Hacking a Power Wheelchair

A Design Project Report
Presented to the School of Electrical and Computer Engineering of Cornell University
in Partial Fulfillment of the Requirements for the Degree of
Master of Engineering, Electrical and Computer Engineering

Submitted by Peng-Ru Lung
MEng Field Advisor: Van Hunter Adams
Degree Date: May 2025

Abstract

Master of Engineering Program

School of Electrical and Computer Engineering

Cornell University

Design Project Report

Project Title: Hacking a Power Wheelchair

Author: Peng-Ru Lung

Abstract: This project enables remote and autonomous control of a power wheelchair by spoofing its internal joystick communication system. The team analyzed the analog signals generated by the wheelchair's joystick and then built a microcontroller-based device that emulated these signals. The system integrates a Raspberry Pi Pico, an external Digital-to-Analog Converter (DAC) for analog signal output and a serial interface for user input. In anticipation of future use cases, the team also implemented an audio localization capability using three microphones and the same microcontroller. This approach offers a creative method for enhancing wheelchair control and accessibility.

Executive Summary

This project explores a creative method to enable remote and autonomous control of a power wheelchair by spoofing its joystick interface. Motivated by a desire to create more dynamic interactions with the wheelchair housed in the Mechanical Engineering Motion Capture Studio Lab, we developed a microcontroller-based system using the Raspberry Pi Pico and an external DAC to emulate the analog signals generated by the joystick. A serial interface allows users to send movement commands directly from a keyboard, and oscilloscope testing was used to validate signal accuracy. To enable users to summon the wheelchair using sound cues, a sound localization module was implemented that estimates sound direction using three microphones and Direct Memory Access (DMA) channels.

Our project was divided across two semesters, Fall 2024 and Spring 2025. In the Fall semester, I focused on familiarizing myself with the Raspberry Pi Pico microcontroller, implementing waveform generation using direct digital synthesis, and experimenting with external devices such as DACs. I also explored Serial Peripheral Interface (SPI) protocols and captured Analog-to-Digital Converter(ADC) signals through microphones. With this foundation, I developed a voltage signal system capable of modifying voltage outputs via keyboard input to simulate joystick signals, preparing the system for further testing and implementation.

In the Spring semester, efforts shifted toward reverse-engineering the communication protocol between the joystick and motor controller, using tools such as oscilloscopes and multimeters to analyze signal paths and socket connections. We also integrated a sound localization module using three microphones and DMA channels to estimate sound direction, enabling sound-based wheelchair summoning.

After two semesters of intensive work and testing, although the emulated system triggered a "joystick error" on the wheelchair, we have gathered detailed signal logs and test results that provide a solid foundation for further debugging. We are currently reaching out to the manufacturer to obtain additional technical details for further adjustments. Additionally, the sound localization system has been successfully implemented, though it is occasionally subject to interference from noise. This project highlights the significant potential of embedded systems in creating flexible, accessible control interfaces for mobility devices.

Introduction

This project aims to enhance the autonomy and accessibility of a power wheelchair by reverse-engineering its joystick interface and developing an embedded system for remote control. By analyzing the joystick's analog signal behavior and communication pathways, we were able to replicate its control commands using a Raspberry Pi Pico and an external DAC. The system allows users to send movement commands through a serial interface, offering a new way of interacting with the wheelchair. In addition, sound localization technology was integrated using three microphones and DMA channels, opening the possibility for sound-based wheelchair summoning. This work demonstrates the potential of embedded systems to create adaptable and user-friendly control solutions for mobility devices.

Background

Power wheelchairs are essential mobility tools for individuals with limited motor functions. While modern models offer joystick-based manual control, users with certain disabilities or specific needs may require alternative control methods. With the rise of low-cost, high-performance microcontrollers and embedded systems, there is growing potential to enhance or replace traditional interfaces with more flexible, accessible, and autonomous solutions.

In the Mechanical Engineering Motion Capture Studio, a traditional power wheelchair is available for exploration and testing. Leveraging the Raspberry Pi Pico—a low-cost microcontroller with robust functionality and comprehensive documentation—we set out to creatively integrate these two resources. Our goal is to develop innovative control methods that demonstrate the potential of embedded technologies in assistive robotics. Additionally, from an educational perspective, enabling remote control of the wheelchair opens up new opportunities for mechanical engineering students. It allows them to design and generate control sequences that can be directly tested on real hardware, enhancing hands-on learning and experimentation.

Problem Statement

To enable remote and autonomous control of a power wheelchair, it is essential to understand how the joystick communicates with the motor controller and to develop a system that can accurately emulate this interface. This involves reverse-engineering the joystick's analog signaling and replacing it with a microcontroller-based solution. Additionally, to support features like remote summoning, the system must be capable of localizing sound sources, which involves implementing a robust sound direction detection mechanism.

Issues to Address

To achieve our goal, we need to address the following issues:

- Reverse-engineering the joystick signal protocol without damaging the original system

- Generating precise signals that accurately mimic the joystick's output.
- Ensuring communication compatibility with the motor controller
- Implementing a reliable sound-based localization system to estimate the sound direction for features like remote summoning.
- Integrating both systems to enable remote wheelchair summoning triggered by a clap sound.

Design - Fall 2024

In the Fall semester, I focused on learning how to program the RP2040 Pico and exploring its architecture and capabilities. This foundation allowed me to develop the necessary background knowledge for generating emulated signals and implementing sound detection. Through this process, I learned how to use an external DAC to output precise voltage signals, as well as how to control these signals via a keyboard using the SPI protocol.

Generating Triangle Wave

To generate the triangular wave, I connected a Raspberry Pi Pico to an external DAC using SPI communication. I precomputed the triangular waveform values and stored them in an array. A timer-based Interrupt Service Routine (ISR) was configured to periodically update the output using Direct Digital Synthesis (DDS).

DDS is a technique used to generate precise waveforms by incrementing a phase accumulator by a fixed amount every clock cycle. The output of the phase accumulator is used as an index to retrieve the corresponding value from the precomputed triangular waveform table. This allows the system to efficiently retrieve the correct value corresponding to the current phase.

Each time the ISR is triggered, it increments the phase accumulator, retrieves the corresponding waveform value from the triangle waveform table, and sends that value to the DAC via an SPI transaction. This produces a smooth, continuous triangular waveform. The wiring diagram is shown below for reference.

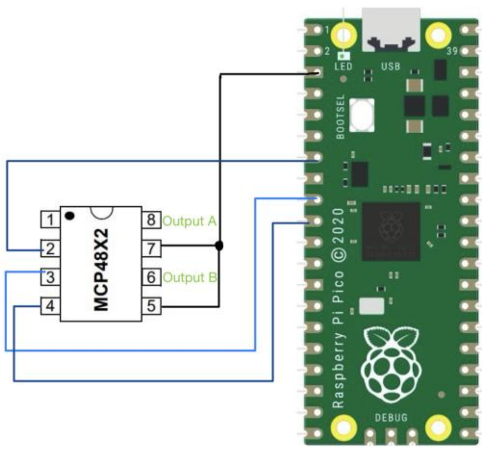


Fig. 1: Wiring Diagram

Design a constant voltage signal based on keyboard input

After getting familiar with the Raspberry Pi Pico, I started building a system that allows user-defined input via a keyboard, bringing it closer to my intended goal. First, I added a wire to connect the serial input to the Pico's Universal Asynchronous Receiver/Transmitter (UART) receive pin, enabling it to receive user input from the keyboard. The Raspberry Pi Pico receives an integer via UART, which it interprets as a user-defined amplitude or waveform setting. Using DDS, the Raspberry Pi Pico generates a corresponding signal based on the user input and outputs it through the DAC. The wiring diagram below illustrates the setup for reference.

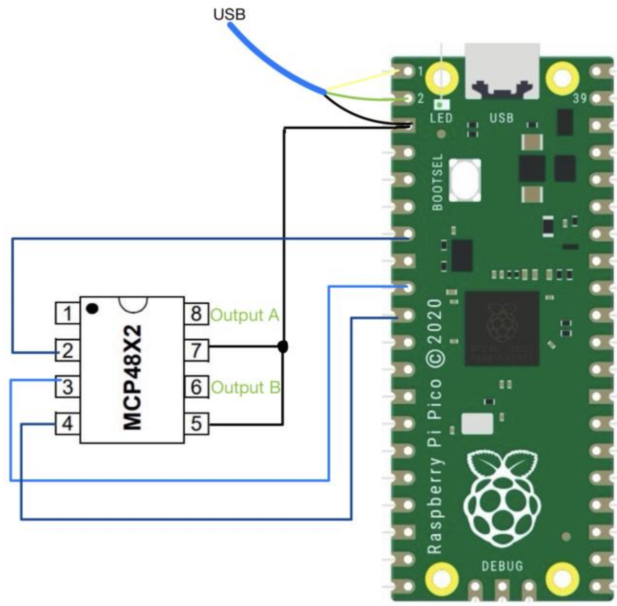


Fig. 2: Wiring Diagram

Testing and Results - Fall 2024

Generating Triangle Wave

To verify that the output met my expectations, I used an oscilloscope to observe the waveform and confirm that the signal matched what I expected.

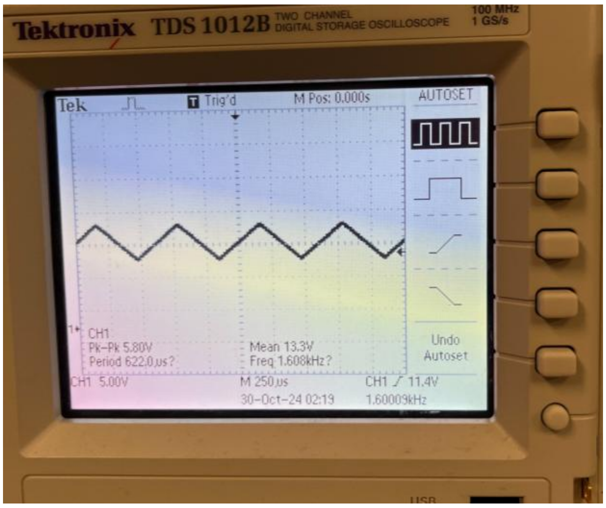
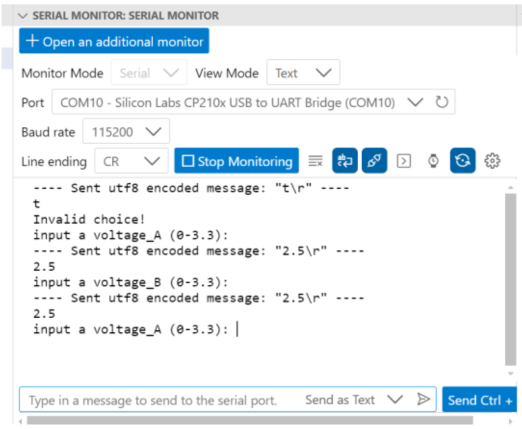


Fig. 3: Oscilloscope with Triangular Wave

Design a constant voltage signal based on keyboard input

To verify that the output met expectations, I used an oscilloscope to observe the generated waveform and confirm it matched the intended signal characteristics. Additionally, I tested the serial interface by sending different user commands to adjust the voltage levels in real-time. The output signals consistently reflected the expected changes, demonstrating both the accuracy of the waveform generation and the reliability of the user control mechanism.



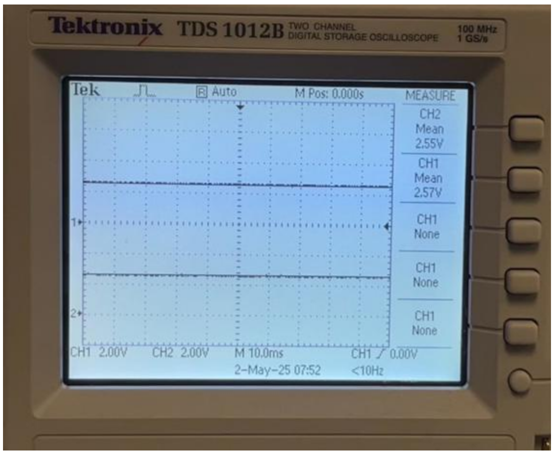


Fig. 4(left): Serial Interface for Real-Time Voltage Adjustment
Fig. 5(right): Oscilloscope Output Responding to User-Controlled Signal Changes

Design - Spring 2025

In the Spring semester, I began reverse-engineering the joystick's communication protocol, identifying how the signals are transmitted between the joystick and the motor controller. This led to the development of an emulated joystick system, which I tested on the wheelchair to ensure compatibility. Additionally, I initiated the sound localization system by setting up three microphones and reading their values through the ADC. I also configured

DMA channels to process the microphone data, performing computations to estimate the sound direction.

The Hacking Process

The first step in the process was to carefully disassemble the joystick to access its internal components and observe both the wiring and the circuit board layout. This allowed us to trace and identify the specific wires responsible for transmitting control signals from the joystick to the motor controller. The left image below shows the power wheelchair located in the Mechanical Engineering Motion Capture Studio. The red rectangle highlights the joystick module, which we aim to reverse-engineer and replace with our custom emulation system. The right image displays a close-up of the original joystick that we intend to replicate and replace through signal emulation.

Fig. 6(left): Power wheelchair in the Motion Capture Studio
with the joystick highlighted in a red rectangle
Fig. 7(right): Close-up of the joystick targeted for replacement

With the internal connections exposed, we could begin analyzing how the joystick communicated with the wheelchair system, which was essential for understanding and ultimately replicating the joystick's communication with the power wheelchair's motor controller. An oscilloscope was used to characterize the solder pads on the joystick's circuit board. The oscilloscope allowed us to visualize the voltage levels and patterns, helping us determine which pins on the board were responsible for transmitting the control signals between the joystick and the wheelchair. The image below shows the internal view of the joystick with our assigned pin numbers. The accompanying table describes the function of each identified pin along with its typical voltage range during operation.

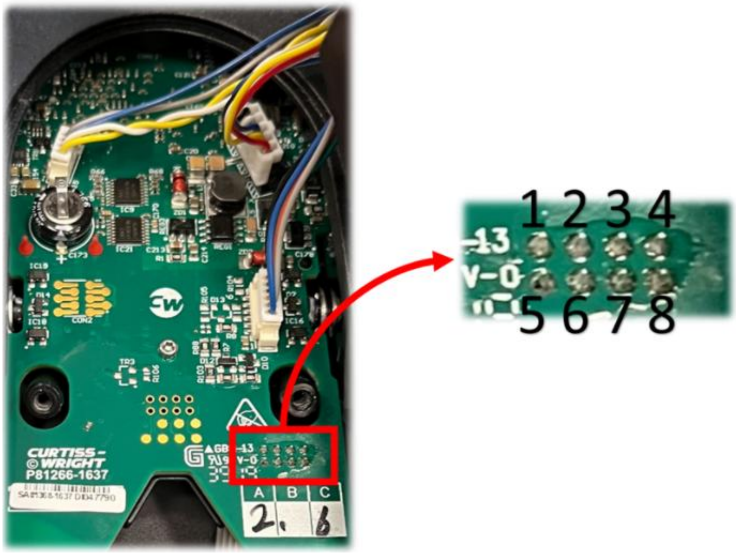


Fig. 8: Internal view of the joystick with labeled pin numbers used for signal analysis

Pin Number	Function	Voltage Range
Pin 1	Ground	0V
Pin 2	Ground	0V
Pin 3	Forward/Backward Signal	1V – 4V (centered at ~2.5V)
Pin 4	Right/Left Signal	1V – 4V (centered at ~2.5V)
Pin 5	Right/Left Signal	1V – 4V (centered at ~2.5V)
Pin 6	Forward/Backward Signal	1V – 4V (centered at ~2.5V)
Pin 7	Ground	0V
Pin 8	Power Supply	5V

Table 1: Pin functions and corresponding voltage ranges observed on the joystick's circuit board

Next, we used a multimeter to trace the wiring on the joystick's circuit board. This helped us match the correct pins to their respective sockets, ensuring that the identified pins corresponded to the correct physical connections. By confirming these connections, we ensured that the signals could be properly manipulated and emulated later in the system design.

Emulating the Joystick

To replace the joystick with my system, it was necessary to accurately emulate the analog voltage signals originally generated by the joystick. I used the Raspberry Pi Pico with an external DAC to replicate these voltage levels with high precision. This hardware setup allowed the system to generate continuous analog outputs that matched the control signals expected by the wheelchair's motor controller.

In addition to signal generation, I implemented a serial input interface that accepts keyboard commands from a computer. This feature enables users to remotely issue directional instructions such as forward, backward, left, or right. The Raspberry Pi Pico processes these inputs and converts them into corresponding analog voltage signals, effectively simulating joystick behavior in real time. This system not only replicates manual control but also establishes the foundation for autonomous or alternative input methods in future extensions of the project.

The software architecture is illustrated below. It operates with a single main thread responsible for continuously reading serial input from the user. When a new command is received, the main thread updates global variables which will be sent to the DAC. These updated values are applied through an Interrupt Service Routine, which performs SPI transactions to ensure timely and accurate signal generation. This architecture enables responsive and smooth control over the wheelchair's movement. The wiring diagram is shown in Fig.10.

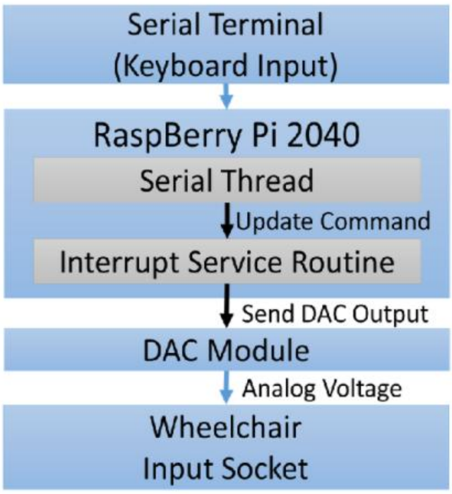


Fig. 9: Software architecture of the emulated system

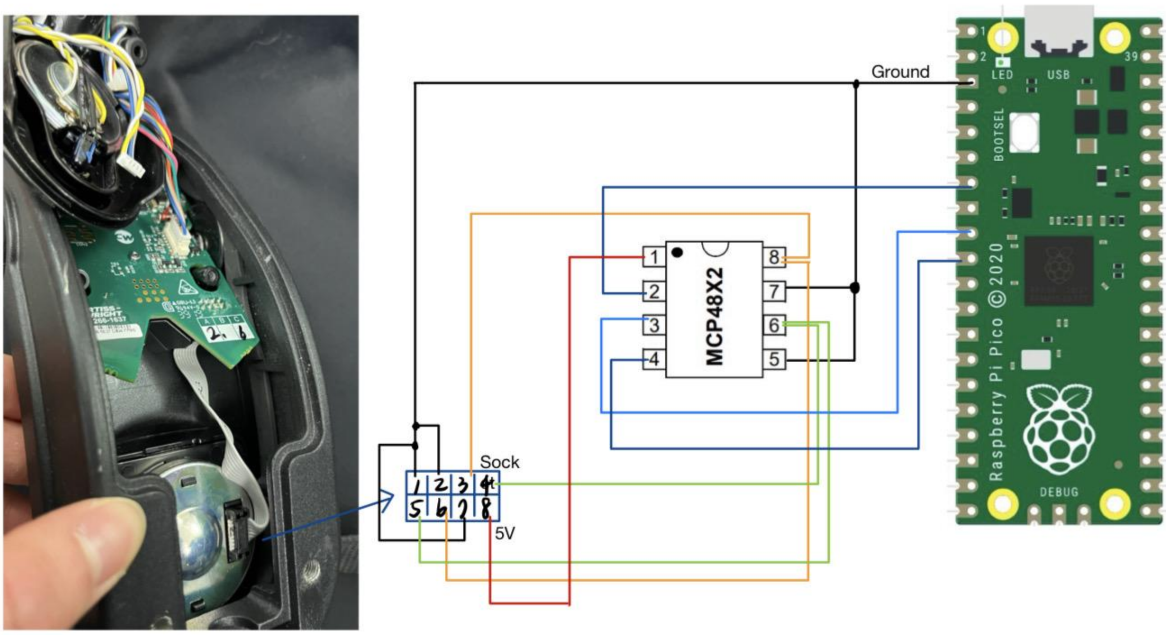


Fig. 10: Wiring Diagram for emulated system

Sound localization

To complement the remote and autonomous control system, a sound localization module was developed using three microphones connected to the Raspberry Pi Pico ADC channels. By configuring DMA (Direct Memory Access) channels, the system was able to continuously sample and buffer audio input from all microphones with minimal CPU overhead, enabling real-time processing.

With the microphone data collected, I computed the cross-correlation as defined by the formula below. Cross-correlation is a signal processing technique that measures the similarity between two signals by sliding one over the other. By analyzing the peak in the cross-correlation output, it is possible to determine the time delay between the arrival of the signal at different microphones. The system can estimate the direction of the sound source by identifying which microphone receives the signal first.

$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f^*[m]\, g[m+n]$$

The microphone and sound source setup is illustrated in Figure 11, while the software architecture for sound localization is shown in Figure 12.

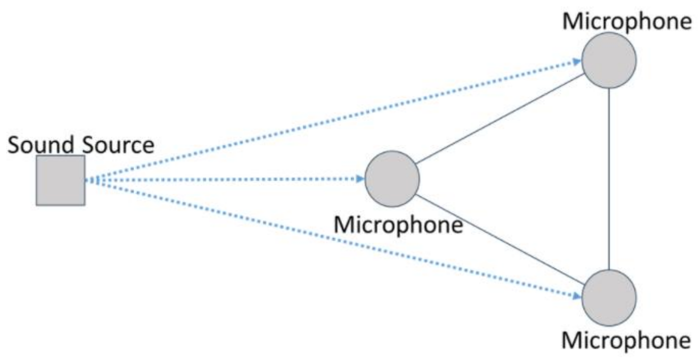


Fig. 11: Microphone and Sound Source Setup

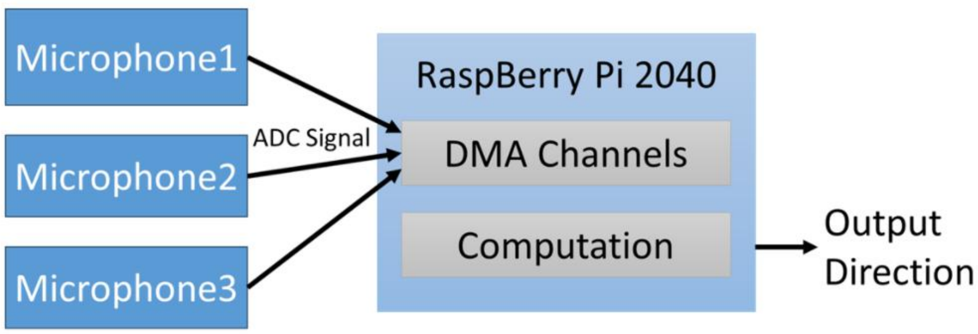


Fig. 12: Software Architecture for Sound Localization

To provide clearer visual feedback on the estimated sound direction, I implemented three LEDs, each corresponding to a specific direction. When the system estimates the direction of the sound source based on the microphone signals and cross-correlation analysis, the corresponding LED lights up to indicate the result. This visual cue allows for quick and intuitive validation of the direction detection algorithm. The wiring configuration for this setup is illustrated in the diagram below. Each LED is connected to a designated GPIO pin on the Raspberry Pi Pico, with a resistor placed in series to protect both the LED and the GPIO pin from excessive current.

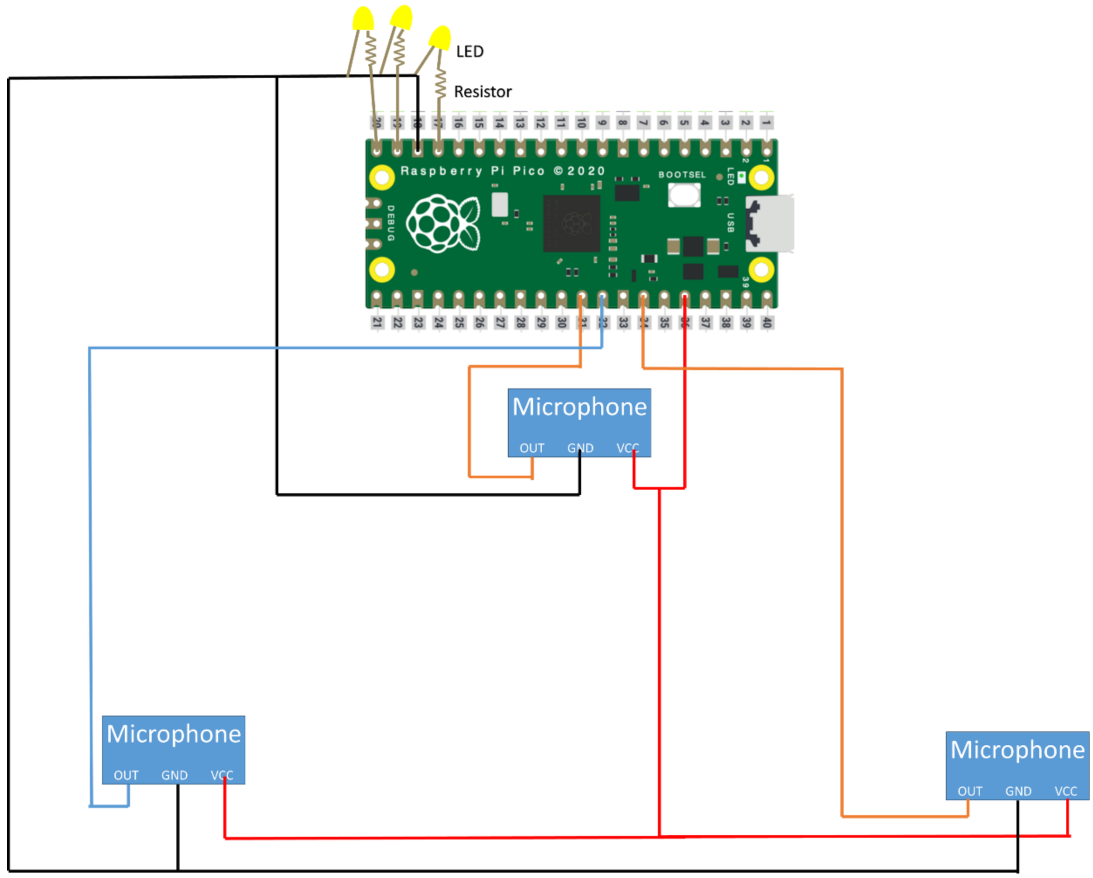
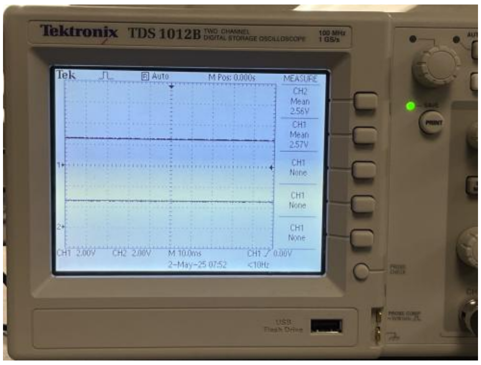


Fig. 13: Wiring Diagram for Sound Localization

Testing and Results - Spring 2025

Emulated System

Upon setting up the emulated system, I initially conducted tests using an oscilloscope to verify the output signals. Additionally, I ensured that the serial interface properly facilitated changes in the voltage signal. The figure below illustrates the oscilloscope output along with a screenshot of the serial interface for reference.



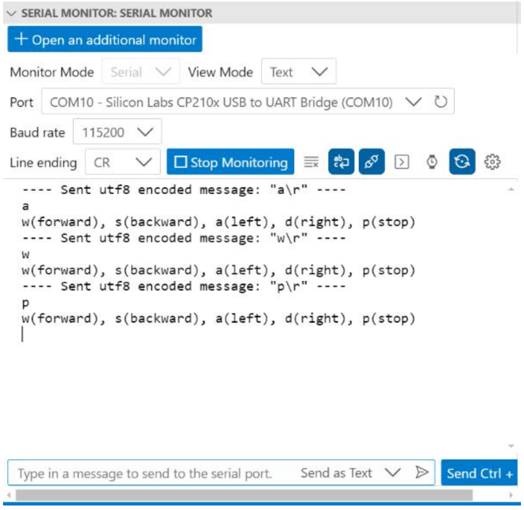


Fig. 14(left): Oscilloscope Output of Emulated System
Fig. 15(right): Serial Interface Command

After confirming that the emulated system was functioning correctly using the oscilloscope, with the waveform matching that of the original joystick signal, I proceeded to wire the system to the actual wheelchair. The image below illustrates the actual wiring setup

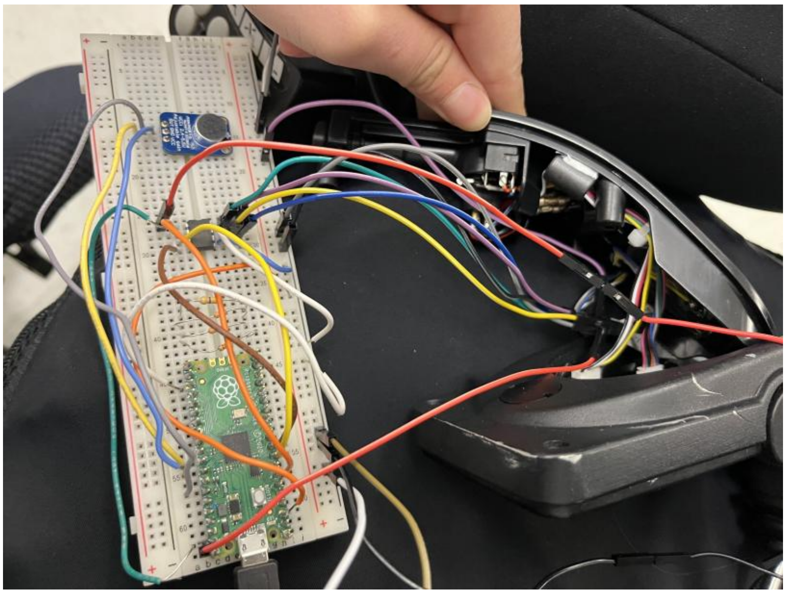


Fig. 16: Actual Wiring Setup to the Wheelchair

However, when the power was turned on, the screen displayed a 'joystick error,' indicating that the wheelchair did not accept our signal, even though the waveform appeared identical to that of the original joystick. Suspecting that the system might also be checking for current draw as part of its validation, I tested the signal using several resistors to simulate a current load. Unfortunately, the result remained the same, and the error persisted.

Fig. 17: Joystick Error

Sound localization

Although the ADC operates at a high internal sampling rate, the data was sampled at 10 kHz to match the frequency range of interest for clap detection, typically between 1000 Hz and 4000 Hz. This sampling rate is sufficient to capture the essential characteristics of the sound while balancing processing efficiency and data storage constraints.

Each microphone channel was calibrated by adding a fixed offset, determined and verified through independent testing, to ensure consistent signal levels across all three inputs.

To estimate the sound direction, cross-correlation was performed between pairs of microphone signals. The central 7/8 segment of one recording was slid across the full length of another, calculating the sum of dot products for each overlap. The sliding window size was determined through empirical testing for optimal results. The peak correlation values and their corresponding indices were then used to infer the direction of the sound source.

To evaluate the performance of the sound localization system, I tested it by clapping from different directions. Since the system uses three LEDs to indicate the detected direction of the sound source, when I tested the system, the corresponding LED was reliably illuminated in response to claps from the associated direction, providing clear and immediate visual feedback. When no sound was present, all LEDs remained off, confirming that the system could correctly identify periods of silence. However, the system occasionally exhibited instability—faint ambient noise or electrical interference would sometimes trigger false positives, causing an LED to turn on despite no intentional sound input.

Conclusion

In this project, I explored the possibility of reverse-engineering the joystick of a power wheelchair and extending its functionality with additional features. By examining the internal wiring and using tools such as an oscilloscope and multimeter, I identified the control signals and their corresponding pins. I then designed and implemented an emulated system, which was first tested on the oscilloscope to check its accuracy and later tested on the actual wheelchair. Although the generated signals closely resembled those from the original joystick, the wheelchair rejected the emulated input, indicating that additional, undisclosed communication mechanisms or validation checks may be in place.

With the detailed documentation and testing results gathered, we are now well-positioned to reach out to the manufacturer or relevant experts for further insights. This could potentially enable successful joystick replacement and pave the way for future enhancements.

For the sound localization system, the core functionality has been implemented. The system is capable of detecting sound direction using microphone input and providing visual feedback via LEDs. However, it requires further refinement and testing to reduce false triggers caused by ambient noise or interference, as these could lead to unintended wheelchair movements.

Overall, this project lays a solid foundation for future development, offering valuable insights into the complexities of embedded systems integration and real-world hardware interfacing.

Appendix

Reference

[1] RP2040 Microcontroller Datasheet [Online]. Available: https://datasheets.raspberrypi.com/rp2040/rp2040-datasheet.pdf
[2] MCP4821/22 12-Bit DAC Datasheet [Online]. Available: https://ww1.microchip.com/downloads/aemDocuments/documents/OTH/ProductDocuments/DataSheets/20002249B.pdf