ME 4000

Engineering Design I

Assignment 6 – Proof-of-Concept Testing (/100 points)

| **Due date:** | See CANVAS for submission deadline and instructions |
| --- | --- |
| **Overview:** | The goal of Assignment 6 is to test the feasibility of the chosen design concept through proof-of-concept testing. |
| **Learning Objectives:** | After completing Assignment 6, students should be able to:   * Develop a proof-of-concept testing plan * Design and construct a prototype * Execute an experimental testing plan * Analyze results and formulate design conclusions |
| **Requirements:** | Assignment 6 will be a report with 5 sections:   1. Summarize the goal of your proof-of-concept testing and test question(s) in an “**Objective**” section (~ 1 paragraph) (/5 points) 2. Design a proof-of-concept prototype to use in your critical function test. The proof-of-concept prototype may consist of your entire device, or one or more components of the device, depending on your project. Write a “**Prototype Design**” section describing the design and your choice of prototype components and materials. Justify the design of the prototype with engineering analysis (calculations, modeling, etc.) (these could be calculations from Assignment 4, or new calculations). Include drawings or sketches that show your design. (/35 points) 3. Construct your prototype. Write a “**Prototype Development**” section of your report documenting prototype construction and assembly (include drawings, photographs, and/or videos of the device as appropriate). “Construct” in this case applies to physical prototypes, computational models, detailed CAD or CFD models, test rigs, etc. (/15 points) 4. Test the proof-of-concept prototype according to the experimental plan developed in step 1. Write a “**Results**” section of your report that summarizes the testing results, including figures and tables that clearly convey your experimental measurements. Include raw experimental data with the assignment submission, either in the main written section of the assignment or as a supporting appendix (whichever makes sense) (/25 points) 5. Analyze your proof-of-concept testing results in a “**Discussion**” section of your report. Be sure to answer the test question posed in step 1. Do the results support continued development of the chosen design concept, or should the selected design be re-evaluated? Justify your conclusions. *Relate your critical function performance to your design metrics.* What are the next steps?(/20 points) |
|  |  |
| **Note on report length:** | There is no length requirement or limit. However, the report should be **complete and concise**. Reports that not concise (i.e., have redundant text, results that are not appropriately summarized, results that are not relevant, etc.) will have points deducted. Reports that are missing important information will have points deducted. Use appendices as appropriate for supporting information or results. |

1. **Objective**

The goal of the quadruped robot’s proof-of-concept testing is to determine the feasibility of the quadruped’s walking cycle to reach our final goal of an autonomous, untethered robot. To do this, it must receive IMU data with low latency, execute accurate positional control, operate without overloading the Orange Pi’s processor, supply enough motor torque, and supply enough power. These parameters are outlined in Table 1. The testing will be done on a test stand comprising one robot leg, the Orange Pi single-board computer, and the ODrive brushless motor controller (BLMC) which will command the motors to go to a specific angular position using the ROS2 Humble Hawkbill development kit. These tests will answer if expanding the design to seven more motors and BLMCs will be feasible–specifically, is the Orange Pi powerful enough to execute control on 8 actuators, does a CAN bus have enough bandwidth for 8 actuators, how much power will need to be supplied to each actuator, what is the load bearing capability of these actuators, and how quickly can these actuators respond–as well as giving the team metrics on which an untethered power system can be designed.

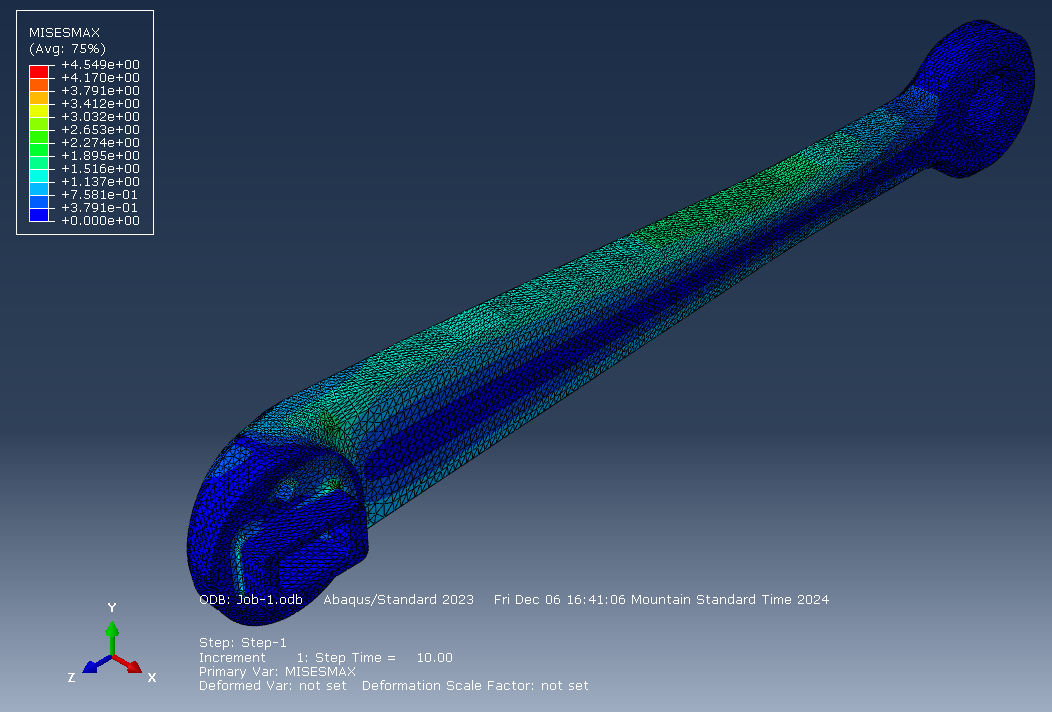
**Table 1.** A series of metrics to determine the feasibility of the quadruped control system.

| Design Specification | Test Specification | Success |
| --- | --- | --- |
| Response Accuracy | Position Control Resolution (Step Size) | <3° |
| Runtime | Power Draw (single motor) | <200 W |
| Response Speed | IMU Latency | <200 ms |
| Response Speed | CPU Load (single motor) | <12% |
| Response Speed | Total data transfer rate on CAN bus (single motor) | <1000 kbps |
| Strength | Stall torque (at end of gearing) | >1.83 Nm |

1. **Prototype Design**

The design of the prototype is a singular actuator and leg from the robot mounted to a test stand. The test stand is 3D printed from PLA. The O-Drive motor driver and Orange Pi 5 are mounted to the test stand and connected. The O-Drive software will demonstrate its motor control capability at a basic level as a stepping stone to creating walking software.

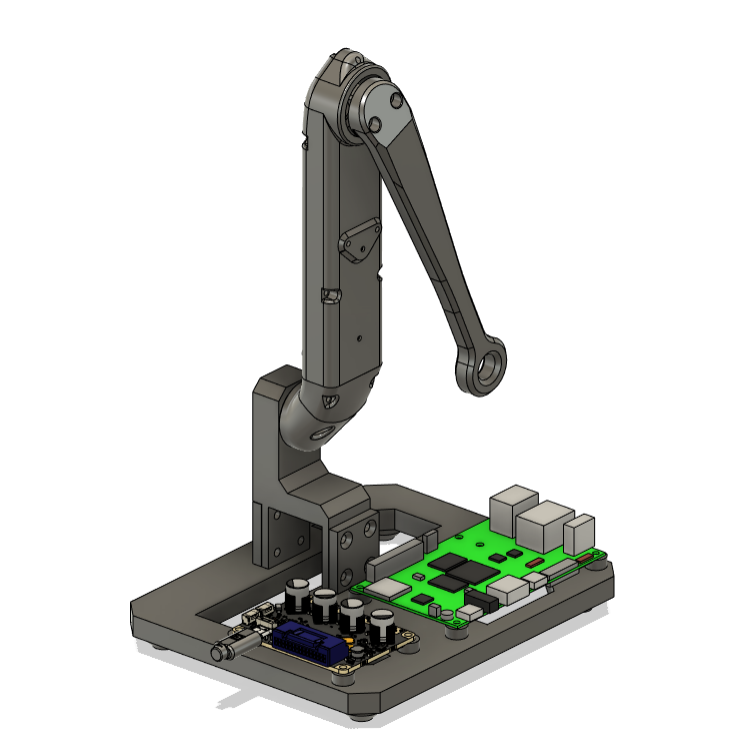
The only material difference is that the current 3D-printed leg is made of PLA and the final version will be made of CF-PLA. In addition to being a request from our sponsors, the reason PLA is used is to be able to rapidly test various leg shapes. In a static loading test, an assumption is made that the PLA is rigid enough to not deform before the motor stalls to ensure the motor limits are accurately validated. Given our stall limit of 1.83 N-m, we can verify the strength of the PLA structure by making sure any structure on the PLA leg can withstand that static load without deforming.



**Figure 1.** FEM Stress analysis of leg portion

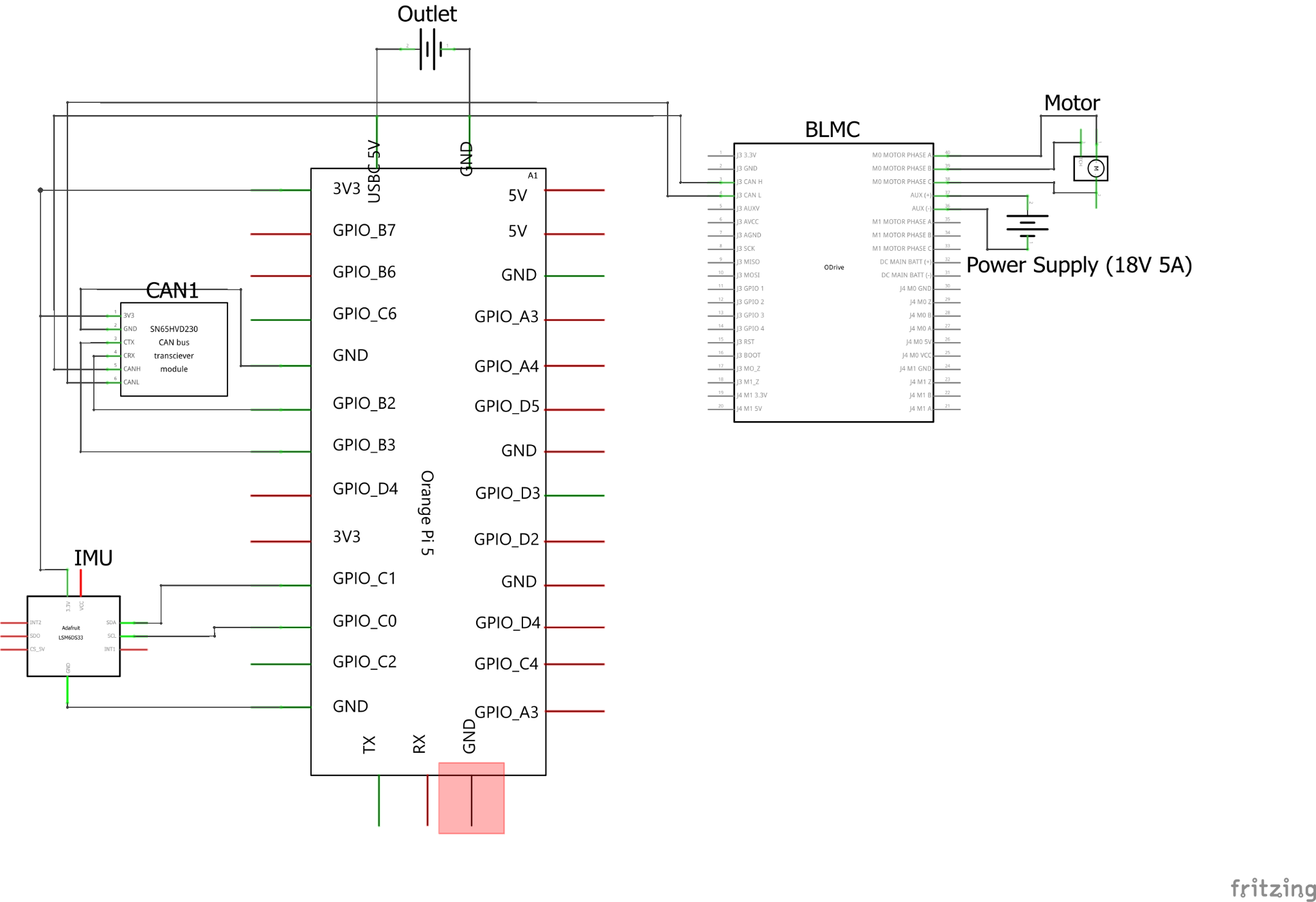
From the FEA analysis, as shown in Figure 1, we expect the stress to be concentrated around the bends with a maximum stress estimated by the model to be 4.55 MPa. From [1], we gather that the yield strength of our leg is 26.082 MPa. Our prototype will have a design factor of safety of 5.7, which is more than enough to withstand testing for motor stall torque.

Component-wise, the only difference is that components are mounted to a 3D-printed test stand instead of the robot body. The computer, motor, motor driver, wire type, and bearings are the same as those used in the planned final quadruped robot. The test stand was modeled in Fusion 360 and shown in Figure 2.



**Figure 2.** Shows the CAD model of the test stand. The Orange Pi (green) and the ODrive BLMC (darker grey to the left) are shown fixed to the base.

To replicate the robot prototype, the proof-of-concept setup uses the control area network (CAN) protocol to connect to the ODrive BLMC and the inter-integrated circuit (I2C) protocol to connect to the IMU. Figure 3 shows the power delivery and wiring paths taken to connect each component.



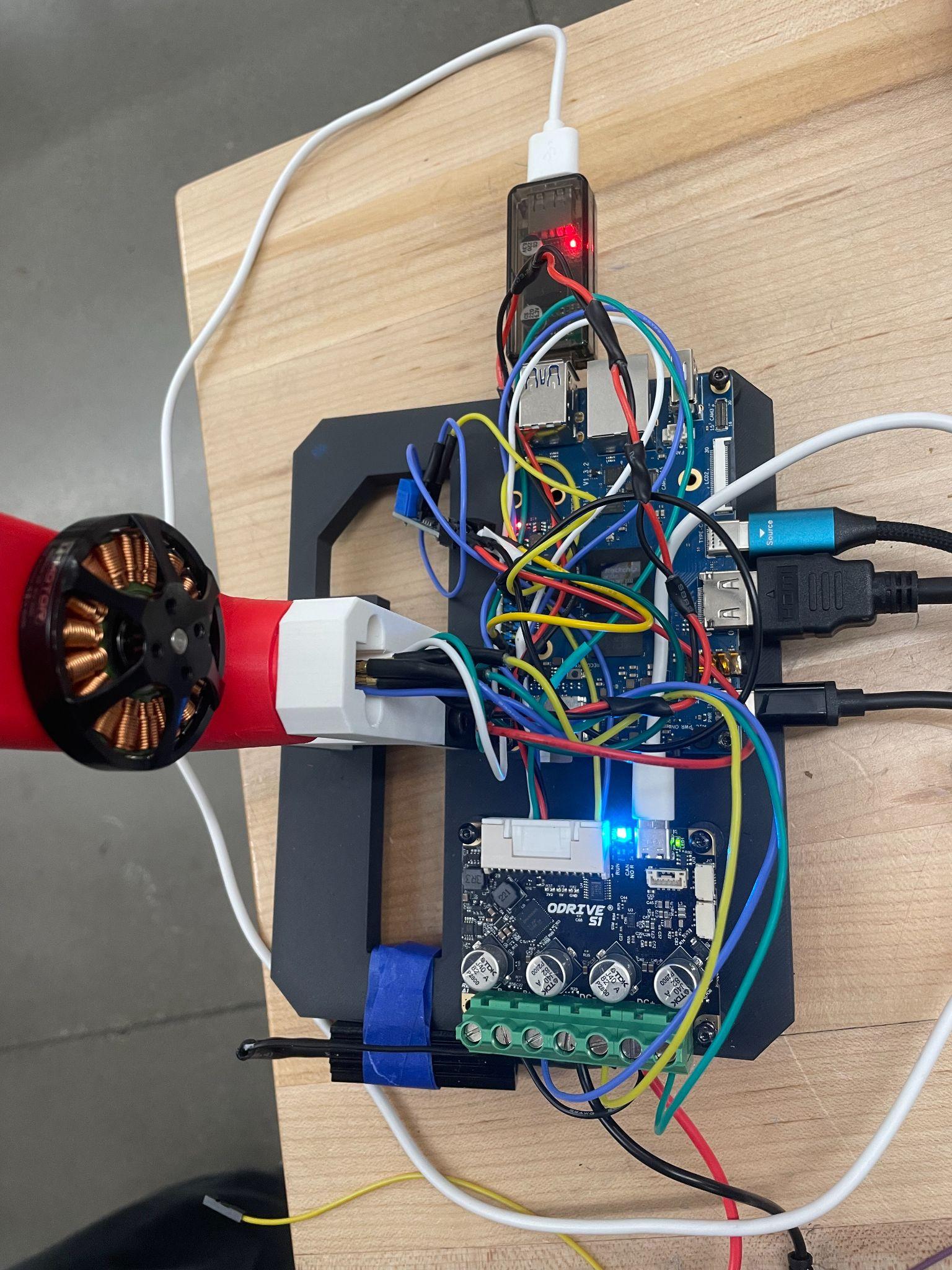
**Figure 3.** Electrical schematic of prototype setup

1. **Prototype Development**

The prototype test stand is printed according to the design seen in Figure 2 with one of the robot’s leg segments connected. Aside from the 3D-printed test stand itself, the components used in the test stand are the same as the final robot. Upon completion of the body prototype, the same parts are used for the standing version of the robot. The physical prototype test platform is shown in Figures 4 and 5. The robot test stand will have a body made with 3D-printed PLA for testing convenience before moving on to CF-PLA for the standing robot designs. The robot uses a CAN communication protocol as it’s able to accept and send commands on one data line by assigning commands and responses to a certain CAN ID corresponding to each BLMC, letting our computer differentiate between control boards with minimal pin usage. This reduces the usage of pins on the Orange Pi, allowing the IMU–attached to the I2C pins–and future components to be connected. Further, the CAN pins on the Orange Pi are connected through a CAN transceiver as seen in Figure 3 to ensure the data coming from the Orange Pi matches the data format coming from the BLMC. To complete the prototype testing, the computer must be able to connect to the BLMC and tell it to go to specific positions. This is done using the supported ODrive Python libraries in a ROS2 Humble node to ensure long-term support and easy modification in the future. To validate IMU usage, the prototype will move the leg according to the way the IMU is tilted. With all of these parts working, the team has high confidence the hardware will work as intended on the standing robot.

|  |  |
| --- | --- |

**Figure 4.** The prototype test stand with one leg attached actuated by the onboard ODrive BLMC and Orange Pi mini-computer.



**Figure 5.** Shows a close-up of the electronics with an Orange Pi mini-computer (top) and ODrive BLMC board (bottom)

1. **Results**

Metric 1: Angular Resolution

A custom bracket was designed and printed to hold an angular protractor in line with the actuator module. An angular protractor was installed and rigidly attached to the robot leg. Small increments were manually commanded to the motor resulting in a step change in the protractor reading. This test achieved step sizes of 0.05°.

|  |  |
| --- | --- |

**Figure 6: Custom bracket CAD model (left), angular protractor assembly (right)**

Metric 2: Power Draw

The O-Drive motor controller is capable of reading the current drawn through the motor and displaying it in the GUI. A weight equal to 25% of the anticipated weight of the robot was mounted to the end of the leg. The actuator was commanded to move back and forth under varied loads as if it were walking. Using the current readings and the power supply voltage a peak power draw was calculated according to equation (1).

(1)

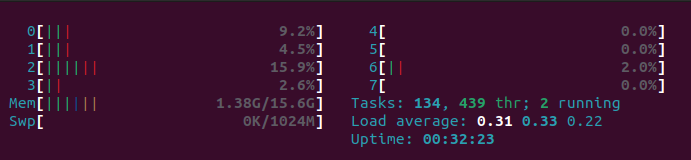
From the average current reading from the ODrive program, an average of 4.4 A was read at stall torque. This results in an approximate peak power draw of 80 W per leg at maximum capacity, which is representative of a standing motion. At half load, which is more representative of the walk cycle, an average of 2.9 A was measured yielding approximately 54 W of power draw. All tests were conducted at the 18 V power rating of our target batteries.

Metric 3: IMU Latency

In the quadruped chassis, the robot will need to frequently reference the IMU to determine the absolute rotation. The rotation influences the leg positions to shift the center of mass closer to the center and avoid tipping from an impulse or ground disturbance. The total read-write rate between the IMU and an actuator should be on the order of 200 ms as prescribed by the Open Dynamic Robot Initiative. This information can be determined first by reading the log output of the ROS2 control node, which outputs an average loop speed (combined reading and writing) of 600 ms. Since this was significantly higher than our metric, we began testing to determine why. Running just IMU filtering or just direct closed loop control yielded results in tens of milliseconds, but combining them created too much overhead. Further, running computations in a C++ node as opposed to the Python nodes yielded orders of magnitude greater computation speed than equivalent Python code.

Metric 4: CPU Load

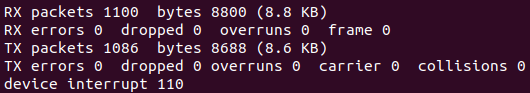
A simple closed-loop control script is run on the Orange Pi and motor driver board to simulate robot control. The CPU usage is averaged over 10 seconds using the Linux terminal. A single motor control process utilized approximately 4.2% of the normalized CPU.



**Figure 7: Linux terminal output of CPU usage.**

Metric 5: Data Transfer Rate on CAN Bus

The same closed-loop control script as before is run on the Orange Pi and motor driver board to simulate robot control. The total data transferred over the CAN bus since boot is then displayed and logged through the Linux terminal, as seen in Figure 8. Closed loop control runs continuously for the next 10 seconds after which it exits and displays the new total CAN bus usage–in this case, 10.2 kb on the RX pin and 9.7 kb on the TX pin. The difference between the total data transferred over both pins divided by 10 seconds yields the average data transfer rate of 0.25 kbps.



**Figure 8: Linux terminal output of CAN bus data transmission.**

Metric 6: Stall Torque

Approximation of the stall torque was achieved by adding weights in the form of 165g clamps at the end of the leg, while in the horizontal position, until it caused the motor to stall. The motor suffered a stall at 495g. The stall torque at 5 A (the limit of the lab power supply) was calculated according to equation (2).

(2)

Calculating torque according to Equation 2 results in an output torque of 0.714 Nm. When batteries are installed the current can be increased to 15 A resulting in an output torque of 2.142 Nm.

A summary of the proof-of-concept test can be found in Table 2. Most metrics are in acceptable parameters aside from the response speed from the IMU which is a limitation of the language used.

**Table 2.** Summary of Results from the Proof-of-Concept Test

| Design Specification | Test Specification | Test Result | Success |
| --- | --- | --- | --- |
| Response Accuracy | Position Control Resolution (Step Size) | 0.05° | <3° |
| Runtime | Power Draw (single motor) | 80 W | <200 W |
| Response Speed | IMU Latency | 600 ms | <200 ms |
| Response Speed | CPU Load (single motor) | 4.2% | <12% |
| Response Speed | Total data transfer rate on CAN bus (single motor) | 0.25 kbps | <1000 kbps |
| Strength | Stall torque (at end of gearing) | 2.14 Nm | >1.83 Nm |

1. **Discussion**

Overall, most of the metrics for feasibility were met or exceeded. Notably, the positional resolution requirement was exceeded by nearly two orders of magnitude. The power requirements for walking suggest that battery capacity can be reduced thus lowering weight and increasing efficiency. Though our IMU test was not able to meet the requirements, it did lead to the discovery of an important limitation with our strategy thus far that we will resolve moving forward. CPU load was about one-third of the maximum. This should leave room for other processes such as sensor interpretation and navigation algorithms. The CAN bus capacity is more than three orders of magnitude larger than the requirement suggesting that it will not be a limiting factor. Finally, the stall torque is sufficient, although it does cause concern that the motors will need to be nearly maxed out to lift the robot through the angles of greatest torque. This concern is somewhat rectified, however, by our ability to choose a smaller battery size. Based on the results of testing, the team will continue with minor changes to the chosen design and components.

The first change we will make is in response to the power requirements test. Since the peak draw for one motor was less than half that of our target, our choice of battery can be switched to a lower-capacity option. This has the advantage of reducing both the weight and cost of the robot. The former results in an increased capacity to carry additional components while the latter improves the attractiveness of our design for the end user. The second change we will make is in the software architecture. This is in response to the failure of the IMU test to meet our response time requirements. In testing each component, we found that the bottleneck in response time was not any of the hardware, but rather in the code. Either the IMU or the motor could respond within tens of milliseconds, far below our requirements, but putting them together in a Python node resulted in enough computational overhead that the best-case scenario if we optimize all our computations and eliminate extraneous code in the script, would be around 400 ms. However, when running tests on an equivalent C++ program, we were able to achieve speeds below 100 ms. Based on this, we will have to rework our software design to offload all computationally intensive and latency-sensitive work to C++ nodes and only use Python for lighter calculations or sensor readings. Outside of these two things, our design will remain unchanged and we can proceed with building the full robot.

Citations:

[1] S. R. Subramaniam, M. Samykano, S. K. Selvamani, W. K. Ngui, K. Kadirgama, K. Sudhakar, M. S. Idris. 2019. “Preliminary investigations of polylactic acid (PLA) properties.” *AIP Conf. Proc.*, 2059 *(1)*.<https://doi.org/10.1063/1.5085981>