Creating STEM Contents: Wind Power

Christian Kirk, Samuel Tando, Lyndon Li

CONCEPT OF OPERATIONS

REVISION – Final
2 December 2023

Concept of Operations
for
Creating STEM Contents: Wind Power

Team 12

Approved by:

____
Project Leader   Date

____
Prof. Kalafatis   Date

____
T/A   Date

Change Record

Rev.	Date	Originator	Approvals	Description
1	2/9/2023	Christian Kirk		Draft Release
2	2/17/2023	Chrsitian Kirk		Revised Subsystems
3	4/29/2023	Christian Kirk		403 Final Report
4	12/2/2023	Christian Kirk		404 Final Report

Table of Contents

List of Tables

No table of figures entries found.

List of Figures

1. Executive Summary

As the world combats climate change by shifting from non-renewable energy to renewable energy, it is important for the upcoming generations to learn about sustainable resources such as wind power. This project aims to give an educational demonstration on how power can be generated from wind in a way that kids can easily understand. The project will use two types of wind turbines, vertical and horizontal, to exhibit the different forms of wind power, and show different uses and advantages of both. The project will show how each turbine behaves differently as they generate power and supply a rechargeable nine volt battery load. The project will be a great tool in schools to teach children about energy and renewable resources.

2. Introduction

This project is meant to be an educational demo of the fundamentals of power generation using wind turbines. This will be done by creating a wind power system that utilizes both a horizontal and vertical turbine. As the world moves towards renewable zero-emission sources of power, it is important to show how wind power is generated and why it is effective.

○ *2.1 Background*

In this project, the wind power systems will be significantly downsized compared to typical wind turbines. This will enhance the ability to educate children and others because the components will be easily visible, tangible, and portable enough to get the system into more classrooms. The project will also be accompanied by visual aids that show the anatomy of the systems, and compare how they behave. This will illustrate how each system works in simpler terms and will hopefully spark the interest of young students to further their education in STEM.

○ *2.2 Overview*

Both wind power systems will consist of a tower, generator, blades, and display. The difference will be the orientation of the two turbines: horizontal and vertical. Both orientations can be seen in the figure below. Each one will be used to compare and contrast the systems.

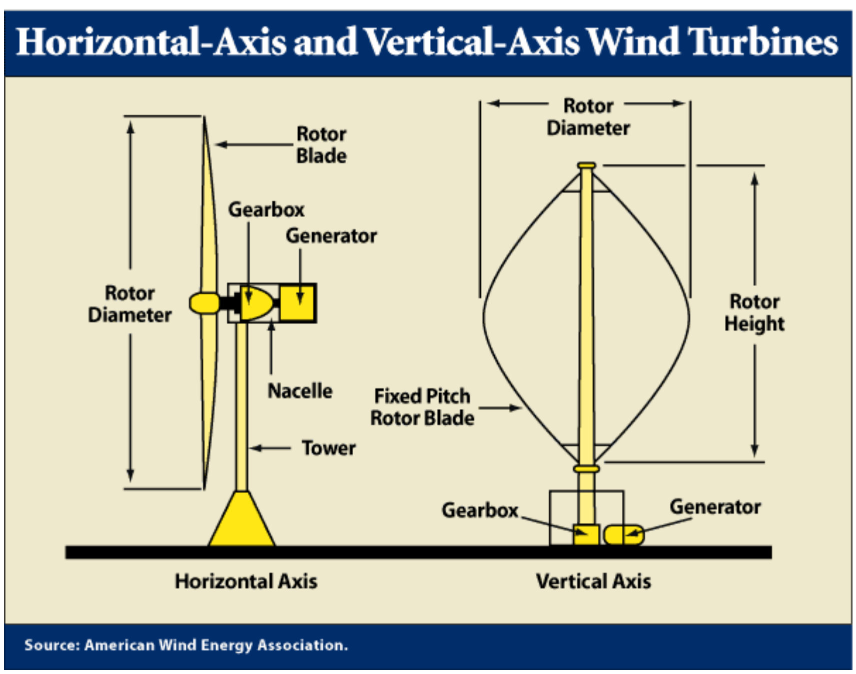


Figure 1: Horizontal and Vertical Turbines

Horizontal-Axis Wind Turbines or “HAWTS” are generally more efficient than Vertical-Axis Wind Turbines (VAWTs) because they are oriented parallel to the wind direction. However, VAWTs are more suitable for urban settings because they are generally smaller and quieter during operation. We intend to show the advantages and disadvantages of both styles by utilizing each in our wind power system. To accomplish this, we will divide this project into subsystems so that each member of the group can focus on accomplishing their task that will be tailored to their skill set. These subsystems will be the microcontroller, the DC-DC converter and the buck-boost converter, and an Android monitoring app.

2.3 Referenced Documents and Standards

- American Wind Energy Association (AWEA) Standards
- International Electrotechnical Commission (IEC) 61400 Wind Turbine Standards
- National Electrical Code (NEC) Requirements

3. Operating Concept

○ *3.1 Scope*

The scope of this project will be to deliver a functional wind turbine, in both horizontal and vertical configurations, by the end of the semester while fitting within the $300 budget. The turbines will be small in scale to serve as a portable educational demo. They will supply power to a nine volt rechargeable battery, and it will be explained that in the industry these same wind power systems can be upscaled to supply power on a much larger scale.

○ *3.2 Operational Description and Constraints*

The wind turbine generator will be attached to a DC-DC converter and a buck-boost converter to supply power to a nine volt rechargeable battery. The sensors from the microcontroller will then be connected to the output of the generator and to the battery load to store the real-time readings that will be sent to the Android monitoring app via Bluetooth. The app will allow the user to set their desired max charge level of the battery. This data will be sent to the MCU and used to flip a charge switch that makes the buck-boost converter stop supplying power to the battery once the desired charge level is reached. The constraints are that the systems need to be portable, within the $300 budget, and fully operational by December 2023.

○ *3.3 System Description*

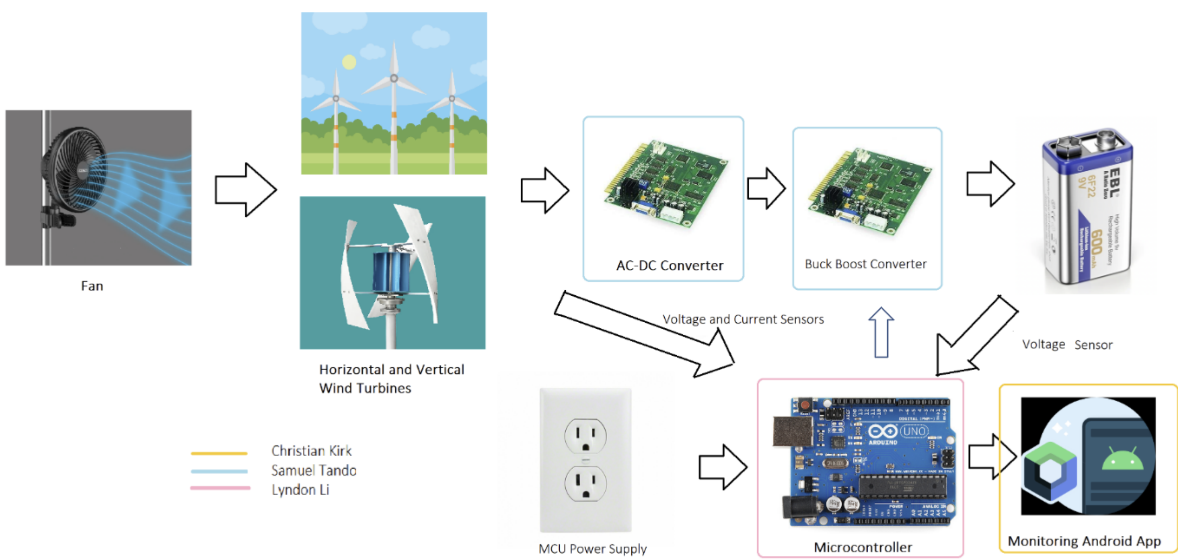


Figure 2: System Block Diagram

There will be three subsystems: the microcontroller, the DC-DC converter and the buck boost converter, and the monitoring app.

1. Microcontroller: This subsystem will be able to take input from the generator and output the wattage to a mobile app. The microcontroller will take in power from an external powersource in order to operate as well as a voltage sensor in order to take in input from the generator. The microcontroller will also generate an output signal to control the charging operation as well as take in input from the battery to output battery capacity to the mobile app. This work will be done mostly by Lyndon.

2. DC-DC Converter and Buck Boost Converter: The DC-DC converter will take in the DC Voltage output from the generator and regulate it to a consistent 5V for the buck boost converter. The buck converter will take the input from the DC-DC converter and it will step it up to a safe voltage level to charge the 9V battery. The Buck Boost Converter will also have a switch that can receive a signal from the MCU to turn on/off the power to the battery.

3. Monitoring App: This app will take input from the microcontroller sensors and then display this data on an Android phone display. The information that will be displayed on the app will be the charge level of the rechargeable battery load, the power output of the generator, and the user specified max charge level of the battery. There will also be a page comparing and contrasting horizontal axis turbines and vertical axis turbines. This subsystem will be Christian's responsibility.

○ *3.4 Modes of Operations*

The two modes of operation for the wind turbine system will be the vertical and horizontal modes of operation. In the vertical mode the vertical turbine would be used and measure the amount of power generated from the wind turbine. The generator will be connected to the microcontroller and use a sensor and display to show the power wattage. The generator will be connected to a rechargeable battery that will connect to a monitoring app which will output a log of the battery level. The horizontal mode operates the same with the same amount of wind, and the two modes could then be compared to see the advantages and disadvantages of using a horizontal turbine and a vertical turbine.

○ *3.5 Users*

The users of this system will not have to be trained in the specifics of how power generation works as the system will be fully contained in the custom housing. However, they will need to be able to operate the Android app on the Moto E device. This will require a basic knowledge of how to use a mobile phone. They will also have the ability to use a fan to interact with the wind turbine generators to generate power to the system and charge the battery. This project is mainly for demonstration purposes so there will be no real world use case in which the project is used as a power source.

○ *3.6 Support*

Support will be given in the form of a user manual demonstrating how to operate as well as the internal structure. The designers will also be available to troubleshoot if the product is not functioning correctly.

4. Scenario(s)

○ *4.1 Educational*

The main use case for this project would be for educational purposes, with a target audience of children in schools. The wind turbines will be portable and sturdy enough to display to an audience how wind turbines can be used to generate electricity.

5. Analysis

○ *5.1 Summary of Proposed Improvements*

The system will provide a heavily scaled down version of a power system with a horizontal and vertical wind turbine, giving children a better understanding and interaction of wind power and power generation.

For the firmware side of the MCU, there are still issues with sensor readings being slightly off due to the ESP32's imprecise accuracy on the onboard ADC. To compensate for this, we've added offset values to the sensors but are still working on calibrating the lower ends and the upper ends of the readings.

○ *5.2 Disadvantages and Limitations*

As it is not a full scale version of an actual wind power system, it cannot replicate the real life scenario of actual wind turbines in their environment, only a simulation of it. It is also very limited in the amount of power that it can generate, which is aimed just to be enough to power a lightbulb or charge a battery.

○ *5.3 Alternatives*

There may be alternatives regarding the components we use for each sub system, including the microcontroller, which we could perhaps use a cheaper or more suitable one rather than the ESP32. We could also eventually produce a more efficient generator or different materials for the wind turbine makeup. An alternative for the app could be developing an IOS app rather than an Android app, but an Android app was opted for because of the affordability of the Android device that we picked (Moto E) without sacrificing performance.

○ *Impact*

As the issue of climate change continues to grow, more people need to be educated about the causes of climate change and how to combat it. Schools nowadays are required to teach about the effects of climate change and how renewable energy can help soften the effects of it. This project would be very useful as a demonstration of different forms of wind energy that are being put into use today, and give children a good understanding of how they work and benefit the environment.

Creating STEM Contents: Wind Power

Christian Kirk, Samuel Tando, Lyndon Li

FUNCTIONAL SYSTEM REQUIREMENTS

REVISION – Final Report

Functional System Requirements

for

Creating STEM Contents: Wind Power

Prepared by:

____
Author Date

Approved by:

____
Project Leader Date

____
John Lusher, P.E. Date

____
T/A Date

Change Record

Rev.	Date	Originator	Approvals	Description
1	2/19/2023	Christian Kirk		Draft Release
2	4/29/2023	Christian Kirk		403 Final Report
3	12/2/2023	Christian Kirk		404 Final Report

Table of Contents

List of Tables

List of Figures

1. Introduction

1.1. Purpose and Scope

This specification defines the technical requirements for the development items and support subsystems delivered to the client for the project. Figure 1 shows a representative integration of the project in the proposed CONOPS. The verification requirements for the project are contained in a separate Verification and Validation Plan document.

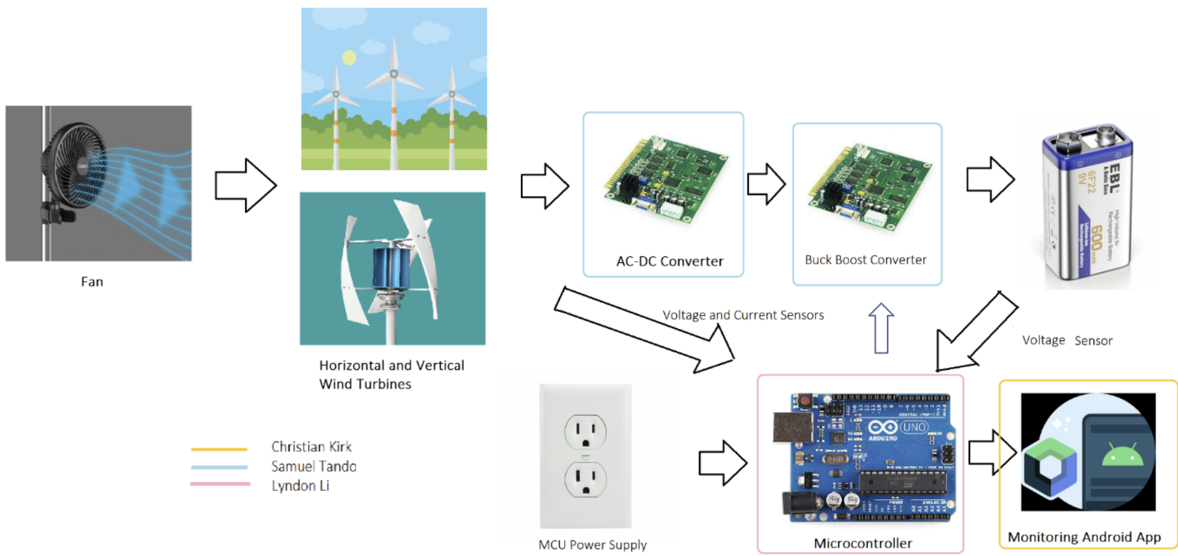


Figure 1. Project Conceptual Image

1.2. Responsibility and Change Authority

Christian Kirk will be the team leader who is responsible for ensuring that the project requirements are met. In order to change the project requirements, the approval of Christian and the sponsor, Wonhyeok Jang, will be needed.

Subsystem	Responsibility
MCU	Lyndon Li
DC-DC Rectifier and Buck Boost Converter	Samuel Tando
Monitoring App	Christian Kirk

Table 1: Subsystem Leads

2. Applicable and Reference Documents

2.1. Applicable Documents

The following documents, of the exact issue and revision shown, form a part of this specification to the extent specified herein:

Document Number	Revision/Release Date	Document Title
N/A	v5.2 2019-12-31	Bluetooth Core Specification

Table 2: Applicable Documents

2.2. Reference Documents

The following documents are reference documents utilized in the development of this specification. These documents do not form a part of this specification and are not controlled by their reference herein.

Document Number	Revision/Release Date	Document Title
	3/1/2018 Version 2.4	ESP32-WROOM-32 (ESP-WROOM-32) Datasheet

Table 3: Reference Documents

2.3. Order of Precedence

In the event of a conflict between the text of this specification and an applicable document cited herein, the text of this specification takes precedence without any exceptions.

All specifications, standards, exhibits, drawings, or other documents that are invoked as “applicable” in this specification are incorporated as cited. All documents that are referred to within an applicable report are considered to be for guidance and information only, except ICDs that have their relevant documents considered to be incorporated as cited.

3. Requirements

3.1. System Definition

The wind power system will be used as an educational demo to show kids the importance of sustainable sources of power in the modern world and to expose them to the STEM field. This power system will comprise three main subsystems: the AC to DC converter and the buck converter, the microcontroller and sensors, and the monitoring application.

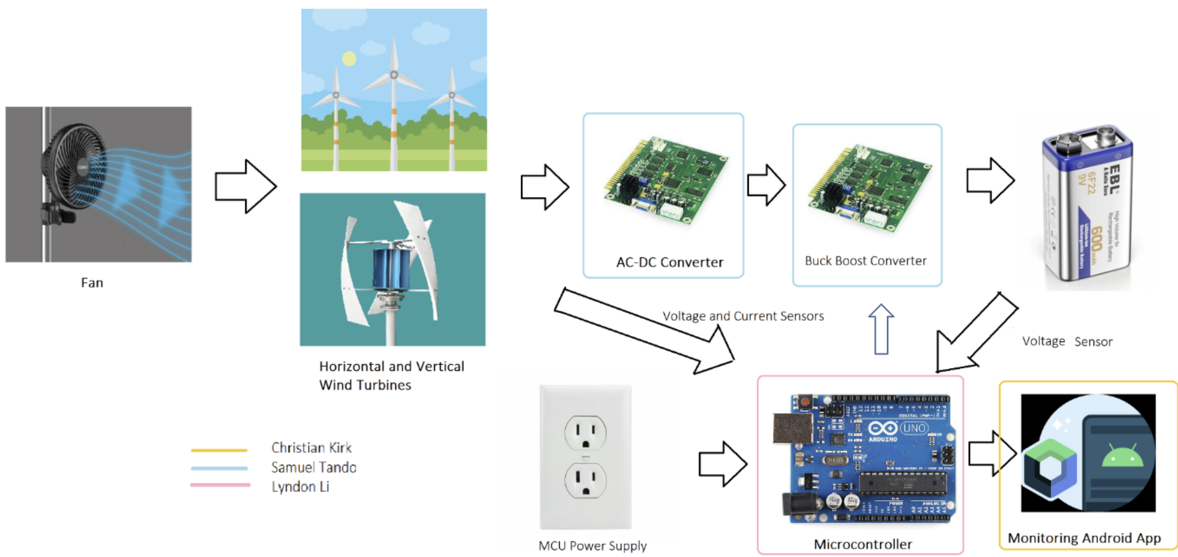


Figure 2: Block Diagram of System

Each of the three subsystems can be seen in the block diagram above. They are color coded based on who is responsible for each. The wind turbine will have a small AC motor that will generate AC that will go into the AC-DC converter, where it will be changed to DC and then the voltage will be stepped up in the buck boost converter to provide 8.4V to the rechargeable battery. The MCU will have voltage and current sensors that will be used to take measurements at the generator (AC motor) and at the load. This sensor data will then be communicated to the monitoring app via Bluetooth and displayed on an Android device. The MCU will get its power from a 5V 2A Power Jack adapter.

3.2. Characteristics

3.2.1. Functional / Performance Requirements

3.2.1.1. Android App Functional Requirements

- The Android app shall have a menu navigation UI that takes the user to the different pages
- The app shall be able to display the real time sensor data from the MCU via Bluetooth
- The app shall take user input for the desired max charge level of the battery and send this data to the MCU
- The app shall compare power output of the vertical axis turbine and horizontal axis turbine

3.2.1.2. Wind Power Output

The Wind Power System shall be able to deliver 8.4 +- 0.2 volts to a rechargeable battery.

Rationale: This is the required input voltage for the battery power delivery system as stated in the specifications

3.2.1.3. DC-DC Converter

The DC-DC Converter shall be able to convert 0-8 V AC generated from the power generator into DC voltage, capping at 5V when the AC voltage exceeds 5V to ensure over voltage protection. The rectifier on the converter will also ensure polarity protection.

3.2.1.4. Buck Boost Converter

The Buck Boost converter shall be able to step up the DC voltage from the AC-DC from 5V to 8.4 to safely charge the 9V battery load.

3.2.2. Physical Characteristics

3.2.2.1. Mass

The total weight of the wind power system shall be less than 10 pounds.

3.2.2.2. Sensor Readings

The current sensor readings will be within 10 mA of actual values and the voltage sensors will be within 20 mV of actual readings.

3.2.2.3. Volume Envelope

The volume envelope of the Wind Power System shall be less than or equal to 12 inches in height, 12 inches in width, and 12 inches in length.

Rationale: This is a functional use case requirement stated by the customer.

3.2.3. Electrical Characteristics

3.2.3.1. Inputs

a. The presence or absence of any combination of the input signals in accordance with ICD specifications applied shall not damage the Wind Power System, reduce its life expectancy, or cause any malfunction, either when the unit is powered or when it is not.

b. No sequence of command shall damage the Wind Power System, reduce its life expectancy, or cause any malfunction.

Rationale: By design, should limit the chance of damage or malfunction by user/technician error.

3.2.3.1.1 Threshold Voltage

a. The battery should stop charging at 100% capacity

Rationale: This is to avoid overcharging and is a request by the customer in order to implement a viable charge controller.

3.2.3.1.2 Input Voltage Level

The input voltage level from the Wind Power System shall be a maximum of 8.4 volts.

Rationale: Specification compatibility, MIL-STD-704F

3.2.3.1.3 External Commands

The Wind Power System shall document all external commands in the appropriate ICD.

Rationale: The ICD will capture all interface details from the low level electrical to the high-level packet format.

3.2.3.2. Outputs

3.2.3.2.1 Data Output- Mobile Application

The mobile application will use the data given from the microcontroller and give visual aid to how much power is being generated and the battery charge level in real time.

3.2.3.2.2 Data Output- Microcontroller

The microcontroller will have a voltage sensor and current sensor from the generator and a voltage sensor from the battery and will output generator wattage as well as battery level percentage to the mobile app via Bluetooth serial.

3.2.3.3. Voltage Output- PCBs

The DC-DC Converter outputs a maximum of 5V DC, and the Buck Boost will increase the 5V to 8.4 V to output to the battery load.

3.2.4. Environmental Requirements

The Wind Power system should be easily portable, weighing less than 10 pounds, and easy to set up. The ideal environment would be in a regular school classroom.

3.2.4.1. Thermal

None of the components should exceed 90 degrees, as the system should be safe for kids in case they touch or interact with it.

3.2.4.2. Electrical

The system should have its wiring safely protected in case children play or interact with the system.

3.2.5. Failure Propagation

The Wind Power System shall not allow propagation of faults at any point in each subsystem.

3.2.5.1. Reverse Polarity Protection

The DC-DC Converter shall contain polarity reversal protection, in the case that the generator is input incorrectly to the converter. If the converter received the wrong inputs, it would cause large consequences for the rest of the system, from the Buck Boost to the load battery charger.

3.2.5.1.1 Upper Voltage Protection

The DC-DC Rectifier shall contain upper voltage protection which will cease to function if too much voltage is applied. This is to avoid any voltage surges in the buck converter and safely load the voltage to the battery.

3.2.5.1.1.1 Battery Overcharge Test

The wind power system will be able to detect when battery load is near max capacity.

Rationale: This is to prevent overcurrent as well as adding functionality to our system in the form of a charge control.

3.2.5.1.1.2 Bluetooth Connectivity Test

The microcontroller will constantly check in with the app to make sure that the Bluetooth connection is stable and the data stream is operational.

Rationale: This is required in order to make sure that the app is delivering accurate and relevant information.

4. Support Requirements

The system will require an Android device with bluetooth compatibility in order to run the monitoring app as well as a wind source in order to simulate real use case scenarios. For the educational purpose of this project, we will be using a hair dryer as our wind source. A USB-C connector will also be needed to supply power to the MCU.

5. Microcontroller Subsystem

b. 5.1 Overview

The main function of the microcontroller will be to calculate the power generated from the generator. This microcontroller will take in the input from the battery load in order to output a signal to the step down converter whether to stop charging or continue charging. The microcontroller will also have a bluetooth module that will report readings to the phone application for monitoring purposes. The microcontroller will also have sensors to read in the voltage generated by the ac generator.

c. 5.2 Interface

5.2.1 ADC Interface

Signal	I/O	Description
GenVoltage	Input	This signal will provide the voltage readings from the Generator
GenCurrent	Input	This signal will provide the current readings from the Generator
BatVoltage	Input	This signal will provide the voltage readings from the battery
MaxBattery	Input	Input signal from Android app that will signal max battery level to charge to

Table 4: ADC Interface

5.2.2 Battery Interface

Signal	I/O	Description
Vin_Bat	Input	Reads the battery level from the battery bank

Table 5: Battery Interface

5.2.3 Step Down Interface

Signal	I/O	Description
Overcharge	Output	Tells the step down converter to stop charging or continue charging.

Table 6: Step Down Interface

5.2.4 App interface

Signal	I/O	Description
BatLevel	Output	Send the app the battery level of the battery bank to display.
AvgPower	Output	Send the app the average power over the specified time frame

Table 7: Application Interface

The microcontroller will have a micro-usb port for debugging as well as a DC power jack connector for power. The microcontroller subsystem will also have ports for the sensors such as the ADC, step down converter, and battery load input.

d. 5.3 Hardware

The microcontroller we will be using is the ESP-WROOM-32 as this fulfills our customer's requirements of having an app that connects via bluetooth as well as a clock speed fast enough to get accurate sample rates for our voltage readings. The ESP32 will be mounted on a PCB built with Altium, which will consist of voltage sensors, a current sensor, as well as a port to provide power to the board.

e. 5.4 Software

The microcontroller is compatible with the Arduino IDE so the custom firmware for the microcontroller will be developed using the Arduino IDE. We will take in an average of 500 readings for our sensors in order to get a more accurate reading as well as real time offset calculations with the help of a lookup table. For the Bluetooth connection, we will be using Bluetooth serial in order to transfer data.

Appendix A: Acronyms and Abbreviations

PCB	Printed Circuit Board
TBD	To Be Determined
USB	Universal Serial Bus
VDC	Volts Direct Current

Creating STEM Contents: Wind Power

Christian Kirk, Samuel Tando, Lyndon Li

INTERFACE CONTROL DOCUMENT

REVISION – Final
2 December 2023

Interface Control Document
for
Creating STEM Contents: Wind Power

Prepared by:

____
Author Date

Approved by:

____
Project Leader Date

____
John Lusher II, P.E. Date

____
T/A Date

Change Record

Rev.	Date	Originator	Approvals	Description
1	2/19/2023	Christian Kirk		Draft Release
2	4/26/2023	Lyndon Li		403 Final Report
3	12/2/2023	Christian Kirk		404 Final Report

Table of Contents

List of Tables and Figures

1. Overview

The purpose of this document is to provide details on how the Microcontroller Unit, Wind Power System, converters, and Mobile Application will interface with each other. The document will describe the inputs and outputs of each component as well as how each system will communicate with each other.

2. References and Definitions

2.1. References

Refer to section 2.2 of the FSR document.

2.2. Definitions

TBD	To Be Determined
MCU	Microcontroller Unit
V	Volts
VDC	Volts Direct Current

3. Physical Interface

3.1. Weight

3.1.1. MCU

Component	Weight
PCB	1.1 oz
ESP32 Microcontroller	1.3 oz
Sensor	.15 oz

Table 1: MCU Weight

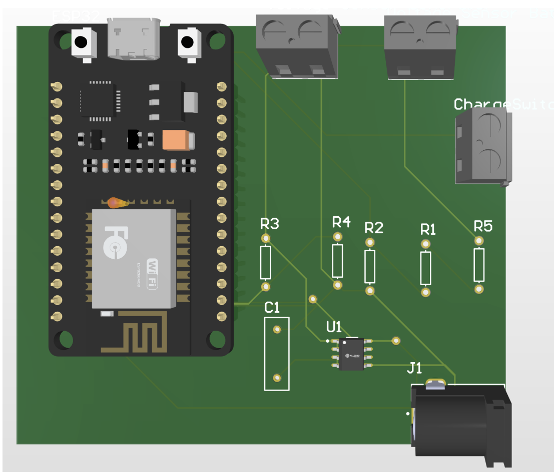


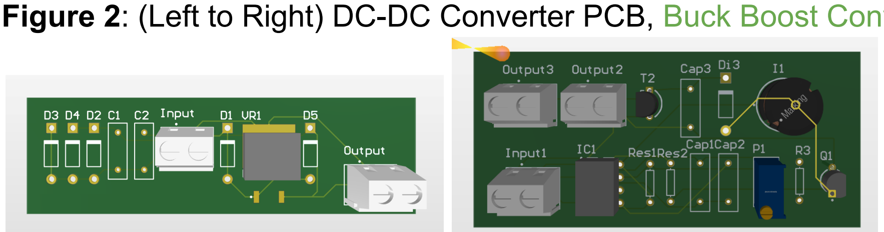
Figure 1: PCB of MCU

3.1.2. DC/DC Converter and Buck Boost Converter

Component	Weight
DC-DC PCB	0.32 oz
Buck Boost PCB	0.62 oz

Table 2: Converter Weight

Figure 2: (Left to Right) DC-DC Converter PCB, Buck Boost Converter PCB



3.1.3. Android Device

Component	Weight
Android phone (Moto e)	5.01 oz

Table 3: Android Device Weight

3.1.4. MISC Non-Subsystem Items

Component	Weight
Tower	TBD
Generator w/ Blades	4.23 oz
9V Rechargeable Battery	1.2 oz

Table 4: Other Weights

3.2. Dimensions

3.2.1. Dimensions of MCU Subsystem

Component	Length	Width	Height
PCB	102.5mm	61.7mm	1.6mm
ESP32 Microcontroller	2.05 in	1.18 in	.59 in

Table 5: MCU Dimensions

3.2.2. Dimensions of DC-DC Converter and Buck Boost Converter Subsystem

Component	Length	Width	Height
DC-DC PCB	78.68 mm	23.63 mm	1.6 mm
Buck Boost PCB	76.58 mm	34.8 mm	1.6 mm

Table 6: Converter Dimensions

3.2.3. Dimensions of Android Device for the App Subsystem

Component	Length	Width	Height
Android Device (Moto e)	124.8 mm	64.8 mm	12.3 mm

Table 7: Android Device Dimensions

3.2.4. Dimensions of Non-Subsystem Items

Component	Length	Width	Height
System Housing	200 mm	170 mm	90 mm
Blades	100 mm	100 mm	34.2 mm
Generator	24.5 mm	24.5 mm	34.2 mm
Rechargeable Battery	48.8 mm	26.0 mm	16.9 mm
Gen to DC-DC Connection PERF Board	75 mm	25 mm	1.6 mm
Boost to Bat Connection PERF Board	60 mm	40 mm	1.6 mm

Table 8: Other Component Dimensions

3.3. Mounting Locations

The wind power system will have small enough weight and dimensions to be easily portable for educational demonstrations at various locations. The PCBs and connection boards will be mounted to the inside of the enclosure through velcro, with jumper wires taped down by tape.

4. Electrical Interface

4.1. Primary Input Power

The primary input source will be a wall outlet with 5V 2A adapter to power the microcontroller unit through a DC power jack connected to the Vin pin on the ESP 32.

4.2. Polarity Reversal Protection

The DC-DC Converter will contain polarity reversal protection using the 4 diode bridge rectifier, in case the generator is plugged in the wrong way into the converter. The full wave rectifier with the diode bridge will allow the converter to nullify the negative voltage, allowing safe output to the buck boost and the rest of the power system.

4.3. Signal Interfaces

Reference the FSR Section 5.2.

4.4. DC-DC Converter

The DC-DC Converter will take the 0-8 V AC input from the fan generator to a constant DC voltage. The DC-DC rectifier will cap the voltage output at 5V if the AC input exceeds 5V to ensure over voltage protection.

4.5. Buck Boost Converter

The Buck Boost converter will take DC voltage from the DC-DC converter and increase the voltage from 5V DC to 8.4V to charge the battery safely. The Buck Boost converter will also receive a 3.3V signal from the MCU to switch on and off power to the battery.

5. Communications / Device Interface Protocols

5.1. Wireless Communications (Bluetooth)

The Microcontroller and the Android phone using the mobile application communicate via Bluetooth serial.

5.2. Host Device

The microcontroller will be using a micro-USB to communicate with the PC when debugging as well as flashing firmware.

5.3. Device Peripheral Interface

The MCU will have a serial port(UART) in order to communicate with the android device and app.

5.4. Android App User Interface

The app has a menu that allows the user to navigate to each of the four pages of the app. Additionally, it has an EditText that allows the user to input the desired max charge level of the battery and a plot of the battery charge level vs time that allows the user to zoom and pan. Additionally, the plot page of the app allows the user to start/stop the plotting to allow the user to view only the time interval that they are interested in viewing.

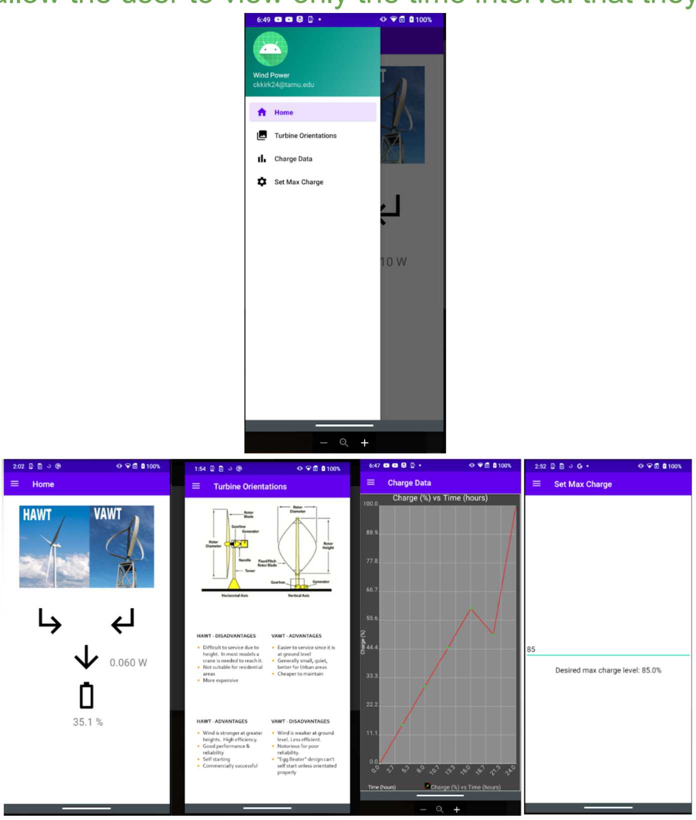


Figure 3: UI of Android app

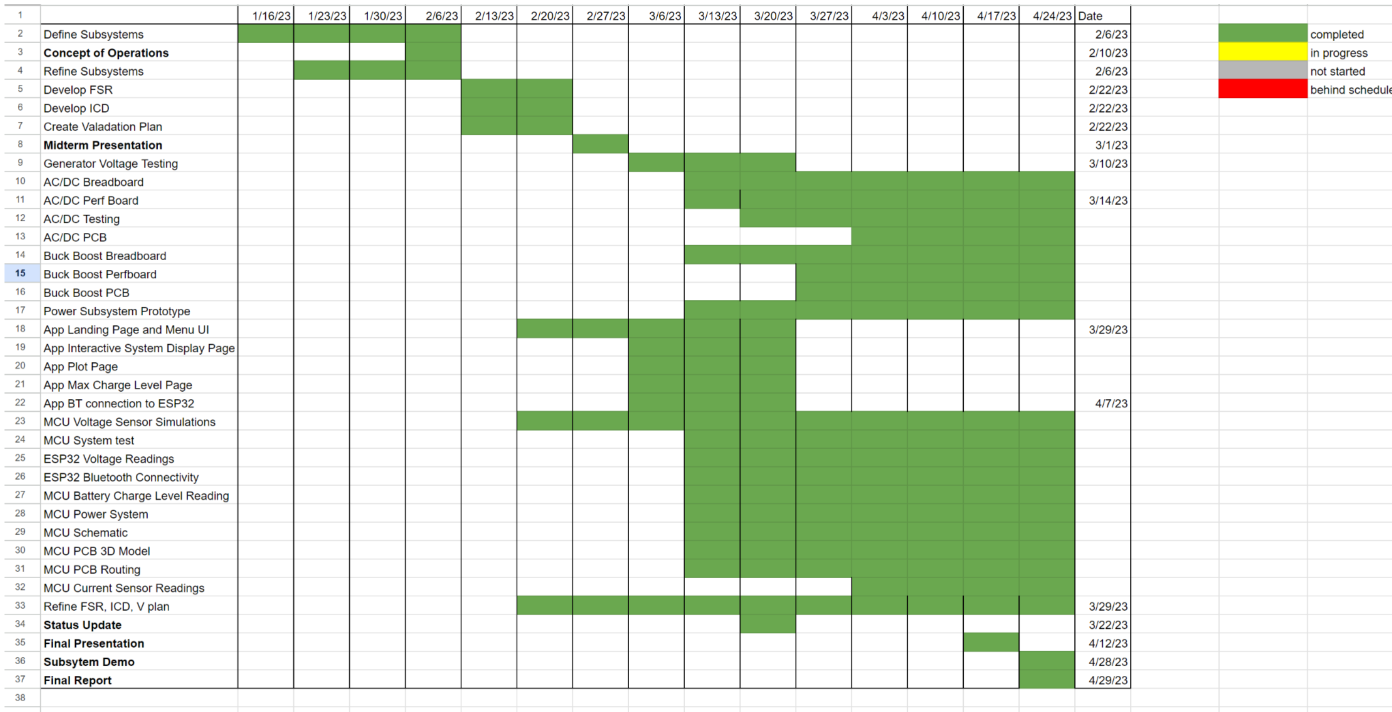
Creating STEM Contents: Wind Power

Christian Kirk, Samuel Tando, Lyndon Li

SCHEDULE AND VALIDATION

REVISION – Final Report
29 April 2023

ECEN 403 Schedule:



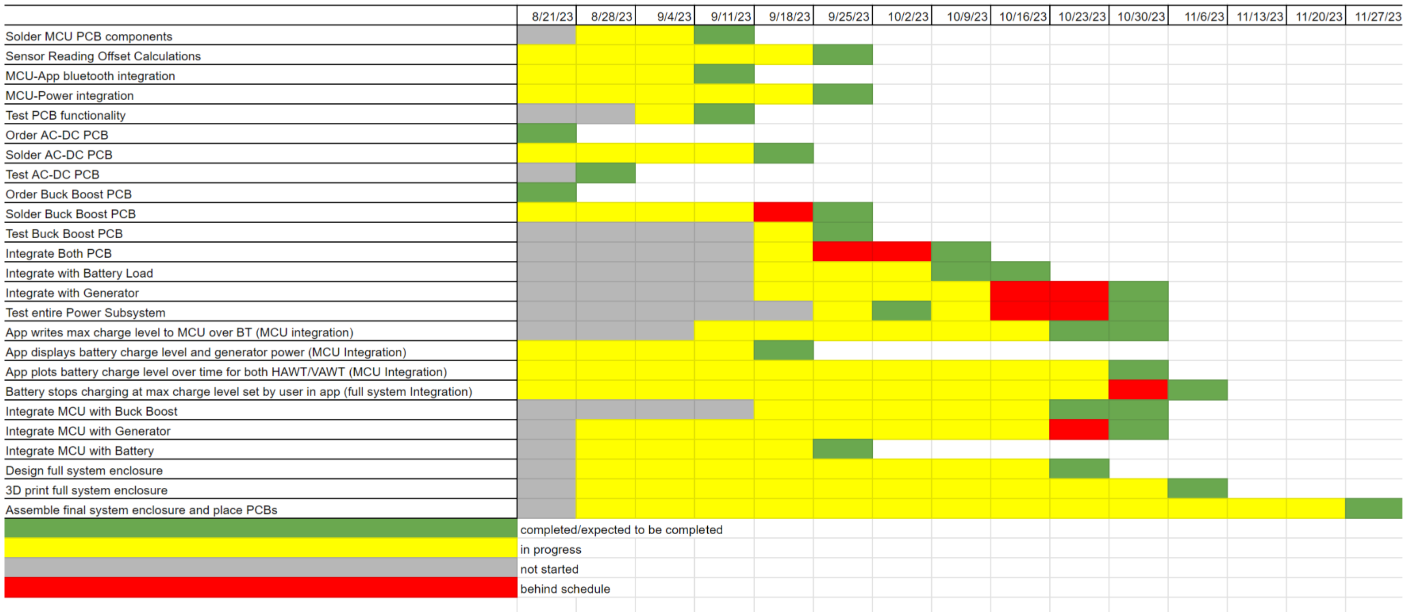
ECEN 403 Validation Plan:

Task	Specification	Result	Owner
Android Device Bluetooth Connection Test	Successfull conncection to ESP32 in App	Complete	Christian Kirk
Data Transfer ESP32 to Android Device Test	Display values from MCU on App home screen (both charge level and power)	Complete	Christian Kirk
Plot Data Page	Plot battery charge level vs time in the corresponding page of the App	Complete	Christian Kirk
Interactive Page	Turbine orientation page has text and images of both orientations	Complete	Christian Kirk
Settings Page	Have a settings page where user can set max charge level	Complete	Christian Kirk
Generator Test	Test voltage levels when wind is applied to it	Complete	Samuel Tando
Interface Connection Test	Make sure that all interfaces are working as intended	In Progress	Everyone
AC-DC breadboard Test	Safely convert AC voltage to DC voltage from 0 to 8.4 Volt range	Complete	Samuel Tando
Buck Converter breadboard Test	Increase DC Voltage to maximum 8.4 Volts, minimum 5 Volts	Complete	Samuel Tando
Linear Voltage Regulator Test	Maintain 5V DC voltage consistently when powered	Complete	Samuel Tando
AC/DC perfboard test	Safely convert AC voltage to DC voltage from 0 to 8.4 Volt range	Complete	Samuel Tando
Duty Cycle Test	Test 555 Timer and ensure that duty cycle changes when potentiometer changes	Complete	Samuel Tando
Housing	Make sure the housing is secure	Not Started	Everyone
MCU Voltage Sensor Simulations	Battery voltage is stepped down from max 9 to 3.3 Volts	Complete	Lyndon Li
	Generator voltage is stepped down from max 24 to 3.3 Volts	Complete	Lyndon Li
MCU Perfboard test	Initial Model with all components created with accurate readings	In Progress	Lyndon Li
ESP32 Voltage Readings	Battery voltage is stepped down from max 9 to 3.3 Volts and readings are within 0.2 V	Complete	Lyndon Li
	Generator voltage is stepped down from max 24 to 3.3 Volts and readings are within 0.2 V	Complete	Lyndon Li
ESP32 Bluetooth Connectivity	ESP32 is able to establish a connection with bluetooth terminal and print out readings	Complete	Lyndon Li
MCU Battery Charge Level Reading	ESP32 Software is able to determine the charge percentage of Battery	Complete	Lyndon Li
MCU Power System	Wall adapter is able to deliver 5V DC Power to the MCU	Complete	Lyndon Li
MCU Schematic	MCU Schematic complete with necessary components and connectors	Complete	Lyndon Li
MCU PCB 3D Model	PCB model is complete with initial placement	Complete	Lyndon Li
MCU PCB Routing	PCB model is routed with intial routing	Complete	Lyndon Li
MCU Current sensor	MCU current sensor is accurate within 5 mA	Complete	Lyndon Li
MCU PCB	All connections working and functioning properly	In progress	Lyndon Li

Performance on ECEN 403 Execution and Validation Plans:

Our plans for this project have been successfully completed in a timely manner. Our subsystems have been validated as individually functional and are ready to begin integration over the next semester.

ECEN 404 Schedule:



ECEN 404 Validation Plan:

Task	Specification	Result	Owner
AC-DC PCB Test	Maintain 5V DC voltage consistently from 0-8.4 V AC Source	Passed	Samuel Tando
Buck Converter PCB Test	Increase DC Voltage to maximum 8.4 Volts from 5V DC Source	Passed	Samuel Tando
Combined Circuits Test	Output 8.4 Volts from combined circuits through 0-8.4 AC Source	Passed	Samuel Tando
Integration with Generator	Power generator with Hair Dryer and output 8.4 V DC with combined circuits	Passed	Samuel Tando
Integration with Battery Load	Input 8.4 V to Battery Load to charge battery with controlled 0-8.4V AC Input	Passed	Samuel Tando
Full Power subsystem test	Charge Battery Load with 8.4V Output from AC Generator with both circuits	Passed	Samuel Tando
Integrate with MCU	Output <0.5V from Buck Boost to Battery when MCU sends 5V signal to transistor	Passed	Samuel Tando
MCU PCB Assembly test	PCB led indicator lit when 5V power supply plugged in. Bluetooth enabled on startup and discoverable.	Passed	Lyndon Li
MCU PCB Functional Test	Sensor readings are updating every second with calculated generator wattage and battery level percent	Passed	Lyndon Li
MCU PCB sensor validation	Sensors are within 0.1 mV or 0.1A of actual value using DC Power Supply	Passed	Lyndon Li
MCU charge control	Charge control pin is toggled low when powered on and toggles high when battery level over 90%	Passed	Lyndon Li
App/MCU integration test 1 (MCU sensor read)	App displays both real time battery charge percentage and power output from generator on the home page	Passed	Christian Kirk
App/MCU integration test 2 (Write to MCU)	The BT output stream from the app is received by the MCU and printed in the MCU's serial monitor	Passed	Christian Kirk
App HAWT/VAWT test	The plot page displays battery charge percentage over time for both the HAWT and VAWT	Passed	Christian Kirk
HAWT 3-D Print Test	3D printed blades generate between 6-7 Volts	Failed	Christian Kirk and Samuel Tando

Performance on ECEN 404 Execution and Validation Plans:

During 404, we were able to finish up our subsystem validation from 403 and get our full system integrated on time. There were a few setbacks during integration that made us take longer than we expected, but overall we executed our schedule and validation plans well and ended up with a system that met the functional requirements.

Creating STEM Contents: Wind Power

Christian Kirk, Lyndon Li, Samuel Tando

SUBSYSTEM REPORTS

REVISION – Final
2 December 2023

SUBSYSTEM REPORTS
FOR
Creating STEM Contents: Wind Power

TEAM 12

APPROVED BY:

____
Project Leader Date

____
Prof. Kalafatis Date

____
T/A Date

Change Record

Rev.	Date	Originator	Approvals	Description
1	4/26/2023	Christian Kirk		403 Final Report
2	12/2/2023	Christian Kirk		404 Final Report

Table of Contents

List of Tables

List of Figures

1. Introduction

The wind power system created in this project is broken down into three subsystems: the power subsystem, the MCU subsystem, and the android app subsystem. The power subsystem converts the AC output of the generator into the appropriate DC voltage to charge the nine volt battery load. The MCU subsystem collects sensor data of current and voltage from the generator and the load and then communicates that data to the Android app via Bluetooth. The Android app acts as a real time display of the wind power system power generation and battery charge level as well as allowing the user to set the desired max charge level of the battery. Below, each of these subsystems is described in more detail along with how they were validated and how they will be integrated together.

2. App Subsystem Report

2.1. Subsystem Introduction

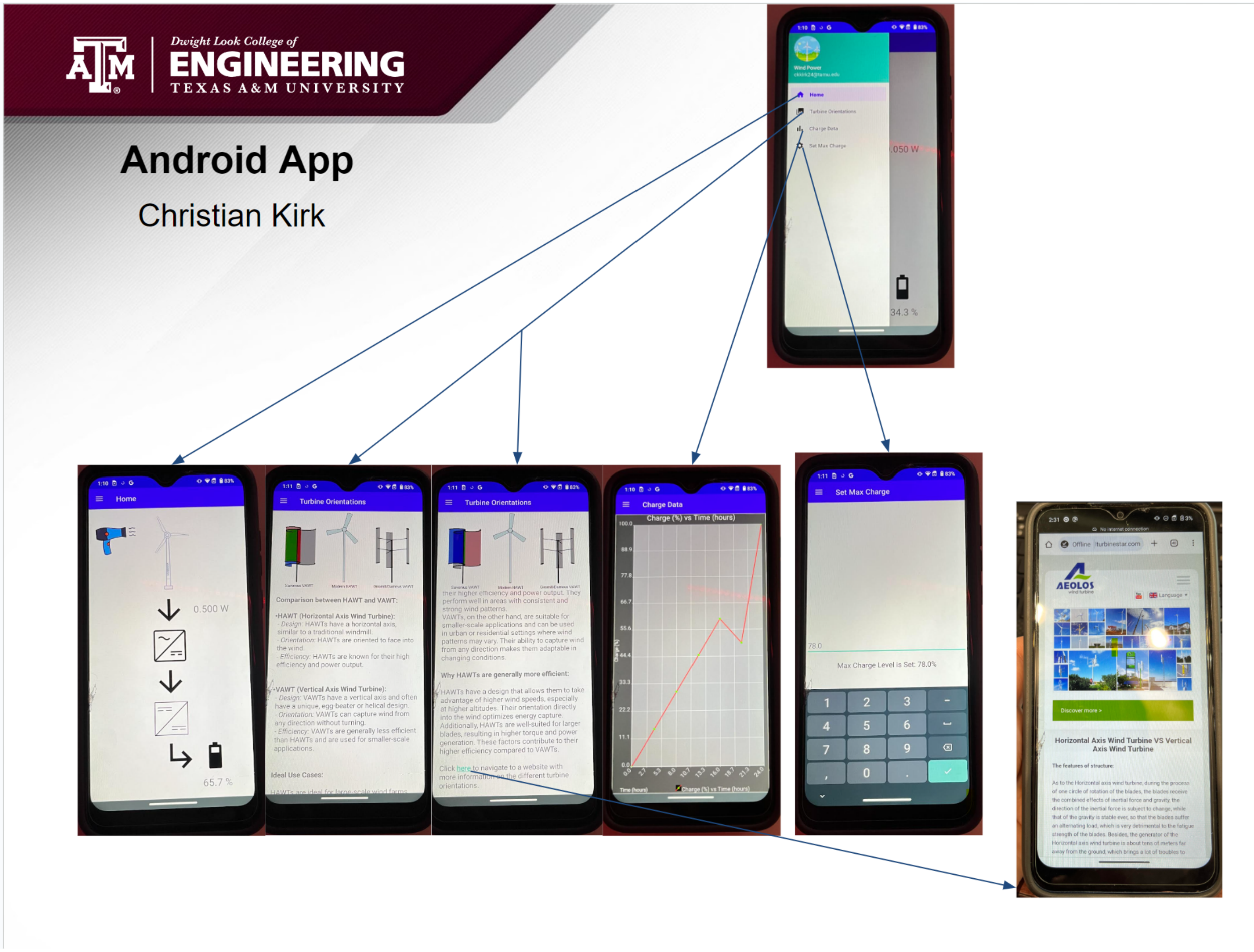
The App subsystem is an Android application that acts as a visual aid for the wind power system. It shows the real time power being produced by either the HAWT or VAWT as well as the real time charge level of the 9 volt battery load. It also acts as an educational tool with a page describing the differences between the two different turbine orientations and the trade offs between them. It also allows the user to set the desired maximum charge level of the battery and plots the battery charge level over time. The purpose of this app is to be an interactive educational tool for kids as they are being presented with our wind power system.

2.2. Subsystem Details

There are four different pages on the app: the home page, the turbine orientations page, the data page, and the set max charge level page. The user can switch between these pages using a menu that was implemented in Android Studio using XML fragment files and an app bar. The home page displays what is going on in the wind power system real time (power, and battery charge level) by connecting to the ESP32 via standard Bluetooth and receiving data from it. The system design page displays animations and scrollable text describing HAWTs and VAWTs. It also has a clickable link that takes the user to a website that has additional information about wind turbines for kids. The data page shows the plot of battery charge level over time. The settings page is where the user can set the maximum charge level of the battery. **Figure 1** below shows the menu and each of the fragment pages it navigates to.

Figure 1: Android App Menu and Pages

Final UI for the integrated system in 404:



2.3. *Subsystem Validation*

For validation of this subsystem, the app was downloaded onto a physical Android device (Moto E) and each of the pages was navigated to in the menu and successfully used as intended. Each page worked correctly with the simulated data. The next step will be replacing the simulated data with the real sensor data from the MCU sensors.

2.3.1 Bluetooth Connection and Simulated Data Validation

The simulated data sent over BT from the ESP32 to the App was the Battery Charge Level in percentage and Power output from the generator in Watts. These values were set in the Arduino IDE as oscillating between 0-100% and 0-0.55 watts respectively. This mirrors what the expected sensor values would be. The values were also set to change every second and the Arduino IDE code was downloaded onto the ESP32. After establishing BT connection, these values were shown in the correct textViews on the app and updated every second as expected.

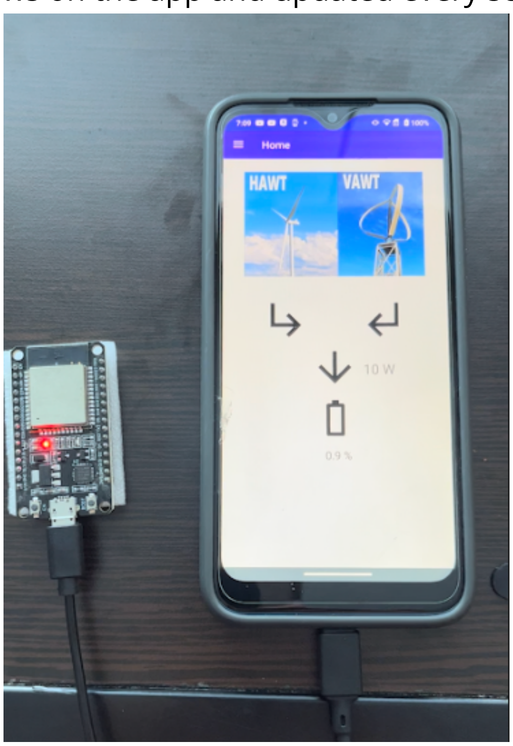


Figure 2: BT Data Validation for App

2.3.2 User Input Validation

The user is able to set their desired max charge level for the battery on the corresponding page of the app. This again was validated on the physical device. It can be seen in the figure below that the user is only able to enter a value between 0-100%. Once the user enters the value, the app determines whether the stored user input is valid or invalid and prints accordingly.

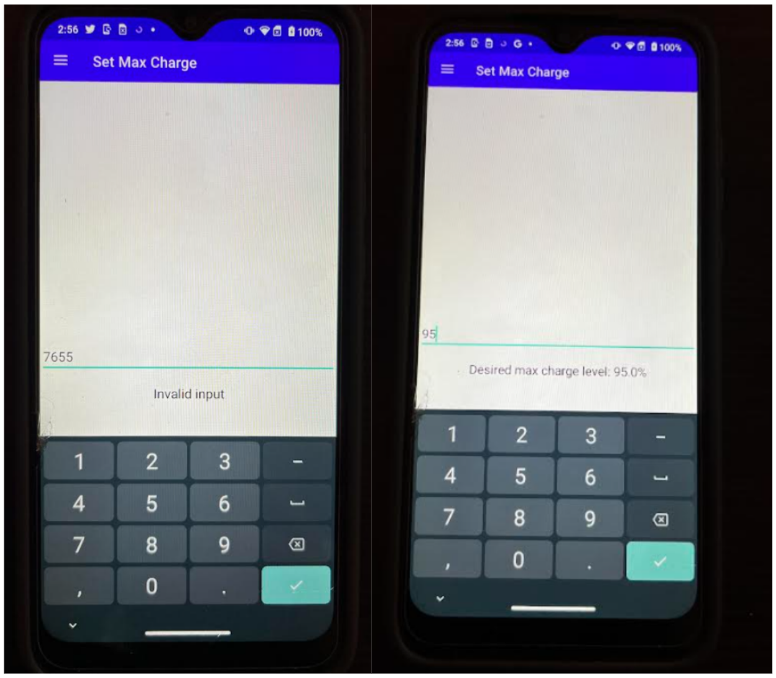


Figure 3: User Input App

2.3.2 Plotting Page Validation

The plot in the app was able to be turned on and off and display the battery charge level of the battery correctly. When turned on, the plot continuously updates with a new data point as soon as it is received from the MCU (every second real time). When turned off, the plot continues storing data points in the background, but the plot does not grow. The advantage of this is that the user can pan and zoom the plot to see the full curve. When turned back on, the plot will plot all of the points since the user turned the plot off. it is a pause functionality. All of this worked as intended, and the plot below shows us first connecting a battery with a charge percentage of around 5% and then quickly swapping it with a battery with a charge level of 87%.

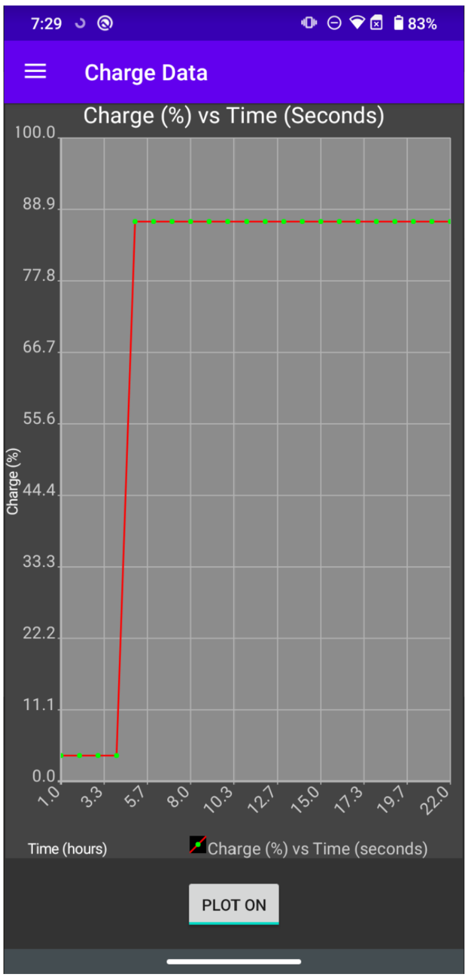


Figure 4: App Plotting Page

2.4. Subsystem Conclusion

Each page of the app works as intended for the individual subsystem. The next step is integrating the actual sensor data from the MCU. The menu UI and the individual pages work as intended.

3. MCU Subsystem Report

3.1. Subsystem Introduction

The purpose of this project is to demonstrate the workings of wind energy for educational purposes so we decided the best way to demonstrate the findings would be to create an app for displaying our data. In order to get that data, we will use a microcontroller to gather the voltage and current readings from the generator and send those readings to a mobile device.

3.2. Microcontroller

3.2.1. Operation

For the microcontroller, we chose to go with the ESP32 microcontroller due to its onboard DAC as well as its Bluetooth capabilities. This would allow us to take in the sensor readings and output an analog value. The Bluetooth capabilities also allowed us to connect to a mobile device easily and transfer data.

3.2.2. Validation

To validate that the ESP32 microcontroller would fit our needs, I used an example Bluetooth serial finder that was built into the Arduino ide. Once I was able to compile the example code and flash the firmware onto my ESP32 board, I was able to verify that the Bluetooth operation was functioning. I further tested the board by sending dummy values from the board into a Bluetooth terminal on my PC and observed that there were no problems. We then had to figure out a way to power on the ESP32 without using the micro USB used to flash the firmware. We went with a power jack connector because we had a 5V 2A wall adapter on hand and when we connected the leads, we found that the ESP32 successfully turned on and was operational.

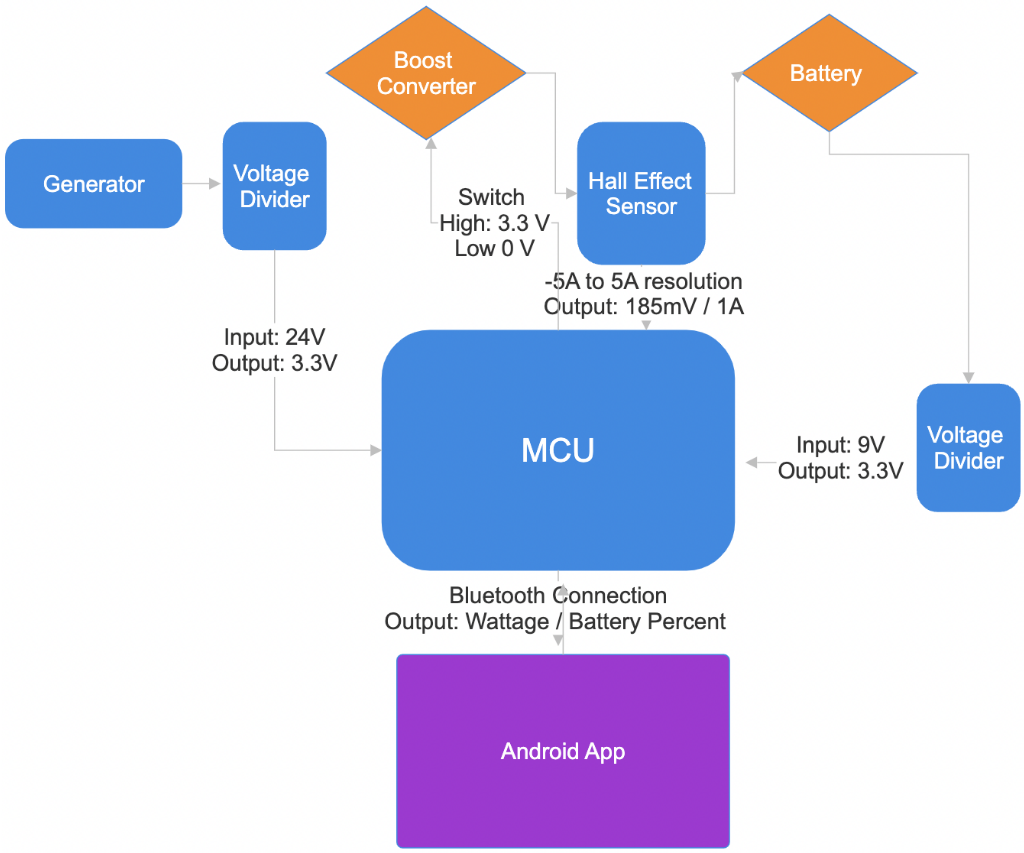


Figure 5: Block Diagram of MCU Subsystem

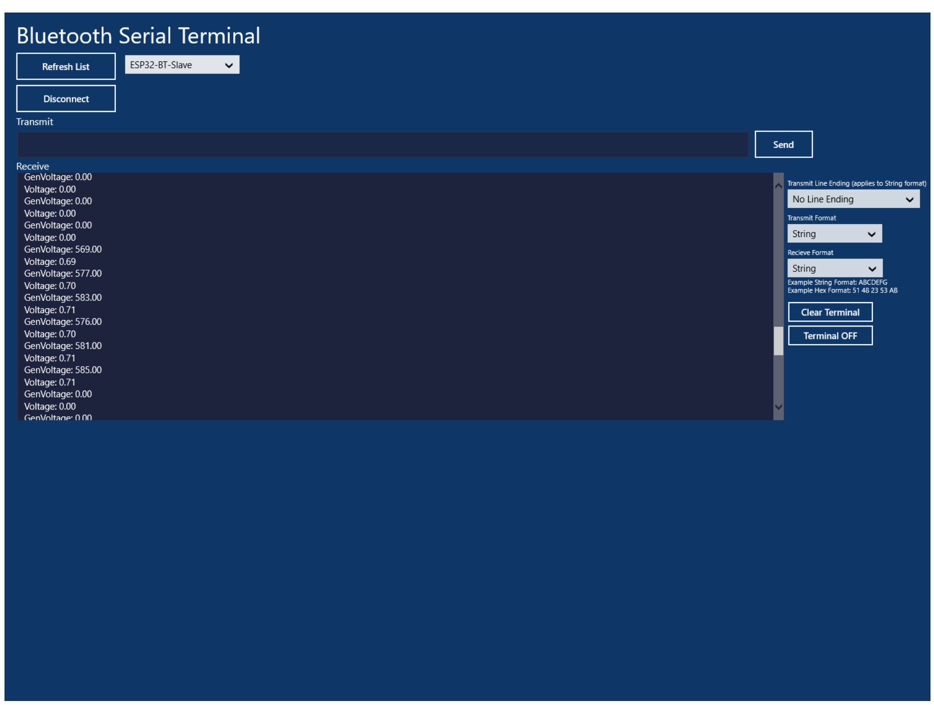


Figure 6: ESP32 Connected to Bluetooth

3.3. Voltage Sensor

3.3.1. Operation

For the voltage sensors, we used a voltage sensor to gather voltage readings from both the battery and the wind generator. The voltage sensor is a voltage divider with resistor values picked out to step down the voltage to 3.3V, which is the max voltage that the ESP32 can take in as input. For both sensors, we chose resistor values that could step down the voltage from 9 to 3.3 as that was the intended max output voltage we wanted from our generator as well as the max capacity for our 9V rechargeable battery.

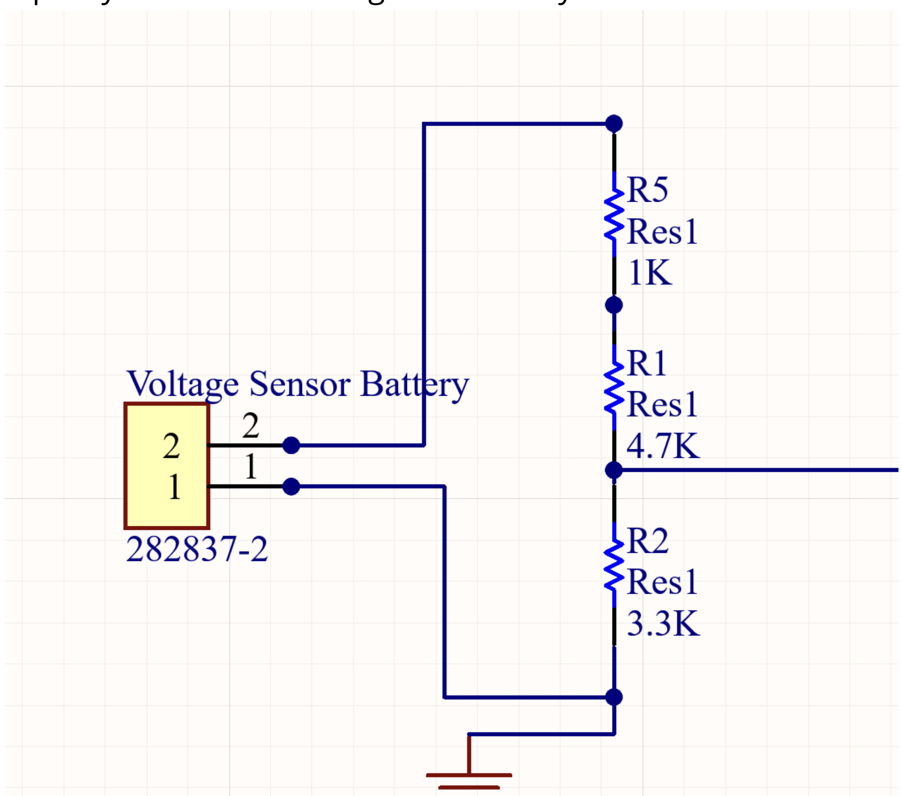


Figure 7: Voltage Sensor Schematic

3.3.2. Validation

To validate the voltage sensor, I used the DC power supply to get a constant voltage reading from the sensors. In order to get the raw input values outputted as the actual readings, we had to first convert the raw values into a voltage reading by utilizing our 12 bit DAC. We then converted that input voltage into the original voltage by multiplying by the step down factor. After that, we found the output values were still off by a nonlinear margin. To combat this, I created a lookup table from the data values and did real time calculations on each voltage reading to get more accurate results.

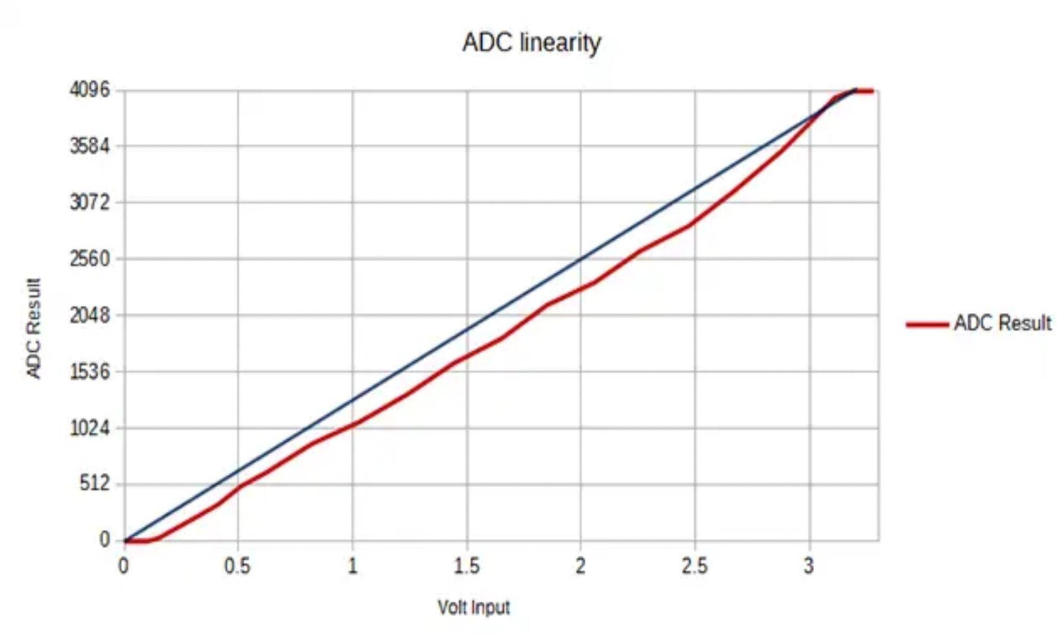
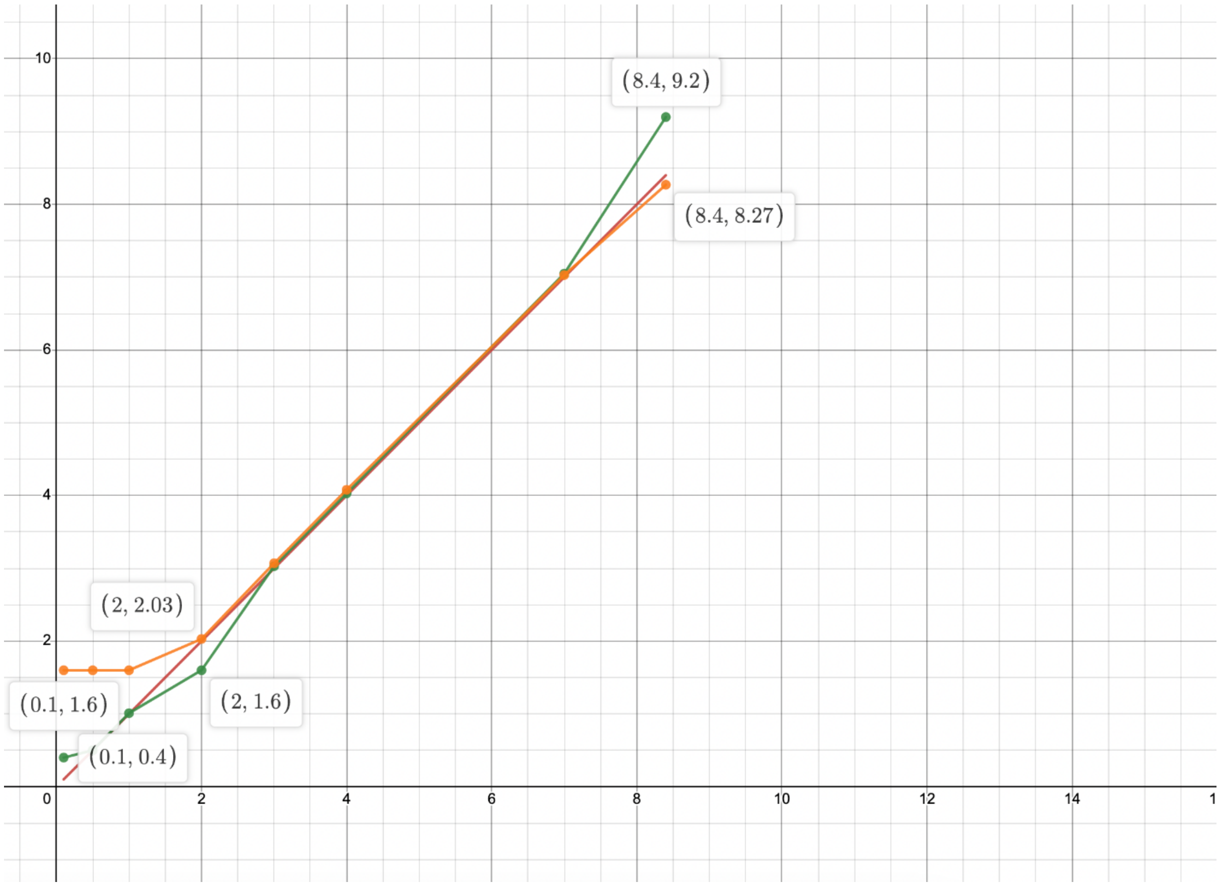


Figure 8: ESP32 Onboard ADC vs Ideal ADC Values



Red = ideal sensor values
Green = Battery Voltage Values
Orange = Generator Voltage Values

Figure 9: Experimental Voltage Readings vs Ideal

	generator	Power supply	Mcu	Battery	Power supply	Mcu
	1	1.04	0.11	1	0.99	0.91
	1.5	1.51	0.76	1.5	1.48	1.63
	2	2.02	1.53	2	2	2.4
	2.5	2.58	2.35	2.5	2.55	3.21
	3	3.03	3.01	3	3.02	3.92
	3.5	3.6	3.85	3.5	3.55	4.7
	4	4.04	4.48	4	4.1	5.52
	4.5	4.56	5.29	4.5	4.6	6.27
	5	5.05	6.01	5	5.09	6.98
	5.5	5.59	6.79	5.5	5.61	7.75
	6	6.12	7.58	6	6.09	8.44
	6.5	6.59	8.28	6.5	6.58	9.19
	7	7.13	9.09	7	7.13	10.01
	7.5	7.59	9.82	7.5	7.6	10.81
	8	8.18	10.69	8	8.17	11.95
	8.5	8.68	11.45	8.5	8.66	13.17
	9	9.18	12.19	9	9.2	14.48
	Difference	Generator	Battery			
	1	0.93	0.08			
	1.5	0.75	-0.15			
	2	0.49	-0.4			
	2.5	0.23	-0.66			
	3	0.02	-0.9			
	3.5	-0.25	-1.15			
	4	-0.44	-1.42			
	4.5	-0.73	-1.67			
	5	-0.96	-1.89			

Figure 10: Data Collected for Lookup Table

```
//lookup table for generator and battery offset values

double genValues[] = {.983,.9877,.9923,.997,1.0016,1.0063,1.0109,1.0156,1.0202,1.0295,1.03415,1.0388,1.04345,1.0481,
1.05275,1.0574,1.06205,1.0667,1.07135,1.076,1.08065,1.0853,1.08995,1.0946,1.09925,1.1039,1.10855,1.1132,1.11785,1.1225,1.12715,
1.1318,1.13645,1.1411,1.14575,1.1504,1.15505};

double batValues[] = {.381,.38555,.3901,.39465,.3992,.40375,.4083,.41285,.4174,.4265,.43105,.4356,.44015,.4447,.44925,
.4538,.45835,.4629,.46745,.472,.47655,.4811,.48565,.4902,.49475,.4903,.50385,.5084,.51295,.5175,.52205,.382,.22675,.0715,-.08375,
-.239,-.39425};
```

Figure 11: Lookup Table for Offset

Once I was happy with the actual values coming out, I had to fix the fluctuations within the output. I was constantly getting dropped readings and readings that were outliers as being too low or too high. To fix this, I went back into the firmware and instead of reading the instantaneous input voltages, I read in an average of 500 samples to get a more accurate reading. This slowed down the update rate, but greatly improved the accuracy.

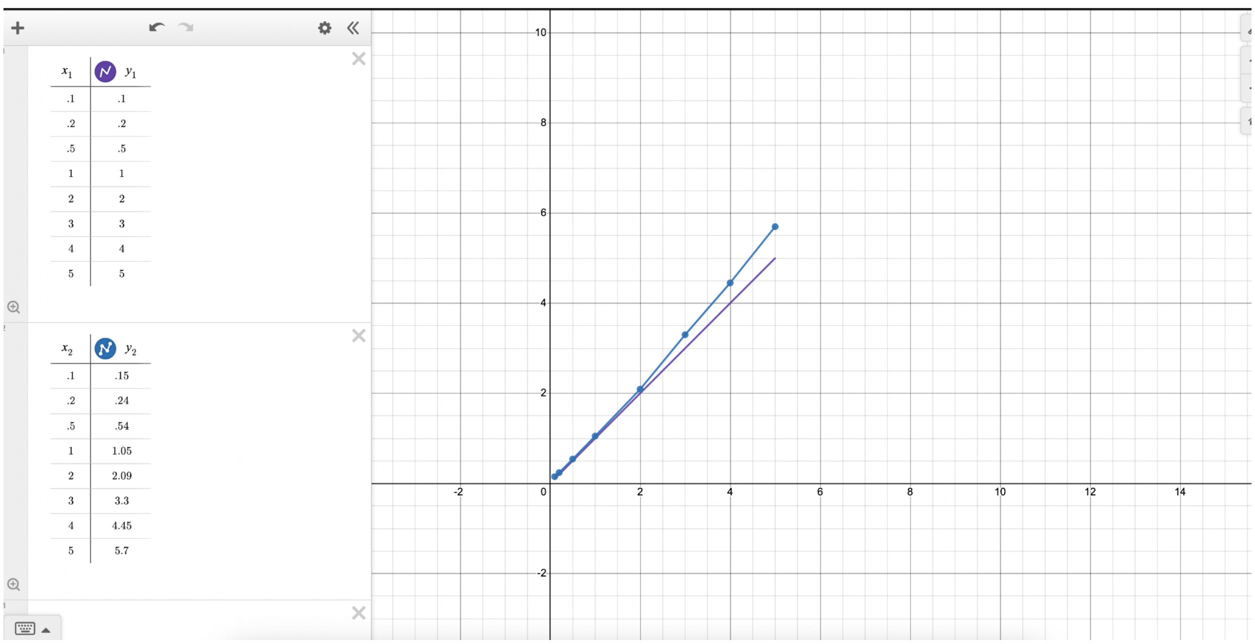
3.4. Current Sensor

3.4.1. Operation

The current sensor is used to measure the current from the generator in order to calculate the power from the generator. This current sensor output is dependent on the input power into the ESP32 so our validation testing and offset calculations are under the condition that our input is 3.3V rather than the final 5V.

3.4.2. Validation

For the current sensor, we decided to use the acs712elctr line of current sensors. We found that the original 30A range sensor had too high of a range and the power output from our generator was too low to be read. We decided to go with the 5A current sensor because it had a higher resolution at lower current which was better for our needs. We also modified our board to have the current sensor be interfaced between the boost converter and the battery in order to measure the current going into the battery. Similar to the voltage sensor, we had to convert the raw input into the actual input from utilizing the DAC. For the 5A version of the chip, each amp of current is supposed to be symbolized by a change of 180 mV. Once we were able to convert our voltage reading into a current reading, we found that the current sensor assumes that the base voltage reading at 0 current will correspond to a reading of 2.5 Volts. This was not the case, as our base value at 0A was around 2.4 V. Due to this, we had to calibrate our current sensor formula with this base value before we could look into our offset value.



Purple = Ideal
Blue = Actual
Figure 12: Current Sensor Readings Ideal vs Actual

From Figure 4, we could see that the actual readings are much larger at the top end compared to the bottom end. We are fine with these results because from our initial readings from the generator, we don't see ourselves being able to reach a full 5A of current. The range in which the sensor is accurate will be enough for our purposes. To verify that our current readings were accurate, I used a DC power supply in constant current mode to get current running across our current sensor, I then applied the same power supply to a multimeter and verified that the values were aligned. Ultimately, we were able to get within the 10mA range of our expected values, fulfilling our original spec needs.

3.5. Charge Control Switch

3.5.1 Operation

The charge control switch is used to turn off the boost converter in order to take a measurement of the battery voltage level for battery level calculations. In order to turn off the boost converter, we implemented a transistor at the end of the boost converter that would turn on or off depending on the charge control output. We tied this switch to an internal timer that would switch every 1000 ms.

3.5.2 Validation

We validated this switching mechanism by using the multimeter to verify that the output signal was switching from 0 to 3.3 Volts. We then tested the boost converter output to see if the output was turning on or off. Once both of these conditions had been met, it was a matter of figuring out what spec our timer should be run at. We found through testing that when the timer was switching every 5 seconds, we would run into issues where the delay would interfere with the calculations and give unstable values. Through our testing, we found that a 1 second switch worked best in terms of update speed and accuracy.

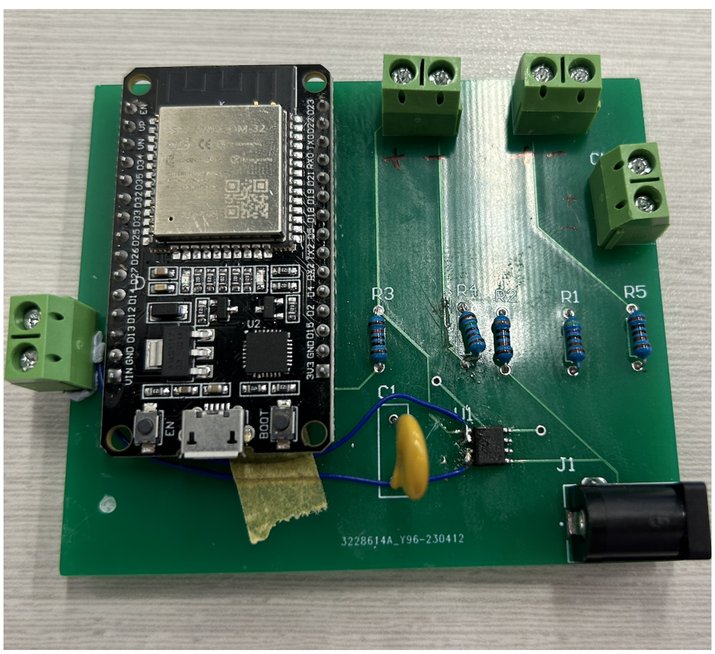


Figure 13: MCU Subsystem

Specification	Measured Power through Digital Multimeter	Measured Power through MCU
Across Resistor Load	3.83V, 4.2 mA	3.81V, 4 mA
Across Integrated System	8.3V, 40 mA	8.35V, 40 mA

Table 1: Test Results

3.6. *Subsystem Conclusion*

Each piece of our subsystem works as intended and the values that are outputting are within our tolerances. The ESP32 is able to be powered on by our intended power source and the Bluetooth data transfer is working without a problem. The charge control switch is successfully switching every second and turning on and off the boost converter for battery readings.

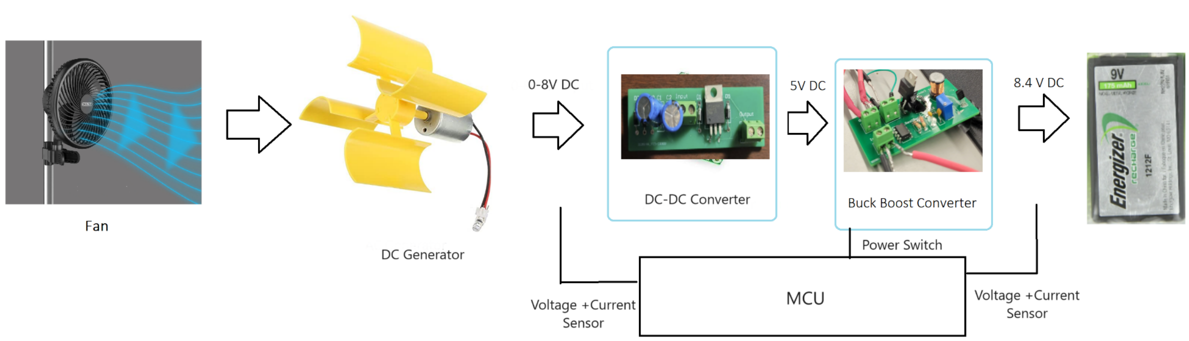
4. Power Subsystem Report

4.1. Subsystem Introduction

The power subsystem represents a scaled down version of a wind turbine power plant, taking the input from a fan generator (wind turbine), and converting the power to power a home (battery load). To safely charge the 9V battery load we will use an DC-DC rectifier to transform the wind powered voltage to a consistent voltage output to the Buck Boost Converter to boost the voltage.

4.2. Subsystem Details

Figure 14: Functional Block Diagram of power subsystem



The power subsystem consists of three main components: the DC Generator, the DC-DC rectifier, the Buck-Boost converter, and the battery load.

The DC Generator produces a voltage proportional to the speed of the turbine depending on the fan's power. The DC-DC rectifier converts the varying 0-8 VDC to a constant DC voltage, which is capped at 5V DC if the DC generator exceeds 5V. The Buck-Boost converter then increases the voltage from 5V to 8.4V, which is required to safely charge the 9V battery load. The MCU and the Android app will monitor the 9V battery load and send a signal to the switch on the buck boost converter to shut off power when it reaches a specific limit or reaches 100% charge.

A PCB design was made for the DC-DC rectifier and Buck boost converter, which was ordered and used for testing.

4.2.1. DC Generator

4.2.1.1. Operation

The DC generator we chose to use was a micro vertical wind turbine manufactured by SYWAN. We chose this specific generator as the output of 0-6.67 V was within the voltage range we desired of 0-8.4V, and worked consistently with different fan wind speeds tested on it. The generator output 0.55 W and around 0.4-0.6

A when tested with a 100 Ohm load, which was ideal as the 9V battery had a 600 mAh rating, meaning the ideal current to charge the battery is at 0.6 A. The design stayed consistent from 403 into the final system integration.

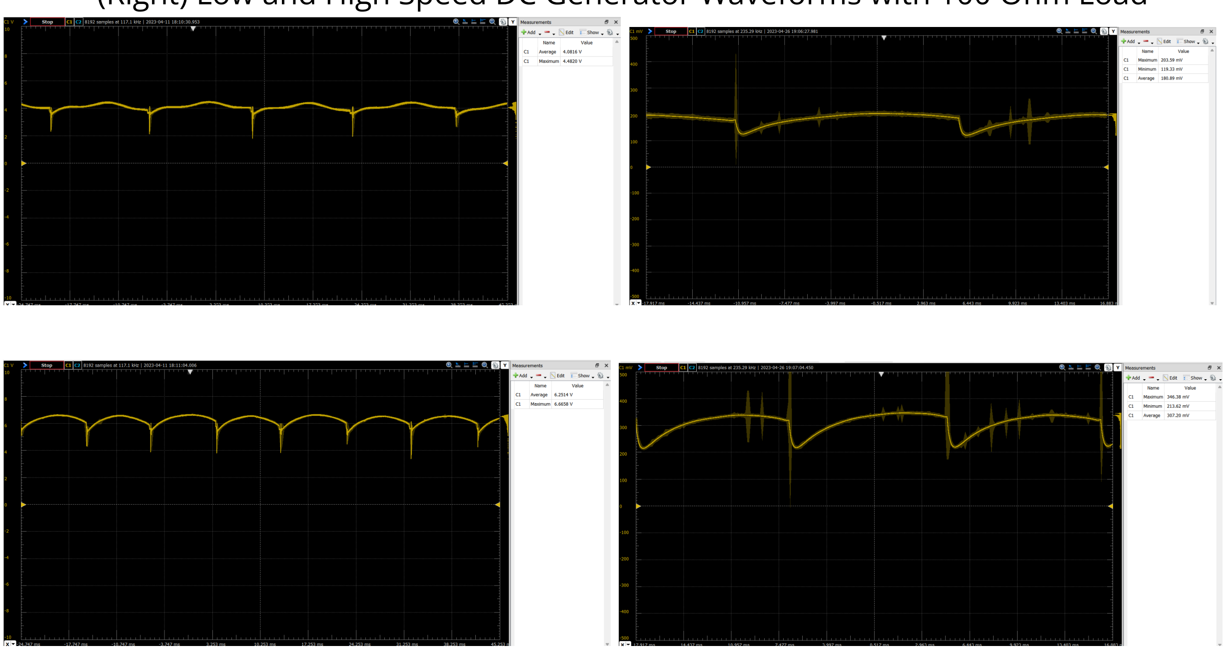
4.2.1.2. Validation

The DC generator was tested with a hair dryer, which had a low and high speed setting, and an AD2 waveform to measure the output. The hair dryer would be aimed at a certain angle towards the generator turbine to generate a constant voltage range.

Table 2: DC Generator Data Measurements

Fan Setting	Average Voltage (V)	Max Voltage (V)	Output Current (mA)	Output Power (W)
Low Speed	4.08	4.48	73.5	0.3 W
High Speed	6.25	6.67	80	0.5 W
Low Speed (w/ 100 Ohm Load)	0.181	0.204	1.8	0.3 mW
High Speed (w/ 100 Ohm Load	0.31	0.35	3	0.93 mW

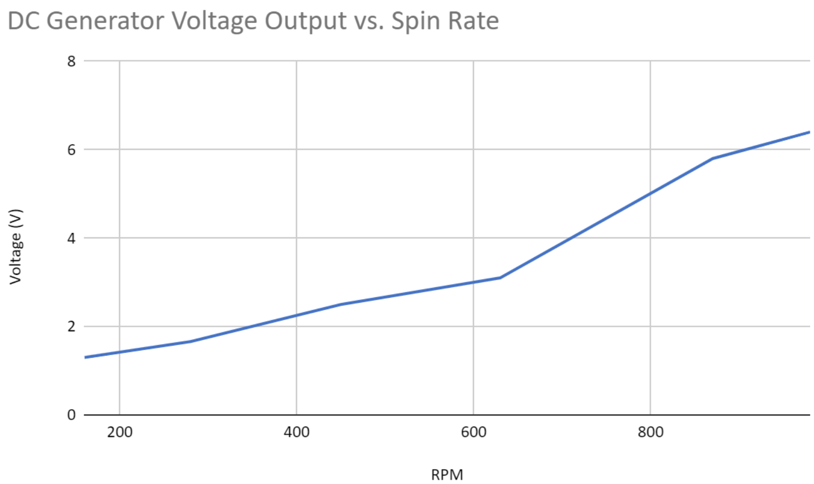
Figure 15: (Left) Low and High Speed DC Generator Waveforms
(Right) Low and High Speed DC Generator Waveforms with 100 Ohm Load



The results showed a somewhat jumpy signal coming from the generator, but the values were within range of tolerance. The waveforms showed a large decrease in voltage once a load was applied, but worked fine after the DC-DC rectifier was added on.

The RPM(rotations per minute) and voltage output correlation was also measured by a tachometer. We attached the generator motor to the takometer which would read the rotation speed, while the generator was hooked up to a multimeter. Once this was completed, we would vary the wind speeds with our fan and collect the RPM and voltage accordingly.

Figure 16: DC Generator Voltage Output (V) vs. Spin Rate (RPM)



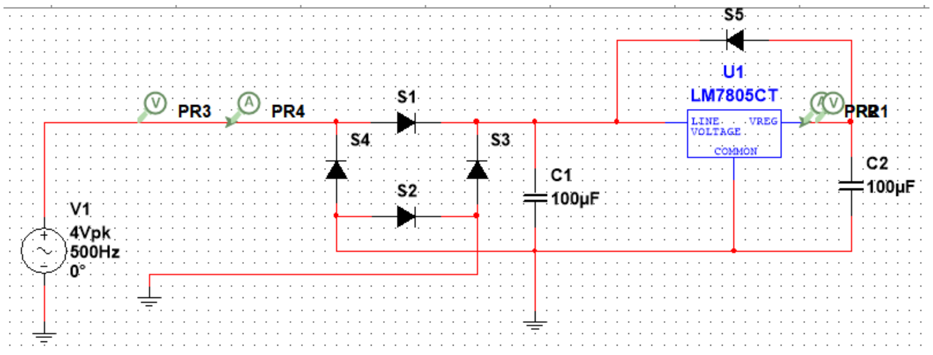
4.2.2. DC-DC Rectifier

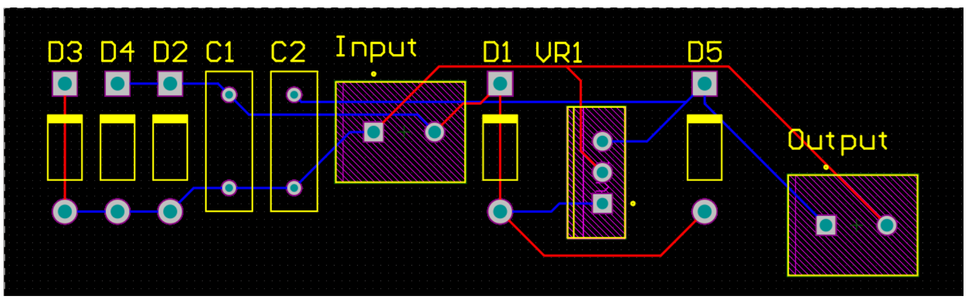
4.2.2.1. Operation

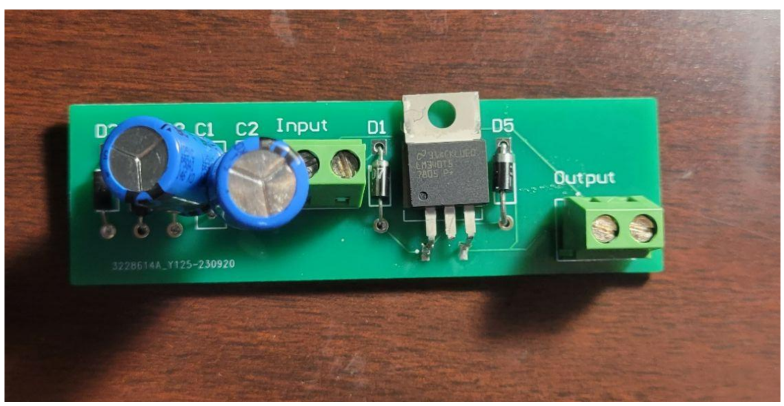
The objective of the DC-DC rectifier is to convert the 0-8 V AC input from the fan generator to a constant DC voltage. For the design of the DC-DC rectifier, we decided to use a full-wave rectifier system with a LM7805CT linear voltage regulator. The full-wave rectifier with a 4 diode bridge rectifier ensures reverse polarity protection, so the DC output will always be positive. The LM7805CT would keep the voltage constant at around 5V in case the input voltage exceeded 5V, ensuring upper voltage protection. The large capacitors help to smooth out the ripples from the generator.

Originally the DC-DC was to be an AC-DC converter, as the original generator was AC, but the functionality was kept the same, as the rectifier could serve as polarity protection and operate the same way.

Figure 17: (Top to Bottom) Multisim Simulation DC-DC Schematic, DC-DC Altium PCB Design, and soldered PCB







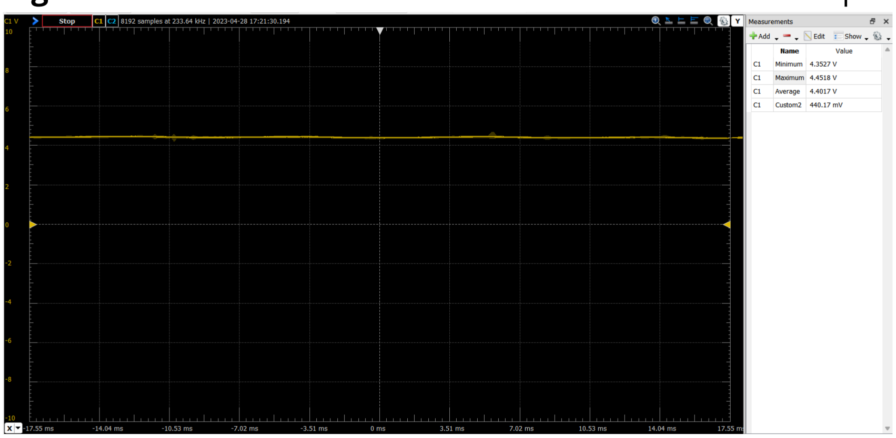
4.4.2 Validation

The DC-DC rectifier was tested with an AD2 to measure voltage and current as well as generate waveforms. We also used the fan generator to test to ensure that the DC-DC was compatible with the input voltage coming from the generator. The load measurements were done by the power source in the FEDC with an electronic load to measure how the current draw would affect the PCB.

Table 3: DC-DC Test Data Measurements

Input (V)	Simulated Output:	Perfboard Output:	Perfboard With 10 Ohm Load:
Sine Wave 0 to 5V	2.83 V	2.58 V	2.40 V 0.24 A
Sine Wave 0 to 8V	5.008 V	4.64 V	
Sine Wave -5 to 5V	2.82 V	3.04 V	2.83 V 0.28 A
Fan Generator (High Setting): 3.6 to 7.4 V Average 7 V		4.76 V	4.45 V 0.44 A

Figure 18: DC-DC Rectifier Waveform with DC Generator Input



The results of the waveforms showed a constant DC voltage each time with very little noise, regardless of the range of the sine waves. The linear voltage regulator was also able to limit the exceeding 5V input to 5 volts.

Table 4: DC-DC Rectifier Load Data

Regulator Input (VDC)	Current Draw (A)	Output Voltage (V)
5	0.116	2.784
6	0.132	3.7
7	0.16	4.55
8	0.143	4.938

4.2.3. Buck Boost Converter

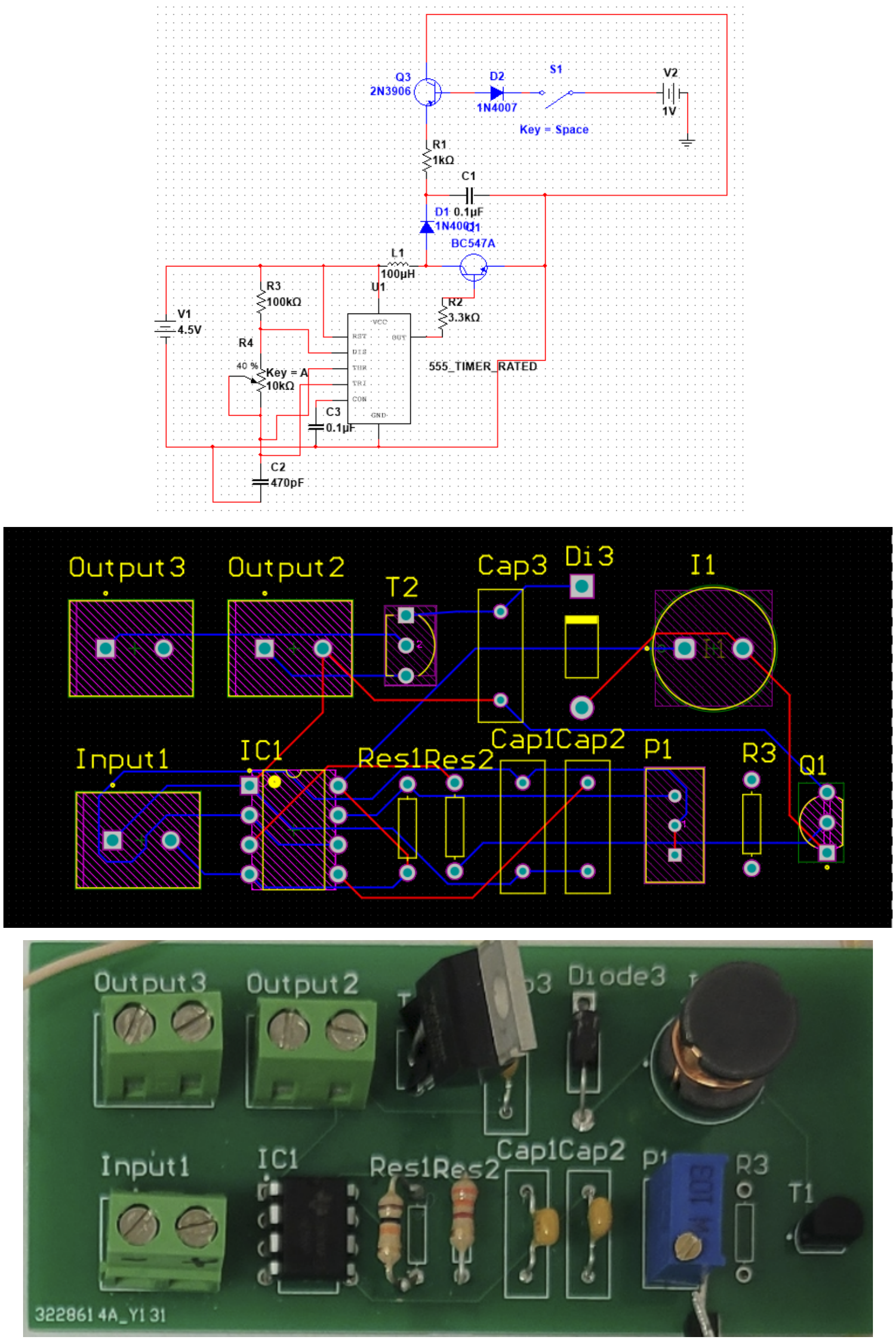
4.2.3.1. Operation

For the design of the buck boost converter, we used a common buck boost topology with a BJT (Bipolar Junction Transistor), 555 timer and 10k Ohm potentiometer for pulse width modulation. The 555 timer IC is used to take the input of the DC and generate a square wave to control the BJT to allow voltage to the inductor.

The original objective of the buck boost converter was to increase the 5V input from the AC-DC rectifier to 8.4 V to safely charge the battery. However, when testing the power system altogether, the DC-DC voltage would drop down to 2.2. After a few attempts to reconcile the problem, we were not able to find a solution, so we decided to instead up the boost from 2.2V to 8.4V.

At the load of the buck boost contains the switching mechanism, which blocks power from the buck boost going to the battery load. This is done by a transistor that will close or open the current going to the battery when a 3.3V signal is sent by the MCU. The buck boost was redesigned from simulation to hardware, as the original design from 403 did not perform as expected.

Figure 19: (Top to Bottom) Multisim Simulation Buck Boost Schematic, Buck Boost PERF Board, Buck Boost Altium PCB Design, Soldered PCB



4.2.3.2. Validation

The Buck Boost was tested with a multimeter and voltage source from the FEDC. To ensure that the buck boost had proper boost, the 555 timer was tested to ensure that the duty cycle varied accordingly. The buck boost would not boost when lower than 2V, but anything past it would be able to have a considerable boost depending on the duty cycle.

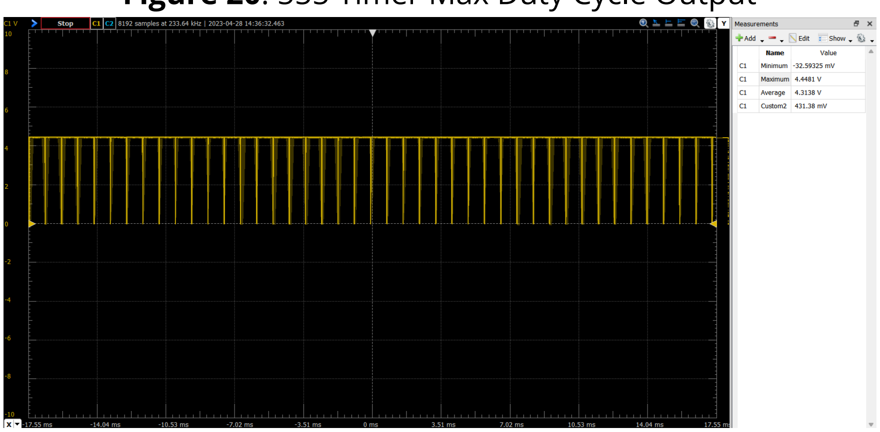
Buck Boost at 98% Duty Cycle, Potentiometer at 9.86k Ohms

Table 5: Buck Boost Input vs. Output

Buck Boost Input (VDC)	Buck Boost Output Voltage (V)
2 V	1.7 V
2.15 V	6.2 V
2.25 V	8.45 V
2.5 V	11.3 V
3 V	12.4V

The 555 timer is a crucial part of the buck boost, as it determines the duty cycle needed to switch on and off the BJT to boost the voltage. When changing the potentiometer, we were able to change the duty cycle accordingly in order to increase or decrease the boost.

Figure 20: 555 Timer Max Duty Cycle Output



To ensure that the switch control worked, we made sure that the MCU was able to turn on the switch with any signal above 3V. To test this, we had the signal input directly from the voltage source and varied the signal voltage, while we kept the buck boost output constant. The switch was able to meet this criteria, and had no problems when integrated with the MCU.

Table 6: Buck Boost Switch Data Table

Signal Input (V)	Switch Off Output (V)	Switch On Output (V)
3	8.34	0.015
3.3	8.3	0.02
5	8.38	0.018

4.3. Subsystem Conclusion

Altogether the power system was able to work as intended with some changing of functionality in the end. The most notable changes made from 403 to 404 was the change in AC-DC to DC-DC rectifier although that did not affect the design of the system, as well as the switching from 5V to 8.4V to a 2.2V to 8.4V boost in order to compensate for the voltage drop detected in the system. The system was able to charge the battery from a nominal high wind speed, and was able to switch on and off power to the battery given a signal of 3.3V. Refer to the system report for battery charge data and nominal system results.

Creating STEM Contents: Wind Power

Christian Kirk, Lyndon Li, Samuel Tando

System Report

REVISION – Final
2 December 2023

System Report

for

Creating STEM Contents: Wind Power

Team 12

Approved by:

____
Project Leader Date

____
Prof. Kalafatis Date

____
T/A Date

Change Record

Rev.	Date	Originator	Approvals	Description
1	12/2/2023	Christian Kirk		404 Final Report

Table of Contents

List of Tables

List of Figures

1. Introduction

This document will show how we designed, constructed, and validated the complete, integrated system, and how our full system met all of the requirements that were laid out to us by our sponsor.

2. Integrated System Report

2.1. System Integration

After each of the subsystems had been created and validated in 403, the next step was to integrate the subsystems. Integration of our subsystems consisted of Bluetooth communication between the MCU and the App in both directions, MCU sensors reading the appropriate voltage and currents from the power subsystem, and finally, the maximum charge level and charge control integration to stop charging the battery at the user-specified max charge percent.

2.1.1. App and MCU Integration

In 403, the app and MCU were already able to connect via Bluetooth and simulated sensor data was sent between them. The challenge in 404 was to send the actual sensor data that was read from the sensors of the MCU to the app. Additionally, the app needed to send the max charge level that the user set in the app to the MCU. Both of these were accomplished. The last step for integrating the app and MCU was the plotting page which required plotting the battery charge level sensor readings over time. This was also accomplished by plotting a data point every time the app read data from the MCU which was approximately every three seconds.

2.1.2. MCU and Power Integration

The integration between these subsystems required the MCU sensors to correctly read the voltages and currents from the generator, battery load, and converters. It also required the charging to the battery to be shut off when the battery reaches the user-specified max charge level from the app. All of this was accomplished.

2.1.3. Integrated System Assembly and Enclosure

Once the subsystems were fully integrated, our system needed an enclosure. Christian created and printed a 3D model for the enclosure, and Sam and Lyndon ensured that the PCBs and routing inside the enclosure were fully functional and visually appealing. Our fully integrated system and enclosure can be seen in the figures below.

Figure 1: System enclosure

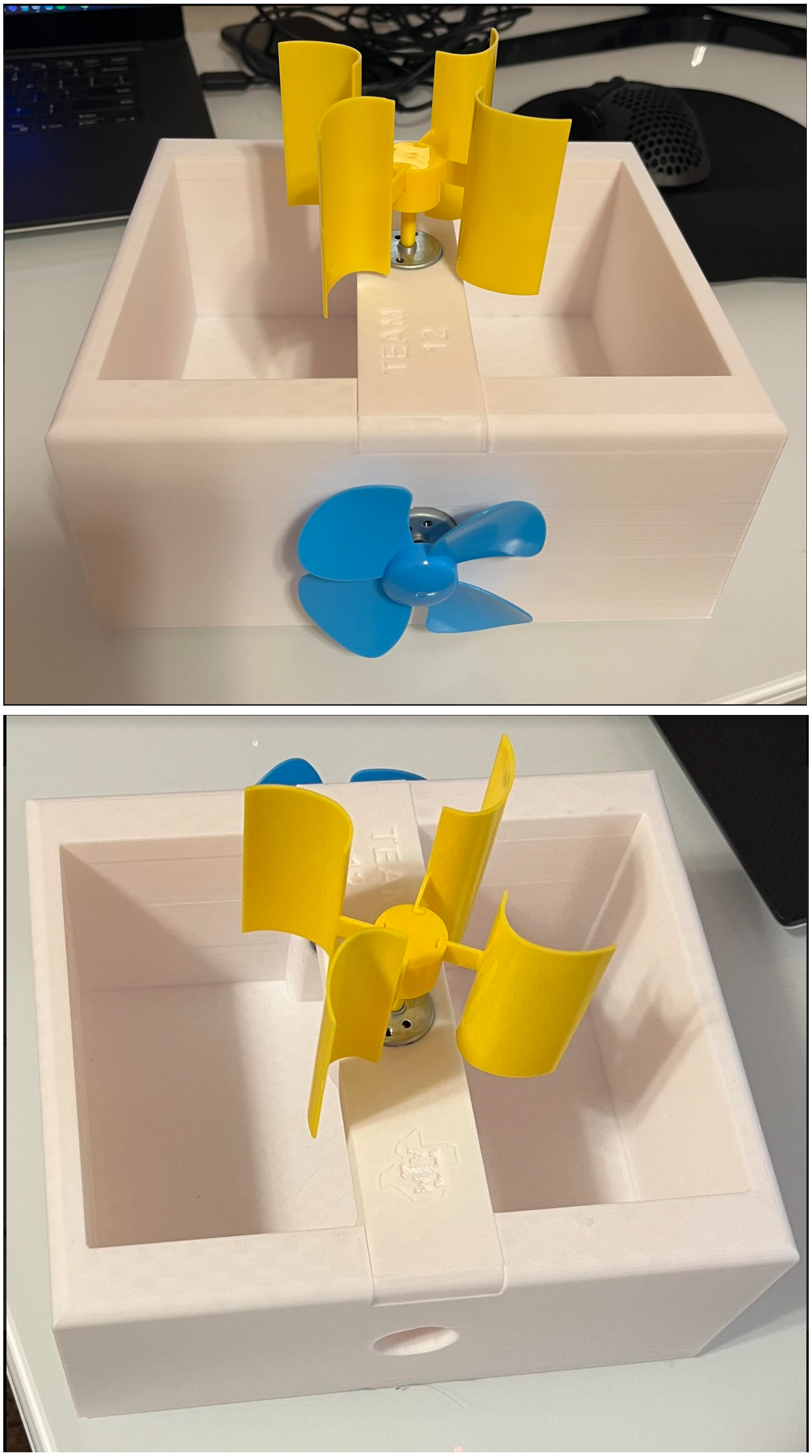
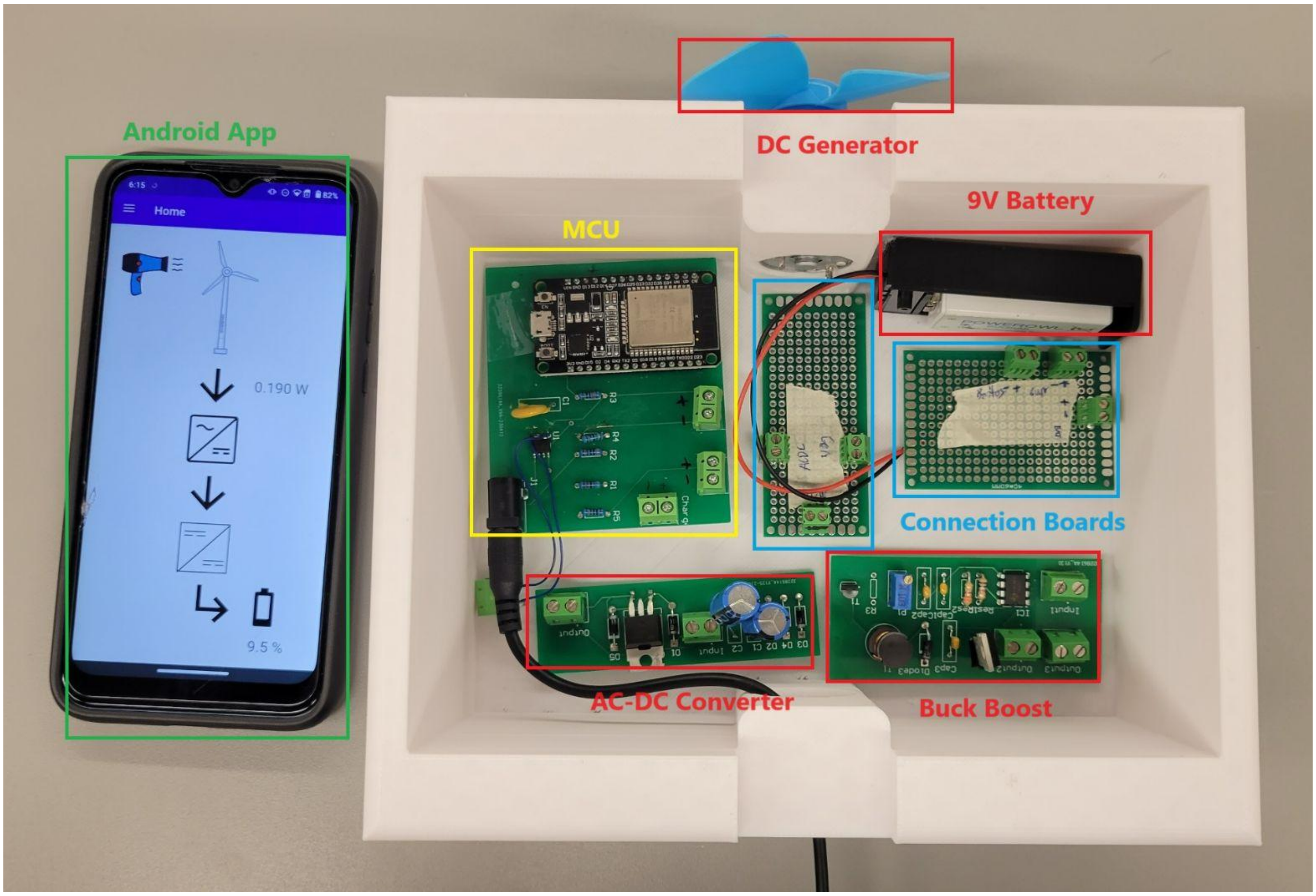


Figure 2: Integrated System in Enclosure



2.2. System Validation

2.2.1. Charge Control Validation

The max battery charge level set by the user in the app was able to turn the charging of the battery off and on as well. This was validated using a multimeter to check if the switch controlling the battery charging was open or closed. A battery that was known to be above the max charge level set by the user was used and the switch was opened, indicating that the charging had been correctly shut off. Then, a battery below the max charge level set by the user was used, and the switch was closed indicating that the battery was charging.

2.2.2. Generator Output/Battery Charge Validation

The Generator output was validated by measuring the voltage and current output at different wind speeds. This was done through using a tachometer in order to relate our reading with a specific rpm (rotations per minute). Throughout our testing, we took measurements of the boost converter output at various wind speeds to make sure the received voltage was around 8 V. This was the case after adjusting the duty cycle of the boost converter through the potentiometer. In order to measure the battery charge, we needed to manually measure the voltage across the battery after several minutes in the system due to the low power output of a small scale wind generator. We verified that after several minutes, the battery was indeed charging. The plot below shows that as wind speed increases so does the rpm of the motor and the output of the generator. Additionally, we have a table showing the charging rate of the battery at max wind speed (1000 rpm).

Figure 3: Generator Output at Various Wind Speeds

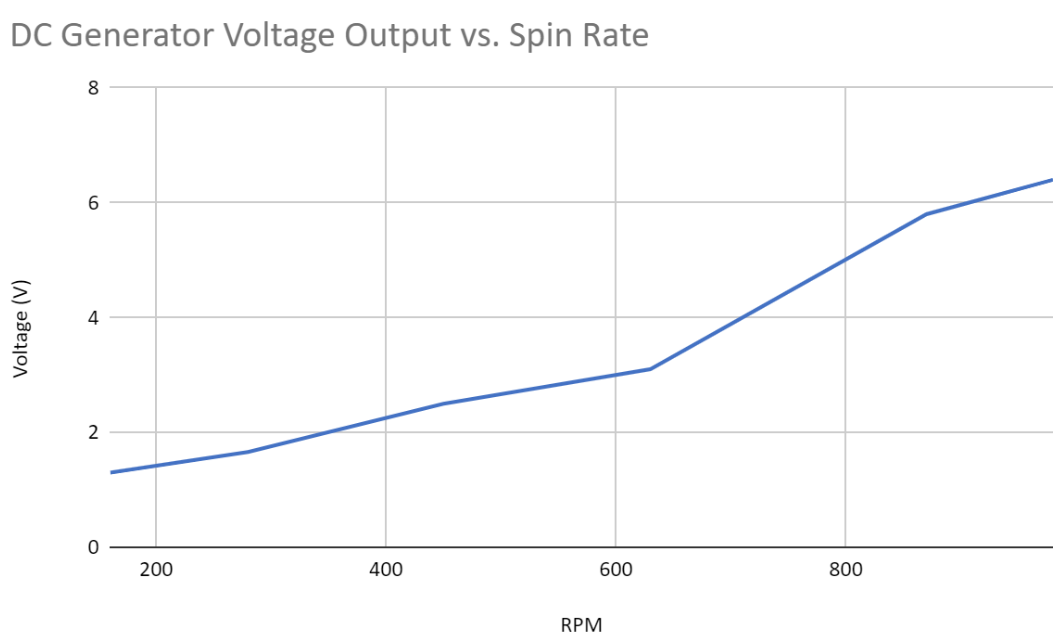


Table 1: Nominal Power System Data

Nominal Power System Test High Wind	Generator Current Out	Generator Voltage Out	Regulator Voltage Out	Buck Boost Voltage Out
Expected	50mA	8V	5V	8.4V
Actual	45 mA	8V	2.2V	8.3V

Table 2: Battery Charging Rate at Max Wind Speed

Battery Charge with high fan setting	Battery Charge after 20 min
7.213 V (85%)	7.311 V (87%)
7.810 V (92%)	7.835 V (93%)

2.2.3. App Display and Plot Validation

The MCU sent the battery charge level and the power output of the generator to the app approximately every three seconds. This was observed using a stopwatch to measure the amount of time it took to generate consecutive points on the plot in the app. Three seconds was about what was expected given that the firmware for the MCU had built-in delays of slightly less than three seconds. The additional milliseconds of delay likely came from delays in the Bluetooth communication which were around 200 milliseconds depending on the distance between the phone and the MCU.

Ultimately, the app successfully displayed the generator power output and the battery charge level in real time on the home page, and the plot showed the battery charge level over time. We could watch on the app when we increased the wind speed on the generator, the power output that was displayed on the app increased, and the battery charge level could be observed as well.

2.2.4. System Enclosure Validation

The enclosure needed to comply with restrictions on volume and weight for portability. It was able to meet the requirements that were set by the sponsor. The final volume was 3060 cubic centimeters and the final weight was well under our requirement of 10 pounds. Two PERF boards were also designed and developed to properly connect the generator and AC-DC to properly enable the voltage and current sensors, as well as a connection board between the buck boost and battery in order to enable the voltage sensor. The enclosure needed to fit all of the seated PCBs nicely and have room for both the horizontal and vertical turbines and both generators to sit snuggly. All of this was accomplished through precise 3D modeling and seating the PCBs such that they were spaced out from each other and elevated.

2.3. System Conclusion

Overall, the fully integrated system was able to take the power output of the generator and use it to safely charge the battery. The app worked as intended for a visual display of the system showing the generator output changing as the wind speed changed, and the plot displaying the battery charging over time. We were also able to control the charge of the battery by stopping the charging once it reached the user-specified max charge level. The system also met the specified portability requirements. Ultimately, the system accomplished the goal we set out for which was creating a portable, small-scale demonstration of wind power.