ECE 445

Senior Design Laboratory

Final Report

Virtual Reality Gloves

Team #58

Ashton Billings
(ashton6@illinois.edu)
Hamza Lutfi
(hamzael2@illinois.edu)
Aditya Nebhrajani
(avn5@illinois.edu)

TA: Jason Zhang

May 7, 2025

Contents

1 Introduction

1.1 Problem

Although research into virtual reality (VR) has been ongoing for decades, even here at our university. It wasn't until the late 2010s that VR technology matured enough to become commercially viable. Early consumer devices, such as the HTC Vive and Oculus Rift[1] were heavy wired systems that required dedicated rooms with external sensors for proper operation. Their graphics performance was limited, and the immature VR ecosystem meant that few immersive experiences were available. Despite the promise of immersive virtual environments, user interaction still relied on traditional handheld controllers with buttons and joystick breaking the illusion of "true" reality.
In recent years, VR headsets have improved dramatically, driven by major industry investments and heightened consumer interest. A landmark moment was Facebook's acquisition of Oculus[2] and its rebranding to Meta, which shifted the company's R&D focus toward the metaverse and VR technology. Modern flagship headsets no longer require external sensors, thanks to built-in tracking systems, and can operate wirelessly. Meta's Quest series, for instance, supports full wireless operation[3] and offers "Air Link" for untethered PC connectivity. Despite these advancements in headset technology, controller design has seen relatively little innovation. Most leading headsets still use one-handed controllers with buttons and joysticks. While the Valve Index introduced finger tracking[4], it remains fundamentally a handheld device. The Apple Vision Pro is a notable exception, using external cameras for controller-free hand tracking[5]. There are also niche products from smaller companies aiming to bring full hand tracking to VR, but these tend to be bulky and expensive. Among these alternatives, two projects stand out as inspiration for our work. The first is the UDCAP VR glove, a kickstarter project offering joint-tracking gloves with built-in buttons and a joystick for controller-like functionality. The second is LucidGloves, an open-source VR glove initiative focused on accessibility and community-driven development [6].
All of these innovations share a common goal: to make virtual experiences feel more real. By enabling users to reach out, grab, and interact with virtual objects as if they were truly tangible, these devices seek to bridge the gap between the physical and digital worlds. Our project builds upon this-developing a system that not only provides realistic haptic feedback, but also preserves the intuitive input features users expect from traditional VR controllers.

1.2 Solution

Our solution to the current limitations of VR interaction is to develop our own hand-tracking gloves, designed to be compatible with existing VR headsets on the market. As previously mentioned, our design takes significant inspiration from the UDCAP Kickstarter project, so it is useful to first summarize their approach:
UDCAP's gloves offer finger joint tracking through proprietary "elastic sensors" placed over each joint. While the specifics of these sensors are not disclosed, we believe they function similarly to strain gauges. Additionally, the gloves feature an index finger module with a Hall-effect joystick and buttons, enabling standard controller input. The system is wireless, communicating over 2.4 GHz with a reported latency of less than 10 ms, and is powered by a

760 mAh rechargeable battery offering 10–15 hours of use per charge. To achieve positional tracking, users must attach an external tracking module such as a Vive Tracker or a Quest controller via adapter mounts sold by UDCAP. These gloves are compatible with SteamVR, making them usable with any mainstream PC VR headset.
Although UDCAP is a Kickstarter product[7], the company behind it consists of a sizable engineering team (reportedly 40–50 members) and has been in development for over three years. We acknowledge that matching their performance and polish is not feasible within the scope of our project. Instead, our goal is to closely emulate the core functionality while introducing specific differences and innovations.

Like UDCAP, we aim to implement finger curl tracking; however, rather than using elastic sensors, we have transitioned to a Hall effect–based sensing approach. Each finger will be equipped with a module containing two perpendicular Hall-effect sensors[8], [9], which detect the relative position of a magnet attached to the rotating joint. This setup allows us to capture 2D magnetic field data and infer angular displacement for accurate finger tracking. These sensor modules connect to the main control board via JST connectors for ease of integration and modularity. In terms of user input, our glove will include a Hall effect joystick mounted on the index finger module to enable directional input. The glove system will remain wireless and be powered by a 2000 mAh, 7.4V Li-ion battery, with onboard voltage regulators providing stable voltage levels to the electronics. For positional tracking, we follow the same approach as UDCAP by mounting a Meta Quest 2 controller directly onto the glove. This leverages the Quest's built-in positional tracking capabilities, allowing seamless integration with SteamVR or Unity environments. For wireless communication, we will use Bluetooth Low Energy (BLE), which operates in the 2.4 GHz ISM band. BLE offers low power consumption and reliable performance for our needs.

At this point, our design takes a different path from UDCAP's. One major addition is our plan to integrate a primitive haptic feedback system, inspired by a combination of LucidGloves V3 and LucidGloves V5 designs. Specifically, we will mount badge reels to each finger and use servo motors to lock the reels when finger collisions occur in the virtual environment. This creates the illusion of physical resistance when interacting with virtual objects, adding realism to the VR experience.

1.3 Brief Description of System Blocks

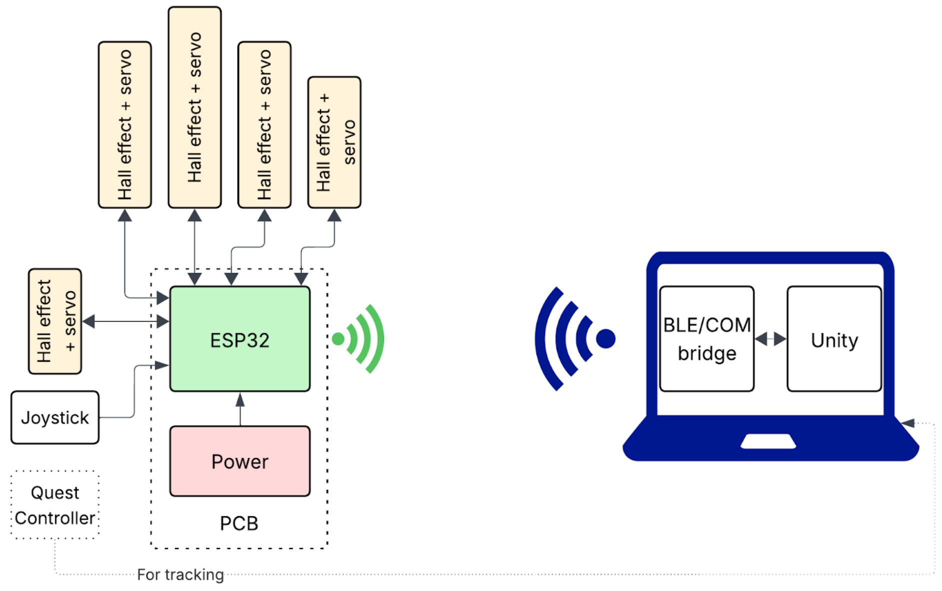

Figure 1: High-level Block Diagram

Our project is divided into the following key functional blocks, as shown in Figure 1:

ESP32 Microcontroller This is the central processing unit of the system[10]. It reads sensor inputs, controls actuators (servos), and communicates wirelessly with the host computer using Bluetooth Low Energy (BLE).

Hall Effect and Servo Modules Each finger is equipped with a hall effect module and a corresponding servo motor. The hall sensors track finger flexion based on magnetic field changes, while the servos provide haptic feedback by applying resistance to finger motion.

Joystick Module A small joystick is included to replicate traditional controller input, enabling the user to navigate VR environments in a familiar way. It interfaces directly with the ESP32.

Power Subsystem The power subsystem regulates and distributes power to the ESP32, sensors, and servos. It ensures the entire system operates reliably from a battery supply.

PCB All components are connected through a custom-designed PCB, which routes power and signals between modules and the ESP32.

Wireless Communication The ESP32 transmits sensor data wirelessly via BLE to a host computer, removing the need for wired communication and enabling untethered use.

Host Computer (BLE/COM Bridge and Unity) The host computer receives BLE data via a custom BLE-to-COM port bridge and parses it for use in Unity. Unity interprets the finger positions and joystick data to animate a virtual hand and facilitate interaction within the VR environment.

Quest Controller While our glove performs finger tracking and haptic feedback, we use an external Quest controller to provide position and orientation tracking within the VR space.

1.4 High-level Requirements

For our design to be successful, our gloves must meet the following requirements:

1. We can move around virtual objects with our gloves, with them properly locking in place for haptic feedback.
2. The communication between the gloves and software has a latency of < 500 ms.
3. The accuracy of our virtual finger curl is $\pm 15\%$ of our actual finger curl.

1.5 Block-Level Design Changes

One of the key design changes made during the semester was in the method chosen for finger tracking. Initially, we planned to use rotary potentiometers (POTs) mounted at the finger joints to measure angular displacement. This approach was appealing due to its simplicity and low cost. However, through early testing, we discovered several limitations. Most notably, potentiometers are mechanical components that suffer from dead zones and mechanical friction. These issues introduced nonlinearity and reduced the reliability of finger tracking especially for the subtle and precise movements required in VR interactions. To address these drawbacks, we transitioned to using Hall-effect sensor modules (mentioned in section 1.3). This setup allows us to track finger curl without direct mechanical contact, eliminating the issues we encountered with potentiometers. Additionally, the Hall effect sensors offer improved precision, durability, and consistency over time. This change improved both the electrical robustness and mechanical reliability of the finger-tracking subsystem, and brought our design closer in performance to commercial alternatives like UDCAP.

2 Design

2.1 Design Procedure

2.1.1 Glove Parts

The design of the VR glove's mechanical system evolved through several iterations, guided by lessons learned from the LucidGloves V3 and V5 projects. V3 was too limited in functionality, while V5—though an improvement with its hall effect sensors and haptic capabilities—suffered from excessive complexity and fragile mechanical construction. These challenges provided critical insight into what aspects to carry forward and what to avoid in our own implementation.

Our goal was to retain the functional strengths of V5—namely, hall-based angular sensing and servo-driven haptic feedback—while drastically simplifying the design to improve printability, durability, and ease of assembly. We prioritized reducing the number of required components, minimizing tight mechanical tolerances, and eliminating parts that had a history of failing under stress or prolonged use. This shift also helped reduce build time and made debugging more accessible when individual components needed replacement or tuning.

The result was a re-imagined architecture that preserved key capabilities but with far less mechanical overhead. We divided the system into two clear subsystems: the finger module, responsible for sensing finger motion and applying resistance, and the mounting and routing system, which securely anchors the modules to the hand and guides the reel strings to the fingertips. More detail about how this system works will be described in the design details section.

2.1.2 Printed Circuit Boards

The VR glove project required custom PCB design to integrate various subsystems, including sensor data acquisition, power management, BLE communication, and servo motor control. The primary challenge was to develop a modular and compact PCB solution capable of real-time performance while minimizing latency and power issues. The final design consists of two main boards: the main control board and the Hall effect sensor breakout board.

The main board houses the ESP32-S3-WROOM-1 microcontroller, power management circuits, UART programming interface, servo motor control outputs, and connections for the Hall effect sensors. Initially, we intended to use USB-C for communication, but due to signal integrity issues related to improper differential pair routing (D+/D- not length and impedance matched), we switched to UART for reliable serial communication. The power subsystem employs two voltage regulators (LM2596S-5.0 for 5V and AMS1117-3.3 for 3.3V), sourced from a 7.4V 2S Li-ion battery. Due to heating issues, we added heatsinks to the regulators to manage thermal dissipation.

The Hall effect breakout board is significantly smaller and mounts directly on the glove's fingers. Each board houses two perpendicular Hall effect sensors to detect the X and Y components of the magnetic field from a rotating magnet, providing a phase-based measurement of finger curl. Soldering these tiny boards was challenging, requiring precise technique to avoid bridging or misalignment.

During the PCB bring-up, we faced an issue where the microcontroller seemed unresponsive. After investigation, we found that a test LED was soldered in reverse polarity, causing it not to light up during programming. Reversing the LED solved the problem, and the microcontroller was correctly flashed.

2.1.3 Firmware

The firmware for the VR glove was designed to achieve real-time finger tracking and wireless communication via BLE using the ESP32 microcontroller. Initially, the project aimed to use potentiometers for finger tracking, but this was later replaced with Hall effect sensors to improve durability and accuracy. BLE communication was chosen to enable wireless data transmission between the glove and a Unity-based VR application. The firmware is structured around the key functions of data acquisition, BLE communication, and haptic feedback control.

The design process began with setting up the ESP32 to read analog signals from the Hall effect sensors, which detect finger curl. Each sensor outputs a voltage corresponding to the magnetic field strength, which changes as the fingers bend. The ADC of the ESP32 samples these signals at a rate of 50 Hz per finger, ensuring smooth tracking.

The next step involved configuring the BLE communication. The firmware was programmed to transmit the processed finger data to Unity using a custom BLE protocol. BLE data packets include positional data from each finger as well as button and joystick states. Haptic feedback was integrated by receiving collision signals from Unity, which triggers the servo motors to lock the badge reels attached to each finger.

To simplify communication with Unity, a BLE-COM bridge was developed using Python. This bridge establishes a virtual serial port interface between the ESP32 BLE device and Unity, allowing Unity to use standard serial communication without directly handling BLE protocols. This modular approach isolates game logic from low-level BLE management, improving maintainability and performance.

The final stage of the firmware design focused on optimizing communication latency and haptic response. The firmware was tested to ensure BLE latency remained below 20 ms[11] and that the haptic system activated within 50 ms upon receiving a collision signal from Unity.

2.1.4 Unity Software

From the beginning, we knew Unity with the XR Interaction Toolkit would be the core of our VR system[12]. It provided a robust framework for headset and controller tracking, physics interaction, and standalone Quest deployment. What wasn't as clear was how to animate the glove fingers using real-time data. This part of the design required more experimentation and prototyping, as Unity doesn't natively support sensor-driven finger animation out of the box.

We considered three main approaches. First, we explored directly controlling the Transform of each finger bone in real time using code. While flexible, this approach quickly became messy and difficult to manage across all joints, especially without built-in constraints or IK support. Next, we tried trigger-based animation — playing a fixed "finger curl" animation when a certain condition was met (e.g., curl > 0.5). However, this lacked nuance and felt jumpy or unnatural. Finally, we tested a system that continuously blended between two keyframe poses, driven by a float value from 0 to 1 representing finger curl. This method, which used Unity's Animator and Blend Trees, proved smooth, low-latency, and highly responsive — and is what we ultimately built the rest of the system around. More details about how this works as well as our other scripts, will go in the design details section.

2.2 Design Details

2.2.1 Glove Parts

The finger module consists of four primary 3D-printed parts that work together to enable both sensing and haptic feedback. The first is a custom inner plate inserted into a modified badge reel, designed to hold magnets used for angular sensing. The second is a general mounting bracket that secures both the badge reel and the servo motor, ensuring stable alignment between the mechanical and sensing components. The third is a side-mounted PCB plate that provides a clean and accessible platform for sensor and control electronics.

The fourth printed part is a bottom board clip, which attaches the entire module to the glove's hand wrap. This clip allows for a secure but removable connection, making it easy to swap or reposition modules during testing or iterative redesign. The badge reel itself is responsible for converting finger curl into reel rotation, and as the magnet rotates inside, its angular position is measured using two perpendicular hall effect sensors. A standard hobby servo motor provides haptic resistance by deploying a locking arm that physically prevents the reel from rotating when needed.

The mounting and routing system ties everything together on the hand. A flexible TPU hand wrap forms the base layer and provides mounting locations for each of the five finger modules. From there, a set of reel guides and finger wraps direct the reel string along each finger, terminating at small plastic fingertip covers. These fingertip pieces include a line holder that ensures the string remains taut and centered during motion. Lastly, we have the central PCB and battery mount located on the wrist. These parts can be seen in Appendix C.

2.2.2 Printed Circuit Boards

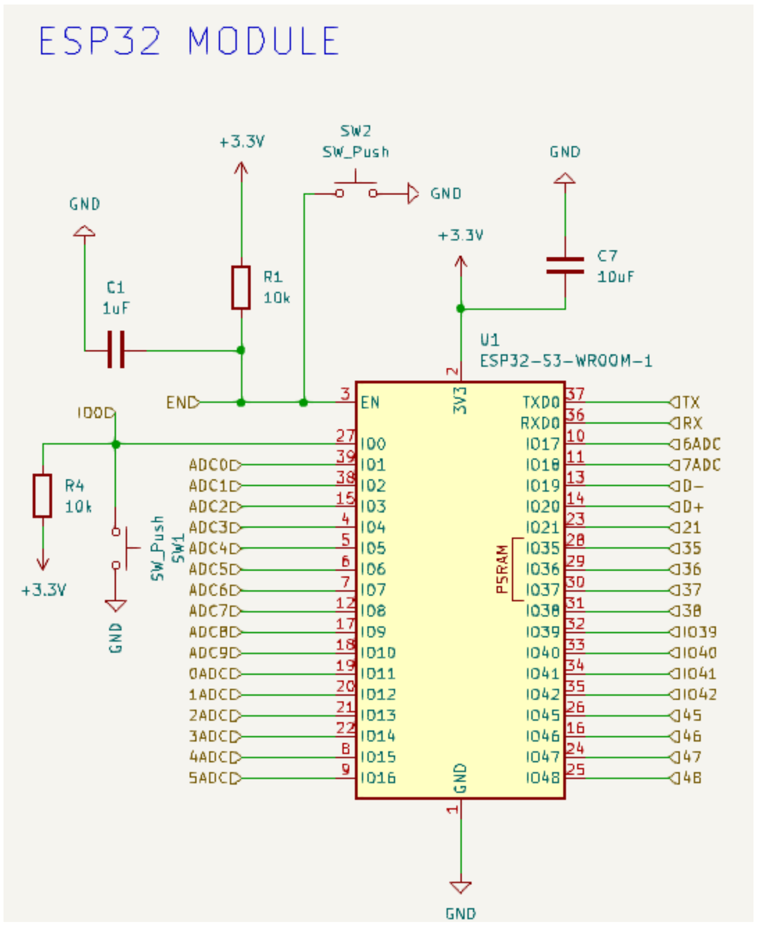

Figure 2: Micrcontroller Schematic

The main PCB is designed around the ESP32-S3-WROOM-1 module. The microcontroller module includes the necessary pull-up resistors for EN and IO0 lines, along with push buttons to manually control boot and reset functions. The UART programming header allows easy flashing and debugging. The PCB also features pin headers for servo motors, Hall effect sensors, and a joystick module.

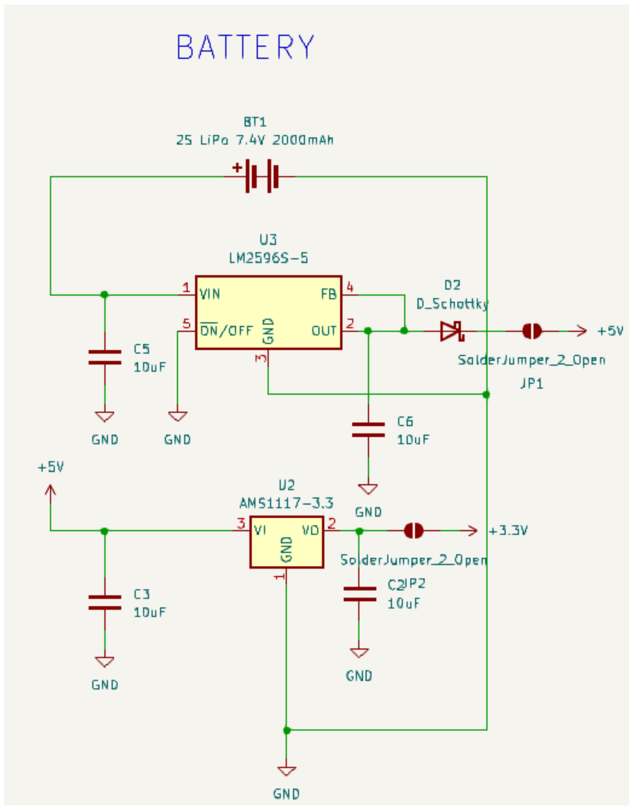

Figure 3: Power Subsystem

Power Management The power subsystem consists of the following components:

- LM2596S-5.0: Steps down the 7.4V battery voltage to 5V, primarily used for servo motors.
- AMS1117-3.3: Further regulates the 5V down to 3.3V for the ESP32 and logic circuitry.

To mitigate overheating, we added heatsinks to the voltage regulators after observing excessive temperature rise during peak current draw (especially from servo motors). The regulators are placed with adequate spacing to allow airflow.

Signal Routing and Communication The original design included a USB-C port for both communication and power. However, the D+ and D- lines were not correctly length and impedance matched, leading to USB communication failures. In the next iteration, we replaced USB communication with UART, which proved to be much more robust.

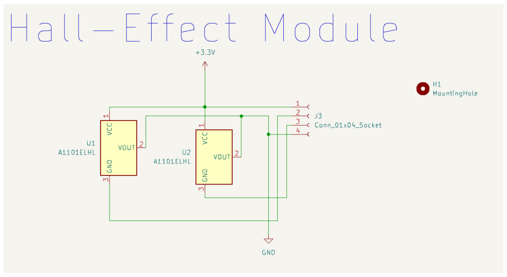

Figure 4: Hall Effect Schematic

Hall Effect Sensor Module The Hall effect sensor boards are compact and designed to be mounted directly near the finger joints. Each board contains two perpendicular Hall effect sensors to detect the X and Y components of the magnetic field. The small board size posed soldering challenges, requiring fine-tipped soldering irons and steady hands.

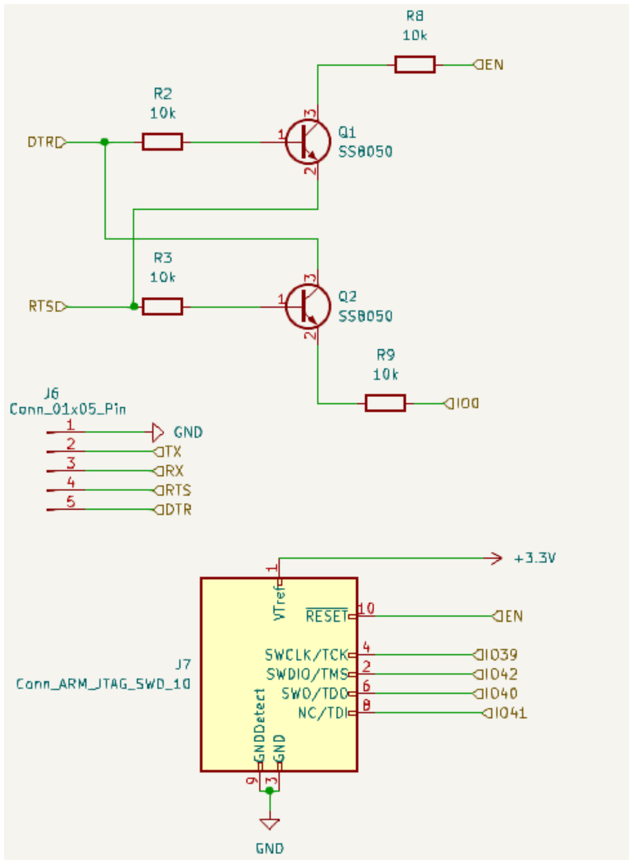

Figure 5: JTAG Debugging Logic

Debugging and Testing During the first bring-up, a reverse-biased test LED misled us into believing the microcontroller was not being flashed. Once we corrected the LED orientation, the microcontroller flashed correctly, and subsequent tests showed proper operation.

We validated power outputs using a multimeter and verified Hall effect sensor responses using a rotating magnet.

These design iterations and modifications led to a stable and functional VR glove PCB capable of handling the demands of real-time finger tracking and haptic feedback.

2.2.3 Firmware

The firmware uses the ESP32's ADC to read analog voltages from the Hall effect sensors. Each sensor is calibrated during initialization to determine the minimum and maximum voltage corresponding to fully contracted and fully flexed finger states. The ADC values are normalized to a range between 0 and 1 using the formula:

$$\text{Flex\_Value} = \frac{ADC - ADC_{min}}{ADC_{max} - ADC_{min}}$$

Signal Processing To reduce noise in the sensor readings, the firmware implements a low-pass filter using an exponential moving average (EMA). The filtered signal is computed as:

$$\text{Filtered\_Value} = \alpha \times \text{Current\_Value} + (1 - \alpha) \times \text{Previous\_Filtered\_Value} \qquad (1)$$

Here, α is the smoothing factor, chosen experimentally to balance responsiveness and stability.

Debouncing To prevent erratic behavior due to signal noise, the firmware applies software debouncing to the button inputs. A debounce delay of 20 ms ensures that spurious voltage changes do not result in false button presses. This is achieved by sampling the button state at fixed intervals and only registering a change if the state remains consistent.

Calibration During startup, the firmware performs a calibration routine to set the minimum and maximum ADC values for each Hall effect sensor. The user is prompted to fully contract and flex their fingers, and the corresponding ADC readings are recorded. These values are stored in non-volatile memory to preserve calibration across reboots.

BLE-COM Bridge for Unity The BLE-COM bridge developed in Python serves as an intermediary between the ESP32 and Unity. By using virtual serial ports, Unity can interact with the glove using standard serial communication rather than complex asynchronous BLE operations. The bridge continuously reads BLE data and transmits it over a virtual COM port, while also sending Unity commands back to the glove when needed. This approach greatly reduces latency and simplifies integration.

BLE Communication BLE communication is handled using the ESP32's native BLE libraries. Data is structured into packets containing finger positions and control states. The Unity application reads these values and maps them to the virtual hand movements. To ensure consistent data rates, a timer interrupt triggers data transmission at fixed intervals.

Haptic Feedback Haptic feedback is managed via GPIO-driven PWM signals that control the servo motors. Upon receiving a collision event from Unity, the corresponding servo motor locks the badge reel, providing a tactile stop to finger movement. The response time was optimized by minimizing data processing steps between signal reception and servo activation.

2.2.4 Unity Software

In our final architecture, the Unity application consists of five major scripts, each with a dedicated role in bridging glove hardware with the virtual environment.

SerialHandler This script manages all communication between Unity and the ESP32 over a virtual COM port. It reads the incoming float data stream (e.g., finger curl, joystick, button states) and can also write back to trigger haptic feedback when needed.

HandCalibration At the start of a session, this script captures the full 6DOF offset between the VR controller and the glove's physical mount. It then uses that offset to transform the virtual hand's position and rotation to match the real glove during runtime, ensuring accurate spatial alignment.

MovementHandler This component processes joystick input from the glove to control avatar locomotion in the VR scene. It enables movement without relying on the physical controller, which helps preserve immersion and opens up hands-free use cases.

FingerAnimator This script takes in the five normalized float values (one per finger) and maps them to parameters in Unity's Animator. These parameters drive Blend Trees that smoothly interpolate between extended and curled poses on each finger, providing lifelike hand motion.

ColliderRecoverer This handles physics interactions by detecting collisions between the virtual fingers and scene objects. It packages this data and can send events back to the ESP32—such as activating haptic feedback when a fingertip touches a virtual surface.

3 Verification

3.1 Overview of Verification Process

To verify that our system met all functional and performance goals, we carried out block-level and system-level tests on both the hardware and software subsystems. Key components such as the power regulation circuit, Hall effect sensors, ESP32 microcontroller, servo motors, and BLE communication pipeline were individually tested against the criteria outlined in our Requirement and Verification table. Where applicable, we used oscilloscopes, multimeters, protractors, Unity test scenes, and timing tools to quantify performance. The full R&V table is provided in Appendix A.

3.2 Power System Verification

The 7.4V Li-ion battery was tested under both idle and load conditions using an oscilloscope. We verified that the AMS1117 provided a stable 3.3V ($\pm$0.15V) and that the LM2596S step-down regulator provided 5.0V ($\pm$0.25V), staying within the 5% tolerance requirement. Ripple was observed to be minimal during active operation with both sensors and servos running.

3.3 Finger Tracking Verification

For each glove finger, we moved joints to known angles using a protractor and recorded the analog voltage outputs from the dual Hall effect sensors. These values were then fitted to a voltage-angle map to determine tracking accuracy. We confirmed that the calculated angles stayed within $\pm 5^\circ$ of actual flexion, meeting our specification.

3.4 ESP32 Communication and BLE Verification

We verified basic ESP32 functionality by uploading a test script that blinked the onboard LED, confirming that the board was programmable and responsive. BLE communication was tested by setting up a serial monitor and confirming that updated finger flexion data was being transmitted and received reliably over the BLE connection.

We also verified the BLE-to-Virtual Serial Bridge component. A known byte pattern was transmitted from the glove to a PC, where it was timestamped upon reception. Measured latencies were consistently under 50 ms, meeting our target.

3.5 Haptic System Verification

To test the haptic locking system, we pressed a button on the glove that triggered servo motors to lock the finger. We then applied an external force to confirm the servo held the finger in place. The system provided sufficient torque to simulate virtual resistance[13]. However, due to time constraints, we were unable to complete the Unity-to-glove BLE feedback loop needed to trigger servo locking from virtual object collisions.

3.6 Summary of Results

All core functional requirements were verified successfully at the hardware and communication levels. The only incomplete verification is the Unity-integrated haptic feedback trigger, which remains as future work. The rest of the design goals—power regulation, sensor accuracy, servo force, communication latency, and BLE responsiveness—were met within the expected tolerances.

4 Cost

4.1 Labor Cost Estimate

Team Member	**Hourly Rate**	**Hours Worked**	**Multiplier**	**Total Cost**
Ashton	\$31/hr	240 hrs	2.5	\$18,600
Hamza	\$31/hr	190 hrs	2.5	\$14,725
Aditya	\$80/hr	180.475 hrs	2.5	\$36,095
Total				**\$69,420**

Table 1: Labor Cost Estimate

4.2 Materials and Parts Cost

Item	**Part Number**	**Qty**	**Unit Price (USD)**	**Total Cost (USD)**
ESP32-S3-WROOM-1-N8R8	ESP32-S3-WROOM-1-N8R8	5	$6.13	$30.65
3.3V Regulator	AMS1117-3.3-SOT-223	5	$0.68	$3.40
5V Regulator	LM2596S-5.0	5	$4.68	$23.40
Schottky Diode	SS14	5	$0.29	$1.45
Capacitor 10 µF	GRM21BR61C-106KE15K	30	$0.14	$4.20
Capacitor 1 µF	CL21B105K-BFNNNE	30	$0.08	$2.40
Capacitor 100 nF	CC0805KRX7R-9BB104	30	$0.08	$2.40
Resistor 10kΩ	ERJ-6GEYJ103V	30	$0.10	$3.00
Resistor 5.1kΩ	RC0805JR-075K1L	30	$0.10	$3.00
Resistor 1MΩ	RG2012P-105-B-T5	30	$0.12	$3.60
Resistor 200Ω	CRGCQ0805F2-20R	30	$0.10	$3.00
LED	LG R971-KN-1-0-20-R18	10	$0.15	$1.50
Tactile Push Button	1825910-6	5	$0.13	$0.65
Pin Headers	PRPC040SAAN-RC	5	$0.64	$3.20
Total				**$86.45**

Table 3: Bill of Materials

5 Conclusion

Our project successfully demonstrated a functional prototype of a wireless VR glove system that supports finger tracking and virtual interaction through Unity. By combining Hall effect sensors, BLE communication, and servo-based haptics, we created a hardware and software system capable of mimicking finger movement in a virtual environment. The glove integrates smoothly with Unity and demonstrates real-time response with virtual hand models, setting the foundation for immersive VR input systems. We were also able to successfully implement basic haptic functionality. Specifically, pressing a physical button on the glove activates the servo motor system, locking the fingers to simulate the sensation of grasping an object. However, due to time constraints, we were unable to complete the Unity integration needed to trigger this locking behavior based on virtual object collisions. This remains one of the key features we would have liked to fully implement. Several other components remain incomplete or in need of refinement. In particular, we had plans to implement onboard position tracking using IMUs and sensor fusion, and introduce splay tracking for more anatomically accurate finger positioning. SteamVR compatibility is also an important future step for broader usability. On the software side, improvements such as refining collider interactions and expanding the test scene complexity would further improve realism. Future iterations should consider redesigning the PCB and reevaluating microcontroller selection.

From an ethical standpoint, we are committed to the principles outlined in the IEEE Code of Ethics[14]. We have strived to ensure the safety, accuracy, and transparency of our design decisions throughout this project. We avoided overclaiming performance and clearly documented known limitations, respecting the responsibility of engineers to contribute to society and avoid harm. In terms of broader impact, this project sits at the intersection of accessibility, affordability, and innovation in VR technology. By building on open-source designs and reducing reliance on traditional VR controllers, our glove has the potential to make immersive VR experiences more natural and inclusive. With further development, this technology could benefit education, rehabilitation, remote collaboration, and gaming, contributing positively to global technological engagement and social inclusion.

References

[1] J. Jerald, *The VR Book: Human-Centered Design for Virtual Reality*. New York, NY: ACM Books / Morgan & Claypool, 2015, ISBN: 978-1970001129.

[2] Facebook, Inc., *Facebook to acquire oculus*, https://about.fb.com/news/2014/03/facebook-to-acquire-oculus/, Press release, Mar. 2014.

[3] Meta Platforms, Inc. “Celebrating 10 years of reality labs.” Accessed 2025-05-07. (Apr. 2024), [Online]. Available: https://about.fb.com/news/2024/04/celebrating-10-years-of-reality-labs/.

[4] Valve Corporation. “Valve index controllers.” (2019), [Online]. Available: https://store.steampowered.com/app/1059550/Valve_Index_Controllers/.

[5] Apple Inc. “Introducing apple vision pro: Apple’s first spatial computer.” (Jun. 2023), [Online]. Available: https://www.apple.com/newsroom/2023/06/introducing-apple-vision-pro/.

[6] LucidVR. “Lucidgloves — diy vr haptic gloves.” (2021), [Online]. Available: https://github.com/LucidVR/lucidgloves.

[7] Udexreal. “Udcap, silk-like vr gloves for steamvr.” (2025), [Online]. Available: https://www.kickstarter.com/projects/udexreal/udcap-silk-like-vr-gloves-for-steamvr.

[8] Texas Instruments, “Angle measurement with multi-axis hall-effect sensors,” Texas Instruments, Tech. Rep. SBAA463A, 2023. [Online]. Available: https://www.ti.com/lit/pdf/sbaa463.

[9] Texas Instruments, “Absolute angle measurements for rotational motion using hall-effect sensors,” Texas Instruments, Tech. Rep. SBAA503, 2024. [Online]. Available: https://www.ti.com/lit/pdf/sbaa503.

[10] Espressif Systems, “Esp32-s3-wroom-1 & esp32-s3-wroom-1u datasheet,” Tech. Rep., 2021. [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32-s3-wroom-1_wroom-1u_datasheet_en.pdf.

[11] Microchip Technology Inc. “Bluetooth low energy connections.” (2023), [Online]. Available: https://developerhelp.microchip.com/xwiki/bin/view/applications/ble/introduction/bluetooth-architecture/bluetooth-controller-layer/bluetooth-link-layer/Connections/.

[12] Unity Technologies. “Unity xr interaction toolkit manual.” (2025), [Online]. Available: https://docs.unity3d.com/Manual/xr_input.html.

[13] Components101. “Mg90s — metal gear micro servo motor.” (2018), [Online]. Available: https://components101.com/motors/mg90s-metal-gear-servo-motor.

[14] IEEE. “”IEEE Code of Ethics”.” (2016), [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html (visited on 02/08/2020).

Appendix A Requirements and Verification Table

#	Requirement	Verification
1	Battery must regulate 7.4V LiPo down to 5V and 3.3V with minimal ripple (±5% tolerance).	Measure output voltages with oscilloscope under idle and load conditions (servos moving, sensors reading). Confirm 5.00V ± 0.25V and 3.30V ± 0.15V.
2	Dual Hall Effect sensors must produce independent voltage signals that correlate to finger flexion angles (±5° accuracy).	Move finger known angles using a protractor. Capture sensor outputs. Fit voltage vector to angle mapping. Confirm measured vs expected flexion within ±5°.
3	ESP32 must be successfully communicated with over a programming/debugging interface and respond to commands.	Program ESP32 to blink its onboard LED using a test script. Observe LED blinking to confirm that the ESP32 is powered, programmable, and responding to communication.
4	Servos must provide sufficient torque to lock the user's fingers in place, enabling realistic haptic feedback during interaction with virtual objects.	Apply external force to a locked finger and confirm that the servos can hold position against a small applied force, simulating physical contact in virtual environments.
5	ESP32 must maintain Bluetooth BLE connection and transmit flexion data.	Set up terminal and confirm reception of updated flexion data.
6	Multithreaded BLE-to-Virtual Serial Bridge must correctly translate BLE packets into COM port-readable data with latency less than 50 ms.	Send known pattern over BLE. Capture it on PC COM port. Measure time between send and receive. Confirm latency 50 ms.

Appendix B Gallery of Key Components

This appendix contains an image gallery of the final product, as well as images of the PCBs and their layout.

B.1 VR Glove Final Build

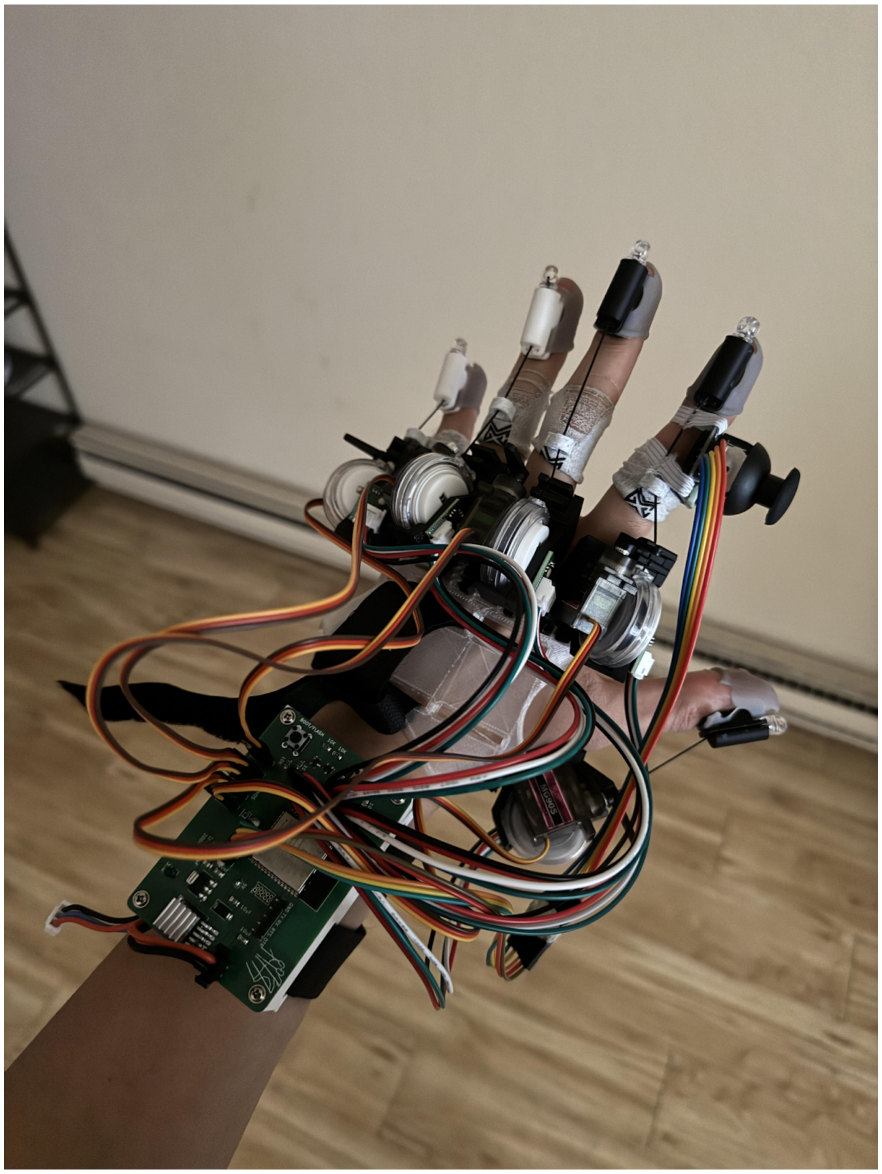

Figure 6: Glove Final

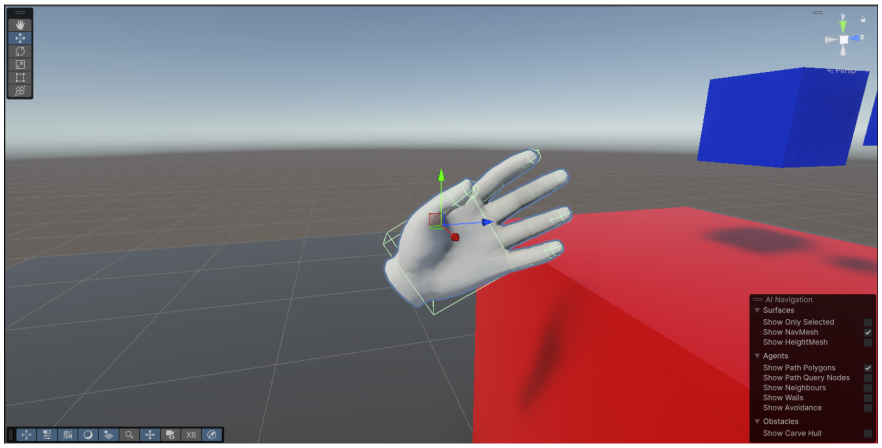

Figure 7: Unity Scene Containing Virtual Hand with Colliders

B.2 Main PCB

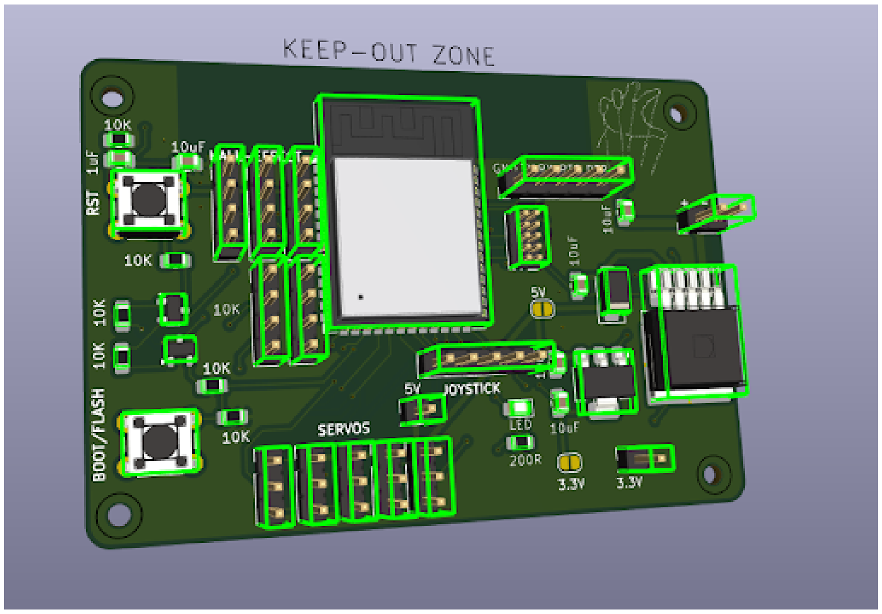

Figure 8: Main Board 3D View

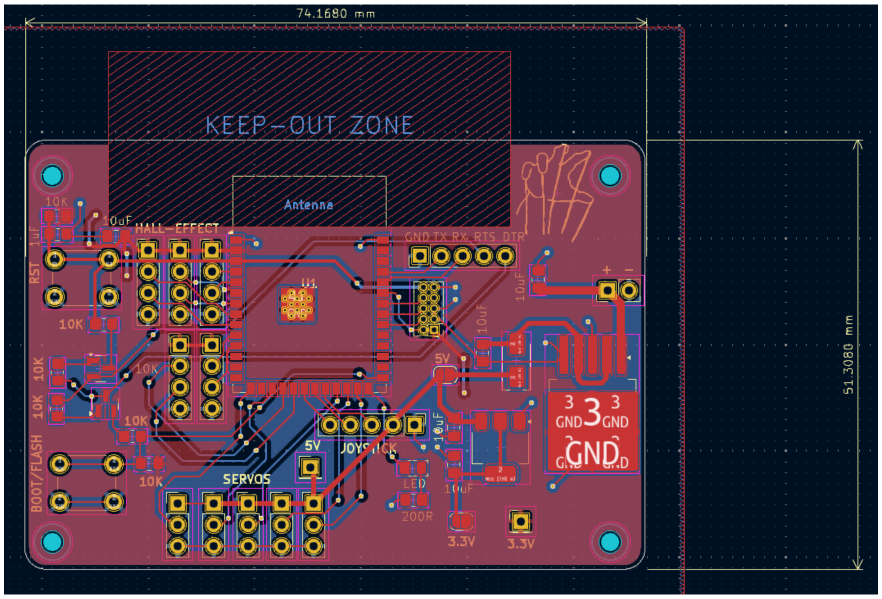

Figure 9: Board Layout

B.3 Hall Effect PCB

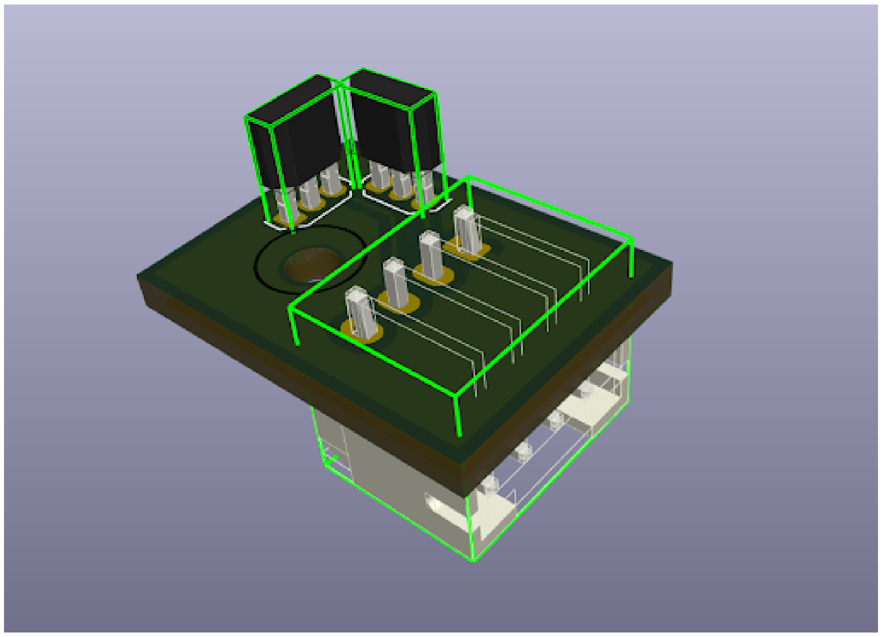

Figure 10: Hall Effect Board 3D View

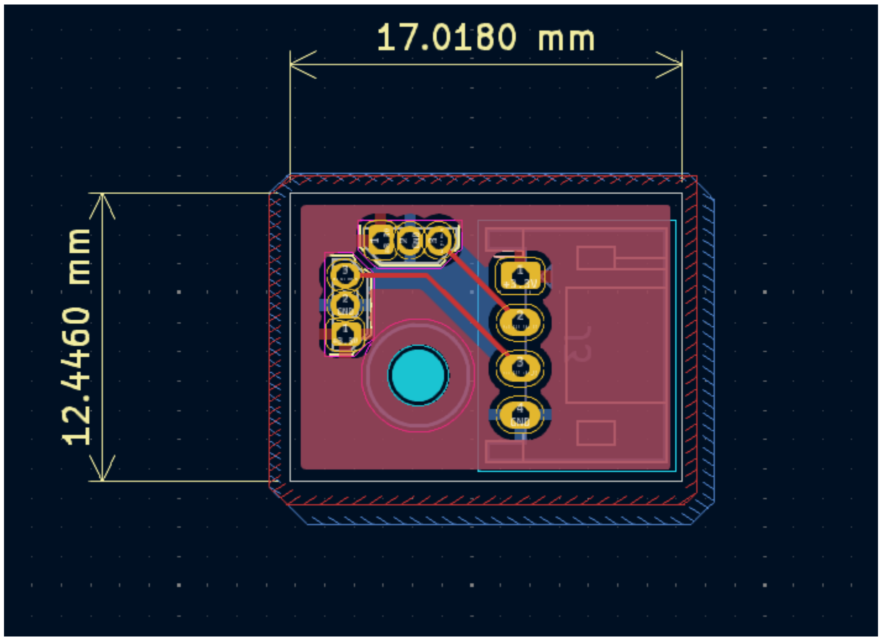

Figure 11: Hall Effect Layout

B.4 Bluetooth Architecture

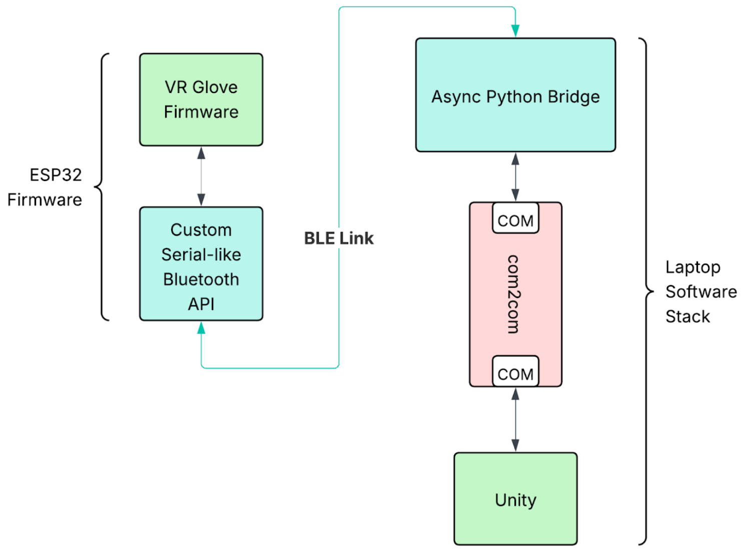

Figure 12: Bluetooth Bridge Architecture

Appendix C CAD Drawings of 3D Printed Parts

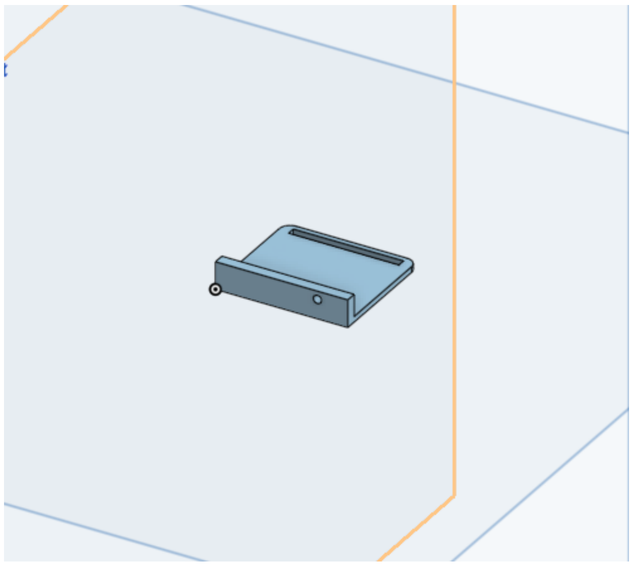

Figure 13: Board Clip

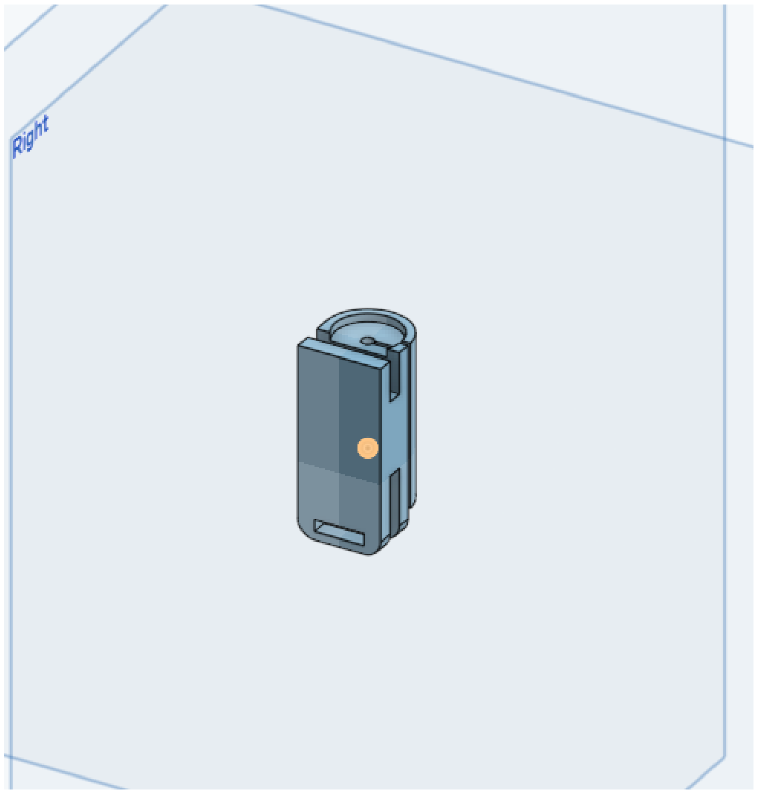

Figure 14: Finger Clip

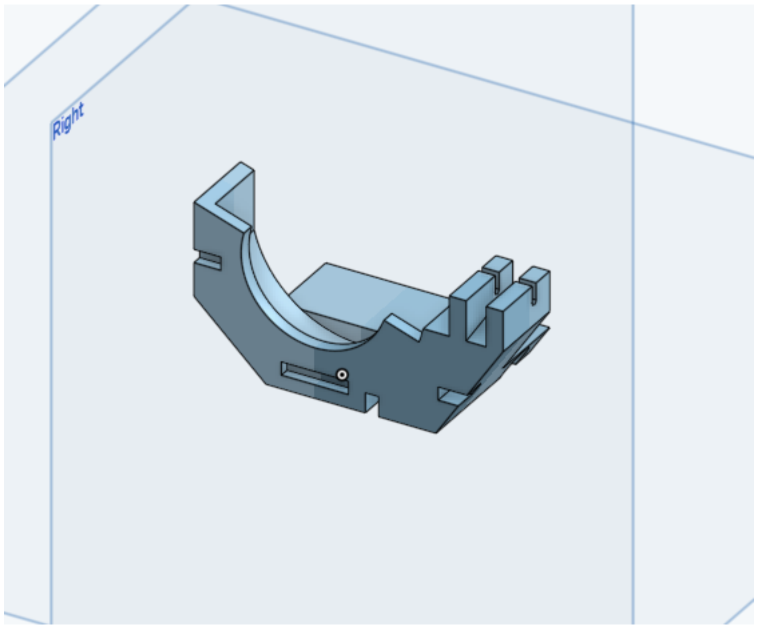

Figure 15: Finger Module

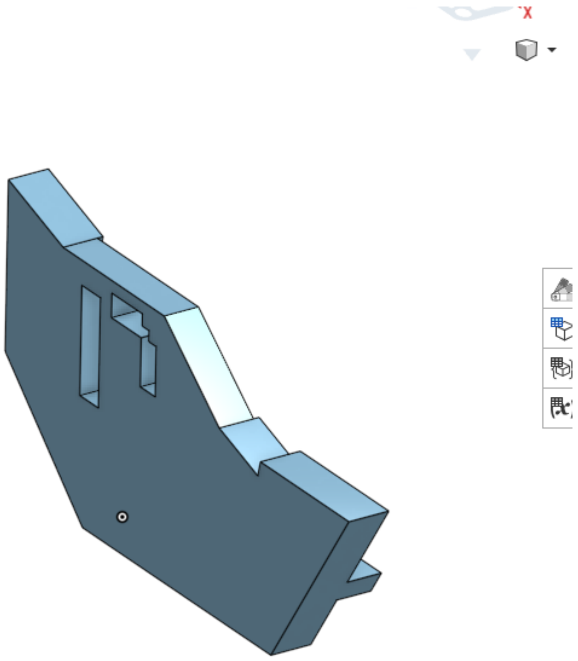

Figure 16: Hall Holder

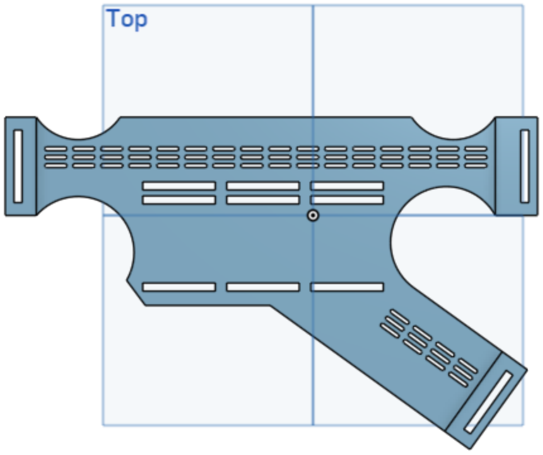

Figure 17: Hand Wrap

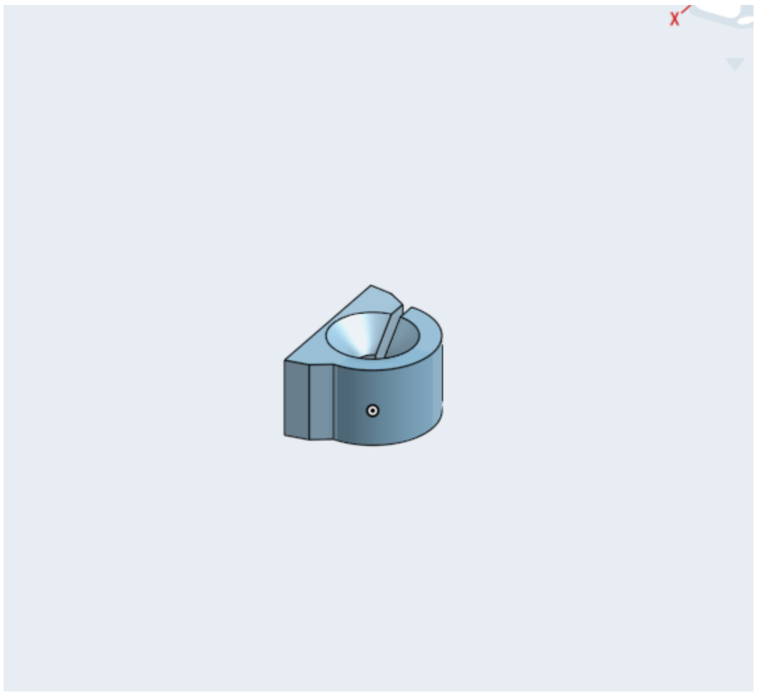

Figure 18: Line Guide

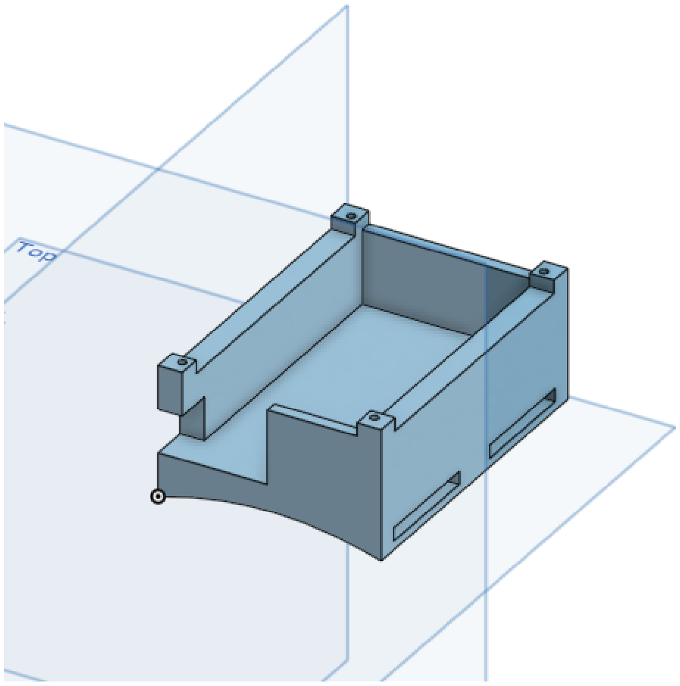

Figure 19: PCB Mount

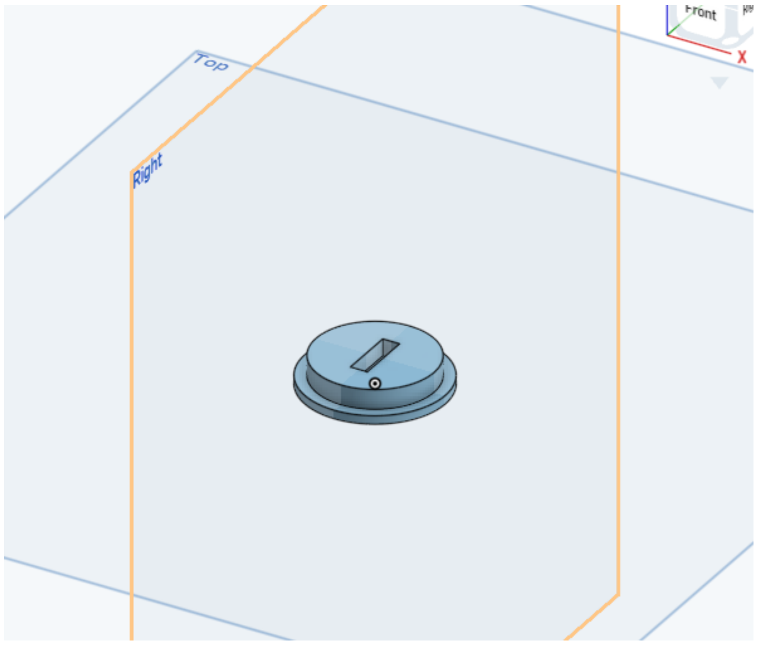

Figure 20: Reel Coin