UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

Modular Desktop Audio Mixer Controller

ECE 445: Senior Design

Group 85

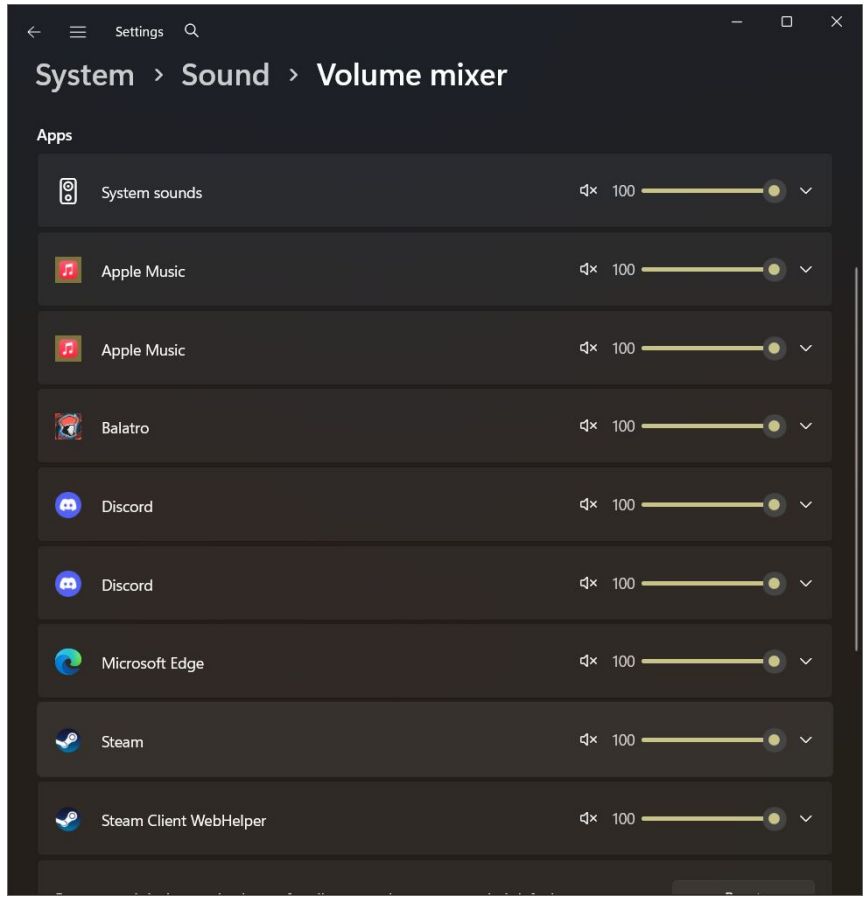
Aarushi Sharma

Dylan Moon

Objective

OS provides audio mixer for volume control of many applications

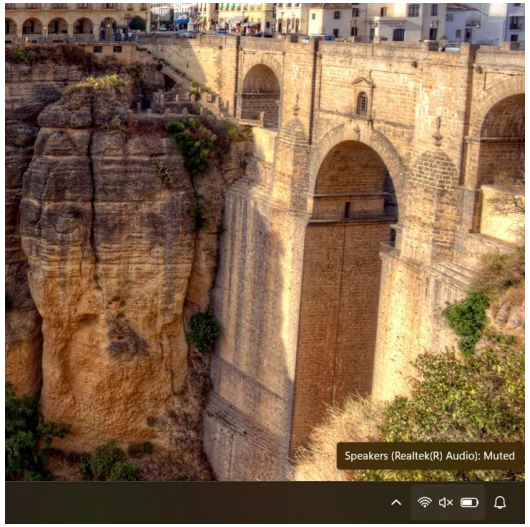
- Power user setup:
  - Application audio in foreground
  - Video call in foreground
  - Music in background
- Gamer setup:
  - Game in foreground (fullscreen)
  - Voice call in background
  - Video on side monitor (YouTube tutorial)
  - Music in background
- **Need to adjust audio volume of apps**

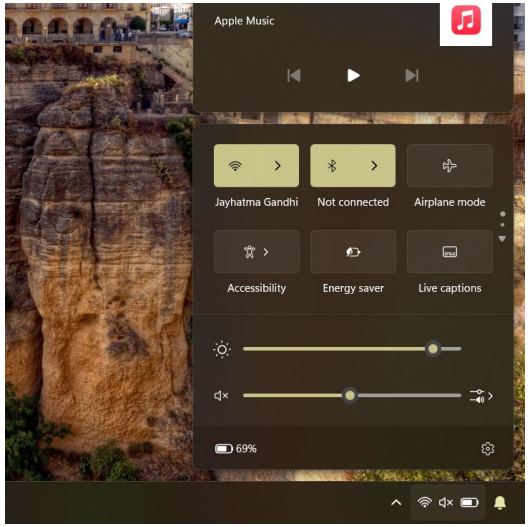


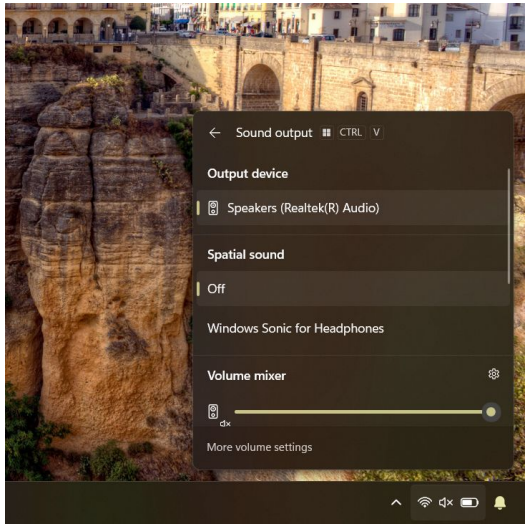
Typical human-computer interaction for desktop computers: the keyboard and the mouse.

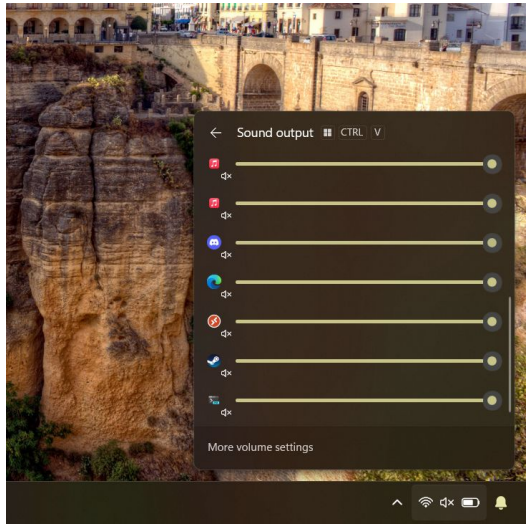
Software audio mixer GUI access isn't straightforward

- Inefficient, obstructive, and distracting
- Buried behind menus









Typical human-computer interaction for desktop computers: the keyboard and the mouse.

Software audio mixer GUI access isn’t straightforward

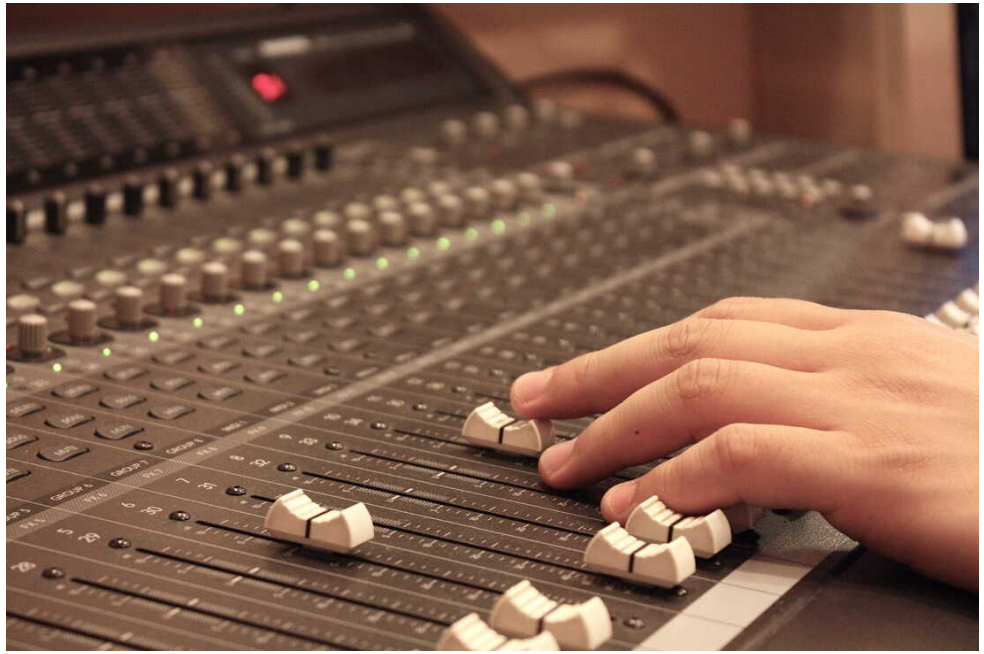
- Inefficient, obstructive, and distracting
- Buried behind menus

Analog control surface to physically interact with:

- Smooth, easy, natural
- Instant physical access
- Save time for power users, avoid interruptions

Audio Faders as Inspiration

- Precise, reliable control interface
- Low latency for bidirectional audio adjustments



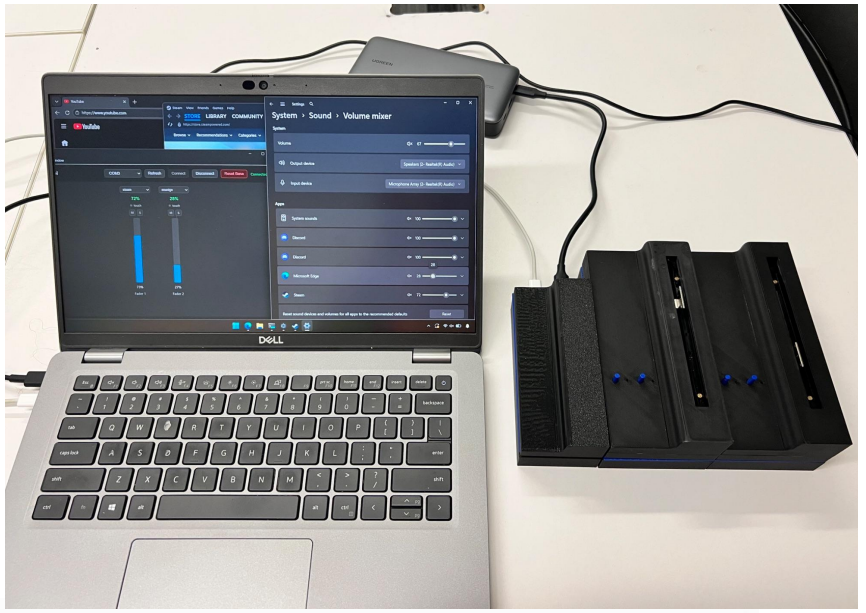
Introduction to a Mixer Control Panel

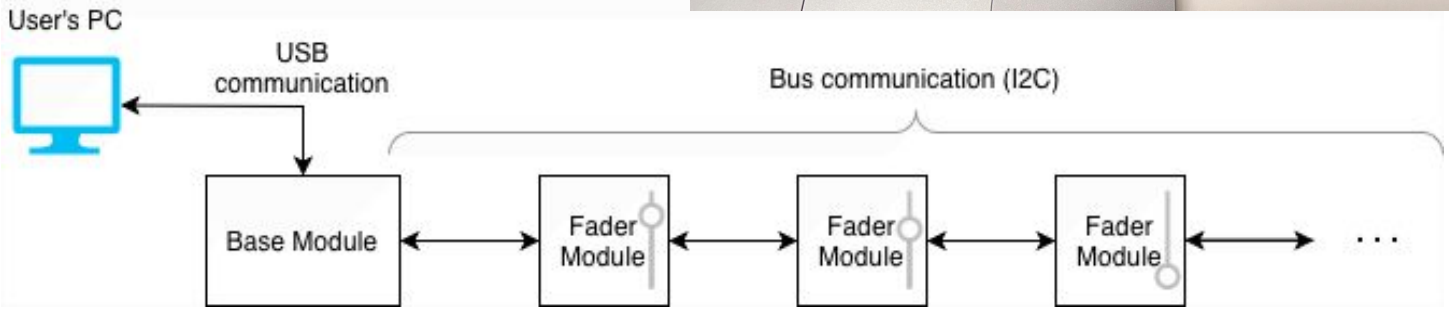
A physical interface for controlling computer audio mixer settings

One base module:
- Connect to computer via USB C
- Drive communication and power for the panel

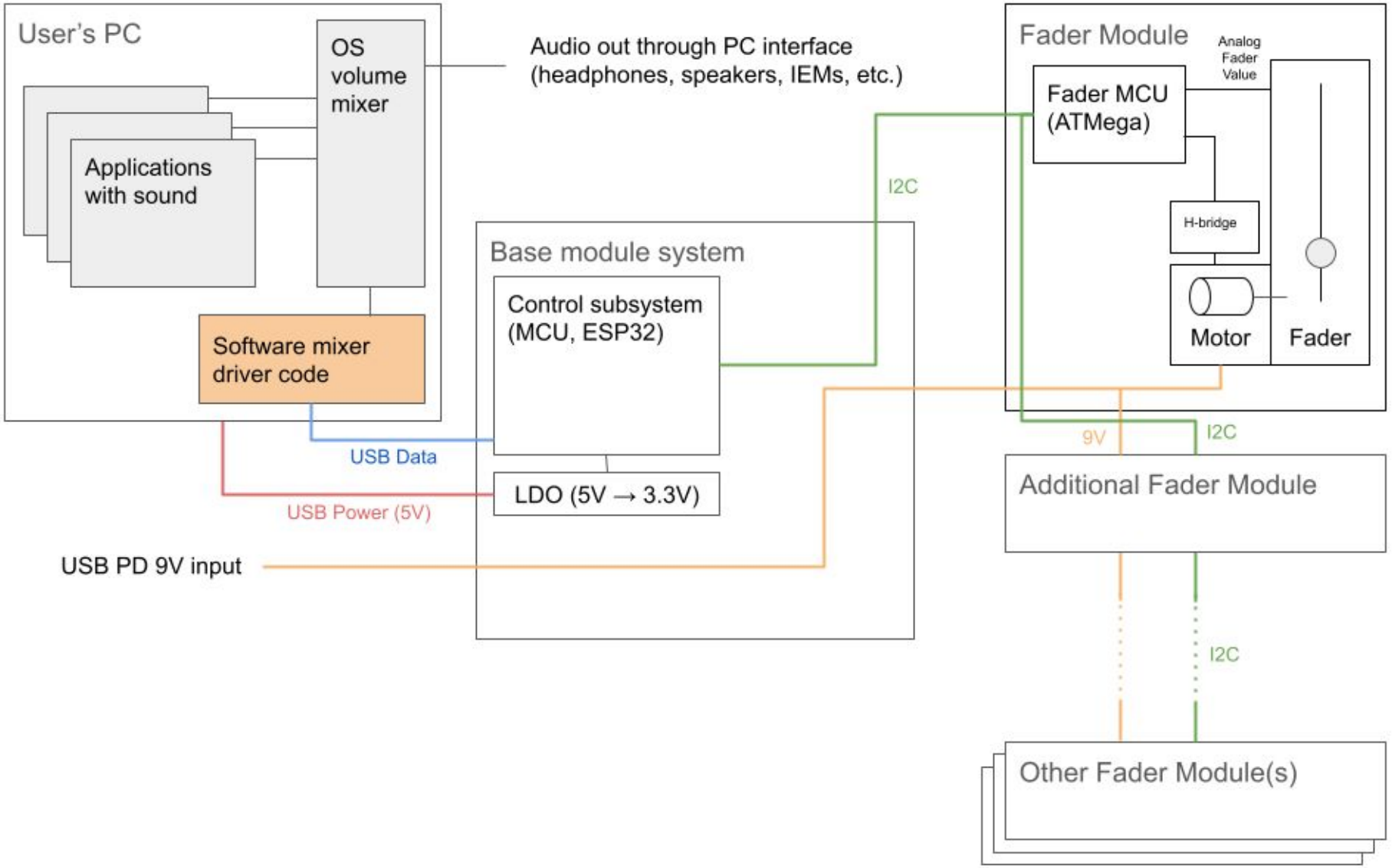
One or more fader modules:
- Magnetically attach as many as desired
- Quick, granular, one-to-one control of individual audio settings

Computer program
- Synchronization between the hardware and the computer's OS
- Setup faders to control certain application's volume





Design: Block diagram



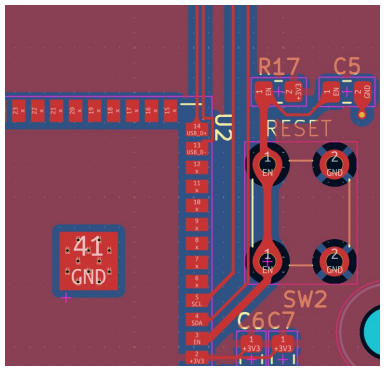
The hardware side of the bridge

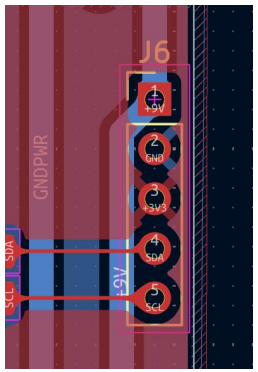
Needs to provide power to the rest of system (connected modules)

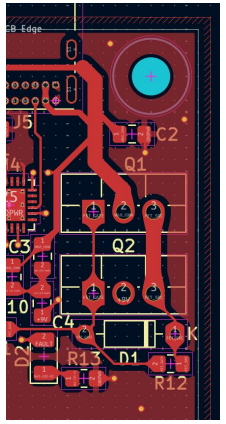
- 3.3V for logic and 9V for high-power components (motor)
- Motor can draw 800mA starting current. Account for 3 motors starting simultaneously
- 2.4A max current. If more motors are required to start at the same time, can offset in firmware
- Required trace width for 2.4A with ΔT of 10℃ is 39.56 ≈ 40 mil
- 80% rule for peace of mind: 50 mil trace width

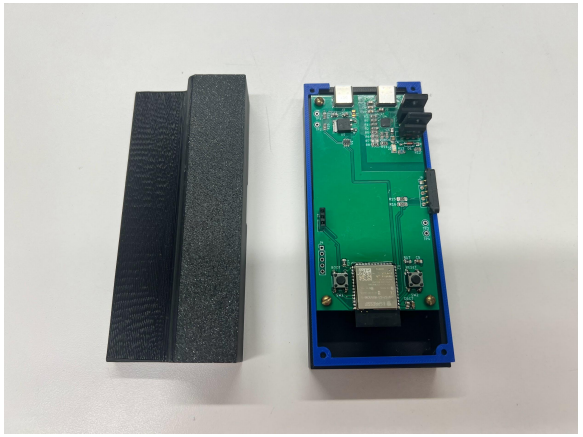
Needs to connect via USB to host PC and provide bus for other components

- ESP32-S3 USB interface
- ESP32-S3 I2C interface on bus



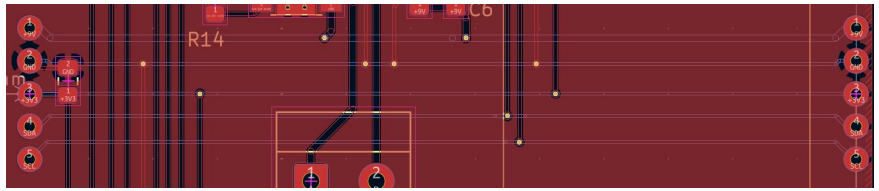






Base Module

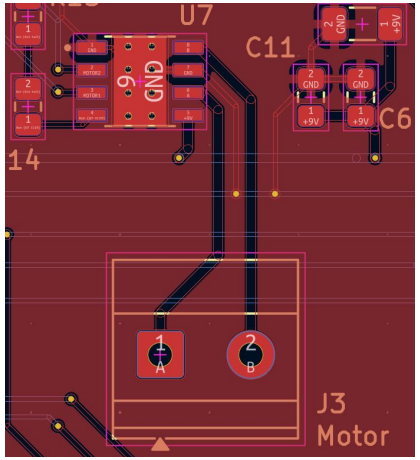
Requirements	Verification
● The Base Module should receive 5V from the USB connection and regulate it to 3.3V and 9V from the external DC source (USB PD) and be able to pass it on through the Pogo pins	● Connect USB and external power source, probe 3.3V and 9V Pogo pins with multimeter. Measurement should be within 2.8V to 3.8V for the 3.3V pin and 8.5 and 9.5V for the 9V pin.
● The Base Module must be able to detect the addition or removal of any Fader Modules from the panel.	● Start with no Fader modules connected. Attach a Fader module. Verify that the software shows the connected Fader module in the configuration screen. Then detach a Fader module. Verify that no Fader modules are present in the configuration screen.
● The Base Module must be able to dynamically manage I2C addresses for connected Fader Modules & inform Fader Modules of their assigned address.	● Implicitly required to detect multiple Fader modules. Start with one Fader Module connected. Attach another fader module. Verify that the Base Module shows two connected Fader modules.

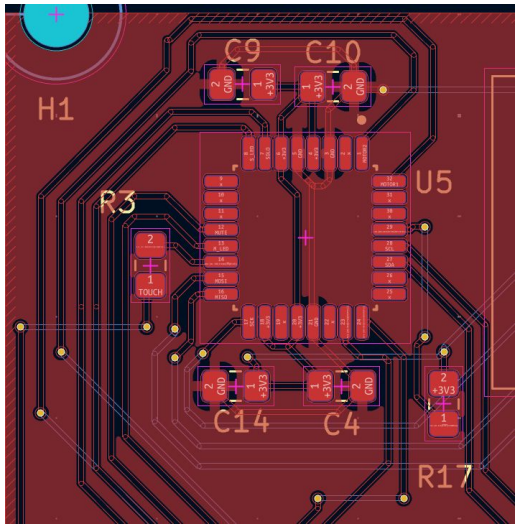


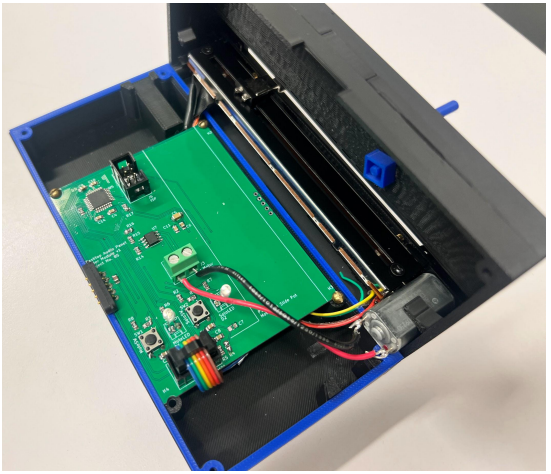
Attachable modules for physical interaction

Motorized linear slider (fader) used to configure an audio setting

- Chained to Base Module or other Fader Modules
- Pogo connections share 12V power and I2C data buses
- On-board ATMega328P manages bidirectional I2C communication
- Manually adjusting the fader updates the computer's software mixer
- Computer-side adjustments update the physical fader's position via its motor
- Simple firmware PD controller







Fader Modules

Requirements	Verification
• Each Fader module must be able to receive messages designated for itself over I2C.	• Adjust the volume in the OS mixer of an application connected to a fader and observe the fader's physical position move to match it.
• Physical adjustment of Faders must be broadcasted to the device over I2C.	• Verify that movement of a Fader is detected by the Base module via debugger or by the OS application
• Computer-side adjustments to audio settings must move corresponding Fader.	• Adjust an audio setting to 50% on the computer. • Verify that assigned Fader module for the setting moved halfway between both ends of the fader.
• All Fader Modules maintain bidirectional continuity over the panel's I2C bus.	• Adjust the volume of an application connected to the last fader and observe that the fader's physical position moves to match it.

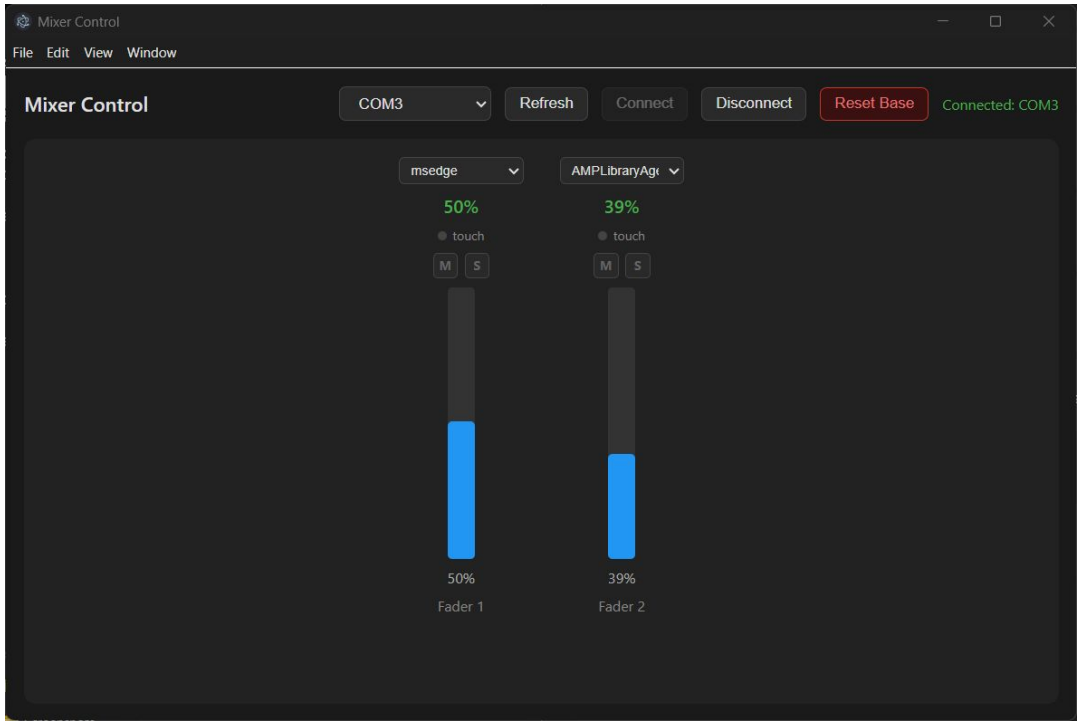
The software side of the bridge

Maintains synchronization

- OS volume mixer settings sync with physical faders
- Bidirectional synchronization

Graphical User Interface (GUI)

- Configure Faders to applications
- Dynamically display connected Faders



Computer Application

Requirements	Verification
• Broadcast audio setting updates from the computer to the Base Module	• Adjustments to an audio setting must be received by the audio panel • Confirm via debugger on the base module • Ensure that adjustments are reflected in Fader position
• Receive Fader position updates from the audio panel and update audio settings	• Confirm that application receives updates from the panel (programmed into the Base module or via movement of a Fader) • Confirm that corresponding computer audio settings are updated
• GUI should be able to configure applications to certain faders	• Connect one Fader module. Assign some audio application to the fader in the GUI. • The App volume should change with Fader movement
• GUI should dynamically display all available Fader Modules	• Connecting and removing a Fader Module to the Base Module should result in the GUI reflecting the Panel Layout within 1 second

Verification Images

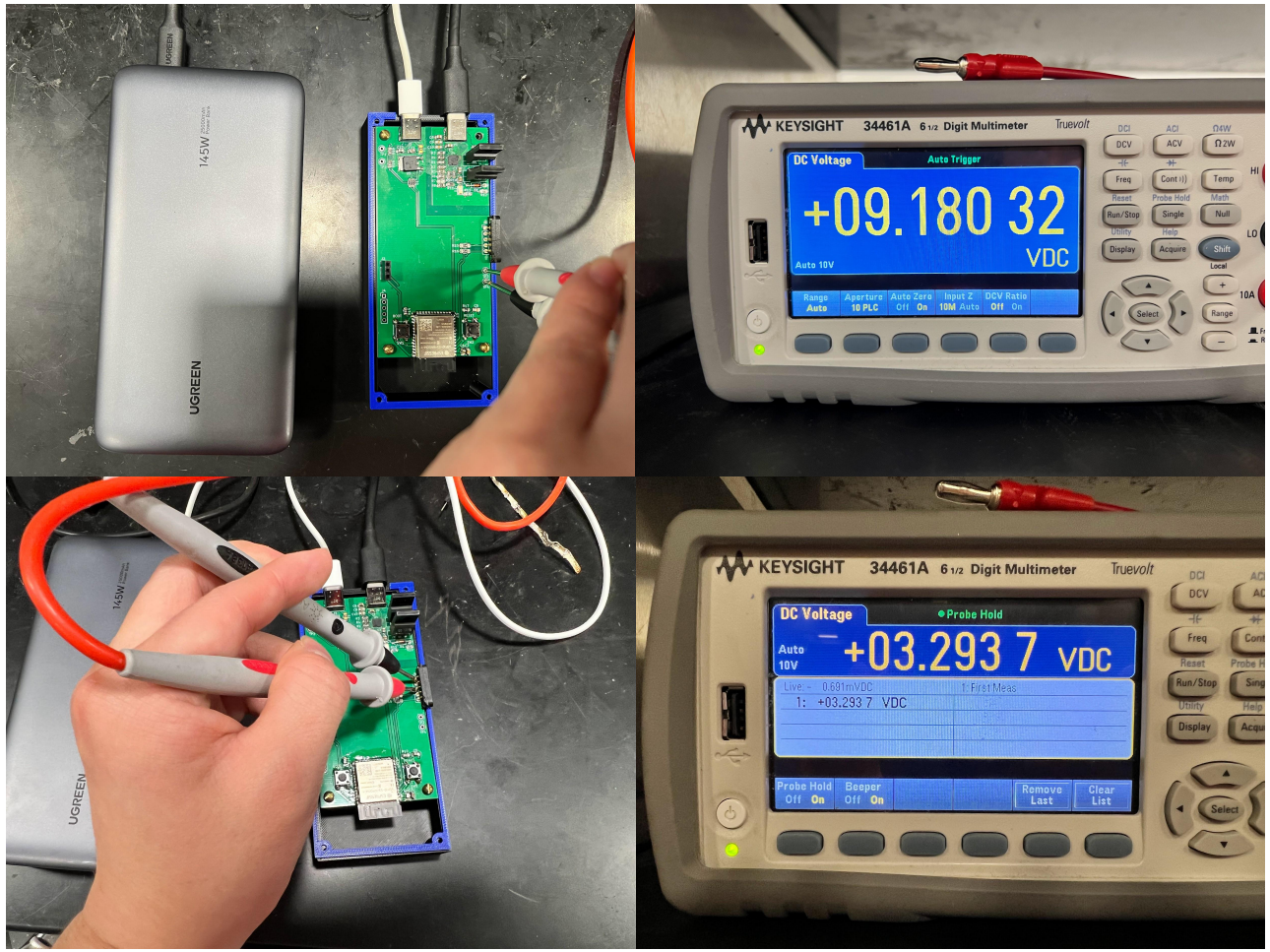
Base: Power supply verification

Top image: 9V

- Measured voltage 9.18V
- Within 8.5V to 9.5V range

Bottom image: 3.3V

- Measured voltage 3.29V
- Within 2.8V to 3.8V range



Verification Images

Base: Connect/disconnect Fader GUI (Software): Dynamically display modules

Top image: No Fader connected

- Base reports no faders connected
- GUI shows no faders connected

Middle image: Connect Fader to Base

- Base reports 1 fader connected
- GUI shows 1 fader connected

Bottom image: Disconnect Fader

- Base reports no faders connected
- GUI shows no faders connected

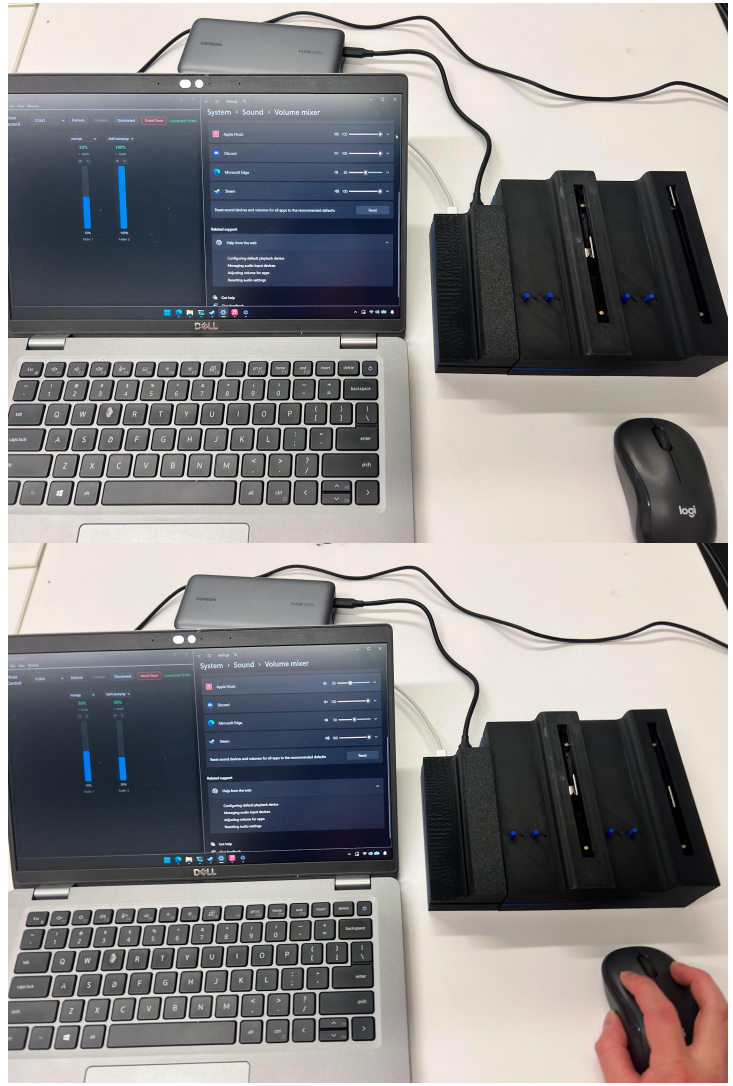
Verification Images

Base: Dynamic I2C (Multiple Faders)

- Base reports 2 faders connected
- GUI shows 2 faders connected
- Dynamic I2C addressing required for both faders to be reported

Verification Images

Fader: Receive I2C messages
Fader: Match OS volume mixer
Fader: Bidirectional continuity
GUI: Send OS mixer updates to Base
GUI: Configure applications to Faders



Top image: Start at 100% volume

- GUI assigned Fader furthest from Base to Apple Music application
- Application volume at 100%, Fader position at 100%

Bottom image: Adjust OS mixer volume to 50%

- GUI reports update to Base
- Base sends I2C message to Fader furthest from Base
- Fader updates physical position to match OS volume setting

Verification Images

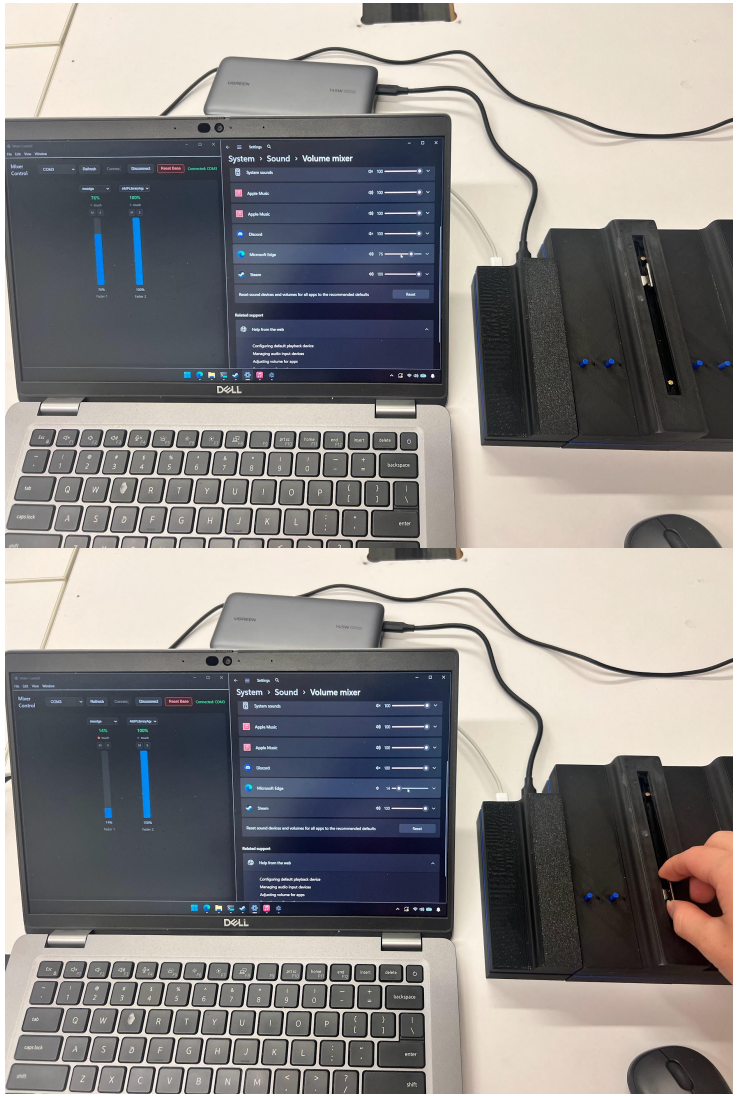
Fader: Broadcast physical adjustments to I2C
GUI: Receive position updates and adjust OS mixer

Top image: Start at 76% volume

- GUI assigned Fader closes to Base to Microsoft Edge application
- Application volume at 76%, Fader position at 76%

Bottom image: Adjust physical fader position

- Fader reports update to Base via I2C
- Base sends message to GUI about position update
- GUI modifies OS volume setting to match fader position



Ethical Considerations

ACM principle 1.6: "informed consent for automatic data collection, and to review, obtain, correct inaccuracies in, and delete their personal data" [1]

- OS design must strictly forgo any collection or storage of user data

IEEE Code of Ethics I Part 7: "to hold paramount the safety, health, and welfare of the public" [2]

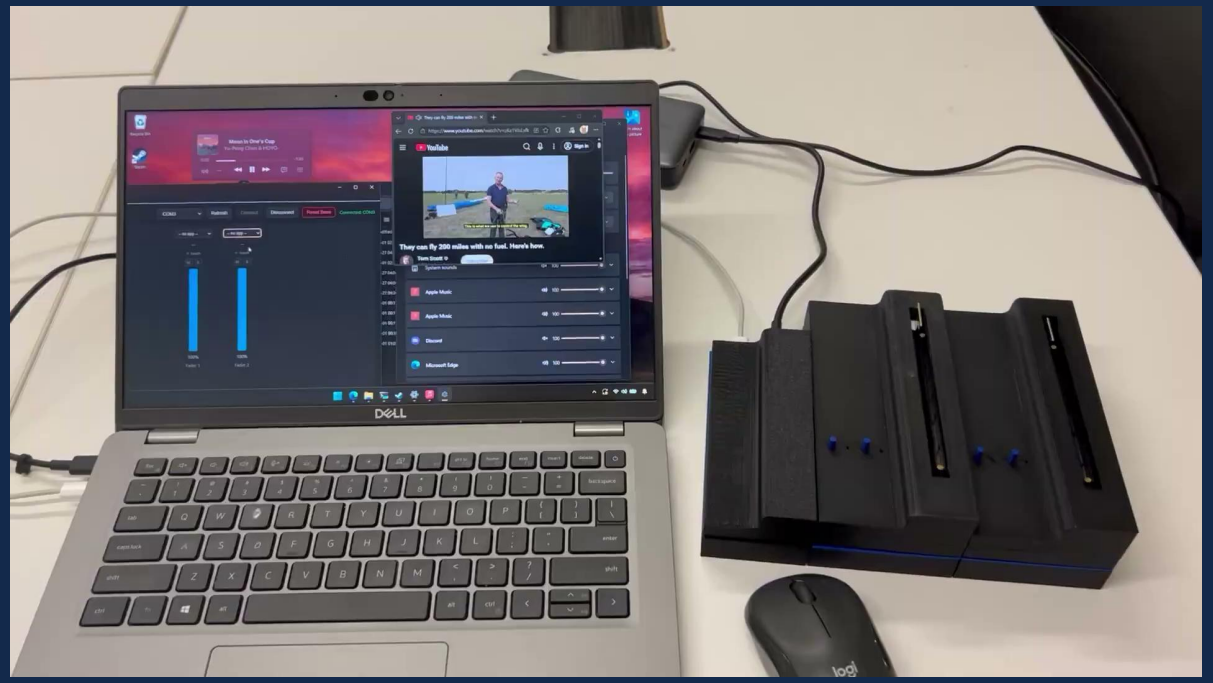
- Address potential hazards such as pinch points on faders and magnetic module interconnects
- Disclose mild magnetic interference to medical devices (e.g., pacemakers) from connectors
- Inform users of any warnings attached to hardware components via clear warning labels

IEEE I Part 5: be "honest and realistic in stating claims or estimates based on available data" [2]

- Remain transparent in latency and accuracy of audio control features through replicable measurements.
- Be receptive and responsive to feedback

[1] Association for Computing Machinery, "ACM Code of Ethics and Professional Conduct," Association for Computing Machinery, Jun. 22, 2018. https://www.acm.org/code-of-ethics
[2] IEEE, "IEEE Code of Ethics | IEEE," Ieee.org, 2020. https://www.ieee.org/about/corporate/governance/p7-8

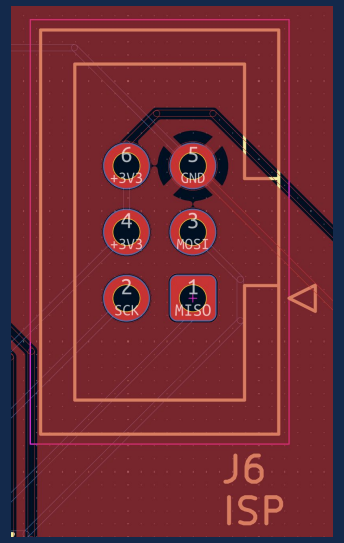
Project Build & Functional Results

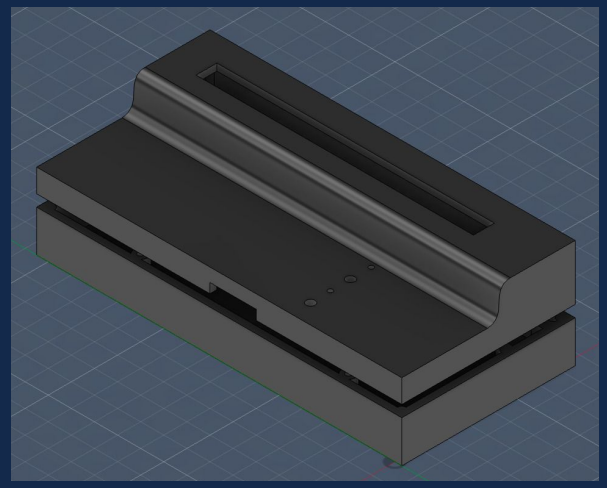


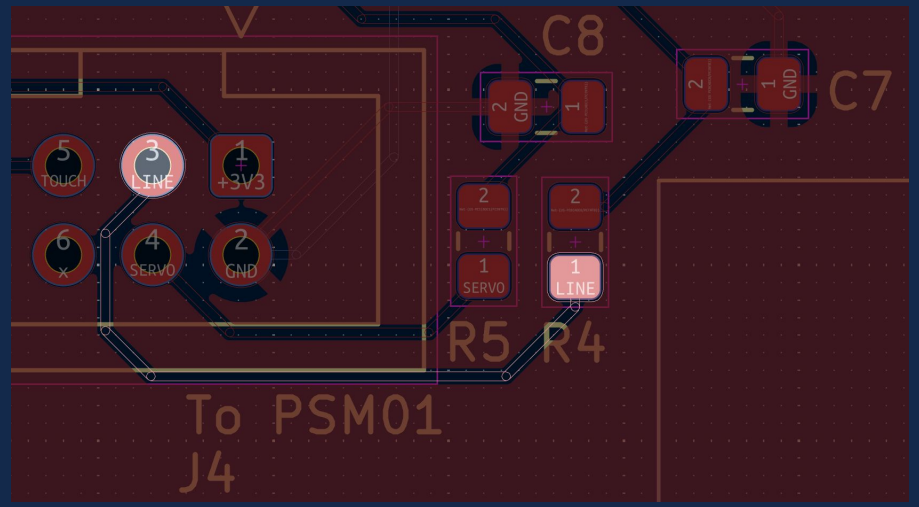
Video demonstrating operation of Audio Mixer Controller

Successes: Everything worked as intended in the end!

- Base Module was able to communicate between PC and faders & deliver appropriate power
- Successful fader actuation within limits of intended positions
- Audio control application could control app volume
- GUI dynamically showed connected faders & able to assign applications







Challenges

Fader ISP Connection

Nonstandard assignments for 6 pin connector
Required external wiring of programmer to assign correct pins

Fader capacitive touch sense

Outdated library (& likely incompatible with intended pin)
Swapped with unused potentiometer track on ADC pin
R4 10k -> 100k, removed decoupling C7
Success with ADC Touch Sense

Fader connector offsets

Machined ~3 mm off on both connector sides
Connectors could reach end of enclosure

Issues flashing ATMega at times

Burned fuses with incorrect configuration caused bricked MCU
Required replacing ATMega and carefully setting configuration

Failed Verifications

Short of 5-pin connectors

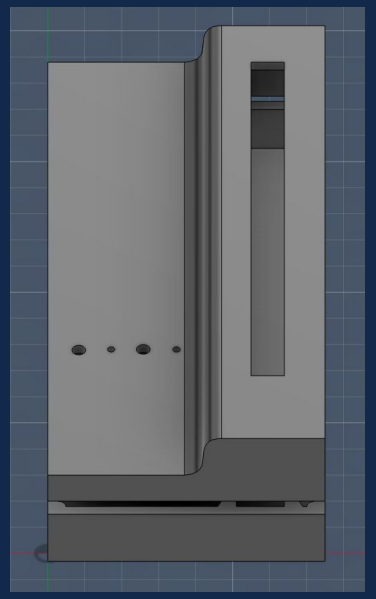
During demo, detaching a Fader module resulted in short

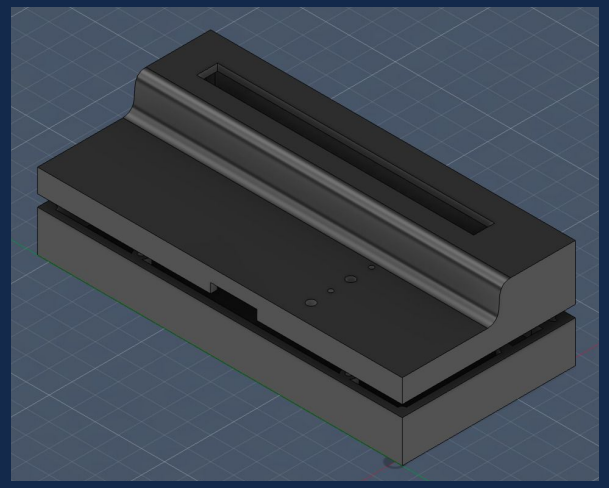
- Damaged I2C pin (SDA) on Base module MCU (ESP32)
- Base module unable to communicate with Fader module

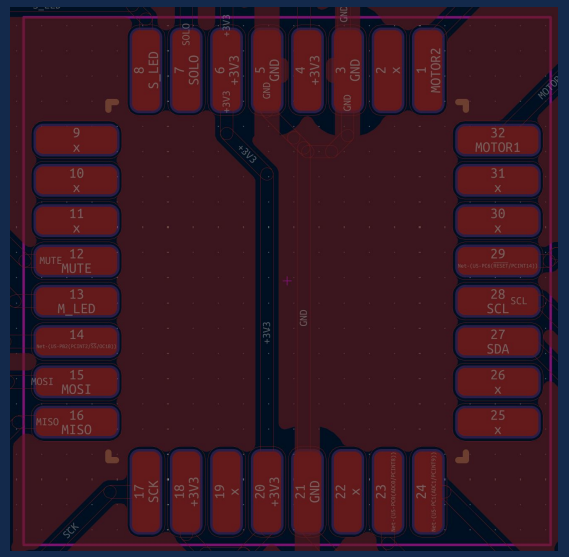
Required replacement of ESP32 on Base module

For future iterations

- I2C protection / buffers
- Keying on housing to prevent unintended pin-pad contact







Insights

A lot of PCB design insights

- Maintain connectors for unused PCB pins
- Lay out dimensions for everything early on for fewer design iterations
- Datasheets may mention resources that are outdated or incompatible
- Ensure standard connector pinouts
- Connector offsets

Improvements

- Better debug for hardware
- Power protection (5-pin connections are bound to slide)
- Better & more robust IDC cables for faders
- Reliable but cost-effective, shorter faders
- Improved tolerances for 3D prints

Thank You!

Questions, comments, concerns?

The Grainger College of Engineering

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN