Project Code: MQP-YM2-1

Wind Surveying Device

Major Qualifying Project Report
Completed in partial fulfilment of the
Bachelor of Science degree
at *Worcester Polytechnic Institute*, Worcester, MA

Submitted by:
Peter Emidy
Christopher Jarrett
Nicholas Pandolfi
Jonathan Redus

Advised by:
Professor Yousef Mahmoud

March 15th, 2019

Abstract

The goal of this project was to create a low-cost device to measure wind speed and direction without an external power supply. Through market research and WPI's energy symposium, a method for obtaining site-specific wind data was found to be desired. By logging wind speed and direction, the device obtains site-specific wind data to help wind-turbine installers analyze the feasibility of a location for wind power production. A working prototype was established with future potential for manufacturability and design integration.

Acknowledgments

This project would not have been possible without the help of several key individuals. First and foremost was Professor Yousef Mahmoud, who advised the group throughout the project. The time he devoted to the project and for his exceptional advice were instrumental in the completion of the project. Additionally, Professor Olinger's assistance from the Aerospace department was crucial for this project, as he provided knowledge on how to test the anemometer and wind turbine in the wind tunnel. His time was incredibly valuable for this project. Finally, none of this could have been achieved without the help of William Appleyard, who assisted by coordinating the purchase of parts for the project.

Table of Contents

Table of Authorship

Section Name	Project Work	Writing
Section 1: Introduction	-	CJ, JR, NP, PE
Section 2: Background	-	CJ, JR, NP, PE
Section 3: Design	-	-
3.1 System Level Design	CJ, JR, NP, PE	CJ, PE
3.2 Power System	CJ, NP	CJ, NP
3.3 Sensors	JR	JR
3.4 Embedded System Design	PE	PE
3.5 Analysis Software	NP, PE	PE
3.6 Final Assembly	CJ, JR, NP	CJ, JR
Section 4: Results	-	-
4.1 Power System	CJ, NP	CJ, NP
4.2 Sensors	JR	JR
4.3 Embedded System	JR, PE	PE
4.4 Analysis Software	NP, PE	PE
4.5 Final System Testing	CJ, JR, NP, PE	JR, PE
Section 5: Conclusions and Recommendations	-	-
5.1 Lasting Impacts of the Device	-	JR, NP
5.2 Future Recommendations	-	JR
5.3 Lessons Learned	-	JR, NP
Works Cited	CJ, JR, NP, PE	NP

List of Figures

List of Tables

Section 1: Introduction

The goal of this project is to create a device that measures wind properties to determine the viability of a location for wind power. The wind surveying device aims to be a lower cost and easier to use alternative to current wind surveying methods. Currently, many wind power companies will construct a tall pole with an anemometer, leaving this device in place for a year to obtain wind pattern data. This method, however, can cost upwards of $1500 plus costs for permanent mounting infrastructure [1]. For large wind companies, this high initial investment for a permanent structure is not too risky, as it is a miniscule investment compared to the large wind turbine fields they install. For smaller companies, this large initial investment just to survey a prospective location for wind power that may not even turn out to be a suitable location can be too costly and risky of an initial investment. For this reason, the proposed device for this project could open the door for smaller wind power companies that could install small to mid-sized wind turbines in areas where large wind turbines are not suitable such as residential areas and farmland.

One reason why the residential wind energy market has not seen the success of residential solar might be because solar companies can easily predict profitability, while wind companies cannot. Solar companies can predict the annual energy production of varying sized solar arrays based on latitude, longitude, and pitch of the panels that would be installed. There are different softwares that can simulate annual energy production based on the size of the installed system and imported weather data such as PVSyst and Helioscope [2]. This allows companies to predict how much money could be saved by installing solar panels and how long it would take for customers to receive full return on their investment. In order for wind power to be effective on a small scale, there must be a simple way to obtain site specific wind data for a low cost. This would allow consumers to see how much money they could save with wind power. This project aims to design a system that can gather all of this data and streamline the process for companies and individuals who are looking to install a smaller wind power system.

1.1 Features and Prospective Usage

The device has features that make it easy to use in the real world. A small wind power company could take the device and put it in place for a few months out of the year to survey a specific location for its wind patterns. The device is cheap and reusable, unlike a costly permanent device, thus minimizing the risk of surveying different locations for their wind turbine potential.

A company should be able to leave this surveying device at a prospective location and not have to worry about connecting it to an external power supply, especially in remote areas. This means that while the device is obtaining wind data, it will also be harnessing energy and storing it in rechargeable batteries for times of low energy supply where the device is still sampling data but is producing less power than it is consuming. It will also require a charge controlling circuit to prevent over charging and over discharging of the batteries and voltage regulation of the generated power. Optimization of the data processing will also to crucial to ensure minimum power consumption within the device.

Section 2: Background

2.1 The Existing Renewable Energy Market

Renewable energy is energy that comes from sources that are naturally replenishing, therefore the supply is limitless. However, most forms of renewable energy are not consistent in their availability. Renewable energy sources include, solar, wind, and hydroelectric. In the United States, renewable energy sources make up 11% of the total energy production as of 2017. Of the renewable sector, solar makes up 6%, wind makes up 21%, hydroelectric makes 25%. Since 2001, the renewable sector has increased substantially [3].

2.1.1 The Current Solar Market

The current solar power market is very well developed and reliable in prediction of return on investment. The range of solar products available cover almost all needs. From large arrays of low-cost, sun-tracking panels to provide power to the grid, to high efficiency micro-panels for use in handheld devices, solar is a very versatile renewable energy source [4]. Solar power's present-day strength is partially due to the ease of predictability of the source of power - the sunlight. The motion of the sun in the sky is relatively simple, and software can easily simulate expected returns when using solar devices using parameters such as the pitch of the solar panels, size of the array, and imported weather patterns for that specific location. Compared to the wind market, the solar market is significantly more developed in its technology and more versatile in the current solutions existing on the market [5].

2.1.2 The Current Wind Market

The current wind market focuses on large scale power installations. This is because the cost of wind energy depends on installation and as turbine size increases, fewer turbines have to be installed to generate the same amount of power. This has resulted in the growth of large offshore wind farms, as well as the design and creation of larger scale turbines in the 1MW to 12MW range. Offshore wind farms have the additional benefit of more predictable wind speeds at a regular direction [6]. The growth of the midscale wind turbine industry has dropped dramatically largely due to stigma around their appearance and the inability to produce as much power as offshore wind. Additionally, viable land space for turbines is dwindling, and in some instances, the government has imposed building and land use regulations which restrict wind farms.

Currently, there are some companies that survey a landscape and then run wind simulations for that surveyed area. These simulations, however, do not have wind data information, but instead are for finding ideal places for a wind turbine out of a selected region assuming the area is already chosen for a wind turbine. The device could be used in conjunction with this process, giving accurate wind data at various data points, thus enabling a much more accurate wind simulation.

2.1.3 Potential Opportunities in the Current Renewable Energy Market

There is still untapped potential in the renewable energy market. In addition to the potential growth of other forms of renewable energy such as geothermal and biomass waste, the renewable energy sector has not yet moved into small-scale or residential scale wind power options.

2.2 Residential Wind Power

Residential wind turbines are turbines designed to be mounted on or around buildings rather than on freestanding structures. Though it might seem that these could run into the same issues as current land turbines, it has been shown that these issues can be worked around. The first issue with land turbines is the idea that they are an eyesore. However, since these turbines are smaller, they do not stand out to nearly the same extent as a large turbine. Additionally, smaller turbines can be worked into the design of buildings and used as aesthetic accents, acting as the opposite of the eyesore people fear they will be [7].

Another issue that land based turbines face is limited land availability [8]. Residential scaled turbines increase the number of locations for turbines to be installed, including cities. At the residential scale, wind turbines also need to compete with solar panels. While solar panels are simpler and a proven technology, there are a few places that wind turbines could be more practical. The first is in cities, where taller buildings can frequently block sunlight from reaching the roofs of shorter buildings, but wind can easily move between them. The second is on farms, where small wind turbines could generate power without taking up as much land for farming, while solar power removes more of a farmer’s land for growing crops. Sometimes if houses do not have a roof facing optimal direction for solar energy production, typically south, or if their roof is not at a pitch close to the degree latitude of which the house is located, solar may not be an effective renewable energy source at that particular location. Thus, residential wind power has several logical applications, and effort can be taken to overcome the limitations of current land-based wind power.

2.2.1 Issues with Residential Wind

Residential wind power has its own set of issues. The most critical issue is that it is not a widely used technology. This means that there are not many people working on this solution, that the solution is not widely considered, and that there is not a way to determine if wind power is a good option. When searching for wind power online, most sources discuss large scale wind companies and installations. Sources on smaller wind turbines are much harder to find. This is further evidence that there are not many companies working in the wind power market outside of large-scale wind turbines.

The last point, that there is no way to evaluate residential wind power options, is what this project aims to address. Through the creation of a residential wind surveying device, people will be able to make informed decisions about their options.

2.2.2 Existing Solutions to Wind Surveying

There are two general solutions to surveying the wind in a potential area to be utilized for wind development; direct measurement, and indirect measurement. To predict the general characteristics of the wind in a particular location, annual global wind patterns can be used. Prediction in such a method will only yield useful data at high altitudes, and will only produce very broad averages for wind speed and direction. When turbulence from land features is negligible at said high altitudes, then prediction using global patterns will yield usable results. However, there are generally large tolerances, and certainty is not high. Consumers want empirical data of the wind’s behavior to be confident enough to invest in high-cost wind turbines with long breakeven times. To obtain empirical data, the wind’s characteristics can be measured directly using an anemometer and vane type design, usually mounted on a high-altitude reinforced pole. Doing so requires a full system design and fully self-sustainable power source. Existing solutions to provide anemometer and vane type measurement usually do not provide their own power, and external assemblies need to be set up to provide sufficient power, usually by means of solar panel, or from the grid [9].

In contrast with the direct measurement of an anemometer and a wind vane, is the indirect measurement from a LiDAR system. Although expensive, LiDAR can measure wind speed and direction at almost any desired altitude. It does so by using lasers to detect and track small particulate naturally present in the air. It can measure wind characteristics hundreds of meters away from the device, and thus can survey an entire area at once instead of just a single point. LiDAR is a form of indirect measurement, and has many associated problems. First and foremost, the price of the units makes them only sensible for application in the largest of wind turbine companies with the biggest tolerance for overhead payment. Secondly, the unit must be stationary and isolated from mechanical movement of any sort. Third, the technology is still developing and accuracy of measurements is not as reliable as that of an anemometer. This combination of factors means that a LiDAR system is only useful for surveying for a full-scale wind farm installation [10].

2.2.3 Proposed Solution

As previously mentioned, a huge remaining issue in the wind power market is a lack of site-specific data for residential areas and other areas that could pose as feasible locations for a smaller wind turbine. Not only is this an issue for locations evaluated for large turbines as well, but is an especially prevalent issue for locations being evaluated for smaller to mid-scale wind turbines [11]. This project aims to tackle this issue by providing site-specific data through wind speed readings, wind direction readings, and a running log of this data that is processed to produce visual aids easily interpreted by users. By providing this site-specific data, companies will be able to more accurately analyze a certain site's feasibility for a wind turbine.

Section 3: Design

3.1 System Level Design

The solution that has been designed on this project has many subsections, electrical and mechanical, that work together to make a final functional product. To simplify the overall structure of the project, both sections will be discussed first at a top level.

3.1.1 Top Level Electrical System Design

The analysis software is the component of the project that takes the raw data collected by the device and turns it into something that can be understood by a user. It does all of the calculations to determine what kind of energy output various wind turbines would get in the area. It needs to be able to read the data collected from the microSD card, perform calculations on it, and graph it in an easy to understand format.

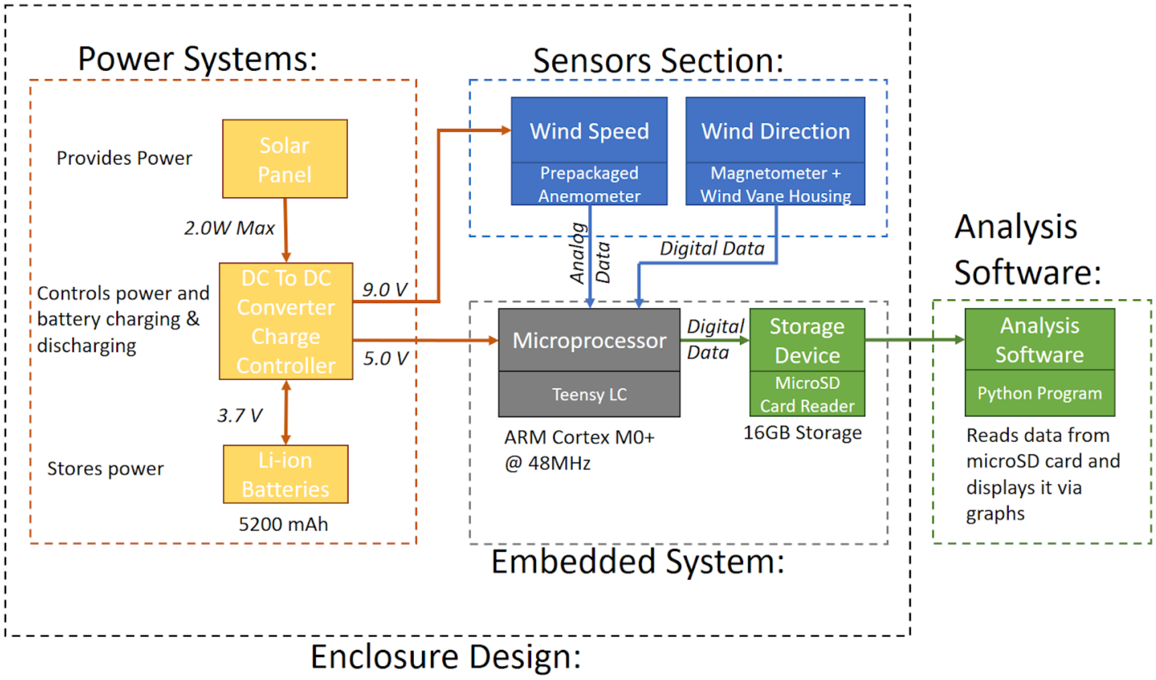

Figure 1: Electrical System Level Block Diagram

3.1.2 Top Level Mechanical System Design

The design requirements for the mechanical system were fairly straightforward. First, it had to be both watertight and sturdy, since the device was meant to be used outdoors in real weather. The wind vane and anemometer needed to be spaced apart from each other, so that they would not interfere with the readings, and the solar panel needed to be in an exposed location, to receive sunlight. For the anemometer, parts needed to be made of plastic so that the magnetic field would not be affected by the enclosure. Finally, the device needed a way to be mounted.

Thus, it was decided that the housing should have two arms coming out of a central T, then turning upwards, with the sensors mounted on top of them. The solar panel could then be mounted on the center of the T. It was also decided that the majority of the system should be made of ¾ inch EMT, since it is rated to protect electronics from inclement weather. The boards were then mounted in a plastic enclosure, with the magnetometer on top.

3.2 Power System

To implement a self-sufficient product to be used out in the field, the design of the power supply was one of the fundamental components tackled first. The final device would have to provide enough power for itself, and be able to store enough power for the most extreme power outages. The product’s ability to function without external power is one of the biggest selling points. Other similar devices on the market do not have this ‘set and forget’ capability, essentially limiting their practical usage cases.

3.2.1 Source of Power

Two options were considered for sourcing power to the device: solar and wind. Solar would be the easiest solution, and offers the most support due to its commonality on many types of small self-powering devices, meaning there are already integrated solutions for power management from solar cells. Solar was not initially considered in the proposal for the final design due to perceived marketing issues. On first glance, the buyer may not trust that the product would prove the viability of wind in a residential area while the device itself uses solar power. Upon further consideration from the team, this problem does not actually apply due to the vast difference in scope of wind power needed to power a tiny device as such versus the scope of a residential wind turbine. It can also be risky to use a wind turbine to power the device if the feasibility of a site is being evaluated for wind power, not yet knowing if a wind turbine actually has potential. However, when the buyer is evaluating the final device, they will most likely not pause and ponder these considerations. For these reasons, it was initially chosen to use a small-scale wind turbine.

3.2.2 Power Management Calculations

Power Consumption:

The table below shows the main components currently in the design of the device, and the forecasted worst-case energy consumption. These numbers represent a prediction, and are just rough estimates. More accurate current consumptions will be attained when empirical tests are performed for each component.

Table 1: Maximum Power Consumption for Each Component

Device	**Worst Case Current (mA)**	**Worst Case Power (mW)**
Teensy LC Microprocessor	20 mA @ 5V	100
SD Card Reader	5 mA @ 5V	25
Magnetometer	1 mA @ 5V	5
Anemometer	20 mA @ 9V	180
Total Power Consumption:		310

Battery Considerations:

Each 18650 series (lithium ion) battery chosen for this device has roughly 2800 mAh of capacity at the nominal 3.7V voltage. Two of these batteries connected in parallel were chosen to give a total of 20.71 watt-hours of energy storage. With 20.71 watt-hours, and an average power consumption of 0.230 watts, this would mean that the device could power itself for approximately 90 hours, or 3.75 days on a full charge of battery and no supplied power.

3.2.3 Circuit Topology and Design for Charging Circuitry

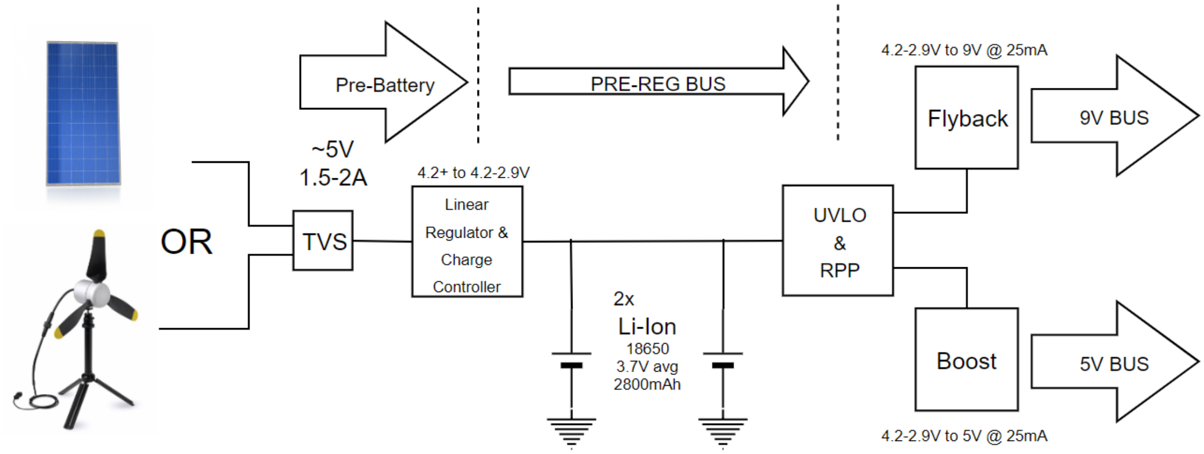

Figure 2: Topology of Power Flow and Conversion for the Device.

The final circuit design is seen above in Figure 2 and described as follows. The power conversion circuitry can use either the solar panel or the turbine for a source of energy. This adds some complexity, but is easy to implement using a low-dropout (LDO) regulator with a high (~16V) upper voltage limit to protect against the open circuit voltage of the solar panel at ~8.5V. The output voltage from the chosen power source must then be regulated to levels applicable for the standard lithium-ion batteries that will be used. This gives a range of 4.2V to 2.9V. Power then branches off as necessary; split between the batteries and the regulators of the next stage. If excess power is produced from the source, and does not need to flow to the output, it will flow into the batteries. If the batteries are charged to 4.2V, the charging circuitry will shut off, until a drop to 3.7V is sensed, for hysteresis. Conversely, if no power is being produced but power is required, the batteries simply feed directly into the converters as needed.

Separating the batteries and the converters is an under-voltage-lock-out (UVLO) stage, as well as a reverse-polarity-protection (RPP) stage. The UVLO prevents the batteries from discharging past 2.9V to preserve their life and to prevent unexpected behavior of the switch-mode IC’s if their input voltage drops below the usable threshold. Once the battery voltage reaches 2.9V, the UVLO stage simply cuts off the batteries from the rest of the circuitry, until they are charged to 3.1V by the power source. The RPP prevents damage to the subsequent converter circuitry if the battery is inserted backwards on accident.

As seen in the diagram above, two voltages will be produced to support the rest of the device. The 9V rail is necessary for only the anemometer, and the 5V for everything else. The 5V rail requires a step-up topology, along with the 9V rail, but the 5V rail is relatively low stress. The lithium ion battery voltage in worst case scenario is around 3V, so stepping up to 5V is not a large demand. The duty cycle of the converter topology is probably the most important thing to look at in applications like these. A very high duty cycle (above 0.5), can lead to subharmonic instability in these current-mode-control (CMC) switcher ICs. The boost topology was therefore determined to be adequate to produce the 5V required at around 25mA worst case scenario.

The 9V rail is much higher in terms of stresses on the components, as is the control loop - if a boost topology were to be used here as well. The obvious choice was to use a flyback based topology. If one were to use a boost converter for this ~3V to 9V step-up, the duty cycle would easily exceed 0.5, and stability would not be guaranteed unless one took an in-depth look at the control loop. To invest in the future of this product, and to make it quite impervious to changes to passive components, the duty cycle would have to be brought down to around 0.4. This can easily be done with a flyback transformer with a 1:3 turns ratio. Coilcraft has many options and one of their products allowed us to make a very dense flyback converter with great transient response and stability. The turn ratio will naturally offset the duty cycle. The higher the turn ratio, the smaller the necessary duty cycle to be able to provide the 9V output.

3.3 Sensors

The sensors are the part of the system that measures physical characteristics. In this project, the two sensors needed were an anemometer, which measures wind speed, and a wind direction sensor.

3.3.1 Considerations for Anemometer

An anemometer is a device that measures wind speed. There are three main kinds of anemometers: cup anemometers, vane anemometers, and hot wire anemometers. Vane anemometers have a turbine on the front of a body with a fin on the back. The fin makes the anemometer turn to face into the wind, so that the turbine faces into headwind. Cup anemometers are simpler, consisting of several hemispherical cups rotating around a central axis. This design allows the anemometer to measure lower wind speeds than the vane anemometer, but means that the anemometer does not measure the wind direction. It measures lower wind speeds because the slightest wind is measured by the anemometer, since it does not have to be turned by the wind before it can measure the wind. The last kind of anemometer, the hot wire anemometer, measures how much a wire cools, and is more commonly used in handheld applications [12]. It was decided to use a cup anemometer, because it can measure lower wind speeds and is the simplest design.

There were a few other desired features for the anemometer. The desired output voltage range was between 0V and 2.56V. This was because the microprocessor has a built in ADC, with a range of 0V to 2.56V, and if the anemometer matched this there would be no need for supporting circuitry. Additionally, it was desired that the anemometer to accept a 5V voltage source, so that it could operate off of the same power supply as the microprocessor.

3.3.2 Considerations for a Wind Direction Sensor

A wind direction sensor is a device that measures the direction of the wind. They operate by measuring the rotation of a vane, which is pushed around by the wind. There was the option to buy a wind direction sensor but decided to originally create one, since they are fairly simple and cost less to build than to buy.

The core of a wind direction sensor is the mechanism to detect the angular position of the vane. The main kinds of sensors considered for this project were rotation sensors and magnetic direction sensors. The rotation sensors can be further broken into Hall Effect sensors incremental encoders, and absolute encoders. Hall Effect sensors are a kind of magnetometer which only measure the strength of a magnetic field. To use them to measure angular position, a magnet is placed above the Hall Effect sensor and a metal plate with gaps is put between the magnet and the Hall Effect sensor, and the plate is connected to the axis of rotation. When the object rotates the plate turns and the magnetic field at the Hall Effect sensor is changed, since if there is a plate there, it shields the Hall Effect sensor from the field, but if there is a gap above the Hall Effect sensor it is not shielded and senses the full magnetic field. This means that the Hall Effect sensor can measure the rate of rotation of an object, but not the position of the object. Incremental encoders also measure the rate of rotation of an object, by outputting a square wave which changes from high to low every *n* degrees (where *n* depends on the specifications of the product). Absolute encoders, however, measure the position of an object. They output a unique code at every position that the sensor measures.

3.4 Embedded System Design

The embedded system is at the center of the project, since it connects all of the various subsections into one unified system. The embedded system is powered by the power delivery system so it can read data from the various sensors and record it, so that it can be interpreted by the analysis software. This system is able to do all these operations via a microprocessor which can turn the input signals into data that can be stored and later interpreted, so that it can be stored and later interpreted, so that it can be understood by the users.

3.4.1 Embedded Systems Considerations

The embedded system is the central element of the wind sensing device. It is responsible for reading the data generated by the sensors and storing the data such that it can be processed by the analysis software. The embedded system requires a constant energy source to power a central microprocessor which performs the data processing. While searching for an embedded system, the following requirements had to be met:

1. Enough inputs for the sensors
2. Low power consumption
3. Can write data to be processed on another computer

4. Easy to prototype designs and program

The first requirement was for the embedded system to have enough inputs for the sensors. This is important because if the system cannot connect to all of the sensors, the device will not be able to perform its duty as a wind surveying device. Therefore, before choosing a particular embedded system, the total number of inputs for the sensors needed to be determined. The two main sensors used on the device were a wind speed sensor and a wind direction sensor. The wind speed sensor was implemented using an anemometer, which requires one analog input. The wind direction sensor was changed in its implementation throughout the project. Originally the wind direction sensor was going to be implanted using an 8-bit absolute encoder, which used 8 digital inputs. However, initial testing of the absolute encoder proved it would not work for the project, so the wind direction sensor was changed to be a magnetometer. The magnetometer chosen requires six total ports for digital communications. Additionally, the embedded system needs inputs to write the data to a storage device. A microSD card was chosen to store the data, and the writer for cards require 4 digital communication ports. The input requirements for the various input configurations is summarized by the following table:

Table 2: IO Ports for the Microcontroller System

	Anemometer	**Wind Direction Sensor**	**MicroSD Card**	**Total**
Original Config	1 Analog Port	8 Digital Ports	4 Digital Ports	13 Ports
Modified Config	1 Analog Port	6 Digital Ports	4 Digital Ports	11 Ports

The second requirement for the embedded system was for it to be low power. By making the embedded system as low power as possible, it would make it easier for the device to be self-sustained using a small solar panel as a power source. However, because the solar panel does not produce energy throughout the entire day, the device will need to be powered using a battery solution overnight. With a low power embedded system, the battery solution can be smaller as it requires less power to run overnight.

The third requirement was for the device to be able to write the data it collects to an external device. This is so that the difficult computations can be done on a separate computer, which reduces the complexity of the microprocessor. If the embedded system is just reading and storing the data, then it does not need to do complex operations on the data which would require a more powerful microprocessor.

The fourth requirement of the embedded system was for it to be easy to prototype designs and programs. This included having a development board, an easy to use IDE/compiler, and thorough documentation for both the development board and the functions of the IDE. The development board is important because it allows for quick prototyping without having to design a board around a microprocessor. An easy to use IDE is important because programming in a higher-level language is quicker and easier than learning a low-level language specific to a microprocessor.

3.5 Analysis Software

The analysis software is where the data collected by the device is processed into a user readable format. The software starts with the data file generated by the embedded system and is able to interpret it to perform more calculations on the data. Some of the calculations include determining the wind speed and wind direction from the anemometer and magnetometer. The software is also responsible for producing graphs for visualizing the data.

3.5.1 Analysis Software Considerations

The analysis software is responsible for representing the raw data recorded by the device. This allows the end user to look at synthesized, graphical data rather than a raw text file. For the software to be successful, it needs to provide helpful visualizations of the data. Therefore, when designing the analysis software, the team thought of methods to group and organize data points, while choosing the optimal language to write the software.

The wind surveying device records data for the wind speed and wind direction over time. These data points are what the software has to work with when producing charts and graphs for the user. Additionally, the objectives of the project should be considered when determining what data to show to the user. The following data and graphs have been determined as useful representations of the data:

- Wind Speed vs Time
- Average Wind Speed in a Direction
- Percent of Time Spent in a Direction
- Power Generation from an Ideal Turbine

The “Wind Speed vs Time” is important as it provides an easy way to see how the wind speed varies through the day(s) during the study which allows the user to get a general idea of how windy the location is. The “Average Wind Speed in a Direction” graph allows the user to see how strong the wind is from varying directions. The “Percent of Time Spent in a Direction” graph lets the user see which direction the wind predominantly originated from. The “Power Generation from an Ideal Turbine” is important because it gives a number for how much power a specific turbine would produce if installed in the testing location.

With the goals of the analysis software considered, the next step was to determine the considerations building for the software. In order to make a program with the specifications above, the following properties had to be met:

- Good string operations
- Good numerical arrays with operations on elements

- Good graphing library for representing data

First, having good string operations is important because the data is loaded from a text file, so being able to turn the text data into usable numbers is important. Next, having good array structures to store and modify data in is important for converting the raw data into readable data and doing further mathematics on it. Finally, a graphing library is important for creating graphs that properly visualize the data.

3.6 Final System Construction

3.6.1 Final System Enclosure

The overall design of the device can be seen in the image below:

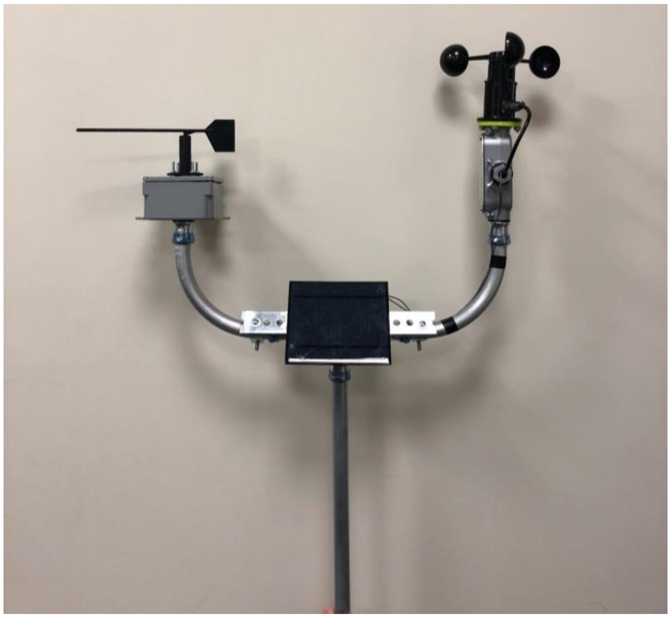

Figure 3: Overall Device Design

The device has a central pole that can be used for mounting. This device was constructed using ¾" electrical EMT conduit for a sturdy and durable design. The piping is connected using rain-tight connectors to prevent water from entering inside the device and piping that could corrode the wiring. As shown in the image, two 90° pipe bends branch off from the central mounting pole. These two pipe bends and the central mounting pole are connected with a rain-tight T conduit connector. A wind vane is mounted on one pole bend and an anemometer is mounted on the other pole bend. A solar panel is mounted on a bracket in between the two pole bends and above the rain-tight T conduit connector as seen in the image below.

Figure 4: Mounted Solar Panel

This bracket was designed to allow the solar panel to be mounted on the device by allowing the solar panel to clear the rain-tight T conduit connector below without interference. This bracket also allows the user to change the pitch of the solar panel by rotating the mounting bracket.

The wind vane is mounted on top of a plastic enclosure as seen in the image below.

Figure 5: Wind Vane Structure and Enclosure

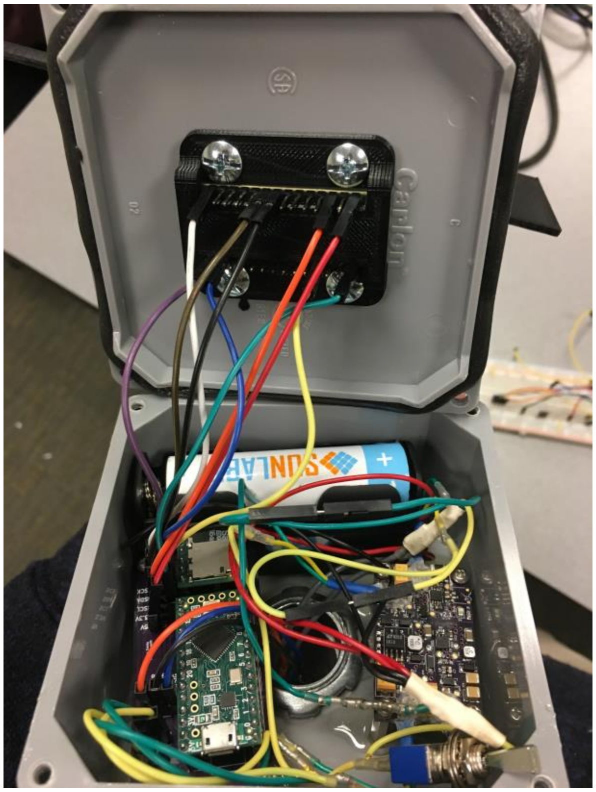

Figure 6: Mounted Electrical System Components

A plastic enclosure was chosen in order for the enclosure to not interfere with the changing magnetic field from the rotating magnet within the wind vane assembly. This plastic enclosure is also waterproof to prevent any water from damaging the microprocessor, SD card reader/writer, charge controller, magnetometer, and batteries all housed inside. The wind vane assembly is bolted to the cover of the enclosure with the magnetometer bolted directly underneath on the inside of the cover. Silicone around the bolts was used to ensure waterproofing.

For the mounting of the anemometer, silicone was also used to ensure waterproofing as seen in the image below.

Figure 7: Mounted Anemometer

There is a silicone bead placed between the 3D printed green adapter plate for the anemometer and the rain-tight T conduit connecter to prevent any water from entering inside the piping. The rubber rain-tight connector can also be seen where the wires from the solar panel and anemometer enter the conduit piping. This rubber rain-tight connector uses the rubber insert to tighten down on the wires, creating a seal that prevents any water from entering the piping.

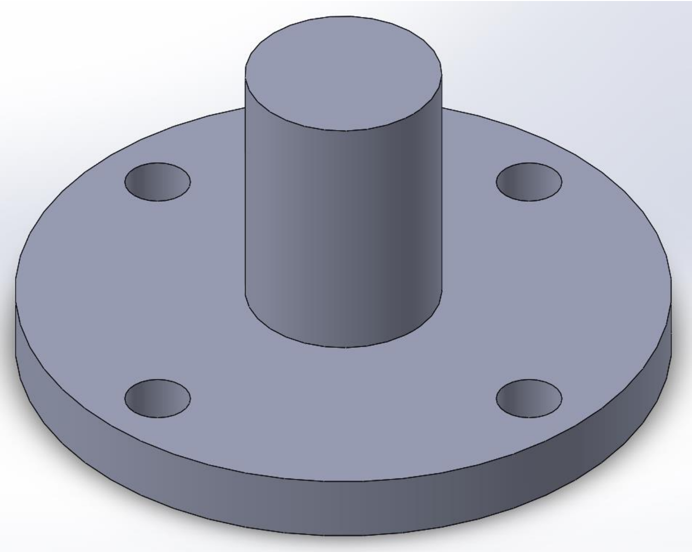

Figure 8: 3D Model of the Adapter Plate

3.6.2 Additional Components

In addition to the housing for the components, the wind vane had to be designed and constructed. The design was done in SolidWorks, and components were either 3D printed or bought. Two bearings were purchased, since they are needed to make the wind vane rotate with little resistance. A neodymium magnet was used so that the magnetic field would be stronger and easily detectable by the sensor. A housing was then designed for the bearings to sit in and a wind vane:

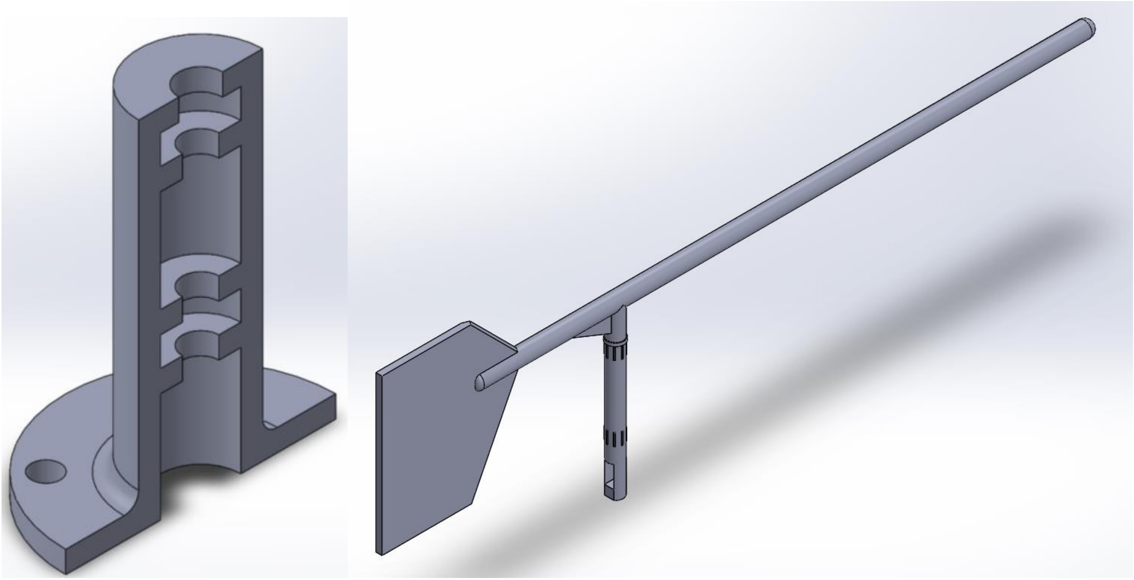

Figure 9: 3D Models of the Bearing Housing (left) and Wind Vane (right)

The wind vane has a large fin on one end to be pushed by the wind, and a long counterbalance on the other end. The counterbalance ensures there are not extra radial forces put on the system, and its slim profile allows for minimal air resistance. The bottom post has two bearings, to ensure that the vane remains steady even in extreme weather conditions. On the bottom of the post, there is an adapter to hold the magnet to the wind vane, so that it rotates with the rest of the wind vane. This lead to the final design:

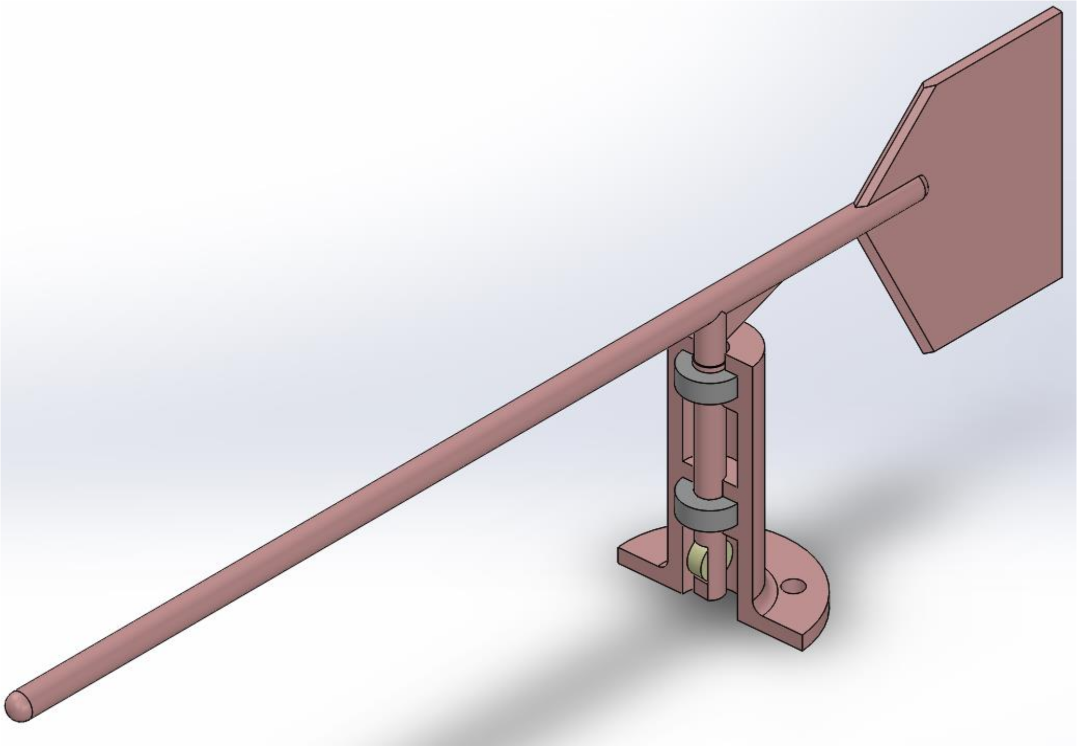

Figure 10: Wind Vane Assembly - Bearings shown in Grey, Magnet in Yellow, and 3D Printed Parts shown in Red

There were a few iterations of the wind vane structure. In the first print, there were no tolerances added, so nothing fit into the place it was designed to. The next iteration had 1mm to 2mm of tolerance added around all measurements which fit significantly better. The initial design of the wind vane also had the magnet holder integrated into it. However, this led to the plastic around the holder being too thin to support the forces needed to put the bearings into place. Thus, the design was changed to the final form, where the post connects to a separate magnet holder.

Section 4: Results

4.1 Power System

4.1.1 Wind Turbine Selection for Prototype

There were a number of options for a miniature wind turbine to supply power to the system. Building a wind turbine or buying one of several existing small-scale turbines were two potential options.

For building a small-scale wind turbine, there were design options similar to the one shown below.

Figure 11: First Do-It-Yourself Mini Wind Turbine Example

This design would cost about $68. A number of issues were found with this design. First, the DC motor used in this design from Pacific Sky Power was listed as a 15W potential source, but many users reviewed the motor to only produce less than 1A at 12V even at wind speeds as high as 60mph. Additionally, the blades in this design were reviewed to only last about 3 months, which would not be sufficient for the desired lifespan of the device. Lastly, this design would not be waterproof, so a waterproof enclosure would have to be designed and tested. For these reasons, the search continued to look into different designs [13].

As mentioned above, the first design found had a number of issues that prevented us from using it for the device. It was decided to search for alternative turbines that addressed the issues discovered in the first design. The next design investigated is shown below.

Figure 12: Second Do-It-Yourself Mini Wind Turbine Example

This design would cost about $63. This design was an improvement from the first example mentioned, but still had some issues. It has a 300W-pulse capable TVS diode that protects the USB device with a regulated 5V USB terminal output. The issue with this design is that the enclosure is still not waterproof and the DC motor would get damaged from moisture without protection. Reviewers highlighted the same issue with blades as the previous example, where the blades did not typically last longer than 3 months. There were also reviews of the blades not being able to withstand more than 40mph winds. With the reoccurring issue of faulty blades and vulnerability to water damage, further investigation was conducted on pre-built mini turbines [14].

As mentioned before, research was also made into buying a premade turbine. The first prebuilt turbine found was from VEVOR, and is shown below:

Figure 13: Pre-Built Mini Wind Turbine by VEVOR

The cost of this pre-built mini wind turbine is $153. Initially, this design seemed to be a promising option. The three-phase AC permanent magnet synchronous generator is rated to produce 100W at 12V, supplying more than enough power to the microcontroller and sensors. The blades have a low cut-in speed of 1.5m/s (3.35mph) and a high maximum wind speed rating of being able to withstand wind speeds of up to 45m/s (100.7mph). The issue with this design, however, is the blade sweep diameter is 35.4 inches, which is far too large for the final device. A turbine of this size would pose many challenges for mounting and could cause wind disturbance that could affect data integrity if it was too close to the anemometer or wind direction sensor [15].

Continuing the search for a pre-designed wind turbine, the wind turbine shown below from SHZOND was found:

Figure 14: Pre-built Mini Wind Turbine by SHZOND

The cost of this turbine is $169. Once again at first glance, this wind turbine seemed to be a promising potential option with its sleek design. Its rated output power of 400W at 24V-57V DC would supply more than enough power to the microcontroller and sensors and the blades have a low cut-in speed to start producing power at 2.5m/s (5.6mph). Once again, however, the size was an issue. The blade sweep diameter is 3.9ft, which is too large for the final design, would be difficult to mount with its size, and could cause too much wind disturbance for the anemometer and wind direction sensor if not mounted far enough away. Thus, the search continued [16].

The last wind turbine that examined, from TEX Energy, is shown below:

Figure 15: TEX:Energy Infinite Air Wind Turbine

This wind turbine costs $130. With its sleek design, the TEX:Energy Infinite Air wind turbine weighs only 626 grams (1.38lbs), generates 7.5-10W at a regulated 5V output, has a blade sweep diameter of only 322mm (12.6 inches), and is IP65 waterproof rated. The definition of the IP65 waterproof rating can be seen below.

Table 3: IP65 Waterproof Rating

IP Rating	Protection	Description	Test Method
IP65 Enclosures	Able to protect against water jets	Water projected by a (6.3 mm) against enclosure from any direction shall have no harmful effects.	Test Duration: At least 15 minutes Water Volume 12.5 litres per minute Pressure: 30 kPa at a distance of 3m

With its small blade sweep diameter, 5V regulated output voltage, and IP65 waterproof design, the TEX:Energy wind turbine appeared to be the optimal wind turbine to power the final device, and is the wind turbine purchased for the project. At $130, it is only about $50 more than if building a wind turbine similar to the options mentioned before, and it already has a much nicer waterproof enclosure design. If building a wind turbine, there would be an additional cost of designing and building a waterproof enclosure, rendering the TEX:Energy turbine to be the more cost effective solution. This wind turbine is also constructed with a universal mount, allowing for many different tripods, clips, and other mounting hardware to be attached. This allows for an easier installation process of mounting the turbine.

The TEX: Energy wind turbine is rated to provide approximately 5 watts at a wind speed of 15 mph. This is treated as an average power generation rate to simplify the preliminary calculations. The next step is to determine the average wind speed in Worcester, Massachusetts. Determined through several sources, a benchmark of about 5 mph average seemed adequate.

Treating the wind turbine as linear with no cut-in speed, on average, it would produce about 1.5 Watts continuously. With the power consumption of 0.230 W average, this would essentially result in 1.27 Watts surplus left over from the wind turbine, which is a large margin and very good for preliminary calculations [17].

4.1.2 Solar Panel Selection for Prototype

In selecting a solar panel for the prototype, the desired power output of the solar panel needed to be determined. To calculate this, the following values were used, along with the max power consumption of 310mW from Table 1:

Step 1: $310mW\ max\ power\ consumption$

Step 2: $4\ hours\ energy\ production\ per\ day\ from\ solar\ panel$

Step 3: $4\ hours/24\ hours\ in\ a\ day\ = 1/6\ of\ the\ day\ for\ energy\ production$

Step 4: $310mW * 6 = 1.86W$

Step 5: $Necessary\ wattage\ of\ at\ least\ 1.86\ W$

The first option for a solar panel to power the device was the 2 Watt Seeed Technology Co. solar panel pictured below along with its specifications.

Figure 16: 2W Seeed Technology Co. Solar Panel

Table 4: 2W Seeed Technology Co. Solar Panel Specifications

Power (W) - Max	2W
Current @ Pmpp	360mA
Voltage @ Pmpp	5.5V
Voltage - Open Circuit	8.2V
Package/Case	Cells
Size/Dimension	180.00mm x 80.00mm x 2.50mm

The size of this 2W panel is large enough to meet the desired minimum solar panel power of 1.86W with a design margin of 8% [18]. For larger design margin purposes, the second option for a solar panel to power the device was the 3W Seeed Technology Co. solar panel shown below along with its specifications.

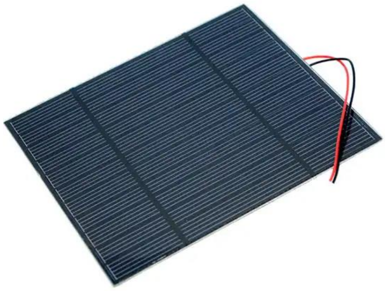

Figure 17: 3W Seeed Technology Co. Solar Panel

Table 5: 3W Seeed Technology Co. Solar Panel Specifications

Power (W) - Max	3W
Current @ Pmpp	540mA
Voltage @ Pmpp	5.5V
Voltage - Open Circuit	8.2V
Package/Case	Cells
Size/Dimension	160.00mm x 138.00mm x 1.50mm

This 3W solar panel is large enough to meet the desired minimum solar panel power of 1.86W with a large design margin of 61% [19]. Looking for a middle-ground option between option 1 and 2, option 3 the Allpowers 2.5W solar panel with its specifications is shown below.

Figure 18: Allpowers 2.5W Solar Panel

Table 6: Allpowers 2.5W Solar Panel Specifications

Power (W) - Max	2.5W
Current @ Pmpp	500mA
Voltage @ Pmpp	5.0V
Voltage - Open Circuit	8.6V
Package/Case	Cells
Size/Dimension	130.00mm x 150.00mm x 5.00mm

This 2.5W solar panel is large enough to meet the desired minimum solar panel power of 1.86W with a design margin of 34% [20]. Since the 2.5W solar panel is of sufficient power, has a larger than 25% design margin, nearly square design, protective film, and is smaller in physical size than the 3W solar panel, the 2.5W solar panel was chosen to power the wind surveying device.

4.1.3 Solar Power vs. Wind Turbine for Power Source

When choosing between solar power and wind power to supply power to the wind surveying device, a number of considerations were evaluated. First, the TEX:Energy Infinite Air wind turbine was tested in the WPI wind tunnel. Tested for the cut-in speed of the wind turbine, as well as the wind speed necessary to produce sufficient power were conducted. An 115Ω load resistance was placed at the output to act as the equivalent load of the final device. The following results were found:

Table 7: Test 1.1 of the TEX:Energy Infinite Air Wind Turbine for Cut-In Speed

Wind Turbine Test 1.1 (115ohm load resistance)					
Frequency (Hz)	**Wind Speed**		**Current Output (mA)**	**Voltage Output (V)**	**Notes**
	m/s	mph			
2.4	2.11	4.72	-	-	
⋮	⋮	⋮	⋮	⋮	Hz was increased in 0.1Hz intervals with no change in output mA or V
4.7	4.14	9.25	-	-	
4.8	4.22	9.45	-	-	
4.9	4.31	9.65	-	-	
5	4.40	9.84	-	-	
5.1	4.49	10.04	-	-	
5.2	4.58	10.24	-	-	
5.3	4.66	10.43	-	-	
5.4	4.75	10.63	-	-	
5.5	4.84	10.83	44	5.1	Cut-In Speed
4.7	4.14	9.25	44	5.1	Minimum speed when blades have already started to spin
4.6	4.05	9.06	-	-	Noisy output voltage
4.5	3.96	8.86	-	-	Noisy output voltage

A second test to determine the cut-in speed of the wind turbine was performed immediately following the first test to see if results were consistent. The following results were found during the second test:

Table 8: Test 1.2 of the TEX:Energy Infinite Air Wind Turbine for Cut-In Speed

Wind Turbine Test 1.2 (115ohm load resistance)					
Frequency (Hz)	**Wind Speed**		**Current Output (mA)**	**Voltage Output (V)**	**Notes**
	m/s	mph			
4.7	4.14	9.25	-	-	
5.5	4.84	10.83	-	-	
6	5.28	11.81	44	5.1	Cut-in speed

After these two tests were conducted, the wind turbine was tested using a variable load to find the maximum power production at various wind speeds. The variable load was implemented using an IGBT, with the gate voltage controlled from a potentiometer. The following results were found:

Table 9: Test 1.3 of the TEX:Energy Infinite Air Wind Turbine for Maximum Power Output

Wind Turbine Test 1.3 (variable load)						
Frequency (Hz)	**Wind Speed**		**Current Output (A)**	**Voltage Output (V)**	**Output Power (W)**	**Notes**
	m/s	mph				
6	5.28	11.81	0.1	5	0.5	
11	9.68	21.65	0.11	5	0.55	
16	14.08	31.50	0.266	4.91	1.30606	
			0.755	4.52	3.4126	
			1.075	4.24	4.558	
21	18.48	41.34	-	-	-	Shutdown

From the results, a wind speed of 41.34mph resulted in the wind turbine going into shutdown, where both current output and voltage output dropped to zero. Upon physical inspection of the turbine, it was noticed that the aluminum enclosure was hot to the touch, and was emitting a smell of burning FR4 material. Testing was concluded after this in fear of the turbine being permanently damaged or at risk of further damage. After a few days, the wind turbine was spun manually to see if there would be any output voltage. With this manual spinning, a steady output voltage of 5V was produced, so further testing in the wind tunnel was to be done.

A second round of testing the TEX:Energy wind turbine was conducted in the same wind tunnel at WPI where the first tests were conducted. For the first test of this session, the turbine was connected to a 100Ω load resistance and the cut-in speed was to be determined. However, during this test the wind speed would be incrementally increased until the turbine shut down and both output voltage and current were lost. This was to simulate how the wind turbine would respond if connected to the device while it was drawing maximum power. The following results were found:

Table 10: Test 2.1 of the TEX:Energy Infinite Air Wind Turbine for Cut-In Speed and Shut Down

Wind Turbine Test 2.1 (100ohm load resistance)					
Frequency (Hz)	**Wind Speed**		**Current Output (mA)**	**Voltage Output (V)**	**Notes**
	m/s	mph			
4	3.52	7.87	-	-	
5	4.40	9.84	-	-	
5.5	4.84	10.83	-	noisy 5V	
5.8	5.10	11.42	52	5	Cut-in speed
6	5.28	11.81	52	5	
10	8.80	19.69	52	5	
15	13.20	29.53	52	5	Shutdown after ~30seconds
					drop wind speed down to 6Hz
6	5.28	11.81	-	-	blades spinning very slow
			-	-	kept at 6Hz - blades slowly start to spin again
					still keeping at 6Hz - voltage & current slowly start to rise
	after 5 minutes still @ 6Hz:		35	3.2-4.2	current will not reach 52mA & voltage will not reach a steady 5V
					completely stop wind tunnel to let turbine cool down

From these results, a cut in speed of 11.42mph was found where the turbine started to output a steady 5V and the desired 52mA. While keeping the same load, the wind speed was increased until 29.53mph was reached and the turbine went into shut down. The turbine was left to cool down in the wind tunnel with the wind tunnel turned off. After this 10 minute period, the same test was repeated with the same load resistance where the following results were found:

Table 11: Test 2.2 of the TEX:Energy Infinite Air Wind Turbine for Cut-In Speed and Shut Down

Wind Turbine Test 2.2 (100ohm load resistance)

Frequency (Hz)	Wind Speed		Current Output (mA)	Voltage Output (V)	Notes
	m/s	mph			
5	4.40	9.84	-	-	
5.2	4.58	10.24	-	-	
5.3	4.66	10.43	-	-	
5.4	4.75	10.63	-	-	
5.5	4.84	10.83	-	-	
5.6	4.93	11.02	-	-	
5.7	5.02	11.22	-	-	
5.8	5.10	11.42	46	5	Cut-in speed
8	7.04	15.75	52	5	
					held steady @ 8Hz for 5min before going to 10Hz - output voltage & current remained steady
10	8.80	19.69	52	5	
					held steady @ 10Hz for 5min before going to 12Hz - output voltage & current remained steady
12	10.56	23.62	52	5	
					held steady @ 12Hz for 5min before going to 13Hz - output voltage & current remained steady
13	11.44	25.59	52	5	
					held steady @ 13Hz for 5min before going to 14Hz - output voltage & current remained steady
14	12.32	27.56	52	5	Shutdown after ~14s
			-	-	Cut right back in
			52	5	Shutdown again @ 45s
			-	-	Cut right back in
			52	5	Held steady to the 5 minute marker
15	13.20	29.53	-	-	Shutdown after 10 seconds

					turned wind speed right back down to 6Hz
6	5.28	11.81	-	-	blades spinning very slowly
			52	5	Output voltage & current came back after ~1 minute

Once again the cut-in speed of the wind turbine to start producing a steady output current of 52mA at 5V was found to be 11.42mph. The turbine also went into complete shutdown once again at a wind speed of 29.53mph, but had also showed signs of unsteady output voltage and current starting at a wind speed of 27.56mph. After these results, the feasibility of using this wind turbine came into question.

The cut-in speed of 11.42mph is quite high, limiting the energy production of the turbine and poses a risk of insufficient power being delivered to the wind surveying device if wind speeds do not regularly remain above 11.42mph. The shutdown speed of 29.52mph was also quite alarming because if wind speeds reach this level or higher, all energy production is lost. This is also a very narrow window of operating speeds from 11.42mph to 29.52mph.

A Final Decision:

The TEX:Energy Infinite Air wind turbine also posed some design issues. The turbine housing did not swivel, so in a natural environment, the turbine would not be capable of staying in headwind as the direction of wind changes. Even if a swivel mechanism were to be implemented, the turbine would still not be capable of staying in headwind because the housing has no vane to keep the blades faced into headwind. On the other hand, a solar panel has no moving parts, and that alone reduces costs for manufacturability of a future product, as well as increases longevity of the device. Due to these concerns, it was concluded that using a solar panel to power the device would be more feasible.

4.1.4 Version 1.0 - 5V Supply with Charger (Wind Compatible)

There were two version of the power converter PCB that were produced, version 1.0, and version 2.1. Version 1.0 was the first version, and had been built under the assumption that wind power would be used. It also assumed that the anemometer would be able to be powered off of 5v like the rest of the circuitry. More of the differences between the two versions will be addressed when discussing version 2.1, the final version. From here, follows a highly detailed description of the design practices and thought that went behind the production of this component of the final project. Please reference Figure 1 below while reading.

Transient voltage suppression (TVS) is an extremely important aspect of any electronics designed to be robust and adaptable. The longer the wires and connectors to the power PCB are, the higher probability that a transient could get picked up by the connection and a large emf is generated, possibly hurting the device. For this TVS need, the team went with a bidirectional TVS diode to clamp high voltages very quickly. To block these high voltage (and high frequency) transients from entering the device in the time it takes to clamp, a ferrite bead was used as a low-pass filter.

For the charging controller, the parallel battery configuration was deemed to be the best due to automatic current reversal from the battery into the device when needed; an IC was not needed to control that. The charging circuitry was loosely based off of a breakout board from Adafruit (Adafruit 259). The topology is simple, and the IC (MCP73833) follows all standard practices and obvious considerations.

For the converter - a simple boost topology - the versatile LMR62421 was chosen as the switching regulator. Its 1.6 MHz switching frequency would allow for the construction of a very dense PCB, and subsequently small power board in general. This is due to the reduced values for the passives as frequency climbs higher and higher. Standard practices were followed from here, and TVS was placed at the 5v output.

Choice of capacitors is also very important. For this extremely high frequency converter (1.6 MHz), a low-ESR capacitor is needed to prevent the ESR-zero from being present in the control loop. Two 47uF ceramics in parallel were chosen as well as a 10uF of a different package to broaden the frequencies filtered out of the switching noise. Bypass capacitors were placed close to IC inputs, as is standard practice.

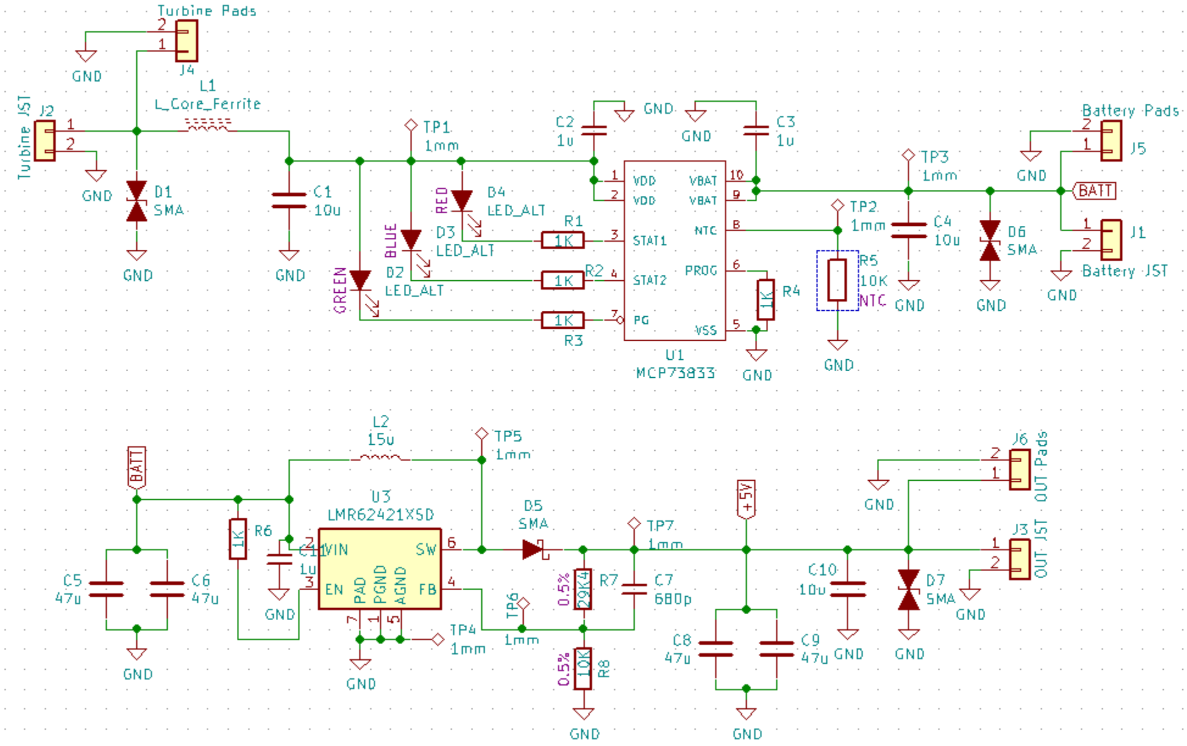

Figure 19: Version 1.0 Schematic for Charging Circuit (top) and 5V Converter (bottom)

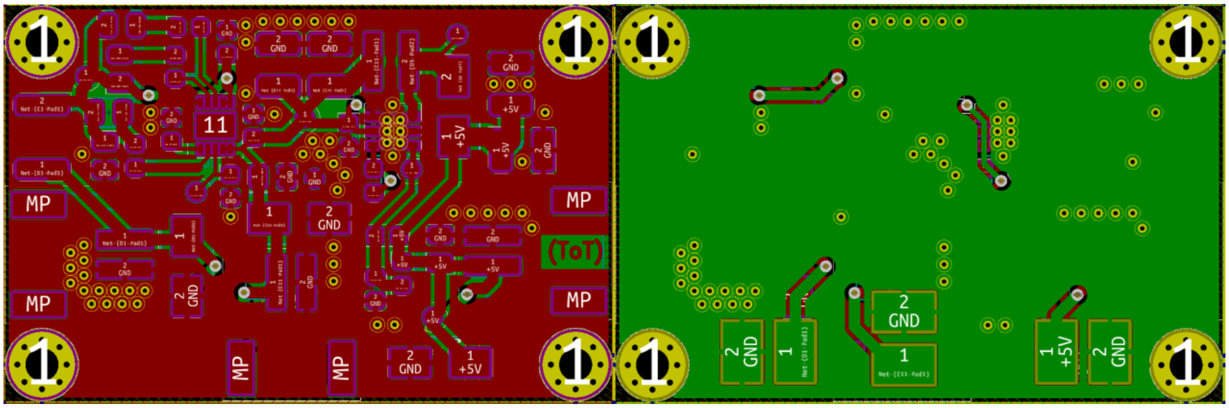

Figure 20: Version 1.0 Rendering of Top and Bottom Copper Layers

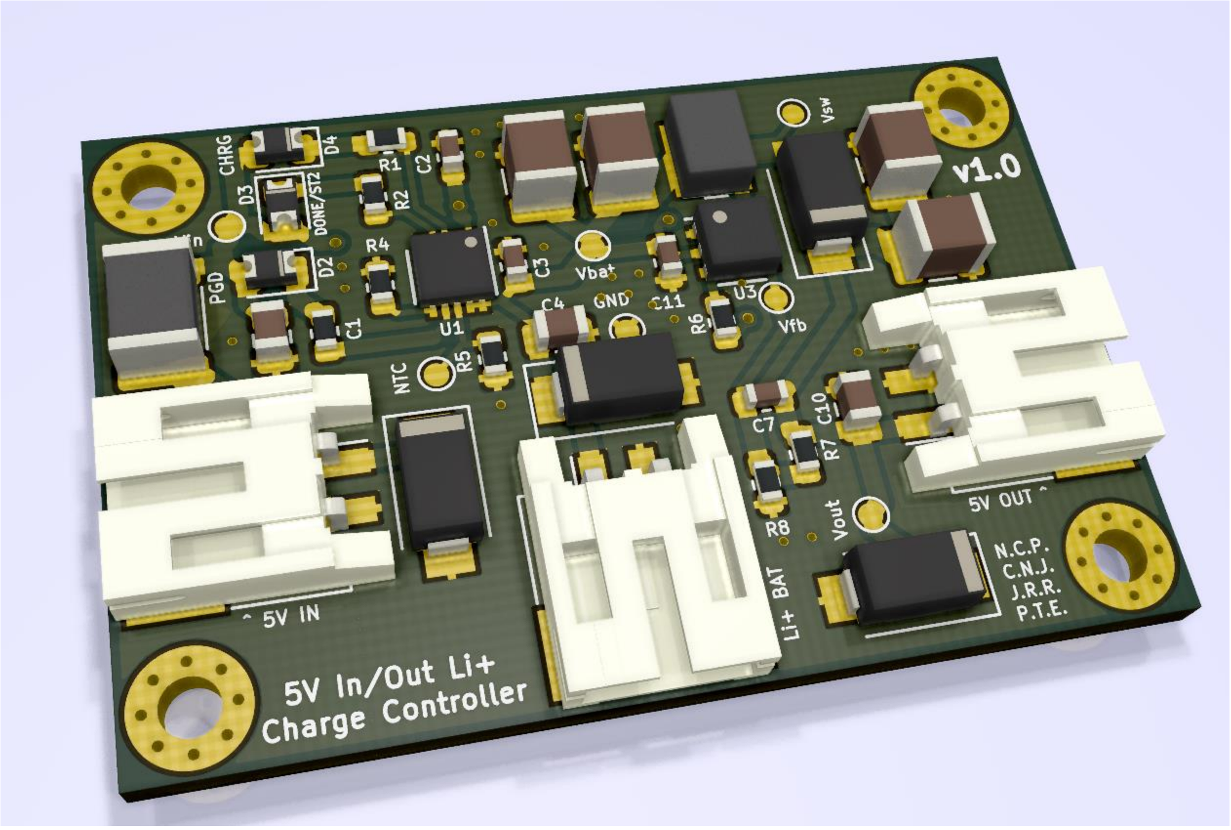

Figure 21: Version 1.0 Rendering of Final PCB

4.1.5 Version 2.1 - 5V and 9V Supply with Charger (Solar Compatible)

Although version 1.0 was successful in achieving its stated requirements, a shift in requirements for the power circuitry naturally called for an updated board. In addition to necessary new features for this second version, small problems with the first version were noted during the build/test process, and were desirable to fix, although not critical to the performance of the product.

Problems with version 1.0 and needs for the second version:

1) Needed to handle large input voltages due to the open-circuit voltage of a solar panel (16V max for solar panels of similar size)
    a) Change in charge controller IC was needed
    b) Preferably a leaded / bigger part (MSOP - SOIC)
        i) Ease prototyping and dissipate heat better
2) Needed a 9V rail, and an entirely different converter topology along with the boost
    a) Change magnetics and size of the boost with the change in power requirements
    b) Shrink the boot footprint, and increase inductance.
    c) Requirement went from ~50mA to ~25mA
3) Proper heat sinking was needed for the linear regulator / charge controller, to dissipate heat into the bottom ground plane as well
4) Integrated UVLO of the switching regulator was not properly working
    a) Needed a discrete UVLO
5) Needed reverse polarity protection
6) Needed bigger bulk capacitance to deal with anemometer drawing inconsistent power
7) Should have used small SMD jumpers to facilitate the building and debugging of complex and risky circuitry
8) Connectors were too big for the low power requirement, and were wasting space
    a) The female receptacle also was not even correct for the males the team had acquired
9) Standoffs were too big
10) Mounting holes should have been placed in multiples of 2.54mm as to be easily mountable to a perfboard

The charging circuitry was only partially changed, with a selection of the TI BQ24202 MSOP IC as the LDO regulator and controller. A simple PMOS was used to provide the RPP, and discrete UVLO was implemented with a load switch and an inverter. The inductance and brand of the boost converter inductor was changed to be a high-quality Coilcraft of 33uH.

The flyback was designed and all proper safeties were implemented, including leakage inductance clamping via diodes D8 & D9, bypass capacitors, and TVS on the output. The LT 1610 was chosen to provide high frequency (1.7 MHz) control loop regulation.

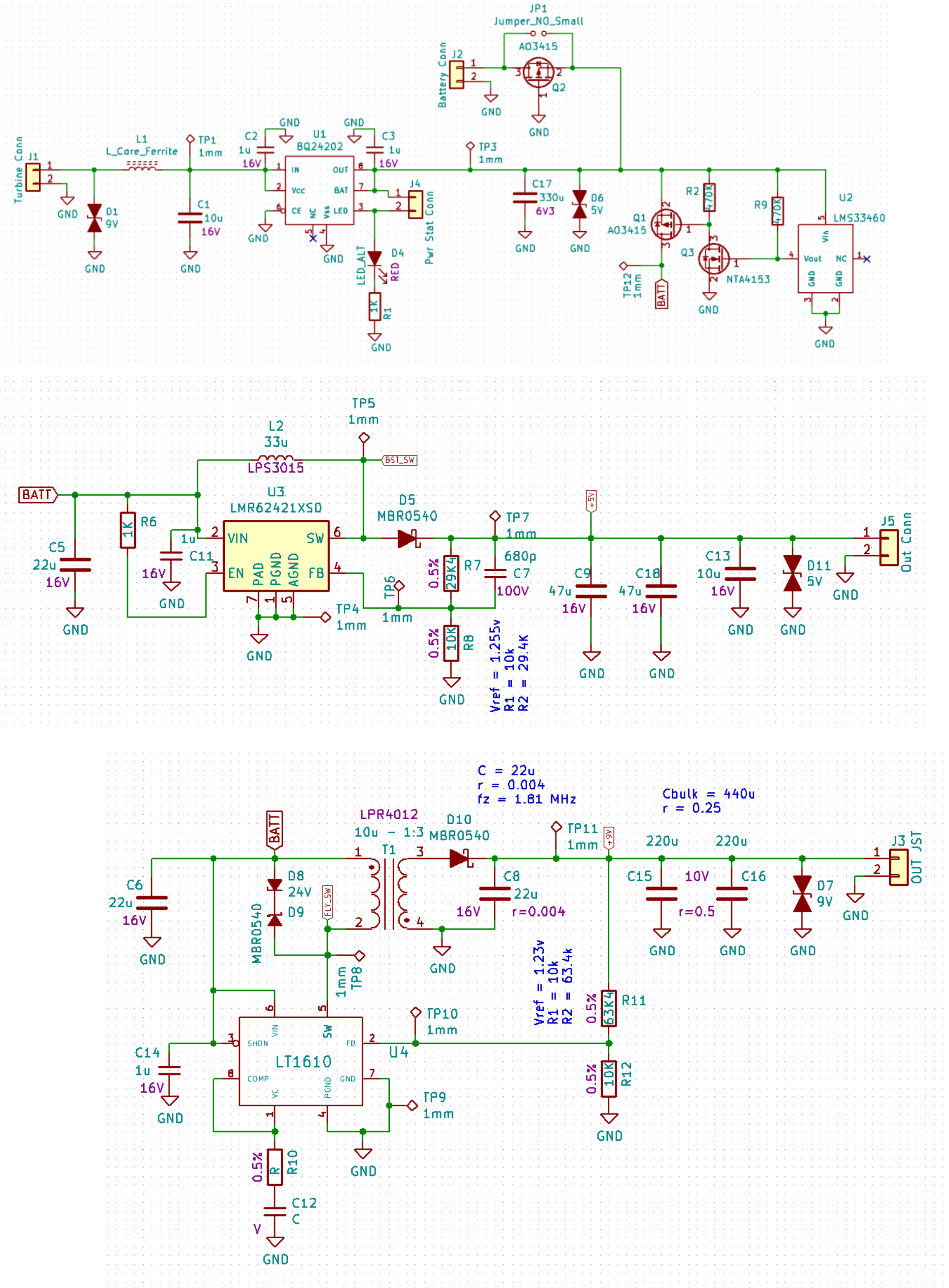

Figure 22: Version 2.1 Schematic for Charging Circuit (top), 5V Converter (middle), and 9v Converter (bottom)

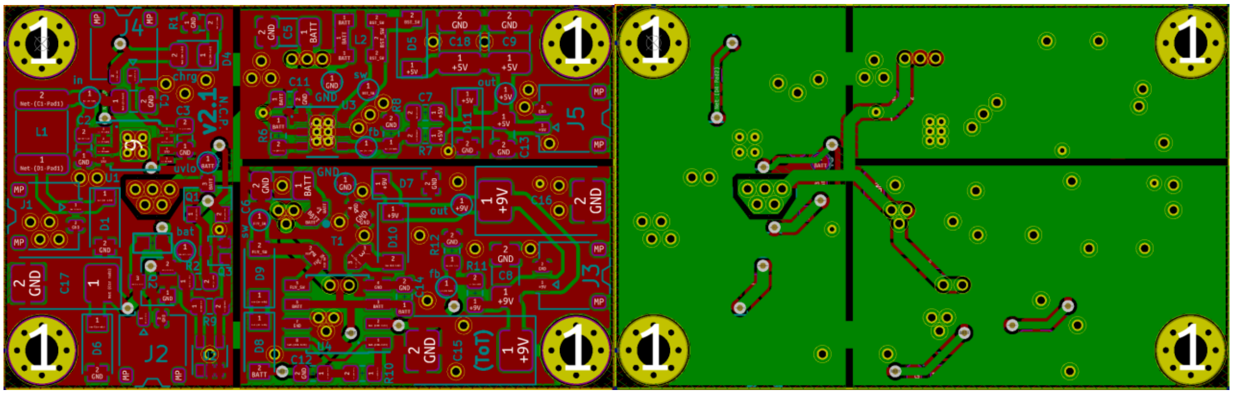

Figure 23: Version 2.1 Rendering of Top and Bottom Copper Layers

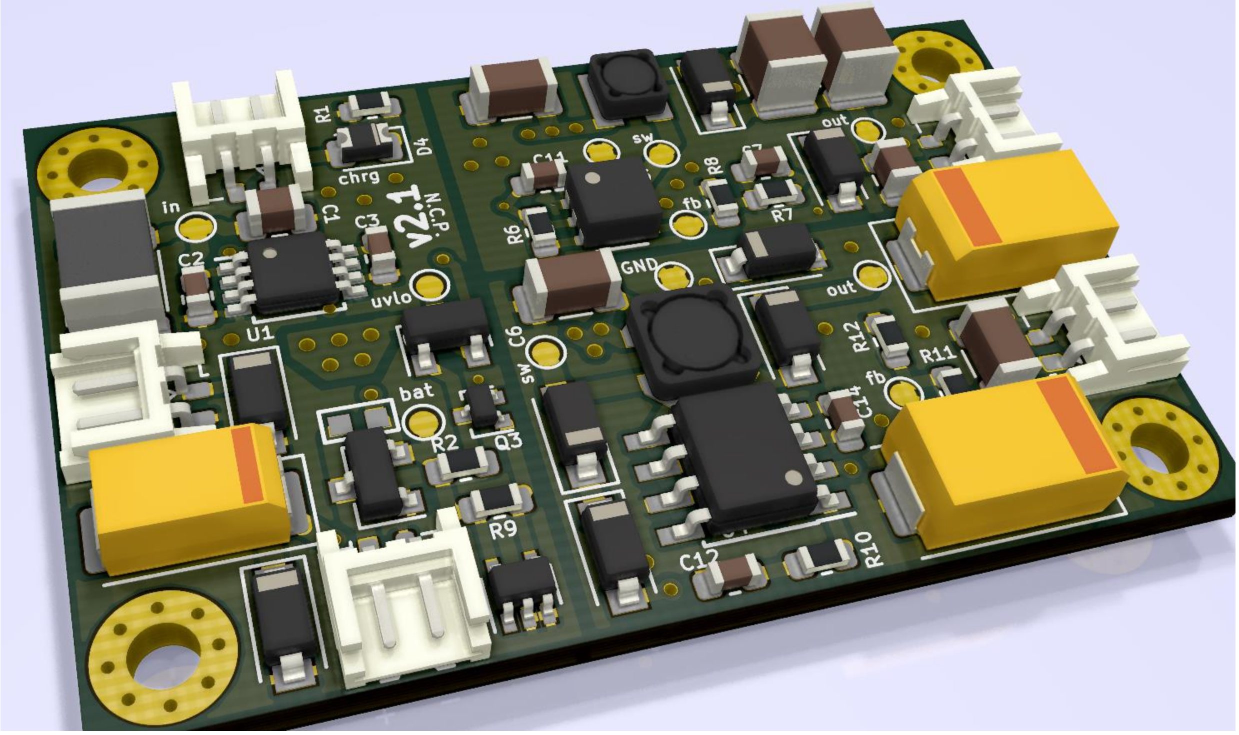

Figure 24: Version 2.1 Rendering of Final PCB

4.1.6 Verification

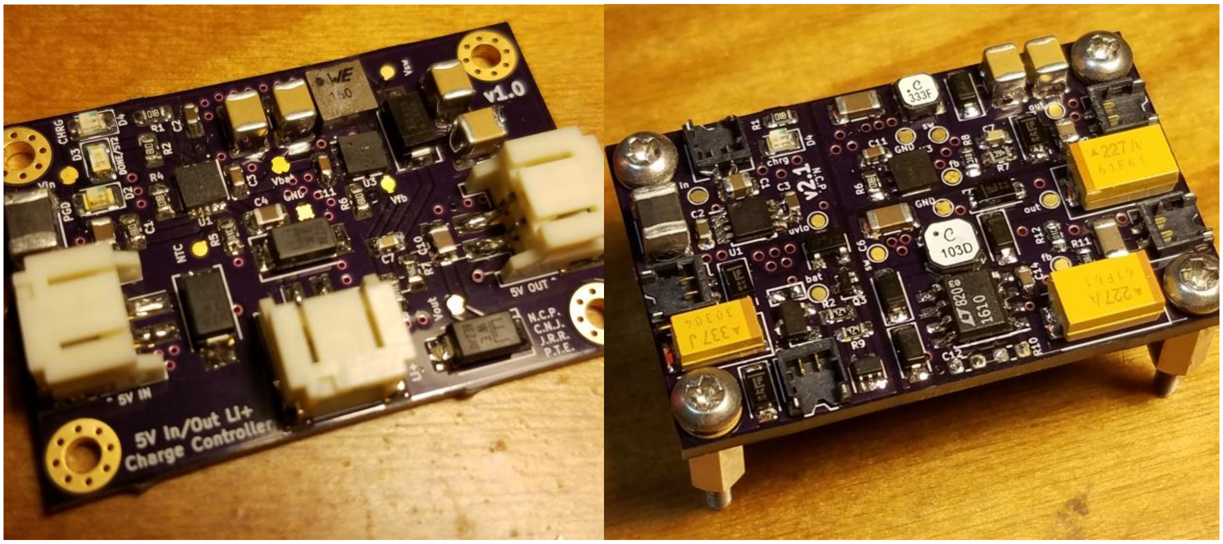

Figure 25: Final Assembled Boards: Version 1.0 (left) and Version 2.1 (right)

For both versions, to verify proper operation, each section was tested independently, and then all tested simultaneously. Proper resistances were calculated to draw the necessary loads, while operation was tested with no batter, uncharged battery, charged battery, no source, full power source, and the combination and matching of all such conditions.

4.2 Sensors

4.2.1 Anemometer Selection

There were two options for where to get the anemometer – an original one could be built, or the purchase a premade anemometer. If building an anemometer, it would have to be carefully balanced, with low frictions bearings and a watertight housing. Given that these challenges would take several weeks to overcome, and given that most of the challenges were mechanical, not electrical, a premade anemometer was chosen.

Figure 26: Adafruit Anemometer

The first anemometer considered was from Adafruit, and is shown above. This one was convenient, because it could be ordered through an American supplier. It is rated to read up to 70 mph wind, with a 0.2 mph cut-in speed. In this speed range, it is rated to output between 0.4V and 2V, which can be easily read by the Teensy. The anemometer also consumes an appropriate amount of power that could reliably be supplied. However, it operates on between 7V and 24V, which would complicate the power system design [21]. Because of this, it was decided to look for other anemometers.

Figure 27: Vienasa Anemometer

The next anemometer examined was distributed by Alibaba, and is pictured above. It is rated to measure up to 100 mph wind, has a cut-in speed of 1mph. It also has an IP-45 rating, meaning it is sufficiently protected from weather - a useful feature for the final product. Additionally, it takes a 5V supply and has a low power consumption, addressing the issue that the Adafruit anemometer had. It outputs a 0V to 5V signal, but the circuitry for addressing this issue is straightforward. However, the shipping time was 22 days (from China), which was far too long to wait to receive a critical portion of the project [22]. Thus, the search for other anemometers continued.

Figure 28: Anemometer from Rika

The next anemometer examined was from Rika, another company based in China. This anemometer was rated to measure wind speeds up to 135 mph, has a cut in speed of 1 mph, and is IP-65 rated, which means it has even better weatherproofing. It also operated off of 5V, had low power consumption, and outputs a 0V to 5V signal, like the anemometer from Vienasa. This anemometer, however, was able to ship within a week, which was acceptable, Unfortunately, when ordering this anemometer, the shipping was going to cost over $200, so it was decided to not use this anemometer either [23]. The company also required a MOQ of 2, as this product was meant to be bought in bulk.

Thus, it was decided to go back to the first anemometer, from Adafruit. This was the only anemometer that could be expected to be received in a reasonable amount of time, for a reasonable price. This choice meant that the power system would need to be modified to supply 9V to the anemometer, but this was a tradeoff worth making.

4.2.2 Anemometer Testing

The anemometer was tested to determine two things: the power consumption, and the relation between the wind speed and the voltage signal. The first test, of power consumption, was to find the exact power consumption of the anemometer, since it only came with a maximum power consumption. This test was done by spinning the anemometer with a motor, and monitoring the current draw. This test showed that the anemometer would draw no more than 19mA, which, with an input voltage of 9V, meant that the anemometer would use no more than

0.171 W. The next test, to determine the relation between wind speed and output voltage was done in the WPI wind tunnel, thanks to the aerospace engineering department. For this test, the output voltage of the anemometer was measured at various wind speeds, and the data was plotted, resulting in Figure 29:

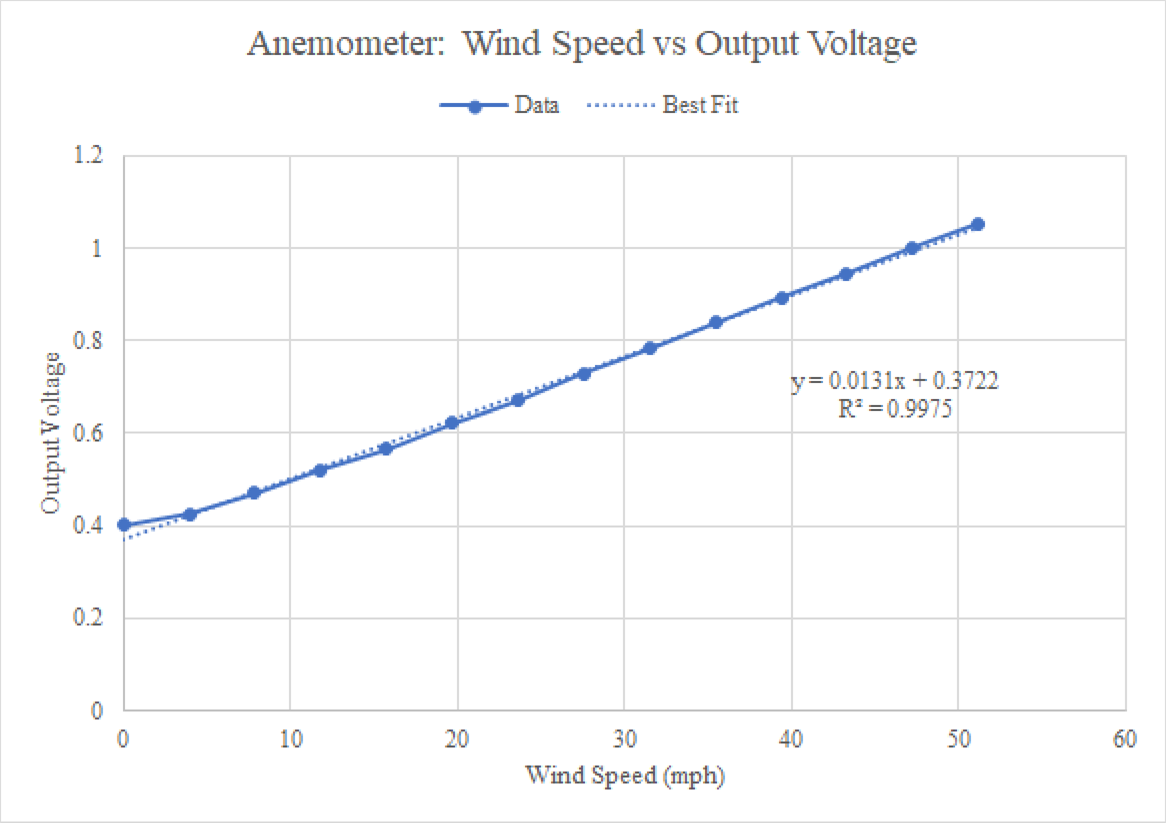

Figure 29: Wind Speed vs Voltage Output of the Anemometer

Figure 29 also shows the best fit curve relating the wind speed to voltage. This provides a way to turn the voltage readings into real values, following the equation:

$$u = 75.88v - 28.175$$

Where 'u' is the wind speed, and 'v' is the voltage. This is important, because it allows the gathered data to be turned into useful information.

4.2.3 Wind Direction Sensor Selection

Initially, the absolute encoder was thought to be ideal for this application, since the position wanted to be known, not the rate of rotation of the vane, and because it seemed easier to use than a magnetometer. However, once the encoder was ordered, it was found to be a critical issue. The encoder had too much internal resistance, meaning that in light to moderate wind, the wind direction sensor would not move. Since it is important to measure the wind direction, even at low speeds, this encoder would not work. Looking for alternative encoders, it became clear that any absolute encoder with low internal resistance would cost too much. However, directional magnetometers cost significantly less, with some individual ICs costing less than $2. Thus, it was decided to use a magnetometer.

The first magnetometer ordered also had some issues. It was designed to work with the Arduino pro mini, which is similar to the Teensy LC used in the device, so it was believed that it would be simple to modify the code to work with the Teensy. However, this proved to be more complicated than anticipated, and under a rush for time, it was chosen to use a different magnetometer that was designed to work with the Teensy.

The ‘prop shield’ from PJRC was chosen, which is the same company that makes the Teensy. To set it up, the power and SPI pins on the Teensy simply needed to be connected to the shield. Then, the test code was run to verify that the magnetometer was working. Once it was verified to be working, the code was modified so that it could be used for the desired purposes.

4.2.4 Wind Direction Sensor Verification

After the code was modified, the magnetometer was tested, to see that it could tell the direction a magnet was pointing. By observing changes in the value of the magnetic field reading, it was verified that the system had been properly set up. Additionally, it demonstrated that the plan for the wind vane is conceptually possible.

Once the wind vane was constructed and integrated it into the finished system, the functionality of the magnetometer was once again tested, along with the rest of the system. Like before, it was tested to see if the magnetometer could detect a magnet spinning above it. Unlike before, the magnet was separated by the housing of the system, and was surrounded by electronics which could potentially cause interference. However, it continued to function as expected, and was able to successfully determine the direction the wind vane was pointing. Thus, the wind direction sensor was successfully implemented.

4.3 Embedded System

4.3.1 Microprocessor Selection

Considering all of these requirements for the microprocessor the search was narrowed down to three common microprocessors: the Arduino Nano, Teensy LC, and Raspberry Pi Zero. A breakdown of each of those microprocessors according to the specifications previously laid out is as follows:

- Arduino Nano [24]
    - I/O
        - 8 Analog I/O (Can be used as Digital I/O)
        - 12 Digital I/O
        - Mini-B USB connector
    - Power
        - Atmega328 (8-bit)
        - No low power modes
        - Older inefficient processor
    - External Processing
        - Serial transmitter/receiver for writing to MicroSD card
    - Development Environment
        - Arduino IDE, lots of documentation of projects done with similar Arduinos
- Teensy LC [25]
    - I/O
        - 25 Analog/Digital I/O ports
        - Micro-B USB connector
    - Power
        - ARM Cortex-M0+ (32-bit)
        - Low power modes available
    - External Processing
        - Can attach
    - Development Environment
        - Can also use Arduino IDE with plugins, similar levels of documentation as Arduino
- Raspberry Pi Zero [26]
    - I/O
        - 40 GPIO ports
        - HDMI + 2 micro-B USB + microSD card reader
    - Power
        - BCM2835 Microprocessor (1GHz CPU + GPU)
        - Efficient but very powerful processor
    - External Processing
        - Can connect USB devices to store data
    - Development Environment
        - Has an operating system built in and can program directly on the device

After reviewing the pros and cons of each of the potential microprocessors the Teensy LC was chosen for its efficient 32-bit processor, feature rich development board, and easy programming suite.

4.3.2 Embedded System Architecture

Hardware Architecture:

The hardware design portion of the embedded system was very straightforward due to the selection of the Teensy LC for the microcontroller. The Teensy LC uses an ARM-Cortex M0+ microprocessor and has a development board with everything necessary for the device to function. This includes a USB microB port for loading code into it, a crystal oscillator for timing, and the I/O mapped to different ports. Everything on the development board is documented thoroughly including the schematics of the development board by PJRC, the company that makes the Teensy.

While most of the hardware design is handled by the Teensy development board itself, there are a few places where additional elements were added to the embedded system. The major addition to the embedded system is the microSD card reader and writer. The microSD card writer is the bridge between the embedded system and the analysis software. The microSD card allows for the microprocessor to store the data from the sensors so that it can be read by the analysis software. The reader itself is a self-contained board, also made by PJRC with inputs for serial communications and power. The board is connected to the Teensy using the Teensy's MISO, MISI, CS, and SCK ports as well as the 3.3V rail to allow for communication and writing data to the microSD card.

The anemometer is connected to the Teensy using one of analog-in ports. The Teensy has a built in 10-bit ADC. The ADC has a voltage range of 0-2.56V, giving a resolution of 0.0025V. From the anemometer testing it was determined that the anemometer has a maximum output voltage of ~1.1V which is within the range of the built in ADC.

The wind direction sensor was implemented by using a magnetometer and a small magnet on a wind vane. The magnetometer is part of a multi-sensor shield for the Teensy that is connected using the MOSI, MISO, SCK, IRQ, SCL, and SDA ports of the Teensy.

The power delivery system connects to the Teensy on the 5V input pin. The only modification to the board was to cut a small trace between the microB USB port to assure that the power was coming from the power delivery system instead of the USB port when debugging.

Software Architecture:

The software side of the embedded system is what allows for the entire system to function properly. The software for the embedded system is written using the Arduino IDE which uses a language similar to C/C++. This is possible due to a number of libraries that allow the Teensy to work with the IDE. The Arduino IDE makes many basic operations much easier to perform, such as configuring input and output for pins, as well as providing libraries to do more complicated functions such as writing serial data. The major functions for the embedded system include: reading the anemometer data, reading the magnetometer data, creating a packet, and writing the packet to the microSD card.

Reading the anemometer data is the simplest part of the software. The pin that it is attached to is configured to be an analog input. When the system goes to collect the sample, it reads the voltage on the pin using the ADC and stores it in a 16-bit integer.

The magnetometer sends its data using SPI. The data is stored using two integers, one for the x-direction and one for the y-direction of the magnetometer, which can be used to determine the angle that the wind vane is facing.

The data is stored in a packet data structure so it can be read later. The packet stores the time of the sample as well as the data from the sensors. The packet was implemented as a struct in the Arduino IDE with a 32-bit integer for the sample number, a 16-bit integer for the anemometer reading, and two 32-bit integers for the magnetometer readings.

The packets are stored in the microSD card, which allows for the data collected by the device to be stored for later analysis. Originally the data in the packet was stored as a binary file on the microSD card, but when the direction sensor was changed to be a magnetometer instead of an encoder it was switched to be a text file. This was due to complications of reading a binary file with the analysis software. The packet is written to the microSD card as a numerical representation of the integers in a text file, with separating characters to allow the analysis software to distinguish between different packets and values. The SD and Serial libraries for the Arduino IDE simplifies the writing to microSD card with their built-in functions.

PCB Design:

Although most of the embedded system design was straightforward due to using prepackaged solutions, a PCB was designed to integrate the separate systems. The main components of the embedded system were the Teensy LC and the microSD card writer, but the anemometer, the magnetometer, and the power delivery system all connected to the embedded system. The PCB design should compact these main components and add headers to connect it to the rest of the system.

The board is a standard FR4, double-sided 1.6mm PCB produced by OSHPark and was designed using KiCAD, a free and open-source PCB editor. It has components installed on the top side and has short hex standoffs to mount to the enclosure. It also was designed to accept female header strips, which would subsequently mate with the male header strips of each subsystem. Additionally, headers for connecting the power board, the anemometer, and the magnetometer board were included to simplify the connections among the different components of the system. Care was taken to determine appropriate trace widths, and the traces were laid out to be as compact as possible. A render of the embedded system PCB can be seen in the figure below:

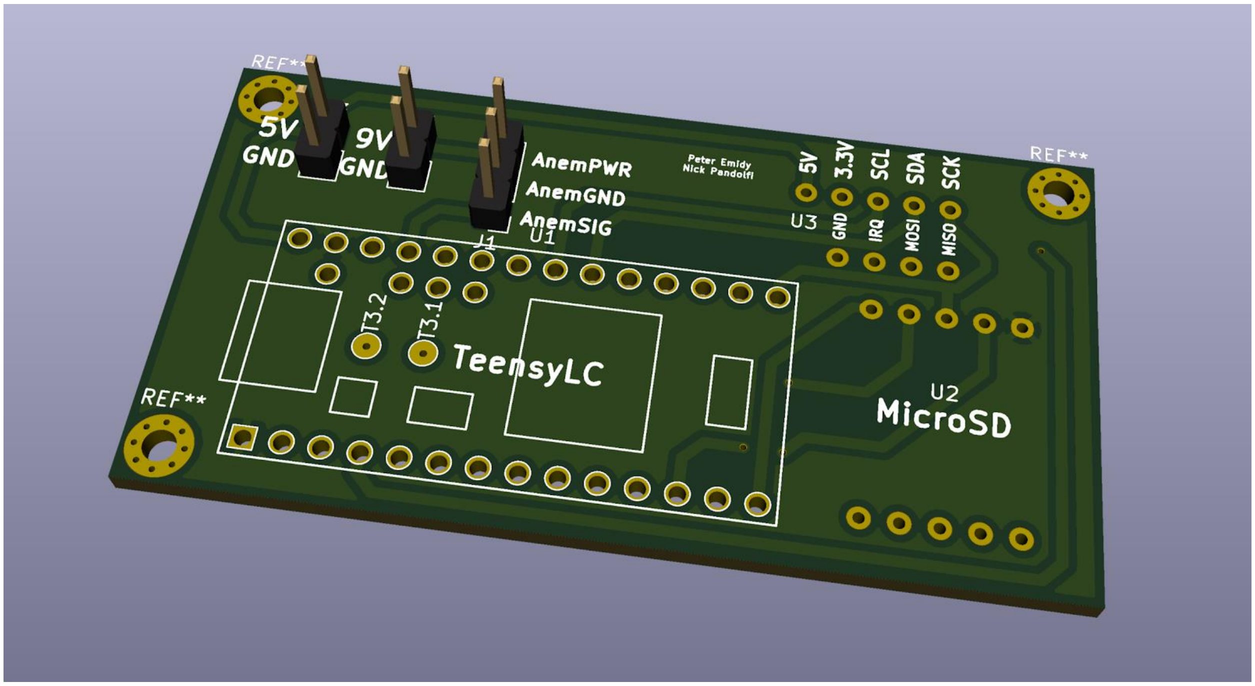

Figure 30: 3D View of the PCB

In the figure above, the PCB has locations for female headers to be soldered onto the board. These headers allow for the daughter boards such as the Teensy LC and microSD Card Reader to be easily replaced for debugging purposes.

4.3.3 Embedded System Results

Testing of the embedded system took place throughout the entire duration of the project, as most changes to the sensors affected the organization of the embedded system. First, the Teensy's hardware was tested by itself, and then the additions such as the microSD card reader, anemometer, and magnetometer were integrated into the overall embedded system. All of this testing was initially done on a breadboard for simplicity. Once all of the components were finalized, the PCB was designed, built, and tested.

First the Teensy LC was tested to ensure that the hardware was functional. This was done by creating a simple program that would blink the LED on the Teensy and write data through the serial port. With the hardware verified, the next section of testing was performed with the anemometer.

The anemometer produces an analog voltage signal that linearly corresponds with the wind speed. To measure this analog voltage signal, one of the analog inputs on the Teensy was used. To initially test the analog input, a voltage divider using a potentiometer was created and the Teensy was configured to read the voltage through one of the pins. Then the voltage was calculated using code in the Teensy's software and sent through the serial port to be read. Next the voltage was measured using a multimeter and was compared to the measured value by the

Teensy. The values were consistent, so the Teensy's ADC was determined as suitable, and the anemometer output voltage signal was measured using it.

The next component of the system that was tested was the microSD card writer. The microSD card writer used was designed by PJRC specifically for use with the Teensy. To test the writer, pins were soldered to the mounting holes and the device was placed on the breadboard. Then the connections were made to the appropriate pins on the Teensy and the microSD card reader. Once the devices were connected properly, test code included in the Arduino IDE was run to determine if the microSD card was being recognized and writing a simple text file to it. When the device was confirmed as working, newer code that would write packets to the microSD card was tested. The functionality of the microSD card writer was confirmed by sending the written packets through the serial port and comparing them on an external computer to assure they were the same.

The final component tested on the embedded system was the magnetometer. The first magnetometer used was made by SparkFun. This magnetometer was designed for use with an Arduino. The device was connected using the appropriate ports and the test programs were loaded onto the Teensy. Unfortunately, the Teensy was never able to communicate with the magnetometer. After further investigation this was because the break out board for this magnetometer was configured to work with the registers of the Arduino, which are different than those of the Teensy. Subsequently, the 'Prop Shield with Motion Sensors for Teensy' by PJRC was chosen, since this board was designed specifically for use with the Teensy. The Prop Shield also included libraries for the Arduino IDE and some example code that was used to test to make sure that the sensors were being read. This code was modified so that it would just collect the necessary information for measuring the wind direction using a magnet. Then a small neodymium magnet was moved around the magnetometer to assure that the magnetometer was measuring the changes in the applied magnetic field. Once the magnetometer was confirmed to be working, the testing code was implemented into the full embedded system code for writing the magnetometer data to the packets.

With all of the components of the embedded system tested using a breadboard, a PCB to consolidate and simplify the connections of the embedded system was designed. Once the PCB was received and the headers were soldered onto the board, the components of the embedded system needed to be tested again. With all of the components installed on the PCB, the full embedded system software was run, but the microSD card was not working. Doing continuity tests across the PCB determined that the microSD card reader daughter board was not properly grounded. To solve this, a jumper wire was soldered between the pin of the daughter board and the ground pin of the Teensy LC.

4.4 Analysis Software

4.4.1 Analysis Software Language Selection

Python 3 was used to prototype the code because it allows for high-level debugging and object-oriented design. Python contains many built-in functions that are very helpful for simple operations on linear datasets such as the ‘list’. Python also contains hash-table type objects such as the ‘dict’ or ‘dictionary’ that will allow for more dynamic lookup of important key-dependent data. Python is very flexible in its usages, and is overall a very safe choice regarding functionality in the prototyping phase. However, Python is slower than its lower level counterparts like C. When not used correctly, Python can be extremely slow at times, so care must be taken to use all of Python’s highly optimized built-in functions to avoid ‘reinventing the wheel’. When used correctly, Python will provide a balance of functionality and performance. Python also has many libraries for easily plotting data for the end user of the device to view in a neat manner.

One of the libraries used was Numpy. Numpy has lots of functions for operations and data analysis on arrays. Most of the data for processing is stored in arrays or lists therefore Numpy provides many tools to efficiently modify it.

For data visualization the Python library Matplotlib was used. This library includes many plotting and graphing tools similar to MATLAB. This allows the user to create multiple charts for representing the different data such as Wind Speed vs Time, Percentage of Time Wind is Blowing in a Direction, and Average Wind Speed in each Direction. These charts are made possible using Matplotlib’s ‘plot’ and ‘bar’ functions that produce line and bar graphs respectively. There are also additional features that allow the user to display the data on the polar plane to give a more intuitive representation of the direction.

4.4.2 Analysis Software Code Topology

The analysis software was implemented using Python 3.7. The program was broken into different sections to accomplish the various goals of the software. These goals included reading the text file produced by the anemometer, modifying the data to be readable, doing calculations to determine wind speed and direction, and graphing the data.

The first section of the analysis software was to load the wind data from the text file produced by the embedded system. In Python there is a built-in function for opening files that works particularly well with text files. Once the file is loaded into the analysis software, a series of string operations are performed on the file to extract the data and store it in appropriate arrays in the program. First, the software searches for ‘:’ within the text file. The ‘:’ is used as a delimiter to differentiate samples of the wind data from each other, therefore the total number of colons in the text file is the number of samples taken by the device. With the number of samples known, arrays of the proper size can be created for storing the identification number, wind speed and wind direction. Next, the software goes through each line of data and splits on ‘,’. In the text file the ‘,’ is used to separate the ID number, the wind speed, and the x-direction and y-direction. With the points split they can be loaded into the arrays created for storing their respective values.

The second section of the code was made to modify the raw data from the text file into a more readable form. This includes changing the wind speed to a value in mph and the direction to be in degrees. The data recorded for wind speed is an integer recorded by the ADC from 0-1024 based on its voltage from 0-3.3V. To get a readable value from the wind speed data, calculations are performed on the value to transform it to a voltage value. Then using the equation determined when calibrating the anemometer in the wind tunnel, the speed in mph is calculated. Next, the data recorded for the wind direction is stored as the x and y directions of the magnetometer. To obtain the direction in degrees from those values, the arctan is applied to the values and the resulting values are converted to degrees by dividing by pi and multiplying by 180 degrees.

The third part of the analysis software was to modify the data further to use in the graphs of the wind direction. To do this, the arrays are sorted by their direction. This makes it easier to determine the average speed and the amount of time spent in each direction. This was implemented by combining the three arrays for ID, wind speed, and wind direction into one 2-dimensional array. By doing this, the new array can be sorted by its direction using an optimized Python function. Once the array is sorted by its direction, the array can be iterated through to determine the average speed and time spent in each direction. Two new arrays are created: a ‘percentDirection’ array and an ‘averageSpeed’ array. Both of these arrays are of length 360 so that they can store their respective information for all 360 degrees. As the direction-sorted-array is being iterated through, it keeps track of how long it stays on the same direction, while also keeping track of the wind speed data for that direction. Once the direction is changed, it stores the percentage of the total time it spent in that direction in the ‘percentDirection’ array and then calculates the average of the wind speed data points and stores it in the ‘averageSpeed’ array. With all of the data analyzed and sorted it is ready to be graphed.

The fourth part of the analysis software is the graphing section. In this section, all of the data that has been modified throughout the software can be visually represented. This project used the Matplotlib library, which contains many different plotting functions. The three graphs that are plotted by the analysis software are wind speed over time, percent of time in each direction, and average speed in each direction. The wind speed over time graph was implemented as a line chart using the wind speed array as the y-axis and the ID number as the x-axis. The other two graphs were implemented using bar charts using a polar coordinate system which allowed them to display the graphs circularly which provided an intuitive representation of the wind direction. The two graphs were plotted using their respective arrays.

4.4.3 Analysis Software Testing

The analysis software was tested throughout its development to ensure that the various functions were working properly. An inherent issue with testing the analysis software was that getting data to analyze takes a lot of time, so a solution was required to get large quantities of data to analyze. This was especially important because the system was not in a state where it could collect data for the majority of the project and waiting until the system was complete to begin working on the analysis software would have resulted in low quality work. To solve these problems, additional software was written to produce simulated wind data that would be similar to the real wind data expected by the finished system. The data from these programs could then be used to test the functions and graphing of the analysis software.

The script created to test the analysis software was a pseudorandom data generator. This program worked in reverse of the analysis software decoder where it generated a data point and then encoded it to a text file using the same packet structure. This method was also helpful in testing the decoding feature of the analysis software because the points generated in the data generator could be compared to the points decoded in the analysis software to make sure they were the same.

The pseudorandom number generator used an algorithm to produce values thought to be similar to real wind data. The algorithm works as follows. There are four variables created: one for a coarse value and one for a fine value for both wind speed and wind direction, and a fifth variable to change the coarse value. For every point created, a small random number is generated for the fine values of wind speed and direction. These variables represent moderate fluctuations in wind speed or small changes in the direction of the wind that occur in small time scales. These values are then added to the coarse values to get the speed or direction recorded for that point. The variable for changing the coarse value was another random variable, where every time it equaled zero, a new random number for coarse value was generated. This signaled a large change in the wind speed or direction. Using this method, large quantities of data with similar properties to wind were generated for testing the analysis software.

With a method for producing data, the graphing functionality of the analysis software could be tested using the pseudorandom data. Using the pseudorandom data generation program, 5000 points of data were created for testing purposes. This generated text file was run through the analysis software and the following graphs were produced:

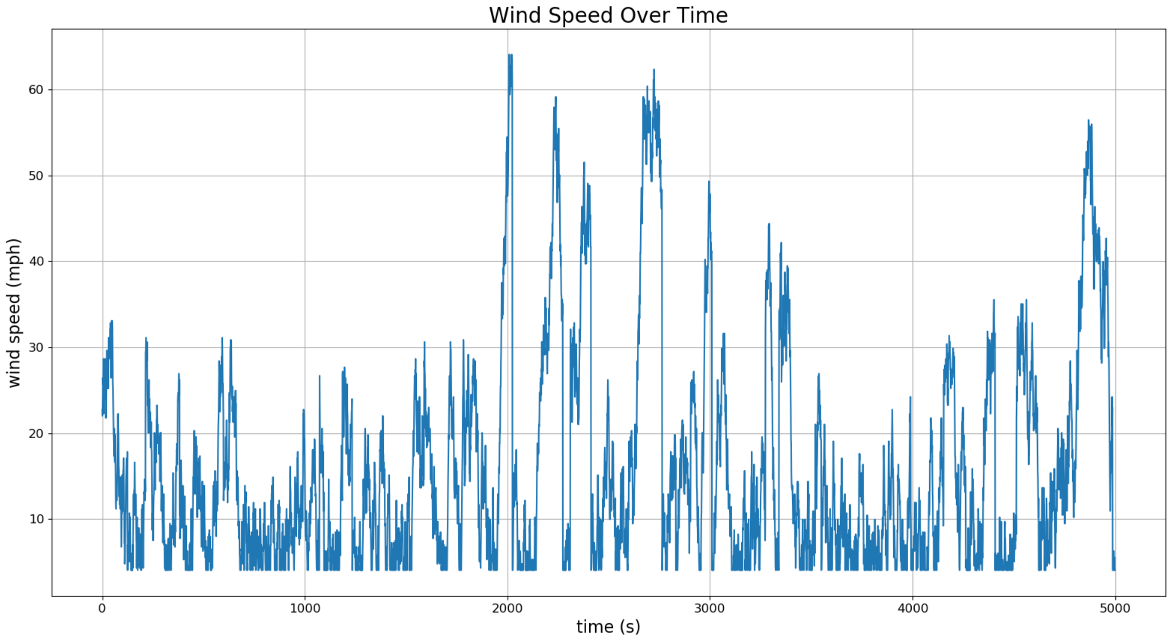

Figure 31: Wind Speed vs Time for the Pseudorandom Data

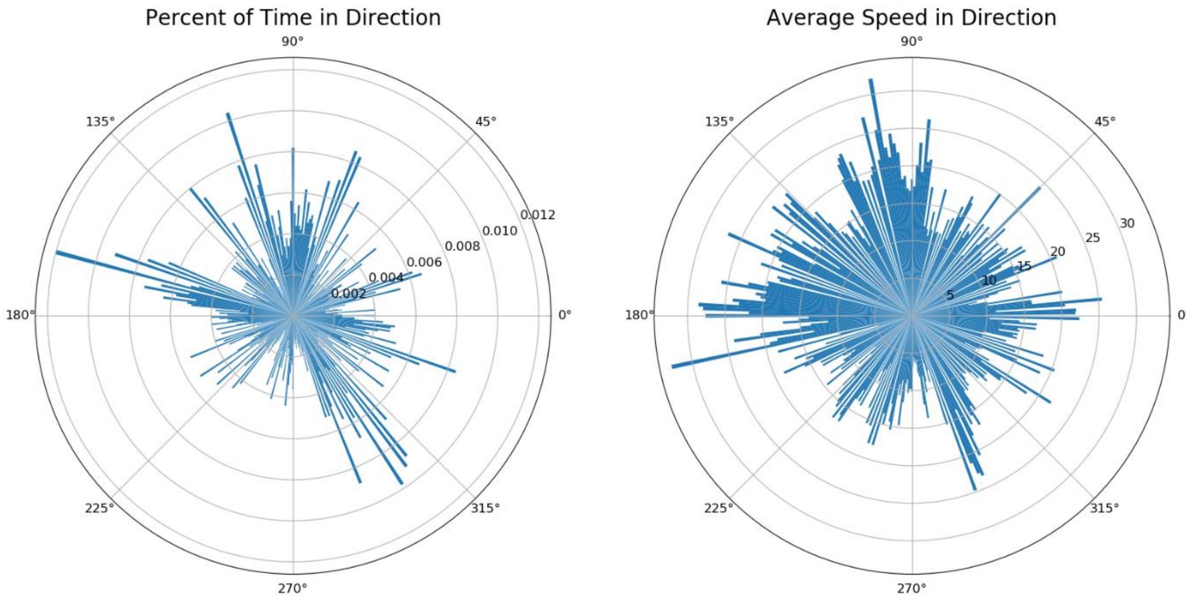

Figure 32: Wind Direction Graphs for Pseudorandom Data

As seen in the figures above, the analysis software produced the graphs as planned. In the 'Wind Speed over Time' graph there are 5000 data points and the wind speed jumps to a particular speed and stays around there for a while when moving to a new wind speed. The 'Percent of Time in Direction' graph also exhibits the same behavior of staying around some general directions with slight variations around it. From these graphs, it is apparent that the analysis software is properly displaying the data from the text file it is reading.

Another important consideration for the analysis software is how efficient it is. Python is not as fast as other programming languages, such as C, because it is an interpreted language rather than a compiled language. It is important to use efficient algorithms so the program will complete quickly, especially with large datasets. The most complex algorithm used in the analysis software is a sorting algorithm with a time complexity of $n \log_2 n$. In the final test, 30 hours of data was analyzed in 9.3 seconds. Assuming that a day of data is analyzed in 8 seconds, extrapolating the test length out to a year would take approximately an hour and 23 minutes.

4.5 Final Device Testing Results

4.5.1 Waterproof Testing

The first test performed with the assembled device was to ensure that it was waterproof. This was important because further testing would have to be done outdoors, meaning that it would be exposed to rain and wind. Thus, if it was not tested and verified to be watertight, the device could have been damaged by water seeping into the enclosure during the final testing.

To conduct the waterproofing test, the electronics were removed so they would not be damaged. Paper was placed in the enclosure so that if water leaked in during the test, it could be seen. Then the device was exposed to conditions similar to heavy rain for several minutes, using a standard shower, shown below:

Figure 33: The Waterproofness Testing Environment

The paper was checked for signs of wetness following the shower. There was some leakage in the branch of the enclosure with the anemometer on it, but this was because the wire insert had not been properly tightened. Additionally, this water did not reach any exposed electronics, so even if there was slight leakage in that branch, it was demonstrated that it would not be an issue. The enclosure that houses the batteries and other sensitive electronic components was completely dry. Therefore, the housing was determined to be sufficiently watertight.

4.5.2 Basic Systems Testing

After the enclosure was tested for waterproofness, the electronics were reintroduced to the system for basic systems testing. This included installing the embedded system PCB, power delivery PCB, and lithium-ion batteries back into the plastic enclosure. All of the devices were connected using the headers, readying the system for testing. A series of tests were devised to ensure that the entire device was working together properly in the final enclosure. These tests were performed were as follows:

- Solar panel begins charging device in moderate sunlight
- Operation of device solely from battery power (absence of sunlight)
- Writing data to the microSD card
- Calibrating the directions of the magnetometer

The first test performed was to verify that the solar panel was charging the li-ion batteries when exposed to direct sunlight. To complete this test, the system was brought outside on a sunny day. When the solar panel was in direct sunlight, the red LED would turn on and when the panel was removed from direct sunlight, the light would turn off.

The second test was performed to verify that the batteries could power the embedded system and anemometer while the solar panel was not functional. This test was completed by shielding the solar panel from light, while the embedded system was left on. At first, the embedded system would turn off at random intervals. With further investigation, it was determined that the contacts that connect the batteries to the power delivery board were loose, causing the random shut-offs. This problem was remedied by soldering new wires between the battery cases and the power delivery board. A switch to turn the power from the batteries on and off was also added. After the changes, the embedded system remained powered on for the duration of the test.

The third test was performed to verify that the data was being properly written to the microSD card. This was done by powering on the embedded system with the microSD card in the reader. While the device was powered up, the anemometer was spun up and slowed down while the wind vane was also turned to various different directions. When the test was concluded, the embedded system was powered down and the microSD card was removed from the enclosure. The microSD card was then plugged into an external computer and the contents of the .txt file were opened using a text editor. The file was scanned to make sure that it followed the general format of the test, verifying that the microSD card was properly saving data.

The fourth test was performed to calibrate the directions of the magnetometer as it relates to the orientation of the enclosure. Once the magnetometer and wind vane were mounted to the top of the enclosure, their position would be fixed, so the cardinal directions (0°, 90°, 180°, and 270°) could be determined. To determine the cardinal directions, the wind vane assembly was mounted to the top of enclosure and the embedded system was turned on. The vane was set to be facing up for 5 seconds, then right for 10 seconds, then down for 15 seconds, then left for 20 seconds. The data was then run through the analysis software. The software produced graphs that allowed the group to determine which direction corresponded to each angle. The angles were then marked on the assembly for mounting purposes.

4.5.3 Testing Location and Strategy

For final testing of the assembled device, it was essential to ensure that the final enclosure was working properly, and that the device was capable of measuring wind data at a specific location. The ideal spot for a test to be conducted would be an elevated location with few obstacles obstructing air flow. The original spot considered for the test was the chimney of the SAE fraternity house at WPI. Due to safety concerns, the test location was moved to the top of the fire escape staircase of the SAE fraternity house. This location is not as ideal as the chimney because it is lower to the ground and its proximity to the building, which would potentially result in less accurate data. The main goal of the test, however, was to test if the device is functional and not for determining the viability of a wind turbine, so these concerns will not affect the integrity of the test.

With the test location determined, the test procedure was performed. The device was mounted using pipe clamps so that it would extend above the railing of the fire escape stairwell. The device was then aligned to magnetic north using the iPhone Compass application so that the solar panel would be facing south. The time the device was powered on was recorded and then the device was sealed up and left for a period of days.

4.5.4 Full Systems Tests 1 and 2

The first test of the system was conducted from Wednesday 3/13/19 to Friday 3/15/19. The device was installed using the procedure discussed above and was checked twice a day to ensure that it was still mounted properly. When the device was taken down on Friday, the microSD card only had 4 seconds of data on it. The 4 seconds of data was due to a failure in the coding of the embedded system that caused the datafile to be overwritten whenever the embedded system powered on. Due to this coding error, the group thought that the device might have temporarily regained power after it was turned off, clearing the previously logged data. Therefore, the error in the coding was rectified so that the datafile would not be overwritten when the device powered up. The device was then prepared for another test.

The second test was conducted from Friday 3/15/19 to Sunday 3/17/19. The device was set up in the same fashion as the first test. When the device was taken down and the data retrieved from the microSD card, the datafile only had 40 minutes of good data on it, followed with approximately 15 minutes of 3-6 second intervals of data. From the data, it was determined that the device lost power after the first 40 minutes of data collection and intermittently turned back on when the battery voltage increased to a certain threshold designated by the UVLO circuitry. The purpose of the UVLO operation is to perform a hard cutoff of power when the battery terminal voltage falls to 3.0V, as to prevent brownouts in the switching converters. The sequence of operations for UVLO is as follows: The UVLO activates once the voltage drops below a certain level, power is cut off, and the terminal voltage starts climbing due to charging and time-dependent open circuit voltage. With the combination of these two effects, the terminal voltage will eventually reach ~3.5V and the power will be turned back on as the UVLO is deactivated (0.5V of hysteresis). The process repeats until the batteries are properly charged. The ULVO worked as it should, but the error in this test was due to the batteries being undercharged at the start of the test. To avoid this error from occurring again, the batteries were charged up to the recommended maximum terminal voltage of 4.2V before the next test.

Additionally, during the second test there was a thunderstorm with strong wind and rain that went through Worcester. While the device did not get any data from the storm because the batteries were undercharged, the electronics survived the weather and were able to reboot with no issue when power was restored. This served as a more real-world scenario for the weatherproofing of the device, so the second test was not a failure because it provided meaningful feedback.

4.5.5 Final Full System Test

The final test started on Monday 3/18/19 at 12:58:20 and concluded on Tuesday 3/19/19 at 19:06:56. This test retrieved over 30 hours of wind data. While the data was generally good, the magnetometer data recorded in the last 11 hours flipped between a single arbitrary point and the correct data. This anomaly was discovered when the data was imported into the analysis software, where it could be seen that the device spent over 17% of its time in the 0° direction, while the rest of the directions are barely visible. This can be seen in the following graph:

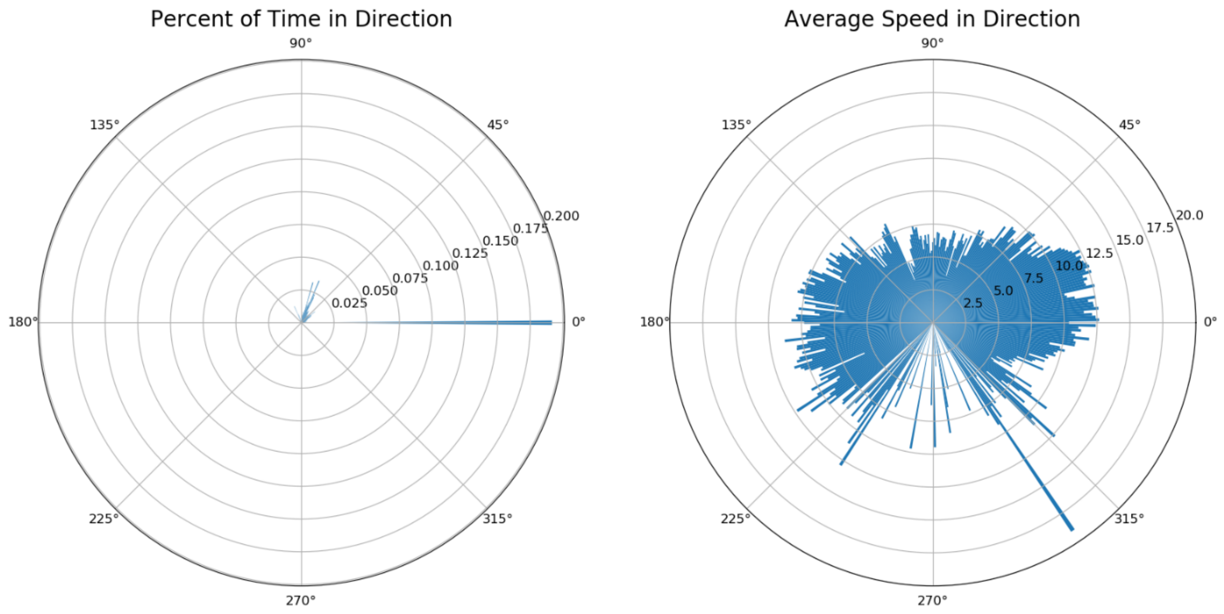

Figure 34: Wind Direction Graphs with 0° Anomaly

After looking through the datafile, the position where the anomaly started occurring was isolated and a new cropped datafile was created with all of the data up to the point where the anomaly began. This new datafile was processed using the analysis software and the following graph was produced:

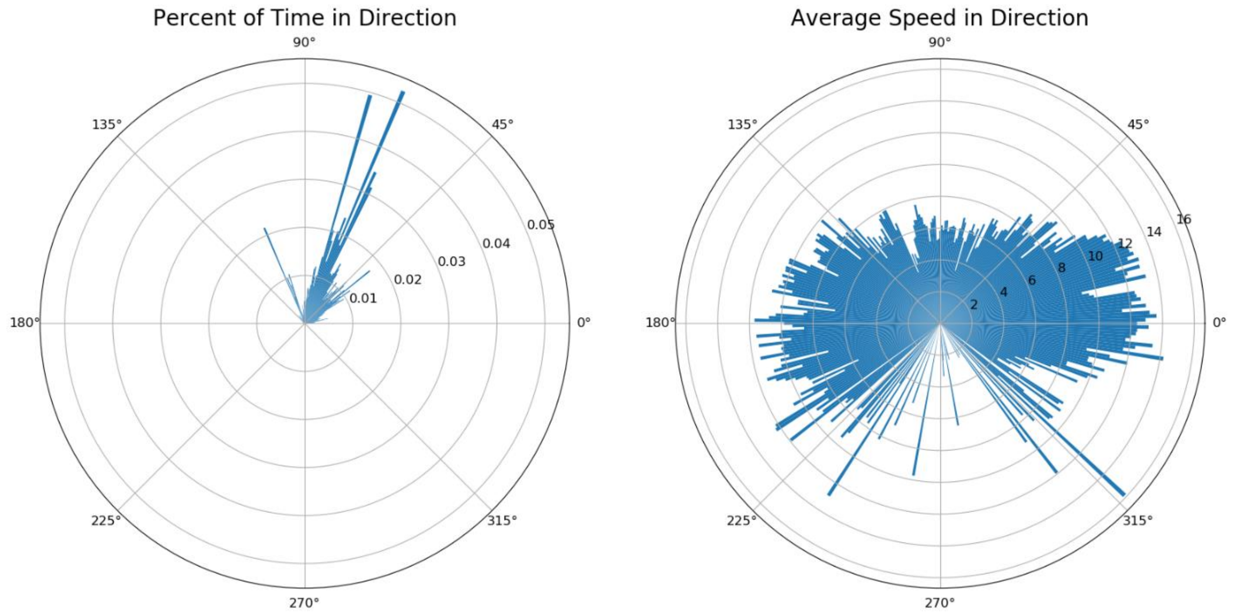

Figure 35: Wind Direction Graphs with Anomaly Removed

From the cropped dataset, it is apparent that the wind came primarily from the ~60° direction with a minimal amount coming from the other directions. As seen in the “Average Speed in Direction” graph, the wind generally averages 8-10 mph. The average wind speed was lower in the range of 225° thru 315° because this was the direction of the house, which explains why there was no wind coming from those directions. The other graph produced from the analysis software is the "Wind Speed over Time" graph. Unlike the wind direction data, the wind speed data was not affected by the wind direction anomaly, so the complete dataset can be used.

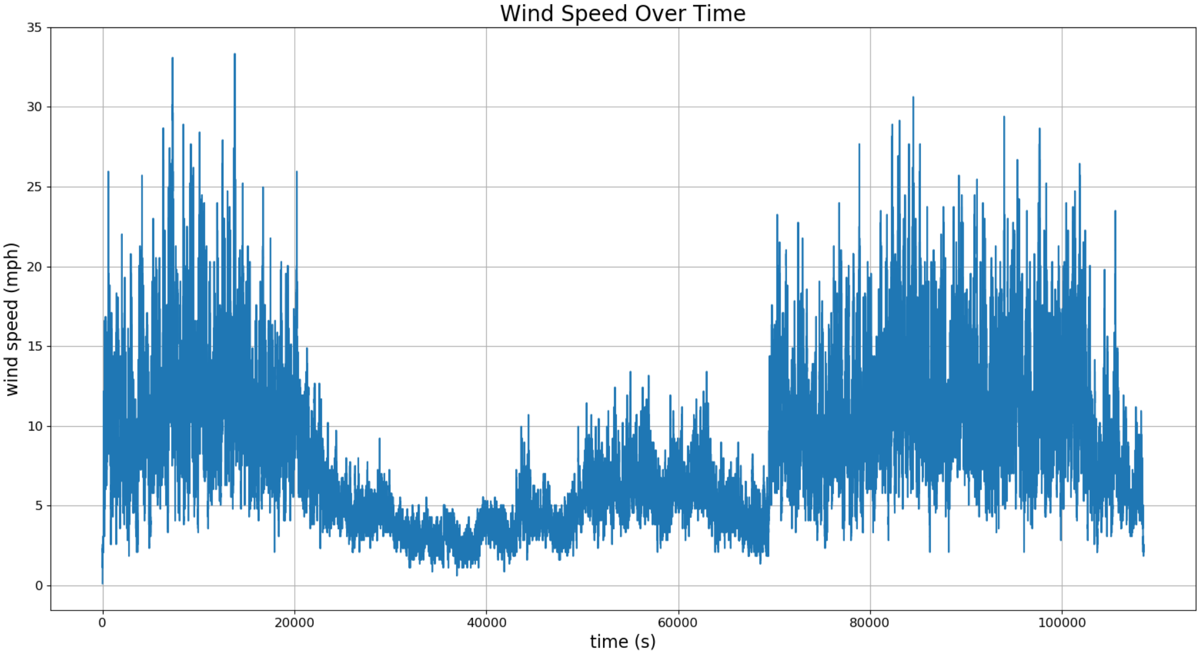

Figure 36: Wind Speed over Time Graph for Test 3

The data in the figure above shows that the average wind speeds were much higher during the day than at night.

Overall, the final test was a success. The device was able to stay powered up and recording data for over a day, and it generally the recorded the data properly. The magnetometer anomaly was unexpected, but further testing should be able to resolve the issue. In this particular case, a slight modification of the analysis software would be able to solve the direction issue. The analysis software also worked flawlessly in processing real data produced by the device. Everything observed in the graphs makes sense based on the physical environment of the test.

Section 5: Conclusions and Recommendations

5.1 Lasting Impacts of the Device

The working prototype this project produced shows promise in the creation of low-cost options for wind parameter measurement. The data gathered can be used to determine the power production of a wind turbine for a specific location. This addresses a need identified by market research, as it would allow wind turbine companies to better understand site-specific wind characteristics of an area. This is essential for the current wind market because it allows for prediction of power production, which in turn can predict revenue generation. This increases the predictability of wind power return on investment to be more competitive with solar power. By expanding different options for renewable energy, this device can help support a future with more clean energy.

5.2 Future Recommendations

5.2.1 Design for Manufacturability

If this project becomes a consumer product, it will need several changes in order to lower manufacturing cost, increase durability, and create a more appealing product.

The most obvious part to change is the magnetometer, due to cost. The current one costs around $20, and has many additional, unnecessary features. There are other magnetometers that cost around a dollar which would still work in the product. The current magnetometer was chosen to simplify development, but now that the concept has been proved, a cheaper magnetometer could be integrated into the product, significantly lowering the final cost.

Similarly, the Teensy could be exchanged for something cheaper by integrating the hardware of the Teensy LC into the other circuitry, instead of using the pre-built daughter board. The Teeny LC is an open-source project, and full schematics are available online to clone. The final PCB solution would ideally have the embedded system components integrated into a single PCB, alongside the power processing circuitry and other peripheral devices. Although this would initially create a large bill of materials, care can be taken to substitute similar components as to reduce unique component count. Full integration reduces costs, simplifies assembly, and reduces Electro Magnetic Compatibility (EMC) issues.

Additionally, the wind vane would need to be altered if the product was to be mass produced. Currently, it is made of 3D printed plastic, which is useful for prototyping, but is expensive and not durable. The wind vane could instead be made using injection molding, which is less expensive for high volume production, and creates a more durable end product [27].

The housing would also be changed for a production scale product. Currently, the device is housed in weatherproof boxes and metal piping, which protects the components from adverse weather. However, in a production unit, it would be better to have it housed in a custom housing to decrease cost and make the product appear more appealing to the end user.

5.2.2 Addition of Altitude Adjustment

One feature that could improve the project would be the addition of a way to measure wind speed at a variety of heights. For this feature, either the device could be attached to say a weather balloon and have an anchoring point with a motor that adjusts the altitude at which the device is sampling wind speeds. An alternative to this feature would be a telescoping pole that is anchored at ground level, controlling the altitude of the wind speed sampling device. A third alternative would be to have the device capable of moving up and down on a fixed pole.

The wind speed sampling device would have to be able to sense its altitude or be in communication with its anchoring point in order to obtain data points for varying altitudes and the corresponding wind speed at that level. This information would then be processed, producing a visual representation of the different average wind speeds at varying altitudes, allowing easy interpretation by the wind turbine company and potential customers of what the optimal height for a wind turbine would be at that specific location.

This feature was not implemented due to several limitations. First, none of the team has had significant experience working with mechanical systems. This meant learning many topics much outside the team’s expertise, not necessarily related to electrical engineering, would have to be done. Additionally, there was not enough time in the project schedule to add this feature. The project schedule was already full, so even if there were experts in this kind of work that did not have to learn how to implement the system, there would be no time for the team to execute this portion of the project.

5.2.3 Proposed System Changes

Although this project is finished, the device successfully created was a prototype, so there are many existing systems that can be refined to improve the design in the future if desired by the team.

There are a few component changes that would improve performance and final assembly. First, the solar panel should have a higher operational voltage. Although the solar panel used in the final prototype is rated at 2.5 Watts, that full power production potential is not met. Having a higher voltage coming from the solar panel allows for power capture in lower-light conditions. There will be less current produced, but the battery will be charging more often. The charging controller used in the power circuitry would act accordingly to limit current being sunk from the solar panel while not reducing voltage to a point where charging is inhibited.

Another hardware change would be the usage of corrosion-resistant, heavy-duty connectors to connect external cables to the fully integrated main board. The male / female through-hole header pins used in the prototype are not vibration hardened and the contacts are not gold or bronze phosphor plated. These connections would most likely not withstand slight humidity leakage into the device that could occur during long term operation, since the device is not hermetically sealed. Also, hard gusts of wind would impose mechanical disturbance to the connectors currently used, possibly shaking them loose, rendering the device inoperable. Ideally, latching connectors designed for high humidity and high temperature environments would be used whenever a board is connected to an external cable. Many companies offer such flexible solutions, including Molex, Amphenol, and TE Connectivity.

The watertight enclosure used to house the electronics was not specifically designed for board mounting. To construct the prototype, the team had to use metal-to-plastic bonding epoxy to secure the steel hex standoffs of the PCBs to the plastic frame of the enclosure. This is not an ideal connection for a vibration hardened device. Many enclosures designed for board mounting have pre-drilled/threaded holes that a PCB can be directly attached to, without the need to drill a hole through the enclosure. The PCB would have to be redesigned around the given dimensions of the mounting holes, so the design flow of the PCB would take a concurrent / iterative approach where the PCB and enclosure would have to be taken into account simultaneously.

A final device would most likely have an included a kit designed to mount the device. This would alleviate the end user from having to find applicable mounting hardware for this device and would ensure that the group could monitor quality of the entire final solution. This extra hardware would likely have a customizable-height mounting pole, a wide base that can be dug into the ground, and be easily assembled and disassembled for efficient shipping.

5.3 Lessons Learned

Over the course of the project, several areas of the project were identified that could be improve upon.

The first area was scheduling the project. It would have been beneficial to create a Gantt chart at the start of A term, so that a better understanding of the timing of certain tasks and how compensation for delays that arose could be gained, in addition to providing a clearer guideline on what tasks were of the highest priority. There were also tasks which took significantly longer to complete than expected, but this is unavoidable, and can only be improved with added experience.

Additionally, keeping better track of design changes could have posed as a benefit, especially how this would impact the rest of the system. This came up when the anemometer was changed to a model that ran on 9V, but it was not realized that the power supply board had not been changed to account for this since the initial idea for the anemometer was that it would operate off of 5V.

Works Cited

[1] M. Froese, “How to choose the best wind-monitoring device for project financing,” *Windpower Engineering & Development*, 15-Aug-2018. [Online]. Available: https://www.windpowerengineering.com/projects/site-assessment/how-to-choose-the-best-wind-monitoring-device-for-project-financing/.

[2] “Renewable Energy Explained,” Renewable Energy Sources - Energy Explained, Your Guide

To Understanding Energy - Energy Information Administration, 13-Jul-2018. [Online]. Available: https://www.eia.gov/energyexplained/?page=renewable_home.

[3] “Renewable Energy Explained,” Renewable Energy Sources - Energy Explained, Your Guide To Understanding Energy - Energy Information Administration, 13-Jul-2018. [Online]. Available: https://www.eia.gov/energyexplained/?page=renewable_home.

[4] C. Maurer, “50 Interesting Facts About Solar Energy,” Solar Power Authority, 25-Feb-2019. [Online]. Available: https://www.solarpowerauthority.com/50-facts-about-solar-power/.

[5] J. Wettengel, “Volatile but predictable: Forecasting renewable power generation,” Clean Energy Wire, 26-Nov-2018. [Online]. Available: https://www.cleanenergywire.org/factsheets/volatile-predictable-forecasting-renewable-power-generation.

[6] H. Roberts, “WHERE THE WIND BLOWS – ONSHORE VS. OFFSHORE WIND ENERGY,” The Future of Energy, 06-Apr-2016. [Online]. Available: http://futureofenergy.web.unc.edu/2016/04/06/where-the-wind-blows-onshore-vs-offshore-wind-energy/.

[7] “HILTON FT. LAUDERDALE - V-Air,” V, Aug-2013. [Online]. Available: http://www.visionairwind.com/hilton-ft-lauderdale/.

[8] R. Gaughan, “How Much Land Is Needed for Wind Turbines?,” Sciencing, 02-Mar-2019. [Online]. Available: https://sciencing.com/much-land-needed-wind-turbines-12304634.html.

[9] “What measures wind speed?,” - Quality Weather Instruments & Gadgets, 14-Nov-2018. [Online]. Available: https://weatherstationary.com/what-measures-wind-speed/.

[10] “Remote Sensing Systems For Wind Measurement,” LiDAR and SoDAR Devices for Wind Energy Assessments (Remote Sensing). [Online]. Available: https://www.ammonit.com/en/produkte/sodar-lidar.

[11] WPI’s Eighth Annual Energy Symposium, Topic: “The Future Grid: Smart, Renewable, and Resilient” Worcester Polytechnic Institute, Worcester, MA, Sept. 27, 2018

[12] “What is an Anemometer and What is it Used For?,” Wind and Weather Tools, 20-Nov 2018. [Online]. Available: https://windandweathertools.com/what-is-an-anemometer-and-what-is-it-used-for/.

[13] “Pacific Sky Power Charger Wind Turbine Generator with Fin” Amazon. [Online]. Available: https://www.amazon.com/Small-Wind-Turbine-Volt-

Generator/dp/B01KY9HMK2/ref=pd_sbs_86_30?_encoding=UTF8&pd_rd_i=B01KY9HMK2&pd_rd_r=4a233190-ba15-11e8-8b29-77fe1a29c361&pd_rd_w=0vljR&pd_rd_wg=piDb1&pf_rd_i=desktop-dp-sims&pf_rd_m=ATVPDKIKX0DER&pf_rd_p=0bb14103-7f67-4c21-9b0b-31f42dc047e7&pf_rd_r=EHR5D6Y5SWS66HW8C9MQ&pf_rd_s=desktop-dp-sims&pf_rd_t=40701&psc=1&refRID=EHR5D6Y5SWS66HW8C9MQ

[14] "Cutting Edge Power Mini USB Wind Turbine Generator - Pipe mount," Amazon. [Online]. Available: https://www.amazon.com/Cutting-Edge-Power-Turbine-Generator/dp/B078ZN9XRD/ref=pd_sbs_86_6?_encoding=UTF8&pd_rd_i=B078ZN9XRD&pd_rd_r=103c1469-ba21-11e8-8732-0b3e20371f0d&pd_rd_w=M3pqk&pd_rd_wg=wkME5&pf_rd_i=desktop-dp-sims&pf_rd_m=ATVPDKIKX0DER&pf_rd_p=0bb14103-7f67-4c21-9b0b-31f42dc047e7&pf_rd_r=T3QJMYMFZ9039F1J1EER&pf_rd_s=desktop-dp-sims&pf_rd_t=40701&psc=1&refRID=T3QJMYMFZ9039F1J1EER.

[15] "Happybuy Wind Turbine 100W 12V Wind Turbine Generator Red Lantern Vertical Wind Generator 5 Leaves Wind Turbine Kit with Controller No Pole (100W 12V, Red)," Amazon. [Online]. Available: https://www.amazon.com/dp/B07C7P64VV/ref=sspa_dk_detail_1?psc=1&pd_rd_i=B07C7P64VV&pf_rd_m=ATVPDKIKX0DER&pf_rd_p=f52e26da-1287-4616-824b-efc564ff75a4&pf_rd_r=CPK9VTH648NY49A469WZ&pd_rd_wg=MwHXm&pf_rd_s=desktop-dp-sims&pf_rd_t=40701&pd_rd_w=GpfFx&pf_rd_i=desktop-dp-sims&pd_rd_r=c5e5e45d-ba14-11e8-a904-6bfe9a9fef2d#customerReviews.

[16] "SHZOND Wind Generator 400W Hybrid Wind Turbine Generator DC 12V/24V Turbine Wind Generator 3 Blades 20A Wind Generator Kit" Amazon. [Online]. Available: https://www.amazon.com/dp/B0792RS9BW/ref=sspa_dk_detail_2?psc=1

[17] "Infinite Air," Texenergy. [Online]. Available: https://www.texenergy.com/products/infinite-air/.

[18] "Seeed Technology Co., Ltd, 2W Solar Panel" Digikey [Online]. Available: https://www.digikey.com/product-detail/en/seeed-technology-co-ltd/313070003/1597-1416-ND/5488052

[19] "Seeed Technology Co., Ltd, 3W Solar Panel" Digikey. [Online]. Available: https://www.digikey.com/product-detail/en/seeed-technology-co-ltd/313070001/1597-1414-ND/5488054

[20] "ALLPOWERS 1PC 2.5W 5V 500mAh Mini Solar Panel Module Solar System Epoxy Cell Charger DIY 130x150mm (Solar Panel Only)" Amazon. [Online]. Available: https://www.amazon.com/gp/product/B00CBT8A14/ref=ox_sc_saved_title_2?smid=ATVPDKIKX0DER&psc=1

[21] "Anemometer Wind Speed Sensor w/Analog Voltage Output" Adafruit. [Online]. Available: https://www.adafruit.com/product/1733

[22] "Hot Sale Wind speed sensor 3-cup anemometer speed sensor 4-20ma" Alibaba. [Online]. Available: https://www.alibaba.com/product-detail/Hot-Sale-Wind-speed-sensor-3_60755301008.html?spm=a2700.7724838.2017115.15.74b4672bTNxrrw&s=p

[23] “RK100-01 304 SS High Strength Wind Anemometer for Wind Generator” Alibaba. [Online]. Available: https://www.alibaba.com/product-detail/RK100-01-304-SS-High-Strength_60598581600.html?spm=a2700.7724838.2017115.95.4d128f6fCSGD60

[24] T. Joseph, “Arduino Nano V3.0 Specification,” TOMSON ELECTRONICS, 17-Oct-2017. [Online]. Available: https://www.tomsonelectronics.com/blogs/news/arduino-nano-v3-0-specification.

[25] “Teensy LC - Low Cost,” PJRC. [Online]. Available: https://pjrc.com/teensy/teensyLC.html.

[26] “Raspberry Pi Zero: the $5 computer,” Raspberry Pi, 26-Nov-2015. [Online]. Available: https://www.raspberrypi.org/blog/raspberry-pi-zero/.

[27] T. Rogers, “3D Printing vs Injection Molding,” 3D Printing vs Injection Molding, 24-Nov-2015. [Online]. Available: https://www.creativemechanisms.com/blog/3d-printing-vs-injection-molding.

Appendices

Appendix A: Embedded_System.ino

```
#include <SD.h>
#include <SPI.h>
#include <Wire.h>
#include <SparkFunIMU.h>
#include <NXPMotionSense.h>
#include <Wire.h>
#include <EEPROM.h>
#include <util/crc16.h>
#include <math.h>

struct packetStruct {
  long pointID;      //32bit - The value of the point
  int windSpeed;     //16bit - The ADC reading from the anemometer
  int windDir[2];    //16bit - The x and y readings from the magnetometer
};
typedef struct packetStruct PacketStruct;

File myFile;
NXPMotionSense imu; // Setup for the prop board
const int chipSelect = 10;

long idNum = 1;
int AnemPin = 23;


void setup() {
  pinMode(AnemPin, INPUT);
  if (!SD.begin(chipSelect)) {
    return;
  }
  imu.begin();  // Initialization for the prop board
}

void loop() {
  PacketStruct testPacket = createPacket(idNum);
  writePacket("datafile.txt", testPacket);
  idNum++;

  delay(1000);
}

// Read the anemometer value
int getAnemReading(int pinNum) {
  return analogRead(pinNum);
}
```

```
// Creates the packet, with data in it, to be sent
PacketStruct createPacket(long id){
  int ax, ay, az, gx, gy, gz, mx; // Unused filler variables
  int my, mz;   // Useful variables

  if (imu.available()) {  // Read everything from the prop board
    imu.readMotionSensor(ax, ay, az, gx, gy, gz, mx, my, mz);
  }

  PacketStruct retPacket;
  retPacket.pointID = id;
  retPacket.windSpeed = getAnemReading(AnemPin);
  retPacket.windDir[0] = my;
  retPacket.windDir[1] = mz;

  return retPacket;
}

// Sends the packet to the SD card
void writePacket(char* filename, packetStruct packet) {
  myFile = SD.open(filename, FILE_WRITE);
  if (myFile) {
      //Write the contents of the packet to the binary file
      myFile.print(packet.pointID);
      myFile.print(",");
      myFile.print(packet.windSpeed);
      myFile.print(",");
      myFile.print(packet.windDir[0]);
      myFile.print(",");
      myFile.print(packet.windDir[1]);
      myFile.println(":");

      myFile.close();
    } else {
       //packet not written
    }
}
```

Appendix B: Analysis_Software.py

```
#use this file to read and process random binary data
import numpy
import math
import matplotlib
import matplotlib.pyplot

#open file #this is the first comment
f = open("DATAFILE.txt", "rt")
data1 = f.read()    #store text file data in string
#print(data1)

#create arrays to store the speed, direction, height, and identity
arrayLen = data1.count(":")
identArray = numpy.zeros(arrayLen)
speedArray = numpy.zeros(arrayLen)
direcArray = numpy.zeros(arrayLen)
#print(arrayLen)

#split data into each packet, separated by :
datalines = data1.split(':')

#fill arrays with proper data
for i in range(arrayLen):
    datapoints = datalines[i].split(',')    #differentiate each point of data, split
by ,
    identArray[i] = int(datapoints[0])
    speedArray[i] = ((int(datapoints[1]) * 3.3/1024) - 0.3722) / 0.0131
    direcArray[i] = math.floor((numpy.arctan2(int(datapoints[2]), int(datapoints[3]))
/ numpy.pi) * 180)

#create the direction angle array from the magnx and magny datapoints

#sort the arrays by time, speed, and direction for graphing purposes
timeSortedArray = [x for x in zip(speedArray, direcArray, identArray)]
#speedSortedArray = sorted(timeSortedArray)
directionSortedArray = sorted([x for x in zip(direcArray, speedArray, identArray)])

#create the arrays for the circular histograms
percentDirectionArray = numpy.zeros(360)
averageSpeedArray = numpy.zeros(360)
```

```
divideNum = 1
speedSum = 0
for i in range(len(directionSortedArray)):
    if(i == len(directionSortedArray)-1):
        percentDirectionArray[int(directionSortedArray[i][0])] = 
divideNum/len(directionSortedArray)
        speedSum += directionSortedArray[i][1]
        averageSpeedArray[int(directionSortedArray[i][0])] = speedSum/divideNum

        #reset variables
        divideNum = 1 #reset the number to divide
        speedSum = 0    #reset the sum of the speeds
    elif(directionSortedArray[i][0] == directionSortedArray[i+1][0]):
        divideNum += 1  #the next element is the same direction
        speedSum += directionSortedArray[i][1]
    else:
        percentDirectionArray[int(directionSortedArray[i][0])] = 
divideNum/len(directionSortedArray)
        speedSum += directionSortedArray[i][1]
        averageSpeedArray[int(directionSortedArray[i][0])] = speedSum/divideNum

        #reset variables
        divideNum = 1 #reset the number to divide
        speedSum = 0    #reset the sum of the speeds

test1 = sum(percentDirectionArray)
print(test1)
print(percentDirectionArray)

#create figure
minimum = 0
maximum = max(percentDirectionArray)
theta = numpy.linspace(0.0, 2*numpy.pi, 360, endpoint=False)

fig = matplotlib.pyplot.figure(figsize=(8,8))
matplotlib.pyplot.subplot(121, polar=True)
matplotlib.pyplot.bar(theta, percentDirectionArray, width=(2*numpy.pi)/360, 
bottom=minimum)
matplotlib.pyplot.title("Percent of Time in Direction", loc='center')

matplotlib.pyplot.subplot(122, polar=True)
matplotlib.pyplot.bar(theta, averageSpeedArray, width=(2*numpy.pi)/360, 
bottom=minimum)
matplotlib.pyplot.title("Average Speed in Direction", loc='center')
```

```
fig2 = matplotlib.pyplot.figure(figsize=(8,8))
matplotlib.pyplot.plot(identArray, speedArray)
matplotlib.pyplot.xlabel('time (s)')
matplotlib.pyplot.ylabel('wind speed (mph)')
matplotlib.pyplot.grid()
matplotlib.pyplot.title("Wind Speed Over Time", loc='center')

matplotlib.pyplot.show()
```

Appendix C: Pseudo_Noise.py

```python
#use this file to create pseudo-random data
import random
import math

numPoints = 5000  #how many points do we want to generate

direcCoarseValue = 0    #general wind direction
direcFineValue   = 0    #small change in direction
direcChange      = 0    #value that will determine if general direction should change

speedCoarseValue = 0    #general wind speed
speedFineValue   = 0    #small change in wind speed
speedChange      = 0    #value that will determine if general speed should change

def writePointToFile(pointNum, windSpeed, windDirection):
    dirX = int(math.cos((windDirection/180)*math.pi)*10000)
    dirY = int(math.sin((windDirection/180)*math.pi)*10000)
    writeString = "{},{},{},{}:\n".format(pointNum, windSpeed, dirX, dirY)

    with open('testData.txt', 'a') as writeFile:
        writeFile.write(writeString)
    return

with open('testData.txt', 'w+') as clearFile:
    pass

for i in range(0, numPoints):
    if(direcChange == 0):
        direcCoarseValue = random.randrange(360)
    else:
        direcFineValue = random.randrange(8) - 4
        direcCoarseValue += direcFineValue

    if(speedChange == 0):
        speedCoarseValue = random.randrange(256)
        if(speedCoarseValue < 132):
            speedCoarseValue = 132
    else:
        speedFineValue = random.randrange(30) - 15
        speedCoarseValue += speedFineValue
        if(speedCoarseValue < 132):
            speedCoarseValue = 132

    writePointToFile(i, speedCoarseValue, direcCoarseValue)

    direcChange = random.randrange(75)
    speedChange = random.randrange(75)
```