Boat #2

2023 Cedarville University Solar Splash Technical Report

Joseph Heise, Joseph Hoyman, Nicolas Knowlton, Ruth Lessen, Asa Mudd,

Bryce Schmitt, Daniel Shoultz, Grace Fearday, Kezia Augusting, Cejay Walker

1 May, 2023

Project Advisors: Dr. Timothy Dewhurst and Dr. Gerald Brown

Executive Summary

The Cedarville University Solar Boat Team has a long history of success at the Solar Splash (SS) competition. To continue this tradition, the 2023 team began with the goal of winning SS in all three races – sprint, slalom, and endurance. The 2022 team was able to win the SS competition with a strong first place score in the endurance event, but only placed third in the slalom event and sixth in the sprint event. Therefore, the 2023 team seeks to improve in the slalom and sprint events through improving the motor and motor controller, steering mechanism, propeller, and boat electronics. Responsibilities have been divided by areas of work among the 2023 team members to achieve these goals.

The first area of work is the motors, motor controller, and dynamometer. Previous teams worked on dyno testing two different Hawk motors (Hawk20 and Hawk40) to quantify their performance and tune the motor controller, but this was not completed. Our team has improved dyno testing the motors by improving the alignment of the motors and installing vibration isolating mounts. Then motor data was collected and sent to the motor controller manufacturer to produce higher torques so that the boat can go faster in the SS sprint and slalom events. After motor testing, it is recommended that the team uses the Hawk40 motor for competition.

The second area of work is the thrust testing and steering mechanism of the boat. To improve the thrust data from the boat, the thrust mount was taken apart and adjusted to eliminate binding. The steering for the SS hull was improved by shortening the tiller arm length to improve the steering ability of the skipper.

The third area of work is the propeller. The 2022 team made an effective propeller for the endurance event, but the propeller they designed for the sprint event did not operate at the desired efficiency. The goal for the 2023 team was to produce a sprint propeller that achieved a high speed in a shorter amount of time. Using OpenProp, a MATLAB software which outputs an ideal blade geometry for certain operating parameters, the team designed a sprint propeller, altering the motor speed design parameters to conform more closely to the value produced by the motor testing. This design used a chord-optimization technique. When tested, this design proved ineffectual, and it was concluded that future sprint propellers should use a broader conventional blade design rather than the thinner chord optimized design.

The last area of work is the boat electronics, specifically the CAN bus, data acquisition system (DAS), and driver interface. The CAN bus is a serial bus used on the boat to communicate between the devices providing battery voltage, motor current, motor speed, and much more. Previously, the team's DAS and display system were awkward, and the tablet display broke. Both systems were improved with the addition of the AEM CD-7L CAN bus display. The improved systems have been tested and verified to be reliable and easier to use. Previous teams also only ran the boat in a backup mode where the terms of the boat operation were defined by the motor controller program. This meant relying on the manufacturer to modify the program, and it was desired that the alternative CAN mode on the controller was used. This required the development of the Boat Operating System (BOS) in a Simulink model giving the team control over how the boat operates. The BOS has been implemented and tested, but an issue with reverse remains.

The team also heavily focused on designing a hull that will fly on hydrofoils. This included hydrofoil strut design and the design and implementation of a closed-loop flight control system. However, since this work will not be seen in competition, it will not be discussed further in this report.

Notation

Notation	Meaning
RPi	Raspberry Pi
BOS	Boat Operating System
CAN	Controller Area Network
GPS	Global Positioning System
SS	Solar Splash
UDT	Universal Drive Train
DHX	Refers to the company Direct winding Heat Exchange Electrical Machines
ACS	Alternating Current Superdrive, the type of motor controller we currently use
AVL	Athena Vortex Lattice
ID	Identifier
CFM	Cubic Feet per Minute
.dbc	CAN database file used to define and interpret CAN messages
Dyno	Dynameter
CAN bus	The two-wire serial bus connecting nodes on the Controller Area Network on which CAN messages are communicated
CAN ID	A number which is used to identify a CAN message defined in a .dbc file
CAN Message	A message sent over the CAN bus related to a specific CAN ID and containing individual CAN signals
CAN Signal	An individual piece of data contained within a CAN message that is identified by its starting bit or byte within its CAN message
AEM	Refers to the company Advanced Engine Management Performance Electronics
AEM CD-7L	The 7-inch carbon digital dash with logging capability made by AEM.
VDM	Vehicle Dynamics Module from AEM which is used primarily for GPS data.
SDL	Senior Design Lab
PCB	Printed Circuit Board
dash	Dashboard
BMS	Battery Management System
EMS	Energy Management System
CU	Cedarville University
DAS	Data Acquisition System

Table of Contents

I. Overall Project Objectives

Our overall goal is to win the Solar Splash Competition in the endurance, sprint, and slalom races. Last year, the winning times for the sprint event and the slalom event were 23.86 seconds and 40.47 seconds respectively. The CU solar boat team and competition winners from 2022 won the endurance event with a total of 72.5 laps in two 2-hour races, which means the average speed was 4.14 m/s. To win the competition, the boat should meet or beat a sprint time of 23.86 s, a slalom time of 40.47 s, and 72.5 laps in the endurance event (depending on the weather).

Our objectives as the 2023 Solar Boat Team are as follows:

- Motor and Motor Controller
  - Determine which Hawk motor should be used through testing.
  - Verify peak motor performance and establish the most efficient operating conditions as design constraints for propeller manufacturing.
  - Update and keep a power budget from experimental data for the actual performance and efficiency of the boat.
  - Gather motor performance data when subjected to high load conditions.
  - Create a simple closed loop cooling system.
- Steering System and CAN Adapters
  - Improve the Solar Splash steering system by redesigning the tiller arm to obtain improved turning capabilities.
  - Design a dynamic code to determine optimal parameters of the steering system.
  - Create CAN adapters for boat functions in order that the boat may operate.
- Propeller
  - Use dyno testing data to design a propeller specific for our boat.
  - Design a sprint propeller with an efficiency of 80% at target speed and sufficient low speed thrust to rapidly accelerate to that speed.
  - Manufacture a propeller to perform similarly to the projected design performance.
- Data Acquisition, Boat Operating System, and Display System
  - Establish a data acquisition system (DAS) that can be used to provide logs of CAN messages easily and consistently such as thrust, motor speed, voltage, and current values.
  - Create a waterproof, sunlight readable, reliable, and easily customizable display system to display race information such as battery voltage, motor current, and boat speed to the skipper.
  - Develop BOS by defining how we want the boat to operate such as maximum speeds in the sprint and endurance events, and enforcing desired safeties, like not allowing motor direction to change while throttle is not at zero.
  - Implement the BOS so that the motor controller responds to CAN messages, and thus the boat can be run in the CAN mode.

II. System Design

A. SOLAR SYSTEM

Current Design: The Solar System is composed of one SP100 and three SP125 Solbian photovoltaic silicon solar panels mounted on a frame over the rear of the boat. Peak power trackers are used to convert the voltage generated to the voltage used to charge the batteries. We made no changes to the solar system this year.

B. ELECTRICAL SYSTEM

1) *Current Design*: For our electrical system, we have a battery box that houses the batteries as well as instrumentation for measuring various voltages and currents. We use Genesis 42EP batteries and Genesis 13EP batteries. Our dashboard contains switches and a potentiometer for controlling different settings and includes hardware for the data acquisition system, including a GPS for measuring boat speed. Schematics are shown in Appendix K. We only made minor changes to the electrical system this year.

C. POWER ELECTRONICS SYSTEM

1) *Current Design*: The electric motors utilized are the Hawk20 and Hawk40 motors. Manufacturer specifications indicate the Hawk20 operates at 48 V, 250 A draw with peak current of 425 A, and an efficiency rating of 95% reaching a rated speed of 4400 RPM. The 2021 Solar boat team decided to use the Hawk40 motor, which has the same current draw and speed specifications as the Hawk20 while rated at 72V and almost double the rated torque.

The DHX Hawk motors are paired with the Inmotion ACS motor controllers, which were selected by the 2020 Solar Boat team due to their efficiency and power handling. According to the 2021 team's spring proposal, the Inmotion ACS controller can operate at 97% efficiency and handle peaks of 550 A. The 2020 team designed a custom heat sink for the Inmotion controller, utilizing water cooling due to restricted room onboard the hull for airflow and forced convection (as well as water being plentiful outside the boat). These motor controllers have proven to be difficult since they require extended amounts of time to program. This past year, we focused on improving the tune of the motor controller and increasing our motor performance to produce torque at higher speeds to improve our sprint and slalom events. We have a dynamometer onsite that allows us to test and record the performance data of our Hawk motors and motor controllers. However, we had issues with vibration losses in the system when being tested.

The previous motor cooling system used a pickup tube at the level of the keel on the transom. The pump sent the water through the motor and then the motor controller, returning into the water. If flow was maintained, the motor was adequately cool because of the low temperature of the lake water. However, the lake water contained algae and particulates which once destroyed a pump and clogged the small water passageways within the motors. The pickup was also exposed to air in sharp turns which caused the pump to get airlocked and halt flow through the motor. These issues caused the motors to overheat, and the motors had to be repaired by DHX.

The cooling system for the motor and motor controller needs to be improved to prevent the motor from overheating. The new system needs to maintain flow through the small passages in the motor (~1 mm diameter) and a motor temperature below 80 °C.

2) *Analysis of Design Concept*: The Land and Sea dynamometer used to collect data on the electric motors was much louder than was reasonable to operate with. Additionally, we knew that the loud volume from system vibrations were audible power losses we were unable to measure. When disassembling the system, we could visually identify a 0.25 in. offset between the dyno shaft and the motor shaft. We then placed a dial caliper on the end of the dyno shaft and measured the differential distance of the mounting plate with respect to the dyno shaft, allowing us to identify how close to perpendicular the plate was to the shaft. We found that at its furthest point, it was as much as 80 thousandths of an inch off from true perpendicular.

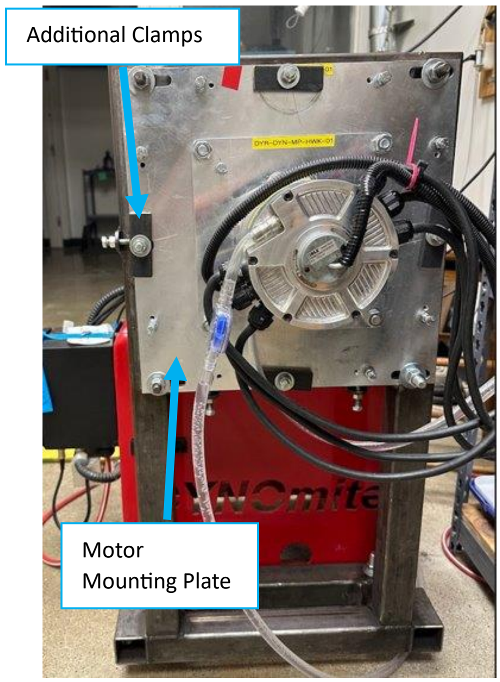


Fig 2.2. *Dyno with labeled motor receiving plate*

We decided to take a 3-step approach to improving the configuration of the dyno to reduce both noise and mechanical losses. First, we clamped the motor mounting plate to the motor frame with 4 additional clamps. These additional clamps and the mounting plate itself are visible in **Fig 2.2**. The mounting plate is made from aluminum and visible bowing could be seen between the original 4 corner clamps.

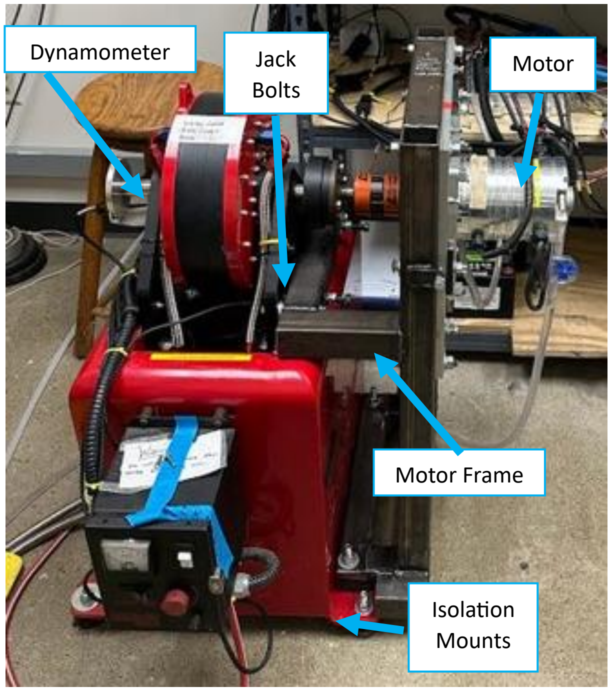


Fig 2.3. *Dyno with labeled parts*

After adding additional clamps, we moved the dynamometer to new concrete expansion anchors and added isolation mounts underneath it. We then changed the dyno's motor frame to mount directly to the dyno instead of the expansion anchors. This allowed the system to be one solid piece that was isolated from vibrating against the concrete floor. Finally, we used jack bolts between the dyno and motor frame to keep the mounting plate perpendicular to the dyno shaft. **Fig 2.3** shows the finished product after making these alignment/vibration adjustments. After placing a dial caliper back onto the dyno shaft, we were able to see an improvement from an 80 thousandth differential distance to $\pm$ 3 thousandths across the face of the motor mounting plate. We then used a straight edge to align the motor shaft and dyno shaft axially.

To prevent damaging motors in the future, we needed to revise the cooling system to reliably keep the motor temperature below 80°C. Considering that the water passages through the motor are very small, we did not want to circulate lake water containing algae and particulates through the repaired motors. We also needed a constant water supply to avoid air locking the pump. Thus, we chose to change to a closed loop cooling system. Then we had to decide how to maintain low motor temperatures. We considered a water-to-water heat exchanger that would attach to the keel of the boat, but we wanted to keep it simple, and the design needed to work on both the SS and hydrofoil hull. It was also difficult to calculate what exactly the heat load is on the system considering we do not know the efficiency of the motor and controller. To keep things simple, we decided to transfer the heat out of the motor into some amount of water that would keep the DHX specification of a maximum motor inlet water temperature of 55 °C.

To calculate the amount of water required, we estimated the amount of energy released in the form of heat by the motor and motor controller in a sprint race to be about 83.6 kJ. We then calculated how much water was needed to absorb that amount of energy, starting at a temperature of 30 °C and without exceeding 45 °C, to give us a reasonable safety factor. We found that we needed at least 1.34 L of water in our reservoir from this calculation. Based on this, we purchased a 2 L reservoir that was suitable for our needs. We also purchased a small heat exchanger as an extra precaution, to expel heat into the air. While we waited to receive the reservoir, we used a gallon milk jug (3.79 L) as a reservoir and found that the motor did not overheat while testing in the sprint configuration. Once we received the reservoir and heat exchanger, we built a hanging bracket mount and plumbed the system together.

3) *Design Testing and Evaluation*: The main objective for the motors was to collect data that could be used to tune the motor controller as well as determine torques and speeds for propeller designs. When laying out testing parameters, we decided on a series of constant speed max-torque tests to give a full range of performance ability for the motor/motor controller combination. Starting at 500 RPM, we held the speed steady and incrementally increased the torque in steps of a few N-m until the motor could not hold its speed any longer. We then repeated this process in 500 RPM increments, ending at 4000 RPM. We utilized PyCAN logger and the dyno software to collect the data for these runs. PyCAN allows for CAN message recording that Inmotion can read to inform tuning decisions while the dyno software was able to export each run's data to an Excel format. We then wrote a MATLAB code where we could quickly import the data and have a standardized plot to easily view the data.

Theoretically, we should be able to produce rated torque up until our field-weakening point. This field weakening point is normally the rated RPM of the motor; however, due to our voltage limitations, it begins at a much lower RPM. The Hawk 40, when subjected to 36 V will begin field weakening at 2200 RPM while the Hawk 20 will begin at 3400 RPM. **Fig 2.4** shows the results of a 3000 RPM test of the Hawk 20. Using theoretical values, we should see the ability of the motor to hold 40 N-m here, however the motor begins to go unstable and therefore cannot reliably produce even 35 N-m of torque.

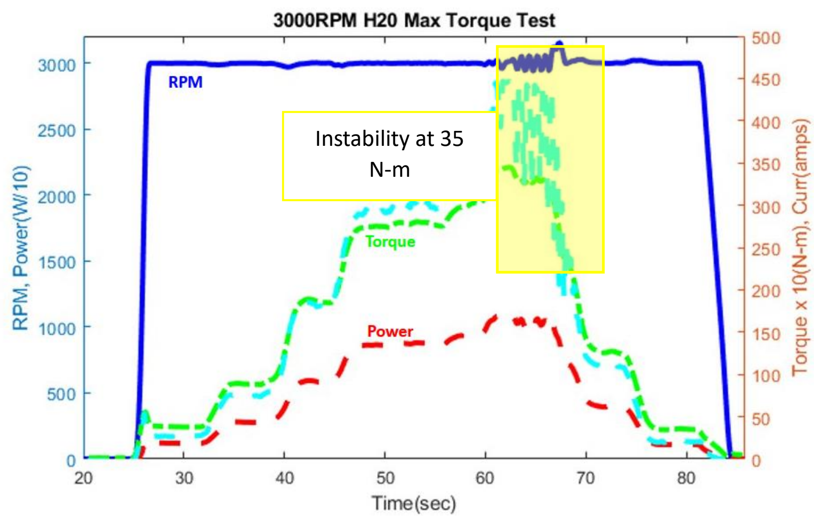


Fig 2.4. *Data collected from the 3000 RPM run of max. torque test of the Hawk20*

Fig 2.5 shows similar results occurred with the Hawk40 at 2000 rpm. The theoretical value for torque should be 67.5 N-m, but the motor could not even maintain 45 N-m of torque.

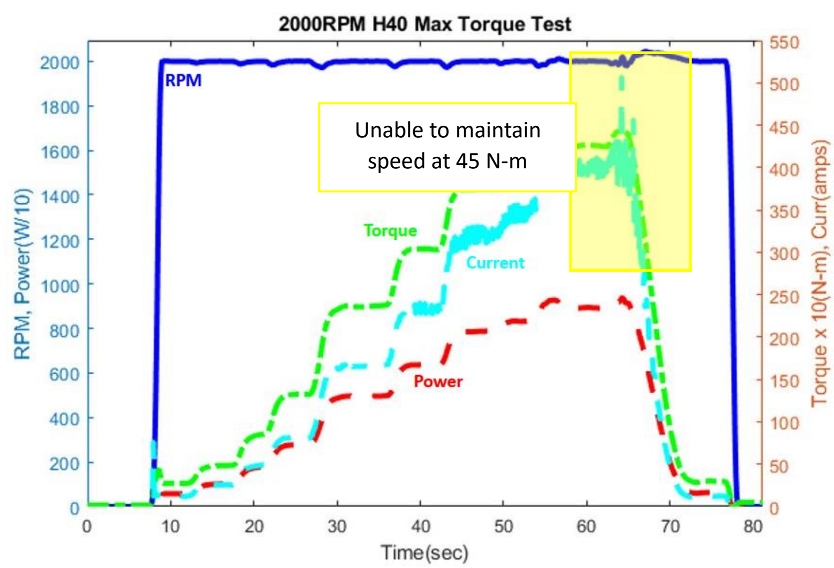


Fig 2.5. *Data collected from the 2000 RPM run of max. torque test of the Hawk40*

Ultimately, the performance of the Hawk motors is currently not where we know they could be. Having this data collected, though, is further than our teams have been able to go in the past. We can use this data to talk with Inmotion about tuning the motor controllers to achieve our needed Hawk motors' performance. As of this report, we have been able to send the data over to Inmotion and are awaiting their response to assist us in the tuning of the motor controllers.

After installing the new reservoir and heat exchanger, and plumbing the system together, we tested the boat in the sprint configuration on April 14th. During our test we monitored the motor temperature with the display and downloaded the log from the Raspberry Pi afterwards.

The plot of the motor current and temperature (**Fig 2.6**) shows that the motor temperature did not exceed 65 °C in high motor current conditions for 250 s, verifying the adequacy of the new cooling system since the motor temperature did not exceed 80°C as specified by DHX.

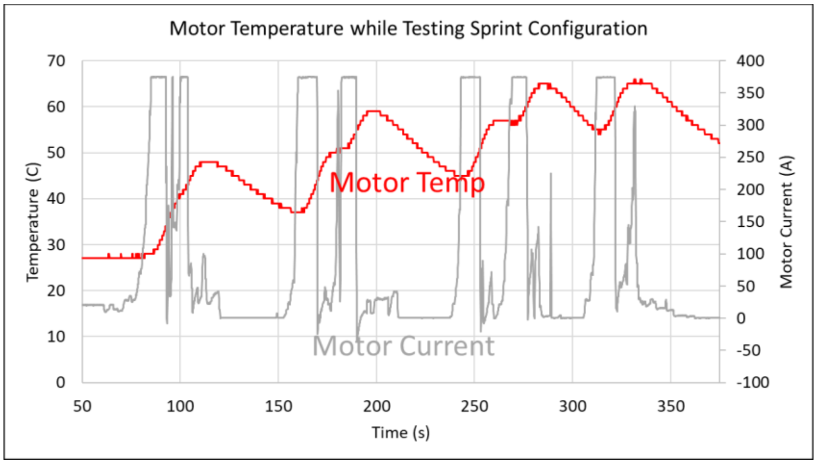


Fig 2.6*. Motor temperature and current data plotted against time from April 14th testing in sprint configuration to verify cooling system performance.*

D. HULL DESIGN

1) Current Design: Previous teams have already performed extensive work optimizing our hull. We will be using the fiberglass, foam, and resin style hull that was developed in 2005 for the Solar Splash race. This design reduces the total weight of the hull to 63 lb by using a combination of a gel coat, fiberglass, resin, catalyst, and foam. The basic components and layout of the Cedarville University Solar Splash boat are as shown below in Fig 2.7.

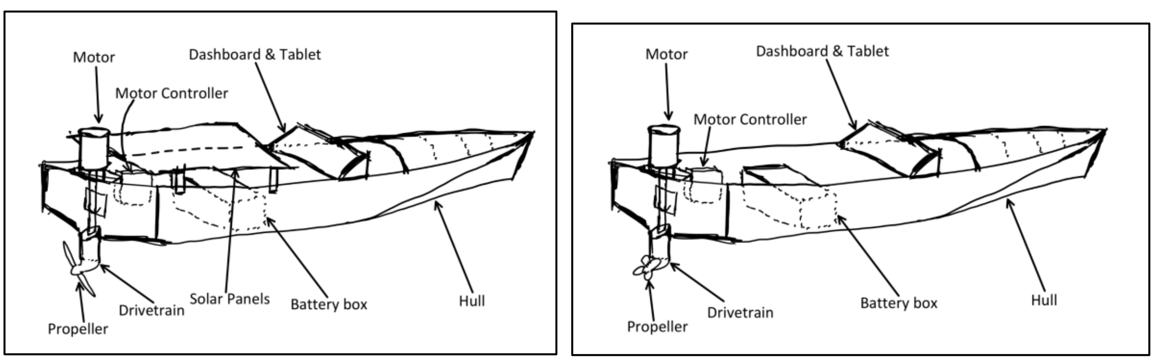


Fig 2.7. Solar Splash boat in endurance configuration (left), and sprint and slalom configuration (right).

The hull has a tapered bow like a canoe that gives the boat low drag and good stability at low speeds when the hull is mostly in the water. The low drag results in more speed from input power, allowing for success in the Endurance event. At the high speeds of the Sprint and Slalom events, the front lifts out of the water and the boat behaves as a planing hull, which reduces drag and maintains handling. This is an effective combination for high-speed events as speed and agility are needed for success. No changes were made to the hull this year.

E. DRIVETRAIN AND STEERING

1) Current Design: The drive train for Solar Splash is composed of the drive motor, motor mount, down-shaft, driveshaft, gearbox, transom mount, tilt assembly, and propeller. The 2021 Solar Splash team manufactured and used a single drive train system for both endurance and sprint races, leading to weight savings of 90 lb.

The SS boat has two propellers—one for endurance and one for sprint/slalom, as well as a pair of contra-rotating propellers. The propellers' efficiency at their design speeds influences the power budget (the total power transferred to drive the boat forward). The sprint propeller is 75% efficient, and the endurance and contra-rotating propellers are 89% efficient. We wanted to design and manufacture a more efficient sprint propeller with better acceleration at lower speeds.

Additionally, the steering system was cumbersome. The skipper would have to pull and remove their hand from the steering cable several times to turn the boat from left to right. This led to a large cable displacement—the amount of cable displaced when the tiller arm turns from left to right (a range of 70°). The SS boat had a cable displacement of 45" completed in two to three hand pulls. There were also design issues with the tiller arm.

2) Analysis of Design Concept: We designed the propellers with OpenProp. We then transferred propeller designs to SolidWorks and milled them on the CNC machine. The blade designs created in OpenProp use the parameters of boat speed, motor speed, boat drag, and several geometric parameters. The boat parameters were already determined by the specifications of our hull and motor, but we had a range of choices for the geometric parameters: blade length, number of blades, and whether or not the blade was chord optimized. Chord optimized propellers have thinner blades with higher efficiencies at their projected operating speed, but from our tests, they appear to have less thrust at lower speeds. After we determined these parameters and generated a design in OpenProp. We transferred it to SolidWorks using a Macro and then we used HSMWorks to design toolpaths for milling the propeller. (For a more detailed analysis of determining design parameters see **Appendix G**). We also analyzed the SS steering system to improve its turning speed and usability.

To determine the optimum propeller size and number of blades, we iterated OpenProp designs, varying the diameter and number of blades until we determined the most efficient values: 2 blades, 0.3-meter diameter (12 inch). We did this as shown in ***Fig 2.8***.

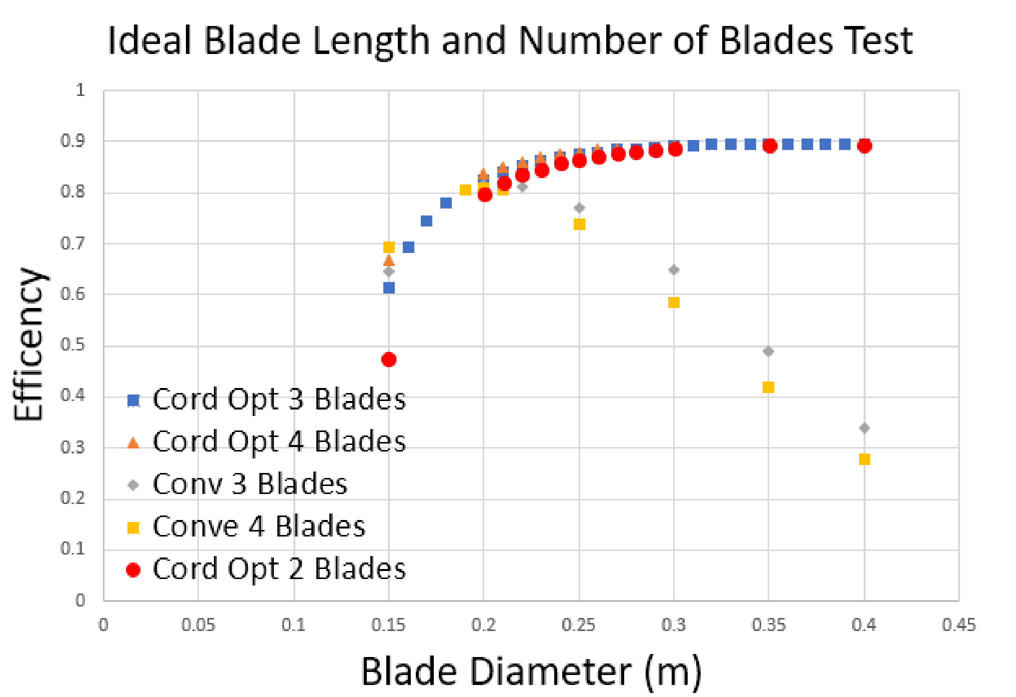


Fig 2.8. *Propeller iterative design record*

We based the projected boat speed on our desired competition speed: 15.6 m/s (35 mph). The motor speed came from last year's race: 3770 RPM. Since we did not have drag data for the SS hull at this point, we used a power budget (recorded in **Appendix G**) to estimate the boat's drag: 800 N (182 LB).

When manufacturing propellers, we ran into two major challenges: deflection of the blades during milling and maintaining zero during flipping. To prevent deflection of the propeller blades during milling, we used parallel tool paths starting at the tip of the propeller and working in towards the hub. This kept additional material between the tool head and the sections of stock which were fixed to the CNC bed. We maintained zero during flipping the part by using pins to keep the stock properly aligned. When we drilled the first pin hole, we set the part zero at the center of it and kept it there in between operations to prevent any error in re-zeroing the part during milling.

For the cumbersome steering system (**Fig 2.9**), we redesigned the tiller arm to reduce the cable displacement and the number of times the skipper must regrip the cable to rotate the tiller arm across its range (**Fig. 2.10**). The maximum and minimum tiller arm angles correspond to the sharpest left and right turns, respectively.

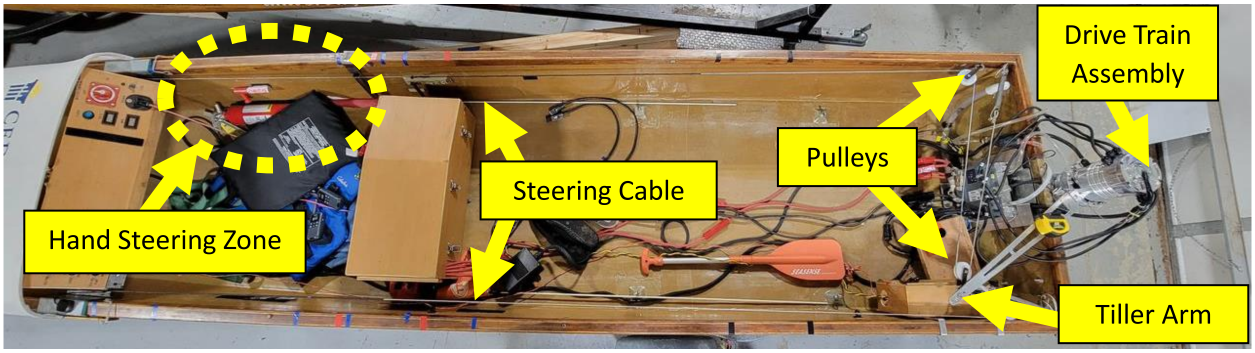


Fig 2.9. *Solar Splash boat steering system components*

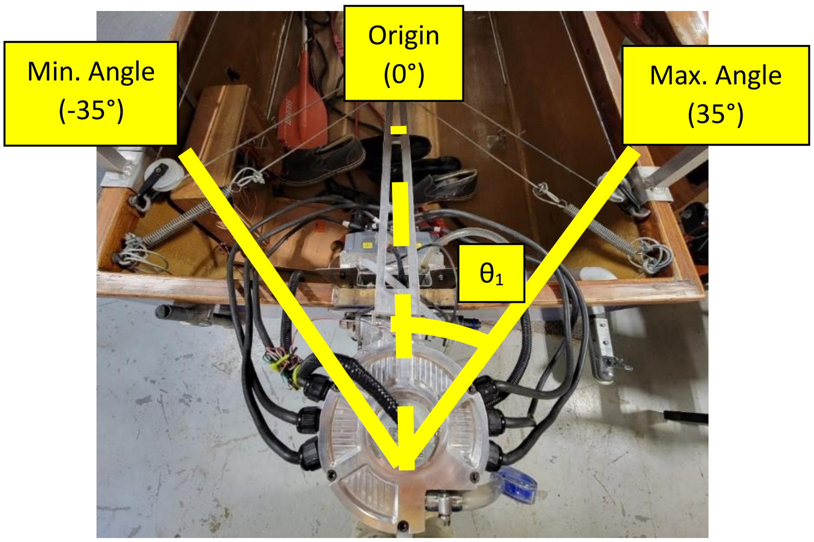


Fig 2.10. *Solar Splash rotational range of tiller arm*

The original system had a cable displacement of 45 inches completed in two to three hand pulls. Ideally, the cable displacement should be approximately 25 inches with only one hand pull, allowing the skipper to constantly keep his hand on the steering cable for quicker turning. This ideal cable displacement came from a test of all the solar boat team members sitting in the boat and doing one comfortable hand pull on the steering cable.

Additionally, there were several design issues with the tiller arm. First, there were only 3 holes on the tiller arm to range the placement of the pulley eye bolt. Because of all the room for improvement, we determined it would be easier to modify only the tiller arm in the steering system. Second, there was a major bending on the tiller arm. In sprint and slalom configuration, the tiller arm would bend significantly compared to the horizontal reference it was supposed to match (**Fig. 2.11**), causing higher bending stress on the tiller arm assembly.

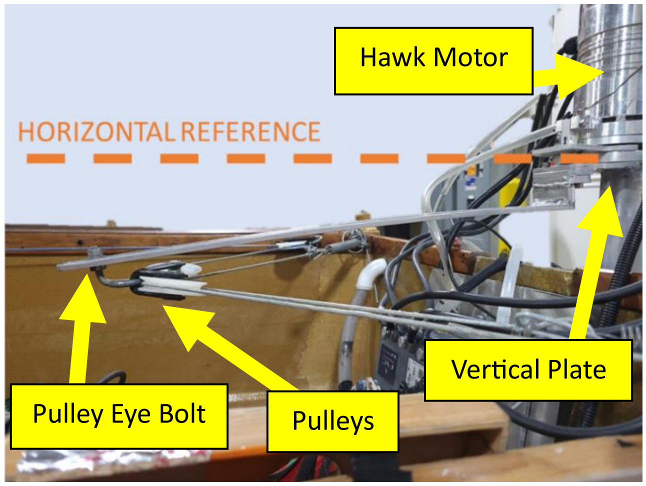


Fig 2.11. *Solar Splash tiller arm assembly – slalom setup*

There were some additional factors to consider in the tiller arm redesign. First, we rotated the torque load cell and Yamato adapter 180° on the drive train by separating and repeating the compression fit between the drive train and Yamato adapter (**Fig. 2.12**). The rotation moved the torque load cell from being right next to the tiller arm to the opposite side of the arm.

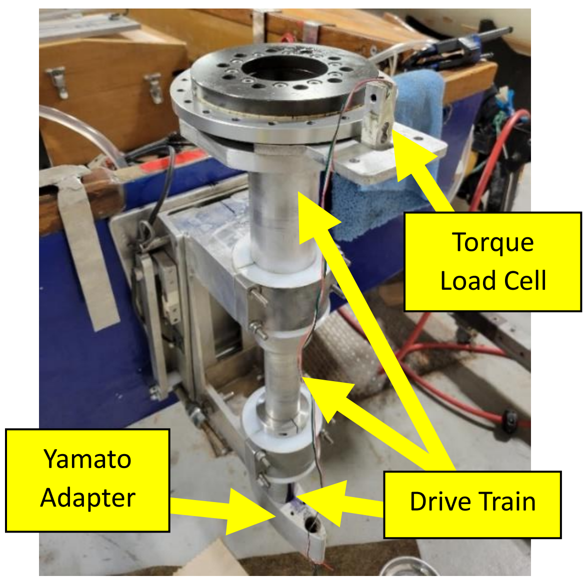


Fig 2.12. *Rotated orientation of torque load cell and Yamato adapter*

We determined the necessary temperatures for the compression fit using coefficients of thermal expansion in a TK Solver code, detailed in **Appendix F**. Third, the vertical spacer we created in the first new tiller arm design has a concern to produce a large moment on the tiller arm. Thus, we eliminated the spacer and modified the vertical plate to be taller. Considering these factors, the final tiller arm design is shown in **Fig 2.13**.

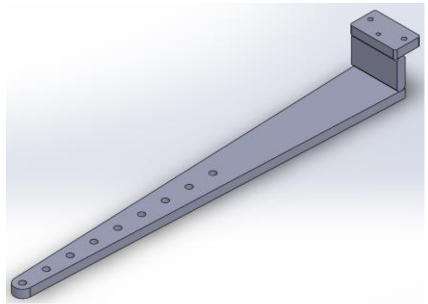


Fig 2.13. *Solar Splash SolidWorks model of 2023 v2 tiller arm design*

We also created a dynamic TK solver design code for the steering system. We based the redesign on a variable tiller arm length, but there are other parameters to consider in creating a more optimal design. This code allows the user to input some known variables such as tiller arm length and pulley positions to iteratively solve for the unknown angles and lengths of cables.

We performed a kinematic evaluation of the steering system using loop equations, Euler's formula, and geometry relationships (detailed in **Appendix F**). **Fig. 2.14** outlines the three loops used in our kinematic loop equation evaluation. Each loop creates two loop equations—real and imaginary—that contain unique and overlapping information, allowing for the equations to relate to each other. The only current limitation on this code is if the length of the tiller arm is less than 10.5 inches, the code will not solve realistically, resulting in extreme angles and lengths.

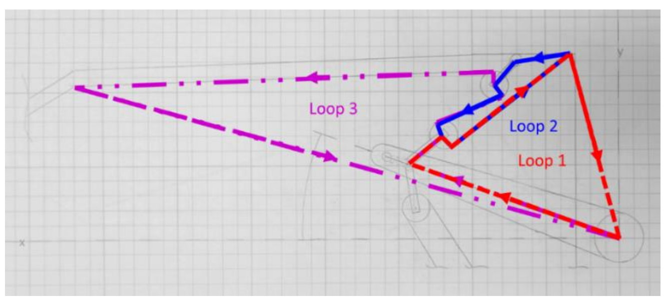


Fig 2.14. *Kinematic loops for Solar Splash boat steering system*

3) *Design Testing and Evaluation*: On the SS drive train assembly, load cells record the boat's thrust and torque. This data is vital for the creation process of new propellers. The load cells connect to the strain amplifier box on the hull, which connects to a computer to record the results.

The load cells work by incorporating the geometry of the thrust mount assembly. **Fig. 2.15** shows a SolidWorks model of the thrust mount assembly. All rods travel through the transom mount. The drive train mount rotates around rod 1, with the bottom part connected to rod 3 through two threaded bolts. Rod 3 is inserted through large holes, allowing it to slightly move in response to the propeller's thrust. The lever arm connects rod 3 to the load cell through a shoulder screw (mirrored on both sides of the assembly). As the propeller applies force onto the drive train, that force is relayed through the drive train mount through the lever arm to the load cell. The displacement of the load cell corresponds to the force of the propeller's thrust.

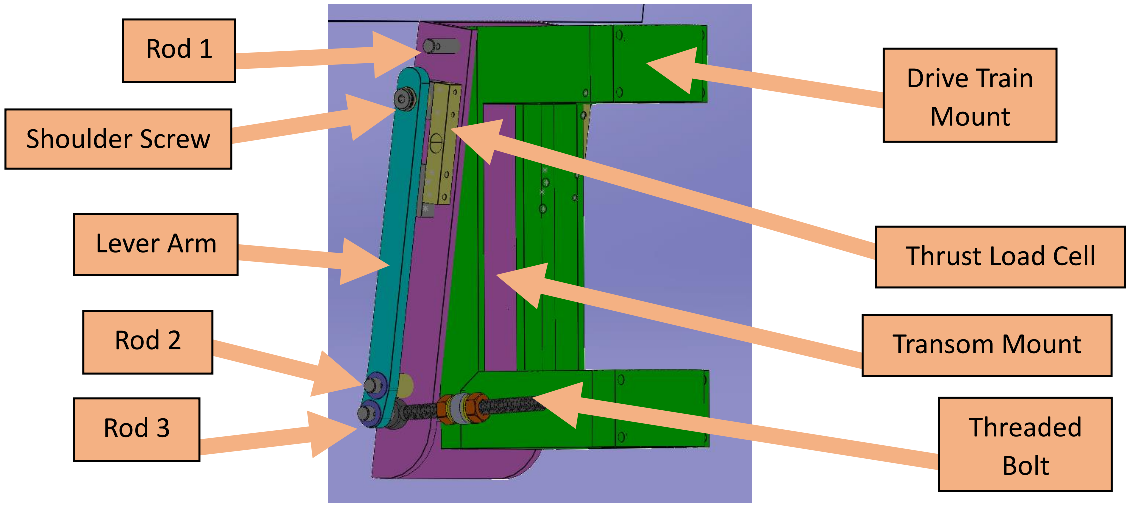


Fig. 2.15. *SolidWorks model of the thrust mount assembly*

Our initial test of the thrust load cell simulated propeller thrust by pushing on the drive train. However, the strain amplifier box did not record any data due to a seizure in the thrust mount assembly caused by inaccurately sized rods and spacers. To correct this assembly, we created a squared diagram of the thrust mount assembly with improved parts (**Fig. 2.16**).

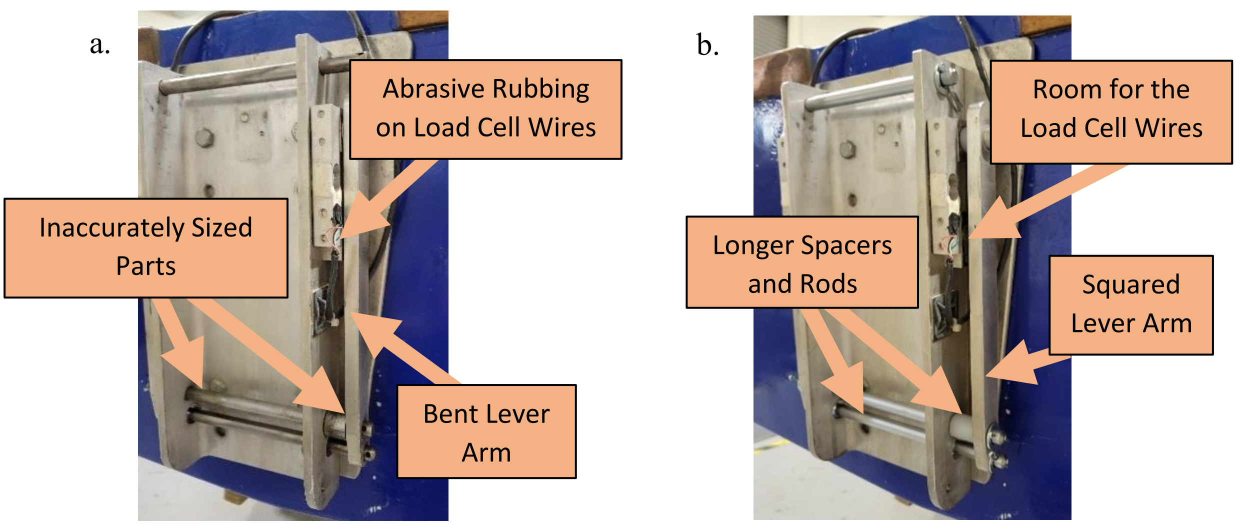


Fig. 2.16. *Thrust mount assembly (**a**) seized and (**b**) corrected.*

To test the accuracy of the load cells, we disassembled the thrust mount assembly from the hull and placed the assembly in a horizontal setup where we hung weights from the propeller to simulate thrust (**Fig. 2.17**). We increased the weight in steps of 20 lb from 0-140 lb. The thrust data from this weight test is recorded in **Fig. 2.18**. It should be noted that the starboard side thrust load cell was settling around 0.6 V instead of continuing a linear trend like the port side data. This load cell should be replaced. We can use the thrust data from the port side load cell since it was accurate and independent of the starboard load cell. Additional testing of the thrust load cell in the lake is needed, as well as testing the torque load cell.

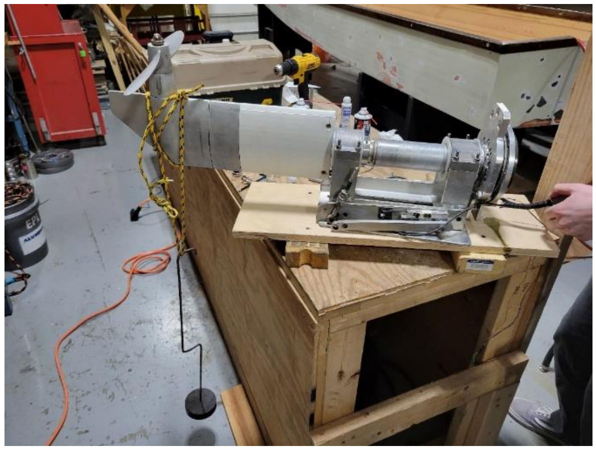


Fig. 2.17. *Incremented weight test for thrust load cells.*

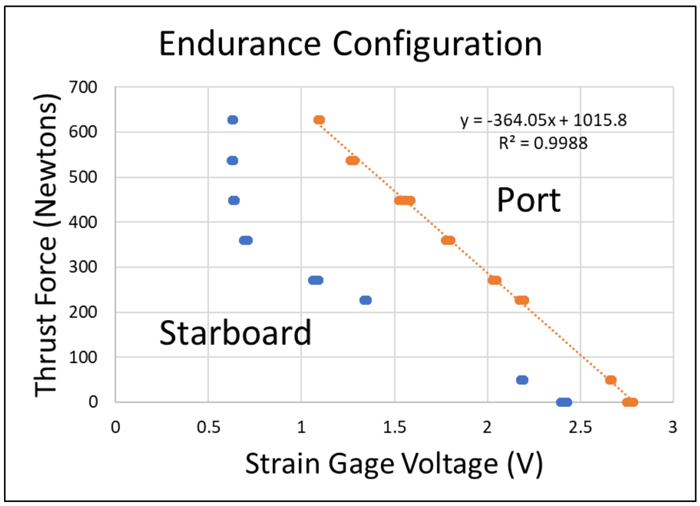


Fig. 2.18. *Incremented weight test for thrust load cells.*

F. DATA ACQUISITION AND COMMUNICATION

1) *Current Design:* The boat currently uses the CAN bus for communication between the driver interface, Raspberry Pi, GPS, ACS motor controller, and CAN adapters as shown in **Fig 2.19**. Data from these devices is then broadcast to all the devices on the bus where it is used for control, recorded, or monitored.

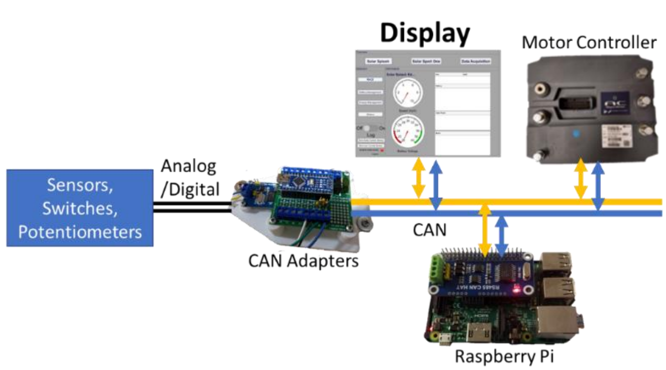


Fig 2.19. *Flow of CAN bus information between devices with MATLAB GUI tablet display.*

Previously the team used a Surface Tablet Pro to display CAN messages with a MATLAB Graphical User Interface (GUI). This display system needed to be replaced because it would overheat and shutdown, MATLAB and Windows caused instability, it was difficult to see, and the GUI had limited functionality and customizability. Just two weeks into the project, the tablet was also damaged by water and completely stopped functioning. Therefore, we required a replacement display that was waterproof, sunlight readable, reliable, and easily customizable.

The DAS used by the team consisted of a Raspberry Pi (RPi) which logged CAN messages. This worked but it had issues: it was difficult to make changes to the logging setup, log files were hard to identify and sometimes lost, and it was difficult to plot data quickly for analysis. These issues needed to be addressed and it was desired to make the DAS easier to use since data is essential for informing design and identifying areas for improvement.

The CAN bus could also be used to communicate messages for operating the boat, the system for accomplishing this is called the Boat Operating System (BOS). The development of this system was limited to a short program on a CAN adapter that was not fully tested or used to operate the boat. Instead, the boat had only run in the backup mode in which inputs from the dash controls were interpreted by a program on the motor controller to operate the motor. This program could only be edited or examined by the manufacturer, Inmotion. An objective for this year's team was to implement the BOS so that we could determine how the motor should respond to the dash controls on our own.

2) *Analysis of Design Concept*: While we looked for a replacement display, we had two main options, either use the previous MATLAB GUI on a new device or find a suitable CAN bus display. A CAN display is a single unit that connects to a CAN bus and comes with some software for designing a GUI to display messages from the CAN bus. Since the MATLAB GUI was difficult to work with and any replacement device would likely still have reliability issues with MATLAB or overheating, we decided to use a CAN bus display. We selected the AEM CD-7L CAN bus display shown in **Fig 2.20**. to be our replacement for the tablet because it met our requirements for being waterproof, sunlight readable, reliable, and easily customizable. This unit was selected over other CAN displays because of its user friendly and free software which we were able to use to design screens for the Sprint and Endurance events before we acquired the display.

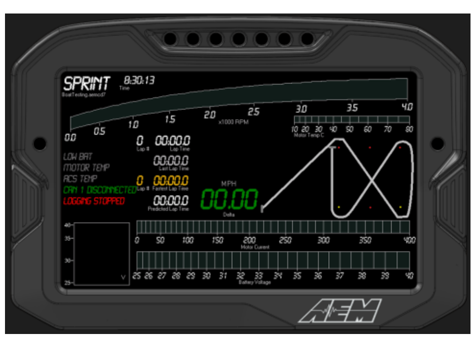


Fig 2.20. *AEM CD-7L display shown in a simulation from AEMDashDesign software with our design of the screen for the Sprint event.*

The objective for the DAS was to fix issues with the Raspberry Pi logging and make it easier to use. The issues with the Raspberry Pi were addressed early in the project but we still desired to make the DAS easier to use. The AEM CD-7L was also purchased to serve this purpose, since it is the “L” model, which has logging capability.

Our objective for the BOS was to implement it in such a way that changes could be made easily. Therefore, we decided to move the BOS from the CAN Adapter code to a Simulink model that would run on the RPi so that it would be easier to understand. The flow of information from the dash controls to the motor controller is now as shown in **Fig 2.21.**

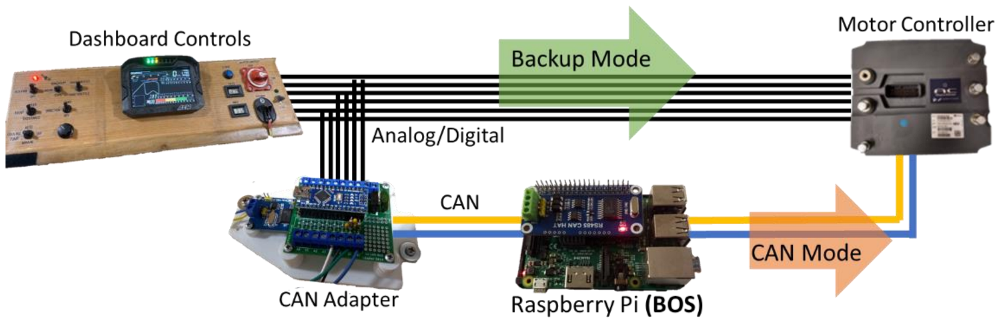


Fig 2.21. *Flow of information from the dashboard controls to the motor controller in the previously used Backup mode and the new CAN mode.*

Implementing this design required modifying the CAN adapter code so that it only converted the analog and digital values from the dash switches and potentiometers to CAN messages for the Raspberry Pi to receive. Then the BOS model on the RPi had to be designed such that the boat operated how we desired in any situation. For more details, see **Appendix J**.

3) *Design Testing and Evaluation*: The new AEM CD-7L display was successfully tested by connecting it to the CAN bus and checking that the messages displayed properly as shown in **Fig 2.22**.

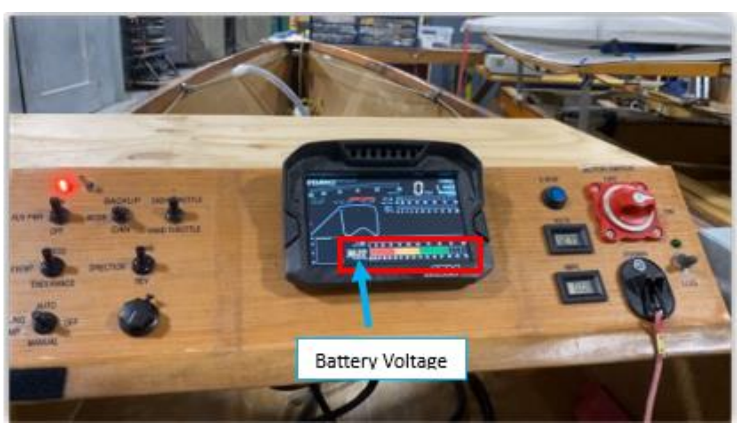
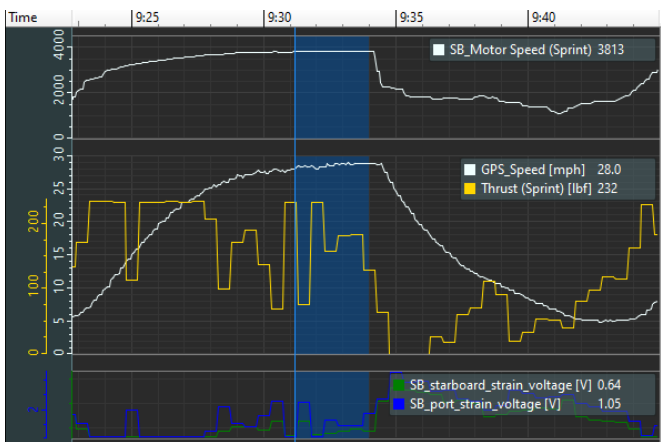


Fig 2.22. *AEM display shown to be functioning by displaying the battery voltage at 38.22 Volts.*

We have also used the display while testing the boat on the lake and have never had an instance of it shutting down (as the tablet would) and it is much easier to read in direct sunlight than the tablet was. Testing on the lake has also proven that its software is easier to work with than the previous MATLAB GUI. We were able to quickly fix an issue where the motor RPM in the Sprint configuration was not showing because it was negative, so we plugged in a laptop and quickly changed the RPM gauge to show negative RPM. These tests confirmed that the AEM CD-7L was sunlight readable, reliable, and easily customizable, therefore satisfying our requirements for a new display system. AEM specifies that this unit is waterproof as well, but we did not test this.

The improved DAS was also tested when we took the boat out to obtain high speed thrust data to estimate hull drag for informing propeller design. We found that compared to the Raspberry Pi, setting up logging on the AEM display was very simple and we could instantly plot log files from the display with the AEMdata software. We also verified that issues with the Raspberry Pi had been resolved since we did not lose any logs and the timestamp of the log was correct so they could be identified. Since we have found the AEM display to be easier to use and more robust than the RPi it will serve as our primary DAS and the RPi will be the backup. A sample of the data for our testing of the boat in the sprint configuration is shown in **Fig 2.23.**



	Min	Max	Avg	ΔT
Thrust (Sprint) [lbf]	74	232	167.09	00:02.818005
GPS_Speed [mph]	28.0	28.9	28.478	00:02.818005

Fig 2.23. *Data log obtained from AEM display from testing in the Sprint configuration shown in AEMdata with traces motor speed, GPS speed, calculated thrust, starboard and port strain plotted against time on the x-axis. Data points of interest for estimating hull drag have been highlighted, and a statistics report for those points is shown below the plots.*

We tested the BOS at multiple stages in its development. First, it was tested within Simulink by simulating the dash controls with interactive knobs and switches. Then we checked the intermediate values and output motor command speed while trying switch combinations and throttle positions to verify that they matched what the BOC specified. After testing in the simulation, the BOS was deployed to the Raspberry Pi. By supplying 36 V power to the dash in the SDL with a power supply, we were able to test everything within the dash, including the new CAN Adapter code, the BOS on the Raspberry Pi, and the AEM display as shown in **Fig 2.24.**

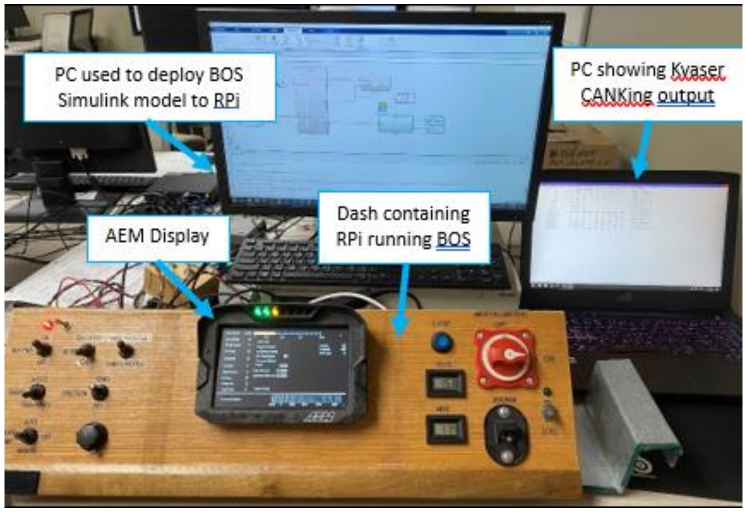


Fig 2.24. *Dash testing setup in the SDL supplying 36V to verify functionality of CAN Adapters, RPi, and AEM Display.*

This setup was used to verify the output CAN messages of the CAN adapter corresponding to the switch and potentiometer positions as well as the output motor speed command from the BOS by monitoring signals with both the AEM display and the Kvaser (a plug-in that allows us to view CAN messages on the bus). Once again, results were checked against the BOC for verification. Finally, we were able to successfully test the BOS on the boat. We found that the BOS worked, and the motor responded to the motor speed command CAN messages output by the BOS. Therefore, our objective is complete, and we can operate the boat in CAN mode. However, the motor controller does not ignore the digital signals from the dash switches like we want, so we are still working with the motor controller manufacturer to resolve this.

III. Project Management

A. Team members and leadership roles

Asa Mudd – Team captain, data acquisition and driver interface

Bryce Schmitt – Motors, motor controller, and dynamometer

Joseph Homan – Propellers

Nicolas Knowlton – CAN adapters and steering system

Ruth Lessen – Hydrofoil boat battery pack

Joseph Heise – Hydrofoil boat flight control system, sensors, and actuators

Daniel Shoultz – Hydrofoil boat strut design

Grace Fearday, Kezia Augusting, Cejay Walker - Electrical

B. Project planning and schedule

Our project started in August 2022. We spent the first month getting to know the boat each of our individual responsibilities. September 2022 – March 2023 we focused on research, design, and development for each of our projects. Then, starting in April 2023 we began our main testing for each project and official data collection of the boat. Starting in August 2022 we were testing the boat out on the water and learning to drive it. In April 2023 we began practicing for the slalom race.

C. Financial and fund raising

Our project is funded by Cedarville University. We have also been given equipment or financial aid from our sponsors, listed in **Appendix L.1**.

D. Strategy for team continuity and sustainability

Every week our team would meet with our advisors and give a verbal presentation with Microsoft PowerPoint to update the team on the previous week's work and to give direction for the next week. During this time, we were able to ask questions and get feedback as well. The team also kept a GroupMe group chat that everyone was active in, and our advisors' doors were always open to us.

E. Discussion and self-evaluation

We believe we worked well as a team. The weekly meetings proved effective at informing each member what was going on, even if the work did not pertain to them directly. As team members we challenged and encouraged each other, giving time as needed to work with each other as needed. Our team captain did a good job of helping all members hit necessary deadlines for reports and presentations, and we saw initiative among other members as well to take the lead when needed.

IV. Conclusions and Recommendations

A. Strengths and weaknesses

Our design has many strengths. It uses the CAN bus for easy communication. The new driver interface screen works well in heat and sunlight and gives a much simpler and efficient way to log data. We have improved the water cooling to be a closed-loop system that will keep the motors clean. We improved how we find the thrust. And we have an improved steering system that will make the boat more responsive and require less readjustment from the skipper.

There are, however, a few weaknesses to note. We are not yet currently getting the most out of our motor. The motor controller is still responding to the digital output from the dash when we are using the BOS. The sprint propeller has not yet been improved for our race. And finally, the dynamic TK Solver steering system code is limited to a 10.5-inch tiller arm.

B. Completion of objectives

We have achieved most of our objectives for this project. We have verified the motor performance through dyno-testing and found design constraints for propeller design through motor and thrust testing. We have also created a closed-loop cooling system for the motor. We were able to design and manufacture propellers and an updated steering system. We also successfully found a new driver display interface, established a reliable data acquisition system, and successfully implemented CAN communication.

Objectives that were not completed are to successfully design a new sprint propeller that will help us achieve our desired sprint time of 23.86 seconds.

C. Reflection of design process

The design process ended up taking much longer than originally expected. We were still able to accomplish most of our goals, but we did so about a month later than originally planned. Our process entailed research, creating design tool and guidelines (like MATLAB codes or prototypes and getting measurements), initial design and testing, then redesign and testing. This process, although slow, seemed to be effective as we were able to complete our work with success.

D. Where we can go from here

Looking to the future, we will need to get a Hawk40 motor, get more concrete and specific propeller design constraints through dyno-testing, design a functioning and improved sprint propeller, and get a boat functioning and race-ready on hydrofoils.

E. Lessons learned

Working on this competition team helped us learn what it looks like to work on an engineering team. We learned what it takes to design a specific outcome within given constraints. We have also learned how to perform good research, professionally interact with other supplying companies, give professional oral presentations, and write up professional written reports. Finally, we learned how to document our work well, as we need to make sure next year’s team is able to continue our work, instead of starting over.

References

N/A

Appendix Table of Contents

Appendix A. Battery Documentation

This year, in line with previous years, we will be utilizing two sets of Sealed Lead-Acid batteries. The first being a set of three Genesis 42EP batteries (denoted as G42EP in Fig. A.1) weighing 32.9 lb. (14.9 kg) each, with a total weight of 98.34 lb. (44.7 kg). The second set is composed of nine Genesis 13EP batteries (G13EP in Fig. A.1), each weighing 10.8 lb. (4.9 kg), with a total weight of 97.2 lb. (44.1 kg). This is in compliance with the new Solar Splash rule 7.4.1 having both of the battery sets under the 100 lb. (45.5 kg) limit. The specification and MSDS sheets for these two types of batteries, which were selected from the available batteries provided by EnerSys are given in Fig. A.1 and the remaining pages of this appendix below.

General Specifications

Battery Type	Nominal Voltage (V)	Nominal Capacity (Ah)	Nominal Dimensions						Typical Weight		Torque		Internal Resistance (mΩ)*	Short Circuit Current (A)*
		10 Hr Rate-Ah	Length in	Length mm	Width in	Width mm	Height in	Height mm	lbs	kg	in-lbs	Nm		
G13EP	12	13	6.89	175.0	3.27	83.1	5.08	129.0	10.8	4.9	50	5.6	21.4	600
G16EP	12	16	7.12	180.8	2.99	75.9	6.57	166.9	13.5	6.1	50	5.6	19.1	675
G26EP	12	28	6.54	166.1	6.89	175.0	4.92	125.0	22.3	10.1	60	6.8	12.3	1150
G42EP	12	42	7.74	196.6	6.50	165.1	6.69	169.9	32.9	14.9	60	6.8	8.8	1480
G70EP	12	72	12.94	328.7	6.54	166.1	6.85	174.0	53.5	24.3	60	6.8	6.1	2100
G200EP	12	200	22.87	580.9	4.92	125.0	12.46	316.5	132.3	59.9	44	5.0	3.15	4000

* Tested per IEC 60896 Part 21

Fig. A.1. *General Specifications for Genesis 13 and 42 EP Sealed Lead-Acid Batteries*

EnerSys
Power/Full Solutions

SAFETY DATA SHEET

Form #: SDS 853024
Revised: AF
Supersedes: AE
ECO #: 1002195

I. PRODUCT IDENTIFICATION

Chemical Trade Name (as used on label):
Non-Spillable Lead Acid Battery

Synonyms:
Industrial Battery, Traction Battery, Stationary Battery,
Deep Cycle Battery

Manufacturer's Name/Address:

EnerSys	Canada Corporate Office
P.O. Box 14145	3-61 Parr Boulevard
2366 Bernville Road	Bolton, Ontario
Reading, PA 19612-4145	L7E 4E3

Chemical Family/Classification:
Electric Storage Battery

Telephone:
For information and emergencies, contact EnerSys' Environmental, Health & Safety Dept. at 610-208-1996

24-Hour Emergency Response Contact:
CHEMTREC DOMESTIC: 800-424-9300 CHEMTREC INT'L: 703-527-3877

II GHS HAZARDS IDENTIFICATION

HEALTH		ENVIRONMENTAL	PHYSICAL
Acute Toxicity (Oral/Dermal/Inhalation)	Category 4	Aquatic Chronic 1	Explosive Chemical, Division 1.3
Skin Corrosion/Irritation	Category 1A	Aquatic Acute 1	
Eye Damage	Category 1		
Reproductive	Category 1A		
Carcinogenicity (lead compounds)	Category 1B		
Carcinogenicity (arsenic)	Category 1A		
Carcinogenicity (acid mist)	Category 1A		
Specific Target Organ Toxicity (repeated exposure)	Category 2		

GHS LABEL:

HEALTH

ENVIRONMENTAL

PHYSICAL

Hazard Statements

DANGER!

Causes severe skin burns and serious eye damage.
May damage fertility or the unborn child if ingested or inhaled.
May cause cancer if ingested or inhaled.
Causes damage to central nervous system, blood and kidneys through prolonged or repeated exposure.
May form explosive air/gas mixture during charging.
Explosive, fire, blast, or projection hazard.
May cause harm to breast-fed children
Harmful if swallowed, inhaled, or contact with skin
Causes skin irritation, serious eye damage.

Precautionary Statements

Wash thoroughly after handling.
Do not eat, drink or smoke when using this product.
Wear protective gloves/protective clothing, eye protection/face protection.
Avoid breathing dust/fume/gas/mist/vapors/spray.
Use only outdoors or in a well-ventilated area.
Contact with internal components may cause irritation or severe burns. Avoid contact with internal acid.
Irritating to eyes, respiratory system, and skin.
Obtain special instructions before use.
Do not handle until all safety precautions have been read and understood
Avoid contact during pregnancy/while nursing
Keep away from heat./sparks/open flames/hot surfaces. No smoking

III. COMPOSITION/INFORMATION ON INGREDIENTS

Components	CAS Number	Approximate % by Wt.
Inorganic Lead Compound:		
Lead	7439-92-1	45-60
Lead Dioxide	1309-60-0	15-25
* Antimony	7440-36-0	2
* Arsenic	7440-38-2	0.2
* Calcium	7440-70-2	0.04
* Tin	7440-31-5	0.2
Electrolyte (Sulfuric Acid (H2SO4/H2O))	7664-93-9	10-30
Case Material:		5-10
Polypropylene	9003-07-0	
Polystyrene	9003-53-6	
Styrene Acrylonitrile	9003-54-7	
Acrylonitrile Butadiene Styrene	9003-56-9	
Styrene Butadiene	9003-55-8	
Polyvinylchloride	9002-86-2	
Polycarbonate, Hard Rubber, Polyethylene	9002-88-4	

Page 1

EnerSys Power/Full Solutions

SAFETY DATA SHEET

Form #: SDS 853024
Revised: AF
Supersedes: AE
ECO #: 1002195

Other: Silicon Dioxide (Gel batteries only) Sheet Molding Compound (Glass reinforced polyester)	7631-86-9 --	1-5	

Inorganic lead and electrolyte (sulfuric acid) are the primary components of every battery manufactured by EnerSys.
Other ingredients may be present dependent upon battery type. Contact your EnerSys representative for additional information.

IV. FIRST AID MEASURES

Inhalation:

Sulfuric Acid: Remove to fresh air immediately. If breathing is difficult, give oxygen. Consult a physician.
Lead: Remove from exposure, gargle, wash nose and lips; consult physician.

Ingestion:

Sulfuric Acid: Give large quantities of water; do not induce vomiting or aspiration into the lungs may occur and can cause permanent injury or death; consult a physician.
Lead: Consult physician immediately.

Skin:

Sulfuric Acid: Flush with large amounts of water for at least 15 minutes; remove contaminated clothing completely, including shoes.
If symptoms persist, seek medical attention. Wash contaminated clothing before reuse. Discard contaminated shoes.
Lead: Wash immediately with soap and water.

Eyes:

Sulfuric Acid and Lead: Flush immediately with large amounts of water for a least 15 minutes while lifting lids
Seek immediate medical attention if eyes have been exposed directly to acid.

V. FIRE FIGHTING MEASURES

Flash Point: N/A | **Flammable Limits:** LEL = 4.1% (Hydrogen Gas) UEL = 74.2%

Extinguishing Media: CO2; foam; dry chemical. Do not use carbon dioxide directly on cells. Avoid breathing vapors. Use appropriate media for surrounding fire.

Special Fire Fighting Procedures:

If batteries are on charge, shut off power. Use positive pressure, self-contained breathing apparatus. Water applied to electrolyte generates heat and causes it to spatter. Wear acid-resistant clothing, gloves, face and eye protection.
But note that strings of series connected batteries may still pose risk of electric shock even when charging equipment is shut down.

Unusual Fire and Explosion Hazards:

Highly flammable hydrogen gas is generated during charging and operation of batteries. To avoid risk of fire or explosion, keep sparks or other sources of ignition away from batteries. Do not allow metallic materials to simultaneously contact negative and positive terminals of cells and batteries. Follow manufacturer's instructions for installation and service.

VI. ACCIDENTAL RELEASE MEASURES

Spill or Leak Procedures:

Stop flow of material, contain/absorb small spills with dry sand, earth, and vermiculite. Do not use combustible materials. If possible, carefully neutralize spilled electrolyte with soda ash, sodium bicarbonate, lime, etc. Wear acid-resistant clothing, boots, gloves, and face shield. Do not allow discharge of unneutralized acid to sewer. Acid must be managed in accordance with local, state, and federal requirements.
Consult state environmental agency and/or federal EPA.

VII. HANDLING AND STORAGE

Handling:

Unless involved in recycling operations, do not breach the casing or empty the contents of the battery. Handle carefully and avoid tipping, which may allow electrolyte leakage. There may be increasing risk of electric shock from strings of connected batteries.
Keep containers tightly closed when not in use. If battery case is broken, avoid contact with internal components.
Keep vent caps on and cover terminals to prevent short circuits. Place cardboard between layers of stacked automotive batteries to avoid damage and short circuits.
Keep away from combustible materials, organic chemicals, reducing substances, metals, strong oxidizers and water. Use banding or stretch wrap to secure items for shipping.

Storage:

Store batteries in cool, dry, well-ventilated areas with impervious surfaces and adequate containment in the event of spills. Batteries should also be stored under roof for protection against adverse weather conditions. Separate from incompatible materials. Store and handle only in areas with adequate water supply and spill control. Avoid damage to containers. Keep away from fire, sparks and heat. Keep away from metallic objects could bridge the terminals on a battery and create a dangerous short-circuit.

Charging:

There is a possible risk of electric shock from charging equipment and from strings of series connected batteries, whether or not being charged. Shut-off power to chargers whenever not in use and before detachment of any circuit connections. Batteries being charged will generate and release flammable hydrogen gas.
Charging space should be ventilated. Keep battery vent caps in position. Prohibit smoking and avoid creation of flames and sparks nearby.
Wear face and eye protection when near batteries being charged.

EnerSys Power/Full Solutions

SAFETY DATA SHEET

Form #: SDS 853024
Revised: AF
Supersedes: AE
ECO #: 1002195

VIII. EXPOSURE CONTROLS/PERSONAL PROTECTION

Exposure Limits (mg/m3) Note: N.E.= Not Established

INGREDIENTS (Chemical/Common Names)	OSHA PEL	ACGIH	US NIOSH	Quebec PEV	Ontario OEL	EU OEL
Lead and Lead Compounds (inorganic)	0.05	0.05	0.05	0.05	0.05	0.15 (b)
Antimony	0.5	0.5	0.5	0.5	0.5	0.5 (b,e)
Arsenic	0.01	0.01	0.002	0.2	0.01	N.E
Calcium	N.E	N.E	N.E	N.E	N.E	N.E
Tin	2	2	2	2	2	N.E
Electrolyte (Sulfuric Acid)	1	0.2	1	1	0.2	0.05 (c)
Polypropylene	N.E	N.E	N.E	N.E	N.E	N.E
Polystyrene	N.E	N.E	N.E	N.E	N.E	N.E
Styrene Acrylonitrile	N.E	N.E	N.E	N.E	N.E	N.E
Acrylonitrile Butadiene Styrene	N.E	N.E	N.E	N.E	N.E	N.E
Styrene Butadiene	N.E	N.E	N.E	N.E	N.E	N.E
Polyvinylchloride	N.E	N.E	N.E	N.E	1	N.E
Polycarbonate, Hard Rubber, Polyethylene	N.E	N.E	N.E	N.E	N.E	N.E
Silicon Dioxide (Gel Batteries Only)	N.E	N.E	N.E	N.E	N.E	N.E
Sheet Molding Compound (Glass reinforced polyester)	N.E	N.E	N.E	N.E	N.E	N.E

NOTES:
(b) As inhalable aerosol
(c) Thoracic fraction
(e) Based on OEL;s Of Austria, Belgium, Denmark, France, Netherlands, Switzerland, & U.K.

Engineering Controls (Ventilation):

Store and handle in well-ventilated area. If mechanical ventilation is used, components must be acid-resistant. Handle batteries cautiously to avoid spills. Make certain vent caps are on securely. Avoid contact with internal components. Wear protective clothing, eye and face protection when filling, charging or handling batteries. Do not allow metallic materials to simultaneously contact both the positive and negative terminals of the batteries. Charge the batteries in areas with adequate ventilation. General dilution ventilation is acceptable.

Respiratory Protection (NIOSH/MSHA approved):

None required under normal conditions. When concentrations of sulfuric acid mist are known to exceed the PEL, use NIOSH or MSHA-approved respiratory protection.

Skin Protection:

If battery case is damaged, use rubber or plastic acid-resistant gloves with elbow-length gauntlet, acid-resistant apron, clothing and boots.

Eye Protection:

If battery case is damaged, use chemical goggles or face shield.

Other Protection:

In areas where sulfuric acid is handled in concentrations greater than 1%, emergency eyewash stations and showers should be provided, with unlimited water supply. Acid-resistant apron. Under severe exposure emergency conditions, wear acid-resistant clothing and boots. Face shield recommended when adding water or electrolyte to batteries, wash hands after handling.

IX. PHYSICAL AND CHEMICAL PROPERTIES

Properties Listed Below are for Electrolyte:

Boiling Point:	203 - 240° F	**Specific Gravity (H2O = 1):**	1.215 to 1.350
Melting Point:	N/A	**Vapor Pressure (mm Hg):**	10
Solubility in Water:	100%	**Vapor Density (AIR = 1):**	Greater than 1
Evaporation Rate: (Butyl Acetate = 1)	Less than 1	**% Volatile by Weight:**	N/A
pH:	~1 to 2	**Flash Point:**	Below room temperature (as hydrogen gas)
LEL (Lower Explosive Limit)	4.1% (Hydrogen)	**UEL (Upper Explosive Limit)**	74.2% (Hydrogen)
Appearance and Odor:	Manufactured article; no apparent odor. Electrolyte is a clear liquid with a sharp, penetrating, pungent odor.		

Form #: SDS 853024
Revised: AF
Supersedes: AE
ECO #: 1002195

SAFETY DATA SHEET

X. STABILITY AND REACTIVITY

Stability: Stable X_ Unstable ___

This product is stable under normal conditions at ambient temperature

Conditions To Avoid: Prolonged overcharge; sources of ignition

Incompatibility: (Materials to avoid)

Sulfuric Acid: Contact with combustibles and organic materials may cause fire and explosion. Also reacts violently with strong reducing agents, metals, sulfur trioxide gas, strong oxidizers and water. Contact with metals may produce toxic sulfur dioxide fumes and may release flammable hydrogen gas.

Lead Compounds: Avoid contact with strong acids, bases, halides, halogenates, potassium nitrate, permanganate, peroxides, nascent hydrogen and reducing agents.

Arsenic compounds: strong oxidizers; bromine azide. NOTE: hydrogen gas can react with inorganic arsenic to form the highly toxic gas-arsine.

Hazardous Decomposition Products:

Sulfuric Acid: Sulfur trioxide, carbon monoxide, sulfuric acid mist, sulfur dioxide, and hydrogen sulfide.

Lead Compounds: High temperatures likely to produce toxic metal fume, vapor, or dust; contact with strong acid or base or presence of nascent hydrogen may generate highly toxic arsine gas.

Hazardous Polymerization:

Will not occur

XI. TOXICOLOGICAL INFORMATION

Routes of Entry:

Sulfuric Acid: Harmful by all routes of entry.

Lead Compounds: Hazardous exposure can occur only when product is heated, oxidized or otherwise processed or damaged to create dust, vapor or fume. The presence of nascent hydrogen may generate highly toxic arsine gas.

Inhalation:

Sulfuric Acid: Breathing of sulfuric acid vapors or mists may cause severe respiratory irritation.

Lead Compounds: Inhalation of lead dust or fumes may cause irritation of upper respiratory tract and lungs.

Ingestion:

Sulfuric Acid: May cause severe irritation of mouth, throat, esophagus and stomach.

Lead Compounds: Acute ingestion may cause abdominal pain, nausea, vomiting, diarrhea and severe cramping. This may lead rapidly to systemic toxicity and must be treated by a physician.

Skin Contact:

Sulfuric Acid: Severe irritation, burns and ulceration.

Lead Compounds: Not absorbed through the skin.

Arsenic Compounds: Contact may cause dermatitis and skin hyper pigmentation.

Eye Contact:

Sulfuric Acid: Severe irritation , burns, cornea damage, and blindness.

Lead Components: May cause eye irritation.

Effects of Overexposure - Acute:

Sulfuric Acid: Severe skin irritation, damage to cornea, upper respiratory irritation.

Lead Compounds: Symptoms of toxicity include headache, fatigue, abdominal pain, loss of appetite, muscular aches and weakness, sleep disturbances and irritability.

Effects of Overexposure - Chronic:

Sulfuric Acid: Possible erosion of tooth enamel, inflammation of nose, throat and bronchial tubes.

Lead Compounds: Anemia; neuropathy, particularly of the motor nerves, with wrist drop; kidney damage; reproductive changes in males and females. Repeated exposure to lead and lead compounds in the workplace may result in nervous system toxicity. Some toxicologists report abnormal conduction velocities in persons with blood lead levels of 50mcg/100 ml or higher. Heavy lead exposure may result in central nervous system damage, encephalopathy and damage to the blood-forming (hematopoietic) tissues.

Carcinogenicity:

Sulfuric Acid: The International Agency for Research on Cancer (IARC) has classified "strong inorganic acid mist containing sulfuric acid" as a Group 1 carcinogen, a substance that is carcinogenic to humans. This classification does not apply to liquid forms of sulfuric acid or sulfuric acid solutions contained within a battery. Inorganic acid mist (sulfuric acid mist) is not generated under normal use of this product. Misuse of the product, such as overcharging, may result in the generation of sulfuric acid mist.

Lead Compounds: Lead is listed as a Group 2A carcinogen, likely in animals at extreme doses. Per the guidance found in OSHA 29 CFR 1910.1200 Appendix F, this is approximately equivalent to GHS Category 1B. Proof of carcinogenicity in humans is lacking at present.

Arsenic: Arsenic is listed by IARC as a Group 1 - carcinogenic to humans. Per the guidance found in OSHA 29 CFR 1910.1200 Appendix F, this is approximately equivalent to GHS Category 1A.

Medical Conditions Generally Aggravated by Exposure:

Overexposure to sulfuric acid mist may cause lung damage and aggravate pulmonary conditions. Contact of sulfuric acid with skin may aggravate diseases such as eczema and contact dermatitis. Lead and its compounds can aggravate some forms of kidney, liver and neurologic diseases.

SAFETY DATA SHEET

Form #: SDS 853024
Revised: AF
Supersedes: AE
ECO #: 1002195

Acute Toxicity:
Inhalation LD50:
Electrolyte: LC50 rat: 375 mg/m3; LC50: guinea pig: 510 mg/m3
Elemental Lead: Acute Toxicity Point Estimate = 4500 ppmV (based on lead bullion)
Elemental Arsenic: No data

Oral LD50:
Electrolyte: rat: 2140 mg/kg
Elemental Lead: Acute Toxicity Estimate (ATE) = 500 mg/kg body weight (based on lead bullion)
Elemental Arsenic: LD50 mouse: 145 mg/kg
Elemental Antimony: LD50 rat: 100 mg/kg

Additional Health Data:
All heavy metals, including the hazardous ingredients in this product, are taken into the body primarily by inhalation and ingestion.
Most inhalation problems can be avoided by adequate precautions such as ventilation and respiratory protection covered in Section 8.
Follow good personal hygiene to avoid inhalation and ingestion: wash hands, face, neck and arms thoroughly before eating, smoking or leaving the worksite. Keep contaminated clothing out of non-contaminated areas, or wear cover clothing when in such areas. Restrict the use and presence of food, tobacco and cosmetics to non-contaminated areas. Work clothes and work equipment used in contaminated areas must remain in designated areas and never taken home or laundered with personal non-contaminated clothing. This product is intended for industrial use only and should be isolated from children and their environment.

The 19th Amendment to EC Directive 67/548/EEC classified lead compounds, but not lead in metal form, as possibly toxic to reproduction.
Risk phrase 61: May cause harm to the unborn child, applies to lead compounds, especially soluble forms.

XII. ECOLOGICAL INFORMATION

Environmental Fate:
Lead is very persistent in soil and sediments. No data on environmental degradation. Mobility of metallic lead between ecological compartments is slow.
Bioaccumulation of lead occurs in aquatic and terrestrial animals and plants but little bioaccumulation occurs through the food chain.
Most studies include lead compounds and not elemental lead.

Environmental Toxicity: Aquatic Toxicity:

Sulfuric acid:	24-hr LC50, freshwater fish (Brachydanio rerio): 82 mg/L
	96 hr- LOEC, freshwater fish (Cyprinus carpio): 22 mg/L
Lead:	48 hr LC50 (modeled for aquatic invertebrates): <1 mg/L, based on lead bullion
Arsenic:	24 hr LC50, freshwater fish (Carrassisus auratus) >5000 g/L

Additional Information:
- No known effects on stratospheric ozone depletion.
- Volatile organic compounds: 0% (by Volume)
- Water Endangering Class (WGK): NA

XIII. DISPOSAL CONSIDERATIONS (UNITED STATES)

Spent batteries: Send to secondary lead smelter for recycling. Spent lead-acid batteries are not regulated as hazardous waste when the requirements of 40 CFR Section 266.80 are met. This should be managed in accordance with approved local, state and federal requirements. Consult state environmental agency and/or federal EPA.

Electrolyte:
Place neutralized slurry into sealed containers and handle as applicable with state and federal regulations. Large water-diluted spills, after neutralization and testing, should be managed in accordance with approved local, state and federal requirements. Consult state environmental agency and/or federal EPA.

Following local, State/Provincial, and Federal/National regulations applicable to end-of-life characteristics will be the responsibility of the end-user.

EnerSys
Power/Full Solutions

SAFETY DATA SHEET

Form #: SDS 853024
Revised: AF
Supersedes: AE
ECO #: 1002195

XIV. TRANSPORT INFORMATION

U.S. DOT:

Excepted from the hazardous materials regulations (HMR) because the batteries meet the requirements of 49 CFR 173.159(f) and 49 CFR 173.159a of the U.S. Department of Transportation/s HMR. Battery and outer package must be marked " NONSPILLABLE" or "NONSPILLABLE BATTERY" Battery terminals must be protected against short circuits.

IATA Dangerous Goods Regulations DGR:

Excepted from the dangerous goods regulations because the batteries meet the requirements of Packing Instruction 872 and Special Provisions A67 of the International Air Transportation Association (IATA) Dangerous goods Regulations and International Civil Aviation Organization (ICAO) Technical Instructions. Battery Terminals must be protected against short circuits.

The words " NOT RESTRICTED" , SPECIAL PROVISION A67" must be provided on an airway bill when air waybill is issued.

IMDG:

Excepted from the dangerous goods regulations for transport by sea because the batteries meet the requirements of Special Provision 238 of the International Maritime Dangerous Goods(IMDG CODE). Battery terminals must be protected against short circuits.

XV. REGULATORY INFORMATION

UNITED STATES:

EPA SARA Title III:

Section 302 EPCRA Extremely Hazardous Substances (EHS):

Sulfuric acid is a listed "Extremely Hazardous Substance" under EPCRA, with a Threshold Planning Quantity (TPQ) of 1,000 lbs.
EPCRA Section 302 notification is required if 1000 lbs or more of sulfuric acid is present at one site (40 CFR 370.10). For more information consult 40 CFR Part 355. The quantity of sulfuric acid will vary by battery type. Contact your EnerSys representative for additional information.

Section 304 CERCLA Hazardous Substances:

Reportable Quantity (RQ) for spilled 100% sulfuric acid under CERCLA (Superfund) and EPCRA (Emergency Planning and Community Right to Know Act) is 1,000 lbs. State and local reportable quantities for spilled sulfuric acid may vary.

Section 311/312 Hazard Categorization:

EPCRA Section 312 Tier Two reporting is required for non-automotive batteries if sulfuric acid is present in quantities of 500 lbs or more and/or if lead is present in quantities of 10,000 lbs or more. For more information consult 40 CFR 370.10 and 40 CFR 370.40.

Section 313 EPCRA Toxic Substances:

40 CFR section 372.38 (b) states: If a toxic chemical is present in an article at a covered facility, a person is not required to consider the quantity of the toxic chemical present in such article when determining whether an applicable threshold has been met under § 372.25, § 372.27, or § 372.28 or determining the amount of release to be reported under § 372.30. This exemption applies whether the person received the article from another person or the person produced the article. However, this exemption applies only to the quantity of the toxic chemical present in the article.

Supplier Notification:

This product contains toxic chemicals, which may be reportable under EPCRA Section 313 Toxic Chemical Release Inventory (Form R) requirements.
If you are a manufacturing facility under SIC codes 20 through 39, the following information is provided to enable you to complete the required reports:

Toxic Chemical	CAS Number	Approximate % by Wt.
Lead	7439-92-1	60
Electrolyte (Sulfuric Acid (H2SO4/H2O))	7664-93-9	10 - 30
* Antimony	7440-36-0	2
* Arsenic	7440-38-2	0.2
Tin	7440-31-5	0.2

See 40 CRG Part 370 for more details.

If you distribute this product to other manufacturers in SIC Codes 20 through 39, this information must be provided with the first shipment of each calendar year.

The Section 313 supplier notification requirement does not apply to batteries, which are "consumer products".

* Not present in all battery types. Contact your EnerSys representative for additional information.

EnerSys
Power/Full Solutions

SAFETY DATA SHEET

Form #: SDS 853024
Revised: AF
Supersedes: AE
ECO #: 1002195

TSCA:

TSCA Section 8b – Inventory Status: All chemicals comprising this product are either exempt or listed on the TSCA Inventory.

TSCA Section 12b (40 CFR Part 707.60(b)) No notice of export will be required for articles, except PCB articles, unless the Agency so requires in the context of individual section 5, 6, or 7 actions.

TSCA Section 13 (40 CFR Part 707.20): No import certification required (EPA 305-B-99-001, June 1999, Introduction to the Chemical Import Requirements of the Toxic Substances Control Act, Section IV.A).

RCRA:

Spent Lead Acid Batteries are subject to streamlined handling requirements when managed in compliance with 40 CFR section 266.80 or 40 CFR part 273. Waste sulfuric acid is a characteristic hazardous waste; EPA hazardous waste number D002 (corrosivity) and D008 (lead).

CAA:

EnerSys supports preventative actions concerning ozone depletion in the atmosphere due to emissions of CFC's and other ozone depleting chemicals (ODC's), defined by the USEPA as Class I substances. Pursuant to Section 611of the Clean Air Act Amendments (CAAA) of 1990, finalized on January 19, 1993, EnerSys established a policy to eliminate the use of Class I ODC's prior to the May 15, 1993 deadline.

STATE REGULATIONS (US):

Proposition 65:

Warning: Battery posts, terminals and related accessories contain lead and lead compounds, chemicals known to the State of California to cause cancer and reproductive harm. Batteries also contain other chemicals known to the State of California to cause cancer. Wash hands after handling.

INTERNATIONAL REGULATIONS:

Distribution into Quebec to follow Canadian Controlled Product Regulations (CPR) 24(1) and 24(2).

Distribution into the EU to follow applicable Directives to the Use, Import/Export of the product as-sold.

Article 33 (1) of the REACH regulation (Reg. EC 1907/2006), which entered into force on 1st of June 2007 in the European Union, requires that manufacturers communicate the presence of Substances of Very High Concern (SVHC) in articles (lead batteries) in concentration greater than 0.1% by weight.

Effective the 27th of June 2018, the European Chemical Agency (ECHA) updated the Candidate List with the inclusion of Lead Metal (CAS No.: 7439-92-1). This inclusion of Lead as an SVHC applies to all of EnerSys Lead based battery products regardless of the design (Flooded, Gel, AGM, etc…).

XVI. OTHER INFORMATION

Revised: 4/7/2020

NFPA Hazard Rating for Sulfuric Acid:

Flammability (Red) = 0
Health (Blue) = 3
Reactivity (Yellow) = 2
Sulfuric acid is water-reactive if concentrated.

DISCLAIMER

This Safety Data Sheet is created by the manufacturer to comply with the requirements of 29 CFR 1910.1200. To the extent allowed by law, the manufacturer hereby expressly disclaims any liability to any third party, including users of this product, including, but not limited to, consequential or other damages, arising out of the use of, or reliance on, this Safety Data Sheet.

Appendix B. Flotation Calculations

The flotation calculations for our boat is shown here.

The surface area of the hull utilizing one layer of 1.25 inch of Nomex honeycomb is 65.0 ft2 and the surface area which utilizes two layers of 0.472 inches of Nomex honeycomb is 7.1 ft2. Thus, the buoyant force provided by the hull alone, neglecting the Kevlar skins is given by Archimedes' principle:

$$BH = \gamma H2O \cdot \sum Aiti\ i\ = (62.4lbft3) \cdot [(65.0\ ft2)(1.2512ft) + 2(7.1\ ft2)(0.47212ft)]$$
$$= 468\ lb$$

where BH is the buoyant force on the hull when submerged, Ai is the surface area covered by a given core thickness, ti is thickness of the core in a given region, and $\gamma H2O$ is the specific gravity of water. Because the batteries are secured to the hull, their buoyant force also contributes the overall buoyant force on the boat. The volume of three 42 EP batteries is less than that of twelve 13 EP batteries and will therefore be used for our calculations.

$$BB\ = 3V42EP \cdot \gamma H2O$$

$$= 3(0.175\ ft3)(62.4lbft3)$$

$$= 33\ lb$$

where BB is the buoyant force of the batteries and $V42EP$ is the volume of the Genesis 42EP batteries. Therefore, the maximum possible buoyant force exerted on the hull is given by the following.

$$Btot\ = BH + BB$$

$$= 468\ lb + 33\ lb$$

$$= 501\ lb$$

Also, the weight of the hull, as given by the power budget, is shown in Table B.1. Based on our calculations, our new hull can easily support its own weight plus a 20% safety factor as the buoyant force of 501 lb. is much greater than the required buoyant force of 335 lb.

Table B.1. *Components and Weight List*

Components	Weight [lb]	
	2023 Sprint	**2023 Endurance**
Solar Array	N/A	30
Batteries	100	100
Drivetrain & Controller	72	72
Hull	53	53
MPPT	N/A	4
Control Panel	5	5
Fabric Fairing	5	5
Miscellaneous	10	10
Total	**245**	**279**
120% Total (Rule 7.14.2)	**294**	**335**

Appendix C. Proof of Insurance

This is our proof of insurance.

ACORD

CEDAR-3 OP ID: CW

CERTIFICATE OF LIABILITY INSURANCE

DATE (MM/DD/YYYY) 04/19/2023

THIS CERTIFICATE IS ISSUED AS A MATTER OF INFORMATION ONLY AND CONFERS NO RIGHTS UPON THE CERTIFICATE HOLDER. THIS CERTIFICATE DOES NOT AFFIRMATIVELY OR NEGATIVELY AMEND, EXTEND OR ALTER THE COVERAGE AFFORDED BY THE POLICIES BELOW. THIS CERTIFICATE OF INSURANCE DOES NOT CONSTITUTE A CONTRACT BETWEEN THE ISSUING INSURER(S), AUTHORIZED REPRESENTATIVE OR PRODUCER, AND THE CERTIFICATE HOLDER.

IMPORTANT: If the certificate holder is an ADDITIONAL INSURED, the policy(ies) must have ADDITIONAL INSURED provisions or be endorsed. If SUBROGATION IS WAIVED, subject to the terms and conditions of the policy, certain policies may require an endorsement. A statement on this certificate does not confer rights to the certificate holder in lieu of such endorsement(s).

PRODUCER 937-324-8492
Wallace & Turner, Inc.
P. O. Box 209
30 Warder Street #200
Springfield, OH 45501-0209
Patrick E. Field

CONTACT NAME: Patrick E. Field
PHONE (A/C, No, Ext): 937-324-8492 FAX (A/C, No): 937-325-1069
E-MAIL ADDRESS: pfield@wtins.com

INSURER(S) AFFORDING COVERAGE	NAIC #
INSURER A : Cincinnati Insurance Company	10677
INSURER B :	
INSURER C :	
INSURER D :	
INSURER E :	
INSURER F :	

INSURED
Cedarville University
251 North Main Street
Cedarville, OH 45314

COVERAGES CERTIFICATE NUMBER: REVISION NUMBER:

THIS IS TO CERTIFY THAT THE POLICIES OF INSURANCE LISTED BELOW HAVE BEEN ISSUED TO THE INSURED NAMED ABOVE FOR THE POLICY PERIOD INDICATED. NOTWITHSTANDING ANY REQUIREMENT, TERM OR CONDITION OF ANY CONTRACT OR OTHER DOCUMENT WITH RESPECT TO WHICH THIS CERTIFICATE MAY BE ISSUED OR MAY PERTAIN, THE INSURANCE AFFORDED BY THE POLICIES DESCRIBED HEREIN IS SUBJECT TO ALL THE TERMS, EXCLUSIONS AND CONDITIONS OF SUCH POLICIES. LIMITS SHOWN MAY HAVE BEEN REDUCED BY PAID CLAIMS.

INSR LTR	TYPE OF INSURANCE	ADDL INSD	SUBR WVD	POLICY NUMBER	POLICY EFF (MM/DD/YYYY)	POLICY EXP (MM/DD/YYYY)	LIMITS	
A	X COMMERCIAL GENERAL LIABILITY; CLAIMS-MADE; X OCCUR; X Empl Liab Defense; GEN'L AGGREGATE LIMIT APPLIES PER: X POLICY; PRO-JECT; LOC; OTHER:			ETD0492220	07/01/2022	07/01/2023	EACH OCCURRENCE	$ 1,000,000
							DAMAGE TO RENTED PREMISES (Ea occurrence)	$ 500,000
							MED EXP (Any one person)	$ 10,000
							PERSONAL & ADV INJURY	$ 1,000,000
							GENERAL AGGREGATE	$ 3,000,000
							PRODUCTS - COMP/OP AGG	$ 3,000,000
								$
A	AUTOMOBILE LIABILITY; X ANY AUTO; OWNED AUTOS ONLY; SCHEDULED AUTOS; X HIRED AUTOS ONLY; X NON-OWNED AUTOS ONLY			ETD0492220	07/01/2022	07/01/2023	COMBINED SINGLE LIMIT (Ea accident)	$ 1,000,000
							BODILY INJURY (Per person)	$
							BODILY INJURY (Per accident)	$
							PROPERTY DAMAGE (Per accident)	$
								$
A	X UMBRELLA LIAB; X OCCUR; EXCESS LIAB; CLAIMS-MADE; DED; X RETENTION $ 0			ETD0492220	07/01/2022	07/01/2023	EACH OCCURRENCE	$ 15,000,000
							AGGREGATE	$ 15,000,000
								$
A	WORKERS COMPENSATION AND EMPLOYERS' LIABILITY; Y/N; ANY PROPRIETOR/PARTNER/EXECUTIVE OFFICER/MEMBER EXCLUDED? (Mandatory in NH) N; If yes, describe under DESCRIPTION OF OPERATIONS below	N / A		ETD0492220	07/01/2022	07/01/2023	PER STATUTE; OTH-ER	
							E.L. EACH ACCIDENT	$ 1,000,000
							E.L. DISEASE - EA EMPLOYEE	$ 1,000,000
							E.L. DISEASE - POLICY LIMIT	$ 1,000,000

DESCRIPTION OF OPERATIONS / LOCATIONS / VEHICLES (ACORD 101, Additional Remarks Schedule, may be attached if more space is required)
Event: Solar Splash 2021 held June 6-10, 2023 at lake adjoining Clark County Fairgrounds, 4401 S. Charleston Pike, Springfield, Ohio 45505

CERTIFICATE HOLDER SOLASPL
Solar Splash 2023
Attn: Jeff Morehouise
309 Newridge Road
Lexington, SC 29072

CANCELLATION
SHOULD ANY OF THE ABOVE DESCRIBED POLICIES BE CANCELLED BEFORE THE EXPIRATION DATE THEREOF, NOTICE WILL BE DELIVERED IN ACCORDANCE WITH THE POLICY PROVISIONS.

AUTHORIZED REPRESENTATIVE
Patrick E. Field

ACORD 25 (2016/03)

Appendix D. Team Roster

This appendix lists the roster for our team. For each member you will find their name, degree program, year, and role.

Asa Mudd – BSME Mechanical Engineering – Senior – Team captain, data acquisition, driver interface

Bryce Schmitt – BSME Mechanical Engineering – Senior – Motors, motor controllers, dynamometer

Joseph Hoyman – BSME Mechanical Engineering – Senior – Propellers

Nicolas Knowlton – BSME Mechanical Engineering – Senior – CAN adapters and steering system

Ruth Lessen – BSME Mechanical Engineering – Senior – Hydrofoil boat battery pack

Joseph Heise – BSME Mechanical Engineering – Senior – Hydrofoil boat flight control system, sensors, and actuators

Daniel Shoultz – BSME Mechanical Engineering – Senior – Hydrofoil boat strut design

Grace Fearday – BS Electrical Engineering – Senior – Electrical

Kezia Augusting – BS Electrical Engineering – Senior - Electrical

Cejay Walker – BS Electrical Engineering – Senior - Electrical

Appendix E. Motor Testing

This appendix has information regarding the motor testing.

The following one drive link to the Dyno Operation folder contains all the starting information necessary to get the dyno operational.

https://cedarvilleuniversity394.sharepoint.com/:f:/r/sites/EngineeringStudents-SolarBoat/Shared%20Documents/Solar%20Boat/Electrical%20Reference%20Material/Dyno%20Operation?csf=1&web=1&e=lfOlI6

Within this folder is the Dyno operation manual, as well as videos detailing the operation of the Dyno and Drivetool. Finally, there is a video detailing how to align the rpm encoder as well as the first motor controller software files you will need to use. The Dyno user manual and how to video are the best places to start.

Drivetool is the software used to program the Inmotion drives and operate them while they are on the dyno.

Using Drivetool:

1. Launch Drivetool from the Dyno room computer.
2. Make sure the drive is connected to the computer via the Kvaser CAN adapter.
3. Click the load settings button in the upper left corner. (The current file (4/24/23) can be found in the electrical reference material folder under Dyno operation, Drivetool files).
4. Ensure the drive has power from the battery box.
5. Click start communications button. You are now communicating with the drive.
6. After this, go to the Par A tap on the right side of the screen as shown in **Fig. E.1**.

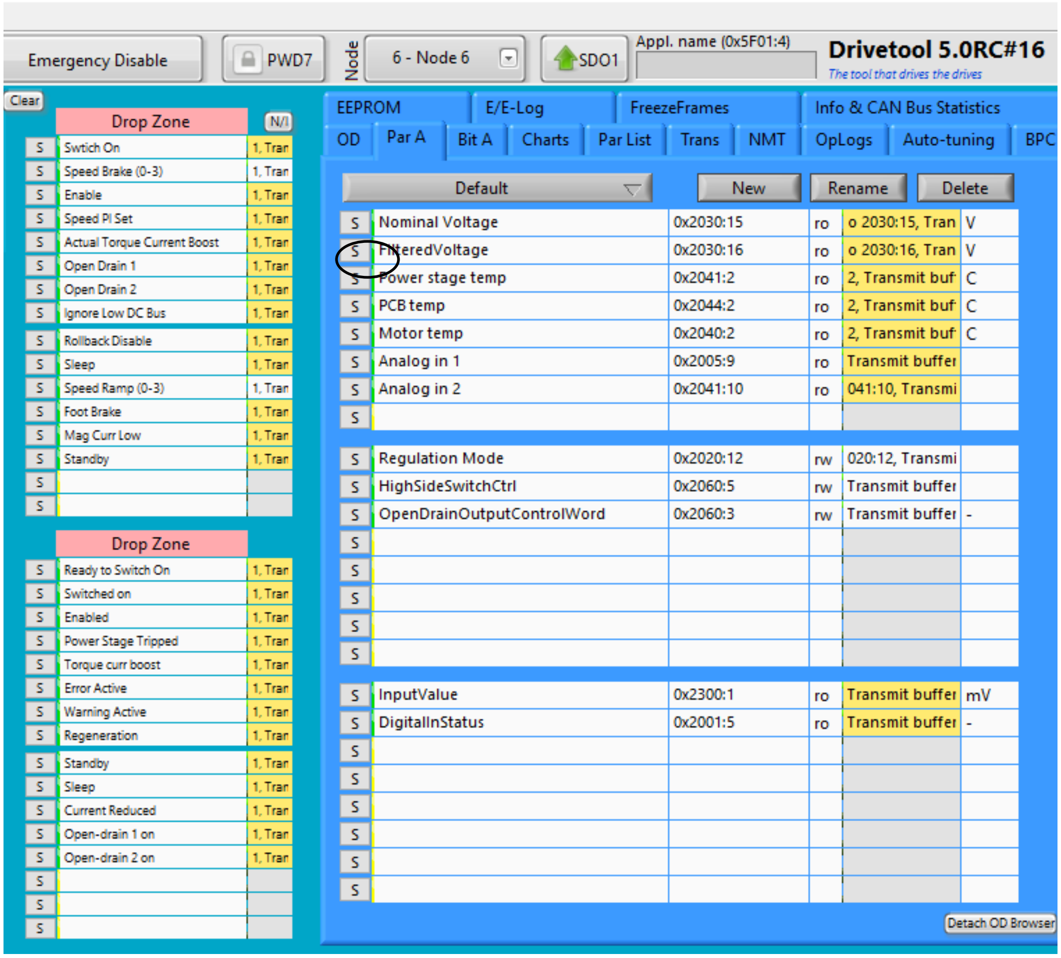


Fig. E.1. *Drivetool settings window.*

7. You will then hit the drop-down menu below Par A. This shows you what the default settings are to run the motor from the computer. (If wanting to run in backup mode from control panel, see #12.)
8. Next, go and turn the control panel Aux Power switch to the on position. This will make all the yellow boxes in Fig.13.1.2 green or white.
9. After that, go to drivetool and click in the OpenDrainOutputControlWord boxes right of this in the defaults window and change the 0 to a 1. (If done successfully, the contactors will close, and you should hear a popping sound, often times you have to close and open this a few times to get it to close correctly.)
10. Finally, all that is needed next is to hit Switch On and Enable on to the "on" position. (Note: once enable is on, you should hear a hum coming from the motor.) See **Fig. E.2** for where the location of Switch On and Enable is.

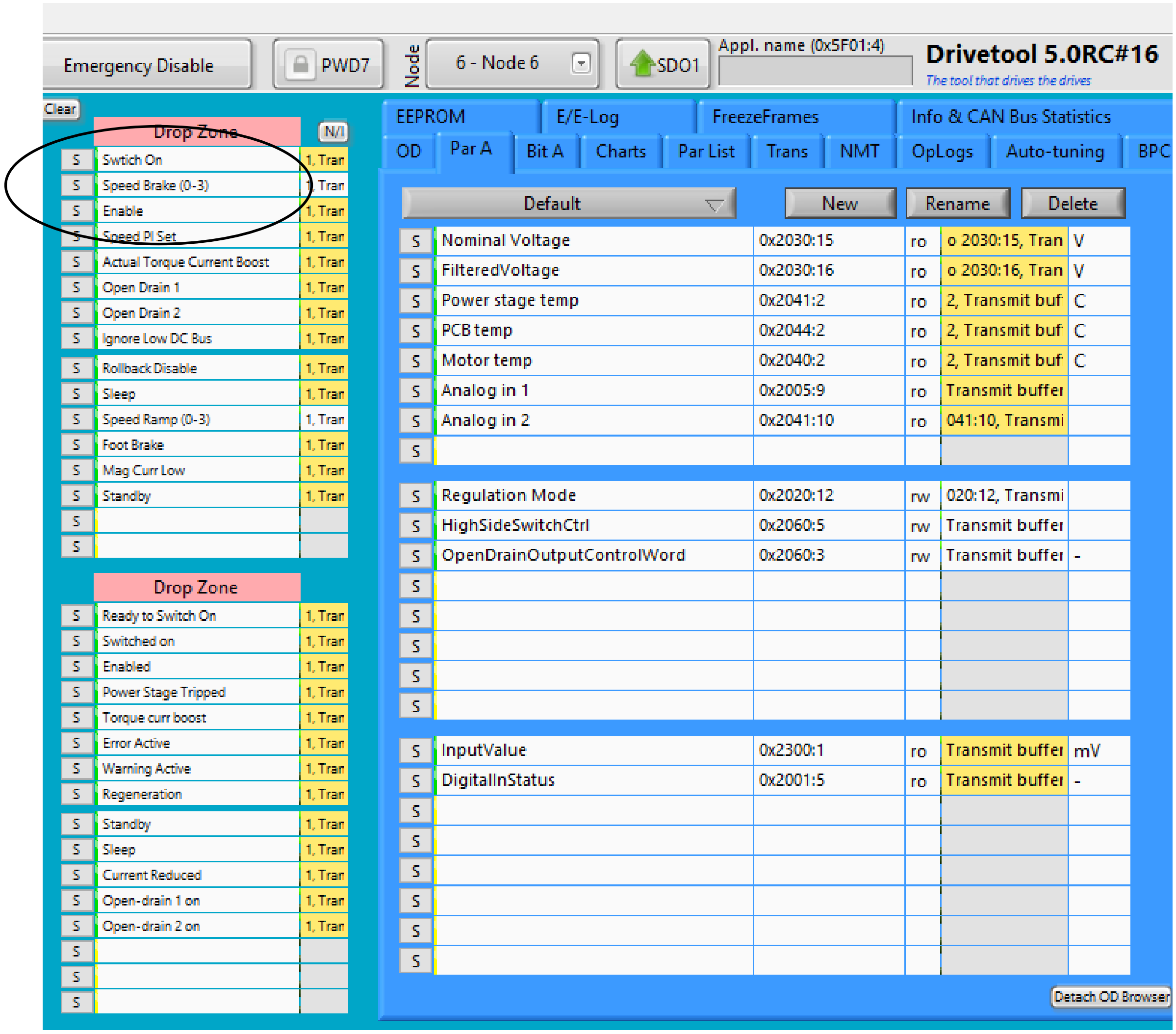


Fig. E.2. *Drivetool settings window after in Par A.*

11. You can now run the motor from the computer.
12. If you would want to run the motor in backup mode from the control panel, go to the default drop-down and click application setup.
13. Once in application set up, you will need to hit PWD7 to unlock the window and change the Application enable to 1. Can be seen in **Fig. E.3**

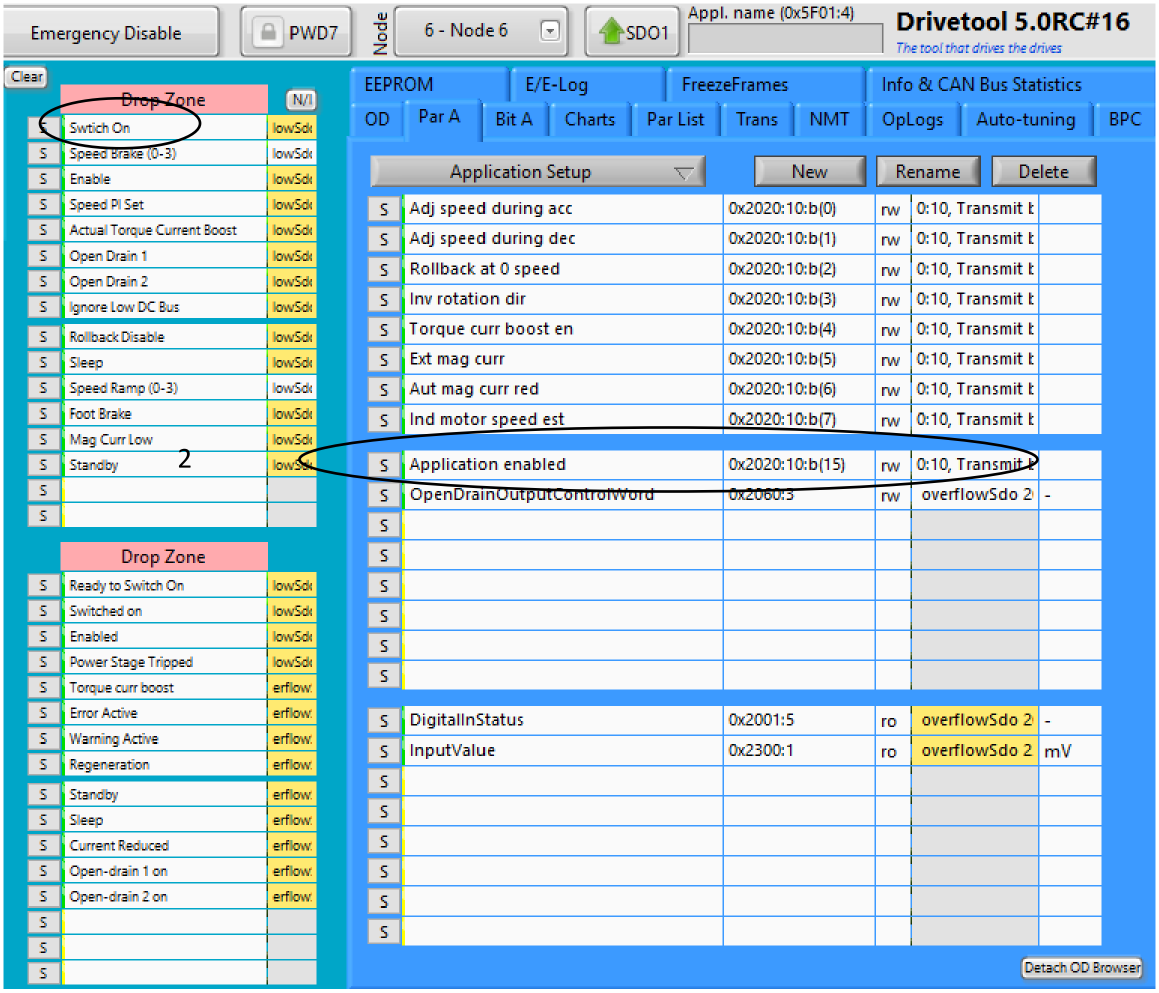


Fig. E.3. *DriveTool settings window after in Application Setup.*

14. Once here, you can resume as normal by turning Aux Power to On instead of going to turning the contactors on like before, just turn the red switch on the control panel labeled MOTOR to on and then you can run the motor from the control panel.

Appendix F. Steering Analysis

The photos in this appendix were used to develop our steering system and dynamic TK Solver code.

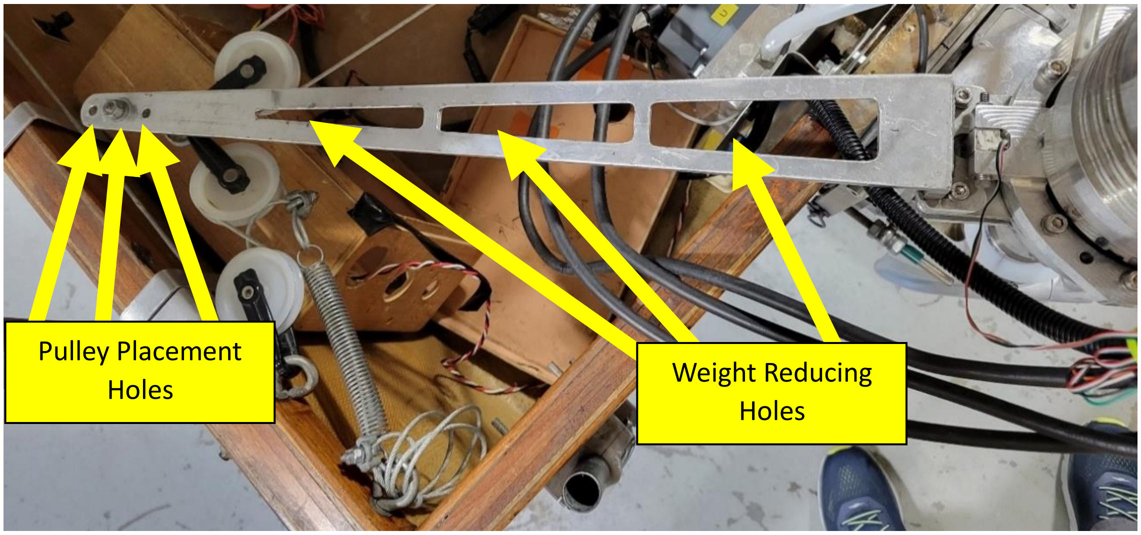


Fig F.1. *SS design limitations of original tiller arm*

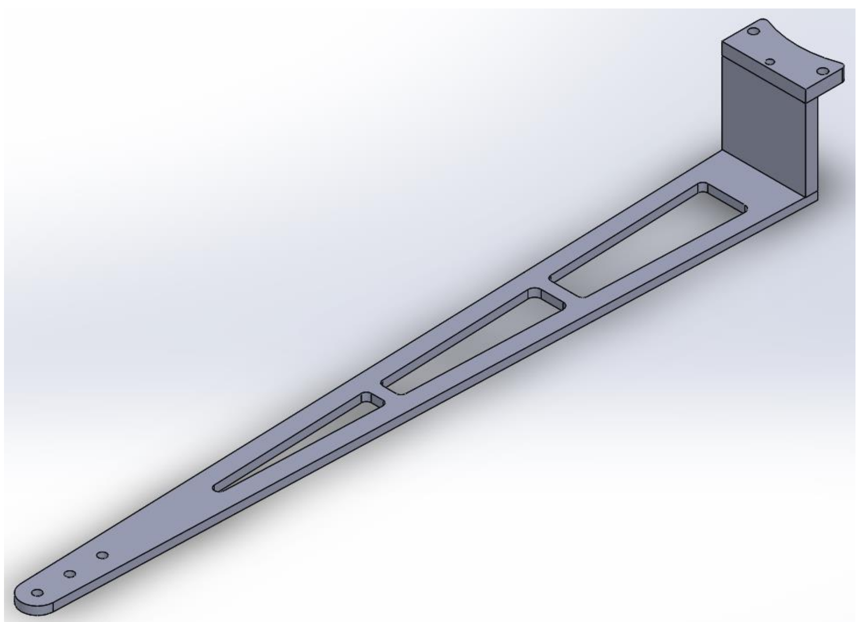


Fig F.2. *SS original tiller arm design*

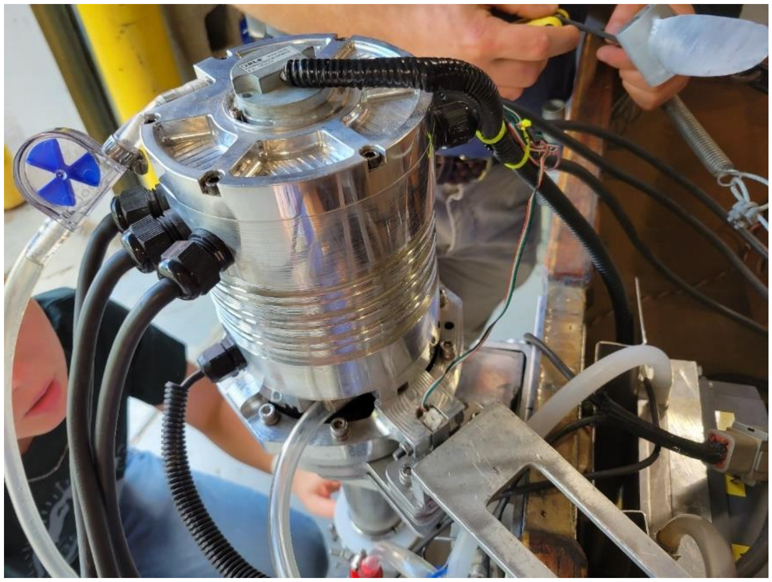
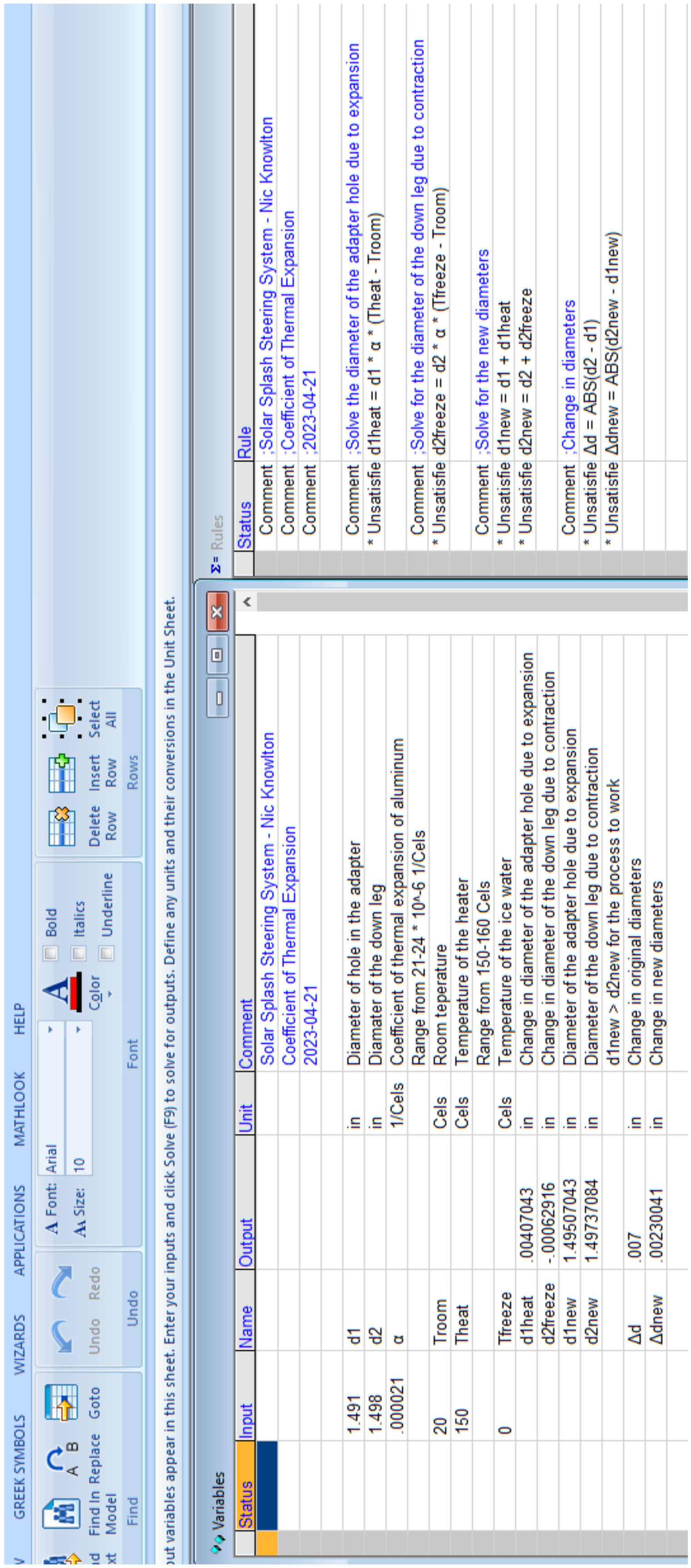


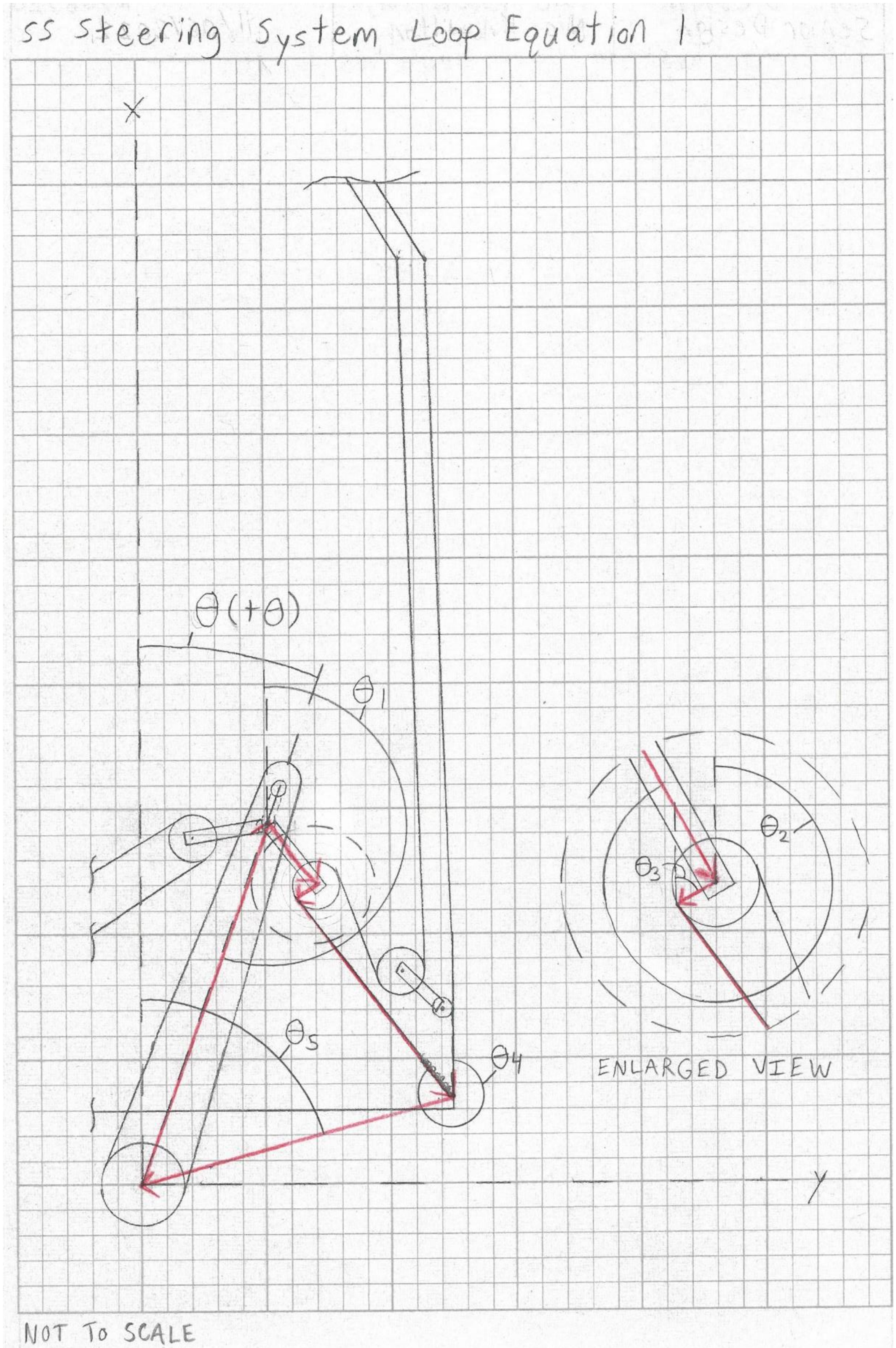
Fig F.3. *Original orientation of torque load cell on Solar Splash boat*



Status	Input	Name	Output	Unit	Comment
					Solar Splash Steering System - Nic Knowlton
					Coefficient of Thermal Expansion
					2023-04-21
	1.491	d1		in	Diameter of hole in the adapter
	1.498	d2		in	Diamater of the down leg
	.000021	α		1/Cels	Coefficient of thermal expansion of aluminum
					Range from 21-24 * 10^-6 1/Cels
	20	Troom		Cels	Room teperature
	150	Theat		Cels	Temperature of the heater
					Range from 150-160 Cels
	0	Tfreeze		Cels	Temperature of the ice water
		d1heat	.00407043	in	Change in diameter of the adapter hole due to expansion
		d2freeze	-.00062916	in	Change in diameter of the down leg due to contraction
		d1new	1.49507043	in	Diameter of the adapter hole due to expansion
		d2new	1.49737084	in	Diameter of the down leg due to contraction
					d1new > d2new for the process to work
		Δd	.007	in	Change in original diameters
		Δdnew	.00230041	in	Change in new diameters

Status	Rule
Comment	;Solar Splash Steering System - Nic Knowlton
Comment	;Coefficient of Thermal Expansion
Comment	;2023-04-21
Comment	;Solve the diameter of the adapter hole due to expansion
* Unsatisfie	d1heat = d1 * α * (Theat - Troom)
Comment	;Solve for the diameter of the down leg due to contraction
* Unsatisfie	d2freeze = d2 * α * (Tfreeze - Troom)
Comment	;Solve for the new diameters
* Unsatisfie	d1new = d1 + d1heat
* Unsatisfie	d2new = d2 + d2freeze
Comment	;Change in diameters
* Unsatisfie	Δd = ABS(d2 - d1)
* Unsatisfie	Δdnew = ABS(d2new - d1new)

Fig E.12. Fig E.4. *SS TK Solver calculation for coefficients of thermal expansion for the drive train and Yamato adapter*



FigF.5. *SS kinematic loop 1*

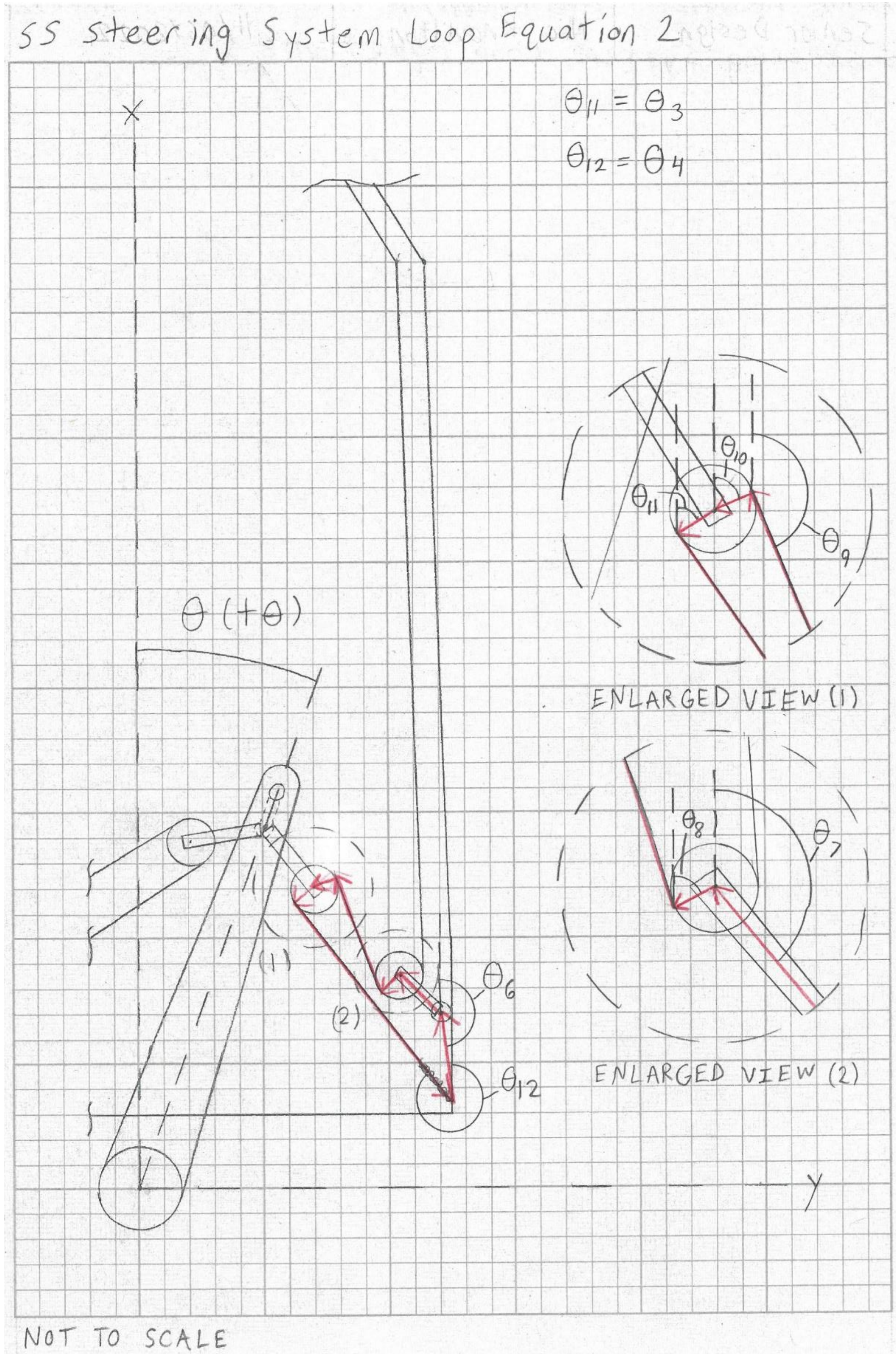


Fig F.6. *SS kinematic loop 2*

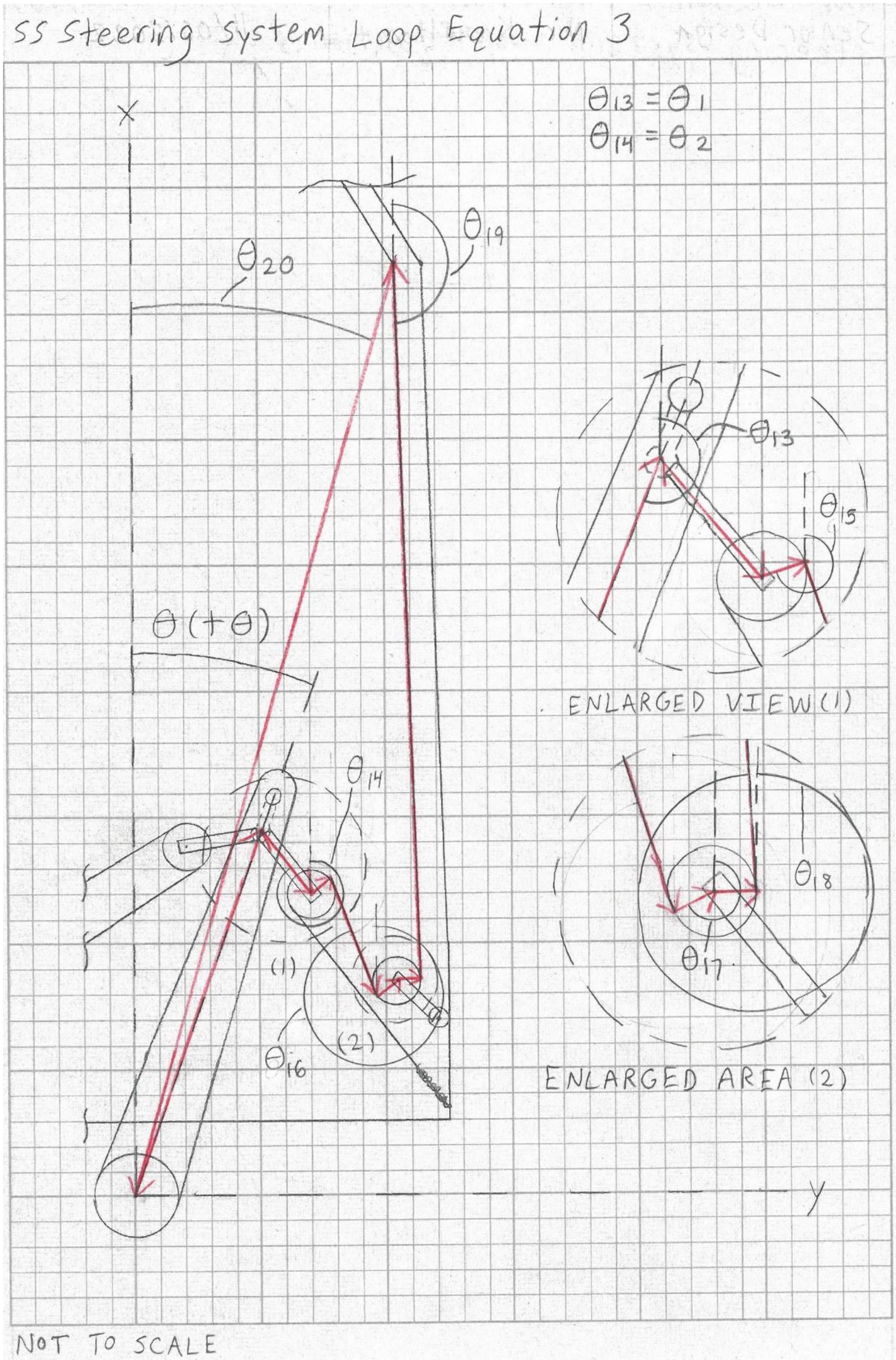


Fig F.7. *SS kinematic loop 3*

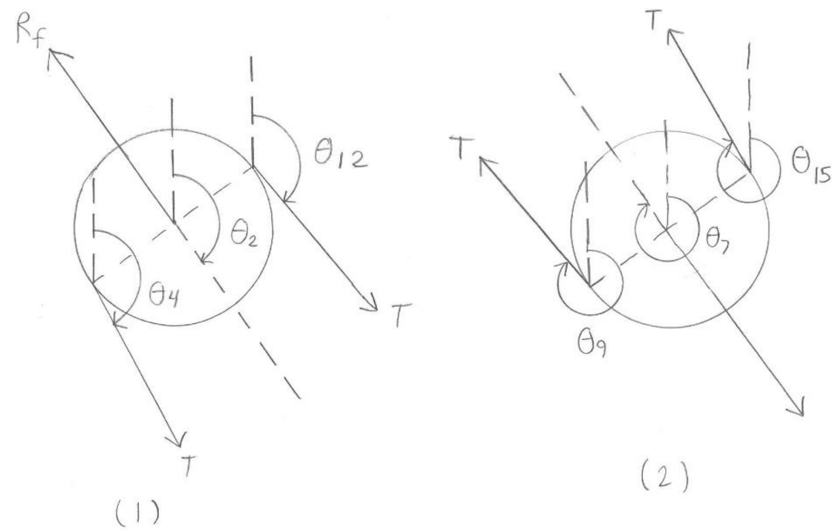


Fig F.8. *SS geometry of pulley 1 and 2*

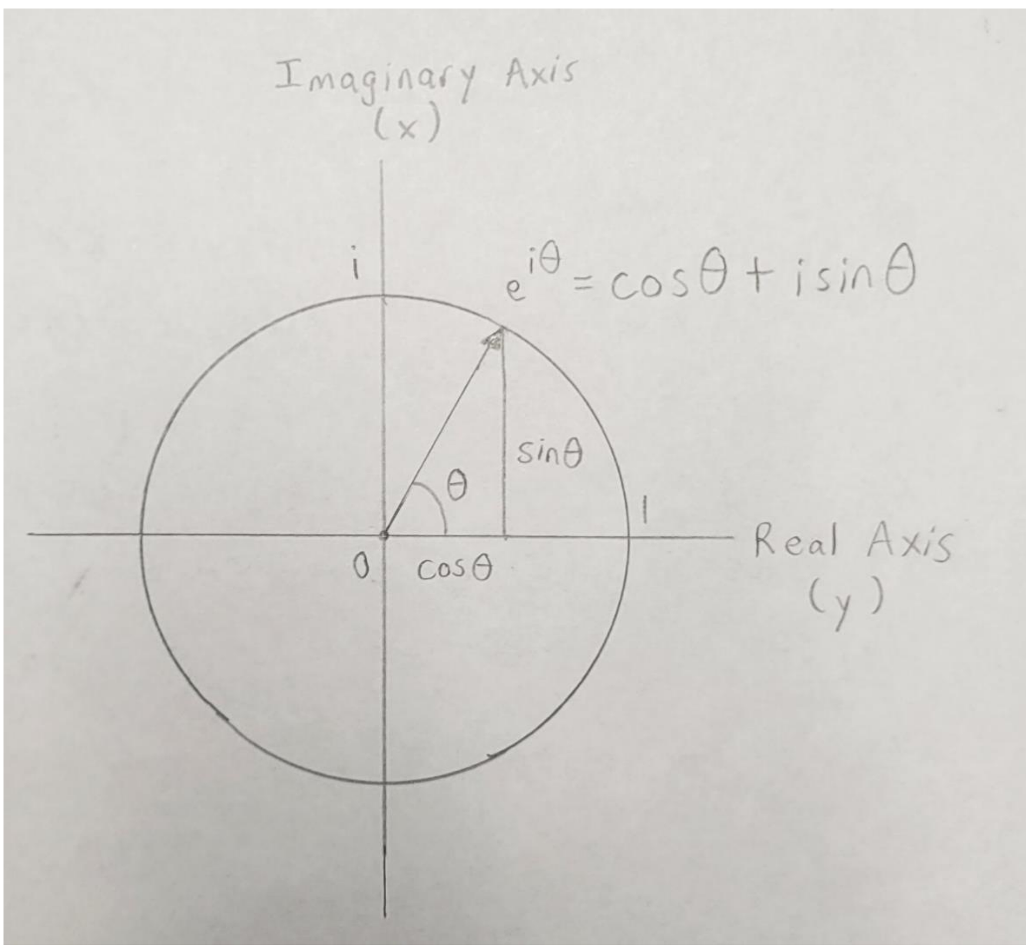
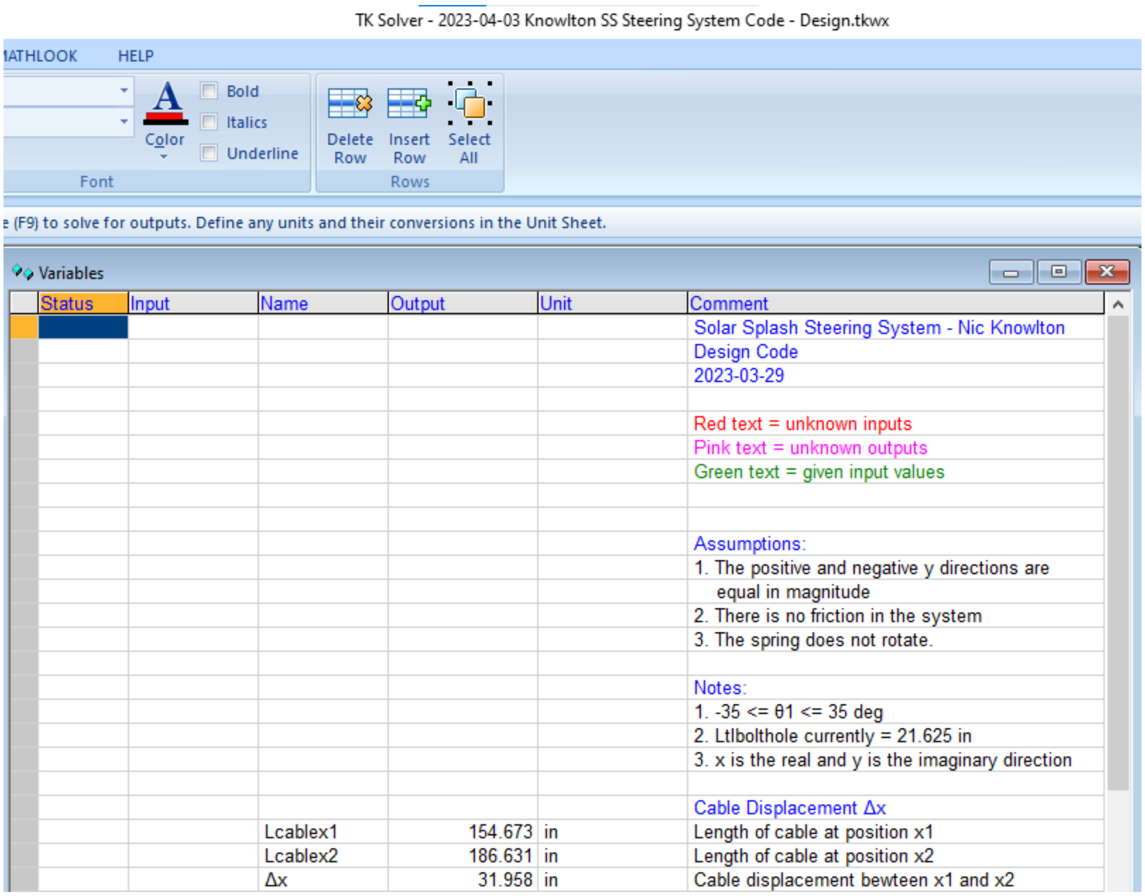


Fig F.9. *Euler's formula geometry*



Status	Input	Name	Output	Unit	Comment
					Solar Splash Steering System - Nic Knowlton
					Design Code
					2023-03-29
					Red text = unknown inputs
					Pink text = unknown outputs
					Green text = given input values
					Assumptions:
					1. The positive and negative y directions are
					equal in magnitude
					2. There is no friction in the system
					3. The spring does not rotate.
					Notes:
					1. -35 <= θ1 <= 35 deg
					2. Ltlbolthole currently = 21.625 in
					3. x is the real and y is the imaginary direction
					Cable Displacement Δx
		Lcablex1	154.673	in	Length of cable at position x1
		Lcablex2	186.631	in	Length of cable at position x2
		Δx	31.958	in	Cable displacement bewteen x1 and x2

Fig F.10. *SS TK Solver design code variables sheet – part 1 of 3*

Variables

Status	Input	Name	Output	Unit	Comment
					Cable Displacement Δx
		Lcablex1	154.673	in	Length of cable at position x1
		Lcablex2	186.631	in	Length of cable at position x2
		Δx	31.958	in	Cable displacement bewteen x1 and x2
					Tiller arm
	11.000	LtIbolthole		in	Length from motor end of tiller arm to center of I-bolt hole holding pulleys 1
	1.438	Lttip		in	Length from center of I-bolt hole to tip of tiller arm
	1.250	LIbolt		in	Length from center of I-bolt hole to center of I-bolt ring
		Ltarm	12.438	in	Total length of tiller arm
	2.875	Lmot		in	Length from center of motor to motor end of tiller arm
					Pulleys
	2.188	Pd		in	Inner diamater of pulley
		Pr	1.094	in	Inner radius of pulley
	2.500	Parm		in	Length of pulley arm
					Known Values for Both Positions x1 and x2
		r1	13.875	in	Magnitude (length) of vector 1 -- tiller arm
		r2	2.500	in	Magnitude (length) of vector 2 -- pulley arm
		r3	1.094	in	Magnitude (length) of vector 3 -- pulley radius
	9.000	r5r		in	Real component of vector 5 -- not real part
	16.733	r5i		in	Imaginary component of vector 5 -- not real part
		r5	19.000	in	Magnitude (length) of vector 5 -- not real part
	245.000	θ5		deg	Angle of vector 5
	3.873	r6r		in	Real component of vector 6 -- not real part
	-1.000	r6i		in	Imaginary component of vector 6 -- not real part
		r6	4.000	in	Magnitude (length) of vector 6 -- not real part
	357.000	θ6		deg	Angle of vector 6
		r7	2.500	in	Magnitude (length) of vector 7 -- pulley arm
		r8	1.094	in	Magnitude (length) of vector 8 -- pulley radius
		r10	1.094	in	Magnitude (length) of vector 10 -- pulley radius
		r11	1.094	in	Magnitude (length) of vector 11 -- pulley radius
		r13	1.094	in	Magnitude (length) of vector 13 -- pulley radius
		r14	1.094	in	Magnitude (length) of vector 14 -- pulley radius
	147.250	r16		in	Magnitude (length) of vector 16 -- not real part
	185.000	θ16		deg	Angle of vector 16
					Positive θ1 Side for Position x1 (+y direction)
	35.000	θ1x1		deg	Angle of vector 1
		θ2x1	95.706	deg	Angle of vector 2
		θ3x1	196.055	deg	Angle of vector 3

Fig F.11. *SS TK Solver design code variables sheet – part 2 of 3*

Variables

Status	Input	Name	Output	Unit	Comment
					Positive θ1 Side for Position x1 (+y direction)
	35.000	θ1x1		deg	Angle of vector 1
		θ2x1	95.706	deg	Angle of vector 2
		θ3x1	196.055	deg	Angle of vector 3
		r4x1	7.363	in	Magnitude (length) of vector 4
F		θ4x1	106.055	deg	Angle of vector 4
		θ7x1	311.906	deg	Angle of vector 7
F		θ8x1	175.356	deg	Angle of vector 8
F		r9x1	4.897	in	Magnitude (length) of vector 9
		θ9x1	265.356	deg	Angle of vector 9
		θ10x1	175.356	deg	Angle of vector 10
		θ11x1	4.644	deg	Angle of vector 11
		r12x1	4.897	in	Magnitude (length) of vector 12
F		θ12x1	85.356	deg	Angle of vector 12
		θ13x1	-4.644	deg	Angle of vector 13
F		θ14x1	88.455	deg	Angle of vector 14
F		r15x1	133.015	in	Magnitude (length) of vector 15
		θ15x1	358.455	deg	Angle of vector 15
					Positive θ1 Side for Position x2 (+y direction)
	-35.000	θ1x2		deg	Angle of vector 1
		θ2x2	92.275	deg	Angle of vector 2
		θ3x2	185.386	deg	Angle of vector 3
		r4x2	22.884	in	Magnitude (length) of vector 4
F		θ4x2	95.386	deg	Angle of vector 4
		θ7x2	313.829	deg	Angle of vector 7
F		θ8x2	179.165	deg	Angle of vector 8
F		r9x2	20.701	in	Magnitude (length) of vector 9
		θ9x2	269.165	deg	Angle of vector 9
		θ10x2	179.165	deg	Angle of vector 10
		θ11x2	.835	deg	Angle of vector 11
		r12x2	20.701	in	Magnitude (length) of vector 12
F		θ12x2	89.165	deg	Angle of vector 12
		θ13x2	-.835	deg	Angle of vector 13
F		θ14x2	88.492	deg	Angle of vector 14
F		r15x2	132.951	in	Magnitude (length) of vector 15
		θ15x2	358.492	deg	Angle of vector 15

Fig F.12. *SS TK Solver design code variables sheet – part 3 of 3*

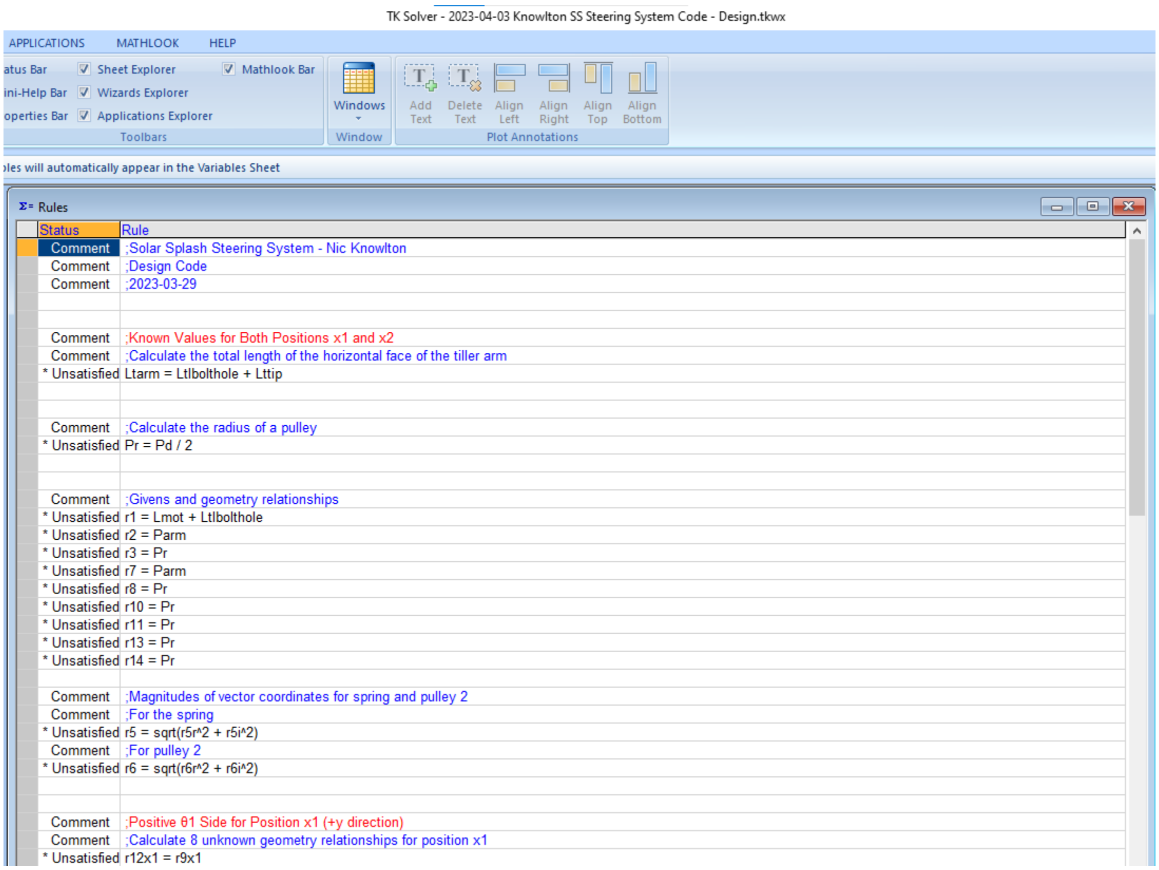


Fig F.13. *SS TK Solver design code rules sheet – part 1 of 3*

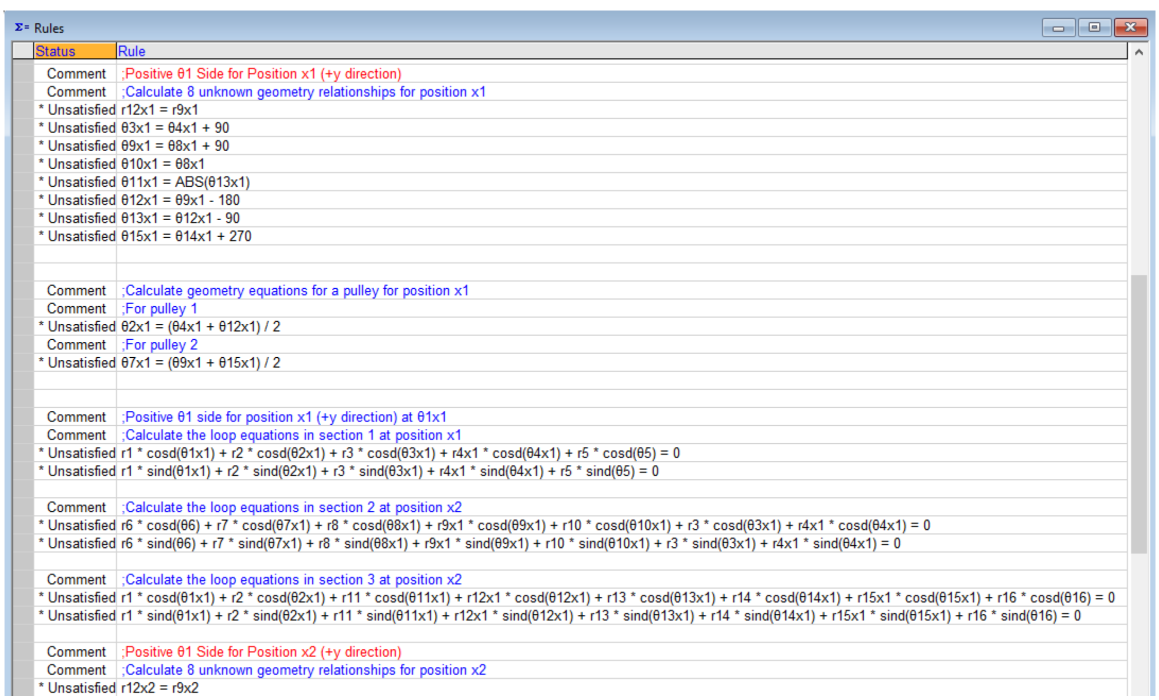


Fig F.14. *SS TK Solver design code rules sheet – part 2 of 3*

Σ= Rules

Status	Rule
Comment	;Positive θ1 Side for Position x2 (+y direction)
Comment	;Calculate 8 unknown geometry relationships for position x2
* Unsatisfied	r12x2 = r9x2
* Unsatisfied	θ3x2 = θ4x2 + 90
* Unsatisfied	θ9x2 = θ8x2 + 90
* Unsatisfied	θ10x2 = θ8x2
* Unsatisfied	θ11x2 = ABS(θ13x2)
* Unsatisfied	θ12x2 = θ9x2 - 180
* Unsatisfied	θ13x2 = θ12x2 - 90
* Unsatisfied	θ15x2 = θ14x2 + 270
Comment	;Calculate geometry equations for a pulley for position x2
Comment	;For pulley 1
* Unsatisfied	θ2x2 = (θ4x2 + θ12x2) / 2
Comment	;For pulley 2
* Unsatisfied	θ7x2 = (θ9x2 + θ15x2) / 2
Comment	;Positive θ1 side for position x2 (+y direction) at θ1x2
Comment	;Calculate the loop equations in section 1 at position x2
* Unsatisfied	r1 * cosd(θ1x2) + r2 * cosd(θ2x2) + r3 * cosd(θ3x2) + r4x2 * cosd(θ4x2) + r5 * cosd(θ5) = 0
* Unsatisfied	r1 * sind(θ1x2) + r2 * sind(θ2x2) + r3 * sind(θ3x2) + r4x2 * sind(θ4x2) + r5 * sind(θ5) = 0
Comment	;Calculate the loop equations in section 2 at position x2
* Unsatisfied	r6 * cosd(θ6) + r7 * cosd(θ7x2) + r8 * cosd(θ8x2) + r9x2 * cosd(θ9x2) + r10 * cosd(θ10x2) + r3 * cosd(θ3x2) + r4x2 * cosd(θ4x2) = 0
* Unsatisfied	r6 * sind(θ6) + r7 * sind(θ7x2) + r8 * sind(θ8x2) + r9x2 * sind(θ9x2) + r10 * sind(θ10x2) + r3 * sind(θ3x2) + r4x2 * sind(θ4x2) = 0
Comment	;Calculate the loop equations in section 3 at position x2
* Unsatisfied	r1 * cosd(θ1x2) + r2 * cosd(θ2x2) + r11 * cosd(θ11x2) + r12x2 * cosd(θ12x2) + r13 * cosd(θ13x2) + r14 * cosd(θ14x2) + r15x2 * cosd(θ15x2) + r16 * cosd(θ16) = 0
* Unsatisfied	r1 * sind(θ1x2) + r2 * sind(θ2x2) + r11 * sind(θ11x2) + r12x2 * sind(θ12x2) + r13 * sind(θ13x2) + r14 * sind(θ14x2) + r15x2 * sind(θ15x2) + r16 * sind(θ16) = 0
Comment	;Cable Displacement Δx
Comment	;Calculate total lengths of cable at positions x1 and x2
* Unsatisfied	Lcablex1 = r4x1 + r9x1 + r15x1 + (2 * Pr * 0.01745 * (180 + (θ10x1 - θ3x1))) + (2 * Pr * 0.01745 * (180 + θ13x1 - θ14x1))
* Unsatisfied	Lcablex2 = r4x2 + r9x2 + r15x2 + (2 * Pr * 0.01745 * (180 + (θ10x2 - θ3x2))) + (2 * Pr * 0.01745 * (180 + θ13x2 - θ14x2))
Comment	;Calculate cable displacement bewteen positions x1 and x2
* Unsatisfied	Δx = ABS(Lcablex1 - Lcablex2)

Fig F.15. *SS TK Solver design code rules sheet – part 3 of 3*

Appendix G. Propeller Design

This appendix shows how a propeller design is brought into Solidworks.

Propeller design starts by downloading OpenProp (https://cedarvilleuniversity394.sharepoint.com/:f:/r/sites/EngineeringStudents-SolarBoat/Shared Documents/Solar Boat/2022-2023/2. Individual Work/5. Joseph Hoyman/2023AdvancedOpenPropLearningandTesting-NotImportant/OpenProp_v3.3.4 Iterative CRP Use this One?csf=1&web=1&e=Y90we6) and opening it in MATLAB. Once it is opened, input operating parameters, and run the program to determine the efficiency and power draw of the design. Upon determining a good design, save the design to the OpenProp folder and use the OpenProp-to-SolidWorks Macro (https://cedarvilleuniversity394.sharepoint.com/:u:/r/sites/EngineeringStudents-SolarBoat/Shared Documents/Solar Boat/2022-2023/2. Individual Work/5. Joseph Hoyman/2023AdvancedOpenPropLearningandTesting-NotImportant/OpenProp to Solidworks Macro.swp?csf=1&web=1&e=aoJYab) to generate geometry sketch contours in a SolidWorks hub file. Once in SolidWorks select each of the geometry sketches in order, starting at the hub, and use the *Lofted Surface* feature to make the 3D shape of one blade (Figure F.1). Finally, use the *Circular Pattern* feature to copy the blade shape around the hub.

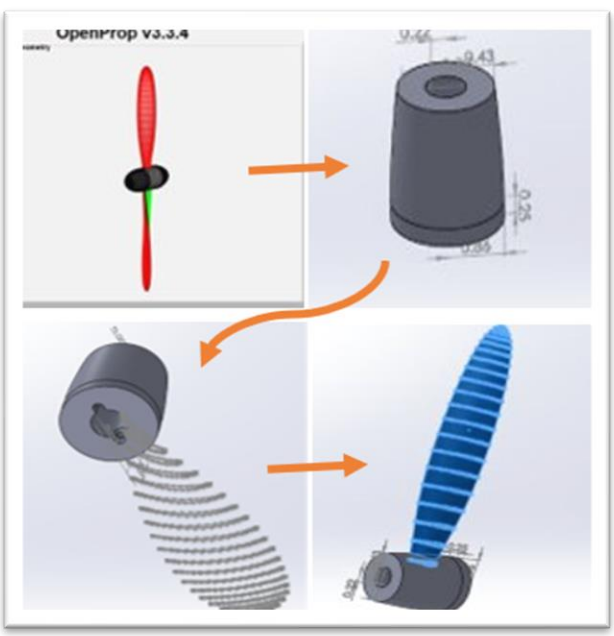


Fig G.1: *The process starts with generating an OpenProp Design, then finding or making a hub in SolidWorks, then generating geometry contours in SolidWorks using the Macro, and finally using lofted surface to generate the 3D blade shape.*

A very helpful tool for understanding the entire process can be found in the overview video made by Caleb Tanner (https://cedarvilleuniversity394.sharepoint.com/:v:/r/sites/EngineeringStudents-SolarBoat/Shared Documents/Solar Boat/2022-2023/2. Individual Work/5. Joseph Hoyman/2023AdvancedOpenPropLearningandTesting-NotImportant/1 Modeled D Fore 1300 Guide.mp4?csf=1&web=1&e=kc6tKq). This video walks through the entire process of turning given design parameters into a SolidWorks Model.

Appendix H. Hull Drag and Motor Torque Data Acquisition

Determining Drag Values from 2022 Race Data

In 2023, we did not have the boat operational for most of the school year and we were therefore not able to find good drag values .The best data we had was one set of 2022 Race Data (https://cedarvilleuniversity394.sharepoint.com/:x:/r/sites/EngineeringStudents-SolarBoat/Shared Documents/Solar Boat/2022-2023/2. Individual Work/5. Joseph Hoyman/2023FirstPropDesignAndDataAnalysis-DIDNOTWORK/2023SprintDataReduction-Basedon2022RaceData.xlsx?d=w812586f20a7e4e868b015af32e241973&csf=1&web=1&e=SnUDeK).

Finding Thrust Using the Power Budget

To reduce this data, we used a power budget model. The idea was to use the known power given by the batteries and assume an efficiency of the motor and prop (**Equations H.1-H.2**) The battery power is shown in ***Fig. H.1***.

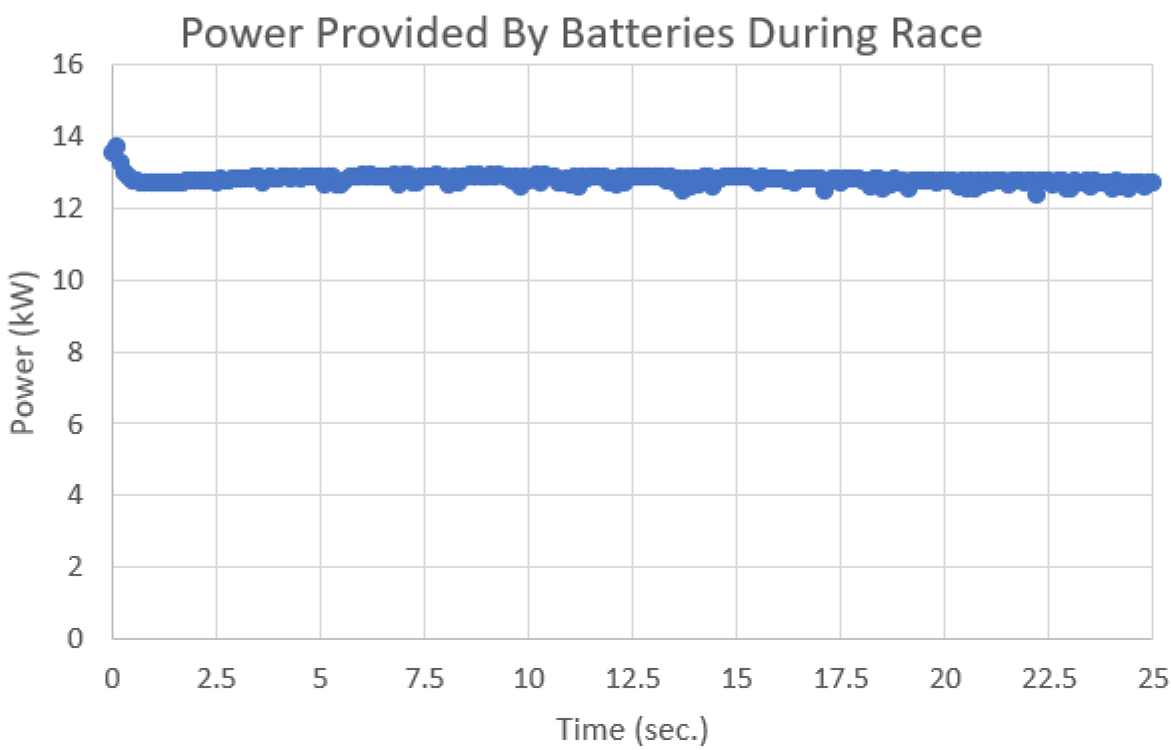


Fig. H.1: *Power put into the system by the batteries over the course of the race.*

$$BatteryPower\ (Current * Voltage) * MotorEfficency = MotorPower\ (RotationalSpeed * Torque) \qquad (H.1)$$

$$MotorPower\ (Speed * Torque) * PropellerEfficency = OutPutPower\ (BoatSpeed * Thrust) \quad (H.2)$$

With the 2022 data, we knew the following variables: *Battery Current*, *Battery Voltage, Rotational Speed,* and *Boat Speed.* We estimated the *Motor Efficiency* based on discussion with our advisor and guessed the *Propeller Efficiency* based on my experience with OpenProp. The two unknowns we were trying to solve were *Torque* and *Thrust*. This gave two equations and two unknowns, by which we could determine the thrust; we plotted it in ***Fig. H.2*** (**Equations H.3-H.4**)

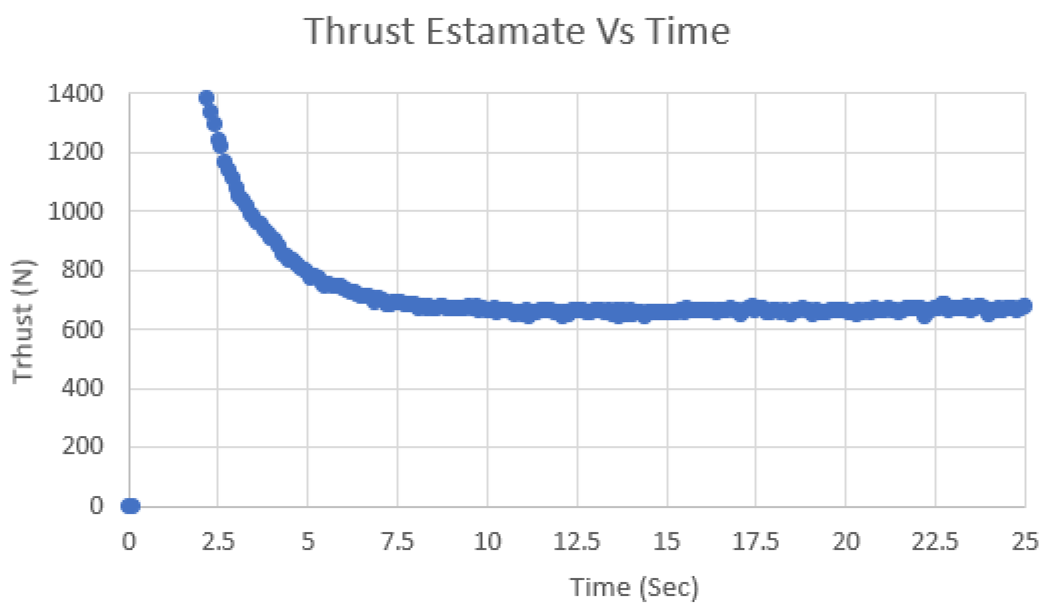


Fig. H.2: *Thrust estimate during sprint race.*

$$(Current * Voltage) * MotorEfficency = RotationalSpeed * Torque \quad (H.3)$$

$$(Speed * Torque) * PropellerEfficency = BoatSpeed * Thrust \quad (H.4)$$

Finding Drag from Thrust:

The next step was to determine the drag. Thrust equals drag at steady state; but during acceleration, thrust includes a drag component.

$$Thrust = Drag + Accelleration\ Force \quad (H.5)$$

We could determine the acceleration using the time derivative of velocity (***Fig. H.3***).

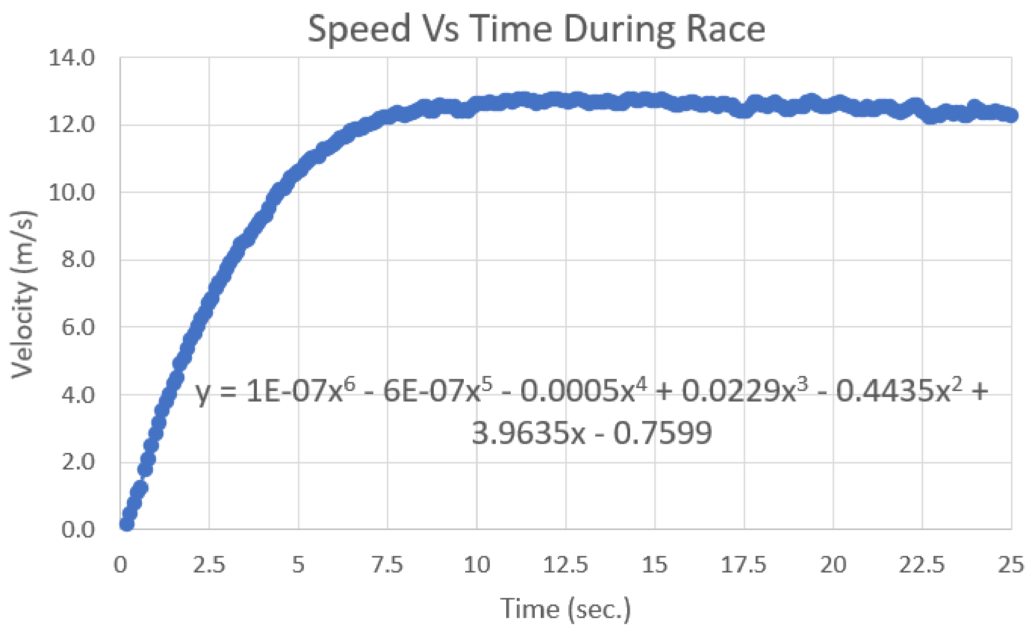


Figure H.3: *Boat velocity over the course of the race with best fit equation. We used the best fit equation to determine acceleration.*

To find the acceleration force, we simply multiplied acceleration by the mass of the boat.

$$Accelleration = Time\ Derivative\ of\ Velocity \tag{H.6}$$

$$Accelleration\ Force = Boat\ Mass * Accelleration \tag{H.7}$$

Finally, we subtracted the acceleration force from the thrust to determine the drag.

$$Drag = Thrust - Accelleration\ Force \tag{H.8}$$

Determining Drag at Target Speed

Drag varies with speed, and we do not have data for the boat when it is travelling at our target speed of 15.6 m/s (35 MPH). To approximate these values, we plotted drag versus boat speed and extrapolated the trend line.

The overall trend line was shaped like a "J", where the low point represented where the boat starts to plane in the water. For my extrapolation, we isolated the planing data and used the trend line feature of Excel to continue the curve up to 15.6 m/s seen in ***Fig. H.4***.

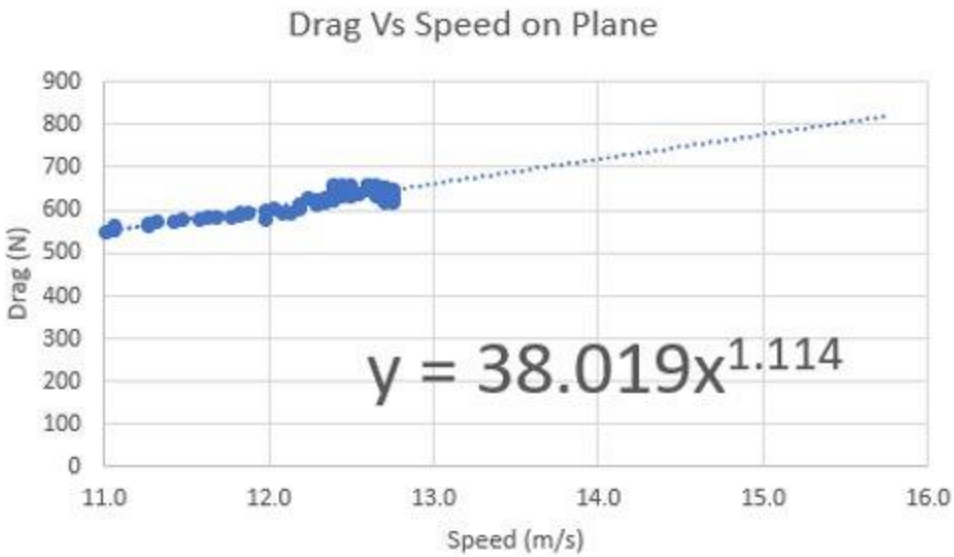


Figure H.4: *Extrapolation of drag up to 15.6 m/s.*

From this extrapolation, we determined that the best guess for boat drag at 15.6 m/s is 810 N (182 LB). This value is similar to the previously projected drag of 800 N. There is uncertainty in this analysis and confirmation came in the form of actual testing later when the thrust load cells were repaired.

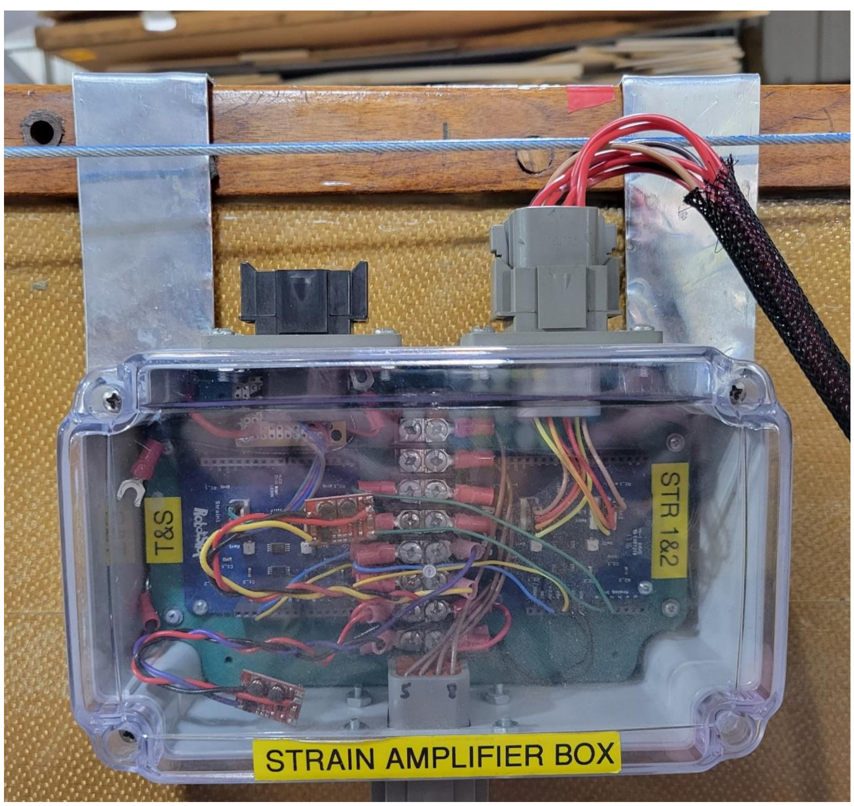


Fig H.5. *SS strain amplifier box*

Appendix I. Data Acquisition and Display System

I.1 Using the AEMDashDesign Software to Setup the AEM Display

A large reason why we selected the AEM display was because the AEMDashDesign software is very user friendly. In the software under the Help tab helpful information is provided for all of its functions as shown in **Fig I.1**.

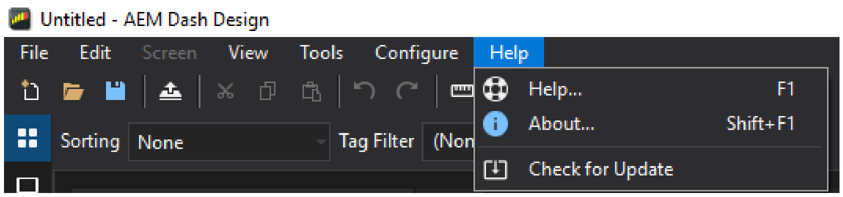


Fig I.1. *Location of the Help tab in AEM Dash Design.*

AEM lists the order one should take making a program for the display:

1. CAN messages must exist before they can be displayed on screens. See the CAN Tab section for more information.
2. Basic setup parameters are defined using the Setup Tab. These include things like shift lights and screen brightness.
3. The Images Tab is used to import all graphic assets for use in your setup.
4. Use the Channels Tab to define custom channel names for CAN messages or to create your own custom channels from scratch.
5. Logic channels must be defined before they can be used as inputs to display items. See the Logic Channels Tab section for more detail.
6. Once all graphics, channels, and logic channels are defined, edit existing screens or design your own using the Design Tab.
7. If your dash hardware is equipped with internal logging, configure your logging preferences using the Logger Tab.
8. Finally, use the Simulate Tab to test your setup. Simulate channel data and watch your screens come to life right on your PC screen.

9. Update the firmware to the latest version. See Updating the Firmware Version in the System User Guide for more detail. If the latest firmware is not installed, attempts to upload the setup file may cause errors.
10. Save your setup and upload it to your dash. See the Uploading a Setup File section.

The current screen designs are located in the Github Repository at GitHub\SolarBoat\AEM.

I.2 Using the AEMdata Software to View Logs

1) Open the AEMdata software
2) Connect to the display with a mini-USB cable while it is powered up
3) Go to Logger>Download Logs
4) Open the log file
5) If it does not open into the template view automatically, go to File>Load Template and navigate to our template which is in the Github Repository at GitHub\SolarBoat\AEM

Appendix J. Boat Operating System

J.1 Boat Operating System Design Explanation

The initial design of the BOS was meant to replicate the function of the BOS code Jonathan Stanhope made to run on the CAN adapter, the main functions from that code are shown below:

```
void runBoatOS() {
      //only change modes when throttle is approximately zero.
    mode = OP_MODE;
    if ((throttle_select == DASH && THROTTLE <= DASH_DEADZONE_LO ||
throttle_select == HAND && THROTTLE <= HAND_DEADZONE_LO)) {
        //only change directions when throttle is zero
        dir_mode = DIRECTION;
        //only change run mode
        run_mode = MOT_ENABLE;
    }
    if(FULL_STOP) {
        e_stop = true;
    }
    if(!MOT_ENABLE) {
        //turn off run mode
        run_mode = false;

        //reset e-stop only when motor is diabled
        e_stop = false;

        //change event type only when motor is disabled
        event = EVENT_TYPE;
    }
    if (OP_MODE == CAN_MODE) {
        sendMotorCommand();
    }
}


void sendMotorCommand() {
    int speed;
    //if in e_stop or not in run mode, send a command of zero.
    if(e_stop || !run_mode) {
        speed = 0;
    }
    else {
        long int max_speed;
```

```
        if(event == ENDURANCE) {
            max_speed = MAX_SPEED_ENDURANCE;
        }
        else {
            max_speed = MAX_SPEED_SPRINT;
        }
        if(throttle_select == DASH) {
          speed = calculateMotorCommand(THROTTLE, max_speed,
DASH_DEADZONE_HI, DASH_DEADZONE_LO);
        }
        else {
          speed = calculateMotorCommand(THROTTLE, max_speed,
HAND_DEADZONE_HI, HAND_DEADZONE_LO);
        }
    }

    Serial.println(speed);
    CANMessage MotorCommand;
    MotorCommand.id = 518;
    MotorCommand.len = 8;
    MotorCommand.data16[0] = speed;
    can.tryToSend(MotorCommand);

}

int calculateMotorCommand(int throttle, long int max_speed, int deadzone_hi,
int deadzone_lo) {
  if(throttle < deadzone_lo) {
    return 0;
  }
  if(throttle > deadzone_hi) {
    return max_speed;
  }
  return (int) (max_speed*(throttle - deadzone_lo)/(deadzone_hi -
deadzone_lo));
}
```

This code simply calculates a motor speed command based on which throttle was being used, the direction, and whether the emergency stop was pressed. The BOS model was built from the BoatWorks Simulink model made by Jonathan Stanhope, at its highest level the BOS has a similar layout but now includes the BOS subsystem and another CAN transmit block as shown in **Fig J.1**.

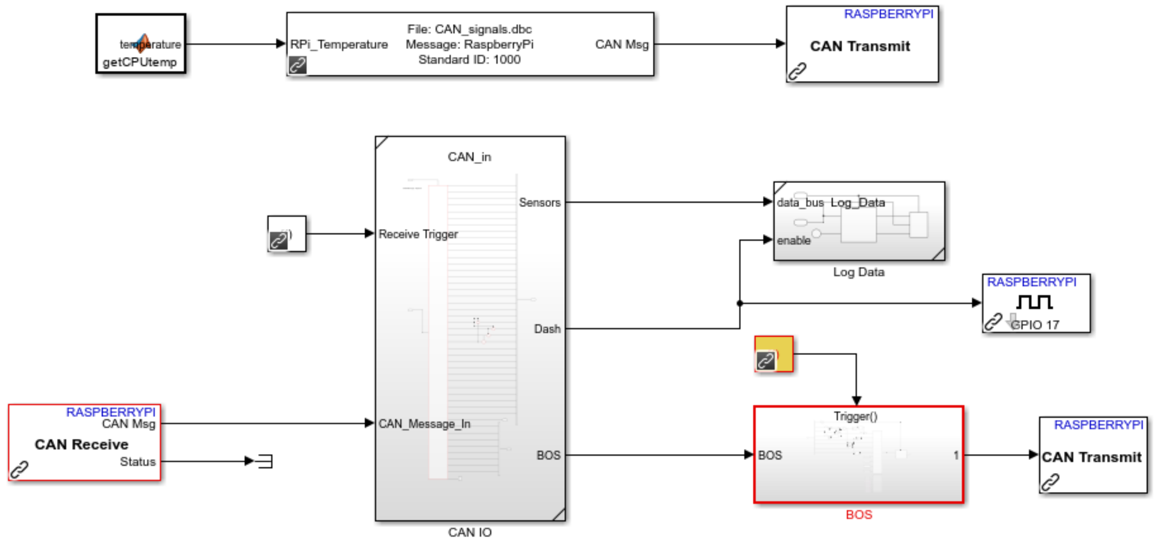


Fig J.1 *Highest level view of the BOS Simulink model.*

This was used mainly because it already had the CAN_in subsystem and logging set up. This subsystem is made up of many CAN unpack blocks which reference the .dbc files to interpret the CAN messages that are input to them from the CAN Receive block shown at the highest level. They then unpack the data according to the .dbc file and name the output signals so that we know what CAN signal they are related to. The CAN unpack block for the SS_Dashboard message is shown in **Fig J.2.** Notice that it is referencing the file CAN_signals.dbc which tells it what the message is named and what the ID is. It also tells the block what each CAN signal within the message is called and all the details of each signal which are defined in the .dbc. Then the data payload of each signal is output on the right of the block and sent out of the subsystem.

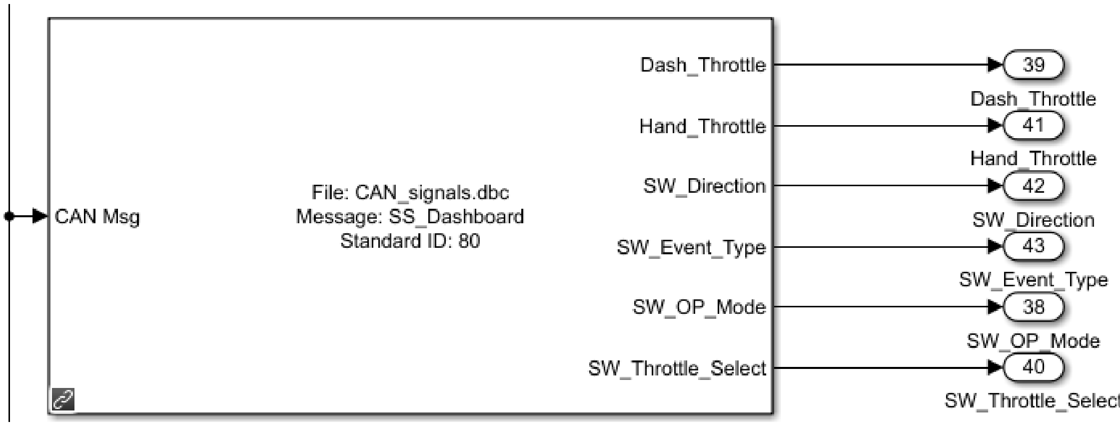


Fig J.2. *CAN unpack block for the SS_Dashboard CAN message which outputs the throttle signals and some of the switch signals received from the dash CAN adapter so that they can be used by the BOS.*

The data from each signal is then sent out of the CAN_in subsystem to the BOS subsystem using a Simulink bus, which is a way to send many connections within one signal wire in Simulink. Currently as shown in **Fig J.1**, the CAN_in subsystem outputs three different buses: Sensors, Dash, and BOS. It would be simpler if it just output all the signals on one bus and we split it to go to the subsystems that use the signals. This way signals could be used by the BOS and be logged at the same time, but it was left this way because currently we don't have a need to log the BOS signals. Notice also that there is a function call generator which is used to trigger the BOS. This function call generate should is set to a sample time of 0.01. This is done to limit the rate at which the BOS message is sent by the model, otherwise it would send too frequently, and the bus would get congested. In the future this could also be improved since we only need to send a motor command speed message when the value changes. This could be done by using a trigger on the CAN pack within the BOS, that somehow is triggered by a rising or falling edge on the motor speed command.

The BOS subsystem is as shown in **Fig J.3.** The main parts of the BOS are the throttle calculation functions, the logic, the command speed calculation function, and the CAN pack.

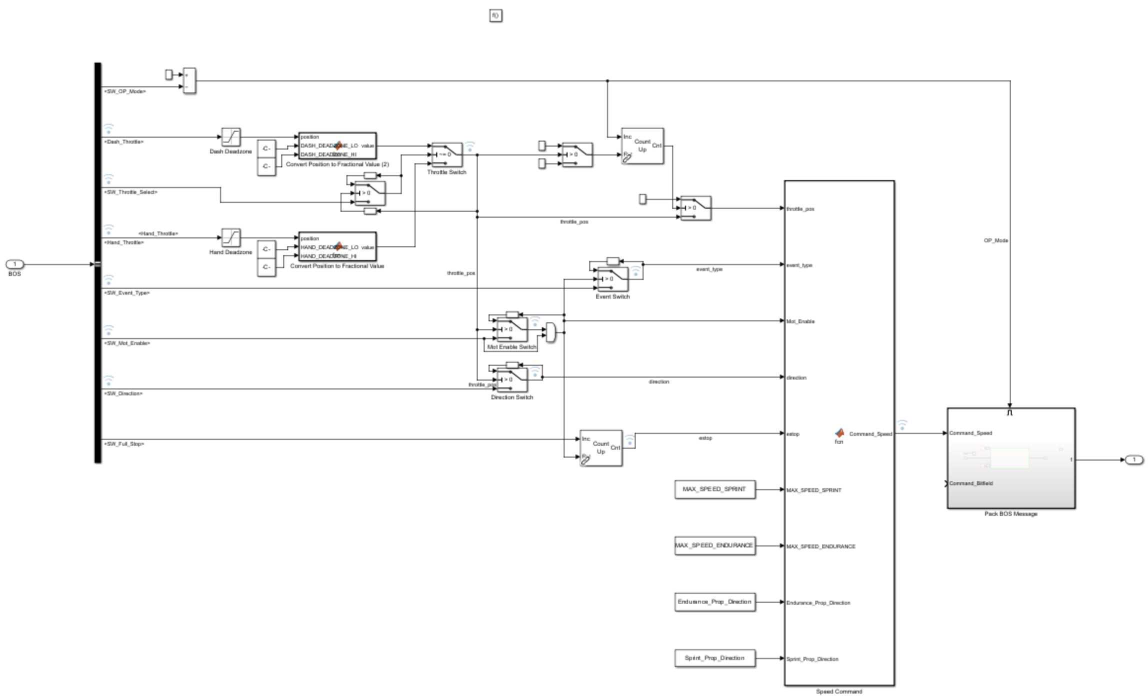


Fig J.3. *The BOS subsystem with (left-right) the throttle calculation functions, logic, the command speed calculation function, and the CAN pack furthest to the right.*

The throttle calculation portion is shown in **Fig J.4**. It takes an input throttle position which comes in as a number, for the dash throttle that is between 0 and 1023. Then the dead-zone values are passed in from the Parameters.m file. The Parameters.m file was created to have one place more setting the values of all the parameters used by the BOS. The code is as follows, notice that the prop values are set to 1 because Dr. Dewhurst did not like the idea of having the motor direction linked to the event type. However, we left it just in case but with both values set to 1 so they don't affect anything.

```
% -------------------------------------------------------------------
%  This file contains the parameters referenced by the BOS.
%  To change their value edit and hit reinitialize from source in model:
%     Modeling > Model Explorer > BOS_RaspberryPi > Model Workspace
% -------------------------------------------------------------------
%  Generated by MATLAB on 24-Mar-2023 16:04:23
%  MATLAB version: 9.12.0.2039608 (R2022a) Update 5
% -------------------------------------------------------------------

DASH_DEADZONE_HI = 1023;

DASH_DEADZONE_LO = 0;

HAND_DEADZONE_HI = 870;

HAND_DEADZONE_LO = 180;
```

```
% Note that the max speed values not only limit the maximum motor speed,
% but also change the scaling of the entire throttle range.

MAX_SPEED_ENDURANCE = 1200;

MAX_SPEED_SPRINT = 4400;

Acceleration_Ramp = 10;

Deceleration_Ramp = 2;

% Input what direction the motor must rotate for the prop to produce
% a forward thrust. So if a positive motor rotation causes a forward thrust
% input 1, and negative motor rotation input -1.

Sprint_Prop_Direction = 1;

Endurance_Prop_Direction = 1;
```

The dead-zone values are determined by setting the throttle to where you would like 0% and 100% to end or start respectively and then reading the value output by the CAN adapter. The dash throttle has a built-in dead-zone near 0% and 100% so we set the values to 0 and 1023 which were read from the CAN adapter in each position respectively. It is important to have dead-zones so that it is easier for the skipper to know they are for certain at 0% or 100% throttle. Then the function takes the difference between the current throttle position value and the lower dead-zone, and divides it by the total range which it gets from the subtracting the upper dead-zone value from the lower dead-zone value as shown:

```
function value = fcn(position,DASH_DEADZONE_LO,DASH_DEADZONE_HI)
    value = (position - DASH_DEADZONE_LO)/(DASH_DEADZONE_HI - DASH_DEADZONE_LO);
end
```

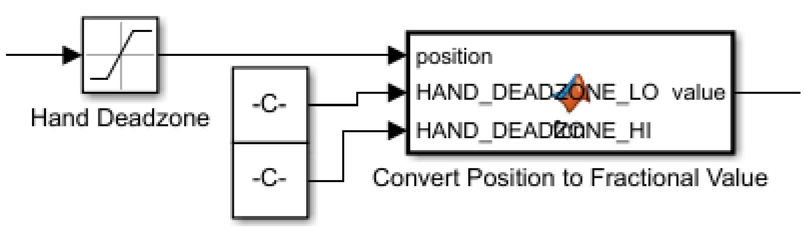


Fig J.4. *Hand throttle calculation function.*

The output of this calculation is a number from value ranging from 0 to 1, which passes through the logic and then into the command speed calculation. The purpose of the logic is to enforce safeties and features defined by the Boat Operating Concept which is included in

Appendix J.2. Within the logic we often use Simulink switches to loop an output until a condition is satisfied to allow it to switch like shown in **Fig J.5.**

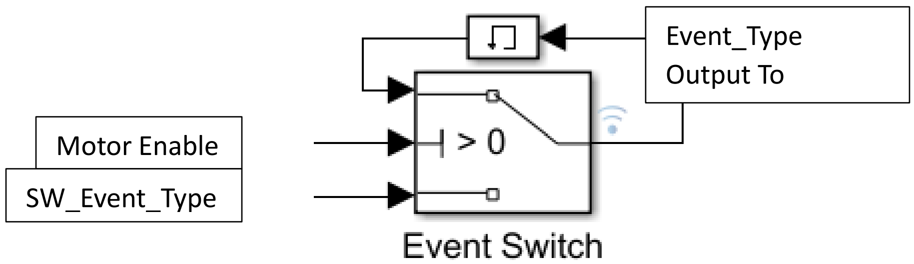


Fig J.5. *Conditional output loop used in BOS logic.*

This switch makes sure that the event type cannot be changed while the motor is enabled. This works because the Simulink switch block passes through input one only when the second input satisfies the condition. In this case that means while the motor enable is high then the switch passes through whatever the last output was using the memory block with the initial condition set to zero. When the motor enable is low then the condition is satisfied so the event type sent to the command speed calculation is set to whatever the value of the switch is. Many of the switches work in this manner, usually the conditional input is either the motor enable or the active throttle position.

This brings us to the command speed calculation function. The code for this function is shown here:

```
function Command_Speed = fcn(throttle_pos, event_type, Mot_Enable, direction,
estop, MAX_SPEED_SPRINT, MAX_SPEED_ENDURANCE, Endurance_Prop_Direction,
Sprint_Prop_Direction)
    % Set the max speed based on the event type
    if event_type == 1
        Max_Speed = MAX_SPEED_SPRINT;
        propfactor = Sprint_Prop_Direction;
    else
        Max_Speed = MAX_SPEED_ENDURANCE;
        propfactor = Endurance_Prop_Direction;
    end

    % A direction of 0 corresponds to REV so multiply by -0.1 which also
    % limits our speed to 20% when in reverse.
    if direction == 0
        direction = -1;
    else
        direction = 1;
```

```
    end

    % The estop button reverses the direction of the motor to bring the
    % boat to a stop and on the second press stops the boat.
    if estop == 1
        estopfactor = -1;
    elseif estop == 2
        estopfactor = 0;
    else
        estopfactor = 1;
    end

    % Motor enable switch
    if Mot_Enable == 0
        enablefactor = 0;
    else
        enablefactor = 1;
    end


    % Calculate the speed command by multiplying the fractional position of
    % the throttle by the maximum speed and direction.

    Command_Speed =
Max_Speed*direction*throttle_pos*estopfactor*enablefactor*propfactor;

end
```

This simply takes the processed inputs from the switches and multiplies them together to get command speed value. This way we get a linear scaling of the throttle position from zero to the maximum speed for the current event type. This means that regardless of the event type the skipper always has the whole throttle range available which is especially important for dialing in an appropriate speed for the endurance event.

After the command speed is calculated, it is sent to the Pack BOS Message subsystem. In this system the BOS CAN message is packed. Currently the BOS message consists only of the command speed signal. The acceleration and deceleration ramp values can also be sent as additional signals, but we still need to find out where in the message the motor controller is expecting those values. This subsystem is enabled by the operating mode switch. This way when we are not in the CAN mode, we are not sending the BOS CAN message. It may appear strange to have the CAN transmit block for the BOS message outside of the BOS, but this is done intentionally. This is done because the Simulink blocks from the Support Package for Raspberry Pi behave inconsistently when placed inside subsystems. They work as expected when the model is initiated by a connected PC, but when the RPi starts the model on its own the blocks do not work.

J.2 Boat Operating Concept

Startup Sequence

This is the order we recommend for starting up the boat to ensure safe operation, but the boat will still work in whatever order as long as the items listed under "required" are done in order.

Recommended

1. Turn on battery box power.
2. Switch on auxiliary power.
3. Turn on cooling pump and confirm flow to motor.
4. Confirm deadman switch is up.
5. Select operating mode (CAN/Backup).
6. Set throttle position to zero.
7. Set motor switch on.
8. Select direction.

Required

1. Turn on battery box power
2. Switch on auxiliary power
3. Set throttle position to zero
4. Set motor switch to on

Shutdown Sequence

1. Set motor switch to off position
2. Hit the shutdown switch for the Raspberry Pi and wait for red LED to turn off
3. Turn off auxiliary power
4. Turn off batter box power

Dashboard Switch Functions

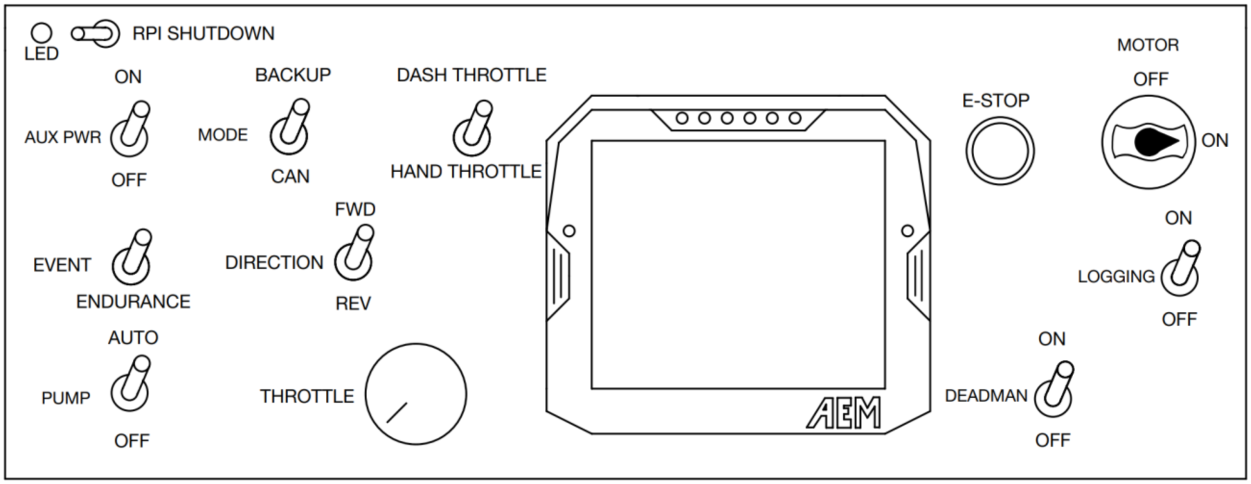


Figure J.6. *Solar Splash dashboard controls layout.*

1. Aux Power

   On: Connects +36V to the DC converts, 12V and 5V, the digital input switches, and the ACS controller. It also enables all the dash control meters and pre-charges the ACS.

   Off: Disables all the dash controls and the ACS motor controller.

2. Operating Mode: CAN/Backup

   ACS:

   On: Immediately changes the ACS controller to be in backup mode. It will read and operate according to the analog signals from the dash.

   Off: It will immediately change the ACS controller to be in CAN mode. In this mode it ignores the analog signals from the dash and operates according to the commands from the CAN bus.

   BOS:

   On: A subsystem within the BOS is enabled which sends the CAN messages that control the motor. When switched from off to on the motor speed command is set to zero until the throttle returns to zero.

   Off: The BOS still runs on the Raspberry Pi, but it does not send the CAN messages to the motor controller to keep bus traffic low.

3. Throttle Select: Dash/Hand

   Backup Mode:

   In backup mode this switch should always be up to select the dash throttle since the hand throttle does not work when in backup mode. We can only send one analog throttle value to the ACS, and the hand throttle and dash throttle cannot be on the same circuit, so we chose to only send the dash throttle.

CAN Mode:

This switch changes which throttle signal is being sent as the motor command. When up it sends the position of the dash throttle and the hand throttle when down. Any change in this switch is ignored until the active throttle position returns to zero, then the active throttle will change according to the position of the switch.

4. <u>Event: Sprint/Endurance</u>

   Backup Mode:

   In backup mode we are not sure of how the ACS is programmed to operate according to this switch. However, the event mode switch in the battery box does change the maximum speed of the motor and it may change some other things such as the current limit of the motor and the ramp acceleration.

   CAN Mode:

   This switch changes the maximum speed of the motor according to the parameter values in the Parameters.m file. In the up position the maximum speed is set to the value of the MAX_SPEED_SPRINT variable, and in the down position it is set to the value of the MAX_SPEED_ENDURANCE variable.

5. <u>Direction: FWD/REV</u>

   Backup Mode:

   This switch changes the direction of the motor but ignores changes until throttle returns to zero.

   CAN Mode:

   This switch changes the direction of the motor but ignores changes until throttle returns to zero. We also have the possibility to limit speed in reverse, but currently our sprint prop requires reverse to go forward so we do not limit it.

6. <u>Cooling Pump: Auto/Off/Manual</u>

   This switch is currently disconnected.

7. <u>Dash Throttle</u>

   Backup Mode:

   This sends an analog value from about 0 – 5V to the motor controller. It must return to zero before changing the direction or motor switch.

CAN Mode:

The position of the potentiometer is scaled based on the current maximum speed and sent to the motor as a speed command. A number of switches require this to be set to zero in order for changes to take effect including the throttle select, direction, and motor switch.

8. AEM CD-7 Display

   This displays information from the CAN bus. The button on the left cycles to the next screen and the button on the right is a min/max value reset button, which we probably won’t use but it also resets the lap times.

9. E-Stop

   Backup Mode:

   We are unsure of what exactly this was programmed to do, but it should reverse the motor direction on the first press and on the second press stop the motor.

   CAN Mode:

   On the first press, the direction of the motor is commanded opposite at the speed the throttle is at to bring the boat to a stop. On the second press the motor speed command is set to zero until it is reset by turning the motor switch off and on again.

10. Motor Switch: On/Off

    Backup Mode:

    The position of this switch is always read by the ACS motor controller, and when it is on the controller provides power from the open drain outputs on the controller which are connected to the contactor to close it. So, in the on position the motor contactor is closed, and in the off position the contactor is opened.

    CAN Mode:

    In the off position the speed command to the motor is always zero. It can be turned on only when the throttle is at zero but can be turned off immediately regardless of throttle input. When it is on the BOS will send a speed command to the motor based on the active throttle value. This switch must be in the off position for the event type to change. The off position also resets the emergency stop.

11. Raspberry Pi Shutdown Switch

    This momentary switch safely shuts down the Raspberry Pi so that log files are not lost.

12. Deadman: Active/Kill

    This switch directly opens or closes the motor contactor and is not processed in software. It must be on for the motor contactor to close. If it is switched off the contactors will open.

J.3 New Dash CAN Adapter Code

CU_CAN_DASH is the Arduino code which is used by the first CAN adapter in the dash. The second CAN adapter in the dash uses the CU_CAN_DASH2 code, and its function is only related to the thrust and torque measurement. The new program for the CU_CAN_DASH was derived from the previous version but stripped down. Now the code simply reads the analog and digital input values which are connected to the dash switches and potentiometers, and the values are assigned to variables:

```
    //Read the input data
    DASH_THROTTLE       = analogRead(A0); // 10 bits
    SW_OP_MODE          = digitalRead(A1);
    SW_EVENT_TYPE       = 1-digitalRead(A2);
    SW_DIRECTION        = 1-digitalRead(A3);
    SW_MOT_ENABLE       = digitalRead(A4);
    SW_FULL_STOP        = digitalRead(A5);
    SW_PUMP_AUTO        = analogRead(A6);
    HAND_THROTTLE       = analogRead(A7);
    SW_LOG_ENABLE       = 1-digitalRead(4); //The log switch is low when in
the on position, so invert the signal.
    SW_THROTTLE_SELECT  = digitalRead(7);
```

Then the CAN message ID and length are defined, and the variables are assigned to certain parts of the messages:

```
    // Initialize CAN Messages Sent
    CANMessage DASH_Message;
        DASH_Message.id = 80;
        DASH_Message.len = 8;
        DASH_Message.data16   [0] = DASH_THROTTLE;
        DASH_Message.data16   [2] = HAND_THROTTLE;
        DASH_Message.data     [4] = SW_THROTTLE_SELECT;
        DASH_Message.data     [5] = SW_OP_MODE;
        DASH_Message.data     [6] = SW_EVENT_TYPE;
        DASH_Message.data     [7] = SW_DIRECTION;

    CANMessage DASH_Message2;
        DASH_Message2.id = 81;
        DASH_Message2.len = 4;
        DASH_Message2.data     [0] = SW_MOT_ENABLE;
        DASH_Message2.data     [1] = SW_FULL_STOP;
        DASH_Message2.data     [2] = SW_PUMP_AUTO_dig;
        DASH_Message2.data     [3] = SW_LOG_ENABLE;
```

The messages and variables starting with SW_ were named so to denote that they are directly related to switch positions only and not necessarily what mode the boat is actually in. For example, if the direction switch were switched to reverse while driving forward with the throttle up, the motor would not turn the opposite direction until the throttle returned to zero. This was done so that in the future if the BOS output messages for what mode the boat is operating in, they could be differentiated from the switch positions in the logs.

Since the order and content of these messages were changed, we had to also update our .dbc file to reflect the changes. Previously the team used two separate .dbc files: CAN_signals.dbc and SS1_Signals.dbc. CAN_signals.dbc contains most of the CAN signals that we use such as those from the ACS motor controller, IPEspeed GPS, battery box CAN adapters, the dash CAN adapters and more. SS1_Signals.dbc contains the signals that are relevant to the hydrofoil boat such as the height sensor sensors, IMU data, and actuator commands. So CAN_Signals.dbc was updated to reflect the changes to the dash changes. Later the two .dbc files were merged using the Kvaser Database Editor. Currently the SS1_Signals.dbc file remains as we make the transition to only using the CAN_signals.dbc file within the Simulink models.

J.4 Using the RPi for the BOS

Deploying the BOS Model to Run on the BOS

Whenever changes are made to the BOS or Parameters.m the model must be redeployed to the RPi for them to take effect, this requires taking the following steps:

1) Make sure your target hardware is correct by navigating the Hardware Tab>Hardware Settings to get to the menu shown in **Figure J.7.**

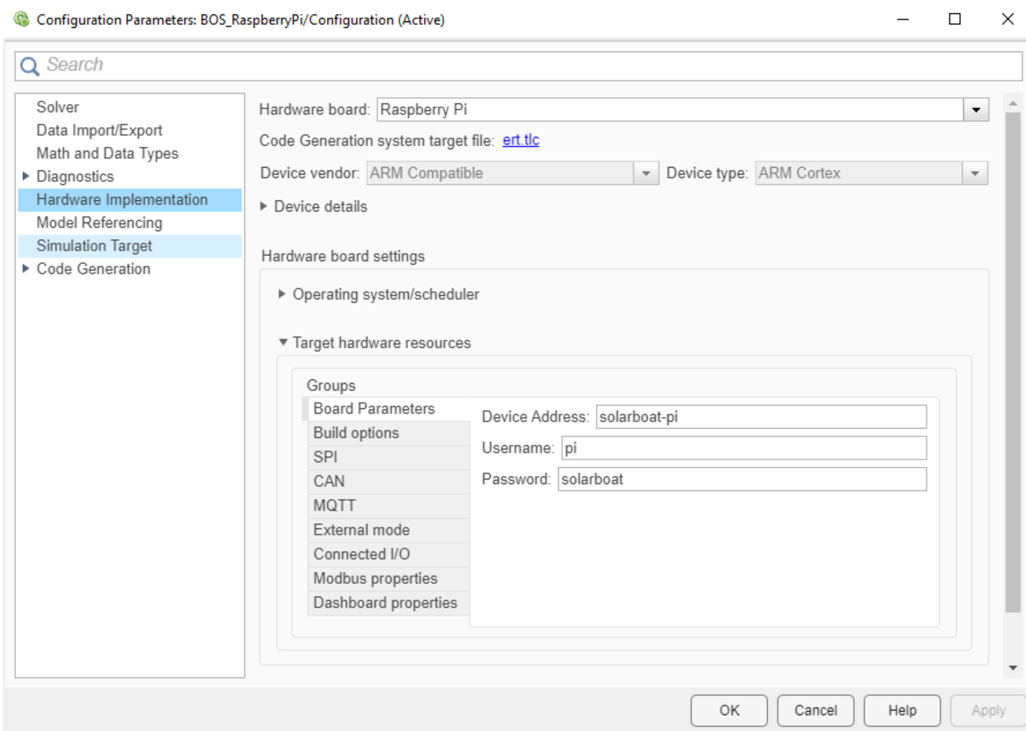


Figure J.7. Changing the target hardware settings within Simulink.

This example is for the dash RPi, if using a different RPi and you don't know the device address you may need to connect to the RPi through ssh and type the command ifconfig to find its ip address under the eth0: listing which you can put in as the device address.

2) Then you can deploy the model to the RPi. If you are testing something and may want to use the Data Inspector feature in Simulink you will need to hit the green play button that says Monitor & Tune. This is very useful for troubleshooting so you can check intermediate values while the code is running on the RPi. If you don't need to do this, we like to hit the Build, Deploy & Start button.

   Note that under the Build options tab shown in **Fig J.7**. you can select whether or not to run the model on boot. This is a nice feature, but when it is used the logging feature does not work on the RPi so we do not use it. Instead we use the crontab task schedule, there is a section on this later in this appendix.

Connecting/Interfacing with the RPi

Note on WinSCP vs. ssh connection.

WinSCP is useful for looking at and manipulating what files are on the RPi, like when you want to transfer log files to your computer. Connecting through ssh is good for executing commands or making changes to the RPi itself such as modifying the config.txt file.

How to connect with ethernet (my preferred method).

1) Ensure RPi has power, either auxiliary boat power or with a micro-USB cable. Pi should startup indicated by the red light on the dash.
2) Connect the RPi to the school network (I think you can also connect it directly to your PC as well) with a ethernet cable in the Senior Design Lab. I would use one from other computers or one I brought in.
3) Open WinSCP on your PC, and follow the login prompt:

   Dash Pi Host Name: pi@solarboat-pi
   Dyno Pi Host Name: pi@solarboatdyno
   Password: solarboat

   Or using ssh (open command prompt on your computer by searching "cmd")

   Use the command "ssh pi@solarboat-pi" then when prompted enter the password "solarboat"

How to connect wireless method (like when it is in the boat).

1) Ensure RPi has power, either auxiliary boat power or with a micro-USB cable. Pi should startup indicated by the red light on the dash.
2) Use a PC to connect to RPi wireless network "CedarvilleSolarBoat" which automatically starts when the Pi has started up. Password is CU_later
3) Open WinSCP on your PC, and follow the login prompt:

   Host Name: 192.168.4.1

   Username: pi

   Password: solarboat

   Or using ssh (open command prompt on your computer by searching "cmd")

   Use the command "ssh 192.168.4.1" then when prompted enter the password "solarboat"

Direct Connection

1) Ensure RPi has power, either auxiliary boat power or with a micro-USB cable. Pi should startup indicated by the red light on the dash.

2) Connect a monitor to the HDMI output of the RPi and a keyboard to connect to one of the USB ports. If the RPi is set up for desktop use you can use a mouse too. Otherwise, this is just like connecting to the pi with ssh which is more convenient IMO.

Info on Using the RPi

Most relevant to our work with the RPi is modifying the config.txt file for setting up modules and other important settings, and the crontab task scheduler for making sure our Simulink models run whenever the RPi starts up. Most of the information to use these features can be found in Jonathan Stanhope's "Guide to Boat Codebase" document, but we will try to put some other helpful information here.

General

Table J.1. *Raspberry Pi commands*

Keystroke/command	Function
CTRL + C	Stop
sudo reboot	Reboots RPi
sudo shutdown now	Shuts down RPi immediately
sudo nano /boot/config.txt	Edit config.txt file
ls	Lists files in current directory
cd	Changes current directory

File Navigation

"ls" lists files in the current directory, "cd" changes the directory, mkdir can make a folder, and ./ runs the executable in a folder. For example at first you are in the home directory so you do ls to list the files, you see MATLAB_ws is a folder which you can enter by using "cd MATLAB_ws", you can also go faster by doing cd followed by the file path like "cd MATLAB_ws/R2022a/C".

Clock module

For a list of commands for the clock module you can use "hwclock -h"

You shouldn't have to reset the clock module time since it is really accurate, after four months I think it is still within a second of real time. If you do have to set it though you can use "sudo date -s 'YYYY-MM-DD HH:MM:SS" to set the system time then use "sudo hwclock -w" to set the clock module to the system time you just set. I think it also updates the time whenever it is connected to the internet as well, so you shouldn't have to worry about it.

CAN Testing

Cangen

It's helpful when trying to send CAN messages to know if the RPi is sending them correctly or if it is a problem with Simulink. You can use the cangen commands to do this which are listed using "cangen -h".

I often use the command "cangen can0 r" to send random messages which I then look for with the Kvasser. Use CTRL + C to stop sending random messages.

ifconfig

If you can't get the RPi to send CAN messages sometimes the CAN channel has not been setup correctly which you can check by doing "ifconfig" which lists the current communication channels. If can0 is listed that means it should be set up correctly.

Config.txt

This file is used to setup modules, switches, adjust settings, and other things.

Lines We Use

```
# Setting for CAN HAT
dtoverlay=mcp2515-
can0,oscillator=12000000,interrupt=25,spimaxfrequency=2000000

# Enable shutdown using switch connected to GPIO3
dtoverlay=gpio-shutdown,debounce=50

# Turn an LED on when power is on
gpio=26=op,dh
```

Crontab

We use crontab to run the Simulink model on startup of the Pi. You can edit this on the Pi by using the command 'crontab -e'. The dash Pi is setup to modify this file with vim which is terrible. I recommend editing with nano instead, but I haven't tried switching it.

```
@reboot sudo
home/pi/MATLAB_ws/R2022a/C/Users/amudd/Documents/GitHub/SolarBoat/BOS/
BOS_RaspberryPi.elf 2>&1 ~/Simulink.log
```

J.5 CAN System Information

BOS Model in Simulink

The dash CAN adapter reads analog and digital values from the dash switches and potentiometers and sends the information in CAN messages to the Raspberry Pi, which is running the Boat Operating System within a Simulink model. The BOS receives these messages and has some processing logic which then sends commands to the motor controller over CAN.

Important note: I have found that the blocks within Simulink from the Raspberry Pi Hardware Support package like the 'CAN Transmit' do not function correctly when they are put inside a subsystem. They may function correctly when running the model through the Run and Monitor mode on a computer connected to the Pi, but when running on the Pi alone they will not function correctly.

CAN Adapters

When uploading code to a CAN Adapter from Arduino IDE you must change the following:

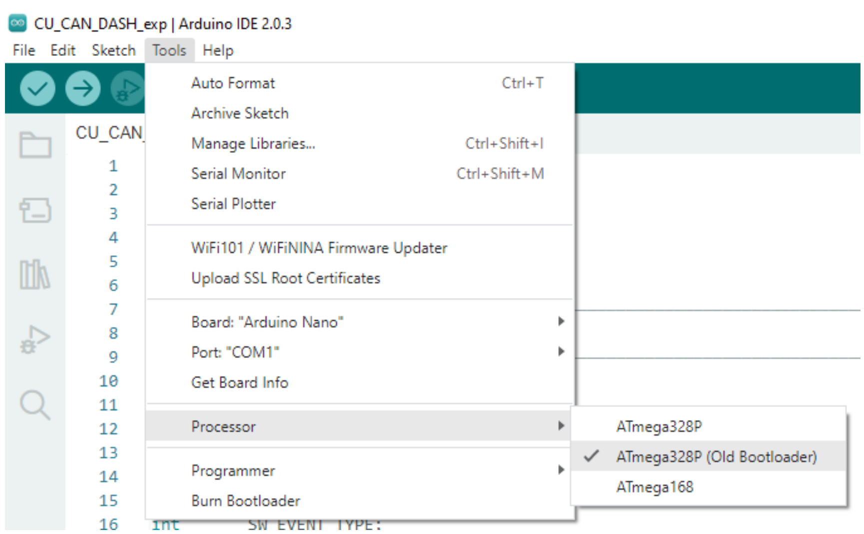


Fig J.8. *Select Old Bootlegger setting in Arduino IDE.*

CAN .dbc Files

The CAN dbc files are the key to interpreting what message is what when using CAN messages. They are used by everything connected to the CAN bus.

Editing the Files:

The format is pretty self-explanatory if you look at them and they CAN be edited manually, but I found the best way to edit the files is by using Kvaser Database Editor. Here is the download link:

https://www.kvaser.com/download/?utm_source=software&utm_ean=7330130981942&utm_status=latest

J.6 GitHub

GitHub is very useful for keeping our codes up to date and tracking our version history. The basic function is that the repository, a folder essentially, is stored on your computer's hard drive and then there is a repository with all the same files on the cloud. Any time you make a change to the files on your computer like adding a line of code, that change will show up on GitHub when you hit the fetch button which causes it to check your files against those in the cloud. Then you add comments describing what you changed, hit the commit button, which commits the change. Then you hit the push button to push the changes to the cloud repository. It also works the other way so that if there is a change to the cloud repository, maybe someone else committed and pushed a change, then when you hit fetch it will show you what changes have been made. Then you can hit the pull button to update the repository on your computer with the changes from the cloud repository.

Setting up GitHub is fairly simple:

1) Go to CedarNet and download GitHub Desktop
2) Make an account with your Cedarville email, or get invited by one of the current team members. Either way you need to be granted access to the Cedarville Solar Boat repository.
3) Then clone the repository to a location on your computer, I recommend the documents folder. (That is usually located at C\name\user\documents on any computer so that when deploying RPi models they are always located in the same directory on the RPi since its directory is built depending on the location of the model

on your computer.) It will then download the repository to your computer, and you will be able to access all of the files.

4) Then you can use the features as described above to commit changes and pull changes from the cloud repository to the repository on your computer.

J.7 Adjustable Parameters

I came up with an idea for making the parameters, that is constants referenced in the Simulink model, adjustable with a push-pull style potentiometer and button. I tested and proved the concept within the dashsimulation.slx Simulink model found in the Github repository at SolarBoat\BOS. The team decided not to implement this feature yet, but future teams may want to do so.

Appendix K. Electrical Schematics

Fig K.1. *Solar Boat Team — Reference Material*

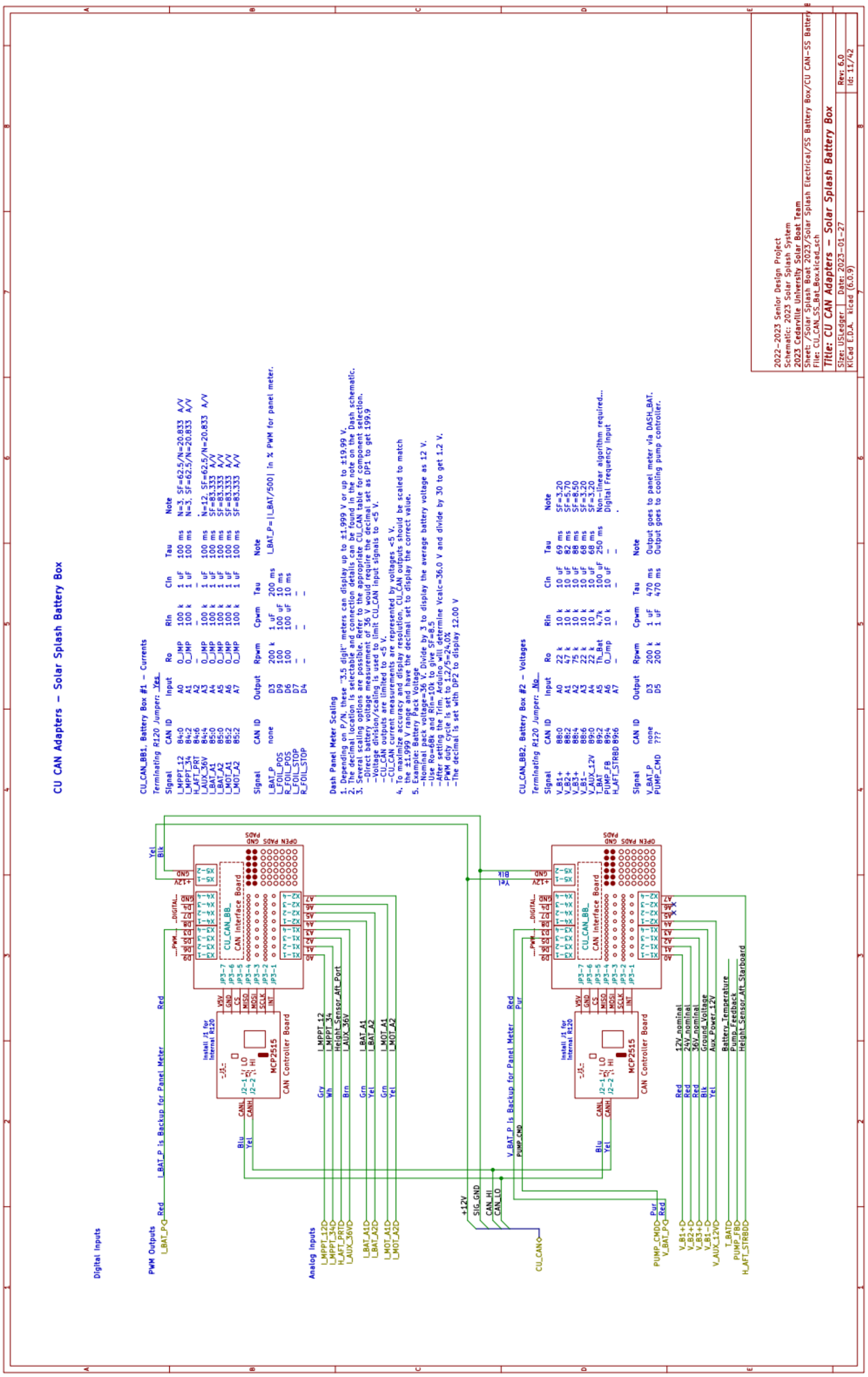


Fig K.2. *CU CAN Adapters – Solar Splash Dash*

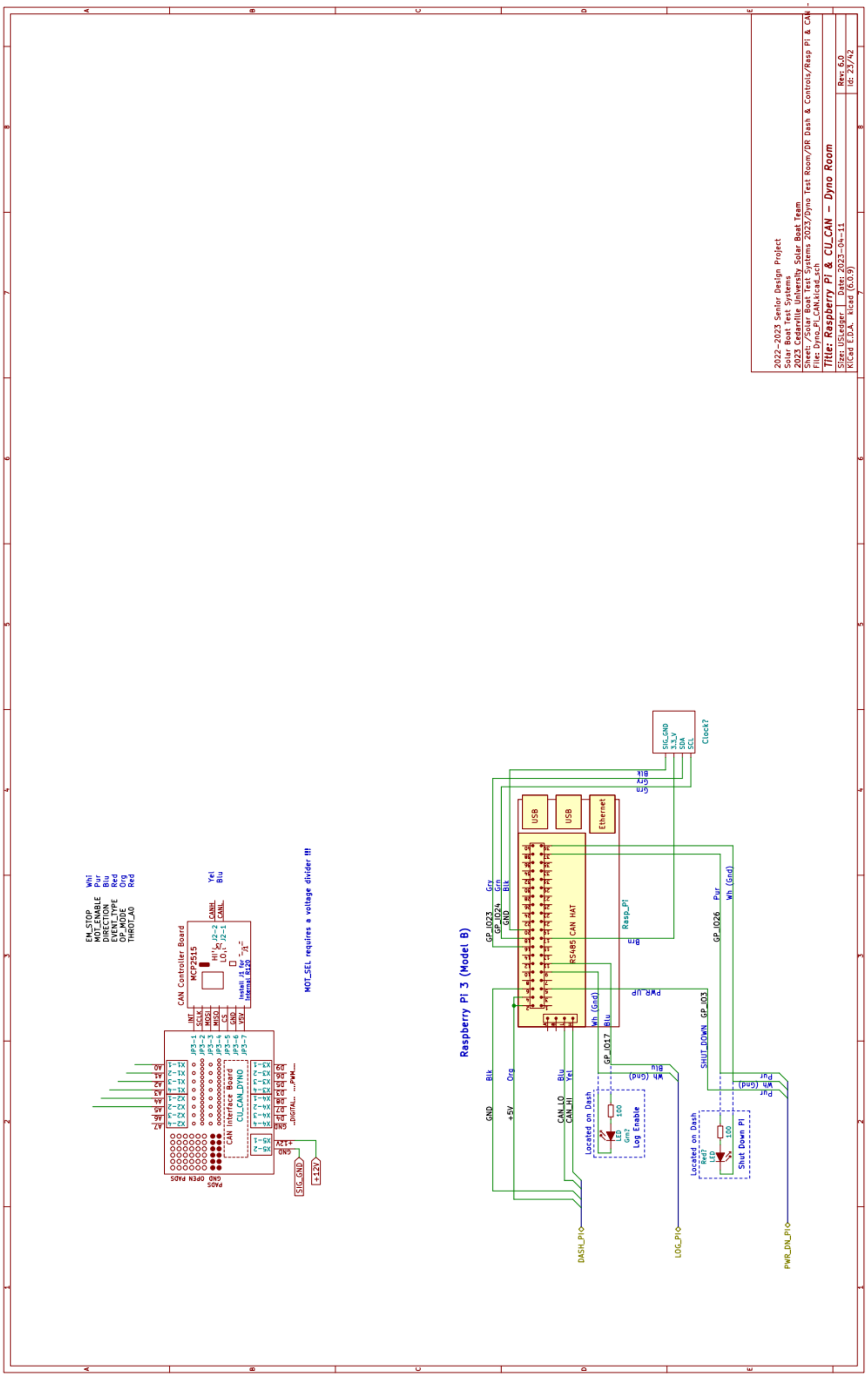


Fig K.3. Raspberry Pi Raspberry Pi & CU_CAN — Dyno Room

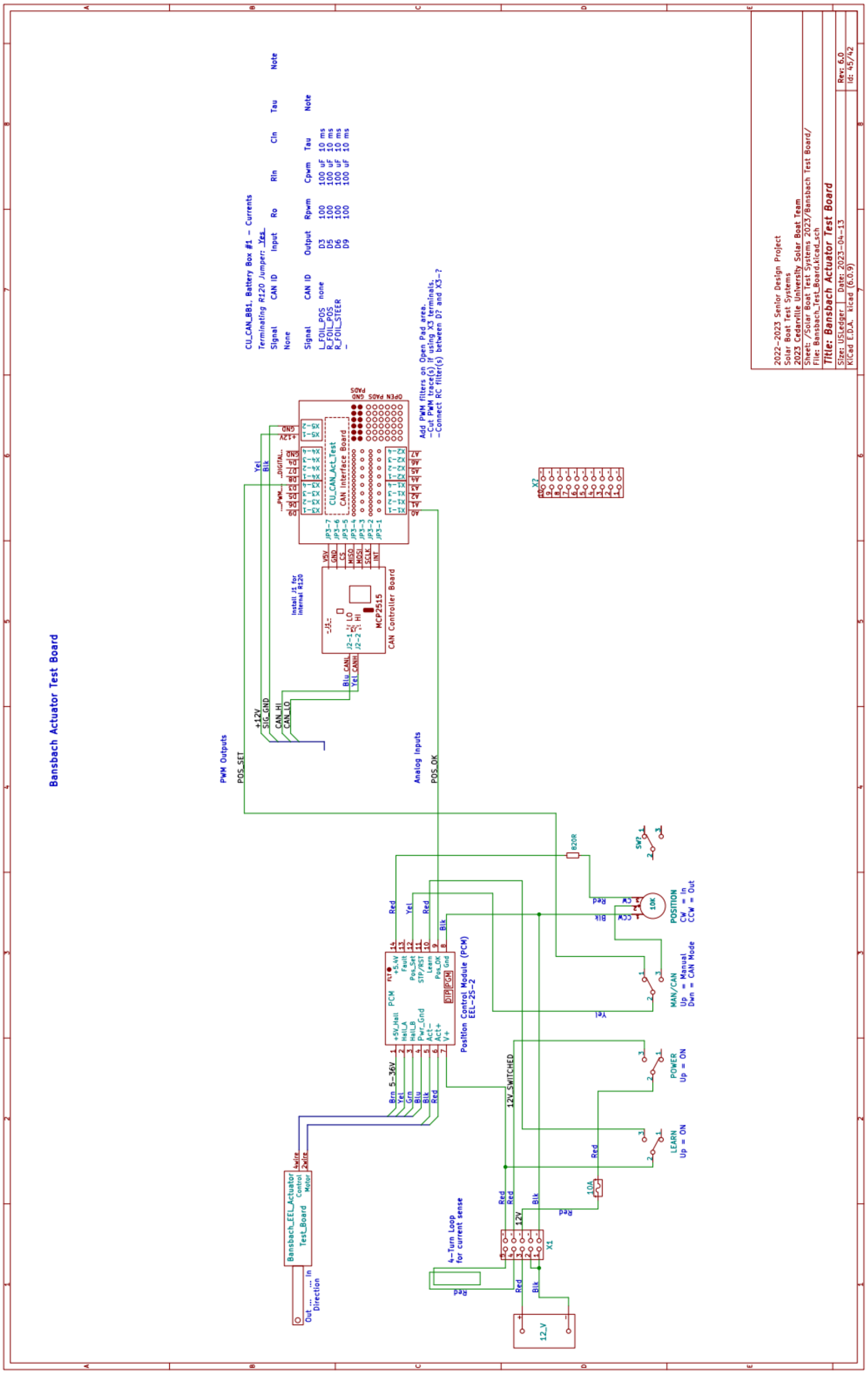


Fig K.4. *Solar Boat Team—Reference Material Bansbach Actuator Test Board*

Appendix L. Contact Information

L.1 Sponsor Contact Information

Contact	Company	Email	Phone
Nicholas Cardaci	Inmotion	nicholas.cardaci@evs-inmotion.com	571.356.8764

L.2 Source Information

Part Name	Part Number	Company Name	Company Address	Company Phone Number	Price
Hawk40 Motor	HO40-5-1-1	DHW Electric Machines	1101 HWY 124, Building 5 Hoschton, GA 30548	678-900-1074	$3,800
AEM CD-7L Display	30-5701	AEM Performance Electronics	2205 W 126th Street, Unit A Hawthorne,CA 90250	310-484-2322	$1,758.98
AEM Vehicle Dynamics Module	30-2206	AEM Performance Electronics	2205 W 126th Street, Unit A Hawthorne,CA 90250	310-484-2322	$396.01