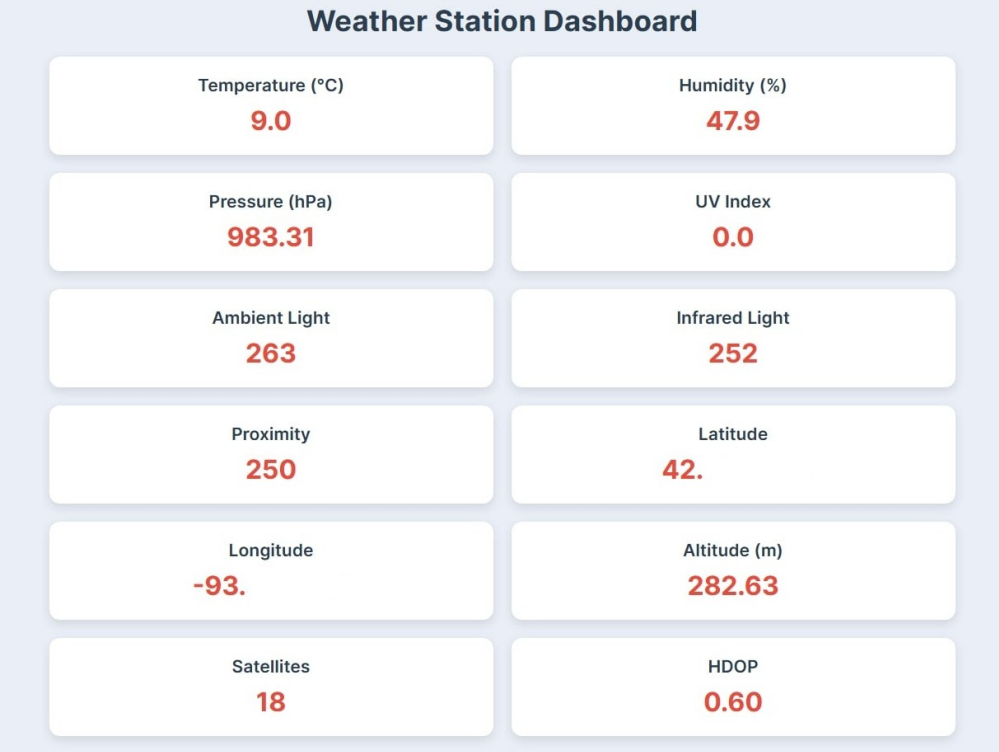


Figure 22: Screenshot of the web dashboard.

5.4 Accuracy Testing

To confirm the accuracy of the displayed measured data, indoor and outdoor tests were conducted, and the results are the following:

Indoor Tests

The device was placed in an indoor environment and was continuously running to collect data. After a random period of time, the readings were compared the the readings of another commercially available device. The readings were consistent with those of the other device, and the discrepancies were negligible, ranging around ± 1 for temperature and humidity. Since the commercially available device didn't read barometric pressure, the Ames Municipal Airport's data were used, and it was noted that there was approximately a ± 30 hPa difference, which can be attributed to various factors, such as altitude. UV Light index was consistently around zero, as no sunlight was hitting the sensor. Lastly, the GPS location was pinpoint accurate, as the coordinates provided accurately indicated the exact location (± 1 meter) where the tests were conducted. It is worth noting that the device was able to obtain the GPS location data only when it was near a window.

Outdoor Tests

The device was placed in an outdoor environment and was continuously running to collect data. After a random period of time, the readings were compared the the readings of another commercially available device. The readings were consistent with those of the other

device, and the discrepancies were slightly higher than in the indoor case, with a range of around ±2 for temperature and humidity. Since the commercially available device didn't read barometric pressure, data from Ames Municipal Airport were used. It was noted that there was approximately a ±30 hPa difference, similar to the indoor case, which can be attributed to various factors, such as altitude. The UV Light index ranged from 1.7 to 2.0, which was consistent with the values reported by the United States Environmental Protection Agency (EPA). Lastly, the GPS location was pinpoint accurate, as before, and the coordinates shown on the dashboard accurately indicated the exact location (±1 meter) where the tests were conducted.

6 Conclusions and Lessons Learned

This project aimed to create a user-friendly, network-connected weather station that reports real-time environmental conditions such as temperature, humidity, pressure, UV index, and GPS location, and succeeded in accomplishing this goal, while adding additional measurements for the ambient and infrared light. During the project, a significant amount of time was spent analyzing and understanding the datasheet of every component to facilitate success down the road. Even with this attention to detail, small mistakes were still made in the first revision that significantly impacted the functionality of the PCB. After fixing these mistakes, the PCB worked splendidly, returning highly accurate real-time data for all of the desired metrics.

Several lessons were learned along this journey, with the major one being the importance of fully understanding the datasheets before beginning any work on the schematic. The foundational knowledge given in the datasheet is fundamental in ensuring the functionality of the designed PCB. A significant amount of knowledge was also gained in the use of KiCAD, specifically good practices in making a schematic and layout. It is important to make the schematic clear and organized to enhance its readability, and the layout should be organized by component while using large trace widths and copper pours to dissipate heat. New hand-soldering techniques were also learned, such as applying solder to the pads of the ICs instead of applying it to the PCB itself for QFN packages, then using the heat gun to melt solder and connect the ICs.

Lastly, during advanced testing of the PCB, it was recommended that the PCB be placed in a heat chamber to test its accuracy. Unfortunately, the heat chamber had metal grates that shorted several components of the PCB and damaged at least one of the ICs. Currently, if a 3.3V input is applied after the buck regulator, the rest of the circuit will draw the maximum amperage and not actually reach 3.3V, indicating that something has been damaged. This also taught a significant lesson in the importance of performing a risk analysis of the tests before performing them to ensure that they will not damage the PCB.

7 Future Work

Since the initially designed PCB required minor revisions, a new board was not soldered. However, with the previously functional board now fried and the Rx/Tx and pull-up resistor

issues resolved in the schematic and layout, a new board should be assembled and tested in the future. Another design change that could be made is to move the temperature sensor (BME280) away from the microcontroller and other ICs for improved temperature isolation. It is hypothesized that the slight discrepancies in the temperature readings are due to heat generated by other components on the PCB. Lastly, due to the damage that occurred to the PCB during testing, as explained earlier, an OLED display wasn't implemented. Therefore, this is one additional item that needs to be included in the next revision.

References

[1] Bosch, "Bme280 combined humidity and pressure sensor datasheet," Online, 2024, rev. 1.24, originally published Nov. 2014, revised Feb. 2024. [Online]. Available: https://www.bosch-sensortec.com/media/boschsensortec/downloads/datasheets/bst-bme280-ds002.pdf

[2] SILICON LABS, "Usbxpress family cp2102n data sheet," Online, 2016, rev. 1.5. [Online]. Available: https://www.silabs.com/documents/public/data-sheets/cp2102n-datasheet.pdf

[3] Texas Instruments, "Mc3x063a 1.5-a peak boost/buck/inverting switching regulators datasheet," Online, 2015, rev. N, SLLS636N, originally published Dec. 2004, revised Jan. 2015. [Online]. Available: https://www.ti.com/lit/ds/symlink/mc33063a.pdf

[4] STMicroelectronics, "Tiny gnss module," Online, 2023, dS12152 Rev 8. [Online]. Available: https://www.st.com/resource/en/datasheet/teseo-liv3f.pdf

[5] Adafruit, "Monochrome 1.3" 128x64 oled graphic display - stemma qt / qwiic," Online, 2019. [Online]. Available: https://mm.digikey.com/Volume0/opasdata/d220001/medias/docus/197/938Web.pdf

[6] SILICON LABS, "Proximity/uv/ambient light sensor module with i2c interface," Online, 2014, rev. 1.0 11/14. [Online]. Available: https://www.silabs.com/documents/public/data-sheets/Si1145-46-47-M01Rev1.0.pdf

[7] Espressif, "Esp32-c3-wroom-02 datasheet," Online, 2025, rev. 1.6. [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32-c3-wroom-02_datasheet_en.pdf