ARM (Actuation via Real-Time Myoelectric Signals) ProsthEEsis

Final Report

EE Senior Design 2024-25, University of Notre Dame

K Blouch, Cassidy Chappuis, Owen Nettles, Madeline Prugh, Andrew Sovinski

Table of Contents

1 Introduction (about 3-5 pages)

The problem being solved.
High level description of your solution.
How well did your design meet your expectations? Does it do what you thought it would, is it as accurate as you thought it would be, etc?

e-NABLE is a global network of volunteers committed to combating the medical device inaccessibility gap by providing functional 3D-printed prosthetic devices at no charge to users. Over the last decade, this community has democratized access to low-cost, scalable, and customizable prosthetic devices through the development of open-source, 3D-printed prosthesis designs. The University of Notre Dame chapter of this network, e-NABLE ND, has laid a strong foundation for this work here on campus, focusing mainly on manufacturing scaled versions of existing, mechanical prosthesis designs and creating devices specific to certain users or tasks.

The ARM ProsthEEsis project seeks to build on e-NABLE ND's past projects by addressing specific challenges in the design and desired functionality of these devices, pushing beyond existing designs through the implementation of advanced features such as myoelectric control. In addition to myoelectric integration, we intend to improve the overall actuation design to better accommodate typical functionalities while minimizing the weight and cost of the device. The ultimate goal of the project is to provide e-NABLE ND, and potentially the greater e-NABLE community, with a myoelectric prosthesis design which can be replicated and adapted to different users with a maximum degree of functionality at a minimal production cost.

Problem Statement:

For children in search of prosthetic devices, the current market presents several challenges. Traditional prostheses are costly and require frequent replacement as children grow. e-NABLE ND, an organization on campus dedicated to creating affordable, 3D-printed prosthetics, provides a solution to this problem. However, their designs are predominantly mechanical, which limits functionality for users who require more advanced, ergonomic solutions. These devices rely on body-powered mechanisms, which can require substantial effort to actuate. For example, elbow-actuated designs require the user to bend their arm to create a fist, limiting their ability to reach or grasp objects naturally. Additionally, the effort needed to keep a fist closed can lead to discomfort and fatigue, making daily tasks cumbersome and reducing the overall functionality of the prosthetic.

Solution Overview:

A promising alternative is the development of a myoelectric prosthesis that uses electromyography (EMG) sensors to detect muscle signals, interpret them, and use these to control the hand position.

Project Success:

2 Detailed System Requirements (several pages)

2.1 Overall System Requirements

The prosthetic device must provide intuitive, reliable, and comfortable control of a prosthetic hand by utilizing electromyographic (EMG) signals captured from the user's residual limb. It must operate continuously for at least a standard workday, ideally for more than eight hours, on a single battery charge. The system shall emphasize user safety, ease of use, accessibility, and long-term durability. All subsystems, including EMG acquisition, power management, motor control, mechanical structure, and user interface, should function cohesively to deliver a seamless user experience.

Component compatibility across mechanical, electrical, and software domains is essential. The device must be lightweight and robust enough for daily use while also allowing easy maintenance and servicing by the user without specialized technical knowledge. Above all, the prosthetic must meet the fundamental goal of restoring functional hand movements to the user in a manner that is natural, dependable, and efficient.

Additionally, the final design along with any and all design decisions, ideas, relevant files, etc. should be well-documented in a manner easy for the non-electrical engineer to understand, so that eNABLE ND can build upon and implement this project design for real users in the future.

2.2 EMG Signal Acquisition & Processing Requirements

The system must be capable of reliably detecting electromyographic (EMG) signals generated by voluntary muscle contractions in the user's residual limb. Surface electrodes shall be positioned to maintain consistent contact with the skin, ensuring signal stability without impeding socket fit or user comfort.

Following acquisition, the raw EMG signal must be processed through a series of hardware and software processing steps. Initially, the hardware must isolate the differential signal of the two input electrodes, and amplify the 0.1-1mV signal to a level suitable for further processing. After amplification, a bandpass filter must be applied to remove unwanted noise outside the typical EMG frequency range, particularly filtering out low-frequency motion artifacts and high-frequency electrical noise. Subsequently, an envelope detector must be used to extract the overall energy of the muscle signal over time, producing a smooth signal suitable for digital analysis.

Upon digitization, the processed signal must be analyzed in real time on the microcontroller using software libraries to implement additional noise filtering. The software must determine the presence or absence of meaningful EMG activity, enabling appropriate control signals to be sent to the mechanical actuation subsystem.

2.3 Power Subsystem Requirements

The prosthetic must be powered in a manner that supports continuous operation for a reasonable amount of time under typical usage conditions. The power system must be built around batteries that are safely removable, allowing users to quickly swap batteries if necessary without requiring specialized tools or technical expertise.

To ensure operational safety and to maximize the longevity of the batteries, appropriate charging and discharging management protocols must be implemented. This includes protections against overcharging, over-discharging, and thermal damage. The power distribution architecture must account for and meet the demands of all integrated subsystems, including the microcontroller, signal processing circuits, motors, and any additional electronics.

Accessibility considerations must ensure that the battery compartment is easily reachable and manageable by the user, allowing for efficient daily maintenance routines.

2.4 Motor Subsystem Requirements

The prosthetic hand must utilize a tendon-based actuation system, relying on strong and durable materials such as high-strength string or fishing line. This material must withstand repeated mechanical stresses involved in opening and closing all five fingers during regular daily use, maintaining consistent performance without fraying, snapping, or excessive stretching.

Motor selection must prioritize performance and reliability. A DC motor will be employed, featuring a spool mechanism connected to its shaft to securely manage the wrapping and unwrapping of the ‘tendons’ during actuation. The motor should reliably turn both directions to close and/or open the hand as the user desires. The motor must be capable of receiving command signals from the microcontroller with minimal latency.

2.5 Hand & Socket Design Requirements

The mechanical design of both the hand, forearm, and socket must prioritize functionality, structural integrity, and user comfort. The prosthetic hand must incorporate articulated joints that allow all five fingers to open and close effectively. Internal tunnels or pathways within each finger must be included to properly guide and protect the ‘tendon’ system, ensuring efficient force transmission and minimizing wear.

The arm socket must serve as a secure housing for all major components, including the servo motors, batteries, microcontroller, and any associated printed circuit boards (PCBs). The arrangement of these components within the socket must consider thermal management, mass distribution, and ergonomic comfort.

The battery must be located in a manner that allows easy user access for charging or replacement, without requiring removal of the socket from the limb.

Any controls such as buttons, switches, or sensors that facilitate operation, calibration, or locking functions must be positioned to be easily accessible by the user's intact hand. Additionally, the entire mechanical assembly must be sufficiently lightweight to permit all-day wear without causing fatigue or discomfort.

While the goal is to successfully hold objects which may be useful in day-to-day tasks, the focus of this project is primarily to assist eNABLE ND with the electrical design, applying electrical engineering skill sets which the club has reported difficulty with in the past.

2.6 User Interface Requirements

The user interface must support straightforward and intuitive operation of the prosthetic device. Electrode placement must be carefully designed to ensure that electrodes do not interfere with the connection between the user's residual limb and the socket interior, while simultaneously ensuring strong and stable EMG signal detection. Strategic placement will help to minimize signal noise and maximize the system's detection accuracy. The arm should be able to stay in a position without constant input from the user, minimizing user fatigue.

A calibration process must be incorporated to tailor the device's EMG signal detection thresholds to individual users. Calibration must be initiated either automatically at startup or through a simple user-initiated command. The system must guide the user through a series of clear and simple steps that collect resting and active EMG signal data. This collected data must then be used to dynamically adjust the detection thresholds, ensuring optimal sensitivity and reliability throughout daily use.

Additionally, user manuals should be created for both the end-user and eNABLE ND. The end-user manual should include information regarding calibration, EMG sensor placement, battery information regarding recharging or replacement, and design information relevant to non-technical maintenance. The eNABLE user manual should include information relevant to those wishing to build upon the existing design (e.g., make updates to the mechanical housing), assembling the arm, and more technical maintenance procedures.

3 Detailed Project Description (20+ pages)

3.1 System Theory of Operation

This section will summarize a high level overview of the system. The overall operation of the system is when the user flexes the targeted muscle, the arm prosthesis will actuate, causing the hand to open or close. Electrodes are placed on the user's muscle, which are connected to a hardware processing circuit on the PCB. The circuit normalizes, filters, and amplifies the signal to optimize it for ADC conversion by the microcontroller. Once the signal reaches the microcontroller, it is further filtered in software. The MCU ultimately outputs a boolean, determining whether the muscle is currently contracted or not. The user is able to calibrate the prosthesis, tuning it to function reliably with their unique electrode placement and level of noise.

Based on the MCU output, the MCU will actuate a motor placed within the 3D arm, which opens or closes the hand. To adequately power the motor, the supplied voltage is split into two lines, 8V for the motor, and 3V3 for the rest of the hardware. The 3D printed arm uses flexible resin for the finger joints, allowing for smoother hand movements. The PCB and motor are contained with the 3D printed forearm, with the batteries placed in an external holder for easier user access.

3.2 System Block Diagram

Figure XX shows the overall system block diagram, illustrating how the system works from a high-level overview. The diagram shows the system divided into subsystems, as well as the interfaces between the subsystems

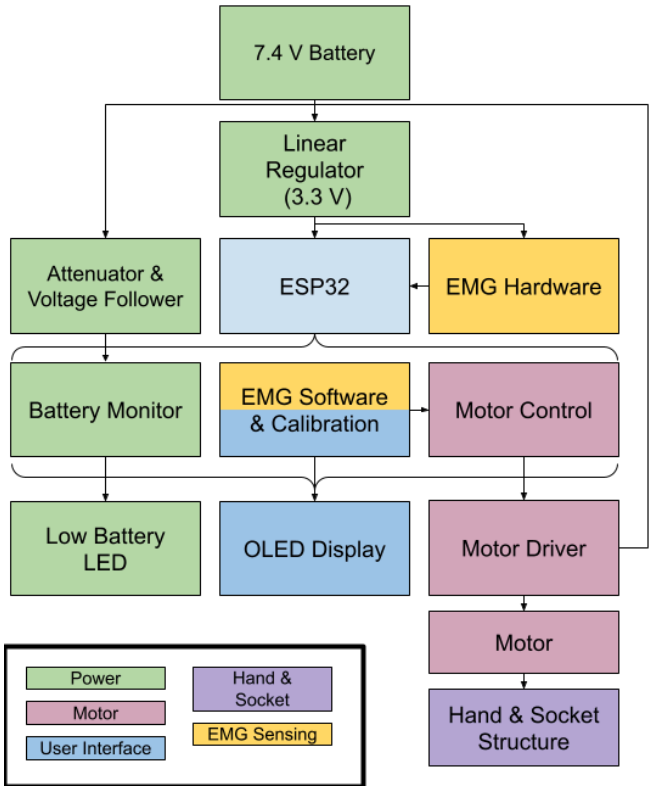

Figure XX. System Block Diagram

3.3 [Subsystem Design Criteria]

Subsystem requirements. These are the requirements that this particular subsystem must meet. (The combination of all of the subsystem and interface requirements must encompass the overall system requirements.)

This should include a schematic (hardware) or flow chart (software) for the subsystem, the function of the subsystem, the interfaces to other subsystems, etc. **There should also be information about why you made the engineering decisions that you did – choice of components, kind of interface, choice of programming language, etc.** Schematics and software need to be described in addition to having a schematic or a code listing. For hardware, you should describe how your circuit works (for example, how your amplifier circuit and filters are designed/operate); for software, you should describe the overall flow of the code (is it interrupt driven, state diagrams, etc.)
If there are communications protocols involved in this subsystem, description and state transition diagrams should be included.
Subsystem Testing. Described how this subsystem was tested to ensure functionality

3.4 Detailed Operation of Electromyogram (EMG) Sensing Subsystem

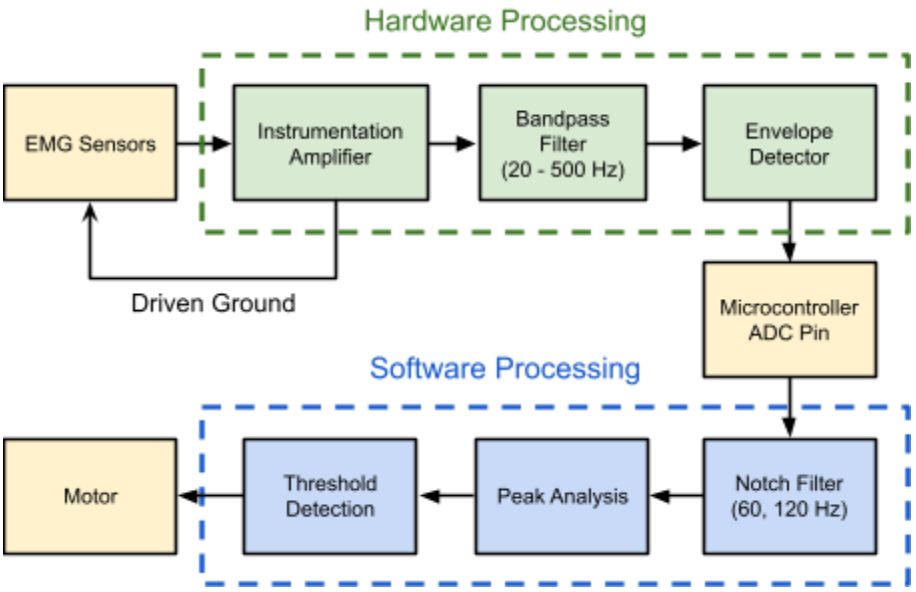

Figure XX. EMG Sensing Subsystem Overview

Requirements

The requirements of the EMG Sensing subsystem are:

1. Acquire an EMG signal from the user's muscle
2. Process the EMG signal in hardware to insure the input to the ESP32 is at a high resolution and within the specifications of the ADC
    - ➢ Apply filters for noise
    - ➢ Amplify the signal to improve ADC resolution
    - ➢ Smooth the signal for easier ADC sampling
3. Process the EMG signal in software to output a binary decision of whether the muscle is contracted or not
    - ➢ Apply a notch filter to remove 60 Hz noise and harmonics
    - ➢ Implement averaging or signal analysis to determine whether a signal is present
    - ➢ Allow for an adjustable detection threshold for user calibration

Hardware

The usable frequency of EMG signals ranges from 20-500 Hz, so the instrumentation is very sensitive to power line noise at 60 Hz, in addition to electrocardiogram (ECG/EKG) interference around 60 Hz. The energy of the EMG signal is not evenly spread throughout the frequency band, and is instead concentrated towards lower frequencies. Additionally, EMG signals are at very low amplitudes, a maximum of 0.1-1mV when using skin electrodes. The majority of hardware processing in the EMG Sensing

subsystem aims to minimize noise and amplify the desired signal before it reaches the microcontroller. Initial breadboard versions of the subsystem were unreliable and very sensitive to external noise, demonstrating that minimizing wire and trace lengths should be a key factor in component selection and PCB design. **Figure XX** is the full schematic for the EMG sensing subsystem.

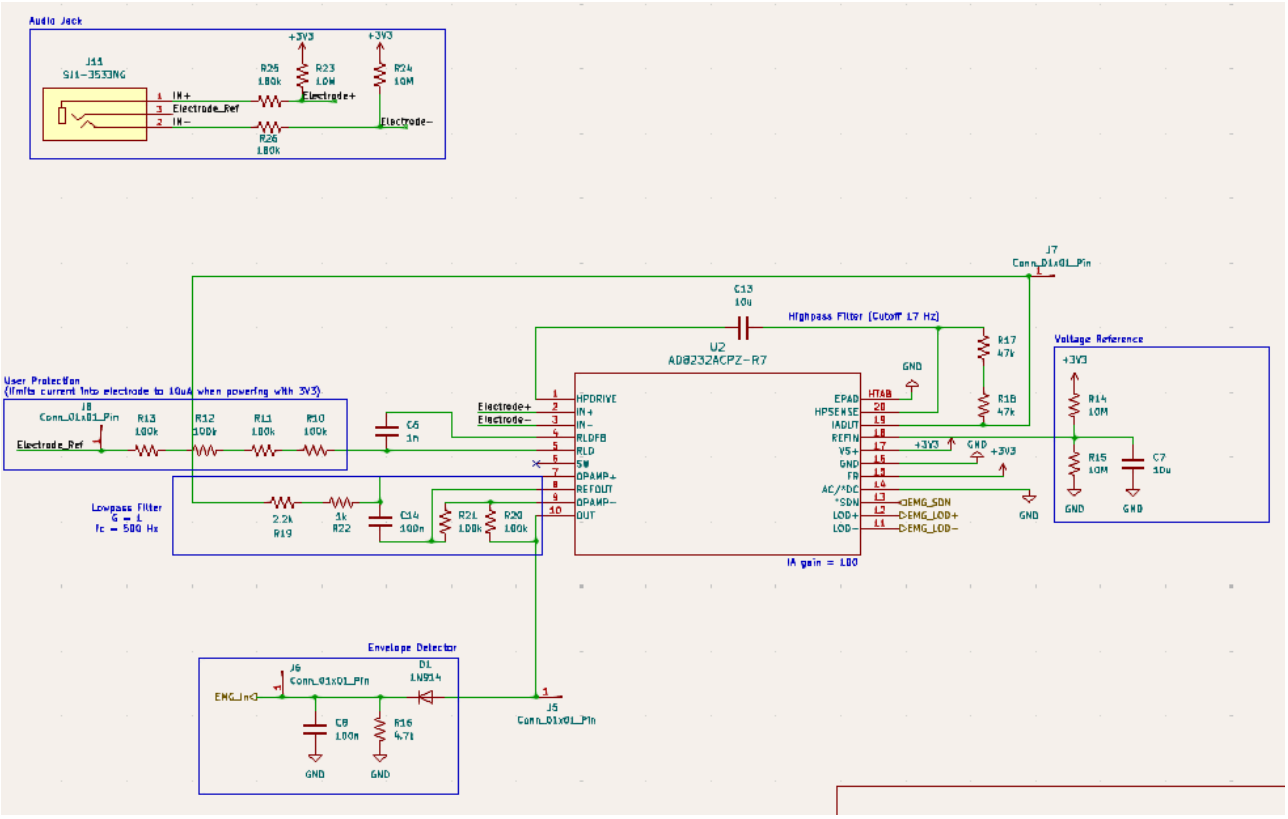

Figure XX. Full Schematic of EMG Sensing Subsystem

The following subsections describe the full hardware design of the EMG Sensing Subsystem. This subsystem utilizes the AD8232, an integrated circuit for heart rate monitoring. The AD8232 was selected because it contains all the necessary signal processing components in a compact, low-cost package. A single chip is only $6, and contains an instrumentation amplifier, driven ground circuit, leads off detection, and extra op-amps to construct a bandpass filter. Though the chip was created for ECG processing, the passive components to set frequency cutoffs are external to the chip, allowing us to tune the processing for EMG frequencies.

a. Electrodes

The standard configuration for detecting biopotential signals is to use a minimum of three electrodes, two for differential signal acquisition, and one as a driven ground. EMG sensing can also be implemented with a two electrode configuration, but the signal will be more noisy and less stable. Electrodes are relatively standard, with minimal differences between different products. Note that ECG electrodes can also be used for EMG due to the overlap in frequency. This

project utilized Covidien 31050522 electrodes, pictured in **Figure XX**, because they were the most available. The electrodes have an adhesive side, with a metal pad coated in an electrolyte jelly in the middle to improve the connection to the user. On the other side, the electrodes have a snap clip.

Figure XX. Covidien 31050522 Electrodes

The two signal electrodes should be placed about 1 cm apart along the direction of the muscle fiber, as shown in **Figure XX**. The ground electrode should be placed on the opposite side of the limb, preferably close to a bone to minimize noise. The three electrodes are connected via snap clips to a three-wire cable, which connects to the board through a 3-pin audio jack.

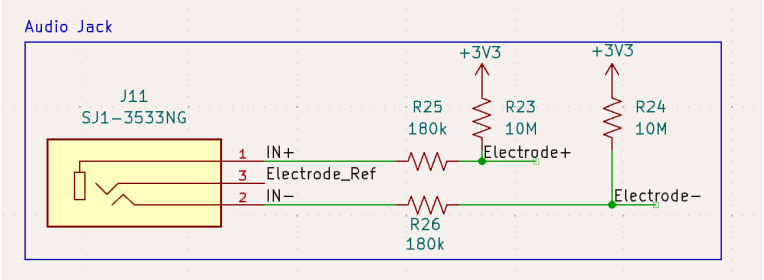

Figure XX. Electrode Input Schematic

At the input terminals of the audio jack, shown in **Figure XX**, there are 180k resistors, recommended by the chip manufacturer to protect the user from fault conditions.

The electrode connection to the user has high impedance, making it challenging to adequately ground the user. As shown in **Figure XX**, the input electrodes are biased to 3V3, which helps to ensure the input signal is operating in the same voltage range as the PCB. The 3V3 bias also allows for “leads off detection” where the AD8232 chip detects if the input electrodes are appropriately connected to the user. Pin 11 (LOD-) of the AD8232 will be high when the IN- electrode is disconnected and is connected to GPIO 7 on the MCU. Similarly, Pin 12 (LOD+) of the AD8232 will be high when the IN+ electrode is disconnected and is connected to GPIO 6 on the MCU. The leads off detection feature is configured in the PCB hardware, but is not currently implemented in the project software as it wasn’t a high priority feature.

b. Driven Ground

In a three electrode EMG setup, the third electrode is used to ground the user. However, the ground electrode cannot be directly connected to the ground plane of the PCB. Due to the high impedance of the electrode-user connection, there will be a voltage difference across the ground connection, which will drift over time. If the user is not adequately grounded, the input EMG signal may drift outside the operating range of the instrumentation amplifier within the AD8232, creating an unreliable output. Additionally, directly connecting the user to the board without a protection circuit could be dangerous for the user if any hardware faults occur, as current could travel back through the electrode into the user.

Our design utilizes the third electrode as a driven ground, also referred to as an active ground or a Right-Leg Drive (RLD) circuit. The RLD terminology originates from ECG configurations, where the ground electrode is typically placed on the patient's right leg, which is the furthest body part from the patient's heart. The RLD circuit takes the common-mode signal from the instrumentation amplifier, which is the noise common to both input electrodes, and feeds it back into the user's body through the third electrode, reducing the noise variations within the user.

For user safety, the output of the right-leg drive circuit has a 400k resistor, which limits the maximum current to the user to 8.25uA when the chip is powered by 3V3.

The driven ground circuit also includes a 1 nF capacitor, which forms an integrator when connected between the RLD FB and RLD pins on the AD8232. The capacitor value can be varied to balance gain and noise rejection, but 1 nF is optimal to reject noise in the range of 50-60 Hz.

c. Instrumentation Amplifier

The signal from the two input electrodes are passed through an instrumentation amplifier, which outputs the difference between the two signals. Because the two electrodes are at different distances along the muscle fiber, the EMG signal will be contained within the differential signal, while the common-mode signal is common noise. The instrumentation amplifier is within the AD8232, with the differential signal output on pin IAOUT. It is configured with 100x gain. **Figure XX** depicts the signal at the output of the instrumentation amplifier.

Figure XX. Relaxed vs. Flexed Signal at Output of Instrumentation Amplifier

d. Reference Buffer

The AD8232 allows the configuration of an internal reference buffer, shown in **Figure XX**, enabling all the op-amps to receive a positive and negative voltage input from the supplied system power. As a result, the output of the bandpass filter has a DC bias of half of the supply voltage. The AD8232 datasheet recommends using high resistor values, specifically 10 MΩ, to minimize the power consumption of the reference buffer voltage divider. However, higher resistor values also increase the likelihood of interference at the input of the reference buffer.

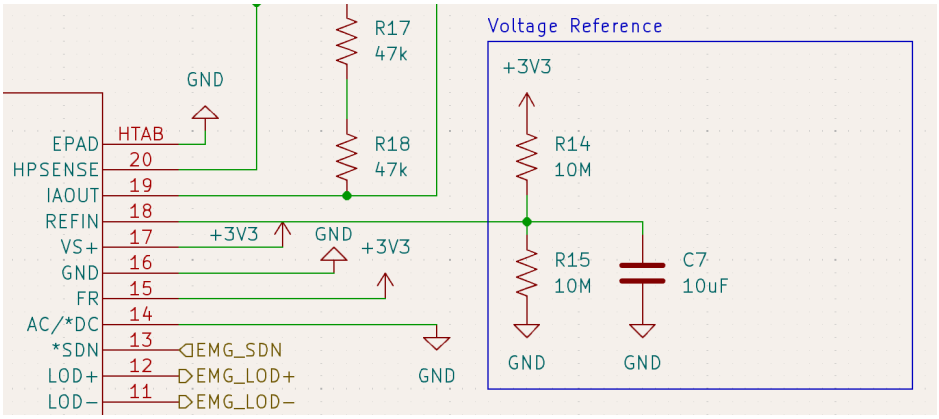

Figure XX. Input to Reference Buffer

To minimize noise, the AD8232 datasheet advises placing the resistors close to each other and the REFIN terminal on the PCB, in addition to adding a capacitor for additional filtering. The proximity of the voltage divider components to the AD8232 is shown in **Figure XX**. A higher capacitor value improves the noise filtering capabilities, but also increases the buffer's settling time after the chip is turned on. The settling time can be estimated using the formula below:

$$t_{settle} = 5 \times \left(\frac{R_{14}R_{15}C_7}{R_{14}+R_{15}}\right)$$

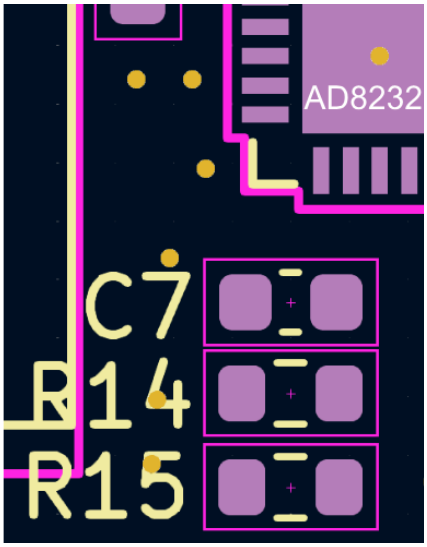

Figure XX. PCB Layout of Reference Buffer Components

e. Bandpass Filter

To filter baseband and higher frequency noise, the design applies a bandpass filter to the differential signal, with a passband range of 17-500 Hz. These frequencies were selected based on the usable frequency range of EMG signals, which is 20-500 Hz. The op-amps used in the bandpass filter are internal to the AD8232.

The cutoff of the highpass filter is set by the following equation:

$$f_{-3dB} = \frac{100}{2\pi RC} Hz$$

The passive components utilized in our design are pictured in **Figure XX**. Note that due to the internal circuitry of the AD8232 chip, the cutoff frequency is affected by the internal 100x gain of the instrumentation amplifier. The AD8232 also allows the implementation of higher order high-pass filters, which were not used in this design. The purpose of the high pass filter is primarily to filter out noise at baseband.

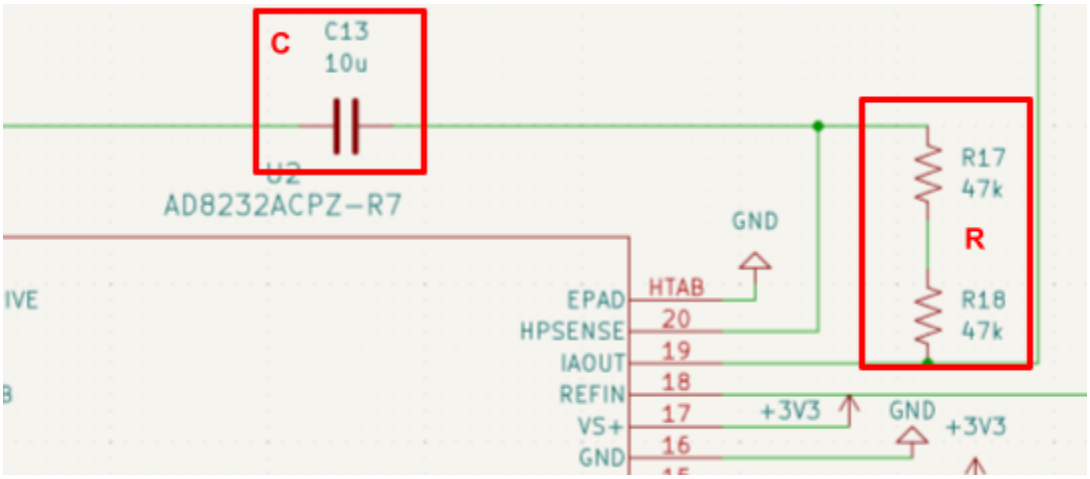

Figure XX. Passive Components of High-Pass Filter

The lowpass filter cutoff is set by the passive components pictured in **Figure XX**, based on the following equation:

$$f_{-3dB} = \frac{1}{2\pi RC} Hz$$

This filter can also be used to amplify the signal, based on the following equation:

$$G = (1 + \frac{R_2}{R_1})$$

For our design, the internal 100x gain of the instrumentation amplifier provided sufficient ADC resolution for reliable EMG detection. Note that the maximum input signal to the ESP32-S3 cannot exceed the specifications of the ADC pin, which is rated for a voltage range of 0-3V3.

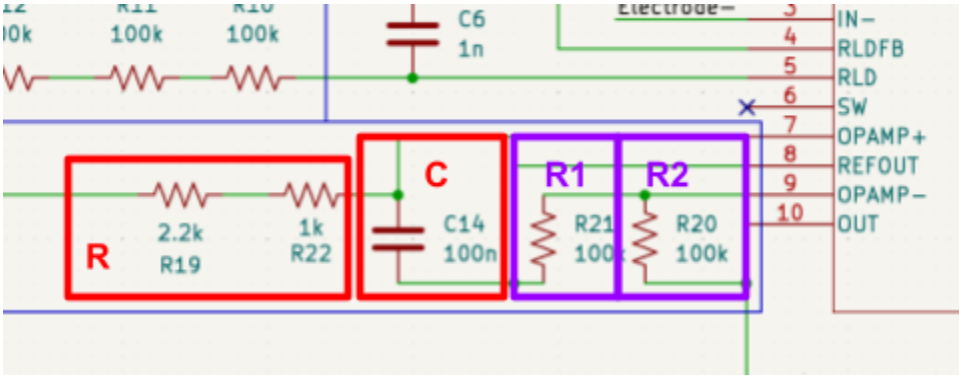

Figure XX. Passive Components in Low-Pass Filter

If a higher order filter is deemed necessary for further product improvements, the AD8232 datasheet provides specifications for the design of a Sallen-Key filter, which would provide a sharper roll off. As previously stated, the majority of energy within the EMG band is centered around the lower frequency range, and the primary noise is at 60 Hz, so the high pass filter is primarily for extraneous

noise, and it is not necessary to have a sharp cutoff for high frequencies. **Figure XX** shows the signal at the output of the low-pass filter.

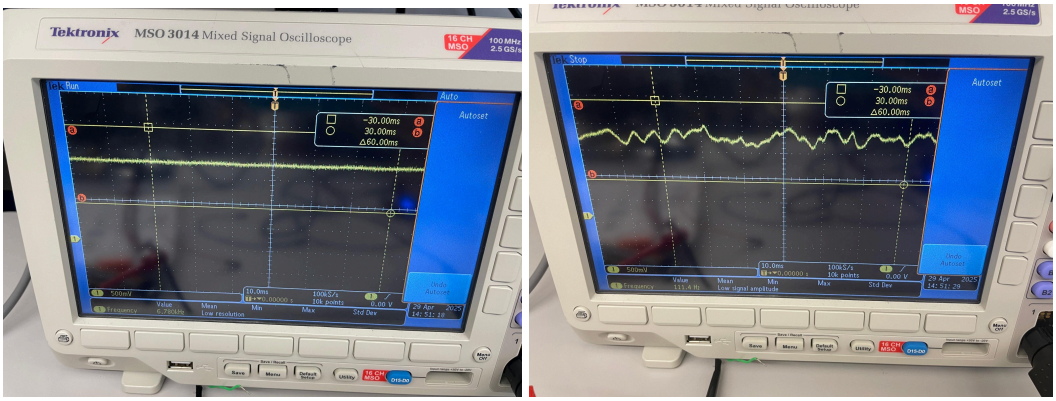

Figure XX. Relaxed vs. Flexed Signal at Output of Low-Pass Filter

f. Envelope Detector

The envelope detector implemented in the PCB design is not fully functioning, but this section describes the ideal behavior of the envelope detector. The envelope detector ideally smooths the output of the bandpass filter, reducing the required sampling rate at the microcontroller. The diode in the envelope detector rectifies the signal, filtering out any negative elements to ensure the signal stays within the range of the ESP32 ADC pin, which is 0-3V3. The time constant of the envelope detector is calculated using the following equation:

$$\tau = RC = (100\,nF)(4.7\,k\Omega) = 0.47\,ms$$

The greater the time constant, the smoother the output signal will be. However, increasing the time constant too much could cause the desired signal to be lost because the detector sensitivity is too low.

In the current design, pictured in **Figure XX**, the signal input to the envelope detector has a DC bias and is centered around 1.8V. Additionally, the time constant is not properly tuned to the desired response. The envelope detector in the PCB circuit is acting as a second low pass filter with a cutoff frequency calculated below:

$$f_c = \frac{1}{2\pi RC} = \frac{1}{2\pi(4.7\,k\Omega)(100\,nF)} = 338.63\,Hz$$

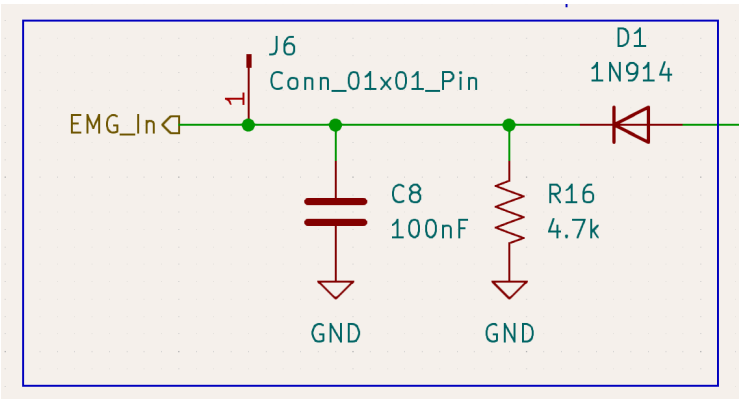

Figure XX. Envelope Detector

Due to the misconfigured envelope detector, some of the higher frequency components of the signal are lost. However, as previously mentioned, the energy of an EMG signal is concentrated to the lower frequencies, so the impact is minimal. The EMG processing software is tuned to function reliably with the existing envelope detector, so no changes were made to the final product. In future revisions, the envelope detector could effectively be removed to save space on the PCB. **Figure XX** shows the signal at the output of the envelope detector, which is the input to the ADC pin of the MCU.

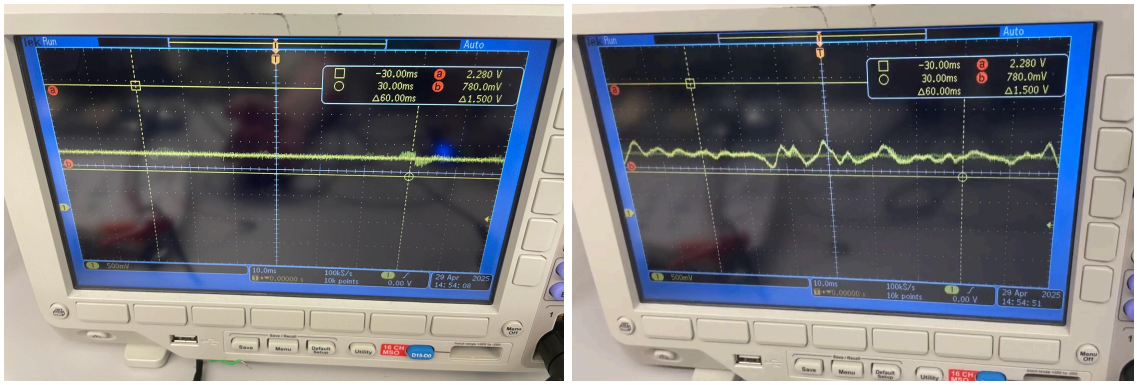

Figure XX. Relaxed vs. Flexed Signal at Output of Envelope Detector

Software

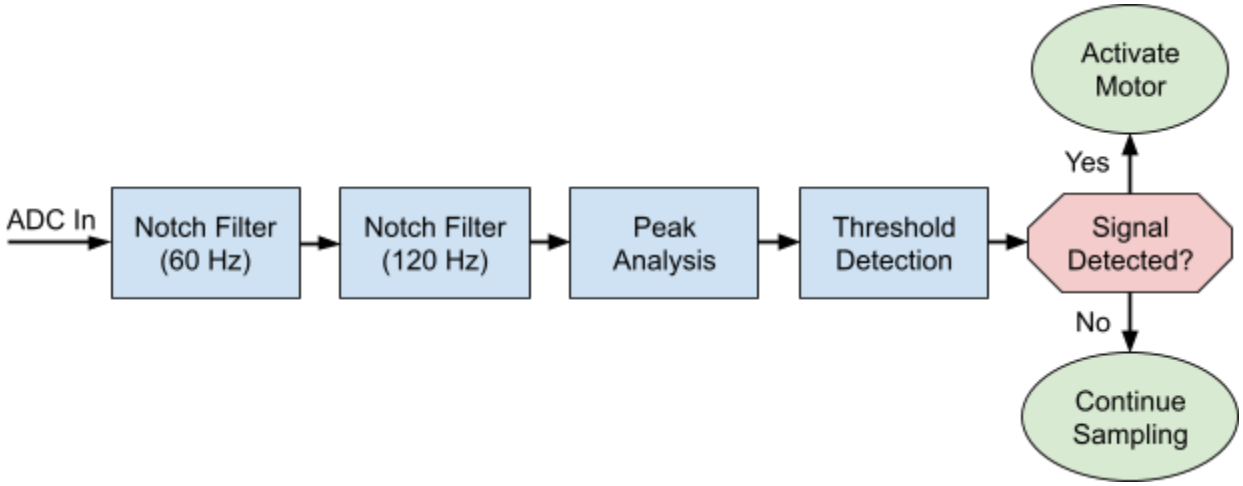

Figure XX. Software Flowchart for EMG Sensing Subsystem

Each sample from the ADC pin of the microcontroller is passed through two notch filters, one at 60 Hz, and the second at the harmonic, 120 Hz. We utilized the Arduino Filters library (documentation) because it has more extensive documentation and examples, and is more user-friendly than the ESP-DSP library. The filtered input signal is analyzed in time blocks of 200 ms. The software was initially designed to average the signal within each time increment, but due to the DC bias on the input signal, the "negative" peaks are not filtered by the envelope detector. As a result, when muscle is contracted, the input signal has higher maximums and lower minimums, which average out to the same as a non-contracted signal input. Instead of averaging the time increment, the software analyzes the maximum signal amplitudes within each 200 ms block, and compares it to a threshold set by the User Interface software. The flow chart in **Figure XX** describes the EMG Sensing software.

The software was tested by printing the digitized value of the ADC signal, and observing how the value changed under different conditions. When implementing the filter, both the pre and post filter values were printed, and we observed a greater difference between the flexed and relaxed signals in the filtered values. The overall software analysis was tested in a variety of rooms to confirm reliability under different noise levels, in addition to testing on multiple different muscles and users. The calibration software, discussed in the User Interface subsection, was also tested concurrently with the EMG software to ensure the signal detection was flexible enough to accommodate different users, muscles, and noise levels.

The testing demonstrated that peak detection was a reliable determination of whether the muscle is contracted. The difference between the maximum of contracted and non-contracted signals is consistent, and large enough that no false positives or false negatives were observed during software testing.

3.5 Detailed Operation of Motor Subsystem

Subsystem Summary:
The design was needed to carry out the physical action of opening and closing the hand appropriately using provided power based on the results of the EMG sensing. When activated by a command signal from the microcontroller, a motor shaft will turn and change the tension in strings that are attached to the finger joints. Originally, the design called for use of a servo motor due to their commonality and ease of use. However, due to the need for continuous rotation and and increased torque, a DC geared motor was chosen instead. Additionally, a motor driver compatible with our power system and with the ability to move the motor at varying speeds and in both the clockwise and counterclockwise direction.

Requirements:

- Hand must be designed to include joints which allow for easy opening and closing of all five fingers. Fingers must include hollow tunnels for stringing.
- Arm socket must include housing for the motor(s), batteries, microcontroller and any additional PCB dimensions, as well as any other additional hardware required.
- Battery location must be accessible by the user for changing or charging batteries.
- The closed-position grip of the hand must be able to hold objects without slippage.
- Any buttons, switches, or sensors for locking or calibration purposes must be in a position accessible to the user.
- Hand and socket must be reasonably lightweight for realistic, daily use.

Hardware:

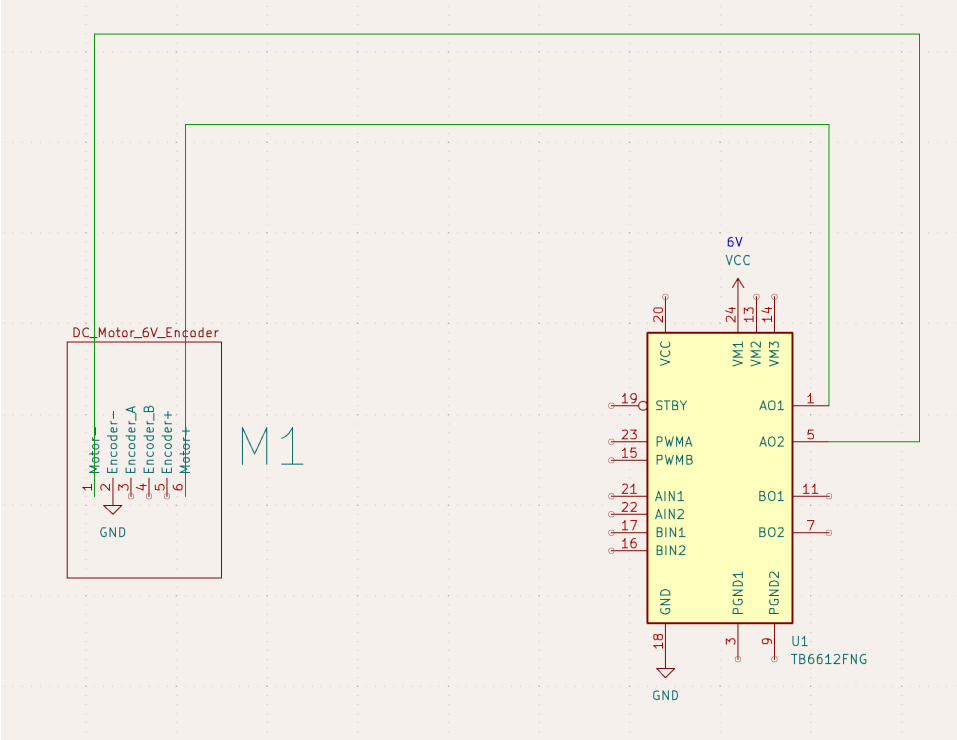

Figure X. Schematic showing connections between the ESP32 microcontroller, the motor driver, and the motor.

a) DC Geared Motor

Choosing a motor was a careful design consideration and a careful balancing act between size and strength/torque. While a stronger motor that could put out greater torque would have allowed for the best possible hand closure,

Will show schematic and describe limits of motor chosen/how implemented

3.6 Detailed Operation of Power Subsystem

Requirements

The requirements for the power subsystem are:

1. Provide necessary power to operate the motor
2. Provide necessary power to operate the signal processing hardware and microcontroller
3. Simultaneously fulfill the previous two requirements, which require different operating points
4. Offer a removable battery for minimal user downtime in the event of full battery discharge
5. Offer a rechargeable battery for user convenience
6. Communicate low battery level to the user
7. Offer protection from overcharging, over-discharging, and short circuits

Hardware

The power system uses two protected 18650 lithium ion batteries in series. The system uses Epoch 18650 2600mAh 8A batteries with built in protection. They are rechargeable via a standard 18650 lithium ion charger and removable. The batteries are 3.7 V, nominally, with a maximum charge voltage of 4.2 V and a minimum safe discharge voltage of 2.5 V. In series, the output is 7.4 V, with a maximum of 8.2 V and discharge cutoff of 5 V. To fulfill the requirement to power the motor, the batteries directly provide power to the motor driver. To power the signal processing hardware and the microcontroller, the voltage is regulated by the stocked AZ1117 voltage regulator. It has a maximum input voltage of 18 V and maximum current of 1 A. There is a large margin between the maximum safe input voltage and the output of the batteries. The maximum current draw that the ESP32-S3 will require is 355 mA while actively using the highest gain RF mode. We only utilize modes with lower power, so there is also a good margin between the maximum safe current of the regulator and the microcontroller. Finally, to monitor the battery charge level, we use a MCP602-E/P operational amplifier as a voltage follower. First, the battery voltage is scaled by a factor of 22 kΩ/69 kΩ via a simple voltage divider. Then, the op-amp follows the reduced voltage, protecting the input of the ESP32 from any spikes in current from the battery. Since the lithium batteries tend to retain their nominal voltage for most of their charge cycle, we illuminate a low power LED when the output voltage is less than or equal to 3.5V. A schematic of the system is below in figure XX.

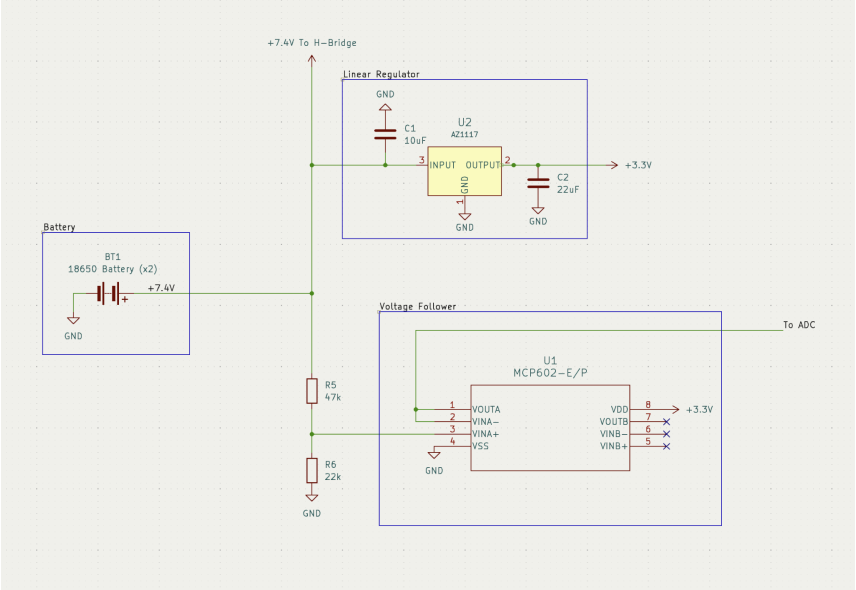

Figure XX. Power board schematic

An additional consideration for the power hardware is the mechanical battery holder. The user needs to be able to access the batteries and the ease of removing them should be reasonable to perform with one hand. Due to the built in protection circuit boards in each battery, they are slightly longer than standard 18650 batteries. To accommodate this, the power system uses a custom made battery holder board that better fits the dimensions of the batteries. The board also features a slide switch to connect and disconnect the batteries from the rest of the system. To connect the external power board to the main board, we use a JST connector. A CAD image of the power board is below in figure XX.

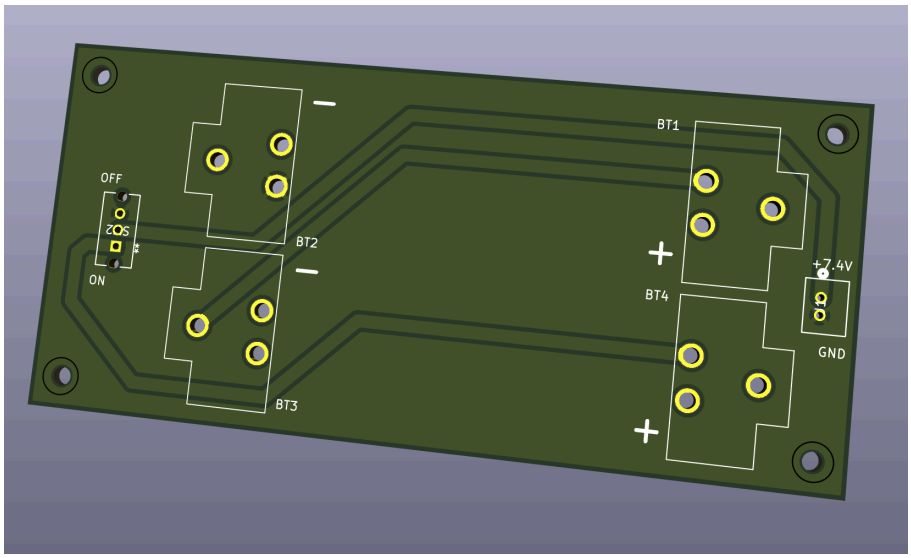

Figure XX. Power Board 3D render

Software

The software component of the power subsystem's main requirement is to monitor the battery voltage. The software needs to notify the user when the charge is low and then shut down the entire arm if the batteries reach their discharge cutoff voltage. To do this, we configure a hardware timer interrupt to set a flag every minute that will trigger a service routine. The service routine will be performed directly after gathering EMG data so that it does not interfere with the reading. During the routine, calling analogRead maps the output of the voltage follower (0-3.3 V) to an integer, x ∈ 0-4095. From this value, we can calculate the battery voltage as follows: $V_{battery} = \frac{22+47}{22} \times 3.3 \times \frac{x}{4095}$ If the battery voltage is below 7 V, a red LED turns on, indicating that charging is needed. Once the battery charge reaches 5 V, the built in protection circuitry in the batteries will disconnect them from the arm.

Subsystem Testing

There are three main tests that need to be performed to ensure that the power subsystem is working. First, we need to ensure that the linear regulator is outputting 3.3V. Second, we need to ensure that the voltage follower tracks its input voltage. Finally, we need to test that the hardware can correctly recognize a low battery voltage event and issue a response to the system.

Since the linear regulator only consists of one part, it was not tested independently and first tested during integration. Our first iteration of the board utilized the AP2112K stock regulator, which could not handle the full charge voltage of the batteries. So, for the first integration test, we relied only on the input from one battery instead of two in series to power the main board. To make up for the higher voltage required for the motor, we utilized an external power source. Moving to the final iteration of the board, we are using the AZ1117. The two regulators have a nearly identical footprint, so the changes made as a result of the integration test were minimal.

To test the voltage follower, we wired the battery to input across a potentiometer. By sweeping the potentiometer and measuring the input voltage to the voltage follower, we found the following output curve shown in figure XX.

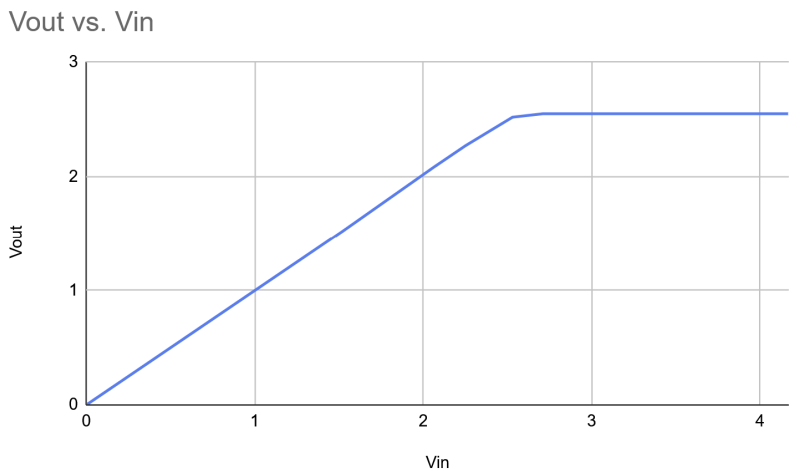

Figure XX. MCP602 op amp curve configured as a voltage follower

The output closely follows the input up to a saturation threshold at 2.55 volts. This test informed the selection of the resistors for the voltage divider. The divider needs to map 8.4 V (maximum battery voltage) to 2.55 V on the voltage follower input. Using two stock SMT resistors, the closest ratio achievable is 22 kΩ to 47 kΩ for a dividing factor of $\frac{22}{22+47} = .319$. This maps the maximum voltage to 2.68 V. This is slightly above the saturation threshold for the voltage follower, but this was an acceptable tradeoff for using only two SMT resistors. The nominal, low battery, and shutoff voltages are all within the linear region of the voltage follower.

The final test is the code. To test that the system can recognize important battery thresholds (7 V and 5 V), we connected the DC power supply across the battery holder terminals and varied the voltage. The system successfully recognized low voltages and turned on the LED.

A Note on Demo Day

Due to delayed shipping of the final board from OSH Park, the demo day product used an older variation of the board that featured the AP2112K linear regulator, only capable of handling a 6 V input. Because of this, we only used one battery. The board also used a green low battery LED as opposed to the red LED intended for the final design.

3.7 Detailed Operation of Hand & Socket Design

Subsystem Summary:

The physical design of the prosthesis should allow for an able-bodied person to demonstrate full functionality on EE Senior Design Demo Day. Additionally, the design should allow for easy adaptability for real-user applications in the future. The main goal of this subsystem is to build upon existing eNABLE designs to provide housing for all necessary electronic hardware and wiring, in addition to means for string-based actuation. User experience should be kept in mind throughout the design process.

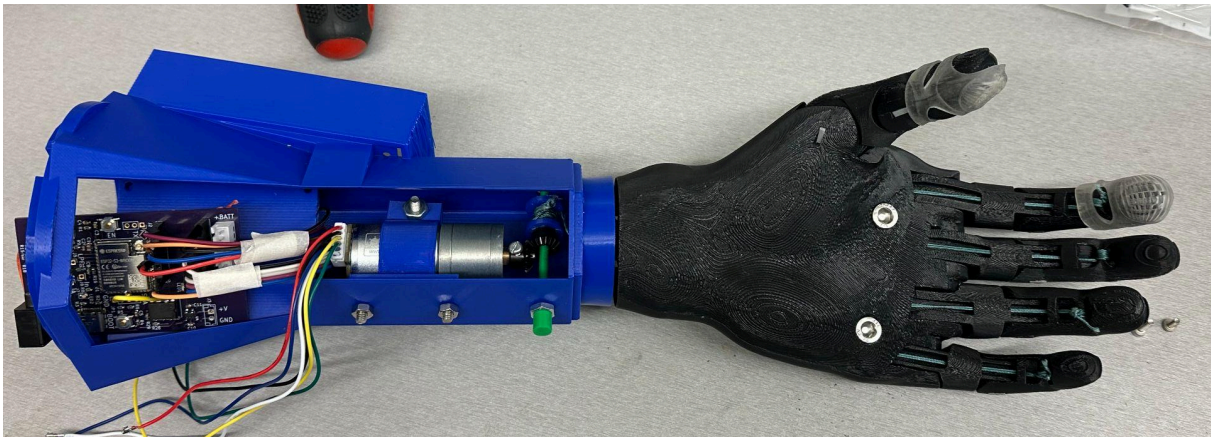

Figure XX. Final Physical Design

Requirements:

The requirements for the physical hand and socket design are:

1. Hand must be designed to include joints which allow for easy opening and closing of all five fingers. Fingers must include hollow tunnels for stringing.
2. Arm socket must include housing for motor(s), batteries, microcontroller and any additional PCB dimensions, as well as any other additional hardware required.
3. Battery location must be accessible by the user for changing or charging batteries.
4. The closed-position grip of the hand must be able to hold objects without slippage.
5. Any buttons, switches, or sensors for locking or calibration purposes must be in a position accessible to the user.
6. Hand and socket must be reasonably lightweight for realistic, daily use.

Theory of Design:

The basic theory of operation for most existing eNABLE hand designs utilizes a stringing mechanism within a 3D-printed hand to facilitate the closing of the hand. As shown in Figure XX below, string is threaded through hollow tracks in the printed fingers, the ends of which are then tied to a whippletree, so that actuation may be performed using a single string.

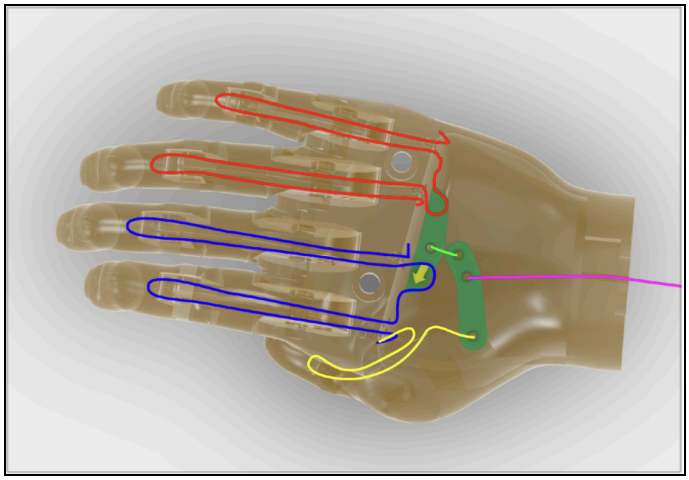

Figure XX. Hand/Stringing Diagram

Since the goal of the project is to leverage electrical engineering expertise to create a design which eNABLE ND can build upon further, we elected to build our hand and arm design off of an existing open-source eNABLE design: the Kwawu 3.0 Socket-Version prosthesis, shown in Figure XX below.

Figure XX. Image of Overall Hand/Socket Basis Design

This design was selected mainly due to the full-circumference socket, which allows optimal space to add means of housing the electronic parts. eNABLE-ND has used the Kwawu Arm design in recent years, with a team successfully printing and assembling the Kwawu 3.0 within the last year. Thus, due to the familiarity and space for housing/mounting which the design provides, the Kwawu 3.0 was selected as the basis for our physical design.

It should be noted that other basis designs from the eNABLE open-source catalog were considered. The assessment of these designs is summarized in the trade study chart in FIgure XX below. Ultimately, our decision relied primarily on documentation of a precedent of former

device success if used previously eNABLE ND, arm volume for potential electronics housing, and some aesthetic preference.

Table YY. Basis Design Trade Study

Existing Device:	Surface Area + Modification Potential	Precedent	Joint Flexibility	Aesthetics
(1) Unlimbited	~½ circumference socket still allows some room for additions but more limited than Kwawu	Used by eNABLE ND but not in recent 2-3 years	TBD	More robot-looking
(2) Kwawu	Full-circumference sockets allows max room for added housing / interface features	Used by eNABLE ND frequently and recently	Some issues with joint flexibility when used by enableND	Very human-hand looking
(3) Reborn	~½ circumference socket still allows some room for additions but more limited than Kwawu	No known precedent	TBD	More robot-looking
(4) Po	Available images indicate full-circumference socket?	No known precedent	TBD	More robot-looking
(5) Kwawu 3.0	Full-circumference sockets allows max room for added housing / interface features	Used by eNABLE ND recently with good results	Good joint flexibility/performance (report from enableND)	Very human-hand looking

The Kwawu 3.0 files are designed in the OpenSCAD software for ease of scaling to user fit. We were able to leverage this scaling feature to create a hand of a reasonable adult hand size, which still allows for proper housing of all parts. The OpenSCAD software files allows for input of features such as the width of the hand, desired length of the forearm, etc. The scaling parameters which were chosen are shown in Figure XX below.

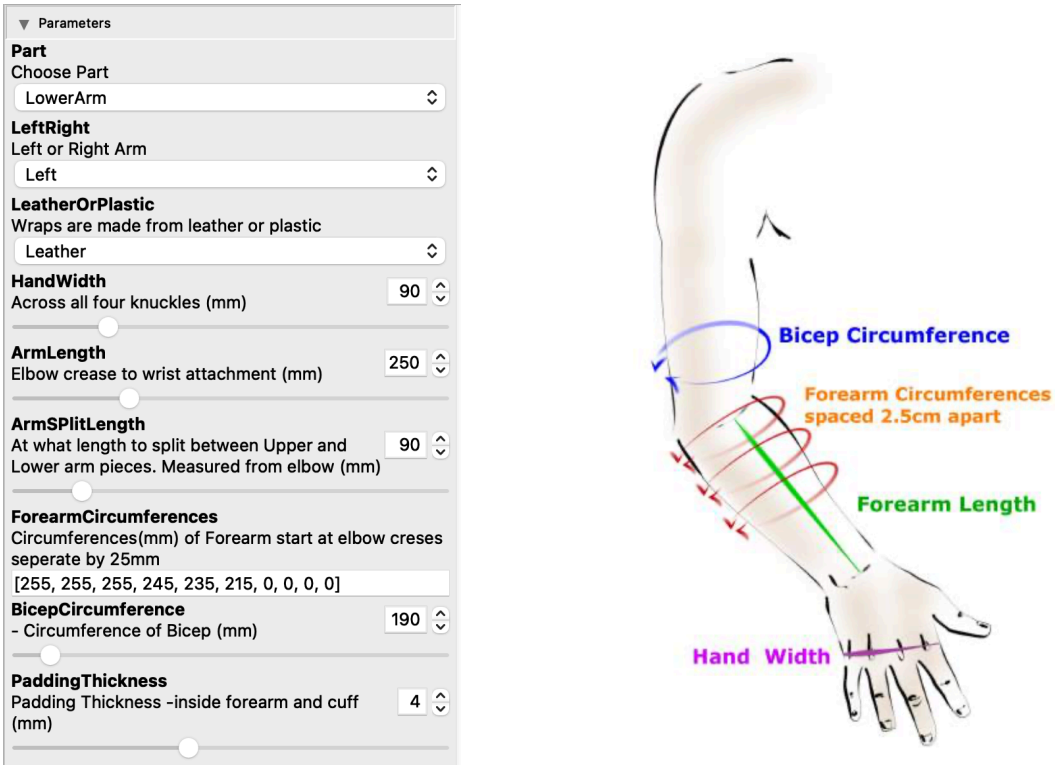

Figure XX. OpenSCAD Scaling Parameters

Making adjustments in the right-hand parameters column will scale all parts of the arm, but the files need to be individually rendered and then exported as .stl files. The LowerArm part is the part subject to further modifications in SOLIDWORKS, as this part is providing housing for the electronics. The remaining parts can be sent straight to print. The exception to this may be the Hinges file, which may need to be slightly scaled up to fit tightly in the finger joint holes of the hand. Scaling can either be done in SOLIDWORKS, or by increasing the HandWidth parameter and then exporting the Hinge file again.

CAD Design - Modifications:

Having scaled the ARM design to a reasonable size in OpenSCAD to accommodate housing and mounting mechanisms for the electronic components, the design of the lower arm needed to be modified in CAD to provide actual housing components. The initial upload of the LowerArm file in SOLIDWORKS, is shown in Figure XX.

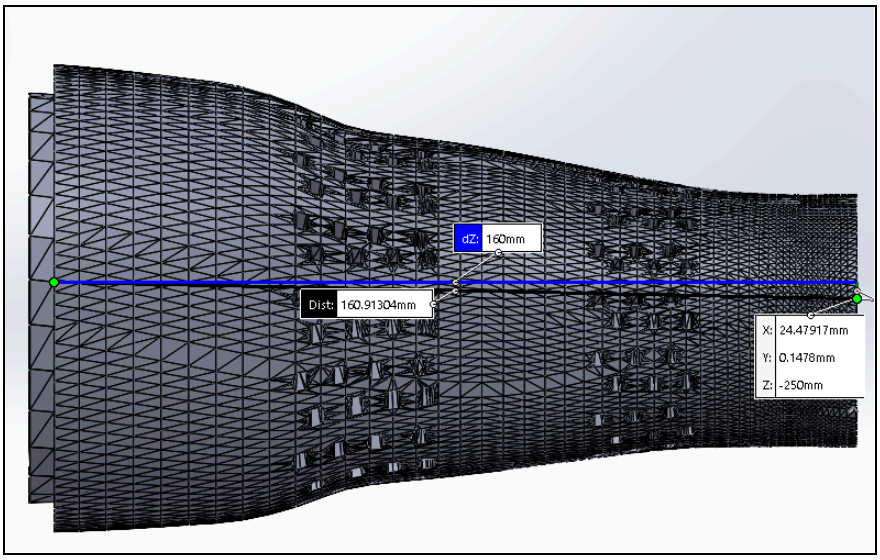

Figure XX. Lower Arm in SOLIDWORKS

While the original basis lower arm design is meant to mimic the shape of a human forearm, this portion of the arm was ultimately replaced with a hexagonal, geometric-style lower arm. This decision was made in order to improve ease of making modifications, mounting the boards and motor by minimizing curved surfaces within the arm. To do this, the section of the arm between the connection ends was removed. It is essential to preserve at least ~10mm on either end of the original solid body in order to preserve the means of connecting the LowerArm to the UpperArm and Palm. These ends can then be saved as separate solid bodies, and the distance between them can then be adjusted to increase the length available for parts housing. Ultimately, an additional 16mm was added between the ends to make room for motor and gear mechanism.

In order to create the replacement geometric forearm, we opted to implement a hexagonal shape to abstractly retain the overall round shape of the LowerArm, while also being able to work with substantially tall flat surfaces for mounting. To do this, a hexagon of the same approximate radius of the wrist connection was extruded from the wrist to the LowerArm center. Another extrude feature was then used to add a drafted hexagon from the end of the former section to the socket connection end. The length of the upper section must be long enough and have a diameter wide enough to accommodate the main PCB dimensions. The lower section (closer to the wrist) must be long enough to house the motor, in addition to any string-spooling mechanisms. The general hexagonal shape is shown in Figure XX.

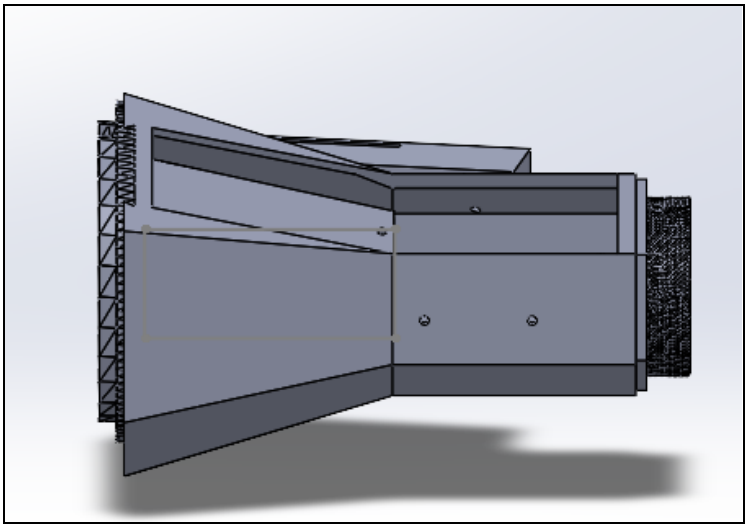

Figure XX. Hexagonal LowerArm

Part Integration:

Figure XX. below shows the overall locations of the electronics within the final LowerArm design. The subsequent sections discuss the design decisions relevant to the integration of each subsystem in the final ARM design.

Figure XX. Electronics Locations in Final Design

a. PCB Mounting

The PCB also needs to be mounted within in the arm, in a position where it is secure but in which the OLED on the board can be seen by the user and any relevant buttons included on the board for user interfacing can be pressed.

For purposes of demonstrating the board design on Demo Day, we opted to leave the lower arm with an open section, so that the board layout and motor location is able to be viewed during demonstrations. A diagonal panel was extruded through the upper section of the lower arm, with mounting holes for securing the PCB in place. In the future, a rotating panel may be implemented for closure and safeguarding the electronics from external elements. Figure XX. offers a clear visualization of this panel location.

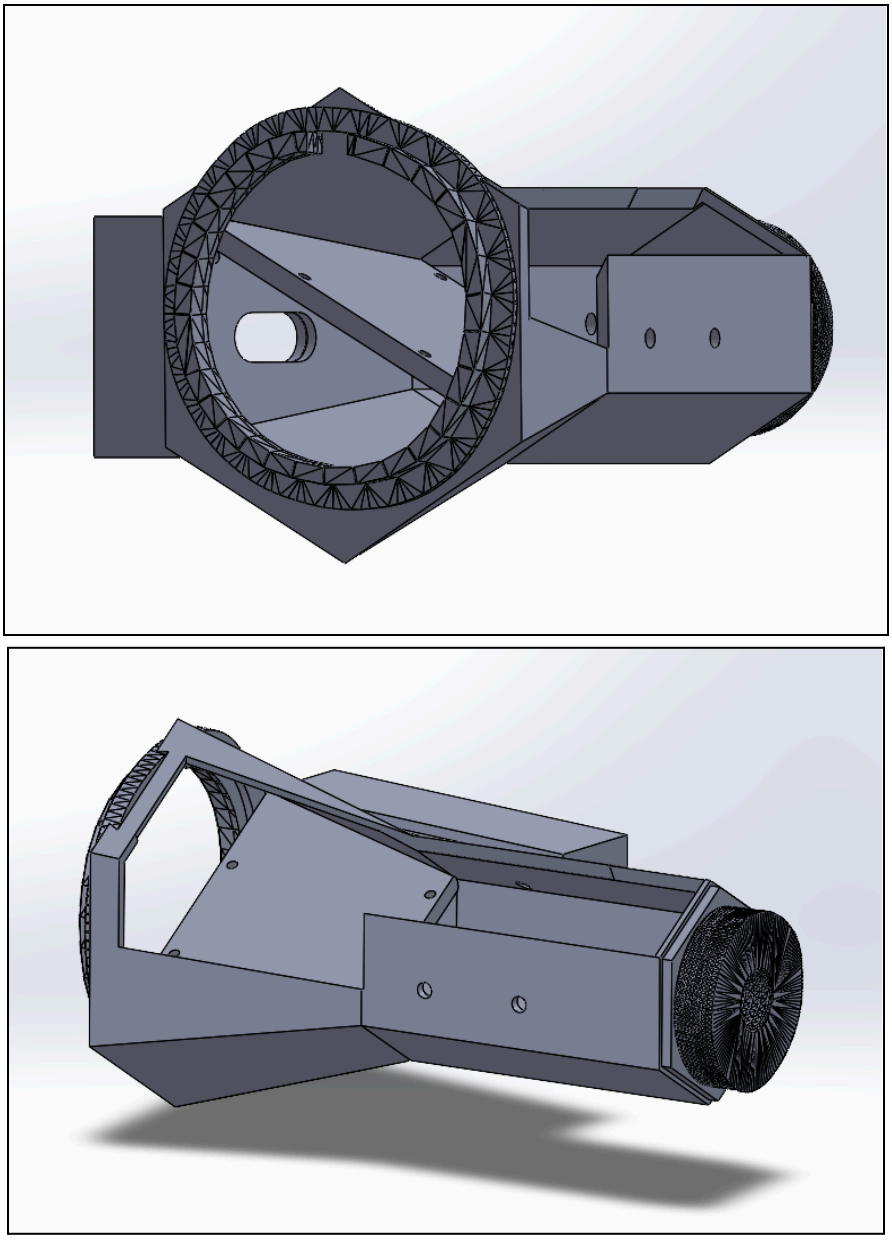

Figure XX. PCB Mounting Panel

b. Battery Housing

The power subsystem also requires means to be housed within the arm. For this purpose, an external box is included on the outside of the lower arm, inside which the power board can be mounted. In SOLIDWORKS, a hole was cut through the base of the power box to the main hollow section of the arm, so that the power cables could be threaded through said hole, and connected to the main PCB without any wiring externally observable. The external box is shown in the figure below.

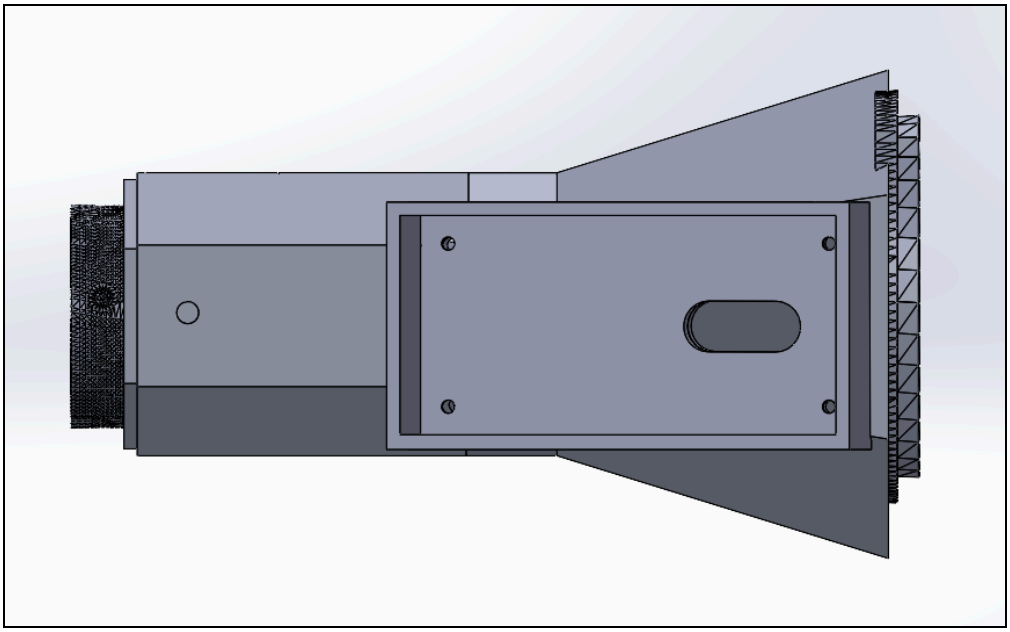

Figure XX. External Box for Power Board

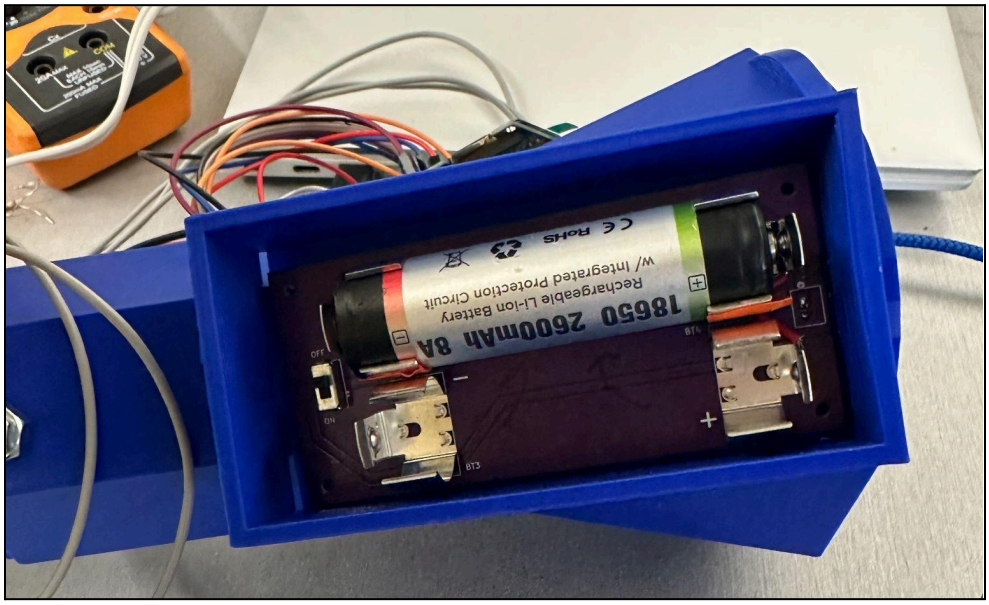

Figure XX. Power Board in External Box

c. Motor Mounting

One challenge of the design was finding a mechanism to mount the motor which would allow the motor to not only fit in the arm, but to also allow for correct tension to be applied to the string from the hand. Ultimately, we opted for a circular clamp which would hold the motor in place and could be mounted with screws on an inner-edge of the hexagonal lower arm piece. This clamp design is shown in Figures XX and YY below.

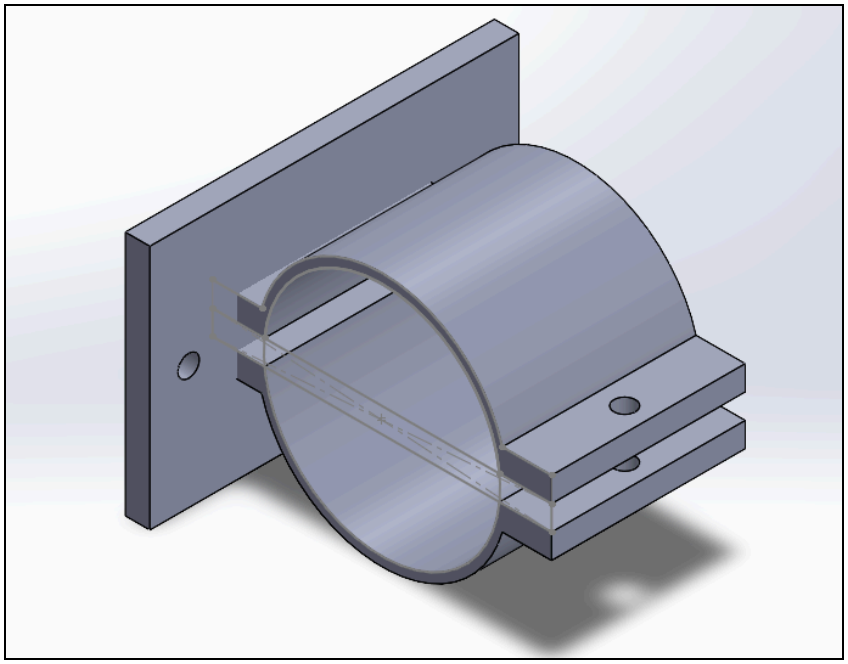

Figure XX. Motor Clamp

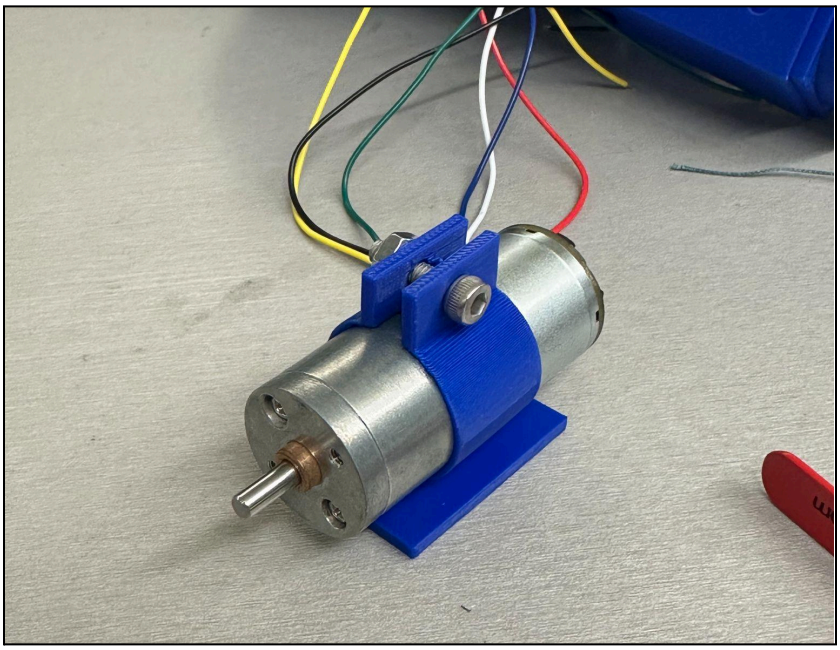

Figure YY. DC Motor in Motor Clamp

d. Stringing Mechanism Integration with Motor

The most challenging part of the physical design was integrating the stringing mechanism of the hand with the motor shaft. This process required iterative designs, as some failed or proved not to be as robust as originally anticipated. Several variables influence the success of the hand-closing mechanism, including the robustness of the knots in the hands, the torque provided by the motor, the flexibility of the material used

in the hinges of the fingers, and the technique of translating the rotational motion (roll to pitch).

Figure XX. Final Spooling Mechanism

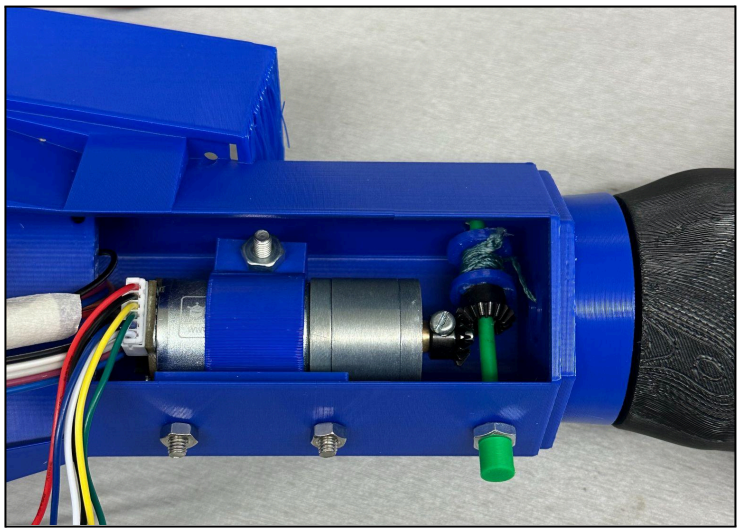

Fig X. Set-up of Failed Spooling Mechanism

3.8 Detailed operation of User Interface Subsystem

3.9 Interfaces

To the extent that the interfaces between subsystems need further explanation, do so. (Parts of this may be covered in the subsystem descriptions.)

Software Interface

The majority of code operations are mutually exclusive functions, which are time sensitive, run for a finite duration, and should not be interrupted. When a muscle flexion is sensed, EMG sensing is temporarily suspended to avoid multiple motor activations from the same muscle contraction. Furthermore, the prosthetic hand only has two discrete states: open or closed. The motor should not be interrupted while changing the hand between states because the software does not track partial states. Additionally, while the hand is calibrating, the normal operation (EMG sensing and motor activation) should be suspended because the software is in the process of updating the threshold.

Considering the desired operation of the prosthesis, the software is configured as a superloop with discrete states. The software states are:

- EMG Sensing
- Motor Activation
- Calibration
- Power Tracking

The software is primarily sampling the EMG signal until an event occurs. When an event occurs, the software temporarily pauses EMG processing to handle a different operation. The potential events and their corresponding operations are:

- Threshold detected → Activate motor
- User button pressed → Enter calibration
- Timer flag → Check battery power
    - Battery is low → Activate warning LED

Figure XX depicts the entire decision tree of the project software.

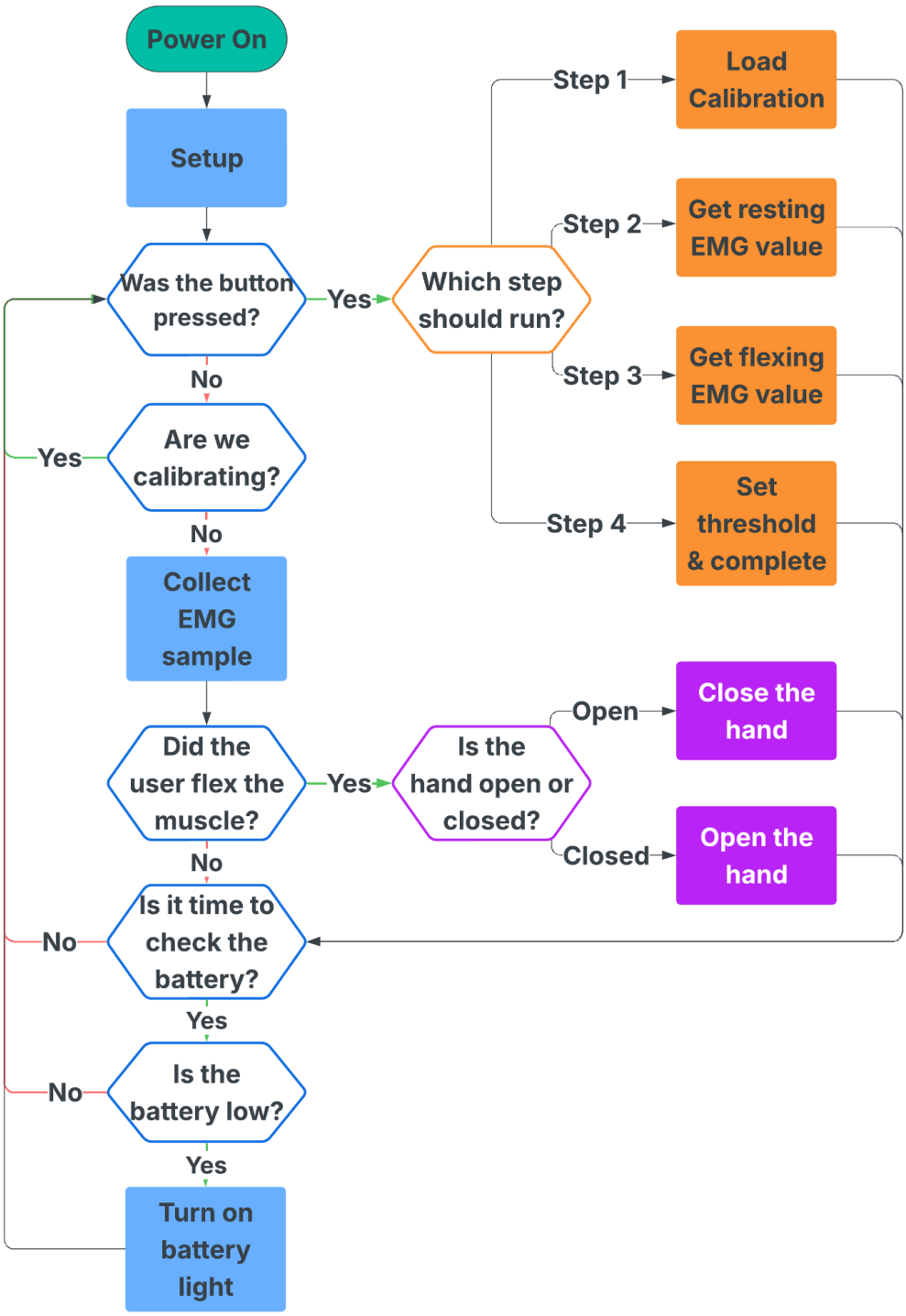

Figure XX. ARM ProsthEEsis Software Operation

PCB Layout

The full PCB layout is depicted in **Figure XX**. The primary design considerations for the PCB layout were:

1. **EMG and Motor Separation:** The motor components were placed on the opposite side of the board from the EMG sensing components to minimize noise in the EMG signal due to the motor sharing a ground with the sensing components.
2. **Motor and Battery Proximity:** Because the motor is the only component powered by the full 8V battery supply, the battery input was placed next to the motor components.
3. **Minimizing Size:** The PCB must fit within the 3D printed forearm, which informed the following decisions:
    a. **Board Dimensions:** The PCB is rectangular to ensure the width is narrow enough to fit within the forearm.
    b. **Audio Jack Orientation:** Rather than facing the outside of the board, the audio jack input is oriented towards the center of the board, allowing the EMG electrodes to be connected without the cable hanging over the side of the board.
    c. **OLED Orientation:** The OLED is oriented to be above the MCU on the board, ensuring it doesn't extrude beyond the board dimensions.

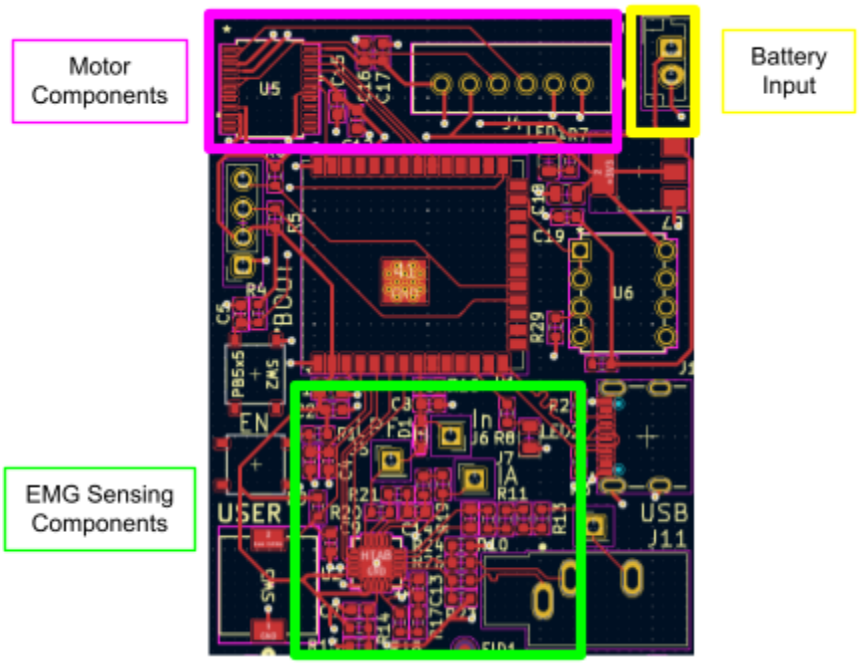

Figure XX. PCB Layout

MCU Selection

We have selected the ESP32-S3 for our project because it has Successive Approximation Register (SAR) ADC pins, which can operate more efficiently with lower power consumption than the standard ADC pins on the ESP32. In addition to having optimized ADC pins, the Espressif ESP-DSP library is also optimized for the S3. In earlier design revisions, we anticipated utilizing this library for software processing, motivating our MCU choice. The ESP32-S3 meets our other listed requirements, including minimum required input/output pins.

The ESP32-S3 variation selected was the WROOM-1U because the chip comes without the antenna module, saving space on the PCB. The antenna module is unnecessary for this project, as the software does not utilize Bluetooth or WiFi.

Any of the ESP32-S3-WROOM-1U memory variations are sufficient as the software only utilizes 1 MB of flash, 2 MB of RAM, and no PSRAM. Note that the Platformio file in the final software version is configured for the memory partition of the ESP32-S3-WROOM-1U-N16R8, which has 16 MB of flash, and 8 MB of RAM.

4 System Integration Testing

4.1 Describe how the integrated set of subsystems was tested.

4.2 Show how the testing demonstrates that the overall system meets the design requirements

5 Users Manual/Installation Manual

(The number of pages depends on how complicated it is to install and use. This section will be different for products that are more “consumer oriented versus business oriented.)

Note

- The board can’t be programmed unless the battery is plugged in
    - To safely connect electrodes from her, the user must be isolated/floating from any wall outlet/power supply
    - Wasn’t worth the space of the added components to allow USB power for programming —> end user won’t need to reprogram the board at all
- can also make estimated component pricing for enable
- also maybe make a code updating guide for enable?

5.1 User Manual

1.1 How to install your product

The first step for using the prosthesis is placing the electrodes. Two electrodes should be placed near the muscle belly, in line with the muscle fibers. The third electrode acts as a ground, and it should be placed near a bone or electrically unrelated tissue. An example of electrode placement is shown in **Figure XX**.

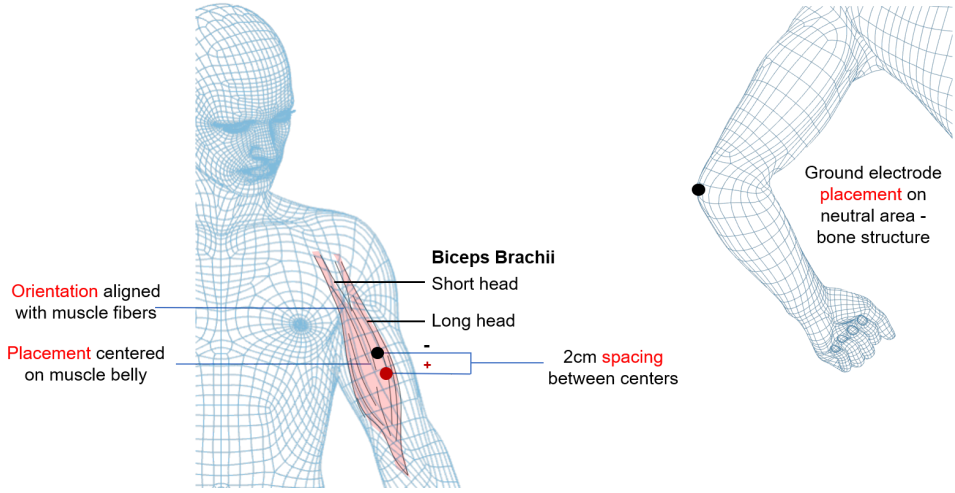

Figure XX. Electrode Placement Guide (source)

Next, the cables need to be snapped onto the electrodes. The green connector should snap onto the ground electrode. The yellow and red connectors should snap onto the muscle belly electrodes.

Then, the arm & socket should be fitted onto the residual limb. Now, the device is ready for setup and subsequent use.

1.2 How to setup your product

Before setting up the device, two batteries should be placed between the snaps in the battery box. To turn on the device, ensure the switch on the battery board is in the “ON” position.

Once the power is on, a blue LED should light up on the main board within the arm. If you see this light, you can proceed to calibration.

6 To-Market Design Changes

Each of the subsystems in the project can be improved in a few ways if the arm was to be taken to market. The first change made to the EMG system would be using a more effective Faraday cage to eliminate electromagnetic interference. The arm currently uses aluminum foil, which reduces some of the noise, but is prone to mechanical gaps which limit its effectiveness. In practice, the user will need to access the inside of the hand to calibrate and change the batteries. This necessitates some type of opening to free space which EM waves can propagate through. Optimally, the Faraday cage would have no gaps in it during normal operation. With just aluminum foil, this is difficult to do. Although other metals, like copper are more conductive than aluminum, aluminum is very light, so we'd likely stick to aluminum as the material going to market. A better option to ensure continuity of the Faraday cage during normal operation would be using some form of sheet metal. Molding a layer of aluminum to the shape of the arm would be impractical, especially if we are trying to keep costs low. A happy medium could look like a solid aluminum lining around the opening in the arm that mates to another aluminum lining on the hatch. This solid aluminum would be fixed to a foil that lines the inside of the arm. To further improve the noise isolation, if needed, we could use a thicker foil or add more layers of foil. This would mainly benefit the operation in very noisy electromagnetic environments (near a generator etc.).

Another additional feature that could be added to the EMG sensing system is an error detection code that recognizes if an electrode falls off of the user. All of the hardware for detecting an electrode disconnection event is already set up to support this system, so the only necessary modifications would be adding a new conditional to the EMG sensing loop. When the electrodes are attached to a person, the EMG hardware gives predictable readings, averaged over a sampling cycle. Adding a conditional to detect significant deviation from these readings would add the error detection functionality.

For the power system design, a very convenient feature to offer to the user is more resolved battery charge tracking. The current system only can alert the user if the battery is “low.” This is because the system only monitors the battery’s voltage, not the amount of power it has used. There are battery monitoring ICs that can be used for this, such as TI’s Impedance Track technology or Coulomb counter ICs. The best way to communicate the charge to the user would be a series of LEDs, a multicolor LED, or a small screen visible from the outside of the arm. The main constraint of adding in more hardware quickly becomes space. During our design process, one of the main changes that we made from our proto-board to our final board was adjusting the layout to better accommodate the limited space available inside the arm. Especially if the end user of the arm is a small child, to have a realistically sized arm, space is extremely limited. Given the current design, the best solution to create more room for added peripherals would be creating a two-sided PCB and utilizing the area around the mounting screws. Due to the time constraint, drastically changing from the protoboard design to a new PCB with more layers would not have been possible. In addition, boards with more layers would drive up costs. Another place for the power system to improve is the electrical connections from the battery board to the main board. Our final design uses JST connectors. The connectors work fine, but aren’t very durable and wouldn’t be reliable. The user is inserting and removing batteries directly next to the JST connector and in general the arm will be subject to daily wear and tear, which could be significant, especially for a child. Opting for a more permanent solution that the user couldn’t accidentally remove while servicing the batteries or during daily life would be best. An easy to market change here would be using Molex connectors instead.

For the hand and socket design, the two main limitations were 3D print quality and joint hinge and threading design. First, the goal of improving print quality would be to make the arm look more like a human arm. The current design has a rough, stepped texture, which approximates the shape of a hand, but doesn’t look or feel much like a real biological hand. Using a 3D printer with a higher resolution between layers of material would reduce the stepped texture and casting the hand with a mold would completely eliminate the issue. Also, using a plastic that more closely matches human skin tone would make the hand and socket look more like a human hand. Due to the constraints of the EIH and our budget, these options were not possible. Another limitation of our 3D print was the texture of the inside of the hand. This texture is very important for how the hand grips objects. The smooth texture of the 3D print is not very sticky, and therefore not very good at picking things up. Using some sort of molded rubber or spray on rubber would improve this. Increased friction between the hand and objects also reduces the force output front he hand needed to successfully hold objects, which makes the system more energy efficient. Finally, having access to higher quality prints, in combination with better mechanical designs, would improve the joint resiliency. This would result in more fluid grasping and would require less torque from the motor to grasp and release the hand.

Second, we experienced major difficulties with the hinges and the threading. For demonstration, the prototype that we present does not spring into an open position when it is not grasped. In other words, the original goal was for the hand to have two

states: firmly closed and firmly open. However the demo day design's states were firmly closed and slack. In the slack state the fingers can freely open and close. This shortcoming is due to a few things. First due to the delayed shipping for the final board, we could only use half of the voltage that we intended to provide the battery. This significantly reduced the maximum torque that the motor was capable of outputting. Second, even with full voltage, the motor likely wouldn't be strong enough to grasp the fingers. This leads to the final cause: too much force required to close the hand. The cause of this is that the joints were too firm. The joint flexion requires deforming soft plastic pieces within each finger. The plastic pieces used in the final design were soft, but not soft enough that the motor could pull against them. To demonstrate the hand, we opted for very soft improvised joints constructed out of wires and duct tape. They allow flexion, but are not elastic enough to return the fingers to a firmly open state. The ideal joints would be somewhere between the original elasticity and the improvised joint elasticity.

7 Conclusions

8 Appendices

Complete hardware schematics
Complete Software listings
Relevant parts or component data sheets (do NOT include the data sheets for the microcontroller or other huge files but give good links to where they may be found.)

Hardware Schematics

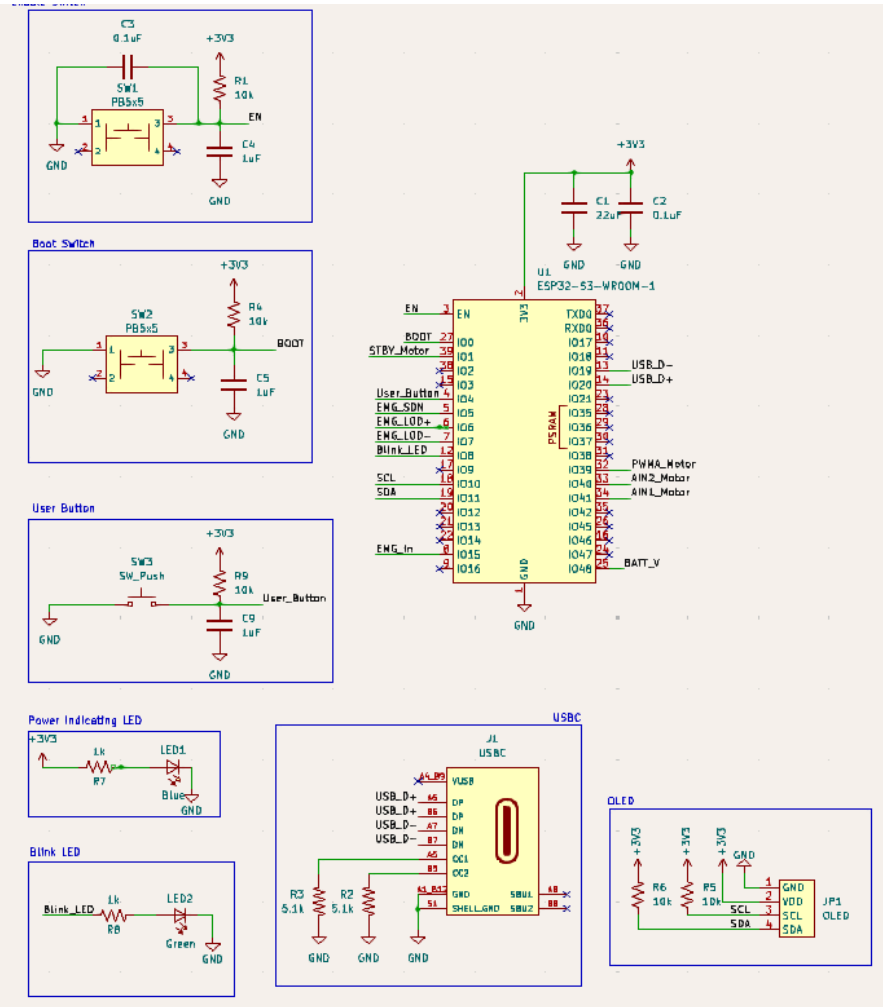

Figure A1. Schematic of MCU and Peripherals

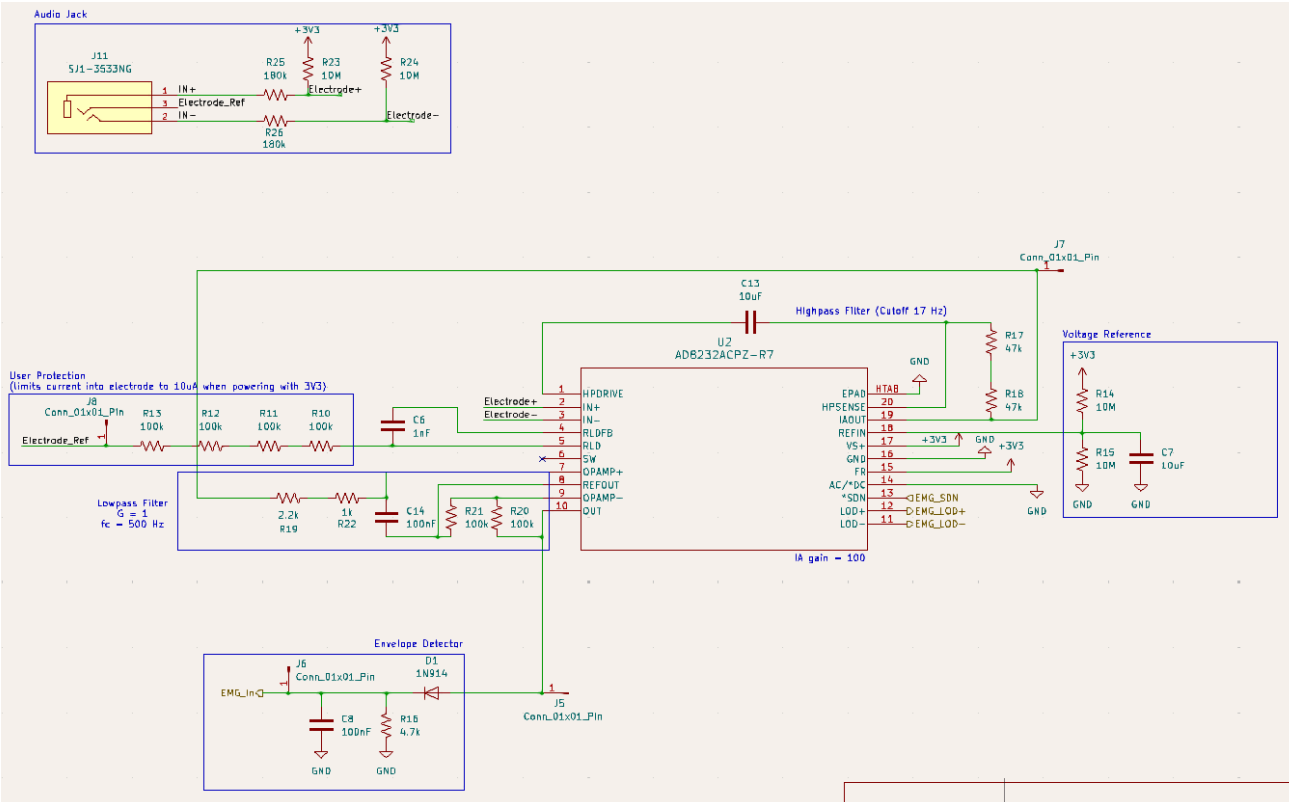

Figure A2. Schematic of EMG Sensing Subsystem

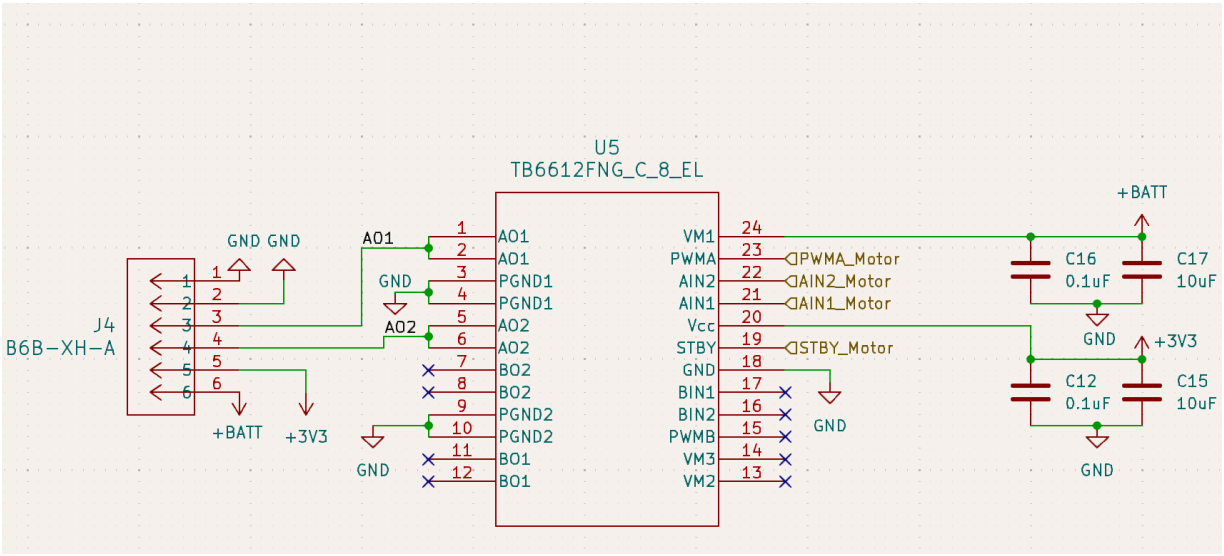

Figure A3. Schematic of Motor Subsystem

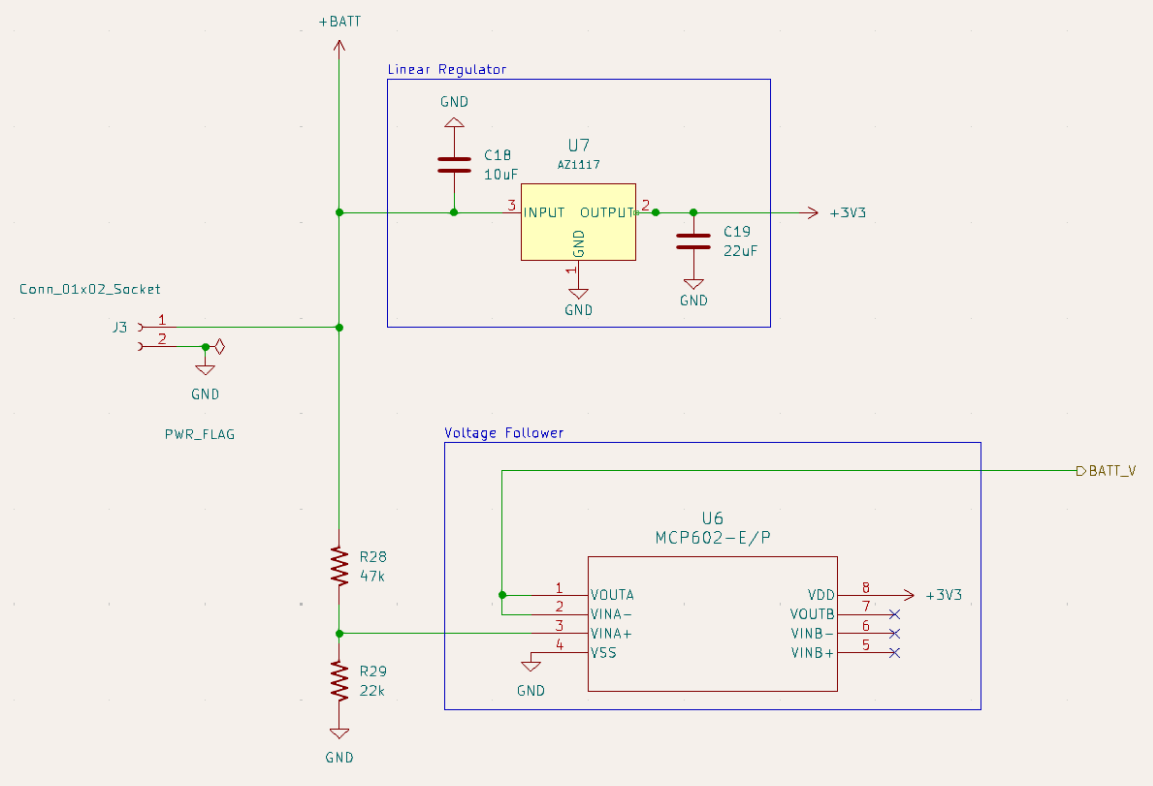

Figure A4. Schematic of Power Subsystem

Software Listing

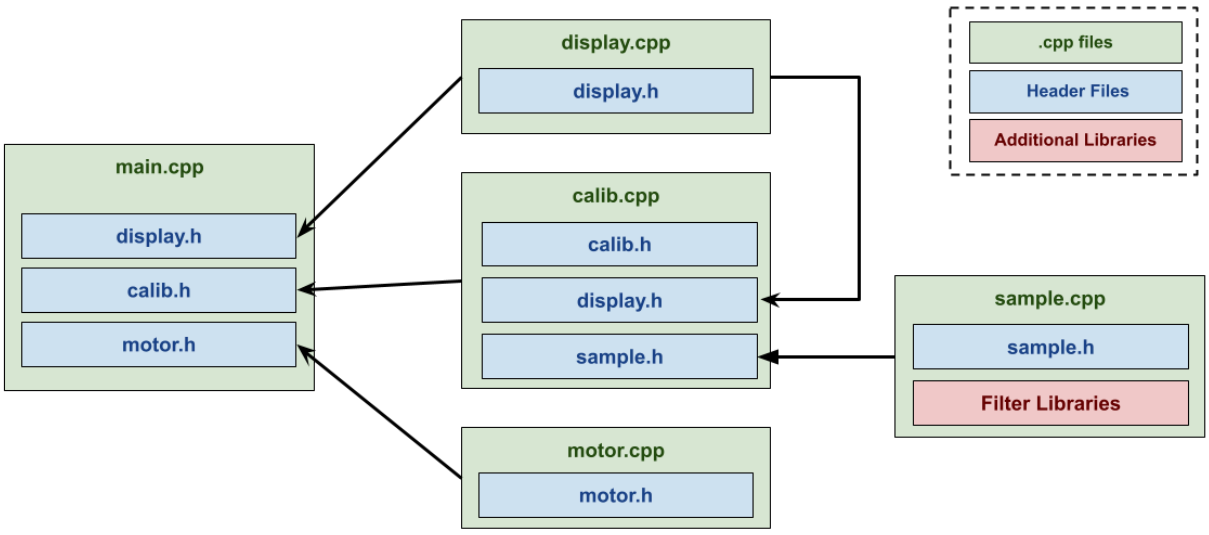

Figure B1. Organization of Software Files

Additional Documentation

- Arduino Filter Library Documentation

Component Data Sheets

- AD8232 (EMG IC)
- ESP32-S3-WROOM-1U (MCU)
- AP2112 (Linear Regulator (Demo Day Board))
- AZ1117I (Linear Regulator (Final Board))
- MCP602-E/P (Op-Amp)