Glove for Real-time Imitation and Performance

Group 21:
Rian Lowery
Christopher Marcoccia
Kane McAgy
Griseld Xhengo

Team Members

Rian Lowery

Computer Engineering

Christopher Marcoccia

Electrical Engineering

Kane McAgy

Electrical Engineering

Griseld Xhengo

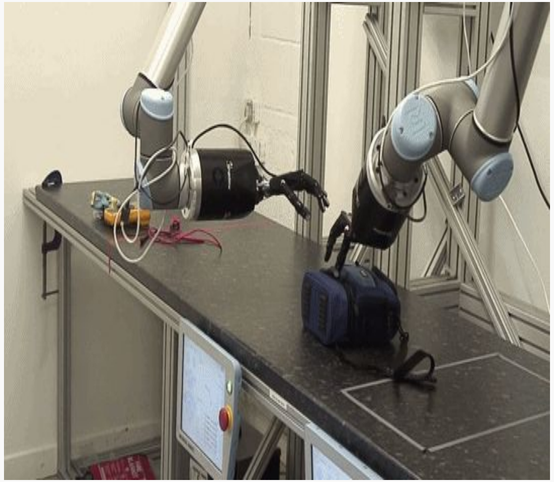
Electrical Engineering

Team roles and work distribution

	Hand	Glove	Software	PCB Design	CAD/Printing	RF Comms
Rian		✔	✔			✔
Christopher	✔		✔		✔	✔
Kane		✔			✔	
Griseld	✔	✔		✔		

Project Problem, Motivation, and Applications

- Some environments and materials are too hazardous for direct human contact
- Current remote interaction methods are limiting
- GRIP can maintain natural interactions without risk to personnel



Experimentation

Technological interaction with exterior environments and harmful substances can mitigate necessity for human contact participation in experimentation.

Exploration

-Attempts to give remote operators more direct interactions with the environment they are studying.
-Remotely piloted submersibles with controllable appendages.

Project Goals and Objectives

Goal type	Description
Basic	• Create a device with a series of sensors that will track the user's hand movements and translate them to GRIP, allowing it to mimic the user with high accuracy. • The hand will be able to have an open or closed grip as well as other standard positions
Advanced	• The system will be able to sustain a low-latency connection with moderate traffic between the user's hand and GRIP, allowing the hand to move with the user with negligible delay. • The hand portion of GRIP will have articulated fingers that can move independently of each other, allowing for more intricate interactions with the environment. Basic movements such as pointing and pinching will be doable.
Stretch	• Incorporate an arm attached to the hand that will be able to mimic the user's arm movement. • Single axis wrist movement • A homing algorithm will be made to calibrate soft and hard stops for the arm, allowing use from different-sized users after the calibration sequence.

Objectives	Description
Basic	• Resolution for flex sensor data needs to be at least 10-bit • Sampling rate at least 50-100Hz per finger • A normalization algorithm is needed to translate the flex sensor range to the servo movement range • Each finger should be able to be fully straight, go halfway, and be fully curled. Each position should be easily achieved and sustained
Advanced	• nRF24101 will be used to achieve wireless communication between the glove and the hand • Condensed packets with only essential information to reduce payloads, allowing for faster communications • Packet loss tolerance needed to ensure functionality, so a single missing packet doesn't halt functionality • Cross-talk prevention to ensure servos do not talk to each other • Soft stops working in tandem with hard stops to ensure fingers cannot exceed the desired operational range
Stretch	• Several gyroscopes will be utilized to measure hand movement in relation to the shoulder. This distance measurement will automatically give the desired elbow angle • The elbow will have a gyroscope solely for side to side postion. Bending that affects the distance of the hand from the shoulder is utilized by the objective listed above • Limit switches to be added to the base of the fingers for calibration, ensuring the motors know where they are on startup, or should any kind of loss of position happen

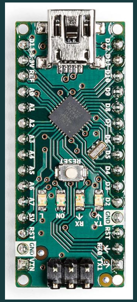
Engineering Specifications

Category	Parameter	Values
Performance	**Runtime**	**≥ 1.5 hours**
	Latency	**≤ 250 ms**
	Operational Voltage (Hand Servos)	**6V ±0.5**
Communication	**Link Reliability**	**Packet loss ≤ 0.5% at 5 m LOS**
	Wireless Transmitter Range	**≥ 10 m**
Actuation / Interface	**Gestures**	**Pointing, open grip, closed grip**

- Crucial specifications that are necessary to ensure a stable system
- Average human reaction time is ~250ms, so latency needs to be on par. Otherwise, visible lag will be seen and felt
- Due to nature of the project, it must be able to function long distances

MCU Selection

	Arduino Nano	**Teensy 4.1**	**ESP32**
Programing Method	Arduino IDE	C/C++ in Arduino IDE	C/C++ in Arduino IDE
Operating Voltage	5V	3.3V	3.3V
Size	18 x 45 mm	17.78 x 61.66 mm	28 x 55.30 mm
Cost	$25.70	$31.50	$11



- Target features for MCU include a small footprint, compatibility with our wireless transceivers, and accurate servo control
- While the ESP32 and Teensy have more features, most are considered overkill for the system
- Debugging and modification is also more user friendly for the Nano over the other two options

Voltage Regulator

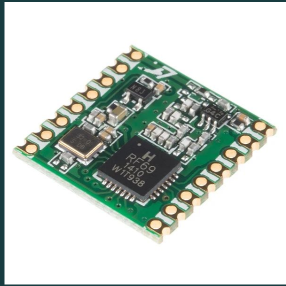
	LM3670	**TLV62569**
Max Current Out	350mA	2A
Output Voltage Range	0.7V - 3.3V	0.6V - 5.5V
Input Voltage Range	2.5V - 5.5V	2.5V - 5.5V
Operating Temperature (°C)	-40 - 85	-40 - 125

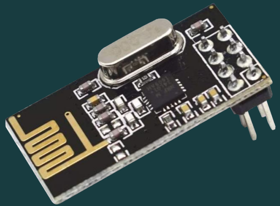
- Regulator is used solely for providing 3.3V and proper current to nRF24
- nRF max current is ~14mA, which will not maximize the use of the TLV
- Chosen regulator ranges do not exceed target values too much

Wireless Communication

	RFM69	nRF24l01
Frequency	315-915 MHz	2.4 GHz
Range	~300m	~50-100m
Penetration	High	Low
Bit Rate	300kbps	2Mbps
Cost	~$9.00	~$2.00

- While more susceptible to noise than the RFM, the nRF has significantly faster data transmission speeds
- Should it be necessary, range and penetration can be improved upon easier than packet transmission rate





Motor Selection

	Servo Motor (MG996R)	Stepper Motor
Control	Closed loop	Open loop
Position Accuracy	High	Can lose steps under load
Return to Position	Automatic	Requires a command
Ease of Control	Simple signal	Step and direction
Movement Torque	High	Low

- The motors will need to have high torque to move the fingers quickly. Steppers lose torque as their speed increases, making the servo motor the optimal choice
- Servo motor encoders will allow to constant positional monitoring and provide an simpler way to adjust soft stops
- MG996r chosen due to its low noise and higher end torque compared to other MG models

Battery

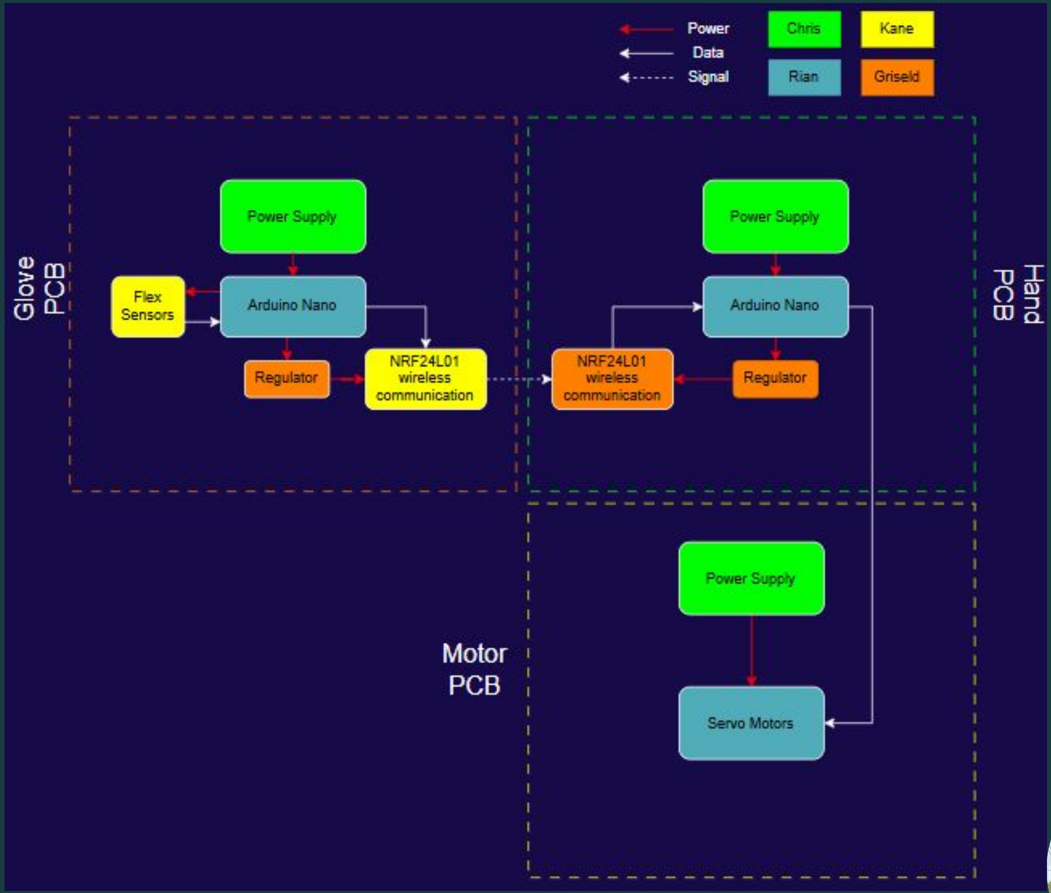
	NiMH 5-cell	**2S LiPo**
Voltage	~6V	~7.4V
Peak Current	20-25A	30-60A
Weight	~351g	~88g per battery
Noise production	Low	Medium

- While chosen battery is heavier, it provides better fit for project
- Low noise production so the RF signals can be as clean as possible
- LiPo battery current production is deemed overkill as max current needed for servos is 15A

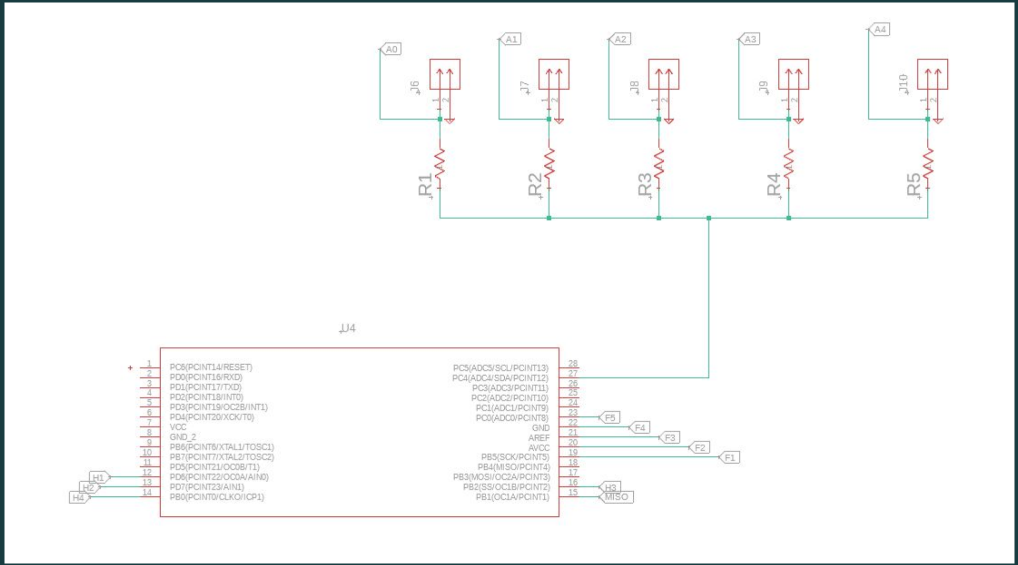
DYNB2442 Dynamite Speedpack2

Hardware Block Diagram

- The PCBs for the glove and the hand are almost identical with some small differences
- The power supply for the nano on the glove and hand will be a 5V 2.4A mobile power bank
- Power supply for the servo motors will be an NiMH



Flex Sensors - Voltage Divider Circuit



Flex sensor prongs and their connection to the Arduino Nano

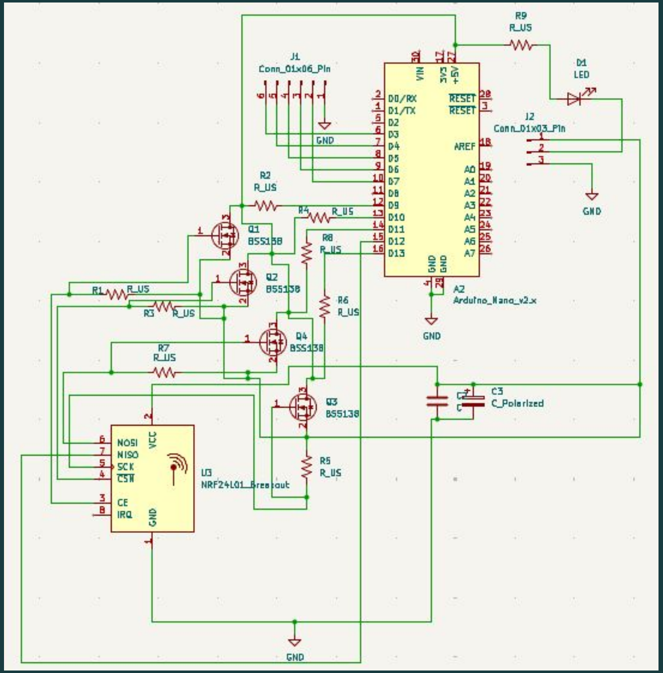
Voltage Divider Circuit to Arduino connections

- Flex sensor signals enter the Arduino Nano's ADC ports
- Flex sensor output ~0–1023
- Algorithms in the software will normalize bending into more linear progression, allowing for smoother servo control
- Smoothing via moving average / low-pass filtering
- Convert mapped value to joint-angle range (0–90°)

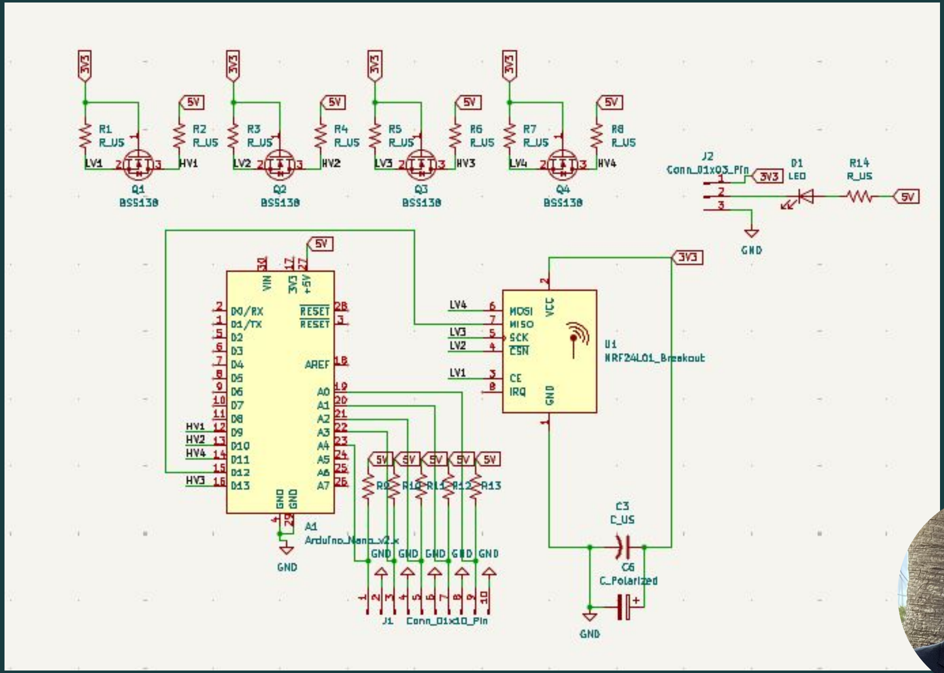
Schematic for the Hand and the Glove

- Hand and Glove PCBs nearly identical in terms of parts utilized. Key differences are in the software and what inputs and outputs the MCU has access to

- Both boards utilize a series of BSS138 MOSFETS to secure communication between the MCU's 5V logic and the nRF24's 3.3V logic. The glove PCB reads a series of analog inputs and transmits them to the receiver, which sends those values to the corresponding digital line that the servos connect to



Hand Schematic

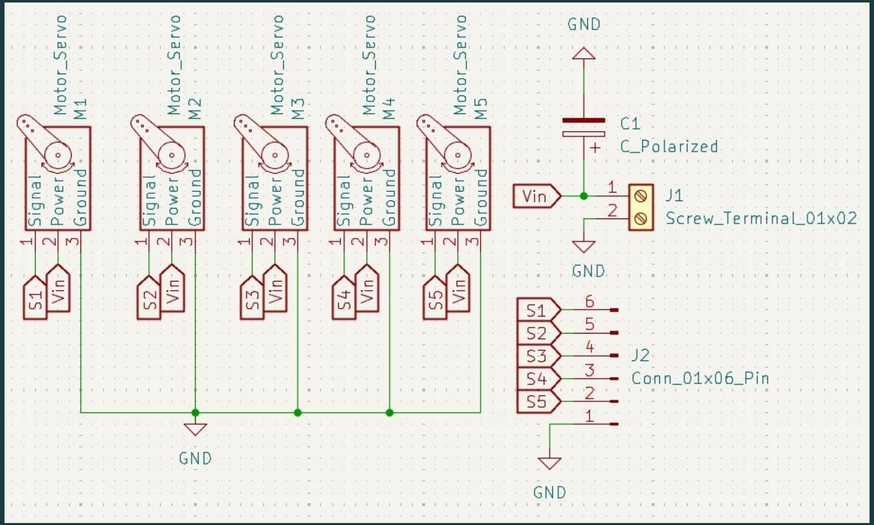


Glove Schematic

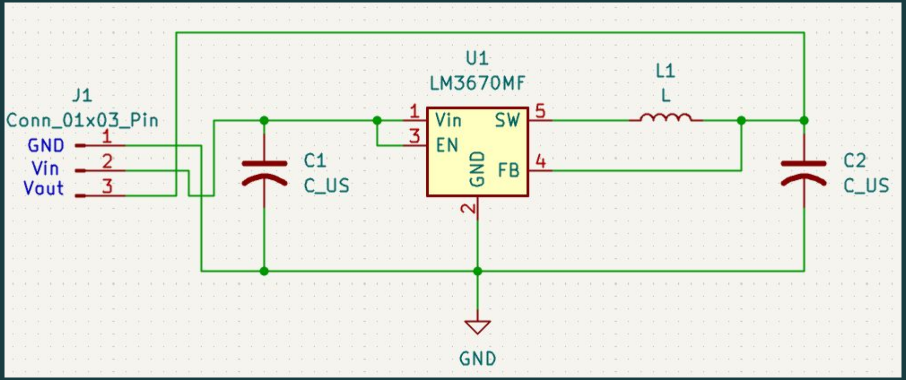
Schematics for the Motors and the Regulator

- Servos all powered by one battery which plugs into the screw terminal. Signals of servos connect to pins which connect to the hand pcb's MCU

- Regulator PCB is not built into the other PCBs. Pins will plug into main boards



Servo motor schematic

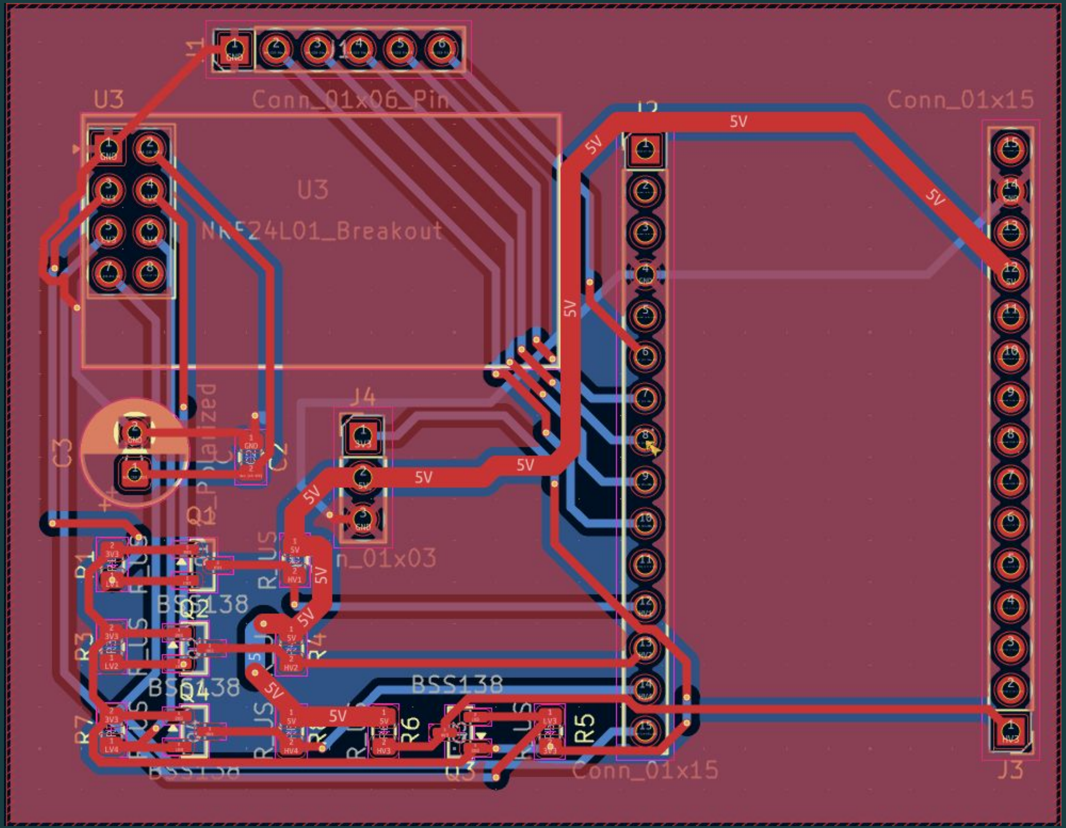


Regulator schematic

Hand PCB Design

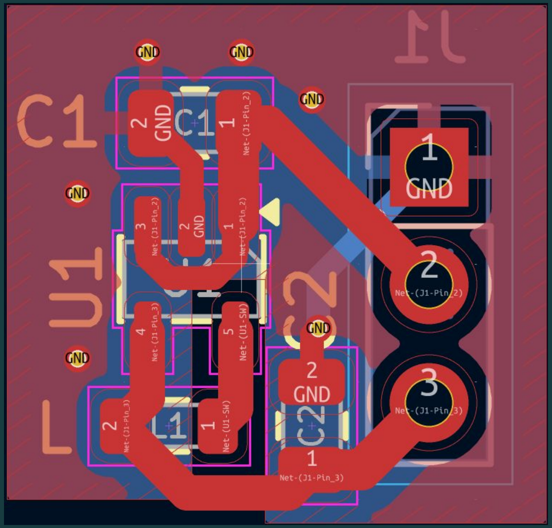
- The hand PCB is strictly designed to receive values and push them to the servos

- Design needs to be as compact as possible while still giving enough room to minimize noise between connections

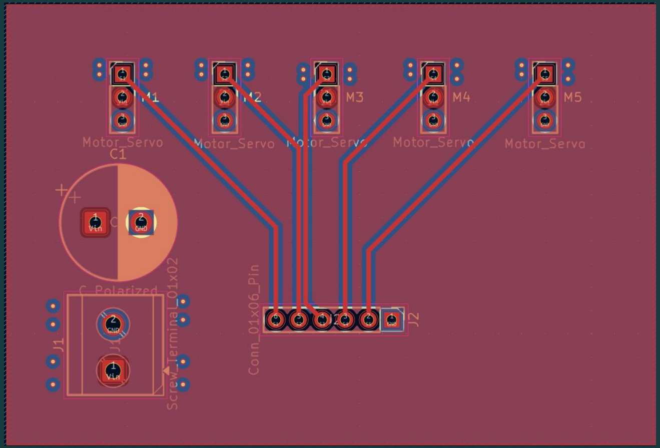


PCB Design for the Regulator and the Motors

- The regulator steps the voltage down to 3.3V and produces 0.3A, which will be more than enough for the nRF24 module on both the hand and the glove PCBs.

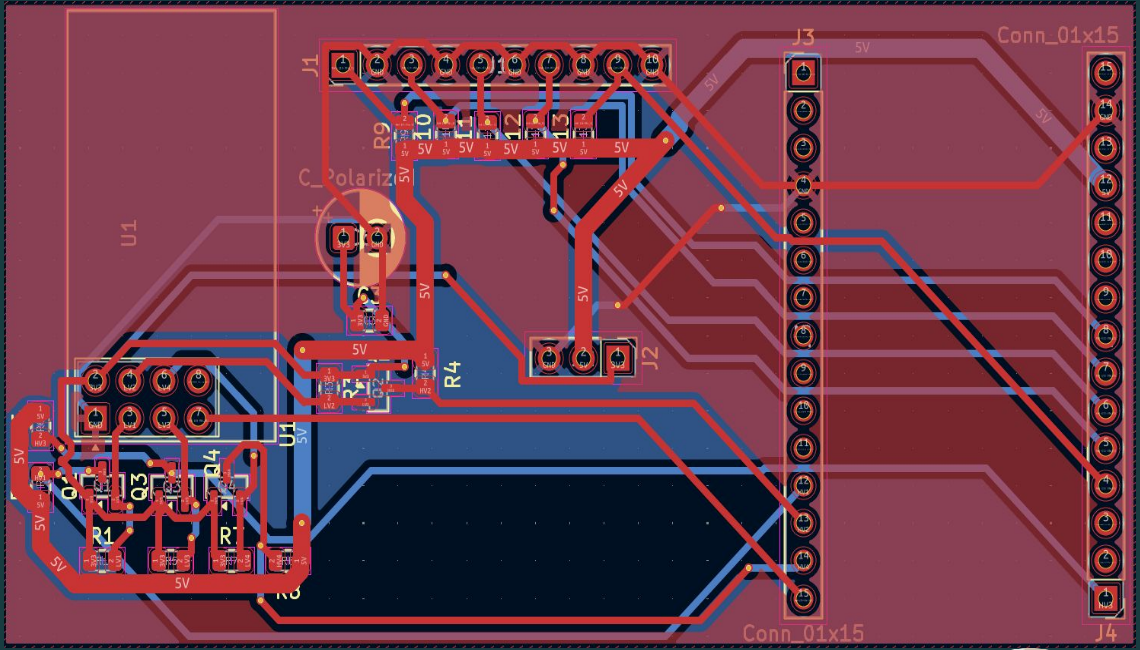


- The PCB for the servos is separate because of the high voltage and current that they will require. It will use a separate power source just for the motors and connect to the hand PCB via connectors and wires.



Glove PCB Design

- Five flex sensors that will take the users movements and convert them into values that will reflect the hand movements.
- Sensors are fed 5V, with a branching trace between the resistor and the prong connecting to the MCU to measure voltage changes



Software Architecture Overview

Glove

Sensor acquisition, calibration, filtering, and angle mapping

- Analog flex sensor acquisition
- Signal conditioning (normalization + curve shaping)
- Joint angle generation

01

Wireless

Low-latency packet transmission via nRF24L01

- Structured packet formation
- Hardware CRC validation
- 50–200 Hz update rate

02

Receiver

Packet decoding and safety validation

- Packet parsing
- Angle extraction
- Timeout detection (failsafe)

03

Servos

Real-time actuation of robotic joints

- Servo angle mapping
- Constrained motion limits
- Smooth positional updates

04

Design decisions supporting real-time performance

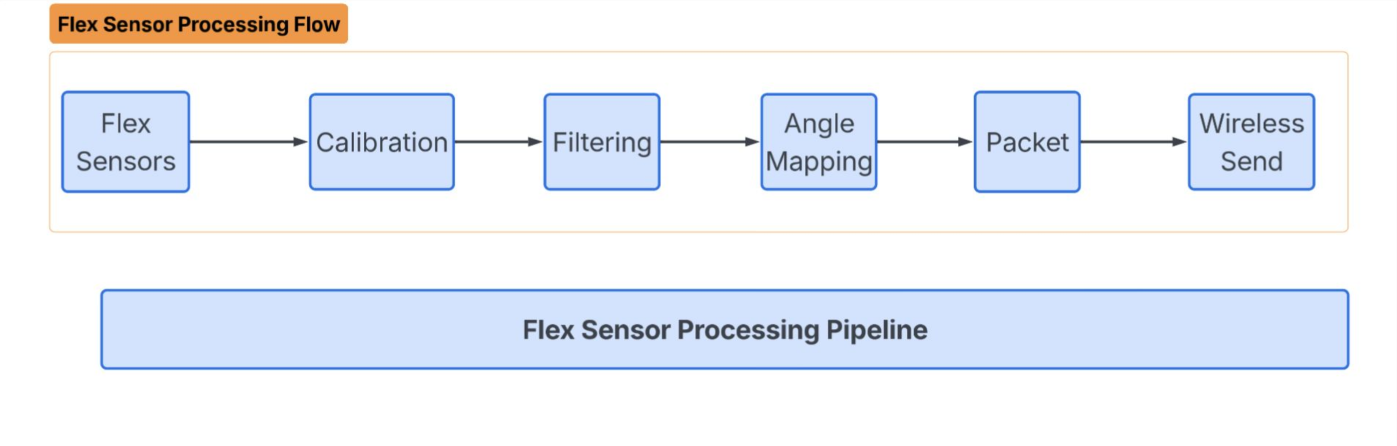
Software Technology Selection & Justification

Component	Options Considered	Selected	Justification
Programming Language	Python, C++	C++	Deterministic timing, low-level hardware access
Wireless Protocol	Bluetooth, WiFi, nRF24	nRF24	Low latency, hardware CRC, simple packet control
Control Architecture	Interrupt-based, Polling Loop	Polling Loop	Predictable timing and simpler synchronization
Servo Control Method	Software PWM, Hardware Timer	Servo Library (Hardware Timer)	Stable pulse timing with minimal overhead

Glove-Side Firmware Architecture

From finger bend → clean angle data → wireless packets (fast + smooth)

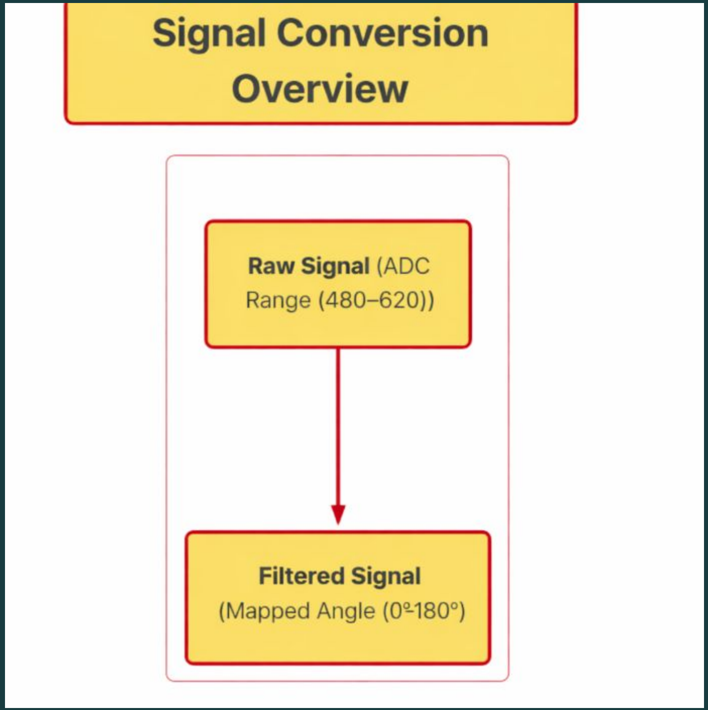
- Arduino Nano (Glove MCU)
- Analog Flex Sensor Acquisition
- User Calibration (Min/Max Mapping)
- Low-Pass Filtering (Noise Reduction)
- Angle Mapping (Sensor → Joint Angle)
- Structured Packet Formation (CRC Included)
- Wireless Transmission via nRF24L01 (200 Hz)



Calibration and Signal Conditioning

Signal conditioning pipeline ensuring stable, user-adaptive, low-latency motion tracking prior to wireless transmission

- Flex sensors vary per user
- Record min/max bend values during calibration
- Normalize sensor range (ADC → 0–100%)
- Apply low-pass filter to reduce noise
- Balance smoothness vs latency



Wireless Communication & Packet Structure

Low-latency, synchronized multi-finger transmission with hardware-level error detection

- nRF24L01 2.4 GHz transceiver
- 250 kbps data rate (stable mode)
- Custom multi-finger packet structure
- Built-in hardware CRC validation
- 50 Hz currently, scalable to 200 Hz

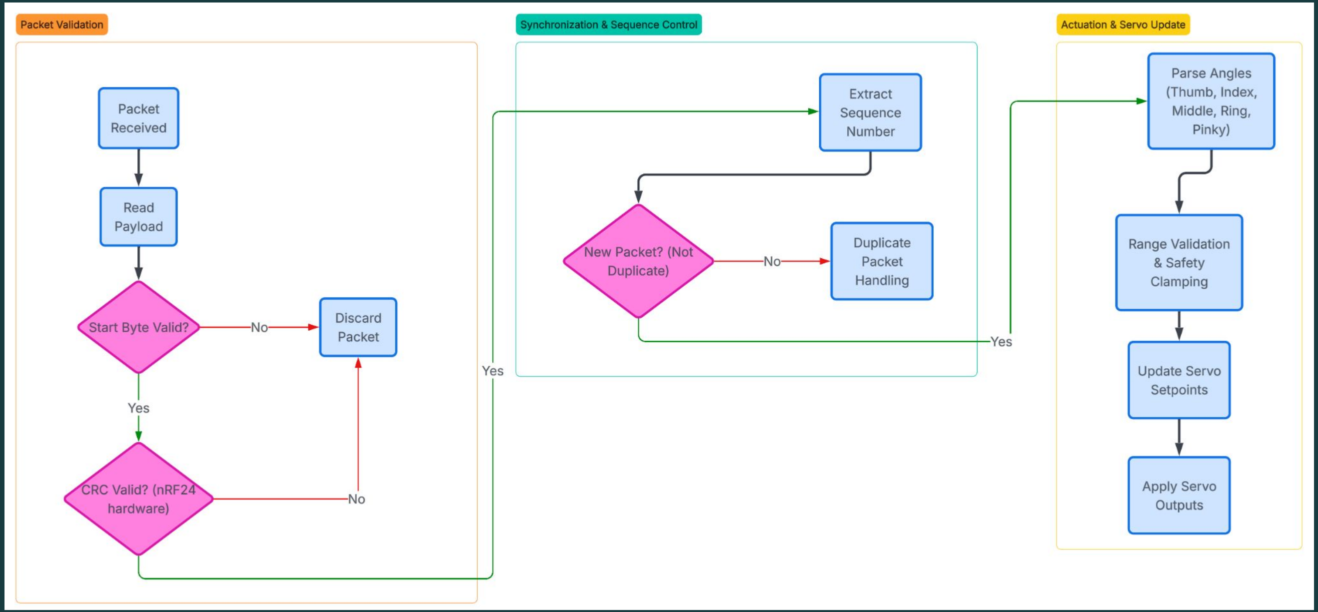
Wireless Data Packet (Glove → Receiver)

Start Byte (1 byte)	Sequence # (1 byte)	Thumb Angle (1 byte)	Index Angle (1 byte)	Middle Angle (1 byte)	Ring Angle (1 byte)	Pinky Angle (1 byte)	CRC (Hardware – nRF24)

Total Payload Size: 7 Bytes
CRC: Automatically handled by nRF24 hardware

Ensures synchronized, error-free angle reconstruction prior to servo actuation

Receiver-Side Processing & Validation



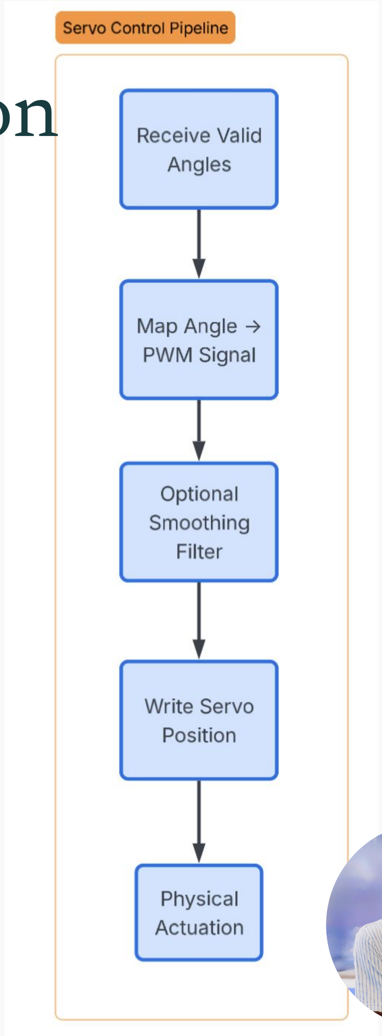
- Hardware CRC prevents corrupted data from reaching control logic
- Sequence tracking detects packet loss/duplicates
- Safety clamping enforces valid servo command bounds
- Failsafe: hold last valid position if packet rejected

Design Goal: Prevent corrupted, duplicated, or misaligned packets from destabilizing actuator motion and communication

Servo Control & Real-Time Actuation

Deterministic real-time angle-to-actuator conversion with stability safeguards

- Map received joint angles to servo PWM signals
- 0–180° angle → calibrated pulse width output
- Real-time update at 50 Hz currently, scalable to 200 Hz
- Motion smoothing to reduce jitter
- Failsafe: hold last valid position on packet loss



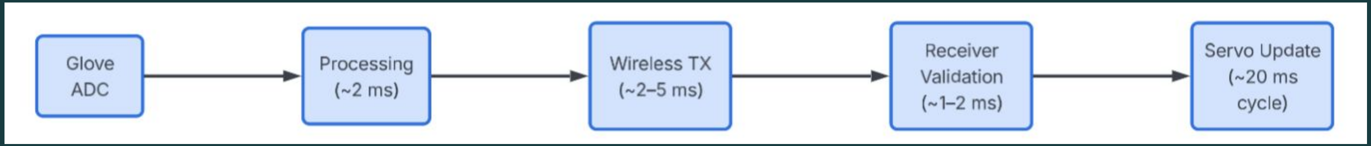
Performance Evaluation & System Metrics

Measured latency, update rate, and motion stability validation

Metric	Target Specification	Measured Result	Status
End-to-End Latency	< 15 ms	~12–18 ms (servo cycle limited)	Within Design Margin
System Update Rate	≥ 50 Hz	~50 Hz (20 ms loop)	Met
Wireless Data Integrity	CRC Validation Required	100% corrupted packets rejected	Met
Motion Stability	No visible jitter	Stable after tuning filter	Met
Packet Loss Handling	No unstable motion	Holds last valid position	Met

End-to-End Latency Breakdown

System meets real-time motion requirements while maintaining packet integrity and actuation stability



Design Challenges & Engineering Tradeoffs

Balancing latency, stability, reliability, and scalability

Technical Challenges

- Sensor noise and ADC variability
- Wireless packet timing unpredictability
- Servo PWM refresh cycle limiting latency
- Multi-finger synchronization
- Preventing unstable actuation on packet loss

Key Tradeoffs

- Filtering vs Responsiveness
- Data Rate vs Transmission Stability
- Update Frequency vs Servo Jitter
- Simplicity vs Protocol Robustness
- Immediate Reset vs Hold-Last-Position Safety

Design Objective: Maintain smooth, synchronized multi-finger motion under real-time constraints.

Opportunities for enhanced performance and robustness

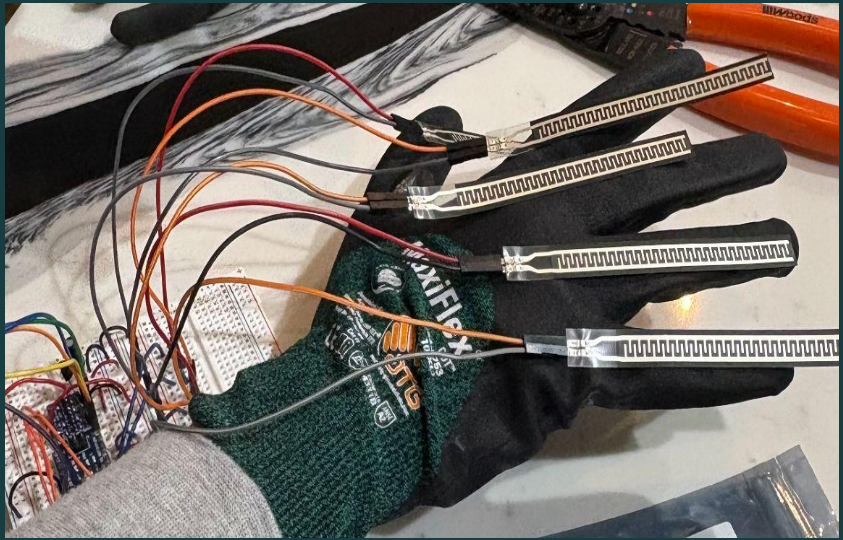
Improvements & System Scalability

- *Increase servo update frequency with higher-speed digital servos*
- *Add interpolation for smoother multi-finger motion*
- *Expand packet structure for additional sensors*
- *Implement sequence numbering for advanced packet ordering*
- *Implement adaptive filtering based on motion speed*

Mounting and Design Breakdown

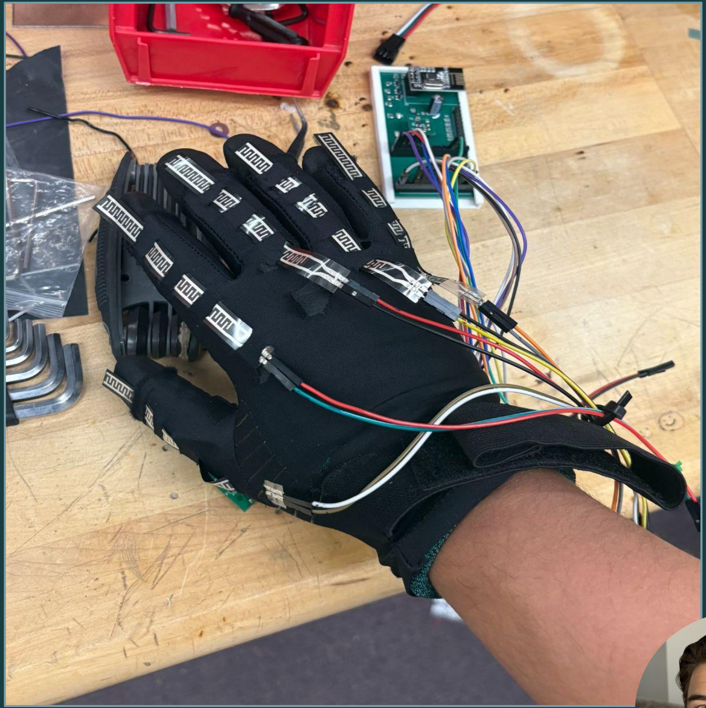
All enclosed:

- Glove will have two layers of material to ensure no contacts touch skin and are contained
- High dexterity was big factor for optimal performance
- Glove PCB
- Flex Sensors
- Velcro Arm band for Battery pack



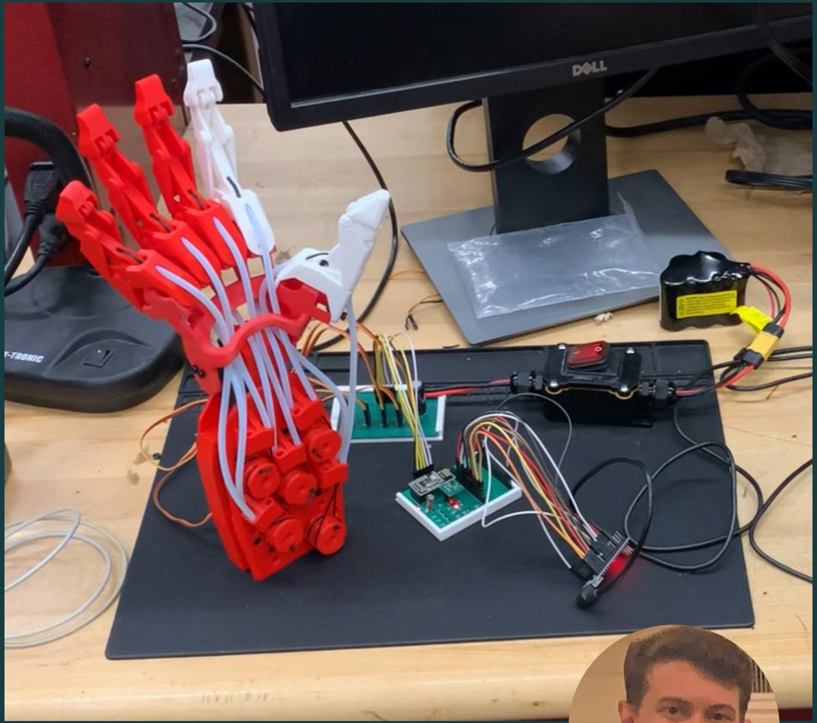
Glove Assembly

- Flex Sensors: ZD10 - 100
- The sensor was slowly bent from its minimum to maximum curvature while its analog output was monitored through the Arduino IDE serial monitor.
- Typical values recorded:
  Unbent state: ~980–1032
  Half bend: ~200–980
  Full bend: ~0–200
- By identifying the minimum and maximum values, the system could map the full flex range directly to 90-0° servo movement. This also allowed dynamic constraint of values within a safe operating window, reducing unnecessary overdriving of the motor.
- PCB mounted to wrist of glove via compression glove velcro strip



Hand Assembly

- Hand is entirely 3D printed using PLA filament
- Mechanical linkage system for fingers, so cords only need to be mounted at the base and not throughout the finger
- Cords will be threaded through PTFE tubing to increase slippage. Custom bearings let the servo pull in both directions
- Electronic components stored in custom container
- Fingers screwed in place to prevent splaying

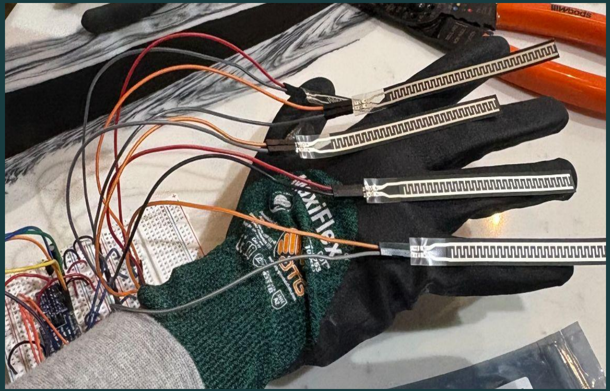


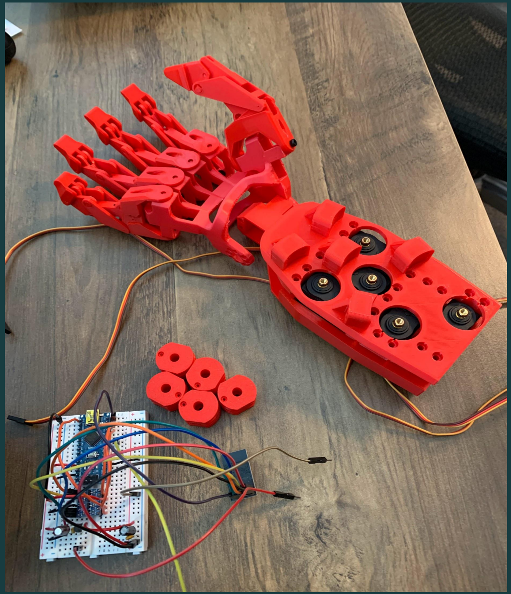
Prototyping and Testing

- Calibration tool utilized to adjust glove sensors to user's hand. Recalibration must occur if another person is to use it. Once calibration is complete, glove is moved to mobile power source and begins transmitting data to receiver

- Testing the hand alone is done through a testbench which sets the servos to various positions. This assists in tweaking fingers and bearings

- In testing, all specified gestures are performed, and distance is increased with each successful run to verify maximum range. Voltage is monitored during heavy use to verify operational range under load





Project Expenses

Part	Cost	Budget
ZD10-100 Flex Sensor	$41.20	
MG996r Servo Motors	$20.99	
nRF24l01	$7.89	
5100mAh 5C NiMH battery	$54.99	
Miady portable charger (2 pack)	$17.99	
Nylon cord	$5.99	
PTFE tube 2.5mm	$6.99	
Kill switch	$9.99	
PCBs	$152.47	
MCU PCBs	$118.54	
Total Expenses	$437.04	$500

- Max budget chosen based on generic necessary part prices, but also giving the team enough wiggle room to experiment or add better parts

- Parts not used in the live demo are not displayed. Any questions about them can be directed to the team

Timeline

Task	Start Date
PCB Assembly	3/2/26
PCB Testing	3/3/26
Complete shift from breadboard to PCB	3/5/26
Full RF comms tests	3/23/26
Full servo power test	4/1/26
Full assembly and final tests	4/14/26
Final Demo	4/23/26

- All tests were completed within the given windows
- At least a week of safety net time has been allotted in case of PCB failure/redesigns or any other issue that arises from final testing

Challenges and Future Improvements

- First generation of the 3.3V regulator PCB had a temperature flaw, the inductor used would get too hot and bottleneck performance. This was fixed by removing a resistor and adding a larger capacity inductor, allowing proper and expected performance

- During testing, servo motors were seen moving to default positions if there was loss of signal between the hand and glove. The index finger and thumb would end up colliding and pushing against one another, stripping bearings and putting pressure on fingers. Solution was to add a subroutine where the hand will deliberately move to a safe position should there be a signal loss.

- Future improvements to the project can be made to make it more streamlined and reliable

- The hand has two powers sources, one for the servos and one for the MCU. The MCU source needs to be activated manually before turning on the servos. A significant improvement would be to power both the servos and the MCU with one battery, resulting in a smaller footprint and simpler startups

- PCB layouts and be modified further to reduce noise the nRF24 signal will encounter. This can be done by reorienting parts and increasing isolation between circuitry

Thank you