To: Dr. Armin Eilaghi

From: CWC GEN

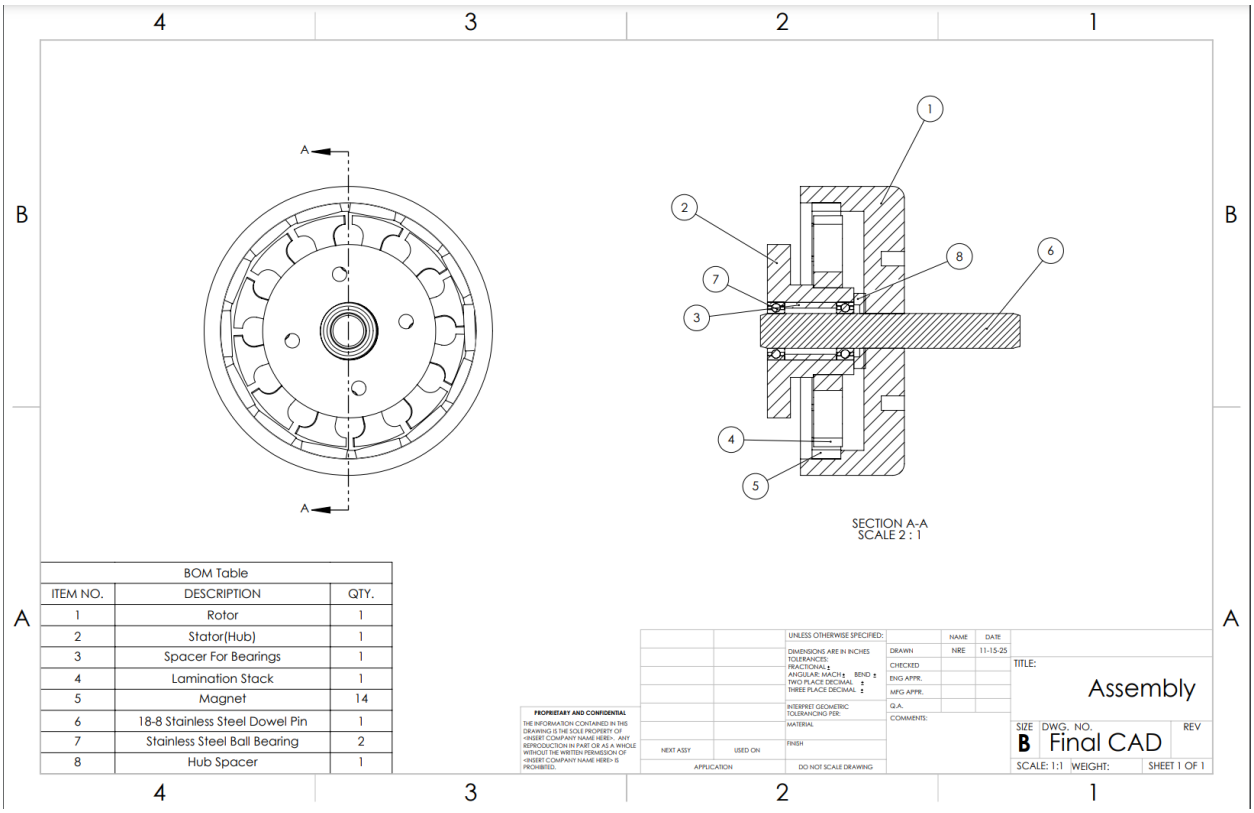
Date: 12/04/2025

Re: Operation/Assembly Manual

1. Generator

1.1. Assembly

The generator is made up of two components, the rotor and stator. See figure one for full CAD assembly. In this design, the rotor is where the magnets are adhered to. The steel shaft is permanently secured to the rotor, as seen in figure 4. The stator is made up of a hub and lamination stack. The lamination stack is permanently secured to the hub by 832-C epoxy. This design has two bearings that connect the stator to the shaft. The bearings are an interference fit to the shaft and are a transitional fit to the hub. To assemble and disassemble the generator, the stator will be pulled off of the bearings separating the two components.



BOM Table		
ITEM NO.	DESCRIPTION	QTY.
1	Rotor	1
2	Stator(Hub)	1
3	Spacer For Bearings	1
4	Lamination Stack	1
5	Magnet	14
6	18-8 Stainless Steel Dowel Pin	1
7	Stainless Steel Ball Bearing	2
8	Hub Spacer	1

Figure 1: Final CAD

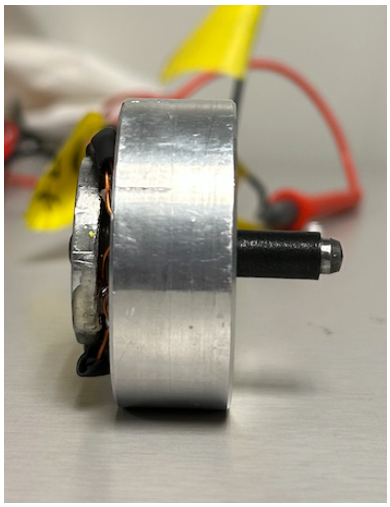


Figure 2: Full Assembly

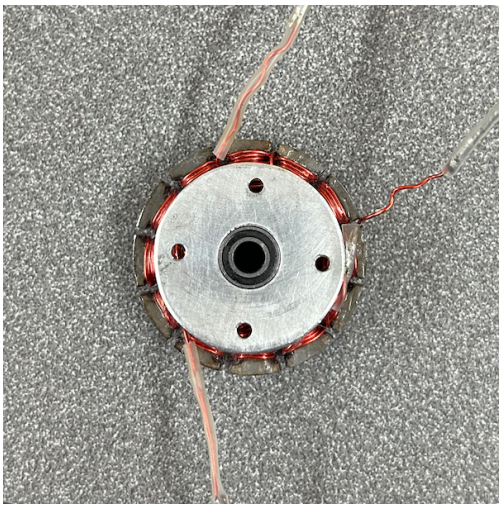


Figure 3: Stator

Figure 4: Rotor

1.2. Operation

To operate the generator on the dynamometer (dyno), the stator will be bolted to a bracket that is bolted to the dyno, so it stays stationary. The shaft will be attached to the drive motor wil1 a coupler. Figure 5 displays how the generator will be mounted. The three phases will be connected to an AC to DC rectifier that is connected to the rest of the sensors, so results can be gathered. It is connected to the rectifier by connecting the phases into wire connectors, see Figure 6 for reference.

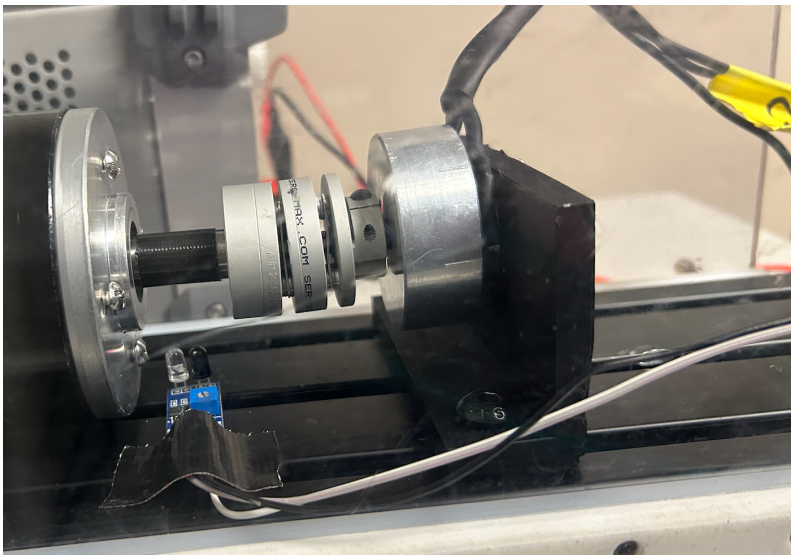


Figure 5: Generator Hooked Up to Dyno

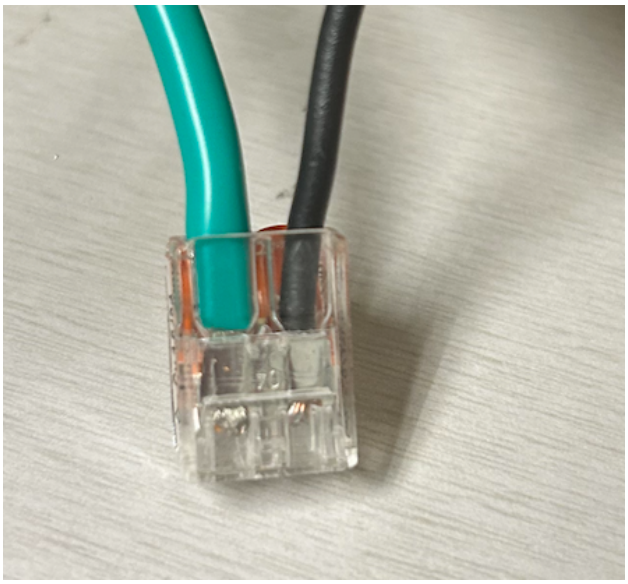


Figure 6: Phase Hook-up

1.3. Potential Failures and Maintenance

Potential failures within the generator are the coils shorting together. This is due to the magnetic field causing the coils to move and rub during operation. Ways to mitigate this issue are to add phish tape around the lamination stack as a layer of insulation. Encapsulating the coils in epoxy within a vacuum chamber will also prevent the coils from moving during operation. The bearings will need to be replaced at 400 operating hours because they will have reached the expected life span.

2. Dynamometer-Integrated Data Acquisition (DAQ) System

2.1. Assembly/Disassembly

2.1.1 Dynamometer

The dyno is comprised of a testing apparatus and a DAQ system. The testing apparatus fits on a t-slot stand where the brackets are tightened on with locking bolts. When testing, a plexiglass shield is fitted around the testing apparatus to protect users from any potential failures. At the back end there is a stand with a torque transducer attached to a 3D printed element adhered to a metal shaft. The shaft is adhered in a bearing stand for support and to allow for free rotation and torque transmission. Another 3D printed element is adhered to the other end of the metal shaft, small enough to allow easy access to the last 3D printed and adhered element where the back end of the motor screws into. The motor shaft has a coupler attached to allow for easy integration of all different kinds of generators. There is black tape wrapped most of the way around the shaft to allow the reflective surface to be read by the infrared sensor for RPM reading. The generator is attached to a stand at the front end of the dyno completing the testing apparatus. Beyond that, the generator leads are hooked up to the 3-phase AC to DC rectifier which is connected to the DAQ system, the motor is hooked up to the power supply, and the torque transducer is wired to an amplifier that is also connected to the DAQ system. A programmable load is also implemented externally to allow for varied testing with induced loads of constant resistance or current. All listed components are shown in Figure 7.

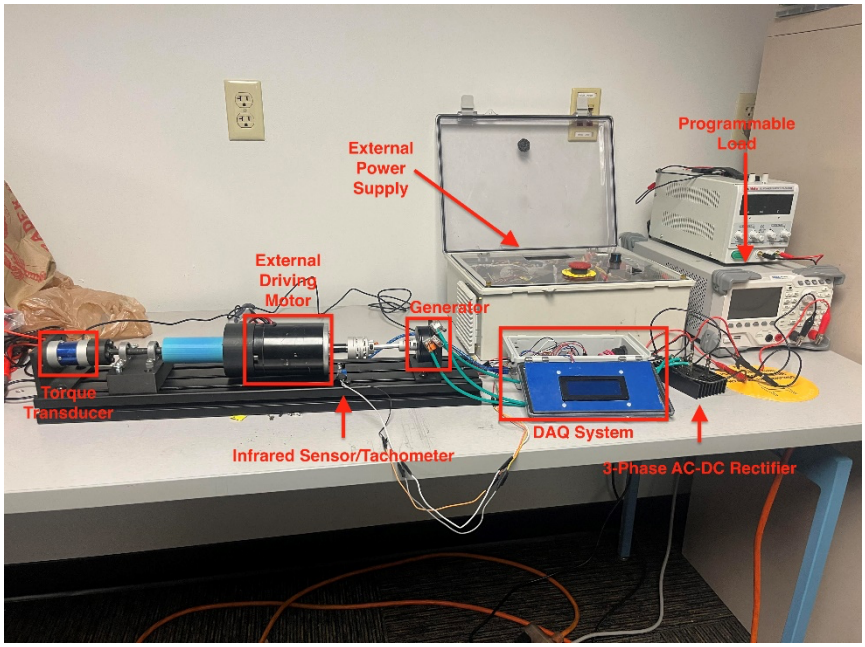


Figure 7: Complete Dyno Setup

2.1.2 DAQ System

The DAQ system is comprised of a voltage divider, current sensor, torque transducer amplifier, infrared sensor, the microcontroller with the source code, an LCD screen, and micro-SD port as shown in Figure 8. The chosen components are shown in Figures 9-14.

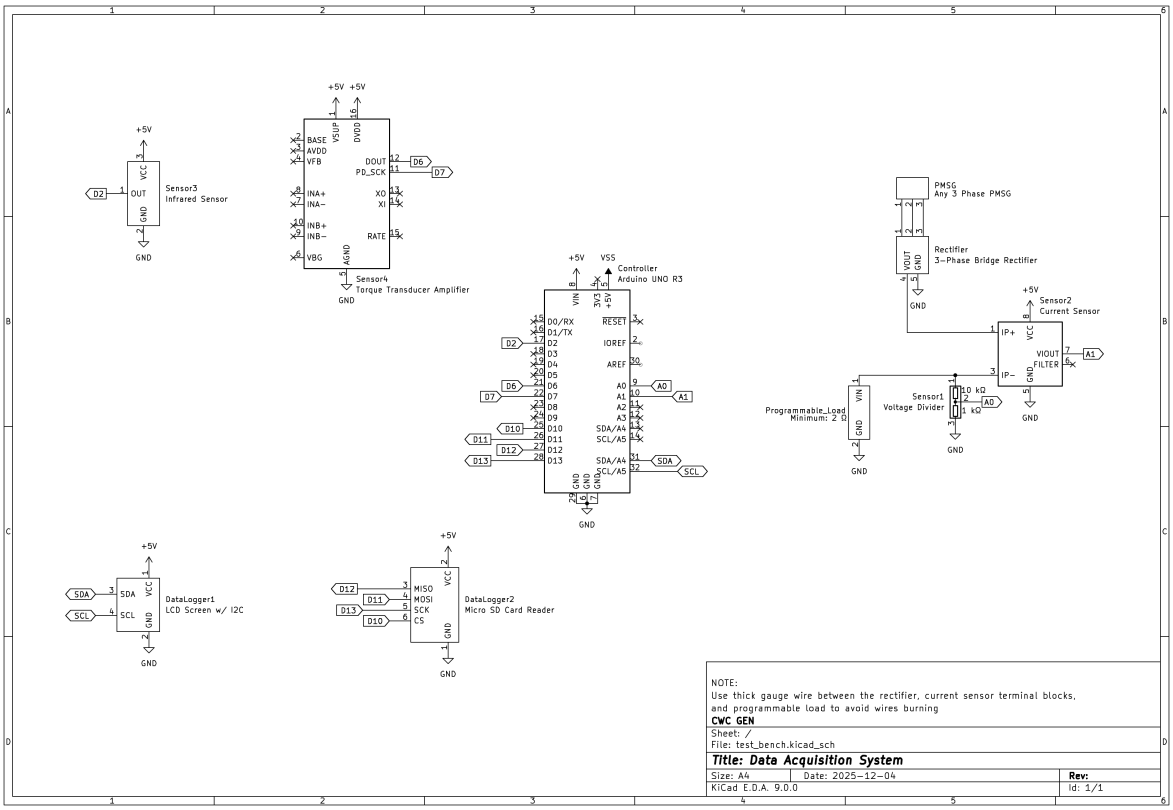


Figure 8: DAQ System Schematic

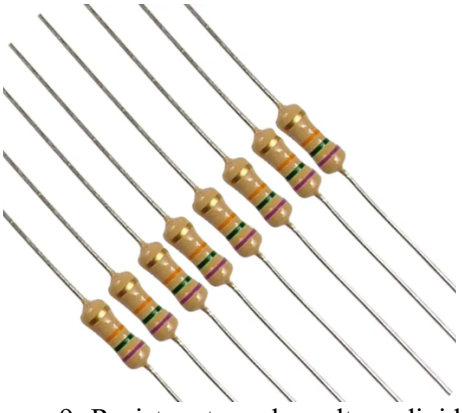


Figure 9: Resistors to make voltage divider

Figure 10: 30 A Current sensor

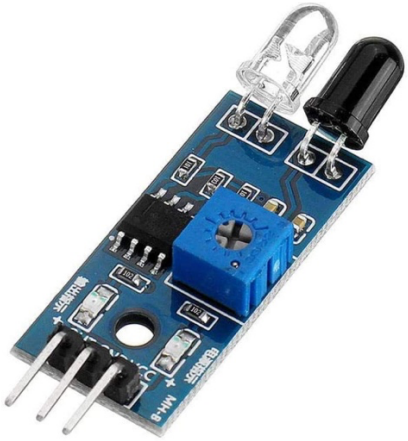


Figure 11**:** Infrared sensor

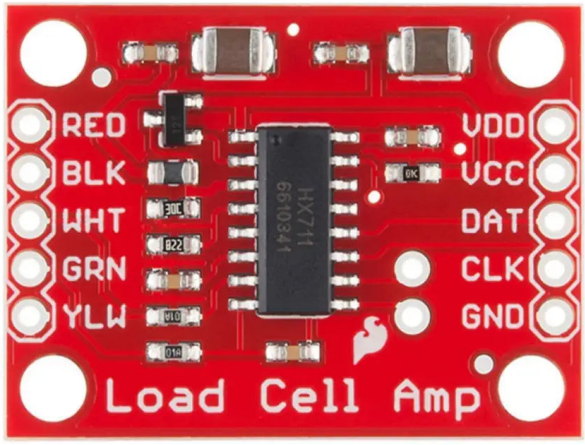


Figure 12**:** Load Cell/Torque Transducer Amplifier

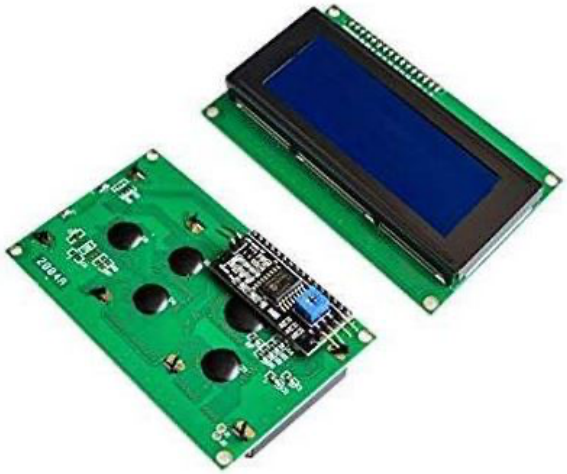


Figure 13**:** 4x20 Cell LCD Screen

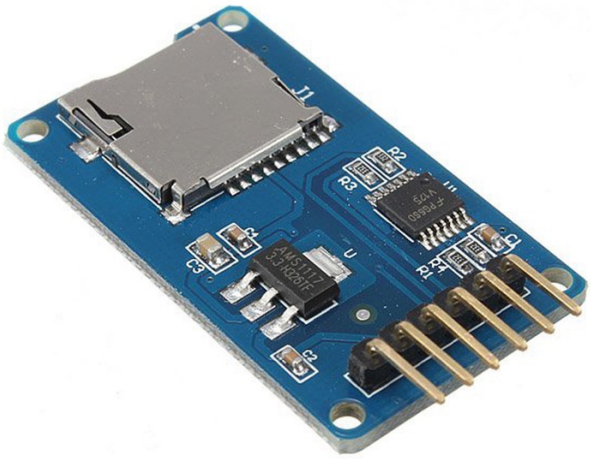


Figure 14: SD Card Reader

The schematic requires mostly soldering, as you see fit, and ribbon cable connectors. As noted in Figure 8, it is **required** to use a thicker gauge wire (i.e. 10 AWG – 12 AWG) between the rectifier, current sensor terminal blocks, and programmable load to avoid wires burning from high current.

After the schematic is soldered and connected, the Arduino IDE must be downloaded to upload the source code to the Arduino found here. The post-processing MATLAB source code can be found there too. To upload the code, connect your computer to the Arduino with the USB port to supply power and communicate with the Arduino. Then, upload the code through the Arduino IDE. If you see measurements being displayed after uploading, everything is set for testing.

When disassembling the DAQ system, it is recommended to salvage the sensors/components and make a new breadboard/protoboard setup for those sensors/components when planning to rebuild.

2.2. Operation

2.2.1 Initialization

If the DAQ system is already built, you may skip this section as it is intended for rebuilding this system.

Before using the DAQ system for testing, check for the values in Table 1. These values will be important to ensure accurate measurements

Table 1**:** Arduino Variable Constants

Variable	Description	Quantifying the Variable
Reference Voltage	Supply voltage from the Arduino in V	Use multimeter to measure voltage between the +5V and GND pins of Arduino
Voltage Divider Resistance 1 (High)	The resistors' resistance from the voltage divider in kΩ	Use multimeter to measure resistance between each resistor
Voltage Divider Resistance 2 (Low)		

Once you've quantified the variables, ensure to apply the values in ArduinoUtitlities.h and VoltageDivider.h from the DAQ system Arduino source code by changing the constant values. Do the same in Constants.m from the post-processing MATLAB source code.

2.2.2 Code Menu

Inside the Arduino source code, there exists a code menu for acquiring data into the SD card. This menu is shown in Table 2.

Table 2**:** Code menu

Code Number	Action	Description
[1]	Start recording	Displays and opens the file the data will be recorded in, asks for sample amount, and then writes the first sampled data
[2]	Continue recording	Writes the next sampled data into the file
[3]	Stop recording	Closes the file and displays what file the recorded data is in
[4]	Reset system	Resets the Arduino

This menu is meant to allow the users to take any sample amount for a given throttle input on the dyno. So, it allows different testing methods to be applied (i.e. take 10 samples at every 10% throttle input increment). Also, if a user enters a negative number, the DAQ system will continuously write data into the

file until stopped. You can use this menu by setting your baud rate to 9600 in the Arduino IDE and setting "NO LINE ENDING" for serial input. Once set, you can enter the numbers to perform the actions. Also, if you do not perform the stop recording action, or remove the SD card prematurely while recording, no data will be saved. Ensure you stop recording before taking out the SD card. Ensure to reset the Arduino if you take out or put in the SD card while Arduino is running.

2.2.3 Calibration

If the DAQ system is already built, you may skip this section as it is intended for rebuilding this system.

For the following measurement calibration, it is recommended to use the code menu to get as many data points as you want for comparing to the actual measurement. When getting raw measurements, ensure to set slope and intercept to 1 and 0, respectively, inside VoltageDividerSensor.h, CurrentSensor.h, and TorqueTransducerAmplifierSensor.h.

2.2.3.1. Voltage

By using a power supply connected to the DAQ system, start from 0 V to the highest voltage by any increment you see best fit. At each step, record the data from the DAQ system. Once each step was recorded from both the measured voltage and actual voltage. Apply those values in VoltageCalibrationData.xlsx in the post-processing MATLAB source code. Within the calibration data, obtain the calibration curve slope and intercept and apply it to VoltageDividerSensor.h from the DAQ system Arduino source code. Assuming you obtained the calibration curve slope and intercept via linear regression line on a graph in Excel, remove the graph so the MATLAB code can only see the data.

2.2.3.2. Current

By using a power supply connected to the DAQ system, start from 0 A to the highest current by any increment you see best fit. At each step, record the data from the DAQ system. Once each step was recorded from both the measured current and actual current. Apply those values in CurrentCalibrationData.xlsx in the post-processing MATLAB source code. Within the calibration data, obtain the calibration curve slope and intercept and apply it to CurrentSensor.h from the DAQ system Arduino source code. Assuming you obtained the calibration curve slope and intercept via linear regression line on a graph in Excel, remove the graph so the MATLAB code can only see the data.

2.2.3.3. Rotational Speed

Rotational speed can be erratic during testing due to non-ideal conditions from lighting and reflectivity. Remedying this includes a 2% error of the reading, which results in a max error around 115 RPM. No action is needed.

2.2.3.4. Torque

The moment arm supplied has notches with 50 mm increments. So, the max distance is 250 mm with respect to the centerline of the dyno motor. With the moment arm, attach it to the dyno as shown in Figure 15.

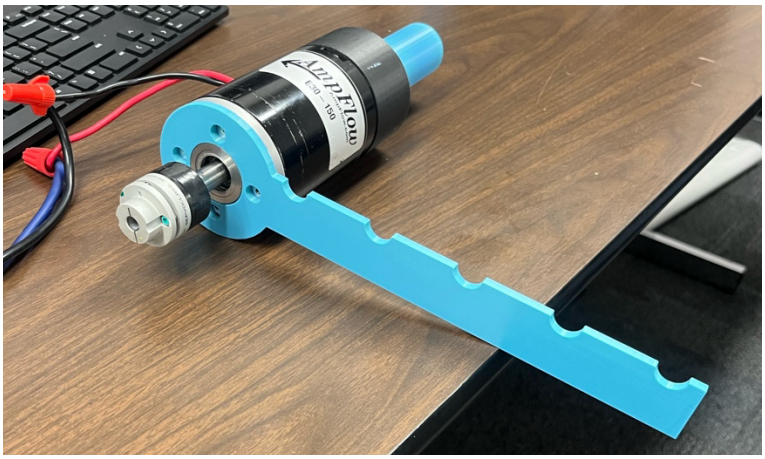


Figure 15**:** Moment arm on dyno

Start from the moment created from the moment arm and add that moment with the moments created from the applied known weights on the moment arm. From the DAQ system, record the data as the torque is applied. With the recorded data, use the digital torque output instead of the actual torque as the digital output will help get the actual torque. Store this in TorqueCalibrationData.xlsx from the post-processing MATLAB source code. Within the calibration data, obtain the calibration curve slope and intercept and apply it to TorqueTransducerAmplifierSensor.h from the DAQ system Arduino source code. Assuming you obtained the calibration curve slope and intercept via linear regression line on a graph in Excel, remove the graph so the MATLAB code can only see the data.

2.2.4 Testing

When testing with the dyno and DAQ system, it is recommended setting the external power supply driving the dyno at certain throttle inputs (i.e. 0%, 10%, 20%, …, 100%). With each throttle input, record the samples you want to take (i.e. 10 samples) with the code menu, then move to the next throttle input to record, and so on. You will do a no-load sweep to get Kv and a loaded sweep with a programmable load to get power from the generators. It is recommended to use the following loads: 2 Ω, 5 Ω, 10 Ω, 15 Ω, 20 Ω, 4.2 A, 4 A, 3 A, 2 A, and 1 A. With all these tests, it is **required** to use the same sample amount across all sweeps for the **same generator** due to the post-processing code workflow. Do **not** quickly change throttle input (i.e. going from 0% to 100% throttle input within a second). When planning to test for long periods, ensure to use a fan on the dyno to reduce overheating of the dyno and DAQ system.

2.2.5 Post-Processing

Once you've obtained no-load and loaded data, you'll be making a folder with your generator data as shown in Figure 16.

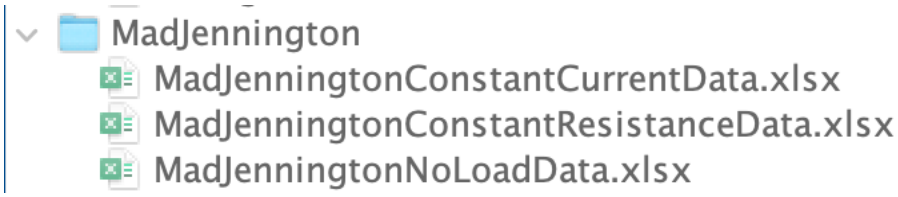


Figure 16**:** Example generator folder

Ensure your file/folder format are as follows:

- **Folders:** [Name of generator with title case].xlsx
- **Constant load data file:** [Name of generator with title case]Constant[Load type]Data.xlsx
- **No-load data file:** [Name of generator with title case]NoLoadData.xlsx

Ensure your constant load data files contain sheets with the title being the number of their load (i.e. "2 ohms", "5 ohms", "10 ohms", etc.). Within those sheets, ensure each contain their respective data from the DAQ system.

Ensure your no-load data file contains the no-load data from the DAQ system.

Once you've formatted your data, put the data folder in the folder where the post-processing MATLAB source code is. Also, the sample amount you took while testing, ensure to change that for each generator if they differ in sample amounts in Constants.m. Then in postProcessData.m, enter you're folder name and run the program to get Kv curve and power curves of your generator with error propagation.

2.3. Maintenance

When there is no generator hooked up to the dyno, ensure something holds it to reduce the stress applied on the extender due to the motor's heavy weight. Ensure no wires are being stretched since the wires going to the motor and torque transducer are long. The piece of tape on the shaft of the dyno motor is bound to wear after going through heat, so keep replacing the tape or apply black paint on it with a slit of metal for the infrared sensor to read.

2.4. Troubleshooting

The dyno does vibrate at times due to the coupler. To mitigate this, provide tension to the coupler attached to the generator to reduce the vibration. If any measurements start jittering between many values, ensure your wires have continuity of 2 Ω or less to ensure minimal resistance between the wired connections.