

# Wheelchair Dynamometer Final Report

Jason Sanchez, Javier Salazar, Olateju Ojeyinka, Rolando Robles

*Department of Electrical Engineering, The University of Texas at Arlington, Arlington, TX 76010*

## Abstract

Fitness is an important aspect in today's society as a rapidly growing industry. Many people exercise to live a healthier lifestyle but also for sport performance. Several machines have been invented to exercise the body in different ways and help stay fit although there are very few machines specifically meant for wheelchair users. The wheelchair dynamometer is designed for wheelchair users to exercise, like an able-bodied individual, when needed as well as track performance parameters important in their field of sport. With a resistance setting, the dynamometer can accommodate a wide range of users at different fitness levels that can see workout/performance parameters through interaction with the included display.

<b>Abstract</b>	<b>1</b>
<b>Introduction</b>	<b>2</b>
<b>Product Background</b>	<b>4</b>
<b>Development Plan</b>	<b>5</b>
Variable Resistance	8
Gate Driver	17
Rectifier	19
Single-Board Computer	20
Torque Measurements	29
Filtering	32
Speed Measurements	33
Analog & Digital Signals	37
Power Requirements	38
Overall System Overview	39
PCB Milling & Testing	40
Natural Rolling Feature - Preliminary Work	44
<b>Budget</b>	<b>52</b>
<b>Conclusion</b>	<b>54</b>
<b>Appendix</b>	<b>56</b>
PCB Manufacturing at UTA	56
User Guide	72
<b>References</b>	<b>80</b>

# Introduction

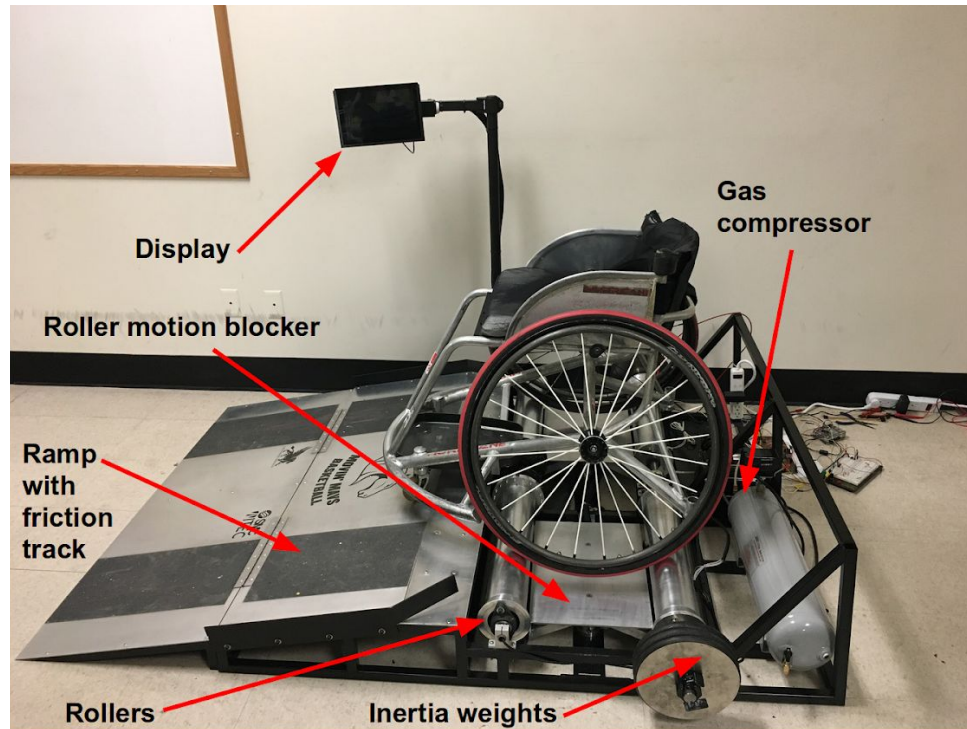


Figure 1. Mechanically complete wheelchair dynamometer.

As shown above, this is the wheelchair dynamometer system with the different components labeled. In the front, we have a ramp with a sandpaper finish section for better wheelchair grip as the user gets on. The roller motion blocker is a metal plate with rubber pads on the bottom. When the switch is activated, the metal plate rises to lock the rollers into position. This is done to make it easier for the user to get in the middle of the two rollers. The gas compressor is included since it is required to raise the metal plates. The display is where the user interacts with the dynamometer system and sees workout information. The inertia weights on the back rollers provide rotational inertia so that the back rollers continue to spin after the user has stopped rolling in order to simulate real-life conditions. Unfortunately, this inertia system does not work well to simulate real-life conditions and the friction issue between the rollers and wheelchair still remains.

This project focuses on implementing the electrical aspect of the wheelchair dynamometer. The main purpose of the wheelchair dynamometer is to measure and keep track of performance parameters of athletic wheelchair users along with general population users. The completed wheelchair dynamometer is required to have an LCD that shows the following: speed, power output, resistance/difficulty level, caloric expenditure, and time. Moreover, the user should be able to control the resistance setting through the display and visualize past workout data. To achieve this, the electronics needs to be controlled by some kind of computer device. However,

based on one of the constraints of this project, the electronics will be controlled by a Raspberry Pi 3, a single board computer. To accomplish a varying difficulty level for the wheelchair dynamometer, we considered a variable resistor which will be implemented with the use of a MOSFET. As mentioned earlier, the device will be capable of measuring speed. The first attempt to do this involved using the in-built hall effect sensor in the motor to get frequency information. However, due to noise issues, the back EMF voltage value from the rectifier is used since this is related to speed. Since we are working with an analog signal, an analog to digital converter will be required at this stage before the information is sent to the single board computer since it only supports digital signals. Furthermore, to implement the dyno functionality of our device, we are required to measure torque. To do this, there is a need for sensing current going through the motor to determine torque information. The signal from the current sense block needs filtering to remove ripple noise from the rectifier and then sent to the ADC before it can be interpreted by the single board computer.

As of now, the motors in the wheelchair dynamometer do not receive power and are operating in what we call the passive mode. Because of the circuit design, there is a connector to allow the possibility of providing power to the motors. One possible, future addition to the dyno system is to power the motors in order to simulate real-life flat path conditions since friction between wheelchair wheels and motors is higher than normal rolling conditions. For example, the rollers cut-off abruptly when the user stops applying force. To counteract this effect, the motors will need to be provided sufficient power whenever an abrupt lower speed is detected by the device. The power should be sufficient enough to slow down the rate at which the rollers stop but not too high to the point where it prevents the roller to stop spinning. This feature will be briefly addressed in the report so that any possible, future team can reference the work already done regarding this topic. The block diagram in Figure 4 will give a more visual representation of the design layout. Each block will be explained further in the report.

## Product Background



Figure 2. Invictus Active Trainer, a similar, commercial wheelchair exercise system. Image Courtesy: Invictus.

Although the main use of the wheelchair dynamometer is to receive data about athletes' performance, it can also be used as a workout machine for wheelchair users. It provides an accessible workout machine for a demographic that lacks options and in doing so expands the target market for this device beyond athletic use. There is presently one company on the market that provides a similar device called the Invictus Active Trainer (IAT) as shown in Figure 2 [0]. This device can show distance, speed, caloric expenditure, and heart rate information through the use of bluetooth-enabled smartphones. It also suffers from the same friction issue between the rollers and wheelchair wheels so it does not correctly simulate real-life conditions. However, our device can provide a wide range of functionalities that this product does not provide. The first and most important functionality is the variable resistance ability. IAT only provides a roller that the wheelchair user can roll their wheels on but there are no adjustments for different difficulty settings. Unlike the wheelchair dynamometer, the IAT does not have an LCD display but does have a mobile app where one can connect to the device via bluetooth. This may be challenging for the elderly market since it requires the ownership and operation of a smartphone device. This



## Development Plan



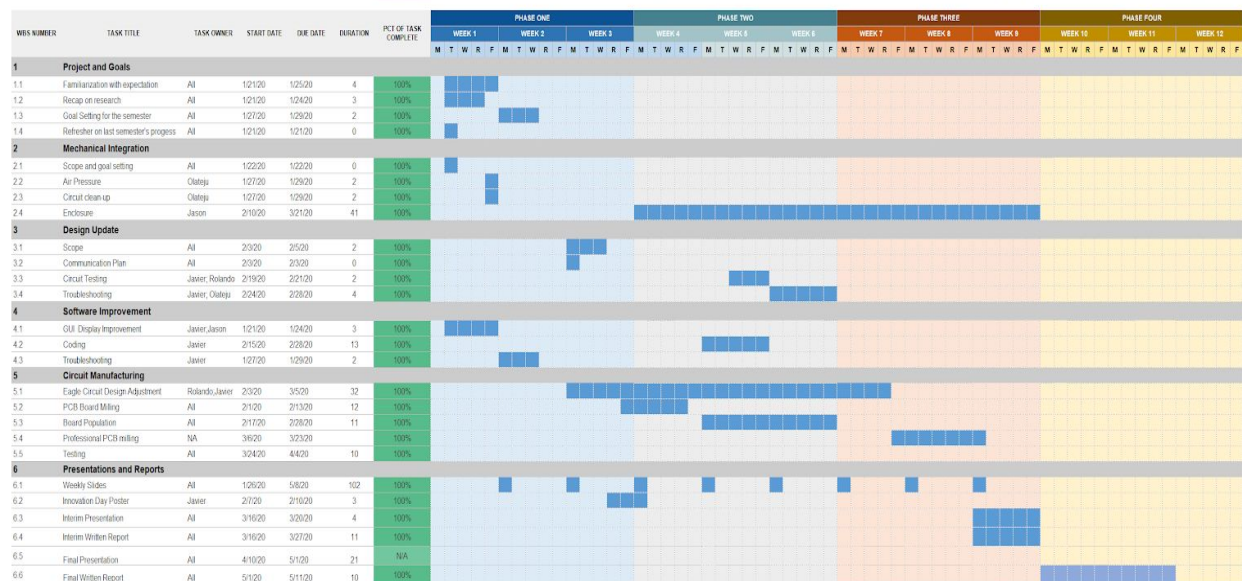


Figure 4. Gantt chart for the spring semester term.

The timeline for this project is eight months or two semesters. The fall semester was spent researching the various implementations of the requirements and coming up with a design that meets the constraints. Furthermore, the parts were ordered, the system was tested on a breadboard, and a board was milled and populated. The software aspect of the project is essentially complete except for some design choices that the sponsors wish to change and minor changes such as adding a button option for changing workout level. For the spring semester, the mechanical system was integrated with the dyno, a new design choice was selected for speed, software improvements were made, and the final PCB designed and assembled. The figure below shows the block diagram of the final design as implemented in the PCB. The sections below will provide a detailed look into each block, other options that were considered to use, why the specific solution was chosen, and what requirements the blocks satisfy. Before going into each individual section, the system block diagram at a top level will be discussed.

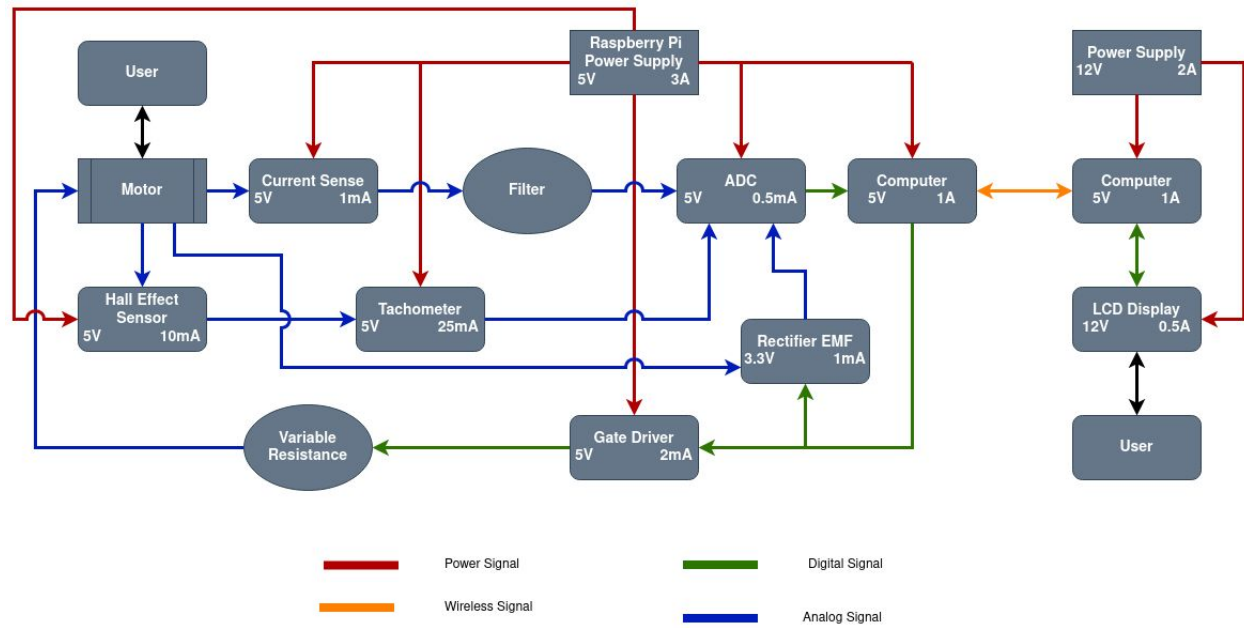


Figure 5. Block diagram for wheelchair dynamometer.

From the figure above, the starting point to observe is the user block on the top left. Here, the user is rolling on the dyno machine and interacting with the motors which is the next adjacent block. The motors are generating current and voltage values that relay important information about the speed and torque of the user operating the machine. In order to get the speed and torque values, a conversion must take place to get the correct values. For speed, the back EMF value of the motor is passed to the ADC using a multi-stage voltage divider. This is done to protect the ADC from high voltage damage as the pin voltage should not exceed supply voltage. The back EMF voltage value is directly proportional to speed (& current as well) so keeping track of both parameters will allow us to calculate speed. For torque, the current sense block is an amplifier with a shunt resistor as an input. The purpose here is to observe and amplify a small differential voltage that is higher when there is more current in the system. This signal is filtered with a low pass filter to reduce noise before being passed to the ADC. The ADC block communicates with the single-board computer via the SPI protocol. The computer that handles computations is linked to the single-board computer inside the display via wifi. This display interacts with the user to set resistance settings and view information. The computer block sends a signal to the gate driver block to set the mechanical resistance setting of the dyno. Finally, the variable resistance block is the actual mechanism and method that is used to change the shunt resistance that the motors experience since the gate driver is only needed to decrease transition times. By changing the electrical resistance, this affects the mechanical resistance experienced by the wheelchair user.

## Variable Resistance

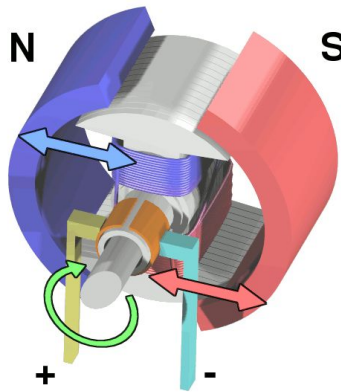


Figure 6. Illustration of simple DC motor model. Image courtesy: Wapcaplet - Wikipedia Commons.

For the variable resistance block, we need to vary the shunt resistance that the motor sees so that the current going from one motor terminal to another is restricted. In effect, this will vary the mechanical resistance required in order to move the motors. From the figure above, we have a DC motor. Note that as a person rotates the motor, the magnet moves past a correctly oriented induction coil that picks up on a changing magnetic field. By Lenz's law that states that electromotive force is proportional to a change in magnetic flux, the direction of the back e.m.f. of an induced field opposes the changing current that is caused by the user [1]. Thus, the current in the induction coil creates a magnetic field that produces a force that counters the direction of present motion and thus requires greater mechanical force to overcome. Clearly, it counters the direction of motion due to conservation of energy. If the electrical force is in the same direction of motion, then we would get more energy than energy supplied by the wheelchair user. This means that the torque required to spin the motor is dependent on the electrical load attached to the motor. For the dynamometer, it is important to understand the appropriate load range that a wheelchair user can operate in so that the variable resistance component can be chosen. An experiment was performed by connecting the three phases to a rectifier and attaching an electrical load across the motor terminals and in doing so revealed the following information shown in Table 1.

Variable Load Experiment Results	
Difficulty Levels	Resistance
1	100 $\Omega$
2	40 $\Omega$
3	25 $\Omega$
4	10 $\Omega$
5	5 $\Omega$
6	0 $\Omega$

Table 1. User difficulty level experiment

There are only six difficulty levels since adding more levels, subjectively, leads to negligible change within each level that the user would not be able to feel so this is the design choice we selected. Important information gained from this experiment is the load range required to have an “easy stroll” setting all the way to “mountain climbing” setting as well as the level of precision needed for the variable resistance component. Now that the range and precision is known, we can discuss various methods of applying an electric load to the motor. One method considered in the beginning is the application of digital potentiometers (digiPOTs) that are digitally controlled and serve the same function of a mechanical potentiometer. The least resistance offered is 1k ohms at a resolution of 256 Taps meaning each increment increases resistance by  $\sim 4$  ohms. The problem with this approach is the fact that these devices are meant for logic-level design and cannot handle the large power loads coming from the motor. More information about power load is discussed further in the report. A second option considered is the use of higher wattage mechanical potentiometers and simply attaching small, step motors that can vary the resistance from user input coming from the display. One thought to keep in mind is the cost required to buy large potentiometers and additional motors makes this a cost-ineffective solution that is also subject to mechanical wear. A third method considered was the use of BJTs and quickly switching it on and off so that the motor sees it as a resistor. The problem with this third method is the high power dissipation by using BJTs. A fourth method considered for achieving variable resistance is the use of a fet-based electronic switch. By switching very quickly between on and off, this switch can trick the motor into thinking it is a resistor since the motor will see the switch via a low-pass filter and thus consider it a resistor. This fourth option is the method chosen for this project since it is cost-effective in that no mechanical components are required and this option is less susceptible to failure. This voltage-controlled resistor (VCR) is such a switch that an input voltage applied to the VCR controls the value of the resistor on the output ports. VCRs are often made using field-effect transistors (FETs) that are advantageous over other transistors because the resistance when the switch is closed  $R_{ds(on)}$  is lower than others for example and thus less power dissipation. From the FET family, the MOSFET and JFET are appropriate

devices for VCR use. The MOSFET is advantageous for our application since it is meant for high noise applications such as switching which is exactly what we are using it for in our case [2]. Moreover, biasing the JFET requires a negative voltage source whereas MOSFET requires a positive gate voltage. Because both devices are similar and we are using positive voltage sources from the single-board computer, a MOSFET is the chosen device for the variable resistance block.

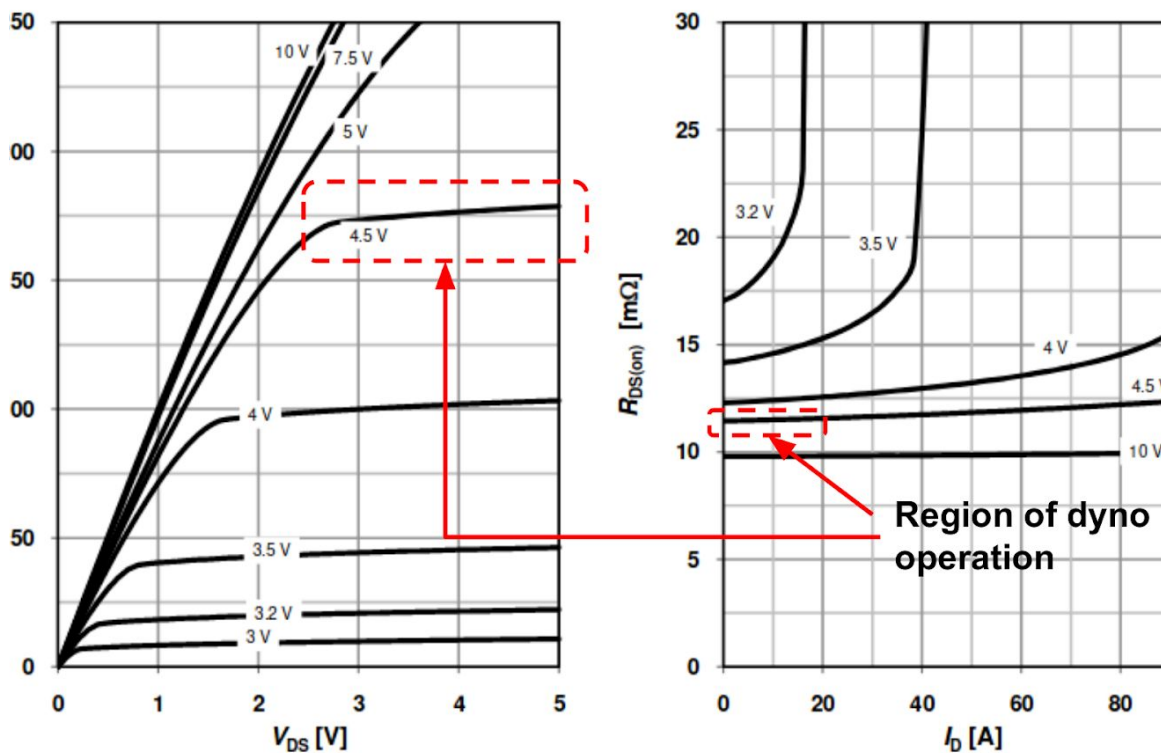


Figure 7. MOSFET regions of operation from mosfet datasheet showing consistent  $R_{ds(on)}$  resistance.

From the figure above, we see the typical three modes of operation known as triode, saturation, and cutoff. The triode region is problematic for our purpose since  $V_{ds}$  (motor voltage) can affect the drain current  $I_d$  (motor current) and thus the user difficulty level. Therefore, as long as the gate source voltage  $V_{gs}$  is lower than  $V_{ds}$  but greater than the threshold voltage  $V_t$  then we will stay in saturation. Saturation is advantageous since any further increase in motor voltage (user putting more power into the system) will lead to a constant current and therefore constant difficulty level. This is verified from the figure above in the red dashed region from the part we selected. Moreover, cutoff is also required as this will block current from flowing (meaning the switch is open). This condition is achieved as long as  $V_{gs}$  is less than  $V_t$ . By quickly switching from saturation to cutoff, we can effectively vary the shunt resistance across the motor since the motor will average things out.



$$I_D = \frac{\mu_n C_{ox}}{2} \frac{W}{L} [V_{GS} - V_{th}]^2 [1 + \lambda(V_{DS} - V_{DSsat})].$$

Figure 8. Drain current equation in mosfet saturation region.

From the figure above, it is clear that the drain current is not completely independent of user power output as seen from the  $V_{ds}$  term and the lambda coefficient. Therefore, it is important to model this equation given datasheet parameters to ensure this dependence is minimal.

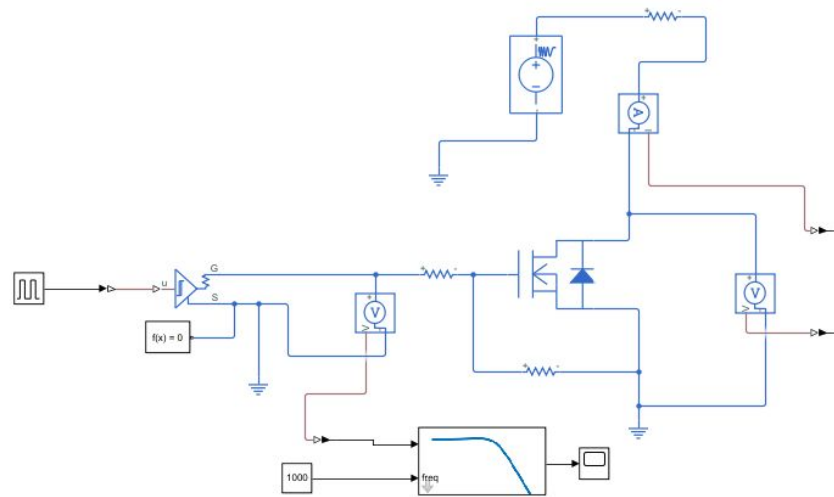


Figure 9. Simulink model of variable resistance block.

From this model, we can simulate the user overcoming rotational inertia and ramping up speed to discover the effect on  $R_{ds}$  over this period. In the figure below, we can see the wheelchair user speeding up as shown in the leftmost graph. This theoretical model is in agreement with experimental results. In the graph on the right, we see the shunt resistance applied to the motor over this period and the negligible  $\sim 1\text{ohm}$  variance experienced. Thus, the simulink model, in theory, is appropriate for the variable resistance block. An important point to note is the effect of inductance on the resistance values. Originally, an oversimplified motor model was used that did not include motor inductance and this was an incorrect assumption as we found a discrepancy from experimentation regarding PWM duty cycle and shunt resistance. Using the experimentally found motor inductance, the model was updated and gave better results regarding correct duty values. This is illustrated from the blue and yellow lines on the rightmost graph below that show how significantly motor inductance can affect simulation values since the original  $1\text{ ohm}$  jumped to nearly  $50\text{ ohms}$ .

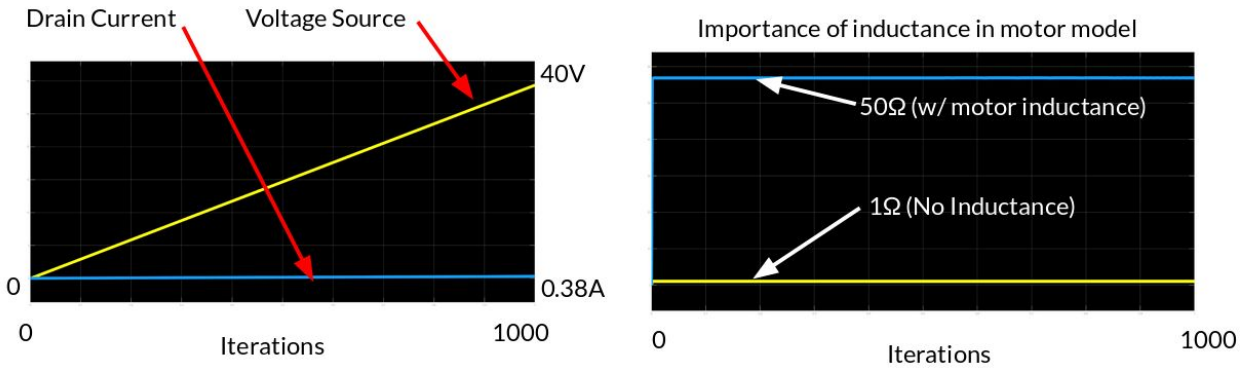


Figure 10. Simulink results of variable resistance model.

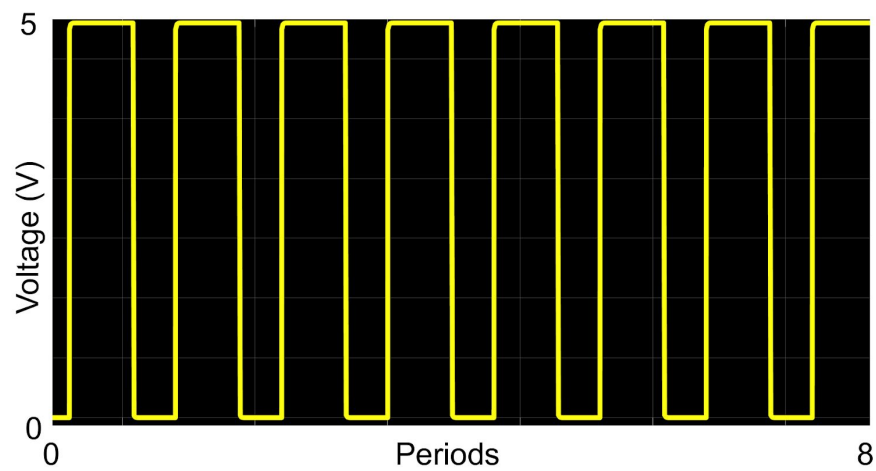


Figure 11. Simulink gate driver output being fed to mosfet.

Important information gained from this simulation is the PWM duty cycle needed to achieve different specific resistances. From the table below, the duty cycle needed to achieve the desired range is shown in the rightmost column. This last column is calculated by running the simulation shown above until the PWM duty cycle produces the desired subjective shunt resistance.

Simulation Results			
Difficulty Setting	Shunt Resistance	Workout Difficulty	PWM Duty Cycle
6	0 $\Omega$	Hardest	100%
5	5 $\Omega$	Hard	30%
4	25 $\Omega$	Medium	7.5%
3	40 $\Omega$	Normal	4.75%
2	100 $\Omega$	Easy	2%
1	Inf. $\Omega$	Easiest	0%

Table 2. Simulink results of proposed circuit.

Since this is an exercise machine but also a dyno, a compromise was made regarding the difficulty settings. For the easiest setting, we have infinite shunt resistance so that the max speed possible can be achieved. For the hardest setting, we have zero shunt resistance so the maximum torque can be achieved. Our sponsor wants this feature so that we can generate more precise force vs. speed graphs based on the axis values rather than perform some kind of interpolation technique. For the values in between, we have subjectively determined what shunt resistance is needed to feel an appropriate change in mechanical force as mentioned before.

Another consideration of this PWM duty cycle is the frequency this PWM signal will communicate with the MOSFET. The range of human hearing is from 20Hz to 20,000Hz generally. Therefore, the PWM frequency should be outside of this range to prevent any whine and audible noises during operation. However, as PWM frequency increases, then the transition time will increase and lead to more power dissipation so 100kHz is a good balance between the two factors considered. So now that these topics have been discussed, the MOSFET requirements must be considered before selecting the right part.

Since no motor datasheet exists for the motor bought, experimentation is needed to figure out motor parameters. Using the previous semester's experiment where one induction machine is used as a motor and the other end of the roller has a generator is useful for finding the motor constants.

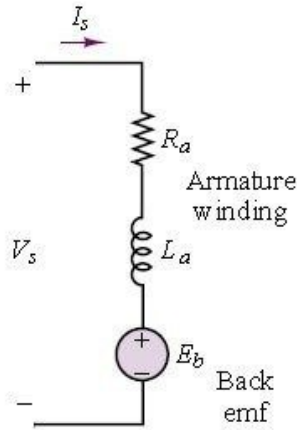


Figure 12. Motor circuit model. Image Courtesy: Circuit Globe.

From the figure above, this is the circuit model for the motors we are working with. Since the experiment keeps current constant, the voltage drop across the inductor is zero. By taking the no-load RPM and dividing the voltage supplied to the motor, the voltage motor constant can be found to be  $K_e = 0.1090$  [Vs/rad]. By applying a constant torque and observing the current during a shorted load, the torque motor constant can be found to be  $K_t = 0.1085$  [Nm/A]. In theory, both of these values should be the same and the fact that the error between them is low means that the values are correct since they were independently found. From both of these constants, the motor resistance can be calculated by the following equation:

$V = IR + K_e \omega$  [EQ1] since the speed, current, and voltage are known during any test point.

The value of motor resistance is 2 ohms. With these found values, we can perform an analysis of what kind of maximum voltage and current values we expect from wheelchair users to ensure the appropriate parts are bought. The world record for wheelchair racing is 18 MPH so the dyno is built for 20 MPH [3]. Taking into account the circumference of the rollers which are 13 1/8", the 20 MPH can be converted to 1,609 RPM. From the mechanical team associated with this project, they state a maximum human power of 186 W. This is a reasonable number considering the power output of arm ergometers are built for 150 W so this is in alignment with a similar workout device [4]. Since  $P = VI = \omega T$ , the torque is calculated to be 1.1 Nm. Thus, using EQ1 gives a voltage value of 40V. This means the voltage across the terminals of the motor can be as high as 40V so this must be considered for part selection. Moreover, the mechanical team gives a maximum force from the user to be 6.25 pounds as given from the excel sheet of experimental and theoretical data. Since  $T = rF$ , this translates to 1.5 Nm given the radius of the rollers which is around 2.1". Via the motor constant calculated, this means the current going through the load can be as high as 15 A. Thus, we are looking for a MOSFET that supports a drain source voltage  $V_{ds}$  of 40V and a drain current  $I_d$  of 15A. The figure below is the comparison table of the power MOSFET used. The IPP12CN10L G is selected even though it is slightly more expensive than

some of the other parts and this is due to a trade-off between power requirements, mosfet on resistance, and gate charge. Ideally, Rds on resistance should be low in order to have low power dissipation and a low gate charge so that the mosfet can be charged quickly. A large gate charge means that it is possible for switching to fail at higher speeds. Essentially, the PWM duty cycle wave can appear more sawtooth in nature rather than square wave so this is problematic for our application.

Mosfet Comparison					
Component Name	Breakdown Voltage Vds	Maximum Drain Current Id	Mosfet On Resistance Rds	Gate Charge	Price
IRFZ24NPBF	55 V	17 A	70 mOhms	13.3 nC	\$0.63
CSD18537NKCS	60 V	56 A	14 mOhms	14 nC	\$1.29
IPP12CN10L G	100 V	69 A	9.9 mOhms	58 nC	\$1.67
IRFB3806PBF	60 V	43 A	12.6 mOhms	22 nC	\$0.83

Table 3. Mosfet selection for variable resistance block.

One final consideration for the variable resistance block is the power dissipation since there is high power going into the MOSFET from the motor. Power dissipation occurs when the MOSFET is on (resistive loss) but also during switching (switching loss). This can be written as

$PD_{Total} = PD_{Resistive} + PD_{Switching}$ . The resistive component can be expressed as

$PD_{Resistive} = (I_{LOAD}^2 R_{DS(on)} V_{out}) / V_{in}$  where it is proportional to the load current, the on resistance, the drain source voltage, and the gate voltage when on. The second term can be expressed as  $PD_{Switch} = (C_{RSS} V_{IN}^2 f_{SW} I_{LOAD}) / I_{GATE}$  where it is related to the reverse transfer capacitance of the MOSFET chosen, the input voltage from the gate driver, the switching frequency, the load current, and the gate current given the driver chosen. Taking these two terms into account gives a power dissipation of around 10 W. Since the thermal resistance of junction to air is 60 C/W from the datasheet, this means the MOSFET selected can only support around 2W without requiring a heat sink. Thus, a heat sink is required.

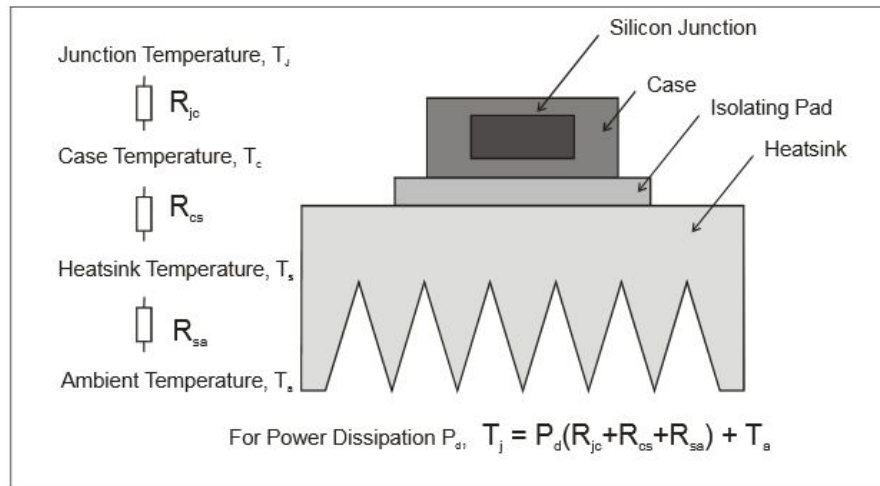


Figure 13. MOSFET thermal resistance model. Image Courtesy: RE-Innovation.

Given the figure above, the junction temperature can be calculated from

$T_j = P D_{Total}(R_{JC} + R_{CS} + R_{SA}) + T_A$ . The junction to case thermal resistance is given in the datasheet to be 1.2 C/W. The case to heatsink is dependent on the thermal paste used but a liberal value of 0.85 C/W can be used. The last thermal resistance is dependent on the heat sink used and the value used that gives an excellent junction temperature is 5 C/W. Finally, the ambient temperature is 25C so all of these values give a junction temperature to be around 105C. The maximum junction temperature the device operates in is 175C but the junction should not be near this for extended periods of time. It is recommended to be less than 75% of the maximum junction temperature to extend the life of the device and prevent damage to the MOSFET. Therefore, 100C < 135C and the mosfet is adequately cooled given the right heatsink that has the 5 C/W thermal resistance.



## Gate Driver

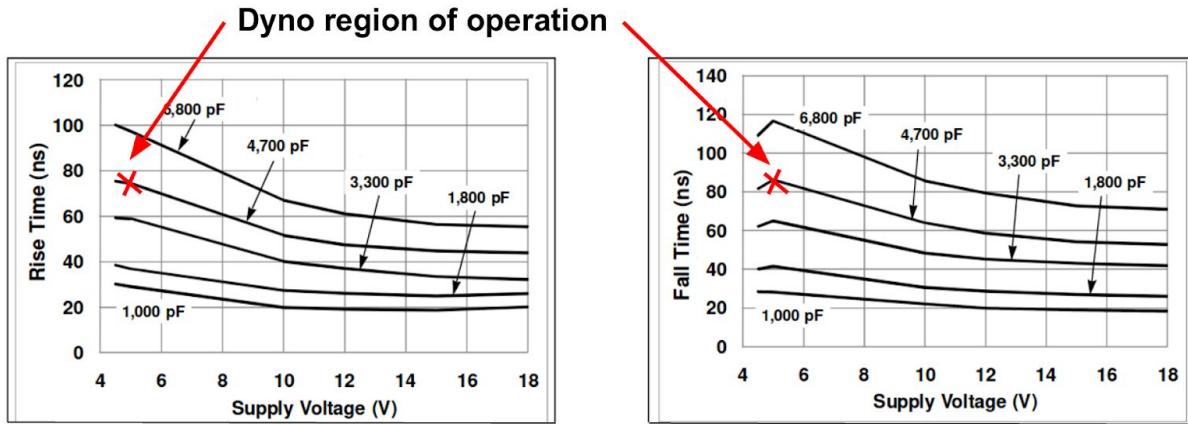


Figure 14. Gate driver rise and fall times for dyno given mosfet parameters and supply voltage.

Regardless of what microcontroller or single-board computer is used, the current provided from the computer is in the milli range out of the output pins. For our project, a power mosfet is required as opposed to a normal mosfet since the motor is a high-power device. Because of the construction of these devices, the input gate capacitance is higher than low-power mosfets. Normally, the gate capacitance is in the pico range but power mosfets have gate capacitance in the nano range. This means that it takes longer to charge the capacitor than low-power mosfets. Thus, this leads to higher transition times during on-off switching. Since the single-board computer provides low current, this is problematic to transition quickly. A gate driver can take the PWM signal and ensure sharp transitions by supplying large current during charge and discharge times. There is no general guideline for what transition times we need for our project. Even though there is no requirement, switching time must be reasonable to ensure low power dissipation and correct functionality. As stated during switching, the gate capacitance of the power mosfet may draw current so quickly that it causes a current overdraw in the microcontroller or logic circuit. Therefore, power mosfets are used in conjunction with gate drivers in the 1-3 A range for most motor switching applications to account for the current overdraw. The maximum drive current must be high enough to drive the gate resistor value and the chosen gate voltage swing [26]. This relationship is stated as follows  $I_{MAX} \geq 0.7 \cdot \frac{\Delta V_{GATE}}{R_{GATE}}$  [26]. From the mosfet datasheet, we have gate resistance of 1.3 Ohms and the voltage swing (5V) is based on the threshold voltage to ensure the mosfet is fully on. Given these values,  $I_{MAX} \geq 2.3 A$  so a 3A gate driver is most ideal based on the power mosfet selected. Transition times, in general, should only occur about 1% of the total time so this translates to a maximum frequency switching support stated as follows  $f_{max} = \frac{1}{200 \cdot (t_{rise} + t_{fall})}$  [27]. This is done to make sure the waveform does not become distorted and the motors fail to adjust the mechanical resistance.

Given a 60nS rise and fall time, this translates to a 400kHz maximum switching speed. This is above our application switching speed so a gate driver with a lower rise and fall time is ideal for our application. The table below compares the different gate drivers and the one selected for our design is the MCP14E10.

From this gate driver, a gate resistor is attached from the output to the gate pin of the mosfet. This is done to reduce ringing and interference caused by switching the mosfet on and off too quickly [19]. Moreover, a gate-source resistor is included with the mosfet in order to ground residual charge and keep the gate pin from being left floating [20].

Gate Driver Comparison					
Component Name	Supply Voltage	Output Current	Rise / Fall Times	Maximum Operating Temperature	Price
IX4424G	4.5-30 V	2 A	60 nS / 60 nS	125 C	\$1.58
MCP14E10	4.5-18 V	3 A	14 nS / 17 nS	125 C	\$1.86
MIC4423ZN	4.5-18 V	3 A	28 nS / 32 nS	70 C	\$2.20
MIC4424YN	4.5-18 V	1 A	35 nS / 35 nS	85 C	\$1.77

Table 4. Gate driver part selection based on parameters discussed.

## Rectifier

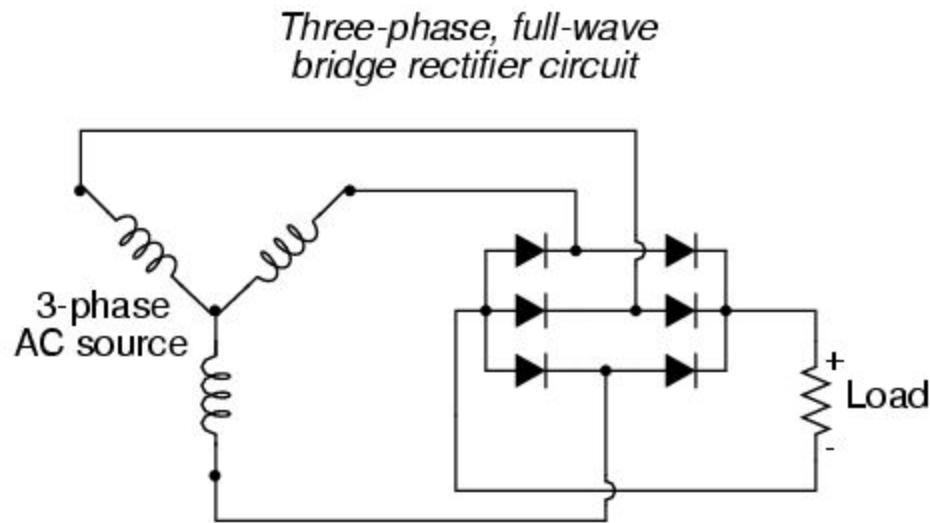


Figure 15. Standard three-phase rectifier connected to load. Image Courtesy: miernik - stackexchange.

Since our dyno involves a three phase motor, the information must be converted to a single dc signal so that we can apply a shunt resistance and also read the total current and voltage coming from the motor. By applying a shunt resistance on the output of the rectifier, we are applying a shunt resistance to each of the phases so this simplifies the design. In order to read the AC signals that the motor is outputting, diodes are needed to convert the signal to DC. A three-phase rectifier is the simplest design choice that works for our application. Since we have three-phase motors, the motors output three different signals which are 120 degrees out of phase from each other. By utilizing the rectifier, it will straighten the three-phase signals and give a single variable load that can be read. Based on the previous calculation in the report, we expect a maximum of 40V and 15A given the load constraints, therefore a three-phase rectifier that can support the load specifications is required. Most of the rectifiers considered in the table below fit the specification but some go above and beyond the necessary requirements. These devices are not ideal because they give roughly  $\sim 120\text{m}\Omega$  resistance per diode while the most cost-effective option that we chose that meets our requirements has  $\sim 40\text{m}\Omega$  resistance per diode. This parameter is important since we need the diode resistance to be low so that we get closer to motor stalling conditions. If the inherent resistance of our circuit is high, then the motor may not give the highest resistance setting possible so it is critical to get the component with the least resistance per diode. Based on this information, the FUS45-0045B was selected that meets our minimum constraints and has the least resistance per diode. The part selected uses Schottky diodes which are constructed differently from the standard silicon diodes which is why this part has a lower resistance per diode than the other rectifiers shown.

Rectifier Comparison					
Component Name	Peak Reverse Voltage	Maximum Surge Current	Maximum Operating Temperature	Diode Resistance	Price
FUO22-12N	1200 V	22 A	150 C	120mΩ	\$6.88
GUO40-12NO1	1200 V	40 A	150 C	80mΩ	\$9.02
FUE30-12N1	1200 V	30 A	175 C	140mΩ	\$11.23
FUS45-0045B	45 V	45 A	150 C	36mΩ	\$6.83

Table 5. Rectifier part selection based on specified constraints.

## Single-Board Computer

One of the requirements for our project is a hosted web server that can display user information during workout sessions. This is a good design choice since a web browser works cross-platform and no specific app has to be coded for android or iOS for example. A microcontroller cannot host a web server so a single-board computer, such as a Raspberry Pi, is required to host the web server and user interface.

Single-Board Computer Comparison					
Computers	Wireless Capabilities	Processor	RAM Size	GPIO	Price
<i>ODROID-C2</i>	None	2 Ghz ARM-A53	2 GB	Analog + Digital	\$40
<i>Libre AML-S905X</i>	None	1.4 Ghz ARM-A53	2 GB	Only Digital	\$45
<i>Orange Pi 4G-IOT</i>	Wifi + Bluetooth	1 Ghz ARM-A53	1 GB	Only Digital	\$45
<i>Raspberry Pi 4</i>	Wifi + Bluetooth	1.5 GHz ARM-A72	1-4 GB	Only Digital	\$35
<i>BeagleBone Black</i>	None	1 GHz ARM-A8	512 MB	Analog + Digital	\$55

Table 6. Comparison of single-board computers.

By researching different components, the following information was found above. Constraints for the dyno are the following: wireless capabilities so that a user can connect to the web server, and analog pins to read speed and torque output along with a PWM channel for variable resistance. However, devices found to meet these requirements started at \$60 which is cost-ineffective. The Raspberry Pi turned out to be one of the cheapest single-board computers that met the wireless requirements. However, this device does not come with analog pins so an ADC must be used in conjunction. This combination of a Pi with an ADC is more cost-effective than buying a device that comes with all the necessary features so this is the chosen single-board computer. The first step is to install the web server software. There are multiple software that achieve this purpose but Apache 2 was chosen since it has around 50% share in the industry meaning there is extensive documentation if something does not work correctly. Other solutions are more advantageous if there are more than a million users accessing the server so this is not a concern. Using [5], the web server can be started locally on the Pi along with PHP which is needed to generate dynamic content on HTML websites. This feature is needed as workout data is stored so the HTML pages must be updated via PHP so end-users can access information. The next step is to enable .htaccess on Apache. This is required so that the workout data directory is defined and the file indexer recognizes the directory. This can be done via [6]. After this, the file indexer can be installed. For this project, the open-source and MIT-licensed modern file indexer h5ai is used to see data at a later point in time. The homepage includes installation instructions for this [7]. From here, the contents of the 'html' folder included in the deliverables can be dropped at '/var/www/html'. Typing 'localhost' or 'dyno.local' in a web browser gives the following result.

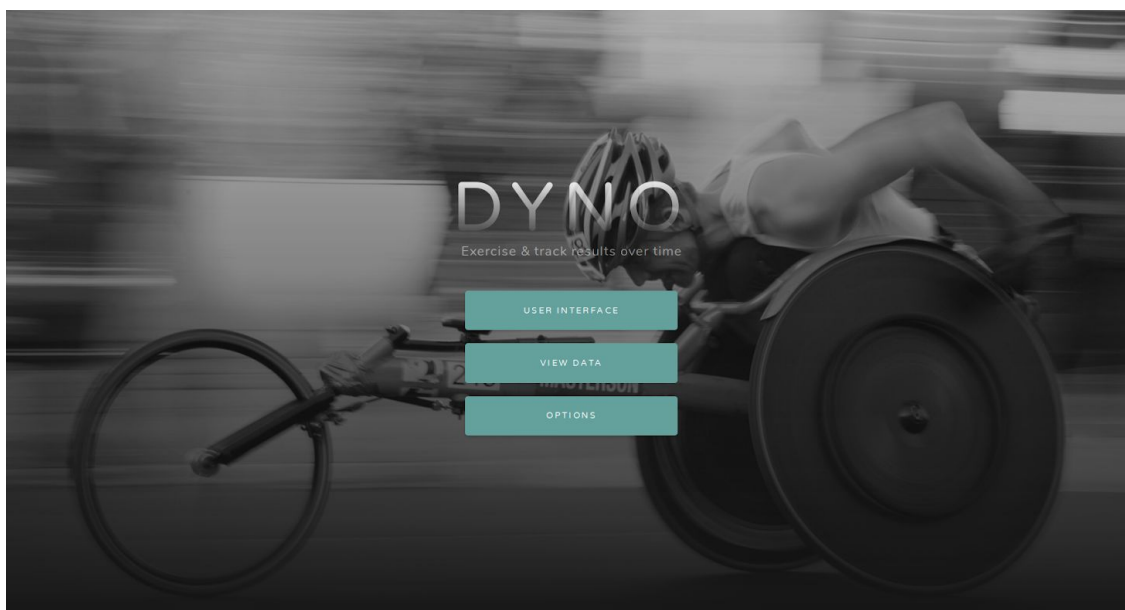


Figure 16. Main page for wheelchair dyno interface.

From the h5ai installation, if the chosen folder name is 'data' located within 'var/www/html' then the button 'View Data' should redirect them to the data directory that contains saved information. The program running in the background saves workout sessions in corresponding folders. If multiple sessions are recorded in one day then the information is recorded as 'sessionX' for example. Data can be downloaded by selecting the check mark that appears when hovering over the folder and selecting the download icon. The following figure below shows how the data is visualized when the user is looking at past data.

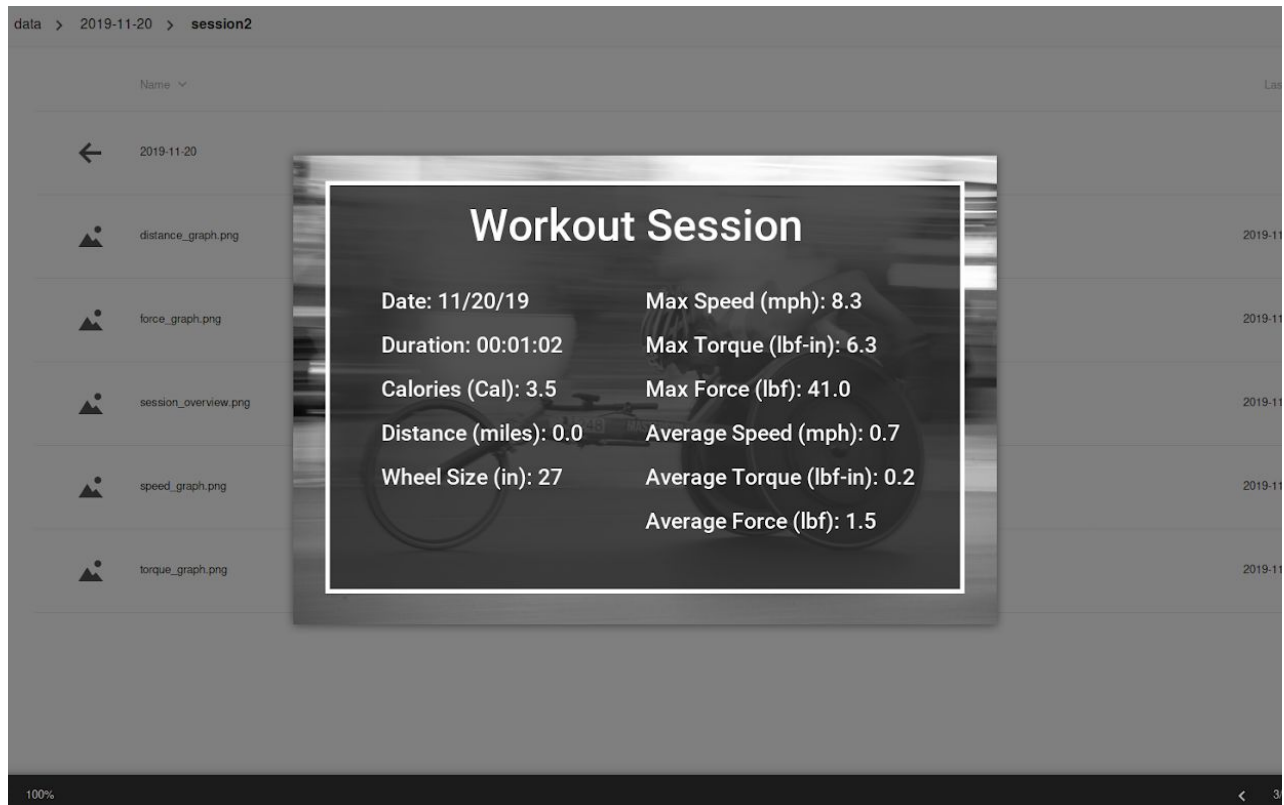


Figure 17. File indexer with previous recorded workout session data.

For the user interface, an internet of things platform is utilized. Caution, a platform is different from a dashboard in that a dashboard only shows data but provides no way to interact with the single-board computer. Analyzing different IOT platforms, Thingsboard was selected due to the amount of documentation available online as opposed to others such as Thingsstream or Thingworx. The installation guide for Raspberry Pi can be selected from the main installation page located here [8]. The thingsboard server is running locally on 'localhost:8080'. By logging in as a 'tenant' described in the installation, the dashboard can be created. Using the dashboard files located in the 'dashboard' directory of the deliverables, the dashboard used during demo can be copied to this new setup. On the dashboard page of Thingsboard, there is an import dashboard feature that allows it to copy over settings from the 'dashboard' directory. Once this is



completed, the following dashboard will become available by selecting the ‘User Interface’ button on the main page.

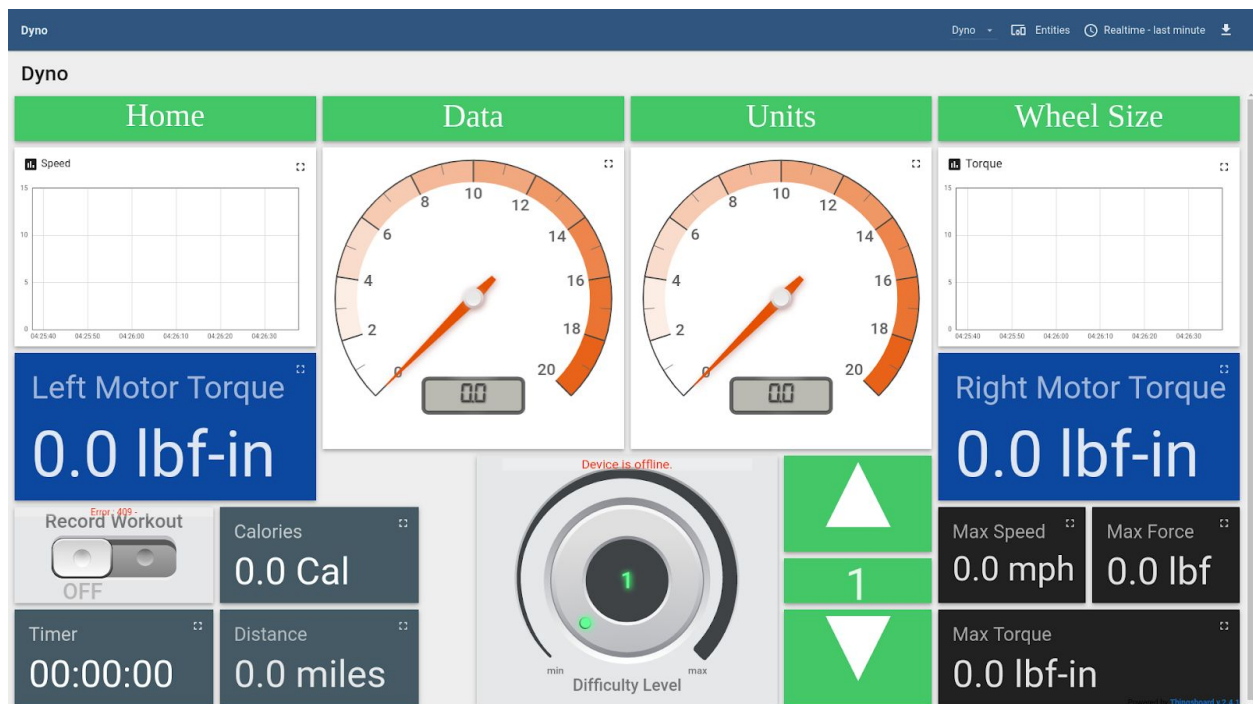


Figure 18. User interface main page.

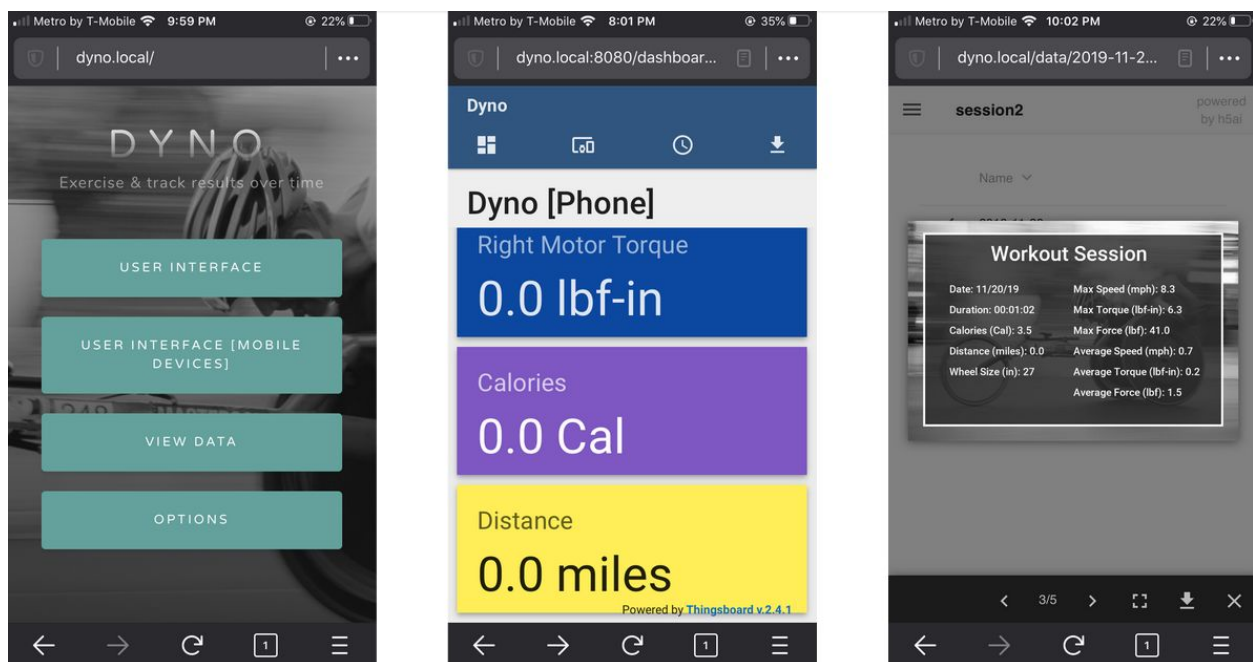


Figure 19. Mobile user interface optimized.

The next step is to install the libraries needed for functionality. The ADC is read using python code via SPI so the Adafruit library can be installed [9]. Moreover, Javascript code handles the PWM duty cycle output so the javascript library for pin interaction must be installed [10]. Once this is done, the code files from the 'code' deliverables can be dropped to the home directory. To get data from the user, 'python adc.py' can be run from the terminal while resistance interaction can be controlled via 'bash motor\_control.sh' if the terminal is located at the home directory. However, this is not needed since the 'startup.sh' file in the deliverables can be added to run during startup such that the necessary code starts running when the machine is turned on. To add the 'startup.sh' file located in the 'code' deliverables, the following tutorial will explain how to edit the startup list [11]. In essence, 'sudo crontab -e' and adding '@reboot sudo bash startup.sh' will achieve this. The final step for the microcontroller setup involves enabling the wireless access point so that any user can connect to the network. The following page explains how to perform this with any network name and password wished to be set [12]. Finally, the hostname must be set to dyno so that 'dyno.local' becomes the main page of the wheelchair dyno when a user types that in the address bar so the following page explains how to set that [13]. For our dyno system the EESD is 'Dyno' and the password is 'dynamometer'. This completes the tutorial for setting up the single-board computer. If there are any issues following the tutorial, the following article explains how to make a backup of an sd card and copy it to another one [14]. This is helpful to make an exact replica of the current sd card that can be used with any Raspberry Pi model 3/4 for further modification if any instructions remain unclear. There are SD card image backups located on the Microsoft Teams group deliverables. These images can be copied to SD cards following the appropriate tutorial [14]. For the display attached to the dyno, the file 'start\_display.sh' in the deliverables must be added to startup using the previous article [11]. However, an SD card image is also provided for the attached display that can be deployed easily.

Moving on to the software files used to complete the project, the software flowcharts for our code is shown below. The software flowchart below is for the main file 'adc\_data.py' that gets ADC values, sets resistance settings, and deals with recording data.

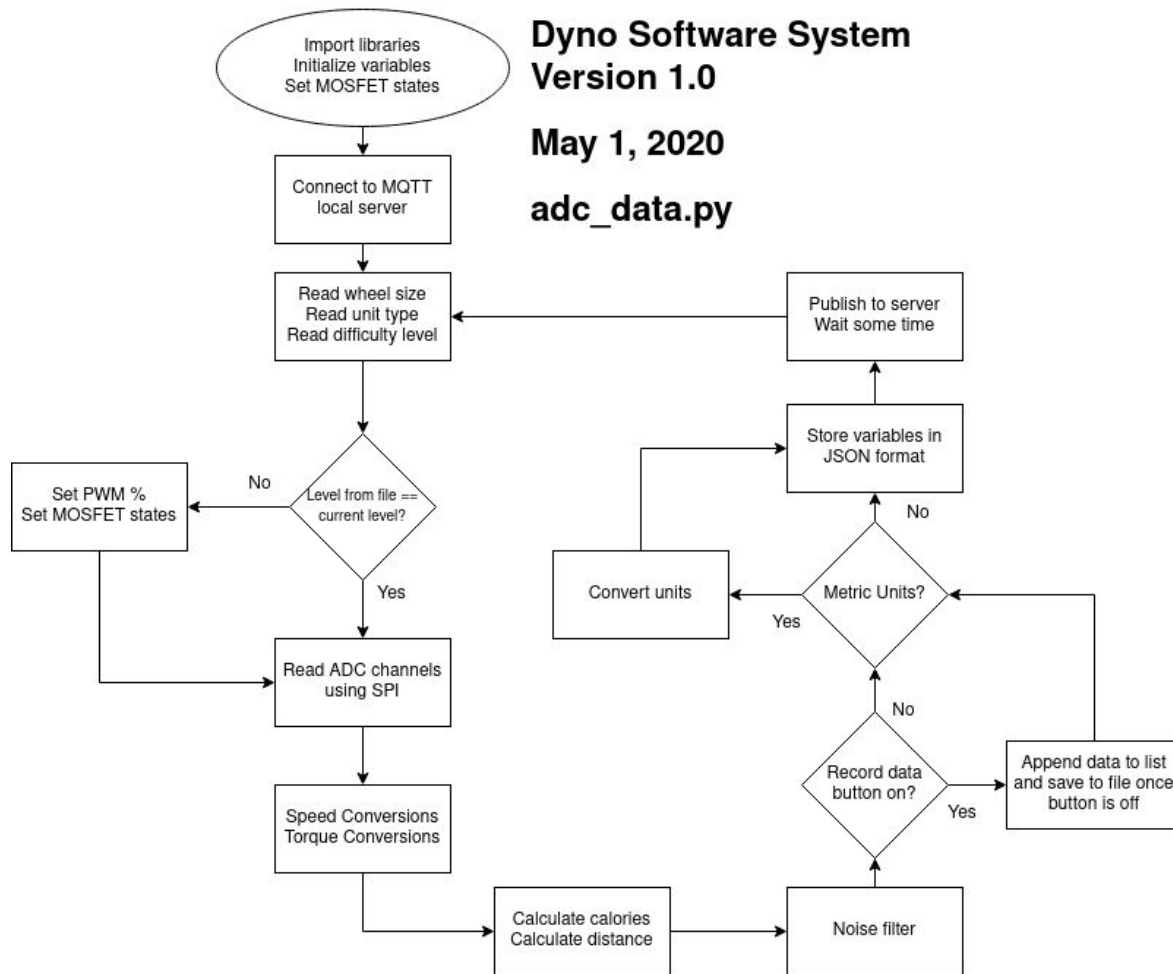


Figure 20. Software flowchart for the main software system.

The figure above illustrates the flowchart for the software system on the dyno. At the beginning of the main file, the necessary libraries are imported that deal with the ADC chip, the MQTT server that receives variables from the code file, and basic libraries for computation. The necessary variables that store speed, torque, distance, etc... are initialized at the start of the file and the MOSFET states are set. This is to ensure that the output of the voltage divider doesn't exceed 5V which would damage the ADC if it did. The next step is to connect to the MQTT server that received data from the file. The thingsboard server is looking at this MQTT server to report data on the GUI. Once this is done, we enter the loop of the file that will stay for the rest of the workout session. At the beginning of the loop, the system will read text files associated with the wheel size, the unit type, and the current difficulty level. If the level in the code does not match the text file, then the code will set the new PWM duty cycle and the MOSFETS for the voltage divider in order to set the new gain. Once this is complete, the channels are read from the ADC using SPI. The ADC values are converted to speed and torque information by the appropriate formulas discussed in this report. From this information, the information is filtered to

prevent small noise from showing up as a speed or torque value. This speed and torque information is used to calculate the caloric expenditure and distance traveled. Once this is done, we check if the user has pressed the record data button. If they have done so, then the variables are appended to a list continuously until the user turns it off. Once off, the data is stored in a CSV file that the plot file then reads and generates the appropriate graphs. From here, the code checks whether the metric system unit flag is turned off. If it is, then the values are converted to the metric type. Finally, all of the variables are stored in a JSON array that is sent to the server. The code starts from the beginning of the loop again and reads wheel size information, unit type, and difficulty level.

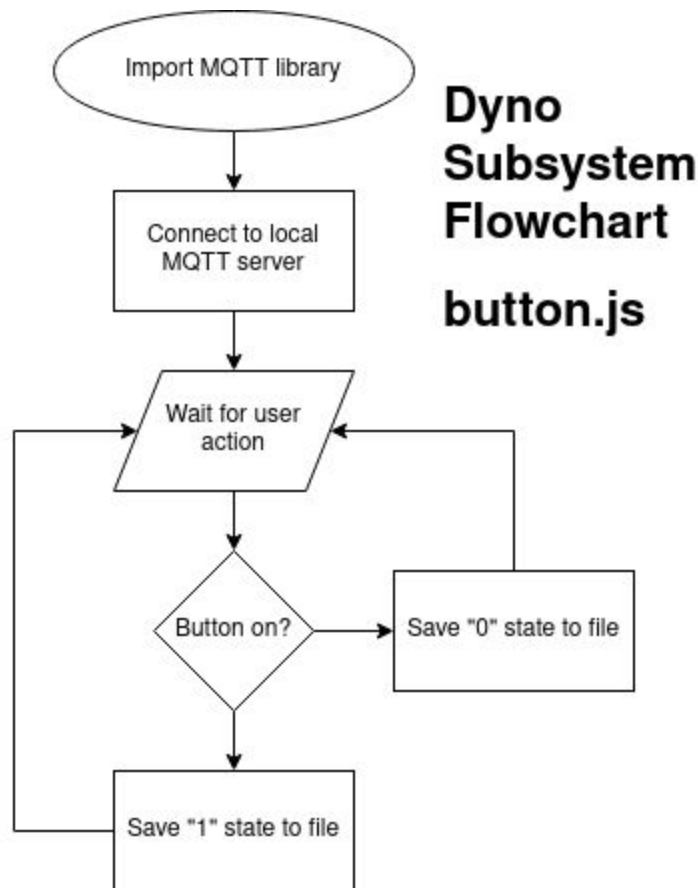


Figure 21. Software flowchart for the recording data subsystem.

From the figure above, this subsystem is responsible for recording the state of the record data button to a file. The main system will read this file to determine whether or not to start recording data as explained above. In this flowchart, the MQTT library is imported since we are listening to the server and waiting for the user to perform an action. Based on the state of the action, the state is recorded to a file.

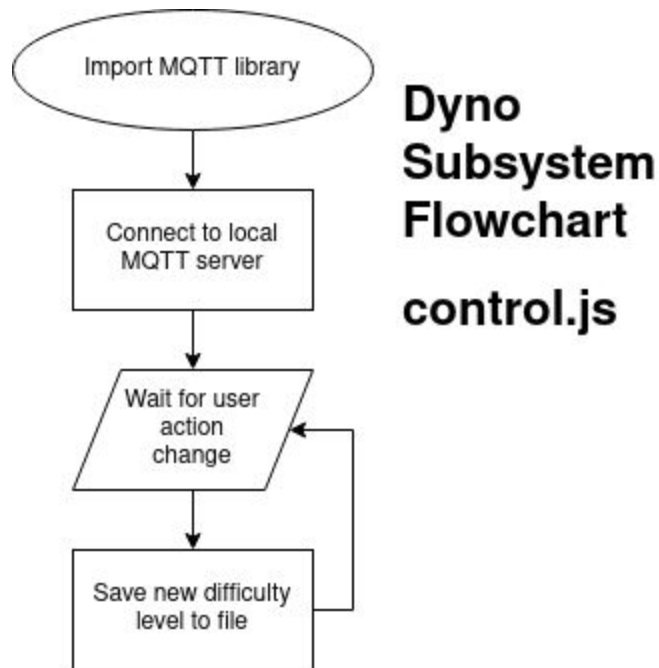


Figure 22. Software flowchart for control knob subsystem.

From above, this software flowchart saves the state of the control knob when it changes to a file. The main system reads this file to update the PWM duty cycle associated with that workout difficulty level. This software flowchart is essentially the same as the record data button subsystem so no extra discussion will be provided regarding this subsystem.

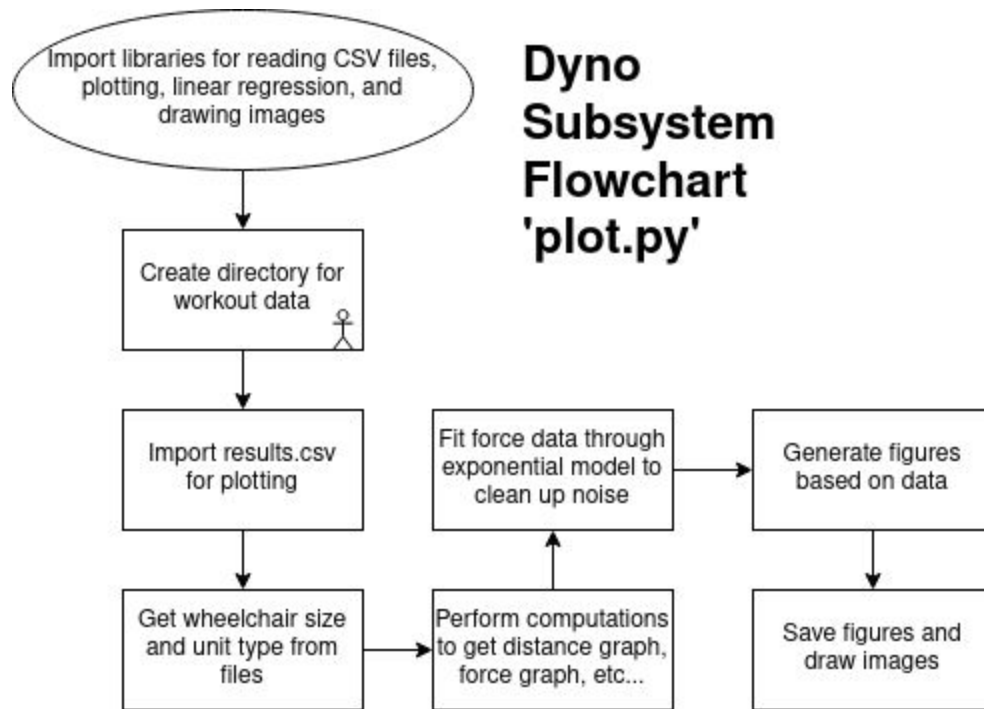


Figure 23. Software flowchart for plotting workout data.

In the flowchart above, this function is responsible for plotting data once the record data button is deactivated. Firstly, the necessary libraries are imported that are required to read the recorded data, plot graphs, perform linear regression on the force data to remove noise, and the library needed to draw images. Once this is done, a directory is created to save all data to this folder. The csv file is imported and wheelchair size and unit type is imported to save information based on the current configuration. Then, the necessary conversions are done to get caloric information, distance over time, force calculations from torque information, etc... The force data is plugged into a linear regression model (exponential) since the data plotted is better understood this way. For example, on the x-axis, we have all speed and no torque and the user can go at various speeds so we have multiple points scattered on the x-axis. This makes it difficult to plot sequential data so this method of linear regression works better to create more meaningful data. Afterwards, the figures are generated and images drawn are saved to the directory.



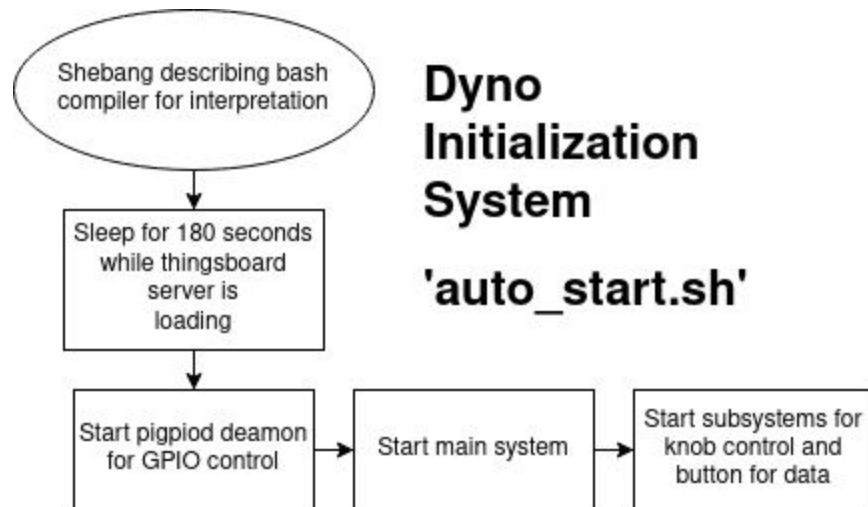


Figure 24. Software flowchart for initialization script.

In the flowchart above, this script is the initialization script that runs when the computer is turned on. In essence, the script waits 180 seconds while the thingsboard server is loading then runs the main system followed by the subsystems.

## Torque Measurements

The relationship between the current through the motor terminals and motor torque can be explained by  $\tau = K_t I$  where  $K_t$  is the torque motor constant and  $I$  is the current. Thus, by sensing the current in the system, we have a direct relationship with torque. There are two main approaches which are direct current sensing and indirect current sensing. The first approach relies on using a resistor with a small resistance so that it does not impact the loading of the circuit in combination with some kind of amplifier that amplifies the small differential voltage. The second approach relies on concepts like a wire creating a magnetic field and detecting the magnetic field via a hall effect sensor. The second approach is commonly used with large ( $>100A$ ) currents while direct approaches work better for smaller currents [15]. Indirect sensing is more complex and thus higher cost but comes at the advantage of noninvasive techniques [16]. Since there was no specific advantage of indirect current sensing required for the application, the direct sensing method is used for the project. With the direct approach, there are two main methods of current sensing which are low-side sensing and high-side sensing. Low-side sensing involves putting the sensing resistor between the load and ground while high-side sensing involves putting the sensing resistor between power supply and load. By our application, this implies we can either put the resistor between the MOSFET and ground or the motor and MOSFET.

	High-side sensing	Low-side sensing
Implementation	Differential input	Single or differential input
Ground disturbance	No	Yes
Common voltage	Close to supply	Close to ground
Common-mode rejection ratio (CMRR) requirements	Higher	Lower
Can detect load shorts?	Yes	No

Table 7. Comparison of low-side and high-side sensing circuits. Image Courtesy: TI.

From the table above, there are different advantages to using high-side sensing circuits which include the ability to detect shorts, less ground disturbances/parasitics, and higher CMRR that leads to better accuracy [16]. However, high-side sensors usually only come in surface mount parts and are more expensive than low-side sensors [16]. Given the project restraints, there is not any necessary feature from high-side sensing required so a low-side sensor is implemented. This is the preferred choice for our sponsor as well.

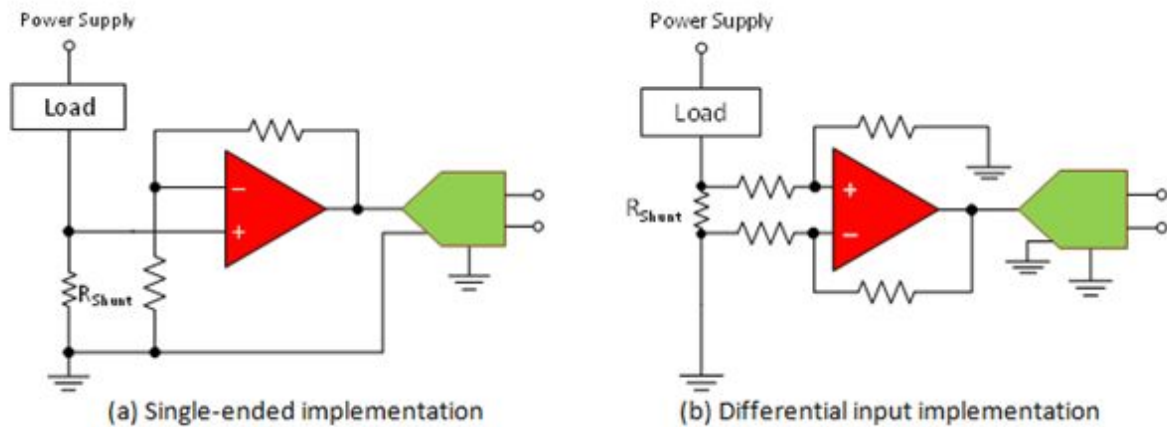


Figure 25. The two different topologies for low-side current sensing. Image Courtesy: TI.

From above, we can see the single-ended topology and the differential input topology that can both be used in low-side sensing applications. While single-ended is a simpler design that utilizes only two less resistors than differential implementation, the differential topology can achieve higher accuracy since the two extra resistors help with parasitics and ground disturbances as explained in [16]. It is important to get an operational amplifier that contains low voltage offset ( $<150\mu\text{V}$ ) so that high precision is achieved along with a differential voltage that

meets the requirements given the maximum expected current of 15A [16]. The part selected that fits these requirements while being cost-effective is the LT1490 as seen in Table 8. Current sense resistors are typically in the 1-100 mOhm range to prevent any loading effects [17]. The value chosen is 100 mOhm so that the power dissipation remains low. By doing this, only a 5W resistor is needed to handle the power. A smaller resistor can be chosen but this means that the resistance comes fairly close to negligible as the PCB lines may have around 5 mOhm resistance. Since the maximum current is 15A, this gives the voltage drop across the resistor. By setting  $Gain = R_2/R_1 = 10$ , this will ensure that the maximum voltage output is 5V which is what the ADC expects the maximum value to be. Thus, the current going through the system is calculated by  $I = V_{ADC}/(10 * 0.1)$  since the gain and current resistor values have to be taken into account. By this equation and the fact that the motor constant is around 0.1 [Nm/A], the torque is calculated from the following  $\approx \frac{V_{ADC}}{10}$  in Nm units. The table below compares different operational amplifiers out in the market that suit our requirements. The two components that make a differential amplifier a good current sense amplifier is the following: input offset voltage and common mode rejection ratio (CMRR). A low input offset voltage (<150 uV) and a high CMRR (>80 dB) are distinguishable characteristics of a current sense amplifier [25]. There is no general guideline for our project but the LT1490 is selected due to a good price to performance tradeoff. Since the team last year reported an inverse parabolic relationship between PWM duty cycle and current, it is possible that the part they selected was not well suited for precision at a larger differential voltage. Therefore, the part selected for current sensing has a lower input offset voltage and higher CMRR than the differential amplifier used last year. This is done to prevent any sensing issues from occurring like the previous group experienced. The part is simulated via MultiSim to ensure the current sensing block is correctly designed. We can see in the figure below that the input wave is amplified with the right gain amount as expected.

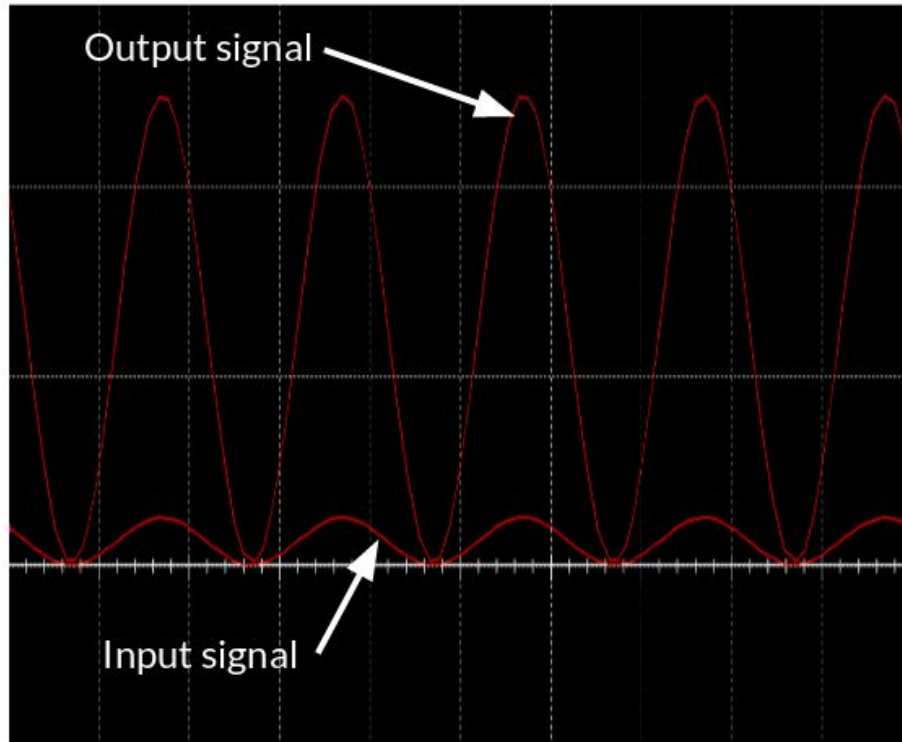


Figure 26. Current sense block simulation results.

Operational Amplifiers Comparison				
Component	Input Offset Voltage	Channels	CMRR	Price
LT1490	110 uV	2	98 dB	\$5.09
OPA2227PA	75 uV	2	130 dB	\$5.48
OPA2336P	125 uV	2	80 dB	\$6.07

Table 8. Comparison of operational amplifiers.

## Filtering

Since the full-wave rectifier converts 3-phase ac power to single phase dc power, the voltage ripple factor is 0.48 [23]. Thus, a large noise level can affect measurements of speed and torque. However, an RC filter can reduce the ripple factor by an order of 10 [24]. A second order RC filter will reduce the ripple factor by an order of 100 [24]. Given the maximum voltage of 40V, this leads to a noise level of  $0.0048 * 40 = 0.192 \text{ mV}$ . The reference voltage is set to 5V and the ADC resolution is 10-bit so this translates to  $\frac{5}{2^{10}} = 5 \text{ mV}$ . Therefore, the noise is within 40 levels

of the true value in the ADC. This is within 4% error in the worst case scenario of a 40V maximum user output. The cutoff frequency is selected to be 5Hz given the advice of our sponsor on this issue. There is no datasheet available for the motors so it is difficult to pinpoint the cutoff frequency in theory. Our sponsor suggested 5Hz would be a good tradeoff for filtering and system responsiveness. Preliminary results confirm a good baseline for the cut-off frequency.

## Speed Measurements

The relationship between speed and back e.m.f. is expressed as  $V_{emf} = K_e \omega$  and thus the speed can be calculated by knowing the back e.m.f. As shown in the figure below, the back emf can be calculated by knowing the inductance and resistance of the motor along with keeping track of current and the rate of change of current.

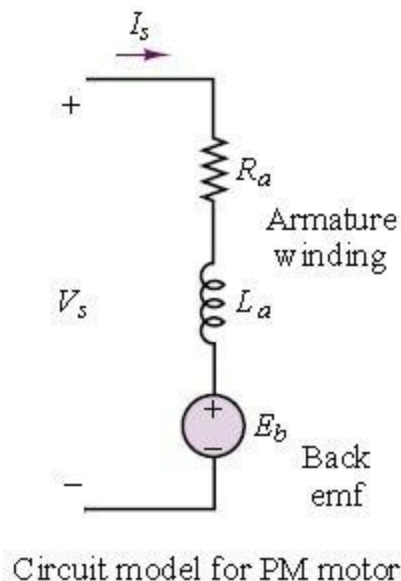


Figure 27. Motor circuit model. Image Courtesy: Circuit Globe.

However, this approach was not initially utilized since it relies on an indirect detection method. Since hall effect sensors are included within the motor, the speed can be directly calculated by noticing the frequency of the output hall effect signal. A hall effect sensor is a transducer that measures the magnitude of the magnetic field. Thus, the amplitude of the signal corresponds to the location of the roller orientation. The motor includes one hall effect sensor per phase so when the sensor is aligned to the magnetic field of the motor poles, then the frequency information of the roller is contained in the signal. Since no datasheet is included for the motors, an experiment

had to be performed in order to understand what kind of output to expect from the hall effect sensor.

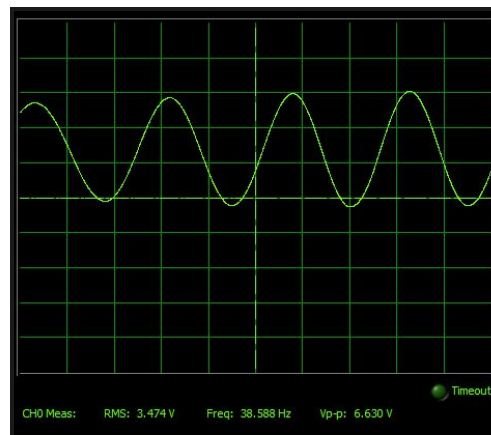


Figure 28. Hall effect sensor output as the roller is spinning.

From the figure above, the output voltage range of the hall effect sensor is from around 0V to 6.6V regardless of how hard the person is spinning the rollers. What does change is the frequency of the sine wave where the theoretical frequency range is found to be 0Hz to 125Hz. Thus, a circuit block is required that extracts frequency information from the signal and maps it to an analog voltage value that corresponds to frequency. By doing it this way, an ADC channel can be used on the ADC chip.

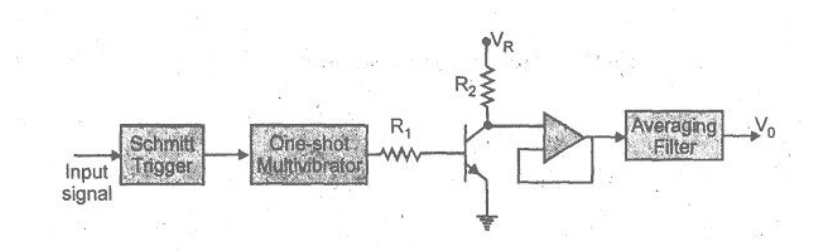


Figure 29. The basic outline of a frequency to voltage block diagram. Image Courtesy: MicrocontrollersLab.

As seen from above, the schmitt trigger converts the sine wave to a square wave that changes in response to the sign of the sine wave. Hence, it is like a zero crossing detector that conveys information about the frequency of the signal [18]. This information is passed to the multivibrator and BJT that generates a pulse with a specific amplitude and frequency [18]. Essentially, some transformations are performed on the Schmitt wave such that it can be fed to the averaging filter where only low frequency content is passed. The averaging filter will output the analog DC signal whose amplitude is related to the  $R_1$  and  $R_2$  values that decide the frequency per volt division. This is the basic diagram for how these converters function. Since

we require a chip with a maximum frequency support of 125 Hz, only chips with the lowest option (10kHz max) were investigated.

Tachometer				
	LM2907N-8	LM2917N-8	LM2917N-14	LM2907N-14
Frequency (kHz)	10	10	10	10
Input Voltage (V)	(-28) - 28	(-28) - 28	0 - 28	0 - 28
Cost	\$2.33	\$2.13	\$2.13	\$2.33

Table 9. Different frequency voltage converters on the market.

Given the fact that the hall effect signal amplitude goes from a slightly negative value to around 6.6V, the appropriate part LM2917N-8 was selected since it is the cheapest component on the market that meets our requirements. From the datasheet, under the applications section, there is a layed out schematic for converting a hall effect sensor signal to an output analog signal as shown in the figure below.

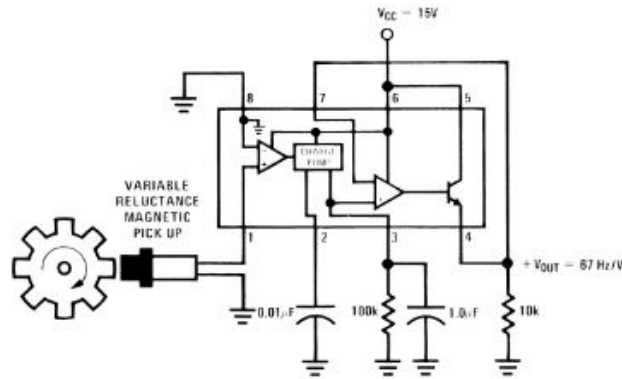


Figure 30. Tachometer layout given in the datasheet similar to our configuration.

The same layout is followed for our dyno project with the exception of R1 and C1. The output voltage is expressed as  $V_{OUT} = f_{IN} V_{CC} R_1 C_1$  where Vcc is the supply voltage and R1 and C1 are components read from left to right in the figure above. Thus, as long as  $R_1 C_1 = \frac{1}{f_{IN}}$ , then the output voltage will go as high as the supply voltage which is 5V per our design. R1 is ideally in the kOhm range and C1 is in the uF range per standard. The second capacitor from left to right, C2, reduces the ripple factor in the output signal. From processing the signal, there will be a ripple factor that is expressed by  $V_{RIPPLE} = \frac{V_{CC}}{2} \frac{C_1}{C_2} (1 - \frac{V_{CC} f_{IN} C_1}{I_2})$ . From this equation, C2 can



be selected by how much ripple is allowed.  $I_2$  is expressed as follows  $I_2 = f_{MAX} C_1 V_{CC}$ . For our application,  $C_2$  is chosen as  $1\mu F$  so that the ripple factor is within a few millivolts of the output signal so that no external filtering is required. Caution must be taken not to reduce the ripple factor greatly since there is a tradeoff between system responsiveness and noise reduction. From this speed sensor the frequency is calculated as  $f = \frac{V_{OUT}}{V_{CC} R_1 C_1}$ . Based on the points discussed above, our values for the constants are as follows:

$V_{CC} = 5V$ ,  $R_1 = 80k\Omega$ ,  $C_1 = 0.1\mu F$ ,  $C_2 = 1\mu F$ . It should be apparent that frequency in Hz units is exactly the same as the number of revolutions completed per second or RPS. From this, an RPM value can be calculated and transformed into MPH by considering the circumference of the wheels. On a final note, the speed measurement block had a noisy ac signal show up that gave a speed value during idle conditions. This problem was solved by connecting circuit ground and earth ground (chassis) together along with a  $3600pF$  capacitor attached to the hall effect signal that removed this interference. However, we noticed noise would still remain while the PWM cycle remained non-zero on the second PCB we designed. This issue could be resolved directly so the tachometer system was replaced with the back emf system for speed measurement as described in the beginning of this section. In other words, instead of utilizing the hall effect signals, the rectifier voltage output is monitored. This way, the relationship  $V_{emf} = K_e \omega$  is straightforward to calculate the speed [rad/s] given that the motor constant is  $0.1$ . However, caution should be taken since the adc can only handle a  $5V$  maximum on the input pins. Thus, some kind of voltage divider is needed to protect the adc and read the rectifier voltage. It is not a non-trivial system since the maximum expected voltage on level 1 is  $30V$  but on level 6 it becomes  $5V$ . Therefore, a fixed-gain voltage divider will not be able to accurately read voltage values on level 6 if that system is implemented. A variable-gain voltage divider system is essential to properly read the analog signals from all workout levels. One way to achieve this is to use a two-stage voltage divider as shown in the figure below.

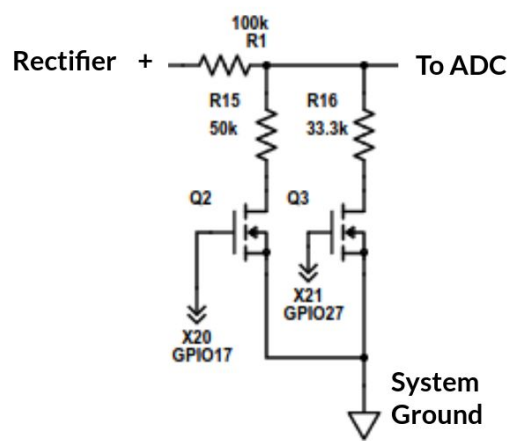


Figure 31. Rectifier voltage divider solution implemented.

From above, there are three stages to the voltage divider system. If both mosfets are off, then we have a gain of 1 which works well for level 5 and 6 since the wheelchair user will not output high voltage. If only the first one is on, then we have a gain of  $\frac{1}{2}$  which is useful for level 4 as a maximum of 10V is expected there. If only the second from the left is on, then we have a gain of  $\frac{1}{3}$  and this is useful for level 3 where it allows for 15V maximum. If both mosfets are on, then we have a gain of  $\frac{1}{6}$  meaning a 30V input is brought down to 5V which is the maximum safe input voltage for the adc. Since the resistors are high resistance resistors, there is a negligible amount of current taken away from the circuit so this should not make any difference in correctly measuring current. Even though the high resistance may be problematic for the adc, the datasheet shows that the adc can handle high input impedances since we are only sampling twice every second. If we required high sampling rates, then the high input impedance becomes problematic. In the figure below, this relationship between acceptable input impedance values and adc clock values are shown. Testing further confirms that the adc does not experience any issues or slowdowns in reading values.

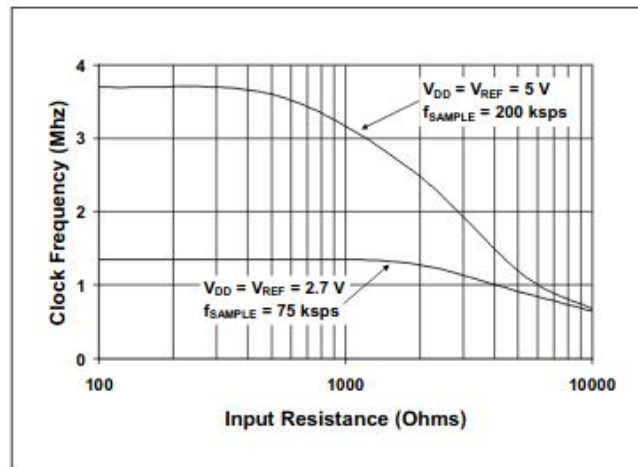


Figure 32. Maximum clock frequency compared to input resistance values.

## Analog & Digital Signals

The analog to digital converter is required since the chosen single-board computer does not contain analog pins. Since we will independently sense both motors, 6 channels are required total. Half for speed sensing and the other half for torque sensing. The tachometer is still included in the final PCB in case another possible, future team needs to utilize it. However, only four input signals are used to determine the essential information. Analyzing parts online, the following figure shows different ADCs found. It is interesting to note that the most inexpensive part is a 10-bit ADC. 8-bit ADCs that have 4 channels cost more than this part. Thus, the

MCP3008 is chosen as the ADC component. The sampling rate is relatively unimportant for this part of the project since we are only taking a sample or two every second and does not demand a high sampling rate.

Analog Digital Converter				
Component	Resolution	Sampling Rate	Channels	Price
MPC3004	10	200 kS/s	4	\$2.14
MCP3008	10	200 kS/s	8	\$2.25
MCP3204	12	100 kS/s	4	\$3.26
TLC0838	8	20 kS/s	8	\$5.06

Table 10. Comparison of different ADCs.

## Power Requirements

For our project, a power supply is required to power the tachometers, gate drivers, amplifiers, and the ADC. The mosfet gate voltage to turn on is typically 3V but can be as high as 4V from the datasheet. Thus, using 5V for the gate driver works to solve this potential problem. Since the ADC, amplifiers, and tachometers support a 5V supply, the entire block diagram will run off a 5V supply as this works well for the different components. Analyzing the power consumption for each component leads to the following table:

Power Consumption			
Component	Voltage	Current	Power
Tachometer (x2)	5V	0.28A	1.4W
Gate Driver (x1)	5V	0.1A	0.5W
Amplifiers (x2)	5V	0.02A	0.2W
ADC (x1)	5V	0.01A	0.1W
Hall Effect (x2)	5V	0.25A	0.25W
Total:	5V	0.66A	2.5W

Table 11. Power consumption of the dyno system.

As seen from the figure above, a 5V 5W power supply is sufficient for our purposes. However, a separate power supply is not required since the Raspberry pi contains a 5V pin. From documentation online, the 5V supply pin can draw a maximum of 1.5A which is above our need

of 0.42A so the Pi's 5V 3A power supply is sufficient for our purposes [21]. Therefore, there is no need to purchase a rectifier to convert AC to DC since the power supply included with the Raspberry Pi will provide enough power. On a final note, each component block contains a bypass capacitor attached from the supply pin to ground so that noise and voltage spikes are reduced in order to prevent noisy output signals [22].

## Overall System Overview

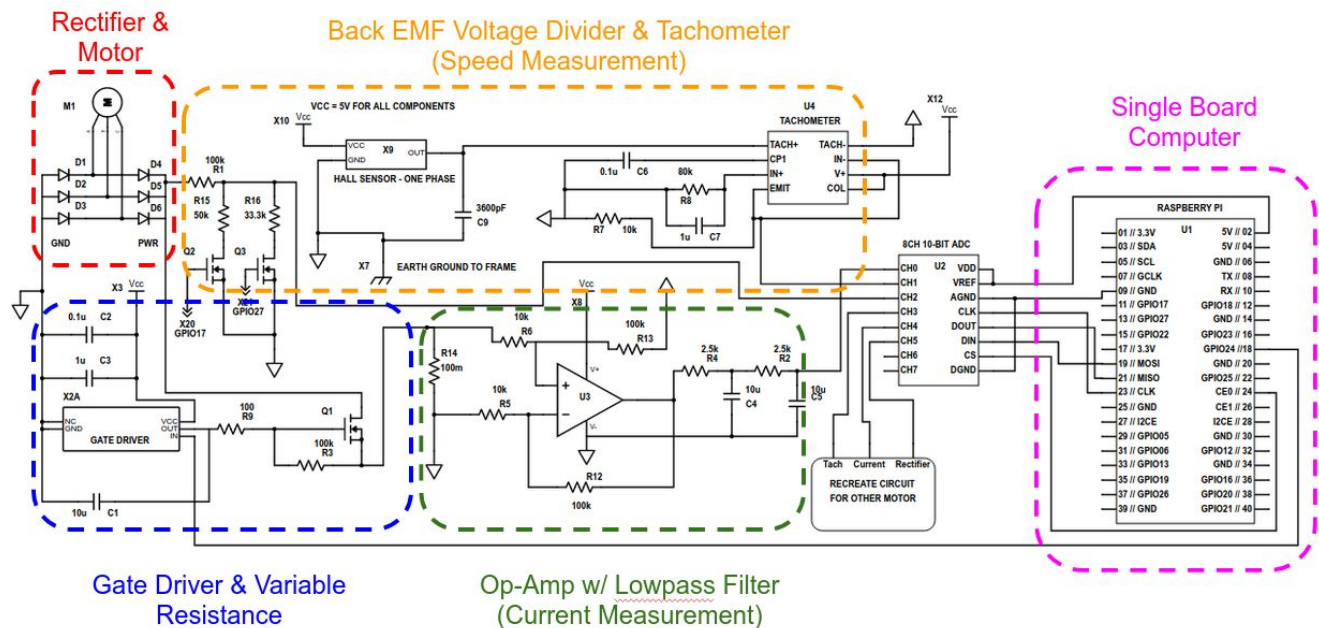
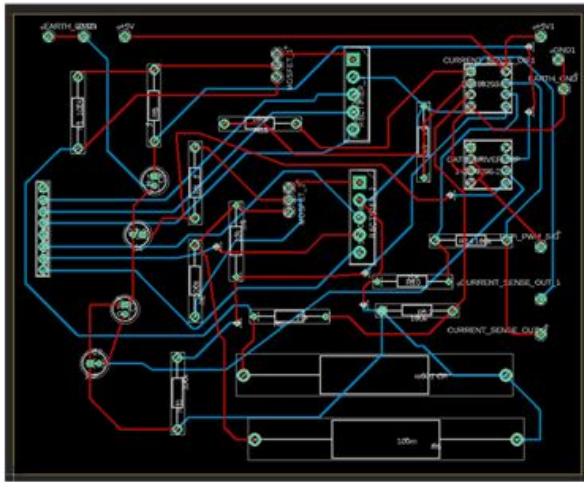


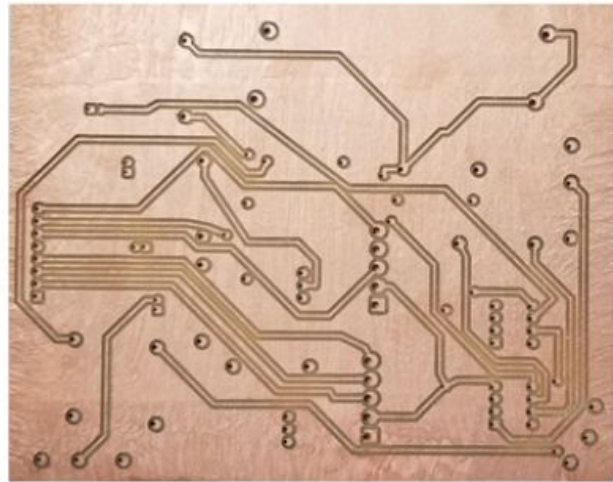
Figure 33. Final circuit diagram for wheelchair dynamometer.

The figure above illustrates the technical block diagram in a schematic form. This diagram is necessary in order to give the full details on how everything is connected together in order for anyone to replicate what was done. The dashed regions represent different blocks discussed in the development plan and the little details mentioned such as attaching circuit ground to chassis in order to remove noise. The overall system functions as expected as shown via video ([Microsoft Stream Link](#)). In the video, we performed experiments designed to test whether components were working as expected before integrating with the actual dyno system. As shown by the video, there is no reason the final design should not work since the PCB went through three iterations and everything is working how we expect it. Moreover, once that was done, the final PCB was integrated with the dyno system and software revisions were made to make sure everything worked as expected. The demo video is accessible via this link ([YouTube Link](#)) showing that the wheelchair dynamometer project requirements have been met. Continuing with the report, the design and testing for the various PCB's is discussed in the next section.

## PCB Milling & Testing



Eagle schematic



Milled PCB

Figure 34. The first board milled at UTA along with the schematic.

As shown above, the first designed PCB layout is designed and optimized in Autodesk Eagle. Using Othermill Pro, the PCB is milled from the UTA makerspace. Due to size limitations and double layer limitations from the makerspace, the entire system diagram needs to be milled on two separate boards. The PCB is tested and achieves almost all functionality with the exception of noise interference in difficulty levels 2-5. In other words, we are detecting speeds from both motors during idle conditions. This is most likely caused by the subtractive manufacturing process used by the mill that leaves the copper plane once the traces are complete. It is problematic because this plane can cause parasitics such as a small capacitance value between separated traces. Therefore, it is likely that the PWM traces are coupling to the speed detection traces since the speed detection block and the variable resistance block are not connected together in any way. Other than this issue, the milled board works as expected. However, a drawback is that we did not use any connectors in the board as everything is connected via male pin headers including information from the Pi. We then started focusing on optimizing the PCB design (including connectors) so that it can be professionally printed via an additive manufacturing process so that there are no parasitic issues but also because our sponsor is looking to get a professional board done for this project.



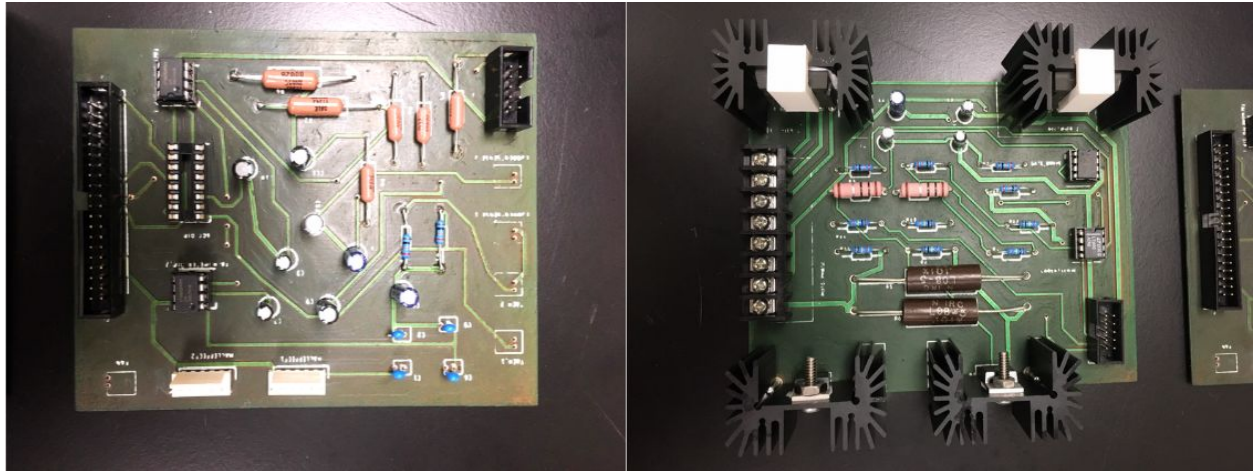


Figure 35. The second board milled at UTA using all available resources.

For the second board milled at UTA, we made some improvements over the first design. First, the leftover copper was removed so that there would not be any parasitic issues and thus no noisy signals although we found out that this was still the case hence the design shift. Secondly, connectors were added to the system to make it easier to use. For example, the Pi is now connected via a 40-pin ribbon cable to the connector on the left. The two boards are connected via the 2x5 IDC small connectors. The motor phase signals are connected via the terminal block that is secured with screws. This is ideal because we expect high current from the motors. However, we experienced some issues with the boards. The boards are warped so they are not flat surfaces making it a bit difficult to push some components through the board. During the electrolysis bath, the board is coated in copper to cover holes and vias. However, it looks like something went wrong during this process since we had continuity issues with the board. With a multimeter, the traces were checked and we noticed that some traces did not complete so we had to remake connections with speaker wire in order to fix these issues. This is not a clean solution since the wires are at the bottom of the board and it is not professional. Despite these issues, the board worked fairly well. Every feature functioned except for the speed sensing. The pwm switching noise is still coupling to the tachometer for a reason we were not able to identify. The hypothesis that the leftover copper caused parasitics was not a correct hypothesis. Looks like the problem is more complicated than the initial hypothesis. This situation made us investigate other possible solutions to gathering speed. It is under these conditions that we switched the tachometer system that utilizes the hall effect signals to the multi-stage voltage divider that keeps track of the back EMF from the rectifier. This change is made for the final PCB. However, the tachometer is still included in the final PCB in case another future team decides to utilize them for another application.

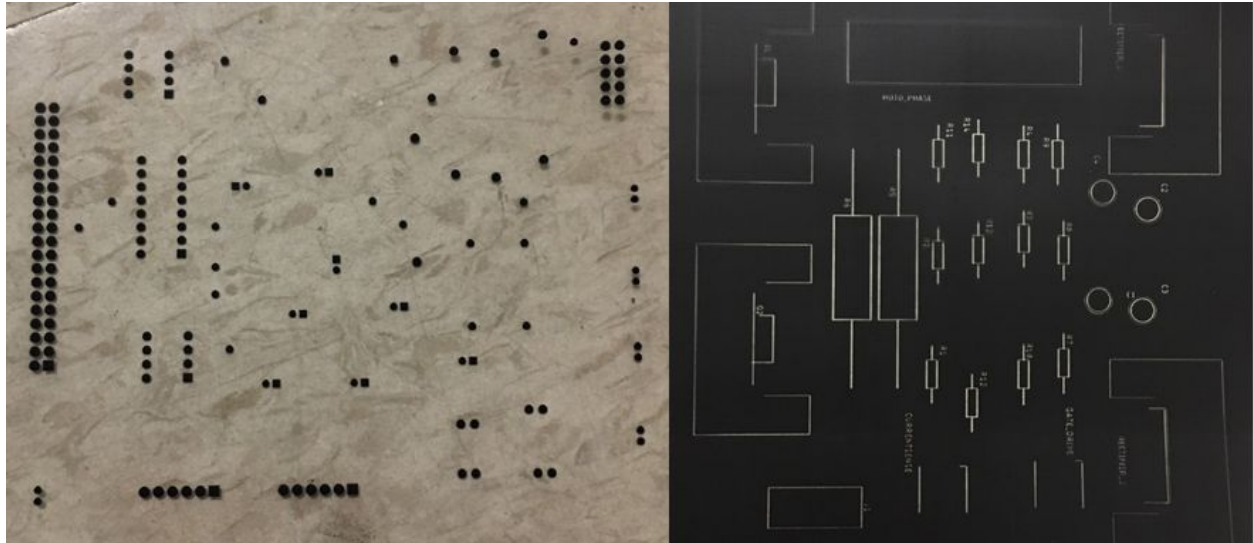


Figure 36. Transparency layers used for the second board solder mask application.

From the figure above, these are the transparency layers that were printed in order to apply the solder mask and silkscreen to our second milled board at UTA. We had the opportunity to apply the solder mask ourselves at the manufacturing lab at Nedderman and learn more about how the full process works. We recorded our experience milling boards at UTA along with a tutorial on how to do so in the appendix section of the report. This is done so that future students of senior design can reference our tutorial in the appendix section to learn more about how to go from Eagle schematic design, to optimized Eagle board, to milling, and post processing. After this second board, we designed and professionally ordered our final PCB as shown below.

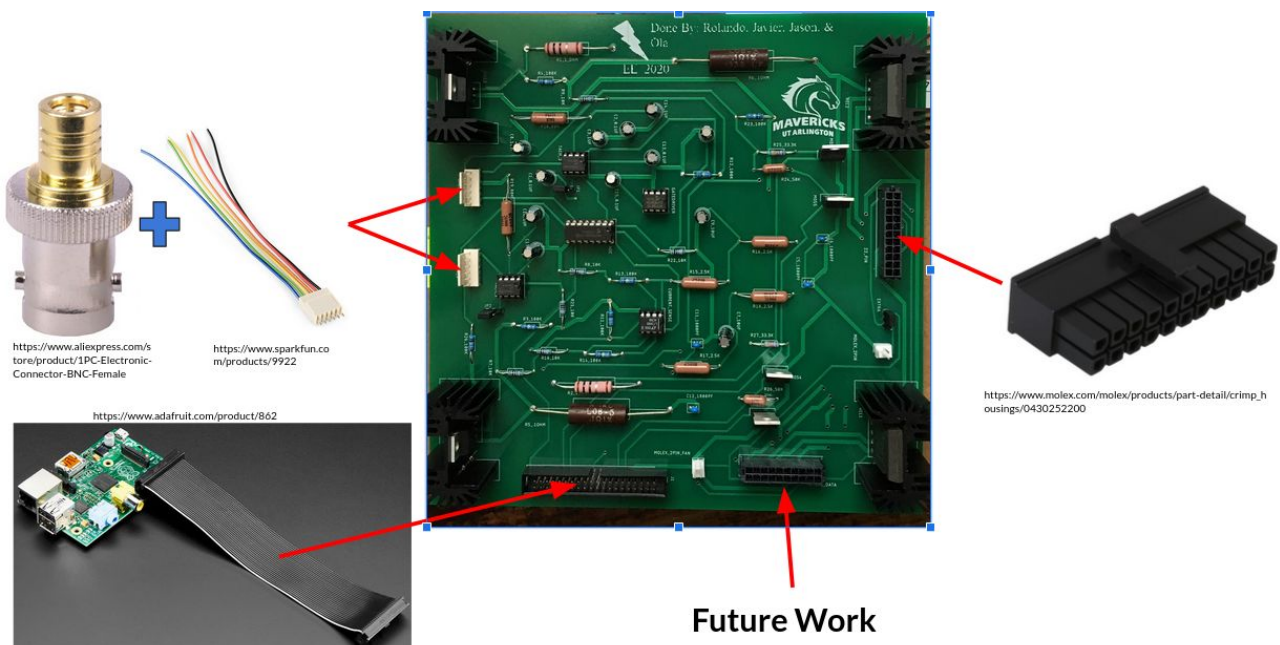


Figure 37. Final PCB professionally etched.



From the figure above, we have the final PCB that was professionally milled by PCBWay. We have the Raspberry Pi connector kept from the second milled board but all other connectors have essentially been replaced. There is a single 22-pin Molex connector that contains all signals coming from the dyno system. This included the motor phases, all the hall effect signals, earth ground, a power supply to power the internal components, and everything else. Therefore, there is only one connector that can be removed without causing damage to the board. This is because the main connector from the dyno attaches to the side of the enclosure so there is strain-relief from the main board. There are 6-pin Molex connectors included for bnc connectors. This is a request made from our kinesiology sponsors so that they can use data acquisition devices. There will be four total bnc connectors which include 2 for speed and 2 for current. Thus, these molex connectors connect the bnc connectors that are attached from the side of the enclosure to the PCB itself. We have a 2-pin molex connector to connect the system fan in order to cool down the Raspberry Pi. Finally, there is a 20-pin molex connector called 'future work' that serves an important purpose. This connector is useful for any future students that may work on the wheelchair dynamometer system. The connector has all of the hall effect signals, the motor phases, and the rectifier output. This is useful in case a team will work on the natural rolling feature of the dyno system. They will have to design their own motor controller and tap into the PCB designed this year. The only possible change they would have to make to the final pcb included is to remove the rectifier since they will implement this feature in a different way. However, having this connector will future-proof this board from any changes that may need to be made. Even though the natural rolling feature that removes friction between wheels and rollers was not implemented, some initial work was done regarding this topic. The following section describes some preliminary work done in this topic that can be continued by a future team.

## Natural Rolling Feature - Preliminary Work

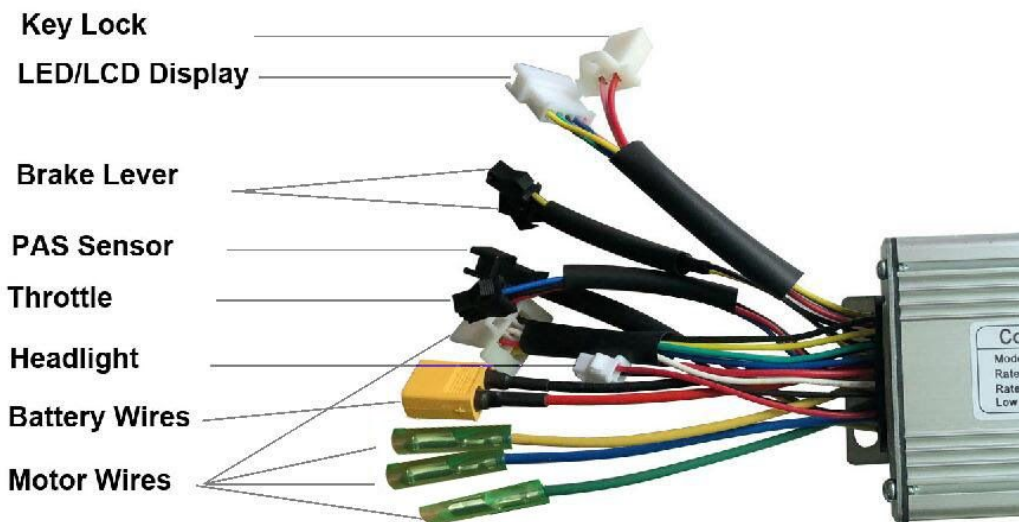


Figure 38. Motor controller layout.

In the figure above, we have the labeled wire diagram for the included motor controllers that came with the motors. We were not able to find the exact model online so the picture is a similar one so there is slight variation. There is a key lock that should be shorted together in order for the motors to turn kind of like an ignition lock. The LCD connector is to view speed and distance information as well as changing speed based on the buttons they provide. The throttle is a potentiometer that can set the amount of power to provide to the motors. The pedal assist sensor (PAS) is a feature that may be relevant for our project since it will assist in 'pedaling' which is basically how adding a natural rolling feature would work. The brake level will help stop the motors. Out of the two motor controllers that we have, only one of them seems to be working. An experiment was conducted to get the motors to spin using this controller. However, we did manage to get the motors to spin but only at a low speed. The throttle feature appears to be a binary on/off signal and not a potentiometer that we were expecting. More testing is required to get conclusive results. Multiple attempts were made to reach the company responsible for the motor controllers to get more information about them so we could use them properly (KT Controllers). We received no replies. We are not aware if the motor controllers have regenerative braking. This is important because the wheelchair user will generate power when they use the dyno so the power has to go somewhere. There is no information about whether or not it goes back to the battery. Without a datasheet, it is difficult to figure out how to implement this controller to our existing system. It may be as simple as connecting the motor phase with the rectifier in our circuit or as complicated as removing the rectifier and implementing an alternative solution to make everything work with the final pcb. Because of these issues, work on this topic halted on the hardware aspect.

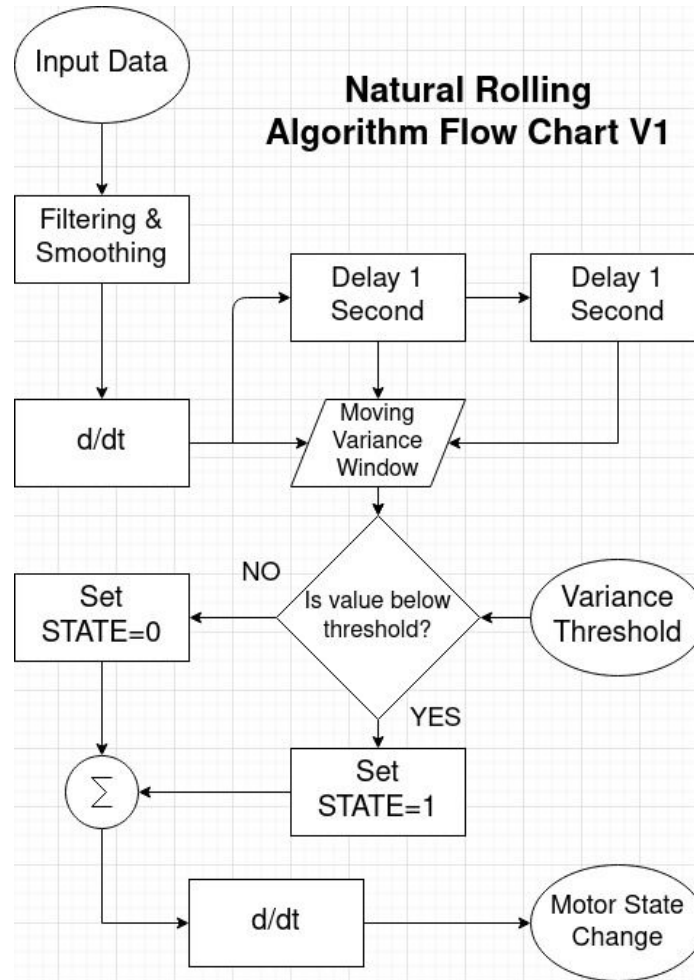


Figure 39. Motor controller algorithm flowchart.

The figure above contains the algorithm flowchart so that the computer can activate the motor controller when needed. In order to explain what is going on, the following figures better illustrate what different blocks are doing in the software flowchart. In essence, the system is designed for a worst case scenario so this implies that there is a significant time delay in regards to motor controller startup. Therefore, the motor should only activate once we know for sure that the user is done rolling. This may not be a fair assumption but if that is so, then the software flowchart can be simplified by only observing the derivative to determine when to turn on and off. The software is designed for a worst case scenario so we will look at a more advanced algorithm in the figures below.

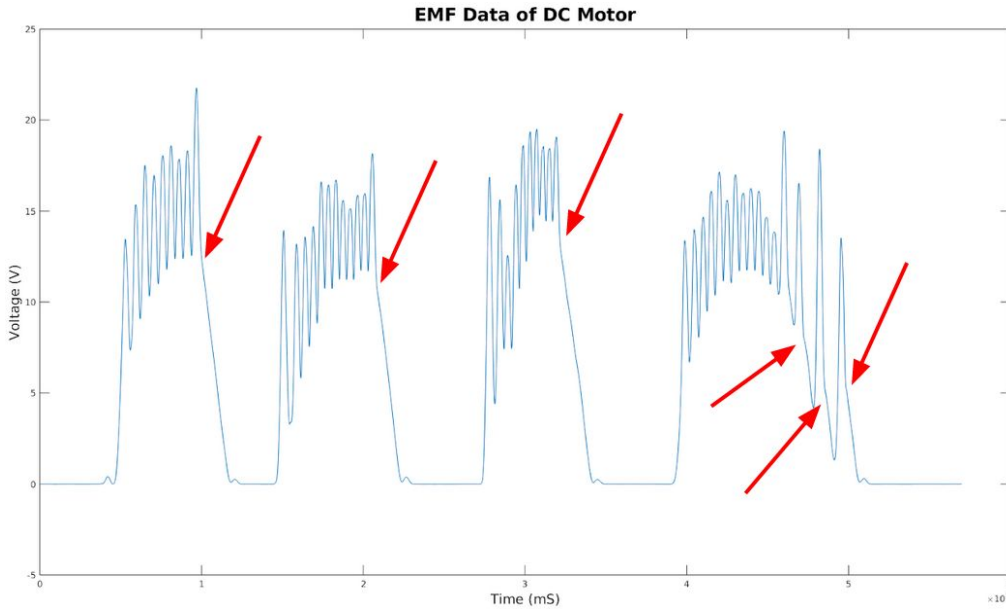


Figure 40. Motor data from the rectifier.

The figure above shows rectifier voltage from the dyno as a person uses the dyno and rolls. The data is collected using the myDAQ with a high sampling rate. The way the algorithm is implemented is by assuming a large motor controller switching time so kind of a worst case scenario. What this means is that while the user is rolling and going back to starting position to roll again, the motor controller will not activate. This assumption may be wrong but that can be solved by simplifying the algorithm so a future team can take the code and flowchart and work from there. The red arrows above illustrate the regions of interest. This is when the motor controller should turn on as the user will most likely not roll again.

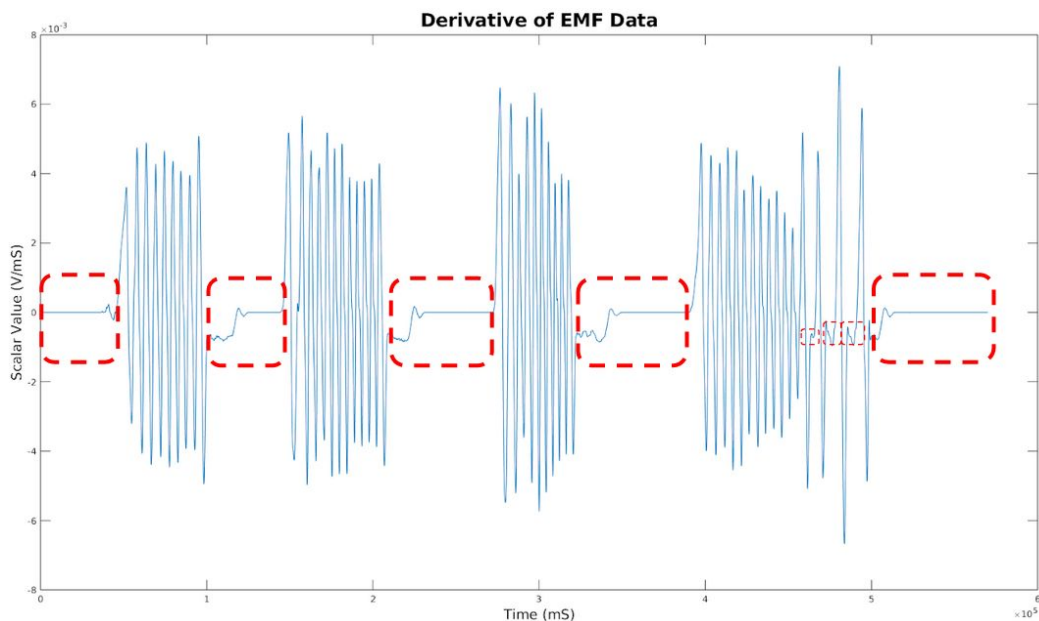


Figure 41. Derivative of previous figure.

If we take the derivative from the previous figure, we get the figure above. This is not surprising since the wheelchair user will have high oscillation speed while using the dyno and a low speed when slowing down. Thus, the red boxes represent when the motor controller should kick on. Since the oscillation sections also have the same magnitude as the red boxed region, the answer is not as simple as thresholding. What must be done is to take a variance window of this figure.

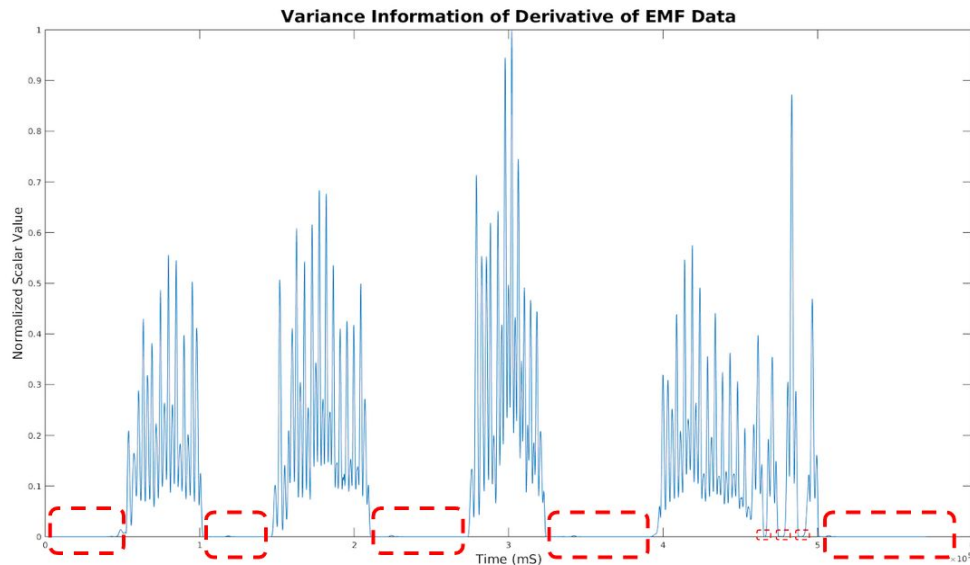


Figure 42. Sliding variance window of previous figure.

Taking a variance window of the previous figure will generate the figure above. What we see is that when the user is actively using the dyno then we have regions of high variance. When the dyno is not rolling, then we have areas of low variance. An important note to make is that all of the results were generated from MATLAB. Caution was taken to ensure that only previous data affects current data. That is, future data does impact current data. The sliding window operation is only looking at past values. This is an important point to make to ensure we have a real-time system. For this sliding window, there is a parameter that specifies how many points to use while the window moves. This is a parameter that was optimized using a heuristic approach. This parameter is robust to new data as seen in the last figure. More on this topic later.

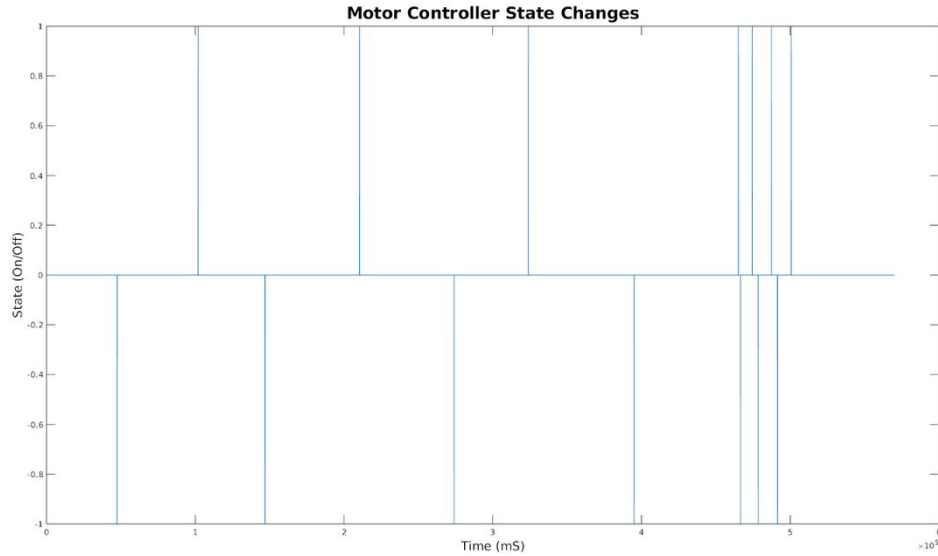


Figure 43. Thresholding the previous figure and taking the derivative.

If we threshold the previous figure, then we will get the figure above. Since thresholding will generate rectangular pulses, then taking a derivative of this signal will generate the correct graph. What is illustrated is only when the change occurs. Of course, there is a parameter here to set how much variance is needed to be considered 'in use' and anything else to be considered 'not in use'. This parameter was also optimized using a heuristic approach. The exact values can be found in the code section of the deliverables.

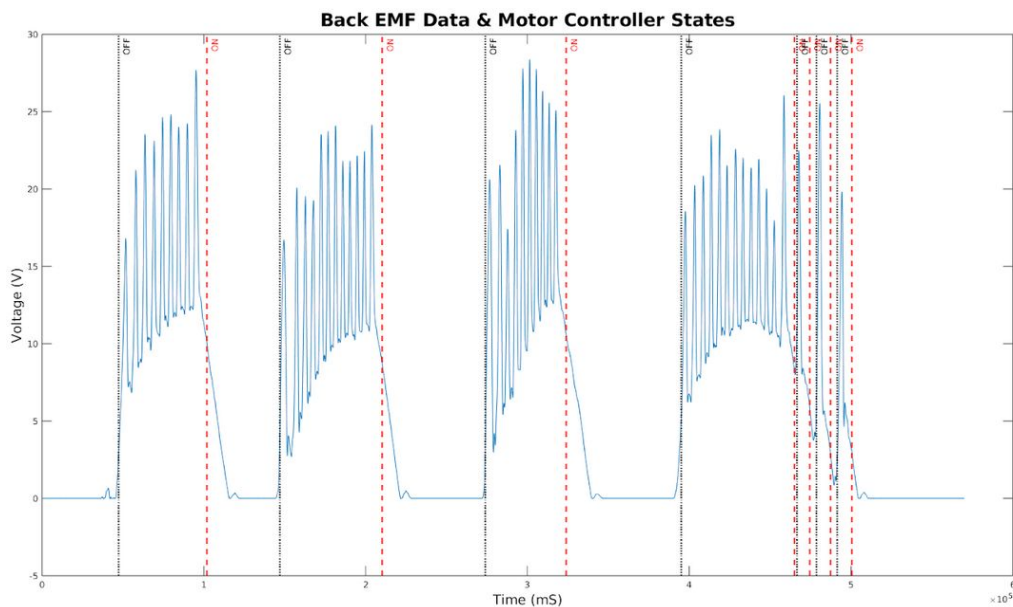


Figure 44. Overlaid previous figure with original data.

Overlaying the previous figure with the original motor voltage graph will result in the figure above. We can see good results in that the motor controller turns on early enough so that it kicks in before the user comes to a complete stop. An important question to ask is whether the

algorithm is robust enough to work on more data. The figure below shows the same algorithm with the same optimized parameters applied. Notice that the results are similar to the previous figure meaning that the algorithm is robust enough to work in different situations. As mentioned before, this algorithm was made under certain assumptions that may not necessarily be true. However, a future team can take this code and develop the algorithm further to fit their needs.

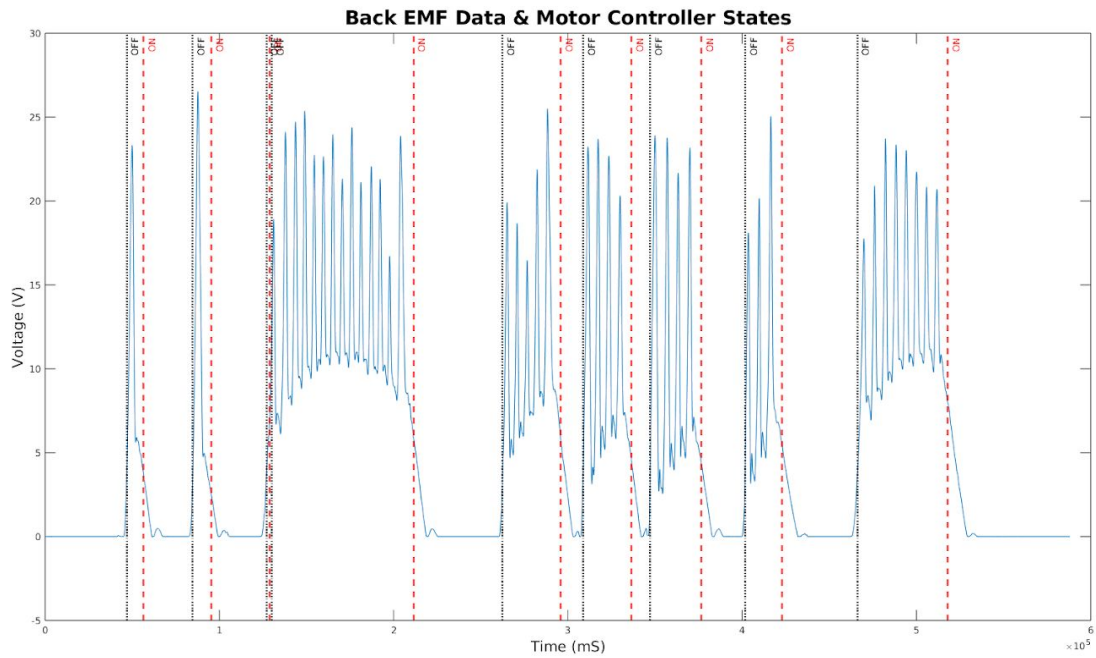


Figure 45. Algorithm applied to a different dataset.



## Enclosure & Final PCB

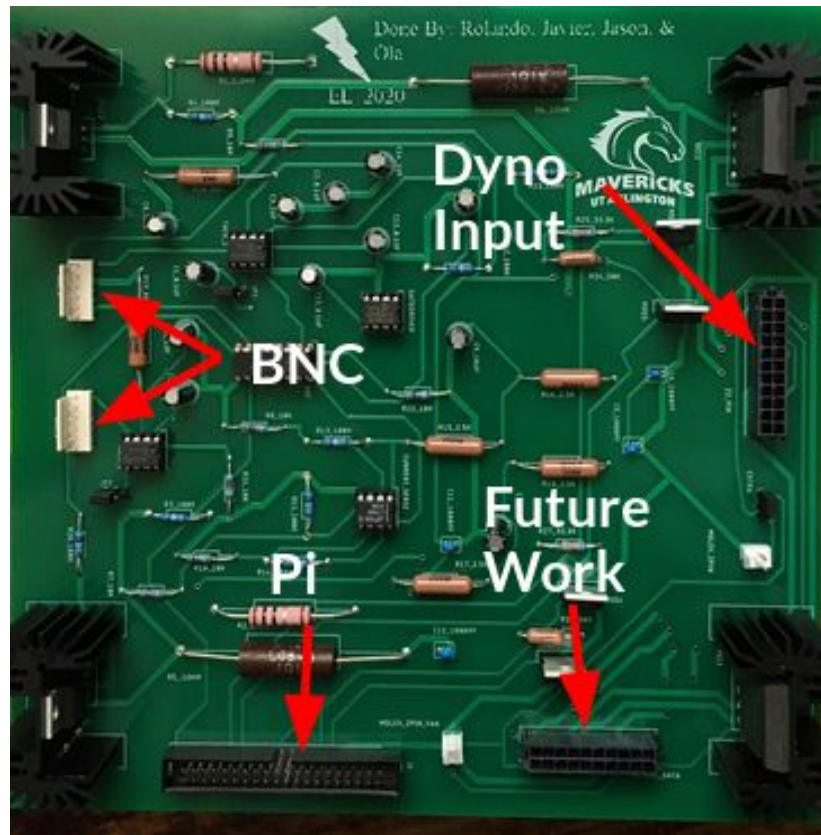


Figure 46. Final etched PCB.

The final professionally etched PCB is a double-layer 2 oz copper FR4 board etched by PCBWay. As seen from the figure above, the Raspberry Pi has an easy connection system that uses a ribbon cable. The BNC connectors for data acquisition are shown above to use the 6-pin Molex connector. The input from the dyno motors, hall effect signals, and power supply are all a single connector that connects to the front of the enclosure via a 22-pin Molex connector. The future work connector has the rectifier and motor phase signals in combination with the hall effect signals in case some possible future team wants to implement the natural rolling feature. This way, the board we currently have would not be outdated since the connector can go off to another board that handles motor controller integration.

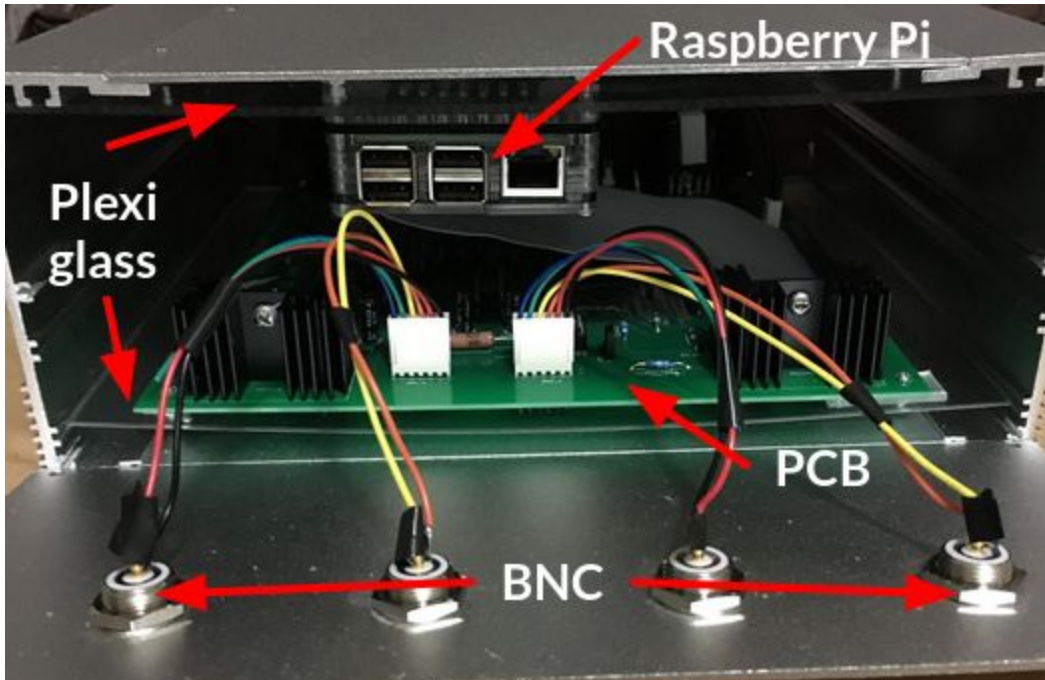


Figure 47. Side view of the PCB within the aluminum enclosure.

The PCB as well as the Raspberry Pi are mounted on plexiglass since the enclosure is larger than the board. This way, the plexiglass slides on the internal grooves of the aluminum enclosure and the PCB is safely mounted on top of it. The same is done for the Raspberry Pi except it's flipped upside down and a ribbon cable joins the connection between the board and the computer.

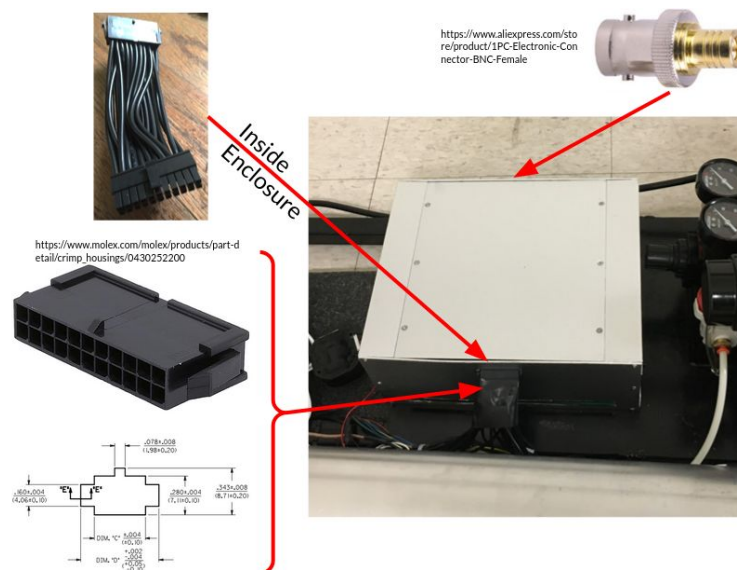


Figure 48. Enclosure system integrated with wheelchair dynamometer.

In the figure above, the enclosure is shown with the 22-pin Molex connector connected to the side of the enclosure. All of the pins on the connector are described in the ‘User’s Guide’ section of the report. From above, there is an internal connector that attaches from the PCB to the side of the enclosure to provide stress relief to the board. At the back of the enclosure, we have the BNC connections that can be used for data acquisition devices that track speed and torque for each motor. More information about this is provided in the ‘User’s Guide’ section of the report. The enclosure remains to be mounted to the frame of the dyno but is otherwise complete. The wires leading up to the Molex connector are cleaned up and secured to the frame of the dyno to have a cleaner area near the enclosure.

## Budget

Each part explained earlier in the report needs to be taken into consideration for the budget. The table below gives the total cost per component that will be needed for the electrical system. Note that some of the following components are provided by one of our sponsors such as the Raspberry Pi 3 but have been included in the budget list anyway in order to have a complete list of the electronic components for the dyno system. This also includes non-essential items like the enclosure and the entire PCB order from PCBWay (5 board minimum).

Dynamometer Budget List					
Part Name	Supplier	Part Number	Quantity	Cost	Type
Mosfets	Mouser	726-IPP12CN10LG	6	\$10.02	Through-Hole
Pin Headers (x20)	Mouser	855-M20-9993045	1	\$1.04	Through-Hole
Gate Driver	Mouser	579-MCP14E10-E/P	1	\$4.23	DIP
Frequency Voltage Converter	Mouser	926-LM2917N-8/NOPB	2	\$4.26	DIP
Operational Amplifier	Mouser	584-LT1490ACN8#PBF	1	\$5.48	DIP
Current Sense Resistor	Mouser	66-LOB5R100FLF	2	\$3.08	Through-Hole
DIP Socket (16P)	Mouser	571-1-2199298-4	1	\$0.23	DIP
DIP Socket (8P)	Mouser	571-1-2199298-2	4	\$0.88	DIP
Heat Sink	Mouser	588-RA-T2X-25E	4	\$7.96	NA
Thermal Paste	Mouser	532-249	1	\$9.38	NA
Heat Shrink Tubing	Mouser	650-829131P019	25	\$2.60	NA

0.1uF Capacitor	Mouser	647-UKL1H0R1KDDANA	4	\$1.44	Radial
1uF Capacitor	Mouser	647-UKL1H010KDDANA	4	\$0.40	Radial
10uF Capacitor	Mouser	647-UKL1C100KDD1TD	4	\$1.20	Radial
1800pF Capacitor	Mouser	810-FG28C0G1H182JNT0	4	\$0.46	Radial
100K Resistor	Mouser	660-MF1/4LCT52R104G	6	\$0.60	Through-Hole
10K Resistor	Mouser	660-MF1/4LCT52R103G	8	\$0.80	Through-Hole
80K Resistor	Mouser	71-RN65C8002B	2	\$1.35	Through-Hole
2.5K Resistor	Mouser	71-RN60D2501F	4	\$1.44	Through-Hole
0.1K Resistor	Mouser	603-FMP300FRF73-1R	2	\$1.18	Through-Hole
3-Phase Rectifier	Mouser	747-FUS45-0045B	2	\$13.66	Through-Hole
Ring-Tongue Connector	Mouser	571-696424-7	2	\$2.64	NA
Analog/Digital Converter	Mouser	579-MCP3008-I/P	1	\$2.19	DIP
Raspberry Pi 3 Model B+	Amazon	B07BC6WH7V	1	\$64.99	NA
Raspberry Case w/ fan	Amazon	B07BTHNW9W	1	\$15.99	NA
Molex 2-Pin PCB	Mouser	538-22-23-2021	2	\$0.34	NA
Jumpers	Mouser	474-PRT-09044	3	\$1.05	NA
Pi Ribbon Cable	Mouser	474-CAB-13028	1	\$2.50	NA
Pi PCB Header	Mouser	474-PRT-13054	1	\$0.95	NA
Molex 22-pin Male	Mouser	538-43025-2200	2	\$2.74	NA
Molex 22-pin Female	Mouser	538-43020-2200	1	\$2.50	NA
Molex PCB Header 22-Pin	Mouser	538-43045-2212	1	\$4.42	NA
Female Crimp Terminals	Mouser	538-43030-0007	65	\$9.50	NA
Male Crimp Terminals	Mouser	538-43031-0007	50	\$8.45	NA
Molex 6-Pin Assembly	Mouser	474-PRT-09922	2	\$3.90	NA
50K Resistor	Mouser	71-RN60C5002D	2	\$0.82	NA

33.3K Resistor	Mouser	660-MF1/4CCT52R3322F	2	\$0.38	NA
20-pin PCB Header	Mouser	538-43045-2012	1	\$3.55	NA
20AWG Wire	Mouser	510-CT2879-0-10	1	\$13.65	NA
Molex 2-pin Housing	Mouser	538-22-01-3027	2	\$0.24	NA
Molex 2-pin Crimps	Mouser	538-08-50-0114	4	\$0.68	NA
Enclosure	Amazon	B07VRKD183	1	\$29.99	NA
32GB MicroSD	Amazon	B073JWXGNT	1	\$7.99	NA
Final PCB	PCBWay	NA	1	\$120.00	NA

Table 12. Complete budget list for electrical system in the wheelchair dynamometer.

## Conclusion

The wheelchair dynamometer is an innovative product initially intended to track the performance of athletic wheelchair users. Nevertheless, the extent of its use was expanded to further allow non-athlete wheelchair users to workout for general health and fitness purposes.

The complete project is subdivided into four different parts:

1. Have a means to vary the difficulty of workout levels to accommodate all users.
2. Track the torque and speed information, along with other parameters derived from this, created by the wheelchair user.
3. Display information via a display with an easy-to-use user interface and record workout data so that past workout information can be accessed.
4. Allow the wheelchair user to experience life-like rolling on the rollers.

Parts one, two, and three of the project are complete. Part four may be assigned to an upcoming senior design team if our sponsor Dr. Woods feels that this is a necessary feature for the dyno to have.

Part one consists of setting different workout levels for the wheelchair user while they are on the machine. This was accomplished by utilizing a transistor as a fast-changing switch that will simulate a resistance across the motor terminals. The resistance levels are set from 1 to 6. Level 1 being the easiest for the user to spin the rotors and level 6 is the hardest. Level 1 is equivalent

to setting a 100-ohm shunt resistance. In contrast, level 6 is equivalent to having zero resistance, or a short across the terminals of the motors, and thus making it very difficult to move the rotors. The circuit design was completed and the components were obtained. The professionally etched board was populated and tested to confirm that every aspect functions as expected. Moreover, the demo video listed earlier in the report shows the completion of the project requirements.

Part two and three consists of being able to track and display the performance of wheelchair users while they are exercising. Both torque and speed, along with derived parameters, must be recorded and shown on an LCD screen which is attached to the frame of the dynamometer. Torque is obtained by utilizing a low-side current sense amplifier. The voltage drop across the resistor is amplified so that the ADC can read and calculate the torque. Originally, the speed was calculated by the tachometer setup as explained earlier in the report. However, the last design change involved utilizing a multi-stage voltage divider configuration that feeds the rectifier voltage to the ADC. In essence, we are utilizing the back emf of the motors since this is directly related to speed information. All of the information is processed by the Raspberry Pi 3, a single board computer capable of hosting a web server. The computer creates a wifi network that any device can connect to it. This will subsequently display all the important data that the wheelchair user is interested in seeing in terms of their performance. The LCD has multi-touch support and the user is able to interact with it through the GUI as shown earlier in the report. Moreover, the user can select the workout level either with the control knob or the button layout adjacent to the control knob.

Part four consists of extending usability by simulating a stroll down a flat path or even going down a slope. That is, whenever the user applies force to the wheels of the chair, momentum can be simulated and the motors will continue to spin and slowly come to a stop. In order to achieve this, a separate circuit will have to be designed and integrated into the system. In this application, the system will have to sense the force applied and detect changes in force to provide some power to the motors that will cause the motors to keep spinning and then gradually come to a halt. This part of the project may be assigned to a possible future senior design team. Some preliminary work has been done on this topic as shown earlier in the report.

As of today, there are not many wheelchair-specific workout machines available at gymnasiums and fitness centers. Therefore, the wheelchair dynamometer will greatly improve the quality of life of a large demographic who had previously been limited in options to routinely exercise at a fitness center. Moreover, this wheelchair dynamometer can also be used in rehabilitation centers and hospitals for other use cases besides tracking sport performance.

# Appendix

## PCB Manufacturing at UTA

### Schematic Process

Once the design of the circuit was complete and tested on a breadboard, a schematic for a printed circuit board (PCB) was needed. Eagle Autodesk is a program that allows a user to create a PCB layout with several tools that help make routing mostly effortless. First, one must acquire the Eagle Autodesk program, there is a student version which is free but does have some size restrictions on it. To begin the process, start by creating a new project which allows for all the files to be saved to that specific folder. This can be done by selecting the file menu, new, and finally project tab.

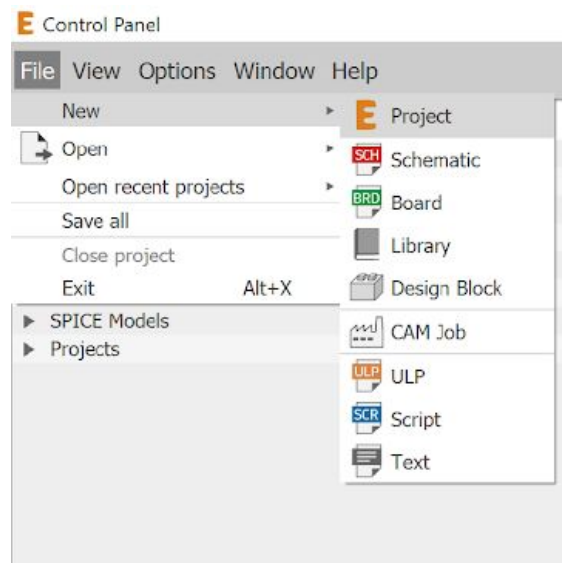



Figure A1. Eagle Autodesk control panel view.

As the above figure shows, the process of the Autodesk project creation is displayed. Once selected, name the folder as pleased, and then right click on the project created. Select the new schematic and the schematic screen will now be visible. You will be able to start adding the parts that are needed. The easiest method to acquire parts is to download universal libraries which carry common resistors, capacitors, headers, and many other common components. Although this method is the easiest way to acquire the part that is needed, another method can be done to ensure the proper component is used. By collecting all the parts required from the Mouser website, it allows the user to acquire the footprint of that component directly from their website. Mouser is a great source for parts needed and also gives the option to download the specific ECAD model for the component needed.




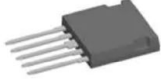


All ▾
Part # / Keyword

Products ▾
Manufacturers
Services & Tools
Technical Resources
Help

All Products > Semiconductors > Discrete Semiconductors > Diodes & Rectifiers > Bridge Rectifiers

### FUS45-0045B

Enlarge

Images are for reference only  
See Product Specifications

Share



Mouser #:	747-FUS45-0045B
Mfr. #:	FUS45-0045B
Mfr.:	IXYS
Customer #:	<input type="text" value="Customer #"/>
Description:	Bridge Rectifiers 45 Amps 45V
Datasheet:	<a href="#">FUS45-0045B Datasheet</a>
ECAD Model:	 PCB Footprint and Symbol


Figure A2. Mouser part page.

As you can see from Figure A2, right beneath the datasheet link is the ECAD model file. This ensures that the right schematic footprint and symbol is being used for the component as far as dimensions and measurements go. Sometimes Mouser does not have the provided ECAD file for a specific component, but another option to acquire that file is by going to componentsearchengine.com. This site is partnered with all the common component sellers such as Mouser and Digi-Key which allows the user to do a search by component category. Simply acquire the Mouser part number from the component needed and search for it using the service.


componentsearchengine.com/search.html?searchString=747-FUS45-0045B&country=GB&language=en&source=1048575

SYMBOLS | FOOTPRINTS | 3D MODELS





Many distributors make claims  
We simply have the largest selection  
of products in stock



ORDER  
WITH  
CONFIDENCE

Components matching your search: 747-FUS45-0045B





Image	ECAD Model ↓	3D	D.S.	Description	Manufacturer	Compare Prices / Stock
	 C4			Bridge Rectifiers 45 Amps 45V	IXYS SEMICONDUCTOR	<a href="#">FUS45-0045B</a>
<div>  In Stock   5.29 GBP <a href="#">BUY NOW</a> </div>						

Figure A3. Component search on componentsearchengine.com.

As the above figure shows, the rectifier was searched with its Mouser part number and was quickly found. If the file is not available, it is possible to request that the file be built according to datasheet specifications. The turnaround time for this option is 48 hours so relatively quick. Moreover, the user can also design it manually by following tutorials found online. The files can

then be downloaded and used. After finding all the parts needed for the circuit, the files must be uploaded to Eagle library. This is done through the library which then allows the user to select 'open library manager'. Once opened, select the available tab and select browse, which allows the user to search the computer for the saved component files.

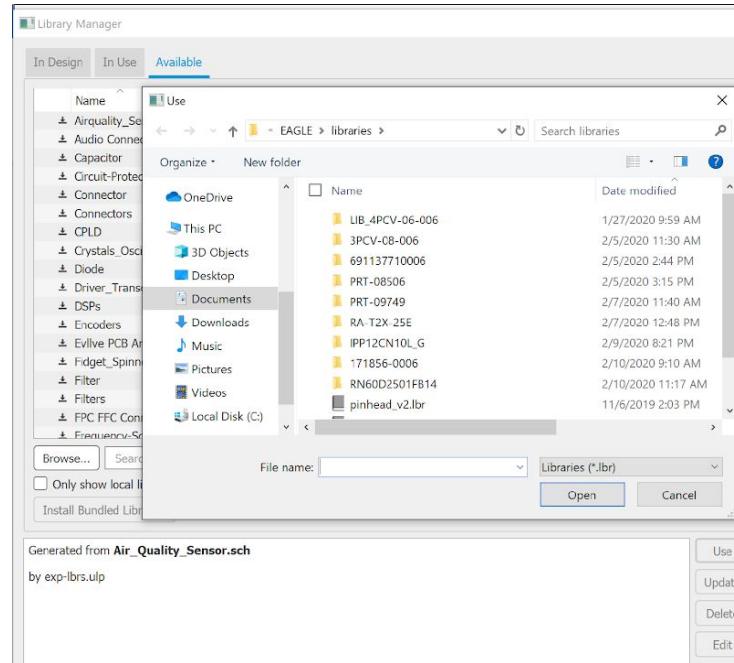


Figure A4. Uploading library files.

Next, select the folder required, and select the Eagle file within that folder. Select the component and select use to ensure the part is added to the Eagle library database. Once the files have been uploaded, the component can now be added to the schematic process. Eagle has several commands on the left side of the screen such as: group, move, mirror, rotate, copy, add part, net, junction, name part, value, etc...

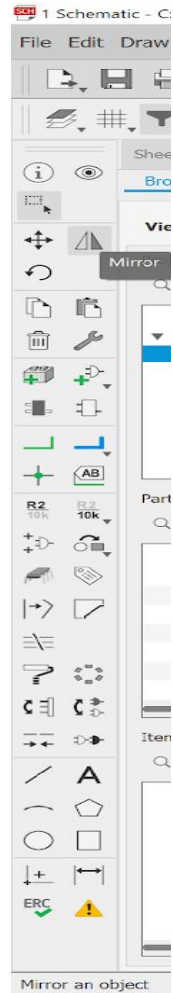


Figure A5. Eagle command selection.

Figure A5 shows the commands on the left side of the viewing screen. These are the main commands that allow the user to interact and control the schematic process. First the add part command is needed to add the component to the screen. The user will navigate through the component selection and acquire the one needed.

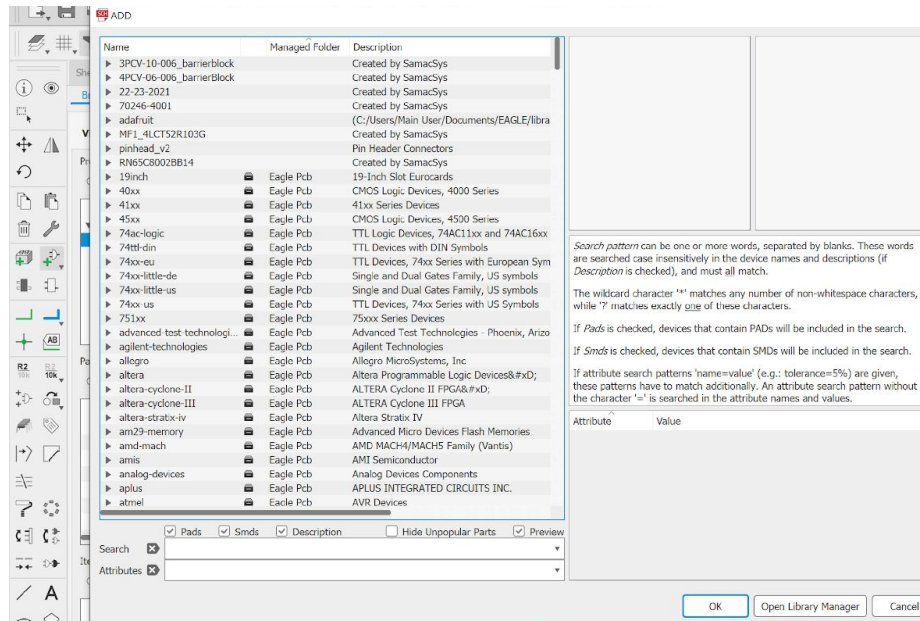


Figure A7. Adding a component to schematic.

As the figure above shows, the user will see the specific components they added at the top and the description reads “created by SamacSys”, which is the componentsearchengine.com organization. Once all the components are added, the user will now do the necessary connections between parts by selecting the NET command.

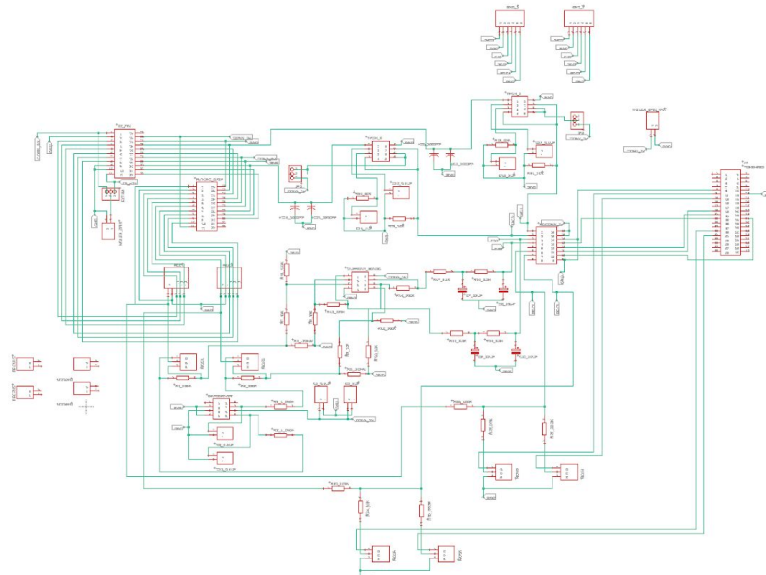


Figure A8.Final schematic view.

Figure A8 shows a fully built schematic once all the connections have been completed. The green lines are the nets (traces or tracks) that connect components to other components as well as all the ground and power connections. Once the user has completed the schematic design, they should revise it several times to ensure that no errors were made. Errors within this part of the

schematic can be critical to finalizing a PCB so ensure that there are no mistakes within this part of the process.

## Converting Schematics to Physical Boards

Once the schematic process has been completed, it is time to start the board layout process. At the top of the schematic view, there is an option command that reads “Generate/Switch to board” which allows the user to create a board file. If a board file has already been saved, then that same command will allow the user to go from schematic to board view and vice versa.

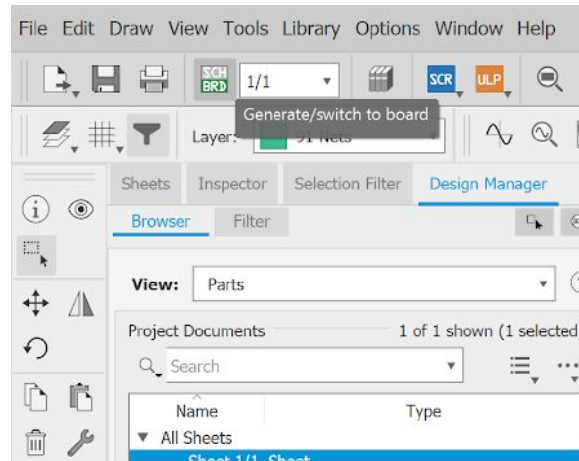


Figure A9. Eagle “Generate/Switch to board” option.

Figure A9 shows the command selection to generate the board file. After the board file is generated, all the component symbols will be on the left side of the screen. The middle square which is outlined will be the board that needs to be populated with the components. A quick suggestion would be to select the dimension command to allow the user to see what size the board will be. In order to get the components within the board, the user needs to drag each symbol(s) to the provided space. Place the components accordingly with enough space between them to ensure proper trace space. Generally, each trace should have at least 10 mil of spacing between them but this depends on applications. Also the edge of the board should have a 10 mil spacing from edge to component to ensure no trace is damaged when the board is cut out.

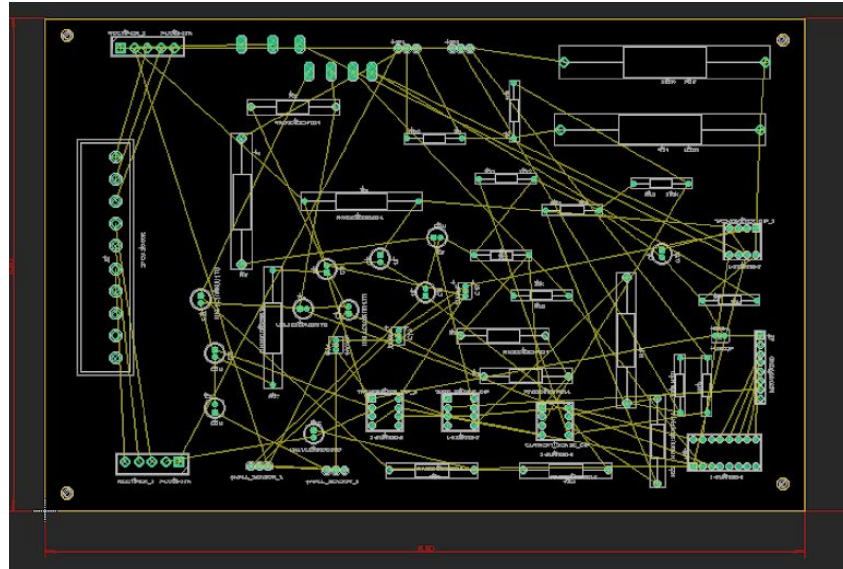


Figure 10A. Component placed on board spacing.

Figure 10A is an example of the components already placed on the spacing provided for the board. A general idea can be seen as to how the PCB will look once it has been milled. Once all components have been placed, there are two options that can be chosen to route each trace on the board. The first option is to route each trace one by one, or the second option is to select the auto router command. This auto router command routes each trace to its best location on the board. Here is where you choose the amount of layers that you want on the board. The more layers the easier the traces will route. The fewer layers, depending on the amount of components, the harder the auto router will work to route the traces to ensure none are touching or crossing. However, you control the auto router. Don't let the auto router control you. Verification and manual touch-up of traces after using the auto router tool is recommended. The width of the trace is very important because it ensures the proper amount of current will flow through the board, whether it be a signal or a power trace. Knowing the amount of current your circuit will see is very important because it allows the user to choose the correct width size for the traces.

Current/A	Track Width(mil)	Track Width(mm)
1	10	0.25
2	30	0.76
3	50	1.27
4	80	2.03
5	110	2.79
6	150	3.81
7	180	4.57
8	220	5.59
9	260	6.60
10	300	7.62

Figure 11A. External track width necessary for current limit.

Figure 11A shows the recommended external track width (mil or mm) for the amount of current (amp) assuming 1 oz copper is used per IPC Class 2 standard. This ensures that the traces are wide enough to sustain the current flow.

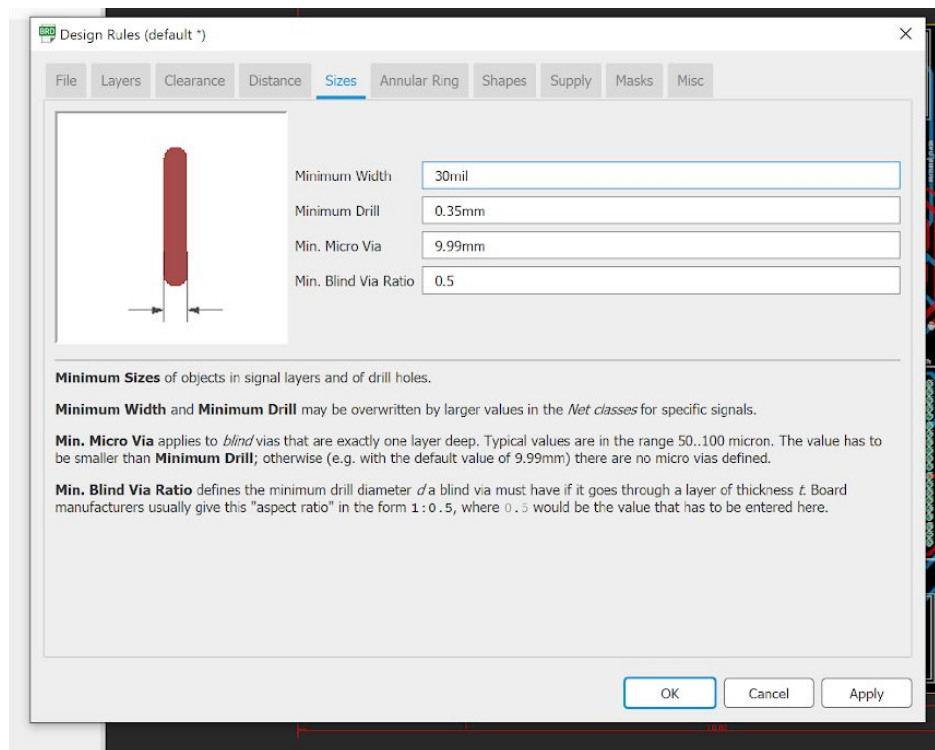


Figure 12A. Design parameters for track width.

Under the edit tab, the user can select the 'design rules' to change the trace width, spacing between widths and board edge, spacing between trace and via, and many other important rules. Figure 12A shows a proper demonstration of the trace width being changed to a proper size for the current. After the width has been set for the traces, the auto routing process can be completed. Depending on the parts that are being used (through hole or SMD), this can affect the amount of layers that are needed. Since the board for this project has all through hole components, a two layer board was chosen as this is the most cost effective solution versus multi-layer boards.



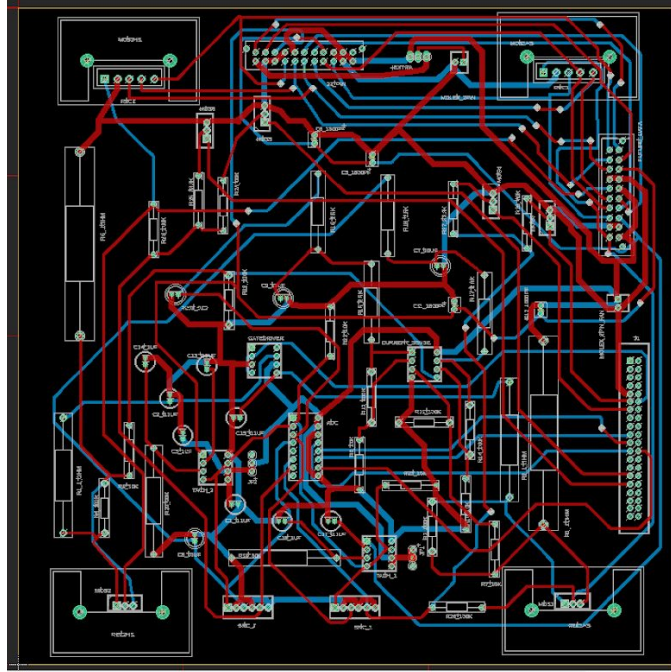


Figure 13A. Final PCB for dyno system.

The figure above shows a completed board file once the auto router is done. Once all the tracks have been auto routed, it is very important to revise all the tracks to ensure none are too close or crossing each other. For each power track, the user must ensure that it does not have a 90 degree sharp turn on it. Instead allow for at least a 120 degree or a chambered line. This will ensure that the power will not leak from track to track thus causing issues within the circuit. For this two layer board, the red tracks signify the top of the board and the blue tracks signify the bottom of the board. Once the design of the board works out okay, acquire the gerber files and drill file from within Eagle to ensure all proper files are saved to send to the manufacturer of your choice. The gerber files can be extracted by selecting file, cam processor, and process job. This will ensure that the user extracts the gerber files (top & bottom copper, top & bottom solder mask, top & bottom silkscreen), drill files, and assembly files (bill of materials). Each one of these files ensures that the design created is constructed to its best accuracy from the manufacturer without ambiguity in the interpretation of the design.



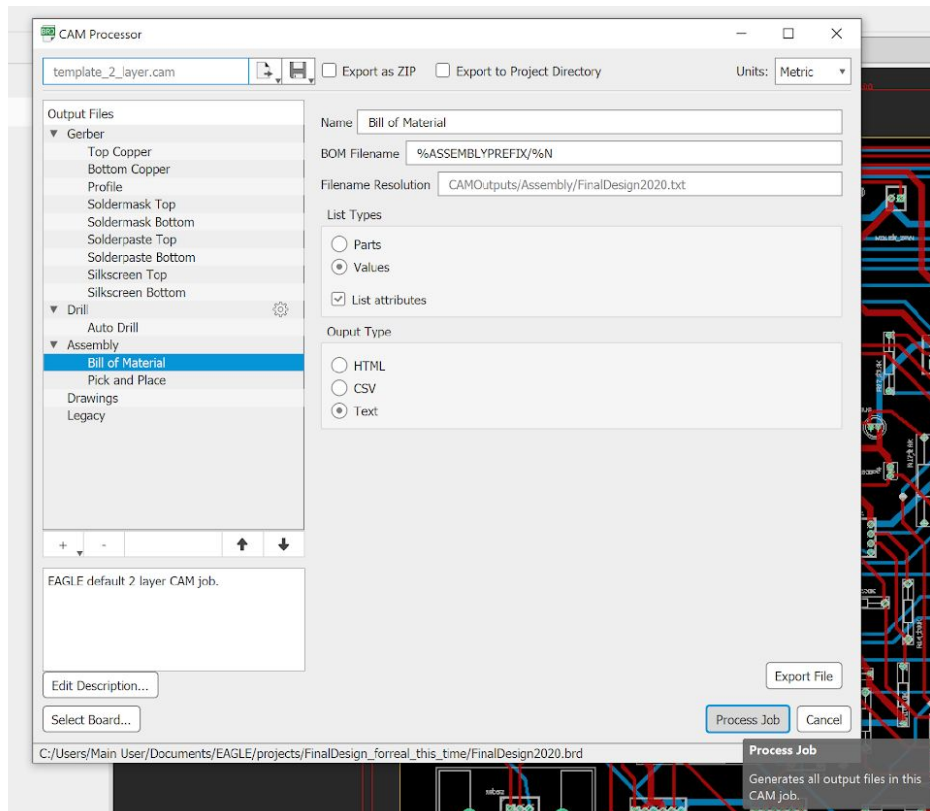


Figure 15A. Cam processor to generate manufacturing files.

The next step is to email or phone the manufacturer of your choice. The team decided to use PCBway, which was recommended by Dr. Wetz since he has used them before and is happy with the quality. The process is straightforward. First have them run a quote based on the number of layers and dimensions of the board to ensure that the cost fits the budget. Then, once they have received your files, the company will process the files and if everything turns out fine, they will approve the build. Although they are based out of China, the turnaround time is less than two weeks. The quoting system used by PCBWay is available on their website and it takes only a few minutes to get an accurate quote with simple information.

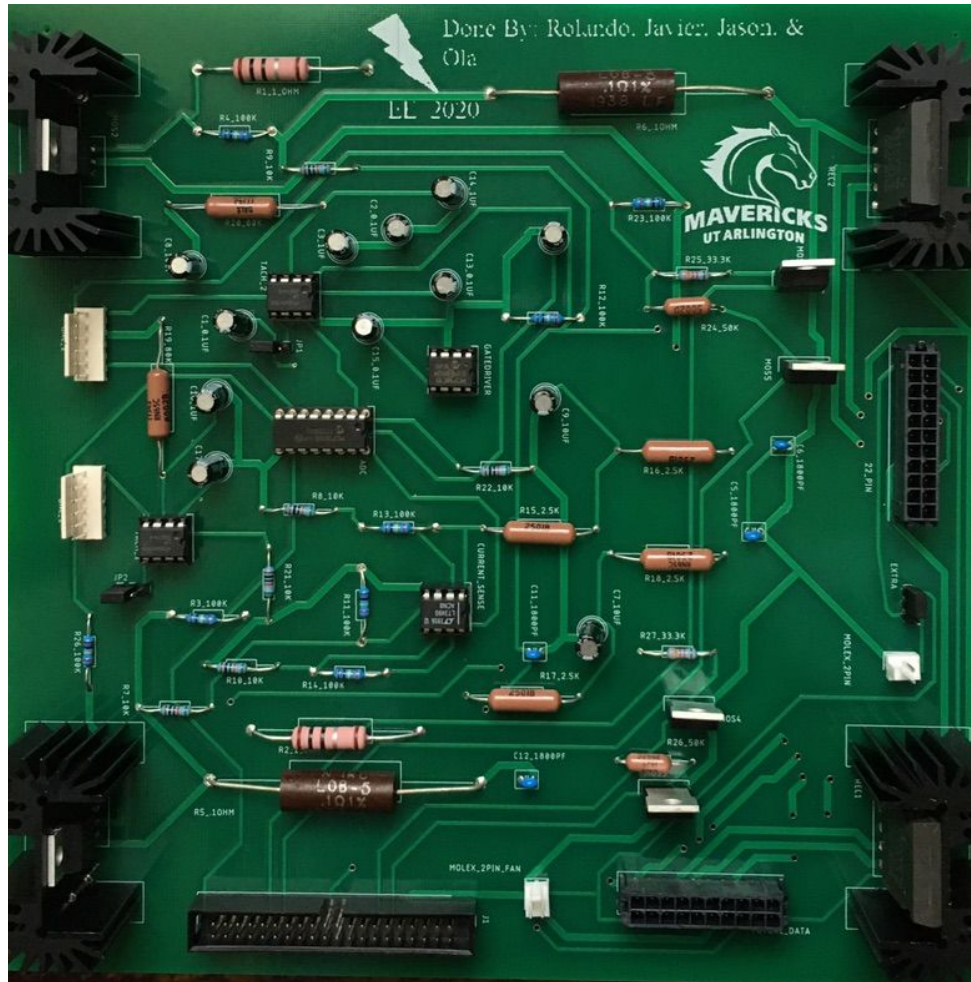


Figure 15A. Professionally etched PCB.

Figure 15A shows the final product which was manufactured by PCBway. The design came out really good. Although having a professionally etched board ensures it will look great, UTA has its own PCB milling laboratory. This allows students to design and create their own boards from scratch. Tige “Todd” Kelly, one of the undergraduate electrical engineering lab technicians, is available to the undergraduate students for solder training classes, equipment training, PCB design and milling, 3D printing, and parts ordering. Todd will direct students as to how the milling process is done and once completed, he will guide the student to allow them to apply the masking screen and silk screen onto the PCB. This gives students a great hands on experience.

## PCB Manufacturing at UTA

In order to start the PCB milling process, the user must first design and create the schematic and board files for the PCB. Once the board file is designed, a useful and time saving process will be to check the board file with a program called “Bantam Tools”. This program is used for the milling machine at the lab and also ensures that all the tracks are correctly spaced and not

touching each other along with verifying that the hole sizes are big enough based on the drill bits available at UTA.

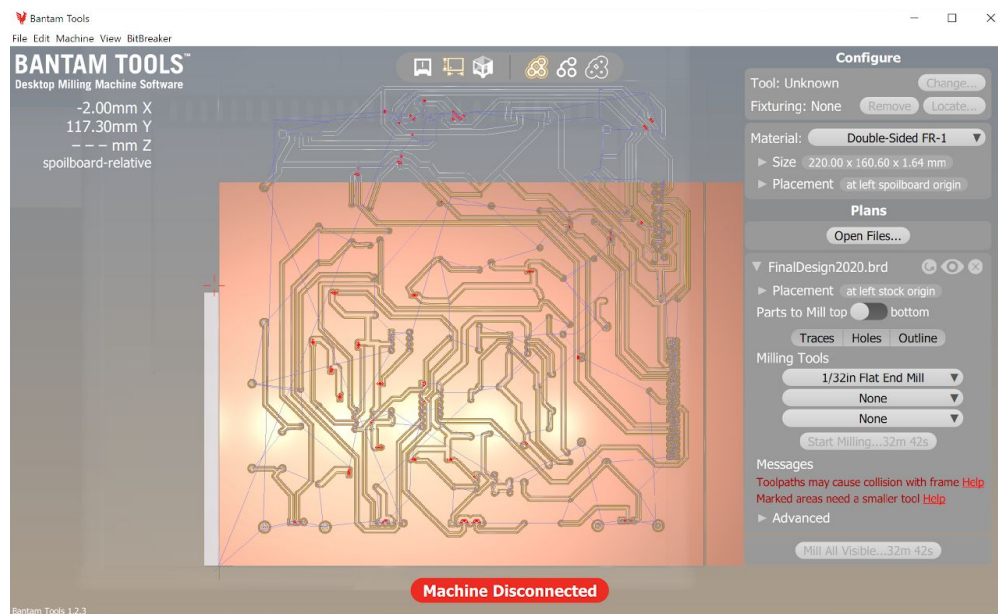


Figure 16A. Bantam Tools view of board file.

The figure above shows a view of how the board file is uploaded to the program. The milling machine is only capable of producing boards no bigger than 4x5" dimensions. Although this is an issue right now, UTA is in the works of getting a larger milling machine that will add the availability of bigger boards.

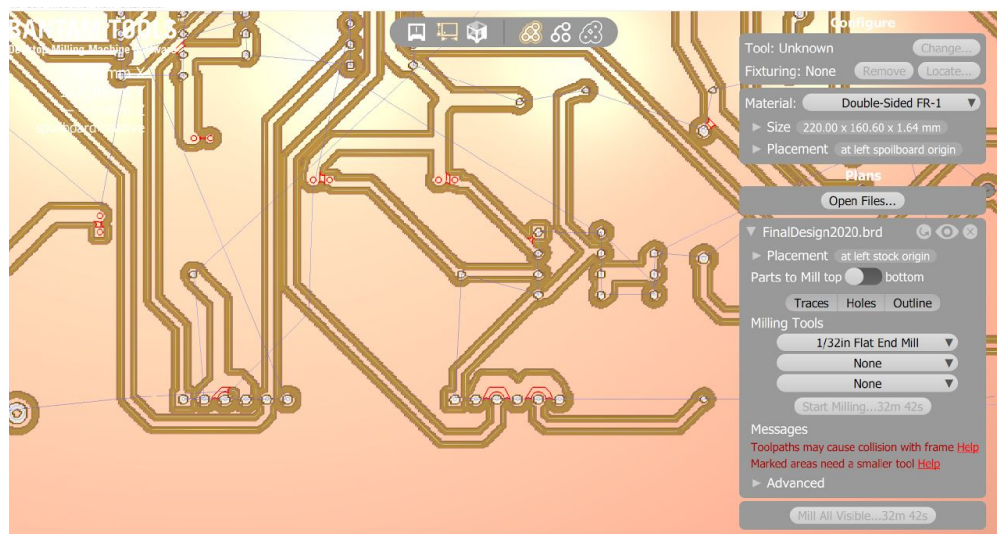


Figure 17A. Bantam Tools program showing errors.

The Bantam Tools program shows you the errors within the board as you can see from the red markings above. This will allow the user to go back and edit the board in order to get rid of those

errors. Another way to eliminate the errors would be to change the milling tools, drill bit, or size down to a finer tool.

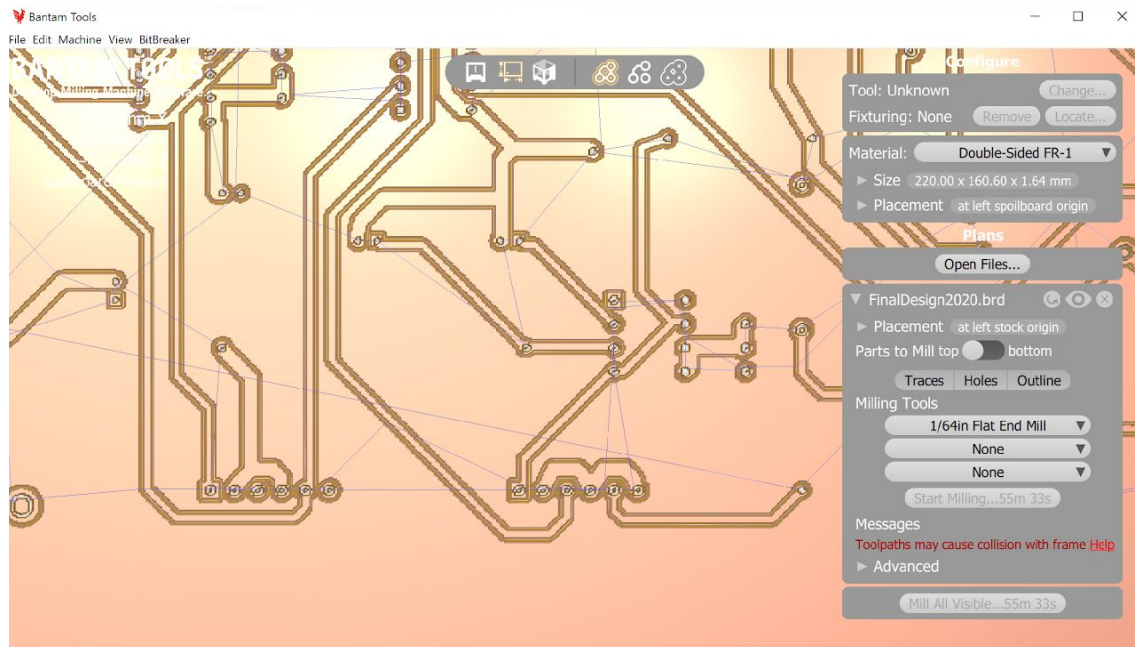


Figure 18A. Milling tool size changed.

As you can see from the above figure, the milling tool was changed from a 1/32" to a 1/64" drill size to get rid of the errors that were showing. What that achieves is to tell the program that a smaller drill bit will be used that allows it to mill it in a finer way. Once that process is done, the files are now ready to be handed to Todd to begin the milling process. This process takes a couple of days due to the fact that it is done in multiple steps. After the electrolysis bath it goes through and copper plating, this is what a board will look like.

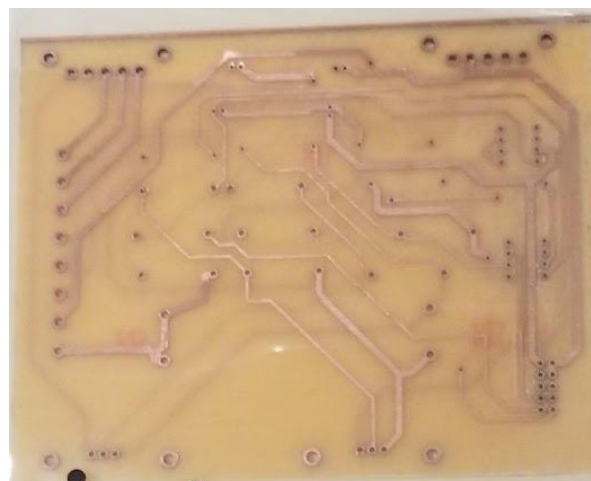


Figure 19A. PCB once milling is complete.



The figure above shows what the board looks like once it's done going through the milling process. The next step will be to add the solder mask and the silk screen onto the board. In order to start this process, the solder mask file and silk screen file will need to be acquired from the cam output within Eagle as explained from an earlier figure titled 'Cam Processor'. The files will then be given to Todd which he will print them out.

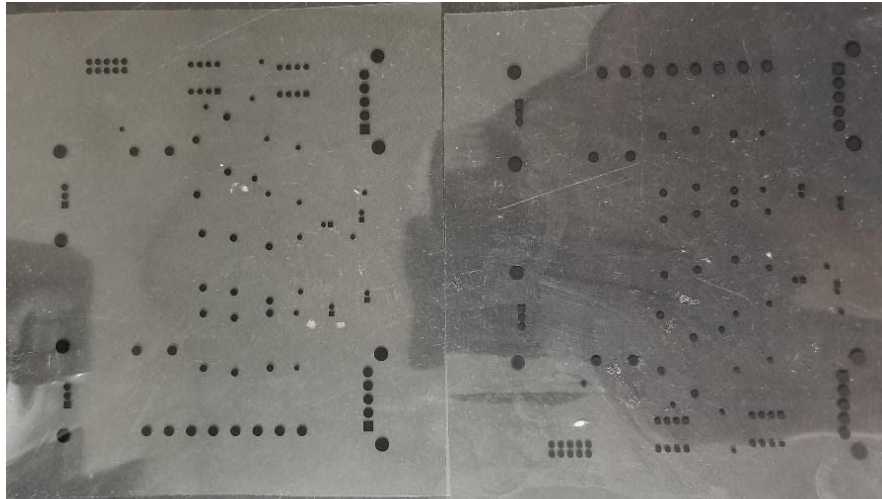


Figure 20A. Clear printing paper of the solder mask layer called photographic.

The figure above shows the layer that is placed on the top and the bottom of the PCB. Once the photographic mask is prepared, the filling paint and curing agent is mixed in a coating dish. A roller is used to apply several thin layer coatings onto the PCB without putting heavy pressure until the entire board is covered.



Figure 21A. PCB inside oven after coating.

As the above figure shows, the PCB is placed in an oven. Once it comes out the oven, a few minutes will be needed for the board to cool down. After this step, it is placed in a machine that exposes it to UV rays which helps make the mask permanent. The photographic paper is used to

protect specific places (such as the holes) that should not have the solder mask. Once it comes out, it is scrubbed to remove the mask in areas that should not have solder mask paint.



Figure 22A. Masking procedure at the UTA laboratory.

The figure above shows the steps that need to be followed in order to add the solder mask (along with silkscreen) onto the PCB. Each step has critical details such as the timing of the oven, the temperature, and many other factors. This poster is found at the UTA PCB laboratory at Nedderman Hall where the process takes place.

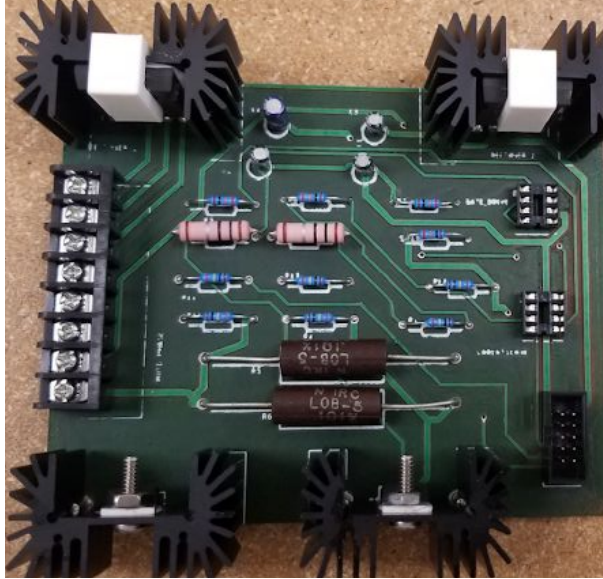


Figure 23A. Assembled PCB created by UTA equipment only.

As the figure above shows, the PCB turns out to be similar to a professionally etched board. Due to the fact that this was the first time this specific process was done by students, there is room for improvement to create a higher quality board. The silk screen came out a bit faded possibly due to the paint not applied properly but this process should improve over time. All in all, the process is fairly doable and just needs some time and patience since most of the procedure has some waiting time. The reason why this board was not used for the final PCB in our project is due to two reasons. The PCB size is limited so we had to create two separate boards for our design. Secondly, there were continuity issues with the boards themselves. With a multimeter, traces were checked and we noticed that some traces did not complete. One possible explanation is the electrolysis bath did not create the holes/vias correctly so there were issues with continuity. We had to use speaker wire at the back of the board in order to make the circuit function correctly. Keep this in mind when utilizing the UTA lab to make a PCB. Check the traces once assembled or you might remain confused as to why the circuit is not working. This issue might not exist at the time the reader is reading this report so verify with Todd or another technician if other students have reported any issues. There are talks about UTA purchasing a \$30,000 milling machine for use sometime next year that is higher quality and capable of bigger board sizes so inquire about using this machine for your projects.



## User Guide

# Wheelchair Dynamometer

## User's Manual

Version 1.0

May 2, 2020

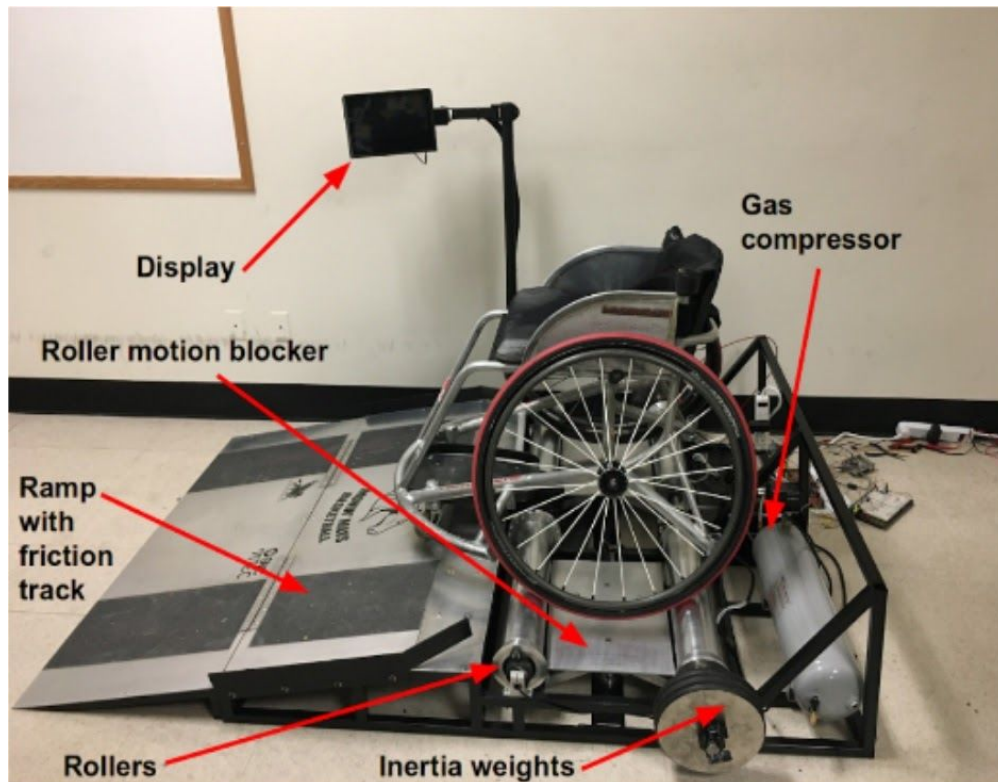


Figure 24A. Mechanically complete wheelchair dynamometer.

This is a document describing the operation of the wheelchair dynamometer co-created by the mechanical engineering & electrical engineering departments at The University of Texas at Arlington. The dynamometer is both a workout machine for wheelchair users and a performance-observance tool that can track speed, torque, caloric expenditure, distance traveled, and other parameters. Its intended users are the kinesiology department and the Movin' Mavs basketball team. In this document, there is a guide for using the mechanical system, handling BNC data acquisition, and input connections to the enclosure, along with a short troubleshooting section.

## Mechanical System

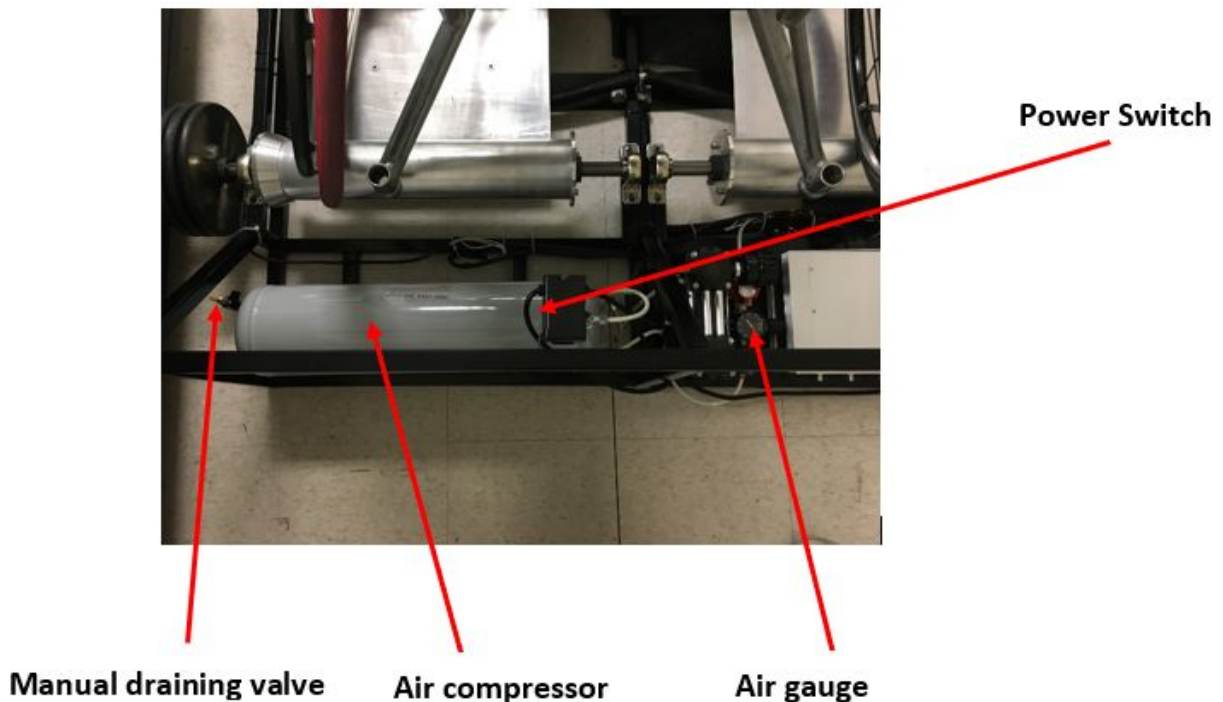


Figure 25A. Mechanical system of the dyno.

Once everything is plugged to the wall and the electrical switch turned on, the power switch of the compressor as shown in the figure below needs to be switched to auto.



Figure 26A. Close up picture of compressor power switch.

Once the gauge shown below reaches around 20 psi, that will be sufficient pressure for a workout session. The power switch can then be switched to the off position. Once done, the knob near the display can be rotated to lift up the platform. This eases the process of the wheelchair user getting onto the dyno without any complication since the rollers are locked into place. After the user is on the platform safely, the knob can be turned again. This moves the platform down and gets the dyno ready for use.



Figure 27A. Close up picture of air gauge that should reach 20 psi.



Figure 28A. Close up picture of manual draining valve.

When the dyno is no longer in use, it is good practice to use the manual draining valve as shown in the figure above to drain the air from the pump until the gauge goes back to zero. However, if this is not done, the air will eventually be drained automatically over a period of time. Common practice is to drain the tank after use.

## Data Acquisition

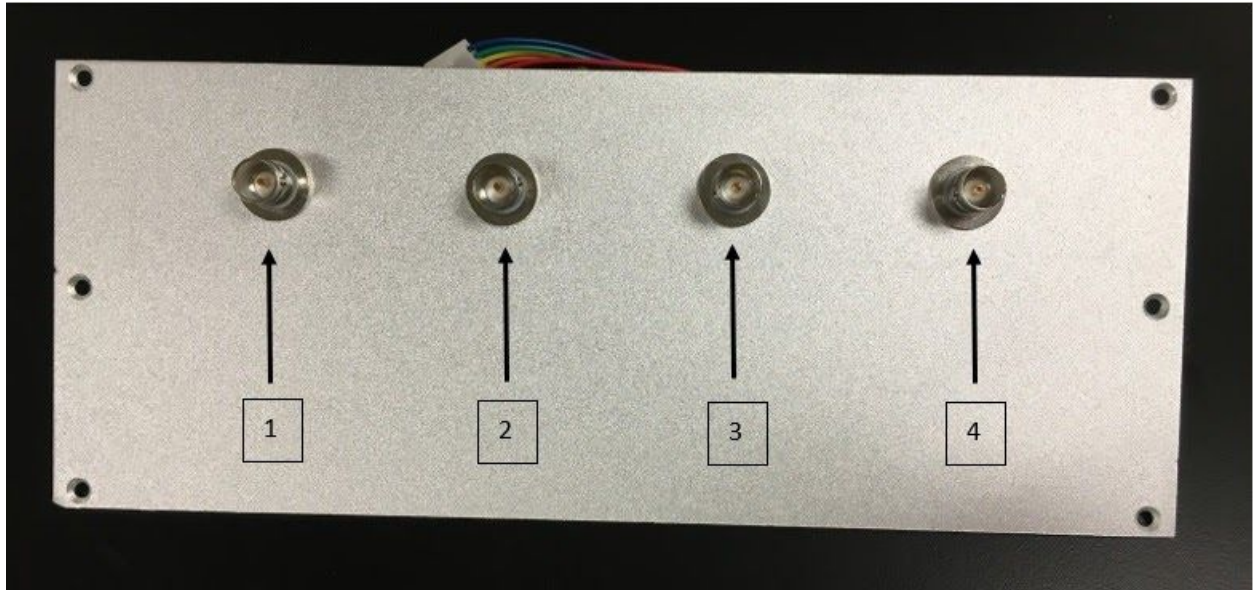


Figure 29A. Back panel with BNC connectors for torque and speed measurements.

BNC Connector	Description
1	Left Motor Speed [0-5V]
2	Left Motor Torque [0-5V]
3	Right Motor Speed [0-5V]
4	Right Motor Torque [0-5V]

### Speed Conversion:

$\omega = 10 * GainFactor * V_{speed} [rad/s]$  based on 0.1 motor voltage constant. GainFactor [V/V] varies depending on level difficulty selected. Level 1: 6, Level 2: 3, Level 3-5: 2, Level 6: 1. From this, the speed can be calculated based on current workout level.

### Torque Conversion:

$\tau = V_{torque} [Nm]$  for all workout levels.



## Input Connection

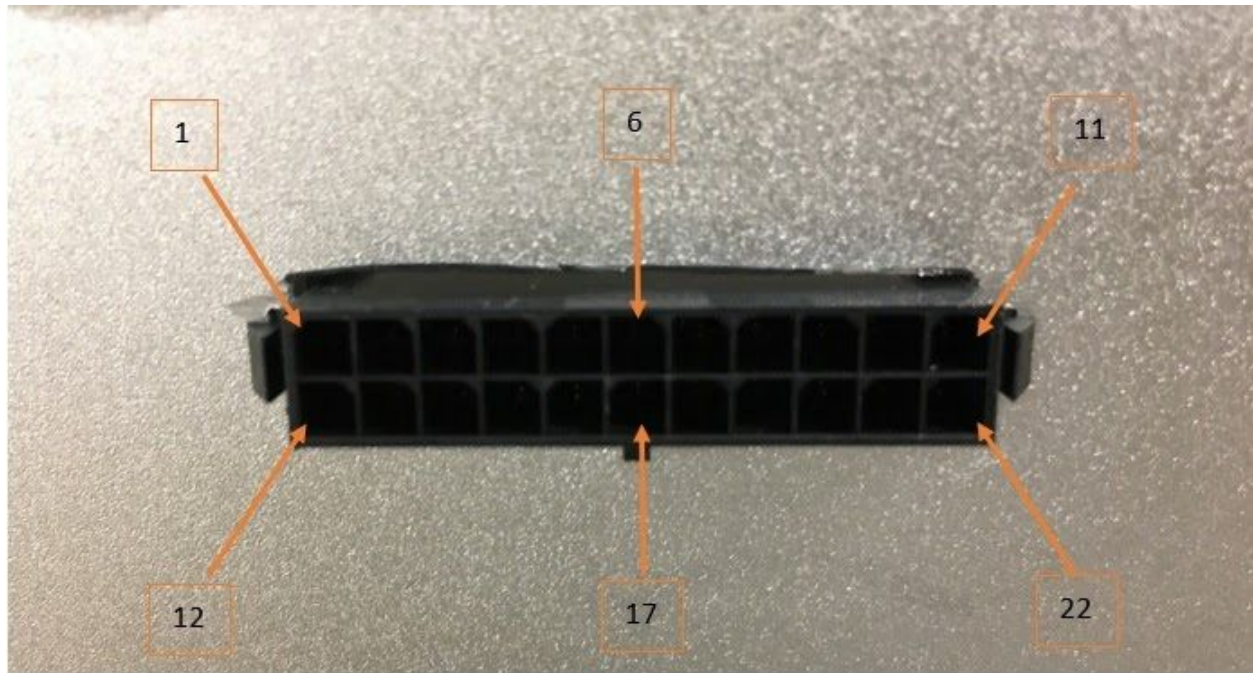


Figure 30A. Molex single connection for all power and signal wires.

There is a single connector that groups all power and signal wires from the dynamometer and directs them to the printed circuit board (PCB) inside the enclosure. The connector is composed of both a header and a wire housing manufactured by Molex. The following table goes into detail on what each connection entails. Every wire is also marked with the pin number in case of future revisions to the dyno.

Pin	Title	Description
1	Power 5V	Main power supply for all electrical systems. Should be 2.5A minimum.
2	GND	Power supply ground.
3	Left Motor Phase A	Phase A for the left motor.
4	Left Motor Phase B	Phase B for the left motor.
5	Left Motor Phase C	Phase C for the left motor.
6	Right Motor Phase A	Phase A for the right motor.

<b>7</b>	<b>Right Motor Phase B</b>	<b>Phase B for the right motor.</b>
<b>8</b>	<b>Right Motor Phase C</b>	<b>Phase C for the right motor.</b>
<b>9</b>	<b>Earth GND</b>	<b>Earth ground that shares circuit ground with metal frame. Reduces noise.</b>
<b>10</b>	<b>GND</b>	<b>Ground for second power supply.</b>
<b>11</b>	<b>Extra 5V Pin (Optional)</b>	<b>Separate Raspberry Pi power from main supply by moving circuit jumper.</b>
<b>12</b>	<b>*Not in use*</b>	<b>Floating Pin.</b>
<b>13</b>	<b>Left Motor Hall Effect Positive</b>	<b>5V supply line for hall effect sensors.</b>
<b>14</b>	<b>Left Motor Hall Effect Negative</b>	<b>Share ground with hall effect and circuit.</b>
<b>15</b>	<b>Left Motor Hall Effect Phase A</b>	<b>Digital hall effect signal for left phase A.</b>
<b>16</b>	<b>Left Motor Hall Effect Phase B</b>	<b>Frequency of signal contains speed information, not amplitude.</b>
<b>17</b>	<b>Left Motor Hall Effect Phase C</b>	<b>Frequency information usually 0-100Hz.</b>
<b>18</b>	<b>Right Motor Hall Effect Positive</b>	<b>Same as the left hall effect.</b>
<b>19</b>	<b>Right Motor Hall Effect Negative</b>	<b>Hall effect ground shared with circuit.</b>
<b>20</b>	<b>Right Motor Hall Effect Phase A</b>	<b>Right motor hall effect signal.</b>
<b>21</b>	<b>Right Motor Hall Effect Phase B</b>	<b>Only one signal used for speed determination.</b>
<b>22</b>	<b>Right Motor Hall Effect Phase C</b>	<b>Same as above.</b>

## File Folder Structure

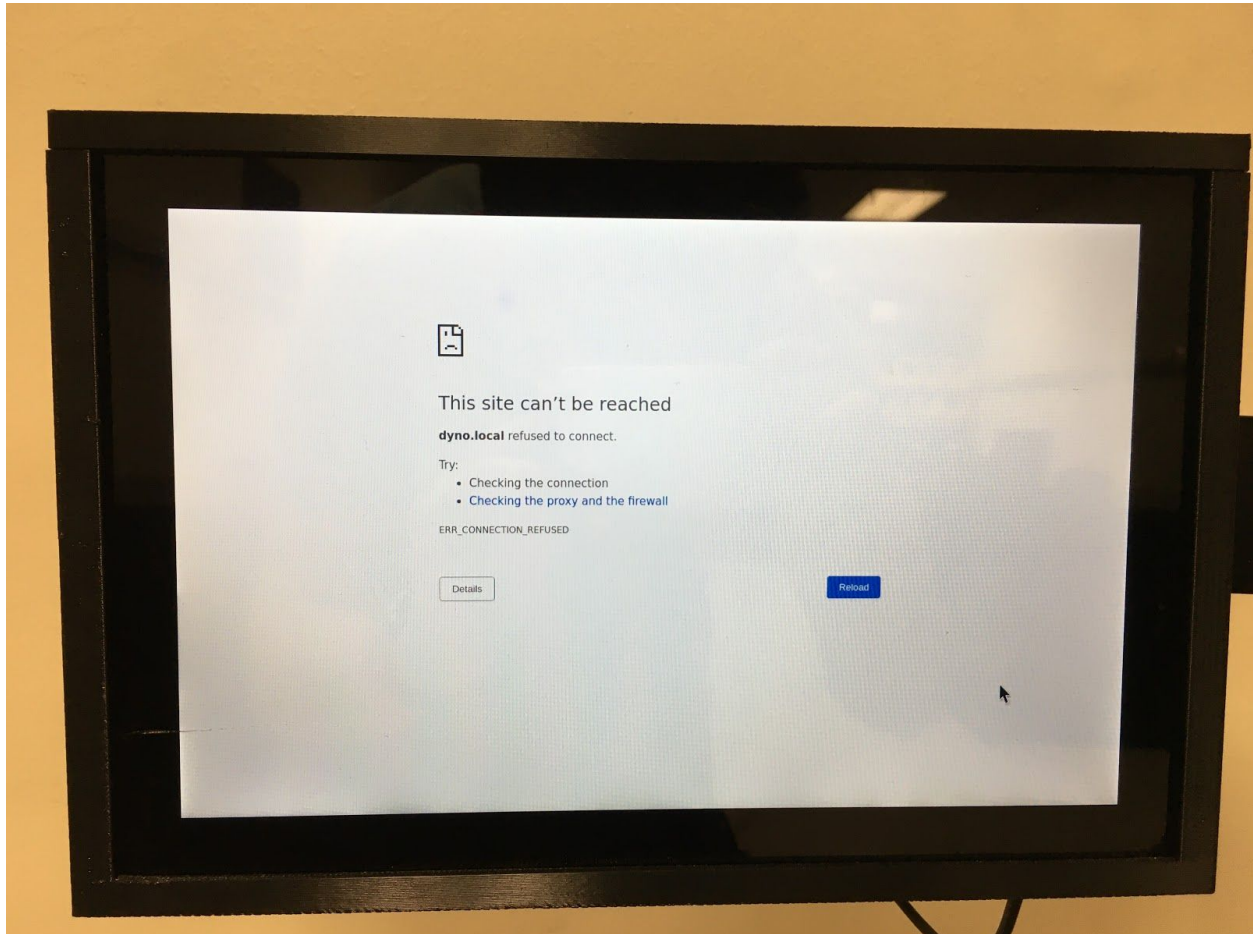
The table below contains descriptions for each folder in the deliverables to better understand where everything is located.

Folder Name	Description
Parts List	Parts lists requested throughout the project. Final list is included in the report.
Second Team Deliverables	The files submitted by the second EE team that worked on this project.
3D Designs	Collection of SolidWorks files for a 3D printed enclosure that was not utilized in the final design.
SD Card - Image Backups	Complete image replicas for both the main system dyno and the attached display that runs the web browser. Can be deployed to new SD cards using the attached program.
PCB Files [Final Board]	The final board design eagle files.
PCB Eagle Libraries	Eagle components used to build PCB such as resistors, connectors, etc...
Simulink Simulations	Simulations for testing the validity of mosfet switching under saturated conditions and the effects of voltage drain on shunt resistance.
Raspberry Pi Code	All of the code + html files used in the main dyno system that are under the hood or backend
Natural Rolling Code	Preliminary work done on the software aspect of natural rolling implementation.
Important Pictures	Motor controller labeling images, final schematic, and final block diagram.
Senior Design PPT & Reports	Final report and presentation for senior design along with interim stuff like presentation video.
Circuit	Relevant research done on best methods to achieve project goals along with first PCB files in the initial testing phase.
Experiments	Experiments conducted before final pcb to test hall effect sensor along with experiment related to final pcb.
Concepts Presentation & Reports	Files from Concepts class.
Quick Start Guide	Starting point that has the user guide for the dyno device.
Videos	Interim presentation, demo video, etc...



## Troubleshooting

Q1: I selected the user interface button but I get this screen. Why is this feature not working?



A1: This means that the thingsboard local server responsible for presenting the user interface is not on yet. Typically, this takes a few minutes after the dyno is turned on. Wait another minute or two and select the reload button once. The main page loads earlier than the UI and allows a user to change settings or view previous data. The UI is the only feature that takes a little bit longer to load.

## References

- [0] "Invictus Active Trainer | Wheelchair Fitness | Series 2", Invictus Active, 2019. [Online]. Available: <https://www.invictusactive.com/invictus-active-trainer/>. [Accessed: 29- Nov- 2019].
- [1] D. J. Griffiths, "Introduction to Electrodynamics," 2017.
- [2] H. Satoh, "Study on increasing the surge capability of a lightning surge protection, semiconductor device", *IEEE Transactions on Electromagnetic Compatibility*, vol. 35, no. 2, pp. 311-315, 1993. Available: 10.1109/15.229440.
- [3] L. H. V. Van Der Woude, H. E. J. Veeger, R. H. Rozendal, G. J. V. I. Schenau, F. Rooth, and P. V. Nierop, "Wheelchair racing," *Medicine & Science in Sports & Exercise*, vol. 20, no. 5, 1988.
- [4] R. Walker, S. Powers, and M. K. Stuart, "Peak oxygen uptake in arm ergometry: effects of testing protocol," *British Journal of Sports Medicine*, vol. 20, no. 1, pp. 25–26, Jan. 1986.
- [5] "How to install a web server on the Raspberry Pi (Apache PHP MySQL)," *Raspberry Pi FR*, 29-Aug-2019. [Online]. Available: <https://howtoraspberrypi.com/how-to-install-web-server-raspberry-pi-lamp/>. [Accessed: 30-Oct-2019].
- [6] "How to Set Up the htaccess File on Apache," *Linode Guides & Tutorials*, 12-Sep-2019. [Online]. Available: <https://www.linode.com/docs/web-servers/apache/how-to-set-up-htaccess-on-apache/>. [Accessed: 30-Oct-2019].
- [7] L. Jung, "h5ai," *larsjung.de*. [Online]. Available: <https://larsjung.de/h5ai/>. [Accessed: 30-Oct-2019].
- [8] Thingsboard, "ThingsBoard installation options," *ThingsBoard*. [Online]. Available: <https://thingsboard.io/docs/user-guide/install/installation-options/>. [Accessed: 30-Oct-2019].
- [9] T. DiCola, "Raspberry Pi Analog to Digital Converters," *Adafruit Learning System*. [Online]. Available: <https://learn.adafruit.com/raspberry-pi-analog-to-digital-converters/mcp3008>. [Accessed: 30-Oct-2019].

[10] Fivdi, “fivdi/pigpio,” *GitHub*, 24-Oct-2019. [Online]. Available: <https://github.com/fivdi/pigpio>. [Accessed: 30-Oct-2019].

[11] Person and wikiHow, “How to Execute a Script at Startup on the Raspberry Pi,” *wikiHow*, 12-Aug-2019. [Online]. Available: <https://www.wikihow.com/Execute-a-Script-at-Startup-on-the-Raspberry-Pi>. [Accessed: 30-Oct-2019].

[12] “Setting up a Raspberry Pi as a Wireless Access Point,” *Setting up a Raspberry Pi as a Wireless Access Point - Raspberry Pi Documentation*. [Online]. Available: <https://www.raspberrypi.org/documentation/configuration/wireless/access-point.md>. [Accessed: 30-Oct-2019].

[13] “Renaming your Raspberry Pi - the 'hostname',” *The Pi Hut*. [Online]. Available: <https://thepihut.com/blogs/raspberry-pi-tutorials/19668676-renaming-your-raspberry-pi-the-host-name>. [Accessed: 30-Oct-2019].

[14] W. Gordon, “How to Clone Your Raspberry Pi SD Card for Foolproof Backup,” *How*, 20-Mar-2018. [Online]. Available: <https://www.howtogeek.com/341944/how-to-clone-your-raspberry-pi-sd-card-for-foolproof-back-up/>. [Accessed: 30-Oct-2019].

[15] T. Agarwal, H. Venu, T. Agarwal, Bhavya, T. Agarwal, H. Bhavya, Harikrishna, Rajeswar, and Imarn, “Fundamentals Of Current Sensor Sensing Concepts and Applications,” *ElProCus*, 18-Mar-2016. [Online]. Available: <https://www.elprocus.com/current-sensor/>. [Accessed: 30-Oct-2019].

[16] “System trade-offs for high- and low-side current measurements,” *E2E*. [Online]. Available: [https://e2e.ti.com/blogs\\_/b/analogwire/archive/2017/06/01/system-trade-offs-for-high-and-low-side-current-measurements](https://e2e.ti.com/blogs_/b/analogwire/archive/2017/06/01/system-trade-offs-for-high-and-low-side-current-measurements). [Accessed: 30-Oct-2019].

[17] B. Schweber, “Using Resistors for Current Sensing: It's More Than Just  $I = V/R$ ,” *Power Electronics*, 30-Mar-2018. [Online]. Available: <https://www.powerelectronics.com/power-management/using-resistors-current-sensing-it-s-more-just-i-vr>. [Accessed: 30-Oct-2019].

[18] N. \*, “Schmitt Trigger: Working with IC555, Transistors, and Applications,” *ElProCus*, 15-Mar-2019. [Online]. Available: <https://www.elprocus.com/what-is-a-schmitt-trigger-working-and-applications/>.

- [19] "Does a MOSFET need a gate resistor?," *Electronics Forum (Circuits, Projects and Microcontrollers)*. [Online]. Available: <https://www.electro-tech-online.com/threads/does-a-mosfet-need-a-gate-resistor.87419/>. [Accessed: 30-Oct-2019].
- [20] "Question about mosfet gate resistor," *Electrical Engineering Stack Exchange*, 01-Aug-1963. [Online]. Available: <https://electronics.stackexchange.com/questions/68748/question-about-mosfet-gate-resistor>. [Accessed: 30-Oct-2019].
- [21] "Raspberry Pi Power Limitations," *Raspberry Pi Stack Exchange*, 01-Oct-1966. [Online]. Available: <https://raspberrypi.stackexchange.com/questions/51615/raspberry-pi-power-limitations>. [Accessed: 30-Oct-2019].
- [22] Ravi, "What is a Bypass Capacitor? Tutorial: Applications," *Electronics Hub*, 19-Apr-2019. [Online]. Available: <https://www.electronicshub.org/bypass-capacitor-tutorial/>. [Accessed: 30-Oct-2019].
- [23] Gonzalez and alex GONZALEZ, "What is 3 Phase Rectifier ? - 3 Phase Half Wave, Full Wave & Bridge Rectifier," *Electronics Coach*, 23-Mar-2018. [Online]. Available: <https://electronicscoach.com/3-phase-rectifier.html>. [Accessed: 30-Oct-2019].
- [24] "Second Order Filter: Second Order Low Pass Filter Design," *Basic Electronics Tutorials*, 10-Feb-2018. [Online]. Available: <https://www.electronics-tutorials.ws/filter/second-order-filters.html>. [Accessed: 30-Oct-2019].
- [25] "Current sense amplifier," *Wikipedia*, 28-Sep-2017. [Online]. Available: [https://en.wikipedia.org/wiki/Current\\_sense\\_amplifier](https://en.wikipedia.org/wiki/Current_sense_amplifier). [Accessed: 30-Oct-2019].
- [26] IGBT and MOSFET Drivers Correctly Calculated. Biel-Bienne: CT Concept Technologie AG, 2019.
- [27] M. Sayed and H. Svensson, "PWM rise and fall time, frequency and duty cycle setting to control voltage using a MOSFET", *Electrical Engineering Stack Exchange*, 2019. [Online]. Available: <https://electronics.stackexchange.com/questions/357049/pwm-rise-and-fall-time-frequency-and-duty-cycle-setting-to-control-voltage-usin>. [Accessed: 01- Dec- 2019].