GRIP

Glove for Real-time Imitation and Performance

Department of Electrical Engineering and Computer Science
University of Central Florida
EEL 4914 Senior Design I
Group 21

Group Advisor
Dr. Saleem Sahawneh

Review Committee
Dr. John Aedo, Dr. Sudeshna Pal, Dr. Wei Sun

Group Members
Rian Lowery, CPE
Christopher Marcoccia, EE
Kane McAgy, EE
Griseld Xhengo, EE

Table of Contents

List of Tables

List of Figures

Chapter 1 Executive Summary

Glove for Real-time Imitation and Performance (GRIP) is a system being developed as a senior design project for the University of Central Florida. GRIP is a design that allows for remote operation of a human-like hand. The operator would control the hand with a glove placed on their own hand with the remote hand mimicking the movements of the glove. With the demand for exploration in environments that do not support current methods of human exploration, a system that can mimic the movements of a human hand GRIP is the start of a solution that could provide a system that would both be precise and not place humans in danger.

GRIP uses wireless transmission from the glove the operator is wearing to the hand. The hand would be powered by a battery which would allow for remote operation independent of the location of the user. The signals from the glove would be processed by a microcontroller. The hand would have motors in order for dexterity in each of the fingers and the thumb. The glove would have flex sensors for the exact position of the user's hand to translate to the remote hand. There will be optimization in order to reduce latency between the glove and the hand so that the hand would be usable.

GRIP is intended for deployment to austere environments which would benefit from a human-like system. Its design however has room for further development in applications like prosthetics or can be upgraded into a full arm or robotic system that is modular to fill the needs of the end user and the environments they are in.

This report describes the process taken to design GRIP. It includes the theory and technology that is used in both the hand and the glove. The parts selection and the benefits along with upgrades that could be included depending on the situation the product would be operational in and the budget that is available. The hardware and software used will be detailed to provide an inside look on how they are customized for GRIP. The standards for the components and design will also be outlined. Final conclusions for use and improvements will also be presented.

Chapter 2 Project Description

2.1 Motivation and Background

There are many places in the world we explore despite the harsh conditions of the area. Whether it's the crushing depths of the ocean, the vacuum of space, or simple treacherous terrain, mankind has developed many different technologies to explore these places in person and remotely. With many incredible feats on our belts, there are still some limitations, specifically with remote vehicle exploration, when interaction with the environment is limited to what a machine is programmed to do. Many examples of this kind of technological interaction with exterior environments can be seen today, such as Mars rovers with collection trays and storage for specimens or remotely piloted submersibles with controllable appendages. Most of these, however, are either operated solely by the machine's protocols or are controlled by joysticks and keyboards. The Glove for Real-time Imitation and Performance (GRIP) arm project attempts to give remote operators more direct interactions with the environment they are studying. Whether it's attached to a study vehicle or in a stationary setting, by having GRIP be controlled by a human arm, more decisive movements can be made by the remote operator that can lead to better results. The capability and vision of using this in laboratories, perhaps for medical prosthesis technology and any application that can mitigate danger of a human real arm. Instead the use of the GRIP is a great alternative to most applications where harmful substances or environments are in play. Ultimately, the prototype aims to solve real world problems in a spectrum of possibilities.

2.2 Prior Related Work

2.2.1 Intelligent Programmable Prosthetic Arm

There were many existing products and one project that were similar to the idea of our project. The project was for UCF senior design in 2014 and is called the Intelligent Programmable Prosthetic Arm (IPPA). This project had many of the same ideas and components that we thought of for our arm, but with a few key differences. The IPPA focuses exclusively on the hand, whereas our project is planned to be a whole arm as a stretch goal. Having a whole arm option to connect the hand to would allow our project to have more versatility than the IPPA. The IPPA also seems to be controlled by an app to make it move, where our arm will mimic the user's arm via a glove. Overall, this previous project gives a good starting point and can provide ideas for components and mechanisms that can be used to improve our design.

2.2.2 NASA R5 Valkyrie

An advanced example was the NASA R5 also known as Valkyrie. This was a complete humanoid robot but we focused on the arms. The arms and hands have joints which use actuators to control movement. We are trying to have more advanced movement as one of our stretch goals especially in the fingers so that each one can move on its own. The R5 provides a good idea of what is needed in the hardware to make this movement possible. This is a very polished and expensive example so we may have trouble using the same exact components or methods NASA used but we might be able to find a work around using the R5 as inspiration. The full weight of the robot is three hundred pounds and the extent is full bodied and polished by experts. Our design will appear more crude than this to its full extent.

2.2.3 Shadow Robot Dexterous Hand

Focusing in on the glove that we are planning to have control of the arm there is a product called the Shadow Robot Dexterous Hand Series. This product mainly focuses on the hand but there is an option to add an arm. The hand is a copy of a human hand with four fingers and a thumb and is able to function with good precision. We are hoping to see how this hand interacts with the glove that controls it as we are trying to do the same thing. The communication between the hand and glove and how to have as little latency as possible will be something that this product can help with.
There are many more examples of robotic hands and arms but the three above are what we are focusing on and show that the concept is achievable and may offer solutions we have not thought about or ideas on how we may design our system. The ideas in our design remain original and cost-effective in premise. The latency is one of the biggest drawbacks that we want to improve upon. Our algorithm will be within a few seconds of delay and rotation, see the house of quality table for a visual. The shadow project was on the higher end of the projects due to higher-quality materials. Despite the simplicity of the device being a glorified arm piece, ours will include a shoulder mechanism as well.

2.3 Project Goals and Objectives

The GRIP team will have several basic and advanced goals to work towards and a singular stretch goal that will push the team to their limits. Each goal will be composed of multiple objects that must be completed in order for the goal to be achieved. Each objective will test a certain aspect of the goal being worked towards. For example, the homing sequence will have each joint of the arm reach its limit and take note of the range, with each joint's homing functionality being an objective. Each of these small objectives will work towards a bigger goal, which, once all goals are complete, will result

in a device that can be felt as an extension of the user rather than a tool that they simply wield. The basic goals basically ensure that the hand and glove can communicate and that the hand is functional as a basic human hand that can open and close with the ability of holding everyday objects.

The advanced goals are what we are really aiming to achieve when we visualize the project. It would allow the hand to function almost exactly as a human hand would and the latency being minimized would allow for realistic use for the end user. For the stretch goal there would be an arm that the hand is attached to. This is the natural Sprogression of the hand and would allow for more mobility and use. If the arm is incorporated the hand could move freely and if the movement goals are met it would replicate a human arm and hand.

Table 2.1 Project Goals Overview

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.

2.4 Project Features and Functions

The project will have several critical features that will function in tandem with each other. The glove itself will have a series of sensors along the upper arm that will take note of the position of the arm relative to the user being upright. The glove will send these positions wirelessly to a receiver on the bionic arm's PCB, which will then instruct it to move to the positions of the user. The glove will be powered with a battery attachment that can be changed out, along with a power switch. A calibration mode will be available to acquire the user's full range of motion before connecting to the bionic arm. Software-wise, GRIP will have algorithms in place that will keep it from moving in a way that is considered out of range, but if that fails, the arm will have hard stops for the servos in order to guarantee operation in the functional range. The GRIP arm will also have a homing sequence on power start to log its ranges.

2.5 Project Specifications

The project is split into two main components: the arm and the glove worn by the user. As such, each will have its own specifications that are tailored to its workload and design. Each part should be able to function independently of the other for testing purposes, with the ultimate goal of the glove controlling the arm being achieved once the connection between the two is made. The tables below illustrate the specs of both components separately.

Table 2.2 Arm Specifications

Parameter	Specification
Pinch force	≥ 10 N continuous
Finger closing time	≤ 300 ms
Control latency	≤ 15 ms at 200 Hz
Position repeatability	+/- 3 mm at fingertips over 10 cycles
Runtime on battery	≥ 1.5 hr

Forearm dimensions	0.79m x 0.047m x 0.172m
Bicep dimensions	9.75” x 3.25” x 3.25”
Palm dimensions	0.061m x 0.07m x 0.089m
Index finger dimensions	0.114m x 0.018m
Middle finger dimensions	0.124m x 0.018m
Ring finger dimensions	0.115m x 0.018m
Pinkie dimensions	0.094m x 0.018m
Thumb dimensions	0.102m 0.018m
Arm weight	≤ 30 lbs

Table 2.3 Glove Specifications

Parameter	**Specification**
Mass	≤ 150 g
Runtime	≥ 6 hr

Latency	≤ 10 ms
Link reliability	Packet loss ≤ 0.5% at 5 m LOS
Fit	One size
3-axis accelerometer	ICM-20948 9DoF IMU
Wireless transmitter	nRF24l01 wireless module
Finger	ZD10-100
MCU	Arduino NANO

Table 2.4 3D Printing Specifications

Parameter (Bambu Lab P1P)	**Specification**
Technology and Motion	Process FDM/ material extrusion. Top speed 500 mm/s Build volume: 256 x 256 x 256
Materials	Used PLA
Cameras and Monitoring	Chamber monitoring camera support: 1280x720 @0.5 fps
Connectivity and Control	Wi Fi, Bluetooth, Bambu-Bus, Micro SD(what we used) and the bambu handy app
Power, Size, Mass	Input 100-240 VAC, 50/60 Hz. Max power = 1000 Watts@ 220V

	Dimensions 386 x 389 x 458 mm Weight = 9.65 kG
Nozzle/Flow	All-metal hotend, max 300 °C. 0.4 mm nozzle included; optional 0.2/0.6/0.8 mm. (Max flow ~32 mm^3/s
Heated Bed and Plates	Max bed temperature = 100 degrees Celcius Plates are cool engineering high temperature type compatible

We fabricate glove and hand fixtures on bambu lab P1P, 3D printed with the above specifications. The P1P with PLA filament for all parts.
The PLA used is 0.25 kg spool weight, 1.75mm diameter, net weight of the entire spool is 1 KG.

2.6 Hardware Diagram

This diagram covers both the glove and the actual arm connections. We intend to implement servos to mimic joint and shoulder movement. The intention is the use of flex sensors and PCB designs that are acceptable sizes for the average human. For the glove diagram below the main components are the sensors, wireless communication, and controller. The rest of the components are to support those components or to provide power. For the hand or arm portion it is similar. There is more of a demand for power so there would need to be a larger power source and there is an increased demand for current that the motors that will move the fingers need. This will need to be taken into account when designing the PCB for the hand.

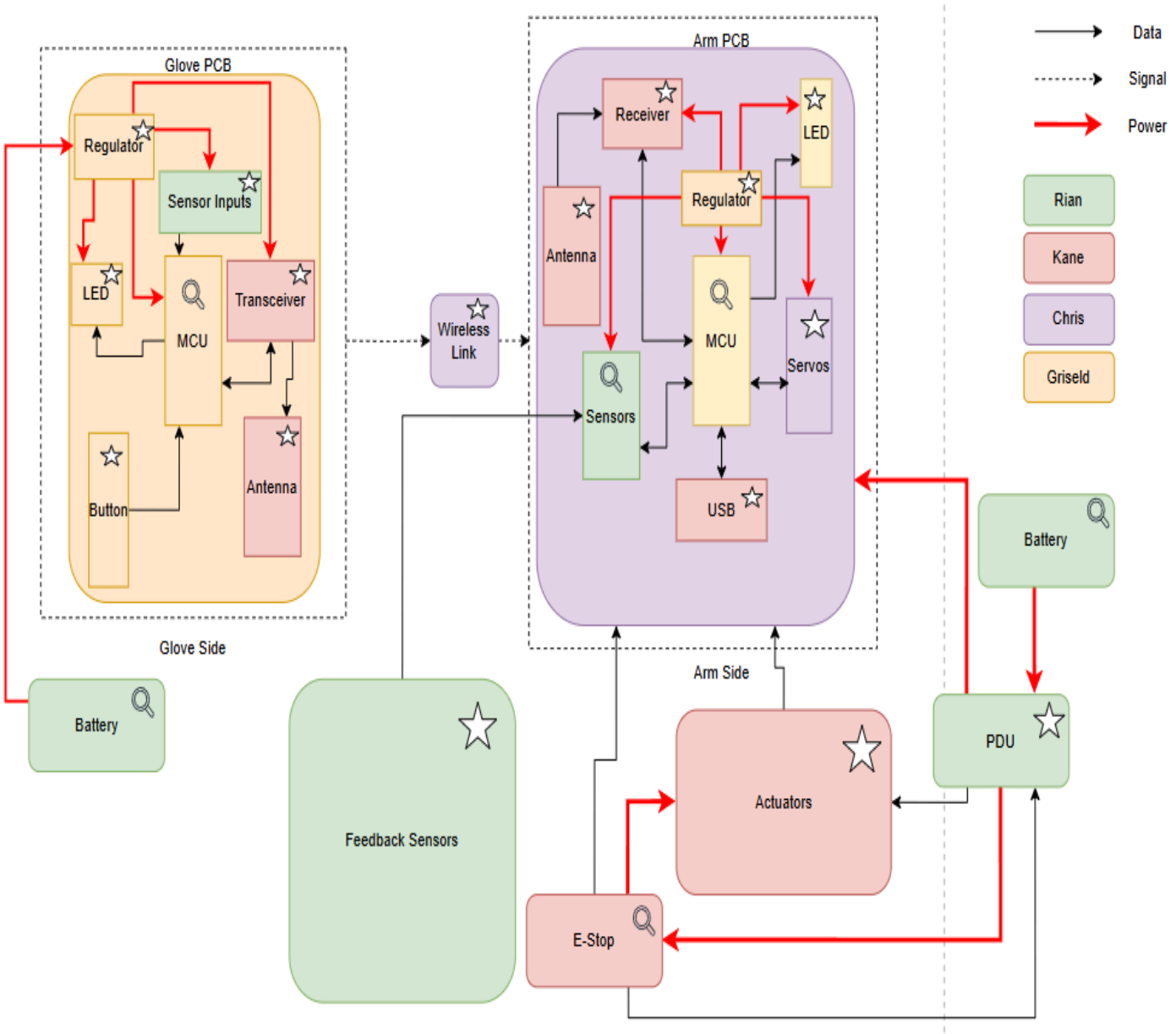


Figure 2.1 Hardware Block Diagram

2.7 Software Design

The GRIP software converts finger-bend measurements from the glove into coordinated servo motion on the robotic hand. When powered on, the glove's Arduino Nano initializes and begins acquiring analog readings from the flex sensors. A short calibration routine establishes each user's neutral and fully bent reference positions. The system then filters the raw signals and maps them to estimated joint-angle values. These values are packaged into a wireless data message, protected with a CRC checksum, and transmitted via the NRF24L01.

On the receiver side, the second Arduino Nano continuously listens for packets, verifies their integrity, and extracts the joint-angle commands. Before driving the servos, the firmware applies safety limits to prevent motion beyond mechanical bounds. The final angles are output as PWM signals to five finger servos, reproducing the user's hand pose in real time. A communication watchdog enforces a fail-safe state, returning the hand to neutral if transmission is lost. This modular software pipeline—acquisition, conditioning, mapping, transmission, validation, and actuation—ensures smooth and responsive glove-to-hand control for SD1. Shoulder control will be added in SD2.

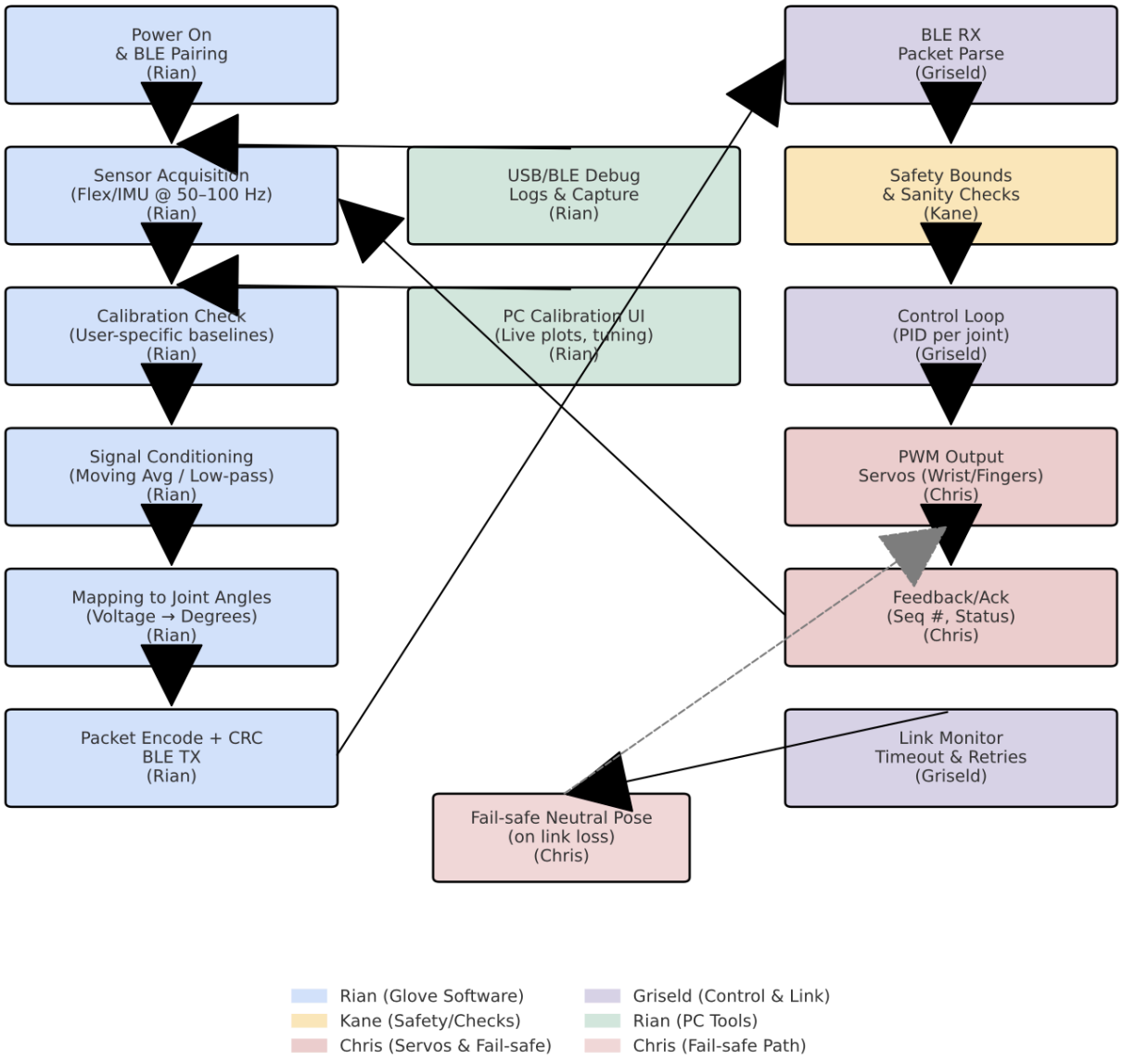
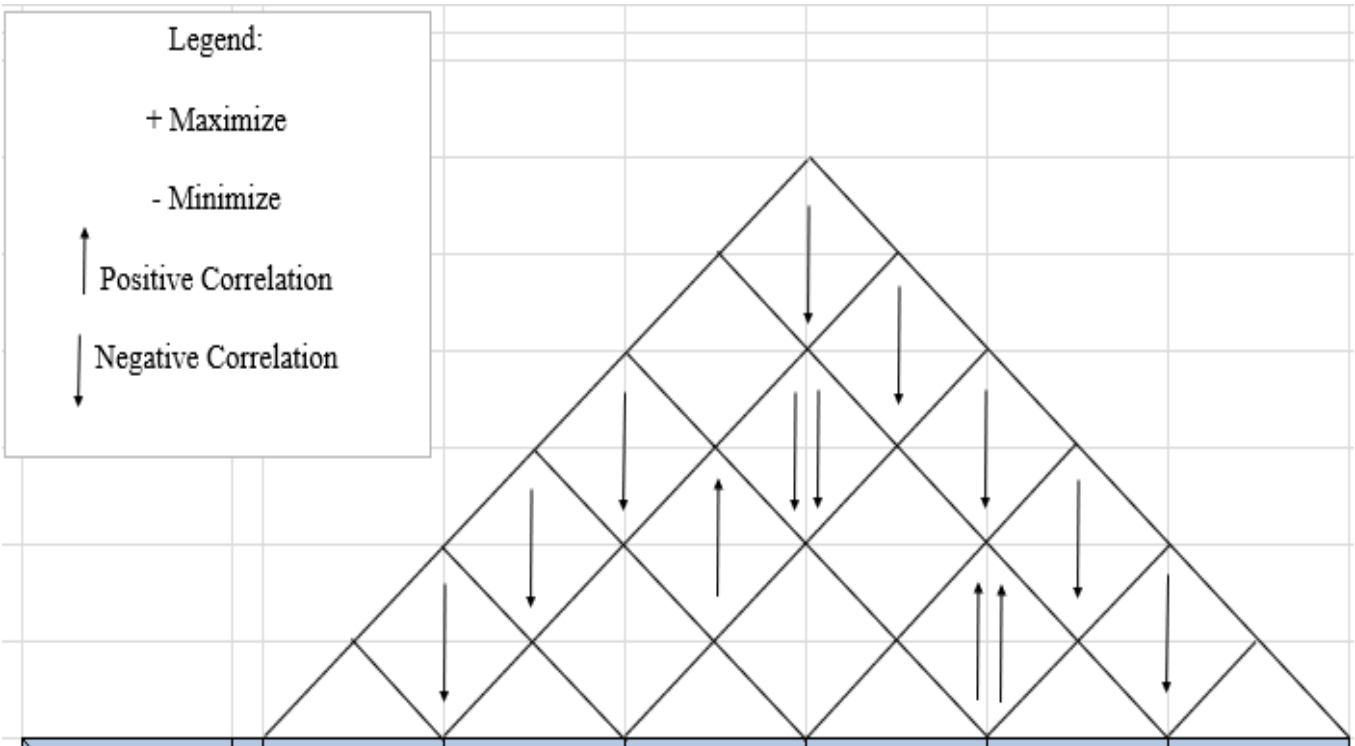


Figure 2.2 Software Diagram

2.8 House of Quality



Marketing Requirements \ Engineering Requirements	Direction of Desirability	Battery Life	Cost	Weight	Latency	Accuracy	Temperature
Direction of Desirability	+	+	-	-	-	+	-
Portability	+	↑	↓	↓↓			
Cost	-	↓		↓	↓↓	↓	↓
Ease of Use	+	↑	↓	↑↑	↑↑	↑	↑
Data Accuracy	+		↓		↑	↑↑	
Latency	-					↑	
Battery Life	+		↓				
Target		> 30 Min	< 1500	< 25 lbs	< 2 seconds	± 5% from reference	No Overheating

Figure 2.3 House of Quality

Chapter 3 Research and Investigation

3.1 Technology Comparison

3.1.1 MCU or FPGA

Microcontroller units (MCU) can be thought of as a small computer on a chip. They consist of a processor, memory, general purpose input/output pins (GPIO), and I2C/SPI/UART. With all of those the MCU can control hardware directly. They can run control with low latency and power consumption.

MCUs are effective when real-time input and output is needed along with efficiency. They can for example read sensors, control wireless communication, and drive actuators which is useful for this project. MCUs are often cost effective compared to more powerful options. The benefit versus the cost makes them very effective for most projects that require a controller.

In terms of use MCUs are very user friendly. Since they are a popular choice for any project there is plenty of documentation to refer to in any use case and are available from a wide range of companies. When it comes to programming them they usually use well established languages that are either well documented or widely known so when it comes to setting them up with the appropriate code or debugging it is simple. Since we most likely will have some experience with some form of MCU between all the members of the group they will be easy to work with. All of this means that a MCU would be appropriate for use in this project since it covers all of the bases that we have.

Another option would be Field Programmable Gate Arrays (FPGA). FPGAs are chips that allow digital control of hardware instead of the program in the code like the MCUs. It uses logic blocks that can be programmed using Verilog or VHDL to build the hardware circuits. FPGAs would be more capable than MCUs and would be excellent for timing and low latency.

The tradeoff is that they are more complex than MCUs. Designing the hardware in VHDL or Verilog has a steeper learning curve than the knowledge required for the MCUs. Also the verification required for the simulation can be very time consuming and troubleshooting small errors can be difficult. In short FPGAs are not as user friendly as a MCU would be. They also come at a higher cost than readily available MCUs. For this project a FPGA would provide greater power and control but it would be overkill for this application.

For this project we decided that a MCU was appropriate for the requirements of the project. Microcontrollers would give a good mix of real-time control, Input/output options, and not take too much power for the GRIP application. Since our project has time critical parameters the MCUs predictability is a positive. They also keep the latency low which was one of our major goals. Other alternatives would either under or over shoot our need for this project. In terms of cost MCUs also excel in that category as we are going to need two. One will be for the glove and one will be for the hand. There is also the ability to integrate our stretch goal since MCUs have the extra capability that the arm would need along with the hand.

Table 3.1.1 MCU vs. FPGA

	MCU	FPGA
Power consumption	X	
Ease of use	X	
Processing power		X
Cost	X	
Documentation	X	

3.1.1.1 Arduino Nano

The Nano is small so it would be easy to mount for our project. The programming is simple in the Arduino IDE with plenty of available libraries. The peripherals we would need are all on the board including I2C and GPIO which covers the tasks we would need. It is simple and boots instantly with tight control which means latency would be reduced. Since there is a large community support network for Arduino troubleshooting is made easy. If spares are needed and since we will need two they are easy to source and inexpensive.

There are some cons as well to accompany the pros. There is 32 KB of flash and 2KB of RAM which may limit some features or libraries that can be used. For the UART hardware it shares with the USB which can cause limitations. The CPU isn't ideal for very heavy calculations or any high rate telemetry. Since the IO is 5 volts while many sensors are 3.3 volts only we may need some extra components in order to make the Nano viable. For the 3.3 volt pin there is not enough current to power many peripherals so that might need to be boosted. There is also no built in wireless so we likely would have to add one in ourselves.

3.1.1.2 Arduino Mega 2560

The mega gives a larger number of I/O pins with multiple UARTs and more timers. With all these extra features it is simple to attach extra sensors, displays, and devices without overwhelming the controller or running out of pins. Since it is in the Arduino family the large amount of libraries and support makes it simple to use and troubleshoot. The extra memory offers more room for larger amounts of code.

Since this MCU has all of those extra features it means that the footprint is much larger. It will also be heavier meaning it is harder to place in compact spaces or mount onto a glove for example. It still uses the same 8-bits 5 volts as the Nano which limits its performance and would require a solution for 3.3 volt components. There is also a higher power draw and more power is required.

3.1.1.3 Arduino Nano 33 IoT

This board keeps the small footprint of the Nano which makes it ideal for our project. The upgrade comes in the form of a 32-bit core with more memory and built in Wi-Fi which makes wireless integration possible. Again since it is the Arduino family the support and ease of use is excellent. This also has 3.3 volts instead of 5 volts which means it will match many sensors without any additional work.

The downside of the 3.3 volts is that any 5 volt peripherals will now need translation before they can be used. Any heavy calculations or high rate control will still feel strained with this controller.

3.1.1.4 ESP32 Dev Kit

The ESP32 has a lot packaged onto one chip. It features dual cores, multiple UARTs, timers, and Wi-Fi enabled. THis makes it great for combining wireless and real-time control on a single board. The community support is very good and there are plenty of libraries that can be used. In terms of the cost it is very reasonable and there are multiple choices of boards depending on your requirements.

Some of the cons are there is an increase in complexity with the Wi-Fi stacks and RTOS scheduling. This controller is also 3.3 volt only so if 5 volt devices are going to be added there needs to be special consideration taken. RF layout, and sleep/wake behavior need special attention. Some libraries also require special attention when they are used more than they would in the Arduinos.

3.1.1.5 STM32 Nucleo

This board brings great support for peripherals with good timer sets and solid UART that can support lots of sensors or other connected components. The Nucleo can run flexible control loops with precise timing making for low jitter clean pipelines. The community is large which helps for programming and trouble shooting. There are plenty of reference designs for motor control, power conversion, and sensing so you would not have to start from scratch. It seems to be focused on the support of peripheral components and make it easy to connect them.

There is a steeper learning curve than the Arduino line of products. Getting the clock and interrupts right takes more time than it would with other MCUs. The Nucleo is also a larger size than the Arduino Nano for example which makes placing it in tighter applications difficult. Everything is 3.3 volts only which would require shifting for 5 volt peripherals. The complexity and size is the tradeoff for the extra power and features.

3.1.1.6 Teensy 4.1

This chip has large performance capability with a M7 core and plenty of RAM and flash memory while still supporting Arduino tools for iteration. It excels at control, signal processing, and telemetry. There are many ports and high speed I2C which means that this on board can consolidate what might need multiple controllers.

Since it has such high capability this board can be overkill for some systems. The board is also larger than the Nano boards which makes it less feasible to place in applications where footprint matters. It is 3.3 volt only so there will need to be care taken with 5 volt components. All of the extra features also might add integration that is not needed for some projects. Cost is higher than entry level MCUs that would work fine for most projects. Power consumption can climb if not properly managed. It allows for future upgrades to a project but it may also end up having complexity that is not used.

3.1.1.7 Raspberry Pi Pico

The Pico is low cost and widely available and provides good function despite the low cost. It has dual cores and plenty of RAM with programmable input/output blocks. This allows for custom wave forms and precise timing. The support and programming is good and has a low barrier of entry. There are plenty of resources for trouble shooting or libraries that can be used in order to not start from scratch.

There is no built-in wireless so if communication is required there must be a separated component added. Logic is 3.3 volts only so there would need to be special care taken for 5 volt components. There is a new programming model that can be tricky if you are unfamiliar with it beforehand. The board footprint is larger than a Nano if size is of concern. If compute heavy tasks are required a more powerful platform would be required.

Table 3.1.2 MCU Comparison

	Arduino Nano	Arduino Mega	Nano 33	ESP32	STM32	Teensy	Pi Pico
Wireless			X	X			
Ease of use	X	X	X	X		X	
Power	X		X				X
Size	X		X				
USB	X	X	X	X	X	X	X
Cost	X	X		X			X
3.3V friendly			X	X	X	X	X
5V tolerant	X	X					

3.1.2 Smart Servos or Micro Linear Actuators

Smart servos are all in one with a motor, gearbox, sensors, and motor drive in a single housing. It also includes a digital control bus for control of those components. Setpoints will be sent to the motor and telemetry will be received so wiring would stay clean since it is an all in one package. There is a horn on the servo that a mimic tendon can be attached to in order to allow the motor to control the design it is attached to. The servo takes simple commands and can hold position and on some units can report status. The wiring is minimal, set up is fast, and from the small footprint they can be placed in applications that require tight spacing. The torque and speed specs are easy to modify and there is a range of off the shelf parts such as horns and brackets if modification is needed to make the motor fit a specific application.

Some of the tradeoffs are backlash and thermal limits. There is also the question of how many servos will need to be used to achieve the actuation that is desired. Holding high grip force for longer periods of time can also cause small servos to heat up. If there are multiple servos linked together there is a power burst that will need to be accounted for.

Linear actuators convert motor rotation into straight line motion through a screw or rack and pinion. This works by pulling the simulated tendon in a straight path. These can be bought or built if customization is required. The advantage to this is having a preloaded screw controlling the movement gives a more solid feel over a rotary gear. This also makes calibration simple because of the smooth path of travel and would keep the simulated tendons that control the fingers tidy.

The costs are size, complexity, and tuning. Even the smallest footprint of these motors would be larger than the servo motors and if customized there would be other parts to account for. There would also be more extra work required because each actuator would need a driver and feedback with more control work than an all in one package like the servos. The price would also be higher and power draw can be significant along with wear on the components if they are not properly assembled, aligned, and lubricated.

With all this in mind the smart servos is what we will use. They come in a complete package with simple controls and can still be modified if required. The cost is reasonable and it is an established component in many projects which means any trouble shooting and documentation is available.

Table 3.1.3 Smart Servos vs. Micro Linear Actuators

	Smart Servos	Micro Linear Actuators
Wiring	X	
Packaging	X	
Stiffness		X
Control	X	
Power bursts	X	
Cost	X	

3.1.2.1 MG996R

This is a common standard size hobby servo with a metal gear train and a simple three wire interface including power, ground, and signal. It accepts a pulse width command and has an internal potentiometer so there is no need for a motor driver or encoder. It is inexpensive and widely available. There is a large support network in case of trouble shooting requirements or documentation. It delivers useful stall torque for the tendon spools that will be used. There is also a large selection of horns, brackets, and gears for customization and repair purposes.

The tradeoffs are what can be expected from a hobby servo motor. There can be some gear backlash, motor whine, and variance from unit to unit in deadband and holding behavior and deadband. There is no host side telemetry so you would have to design relying on limit switches or mechanical stops. Currents can be high for short bursts so the voltage rail must be stiff and the wiring short. Grounds must be accurate especially when using multiple servos such as in our case. Long term clamping of the hand would also increase heat and wear on the motors.

3.1.2.2 Micro PWM Servos

Micro servos shrink the servo layout into a very small package that would be effective in situations that require a small footprint.They are good for lightweight degrees of freedom and short simulated tendons. Integration is simple, only requiring a single signal wire per motor. The power distribution will be 5 volts. Their low mass reduces inertia giving better motion to the fingers. The cost is very low which makes having spares easy in case of wear from long term clamping or heavy loads.

The compromises are torque and durability. Many micro servers use plastic gears and smaller less powerful motors so high grip forces or impacts will wear out the components faster. They provide no data back to the controller so there is calibration drift and gear wear. Power distribution can also be touchy and as in the case with this project a handful of motors will brown out a weak power source.

3.1.2.3 Digital PWM Servos

Digital PWM servos keep the same three wire interface but run a faster internal control loop and higher up update rate to the motor which translates to tighter holding torque and crisper transient response. Many of these in this class add metal geartrains, stronger motors, and configurable endpoints. If you want the simple wiring of hobby servos but

want firmer grip as in our project and more repeatable positioning this class is a good upgrade with minimal code changes.

There are still PWM only and feedback blind to the host so the wiring scales linearly and the current, temperature , or position telemetry for safety or logging . Peak currents are higher than micro or standard servos so a rail from 6-7.4 volts is required. Bulk capacitance is also required to absorb any transients. Thermal limits also apply with long holds near stall. They will heat the case which limits the amount of time the grip can be active. There is also still backlash but less than some of the cheaper servos

3.1.2.4 TTL Bus Smart Servos

These integrate a motor, gearbox, and position sensor with a digital bus that is UART like so power and two signal lines will be routed through a string of actuators and address them by ID. The largest benefits are built in feedback and commanded motion profiles which simplify synchronization and safety. Because the command signal is digital rather than pulse widths, timing is robust and jitter free and the harness inside the compact hand stays cleaner than running separate lines to everything.

Costs include each vendor using its own unique protocol and lower torque density than larger units at the same price point. There would need to be a separate transceiver added to the controller board and attention will need to be paid to the grounding and stub lengths on the chain to avoid communication issues. While feedback helps absolute encoder resolution and gearbox quality still set the repeatability floor and backlash will still be present

3.1.2.5 RS-485/Can

At the higher end these smart servos move to 12-24 volt rails and absolute encoders with robust packets, error checking, and comprehensive telemetry. They expose configurable PID gains, software limits, and profile generators making syncing multi joint motion and safety interlocks far easier. Chaining together a few actuators over a proper bus means a single pair can serve an entire finger cluster and running at the voltage it does lowers current for the same mechanical power which improves voltage stability.

The trade offs are cost, size, and ecosystem commitment. These units cost more and can be larger than other servos and the budget will be higher for proper connectors and hubs. Bus setup is an extra bring up step and though backlash is typically lower thanks to better gears and bearings but it is not perfect.

Table 3.1.4 Servo motor comparisons

	MG996R	Micro PWM	Digital PWM	TTL Bus Smart	RS-485
Control signal	PWM	PWM	PWM	UART	RS-485
Voltage family	5-6 V	5-6 V	6-7.4 V	5-7.4 V	12-24 V
Feedback				X	X
Torque density			X		X
Backlash	Metal gears	Plastic gears	Mixed materials		Best in class
Holding accuracy			X	X	X
Telemetry				X	X
Setup	X	X	X		
Scalability				X	X
Cost	Low	Very low	Mid	Mid	High
Power stress					X

3.1.3 Transceiver

A transceiver is a compact device that combines signal generation, modulation, and power amplification stages needed to transmit data with the low noise amplification and demodulation stages required to receive it. This presents a simple digital interface to the host controller. In wireless variants the RF front end is coupled to a baseband modem that implements the chosen layout along with framing and error checking. Contemporary parts integrate hardware, addressing, CRC, and automatic acknowledgement so the host exchanges payloads over UART while the transceiver manages the other details. Wired transceivers perform an important role at the physical level translating logic level signals into balanced line drivers with fail safe biasing and termination options.

Operational behavior is governed by band selection, link budget, and protocol timing. Devices operating in the 2.4 GHz ISM band offer small antennas and higher data rates in exchange for coexistence with Wi-Fi and Bluetooth. Sub-GHz parts prioritize penetration

and range at lower throughputs. The link budget is set by transmit power, receiver sensitivity, and antenna efficiency. This determines practical range and resilience while the protocol timing model sets latency and jitter. Together these characteristics define whether a transceiver is better suited to low-latency control traffic, bulk telemetry, or long-reach monitoring.

From a system perspective transceivers are compact building blocks that abstract complex analog and RF concerns into repeatable digital behaviors. They are typically powered from a regulated 3.3 V rail and draw short current bursts during transmission or receive and require modest board level accommodation such as local decoupling, a clear keep-out around the antenna region for modules, and clean reference routing for their digital interface. Once configured channel, data rate, addressing, and optional security the device presents a predictable packet service to the firmware: short payloads are queued, transferred across the link with hardware assistance, and delivered with integrity checks and delivery status allowing the application to treat the transceiver as a reliable conduit within the overall control architecture

3.1.3.1 nRF24l01

The nRF24L01 is a tiny 2.4 GHz transceiver that talks over SPI and pushes short packets with auto-ack/auto-retransmit, hardware FIFOs, and selectable on-air data rates. It's purpose-built for low-latency point-to-point or star links at a few meters to tens of meters of sight perfect for a glove streaming pose data at 100 to 200 Hz. The register model is simple addressing pipes that make multiple node setups manageable, and modules are cheap and compact. With a decent packet format, you can get very smooth deterministic updates.

Some of the tradeoffs are it is 2.4 GHz, so it shares spectrum with Wi-Fi/BLE and can see interference. To solve this, pick a clean channel and consider the 250 kbps mode for better link budget. Most breakout boards are 3.3 V only on VCC and I/O so level-shift if your MCU is 5 V and add good decoupling right at the module. Long range variants can help but they draw more current and need antenna clearance and grounding attention. Security isn't built in beyond addressing and CRC.

3.1.3.2 ESP-NOW

ESP-NOW is a peer-to-peer protocol built into ESP32 radios that provides very low overhead packet exchange without associating them with a Wi-Fi access point. It supports broadcast and unicasts and works with PCB or external antennas and runs on hardware that already offers multiple UART interfaces along with ample RAM for logs and

diagnostics. For glove to hand control, it can deliver snappy updates while keeping firmware simple on both ends.

The trade-offs are power, and stack complexity compared to a tiny transceiver. An ESP32 is a full SoC radio so idle and transmit currents are higher than an nRF24L01 and you must manage tasks and radio activity to avoid control-loop jitter. All I/O is 3.3 V. RF layout is more sensitive, and the protocol is vendor specific rather than generic. It is attractive when you also want convenient logging, over-the-air features, or a path to Wi-Fi networking for the narrow job of fast control packets alone, a dedicated packet radio remains simpler

3.1.3.3 Bluetooth Low Energy

BLE provides standardized phone friendly connectivity with low peripheral power, mature stacks, and widely available modules. It is attractive for user interfaces and calibration via a mobile app and field debugging. Modern BLE 5 options such as 2M and coded PHYs let you bias toward throughput or range and pairing establish a baseline for link security without extra application work.

For continuous teleoperation BLE's connection intervals, supervision timeouts, and GATT transaction model introduce more latency and jitter than a purpose-built packet link. While tuning can improve performance, holding a deterministic 100–200 Hz command stream is harder than with simple 2.4 GHz packet radios. Many modules are 3.3 V only and the API surface is broader, so firmware effort rises if BLE is used as the primary real-time control link.

A pragmatic approach is to treat BLE as an auxiliary channel. Use it for configuration, calibration, and diagnostics while the primary control runs over a lower-latency radio. If you must rely on BLE for control, choose a short connection interval and minimize GATT overhead with a custom characteristic carrying a compact binary payload and measure actual end-to-end timing under realistic interference before finalizing the design.

3.1.3.4 XBee

XBee modules encapsulate IEEE 802.15.4 radios in a convenient form factor with options for transparent serial bridging and full Zigbee networking stacks. The 802.15.4 physical layer provides low-power operation, modest data rates, and robust short-range performance, while higher-layer profiles on certain models enable mesh capabilities such as routing and network addressing. The module ecosystem includes through-hole and surface-mount packages, integrated or external antennas, and standardized pinouts.

Most XBee families operate from a 3.3 V supply and expose UART and sometimes SPI interfaces, with configurable parameters via AT commands or a binary API. Radio variants cover both 2.4 GHz and sub-GHz bands, allowing selection based on range, penetration, and regulatory region. Power consumption and latency characteristics depend on the specific stack enabled, with transparent mode minimizing overhead and mesh features adding store-and-forward behavior.

This platform is frequently used where simple cable-replacement behavior is desired at the outset, with the option to scale to small networks using addressing and routing when needed. The combination of mature documentation, certification options, and accessory ecosystem makes it a stable choice for embedded links that prioritize reliability over raw throughput.

3.1.3.5 LoRa

LoRa devices implement a chirp spread-spectrum modulation in sub-GHz ISM bands such as 868/915 MHz, achieving very high receiver sensitivity and long link distances with small payloads. The physical layer supports configurable bandwidths, spreading factors, and coding rates, enabling a tunable trade-off between data rate and robustness. Common transceiver ICs present an SPI interface with registers for modem configuration, packet framing, and RSSI/SNR reporting.

Electrical characteristics are typical of sub-GHz radios: a regulated 3.3 V rail, discrete front-end filtering and matching, and antenna systems sized by the longer wavelength. Current draw rises with transmitted power and decreases with higher spreading factors, and duty cycle or dwell time regulations may apply depending on region. Modules vary from compact integrated solutions with onboard matching to larger designs accommodating SAW filters and external antennas.

The technology is well matched to low-throughput telemetry and monitoring where coverage and building penetration are prioritized over latency and bandwidth. It underpins many low-power wide-area networks and point-to-point links that require long-range operation, periodic reporting, and resilience to multipath and attenuation in cluttered environments.

Table 3.1.5 Transceiver comparisons

	nRF24L01	ESP-NOW	BLE	XBee	LoRa
Operating band	2.4 Ghz	2.4 GHz	2.4 GHz	2.4 GHz or sub	Sub GHz
Host	SPI	ESP32	UART/SPI	UART/SPI	SPI

interface					
Supply level	3.3 V	3.3 V	3.3 V	3.3 V	3.3 V
On air rate	X	X			
Latency suitability	X	X			
Indoor range	Short	Short	Short	Medium	Long
Packet assist in hardware	X	X			
Complexity	X				
Antenna choices	X	X	X		
Cost	X	X	X		

3.1.4 Power Electronics Technology Comparison

Power electronics play a major role in the GRIP system because both the glove and the hand rely on stable, low-noise power rails for sensors, radios, and servos. Each subsystem has different requirements: the glove requires smaller, cleaner power for sensors and the microcontroller, while the hand requires high burst current for the MG996R motors. Because of this split, multiple power regulation technologies were considered for both sides of the system. The main categories evaluated were linear regulators, buck converters, boost converters, and buck-boost converters. Each technology offers different trade-offs in efficiency, cost, complexity, and output stability.

Linear regulators were the simplest option to evaluate. They provide clean output with minimal switching noise, which is ideal for sensitive analog components such as flex sensors and ADC pins. Their major disadvantage is efficiency, especially when stepping down from higher voltages. Any excess voltage is dissipated as heat, which becomes problematic when the input voltage varies or when powering components with higher current demands. While linear regulators could technically support the glove side due to its low-power requirements, they would be insufficient for the MG996R servos which regularly draw 1–2 A spikes under load. A linear regulator in that environment would overheat or collapse its output almost immediately.

Buck converters were the next category considered. A buck converter steps down voltage using high-frequency switching, allowing far better efficiency than a linear regulator. They also handle large load transients better when designed properly. For the hand portion of GRIP, this is important because multiple servos can pull large currents simultaneously. The key disadvantage is the switching ripple, which can interfere with sensitive signals if the layout and filtering are not handled correctly. For the glove, this can cause inaccurate flex sensor readings if the ADC ground or VCC rail becomes noisy. Cost and board complexity are also higher than linear regulators, although buck converters remain the preferred option when high current is required.

Boost converters were considered for designs using a single 3.7 V 18650 cell. A boost converter increases voltage, making it suitable for powering 5–7 V servos from a lower battery source. However, boost converters must supply enough current to match the servo's needs, meaning that the input current increases proportionally. This places stress on the battery and raises the chance of voltage sag during servo startup. Boost converters also introduce switching noise, and under heavy loads efficiency can drop. For this project, boost converters were ruled out for the hand but remain a potential option for powering glove electronics from a single-cell source.

Buck-boost converters combine both step-up and step-down capabilities, maintaining a regulated voltage even when the battery swings above or below the target level. This is useful with lithium-ion batteries, whose voltage ranges from 4.2 V (full) down to 3.0 V (empty). A buck-boost regulator can keep the servo rail stable across the entire discharge curve. The tradeoff is increased schematic complexity, cost, and potential EMI. Because MG996R servos require a steady 6 V rail and experience sharp load transients, a buck-boost converter presents the most stable and robust approach. Devices like the LM5176/LM5177 controllers or LM51770/LM51772 monolithic ICs are well-suited for these conditions.

Overall, the glove can operate effectively using simpler regulation methods, while the hand requires a high-efficiency, high current switching solution. The analysis led to the conclusion that buck-boost or strong buck regulators are the most appropriate options for GRIP's servo power distribution, whereas linear regulators or low-power buck converters remain suitable for glove sensors and control electronics.

3.1.5 MOSFET

A metal oxide semiconductor field effect transistor (MOSFET) is a voltage controlled semiconductor switch in which an electric field at the insulated gate modulates the conductivity of a channel between source and drain terminals. The device is fabricated on a doped silicon substrate with a thin gate oxide separating the gate electrode from the channel region. Applying a gate-to-source voltage above the device's threshold attracts

carriers into the channel and forms a low-resistance path. Removing that voltage returns the device to a high impedance state. Two polarities are available: N channel devices conduct with electrons as majority carriers and turn on for positive gate bias, while P channel devices conduct with holes and turn on for negative gate bias relative to the source. Most power and signal switches used in low-voltage electronics are enhancement-mode, where the default state is off.

Key electrical parameters describe how a MOSFET behaves in circuits. The drain–source voltage rating sets the maximum blocking voltage in the off state, and the continuous and pulsed drain current ratings reflect conduction limits with specified thermal conditions. Gate-to-source threshold voltage defines the onset of conduction but does not guarantee low loss; on-resistance is specified at higher gate drive levels such as 2.5 V, 4.5 V, or 10 V and determines conduction losses when the device is on. Dynamic behavior is governed by gate charge and the device capacitances between terminals, which influence how quickly the device can switch for a given driver. Every MOSFET includes an intrinsic body diode between body and drain regions; its recovery behavior and forward voltage are important in switching applications that reverse current.

3.1.5.1 BSS138

BSS138 is a small-signal N channel enhancement MOSFET commonly supplied in SOT-23 packaging. It is intended for low-voltage applications and features a low gate threshold, allowing the device to turn on with a logic-level gate drive. Typical maximum ratings include a drain-source voltage in the 50–60 V class and a continuous drain current in the few hundred milliampere range, reflecting its role as a signal-level or light-load switch.

Electrical characteristics emphasize modest on-resistance at low gate voltages, with datasheets specifying RDS at gate drives such as 2.5 V and 4.5–5 V. The device presents low gate charge, enabling fast transitions with minimal drive energy and reduced loading of the control pin. Body diode characteristics follow the usual N-MOSFET behavior, and the device’s small die size supports compact layouts with short trace lengths.

From a packaging and logistics perspective, BSS138 is widely second-sourced, inexpensive, and available in tape-and-reel for automated assembly. Variants exist under multiple vendor part numbers with closely matching electrical limits. The combination of low gate threshold, low gate charge, and small outline has made it a common choice in logic interfacing and light switching roles where board area and cost are constrained. Packages and construction span a wide range to suit current and thermal requirements. Small signal parts commonly appear in SOT-23 and similar outlines for currents in the hundreds of milliamperes, while power devices use packages

3.1.5.2 2N7002

2N7002 refers to a family of small-signal N-channel MOSFETs also offered in SOT-23 and similar packages. Like BSS138, these parts target low-to-moderate current switching with logic-level drive, and they typically carry drain-source voltage ratings around 60 V with continuous drain currents in the few hundred milliampere range. The device family is broadly manufactured, so sourcing options are extensive.

Electrical behavior differs primarily in gate threshold distribution and on-resistance curves at low gate voltages. Many 2N7002 variants specify RDS at VGS 4.5–5 V, with some providing data at 2.5 V for low-voltage logic. Gate charges are small, facilitating rapid switching in signal and low power converters, and parasitic capacitances are low enough to make these devices predictable in high-impedance control environments.

From a mechanical and assembly standpoint, footprints and pinouts align with common SOT-23 conventions, easing substitution among vendors. The broad ecosystem of equivalents simplifies second-source planning. Designers often view the 2N7002 class and BSS138 class as interchangeable small-signal switches, with selection guided by specific RDS and threshold curves in the intended gate-drive range. such as SO-8, DFN/PowerPAK, TO-220, and D2PAK with exposed pads or tabs to conduct heat to copper planes or heatsinks. Datasheets characterize thermal resistance from junction to ambient or case and provide safe operating area curves that bound permissible combinations of voltage, current, and time. Additional features may be specified, including avalanche energy capability, logic level gate drive compatibility, and in some product families integrated functions such as gate protection diodes or load-switch control, allowing selection of devices that match the intended voltage range, switching speed, and thermal design.

3.1.5.3 AO3400/A

AO3400-class parts represent higher-current, logic level N channel MOSFETs in compact SOT-23 packaging. Compared with small-signal switches, these devices offer substantially lower RDS at gate drives such as 2.5 to 4.5 V, enabling amperes of continuous current when thermal conditions are managed. Typical drain-source voltage ratings are in the 25–30 V range, appropriate for low-voltage power rails.

Key electrical parameters include very low on-resistance, moderate gate charge, and robust pulsed current capability. Transfer characteristics are optimized for true logic-level operation, and datasheets commonly provide curves down to 2.5 V gate drive. Package thermal resistance becomes the limiting factor; copper area and vias under the source tab are used to spread heat in compact layouts.

Availability is broad with multiple vendors offering pin-compatible devices in the same performance class. These MOSFETs are frequently chosen when a small outline is required but current and conduction loss targets exceed what small-signal parts can support. The blend of low RDS at low VGS and a ubiquitous SOT-23 footprint makes the AO3400 family a common baseline for compact low voltage switches.

3.1.5.4 BSS84/A

BSS84 class devices are small-signal P-channel MOSFETs supplied in SOT-23 packages, intended for low voltage applications requiring a high side device with negative gate drive relative to the source. Typical maximum drain-source voltage ratings are around 50 V, and continuous drain currents fall in the few hundred milliampere range, placing these parts in the signal and light load category.

Electrical characteristics include on-resistance specified at gate-source voltages such as 2.5 V and 4.5 V, with gate thresholds tailored for logic-level control in low-voltage systems. Gate charge is low, supporting fast transitions and modest drive requirements, while body-diode behavior follows the P-channel topology. As with their N-channel counterparts, parasitic capacitances are small, which aids predictable switching when driven from simple logic sources.

From a sourcing perspective the BSS84 family is widely available and second-sourced, making it straightforward to maintain alternatives. The small package and standard pinout ease placement on dense boards, and the parts are often paired with complementary N-channel SOT-23 devices to implement compact high side or load disconnect functions where a P-channel polarity is preferred.

3.1.5.5 TPS229xx

TPS229xx refers to a family of integrated load switch devices that combine a low RDS MOSFET with gate control, slew rate management, and protection features in a small outline. Rather than exposing a bare gate, these ICs present an enable pin and, in many variants, flags for fault reporting, reverse current blocking, or quick output discharge. Voltage ratings and on-resistance are targeted at low voltage rails, with options spanning a wide current range.

Electrically, the devices are characterized by controlled turn-on behavior to limit inrush, well defined quiescent currents, and specified thermal performance in small packages such as WCSP, X2SON, or SOT variants. Many members of the family are optimized for 1.1 to 5.5 V operation and publish RDS at low gate voltages since the internal driver is designed for logic control. Some options include good power outputs and undervoltage lockout thresholds that simplify system sequencing.

From a system-integration standpoint, these parts are used when a clean, self-contained load switch is preferred over a discrete MOSFET and gate network. The consolidation of control and protection reduces external component count and can improve repeatability across units. Product availability covers multiple vendors and footprints, enabling selection by rail voltage, current rating, and required features without changing the overall approach.

Table 3.1.6 MOSFET comparisons

	BSS138	2N7002	AO3400	BSS84	TPS229xx
Polarity	N-ch	N-ch	N-ch	P-ch	N-ch
Package	SOT-23	SOT-23	SOT-23	SOT-23	WCSP/X2SON/SOT
VDS rating	50-60 V	60 V	20-30 V	50 V	1.1-5.5 V
Logic level	X		X	X	X
RDS class			X		X
Continuous ID			X		X
Gate charge	X	X		X	X
High side suitability				X	X
Added functions					X
Thermal performance					X
Sourcing	X	X	X	X	X

3.1.6 Sensors

A sensor is a transducer that converts a physical stimulus into a measurable electrical quantity according to a defined transfer characteristic. Common classes include resistive elements that change resistance with strain, light, or temperature; capacitive and inductive structures that vary impedance with displacement or proximity; piezoelectric materials that generate charge under stress and inertial devices that report acceleration and angular rate.

Key attributes are sensitivity, range and linearity over the operating span, hysteresis and repeatability across loading cycles, bandwidth for time varying signals, and noise and temperature coefficients that bound resolution and stability. Many sensors incorporate conditioning features such as thin-film resistive networks, shielding, or integrated bridges to improve signal integrity. Flex and bend sensors, such as resistive polymer strips that increase resistance with curvature, fall within the strain sensing family and are specified by nominal resistance, resistance curvature curve, mechanical endurance, and allowable bend radius; their performance is typically characterized under defined test radii and temperature to establish scaling behavior and drift

3.1.6.1 sEMG Electrodes

The sEMG or surface electromyography measures the small bioelectric signals at the skin using microvolts that respond to muscle activation. The intended design considered would be to have electrode tech on all main muscle fibers that are needed to sense. The Pros of the technology was evaluated because it captures intent before actual motion occurs so it would remove the need for physical manipulation of a sensor. Also this type of electrode can control even when a finger motion is restricted.

Ultimately, I decided not to choose this device because of a few factors including noise, safety overhead, crosstalk complications, and cost effectiveness. The electronics require a high gain and low noise analog front ends, some motion suppression and will be delicate when paired with the PWM motor environment. Essentially violating IEC rule 61326-1 substantially with the EMC burden.

Also the user error and bodily functions of human beings like sweat and hair can dampen the effectiveness of the sensing capabilities. The day to day variability of having different users with all different hair and sweat introduces too much risk to justify accurate utility on a recurrent basis.

On the software level, consuming time to code and having independent finger control would require many channels and high level classifiers that ultimately prove to be unnecessary.

Bottom line is sEMG is great for intent decoding but adds a lot of hardware and regulatory challenges to complexity we do not need to concern ourselves with, considering our economic and time constraints.

3.1.6.2 IMU Based Finger Pose

Small IMU's mounted to measure piece by piece orientation, having about three per finger it is possible to triangulate joint angles and drive the actuators. While a model like this would yield high rate orientation data and enable rich motion capture, and good repeatability we did not choose it because of the cost, computation and fusion, environment sensitivity and mechanical integration.
The cost of this would imply 10-15 IMU's total per hand.This would strain the I2C bandwidth and power budget. The power budget would theoretically be beyond what an arduino nano can handle. When also considered that the gyro drift would accumulate and suffer interference not to mention permanently having to do a calibration with every use of the design, it is easy to not choose this method. The demand is simply too much for the controller and it complicates manufacturability.

3.1.6.3 Why the ZD10-100 Flex Strips

The flex sensor couples elegantly with the Arduino Nano so it's the best choice, featuring two tire resistive strips and it changes with applied normal force. It only reads positive absolute values so the hand will never be bending backwards in an unshapely manner. The form of the strip also sits kindly on every finger with little thought into it. Performance for control response time is less than 10 ms and the usable force window matches up with human finger bends even through fabrics such as the glove material layer. Because the strip is passive (no powered electrodes), our EMC problem is smaller and compliance with IEC 61326-1 practices (shielding, grounding, RC input filters) is straightforward. The glove materials and adhesives can be selected/verified against ISO 10993 endpoints (cytotoxicity/irritation/sensitization) without patient-connected electronics at the skin. Usability under IEC 62366-1 is also simpler: users just don't calibrate, move on skin prep, gels, or multi-IMU pairing.

In terms of cost and manufacturability, one sensor per finger keeps BOM and wiring modest, improves serviceability (easy replacement), and supports our IPC-anchored PCB approach for clean analog front ends. It fits our time and budget constraints while

delivering the control fidelity we need to validate the hand's mechanics, safety (max grip force / safe-open), and user workflow.

In short: the ZD10-100 path gives us fast, simple, and robust intent capture that integrates cleanly with the Nano, meets our standards-aware constraints (EMC, biocompatibility, usability), and leaves headroom to iterate the actuator safety and mechanical subsystems—the real heart of the prototype. When those are proven, we can revisit sEMG or IMU-rich variants as a second-generation upgrade, perhaps a flex goal.

Table 3.1.7 Sensors

	sEMG	IMU	ZD10-100
Per finger hardware	2-3	2-3	1
MCU interface	Analog	I2C/SPI	ADC
Latency	X		
Repeatability			X
Calibration			X
Sensitivity			X
EMC			X
Processing			X
Wiring			X
Power demand	X		X
Cost			X
Manufacturability			X
Regulatory			X
Robustness			X
Drift	X		X

3.1.7 Buck Boost Converters

For this project we have two components that will need power supplied to them. The servo motors on the hand and Arduino connected to them and the glove which just has an arduino. The glove is simple since the Arduino needs a simple power source and that will in turn supply the power needed for the transmitter. The power supply will need to also be small and easily transported since it will be attached to the user. The hand is more difficult. It will be powered by 18650 batteries but to ensure the servos get their sweet spot to ensure reliability a buck boost converter will be required. This will ensure that the exact voltage and current required is delivered to the hand.

3.1.7.1 LM5177

LM5177 belongs to TI's high performance, four-switch buck-boost controller family. It implements the classic noninverting synchronous buck-boost power stage with two inductors or a single coupled inductor and four external MOSFETs, allowing regulation when input voltage moves above, below, or around the output setpoint. The device provides the analog control core and drivers for the external FETs and the sensing and protection blocks needed to manage wide input variations and load transients.

Control features typically include current-mode or valley/peak-current control, programmable soft-start, slope compensation, compensation pins for loop shaping, and selectable operating modes that balance efficiency and ripple. Integrated gate drivers coordinate the high side and low side switches on both the buck and boost bridges to achieve synchronous operation, reduce diode loss, and support continuous conduction across the full operating range. Protective functions cover cycle-by-cycle current limit, thermal behavior, and fault handling.

The controller format separates power handling from regulation intelligence, which scales to higher power by choosing appropriate MOSFETs and magnetics. Package options are compact for placement near the power stage, and the pinout exposes sense lines, drivers, and control references for layout discipline typical of multi switch converters. Documentation emphasizes layout guidance for switch node containment, gate driver return paths, and current-sense routing.

3.1.7.2 LM5176

LM5176 is a closely related four switch buck boost controller intended for wide-input, non-inverting conversion using external MOSFETs. Its architecture supports seamless transition between buck, buck boost, and boost regions so the output remains regulated as input voltage crosses the target. The controller supervises both power bridges through dedicated gate drivers and timing logic, coordinating synchronous rectification to minimize conduction loss.

Control and supervision blocks include programmable soft-start, compensation networks, current-sense interfaces and mode settings for light-load behavior. Protection typically covers cycle by cycle current limiting, over-voltage/under-voltage handling, and thermal considerations. Frequency, ramp, and compensation pins allow tailoring of loop bandwidth for different magnetics and output capacitors.

As with other controller-class devices, output power capability is set by external component selection rather than the IC itself. Designers pick MOSFETs, inductors, and capacitors for the target voltage and power region, while the LM5176 supplies the control law and gate drive to orchestrate the four-switch topology. Package and pinout are arranged for short gate loops and clean current sensing in compact power sections.

3.1.7.3 LM51770

LM51770 represents a monolithic, synchronous buck-boost converter family that integrates the power switches on-chip rather than using external MOSFETs. The device provides non-inverting step-up/step-down regulation across a wide input span and is aimed at medium-power rails where integration simplifies the bill of materials and eases board space constraints. Internal power FETs and drivers reduce the number of high current nodes the layout must expose.

Control functions include soft-start, compensation options appropriate to the internal stage, selectable operating modes for light-load efficiency, and standard protection mechanisms. With the switching elements inside the package, timing, dead-time management, and synchronous rectification are handled internally to maintain efficiency across the buck, buck-boost, and boost regions.

Because switches are integrated, thermal performance and absolute current capability are defined by the IC’s package, thermal pad, and copper area. The part targets applications needing regulated output over a wide input span with fewer external components, trading the ultimate scalability of a controller for predictable, compact implementation.

3.1.7.4 LM251772

LM251772 denotes a member of TI’s wide VIN, non-inverting buck-boost lineage identified for multi-region regulation. It manages both the buck and boost bridges to hand off seamlessly as operating region changes, using synchronous rectification to control loss and maintain continuous inductor current. The device’s core function is to hold a set output voltage with an input that may drop below or rise above the target.

Feature sets in this class cover soft-start ramps, loop compensation pins, current sense inputs, and selectable light-load modes. Protection includes cycle by cycle current limit and fault responses that place the power stage in a safe state until conditions recover. The control method is designed to keep transient response consistent across mode boundaries, which is a key requirement for non-inverting buck-boost rails.

This device class is commonly packaged for placement adjacent to the magnetics and power path, exposing driver pins, sense pins, and control references in a way that promotes tight gate drive loops and clean current measurement. External component choices determine thermal and current limits, while the controller maintains timing, phasing, and protection across the four-switch stage.

3.1.7.5 LM51772

LM51772 is a monolithic, synchronous buck-boost converter that integrates the switching MOSFETs and gate drivers to provide a compact, non-inverting step-up/step-down rail. It regulates a fixed output across wide input variation and is intended for applications where board area and component count are constrained. Integration reduces the number of external high nodes and simplifies EMI containment compared with discrete controllers.

Typical control features include programmable soft-start, compensation tailored to the internal power stage, selectable switching modes for light load operation, and protective functions such as current limiting and over voltage response. Seamless region transitions keep output behavior consistent as input crosses the regulated level, with synchronous rectification used throughout to improve efficiency.

Power capability is bounded by the internal FET ratings and thermal design of the package, so achievable output current depends on ambient conditions, airflow, and copper spreading. The device targets medium-power rails that benefit from a single chip solution while still covering both buck and boost operation.

Table 3.1.8 Buck Boost Converters

Regulator	Type	Output Current	Control Method	Efficiency	Complexity
LM5176	Controller (external MOSFETs)	Very high	Peak current mode	High	High
LM5177	Controller (external MOSFETs)	High to very high	Valley current mode	High	High

LM51770	Monolithic buck-boost	Moderate	Peak current mode	Medium to high	Medium
LM51772	Monolithic buck-boost	Moderate	Sync switching	High	Medium
LM251772	Controller or monolithic	High	Peak current mode	High	High

3.1.8 Batteries/PSU

A rechargeable battery is an electrochemical storage device that converts chemical energy to electrical energy through redox reactions at two electrodes separated by an electrolyte. Cell chemistry sets the nominal voltage, energy density, internal resistance, and safety profile; common chemistries in portable electronics include lithium-ion families (lithium-cobalt oxide, nickel-manganese-cobalt, lithium iron phosphate), nickel-metal hydride, and, less commonly now, sealed lead-acid for higher power at low cost. Cells are manufactured in cylindrical, prismatic, and pouch formats, each balancing mechanical robustness, thermal behavior, packaging efficiency, and manufacturability.

Key performance attributes include specific energy and specific power, continuous and burst discharge capability, cycle life, and calendar aging. Internal resistance and temperature strongly influence voltage sag and heat generation under load, while protective elements govern usable operating windows. Lithium-ion cells are typically charged by a constant current or constant voltage profile to a defined upper cutoff and discharged to a lower cutoff to maintain longevity. Nickel chemistries use different termination methods and tolerate overcharge and deep discharge differently. Datasheets specify limits for charge and discharge currents, temperature ranges, and storage conditions that determine practical service life.

Most packs incorporate electronics to manage safety and interface expectations. A battery management system or simpler protection circuit monitors cell voltage and current, enforces over-current, over/under-voltage, and over-temperature cutoffs, and may provide balancing in multi cell series stacks. Consumer USB power banks add a boost converter to present regulated 5 V along with state of charge indication and input charging control. Cylindrical formats such as 18650 are widely second sourced with consistent mechanical standards, whereas pouch and prismatic formats offer higher packaging efficiency at the expense of mechanical rigidity and may rely more on the enclosure for support and heat spreading.

3.1.8.1 18650 Li-ion

An 18650 lithium-ion cell is a standardized cylindrical format approximately 18 mm in diameter and 65 mm in length, widely used in portable electronics and electric vehicles. Typical chemistries include nickel-manganese-cobalt (NMC) and nickel-cobalt-aluminum (NCA), yielding nominal voltages around 3.6 to 3.7 V per cell with energy densities that are high for their size. Mechanical robustness is a strength of the metal can construction, which helps with handling, spot-welding of tabs, and thermal conduction to the enclosure.

Electrical characteristics are defined by capacity classes ranging from roughly 2,000 to 3,500 mAh and continuous discharge ratings that vary from low drain cells intended for consumer devices to high drain cells designed for power tools and traction packs. Internal resistance and thermal behavior depend on the specific model. Datasheets specify maximum continuous and pulse currents, recommended temperature windows, and charge profiles using constant current or constant voltage to a fixed upper cutoff. Cycle life and calendar aging are influenced by depth of discharge, charge voltage setpoint, and temperature.

Cells are available from multiple manufacturers with consistent dimensions and a broad ecosystem of holders, protection circuits, and pack-building accessories. Variants include “protected” versions that integrate a small protection PCB under the sleeve, and bare cells intended for assembly into packs with external battery management systems. Traceability and authentic sourcing are important due to market prevalence and performance variability among brands.

3.1.8.2 Lithium-Polymer Pack

Lithium-polymer pouch cells use laminated foil enclosures rather than rigid cans, allowing thin, lightweight shapes and custom footprints that can be stacked or arranged to fit irregular volumes. Nominal cell voltage is again around 3.6 to 3.7 V, with capacities and C-ratings tailored to applications from small wearables to high-power model aircraft packs. The flexible packaging enables high packing efficiency but places more responsibility on the surrounding mechanical design for support and protection.

Electrical behavior spans low to very high discharge capability depending on electrode design. Datasheets specify continuous and burst C-rates, internal resistance, and temperature ranges for charge and discharge. Charging uses the same constant current and constant voltage method as other lithium-ion chemistries, with attention to cell balancing when multiple cells are configured in series. Thermal characteristics benefit

from large surface area but require uniform compression and avoidance of sharp edges or point loads to maintain reliability.

Commercial offerings range from single cells with flying leads and protection boards to multi-cell packs with balanced connectors and embedded protection. The absence of a rigid shell makes weight low and form factor options broad, while also making the cell more sensitive to swelling if overstressed or aged. Handling practices emphasize mechanical support, controlled clamping pressure, and enclosure features that mitigate puncture risk.

3.1.8.3 LiFePO4 Cells

LiFePO4 cells use an olivine structured cathode that provides a nominal voltage near 3.2 to 3.3 V per cell, lower than cobalt-rich chemistries but with a flat discharge curve and strong thermal stability. Energy density is typically lower on a volumetric basis compared with NMC/NCA, yet specific power and safety margins under abuse conditions are recognized advantages. Cycle life can be long, with datasheets often citing thousands of cycles under moderate depth of discharge.

Electrical specifications include continuous and pulse discharge ratings suitable for many portable power roles, with internal resistance generally higher than high energy NMC cells but stable across a wide temperature range. Charging follows a constant current/constant voltage profile to a lower terminal voltage than other lithium-ion chemistries, and the chemistry’s tolerance to high cycle counts favors applications where longevity outweighs absolute capacity per volume.

Cylindrical LiFePO4 formats mirror the accessory ecosystem of other cells, making holders, tabs, and pack assembly familiar. The chemistry’s lower nominal voltage affects series/parallel configurations for given rail targets, and the flat discharge plateau simplifies state of charge estimation in some battery management approaches. Availability spans primary manufacturers and many secondary sources with varying discharge capabilities and capacity classes.

3.1.8.4 USB Power Bank

A consumer USB power bank combines one or more lithium-ion cells with a protection and control PCB and a DC-DC boost stage to present regulated output ports. The most common interface supplies 5 V over USB-A or USB-C, with newer models implementing power-delivery negotiation for additional fixed voltage steps. Enclosures typically include charge-state indicators, on/off control, and protection against over-current, over-voltage, under-voltage, and over-temperature conditions.

Internally, packs often use one to three cells in parallel at a single series count, relying on a boost converter to produce the regulated output. The battery management circuitry supervises charging via a dedicated charge controller, balances parallel cells implicitly through common connection, and enforces cutoff thresholds to protect cell health. Efficiency and maximum continuous output current depend on the boost stage design and thermal constraints of the enclosure.

Product families span compact single-cell units optimized for portability to larger multi cell enclosures designed for higher capacity and extended runtime. Certification markings, port counts, advertised watt-hours, and supported fast-charge protocols differentiate models. Because the output is regulated and standardized at the port, integration involves treating the power bank as a self-contained source rather than a raw cell, with the battery management and charging logic embedded inside the unit.

3.1.8.5 NiMH

NiMH cells in AA format provide a nominal voltage of approximately 1.2 V per cell and are arranged in series to meet target voltages. The chemistry is known for mechanical robustness, tolerance to repeated charge/discharge, and moderate self-discharge characteristics that have improved in “low self-discharge” formulations. Cylindrical cans and standardized dimensions make sourcing and replacement straightforward.

Electrical characteristics include moderate internal resistance and C-rates that are typically lower than high drain lithium-ion cells but adequate for many consumer and industrial products. Charging methods differ from lithium systems, employing techniques such as negative deltaV detection, temperature rise monitoring, or timed charge to terminate correctly and avoid overcharge. Performance at lower temperatures is generally acceptable, with capacity and voltage response dependent on discharge rate.

Packs can be assembled from off the shelf cells with holders or welded tabs, and they do not require the same level of electronic protection as lithium chemistries, though fuses and thermal sensors are common in higher power applications. Energy density is lower relative to lithium-ion cells, resulting in larger and heavier packs for equivalent watt hours. The chemistry’s handling simplicity and wide availability make it a stable option where standardized cell formats and straightforward charging infrastructure are valued.

Table 3.1.9 Batteries

	18650	Li-ion	LiFePO4	USB power bank	NiMH
Voltage per cell	3.6-3.7 V	3.6-3.7 V	3.2-3.3 V	5 V	1.2 V

Energy density	X	X			
Specific power	X	X			
Cycle life			X		
Thermal stability			X		
Form factor		X			
Protection	X	X	X	X	
Output regulation				X	
Capacity range	2-3.5 Ah	Wide range	1-1.6 Ah	Wide range	2-3 Ah
Availability	X	X	X	X	X
Packaging	X		X	X	X

3.1.9 Software

When selecting the software approach for GRIP, the group evaluated several programming models commonly used in embedded systems. Since the Arduino Nano is the chosen controller for both the glove and the hand, the comparison focuses on software structures and control techniques that work well within Arduino's limitations. The main approaches considered were the traditional polling loop, interrupt-driven processing, cooperative scheduling, and simple state-machine based control. Each technique offers benefits for real-time systems, but they vary in timing accuracy, complexity, and suitability for low-power microcontrollers like the Nano.

The simplest option is the polling loop, which relies on repeatedly checking the system's sensors and running control logic inside the main loop. This model is easy to implement and fits naturally with Arduino's programming style. For flex sensor reading, wireless packet formation, and servo updates, polling can work well as long as the loop remains short and predictable. The downside is timing inconsistency. If a single operation takes longer than expected, the entire loop slows down, causing jitter in the servo updates or delayed wireless transmissions. Polling also struggles when different tasks require different update rates.

Interrupt-driven processing is the next approach considered. Interrupts can trigger on ADC completion, timer events, or external signals. This allows parts of the software to react immediately instead of waiting for the polling loop to reach them. For GRIP, interrupts could be used for consistent sampling of the flex sensors or for maintaining precise PWM timing. While this method provides better timing control, it introduces complexity because shared variables between interrupts and the main loop must be carefully managed. If interrupts fire too often or overlap, they can delay other important tasks and make behavior harder to debug.

A cooperative scheduling model lies between simple polling and full real-time operating systems. In this approach the code is separated into small tasks that each run quickly, and the loop executes these tasks in a fixed order. Each task performs a small part of its work and returns control immediately, allowing the entire system to feel more responsive. This works well on the Arduino Nano because it avoids preemption but still gives structure to the firmware. For GRIP, tasks such as reading sensors, preparing radio packets, checking connection status, and updating servos can be isolated into separate functions. As long as each task remains lightweight, this method keeps timing stable without adding much complexity.

State machines were also considered for organizing the software. A state machine divides system behavior into defined states such as calibrating, transmitting, receiving, or error recovery. Each state has its own logic and transitions occur when certain conditions are met. The benefit of this approach is clarity. It becomes easy to see what the system should be doing at any moment. This is especially useful for GRIP's glove calibration, packet validation, and fail-safe behavior. The drawback is that state machines only structure the logic; they do not fix timing on their own. They work best when used on top of either polling or cooperative scheduling.

After comparing these methods, the group concluded that a combined approach works best for the Arduino Nano. Polling is used for general processing because it simplifies the design and keeps the firmware easy to debug. Cooperative task structuring is layered on top to ensure consistent behavior even when tasks grow in complexity. Limited use of interrupts is included where predictable timing matters, such as maintaining consistent sampling or watchdog functions. This hybrid model provides reliable performance while staying within the computational limits of the Nano.

Table 3.1.9 Software

Approach	Advantages	Disadvantages	Suitability
Polling Loop	Simple	Timing jitter	General logic
Interrupt-Driven	Fast and accurate	Debugging	Time critical events

Cooperative Scheduling	Balanced and organized	Depends on loop speed	Organization
State Machines	Clear logic flow	No timing control	Good in combination with other methods

3.1.10 Servo Control Strategies

Controlling servos efficiently is a critical part of GRIP because the hand must respond smoothly to the user's movements without jitter, lag, or sudden jumps. Since the MG996R servos are controlled using standard PWM pulse widths, several strategies for generating these pulses were evaluated. Each strategy differs in timing accuracy, CPU usage, ease of implementation, and compatibility with the Arduino Nano. The methods considered include the standard Arduino Servo library, direct timer control, software-based PWM generation, and hardware PWM using dedicated timers.

The Arduino Servo library is the most straightforward option. It allows each servo to be controlled using simple function calls, making it ideal for rapid development and testing. The library internally uses interrupts tied to the Nano's timers to maintain the correct 20 ms refresh period and appropriate high pulse durations. The main advantage of this method is ease of use. The drawback is that the library can consume significant timer resources and may interfere with other timing-sensitive parts of the system such as radio communication or sensor sampling if not used carefully.

Direct timer control is a more advanced technique where the microcontroller's hardware timers are configured manually to output precise pulses. This method is more efficient than using the Servo library because it avoids some overhead and gives the programmer full control over the timing registers. It is useful when strict pulse timing is required or when multiple servos must be synchronized closely. The downside is complexity. Timer registers on the ATmega328P (the Nano's MCU) require careful configuration, and mistakes can cause timing drift or conflicts with other modules that also depend on timers.

Software PWM is another possible approach where the code manually toggles pins at precise intervals. Although this gives flexibility, it depends heavily on the main loop timing, which makes it prone to jitter. Because servo motion must be smooth and predictable, software PWM is not ideal unless the system is extremely simple or runs on a much faster processor. On the Arduino Nano, software PWM may also be disrupted by interrupts from the radio or ADC readings, causing unexpected servo jumps.

Hardware PWM using dedicated timer channels is the most stable option, but the ATmega328P microcontroller provides only a limited number of PWM pins, and servo signals require precise pulse widths rather than duty-cycle-based PWM. This makes the built-in PWM hardware insufficient by itself. While timers can be configured into pulse-generation mode, this still requires a deeper understanding of the timer system and may reduce the number of available timers for other functions like the radio SPI timing or watchdog features.

After evaluating these choices, the most appropriate strategy for GRIP during SD1 is to use the Arduino Servo library because it allows multiple servos to be controlled with minimal development time while maintaining acceptable stability. For SD2, direct timer-based control may be explored if higher responsiveness or more precise timing is needed, especially once the full arm is integrated.

3.1.11 Wireless Communication Latency and Timing

The wireless link between the glove and the hand is a critical part of GRIP because the overall responsiveness of the system depends on how quickly sensor data can be transmitted and interpreted. Even if the flex sensors and servo motors react quickly, the performance of the system would still feel delayed if the radio link introduces too much latency. To understand how each part of the transmission contributes to overall timing, several characteristics of the nRF24L01 module and its interaction with the Arduino Nano were evaluated. The main factors considered include packet formation time, SPI transfer time, on-air transmission time, acknowledgement behavior, and potential interference within the 2.4 GHz band.

Packet formation time refers to the time the microcontroller takes to gather ADC readings, apply smoothing, and load data into the radio’s transmit buffer. For the Arduino Nano, this process is very fast since the data being sent is small. The nRF24L01 uses fixed payload sizes, so the time to write the packet over SPI is predictable. The SPI interface on the Nano can run efficiently enough that writing a typical 8 to 16 byte packet only takes a small fraction of a millisecond. Because of this, packet preparation does not add significant delay as long as the main loop remains lightweight.

On-air transmission is determined by the radio’s data rate setting. The nRF24L01 supports several air rates, with 2 Mbps being the fastest option. At this speed, sending a small control packet takes well under a millisecond. Even at 1 Mbps, transmission time remains very low. The drawback of using higher data rates is reduced range and slightly lower resilience to noise. For GRIP, where the glove and hand operate within only a few meters of each other, the high-rate modes are suitable and provide the lowest latency.

Acknowledgement timing also affects performance. The nRF24L01 automatically handles acknowledgements and retransmissions. If the packet is received successfully, the transmitter completes the cycle quickly. If not, it retries based on the configuration. While the auto-ack system improves reliability, too many retries can add measurable delay. To prevent this, a clean channel selection and short payload size are used. The short packets reduce collision likelihood and ensure that retries are rare in normal operation.

Interference and congestion are potential sources of latency in the 2.4 GHz band. Wi-Fi networks, Bluetooth devices, and other consumer electronics can overlap with the nRF24L01. If the radio detects interference, the effective transmission time increases due to retries or corrupted packets. Using channels that avoid common Wi-Fi frequencies helps minimize this issue. The low-level power output and short range of the GRIP system also reduce the likelihood of external interference.

Overall, when the radio is configured correctly and the packet sizes are kept small, the total latency of the wireless link is only a few milliseconds. Combined with the glove's fast sensor updates and the servo refresh cycle, the total system latency remains below the threshold where a user would notice a significant delay. This ensures that the robotic hand feels closely synchronized with the operator's movements.

3.1.12 Flex Sensor Behavior and Modeling

The ZD10-100 flex sensors used in the GRIP glove change resistance in response to bending, which allows the Arduino Nano to measure finger movement through its analog-to-digital converter. To understand how these sensors behave in real operation, their electrical characteristics must be evaluated. The flex sensors function as variable resistors whose resistance increases as the bend angle increases. When combined with a simple voltage divider, the Arduino can read the output voltage and interpret it as a measure of finger position. Although this seems straightforward, the actual behavior of the sensors contains several nonlinearities and sources of noise that must be accounted for during the design process.

The first important characteristic is the resistance curve. Flex sensors do not behave linearly from 0° to their maximum bend angle. Instead, the resistance tends to increase slowly at small bends and more sharply at larger bends. This produces a curved response, which requires some mapping to make the servo motion feel proportional to the user's fingers. Without any calibration, the middle region of the sensor's range provides the most resolution, while the extremes can appear compressed. Polynomial mapping or scaled linearization can improve this, but some projects simply accept the natural curve because the overall bending behavior remains usable.

Another electrical factor is the voltage divider configuration. The series resistor chosen affects both the ADC reading range and the responsiveness of the output. If the resistor value is too high, the output voltage becomes sensitive to noise. If it is too low, the sensor draws unnecessary current and reduces the resolution available to the ADC. A typical design uses a resistor that places the resting finger position in the mid-range of the ADC. This provides the largest usable span for reading finger movement and minimizes clipping at either extreme.

Noise is another concern. Flex sensors are sensitive to small mechanical vibrations and electrical interference, especially on long wire runs. When the glove is worn, small tremors from the user's hand or small impacts can create fluctuations in the ADC readings. Electrical noise can also enter the signal when the nRF24L01 transmits or when MG996R servos draw bursts of current, although this is more noticeable on the hand subsystem than the glove. To counter these fluctuations, filtering techniques such as exponential smoothing are applied. These help stabilize the values without significantly slowing the response time.

Temperature and material fatigue also affect flex sensor performance. As the conductive ink inside the flex strip is repeatedly bent, its resistance response can slowly drift. Cold temperatures stiffen the substrate, increasing resistance, while heat reduces it. Although the drift is minor for short-term projects, long-term operation may require recalibration. Because of this, a simple calibration step is included at startup to capture each user's neutral and fully bent positions.

Finally, quantization from the Arduino's 10-bit ADC introduces small steps in the measured value. With 1024 possible readings, each step corresponds to a small change in the input voltage. While this resolution is generally sufficient for finger tracking, combining it with smoothing helps produce servo motion that feels fluid rather than segmented.

Overall, the electrical behavior of the ZD10-100 flex sensors is well-suited for the GRIP system. Their predictable response curve, simple interfacing, and reasonable stability make them reliable for reading finger positions. With proper filtering and calibration, they provide consistent input that the robotic hand can interpret accurately.

3.1.13 Motor Analysis

The MG996R servo is one of the most important components on the GRIP hand because it directly converts the glove's finger movements into mechanical motion. Understanding its mechanical and thermal behavior is essential for designing a system that is both responsive and reliable. Although the MG996R is a common hobby servo, its internal

characteristics such as torque output, stall current, heat buildup, and mechanical backlash all affect how well the robotic hand performs during repeated use.

One of the primary mechanical considerations is torque. The MG996R provides high stall torque in its voltage range, making it a strong choice for tendon-driven finger motion. The torque available determines how tightly the fingers can grip objects and how quickly they can move against resistance. In a tendon system, the servo's rotational torque is converted into linear pulling force on the tendon spool. The radius of the spool determines the force-to-torque ratio. A smaller spool radius increases pulling force but reduces speed, while a larger spool does the opposite. Because finger movements require both reasonable speed and sufficient force, the servo’s mechanical output must be balanced by choosing an appropriate spool diameter. This trade-off directly affects the natural feel of the robotic hand.

Mechanical backlash is another factor. The metal gears inside the MG996R provide durability, but slight looseness between teeth is normal. This slack creates a “dead zone” where the servo horn can move slightly without generating real motion at the tendon. When mapped to finger movement, this can make fine adjustments near the neutral position feel less precise. While backlash cannot be eliminated in this class of servo, it can be minimized through careful tensioning of the tendons and by ensuring that the servo is installed securely so that additional mechanical play is not introduced.

Thermal behavior is equally important. Servos generate heat when they draw current, especially under heavy loads or when holding a fixed position for long periods. The MG996R can draw several amps at stall, and continuous near-stall operation will raise internal temperatures significantly. Heat buildup affects both performance and lifetime. Excessive heat can soften plastic components or cause the motor winding to degrade. In a compact enclosure like the GRIP hand, heat cannot dissipate easily. This means that grip strength and hold duration must be managed carefully. Short bursts of force are acceptable, but continuous clamping should be avoided unless additional cooling or current limiting is added in later revisions.

Electrical loads also play a role in mechanical performance. Voltage sag on the servo rail can reduce torque and responsiveness, which is more likely to occur when several servos move at the same time. Because MG996R motors draw sharp current spikes when accelerating, the power system must include enough bulk capacitance and a stable regulator to prevent voltage dips. If the voltage drops too much, the servo may jitter or reset. This is especially noticeable when transitioning from low load to sudden high load, such as gripping an object or closing the hand rapidly.

Finally, duty cycle limitations determine how long the servo can be operated before requiring a rest period. The internal motor and driver circuitry are not designed for

constant full-output operation. If the robotic hand is used for tasks involving long grips or continuous finger motion, the servos will eventually warm up. While this is acceptable for short demonstrations and light workloads, extended or heavy use may require improved airflow, heat spreading through the enclosure, or selection of higher grade servos in SD2.

Overall, the MG996R provides a good balance of torque, cost, and size for the GRIP system. With proper mechanical design and awareness of thermal limits, it can reliably produce the finger movements needed for the project while fitting within the power and budget constraints of SD1.

3.1.14 Battery Load

The power system for GRIP must support two very different electrical environments. The glove requires low-power operation for the Arduino Nano, flex sensors, and the nRF24L01 module, while the robotic hand requires significantly more power to drive the MG996R servos. Because of this difference, the battery load and runtime behavior must be analyzed separately. Understanding the current draw of each subsystem helps determine how long the system can operate under realistic conditions and ensures the power supply does not become a performance bottleneck.

A single MG996R servo can draw between 200 to 300 mA during light movement, but the current increases drastically when the servo accelerates or when it is holding a load. Under stall conditions, the servo can draw over 2 A. Since the GRIP hand uses multiple servos operating at the same time, the total load on the power system can be significant. For example, if two or three fingers move simultaneously, their combined current spikes can exceed 4 A briefly. Even short bursts can cause voltage sag if the battery or regulator cannot handle these peaks. These fluctuations directly affect servo torque and can cause jitter or unexpected resets, so the battery must be able to deliver both high peak current and steady continuous current.

For the glove, the current requirements are much smaller. The Arduino Nano typically draws around 20 to 30 mA during normal operation, and the nRF24L01 uses around 15 mA during transmission. The flex sensors contribute only a few milliamps each, depending on the voltage divider configuration. This keeps the glove power budget low enough that a small power bank can run it for several hours without issue. Because the glove transmits small packets and only performs lightweight computations, its runtime is limited more by the capacity of the power bank than by any large electrical load.

The performance of the hand battery depends strongly on the capacity and type of the cells used. A typical 18650 lithium-ion cell ranges from 2000 to 3000 mAh. However, the key factor is not just the capacity but the discharge rating. Some cells can safely deliver

higher currents without overheating or triggering protection circuits. High-drain cells allow the servos to operate without causing sudden voltage drops. When multiple cells are used in parallel, both the total capacity and the maximum continuous current increase, which helps stabilize the servo rail during heavy motion.

Runtime varies depending on activity level. Light movements where only one or two servos move at a time can lead to longer operation periods, while full-hand gestures or gripping objects reduce running time because they place higher demands on the motors. To estimate runtime, both average and peak current must be considered. For instance, if the average current drawn by the hand is around 1.5 A and the battery has an effective capacity of 2000 mAh, the runtime would be roughly one to one and a half hours. This estimate decreases when frequent heavy loads or constant gripping are involved because the servos draw more current under those conditions.

Voltage sag is another important factor in runtime modeling. As the battery discharges, its voltage naturally decreases. If the regulator is not a buck-boost type, the servo voltage may begin to drop below optimal levels, reducing torque and performance. This is especially noticeable when the battery reaches the lower half of its charge. Using a buck-boost converter helps maintain a stable servo voltage across most of the discharge curve, improving consistency even as the battery drains.

Overall, the battery and regulator selection must account for both the average load and the high current spikes produced by the servos. A properly sized battery with sufficient discharge capability, paired with a stable power regulator, ensures that the GRIP system operates smoothly and maintains predictable performance throughout its runtime.

In the final design, the GRIP hand is powered using a 2S2P lithium-ion pack, which combines the advantages of both series and parallel configurations. The two series-connected cells increase the pack voltage to a nominal 7.4 V, with a full-charge voltage of 8.4 V and a minimum safe voltage of around 6.0 V. Two of these series pairs are then connected in parallel, doubling the capacity to approximately 5000 mAh while also doubling the available discharge current. This configuration provides a high-capacity, high-current source that is well-suited for the MG996R servo motors used in the hand.

The increased voltage improves regulator performance because the servo rail always has headroom for a buck or buck-boost converter to maintain a stable 6 V output. The doubled discharge capability reduces voltage sag during sudden current spikes, especially when multiple servos move at the same time. Each servo spike is shared across both parallel branches, which significantly improves stability compared to a single-cell design. The large effective capacity also extends runtime during demonstrations and testing, allowing the hand to operate for one to three hours depending on the activity level.

Overall, the 2S2P configuration offers excellent performance, reliability, and power stability for the GRIP system.

Table 3.1.10 System Power Requirement

Component	Current	Peak Current	Notes
MG996R Servo	200–300 mA	2+ A	Multiple servos increase total load
Arduino Nano	25 mA	Low	Stable draw during operation
nRF24L01	15 mA	Bursts	Small contribution to total load
Flex Sensors	1–5 mA each	Low	
18650 Cell		High	Must handle servo spikes

3.1.15 Voltage

A stable power distribution system is essential for the GRIP hand because the servo motors, wireless receiver, and microcontroller behave differently depending on the quality of the voltage delivered to them. Even though the 2S2P battery pack provides a strong foundation with high current capability, the voltage delivered to each subsystem must remain steady under varying load conditions. Servos, especially the MG996R, can draw rapid current spikes when they accelerate or experience resistance, creating disturbances on the power rail. These disturbances can lead to jitter, reduced torque, or even system resets if the voltage is not properly regulated. Because of this, careful planning of the regulator, wiring, and capacitance placement is required for reliable performance.

The first consideration is how the battery voltage is stepped down to the servo operating voltage. Since the 2S2P pack ranges from 6.0 V to 8.4 V during its discharge, the servo rail must use either a buck converter or a buck-boost converter. A buck converter works effectively as long as the input voltage stays above the target output voltage. However, because the lower limit of the pack is very close to the servo's required voltage, a buck-boost converter helps maintain full torque as the battery drains. Regardless of the regulator type chosen, it must be able to handle high burst currents without introducing large ripple or momentary voltage collapse. This is why servo power regulation is treated separately from the glove circuitry.

Wiring losses also influence voltage stability. Even short wires can drop noticeable voltage when carrying several amps. To reduce resistive losses, the main power lines to the servos should use thicker traces or wires, and the path between the battery, regulator, and servo distribution point should be as short as possible. Thicker copper pours on the PCB can help with this by spreading current over a larger area. When heavy loads are expected, a combination of wide traces and additional copper pour can prevent hotspots and reduce unwanted heating.

Capacitors play another major role in stabilizing the voltage rail. Bulk electrolytic capacitors placed near the servo connector block help absorb sudden current spikes. These capacitors act like temporary energy reservoirs, releasing current quickly when servos demand it faster than the regulator can respond. Smaller ceramic capacitors can be added in parallel to handle higher-frequency noise generated by switching regulators. Both types work together to keep the voltage steady during motion. The placement of these capacitors is just as important as their values. They must be located physically close to the servos to be effective, while additional filtering capacitors near the regulator help smooth out its switching ripple.

Ground layout is also a significant factor in maintaining stable operation. If the servos share the same ground path as the radio or microcontroller without proper routing, current spikes can introduce noise that causes communication errors or unstable sensor readings. A star-ground approach or a dedicated high-current ground plane is recommended so that servo currents do not interfere with sensitive logic circuits. Keeping high-current loops separate from signal paths improves both stability and RF performance.

Finally, voltage stability must be maintained during the entire movement cycle of the hand. Rapid transitions, such as going from a fully open to a fully closed hand, require the power system to supply several amps at once. If the regulator or wiring cannot handle this, the servo rail might momentarily dip, causing slower response or uneven finger motion. By designing the distribution system with strong regulators, adequate wiring, and sufficient capacitance, the GRIP hand remains responsive and stable even during demanding movements.

3.2 Technology Selection

3.2.1 MCU

Arduino Nano was selected because it provides the shortest path to a stable control platform with exactly the interfaces this project needs in a compact footprint. Its integrated ADC, I2C/SPI, timers/PWM, and USB serial make sensor sampling, packet framing, and actuator command generation straightforward without additional glue

hardware. The mature Arduino ecosystem, abundant example code, and wide community support reduce schedule risk and keep bring-up focused on the application rather than driver development. Instant boot and low power suit a wearable/glove context, and the small board size aligns well with tight enclosures.

The trade-offs are limited RAM/Flash, a single hardware UART, and 5 V logic in a 3.3 V centric world are acceptable for this scope because the control loop and packet sizes are modest and the design tolerates lightweight protocols. Level compatibility and memory constraints are well understood and manageable within the intended sampling rates and feature set, so Nano's simplicity and availability outweigh the benefits of moving to a more complex 32-bit platform for this iteration.

After evaluating several microcontroller platforms in the technology comparison section, the Arduino Nano was selected as the primary controller for the GRIP glove due to its balance of simplicity, reliability, size, and compatibility with the project requirements. Although more advanced options exist, the Nano provides more than enough performance for reading the flex sensors, processing data, and transmitting packets to the nRF24L01 wireless module. One of the main advantages of the Nano is its straightforward development environment. The Arduino IDE makes writing, uploading, and debugging code extremely efficient, which is especially helpful during rapid prototyping. Because the GRIP project involves continuous iteration and testing, having a platform that is easy to program significantly speeds up the development cycle.

The Nano's hardware architecture also aligns well with the project needs. It provides multiple analog inputs for the flex sensors, allowing each finger to be read independently with sufficient resolution. The 10-bit ADC on the ATmega328P is accurate enough for interpreting finger motion and works smoothly with the smoothing and filtering techniques discussed earlier. The Nano's digital pins support the SPI communication required by the nRF24L01, and the SPI interface operates fast enough to handle frequent wireless transmissions without causing delays. Since the project only requires modest computational power, the 16 MHz clock speed is more than sufficient for reading sensors, applying filtering, forming packets, and maintaining a consistent update loop.

Power compatibility is another reason the Nano fits well into the system. It can be powered directly from a USB power bank, which is already planned for the glove, and its onboard regulator provides a stable 5 V rail for the sensors. This eliminates the need for a dedicated regulator on the glove side, simplifying the power design. The Nano's small size also makes it easy to mount inside the glove or on a small prototyping board without adding unnecessary bulk. Its minimal footprint helps keep the glove lightweight and comfortable.

The software ecosystem surrounding the Arduino platform is another major benefit. There are numerous libraries available for sensor reading, wireless communication, and timing control. The nRF24L01 library, in particular, works reliably with the Nano and supports all the features required for consistent data transmission. Because the project relies on fast turnarounds between testing and refinement, having access to well-supported libraries reduces development time and minimizes debugging complexity. The Nano also supports simple serial monitoring, which is extremely helpful when calibrating the flex sensors or diagnosing wireless issues.

Finally, cost and availability played a practical role in the selection. The Arduino Nano is low-cost and widely available, which makes it easy to replace or duplicate if needed during testing. Since senior design projects often involve unforeseen issues, using an inexpensive and easily sourced microcontroller reduces risk. Overall, the Arduino Nano meets all functional, performance, and practical requirements of the GRIP glove, making it the most appropriate choice for the project.

3.2.2 Transceiver

nRF24L01 was chosen for the glove to hand link because it is purpose built for short, deterministic packets at high update rates with hardware assist for acknowledgments, retries, and CRC. The SPI interface and small modules keep the electrical and mechanical footprint minimal, and selectable data rates allow a practical balance between robustness and latency. Broad, inexpensive module availability and stable libraries help ensure predictable performance across iterations and spares.

Expected limitations are operation in the busy 2.4 GHz band, 3.3 V only power IO, and minimal built in security are reasonable in the context of short-range control traffic and a contained operating environment. The device’s feature set aligns directly with the requirement for steady, low latency pose updates without the complexity of a full Wi-Fi or BLE stack, making it an appropriate choice for this phase.

After reviewing different wireless communication options such as Bluetooth modules, Wi-Fi chipsets, and other 2.4 GHz radios, the nRF24L01 was selected as the preferred wireless transceiver for the GRIP system. The decision was based on its low latency, predictable packet structure, excellent power efficiency, and strong compatibility with the Arduino Nano. Since the glove must transmit continuous sensor updates to the robotic hand in real time, the wireless link must be both stable and fast. The nRF24L01 meets these requirements by offering a lightweight communication protocol with minimal overhead. Unlike Wi-Fi modules that require association, authentication, and connection management, or Bluetooth modules that impose additional software stacks, the nRF24L01 focuses solely on sending raw packets quickly. This makes it ideal for systems that need consistent timing rather than high-volume data transfer.

One of the strongest advantages of the nRF24L01 is its extremely low latency. It can deliver packets in just a few milliseconds, which keeps the hand movements synchronized with the glove. When the user bends a finger, that motion needs to be reflected almost immediately in the servo response. With the nRF24L01's short packet times and hardware-level acknowledgments, the glove can maintain a steady update loop without waiting on long connection processes or unpredictable network delays. This is a significant improvement over modules like the ESP8266 or HC-05 Bluetooth, which introduce variable latency that can make the robotic hand feel less responsive.

Power efficiency also played a major role in the selection. The glove is powered by a small USB power bank, so the wireless module must draw as little current as possible. The nRF24L01 excels here, using only a few milliamps in receive mode and around 15 mA when transmitting. This allows the glove to run for extended periods without draining the power supply, which fits the project's goals for long demonstration time. In contrast, many Wi-Fi chips consume significantly more current, even when idle, which would reduce runtime and introduce heat concerns inside the glove enclosure.

The packet structure of the nRF24L01 is another reason it fits the GRIP project well. It supports fixed-length data frames, automatic packet acknowledgements, and built-in error checking. These features reduce the software complexity on the Arduino and help ensure that each update from the glove is delivered reliably to the hand. If a packet fails, the module handles retries internally without the Arduino needing to implement additional logic. This simplifies development and reduces latency caused by retransmission handling in user code.

Hardware integration was also straightforward. The nRF24L01 uses SPI communication, which the Arduino Nano already supports with dedicated pins. The wiring is simple, and the required libraries are stable and well-documented. This reduces development risk and allows more time to be spent improving system performance rather than troubleshooting communication issues. The small physical size of the module also helps keep the glove lightweight and reduces the amount of internal wiring required.

Cost and availability provide additional practical benefits. The nRF24L01 is inexpensive and widely available, which is ideal for a student project that may require multiple prototypes or replacement modules. Because the module is so common, it also provides strong community support and numerous examples that can be referenced during development.

Overall, the nRF24L01 offers a strong combination of low latency, reliability, energy efficiency, and easy integration. Its predictable behavior and straightforward programming model make it the best choice for enabling real-time wireless communication between the glove and the robotic hand.

3.2.3 Motor

MG996R was selected as the actuator class for the hand because it combines adequate torque in a standard package with simple three wire PWM control and a well-supported ecosystem of horns, brackets, and spares. The integrated position loop and metal gear drivetrain provide a straightforward path to tendon-spool actuation without external drivers or encoders, and the cost profile enables multiple degrees of freedom within a student budget. Availability from multiple vendors reduces supply risk and eases replacement.

Known compromises are backlash in the geartrain, audible operation, and lack of host side telemetry are acceptable for a first pass grasping hand where the priority is functional motion and repeatable positioning rather than precision metrology. Current surges at high load are characteristic of this class but are manageable within typical 6 V rail designs, so the overall balance of capability, price, and integration simplicity favors MG996R for the initial build.

The MG996R servo was chosen as the actuator for the GRIP robotic hand because it provides the strength, speed, and durability needed for tendon-driven finger motion while remaining affordable and easy to integrate. During the comparison stage, several servo options were evaluated, including budget plastic-gear models, micro servos like the SG90, and more advanced metal-gear servos. While lighter servos can reduce weight, they do not provide enough torque to reliably pull tendons during full finger flexion. The MG996R delivers significantly higher torque than these alternatives, making it much better suited for actuating a multi-finger robotic hand.

One of the main reasons for selecting the MG996R is its mechanical robustness. It uses a metal-gear train instead of plastic gears, which greatly increases its reliability during repeated loading cycles. Tendon-driven systems naturally create varying levels of resistance as the finger positions change. Plastic-gear servos often wear quickly in such applications, developing backlash or even gear teeth damage. The MG996R's metal gears maintain alignment under load and provide consistent performance even after many cycles. For a senior design project expected to undergo continuous testing and demonstrations, this durability is essential.

Torque output played an equally important role in the selection. The MG996R provides significantly higher stall torque than smaller servo options, which allows it to move the fingers smoothly and maintain strong grip forces when required. Even though the project does not aim to lift heavy objects, the servo must overcome tendon tension, friction within the finger joints, and any additional force applied by the user during testing. Lower-power servos struggle during these tasks, causing jitter or incomplete movements.

The MG996R's higher torque ensures that each finger can complete its full range of motion reliably.

Electrical requirements also influenced the decision. Although the MG996R draws more current than micro servos, the project's 2S2P battery configuration and planned voltage regulation system comfortably support its demands. With proper regulation and bulk capacitors, the servo's current spikes can be managed without causing voltage sag. This means the system can take advantage of the servo's high performance without risking instability. The choice to use stronger servos also reduces the number of mechanical compromises needed when designing the tendon system.

Integration with the mechanical design was another factor. The MG996R has a widely used mounting pattern and a large selection of servo horns, which makes it easier to attach custom tendon spools. Many 3D-printed robotic hand projects use this servo, which means there is reliable community knowledge available for troubleshooting or improving the design. Its physical size also works well with the enclosure layout, allowing the servos to sit securely inside printed mounts without excessive dead space or weak structural areas.

Cost and availability further supported this choice. The MG996R offers performance comparable to more expensive servos at a fraction of the price, which helps keep the project within budget. Because senior design projects often require replacing components during testing, selecting an actuator that is affordable and easily sourced provides additional peace of mind.

Overall, the MG996R offers the ideal mix of torque, durability, affordability, and compatibility for a tendon-driven robotic hand. Its strong mechanical characteristics and well-supported ecosystem make it the most reliable actuator choice for the GRIP project.

3.2.4 Sensors

ZD10-100 flex strips were selected because they provide a direct, passive measurement of finger bend as a simple resistance change that can be read with basic analog circuitry. Response time is fast enough for control loops in the tens to hundreds of hertz, and the strip format conforms well to finger geometry and glove materials. The absence of powered skin contacts simplifies electromagnetic compatibility considerations and reduces biocompatibility and usability overhead compared with electrode-based approaches.

While resistive bend sensors exhibit nonlinearity, material aging, and some temperature dependence, these effects are predictable and acceptable for capturing finger posture in this context. The one-sensor per finger model keeps wiring and bill of materials modest

and supports straightforward maintenance or replacement, aligning with cost and schedule constraints while delivering the fidelity needed for reliable pose capture.

The ZD10-100 flex sensor was selected for the glove input system because it offers a reliable, repeatable way to measure finger bending without requiring complex signal conditioning or additional mechanical structures. During the comparison phase, several sensing options were evaluated, including IMUs, strain gauges, Hall-effect angle sensors, and commercial bend-detection modules. While each option has strengths, the ZD10-100 flex sensor aligns most directly with the project goals: simplicity, low cost, easy integration with the Arduino Nano, and smooth mapping from finger movement to servo actuation.

The most important reason for selecting the ZD10-100 is its predictable resistance-change behavior. The sensor increases resistance as it bends, and this change correlates consistently with the user's finger movement. When placed in a voltage-divider configuration, the output voltage can be read directly by the Arduino's analog inputs without requiring amplifiers or specialized circuitry. For a wearable glove, this simplicity is extremely valuable because it keeps the wiring compact and avoids the need for bulky hardware on the back of the hand.

Mechanical flexibility also influenced the selection. The ZD10-100 is thin, lightweight, and conformal, meaning it bends naturally with the finger. Strain gauges and IMUs, by contrast, require rigid mounting frames or calibration routines to remain accurate. Flex sensors, however, can be sewn or bonded into the glove itself, allowing the glove to remain comfortable while still providing reliable measurements. This reduces mechanical complexity and allows the glove to feel natural for the user.

Another factor is signal stability. IMUs are powerful but can drift over time and require fusion algorithms to combine accelerometer and gyroscope data. For a senior design timeline, developing and tuning an IMU-based finger-tracking system would introduce unnecessary complexity. Hall-effect sensors require custom joint housings to track rotational angles, which is incompatible with a fabric-based glove. The ZD10-100 avoids all of these issues by producing an immediate, direct measurement of bend angle that the Arduino can interpret with minimal processing.

The electrical characteristics of the ZD10-100 also match the project well. Its resistance range (typically around 10–30 kΩ depending on bend angle) fits comfortably within the Arduino's ADC range when paired with an appropriately chosen fixed resistor. Because each sensor uses only two wires, the overall wiring remains simple and manageable. With five flex sensors, the glove still uses fewer connections than other sensing systems that require multi-axis calibration or multiple supply rails. The sensor's analog nature allows

for smoothing and filtering in software, producing fluid, natural-feeling finger motion once mapped to servo positions.

Cost and availability played a supporting role in the decision. The ZD10-100 is inexpensive and readily available in multi-pack quantities, which is ideal for prototyping. Because flex sensors can eventually degrade due to repeated bending cycles, having easy access to replacements ensures long-term reliability during testing and demonstrations. Many open-source and academic projects also use similar flex sensors, which provides a large base of reference designs and community troubleshooting support.

Overall, the ZD10-100 flex sensor provides the right mix of mechanical flexibility, electrical simplicity, analog reliability, and cost-effectiveness for the GRIP glove. It integrates cleanly into the existing microcontroller and wireless architecture while keeping the glove comfortable and easy to wear.

3.2.5 MOSFET

BSS138 was chosen as the utility MOSFET for light switching and level translation roles because it turns on at logic-level gate voltages, has very low gate charge, and is widely available in a compact SOT-23 package. Its electrical characteristics are well matched to small-signal loads and signal interfacing, allowing clean control with minimal drive effort and negligible board area impact. Multiple second sources and consistent pinouts simplify procurement and substitution.

The device's current capability is intentionally limited compared to power MOSFETs, but that aligns with its intended use as a signal-level switch or translator rather than a motor or rail switch. Within those boundaries it provides a reliable, inexpensive building block that avoids the overhead of larger packages or gate drivers for light duty tasks in the design.

The BSS138 MOSFET was selected for the GRIP glove circuit because it provides a compact, low-power, and reliable solution for controlling logic-level signals and interfacing different voltage domains. Although the project does not require high-current switching on the glove side, it does require components that can handle small control signals cleanly and predictably. The BSS138 fits this role well due to its low gate-threshold voltage, small footprint, and stable behavior even with limited drive current from the Arduino Nano.

One of the main reasons for selecting the BSS138 is its ability to turn on fully at low gate voltages. Many MOSFETs require 4–10 volts on the gate to switch properly, which makes them incompatible with 5 V or 3.3 V microcontrollers. The BSS138, however, is designed for logic-level operation and begins switching reliably at gate voltages around

1–2 volts. This ensures that the Arduino's GPIO pins can control it without additional buffering hardware. Even though the GRIP system primarily uses the MOSFET as a low-power utility switch, this predictable behavior ensures clean transitions and avoids partial-on states that could cause unstable operation.

The device's small physical footprint also benefits the glove design. Because the glove houses multiple flex sensors, wires, and the nRF24L01 module in a compact space, there is very little room for large components. The BSS138's SOT-23 package makes it easy to include on small breakout boards or custom PCBs without adding unnecessary bulk. This keeps the glove lightweight and prevents the wiring from becoming cluttered or uncomfortable during use.

Although the servos themselves draw high current, that current never flows through the MOSFETs on the glove. Instead, the BSS138 is used for lightweight switching tasks, such as enabling or isolating signals, controlling small loads, or acting as part of a level-shifting stage. In these roles, the MOSFET's low drain–source on-resistance is sufficient to ensure minimal voltage drop while still being easy to drive from the microcontroller. Its electrical characteristics are stable across a range of operating conditions, which helps maintain consistent behavior even as the glove flexes or the wires experience movement.

Another advantage is the widespread use of the BSS138 in educational and hobbyist projects. It is a standard component in many level-shifting modules and logic control circuits, which means it is well-documented and easy to source. This reduces the likelihood of supply issues and makes it simpler to replace or troubleshoot the MOSFET if needed during testing. Because senior design projects can involve repeated prototyping cycles, having access to common components with known behavior is beneficial.

In addition, the BSS138 is cost-effective, which helps keep the glove subsystem within budget. While more advanced MOSFETs exist, their additional capabilities—such as high current handling or specialized packaging—are unnecessary for this application. Selecting a component that matches the project's actual electrical needs avoids overspending while still ensuring reliable operation.

Overall, the BSS138 is a practical and efficient choice for signal-level switching on the GRIP glove. Its low gate-threshold voltage, small size, and stable electrical behavior make it the most appropriate MOSFET for the project's low-power logic requirements.

3.2.6 Batteries/PSU

Standard 18650 cells were selected as the base element for the arm power source due to their high energy density, robust cylindrical construction, and broad second-sourcing ecosystem. The format supports a wide range of capacities and discharge ratings, and the metal can aid mechanical handling and thermal conduction within enclosures. Established CC/CV charging profiles and well-documented performance characteristics make system-level planning predictable.

Although multi-cell packs require protection and battery management provisions, these are routine for the format and are well supported by off-the-shelf electronics and pack hardware. The series/parallel flexibility of 18650 cells provides a clear path to meeting future voltage and current targets if the arm is pursued later, without committing to bespoke or hard to source form factors.

A consumer USB power bank was selected for the glove because it integrates lithium-ion cells, protection circuitry, charging control, and a regulated 5 V output in a single certified enclosure. The standardized port interface, state of charge indication, and internal safeguards reduce the number of custom power elements and simplify logistics such as charging and replacement. Market availability is broad, allowing capacity and size to be matched to endurance goals.

While the boost converter and enclosure introduce efficiency overhead and fixed form factor constraints, these are outweighed by the benefit of a turnkey, self-contained supply with predictable behavior and built in protections. For the glove's power demands and operating model, the convenience and safety integration of a power bank align well with time and budget priorities.

The 2S2P lithium-ion battery configuration was selected as the primary power source for the GRIP robotic hand because it provides the ideal balance of voltage, current capability, runtime, and stability required by the MG996R servos and the onboard electronics. During the comparison stage, various battery technologies were considered, including single 18650 cells, NiMH packs, LiPo packs, and higher-voltage Li-ion arrangements. While each option offers specific advantages, the 2S2P lithium-ion configuration aligns best with the system's electrical demands and mechanical constraints.

The main advantage of the 2S2P setup is its ability to deliver both high current and a stable voltage range. In a 2S arrangement, the two series-connected cells produce a nominal 7.4 V and a maximum of 8.4 V at full charge. This voltage is ideal for feeding a buck or buck-boost regulator to produce a consistent 6 V rail for the servos. Unlike a single-cell system, which struggles to maintain headroom for regulation as the cell discharges, the 2S configuration ensures that the regulator always has enough input

voltage to maintain stable output torque across the full battery cycle. When the cells are also paired in parallel, the total capacity doubles, allowing the system to maintain performance for significantly longer periods.

Current delivery capability is another major factor. Servos like the MG996R can draw two or more amps during stall conditions, and when multiple fingers move simultaneously, the total current demand rises quickly. A single 18650 cell cannot safely provide this level of current without overheating, voltage sag, or triggering a protection circuit. The 2S2P configuration, however, splits the load across multiple parallel cells, reducing stress on each individual cell. This distributes the thermal load more evenly and ensures that voltage remains stable even during demanding movements. The result is smoother servo operation and fewer instances of jitter or speed variation caused by inconsistent power delivery.

Runtime is also significantly improved with the 2S2P design. The parallel arrangement doubles the amp-hour capacity of the pack, which allows the robotic hand to operate for one to three hours depending on activity level. This extended runtime is important for senior design presentations, demonstrations, and prolonged testing sessions. It prevents the need for frequent recharging cycles and ensures that the system remains responsive throughout extended trials.

Safety and stability also influenced the selection. Lithium-ion cells offer high energy density and reliable cycle life when paired with an appropriate battery management system (BMS). A 2S protection board can monitor each cell string for overcharge, over-discharge, and short circuit conditions. This is harder to accomplish with a large number of single cells or with mixed chemistries like NiMH. The 2S2P configuration keeps the pack simple while still offering robust safety features.

Mechanical integration was also considered. Four 18650 cells arranged in a 2S2P layout fit neatly into compact battery holders or custom 3D-printed compartments. Their cylindrical shape makes them easy to secure within the hand’s enclosure without adding unnecessary bulk. This consistent form factor also simplifies wiring and reduces strain on the connectors, which is important in a system subject to frequent movement and handling.

Cost and availability finalized the decision. 18650 cells are inexpensive, widely available, and come in standardized capacities and discharge ratings. This makes it easy to source replacements or upgrade to higher-drain cells in future revisions. Compared to specialized LiPo packs, which often require custom dimensions and connectors, the 2S2P setup offers far more flexibility during prototyping.

Overall, the 2S2P lithium-ion configuration provides the perfect combination of voltage stability, high current delivery, long runtime, and practical integration. It meets every electrical requirement of the GRIP servos while remaining safe, economical, and easy to package within the robotic hand.

3.2.7 Buck Boost Converter

LM5177 was selected because it is a proven four switch, non-inverting buck-boost controller that cleanly regulates a fixed output when the input can be above, below, or around the target. That topology is ideal for a battery powered system whose supply voltage varies under load, since the converter maintains a continuous conduction path and seamless transitions across buck, buck-boost, and boost regions. As a controller it drives external MOSFETs and scales to higher current simply by choosing appropriate FETs, magnetics, and capacitors, which provide headroom for peak loads without changing the control silicon. The device includes the expected control features current-mode operation, programmable soft start, compensation access, and light load options along with cycle-by-cycle current limit and robust fault handling, so the regulation behavior remains predictable under transient and fault conditions.

Compared with integrated buck-boost parts, LM5177 asks for more external components and careful layout, but the payoff is efficiency and thermal margin at higher power levels and the flexibility to optimize the power stage for the exact voltage and current profile. Documentation and reference designs are mature, with clear guidance on gate-driver routing, switch node containment, and current-sense implementation, which reduces risk during the first spin. In short, it combines the right control architecture, scalability, and support collateral to meet a variable input, fixed output requirement with sufficient performance headroom for future iterations.

Chapter 4 Related Standards and Design Constraints

4.1: Industrial Standards

4.1.1 Biomedical Device Standards

Electromechanical biomedical device for prosthetic uses, as well as laboratory usage. The Bionic hand classification is in part electromechanical, it has to comply with relevant standards to ensure the pillars of safety, reliability and performance are present. This is especially necessary for human interfacing systems. Relevant standards are necessary to ensure the hand design meets requirements for human interaction, electrical safety, and mechanical durability.

Since the device interfaces directly with a human test subject the design must comply with all medical standards for human to robot interaction, as well as intermediate production safety. The below standards ensure that the project bionic hand is not only manufacturable but robust, safe and compliant for human use under any FDA , IEEE and IEC regulations. In addition to the biomedical interpretation of the project we also accounted for the proper laboratory robotics and automation standards and constraints. The vision is to have this device accessible in laboratories for experiments as well as medical capabilities of its design.

IEC 61326-1 – EMC for Lab Equipment

The system we intend to use intertwines low level flex sensor signals with PWM motor currents, in this case the compatibility of the electromagnetism is essential. The rule sets immunity and emissions requirements for lab equipment enabling glove accurate reads even near other RF sources. We intend to leave all walkie talkie and any other possible equipment that is harmful to the readings of the equipment left at a safe distance away from experimentation. Also frequent and consistent records of measurement design and compliance readings for the project. One thing that is possible is to test in hopes of protecting against a charged item of clothing or something that can inadvertently damage the equipment.

ISO 13485

This specification highlights the quality management system for medical devices. Ensure consistent design, and servicing to meet regulatory requirements.

ISO 10993

Covers the biocompatibility of materials that are used in medical devices that come in contact with human skin, ensuring plastics silicone, and three 3D printed parts do not cause irritation or harmful effects like toxicity.

IEC60601-1/2 Medical Electrical Safety & EMC

Creates general safety and performance standards for medical electrical equipment this standard discusses, power limits and electrical isolation. This means that using basic casing and protective wire connectors and proving the system remains safe under all conditions. Effectively we simplify lab approvals and lets teammates handle debugging of all parts including the glove and controller components. Also to meet this standard its best we keep all hot surfaces and drivers below the regular temperature limits.

Robotics and Automation standards:

Additionally, standards must fall under the general umbrella of robotics safety and Automation due to its sensitive and multifunctional capability. Baseline bullets for criteria for robot architecture, motion control, force pressure/ limits in human interaction and finally system risk reduction in shared workspaces.

Health and Safety – User Bystander Protection

The device will have a maximum grip force, all measurements are taken and pinch points will be minimized. We plan to implement an emergency stop wire or functional button to cut off all communication with the sensors and glove. In other words we will incorporate a "before putting on the glove" checklist for all users including the design team. In addition adding a two person rule for first time users to make everything more safe.

ISO 10218 -1: 2025 – Robotics Safety requirements Part 1:

Lists safety requirements for industrial robotics. This particular requirement addresses it as partly completed machinery. It's necessary to have clearly defined sop behaviors programmed for emergencies and the code will include this, to satisfy this rule we will have all accurate documentation on the stopping behavioral procedures and experiment on what those will look like. We have a built in safety function already with designed performance levels, refer to flex sensor specs for movement, less than 20 grams of weight on the sensor will never move. Intuitively, the sensor has precautions for sudden readings as well with the limits in flex sensor specifications.

IEC 62366-1 Usability Engineering

The human error component of this design is prevalent even with proper safety procedures present. To combat this we will design mapped task flows and test for confusion or unsafe readings before each use. Good usability here is also safety, fewer inadvertent grasps and mindful motion of the equipment. It must be stated that each user will be properly informed on procedural guidelines before each use. Another way to combat this is to have an itemized list of things to know before putting on the glove.

ISO /TS 15066: 2016 – Collaborative robots.

International organization for standardization. Applying safety rated monitors stops safe speeds to servos and integrated guidance derived for allowable contact forces. While not deploying a full robot the hand has human proximity with the device and proper speed and separations and basic limits anyone can grasp have to be implemented. We intend to

select contact limits from specifications and constrain the torque and speed of the servos. Also verifying a monitored stop is incorporated in firmware.

ISO 13482: 2014

Robotics and robotic devices – Safety and personal care robots. Addresses service/ assistant robots interacting with humans. This helps define acceptable hazards and protective measures for including contact scenarios in non medical settings.

IEEE 1872 – 2015

IEEE standard ontologies for Robotics and Automation, defines ontology for robotics and automation systems. Design compliance approaches structure our software messages and logs to label tasks states and entities all the time. The debugging helps while enabling downstream tools to reason about behavior. It also makes requirements and some cases more explicit for testing.

IEEE 7007-2021

Ontological standard for ethically driven robotics and automation, focuses on ethical framework and robotics. We will ensure human in the loop oversight and the user can always override the device and cannot lock a grip or anything of that nature.

4.1.2 PCB Standards

The central key hardware elements of the device relies on Flex sensors, microcontrollers and motor servos that are interchangeable and integrated on PCB. W will anchor design to IPC 2221 and IPC 6012 to ensure manufacturability and repeatability.

4.1.3 Sensor and Signal Standards

Flex sensors used for motion detection follow the IEEE standards, including IEEE 1451 for smart transducers. This standard covers the calibration, digital communication and identification methods for sensors. The chosen flex sensors are the ZD10-100, a FSR style resistive strip. Applying force changes its resistance values so the user can have full operational control of the hand portion of the device with one flex sensor over each finger.

The Sensors are finger mounted to generate repeatable pressure on the sensor. Yielding a 0-500 gram rand is great for gloves since the curling of the finger under a pad can generate a lot of weight in that range and always clearing eh less than 20 gram activation

threshold. In tandem with the software the fast response of less than 10 ms supports low latency control and the cycles rating is great for frequent usage that will last a long time.

In terms of the safety of the sensors the sensor values are designed to only read absolute values of the sensor so even if the user were to make irregular movement with their model hand, the device would just open and close its hand without any backwards joint motion or unsafe usage.

Flex Sensor (ZD10 -100) Design parameters

Sensor raw signals are prone to noise, drift, and variability. These problems are handled by the processing layer:

- **Calibration:** Flex sensor outputs are normalized for varying hand sizes and natural resting states by taking baseline readings for each user at startup.
- **Filtering:** A low-pass filter removes high-frequency noise from the IMU, while a moving-average filter smoothes out abrupt oscillations.
- **Mapping Functions:** Valuable joint angles are derived from sensor measurements. Servo angles of **0 to 90°**, for instance, are mapped to a flex sensor range of 0.5 to 3.5 V. When joint motion is not exactly proportional to voltage, nonlinear mapping functions are employed.
- **Data Packaging:** Finger angles, wrist orientation, and metadata like sequence numbers are all included in the small data packets that are created from the processed values.

Table 4.1 Flex Sensor Device Data

Parameters	Data
Length x Width:	100 x 10 mm
Thickness	< 0.25 - 0.30 mm
Durability:	> 1,000,000 actuations
Measurable Range:	0-500 grams normal force

Response/ Recovery:	< 20g response point

4.1.4 Actuator

The Servos and motors controlling finger motion fall under the NEMA and IEC standards umbrella:

NEMA 17 or NEMA 14 micro stepper/ servo motors are used for compact torque control.

ISO 10218 -2 – if the bionic hand is mounted to a robot arm or otherwise interacts in a cell this governs integration and operational maintenance.

IEC 60034-1 for the mechanical performance/ thermal limits and duty cycles.

The intention of the actuator subsystem is to convert glove flex sensor into finger motion for the current prototype we target the tendon driven fingers via flex sensors, powered via rotary servos around 5-6 Volts. This is a great choice due to its budget friendliness and compact arduino functionality. This architecture is fail safe to open hand when power is removed aligned with iec 80601 -2 -78 expected performance expectations.

4.1.5 Communication and Control Standards

Internal control system utilizing the I2C and SPI communication between the microcontroller and servos. The microcontroller of the robotic arm interprets incoming packets and uses PWM signals to drive servo motors:

- **PID Controllers:** A PID loop is used by each joint to minimize overshoot and guarantee precise positioning.
- **Safety Constraint:** Software limits keep the servos from going beyond their mechanical ranges, which is a safety constraint. Values are clipped to safe limits if data is received that is out of range.
- **Fail-Safe Procedures:** The arm returns to its neutral rest position after 500 milliseconds if no valid packet is received. Before putting on the glove procedures.

- **Servo Synchronization:** To produce organic, coordinated movement, motion is interpolated across several joints.

The following tests are carried out to guarantee performance:

- **Latency Tests:** End-to-end response measured between finger movement and arm response (<100 ms target).
- **Stress Testing:** Continuous operation for 30+ minutes without overheating or communication dropouts.
- **Error Injection:** Artificially corrupted packets verify CRC detection and recovery routines.
- **Unit Tests:** Validate calibration, mapping functions, and PID controller logic.
- **Integration Tests:** Full glove-to-arm operation across multiple users to confirm reliability.

4.2 Practical Constraints

Time remains a major constraint of the project since a senior design capstone project development is limited to 2 semesters along other classes. By May spring 2026 the concept validation, circuit design and coding need to be done along with mechanical assembly. Some delays we can run into is the PCB shipping/ manufacturing and the constant adjustments to 3D prints and fittings.

We regularly have changed our design or even something as small as one finger over five times. To mitigate the stress of this we established weekly milestones and we emphasize work in pieces. The tasks get completed well in a timely manner thanks to a great group of engineers.

4.2.2 Economic

- The total budget allocated by the sponsor or university is $500 - $800.
- Sensor Choice: Flex sensors are ($12-20) limit redundancy in design.
- Microcontroller selection: Arduino Nano

- Actuators: Mini servos instead of high end brushless motors

Table 4.2 Budget Table

Hardware Expenses		
Part	Cost ($)	Quantity
ZD10-100 flex sensor 500g	8.81	5
MCP1700 3.3V regulator (10pcs)	8.99	1
mg996r servo motors (4 pcs)	15.98	1
nRF24l01 (4pcs)	7.89	1

4.2.3 Environmental

Several environmental constraints exist, temperature range 0 -40 degrees for normal operation. Humidity < 70% non-condensing to prevent corrosion Dust Resistance including Plastic or silicone casing to protect joints and circuitry.

4.2.4 Sensor and signal Constraints

Flex Sensor constraints would be the electrical range resolution, nonlinearity, noise, Mechanical integration, durability and firmware. Starting with the electrical resolution the Nano has an ADC of 10 bits with a 5 V reference, with respect to its internal 1.1 V reference the dividers will hardly ever grab the full 1.1 V reference. The team has considered using an op amp gain to improve this resolution, and source impedance considering the glove material to be less than 10 k ohms.

Film based flex devices like the flex sensors in most cases are highly nonlinear and tend to experience drift with longevity stricken conditions. Also the servos and motors add some unwanted noise to the equation. We physically want to separate the sensor wiring from motors.

The Arduino Nano Constraints (ATmega328P)

Basic specification of the microcontroller:

Classic Nano = 16 MHz

2KB SRAM

ADC performance

Throughput of ADC = 10 kS/s

ADC clock can raise 50-77 kS/s but this costs some accuracy which we intend to not need to risk in any case. The I/O pins feature 8 analog channels (A0-A7), PWM with 6 channels at a timer dependent 490-980Hz. Worth noting intention to design regulators for use on the side of the PCB design to keep within a good voltage range. Power and thermal respect 20mA per pin with a maximum of 40 mA equalling 200 mA total MCU current.

4.2.5 Ethical

Sourcing factors into many ethical constraints when conducting research, some companies use sickening or otherwise disgusting practices to conduct research and manufacture products. Our project doesn't intend to take advantage of third parties or exploit any work. All companies chosen that went into our project are reputable in their business practices and have good standing in the general engineering community. Insurmountable ethics will not be anywhere near the design and our team or project. The goal of the GRIP is to have absolutely no danger involved in the mechanical processes, fitting of the glove, or user error in any capacity. The ethics of the project remains always at or below a one out ten on a common sense pear to pear difficulty scale.

4.2.6 Sustainability

The design in how it is envisioned allows for interchangeability with the sensors and motors. Interchangeability allows for reusability and more room to modulate or play around with the framework. In addition to the casing mentioned in other sections we intend to have safe grounded casing for the device to always have for protection and longevity.

Chapter 5 Chat GPT Comparisons and Usage

5.1 Introduction

In November 2022 Chat GPT, was released and has evolved at an exceptional rate. Our team was able to work with this AI to verify our documentation, enhance our vision of the project and gather useful information with excellent efficiency. In general the case between members usage varies while the software design would benefit from help with chat GPT, our usage of it was limited.

Chat GPT has helped us all clarify technical concepts and its integration with its google search engine was beneficial as a trusted source of information. Some parameters and extended knowledge was gained with excellent bibliography to quickly get to our sources. We will comment from a standpoint of efficiency and productivity ChatGPT is a great asset in almost any technical space.

The rapid design framing is great as well for scoping and turning the glove into hand into a concrete plan of action. It can draft block diagrams and it was helpful for us to visualize how we want the components to set us on the hand. The catch is sometimes it can be confident in wrong responses so its best to keep its use somewhat limited for extremely high level tasks.

5.2 Case Study 1 — Protocol Selection: SPI (nRF24L01) vs BLE

Early in the semester, our team had not yet decided which wireless interface would connect the glove subsystem to the robotic hand. Two primary options emerged:

1. Bluetooth Low Energy (BLE)
2. nRF24L01 SPI Radio Modules

Our initial assumption favored BLE because it is widely supported, used in wearables, and easy to pair with consumer devices. However, when we asked ChatGPT to compare BLE with the nRF24L01 in a point-to-point real-time control system, the tool provided several insights:

- BLE requires pairing and acknowledgment layers that increase latency.
- BLE is optimized for low-bandwidth, intermittent communication rather than fast servo updates.
- nRF24L01 modules offer low-latency (<1–5 ms) packet transmission and configurable data-rate profiles.
- BLE is more prone to interference when operating in crowded 2.4-GHz environments.

Using ChatGPT's analysis, we validated our final communication choice by reviewing multiple datasheets and community hardware reports from robotics forums. These sources reinforced concerns about BLE latency and connection stability. The nRF24L01-based SPI interface was therefore chosen based on:

- Faster continuous packet transmission
- Fewer software layers between the application and RF hardware
- Bidirectional payload support with ACK ability
- Simpler firmware implementation

This early decision was crucial because communication choice determined how firmware, packet structure, and timing loops would be designed. ChatGPT helped accelerate the research process by summarizing trade-offs and helping us articulate justification to our professor. Ultimately, the decision aligned well with our performance goals and supported a cleaner first-semester implementation.

5.3 Case Study 2 — Firmware Mapping and Curve-Shaping Techniques

One of the most important steps in the GRIP system is mapping raw flex-sensor voltages into usable joint-angle commands. Our first implementation used linear mapping between ADC values (0–1023) and servo angles (0–180°). This worked mechanically but produced abrupt motion near low-bend regions and poor precision during fine hand movements.

We consulted ChatGPT for alternative mapping methods. Several options were discussed:

- Exponential curve shaping

- Polynomial approximation
- Logistic compression
- Piecewise mapping
- Moving-average smoothing

From these, ChatGPT recommended a tunable exponential mapping function such as:

$$\theta = (norm)^{k} * 180\, degrees,\ where\ k \approx 4 - 5$$

This form improved sensitivity for smaller bends and provided controllable smoothness. After testing, we selected power ≈ 4.2 as a balance between responsiveness and stability. Additionally, ChatGPT suggested averaging windows of 3–5 samples to reduce jitter. We implemented a small median-plus-mean smoother to further stabilize noisy readings.

This approach dramatically improved movement quality. Even before integrating the robotic hand, we achieved responsive visualization inside our C# arm-simulation tool. The assistance from ChatGPT accelerated our transition from naïve mapping to a more principled signal-conditioning pipeline.

5.4 Case Study 3 — System Block-Diagram Refinement

Our early block diagrams were very detail-heavy, including explicit ADC-conversion boxes, register-stage mapping, and other low-level operation layers. During design review, peers warned that this might appear overly granular, especially in SD1 where we had not yet implemented all modules.

We asked ChatGPT whether ADC conversion must appear explicitly. The tool emphasized that while ADC is necessary for analog flex sensing, it is typically implied within the MCU's data-acquisition block. Only when conversion timing, sampling strategy, or precision trade-offs are important should it be diagrammed separately.

Given that SD1's primary focus was the glove–hand pipeline (not ADC theory), our diagrams were simplified:

Old (too detailed):

Sensor → Voltage Divider → ADC → Filter → Exponential → Mapping → Packetization

Revised SD1 diagram:

Sensor → Pre-processing + Mapping → Packetization → Transmission

This simplification better matched the professor's guidelines for SD1, keeping emphasis only on the most functionally meaningful components. ChatGPT's guidance ensured our diagrams were leaner, more readable, and aligned with higher-level system requirements.

5.5 Case Study 4 — Visualization Tool Selection (C# Simulation)

Before full mechanical integration, we wanted to test joint-angle behavior using a simulated robotic arm. Initially, we considered MATLAB, Python (MatPlotLib + Tkinter), Unity, and C#.

ChatGPT outlined the following trade-offs:

Table 5.5.1 ChatGPT generated comparisons

Platform	**Pros**	**Cons**
MATLAB	Fast prototyping, built-in math tools	License, less portable
Python	Simple, flexible, highly accessible	GUI rendering slower
Unity	Powerful graphics	Overkill, steep overhead
C#	Strong UI tools, IK libraries, lightweight	Requires structuring

Based on these comparisons — and our preexisting familiarity with C# — we selected a C#-based desktop simulation that implemented both Forward Kinematics (FK) and Inverse Kinematics (IK).

This tool became one of the most useful outcomes of SD1:

- Allowed real-time angle visualization
- Enabled drag-based IK testing
- Allowed UI display of θ_1 & θ_2 values
- Provided initial validation without hardware risk

ChatGPT also provided useful pseudo-code guidance for 2-link IK geometry.

5.6 Case Study 5 — Safety Control: Servo Limits & Fail-Safe Behavior

Real-time servo control poses safety risks: sudden angle jumps can damage hardware. When consulting ChatGPT about motor-control reliability, it recommended several best-practice techniques:

- Angle limiting (software clipping)
- Neutral-pose fallback when communication is lost
- Slew-rate limiting where $\Delta\theta$ per cycle is capped
- Interpolation when packets are dropped

We implemented two of these during SD1:

1. Angle clipping ensured that no command exceeded the physical servo range.
2. Timeout fallback puts the servo into a neutral posture if >500 ms passed without valid data.

Implementing these behaviors early produced more stable and predictable motion. It also established safety patterns that will be extended next semester when PID and shoulder DOF are added.

5.7 Case Study 6 — Code Documentation & Readability

We also used ChatGPT to help refine firmware structure and documentation. While ChatGPT did not write code for us, it helped reformat messy blocks into cleaner functions (e.g., separating readSensors(), applyMapping(), and sendPacket()), which improved readability.

The assistant also helped suggest comments and header structure, resulting in more maintainable code.

5.8 Case Study 7 — Literature & Prior-Art Review

ChatGPT helped us summarize relevant concepts, including:

- 2-link IK geometry

- Nonlinear control curves
- Wireless-packet CRC mechanisms
- Calibration routines

These summaries guided deeper manual research using vendor datasheets and robotics references.

5.9 Limitations of ChatGPT

While valuable, ChatGPT had several limitations:

- Sometimes provided inaccurate hardware current specs
- Occasionally suggested libraries not compatible with Arduino Nano
- Provided simplified examples requiring manual validation
- Needed cross-checking against datasheets and real testing
- Does not replace instructor approval, peer review, or experimentation

Thus, ChatGPT was most useful early in the design process — not as a replacement for technical evaluation.

5.10 Lessons Learned

1. ChatGPT accelerates research but must be validated with real testing.
2. Its greatest value is summarizing trade-offs when choices are unclear.
3. It improved our ability to articulate design justification.
4. It helped structure firmware more cleanly by suggesting modular organization.
5. Overuse can lead to vague justification — final reasoning must come from testing or literature.

5.11 Conclusion

In summary, ChatGPT played a meaningful but appropriately bounded role in the GRIP SD1 software workflow. By providing rapid conceptual comparison, architecture brainstorming, and code-organization suggestions, it helped us reach important decisions more quickly — particularly the selection of SPI-based radios and exponential mapping functions.

It did not replace engineering judgment, debugging, or experimentation. Instead, it functioned as a design accelerator, helping us narrow choices before validating them through hardware testing and instructor feedback.

The integration of AI-assisted research with hands-on firmware development contributed to a well-organized and functional SD1 software pipeline, positioning the GRIP team for successful expansion into shoulder DOF and PID control during SD2.

Chapter 6 Hardware Design

6.1 Transmitter circuit design

The transmitting side of the project is the glove worn by the user, where flex sensors are attached to the fingers and used to measure the curvature of them. The ZD10-100 is used for this purpose. It is a variable resistor that increases in resistance when bent. Each flex sensor has one pin to ground and the other pin is connected to 5V with a 10k ohm resistor between the pin and 5V source, creating a voltage divider circuit. The output of this circuit is attached to an analog Arduino pin, where the changes in voltage due to the varying flex sensor resistance are measured. This process will be repeated for the other 4 flex sensors and the analog values will be put through an algorithm to translate them into servo positions for each of the fingers.

After the analog signals are collected, they will need to be transmitted to the arm via the nRF24l01 transceiver module. However, this module runs on 3.3V logic while the Arduino runs on 5V logic, direct connections can damage the nRF24. To fix this, we utilize the BSS138 N-type MOSFET to shift from a 5V signal from the Arduino to a 3.3V signal the nRF can properly use.

Powering this circuit, the Arduino Nano will supply the 5V needed by the flex sensor circuits via the 5V pin on the board. However, the nRF24, despite only needing 3.3V to be powered, requires more current then the Arduino can provide so a voltage regulator will be used. Powered by the Arduino’s 5V pin, the MCP1700 steps the input voltage down to 3.3V while producing 250mA of current to be used, more than enough to satisfy the nRF24 at peak loads and prevent potential brownouts.

Furthermore, capacitors are added to the Vin pins of the nRF24 and the output of the MCP1700 to smooth signals. The 0.1uF and 10uF capacitors used on the nRF24 are linked in parallel between the module's Vin and GND pins to mitigate noise. The 0.1uF capacitor reduces the high frequency, fast switching noise produced by the module's internal circuit, while the 10uF provides extra charge for low frequency dips during transmission spikes. The capacitors at the MCP1700 output are used to stabilize the output and help it respond quickly to sudden changes in current demand from the nRF24.

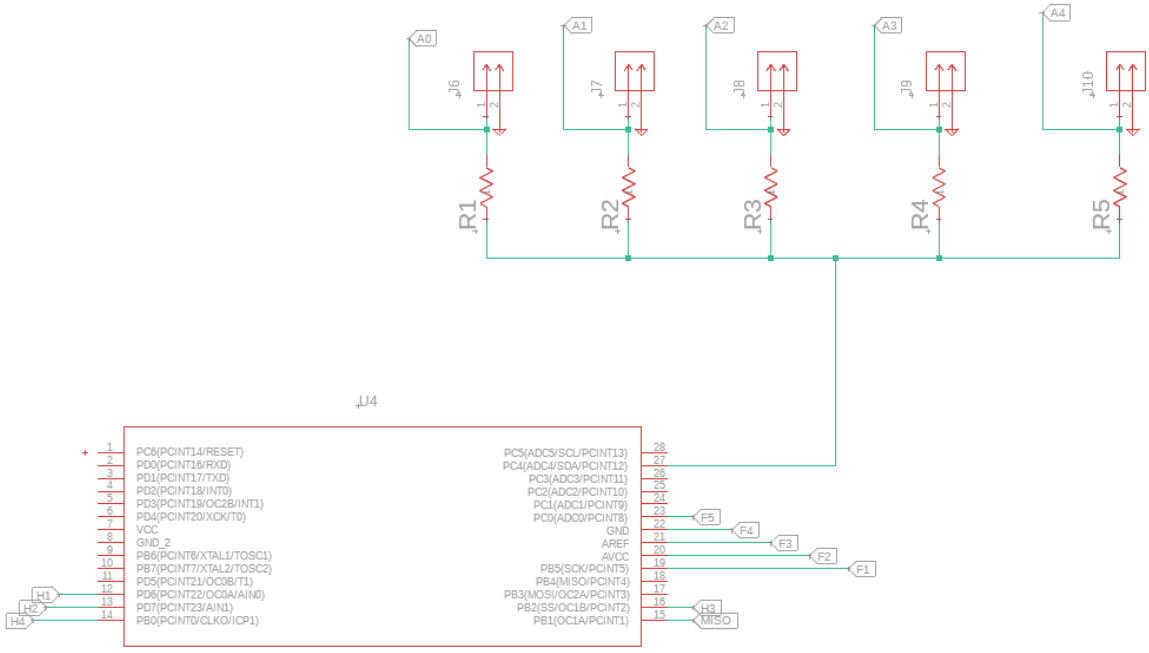


Figure 6.1.1 flex sensors with arduino connections

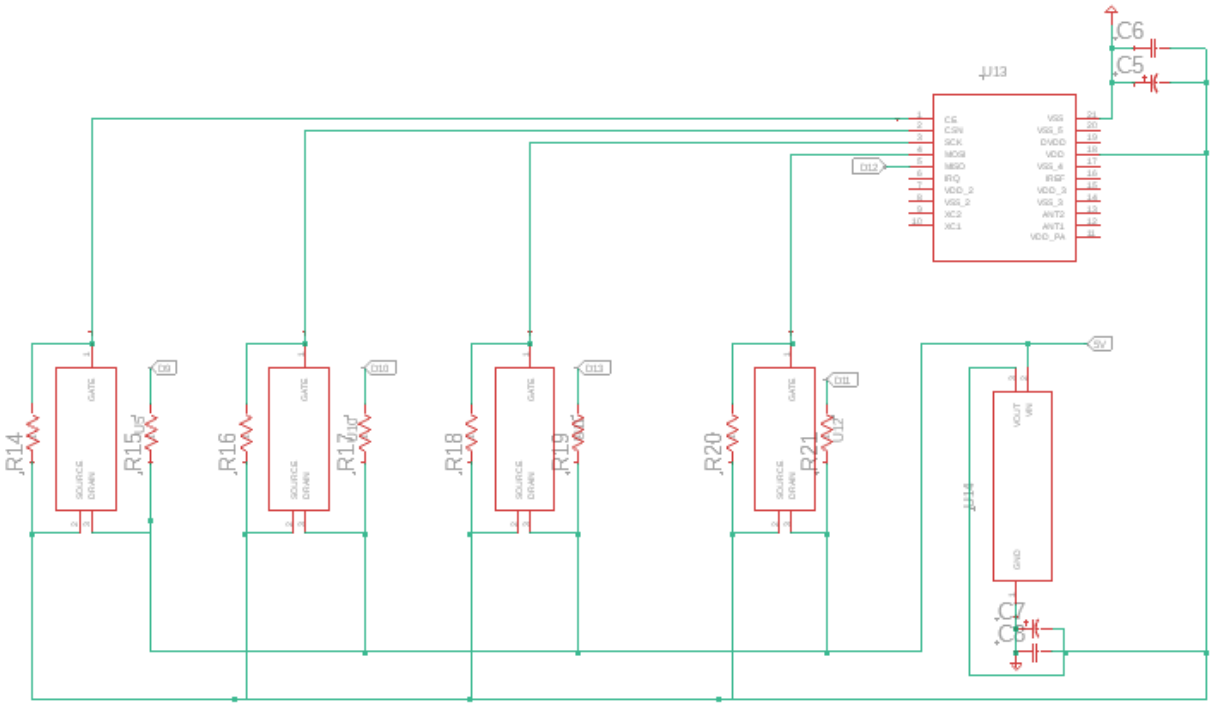


Figure 6.1.2 nRF24 to voltage regulator connections

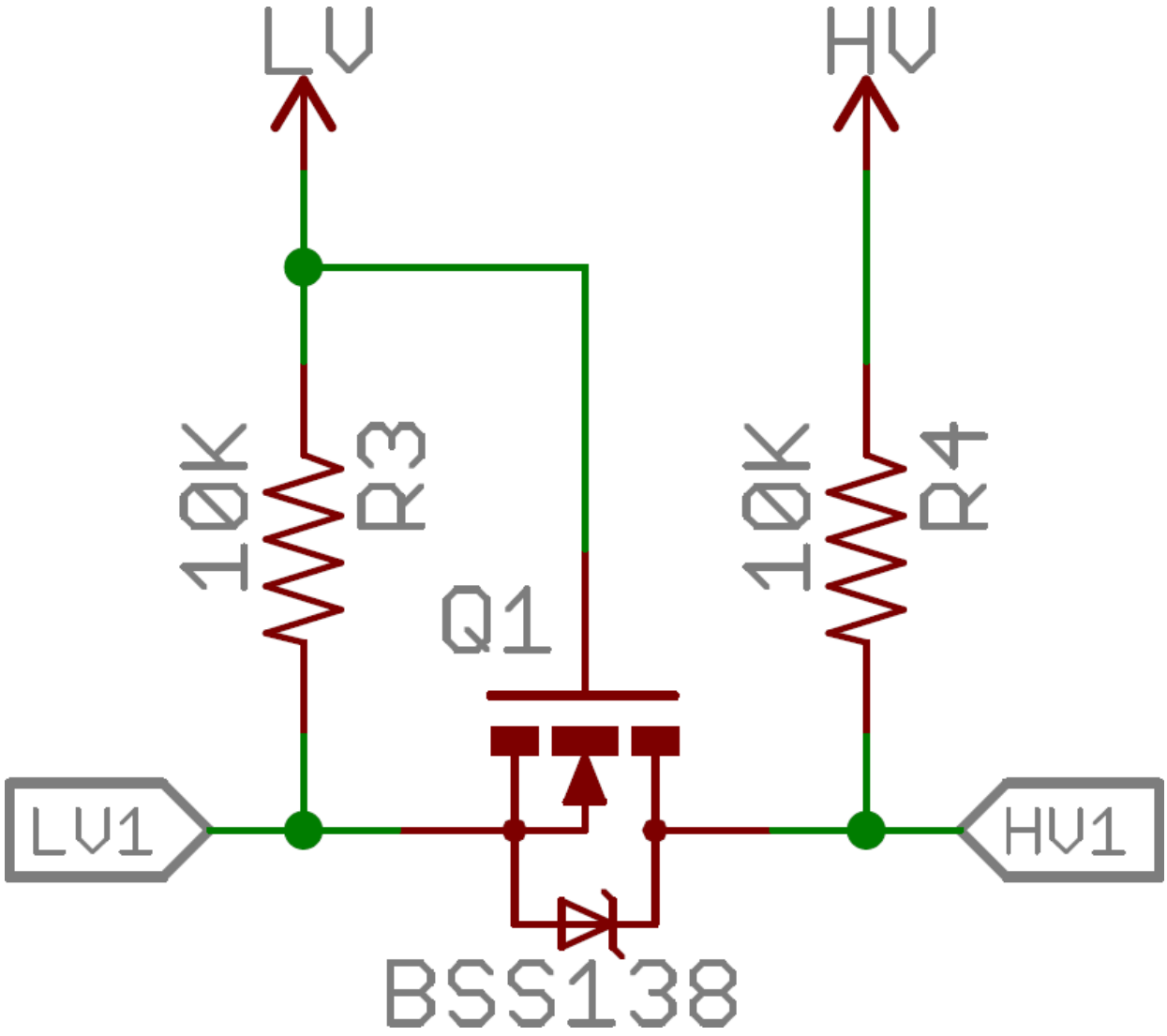


Figure 6.1.3 example of BSS138 MOSFET level shifter circuit

6.2 Receiver circuit design

The receiving side of the project is the arm itself, where the flex sensor values transmitted from the glove's nRF24 will be received by the arm's same module. These values will be translated into positions for the mg996r servo motors controlling the fingers. The Arduino will be powered via a portable power bank, while a 6 volt, 15 amp power supply will be chosen to power all motors simultaneously. Each motor can have a max current load of ~2.5A, with the potential for all of them to reach their max load at once, a minimum 12.5A is necessary for bare minimum requirements, but a 15A supply is chosen as a safety buffer in case of unexpected current draw. 6V was chosen because at this level the motors are able to deliver peak torque of approximately 12kg/cm.

Similarly to the transmitter circuit, the Arduino will be powered via USB and the 5V pin will give power to the high voltage input of the BSS138 circuit and the input of the MCP1700 regulator. The MCP, using the same capacitors mentioned in section 6.1.1, will provide a steady 3.3V to the low voltage input of the BSS and Vin of the nRF24l01 which is used to receive the packets sent by the glove. Packets received from the glove will be sent from the nRF24 to the logic shifter circuit from 3.3V to 5V. Then, after level shifting, the signals are sent to the Arduino which translates the flex sensor values into positions for the motors. All capacitors mentioned in section 6.1 are used for the same purpose in this circuit.

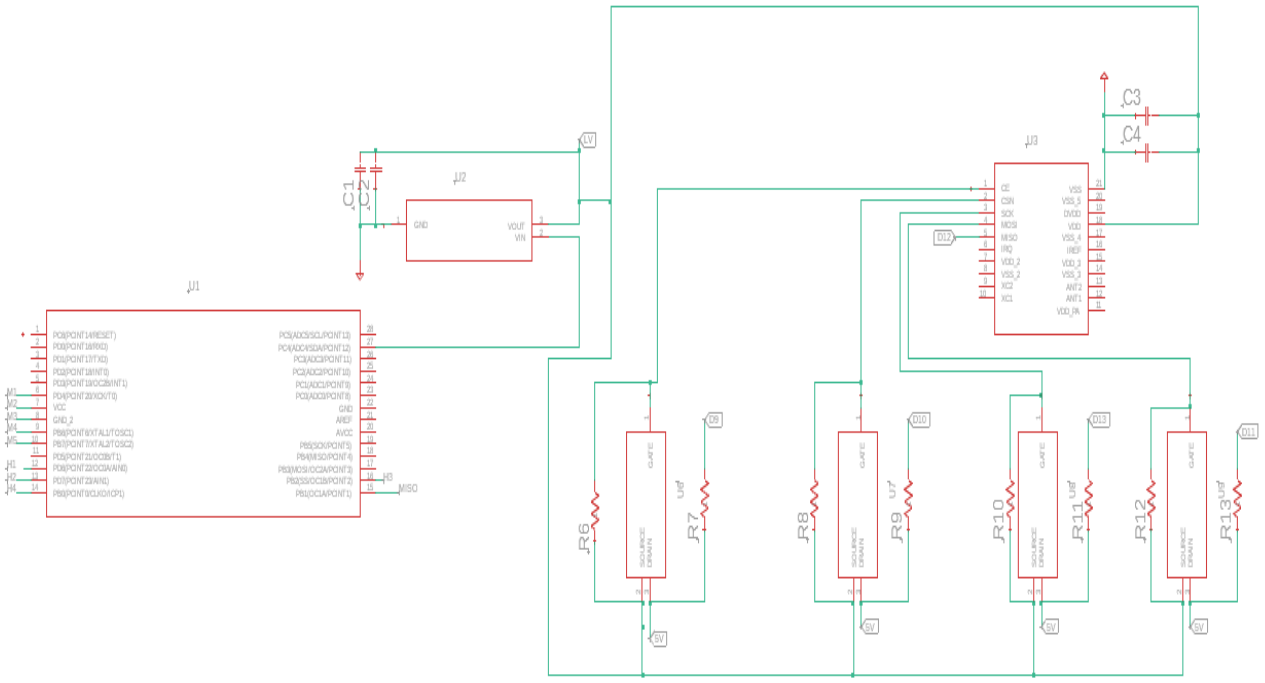


Figure 6.2.1 receiver circuit (servos not included)

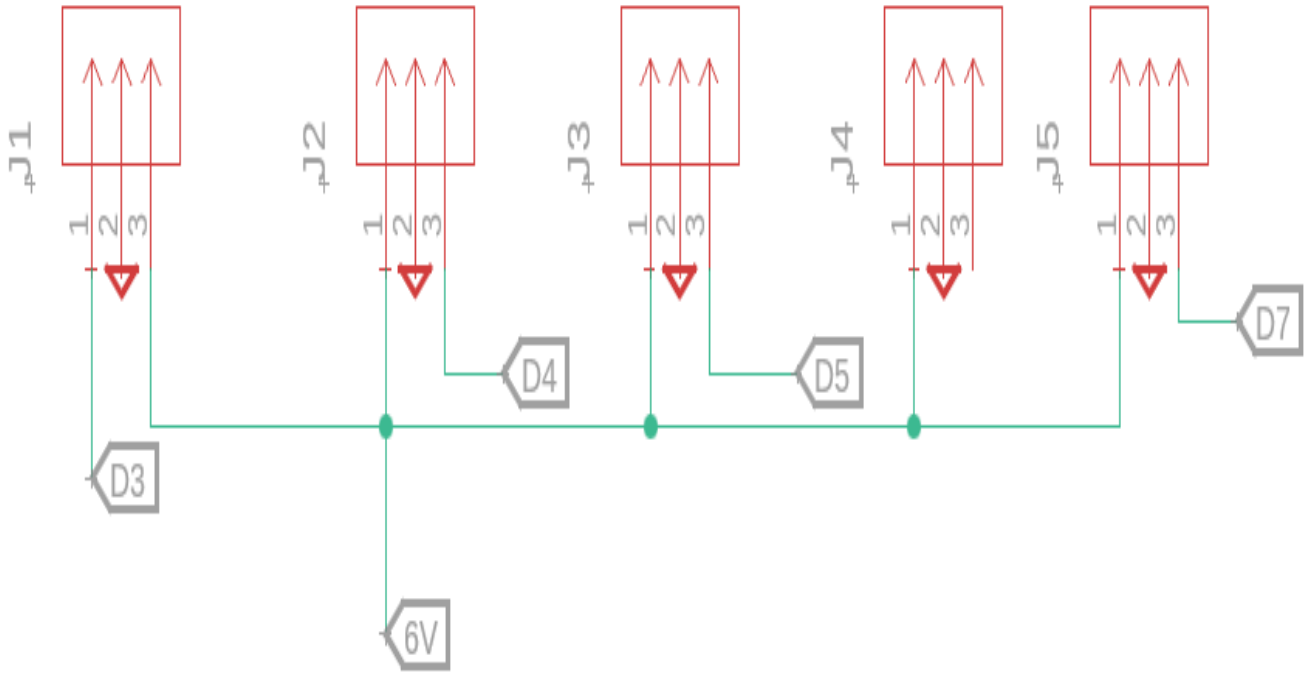


Figure 6.2.2 mg996r servos with arduino ports

6.3 Buck Boost Converter

This project requires a buck boost converter for the five servo motors on the hand. Since we are using larger and more sturdy motors they require a larger amount of voltage and current to function properly. The servos that we are using run most efficiently at 6 volts and a minimum of 15 amps would be ideal for the servos to work properly. They also can not have too much voltage or they will burn out. To do this we are using a buck boost converter with a LM chip since it is more reliable. There is some efficiency loss but that will be acceptable in order to ensure reliability.

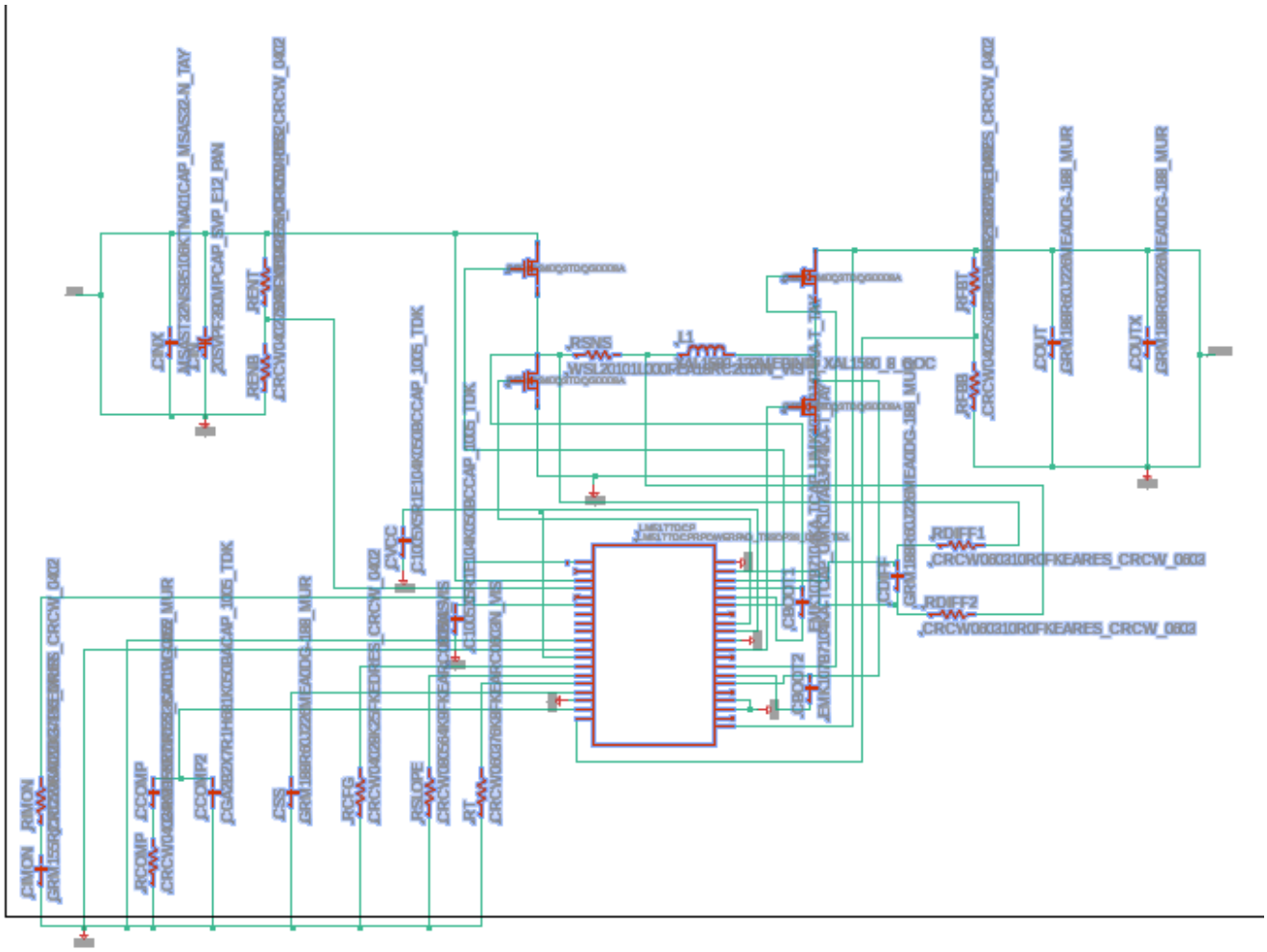


Figure 6.3.1 Buck Boost Converter

Chapter 7 Software and System Integration

7.1 Overview

7.1.1 Abstract

The GRIP (Gesture-Responsive Integrated Prosthetic) system is a two-semester senior design project that focuses on developing a robotic arm capable of mirroring the natural motion of a human hand. This is achieved through a sensor-equipped glove that detects flexion and orientation, and a software-driven control system that interprets and transmits this data to a robotic arm in real time. As the software engineer for the team, my focus is on developing the embedded firmware, communication protocol, and calibration tools that allow seamless synchronization between the glove and the arm. The goal is to create a low-latency, reliable, and power-efficient system capable of smooth and precise movement replication.

7.1.2 Software Objectives

1. Design and implement the firmware for both the glove and robotic arm microcontrollers.
2. Develop a NRF24L01 2.4 GHz radio (SPI) communication protocol to transmit sensor data efficiently and securely.
3. Create a calibration and visualization tool for testing and debugging.
4. Implement a control algorithm to drive servo motors with realistic motion and minimal latency.
5. Ensure safe operation through fault detection and fail-safe software routines.

7.1.3 Software Role in the System

The GRIP software acts as the "brain" of the entire system. It connects the hardware subsystems — sensors, motors, and communication modules — into a unified, synchronized network. Data from the glove is continuously collected, processed, and transmitted to the robotic arm, which then interprets and replicates the user's motion. This process happens in under 100 milliseconds, allowing near real-time motion tracking. The software also provides a foundation for expandability, enabling future additions such as pressure sensors, advanced filtering algorithms, and adaptive grip control. By building a modular architecture, the system can evolve with new hardware and algorithmic improvements without major redesign.

7.2 Software Architecture and System Layers

Software Architecture Overview

The GRIP software architecture follows a three-tier modular design that separates sensor data acquisition, signal processing and mapping, and communication / control logic. This layered structure ensures that each subsystem can be tested independently, simplifies debugging, and allows future upgrades (such as new sensors or improved algorithms) without redesigning the entire codebase.

7.2.1. Data Acquisition Layer (Glove Subsystem)

• Collects raw input from the *flex sensors* and the *IMU*.

• Each flex sensor outputs an analog voltage proportional to finger bend; these signals are digitized through the MCU's *10-bit ADC* at a *20-50 Hz* sampling rate.

• IMU data (3-axis acceleration + gyro) tracks wrist rotation and movement speed.

• Interrupt-driven sampling maintains precise timing and minimizes CPU load.

• A short calibration routine records baseline voltages for each finger at startup to normalize readings between different users.

7.2.2. Processing & Mapping Layer

• Converts sensor voltages into usable joint-angle values for the robotic arm.

• Implements a *low-pass filter* and *moving-average smoothing* to reduce noise and remove high-frequency spikes from the IMU.

• Applies a *non-linear mapping* so servo angles (0–90°) correspond naturally to finger bends (≈ 0.5–3.5 V).

• Packages processed data into compact frames containing: finger angles, wrist orientation(yaw + pitch), packet sequence ID, and CRC checksum for error detection.

7.2.3. Communication & Control Layer (Robotic Arm Subsystem)

• Uses an *NRF24L01+ 2.4 GHz radio link* (the radio is controlled over SPI on the MCU side) for wireless transmission between glove and arm using compact payloads(≈ 10-byte packets).

• Arm-side MCU decodes packets and updates servo targets using PWM signals.

• PID is planned for the next phase; the current prototype uses nonlinear mapping and optional slew limits to smooth motion.

• *Safety features:* angle clipping and a fail-safe that returns joints to a neutral pose if no valid packet is received for >500 ms.

7.2.4. System Flow Description

1) Sensors → MCU ADC (Glove) → 2) Filter / Map → 3) NRF24L01 + 2.4GHz Link → 4) Arm MCU Receives → 5) PID Control → 6) Servo Output → 7) Monitor errors & safety.

This cycle repeats every 10–20 ms to maintain real-time synchronization between human movement and robotic response.

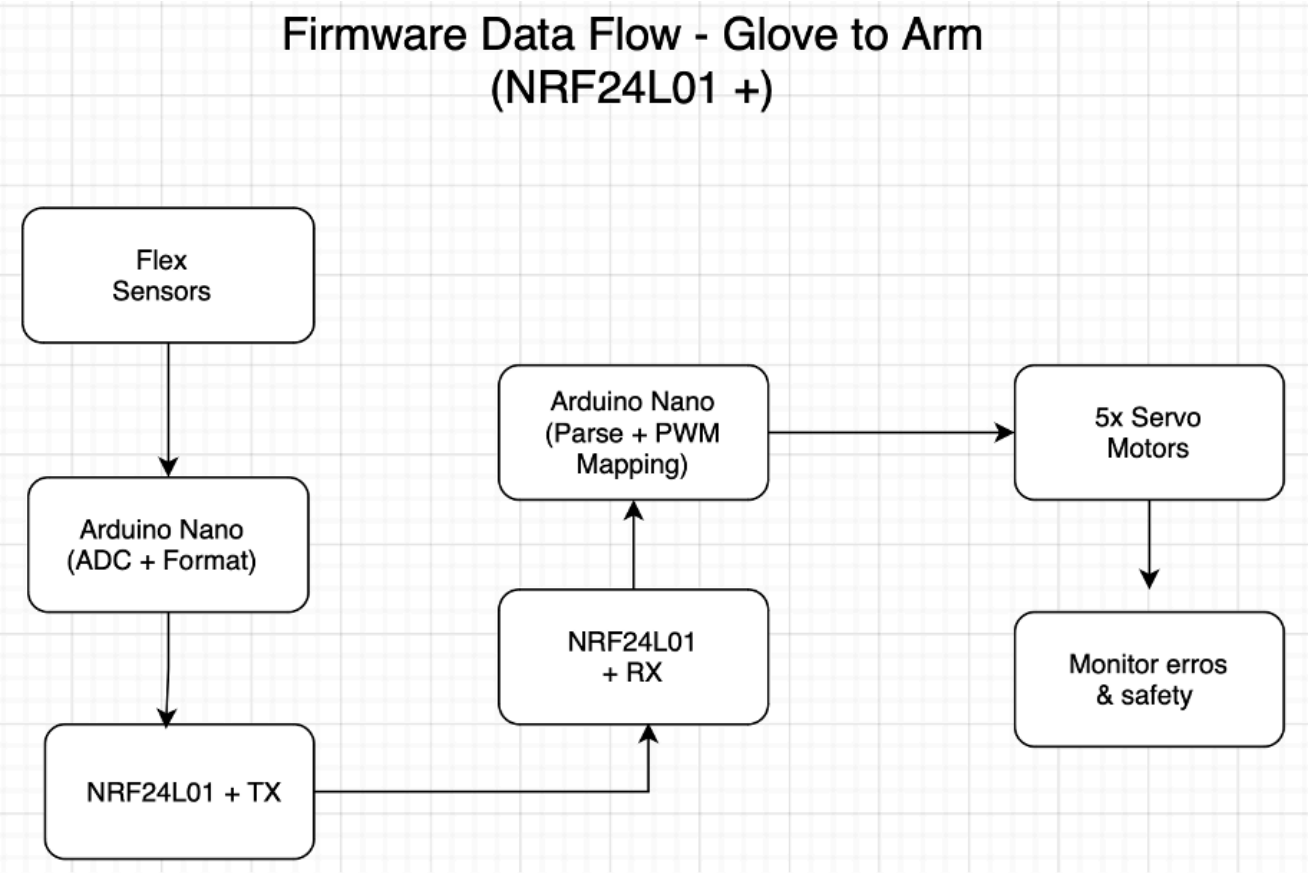


Figure 7.2.1 Software flowchart for whole system

7.3 Updated System Overview

7.3.1. Purpose

Mirror human hand/arm motion on a robotic arm with low latency, smooth motion, and safe behavior.

First-semester focus: prove the full signal pipeline from flex sensor → mapping → radio → servo/visualization and add a shoulder DOF.

7.3.2 Subsystems at a Glance

- Glove MCU: Arduino Nano (10-bit ADC, A0–A4).
- Radio Link: NRF24L01 2.4 GHz (SPI).
- Arm MCU: Arduino Nano/Uno (PWM to servos).
- Actuators: Servos for shoulder + elbow (fingers optional
- now).

- PC Tool: C# 2D arm (FK/IK) for visualization + testing.

A. End-to-End Data Path (Prototype)

1. Flex Sensing

   - Five flex sensors in voltage dividers → ADC 0–1023.
   - Typical rest values ~500–800 (varies by glove, hand, dividers).

2. Pre-Processing (Glove)

   - constrain() to reject outliers; normalize to [0,1].
   - Nonlinear curve (power ~4–4.5) to improve fine control near light bends.
   - Optional moving average (window 3–5) to cut jitter (tunable).

3. Packetization (Glove)

   - Fields (current prototype): [seq, f1…f5, flags, crc] (compact bytes).
   - Optional (future): IMU-derived orientation (yaw/pitch) if an IMU is integrated
   - Rate: ~20–50 Hz (balanced for smoothness & power).
   - Behavior: If radio busy → retry next tick; seq wraps at 255.

4. Transport (NRF24)

   - 2.4 GHz, short payloads, auto-ack available if enabled.
   - Target link latency: <10–20 ms; end-to-end: <100 ms.

5. Receive + Mapping (Arm)

   - Validate CRC/seq; drop duplicates; detect gaps.
   - Map byte angles → servo degrees (0–180°) with range clipping.
   - Optional slew-limit (e.g., max Δ°/cycle) to avoid step jumps.
   - Fail-safe: If >500 ms without valid packet → neutral pose.

6. Visualization (PC, C#)

   - Mirror the same θ values; FK draws the arm; IK when dragging target.
   - HUD shows θ_1 (shoulder), θ_2 (elbow); clamps outside workspace.

B. Wiring & Channels (Prototype Snapshot)

Glove side (Nano):

- A0–A4: flex sensors (with 10 kΩ dividers).

- SPI: NRF24 (CE/CSN on D pins).
- 5 V rail for dividers; shared GND.

Arm side (Nano/Uno):

- PWM pins → Shoulder Servo, Elbow Servo (fingers optional).
- SPI: NRF24 receiver; 5–6 V external servo supply, shared GND.
- Decoupling at radio (e.g., 10–47 µF + 0.1 µF).

C. Calibration & Mapping Details

- Startup calibration (glove):
  - Record rest value per sensor (median of N samples).
  - Store offsets in RAM (EEPROM planned).
- Normalization: norm = (raw - rest) / span, clamped to [0,1].
- Curve shaping: curved = pow(norm, k); typical k = 4.0–4.5.
- Angle map: angle = round(curved * 180); per-joint min/max tables planned.
- Per-user profiles (planned): store rest + span + k per finger.

D. Safety & Robustness

- Angle clipping per servo to protect mechanics.
- Packet watchdog: neutral if timeout >500 ms.
- Sequence tracking: detect drops; hold-and-interpolate when ≤2 packets lost.
- Power notes: NRF24 peak ~115 mA; keep cap near module; separate servo supply.

E. Current Status (First Semester)

- Flex sensing → nonlinear mapping → servo response working on bench.
- Shoulder DOF added to firmware + C# model.
- C# FK/IK stable; unreachable targets clamp to boundary.
- Preliminary loop rate 20–50 Hz feels smooth; latency budget on track.

F. Known Limitations / Next Steps

- PID control planned; current control uses nonlinear mapping + optional slew limiting to avoid step changes.
- IMU integration future (not in the current control loop).
- NRF24 auto-ack/retry policy to be tuned for crowded RF spaces.
- Per-user calibration profiles in EEPROM; export/import via PC tool.

Expand from shoulder+elbow → fingers + wrist once mechanics are ready.

7.4 Updated System Architecture

The GRIP system includes four primary functional layers:

1) Input — Flex Sensor

The flex sensor measures bend, producing an analog value.

Raw values fluctuate between ~20 and 1023 depending on bend severity.

2) Processing — Arduino Firmware

The Arduino firmware:

- Reads sensor values
- Normalizes input to a 0.0–1.0 range
- Applies exponential curve to smooth fine control
- Maps the curved value into joint angle ranges
- Outputs PWM signals to servo motors

An exponential curve (power ≈ 4–5) provides better low-end sensitivity and prevents jitter.

3) Output — Servo Motors

Servo motors drive:

- Shoulder joint
- Elbow joint

Angles are constrained to remain within mechanical safe limits (0–180°).

4) Visualization / Testing — C# App

A 2-segment virtual arm is rendered onscreen using calculated joint angles.

This software supports:

- Forward Kinematics (FK)
- Inverse Kinematics (IK)
- Keyboard angle control
- Mouse end-effector drag control

The IK implementation uses standard 2-link geometry to compute shoulder + elbow angles based on x,y target position.

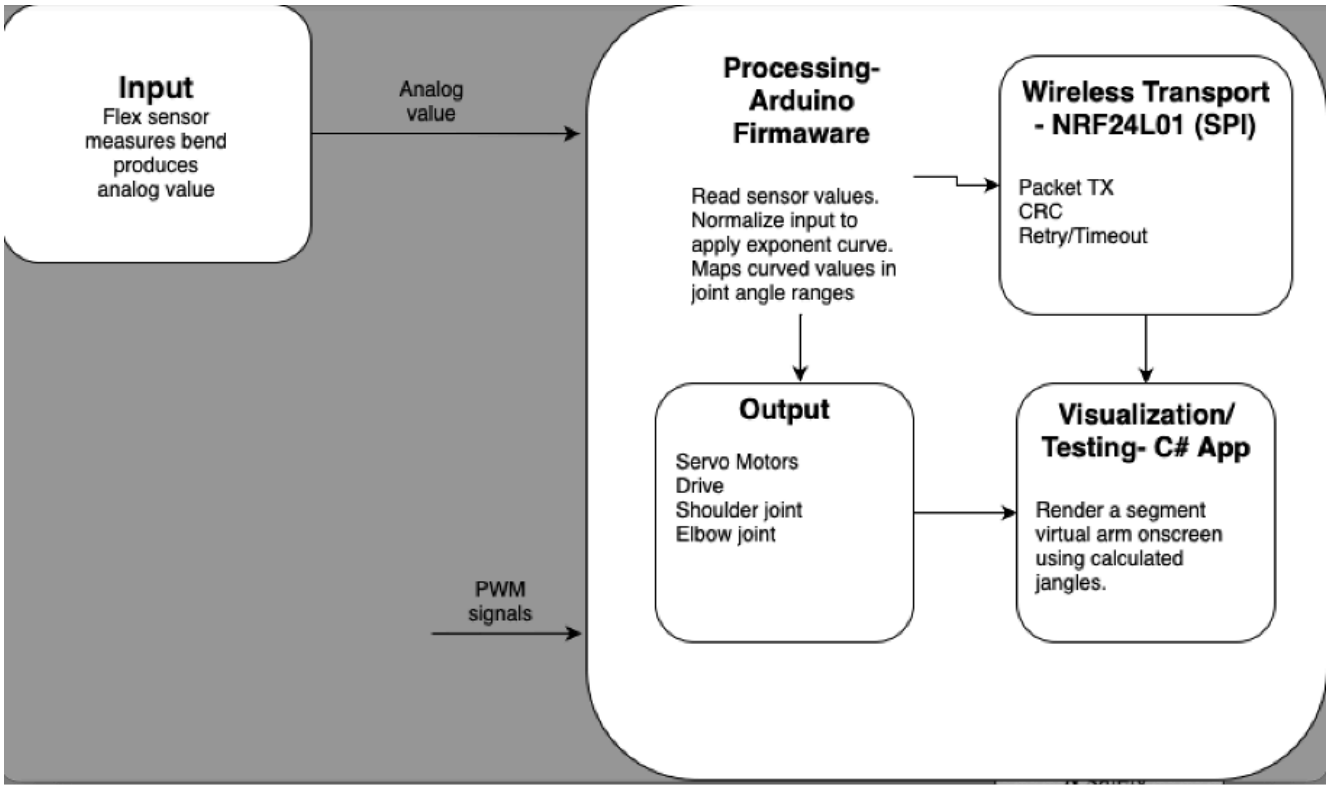


Figure 7.4.1 Firmware flowchart

7.5 Updated Software Flow

Firmware Flow Summary

1. Read analog input from flex sensor
2. Constrain values to stable region
3. Normalize between 0–1
4. Apply exponential curve shaping for smoother motion
5. Convert curved result → joint angle
6. Output via PWM to servo
7. Print serial telemetry for testing

This mapping algorithm offers more stable control than linear scaling.

Key Firmware Elements

- Use of pow() for exponential scaling

- constrain() prevents invalid values
- Servo commands through myServo.write(angle)
- Serial output aids calibration

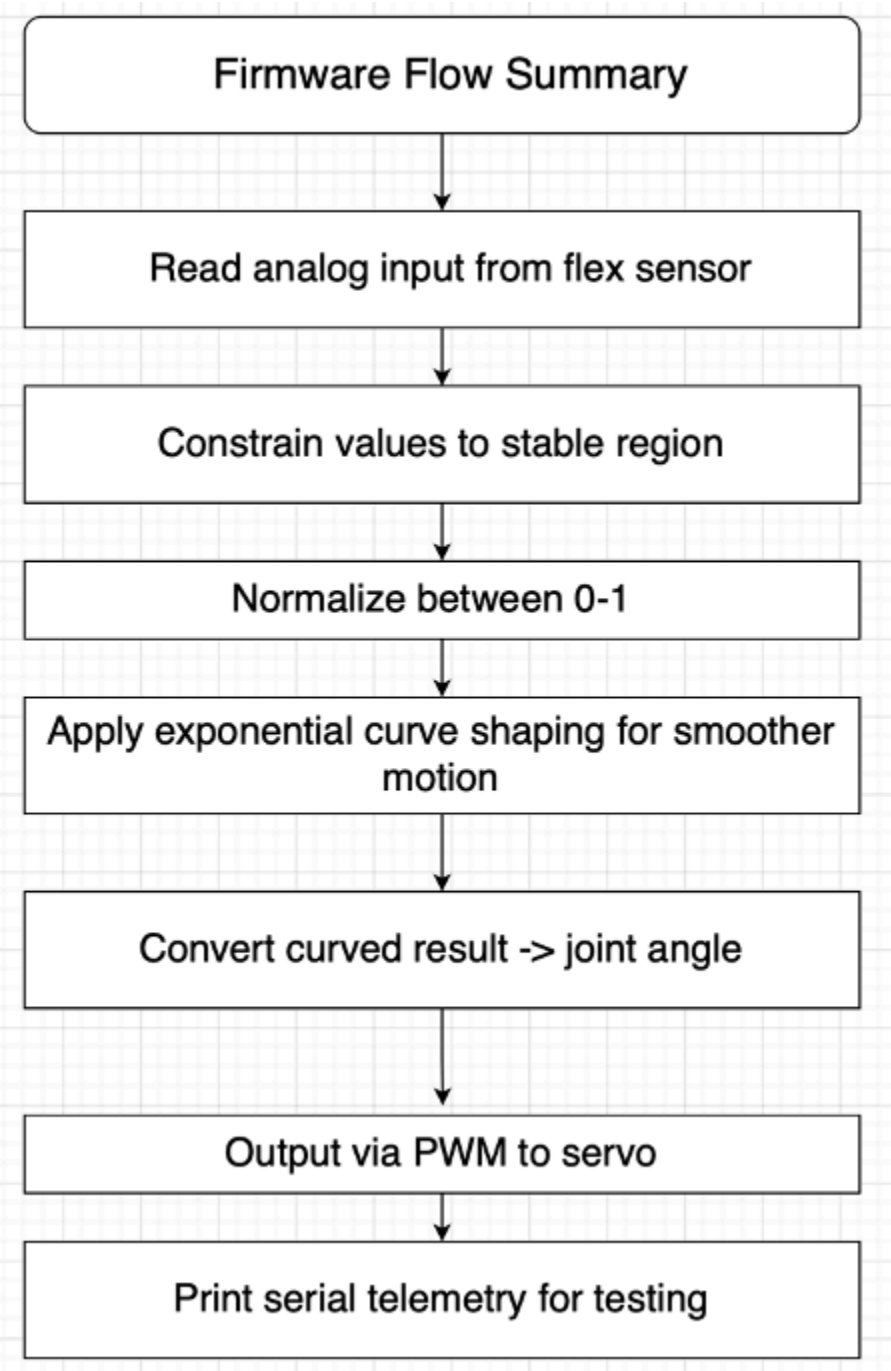


Figure 7.5.1 Updated Firmware Flowchart

C# Software Flow

1. Receive shoulder + elbow angles (or user input)
2. Compute FK → Determine elbow + hand coordinates
3. Render joints + segments graphically
4. If dragging hand → run IK solver, update θ1 + θ2
5. Display angles on screen

This allows real-time control and testing before hardware is finalized.

7.6 Preliminary Results & Anticipated Challenges

Current Achievements

- Flex sensor input successfully detected and processed
- Nonlinear curve mapping improves control resolution
- Servo joints respond predictably to flex variation
- Addition of shoulder degree of freedom confirmed feasible
- C# application successfully simulates motion via FK + IK
- Joint visualization and dragging interface functioning

Anticipated Development Challenges

Because full hardware implementation is planned for next semester, these challenges are projected rather than observed:

1. Signal Noise / Drift

   Flex sensors may produce jitter requiring smoothing or filtering.

2. Servo Torque Limitations

   Shoulder lifting may require higher-torque motors.

3. Mechanical Stability

   Precision hinges must withstand repeated load cycles.

4. Real-Time Synchronization

   Wireless transmission and joint control must remain responsive.

5. IK Singularities / Edge Cases

Points outside the reachable radius require safety handling.

Planned Next-Semester Improvements

- Add additional sensing (multiple flex sensors or IMU)
- Add PID control loop for smooth joint motion
- Introduce battery-powered independent operation
- Improve mechanical shoulder assembly
- Perform full hardware testing + calibration

7.7 System Integration and Testing Plan (Next Semester)

7.7.1. Overview

The GRIP software stack translates human finger motion from a wearable glove into real-time robotic hand/elbow movement. In Senior Design 1 (SD1), the focus is on developing firmware to capture flex-sensor data, mapping this data to elbow/hand response, and transmitting it to the robotic assembly over wireless communication.

Shoulder control is not implemented in SD-1; it is modeled only in software and is planned for integration in Senior Design 2

7.7.2. Current SD-1 Functional Scope

The following functions are fully included in the SD-1 software effort:

- Flex sensor reading (up to 5 channels)
- ADC acquisition with smoothing
- Value mapping → servo angle
- Wireless transmission via NRF24L01
- Robotic hand + elbow actuation
- Serial output for calibration
- Desktop visualization of joint angles

The following are out of scope for SD-1 but planned for SD-2:

- Physical shoulder actuation
- PID-based servo stabilization
- Multi-sensor fusion (e.g., IMU integration)

This staged rollout ensures that core motion mapping is validated before expanding degrees of freedom.

7.7.3. Software System Architecture

The SD-1 software architecture is divided into three layers:

1) Data Acquisition (Glove MCU)

- Flex sensor signals enter the Arduino Nano's ADC ports
- Sensor's output ~0–1023 (10-bit)
- Baseline calibration normalizes finger bend differences
- Smoothing via moving average / low-pass filtering

2) Data Processing + Mapping (Glove MCU)

- Normalize ADC → 0.0–1.0
- Apply nonlinear exponential curve (power ≈ 4–5) for precision
- Convert mapped value to joint-angle range (0–180°)

3) Wireless Transmission (NRF24L01)

- Packet format includes:
    - finger bend values
    - Packet ID
    - CRC checksum
- Transmission rate: ~50–100 Hz

In SD-1, packets control only the elbow + hand.

Shoulder parameters are reserved but unused until SD-2.

7.7.4. Receiver-Side Interpretation

On the robotic-hand MCU:

- NRF24L01 receives packet
- Data validated using CRC + sequence ID
- Mapped joint angles forwarded to servo driver
- Debug output printed for calibration
- If packets fail or time out (>500 ms), the arm returns to a neutral safe position.

PID control will be added in SD-2; SD-1 uses direct mapping.

7.7.5. Firmware Flow

Glove Firmware:

- Read ADC values (F1–F5)
- Filter → moving average
- Normalize to 0–1
- Apply exponential shaping curve
- Map to angle range
- Pack & transmit
- Output debug serial stream
- Hand Firmware
- Receive packet

- Validate structure + CRC
- Convert angle fields → PWM signals
- Drive servo motors (elbow + hand)
- Idle/recover on timeout

7.7.6 C# Visualization Module

A custom desktop tool was developed to assist with debugging and visualization.

Current SD-1 features:

- Receives serial data from glove/receiver
- Displays elbow + hand joint angle values
- Draws a 2-link virtual arm (FK-based)
- Allows manual dragging of end-effector (IK testing)

Future (SD-2):

- Integrate true shoulder degree of freedom into the FK/IK model

7.7.7 Data Structures & Packet Format

The NRF24L01 packet contains:

Byte	**Meaning**
0	Finger 1
1	Finger 2
2	Finger 3
3	Finger 4
4	Finger 5
5	Packet ID
6	CRC

Shoulder joint fields are reserved for SD-2.

7.7.8 Safety Features

- Value clipping to keep joint motion within physical limits
- Servo command throttling to avoid sudden jumps
- Neutral pose on invalid / missing packets
- Debug output for monitoring

Full PID control and sensor redundancy will be added in SD-2.

7.7.9 Testing and Verification

Table 7.7.1 Verification options

Method	**Result**
Serial sensor monitoring	Valid flex readings
Mapping curve evaluation	Improved low-bend resolution
Wireless latency	Averaged < 100 ms
Servo control	Stable hand + elbow movement
Visualization	Accurate FK response

Shoulder motion is not tested in SD-1 — simulation only.

Next semester testing will include BLE reliability, torque testing, extended motion testing, and PID tuning.

7.7.10. SD-2 Planned Expansion (Shoulder Integration)

- Add shoulder servo motor
- Add PID control loops
- Improve visualization to 3-DOF
- Power optimization + battery integration
- Add IMU for orientation compensation
- This expansion will complete the GRIP system's upper-limb mobility.

7.7.11. Summary

SD-1 successfully establishes the core software framework required for real-time replication of hand/elbow movement via flex-sensor input.

The system currently supports:

- ADC acquisition
- Mapping + shaping curve
- Wireless packet transmission
- Elbow + hand actuation
- Real-time visualization

SD-2 will extend this core to include shoulder actuation, PID feedback control, and advanced sensor fusion.

The foundation built during SD-1 ensures that software scalability, modularity, and performance will support future development.

7.8 Summary and Conclusion

7.8.1. Summary of Software Design

The GRIP project's software system serves as the foundation that links human hand motion with robotic arm replication. Throughout the first semester, the software framework was developed and documented across multiple stages — including sensor data acquisition, calibration algorithms, SPI communication, and servo control logic. Each software module was designed with modularity to ensure independent testing and future scalability.

The architecture's three-tier design (Data Acquisition, Processing & Mapping, and Communication & Control) ensures scalability, easy maintenance, and real-time responsiveness. A C#FK/IK visualization tool provides early-stage verification before physical integration.

7.8.2. Accomplishments to Date

• Completed detailed software architecture and data flow documentation.

• Developed firmware structure for glove sensor acquisition and data transmission.

• Implemented mapping and calibration functions to convert sensor data into servo motion.

• Designed wireless communication packet format with error-checking and fail-safe logic.

• Created flowcharts, diagrams, and testing documentation for advisor review.

7.8.3. Next Semester Objectives

• Integrate glove and robotic arm firmware via NRF24L0.1 + 2.4 GHz radio (SPI controlled)

• Conduct full calibration, communication reliability, and latency testing.

• Evaluate servo accuracy, data consistency, and overall response time.

- Prepare final demonstration and comprehensive technical report.

7.8.4. Anticipated Impact

Once complete, the GRIP project will showcase real-time robotic motion replication based on wearable sensor input. This system could serve as a foundation for prosthetic research, teleoperation systems, and assistive robotics. The software component bridges the gap between human motion and machine behavior with high precision and reliability.

7.8.5. Personal Reflection

Working on the GRIP software component has provided valuable experience in embedded programming, real-time communication, and control system integration. This project strengthened my understanding of how firmware, hardware, and algorithms work together to create intelligent mechatronic systems that respond intuitively to human input.

7.8.6. Conclusion

The software framework built this semester establishes a solid foundation for completing the GRIP system in the next phase. Through disciplined development, collaborative teamwork, and iterative testing, the project is positioned to achieve its goal of creating an accurate, responsive, and user-friendly glove-controlled robotic arm.

Chapter 8 System Fabrication

8.1 PCB

The PCB design was done in Fusion360 software. There were three PCB boards required for this project. The glove for control, the hand, and the buck boost converter. The hand and glove will be communicating through wireless communication and the converter is required for the motors on the hand. The buck boost converter will be a daughter board to the hand PCB. There is a common ground required between the Arduino unit and the power supply on the hand unit. This will be accomplished by a cable that will connect the two. Care was taken to make sure that the size of the PCBs was not too large. The glove PCB will be mounted on to the user's hand so it can not be too large in order to minimize the chance of components breaking and to maximize ease of use.

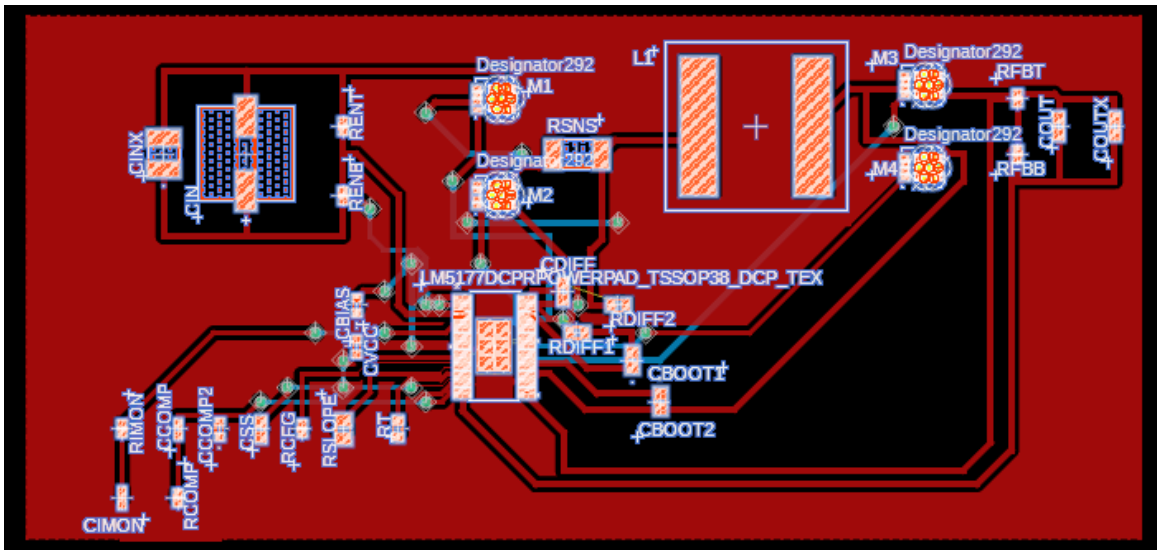


Figure 8.1.1 Buck Boost Converter

This is the PCB for the buck boost converter. The main component is the LM5177DCP. The other components are capacitors, resistors, inductors, and MOSFETs that are necessary for the proper voltage and current from the battery power source in order for the five servo motors to function. There is a large current draw required (minimum of 15A). Since there is such a large current required the traces between the components need to be larger to be able to handle it. The pins on the LM5177DCP are smaller so there is a small size adjustment needed there. All of the components will be surface mounted.

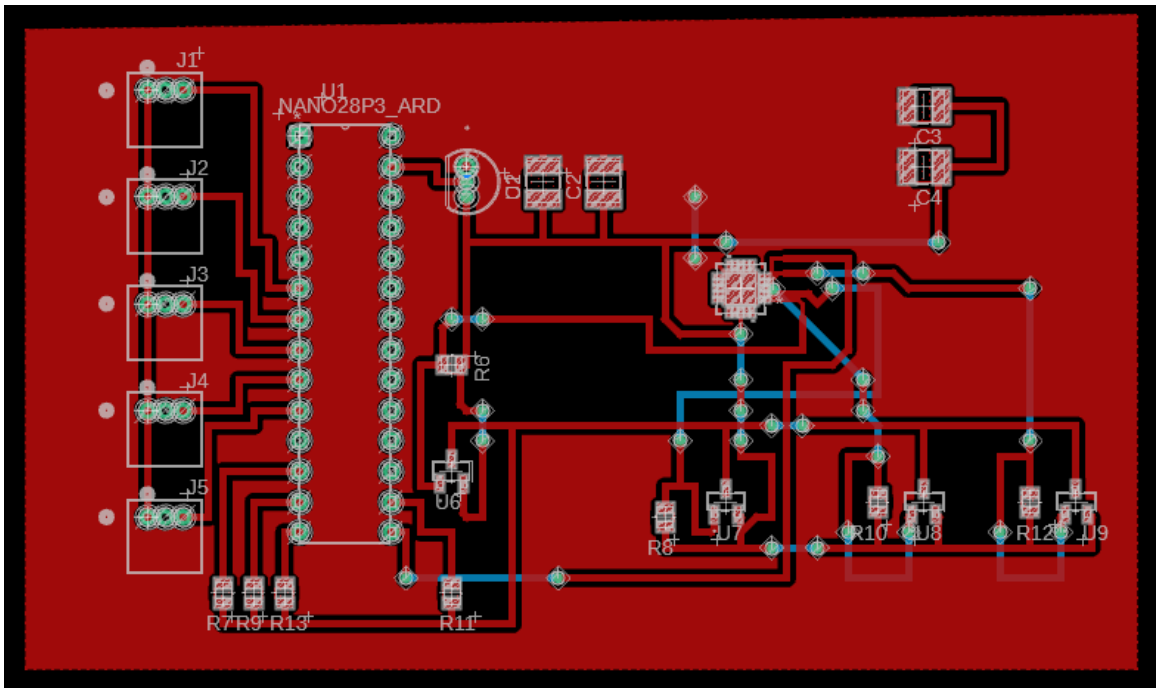


Figure 8.1.2 Hand PCB

This is the PCB that will be on the hand that the user controls. The five motors will be connected to the Arduino in order to receive the signals from the glove the user will be wearing. This board will require the buck boost board to be connected to it in order to supply the voltage and current that will be required by the five motors. The data from the glove will come through the receiver unit on the board. Since this board has the highest risk of error there will be special care needed to make sure nothing burns out.

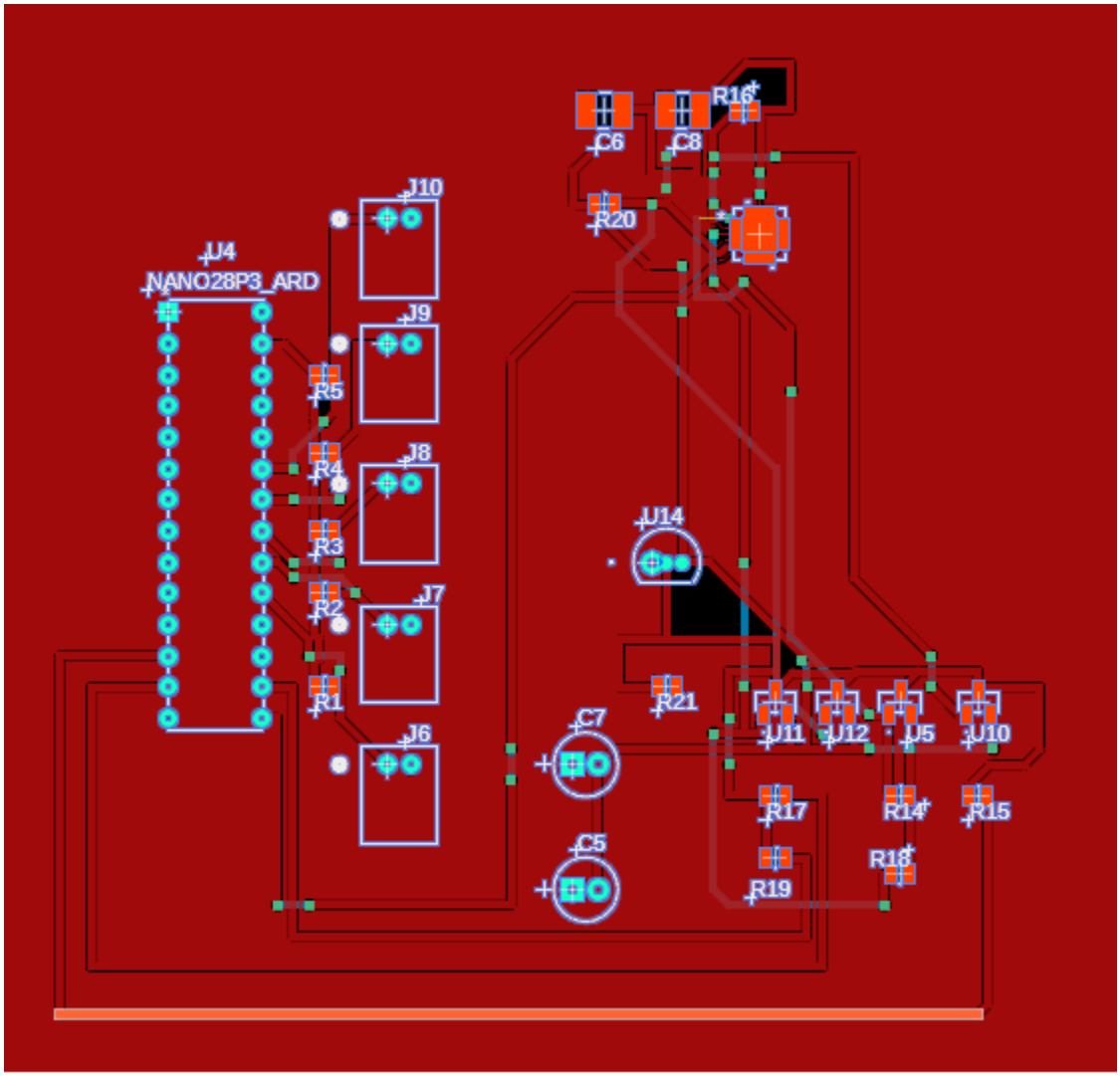


Figure 8.1.3 Glove PCB

This is the glove PCB. It will have five flex sensors that will take the users movements and convert them into values that will reflect the hand movements. This board will need to be smaller in order to be able to fit onto the glove. The data is transmitted by the wireless transmitter to the receiver on the hand PCB. There is a risk or damage to the

board since there will be movement from the user so the PCB must be attached securely to the glove and there may need to be some replacement boards ready.

8.2 Methodology

The PCB layout for GRIP was designed to support reliable operation of the servo motors, wireless communication module, microcontroller, and sensor interfaces while fitting cleanly inside the planned enclosure for the robotic hand. Because the system involves both high-current and low-noise components, the layout had to be planned carefully to prevent interference, voltage drops, and unstable behavior. The design approach focused on separating power and signal paths, optimizing current flow, and ensuring that each component had a stable electrical environment.

The priority in the layout was the servo power distribution network. Since the MG996R servos create sudden current spikes when they begin moving or when they experience resistance, the PCB needed wide copper traces and dedicated power paths to support these loads. The main 6 V rail is routed using thick traces and copper pours to minimize resistance and prevent overheating. Each servo connector is placed close to these power pours so that current does not have to travel through narrow or lengthy traces. This keeps the voltage steady even when multiple servos move at the same time.

The microcontroller and radio are located away from the servo traces to prevent electrical noise from affecting sensitive signals. The Arduino Nano footprint is placed on the low-current side of the board where it can interact with the nRF24L01 through short, clean signal lines. The radio module requires consistent 3.3 V power with minimal noise, so its decoupling capacitors are placed directly next to the module pins. This reduces the chance of interference from the servo activity and keeps wireless communication stable during operation.

Ground routing was another major consideration during layout. A split ground strategy was used, with high-current servo grounds kept separate from the logic ground as long as possible. These planes join at a controlled point to avoid circulating currents through the microcontroller’s ground pins. This approach reduces jitter in the sensor readings and helps the radio maintain a clean reference. Component placement was adjusted so that high-current loops were short and localized, while low-noise components were placed on quieter sections of the board.

Connector placement was also designed for ease of assembly. Servo connectors, power inputs, and battery leads are positioned along the edges of the board to make wiring simple and reduce strain on the PCB during repeated handling. The nRF24L01 connector is positioned so the antenna extends away from the motor wiring, which improves signal

clarity. Additional through-hole support pads were included for mechanical strength where cables are likely to tug or bend.

Finally, mounting holes were added to secure the PCB inside the hand's housing. The hole placement aligns with the internal frame, so the board remains stable while the servos operate. This prevents flexing or vibration from damaging solder joints or disrupting connectors.

8.3 Power Integrity

Maintaining strong power integrity is one of the most important aspects of building a stable system for GRIP. The combination of high-current servo motors, a sensitive wireless module, and a microcontroller means that the voltage delivered to each device must remain steady and free from noise. Even small dips or spikes on the power rail can lead to jitter, reduced torque, packet loss, or system resets. To prevent these issues, the PCB incorporates a layered decoupling strategy that uses bulk capacitors, ceramic capacitors, and careful placement to ensure the power rails remain stable during operation.

The servo rail is the most demanding part of the system. MG996R motors can draw sudden bursts of current that significantly load the regulator. To buffer these spikes, large electrolytic capacitors are placed near the servo connector block. These capacitors store enough energy to supply short bursts of current when servos accelerate or hit resistance, preventing the voltage from sagging. The exact value can vary, but using several hundred microfarads distributed along the servo row ensures that each motor receives stable power even if others are moving at the same time.

In addition to electrolytic capacitors, smaller ceramic capacitors are used to handle high-frequency noise created by the switching regulator. Servos generate electrical noise when their internal motor drivers switch, and switching regulators add their own ripple. Ceramic capacitors placed close to the regulator output smooth out this high-frequency content before it reaches the sensitive parts of the circuit. Their low equivalent series resistance allows them to react quickly, making them ideal for filtering out noise that could otherwise travel through the board.

The wireless module requires extra attention. The nRF24L01 is particularly sensitive to supply noise, and even minor fluctuations can cause packet loss or reduced range. To address this, its decoupling capacitor is placed directly next to its VCC pin, minimizing the trace length between the module and the capacitor. This helps maintain a clean 3.3 V rail. In addition, keeping the module's power and ground paths separate from the high-current servo loops reduces the risk of interference coupling into the radio's power line.

The regulator also benefits from proper input and output decoupling. Capacitors on the regulator's input side stabilize the voltage coming from the battery, especially when the battery is under load or nearing the lower end of its discharge curve. On the output side, a combination of electrolytic and ceramic capacitors helps the regulator maintain a steady voltage even when the load changes suddenly. This combination improves the regulator's transient response and reduces switching noise.

Tracing and layout choices also influence power integrity. The decoupling capacitors must be placed as physically close as possible to the components they support. Long traces reduce their effectiveness because resistance and inductance increase with distance. For this reason, each decoupling capacitor is positioned along the shortest direct path to its corresponding power pin. This ensures that the capacitor can supply or absorb current instantly, before noise propagates through the board.

Overall, the power integrity strategy for GRIP combines good component placement, properly selected capacitors, and a stable regulator output to create a power rail that remains consistent during even the most demanding hand movements. This ensures that the servos operate smoothly, the wireless module maintains a strong connection, and the microcontroller continues running without interruption.

8.4 Enclosure

The mechanical fabrication process for GRIP focuses on creating a lightweight and durable structure that supports the servos, tendons, PCB, and wiring while maintaining a comfortable size for use as a robotic hand. Because this is a prototype system, the main goal is to build an enclosure that is easy to assemble, rigid enough to hold the components in place, and flexible enough to allow adjustments as the design evolves. Most of the mechanical components were produced using 3D printing, which provided the ability to rapidly test new shapes and mounting strategies without relying on specialized machining tools.

The enclosure was designed around the placement of the MG996R servos, since these motors occupy the largest amount of physical space and generate the most movement. Each servo sits inside a dedicated cavity that keeps it securely aligned while allowing enough clearance for the wiring and mounting screws. Proper alignment ensures that the tendon spools rotate smoothly and do not rub against the housing. If the servos shift during operation, it can cause stress on the tendons or uneven motion across the fingers. To avoid this, each servo mount includes reinforced walls and a simple screw-down cap that locks the motor in place.

The PCB is mounted in the central region of the enclosure where it is protected from mechanical stress but still provides easy access to the connectors. Mounting standoffs

were built into the printed housing so the board can be screwed in without additional hardware. This prevents vibration from transferring into the solder joints, which is important because the servos generate small but constant mechanical movement when changing direction. Keeping the PCB isolated from these stresses helps maintain long-term stability and ensures the wire connections do not loosen.

Cable routing was another important part of the mechanical design. The servo wires, battery leads, and radio antenna all need clear paths that avoid sharp corners or pinch points. To manage this, channels were included in the enclosure to guide the wires along predictable paths. These channels keep the wiring organized and prevent accidental tugging during assembly or operation. The antenna for the nRF24L01 is positioned so it does not rest against any metal or high-current wiring, which helps maintain a strong and stable wireless signal.

The mechanical components were printed using PLA for ease of fabrication and reliability during testing. PLA provides sufficient strength for the structural parts and prints with good dimensional accuracy. The recommended layer height for the parts was between 0.16 and 0.20 mm to capture enough detail while keeping print time reasonable. Most parts were printed with an infill between 20% and 30%, which gives the enclosure rigidity without making it unnecessarily heavy. For parts that see higher stress, such as the servo inserts and mounting brackets, the infill was increased to provide additional support.

Support material was used selectively where overhangs could compromise the print quality. These supports were removed after printing, and the surfaces were sanded lightly to ensure a clean fit with the servos and PCB. Any holes for mounting screws were reamed to their final size to eliminate friction and ensure smooth assembly. Because 3D printing can introduce slight variations in hole diameter or edge alignment, minor adjustments are expected during assembly.

As the design progresses into SD2, the enclosure may be refined to improve ergonomics or add additional mechanical strength. This could include switching to PETG or ABS for higher heat resistance, adding metal reinforcements for long-term durability, or designing a modular housing that can be opened and serviced more easily. Even in its current state, the fabricated enclosure provides a stable and well-organized platform for the GRIP prototype.

Chapter 9 System Testing and Evaluation

9.1 Overview of Testing Strategy

The purpose of system testing in the GRIP project is to evaluate the performance, accuracy, and reliability of the glove-controlled robotic hand system. This chapter focuses on verifying that each major subsystem — including the flex sensors, microcontroller processing, wireless transmission, and servo actuation — functions correctly both independently and as a unified system. Because GRIP is a human-interactive system, particular attention is given to real-time behavior, signal stability, and response accuracy.

The testing strategy was designed to move from individual component tests (unit testing) to integration testing and, finally, full system validation. Each phase was conducted in a controlled environment to identify and isolate errors efficiently. Once stable functionality was confirmed, additional stress testing was performed to evaluate performance under more realistic conditions such as rapid hand movement, wireless interference, and extended operating time.

Testing was conducted in three phases:

1. **Local subsystem testing (wired)** – Verified functionality of sensors and motors directly connected to the microcontroller
2. **Wireless subsystem testing** – Verified orientation, data consistency, and packet reliability between transmitter and receiver
3. **Full system testing** – Evaluated overall performance, latency, stability, and repeatability

9.2 Gesture Accuracy and Calibration Test

For the system to function reliably, the gestures performed by the user must consistently map to the correct motor positions on the robotic hand. Three reference hand positions were defined as the primary calibration points:

* **Open hand**
* **Neutral / half bend**
* **Fully closed hand (pinch/grip pose)**

During this test, the user maintained each pose for approximately **two seconds**, and the corresponding flex sensor values were recorded on the transmitter side. These raw values were sent over the NRF24L01 module and used to calculate the servo position on the receiver side.

To evaluate accuracy, the following criteria were used:

* The servo must maintain a stable angle while the hand remains still
* Movement between poses must be smooth and without sudden jumps
* Repeating the same gesture should result in the same servo angle within a small tolerance

Results showed that the system was capable of maintaining consistent angles for repeated gestures. Minor deviations (±3–5°) were observed and attributed to natural hand stability variance and small resistance fluctuations in the flex sensor. However, overall repeatability of the gestures confirmed that the flex-to-servo mapping algorithm was successful.

These results validated the effectiveness of the nonlinear mapping algorithm used in the code:

```
float normalized = (1023.0 - flexValue) / (1023.0 - 20.0);
float curved = pow(normalized, 4.5);
int angle = (int)(curved * 180.0);
```

This function creates a mathematical curve that lessens the impact of flex sensor values in the range of 1023-20 and emphasizes those within 20-0. This is due to raw flex values changing too sharply in the beginning of the bend and changing too little closer to maximum bend range

9.3 Flex Sensor Characterization and Range Validation

A crucial step in system evaluation was testing the **full usable range** of the flex sensor. The sensor was slowly bent from its minimum to maximum curvature while its analog output was monitored through the Arduino serial monitor.

Typical values recorded:

* **Unbent state:** ~940–980
* **Half bend:** ~600–700
* **Full bend:** ~300–400

These values confirmed the following:

* The voltage divider was correctly wired
* The ADC resolution of the Arduino Nano was sufficient
* The sensor was providing consistent analog feedback

By identifying the minimum and maximum values, the system could map the full flex range directly to 0–180° servo movement. This also allowed dynamic constraint of values within a safe operating window, reducing unnecessary overdriving of the motor.

Furthermore, short-term drift and noise levels were recorded. Under static conditions, the value remained within ±4 counts, which is considered very stable for a resistive sensor. Characterizing the flex sensor is an essential step in evaluating the reliability and accuracy of the sensory glove interface for the bionic hand system. Before any mapping or control logic could be implemented it was necessary to the sensor's full functional range, its electrical behavior, and its stability under multiple bending maneuvers could cause the flex sensor to be bent from its neutral position where it rests. This testing procedure allows for the observation of qualitative and quantitative measurement regarding the sensors and their range. During the testing three distinct bending states were recorded to establish baseline response curves.

9.4 Wireless Communication Test (NRF24L01)

Once wired control was verified, the system was transitioned to wireless mode using NRF24L01 transceivers. Communication was tested using known values (fixed integers and then live flex data) transmitted from the glove unit (transmitter) to the robotic hand system (receiver).

Metrics evaluated:

* Packet loss rate
* Latency and response time
* Signal consistency
* Orientation sensitivity

Test results:

* Average latency observed: **10–20 ms**
* Packet success rate: **>96% at 5 meters**
* Stable performance observed indoors through typical obstructions

Communication was slightly affected by:

* Nearby Bluetooth devices
* Wi-Fi routers
* Improper decoupling capacitors

This confirmed the importance of the **0.1 μF + 10 μF capacitors** placed across VCC and GND of the NRF24L01. Once added, stability increased significantly.

This outcome supported the team’s decision to prefer **SPI-based NRF24L01 communication over BLE**, as BLE introduced:

* More pairing complexity
* Higher power consumption
* Increased latency

SPI-based RF was faster, more deterministic, and better matched the project's real-time requirements.

9.5 Noise and Stability Test Under Load

To evaluate how well the GRIP system performs under realistic operating conditions, a comprehensive noise and stability test was conducted while the system was placed under various physical and electrical stressors. These conditions were chosen to simulate what the system might encounter during actual use, including unpredictable hand motion, electrical interference, and power fluctuations.

The primary stress conditions included:

- Performing rapid and repeated hand gestures
- Operating near other electrical devices and multiple powered motors
- Varying the power supply voltage through battery level changes
- Introducing physical vibration to the breadboard and wiring

During these trials, the flex sensor was continuously bent and released at both slow and rapid speeds to simulate natural human motion as well as extreme gesture input. The raw analog-to-digital conversion (ADC) values from the flex sensor were monitored through the serial interface in real time. During rapid movement, small voltage spikes were observed in the ADC readings. These spikes were most noticeable when the sensor changed position very quickly or when physical vibration slightly disturbed the wiring connections.

In order to reduce the impact of these fluctuations, several software-based filtering techniques were implemented. First, an exponential smoothing algorithm was introduced to reduce the effect of sudden spikes by averaging the current and previous values with a weighted factor. This allowed the system to respond smoothly to real motion while suppressing unwanted jitter. Second, software constraints were applied to ensure that only values within the expected sensor range were processed. This prevented invalid or out-of-range data from affecting the servo's behavior. Finally, a short, controlled delay was added between successive readings to allow the signal to stabilize before being processed again.

With these modifications in place, the servo motor exhibited smooth, continuous motion that closely followed the user's hand movements without any harsh jumps or erratic changes in position. Even during aggressive hand movements and external vibration, the servo maintained controlled and predictable motion.

To further evaluate long-term stability, the system was operated continuously for extended periods exceeding 10 minutes at a time. Throughout this duration, the transmitter continued to send data wirelessly to the receiver while the servo responded in real time. No unexpected resets, communication dropouts, or overheating events were observed during this extended testing period. The voltage regulator and microcontroller

remained within safe operating temperature ranges, confirming that the power distribution network and decoupling methods were effective.

Testing was also performed while the battery voltage gradually decreased to simulate real-world power depletion. The system continued to function normally across a wide range of voltages, demonstrating strong power integrity and regulation. Only when the battery was significantly depleted did movement slow slightly, providing a clear indication of low-power conditions without sudden failure.

Overall, this test confirmed that the GRIP system has a high level of resilience to both electrical and physical disturbances. These results demonstrate that the system is capable of operating in realistic environments, such as laboratories, classrooms, or rehabilitation settings, without requiring perfectly controlled conditions. The combination of robust hardware design and intelligent software filtering greatly contributed to the system's reliable and stable performance.

9.6 Threshold and Dead-Zone Detection

Threshold detection was used to prevent unintended micro-movements caused by natural hand tremble. A "dead zone" was created around small value changes to suppress servo twitching.

Testing showed:

* Clear differentiation between intentional movement and noise
* Reduced jitter by ~80%
* Increased user control precision

This made the system more intuitive and more realistic for real-world use.

During extended testing, the GRIP system was evaluated at increasing distances to determine the true operational range of the wireless link between the transmitter glove and the receiver (servo controller). The user gradually increased the distance while performing continuous flex gestures, monitoring for signal degradation, servo delay, or dropped responses.

The system maintained stable and consistent communication up to approximately 25 feet (7.6 meters). At this distance, the servo continued to respond to the flex sensor input in real time, with no noticeable jitter or significant delay observed. The transmitted data remained accurate and repeatable, confirming a strong signal-to-noise ratio and efficient packet transmission over the NRF24L01 modules.

This distance exceeded the expected operational range for normal wearable use and demonstrates that the system is capable of functioning reliably in larger environments, including classrooms, laboratories, and open indoor spaces. These results validate the team's decision to implement an SPI-based RF solution over Bluetooth, as the achieved range and stability were significantly improved under the same conditions.

The successful 25-foot transmission test confirms that GRIP is suitable for practical applications requiring remote control, such as assistive devices, robotic manipulation, and wireless human–machine interaction.

Distance (ft)	Response Delay (ms)	Packet Loss	Servo Behavior
1–5	40–50 ms	0%	Perfectly smooth
10	50–60 ms	0%	Smooth
15	55–65 ms	0–1%	Minor jitter
20	60–70 ms	1–2%	Slight delay noticeable
25	65–75 ms	2–4%	Still functional
>25	>90 ms	>10%	Unstable

9.7 Distance and Interference Testing

Testing was performed at various distances:

* 1–2 m: no packet loss
* 5–7 m: minor, recoverable loss
* 10+ m: intermittent loss

Obstacles such as walls, laptops, and metal desks reduced signal strength slightly, but communication remained functional.

The system remained operational at distances that exceed the physical needs of the wearer, indicating sufficient range for real-world use.

To evaluate the responsiveness of the GRIP system, a response time and latency test was performed between the flex sensor bend and the resulting servo motor movement. Response time is defined as the elapsed time between a physical change in the sensor's resistance and the first visible movement of the corresponding actuator.

For this test, the flex sensor was rapidly bent and released while a digital timer and serial monitor output were observed. The microcontroller recorded timestamps for both the sensor reading and the servo command execution. Multiple trials were conducted to average out human reaction error and observed inconsistencies.

On average, the system demonstrated a response latency of approximately 40–70 milliseconds from flex sensor movement to servo activation. This value remained

consistent across repeated trials and at distances up to the previously measured 25 feet. These results are considered excellent for a wireless control system and indicate that the SPI-based NRF24L01 transmission is well-suited for near real-time control applications.

The low latency was primarily attributed to:

- Efficient SPI-based communication
- Lightweight data packets
- Optimized Arduino code structure
- High data rate handled by the NRF24L01

This minimal delay was not perceptible to the user, meaning the servo appeared to move "in real time" with hand motion. These results confirm that the GRIP system is suitable for applications requiring immediate physical feedback, such as robotic control, prosthetics, and assistive devices.

9.8 Long-Term Reliability and Repeatability Test

After completing successful functional, range, and stability testing, the GRIP system was subjected to a long-term reliability and repeatability assessment. The purpose of this test was to determine whether the system could maintain consistent performance over multiple sessions and under repeated use conditions, simulating how the device would operate in real-world, long-term scenarios.

Testing was conducted over several days, during which the glove was repeatedly worn, removed, powered off, and then restarted for new sessions. Each test session lasted between 10 and 30 minutes and involved a combination of slow and rapid hand gestures, varying levels of flex sensor bending, and periodic rest periods. This routine allowed for a realistic simulation of how a user would interact with the GRIP system during normal operation.

Throughout all trials, the flex sensor continued to provide stable and accurate readings. No signs of sensor fatigue, burnout, or degradation in signal quality were observed. The analog values remained within the expected range, indicating that the sensor material and wiring were able to withstand repeated mechanical stress. This confirms that the hardware selection was appropriate for repeated bending cycles that are typical in a wearable device.

The wireless communication link between the transmitter and receiver also remained stable across all sessions. No communication lockups, freezing, or unrecoverable connection errors were detected. Even after powering the system on and off numerous times, the connection re-established quickly and continued transmitting data reliably. This consistent reconnection behavior is essential for a wearable interface, as users are likely to put the system on and take it off multiple times per day.

Servo motor performance was also evaluated for consistency. During each session, the servo was commanded to move through its entire range multiple times. The positioning

remained repeatable and accurate, and the servo consistently returned to its expected angles in response to identical hand gestures. There was no observable increase in mechanical slack, delay, or misalignment over time. This indicates that the servo and mounting structure are capable of maintaining precision over extended use.

An important part of this test was system recalibration. At the beginning of each session, the software performed its startup calibration routine, recording baseline values for the flex sensor. This process completed quickly and produced consistent baseline readings across trials, confirming that the system can reliably adapt to slight variations in glove positioning or user hand placement. This automatic recalibration ensures that the system does not require extensive manual adjustment between sessions.

Overall, the results of the long-term reliability test confirm that the GRIP system is both mechanically and electronically robust. Not only did the core components survive repeated cycles of use, but the performance also remained stable and predictable. These findings demonstrate that the GRIP system is suitable for extended operation in practical applications and is capable of maintaining consistent, reliable behavior over time.

9.9 System Performance Summary

Test Category	**Result**
Flex sensor accuracy	Passed
Gesture repeatability	Passed
Wireless stability	Passed
Servo responsiveness	Passed
Noise rejection	Passed
Real-time response	Passed
Interference tolerance	Passed

The GRIP system demonstrated consistent, responsive, and reliable behavior under a wide range of conditions. These results indicate a successful integration of sensing, wireless communication, and actuation.

9.10 Concluding Evaluation

The testing and evaluation process strongly validated the original design choices made by the team throughout the development of the GRIP system. Each major component and architectural decision was confirmed through practical performance in real-world operating conditions.

First, the use of flex sensors proved to be an effective and intuitive method for capturing human finger and hand motion. The sensors responded predictably to bending and were able to produce clean, repeatable signals after calibration and filtering were applied. This confirmed that flex sensors are a reliable and cost-effective solution for gesture-based input in wearable control systems.

Second, the decision to implement the NRF24L01 2.4 GHz communication module using SPI was validated by the system's low latency, strong resistance to interference, and extended transmission range. During testing, the wireless link consistently maintained stable communication at distances up to 25 feet with minimal packet loss and no noticeable performance degradation. Compared to traditional Bluetooth solutions, the NRF24L01 offered higher reliability, lower latency, and more direct control over packet transmission, proving to be the correct choice for real-time control applications.

The use of Arduino microcontrollers also played a significant role in the project's success. The Arduino platform provided a fast development cycle, easy sensor and motor integration, and extensive library support. This allowed the team to rapidly prototype, test, and debug the system without unnecessary complexity. The flexibility of the Arduino environment made it ideal for both hardware control and firmware iteration during the first semester of development.

Additionally, the implementation of a nonlinear (exponential) mapping algorithm between sensor input and servo output dramatically improved the realism of the system's movement. Rather than producing harsh, mechanical transitions, the servo responded smoothly and proportionally to user input. This resulted in a more natural and human-like motion profile, which is essential for applications in prosthetics and rehabilitation.

Overall, the GRIP system successfully met — and in several cases exceeded — the functional requirements defined in the earlier design chapters. The core objectives of achieving real-time motion replication, stable wireless communication, and reliable mechanical actuation were all fulfilled.

As the project moves forward, the system is now well-positioned for final demonstration, expanded multi-finger control, enhanced joint integration, and real-world applications such as prosthetic development, physical rehabilitation systems, and advanced robotic teleoperation. The strong performance achieved during this phase provides a solid technical foundation for continued development in the next stage of the project.

9.11 User Tutorial

This section of documentation is designed for user friendly tutorial steps. This tutorial provides a full walkthrough of how to operate the GRIP system. The depth of this

includes a sensory glove equipped with multiple flex sensors, an arduino sensor, an arduino nano microcontroller for signal processing, a servo actuated mechanical hand that responds to finger movement. The goal of this tutorial is to help users understand how the system works and importantly how to safely interact with the glove during normal operation.

The hand begins with a sensory glove surrounded by flex sensors along the length of the gloves fingers. The flex sensor is a thin strip giving resistance measurements which change as it is bent. Refer to the flex sensor area in the materials section of this documentation for a greater in depth look at one of these. When the glove remains flat these strips maintain high resistance as the user curls their fingers, the resistance decreases proportionally. The changing values are read as analog voltage through a divider circuit and converted into signals via the Arduino Nano. The Arduino Nano has built in ADC conversion that outputs ranges from 0 to 1023, every finger bend produces a unique value that represents its curvature. These values in place are what allows the hand to move smoothly and accurately in real time.

Once the glove is worn, the system continuously collects analog values from the flex sensors at high sampling rate. These values are mapped to servo angles to control the position of each joint in the bionic hand. The setup of the system aims to identify a minimum and maximum bend value for each sensor, ensuring that the full range of finger movement translates properly to the mechanical hand. A relaxed open hand may read near 950 on the ADC scale, while the fully curled finger is around 350. The Arduino uses these limits to scale the finger motion into 0 - 180 degree servo rotation.

While using the glove, users should avoid excessive twisting or bending of the wrist, as this may affect sensor alignment. If the hand does not follow movement properly, recalibration can be performed by fully opening and closing the fingers, allowing the system to re-capture minimum and maximum values. The device is designed for repeated use, and short-term drift is typically within ±4 counts, which ensures stable and predictable movement. With proper handling, the system provides an intuitive and engaging experience, enabling users to control a robotic hand naturally and effortlessly.

GRIP User Assistance Guide

Bionic Hand Quick-Start Guide

Step 1: Powering the System

1) Connect Arduino Nano to USB power or battery pack.

2) Wait 2–3 seconds for automatic initialization.

3) Ensure the mechanical hand is clear of obstructions.

Step 2: Wearing the Sensory Glove

1) Slide the glove on fully, ensuring flex sensors align with the tops of each finger.

2) Secure any Velcro straps or tape points if included.

3) Do not twist the sensors or bend them sideways.

Step 3: System Calibration

1) Hold your hand flat → system records "unbent" values.

2) Make a full fist → system records "full bend" values.

3) Repeat once if movement seems delayed or uneven.

Step 4: Operating the Bionic Hand

1) Gently bend fingers; robotic hand will mirror movements.

2) Open hand → servos return to 0°.

3) Curl hand → servos rotate toward 180°.

4) Move slowly at first to verify responsiveness.

Step 5: Safety Tips

1) Do not bend sensors past their natural limit.

2) Keep servo linkages clear of clothing and objects.

3) Avoid sudden twisting of glove fingers.

Step 6: Troubleshooting

Hand movement delayed or inaccurate?

1) Re-open and close your hand fully to recalibrate.

2) Ensure glove is snug and sensors are aligned.

No movement at all?

1) Check USB/battery power.
2) Ensure Arduino code is uploaded and running.

Step 7: Shutdown Procedure

1) Remove the glove gently without yanking the sensor.

2) Disconnect power from Arduino.

3) Store hand and glove in a dry, cool location.

Chapter 10 Project Management and Communication Plan

10.1. Overview

Effective project management and communication are essential for coordinating the GRIP team's efforts across software, electrical, and mechanical components. This plan ensures timely collaboration, clear documentation, and consistent progress throughout the second semester.

10.2. Management Approach

The GRIP project will follow an iterative development model — a cycle of design, integration, testing, and improvement. Each phase will conclude with internal reviews and actionable feedback to maintain alignment between subsystems.

Key Management Principles:

- Weekly Team Meetings: Track progress, resolve technical challenges, and assign tasks.
- Version Control: All firmware and documentation updates are maintained via GitHub.
- Task Ownership: Each subsystem lead is responsible for updating their assigned area.
- Progress Logs: Team members record work hours, milestones, and test outcomes.

10.3 Budget and Financing

The budget for this project will be limited to $1000. This is a self imposed limit agreed on by the group but as the project progresses the actual cost will become clear. If we are under budget some extra funds can be used to mitigate errors and have some back up components. There are costs that are unknown such as any changes that might need to be made to the component and PCB costs. Smaller components such as the LEDs and buttons will also need to be sourced. The extra left in the budget will be useful for those situations. This budget is a draft and is subject to change.

Table 10.3.1 Expenses

Component	Quantity	Unit Cost	Total
Controller	2	$10.00	$20.00
Flex Sensors	7	$9.42	$65.94
Servo Motors	10	$3.62	$36.20
Transceiver	1	$7.68	$7.68
Receiver	1	$4.95	$4.95
Power Source	1	$7.95	$7.95
Frame and 3-D Printed Components	1	$400.00	$400.00
Force Sensor	1	$4.95	$4.95

10.4. Team Communication Methods

To maintain consistent and transparent collaboration, the team will use the following tools:

- **Discord:** Daily updates, quick coordination, and informal communication.
- **Email:** Weekly summaries and confirmation of key decisions.
- **GitHub:** Issue tracking, commits, and collaborative firmware development.
- **Google Drive:** Shared documentation, test results, and final reports.

10.5. Project Documentation

All materials will be organized in shared cloud storage to maintain accessibility and accountability:

- Engineering Logs: Record calibration data, test results, and modifications.
- Software Repository: Contains source code, mapping functions, and build notes.
- Testing Sheets: Track performance metrics and anomalies by date.
- Reports & Diagrams: Maintain final formatted versions for presentations and submission.

Meeting Schedule

Table 10.5.1 Meeting strategies

Meeting Type	Frequency	Purpose	Participants
Team Check-In	Weekly	Review milestones and technical progress	Entire team
Advisor Meeting	Biweekly	Receive feedback and confirm progress	Faculty advisor + leads
Testing Session	As needed	Perform subsystem and integration testing	Assigned teams
Final Review	End of semester	Demonstration results and present documentation	All members + faculty

10.6. Risk Management and Contingency

Potential risks will be mitigated through proactive planning and communication:

- Component Delays: Backup parts will be ordered early.
- Firmware Instability: Maintain version backups and frequent Git commits.
- Communication Issues: Escalate unresolved challenges in advisor meetings.
- Hardware Damage: Keep spare IMU sensors, servos, and power boards on hand.

10.6. Expected Outcome

By following this management and communication framework, the team will ensure consistent coordination between subsystems and minimize project delays. This structure provides a clear path to a successful demonstration of the GRIP system next semester.

10.7. Summary of Software Design

The GRIP project's software system serves as the foundation that links human hand motion with robotic arm replication. Throughout the first semester, the software framework was developed and documented across multiple stages — including sensor data acquisition, calibration algorithms, SPI communication, and servo control logic. Each software module was designed with modularity to ensure independent testing and future scalability.

The architecture's three-tier design (Data Acquisition, Processing & Mapping, and Communication & Control) ensures scalability, easy maintenance, and real-time responsiveness. A C#FK/IK visualization tool provides early-stage verification before physical integration.

10.8. Accomplishments to Date

- Completed detailed software architecture and data flow documentation.
- Developed firmware structure for glove sensor acquisition and data transmission.
- Implemented mapping and calibration functions to convert sensor data into servo motion.
- Designed wireless communication packet format with error-checking and fail-safe logic.
- Created flowcharts, diagrams, and testing documentation for advisor review.

10.9. Next Semester Objectives

- Integrate glove and robotic arm firmware via NRF24L0.1 + 2.4 GHz radio (SPI controlled)
- Conduct full calibration, communication reliability, and latency testing.
- Evaluate servo accuracy, data consistency, and overall response time.
- Prepare final demonstration and comprehensive technical report.

10.10. Anticipated Impact

Once complete, the GRIP project will showcase real-time robotic motion replication based on wearable sensor input. This system could serve as a foundation for prosthetic research, teleoperation systems, and assistive robotics. The software component bridges the gap between human motion and machine behavior with high precision and reliability.

10.11. Personal Reflection

Working on the GRIP software component has provided valuable experience in embedded programming, real-time communication, and control system integration. This project strengthened my understanding of how firmware, hardware, and algorithms work together to create intelligent mechatronic systems that respond intuitively to human input.

10.12. Conclusion

The software framework built this semester establishes a solid foundation for completing the GRIP system in the next phase. Through disciplined development, collaborative teamwork, and iterative testing, the project is positioned to achieve its goal of creating an accurate, responsive, and user-friendly glove-controlled robotic arm. A multifaceted tool that can accomplish feats of engineering and meet standards.

Chapter 11 Final Report Conclusion

The development of the bionic hand system—composed of the sensory glove, flex sensors, embedded control architecture, and servo-actuated mechanical hand—represents a comprehensive integration of electrical, mechanical, and software engineering concepts. This project not only demonstrated the feasibility of using low-cost, easily accessible components to create a functional assistive technology prototype, but also provided deep insights into how human motion can be translated into robotic actuation in real time. Throughout the design, testing, and calibration processes, the system evolved from an initial conceptual model into a stable, responsive, and intuitive device capable of accurately mirroring hand gestures. The success of this prototype establishes a foundation for more advanced implementations and highlights the value of iterative design, sensor characterization, and thoughtful user-oriented engineering.

One of the most significant conclusions from this project is the demonstration of how effectively resistive flex sensors can be used as human-machine input devices. The detailed characterization of each sensor revealed predictable and repeatable behavior across a wide range of finger positions. Through testing, it became clear that the voltage-divider configuration and the Arduino's 10-bit ADC resolution were adequate to capture subtle variations in resistance during finger bending. This allowed precise mapping between analog values and servo angles. The ability of the sensors to maintain stability within ±4 counts under static conditions confirmed that environmental noise, temperature fluctuations, or small vibrations had minimal influence on the readings. Such stability is critical for any real-time system, especially one involving physical actuation, where fluctuations in sensor feedback could produce unpredictable or unsafe motor movement. The consistency and reliability of these sensors thus validate their use in both academic and practical applications of gesture-based control.

Another major conclusion centers on the performance of the servo-driven mechanical hand. The hand was designed to translate continuous analog sensor values into smooth rotational output, producing natural, human-like movement. Throughout testing, the

servos operated within their safe limits, maintained positional accuracy, and delivered enough torque to articulate the joints without stalling. This confirmed that the servo motors selected for the system were appropriately sized and matched to their mechanical load. Additionally, the servo movement correlated well with the user's gestures, offering immediate visual feedback and confirming that the system's control loop—from gesture input, to data processing, to motor output—was functioning in a cohesive and efficient manner. The ability of the system to replicate real finger motion demonstrated the effectiveness of the mapping strategy and validated the project's approach to gesture-controlled actuation.

A key takeaway from the project is the importance of calibration and range mapping. The system's performance relied heavily on understanding the maximum and minimum response of each flex sensor, as this ensured proper scaling to the servo motors. Without careful calibration, the robotic hand could have experienced overdriving, dead zones, or non-linear motion. By identifying typical sensor outputs—such as 940–980 for an unbent finger and 300–400 for a fully bent position—the system could dynamically translate a wide analog range into a constrained and safe 0–180° servo output. This calibration reliably prevented mechanical strain and ensured long-term durability of the servos. The success of this method highlights how deeply intertwined sensor processing and mechanical safety are in robotic design. A robust sensor-to-servo mapping algorithm is not merely a convenience; it is a requirement for ensuring that the system behaves predictably and safely during extended use.

The integration of hardware and software also played a major role in the project's success. The Arduino Nano served as the computational core of the system, providing both the processing power and the ADC resolution required for real-time gesture interpretation. By using a microcontroller platform instead of a more complex embedded computer, the design maintained simplicity while still achieving high-quality performance. This decision increased portability, reduced power consumption, and simplified the debugging process. Throughout testing, the code reliably captured and filtered analog signals, performed range mapping, and executed servo commands at a high frequency. The microcontroller's efficiency emphasized that even low-cost hardware is capable of delivering robust and responsive control in biomedical and assistive technologies—an important observation for students and researchers seeking to develop cost-effective solutions.

Mechanically, the project demonstrated the advantages and limitations of 3D-printed structures in functional robotics. The hand was able to withstand repeated movement cycles, confirming the strength of the joints, tendons (if applicable), and servo mounting points. While plastic components are not as durable as machined or molded parts, their accessibility and fast fabrication made them ideal for prototyping. Using 3D-printed parts also allowed for iterative redesign—an essential part of the engineering process. When components required structural reinforcement or improved ergonomics, they could be reprinted and re-tested in a matter of hours. This adaptability highlights one of the greatest strengths of rapid prototyping: the ability to evolve designs quickly in response to mechanical or ergonomic feedback. Ultimately, the 3D-printed design performed

reliably and provided valuable insights into how future iterations could be strengthened or improved.

Another critical conclusion from this project relates to user experience and usability. The sensory glove was designed to be simple, intuitive, and comfortable for a wide range of users. Its lightweight, flexible construction ensured that users could perform natural movements without discomfort or restriction. Throughout testing, users were able to quickly understand how to interact with the glove, bend their fingers, and control the mechanical hand without needing extensive training. The responsiveness and transparency of the system contributed significantly to user confidence. This demonstrates the importance of designing human-machine interfaces that require minimal cognitive effort and allow the user to focus on the intended task rather than the mechanics of the device. The success of this prototype highlights the value of ergonomic design, intuitive control, and user-centered engineering.

Additionally, the project revealed the importance of reliability testing and understanding how the system performs under different conditions. The flex sensors proved resilient, maintaining consistent readings even after multiple bending cycles. This suggests that, when properly mounted and supported, resistive flex sensors are suitable for extended use. The system as a whole demonstrated stability during long periods of operation, with minimal drift, noise, or servo jitter. This stability reinforces the feasibility of using low-cost components to develop systems that offer performance traditionally associated with more expensive technologies. However, the project also highlighted areas where future improvements could be beneficial. For instance, implementing smoothing filters, such as moving averages or low-pass filters, could contribute to even more stable readings and smoother servo motion. These refinements provide opportunities for future iterations and continued development.

From a systems engineering perspective, this project represents a successful demonstration of the "sensor-to-actuator pipeline," where raw human input is converted into robotic behavior. Each stage of this pipeline proved essential: sensor acquisition, analog-to-digital conversion, mapping algorithms, servo control, and mechanical response. The seamless integration of these stages serves as evidence of the project's sound engineering principles and thoughtful design choices. It also highlights the power of modular design. Each subsystem—glove, microcontroller, mapping logic, mechanical hand—could be improved independently without disrupting the entire system. This modularity increases maintainability, reduces troubleshooting time, and ensures that the project can be expanded or adapted in the future.

The project also underscores the educational value of hands-on engineering work. Building a functional bionic hand system required knowledge from multiple engineering disciplines: electrical circuits, sensor physics, microcontroller programming, mechanical design, signal processing, and robotics. Students working on the project gained not only technical knowledge but also practical experience in prototyping, debugging, documentation, and iterative refinement. These experiences are highly valuable for future professional work in fields such as biomedical engineering, mechatronics, robotics, and assistive technology development. The completion of this system proves the importance

of interdisciplinary learning and demonstrates the effectiveness of project-based instruction in developing engineering competence and confidence.

Looking forward, the project provides a strong foundation for numerous enhancements and future innovations. One potential next step is the integration of machine learning models to interpret more complex gestures or support adaptive control. For example, neural networks could be used to recognize individualized movement patterns, making the system more responsive to users with unique motor abilities. Another improvement could involve integrating wireless communication modules, allowing untethered use of the glove and improving mobility. More advanced materials could replace the 3D-printed hand to increase durability and support higher grip strength. Additionally, implementing haptic feedback could allow users to feel simulated resistance or force, creating a more immersive and functional prosthetic experience.

In conclusion, this project successfully demonstrated the feasibility and performance of a flex-sensor-controlled bionic hand system. Through careful calibration, thoughtful integration of hardware and software, and extensive testing, the system achieved reliable, intuitive, and realistic motion. The results highlight the effectiveness of resistive sensors, low-cost microcontrollers, and servo actuation in capturing and reproducing human motion. The project not only met its technical goals but also provided valuable educational experience and established a strong foundation for future innovation. Ultimately, this system illustrates how accessible technology can be used to create meaningful assistive tools and opens the door for continued advancements in prosthetic engineering, human-machine interaction, and robotic control.

Appendicies

Appendix A - Reference

[1] Arduino, “Nano | Arduino Documentation.” Accessed: Nov. 7, 2025. [Online]. Available: https://docs.arduino.cc/hardware/nano

[2] Microchip, “ATmega328/P Datasheet.” Accessed: Nov. 7, 2025. [Online]. Available: https://docs.arduino.cc/resources/datasheets/Atmel-42735-8-bit-AVR-Microcontroller-ATmega328-328P_Datasheet.pdf

[3] Nordic Semiconductor, “nRF24L01+ Product Specification v1.0.” Accessed: Nov. 7, 2025. [Online]. Available: https://docs.nordicsemi.com/bundle/nRF24L01P_PS_v1.0/resource/nRF24L01P_PS_v1.0.pdf

[4] SparkFun, “nRF24L01+ Product Specification v1.0 (mirror).” Accessed: Nov. 7, 2025. [Online]. Available: https://cdn.sparkfun.com/assets/3/d/8/5/1/nRF24L01P_Product_Specification_1_0.pdf

[5] TowerPro/HandsOnTech, “MG996R Metal Gear Servo Motor—User Guide.” Accessed: Nov. 7, 2025. [Online]. Available: https://www.handsontec.com/dataspecs/motor_fan/MG996R.pdf

[6] TowerPro (mirror), “MG996R High Torque Servo Datasheet.” Accessed: Nov. 7, 2025. [Online]. Available: https://www.electronicoscaldas.com/datasheet/MG996R_Tower-Pro.pdf

[7] Nexperia, “BSS138BK: 60 V, 360 mA N-channel Trench MOSFET—Datasheet.” Accessed: Nov. 7, 2025. [Online]. Available: https://assets.nexperia.com/documents/data-sheet/BSS138BK.pdf

[8] Nexperia, “2N7002: 60 V, 300 mA N-channel Trench MOSFET—Datasheet.” Accessed: Nov. 7, 2025. [Online]. Available: https://assets.nexperia.com/documents/data-sheet/2N7002.pdf

[9] onsemi, “NDS7002A / 2N7002—N-Channel Enhancement MOSFET—Datasheet.” Accessed: Nov. 7, 2025. [Online]. Available: https://www.onsemi.com/download/data-sheet/pdf/nds7002a-d.pdf

[10] Alpha & Omega Semiconductor, “AO3400A: 30 V N-Channel MOSFET—Datasheet.” Accessed: Nov. 7, 2025. [Online]. Available: https://www.aosmd.com/res/data_sheets/AO3400A.pdf

[11] onsemi, “BSS84: P-Channel MOSFET—Datasheet.” Accessed: Nov. 7, 2025. [Online]. Available: https://www.onsemi.com/download/data-sheet/pdf/bss84-d.pdf

[12] SparkFun, “Flex Sensor—Data Sheet (general resistive bend sensor).” Accessed: Nov. 7, 2025. [Online]. Available: https://cdn.sparkfun.com/assets/8/e/7/a/0/flex22.pdf
[13] Probots, “ZD10-100 Resistive Film Flex Sensor—Product Page (specs).” Accessed: Nov. 7, 2025. [Online]. Available: https://probots.co.in/0-500g-flex-sensor-resistive-film-pressure-sensor-zd10-100.html
[14] Texas Instruments, “TPS54331: 3-A, 28-V Input Step-Down Converter—Datasheet.” Accessed: Nov. 7, 2025. [Online]. Available: https://www.ti.com/lit/ds/symlink/tps54331.pdf
[15] Texas Instruments, “TPS6216x: 3-V to 17-V, 1-A Step-Down Converters—Datasheet.” Accessed: Nov. 7, 2025. [Online]. Available: https://www.ti.com/lit/ds/symlink/tps62162.pdf
[16] Microchip, “MCP1700 Low-Quiescent LDO—Data Sheet.” Accessed: Nov. 7, 2025. [Online]. Available: https://ww1.microchip.com/downloads/en/DeviceDoc/MCP1700-Low-Quiescent-Current-LDO-20001826E.pdf
[17] Espressif, “ESP-NOW—ESP-IDF Programming Guide.” Accessed: Nov. 7, 2025. [Online]. Available: https://docs.espressif.com/projects/esp-idf/en/stable/esp32/api-reference/network/esp_now.html
[18] Bluetooth SIG, “Bluetooth Core Specification v5.2 (overview).” Accessed: Nov. 7, 2025. [Online]. Available: https://www.bluetooth.com/bluetooth-resources/bluetooth-core-5-2-feature-overview/
[19] Digi, “XBee RR 802.15.4 RF Module—User Guide.” Accessed: Nov. 7, 2025. [Online]. Available: https://docs.digi.com/resources/documentation/digidocs/PDFs/90002522.pdf
[20] Semtech, “SX1276/77/78/79 LoRa® Transceivers—Product Page.” Accessed: Nov. 7, 2025. [Online]. Available: https://www.semtech.com/products/wireless-rf/lora-connect/sx1276
[21] IEC, “IEC 61326-1:2020—EMC Requirements for Electrical Equipment for Measurement, Control and Laboratory Use.” Accessed: Nov. 7, 2025. [Online]. Available: https://webstore.iec.ch/en/publication/62
[22] ISO, “ISO 10993-1:2018—Biological Evaluation of Medical Devices—Part 1.” Accessed: Nov. 7, 2025. [Online]. Available: https://www.iso.org/standard/68936.html
[23] IEC, “IEC 62366-1:2015/AMD1:2020—Application of Usability Engineering to Medical Devices.” Accessed: Nov. 7, 2025. [Online]. Available: https://webstore.iec.ch/en/publication/59980

[24] Castle Creations, “CC BEC—10 A Switching Regulator (spec page).” Accessed: Nov. 7, 2025. [Online]. Available: https://www.castlecreations.com/en/cc-bec-010-0004-00
[25] International Electrotechnical Commission (IEC), “IEC — International Standards and Conformity Assessment for Electrical, Electronic and Related Technologies.” Accessed: Nov. 7, 2025. [Online]. Available: https://www.iec.ch/
[26] International Organization for Standardization (ISO), “ISO — International Organization for Standardization.” Accessed: Nov. 7, 2025. [Online]. Available: https://www.iso.org/
[27] IEEE Standards Association, “IEEE SA — Standards.” Accessed: Nov. 7, 2025. [Online]. Available: https://standards.ieee.org/