TEAM 24 FINAL REPORT

By

Abhyan Jaikishen

Christopher Zhang

Daniel Hsu

Final Report for ECE 445, Senior Design, Fall 2022

TA: Jason Paximadas

07 December 2022

Project No. 24

Abstract

Team 24 designed and built a grounded rope management system for belaying. The portable system was created to fully replace the responsibilities of a belayer when top rope rock climbing. In short, the system can belay, lower, and allow climbers to rest during operation using an iPhone application. The constructed system can catch falls, belay at a certain speed, and can lower climbers safely. A climbing device, called a grigri, was used to catch falls. Rock climbing is inherently dangerous and while testing the system, all members made sure to take the proper precautions. The system has yet to be tested in a proper climbing environment due to liability issues and a lack of outdoor, top rope climbing in the Urbana-Champaign area.

1. Introduction

1.1 Problem

Rock climbing is an inherently high-risk sport that involves being at dangerous heights with not a lot of safety equipment. Therefore, whatever safety system used must be very reliable. While there are some types of rock climbing where the climber does not even have a rope, this is usually done at lower heights where any falls are controllable and would not result in injury. Top rope is a type of climbing where a rope is fed through a pre-mounted pulley or hook at the top of a climb and attached to the climber on one end and the belayer on the other. The belayer's job is to control the rope's slack and keep the climber safe while going up, as well as lower the climber at the end.

With this human belayer system provides the drawback that a partner is necessary to climb top rope. While climbing in a group or with a partner is inherently safer, there are many people who would rather climb alone and would accept the risks of doing so. However, in the case of top rope, that is not entirely possible in the current market.

Additionally, there are many jobs that require people to climb to dangerous heights such as cell tower maintenance that have a similar safety requirement as rock climbing. While many of these use cases have their own safety systems, many of these systems rely on expensive cable rails and pre-built systems.

1.2 Solution

We designed a grounded-rope management system that fully replaces all jobs of a typical belayer. While there are auto-belays that exist on the market, they are expensive (>$2000) and are only fixed, indoor solutions, which must be pre-mounted at the top of a climb. Thus, existing solutions are less practical for outdoor climbers, anyone who wants to enjoy climbing alone, or even workers who need to climb high towers. Our system can be more flexible as well as cut costs compared to the existing systems.

The system takes advantage of a grigri (a simple rope-management tool for belayers) and emulates human operation of the grigri using 2 motors. The system runs on a 12V battery, utilizes an ESP32 Microcontroller for communication, and is paired with an iOS app that communicates via Bluetooth.

1.3 High Level Requirements

The final product we built satisfies our original high-level requirements:

- Rope system must be able to maintain an acceptable level of tension (such that excess rope length is less than 4 feet) on the rope while the climber ascends without actively pulling the climber up the wall.
- The climber must be able to communicate with the rope system wirelessly (range of at least 50 feet) to give the following commands: stop, start, and lower.
- Rope system must be able to handle the climber's fall safely by catching the fall within 4 feet.

2. Design & Verification

2.1 Introduction and Procedure

The most important considerations for the design were capability and reliability. So, in deciding on our subsystems and their corresponding design, that was the focus. The main blocks were separated into the power supply, mechanical, control, and software subsystems (Figure 6). The power supply subsystem provides a separate higher voltage line for the mechanics and a lower voltage line for the digital side. The control subsystem consists of the ESP32 microcontroller, which handles all the state control, handling the data input and output from the mechanics and the user interface. The mechanical subsystem consists of the motor system and physical chassis design. Finally, the software subsystem contains the user-facing mobile application, which interacts wirelessly with the firmware on the microcontroller.

2.2.1 Power Subsystem

The power subsystem includes the battery and a linear voltage regulator (Figure 9). The main purpose of the power subsystem is to power the system using two main power lines: 12V and 3.3V. The 12V power runs through the motor controllers to motors, with all other components on the PCB are on the 3.3V line. While we originally intended to use the MCP1703T, the part became out of stock at all major components retailers before the department could place the order. Instead, we pivoted to use an LM317T, a larger linear voltage regulator that we placed on a breadboard.

2.2.2 Control Subsystem

The control subsystem consists of the ESP32-WROOM-32D microcontroller, which we will refer to as the ESP32, and the motor controllers, all of which were soldered onto the PCB. The ESP32 receives and sends signals over Bluetooth Low Energy (BLE) to tell the motor controllers how to drive the motors. We devised a state machine (Figure 1) that the ESP32 follows to change motor control for belaying that best emulates how a human would belay. Initially, we used the MC33931VW motor controllers to detect current feedback and drive the motors using PWM. A shunt resistor was used in line with the motor controller feedback pin which provided 0.24% of the load current as a reference current. This provided a voltage that the motor controller was able to read. During falls, the load current would sharply increase, flagging the system to change states. To keep the rope taught, the motors were driven until they could no longer spin, i.e., all the slack in the rope was taken up. Given that the motors were different for each hand, software adjustment was done to the duty cycle to match the speeds for adequate belaying.

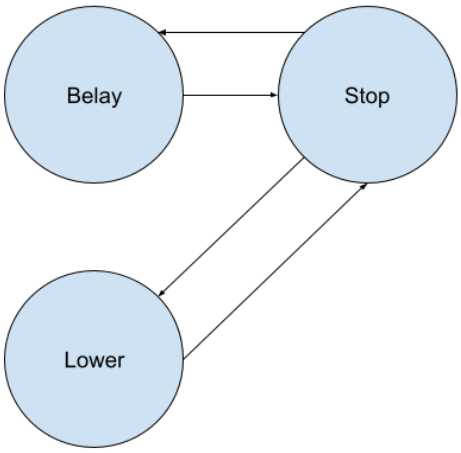

Figure 1: State Machine for Belay System

2.2.3 Mechanical Subsystem

The mechanical subsystem consists of the 2 motors, the mechanical clutch system, the grigri, and the servo. Initially, two LP3VA motors were supplied by the machine shop for use in the project. During the testing phase, however, we realized that the gear boxes in both motors were destroyed. Multiple gears had multiple missing teeth, causing little to no rotation (Figure 10). Fortunately, Gregg from the Machine Shop was able to supply two different motors (which differed in torque and RPM).

Mechanical challenges were a large concern for this project, but we worked closely with the Machine Shop to construct a fully working design. Motors were mounted in a diagonal position surrounding the grigri and the grigri was pinned in a certain orientation to allow the rope to form an 'S' shape through the grigri. This setup allows the most friction between the rope and grigri- enough to the point where the machine is still able to belay, but also lock-up in the event of a fall.

One of the largest components of catching a fall was the concern of damaging the motors or generating back-EMF that would damage components on the PCB. The final design used a multi-shaft clutch system (Figure 12). The smaller shaft fits flush into the larger, threaded shaft which contains a small Teflon plate to allow for slipping, which prevents gearbox damage during falls.

Unfortunately, due to time constraints and part supply issues, we were unable to mount the servo to pull the lever on the grigri.

2.2.4 Software Subsystem

The software subsystem consists of both the user interface in the iOS mobile app, as well as the ESP32 firmware that controls the entire system. As described in the control subsystem, the system operates on a state machine, so that can be controlled by the user through the mobile app. The phone and microcontroller communicate over BLE. This was chosen for both development and design reasons. First off, Apple does not allow for unregistered Bluetooth peripherals using normal Bluetooth, so in the development process, BLE was the only option.

Figure 2: Belay App Home Screen

The mobile app both sends messages to the microcontroller and receives feedback, displaying the current state of the machine to the user. The interface was designed to be simple and easy to use, since the user cannot dedicate much attention to the app while climbing.

2.3 Verification

2.3.1 Completed Project

We experienced several roadblocks along the way for every subsystem, but we will only discuss the most significant obstacles in this section. These obstacles, for better or for worse, changed the course of our projects and required pivoting.

The first major obstacle we experienced was regarding the mechanical design. Initially, we planned to use a one-way bearing to deliver torque in one direction, while allowing freewheeling in the other. At the suggestion of the machine shop, we took measurements for what shaft sizes we required and discussed the part with Gregg and Skee. Our group and the machine shop later learned that the one-way bearing does not work as anticipated, given the way we were required to fix our shaft. After lengthy discussion, we decided to shift to a spring-clutch system. As described earlier, the clutch system allows

for slipping during high-force falls, but still allows us to deliver torque when belaying. This system worked perfectly for our application. This system, combined with the use of our 12V battery, also solved our back-EMF issue. If the clutch system were to fail (although this never occurred during testing) and the motors were to generate a voltage back into the system, the battery would simply recharge. The very same concept is used for recharging electric vehicles and skateboards during breaking.

The final obstacle unfortunately became an issue the night before the demo. During normal testing of our system, our voltage regulator suddenly started to heat up to a physically painful temperature. We began to probe our PCB for the issue and realized that the current draw to the board was 1A, which was significantly higher than anticipated or needed for the system. Normal load due to the ESP32 and motor controllers was around 0.5A at its peak. We unfortunately realized we had shorted some components on our board and after hours of testing could not pinpoint the issue. To solve this issue, we quickly pivoted to using an ESP32 development board and L298 motor controllers (which unfortunately had a different system for current sensing, requiring the use of an Op-Amp). While were able to get this system completely working for the demo, it was less than ideal given the current sensing setup of the new board. With a proper amount of time, we would have been able to debug the PCB correctly.

These obstacles provided valuable insight for future design considerations. Adding feedback protection and more diodes along the higher voltage lines might have prevented the board from frying. Moreover, breaking out more pins on the ESP32 and the usage of switches on the PCB would have likely saved us lots of trouble when programming the ESP32 (and getting it to boot in the correct mode). Although these considerations would have been helpful in hindsight, the 6th iteration of our PCB that was functioning up until the demo was a large improvement over the first iterations (Figure 8). Earlier testing could have been achieved if parts had arrived earlier, which would have likely eliminated these issues.

2.3.2 Requirements Discussion

Appendix A contains the full breakdown of the RV tables we had used for this project. We met all our requirements except for the lowering requirement. Lowering required the servo, which we were not able to implement in time. However, it is worth noting that we could properly entering the lowering state (i.e., our finite state machine worked as intended), but simply lacked the servo to implement this feature. To compensate, we drove the motors in the reverse direction to feed rope the other way when the grigri was not engaged.

The current-sensing capability was missing from our final demo build since we fried our PCB with the working motor controllers and current sensing circuit the night before. Earlier we had shown that we were able to detect current (and therefore voltage) changes during falls and the ESP32 was able to read these values and scale them correctly.

Testing procedures are described in the corresponding RV table in the appendix, along with the measurements for quantitative tests. Below is a generated graph for the signal time requirement we had for the control unit.

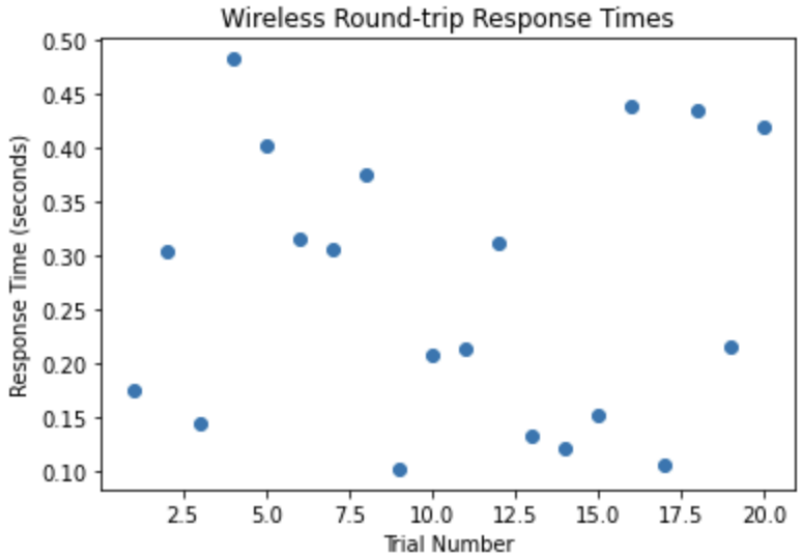

Figure 3: Wireless round-trip response times

Figure 3 shows the measured response times in 20 trials of sending and receiving responses between the mobile app and microcontroller. The average response time came out to be 0.2676 seconds, and maximum 0.4823 seconds, which are well within our requirement of 1 second.

3. Costs and Schedule

3.1 Parts Cost

Table 1. Part and Cost List

Parts	Part Name & Link	Qty	Price ($)
Microcontroller	ESP32-WROOM-32D	1	14.59
Left Hand Motor	1LPV3A	1	289.99
Right Hand Motor	1LPV6A	1	289.99
Grigri Servo	FS5106B	1	15.49
Motor Controller	MC33931VW	2	23.21
Lead Acid Battery	NP4-12	1	45.41
Voltage Regulator	MCP1703T	2	0.73
Grigri*	Petzl Grigri	1	99.99
Carabiner*	Spirit Lock Carabiner	1	16.95
Rope	Twisted Polyester Rope	1	7.99
Cost Summary	Total Parts Cost (including items in possession)	$828.28	
	Total ECE Student Labor Cost	$20,475	
	Total Machine Shop Labor Cost	$1,240	
	Total Cost	$22,543.28	

3.2 Labor Cost

ECE Student Labor Cost: To estimate the student labor cost, we found the average starting salary of a CE undergraduate from UIUC to be ~ $\$105{,}000$ per year. Assuming the average graduate works 50 weeks per year, and 40 hours per week, the average hourly salary would be roughly $\$52.50/hr$. Assume each student in the group spends roughly 10 hours per week with 13 weeks in the project. Given that there are 3 students, the total ECE student Labor Cost for this project would be:

$$3\ students * 10\ hours/week * 13\ weeks * \$52.50/hour = \$20{,}475.$$

Machine Shop Labor Cost: According to PayScale, the median Machinist pay is $\$32.26/hr$. We spoke to the machine shop and estimated that the machine shop will spend 40 hours mounting the motors to the board, building a battery mount, and creating the clutch system.

$$40\ hours * \$31/hour = \$1{,}240.$$

3.3 Total Cost

The total cost for this project would be an estimated $22,543.28, which is simply the sum of the parts and labor cost. Note that several costs were avoided thanks to the help of the machine shop and climbing equipment already in possession. Both motors were provided by the machine shop and material cost for the build was not accounted for here. Fortunately, the team already possessed a grigri and carabiner, which typically retail for around $110 combined.

3.4 Schedule

Dates	Task	Team member
9/26 – 9/30	Worked on Design document	Abhyan
		Chris
		Daniel
10/3 – 10/7	Attended design review Selected parts	Abhyan Chris
	Attended design Review Designed Schematic	Daniel
10/10 – 10/14	Attended PCB Review	Abhyan
		Chris
	Created Schematic	Daniel
10/17 – 10/21	Met with Machine Shop for Mechanical Design Fixes Created PCB	Abhyan
	Met with Machine Shop for Mechanical Design Fixes	Chris
	Created PCB	Daniel
10/24 – 10/28	Met with Machine shop for fixes Fixed PCB Ordered parts	Abhyan
	Met with Machine shop for fixes Ordered parts	Chris
	Fixed PCB	Daniel
10/31 – 11/4	Restarted schematic to fix several issues and fixed PCB (V3) Ordered more parts and headers Updated mechanical design Acquired new motors from Machine Shop	Abhyan
	Restarted schematic to fix several issues Ordered more parts and headers Completed ESP32 Interface Code (Bluetooth working) Updated mechanical design Acquired new motors from Machine Shop	Chris
	Servo design Development board motor driving	Daniel
11/7 – 11/11	Revised Design Document Worked on current feedback Worked on ESP32 Software	Abhyan
	Worked on iOS app Worked on ESP32 software	Chris
	Revised Design Document	Daniel
11/14 – 11/18	PCB Soldering Mock Demo	Abhyan
	Worked on ESP32 Software Worked on iOS app PCB Soldering Attended Mock Demo	Chris
	Attended Mock Demo	Daniel

11/21 – 11/25	-	Abhyan
	-	Chris
	Debugged motor controller issues Met with machine shop	Daniel
11/28 – 12/2	Prepared for Final Demo	Abhyan
	Prepared for Final Demo	Chris
	Prepared for Final Demo	Daniel
12/5 – 12/9	Worked on Final Presentation and Report	Abhyan
	Worked on Final Presentation and Report	Chris
	Worked on Final Presentation and Report	Daniel

Table 2: Schedule of completed work

4. Conclusion

4.1 Accomplishments

Overall, we were able to meet all our high-level requirements in the end, and at some point, were able to meet the subsystem requirements. While unfortunately, we shorted our final PCB before demo time, we were able to get the current feedback from the motor controllers working and measured by the microcontroller.

The other subsystem requirements were also fulfilled, even on our last-minute, improvised breadboard setup. The wireless communication and software systems were fully working within our required latency and range values. The microcontroller was able to control the motors and sync them sufficiently to raise and lower the rope, as well as take in user input to switch between the belay, lower, and stop states. Finally, we were able to split the 12V battery into the higher voltage motor line and the lower voltage digital component line, letting our whole system work off one battery. Most importantly, as mentioned before, we were able to perfectly catch falls during operation.

4.2 Uncertainties

There were many setbacks and limitations that we faced. With both the global chip shortage and some troubles with PCB design, our timeline got pushed far back, losing a lot of time to delays, parts being out of stock, as well as having to redesign and reorder parts. Our final product did not include any current feedback detection since we were on an improvised circuit with different motor controllers.

We also had two motors with different RPMs, since we could only use spare motors that the machine shop graciously provided us. The slower of the two was considerably slower, which resulted in having to adjust for it and run everything at a slower rate, providing a less-than-ideal rate of belaying. A more finished product would want to use a pair of the same, faster motors to be smoother and faster: more suited for actual belaying.

Additionally, our mechanical design was far from perfect, having unreliable rope coiling and tangling due to spools that were too wide. This would want to be remedied for a commercial product.

4.3 Ethical considerations

Rock climbing, especially rope climbing, is an inherently dangerous sport. Safety is of the utmost concern and several systems and techniques are used to mitigate risks. Using the correct equipment and taking extra precaution prevents injury and potential death.

When we designed this rope-management system, we followed climbing and mechanical industry standards, in addition to documenting the full process.

There are some ethical concerns regarding unintended usage of the system. However, we were always the only users of this system, as we are familiar with the controls and dangers. Under no circumstances did we allow any untrained or non-group members to use the belay system. When active, the system was always be under supervision to prevent any misuse and resulting damage. This is in accordance with IEEE Code of Ethics (section 7.8.I.1).

All group members have completed the lab safety training and exercised caution when working in dangerous environments. When working in the machine shop, we took adequate precautions to avoid injuries and accidents. This is in accordance with IEEE Code of Ethics (section 7.8.III.10). Throughout the semester, we received feedback from TAs, Professors, and the machine shop employees, to alleviate concerns and pursued honest work (section 7.8.I.5).

4.4 Future work

There are several improvements and points of future work we would like to cover before finalizing this product. We hope to improve user experience with a better point of control for the belay system; something like a wristband RF module to send basic stop, start, lower signals. This would be more useable and allow for extended range.

Smaller improvements include:

- Better motors (the same model with higher RPMs)
- More reliable threading: better knurling and indents for the rope
- Placing the servo for the lever on the grigri
- A fail safe to lower the climber if the system battery is low or the RF module signal drops
- Mounting brackets for pinning the system down outdoors

This project was an incredible learning experience from the start to finish. Learning about iterative design, picking parts, designing a PCB, iOS development, soldering, etc. was invaluable and highly enriching. However, the most invaluable experience was problem solving under time and budget constraints. This system could eventually go to market for climbers and other professions that use harnesses for climbing structures. We succeeded at solving a problem for which there is no current viable product out on the market, whether it be for traditional sport climbing or other belay applications.

References

[1] *"IEEE Code of Ethics."* [Online]. Available: https://www.ieee.org/content/dam/ieee-org/ieee/web/org/about/corporate/ieee-code-of-ethics.pdf

[2] "CEN - EN 15151-1 - Mountaineering equipment - Braking devices - Part 1: Braking devices with manually assisted locking, safety requirements and test methods | Engineering360," *standards.globalspec.com*. [Online]. Available: https://standards.globalspec.com/std/1540615/EN%2015151-1

[3] G. E. O. of M. and Communications, "Salary Averages," *ece.illinois.edu*. [Online]. Available: https://ece.illinois.edu/admissions/why-ece/salary-averages

[4] "UIAA | Safety Standards UIAA," *UIAA*. [Online]. Available: https://theuiaa.org/safety/safety-standards

[5] VDiff, "How To Belay: Top Rope Basics - Learn To Rock Climb - VDiff Climbing," *VDiff*, Jan. 03, 2018. [Online]. Available: https://www.vdiffclimbing.com/basic-top-rope-belay/

[6] S. built by: Salary.com, "Machine Shop Tool & Die Maker III Salary in Illinois," *Salary.com*. [Online]. Available: https://www.salary.com/research/salary/alternate/machine-shop-tool-and-die-maker-iii-salary/il

[7] "TRUBLUE iQ Auto Belay," *Head Rush Technologies*. [Online]. Available: https://trublueclimbing.com/trublue-iq-auto-belay

[8] "Climb Safe: How to belay with the Grigri," *www.youtube.com*. [Online]. Available: Climb Safe: How to belay with the Grigri

[9] "ESP32 Wi-Fi & Bluetooth Modules I Espressif," *www.espressif.com*. https://www.espressif.com/en/products/modules/esp32

[10] "ESP32 Bluetooth Classic with Arduino IDE - Getting Started | Random Nerd Tutorials," May 10, 2019. https://randomnerdtutorials.com/esp32-bluetooth-classic-arduino-ide/

[11] "Apple Developer Documentation," *developer.apple.com*. https://developer.apple.com/documentation/corebluetooth

[12] "In-Depth: Interface L298N DC Motor Driver Module with Arduino," *Last Minute Engineers*, Nov. 28, 2018. https://lastminuteengineers.com/l298n-dc-stepper-driver-arduino-tutorial/

[13] "PACE Inc. Soldering SMD fine pitch component (8a) Hot Air Mini Wave Soldering," *www.youtube.com*. https://www.youtube.com/watch?v=IgrEbJN8dcQ (accessed Dec. 08, 2022).

Appendix A

Requirement and Verification Tables

Power Subsystem

Requirement	Verification	Status
1. Internal Power Dissipation of the Linear Voltage Regulator must not exceed 1.006W.	A multimeter will be used to determine input voltage, output voltage, and output current. This can then be used to calculate the Internal Power Dissipation of the chip. The calculated value should not exceed 1.006W. The formula used is $P_{LDO(MAX)} = (V_{in(MAX)} - V_{out(MIN)}) * I_{out(MAX)}$ = (12V - (0.97*2V)) * 0.1A = 1.006W.	Y

Table 3: RV table of the power supply

Control Unit

Requirement	Verification	Status
1. The control unit should be able to detect a fall and signal the app that a fall has occurred. A fall should cause an estimated current change from 1.8A to 2A.	This can be verified by utilizing the feedback pins on the motor controllers. 0.24% of the load current is the reference current the feedback pin provides. By running this current through a 100 Ohm resistor, we receive a reference voltage. When the current changes from 1.8A to 2A, under a higher load, a difference in reference voltages (0.48 - 0.432 = 0.048V) can be identified. a. Normal: (0.24% * 1.8A) * 100 Ohm = 0.432V b. Load: (0.24% * 2A) * 100 Ohm = 0.48V	N
2. The system can enter the 3 states given commands from the mobile app: belay, lower, stop.	These states will be verified using an LED and visual confirmation. c. Belay: Both motors spin at the same rate, pulling away excess rope d. Lower: The motors are stopped and the microcontroller is able to send a “lower” command e. Stop: The motors are stopped and the weight on the rope will not move.	Y

Table 4: RV table of the control unit

Motors and Mechanics

Requirement	Verification	Status
1. Mechanics must be able to pull rope through gri-gri alone without stalling at a rate of 2±0.2 inches/second.	Drive motors to pull rope through gri-gri, measure rate by length of rope. A simple mark on one point of the rope, combined with a reference point can show how much rope has been pulled through the system. Monitor the current draw of motors with a multimeter/oscilloscope to check if stalling. Tests showed that the grigri was able to pull rope at a rate of approximately 1.8 inches per second, which was slower than originally required. However, the requirement was modified with the new motors with lower RPM in mind.	Y
2. System must be able to handle the force of a climber's fall. • Handle any back-emf of motors' backspin (current should not exceed 0.5A) • Mechanical subsystem should withstand fall force	Procedure: • Monitor current between power source and motor system using a multimeter or oscilloscope. • Simulate fall by pulling rope with significant abrupt force (or dropping a weight). The system was able to handle the generated back-EMF, and no components were damaged during the fall tests.	Y
3. Both motors run at the same speed (given that they are different motors), maintaining an acceptable level of slack.	This can be verified by measuring the RPM of each motor after determining the correct voltage to drive each motor at. RPM can be measured using an encoder or marking a point on the shaft and timing how long one rotation takes, which can then be converted to RPM. An acceptable level of slack can be visually verified. The required speed to belay at a rate of 2±0.2 in/sec will be found through trial and error. As shown, the motors were able to run at a similar speed and belay properly.	Y

Table 5: RV table of the motors and mechanics

Communications and Software

Requirement	Verification	Status
1. Wireless communication range should work reliably within 50 feet (need a requirement because using Bluetooth low energy).	Procedure: • Stand at ranges of 10, 20, 30, 40, and 50 feet from the system. • Issue commands from app • Verify correct command was executed based on LED response and app display	Y
2. The round-trip time of sending a signal from the phone and receiving a state-change notification back from the controller is under 1 second.	Round-trip time was verified through a timer built into the iOS application. The app starts a timer when sending a signal to the ESP32. Once the ESP32 sends a signal back to the app and is received, the app stops a timer. All commands and their responses were verified to be under 1 second.	Y

Table 6: RV table of the communications and software

Appendix B

Figures and Diagrams

Visual Aid(s)

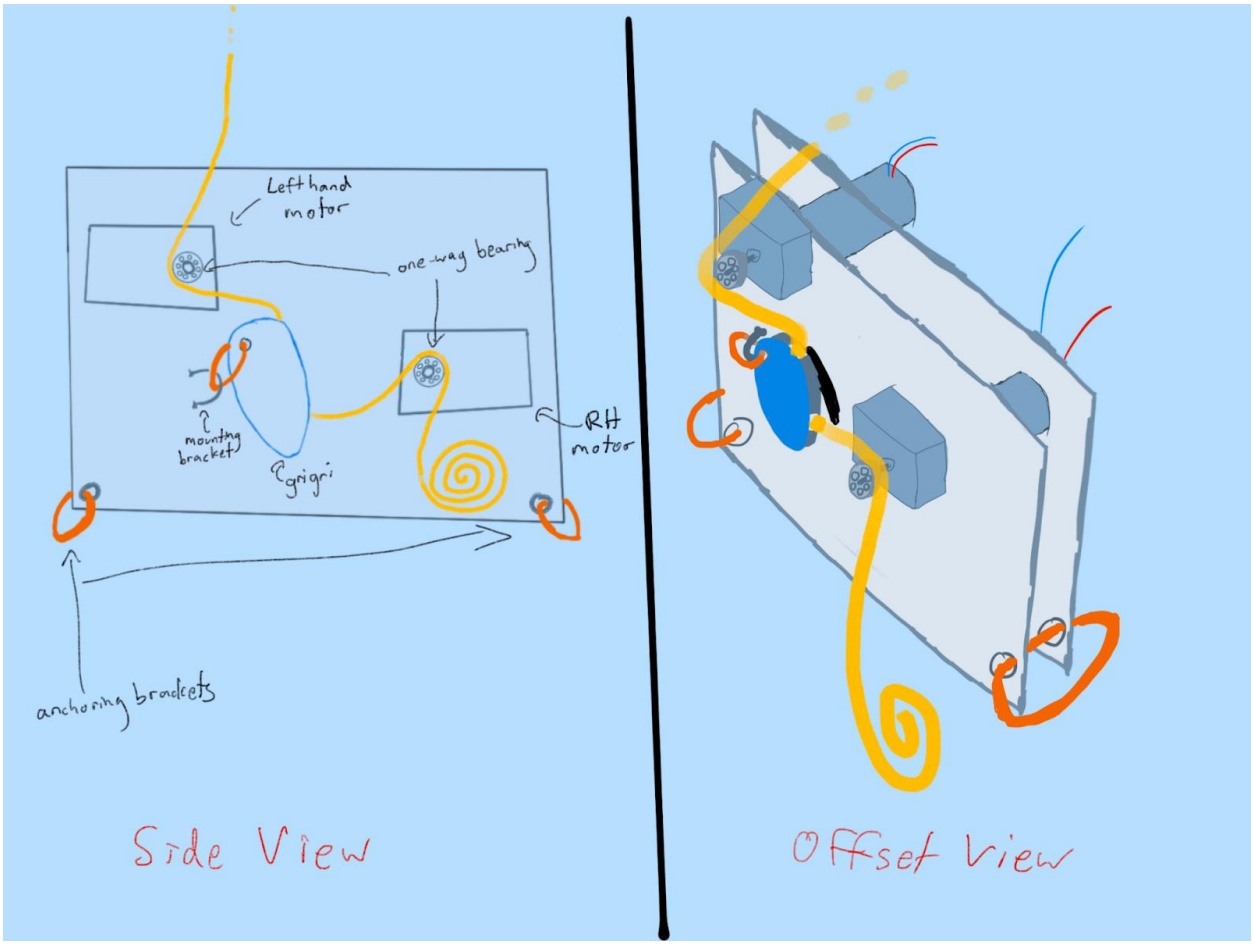

Figure 4: Rough sketch of the belay system.

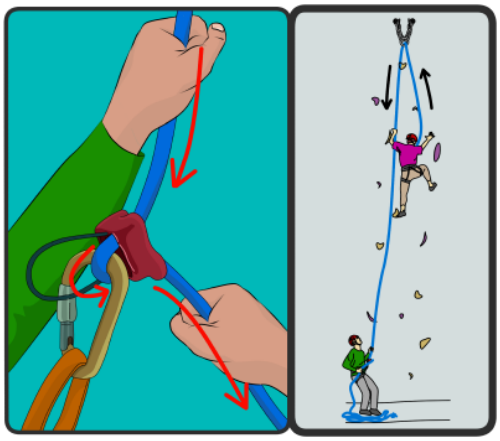

Figure 5: Typical top-rope climbing setup. In this example the belayer is using an ATC instead of a grigri, but the same principles apply.

Block Diagram

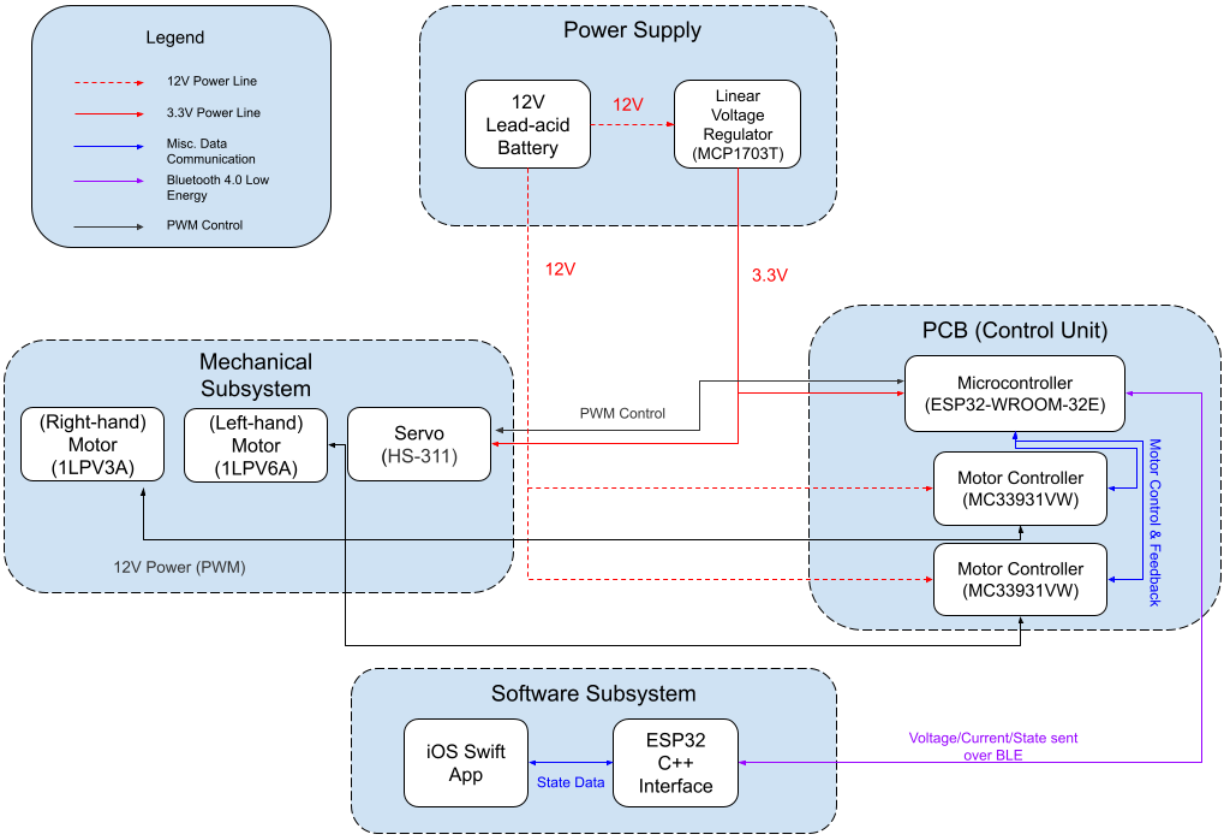

Figure 6: Block Diagram for the belay system. Note: The ESP32 uses I2C, but we are not utilizing it for this project.

Circuit Schematic

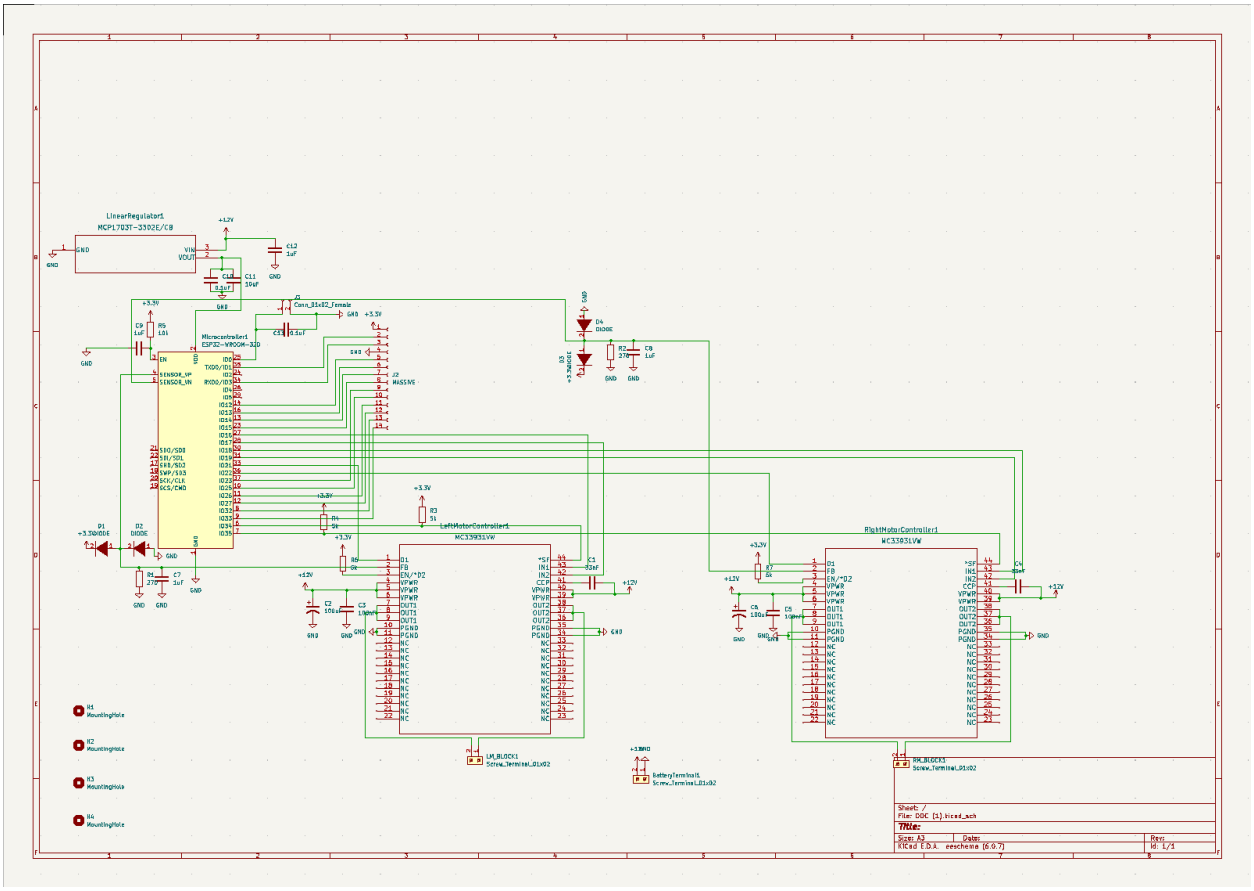

Figure 7: Schematic layout

PCB

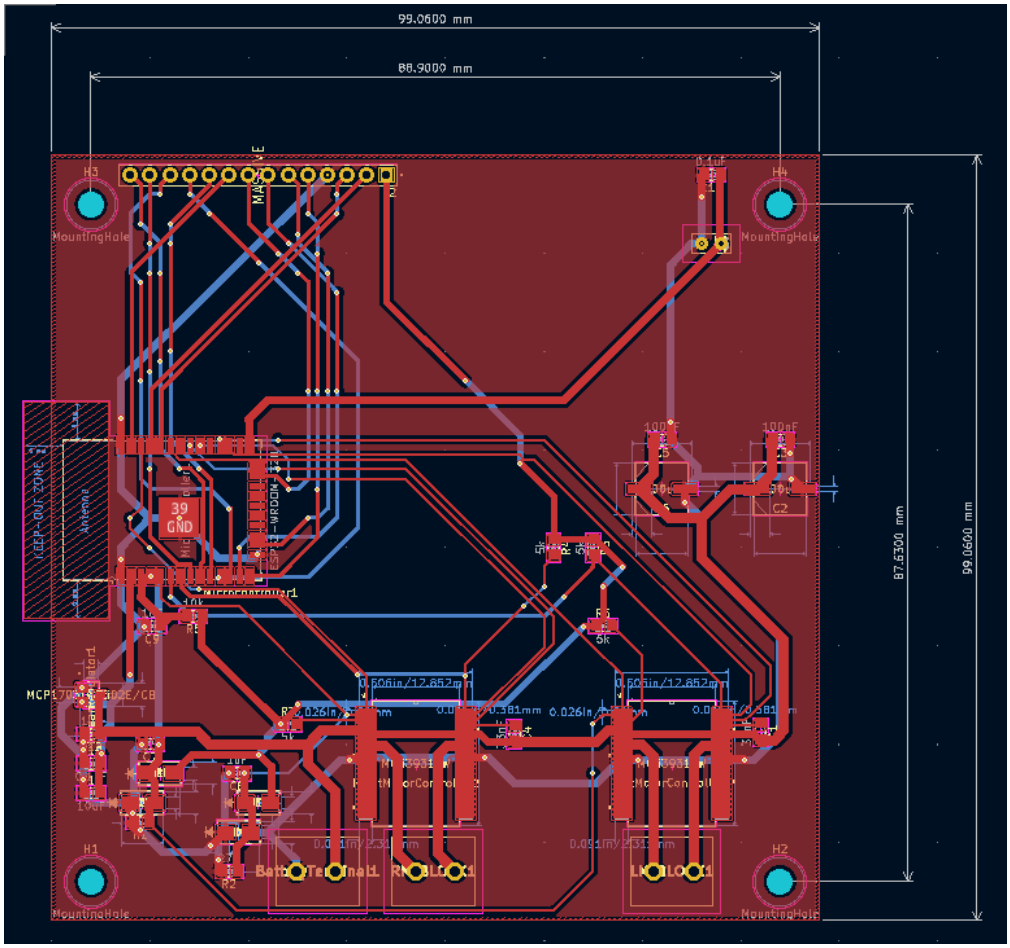

Figure 8: PCB design

Miscellaneous

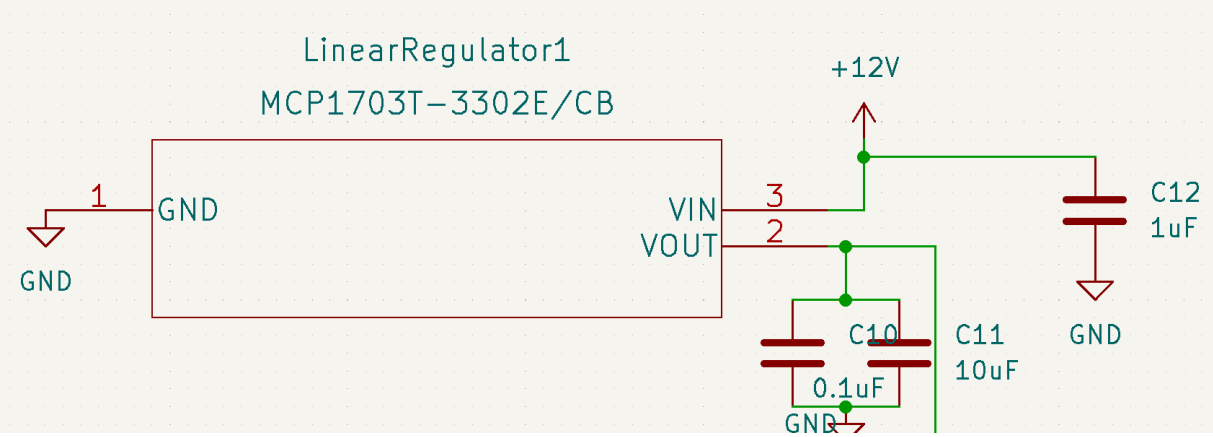

Figure 9: Voltage regulator

Figure 10: Broken gearbox

Figure 11: Portable rope management system

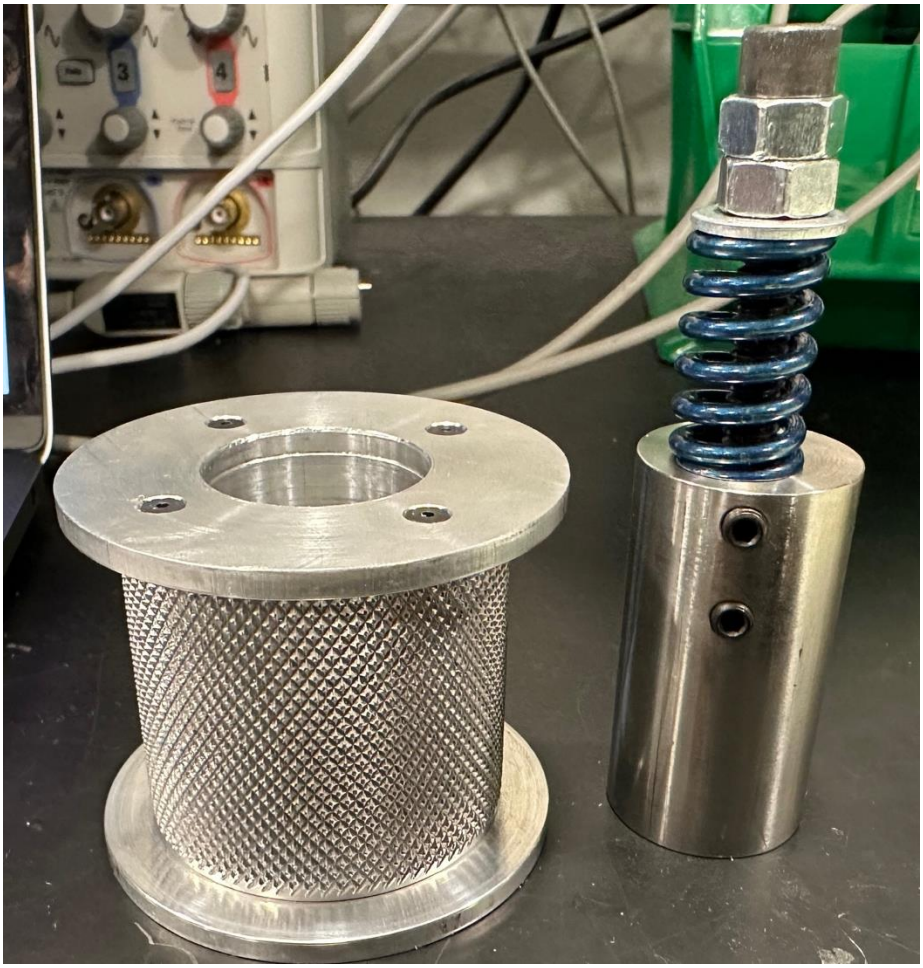

Figure 12: Shaft system with spring