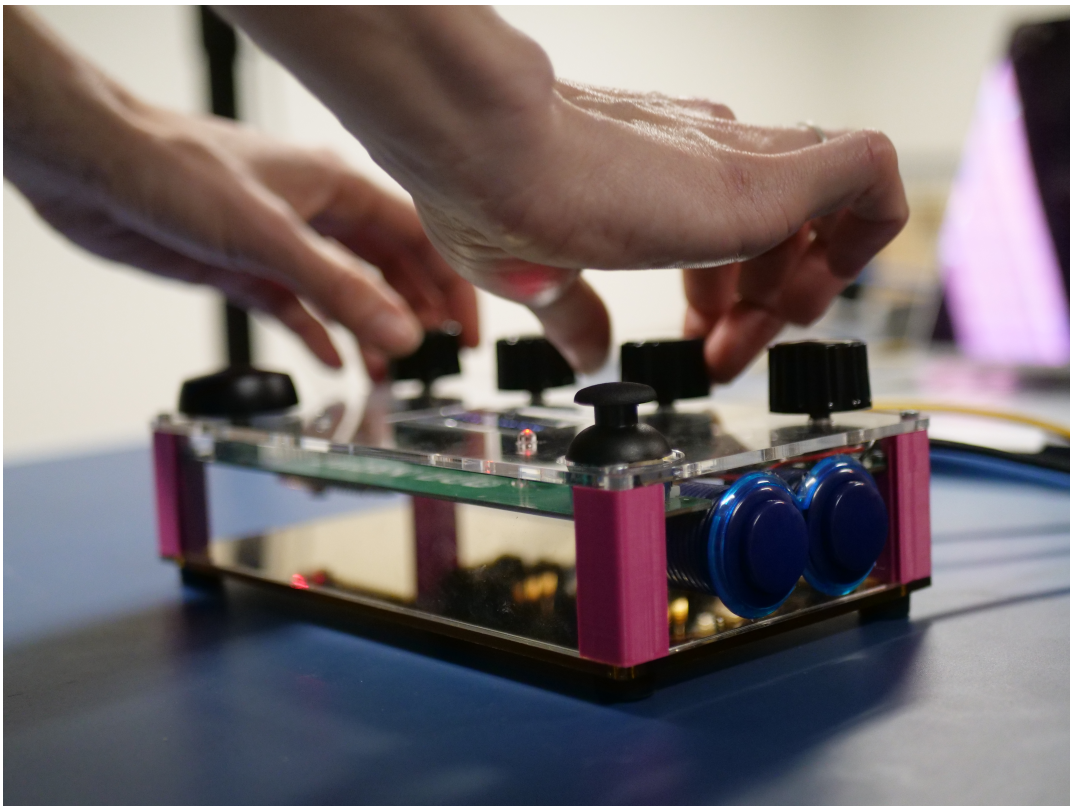
DAISY DUB

- a modular and updateable real-time audio effect for music production and performance -

Master Thesis

Rasmus Kjærbo, Leo Fogadić & Oliver Winkel

Group: Fri-74523-1



Aalborg University
Sound & Music Computing

This thesis was created using a combination of software and hardware tools. The design and development of the electronic hardware and PCBs were done using KiCad and Fusion 360, while LTSpice was used for circuit simulations. PCBs are manufactured by JLCPCB in China. For audio signal processing, the Max programming environment and Gen were used. Compiling of DSP code is initially done with the Oopsy Max Package and then further edited for the addition of special features not available in the original Oopsy Max package. Editing of the C++ code is done with Visual Studio Code, where it is finally flashed to the Daisy Seed development board from. Code is available on our GitLab repository[1].

For case manufacturing, Ultimaker 3s 3D printer was used. For laser cutting an Epilog Fusion M2 40 was used. Adobe Illustrator was used to modify the laser cutting files exported from Fusion 360. Laser cut materials acquired from cotter.dk

Coding and programming were done in Visual Studio Code, while the audio measurements were performed using the Audio Precision tool at Sound Hub Denmark. Preliminary audio measurements were performed with Ableton Live 11.2.10 using iZotope Insight with a Fireface UCX sound card connected via USB, and the oscilloscope used was the Keysight DSOX1102G Digital Oscilloscope. Further investigation of signals was performed with Matlab 2022b on macOS 12.6.3, Audio Toolbox, Communications Toolbox, DSP System Toolbox, and Image Processing Toolbox. Flow charts and signal flow diagrams made with Microsoft Visio and Lucid.app. Batch figure and image processing with ImageMagick.

This combination of tools and equipment allowed for a comprehensive design and testing process, ensuring the accuracy and quality of the final product.

The authors would like to thank the supervisor Stefania Serafin <sts@create.aau.dk> for guidance in writing the report, Sound Hub Denmark for guidance on electrical design and manufacturing, tools, office space and equipment, along with Jens Nørmølle Rasmussen <jlr@soundhub.dk>, Dirk Klijn <dirk@lospeakers.com> for mechanical design, and Andreas Wetterberg <wetterbergmusic@gmail.com> for software design and UI discussions. To friends, family, and everyone involved, thank you for your support and guidance in making this thesis and product a reality.

[1] https://gitlab.com/northernstructuresaudio/daisy-dub

AALBORG UNIVERSITY

STUDENT REPORT

Title:
DAISY DUB

Theme:
Sound & Music Hardware Innovation

Project Period:
Fall 2022 / Spring 2023

Project Group:
Fri-74523-1

Participant(s):
Rasmus Kjærbo
Leo Fogadić
Oliver Bjørk Winkel

Supervisor(s):
Stefania Serafin

Copies: 0

Page Numbers: 117

Date of Completion:
March 10, 2023

Abstract:

This paper presents the development of a versatile and modular real-time audio effect unit called the Daisy Dub for music producers, performers and DJs. The device utilises the state of the art Daisy Seed development board by Electrosmith and comes with a custom made PCB and hardware case. It features a range of real-time audio effects, with an emphasis on creative delays and includes a range of modulation effects and filters in its feedback path. In addition, the unit is compact and portable, with an interactive graphical interface, four knobs and two arcade buttons for performance control, an encoder for menu diving, and an OLED screen for spectrum analysis. It features quadrophonic audio processing in real-time and is portable and powered by USB Type-C. DSP is developed with Gen by Cycling '74 and took inspiration from state of the art hardware audio delay units and modular real-time audio effect processors. A series of DSP and hardware evaluation tests were performed along with two usability tests, from the early cardboard prototype to the final manufactured device to evaluate the effectiveness and usability. Our research demonstrates the feasibility and potential of creating a versatile and modular real-time audio effect for music production and live performance. The Daisy Dub offers a modern take on contemporary real-time audio effects, emphasising a delay effect, and we believe it contributes to the field of audio effects and the music making industry in general.

Contents

Preface

Aalborg University, March 10, 2023

This thesis stands as the final project of the Sound & Music Computing Master's programme at Aalborg University, Copenhagen. The work was carried out between October 2022 and March 2023 in collaboration with Sound Hub Denmark. The thesis documents the development of Daisy Dub, a modular and scalable, repairable, hackable, open-source unified hardware and software solution for audio effects geared towards musicians, performers, producers and DJs. It is the result of a collaborative effort between a group of engineers and musicians who share a passion for creating innovative and high-quality tools for sound design and music production. The project is intended to provide an alternative to existing commercial products that lack the flexibility and hackability that many musicians and sound designers seek. Code, files and figures are available on our GitLab repository[2].

Rasmus Kjærbo
<rkjarb19@student.aau.dk>

Leo Fogadić
<lfogad20@student.aau.dk>

Oliver Bjørk Winkel
<owinke17@student.aau.dk>

[2] https://gitlab.com/northernstructuresaudio/daisy-dub

Chapter 1

Introduction

In recent years, the rapid advancement of microprocessor technology has greatly impacted the audio processing and sound design field [8, 6, 7, 10, 11, 12, 15, 9, 19, 18, 30, 20]. The use of microprocessors in musical devices has enabled a new level of modularity, flexibility, and real-time audio effects. The development of compact and portable devices, such as the Daisy Seed development board by Electrosmith[9], has made it possible for musicians and sound designers to create high-quality audio in a wide range of settings, from music production to live performance. This thesis focuses on the further development of the Daisy Dub[12] (presented at SMC21[1]), a real-time audio effect unit that utilizes the state of the art Daisy Seed development board. The Daisy Dub was originally developed as a cardboard prototype, but this report explores the process of taking it from a prototype to a production ready multilayered printed circuit board (PCB). The focus is on how microprocessor technology has enabled the creation of a versatile and modular real-time audio effect unit, which can be used in a wide range of music tech and STEAM education contexts. This research paper will examine the technical and design challenges encountered in developing the Daisy Dub (see figure 1.1), including using the Electrosmith Daisy Seed development platform and Cycling '74 Gen software, as well as exploring various audio delay units and their design principles. The research results demonstrate the feasibility and potential of creating a versatile and modular real-time audio effect unit for music production and live performance.

Throughout this thesis, we will explore the technical and creative aspects of audio effect design, as well as the importance of usability testing and user feedback in the development process. We hope that Daisy Dub and the research presented in this thesis will contribute to the advancement of modular and updateable real-time audio effects for music production and performance.

[1] `https://nordicsmc.create.aau.dk/wp-content/NordicSMC/Nordic_SMC_2021_paper_29.pdf`

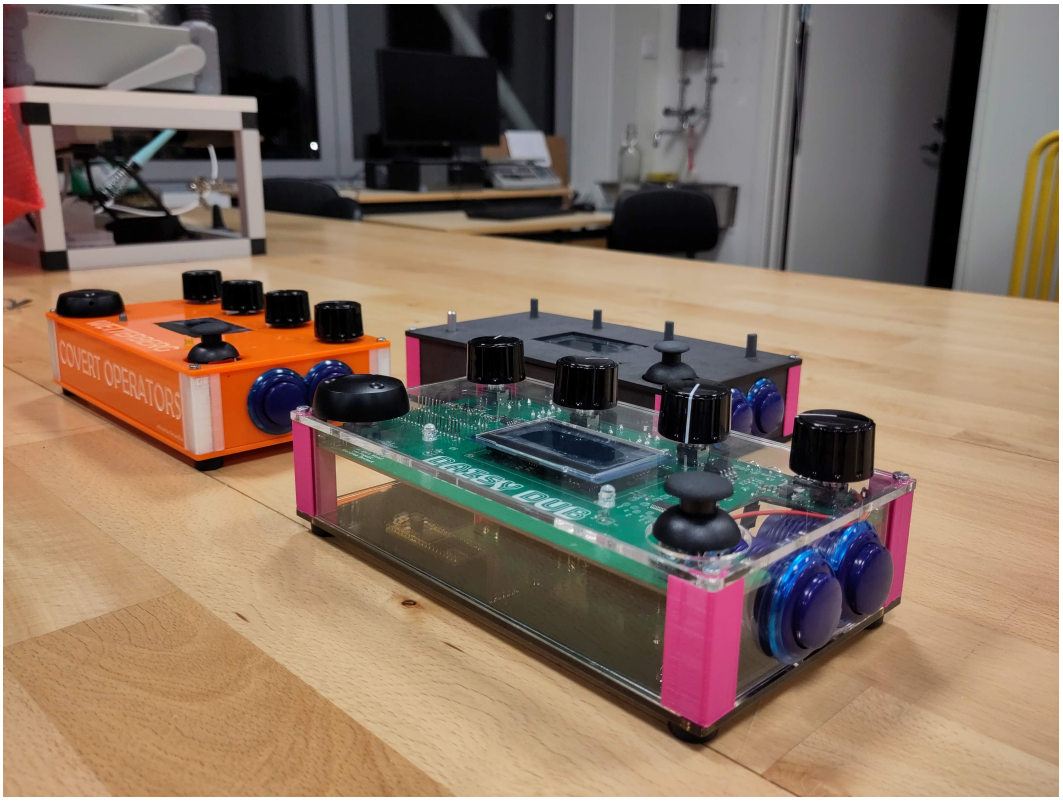


Figure 1.1: Daisy Dub prototypes in three different case designs.

1.1 Background and motivation

The background and motivation for this thesis is the need for versatile and modular performative real-time audio effects for music producers and performers. Currently, the market is dominated by effects pedals that are not easily updateable or customizable, and digital audio workstations (DAWs) offer a range of effects but are not optimized for live performance without further programming and external hardware controllers. Electrosmith's Daisy Seed platform and the Cycling '74 Gen programming environment offer a solution to this problem by providing a programmable development board and a software environment that allows for the creation of physical custom audio effects. As the prototype that this project builds upon is a delay effect, the background of this type of effect as well as its impact will be explored.

1.2 Delay effects

Delay is a fundamental audio effect in music production and performance[4, 6, 7, 15, 20, 21, 23, 26, 29]. It creates an echo-like effect by repeating a signal at a specified time interval and can be used to create various sounds, from subtle ambience to more extreme effects. There is a wide range of delay units available on the market, ranging from simple analogue pedals to more advanced digital units with a wide range of features and capabilities. These units can be used in various settings, including recording studios, live performances, and DJing. Delay effects are a common staple in music production and performance, with a range of applications and variations. In addition to simple delays, which repeat a sound at a set interval, there are also modulated delays, which modulate the delay time to create a sweeping or pulsing effect, and multi-tap delays, which allow for the creation of complex rhythms and patterns. Other variations include ping-pong delays, which bounce the delayed sound back and forth between the left and right channels, and reverse delays, which play the delayed sound in reverse. Historically, delay effects have played a significant role in shaping the sound of popular music. The use of tape delays in the 1950s and 1960s helped define the sound of rock and roll and laid the foundation for modern delay effects. Iconic guitarists such as Eddie Van Halen and Jimmy Page made extensive use of delays in their playing, helping to popularize the effect and establish it as a fundamental tool in the musician's toolkit.

1.2.1 Iconic and historic records that used early tape delay effects [authors abbreviated list]

- "I Am the Walrus" by The Beatles (1967): This song used a tape loop of the intro played backwards to create a disorienting and psychedelic effect. The song features a number of experimental sound effects, including a tape loop of a radio broadcast and a choir singing "Oompah, oompah, stick it up your jumper". The backwards guitar part in the intro was recorded normally, and then played backwards and added to the mix.
- "Echoes" by Pink Floyd (1971): This song is perhaps one of the most famous examples of tape delay, with the effect being used to create a sense of space and atmosphere. The song features a number of experimental sound effects, including tape delay on the drums and vocals, as well as a ping-pong delay effect. The song is known for its extended instrumental passages and atmospheric soundscapes.
- "Heroes" by David Bowie (1977): The song features a distinctive guitar sound created using a "Frippertronics" technique, which involves playing and recording a guitar part onto two tape machines simultaneously, and then playing the recording back while recording another part on top of it. The result is a layered sound with a delay effect.
- "Surrender" by Cheap Trick (1978): The guitar solo on this song was created using a tape delay effect, giving it a distinctive and memorable sound.
- "The River" by Bruce Springsteen (1980): This song features a prominent use of tape delay on the vocals, giving them a unique and memorable sound.

1.2.2 Iconic uses of delay; Delia Derbyshire

Delia Derbyshire was an iconic British sound artist and engineer who was active in the 1960s and 1970s. She is best known for her work with the BBC Radiophonic Workshop, where she was responsible for creating innovative and experimental sound effects and music for radio and television. One of Derbyshire's most famous works is the electronic theme music for the BBC science fiction series "Doctor Who," which she created using tape manipulation and other experimental techniques. She was also a pioneer in the use of electronic music instruments, such as the Ondes Martenot, and was one of the first musicians to use computer-generated sound in her compositions. In addition to her work with the BBC, Derbyshire was also a member of the avant-garde music group White Noise, and her work with tape manipulation and electronic music has had a lasting influence on the development of electronic music and sound design [16].

1.2.3 Iconic uses of delay; King Tubby

King Tubby, born Osbourne Ruddock in 1941, was a Jamaican sound engineer and record producer who is widely regarded as a pioneer in the field of dub music. Tubby's work was driven by an experimental approach to sound engineering and a deep understanding of electronics. He was known for his innovative techniques, such as using custom-built mixing consoles and incorporating effects like echo and reverb into his music. Tubby was a self-taught engineer who began his career repairing electronic equipment. He quickly became known for his ability to fix and modify audio equipment, and he began working as a sound engineer at a number of local recording studios. It was here that Tubby developed his signature sound, which was characterized by heavy use of echo and reverb, as well as the incorporation of other effects like phasing and flanging. Tubby's work was a major influence on the development of dub music, a subgenre of reggae that emerged in the 1970s and is known for its heavy use of echo and reverb. Tubby was a key figure in the evolution of dub, and his work inspired countless other producers and engineers to experiment with electronic music. His contributions to the field have had a lasting impact, and his legacy is still felt today in the work of contemporary electronic musicians and producers. There are several iconic albums by King Tubby that have had a significant impact on the world of electronic music. These include: "Dub From the Roots" (1975)[2]: This album is considered one of King Tubby's most influential works, and it features a number of tracks that showcase his innovative use of delays, reverb, and other effects to create a unique and immersive soundscape. "The Roots of Dub" (1975): This album is another classic from King Tubby, and it features a number of tracks that showcase his mastery of the dub style. The album is known for its powerful and resonant bass lines, as well as its use of delay and reverb to create a sense of depth and space [5].

1.2.4 Iconic uses of delay; Else Marie Pade

Else Marie Pade was a Danish composer and electronic music pioneer who played a significant role in the development of electronic music in the mid-20th century, particularly in Denmark. Pade was a pioneer in the use of tape manipulation in her compositions, which involved recording and manipulating sounds using magnetic tape. In the 1950s, Pade studied at the Danish Radio Experimental Studio, where she worked with tape manipulation techniques and began creating electronic music compositions. Pade's use of tape manipulation was innovative for its time and allowed her to create unique and experimental soundscapes in her music. One of Pade's most notable works is the 1958 composition "Symphonie Magnétophonique," which was created using tape manipulation techniques. This piece is considered a pioneering work of electronic music, and its use of tape manipulation has had a lasting influence on the genre. Pade's use of tape manipulation in her compositions

[2] https://www.discogs.com/artist/25872-King-Tubby

was groundbreaking, and she remains an important figure in the history of electronic music. Her work continues to inspire and influence contemporary electronic musicians and composers [16].[3]

1.2.5 Iconic producers, artists, and engineers working with tape, digital hardware, and software delays [authors abbreviated list]

- Delia Derbyshire: Pioneering electronic musician and composer responsible for creating the iconic Doctor Who theme using tape delay and other electronic music techniques.
- King Tubby: Jamaican producer and engineer is credited with creating the dub music genre, which extensively used tape delay effects to create a distinctive and otherworldly sound.
- Brian Eno: Musician, producer, and composer is known for his pioneering work in ambient music, which often used tape delay to create immersive and atmospheric soundscapes.
- Giorgio Moroder: Italian producer and composer known for using electronic music and tape delay in disco and pop music, including Donna Summer's hit "I Feel Love."
- Annie Lennox: Musician and composer known for using tape delay in her solo work and as part of the Eurythmics, particularly on the song "Sweet Dreams (Are Made of This)."
- Daniel Lanois: Producer and engineer is known for his work with artists such as U2 and Bob Dylan, and for using tape delay and other effects to create a distinctive and atmospheric sound.
- The Edge: Musician and composer known for using tape delay as part of U2, particularly on the song "Where the Streets Have No Name."
- Robin Guthrie: Musician and producer known for using tape delay as part of the Cocteau Twins, particularly on the song "Lorelei."

In the digital age, delay effects have become even more versatile and powerful, with software-based delays offering a range of features and flexibility. These software delays can be easily integrated into a digital audio workstation (DAW), allowing for precise control and manipulation of the delay time, feedback, and wet/dry mix. Modulation sources, such as envelopes and LFOs, can also modulate software delays, opening up even more creative possibilities.

Despite their widespread use, delay effects continue to evolve and offer new possibilities for music production and performance. As technology and software continue to advance, we will likely see even more innovative and creative uses of delay in the future.

[3] `https://kvindebiografiskleksikon.lex.dk/Else_Marie_Pade`

Chapter 2

State of the art

Delay effects are an integral part of the audio processing toolkit for music producers and performers, providing a range of options for creating echo, reverb, and other temporal effects. In recent years, modular real-time audio effects (MRTAEs) have emerged as a popular platform for designing and implementing cascade of effects, e.g. effect chains, offering a high degree of flexibility and customization for users. Through this analysis, we aim to provide a comprehensive overview of the state of the art in modular real-time audio effects with an emphasis on delay units for music producers and performers and to offer insights and guidance for researchers and practitioners working in this field.

2.1 Overview of modular real-time audio effects

Modular real-time audio effects have been a staple in music production and performance for decades, providing a means for musicians to manipulate and shape the sound of their music in real-time. These effects can be hardware-based, such as pedals and rack-mounted units, or software-based, such as digital audio workstations (DAWs) and plug-ins. One key feature of modular real-time audio effects is their ability to be customized and configured to meet the specific needs of the user. This can be done through the use of patch cables, which allow users to create complex signal processing chains by connecting different modules together. In recent years, there has been a resurgence of interest in modular real-time audio effects, with a number of companies producing hardware units and software plug-ins that allow users to create their own customized effects. These units often utilize digital signal processing (DSP) technology, which allows for greater flexibility and higher levels of performance compared to traditional analog effects.

2.2 Delay units for music production and performance

Delay is a fundamental audio effect that has been used in music production and performance for many years.

There are a wide range of delay units available on the market, ranging from simple analog pedals to more advanced digital units with a wide range of features and capabilities. These units can be used in a variety of settings, including recording studios, live performance, and DJing.

2.3 Abbreviated list of iconic delay effects from the 1970s till today:

Years	Delay effect
1970s	Echoplex EP-3: One of the first widely-used delay pedals, the Echoplex EP-3 was introduced in the 1970s and became a popular choice for creating echo and reverb effects in recording studios.
	Roland Space Echo: Another pioneering delay effect, the Roland Space Echo was a popular choice for creating spatial effects in the 1970s and continues to be used today in recording studios and live performances.
1980s	Lexicon PCM42: A digital delay and reverb processor that was widely used in recording studios in the 1980s.
	Roland SDE-2500: A digital delay processor that was introduced in the 1980s and became a popular choice for recording studios.
1990s	Eventide H3000: A popular digital effects processor that was widely used in recording studios in the 1990s. It offered a range of delay, reverb, and pitch-shifting effects.
	TC Electronic 2290: A popular digital delay and effects processor that was widely used in recording studios in the 1990s.
2000s	Eventide TimeFactor: A popular digital delay and effects pedal that was widely used in recording studios and live performances in the 2000s.
	Strymon El Capistan: A popular digital delay and effects pedal that was widely used in recording studios and live performances in the 2000s. It offered a range of tape delay and modulation effects.
2010s	Strymon Timeline: A popular digital delay and effects processor that was widely used in recording studios and live performances in the 2010s. It offered a range of delay, reverb, and modulation effects.

Table 2.1: An abbreviated table narrowed down to units that were especially popular in recording studios and for music producers and performers.

2.4 Comparison of Roland SDE-2500 and Strymon El Capistan

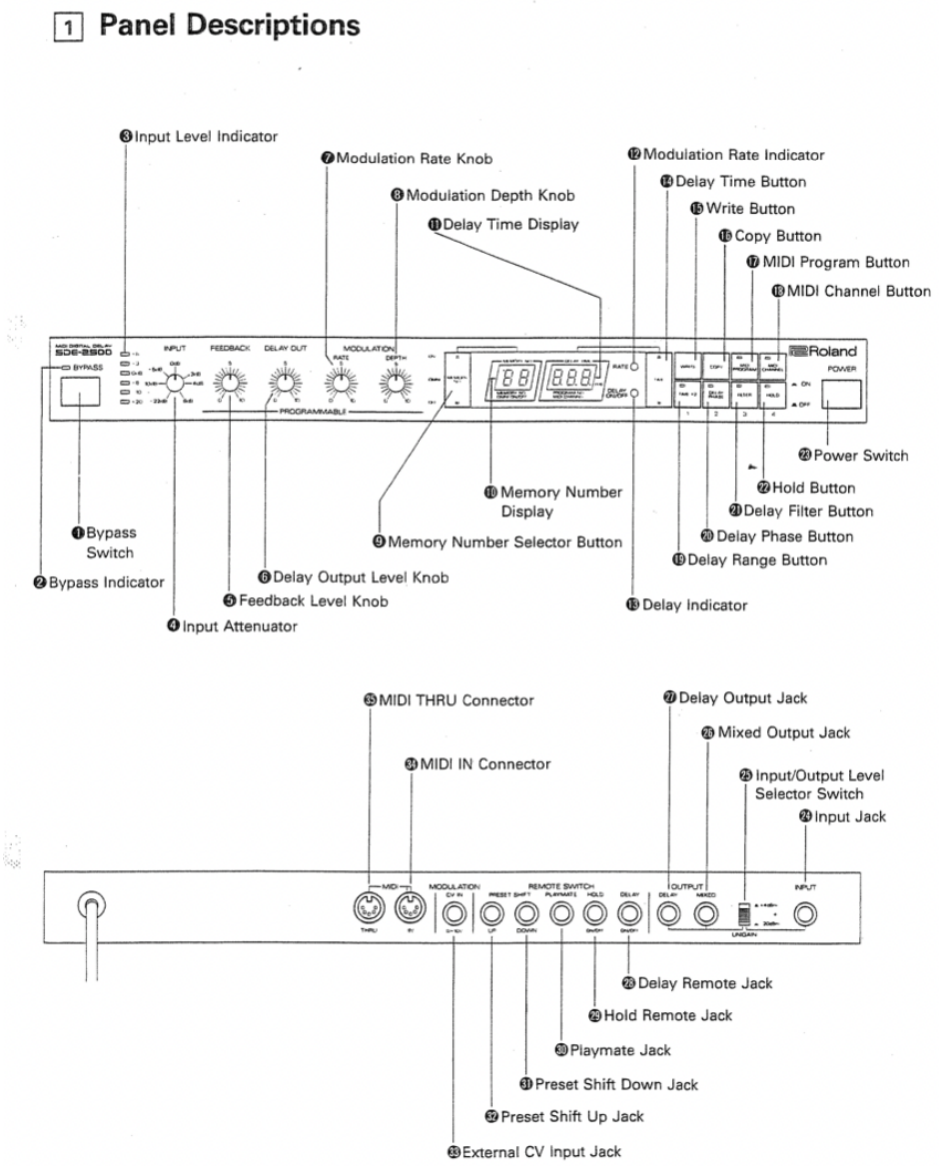


Figure 2.1: Roland SDE-2500 panel descriptions

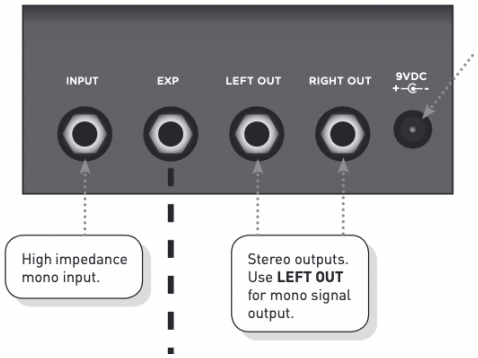


Figure 2.2: Strymon El Capistan connections

1. **Format**: The Roland SDE-2500 is a rackmount processor[25], while the Strymon El Capistan[27] is a stompbox-style pedal. This means that the SDE-2500 is intended to be used in a studio setting, while the El Capistan is more portable and can be easily used in a live performance context.
2. **Digital Signal Processing**: The SDE-2500 and El Capistan use DSP to create delay and effects algorithms. The SDE-2500 uses a 15-bit digital audio processor and has a maximum delay time of 750 ms, while the El Capistan uses a 24-bit 96 kHz digital audio processor with 20 seconds maximum delay time. Using a higher-resolution processor in El Capistan can result in higher quality and more detailed sound.
3. **Delay types**: El Capistan offers three different types of delay: fixed head, multi-head, and single head. The fixed head mode emulates the sound of a single fixed playback head on a tape machine, while the multi-head mode emulates the sound of multiple playback heads with varying spacing. The single head mode provides a clean, clear digital delay. The El Capistan is known for its accurate and realistic tape delay emulation, which allows users to create warm and authentic tape-style delays.
4. **Modulation and effects**: The SDE-2500 offers various modulation types depending on the setting for modulation time. El Capistan offer the same type of delay time modulation as well as Wow & Flutter, Tape Age, Spring reverb, Tape Crinkle, Bias, Low End Contour and Sound on Sound. The El Capistan is known for its rich and complex modulation capabilities, which allow users to create a wide range of modulation effects with great depth and nuance.
5. **Control**: The SDE-2500 and El Capistan have different control schemes, with the SDE-2500 offering a more traditional set of knobs and buttons for delay and modulation and a small LCD. El Capistan offers a reduced set of primary controls but adds a convoluted layer of secondary knob functions to the primary controls. This can affect the devices' ease of use and flexibility, depending on the user's preferences.
6. **Connectivity**: The SDE-2500 and El Capistan offer a range of connectivity options for interfacing with other audio equipment. The SDE-2500 has a single 6.35 mm jack input and output and a separate send and return for connecting to external effects processors (see figure 2.1). It offers an additional 'Mixed Output Jack'. It also has a footswitch input for external control. The El Capistan has a mono 6.35 mm jack input and two 6.35 mm jack outputs (stereo) and an expression pedal input for continuous control (the newer revision El Capistan dTape Echo V2 also has USB Type-C and MIDI via 6.35 mm jack). See figure 2.2 for V1 connections.
7. **Memory**: Only the SDE-2500 offer memory functionality for storing and recalling delay settings. The SDE-2500 has 64 memory locations.

8. **Size and weight**: As mentioned previously, the SDE-2500 is a rackmount processor, while the El Capistan is a stompbox-style pedal. This means that the SDE-2500 is larger and heavier than the El Capistan, which can be a consideration for users who need a more portable device.
9. **Compatibility**: Both the SDE-2500 and El Capistan are compatible with a range of audio equipment, including mixers, amplifiers, and audio interfaces. The SDE-2500 is a rackmount processor, requiring a rack for mounting, while the El Capistan is a stompbox-style pedal that can be easily placed on a pedalboard.
10. **Sound quality**: Both the SDE-2500 and El Capistan are known for their high-quality sound, with both devices offering a range of delay and effects algorithms designed to provide a high level of clarity and detail. The SDE-2500 uses a 15-bit digital audio processor, while the El Capistan uses a 24-bit digital audio processor, which can result in higher quality and more detailed sound in the El Capistan.
11. **Price**: The SDE-2500 and El Capistan have different price points, with the SDE-2500 typically more expensive than the El Capistan since it is vintage. This can be a consideration for users working within a budget or having specific financial constraints.
12. **Availability**: The SDE-2500 and El Capistan are still available on the market today, although the SDE-2500 may be more difficult to find due to its age. This can affect the availability and accessibility of the devices, depending on the user's location and resources.
13. **Reputation**: Both the SDE-2500 and El Capistan have strong reputations within the audio community, with both devices being widely used and highly regarded by professionals and enthusiasts alike. The SDE-2500 is known for its classic and vintage character, while the El Capistan is known for its high-quality sound and versatility.

The Roland SDE-2500 and Strymon El Capistan are iconic and highly-regarded delay and effects devices widely used in recording studios and live performances. While they have some similarities, they also have some key differences that may make one device more suitable than the other, depending on the user's specific needs and preferences.

2.5 Contemporary modular real-time audio processors

When comparing state of the art devices there are several factors to consider. The four devices in Table **??** - 1010audio Blackbox, 1010audio Bluebox, Moddevices Duo X, and Otomachines BIM - are all real-time audio effect processors that offer a range of effects and processing options. Here is an overview of each device and its respective company:

1010audio is a company based in Los Angeles, California that specializes in the design and manufacture of modular synthesizer modules and real-time audio effect processors. The company's Blackbox[1] and Bluebox[2] devices are both portable, standalone audio processors that allow users to create and manipulate audio on-the-go [2, 1].

The Blackbox features a 1 stereo channel input and 3 stereo channels output + 1 stereo channel headphone output, touch screen interface, built-in effects, and recording capabilities, while the Bluebox features a stereo input/output, touch screen interface, built-in effects, and support for third-party plugins. Both devices are designed with modularity in mind, allowing users to expand their functionality by connecting them to other modular synthesizer modules.

Moddevices is a Portuguese company that designs and manufactures the Duo X, a real-time audio effect processor that is designed to be a versatile, all-in-one solution for musicians and audio professionals. The Duo X features a touch screen interface, built-in effects, customizable effects, and

[1] `https://1010music.com/product/blackbox`
[2] `https://1010music.com/product/bluebox`

support for third-party plugins. The device is capable of high-quality audio processing and can be used in a variety of applications, from music production to live performance.

Otomachines is a French company that specializes in the design and manufacture of glitchy, stuttering audio processors. The BIM is one of the company's flagship devices, featuring a range of unique and experimental effects that are designed to push the boundaries of traditional audio processing. The BIM features a knob-based interface, built-in effects and a focus on glitchy, stuttering effects.

When comparing these devices, it is clear that they all offer a range of features and processing options that can be used in a variety of applications. The 1010audio Blackbox and Bluebox devices are notable for their modularity and portability, making them ideal for on-the-go audio processing. The Moddevices Duo X offers a versatile, all-in-one solution with customizable effects and support for third-party plugins, while the Otomachines BIM offers a unique and experimental approach to audio processing with a focus on glitchy, stuttering effects. In terms of user interface, the 1010audio devices and the Moddevices Duo X all feature touch screen interfaces, while the Otomachines BIM features a knob-based interface. Control options also vary between devices, with MIDI and CV control options available on some devices. Overall, these devices offer a range of options for audio processing and manipulation, and are well-suited for a variety of applications in music production and performance. Their respective companies are all dedicated to providing high-quality audio products that push the boundaries of traditional audio processing, and are well-regarded within the music technology industry. Another important factor to consider is the range of built-in effects that each device offers. The 1010audio products and the Moddevices Duo X both include a range of built-in effects, such as reverbs, delays, and distortion. The Otomachines BIM, on the other hand, is more focused on creating glitchy, stuttering effects and may not offer as much variety in terms of effects. It is important to consider the interface and control options that each device offers. The 1010audio products and the Moddevices Duo X both feature touchscreen interfaces, which can make it easier to navigate and control the device. The Otomachines BIM, on the other hand, features a more traditional knob-based interface. Overall, each of these devices offers its own unique set of features and capabilities. The 1010audio products are highly modular and customizable, while the Moddevices Duo X offers a high degree of flexibility and an open platform for creating custom effects. The Otomachines BIM is more focused on creating glitchy, stuttering effects and may be better suited for musicians who want to create a specific type of sound.

The philosophy behind these devices is centered around modularity, customization, and hackability. The idea is to provide musicians and sound designers with a set of tools that are flexible enough to adapt to their specific needs and creative goals. Modularity is a key aspect of these devices, allowing users to mix and match various modules to create a customized signal chain. This modularity not only enables users to create unique sounds and effects, but it also makes the devices more versatile and adaptable to different use cases. Customization is another important aspect of the philosophy behind these devices. Many of these products offer an open platform for creating custom effects or software, which allows users to tailor the devices to their specific needs. This customization can range from simply adjusting parameters to building entirely new effects or software from scratch. Hackability is also a core part of the philosophy behind these devices. Users are encouraged to explore and experiment with the devices in new and innovative ways by providing an open platform for custom software and effects. This hackability can result in unexpected and creative device uses, leading to new and exciting sounds and effects.

In summary, the philosophy behind these devices is centered around providing musicians and sound designers with flexible and adaptable tools that can be customized and hacked to meet their specific creative goals. This approach empowers users to explore and experiment with new sounds and effects, pushing the boundaries of what is possible with audio processing technology.

Feature	**Blackbox (1010audio)**	**Bluebox (1010audio)**	**Duo X (Moddevices)**	**BIM (Otomachines)**
Input	1 stereo 3.5 mm TRS	6 stereo 3.5 mm TRS	Stereo in	Stereo in
Output	4 stereo 3.5 mm TRS (1 stereo out is headphones output)	3 stereo 3.5 mm TRS	Stereo out	Stereo out
Audio resolution	24-bit	24-bit	32-bit floating-point	12-bit
Sample rate	48 kHz	48 kHz	Up to 48 kHz	Up to 48 kHz
Processor type	ARM STM32H743XIH6 Cortex M7 + DP-FPU + DSP 2MB Flash 1Mb RAM, 480Mhz	ARM Cortex-A9	Quad-core 1.5GHz ARM Cortex-A53	Undetermined
Effects	Built-in effects	Built-in effects	Built-in customizable effects	Built-in effects
Interface	4 encoders, 8 navigation, 3 transport, 2 small (back and info)	4 knobs, 8 navigation buttons and 3 transport buttons	2 encoders, 8 knobs and 4 buttons	6 knobs and 8 buttons
Screen	Touch screen	3.5″ touch screen	2 2.8" LED screens	No
Control options	MIDI 3.5 mm TRS, USB HOST	MIDI 3.5 mm TRS	MIDI, CV	MIDI in
Store & recall	Project settings, samples	Project settings, parameters, outputs, EQ, effects	UI assignments, ranges and sensitivity, saving and recalling profiles, presets and pedalboards	Presets
SD-Card	Yes	Yes	No	No
Modular	Yes	No	Yes	No
Power	USB 2.0 A-B or AC-adapter	USB 2.0 A-B or AC-adapter	AC-adapter	VDC-adapter
Recording capabilities	Yes	Yes	No	No
Open platform	No	Yes	Yes	No
Release date	2019	2021	2018	2018
Price	€569	€660	€750	€350

Table 2.2: Feature comparison chart of state of the art modular real-time audio effects processors

2.6 Why does the World need yet another audio effect for real-time processing?

The music industry is constantly evolving, and there is a demand for new and innovative audio effects that can help artists and producers create unique and compelling sounds. One potential solution to this demand is the development of a modular and updateable real-time audio effect, which could offer a range of benefits to music producers and DJs. One key advantage of a modular and updateable audio effect is its flexibility and adaptability. Music producers and DJs often rely on a wide range of different audio effects and processing tools in their work, and the ability to easily update and customize an audio effect could make it more suitable for use in a variety of different applications and environments. Another potential benefit of a modular and updateable audio effect is the ability to leverage advances in technology and digital signal processing techniques. As these techniques continue to evolve, it is likely that new and improved audio effects will become possible. A modular and updateable audio effect could incorporate these advances to provide enhanced performance and capabilities that are not available in existing products. In addition to its technical benefits, a modular and updateable audio effect could also offer a unique set of features and capabilities that are not available in existing products. This could make it particularly appealing to music producers, performers, and DJs looking to expand their sound palette and create new and innovative sounds. The development of a modular and updateable real-time audio effect could offer a range of benefits to music producers, performers, and DJs, including flexibility, adaptability, and the ability to leverage advances in technology and digital signal processing. As such, it could be an important and valuable addition to the music industry.

2.7 Foundational work

This thesis is building upon the work done in [13]. The report contains the work done to develop the first prototype of the Daisy Dub and the first user test of said prototype. The report ends with the findings from that user test being used for improvements of the prototype, so before any further work is done on the Daisy Dub these improvements should first be tested.

2.7.1 Initial testing

This testing will be done by performing a usability test on the updated prototype followed by an exit interview. By doing this it should be possible to determine if the adjustment has fixed the issues they were implemented to solve, as well as get an idea of other possible adjustments that should be made.

Setup

The test will be performed by two people: A test conductor and a note taker. The conductor will be responsible for giving the test subject the tasks at hand and also be the person communicating with them throughout the test. The note takers job is to document everything during the experiment, both the errors that may occur during the tests, as well as the answers given during the exit interview.

The subject will be seated on one side of a table, with the prototype in front of them. The test conductor will be seated next to them, with the instructions in front of them. The note taker will be sitting on the other side of the test subject and have a laptop in front of them. A picture of the test setup can be seen in figure 2.3.

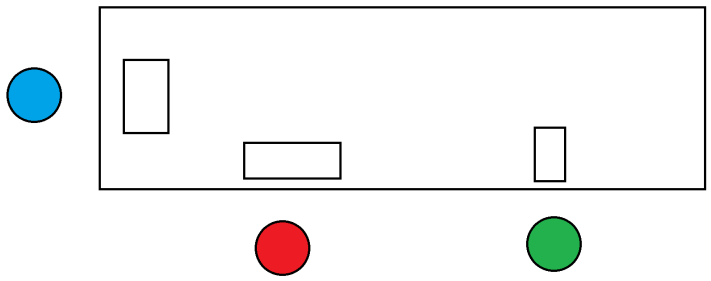


Figure 2.3: A picture of the test setup for the user test. Here we see the note taker in the foreground, and the test subject using the Daisy Dub

Procedure

Before the start of the testing, the subject will be given a quick overview of the prototype by the conductor, and then the subject will have a chance to familiarize themselves with the Daisy Dub. Once the subject believes they are familiar with the delay, the conductor will give them the tasks one at a time. The test subject will also be prompted to think aloud during the testing, as to better understand their thought process.

After the tasks have been completed, the subject will be asked to fill out a System Usability Scale (SUS) questionnaire, followed by an exit interview. The manuscript for the test, including the exit interview, and the SUS questionnaire can be found in Appendix E.

Participants

These tests were done at Electronic Music School Rumkraft in Copenhagen where the tests were done over two sessions. Due to these tests being done at Rumkraft the participants all have at least some knowledge about music effects, which means that they know all of the basic terminology used in the test. No more than five people were tested as any more testing would be a poor return on investment[17]. Testing five people should therefore uncover most of our usability issues without superfluous testing.

Findings

This test showed that the most pressing issue is a visual overhaul of the menu, as every subject had some comment or complaint about an aspect of the menu. These issues included names being confusing, not enough separation between variables, and being unable to tell that the menu continued beyond a certain point. For the next prototype a visual upgrade of the menu should be a priority in order to create a clearer separation between controls.

Multiple subjects also wanted easier control over different features, the most frequent one being the master dry/wet. However it was also brought up that the delay time should be easier to manipulate once it was dialed in, for instance being able to halve the time, or put it into triplet or

dotted notes of the current position. These adjustments seem very useful, and therefore possible solutions to expand which parameters are controllable should be considered, for instance a solution for altering the current selected delay time. Alongside these adjustments several subjects also noted that it is difficult to accurately dial in the delay time. Because the range is so large, it was hard to get the delay to match the timing of the input perfectly. While this can be a desired feature for some users as it gives it a less quantized sound, it should be possible to keep that while still offering the option of finer adjustments.

The last improvement suggestion from this test is to add a bypass control. This would be in the form of a button that would allow the signal to pass through the delay only when held. This would allow the user to send out short bursts of the delay when using it as a send effect, which we were doing was done in this test. Normally when using send effects this is controlled by adjusting the send channel volume slider, but a bypass button or switch would allow for a different faster type for bypassing.

Most of these concerns were raised during the exit interview, although it was clear throughout the testing when a subject had trouble navigating the menu. Throughout the exit interview participants were also asked to put numerical values on statements related to their interaction with the Daisy Dub

The statements were as follows:

1. On a scale of 1 to 10, where 1 is completely unexpected and 10 is exactly as expected, did the Daisy Dub react as you expected?
2. On a scale of 1 to 10, where 1 is very hard and 10 is very easy, how difficult was it to find the right setting?
3. On a scale of 1 to 10, where 1 is very hard and 10 is very easy, how well were you able to change a desired parameter?
4. On a scale of 1 to 10, where 1 is very bad, and 10 is very good, how is the sound quality of the Daisy Dub?
5. Was there any interaction which you found inconvenient or would like to work differently?

The first four of these questions forced to participants to put a number on their interaction, and as such have an opinion on certain aspects of the Daisy Dub. By doing this it should be easier to see if any of these areas are specifically lacking. The last question is instead very open. This allows the test subject to talk about any issues or annoyances they may have found with the Daisy Dub during testing that was harder to see for the note taker, or is not covered by the other questions.

For the numerical value questions only the second test subject scored a question below a 7, and gave the sound quality a rating of 6. In further developing the Daisy the sound quality should improve, as the prototype at this stage is on a breadboard, which allows for a lot of accidental additional noise in the signal. Except for this one answer all of the participants scored the various issues with a 7 or higher, which was deemed acceptable.

For the System Usability Scale the Daisy Dub scored a 79.5, which is well above 68, which is considered an above average score [28]. When further developing the Daisy Dub, the expectation will be to sustain or improve this score.

As the work for this thesis started, the project was selected for the SoundTech Incubator, a part of their Beyond Beta program[3]. This means that for this thesis Sound Hub Denmark will be providing additional supervision for the technical parts of the development of the Daisy Dub.

[3] `https://www.beyondbeta.dk/incubation-programs/soundtech`

2.8 Research questions and objectives

The research questions for this thesis are:

- What are the design considerations for a compact and low-cost modular and updateable real-time audio effect for music production and performance?
- How can the firmware development and update utility for Daisy Dub be implemented and evaluated?
- How can the usability and user experience of Daisy Dub be tested and improved?

The objectives of this thesis are:

- To design and implement the mechanical and electrical aspects of Daisy Dub.
- To improve upon the current software, and add further functionality.
- To develop and evaluate the firmware update utility for Daisy Dub.
- To conduct usability testing and a workshop evaluation of Daisy Dub.
- Formulate a framework for a STEAM project involving Daisy Dub.

Chapter 3

Proposing a versatile and modular real-time audio effect unit

In recent years, there has been a growing demand for audio effect units that offer a high level of flexibility and customization for music production and live performance. In this chapter, we propose a new unit that aims to meet this demand by incorporating the best features of existing delay devices, such as the Roland SDE-2500 and Strymon El Capistan, and state of the art modular real-time audio effects like the ones made by the companies 1010audio[1] and mod.audio[2] and adding new features that are designed to enhance the unit's versatility and functionality.

3.1 Format and digital signal processing

The new unit is designed to be a compact and portable unit, with a rugged and durable design suitable for live performances. It is powered by USB Type-C, allowing it to be easily powered via existing power sources and cables from a laptop, outlet or portable battery pack. The unit uses a high-resolution digital audio processor, similar to the one used in the Strymon El Capistan, to provide a high-quality and detailed sound. Additionally, the unit offers quadrophonic audio processing on both input and output, allowing users to create and manipulate 4-channel audio signals.

3.2 Delay types and modulation

The new unit offers a range of delay types, including digital, tape, and analog emulation, similar to both the SDE-2500 and El Capistan. The unit offers a range of modulation effects, like both the SDE-2500 and El Capistan, but with even more options and control. It also allows users to choose custom modulation parameters using an interactive graphical interface, somewhat similar to the 'secondary' button features on El Capistan but with a much easier overview by having it available on the display.

[1] `https://1010music.com/`

[2] `https://mod.audio/`

3.3 Control and connectivity

The new unit has four knobs for performance control, allowing users to easily adjust delay and effects parameters on the fly. It also has an encoder for specific sound design menu diving, allowing users to access a wide range of advanced sound design options. The unit has two arcade-style push buttons and a XY-axis thumb joystick with a push button for performance interaction. An OLED display is used for selectable spectrum analysis of input and output and menu diving, providing users with a clear and intuitive interface. The new unit offers a range of connectivity options, including double stereo jacks for audio input and output, making it possible to use the device in either mono mode, stereo or quadrophonic. A mixing matrix for doubling inputs to the outputs is selectable from the menu. The device also features MIDI (Musical Instrument Digital Interface) input and output ports used for syncing tempo with other devices and instruments. MIDI support provides more parameter control from an external controller, or it can be used to send MIDI messages to control other MIDI-compatible devices.

3.4 Memory and programmability

The new unit offers a large number of memory locations to save presets, similar to some of the devices in the state of the art, but with additional features such as the ability to save and recall entire effect chains and create custom preset libraries. The unit is programmable and updateable, allowing users to customize and update the unit's algorithms and features. This can be done by connecting it to a computer via USB.

3.5 Compatibility and sound quality

The new unit is compatible with a range of audio equipment, including mixers, amplifiers, and audio interfaces. Its range of input and output options, including stereo, mono, and quadrophonic, allows for flexibility in a variety of setups. The unit offers a high-quality sound, with a range of delay and effects algorithms that are designed to provide a high level of clarity and detail. Its high-resolution digital audio processor, quadrophonic audio processing, and interactive graphical interface allow users to create a wide range of sounds with great depth and nuance.

3.6 Flexibility and ease of use

The new unit's quadrophonic audio processing and interactive graphical interface allow users to create and manipulate complex multi-channel audio signals. This could be particularly useful for electronic music producers, sound designers, and experimental musicians who are looking to push the boundaries of audio processing. The unit's control scheme, which combines traditional knobs and buttons with an interactive graphical interface, provides users with a range of options for adjusting and manipulating parameters. This makes the unit easy to use for both beginners and experienced users, and allows for both intuitive and precise control.

3.7 Portability

The new unit is compact and portable. Combined with its USB Type-C power delivery and programmability, this makes it easy to use in a variety of settings. It can be easily transported to gigs, recording sessions, and other events, and can be easily integrated into a wide range of audio setups.

3.8 Summary

In summary, the new unit described in this chapter is an extremely versatile and powerful tool for recording studios and live performances. Its quadrophonic audio processing, interactive graphical interface, and programmability allow users to create and manipulate complex audio signals with great flexibility and creativity. Its range of delay and effects options, combined with its intuitive control scheme, make it easy to use for both beginners and experienced users. Its compact and portable format, combined with its USB Type-C power delivery and compatibility with a range of audio equipment, make it a convenient and practical tool for a wide range of audio professionals. In conclusion, the new unit represents a significant advancement in the field of real-time audio effects, and has the potential to significantly enhance the capabilities of music producers and performers.

3.9 Daisy Dub Delay

One of the main design considerations for a modular and updateable real-time audio effect like the Daisy Dub is usability. It is important that the device is easy to understand and use for the target user group, which in this case includes music producers, DJs, and performers. This may involve considering factors such as the layout of controls and displays, the availability of documentation and tutorials, and the overall user experience.
Repairability: Given the high demand and use that music producers and performers may have for the device, it is important that it can be easily repaired or maintained in the event of a problem. This may involve designing the device to make it easy to access and replace components, or including features such as diagnostic tools or self-testing capabilities.
Modularity: The ability to easily add or remove components from the device allows users to customize it to their specific needs and adapt it to different situations or use cases. This may involve designing the device in a way that allows users to easily swap out different components or modules, or to easily customize the device.
Flexibility: The ability to adapt the device to different situations or use cases is crucial for music producers, DJs, and performers who may need to use the device in a variety of different contexts. This may involve designing the device in a way that allows users to easily reconfigure it for different purposes, or adding features such as programmability or updateability to allow users to easily modify the device over time.
Performance: The device must be effective in achieving its intended purpose, which is real-time audio processing. This may involve considering factors such as audio quality, processing speed, and power efficiency.
Ergonomics: The device should be designed with respect to the user's physical and cognitive abilities to enhance the overall user experience. This may involve factors such as the size and shape of the device, the layout of controls and displays, and the overall user experience.

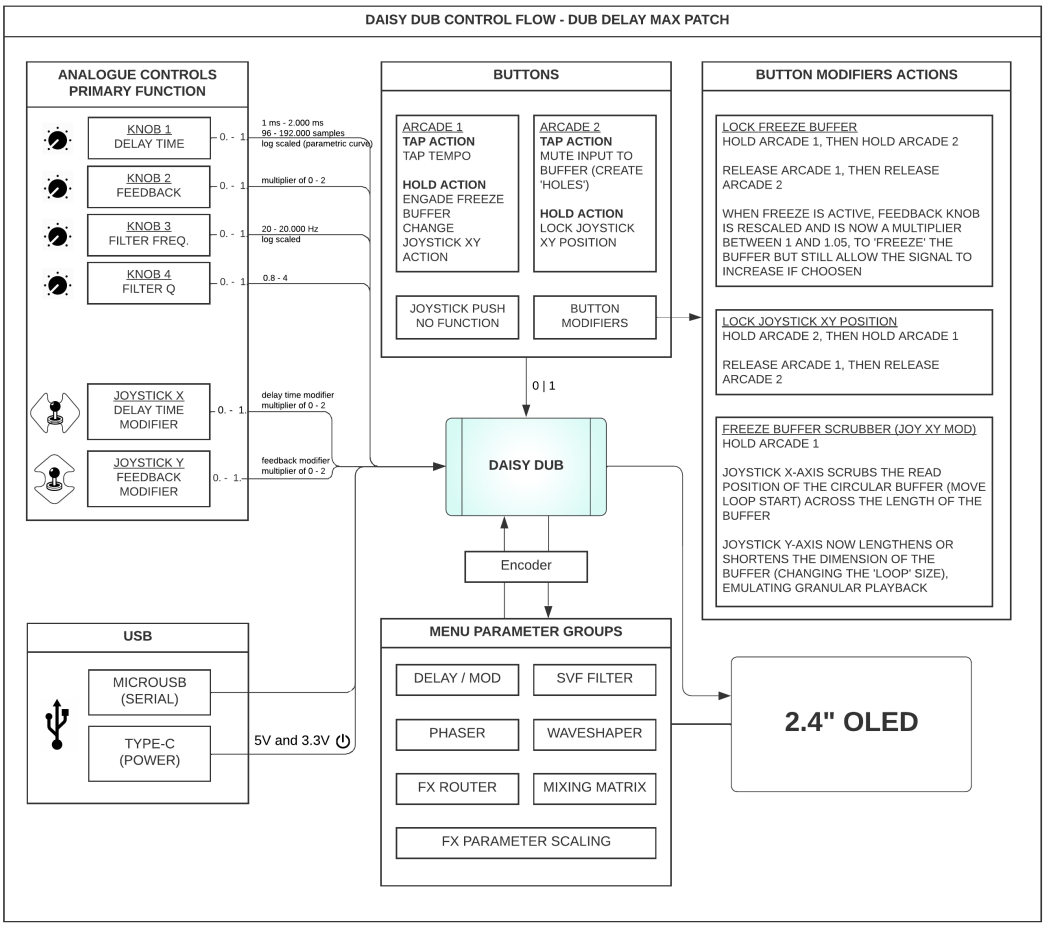


Figure 3.1: Daisy Dub delay patch - control flow diagram with scaling and features

Format and digital signal processing

Daisy Dub is housed in a portable case, with a rugged and durable design suitable for live performances. It is powered by USB Type-C, allowing it to be easily powered from a laptop or portable battery pack. The unit uses a high-resolution digital audio processor, similar to the one used in the Strymon El Capistan, to provide a high-quality and detailed sound. Additionally, the unit offers quadrophonic audio processing on both input and output, allowing users to create and manipulate 4-channel audio signals.

Delay and modulation

Daisy Dub offers a delay patch with sonic quality similar to the SDE-2500 and El Capistan. It also offers additional modifications to the delay, with a freeze buffer and freeze buffer scrubber function (see figure 3.1) to provide even more flexibility and creativity. The unit offers a range of modulation effects, like the SDE-2500 and El Capistan, but with even more options and control. It also allows users to choose custom modulation parameters using an interactive graphical interface, similar to the secondary button features on El Capistan but with a much easier overview having it on a display.

Control and connectivity

Daisy Dub has four knobs for performance control, allowing users to adjust delay and effects parameters on the fly. It also has an encoder for specific sound design menu diving, allowing users to access various advanced sound design options. The unit has two arcade buttons and a thumb joystick for performance interaction, allowing users to create dynamic and expressive effects. An OLED screen is used for spectrum analysis and menu diving, providing users with a clear and intuitive interface. Daisy Dub offers a range of connectivity options, including stereo and mono input and output and quadrophonic input and output.

Memory and programmability - unfulfilled

Daisy Dub should be able to save and recall entire effect chains and create custom preset libraries. The unit is programmable and updateable, allowing users to customize and update the unit's algorithms and features. This can be done using a computer and a USB connection.

Compatibility and sound quality

Daisy Dub is compatible with various audio equipment, including mixers, amplifiers, and audio interfaces. Its range of input and output options, including stereo, mono, and quadrophonic, allows for flexibility in various setups. The unit offers a high-quality sound, with a range of delay and effects algorithms designed to provide high clarity and detail. Its high-resolution digital audio processor, quadrophonic audio processing, and interactive graphical interface allow users to create a wide range of sounds with great depth and nuance.

Flexibility and ease of use

Daisy Dub's quadrophonic audio processing and interactive graphical interface allow users to create and manipulate complex multi-channel audio signals with great flexibility and creativity. This could be particularly useful for electronic music producers, sound designers, and experimental musicians looking to push the boundaries of audio processing. The unit's control scheme, which combines traditional knobs and buttons with an interactive graphical interface, provides users with various options for adjusting and manipulating parameters. This makes the unit easy to use for both beginners and experienced users and allows for intuitive and precise control.

Portability

Daisy Dub's compact and portable format, combined with USB Type-C power and programmability, makes it easy to use in various settings. It can be easily transported to gigs, recording sessions, and other events and can be easily integrated into a wide range of audio setups.

Chapter 4

Design and implementation

The Design and implementation chapter of this thesis aims to provide a comprehensive overview of the design process and technical considerations involved in the creation of the Daisy Dub. One of the main design considerations for a modular and updateable real-time audio effect like the Daisy Dub is usability. It is important that the device is easy to understand and use for the target user group, which in this case includes music producers and performers. This may involve considering factors such as the layout of controls and displays, the availability of documentation and tutorials, and the overall user experience.

Another important design consideration is repairability. Given the high demand and use that music producers and performers may have for the device, it is important that it can be easily repaired or maintained in the event of a problem. This may involve designing the device in a way that makes it easy to access and replace components, or including features such as diagnostic tools or self-testing capabilities. Modularity is also an important design consideration for the Daisy Dub. The ability to easily add or remove components from the device allows users to customize it to their specific needs and adapt it to different situations or use cases. This may involve designing the device in a way that allows users to easily swap out different components or modules, or creating a system that allows users to easily customize the device.

Flexibility is another important design consideration for the Daisy Dub. The ability to adapt the device to different situations or use cases is crucial for music producers and performers who may need to use the device in a variety of different contexts. This may involve designing the device in a way that allows users to easily reconfigure it for different purposes, or adding features such as programmability or updateability to allow users to easily modify the device over time.

Performance is also a key design consideration for the Daisy Dub. The device must be effective in achieving its intended purpose, which in this case is real-time audio processing. This may involve considering factors such as audio quality, processing speed, and power efficiency. Lastly ergonomics is another important design consideration for the Daisy Dub. The device should be designed with respect to the user's physical and cognitive abilities in order to enhance the overall user experience. This may involve considering factors such as the size and shape of the device, the layout of controls and displays, and the overall user experience.

4.1 Demand specification

In order to try and live up to these demands to the design and implementation a demand specification was created, to give an overview of the most important goals that should be met in the development. The demands will be split into three categories; functional demands, quality demands, and environmental demands. These demands will be tested after implementation to ensure that they are met.

4.1.1 Functional demands

In order to ensure that the Daisy Seed is performing at a baseline level, some functional demands are set up. These demands will be in relation to the electric and mechanical design. To ensure that the Daisy Dub is able to stay relevant for as long as possible, it would be preferable to use a USB Type-C connection as this is the emerging new standard for power and data delivery. For sound input and output, it was decided to have the capacity for quad audio. This means that there are four input channels, as well as four output channels. The initial Daisy Dub prototype was playable even without any input signal, as a small amount of noise is injected into the signal, which becomes audible if the user turns some of the controls, like the feedback up high enough. This is a feature that will be kept going forward. Being playable without an input also means that the delay time is not synchronized however, it should also be possible to synchronize with a performance, or production if that is desired. This would add an extra use case for the product, and make it usable in more scenarios. As the Daisy should be usable in a performance setting, it would be desirable to add a way for the users to store and recall certain settings easily, so that the user doesn't have to set everything up by ear at every performance. Because the goal of this project is to create a product that is customizable and hackable, exposing the analog inputs on a PCB for the Daisy Dub would be great, as it would allow the user to switch out the user interface controls. Finally one of the most requested features during the testing was the ability to modify the delay time, which means that the user should be able to get into the half or double value of the current delay time. It was also requested that this feature should include the ability to go to dotted or triplet notes.

With these considerations in mind, the following demands were formulated:

1. Power should be delivered to the device through a USB Type-C port
2. It should be possible to update the software on the Daisy Seed through a USB Type-C port
3. Quad audio capability
4. Usable without external inputs
5. It should be possible to send MIDI data to the device to synchronize the tempo with a performance
6. The user should be able to store and recall presets
7. Expose analog inputs on the PCB
8. Delay time modifiers

4.1.2 Quality demands

The functional demands are important to ensure that the Daisy Dub works as a device, but it also needs to be desirable to use for potential customers. To ensure that this is the case some quality demands need to be set. In section 2.7.1 the initial prototype reached a SUS score of 79.5. Ideally the next iteration should at least retain this score but preferably improve it. It is also important to consider the use cases for the product, and for the Daisy Dub it is primarily for music producers,

performers and DJs. As is evident in similar products discussed in section 2, the form factor is fairly small, as it allows for the product to be easily carried around, but still allows for big enough controls that it is possible to comfortably use in a performance setting. In these same settings it is important that the product does not have a low-friction underside so that it slides around on the table at the slightest disturbance. This is luckily easily fixed by either giving the Daisy Dub rubber feet, or a type of non-slip surface on the bottom. Should the user wish to flash new software onto the Daisy Seed they have to press two buttons directly on the board. The final design for the Daisy Dub will not have the Daisy Seed exposed, and therefore it would be a hassle to access these buttons. To combat this some software should be implemented so that this bootloader can be accessed without removing some of the casing.

With these considerations in mind, the following demands were set:

1. Should achieve a score of at least 79.5 on a SUS scale during user testing
2. The form factor should be small enough to carry around easily, but big enough to comfortably use for performance
3. Some form of anti-slip
4. It should be possible to flash new software to the product without accessing the Daisy Seed within

4.1.3 Environmental demands

The environmental demands for this product fall under the concept of reducing the number of similar products that an end user might need. The Daisy Dub should be customizable both in terms of appearance, but also in terms of control interfaces, to allow potential users to tailor it to their needs. The hope is that different algorithms should be developed for the Daisy Dub, so that a potential user could consolidate their hardware needs into this product, instead of having separate products for each effect. With this in mind, it is important that the circuitry is fairly generalized, and that any specifics that might be needed for a certain effect are done in code. This would allow the user to simply put new software on the unit in order to change its use.

With these considerations in mind, the following demands were established:

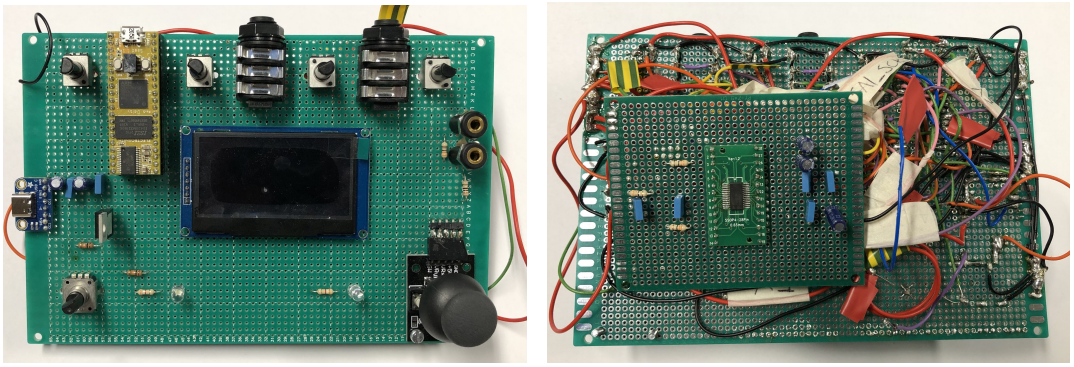
1. The parts most susceptible to damage or customization should be easily replaceable
2. Should be as non-specific in design as possible
3. User should be able to put new code onto the device

4.2 Electrical design

4.2.1 Considerations

The electrical design of the Daisy Dub has significant implications for several aspects, such as audio processing, user interaction, and repairability. This section will discuss the relevant electrical design considerations for each of these key areas.

Audio processing is a crucial aspect of the product's electrical design. The system responsible for real-time audio processing must meet the performance and quality standards of the target audience. This may entail careful consideration of factors such as the choice of audio processing algorithms, suitable hardware and software for processing, and the overall architecture of the audio processing system. It is also essential to consider the audio input and output interfaces, as well as the means of connecting the device to external audio sources and destinations.



(a) Front **(b)** Back

Figure 4.1: Daisy Dub prototype

User interaction is another significant aspect that requires careful consideration during the electrical design of the product. The user interaction system must be intuitive, responsive, and easy to use, allowing the user to interact with the device effectively. This may involve considerations such as the arrangement and layout of the device's various controls and interfaces, the choice of sensors and actuators, and the design of the user interface software. The means of connecting the device to external control sources and destinations, such as computers or other devices, must also be considered.

In addition to audio processing and user interaction, signal processing requirements must also be considered in the electrical design of the Daisy Dub. Suitable signal processing algorithms, hardware, and software must be selected to meet the target audience's needs. The signal input and output interfaces, as well as the means of connecting the device to external signal sources and destinations, must also be considered.

Finally, the electrical design must consider the product's repairability. This may entail designing the system for easy diagnosis and repair, using standard and readily available components, and providing suitable documentation and support for maintenance and repair. By designing the product with repairability in mind, the costs and downtime associated with maintenance and repair can be reduced, and the overall reliability of the product can be improved.

In conclusion, the Daisy Dub's electrical design requires careful consideration of several factors, including audio processing, user interaction, signal processing, and repairability. By taking these factors into account, it is possible to develop a functional, reliable, and user-friendly product that meets the needs and preferences of the target audience.

4.2.2 Implementations

This section will describe the work and the troubles encountered while implementing certain hardware aspects of the Daisy Dub. Components and circuits integrated into the device were tested on breadboards and prototyping boards. One of the prototypes is shown in figure 4.1. The components and circuits used required 3.3V and 5V, and an estimate of the current needed for the device is 150mA.

USB Type-C port

The decision to power the Daisy Dub device through a USB Type-C port was motivated by the need to conform to contemporary technology standards and enhance user convenience by providing a standardized port. USB Type-C supports power delivery up to 48V/240W and rapid data transfer. During the process of integrating the desired port, it was discovered that the data pins (D- and D+) exposed on the Daisy Seed development board are OTG (On-The-Go) pins which only allow it to be used as a USB host, and not as a USB device, which was the intention for the Daisy Dub.

Considering these limitations, the developed prototype incorporates a USB Type-C port for power delivery and utilizes the onboard micro USB port for data communication. This configuration should suffice until a more permanent solution is achievable since the onboard connection is only necessary when flashing firmware updates to the Daisy Seed board.

Since the device needs to be powered by 5V, a simple resistor method is used where an upstream facing port must connect a valid pull-down resistor to ground to both CC1 and CC2 pins[24]. The value of the resistors must be 5.1kΩ for the port to provide 5V and up to 3A.

PCM3060 audio codec

A feature desired for implementation on the Daisy Dub is the ability to have four audio input and output channels, as it would broaden the range of applications for which it could be utilized. The Daisy Seed hardware platform, however, provides only two audio input and output channels, thereby necessitating the incorporation of an external audio codec within the device's design. Specifically, Texas Instrument's PCM3060 audio codec[1] has been selected as the preferred codec for this purpose; a 24-bit asynchronous stereo codec with a 96/192kHz sampling rate.

The configuration and control of the codec are facilitated by the utilization of a 2-wire I2C communication protocol, while the transfer of clocks and audio signals occurs via the Serial Audio Interface (SAI). It is expected that the expansion of audio channels will result in an enhanced user experience, thereby positioning the Daisy Dub device as a more versatile and attractive option for a broader range of potential users.

User interface

Daisy Dub's user interface includes several components; four potentiometers, a rotary encoder, a thumb joystick and two arcade buttons which are connected to analog and digital input pins on the Daisy Seed board. The user interface also includes two LEDs and a 2.42″ OLED display with an SSD1309 driver. These components are used to give visual feedback to the user. The LEDs are connected to digital pins on the Daisy Seed board while the OLED display communicates with the board over Serial Peripheral Interface (SPI) bus protocol.

MIDI

The inclusion of MIDI functionality within an audio device offers a multitude of opportunities for the integration of the device within an existing setup, syncing the tempo between devices, as well as facilitating external control over its features. MIDI signals in this implementation are transferred over a 3.5mm audio jack.
Receiving MIDI data from an external device through an audio jack involves transmitting the raw stream of bytes from the MIDI controller to a microcontroller via a Schmitt trigger output optocoupler. The Schmitt trigger isolates the MIDI stream by flashing 0 and 1 bits through an LED and then

[1] `https://www.ti.com/product/PCM3060`

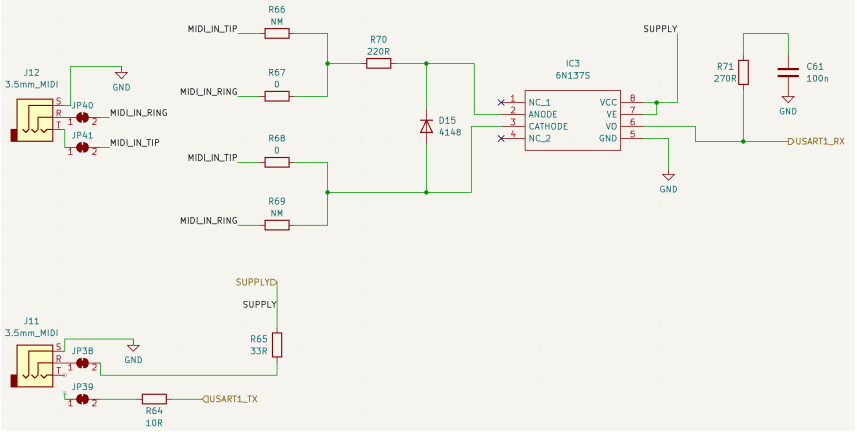


Figure 4.2: MIDI circuits schematics

converts them into the universal synchronous and asynchronous receiver-transmitter (USART) serial communication protocol, which facilitates the transmission of MIDI messages [14]. The output MIDI connection is established between the USART transmit pin on the microcontroller and an external device, allowing for the transmission of MIDI messages.
The schematics for these circuits (see figure 4.2) was adapted from the Daisy Patch[2], a product designed by Electrosmith that uses the Daisy Seed board and implements the MIDI functionality over an audio jack. Due to time constraints, these circuits were not tested before manufacturing.

Analog circuits

Through the supervision and guidance provided by Sound Hub Denmark, it was decided to add analog circuits to the Daisy Dub in order to properly handle audio inputs and outputs in a way that matches production-ready products. Three types of circuits were made: a bias generator, audio input buffers, and differential to single-ended (DSE) conversions. The bias generator is a simple circuit that uses a voltage divider to create two different bias outputs, as these were needed in the other circuits. The schematic of a bias generator is shown in figure 4.3. For the other two types of circuits, two variants were made: one for the Daisy Seed, and one for the audio codec.

Input buffers
These circuits take their input from the input jack connectors on the Daisy Dub, and put a buffer between those connectors and their destination, as well as filter the signal. The buffering is done to prevent unassociated parts of the electronic circuits from affecting the signal, and the filtering is done to get rid of any potential low frequency noise, as well as any DC offset that might be present. The schematics for these two circuits can be seen in figure 4.4.

Differential to single-ended conversion
These circuits are on the opposite end of the signal chain, and take the output from either the Daisy Seed or the audio codec, and output them to the output jack connectors on the Daisy Dub. These combine signal's plus and minus, effectively removing the shared information of those inputs. The schematics for these circuits are shown in figure 4.5.

Validation

[2] `https://www.electro-smith.com/daisy/patch`

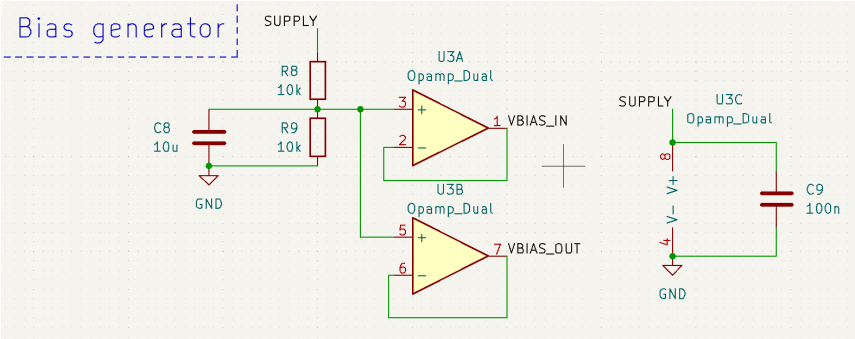
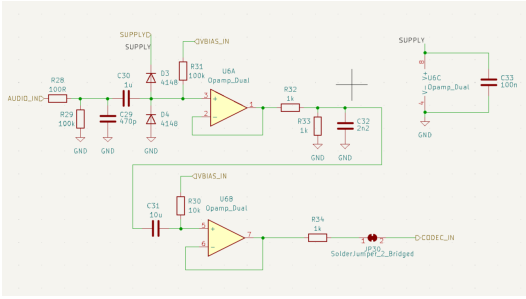
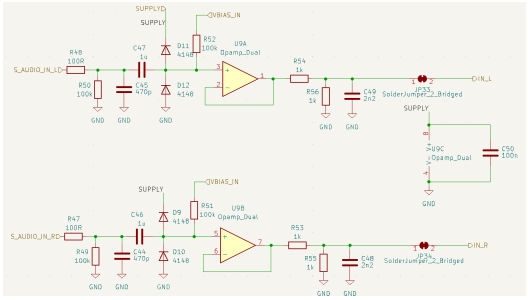


Figure 4.3: Bias circuit schematic

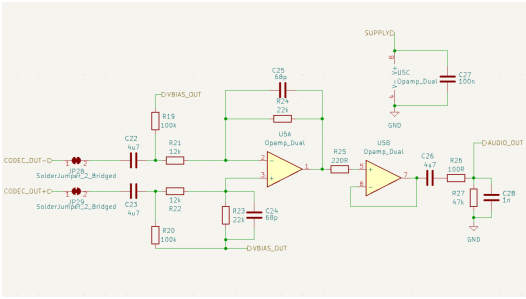


(a) The input buffer that passes the input signal from a jack connection to the codec.

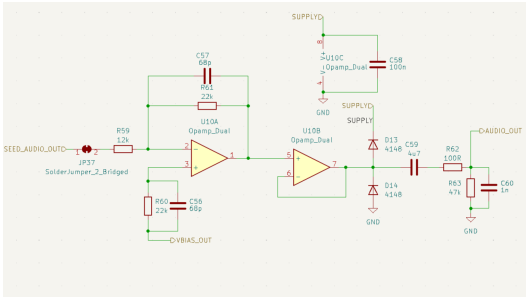


(b) The input buffer that passes the input signal from a jack connection to the Daisy Seed

Figure 4.4: Input buffer circuits schematic



(a) The DSE conversion circuit; passes output signal from the codec to a jack connection



(b) The DSE conversion circuit; passes output signal from the Daisy Seed to a jack connection

Figure 4.5: Differential to single-ended conversion circuits schematic

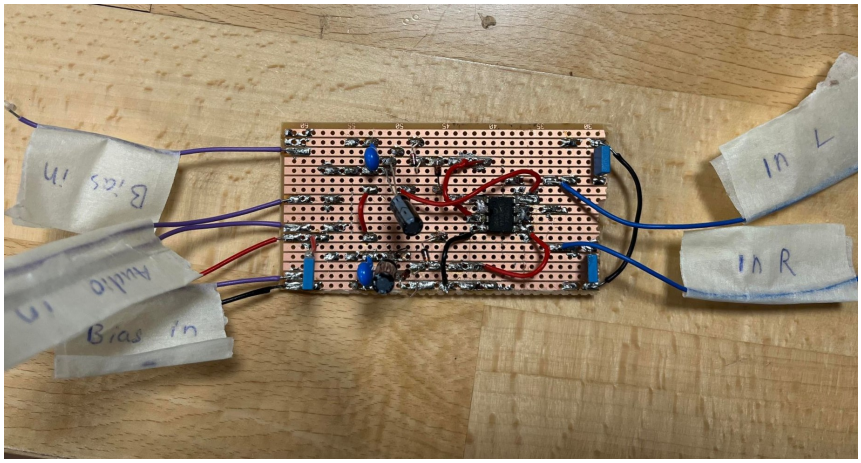


Figure 4.6: Input buffer circuit (Daisy Seed) prototyped on a stripboard

In order to ensure that the analog circuits have the intended effect, they were validated on prototyping boards. One of the test circuits can be seen in figure 4.6.

The bias circuit was tested by connecting it to power and ground. The results produced half of the input voltage of both outputs, which is what was expected. The bias generator circuit was tested first, as the output of this is used in the other circuits.

To test the input buffer circuits, two tests were conducted. First, the input was given a sine sweep from 20 Hz to 100 kHz, and the output was analyzed. Both the sweep and analysis were done on a Keysight DSOX1102G Digital Oscilloscope. Sine sweep was set to end at 100 kHz instead of 20 kHz, as the only options for the end frequency were 10 kHz and 100 kHz. The latter was selected because the response from 10 kHz to 20 kHz needed to be tested. The analysis of this sweep showed that the signal on the output matched the input signal, thus validating the circuit. The next test was providing the circuit with an actual audio input using a jack connection, and sending the output to a pair of speakers. This was a less objective test and was done to see if we noticed any artifacts in the audio or other issues that may be difficult to detect purely from an analysis of a sine sweep.

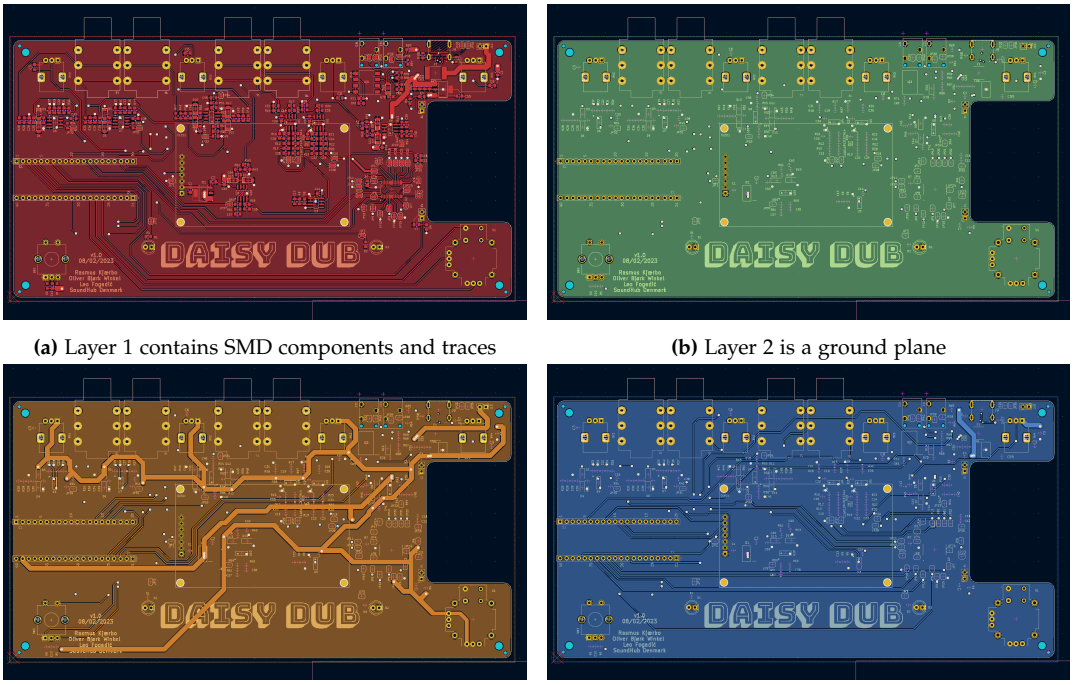
The DSE circuits were tested by running a test similar to the one done for the input buffer circuits. Due to the nature of how the circuit works and us only making one for the testing, it would not sound right to do a test with speakers, so for this, we settled with getting the correct output using a simplified input.

Assuring that all the analog circuits were giving the expected results, the designs were ready to be implemented on a printed circuit board.

4.2.3 PCB Production

While and after prototyping and validating circuits and components, a PCB design was made using KiCad[3], an open-source electronics design automation suite. The process included designing the circuits in the schematic editor according to prototypes built on breadboards and prototyping boards. The full schematic can be found in Appendix D.
Subsequently, footprints for every component in the schematic had to be either sourced online or, if not found, designed by examining the dimensions of the component found in its datasheet. Additionally, 3D models representing each component were sourced to aid the mechanical design of the

[3]https://www.kicad.org/



(a) Layer 1 contains SMD components and traces

(b) Layer 2 is a ground plane

(c) Layer 3 contains the power supply and other traces

(d) Layer 4 contains the rest of the traces

Figure 4.7: PCB layers

enclosure. After sourcing and creating all of the aforementioned files, the PCB layout was designed. Due to the complexity of the board, and the choice to allocate one layer for a ground plane, the design was split into four layers. The front layer contains all surface-mounted (SMD) components and several through hole components that needed to be on the top of the board, such as the user interface components. The front layer also integrated the majority of the traces. The second layer was designed to be used as a ground plane, while the third layer integrated wider power supply traces and other traces. The back layer was designed to include through-hole components which were supposed to be on the bottom side of the PCB, like the audio jacks, MIDI jacks, and pin headers for the Daisy Seed development board. The bottom layer also included the rest of the traces. The majority of traces are 0,3mm wide, while the power traces are 2mm wide. Besides components and tracks, closed solder jumpers were added to several connections on the board in case a connection had to be altered after manufacturing. Pin headers with long legs were also used for the Daisy Seed board in case of a need to rewire components and cirucits. All layers of the board are shown in figure 4.7.

Five boards were manufactured by JLCPCB[4], and all surface-mounted components were assembled by JLCPCB's assembly service[5]. Through-hole components on the top side of the PCB were also assembled and soldered by JLCPCB, while the components on the bottom side were soldered after receiving the boards.

[4] `https://jlcpcb.com/`

[5] `https://jlcpcb.com/smt-assembly`

4.3 Software design

The software framework for the Daisy Dub project is primarily written in Gen by Cycling '74, an addon for their visual programming language, Max. Gen is a visual programming language that allows for the creation of custom digital signal processing algorithms, and is an intuitive way to design new audio effects. The visual interface makes it easy to understand how signals are routed and processed, and the real-time preview allows for rapid iteration and fine-tuning of the algorithm. By integrating the gen code with the Daisy Dub hardware, users will have a powerful and flexible platform for creating their own custom audio effects.

The Oopsy package from Electrosmith is another key tool for implementing custom algorithms on the Daisy Dub hardware. It is a Max package that allows for the creation of custom firmware for the Daisy Seed platform. With Oopsy, users can easily create new effects using gen code and deploy them to the hardware in a streamlined and efficient manner. The package includes a library of pre-built objects and examples to help jumpstart the development process. Integrating gen and Oopsy together provides a complete and flexible solution for developing custom effects for the Daisy Dub hardware. The visual programming interface of gen allows for rapid prototyping and fine-tuning of the effect algorithm, while Oopsy provides a robust and reliable way to deploy the custom effect to the hardware. Together, these tools make it easy to create unique and powerful audio effects that can be easily shared with the wider community.

4.3.1 Daisy Seed by Electrosmith

The Daisy Seed platform is a highly suitable choice for developing Daisy Dub due to its open-source nature, strong focus on audio processing, and ability to handle real-time processing. One of the key benefits of using the Daisy Seed platform is its open-source nature, which allows for a high level of flexibility and customization in the development process. This is especially important in the case of Daisy Dub, as the device is intended to be both modular and updateable. By utilizing an open-source platform such as Daisy Seed, developers have the freedom to customize and tweak the firmware as needed, rather than being limited by proprietary restrictions. In addition to its open-source nature, the Daisy Seed platform is also well-suited for audio processing applications due to its powerful and efficient hardware. The platform features a powerful ARM Cortex-M7 MCU, running at 480MHz, capable of handling high-resolution audio processing with minimal latency. This makes it ideal for use in real-time audio effects such as Daisy Dub, which requires rapid processing to produce seamless, natural-sounding effects and delays. It includes a 24-bit analogue-to-digital converter (ADC) and a 24-bit digital-to-analogue converter (DAC) at 96kHz (AC-Coupled). It runs on a powerful 32-bit processor and boasts a plethora of GPIOs. This makes it ideal for developing real-time audio effects such as Daisy Dub. In addition to its audio processing capabilities, the Daisy Seed platform also includes a range of connectivity options, including USB and microSD card support. It has connectivity for MIDI I/O, PWM outputs, and various serial protocols for external devices (SPI, UART, SAI/I2S, I2C) as well. This allows for easy integration with other devices and systems, as well as convenient data storage and transfer. It is programmable in C++, Arduino, Max/MSP Gen, Pure Data, and Rust leaving plenty of options for both paid programming environments as well as open-source. Overall, the combination of the Daisy Seed platform's open-source nature, strong focus on audio processing, and ability to handle real-time processing make it an excellent choice for the development of Daisy Dub. By leveraging these capabilities, developers can create a highly customizable, efficient, and effective modular audio effect that is ideal for use in music production and performance.

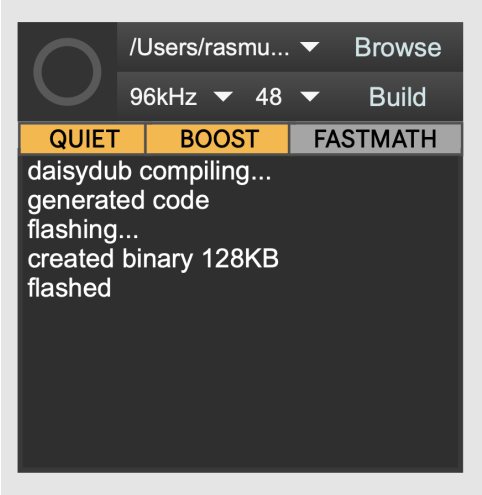


Figure 4.8: Oopsy abstraction from the Oopsy Max Package by Electrosmith. Compiles gen~ patchers to C++ code and flashes the Daisy Seed development platform.

4.3.2 Gen by Cycling '74

Gen is a programming language and visual programming environment developed by Cycling '74 for creating interactive, real-time audio and visual processing applications. It is particularly well-suited for use in the Max programming environment, which provides a graphical user interface (GUI) for building and interacting with Gen programs[3, 6, 7, 15, 29]. Gen is based on a functional, dataflow programming model, in which complex programs are created by composing simple functions and processing blocks. It provides a large library of built-in functions and objects that can be used to create a wide range of audio and visual processing applications, and also allows users to create their own custom functions and objects. One of the key features of Gen is its ability to process data in real-time, making it well-suited for use in interactive audio and visual applications. It provides a variety of functions and objects for generating and processing audio and video signals, as well as for interacting with external devices and sensors. Gen also includes support for a number of advanced features, such as multithreading, GPU acceleration, and network communication, which can be used to create more complex and powerful processing applications. It is also highly customizable, with a wide range of options for controlling the appearance and behavior of Gen programs. Gen is a powerful and flexible tool that is widely used by artists, musicians, and creative coders for a variety of applications. It is particularly well-suited for use in real-time audio and visual processing, and is a key component of the Max programming environment.

4.3.3 Oopsy - A Max package for flashing gen patches

The Oopsy Max package is a software development kit (SDK) developed by Electrosmith for compiling Gen patches to functional and efficient C++ code running on the Daisy platform[9]. Oopsy is designed to simplify the process of developing applications for the Daisy platform, providing a set of tools and libraries for interacting with the hardware and software components of the platform. gen patches are compiled and flashed directly to the Daisy Seed development platform via the Oopsy abstraction (see figure 4.8). The package is readily available via Electrosmith's GitHub

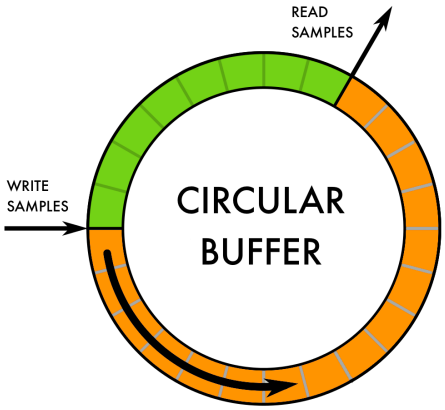


Figure 4.9: Circular Buffer Diagram - the buffer rotates and the write position is fixed while the read position can be changed by the user.

repository[6].

4.3.4 Circular buffer delay

A circular buffer delay audio effect is a type of audio processing effect that allows for the creation of delay and echo effects in real-time. It works by storing a certain amount of audio samples in a circular buffer and continuously reading and writing to this buffer at different points in the audio signal, overwriting the oldest data with new data as it becomes available (see figure 4.9). This allows for efficient storage and retrieval of data and real-time processing of the data stream. One of the key advantages of using a circular buffer for delay effects is that it allows for low-latency processing, as the delay time can be changed on a per-sample basis. This makes it well-suited for use in live performance settings where delay times may need to be adjusted quickly and accurately, and especially for those looking to explore creative uses of aggressive feedback, self-oscillation, and feedback-path signal processing. Adding in parameters to manipulate both the delay time and feedback amount allows for creative exploration of sounds, which can be used to create delay, echo, or self-oscillation. The use of a circular buffer delay in the Daisy Dub allows for either simple delay effects reading and writing samples to the circular buffer at a specified delay time, or a more complex type of delay involving filtering, saturation, phasing, or modulating the delayed signal in various ways.

The ability to process audio in real-time with low latency is an essential aspect of the design and implementation of performance-oriented audio effects and will continue to play a central role in the development of new and innovative audio-processing tools [4, 6, 7, 23, 26, 29].

4.3.5 Effects in the feedback path

The implementation of effects such as a state variable filter (SVF), waveshaping and phasers in the feedback path of a delay can greatly enhance the creative potential of the effect. In the engineering and programming aspect of a circular buffer delay, adding effects like a SVF, waveshaping and phaser

[6] `https://github.com/electro-smith/oopsy`

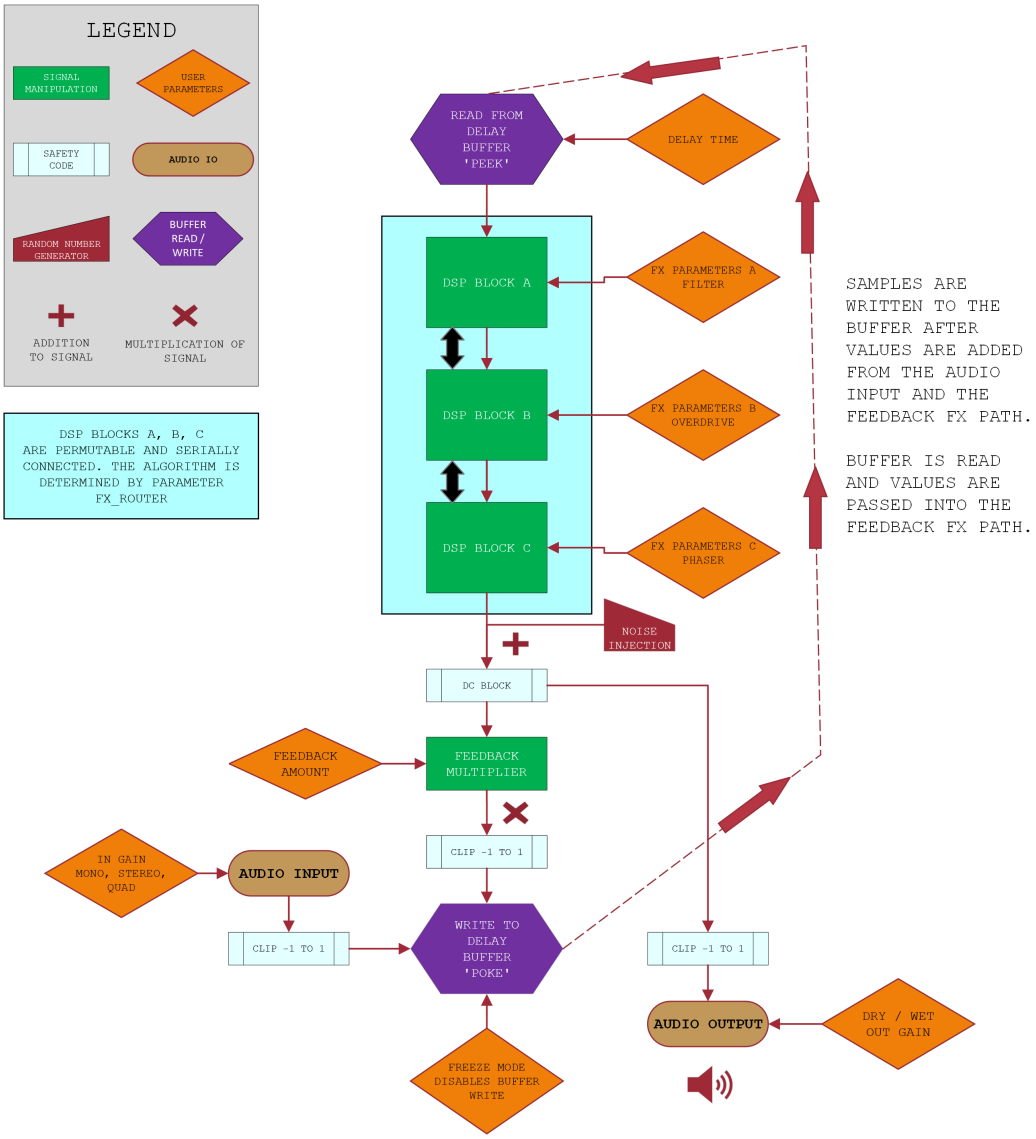


Figure 4.10: Daisy Dub DSP flowchart for a single audio channel. The circuit is doubled for the second input channel for true stereo processing. The effect blocks A, B, C are serially connected and interchangeable via the fx_router parameter, so the user can change the order of the effects; filter, waveshaping, phaser.

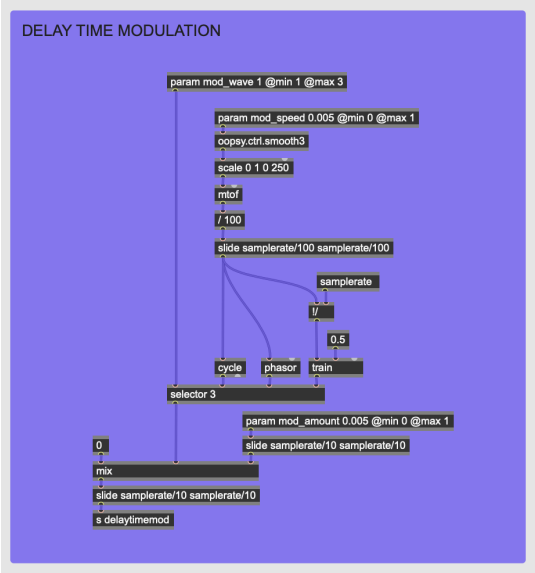


Figure 4.11: Delay time modulation section of the gen patch

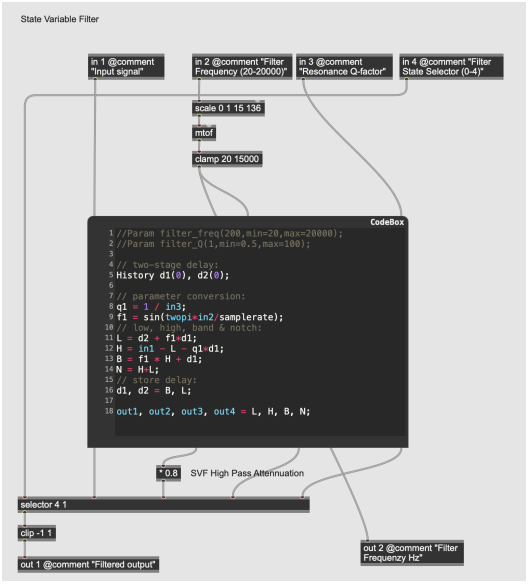
in the feedback path can have a significant impact on the sound and character of the delay (see figure 4.10) [4, 6, 7, 26, 29].

A state variable filter: is a type of filter that allows for the simultaneous control of multiple filter parameters, such as cut-off frequency and resonance. With simple math, it is possible to tap out a low-pass, hi-pass, band-pass and notch signals from the same filter algorithm (see fig 4.12a). When added to the feedback path of a delay, it can shape and manipulate the frequency spectrum of the delayed signal, creating a range of tonal possibilities. Increasing the filter resonance and utilising the feedback of the delay will make the device self-oscillate and generate a tone around the filter's cutoff frequency [4, 6, 7, 26, 29].

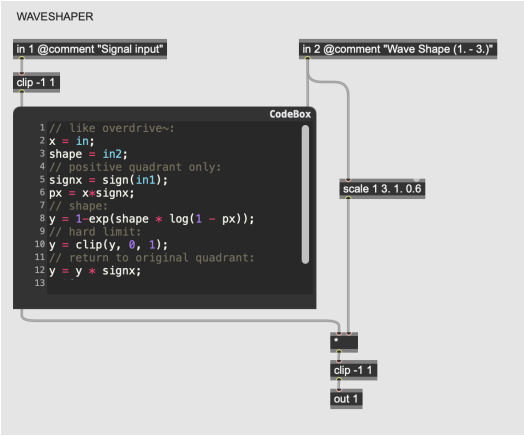
Waveshaping: is a non-linear process that adds harmonics to the signal and can be used to create a range of saturated and overdriven sounds (see figure 4.12b). Adding a waveshaper to the feedback path can create a sense of grit and dirtiness in the delayed signal, resulting in character, warmth, or overdrive [4, 6, 7, 26, 29].

Phaser: is an effect that uses a series of all-pass filters (see figure 4.13b) to shift the phase of different frequency bands in the signal. When added to the feedback path, it can create a sweeping and swooshing sound, adding movement and dimension to the delay (see figure 4.13a) [4, 6, 7, 26, 29].

Messing with the feedback path of a delay can be 'dangerous', though, as it has the potential to create extreme and potentially damaging levels of gain. However, when used with care and intention, adding effects like a SVF, waveshaping, and phasers in the feedback path can be incredibly beautiful and artistic. These effects can add depth, character, and complexity to the sound and can be used to create a wide range of tonal and timbral variations. When used creatively and judiciously, the feedback path of a delay can be a powerful tool for sculpting and shaping sound expressively and imaginatively. If not adequately controlled by the developers, the feedback can quickly become overwhelming and cause the system to become unstable. In uncontrolled circumstances, it is essential for the engineer or musician to have a strong understanding of the technical aspects of the system

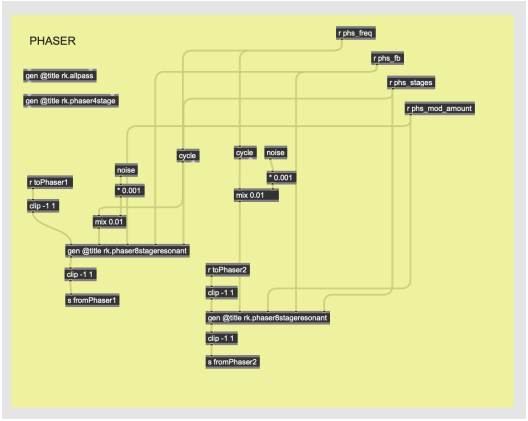


(a) State Variable Filter with user-changeable cutoff frequency and filter resonance

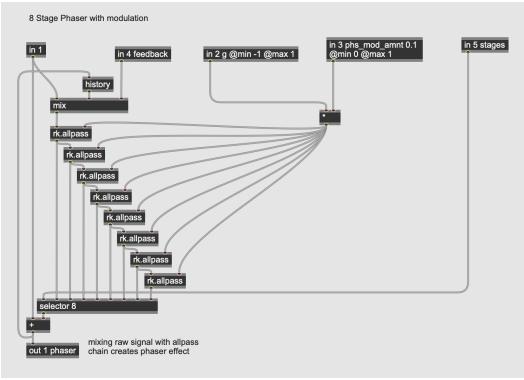


(b) Waveshaper algorithm implementation in gen codebox

Figure 4.12: SVF and waveshaper algorithm



(a) Stereo phaser implementation in the feedback path of the delay



(b) 8-stage parametric resonant phaser created by a series of all-pass filters

Figure 4.13: Phaser implementation and gen algorithm

and to use caution when experimenting with these techniques not to cause the system to become unstable. In our case, the extremities of the parameters have been adjusted so the user can freely explore the full range and capability of the device. See figure 4.15 for the complete gen patch overview.

One of the pioneers of dub music, King Tubby (as mentioned in 1.2.3) was known for his innovative use of feedback and effects in the feedback path of delays. He would often manipulate the feedback and effects in real-time during his performances, creating a constantly evolving and dynamic sound. This use of feedback and effects in the feedback path allowed him to craft a unique and memorable sound that has had a lasting impact on music to this day and onwards.

4.3.6 Signal flow of Daisy Dub

Daisy Dub consists of various inputs and outputs. In figure 4.14 an overview of the general user interface and inputs/outputs (IO) are presented.

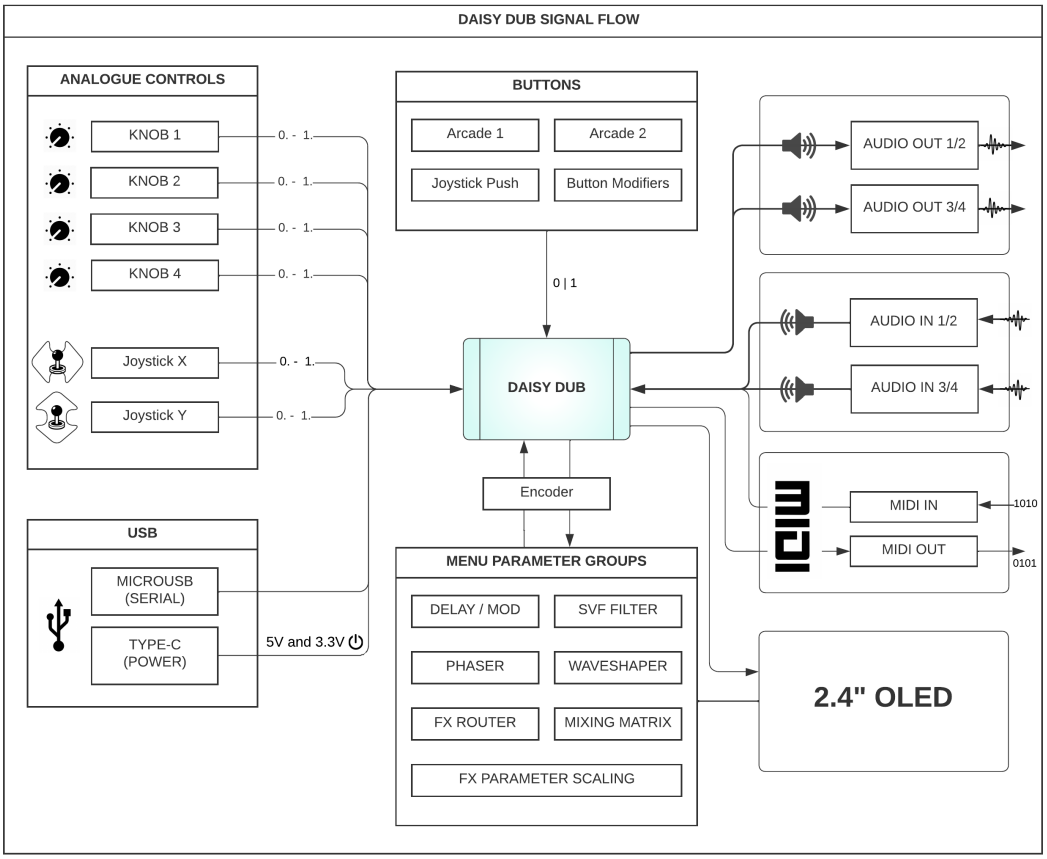


Figure 4.14: Daisy Dub signal flow overview

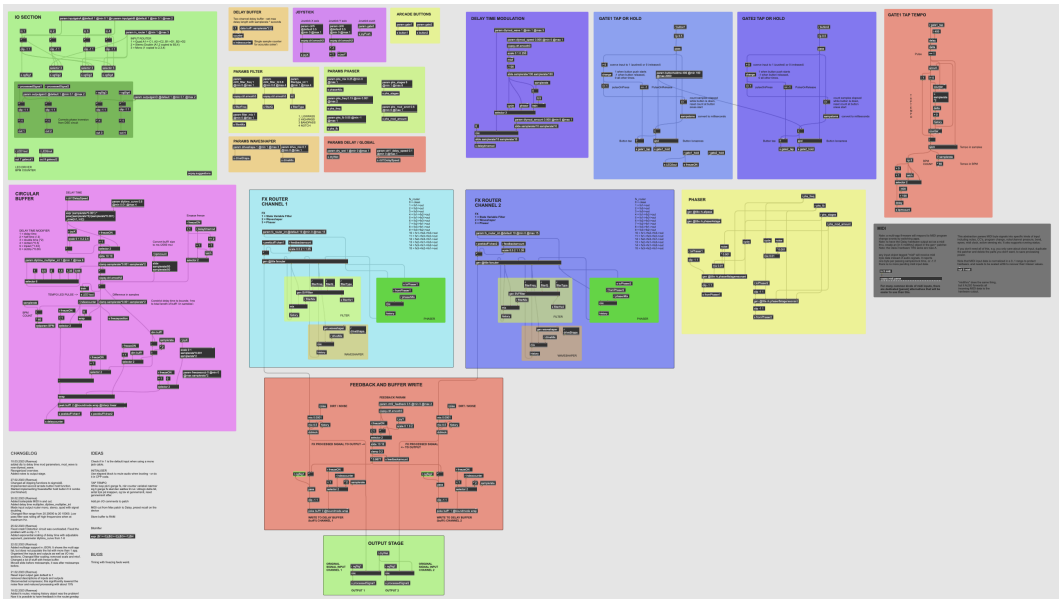


Figure 4.15: Daisy Dub gen patch overview

4.3.7 DSP evaluation of the Daisy Dub

Evaluating DSP is crucial in ensuring high-quality audio signal processing for music. This chapter focuses on the DSP evaluation of the audio algorithms used in the Daisy Dub hardware and the overall audio circuits constructed. This chapter aims to evaluate the performance of the DSP code and hardware circuitry in terms of accuracy, noise levels, and frequency response and to ensure that the resulting audio output meets the high-quality standards required for music production and performance. Through this evaluation, we can ensure that the Daisy Dub hardware is capable of producing high-quality audio effects and is suitable for use by musicians and audio engineers.

Two tests are conducted; one using a tone generator from a laptop running Max and measurements taken with an Audio Precision APx517 acoustic audio analyzer at Sound Hub Denmark. The other was conducted with a laptop and a Fireface UCX sound card, sending audio out of the sound card into Daisy Dub input 1 and returning the processed signal back to the sound card from Daisy Dub output 1 (see figure 4.17b). To establish a baseline for testing, a pass through test with only the sound card roundtrip latency was performed (see figure 4.17a). The recordings were made with Ableton Live and analysed with iZotope Insight[7] (see figure 4.16).

[7] `https://www.izotope.com/en/products/insight.html`

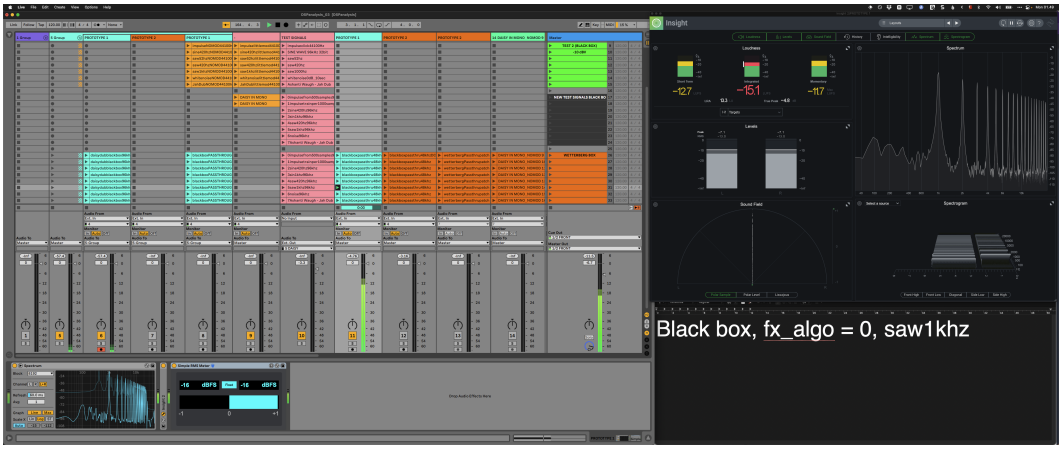
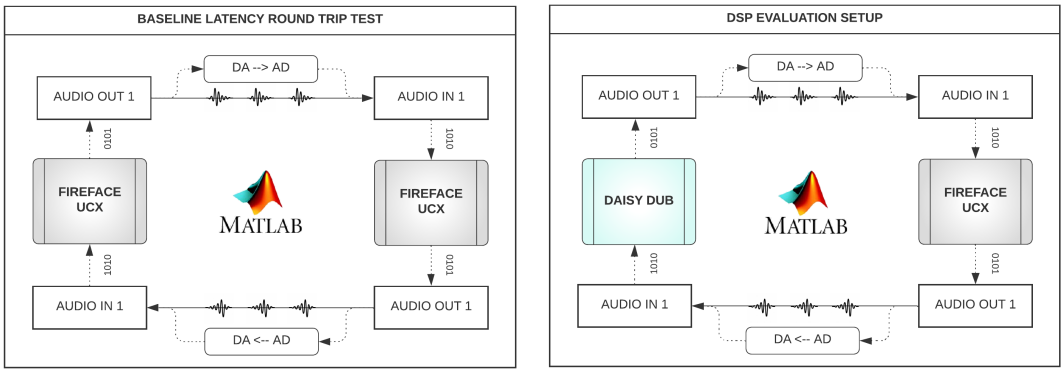


Figure 4.16: Ableton Live Session View test setup to record roundtrip processing from sound card to Daisy Dub to sound card

The recordings were done with input gain and output level fixed at 0 dB with -10 dBV voltage level. -10 dBV is a voltage level commonly used in consumer audio equipment, such as CD players, MP3 players, and other consumer electronics. It is a measure of the voltage level of the signal relative to a reference level of 1 volt. Practically, -10 dBV represents a voltage level of 0.316 Vrms (root mean square) or 0.447 V peak-to-peak. It is commonly used as a reference level for consumer audio equipment, and it is typically the maximum output level of devices such as smartphones, laptops, and portable music players. When using professional audio equipment with a -10 dBV output, it is important to note that the signal level may be lower than that of a +4 dBu output. This may result in a lower signal-to-noise ratio and less headroom in the signal [22, 4].



(a) Daisy Dub signal flow baseline latency test setup diagram

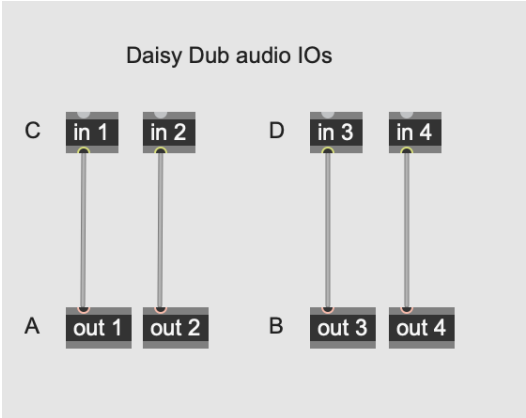
(b) Daisy Dub flow diagram for signal analysis test setup

Figure 4.17: Daisy Dub signal flow baseline latency test and signal analysis test setup diagram

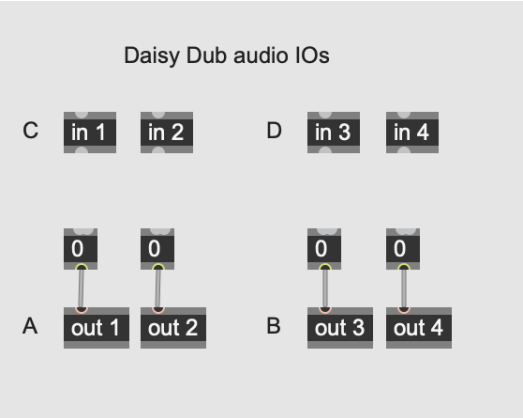
Signal analysis with Audio Precision APx517

Signal analysis for an audio effect involves analyzing the incoming and processed audio signals to evaluate the effect of the audio effect. This includes examining characteristics such as frequency response, harmonic distortion, signal-to-noise ratio, time-domain response, and other parameters

that affect the quality and character of the audio signal. The goal of the signal analysis is to ensure that the audio effect is achieving the intended result and to identify any potential issues or areas for improvement. This information can then be used to make adjustments to the audio effect settings or to improve the underlying algorithms used to process the audio signal. To evaluate that the effect is meeting the desired goals and providing high-quality audio processing, a series of analysis tests were made to ensure the quality of the analogue input and output circuitry and the AD (analogue to digital) and DA (digital to analogue) conversion (see figure 4.20 for test results). To measure the quality of circuitry, a simple pass-thru DSP patch was flashed to Daisy Dub (see figure 4.18a). This patch connects input 1 directly to output 1 with no onboard processing. A similar connection is made from input 2 to output 2. To evaluate only the DA conversion and the analogue output circuitry, a simple gen patch with a bypassed input and only zeroes being sent to the PCB output 1 and 2 was flashed (see figure 4.18b). Starting from scratch and making simple patches has been made fast and easy with a custom Oopsy template tailored for Daisy Dub (see figure 4.19).



(a) Gen patch audio pass-thru connection input 1 to output 1 and input 2 to output 2 (C to A). No further DSP processing is running on the board



(b) Gen patch audio inputs bypassed. Audio outputs constantly output zeroes at sample rate. No further DSP processing is running on the board

Figure 4.18: Two simple gen patchers for audio quality analysis. Pass-thru and bypassed inputs presented to test the analogue audio circuits, the AD-DA conversion, and the output circuits.

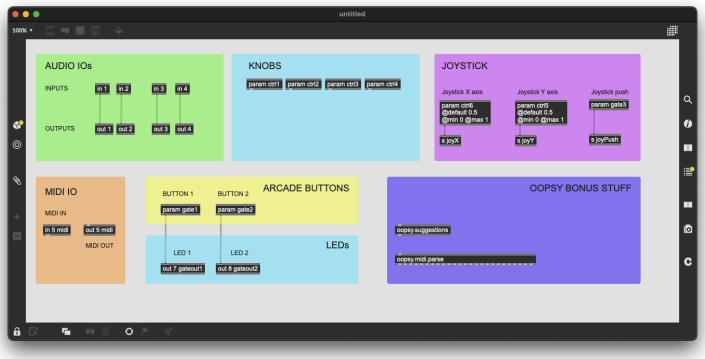
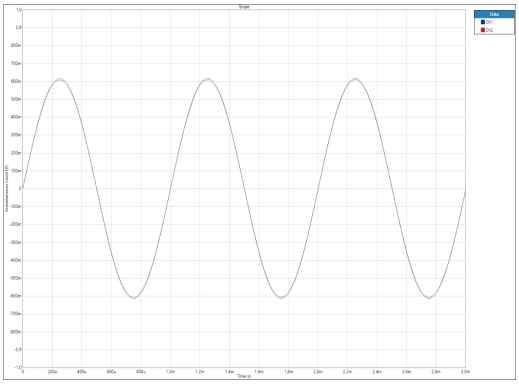
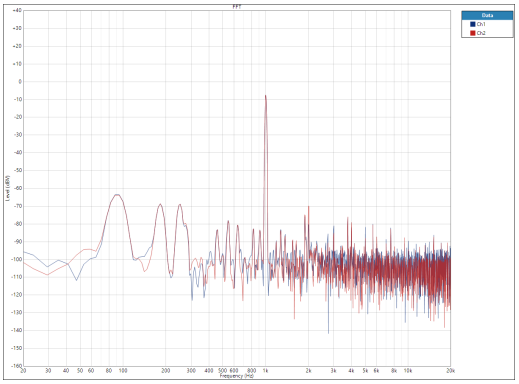


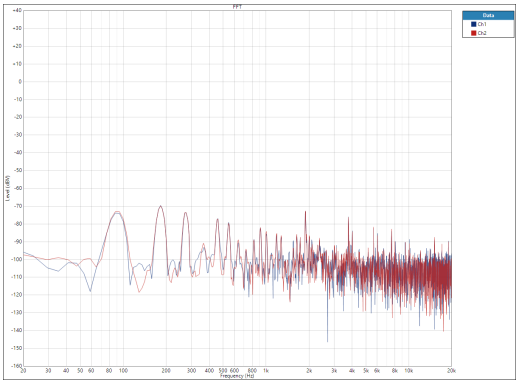
Figure 4.19: Oopsy Daisy Dub clean template



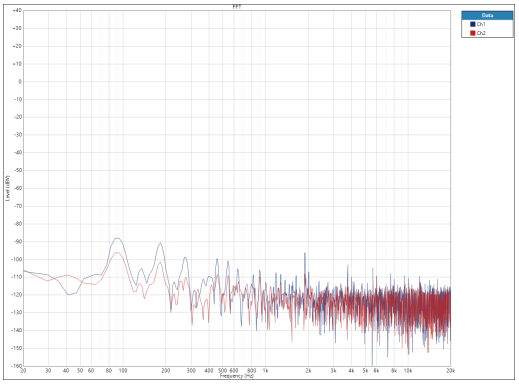
(a) Clean DSP algorithm with PCB input 1 routed directly to PCB output 1 (see figure 4.18a). Similar routing for input and output 2. Scope visualization of 1kHz pure sine wave fed to PCB audio inputs 1 and 2 (Daisy input C) and measuring the resulting output from PCB audio channels 1 and 2 (Daisy output A).



(b) Clean DSP algorithm with PCB input 1 routed directly to PCB output 1 (see figure 4.18a). Similar routing for input and output 2. FFT visualization of 1kHz pure sine wave fed to PCB audio inputs 1 and 2 (Daisy input C) and measuring the resulting output from PCB audio channels 1 and 2 (Daisy output A).



(c) Self-noise measurements with PCB input 1 and 2 (Daisy input C) shorted directly to ground. Clean DSP algorithm with PCB input 1 routed directly to PCB output 1 (see figure 4.18a). Similar routing for input and output 2. FFT measurements of PCB output 1 and 2 (Daisy output A).



(d) Self-noise measurements with PCB input 1 and 2 (Daisy input C) shorted directly to ground. DSP algorithm bypasses input 1 and input 2 (Daisy input C), so the measurement is independent of the input circuits and AD conversion. Output 1 and 2 (Daisy output A) outputs zeroes (see figure 4.18b). FFT measurements of PCB output 1 and 2 (Daisy output A) FFT channel 1 and 2.

Figure 4.20: PCB noise measurements were done with an Audio Precision APx517 acoustic audio analyzer at Sound Hub Denmark. FFT and scope visualisations presented. All DSP code is run at 96.000 Hz sample rate with Oopsy BOOST ON and FASTMATH OFF. Pass-thru patch with no FX (see figure 4.18).

Spectrum analysis

In audio engineering, evaluating the effects of audio processing devices on audio signals is essential. Spectrum and spectrogram analysis can provide valuable insights into the frequency and time-domain characteristics of such signals. In this section, we will explore the use of spectrum and spectrogram analysis for evaluating the effects of the Daisy Dub on three test signals - a sine wave,

a sawtooth wave, and gaussian white noise. All test signals are generated in Max at 96 kHz sample rate (see figure 4.21). The test signals will be played from the sound card, passed through the audio effect device, and re-recorded with the sound card for analysis (see figure 4.17b for diagram setup).

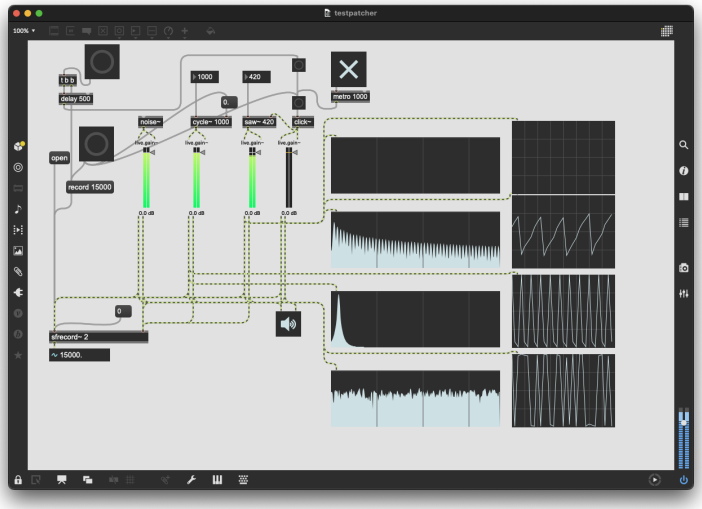
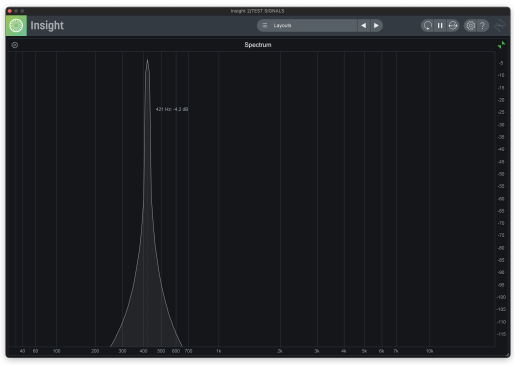
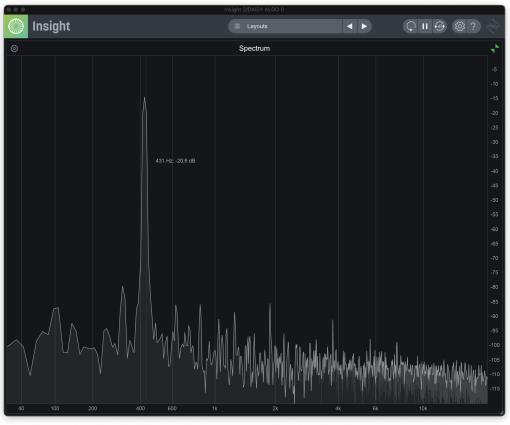


Figure 4.21: Max test audio file generator patch

In figure 4.22 the difference between a 1 kHz sine wave test signal (figure 4.22a) and the recorded signal (figure 4.22b) are seen. In figure 4.23 the same analysis is performed, but on a 1 kHz sawtooth wave. The algorithm tested is fx_router = 0, which bypasses all FX blocks, and only runs the signal through the circular buffer delay. Feedback is set to 0 and delay time is set to minimum, 1 ms (96 samples at 96 kHz sample rate).

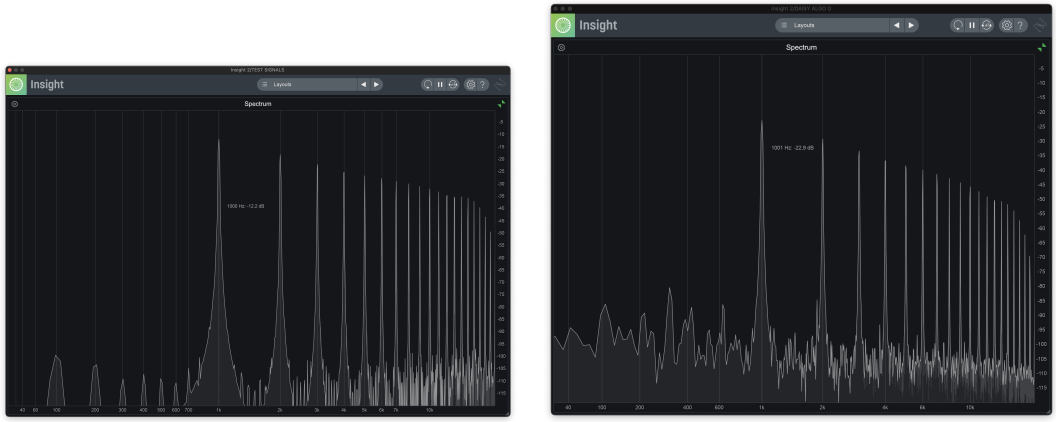


(a) 420 Hz sine wave - test signal played from Fireface UCX sound card



(b) 420 Hz sine wave - recorded test signal from Daisy Dub - FX bypassed (fx_router = 0)

Figure 4.22: Spectrum of 420 Hz sine wave played from Fireface UCX into Diasy Dub and rerecorded. All signals are 96 kHz sample rate.



(a) 1 kHz sawtooth wave - test signal played from Fireface UCX sound card

(b) 1 kHz sawtooth wave - spectrum of recorded test signal from Daisy Dub - FX bypassed (fx_router = 0)

Figure 4.23: Signal analysis spectrum of 1 kHz sawtooth wave played from Fireface UCX into Diasy Dub and rerecorded. All signals are 96 kHz sample rate.

In figure 4.24 a spectrogram plot of a 420 Hz sine wave is shown on the left side, and the processed sine wave with fx_router = 2, which runs the signal through the waveshaper algorithm. Delay time is 1 ms (96 samples) and feedback is 0. The cyclical changes of the high frequencies, from 4 kHz and up on the test signal (left side of figure 4.24) is the result of a slow and minuscule amount of delay time modulation. It was not intended to be on for this test, but it had little effect on showing the effects of the waveshaper algorithm. On the right side of the figure, clear overtones appear resulting from the waveshaper algorithm.

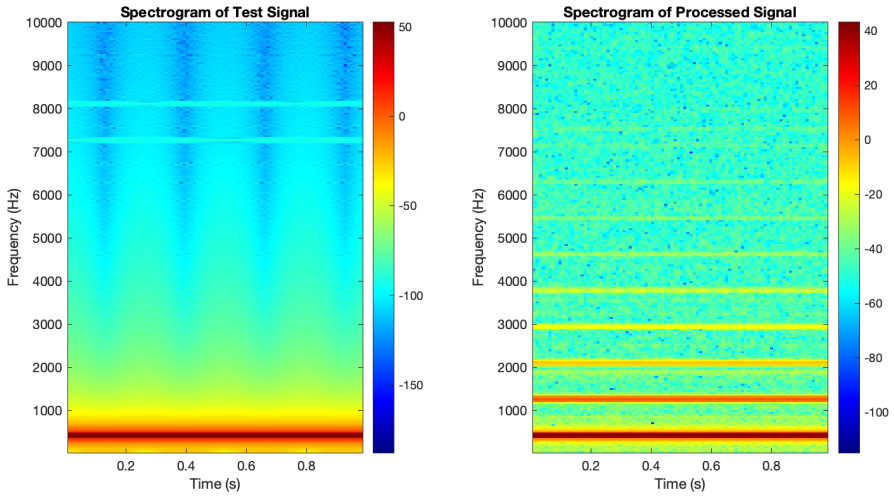


Figure 4.24: Spectrogram comparison fx_router = 2 (waveshaper), sine 420 Hz (parameters: drive mix=0.5, shape=2, ylim [0 10000]

Comparison of spectrograms for two algorithms processing sawtooth waveform at 1 kHz (see figure 4.25 and 4.26). To test the filter algorithm, a test signal was recorded with and without the

filter processing the test signal. In figure 4.25, a spectrogram comparison is shown with the filter algorithm (and all other FX) bypassed. In figure 4.26 the same test is performed but with the SVF filter active, set to lowpass with a cutoff frequency at 20.000 Hz and resonance (Q-factor) at 1.

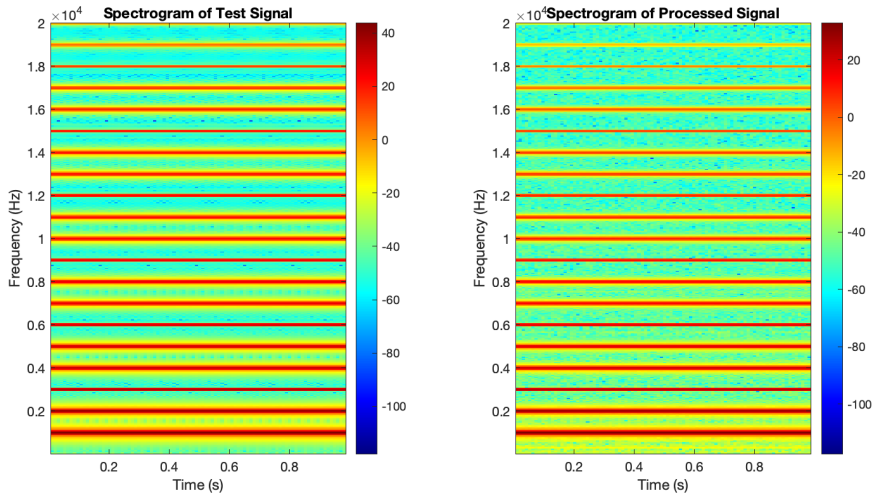


Figure 4.25: Spectrogram comparison fx_router = 0 (all FX bypassed) only delay active with 0 feedback and 96 samples delay time, sawtooth wave 1 kHz

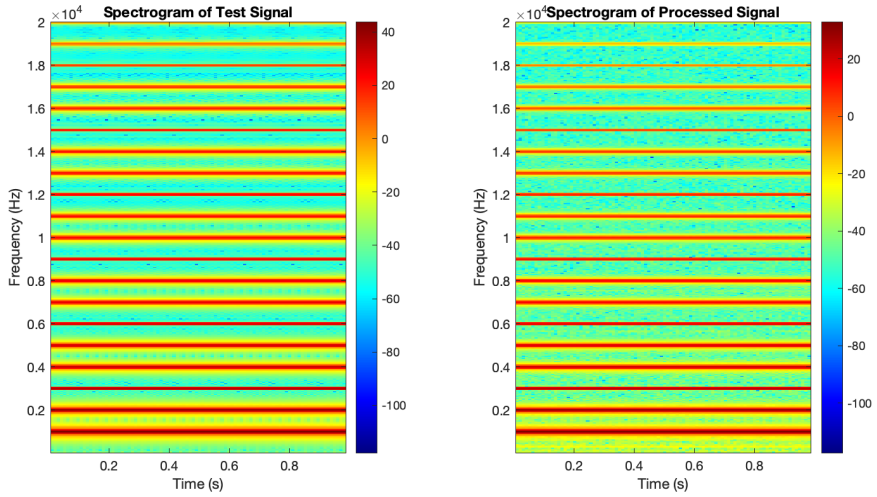


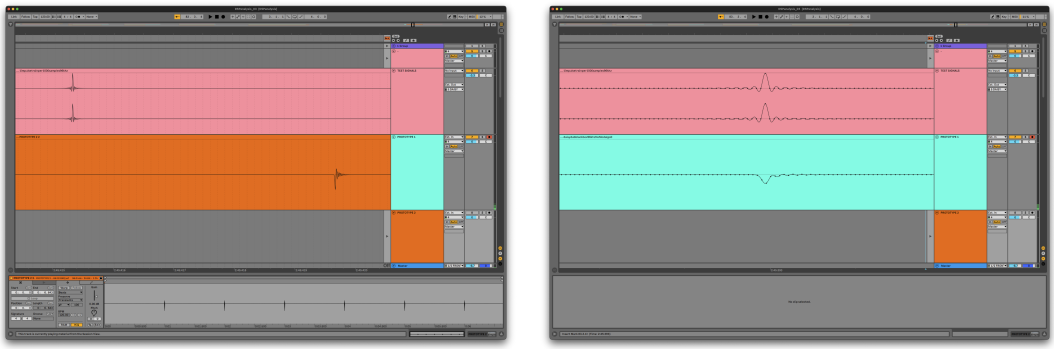
Figure 4.26: Spectrogram comparison fx_router = 1 (lowpass SVF) and delay, cutoff frequency 20.000 Hz, sawtooth wave 1 kHz

See Appendix C for more figures.

Time-domain analysis

In this section, we analyze the time-domain characteristics of signals played from a sound card, passed through the device, and then re-recorded with the sound card. Specifically, we examine the behaviour of an impulse signal, a 1 kHz sine wave, a 1 kHz sawtooth wave, and dub music signals when processed through Daisy Dub. The time-domain analysis of these test signals is an essential part of understanding the behaviour of the audio effect device. By examining the signals' waveform before and after processing through the device, we can determine if the device has introduced any unwanted changes to the signal. This analysis can also help identify any potential issues with the device, such as distortion, phase changes or delay.

In figure 4.27 the latency introduced can be seen (figure 4.27a). It is estimated to $\approx$ 4ms. When shifting the recorded signal $\approx$ 4ms back towards the test signal, a clear phase inversion is seen (figure 4.27b).

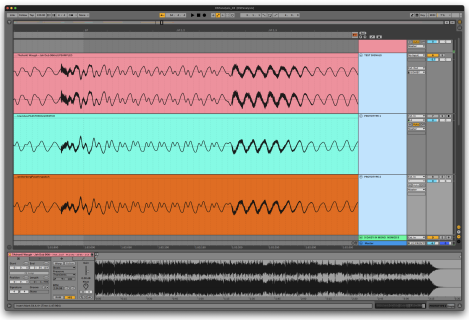


(a) No latency correction

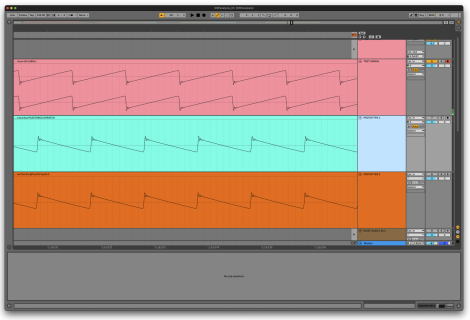
(b) $\approx$ 4ms latency correction, zoomed in

Figure 4.27: Examining latency and phase changes for Daisy Dub patch, impulse signal, 96 kHz sample rate, no delay time modulation, feedback = 0, delay time = 1 ms (96 samples)

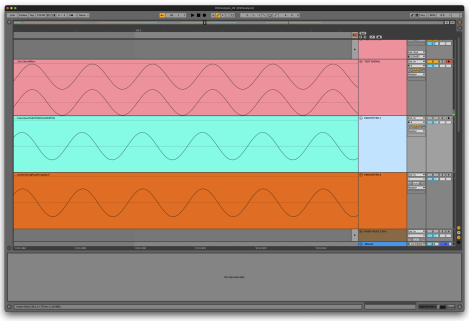
The time-domain analysis tests shown in figure 4.28 are recorded with a gen patch processing audio in pass-thru, as seen in figure 4.18a).



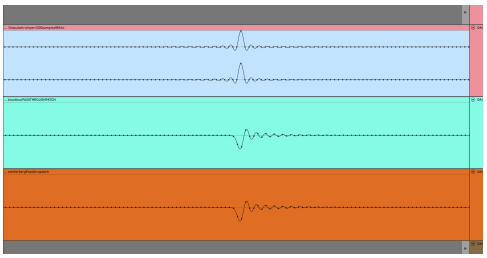
(a) Dub music signal, latency corrected



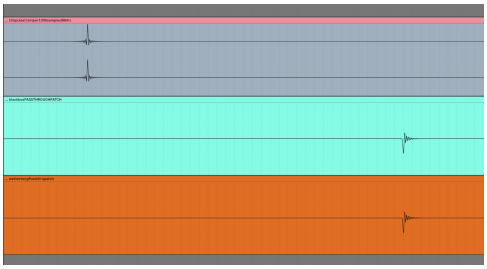
(b) 1 kHz sawtooth wave, latency corrected



(c) 1 kHz sine wave, latency corrected



(d) Impulse, latency corrected



(e) Impulse, NO latency correction

Figure 4.28: Pass-thru recordings of various waveforms passed from Fireface UCX through Daisy Dub and back to Fireface UCX with Ableton Live

Signal analysis with Matlab

To establish a baseline for audio latency testing, various sample rate and buffer size values are tested with *audioLatencyMeasurementExampleApp* from Matlab Audio Toolbox[8]. The test is performed by directly connecting a jack cable between output 1 of the Fireface UCX sound card to input 1 of the sound card (see figure 4.17a and figure 4.29). A test signal is played out through output 1 and is simultaneously recorded via input 1. The *audioLatencyMeasurementExampleApp* then performs a latency test of the desired values set by the user parameters.
The test results are seen in Table 4.1. It was decided to run further latency tests with two values for sample rate, 48.000 and 96.000 Hz. Two values for buffer size were chosen, 256 and 512 samples per

[8] `https://se.mathworks.com/help/audio/ug/measure-audio-latency.html`

```
audioLatencyMeasurementExampleApp(
    'SamplesPerFrame',[128 256 512], ...
    'SampleRate',[48e3 96e3], 'Device', ...
    'Fireface UCX (23574731)', 'IOChannels', [1 1])
```

Table 4.1: Matlab *audioLatencyMeasurementExampleApp* results with pass-thru audio, Fireface UCX output 1 connected to input 1. See Appendix B for Matlab code.

Samples Per Frame	Sample Rate (Hz)	Latency (ms)	Underruns
128	48.000 Hz	NaN	896
128	96.000 Hz	NaN	4.608
256	48.000 Hz	11.062	0
256	96.000 Hz	NaN	256
512	48.000 Hz	29.729	0
512	96.000 Hz	17.552	0

frame. Three of the four values performed consistently without buffer underruns and was selected. See Appendix B for Matlab code.

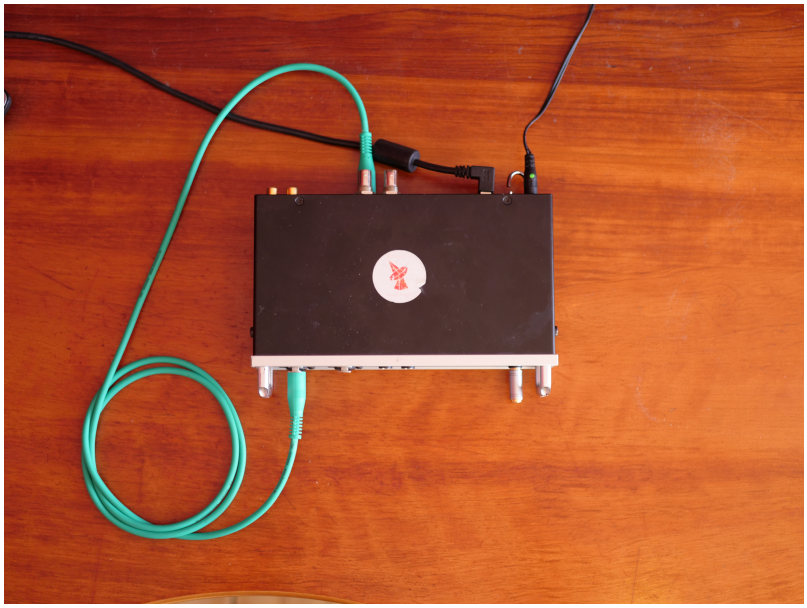


Figure 4.29: Passthru audio latency test with Fireface UCX sound card

The following tests were performed with the same setup in session without interruption. The results are seen on figure 4.30.

Table 4.2: Matlab *audioLatencyMeasurementExampleApp* results with audio passing from Fireface UCX output 1 to Daisy Dub input 1, then from Daisy Dub output 1 to Fireface UCX to input 1. See Appendix B for Matlab code.

	48 kHz 256 samples	**48 kHz 512 samples**	**96 kHz 512 samples**
Sound card pass-thru	11,1	29,7	17,6
FX bypassed (dry/wet = 0)	17,9	31,2	19,0
Dry/wet = 100, FX bypassed	19,1	32,5	20,3
Filter low pass, cut-off 50 %	19,4	32,7	20,6
Filter low pass, cut-off 20 kHz	19,2	32,5	20,3
Waveshaper	19,2	32,5	20,3
Phaser	19,2	32,5	20,3
Filter -> Waveshaper -> Phaser	19,2	32,5	20,4

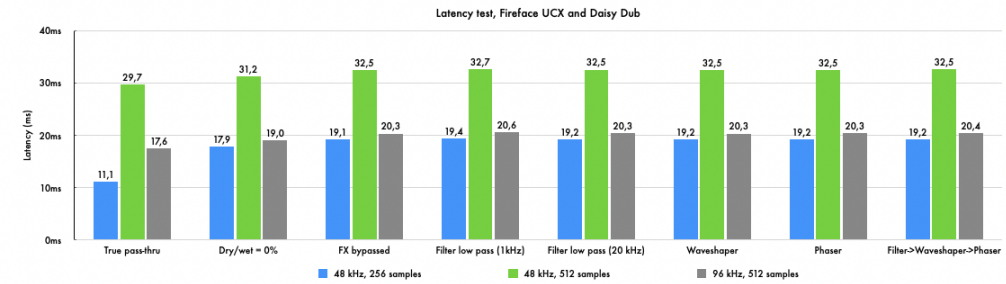


Figure 4.30: Matlab *audioLatencyMeasurementExampleApp* test results from eight different signal processing scenarios. Tests are performed with a Fireface UCX audio interface and Daisy Dub. See Appendix B for Matlab code.

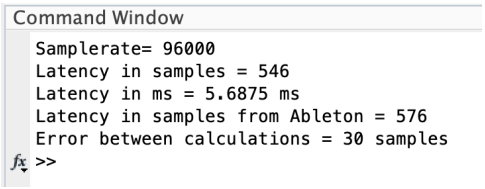
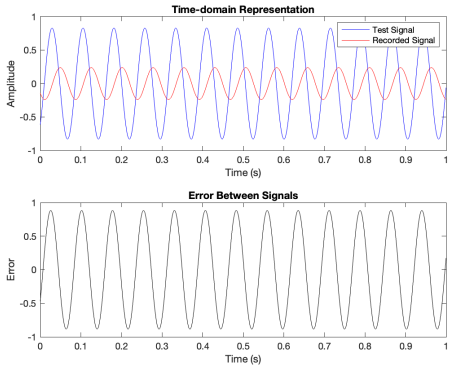
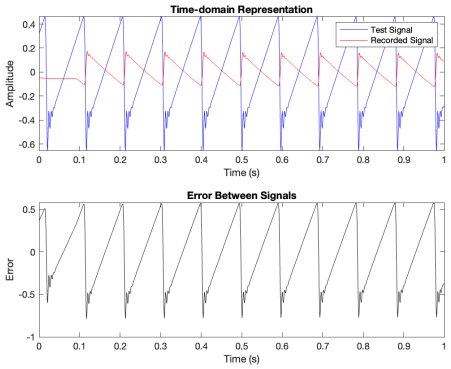


Figure 4.31: Signal analysis with an impulse audio signal pass-thru between Fireface UCX sound card and Daisy Dub with fx_router = 0 (all FX bypassed). This test determines the latency for the following signal comparisons. See Appendix B for Matlab code.



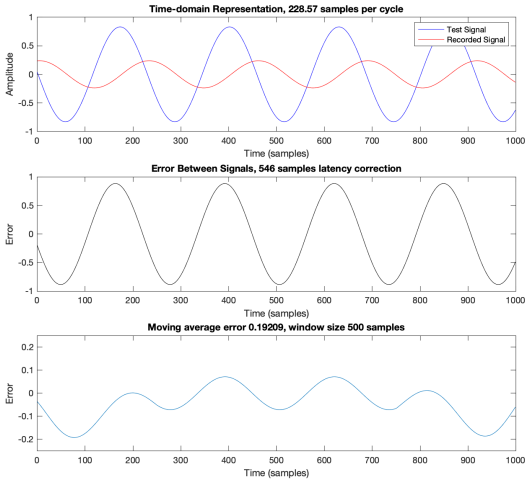
(a) Time domain analysis fx_router = 0, signal is a pure sine wave at 420 Hz frequency



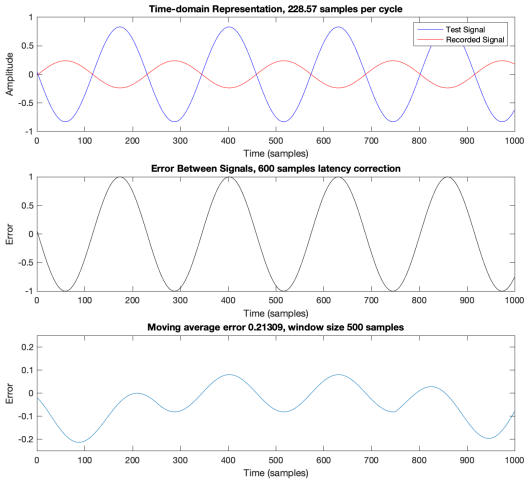
(b) Time domain analysis fx_router = 0, signal is a sawtooth wave at 1kHz frequency

Figure 4.32: Initial time domain analysis of test signals; sine wave 420 Hz and sawtooth wave at 1 kHz frequency. The recorded signal is latency corrected to match the test signal with a value of 546 samples (see figure 4.31 for calculation and Appendix B for Matlab code).

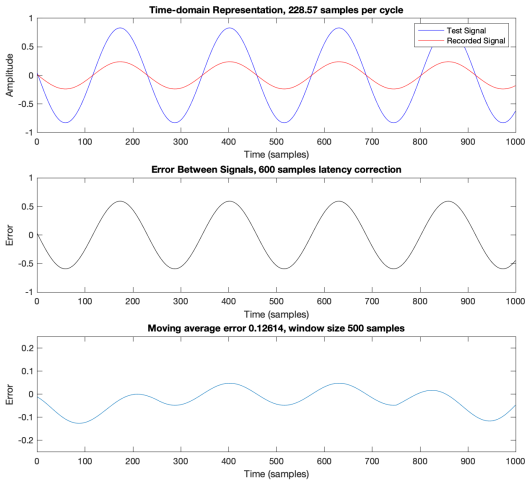
An initial inspection of time domain plots revealed a discrepancy between the calculated latency and how the waveforms matched up. Furthermore, the signal appears to be phase inverted. Initial inspections are seen in figure 4.32. Further inspection, changing the latency compensation and inverting the phase significantly lowered the error curve. A time domain plot with an error curve and moving average calculation of error are seen in figure 4.33 and 4.34.



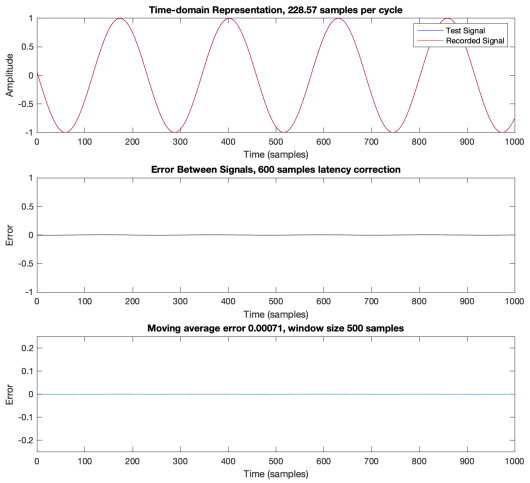
(a) Sine wave 420 Hz (fx_router = 0) 546 samples latency compensation, NOT normalized, NOT phase inverted.



(b) Sine wave 420 Hz (fx_router = 0) 600 samples latency compensation, NOT normalized, NOT phase inverted.



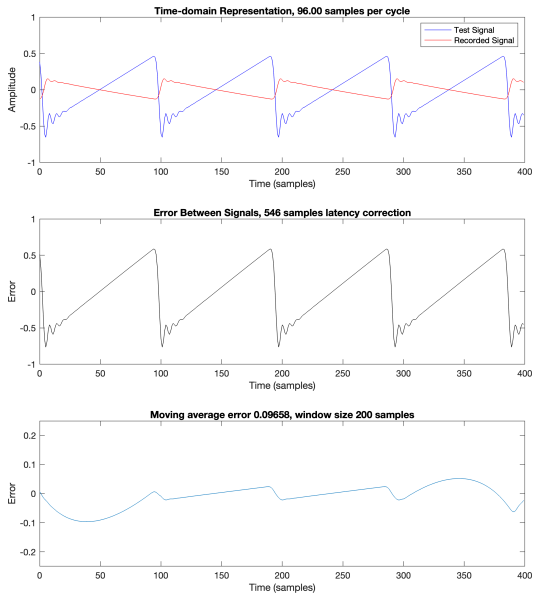
(c) Sine wave 420 Hz (fx_router = 0) 600 samples latency compensation, NOT normalized, phase inverted.



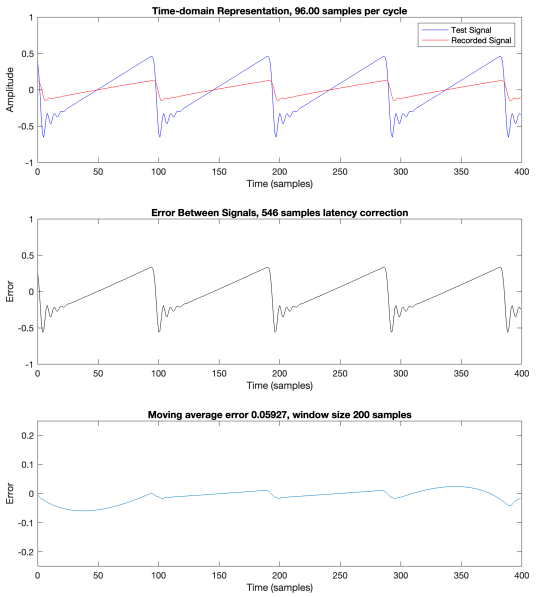
(d) Sine wave 420 Hz (fx_router = 0) 600 samples latency compensation, normalized, phase inverted.

Figure 4.33: Time domain analysis with latency compensation, normalization and phase inversion for 420 Hz sine wave. See Appendix B for Matlab code.

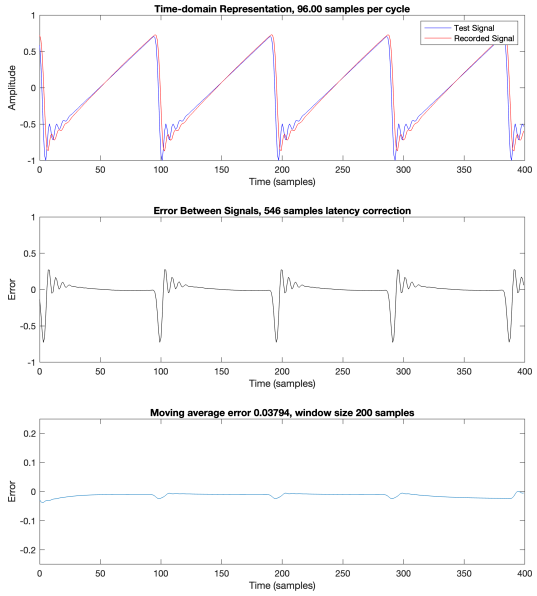
The uneven curve of the sawtooth waveform results from Gibbs' phenomenon for waveforms near points of discontinuity. The sawtooth waveform has sharp, abrupt transitions at the end of each cycle, resulting in a jagged appearance when viewed in the time domain. This sharp transition produces high-frequency harmonics, absent in a sine wave, resulting in the so-called Gibbs phenomenon. The high-frequency harmonics cause ringing or overshoot near the discontinuity, making the waveform appear jagged.



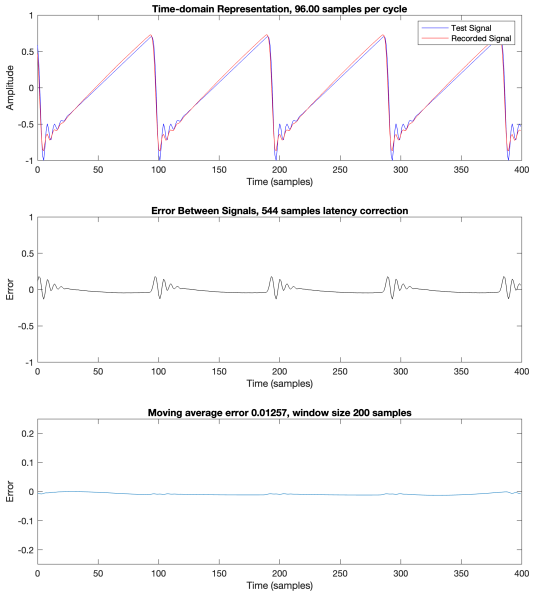
(a) Sawtooth wave 1 kHz (fx_router = 0) 546 samples latency compensation, NOT normalized, NOT phase inverted.



(b) Sawtooth wave 1 kHz (fx_router = 0) 546 samples latency compensation, NOT normalized, phase inverted.



(c) Sawtooth wave 1 kHz (fx_router = 0) 546 samples latency compensation, normalized, phase inverted.



(d) Sawtooth wave 1 kHz (fx_router = 0) 544 samples latency compensation, normalized, phase inverted.

Figure 4.34: Time domain analysis with latency compensation, normalization and phase inversion for 1 kHz sawtooth wave. See Appendix B for Matlab code.

After thorough testing of the analog circuits, it was decided to invert the phase on all outputs of the gen patch. This remedied the phase inversion caused by the DSE conversion circuit (figure 4.5). A new trial of an impulse signal was recorded and the resulting corrected phase can be seen on

figure 4.35.

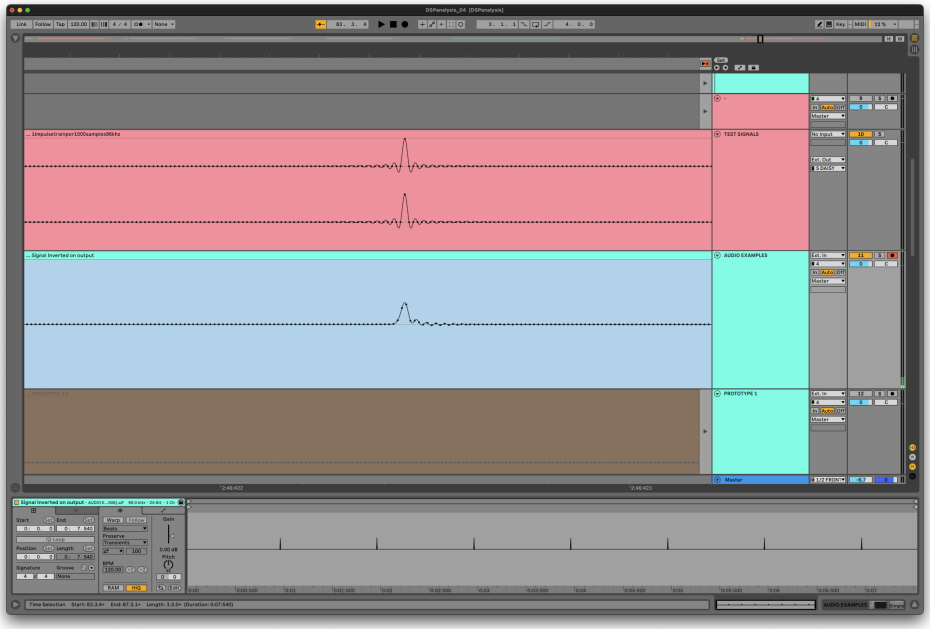


Figure 4.35: Latency test Ableton Live, Daisy Dub fx_router = 10, phase inverted on all Daisy Dub outputs

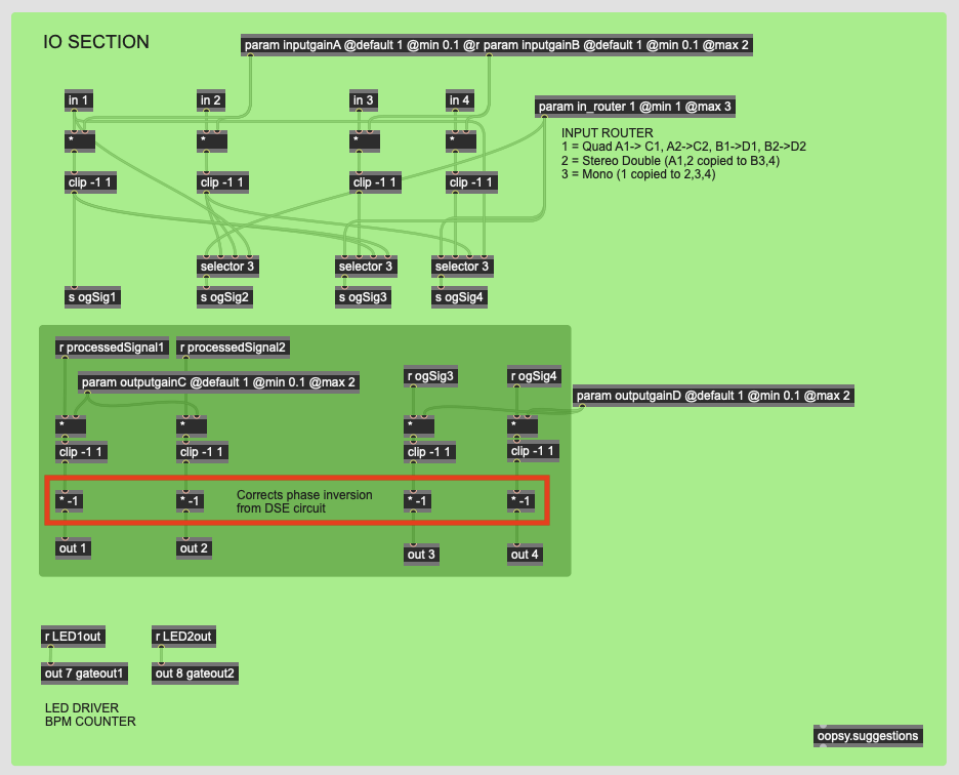


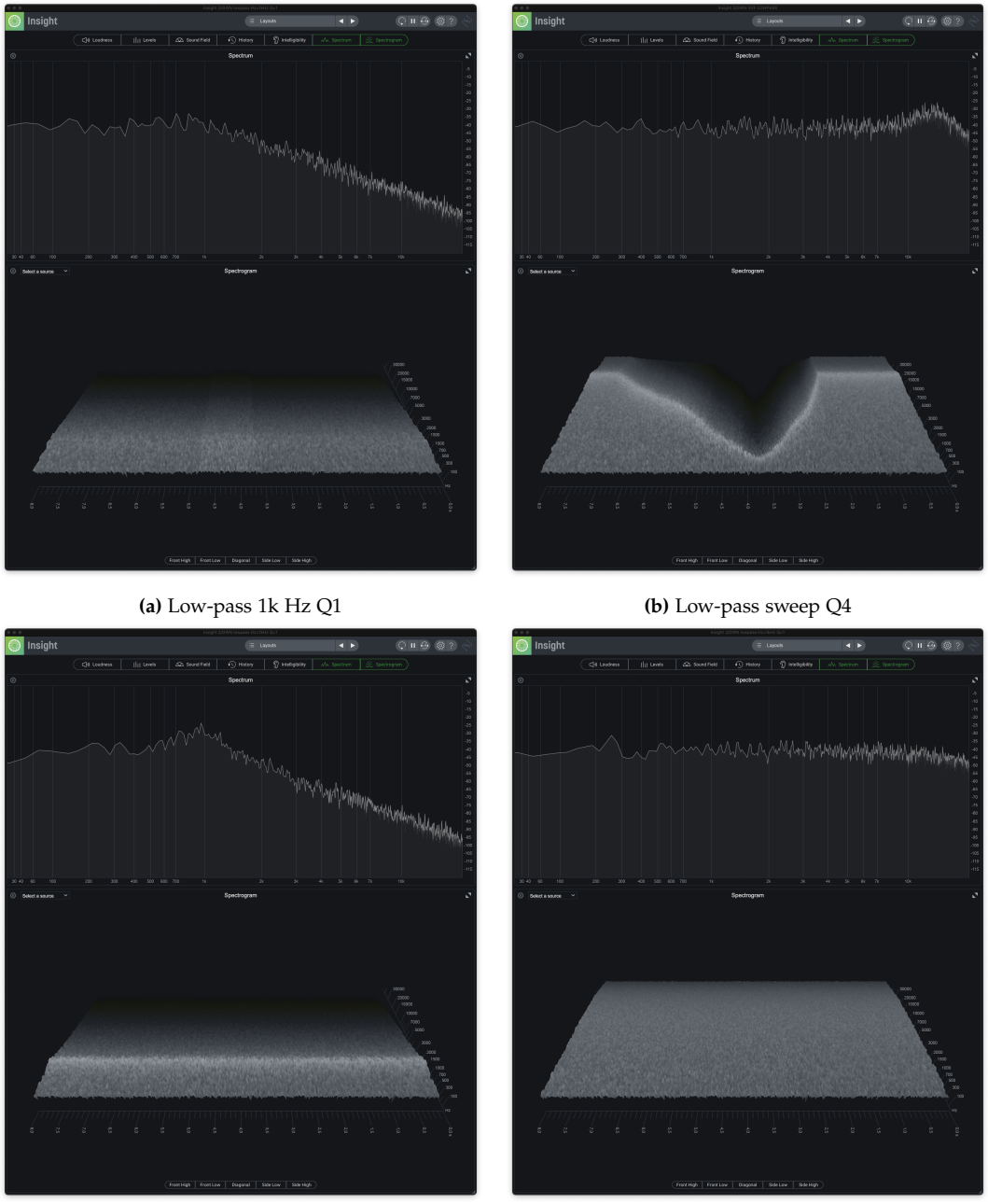
Figure 4.36: Daisy Dub main gen patch with updated output section, phase inversion ($* -1$) on all Daisy Dub outputs (dark green area, red box)

See Appendix B for Matlab code and more figures.

4.3.8 Filter DSP evaluation

In the field of audio signal processing, the use of filters is crucial for achieving desired audio effects, removing unwanted noise, and emphasizing specific frequency bands. The effectiveness of a filter is dependent on various factors, such as its frequency response, resonance (Q-factor), and type. In order to evaluate the performance of different filters under varying conditions, a common practice is to apply a white Gaussian noise signal and measure the filter's output in the frequency domain. This evaluation method allows for quantitative analysis of a filter's ability to shape the frequency content of a signal. In this section, we present the results of a filter evaluation using a white Gaussian noise test signal and varying filter frequency, resonance, and type.

Low-pass filter evaluation



(a) Low-pass 1k Hz Q1

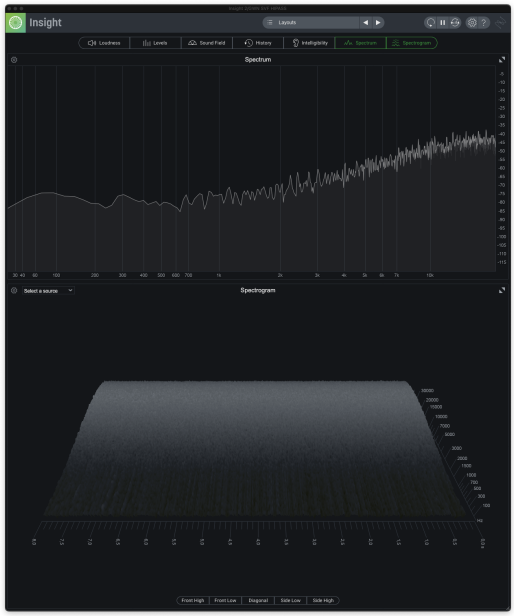
(b) Low-pass sweep Q4

(c) Low-pass 1 kHz Q4

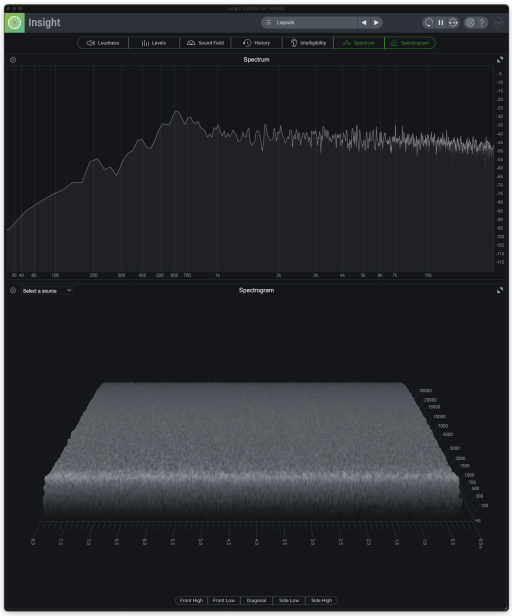
(d) Low-pass 20 kHz Q1

Figure 4.37: Low-pass filter tests, white Gaussian noise

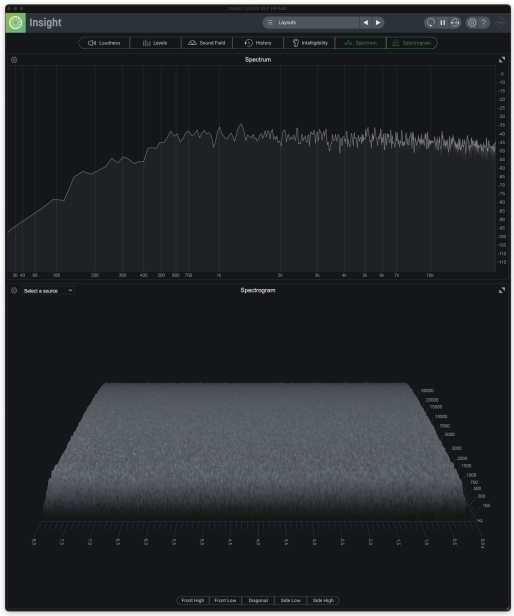
High-pass filter evaluation



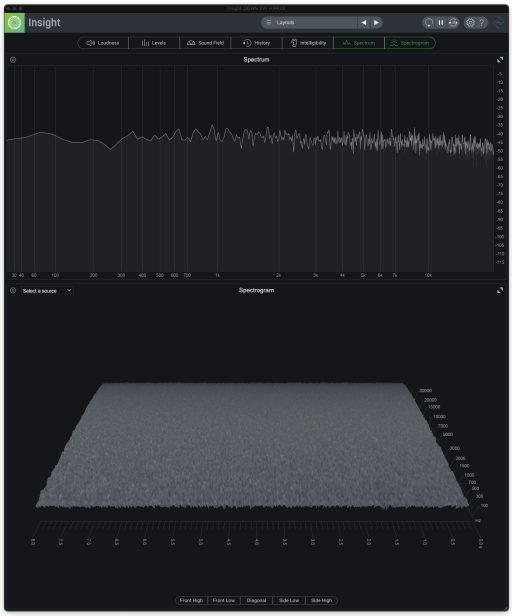
(a) High-pass 20 kHz Q1



(b) High-pass 1 kHz Q4



(c) High-pass 1 kHz Q1



(d) High-pass 20 Hz Q1

Figure 4.38: High-pass filter tests, white Gaussian noise

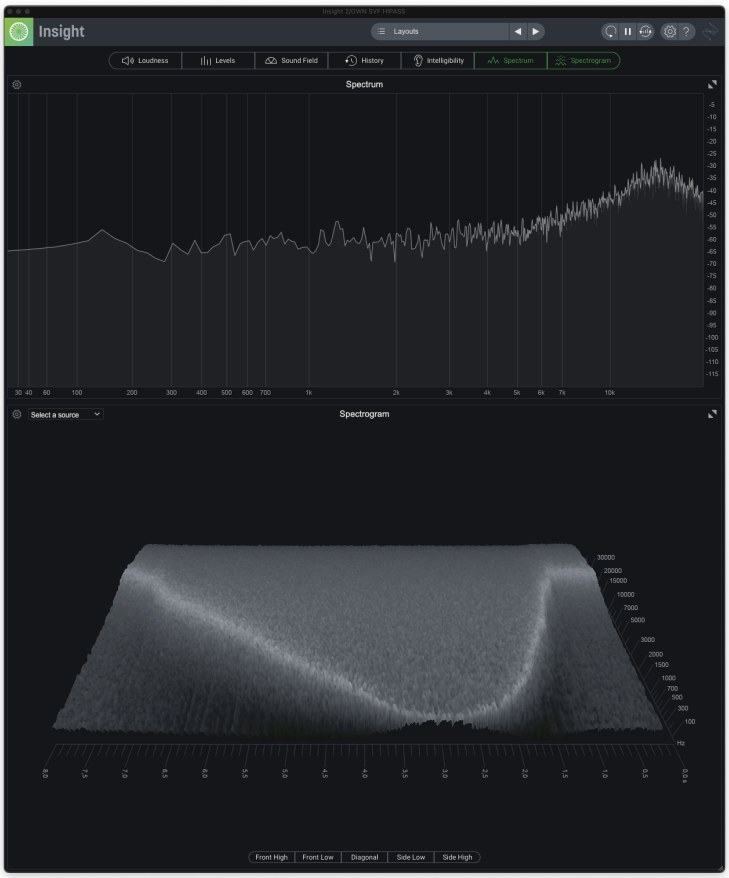
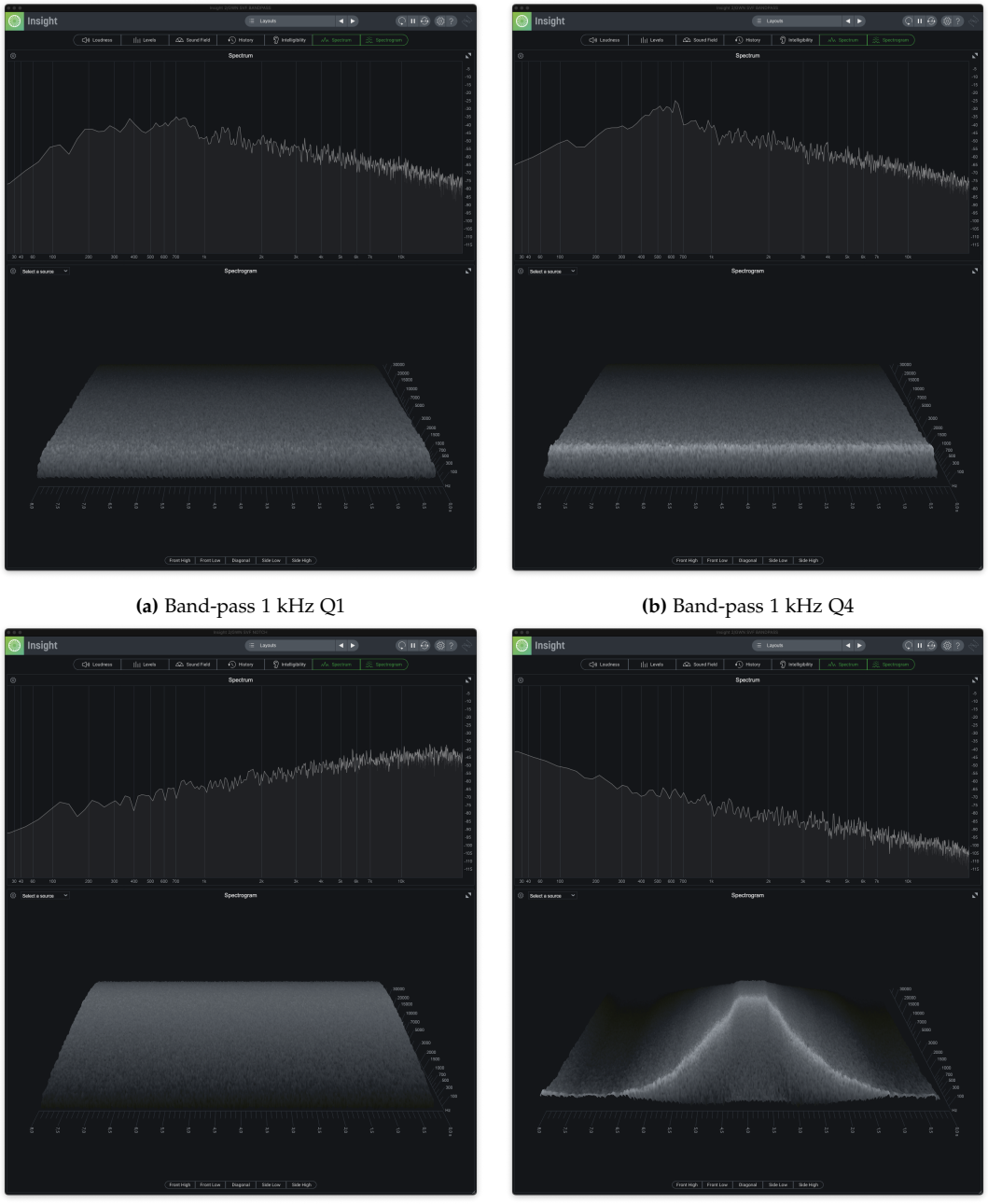


Figure 4.39: White Gaussian noise, high-pass filter sweep, Q4

Band-pass filter evaluation



(a) Band-pass 1 kHz Q1

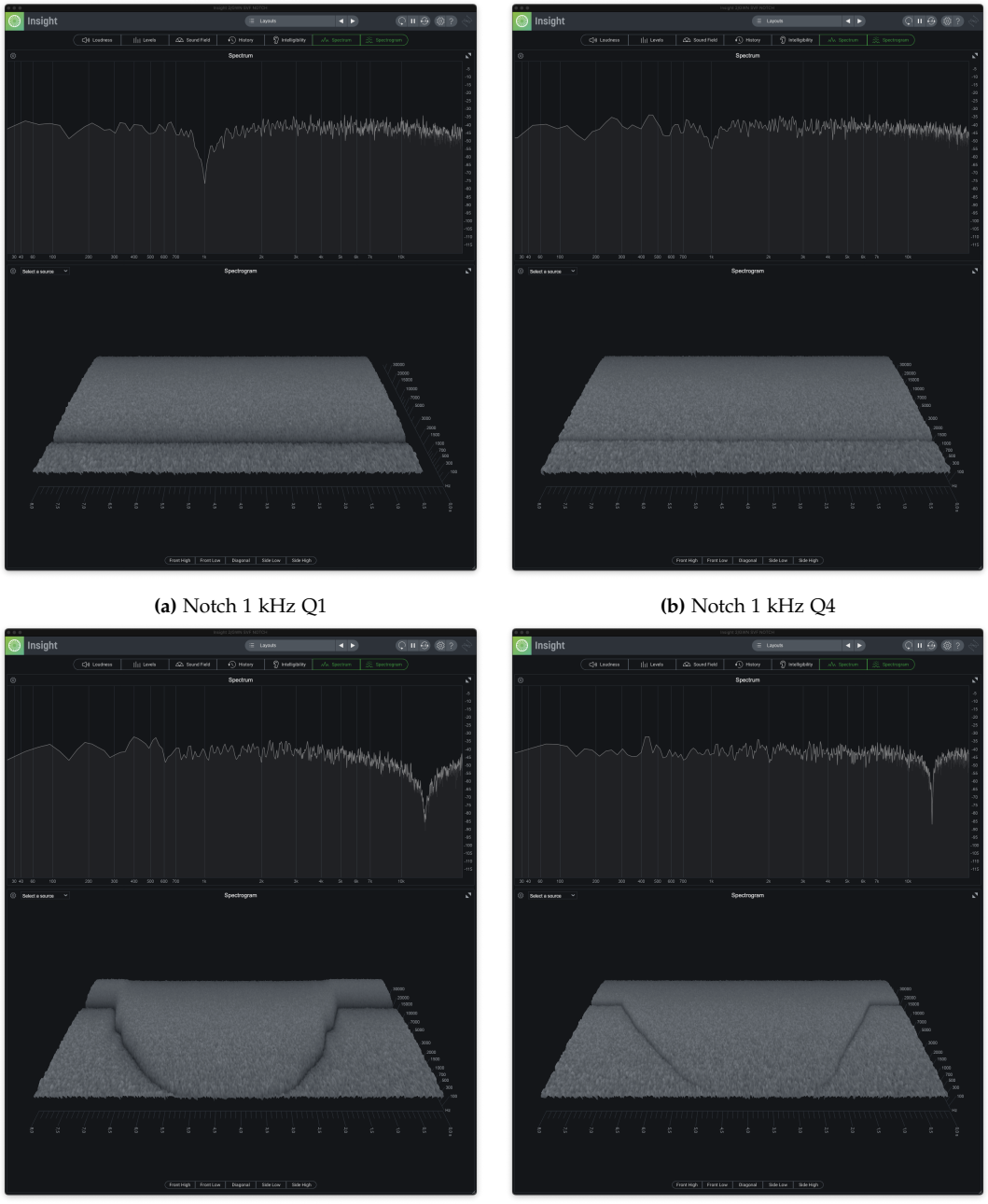
(b) Band-pass 1 kHz Q4

(c) Band-pass 20 kHz Q1

(d) Band-pass sweep Q4

Figure 4.40: Band-pass filter tests, white Gaussian noise

Notch filter evaluation



(a) Notch 1 kHz Q1

(b) Notch 1 kHz Q4

(c) Notch sweep Q1

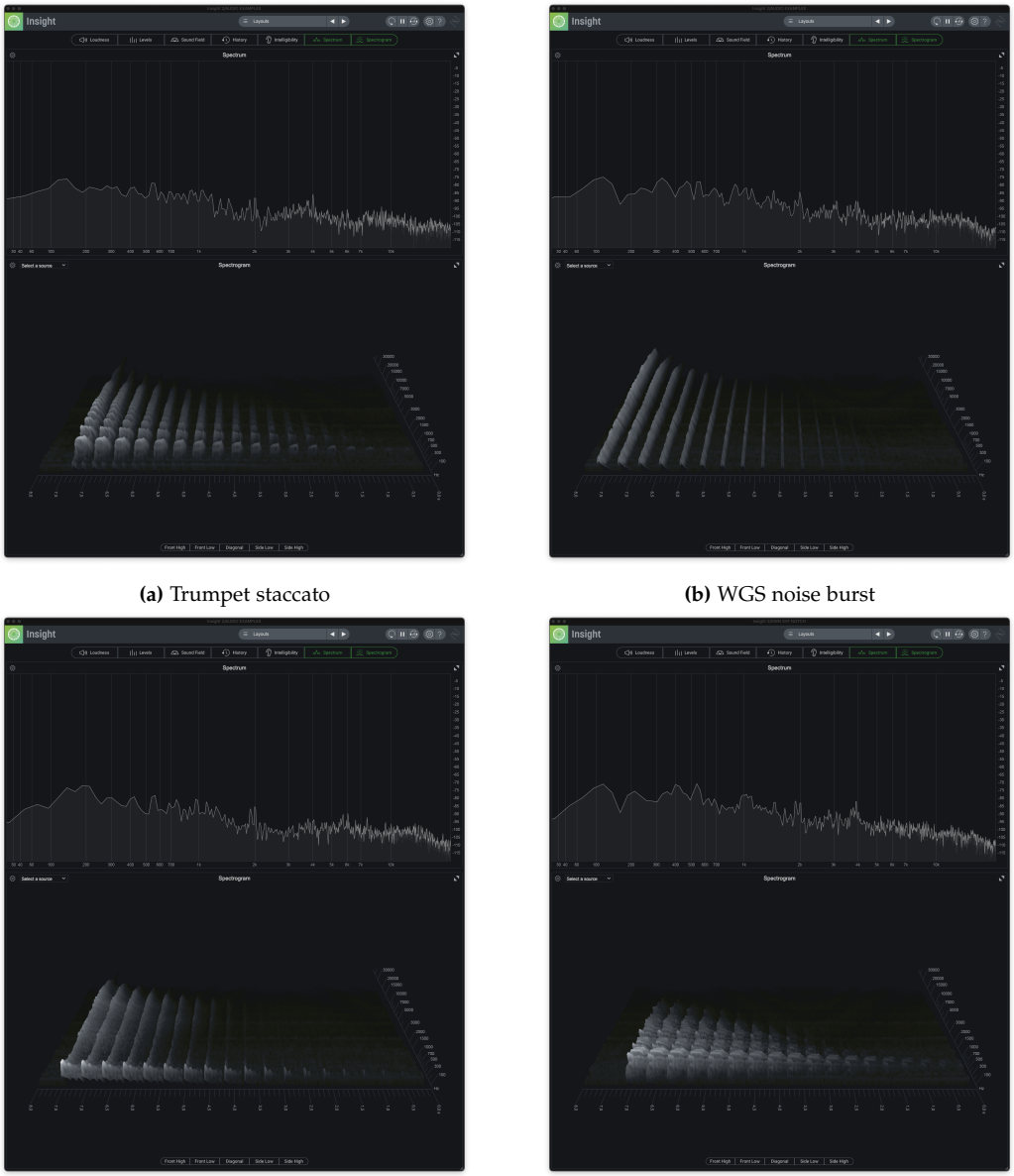
(d) Notch sweep Q4

Figure 4.41: Notch filter tests, white Gaussian noise

4.3.9 Daisy Dub complete DSP evaluation

In this section, we analyze the performance of the complete DSP chain of the Daisy Dub device. This includes delay, modulation, noise, filtering, waveshaping, and phasing. We evaluate the chain's ability to process various test signals, including guitar skank, snare drum, white Gaussian noise burst, and trumpet. The analysis is performed using both spectrogram and spectrum methods with Ableton Live and iZotope Insight. In the first part of the evaluation, we examine the chain's performance with feedback levels below 1 (see figure 4.42). We measure the chain's ability to preserve the quality of the original signal while applying various effects and filtering techniques. In the second part of the evaluation, we increase the feedback levels above 1 and adjust parameters such as delay time, filter frequency, feedback, and filter resonance. We examine the impact of these changes on the processed signal's quality and evaluate the chain's ability to maintain stable, but aggressive operation at high feedback levels, short delay times, and high filter resonance (see figure 4.43). With a feedback level above 1, the device will start self-oscillating. Increasing filter resonance will increase this by a magnitude. Using the filter frequency can sculpt this aggressive feedback while changing the 'ringing' tone of the filter around the cutoff point. Audio examples are available on GitLab[9].

[9] `https://gitlab.com/northernstructuresaudio/daisy-dub/-/tree/main/audio-examples`



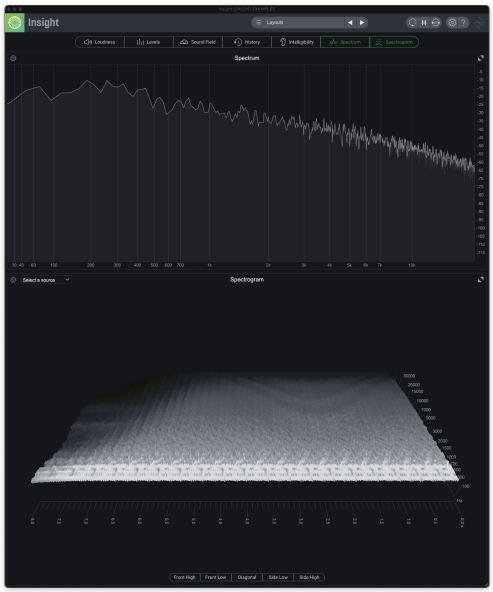
(a) Trumpet staccato

(b) WGS noise burst

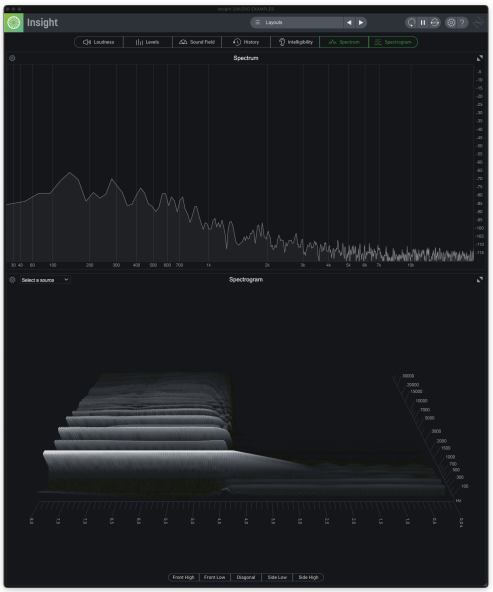
(c) Snare drum hit

(d) Guitar skank

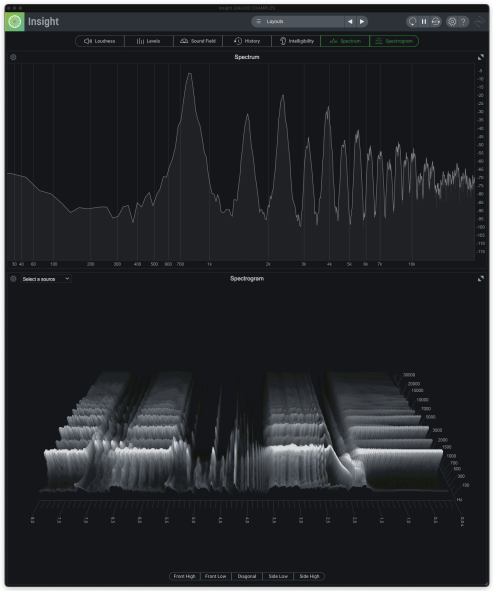
Figure 4.42: Daisy Dub complete DSP test with various signals, fx_router = 10, feedback < 1, low-pass filter



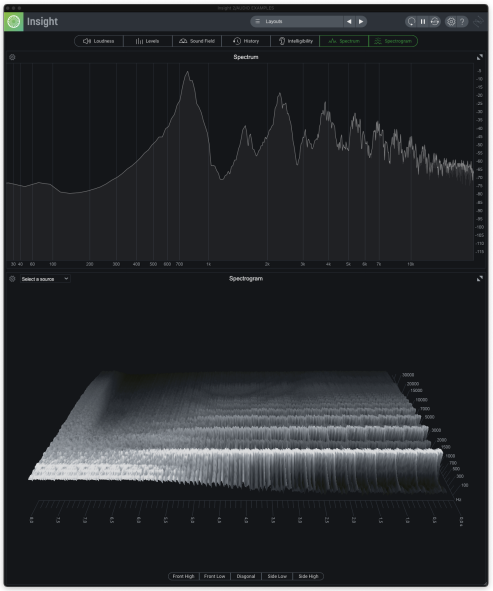
(a) Feedback > 1, sweeping filter frequency



(b) Self-oscillation dying out



(c) Feedback > 1, high resonance, rapidly adjusting delay time



(d) Feedback > 1, increasing filter resonance

Figure 4.43: Daisy Dub complete DSP test with various signals, fx_router = 10, feedback > 1, low-pass filter, adjusting delay time, feedback, filter frequency and filter resonance

4.4 Mechanical design

The mechanical design of a portable, modular and updateable real-time audio effect for music producers and performers significantly impacts various aspects of the product, including ergonomics, performance, and repairability. In this section, we will focus on these key areas and discuss the mechanical design considerations that are relevant to each. One key aspect of the mechanical design of this product is the ergonomics of the device. As a performance-oriented tool, the device should be designed to be comfortable and easy to use for music producers and performers. This involves considerations of the size and shape of the device, the layout and arrangement of the various controls and interfaces, and the use of materials and finishes that are comfortable and easy to grip. It is also essential to consider the overall balance and weight of the device. This is, however, out of the scope of this project, as these considerations become relevant only after ensuring that the product's functionality is established. Another critical aspect of the mechanical design is the layout and arrangement of the various controls and interfaces. For this product, this includes four knobs, one encoder, an OLED display, a thumb joystick, two arcade buttons, and two LEDs, as well as a USB Type-C connection and four audio inputs and four audio outputs as double stereo pairs. It is important to consider the ergonomics of these controls and interfaces, as well as their accessibility and ease of use, in order to ensure that the product is intuitive and user-friendly. Performance is another key aspect of the device. As a real-time audio effect, the device should be designed to perform to the required standards of the target audience. This involves considerations of the design of the audio processing and signal processing systems, the selection of suitable hardware and software, and the overall architecture of the system. It is also important to consider the audio input and output interfaces and the means of connecting the device to external audio sources and destinations. Ensuring that the Daisy Dub lives up to these standards would require a lot of user testing with this specific focus, and while it is important for an overall user experience, it is outside the scope of this project to perform such testing. Instead, the mechanical design of the Daisy Dub will draw upon existing hardware with similar use cases. This should suffice unless any issues arise due to this design during the evaluation. In addition to ergonomics and performance, the mechanical design of the product should also consider repairability. This involves considerations about the system's design for ease of diagnosis and repair, the use of standard and readily available components, and the provision of suitable documentation and support for maintenance and repair. By designing the product with repairability in mind, it is possible to reduce the costs and downtime associated with maintenance and repair, and to improve the overall reliability of the product.

Overall, mechanical design requires careful consideration of a wide range of factors, including ergonomics, performance, and repairability. Considering these factors, it is possible to develop a product that is comfortable and easy to use, performs to the desired level, and is reliable and easy to maintain.

The design of the case for the Daisy Dub is based on the original case from the initial paper [13]. Although after the testing done in section 2.7.1, some features were added, so they will need to be incorporated into the design as well. Additionally, the backside of the Daisy Dub will have to accommodate the additional in and outputs. As the Daisy Dub is intended to be a DIY project, the case should be simple to assemble, and also simple to swap out materials for the case if that is desired. With this in mind, it was decided that the sides of the case should be all straight, as this would be a lot easier for most people to source, rather than having to curve material in any way to build the case. This decision means that the corner connecting pieces have to be able to accommodate all of the sides, as well as provide a place to secure the PCB.

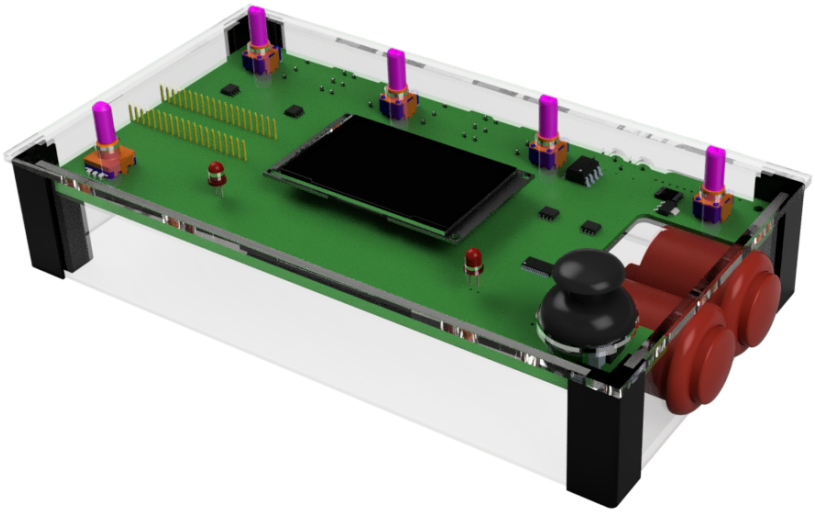


Figure 4.44: 3D model of the Daisy Dub rendered with Fusion 360

4.4.1 Fusion 360

In order to create a model of the casing for the Daisy Dub Fusion 360 was used. This allowed for iteration of the design, specifically in conjunction with the development of the PCB, as it is possible to import a step file of the PCB to make sure that everything fits together. The final casing for the Daisy Dub consists of six different plates and four identical corner pieces. A 3D model of the full case including the PCB can be seen in figure 4.44.

4.4.2 Case plates

The sketches on the individual plates are exported out of Fusion 360, and used for laser cutting material for the six plates. In order to test the fit of all the plates they were first cut out of 3mm black veneer, as this material was easily available and cheap to test on. By doing this testing it was possible to adjust the height of the front, back, left, and right plates so that there was not a gap between them and the top plate when assembled. The cuts for all of the parts which stick out of the box were also given some offset to ensure that they would be able to fit without any issues. Text was added to the back plate to indicate what the different connections are for, and a USB logo was added on the left side to signify that data transfer is done through that port. The back of the Daisy Dub with the text can be seen in figure 4.45.

4.4.3 Side pieces

For the corner pieces, Fusion 360 allows for direct transfer to Ultimaker Cura, which was used to 3D print the pieces. It was decided that 3D printing these side pieces would be the best option, as this would allow for custom pieces that fit the exact dimensions, as well as rapid prototyping. 3D printing has also become a lot more accessible in the last few years, and as such, it is expected that the use of 3D printing will not exclude people from creating the Daisy Dub as a DIY project. The inspiration for the design of the corner piece is taken from corner extrusion profiles, which will typically be made out of metal. A 3D model of the corner piece can be seen in figure 4.46. The corner piece is designed in such a way that the top and bottom are screwed into them, and then the rest of

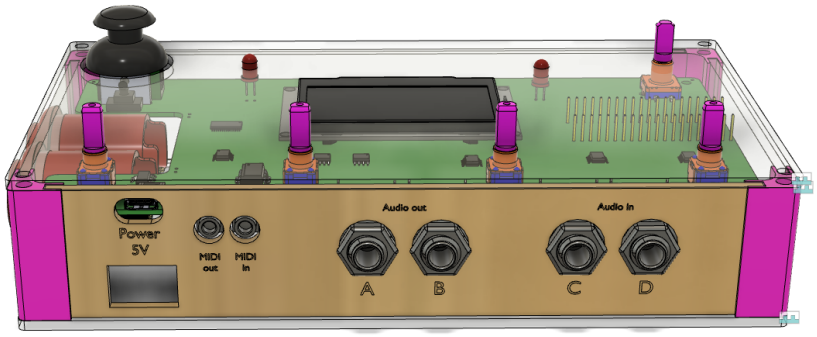


Figure 4.45: 3D model of the back of the Daisy Dub rendered with Fusion 360

the sides are held in place between them. In order to screw into a 3D print brass insert are heated up and placed into the plastic of the print. This allows for the top and bottom plate and the PCB to be secured by screws.

4.5 Acceptance testing

In order to assure that the demands set in section 4.1 are met, some checks will be done on the final implementation of the new prototype. With this final implementation, all of the functional demands are currently easily verifiable and have mostly been discussed in the previous sections. The exception to this is the ability to save and recall presets. This feature was not implemented in this prototype due to time constraints but is highly prioritized in the further development of the Daisy Dub. Most of the quality demands is also already answered. As this acceptance test is done before evaluating the prototype, it is not yet possible to say what SUS score it achieves. It is however known that the form factor of the prototype is currently within the limits set by the demands, and with the addition of rubber feet to the design the anti-slip has also been achieved. A way to flash new code without direct access to the Daisy Seed has not yet been implemented due to time constraints, but during the development, it was discovered that it was possible to put the Daisy Seed into bootloader mode through other inputs than the two buttons directly on the seed. The environmental demands have been deemed fulfilled but are still subject to change if any issues should arise during evaluation. It is believed that the development facilitates the fulfillment of these demands, but as more users get to use the Daisy Dub, new information might arise that changes their status of them.

4.6 Usability testing of Daisy Dub

Usability testing is an important step in the development process for any new product. It can help identify any issues or challenges that users may face when using the product and can provide valuable insights into how to improve the user experience. There are several different approaches that can be taken when conducting usability testing of a modular and updateable real-time audio effect. One common approach is to conduct usability testing with a small group of representative users, such as music producers or DJs, who represent the product's target

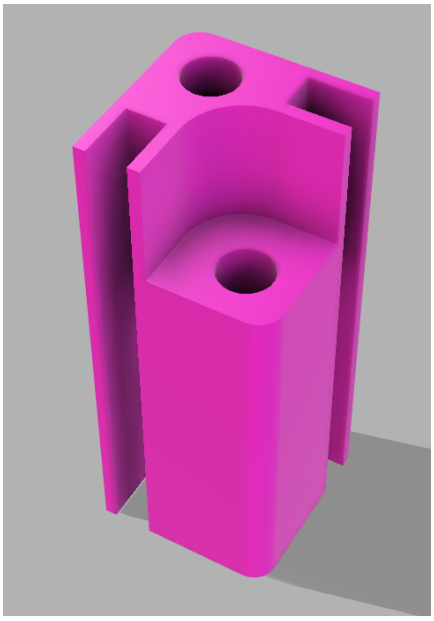


Figure 4.46: 3D model of the corner pieces rendered with Fusion 360

Category	Demand	Status
Functional	Power through USB Type-C	Fulfilled
	Data through USB Type-C	Not fulfilled
	Quadrophonic audio IO capabilities	Fulfilled
	Usable without external audio input	Fulfilled
	MIDI IO functionality	Implemented (untested)
	Preset store and recall	Unfulfilled
	Expose analog inputs on the PCB	Fulfilled
	Delay time modifiers	Fulfilled
Quality	SUS score $\geq$ 79.5	Undecided
	Form factor	Fulfilled
	Anti-slip	Fulfilled
	Custom bootloader from the menu or button combo	In progress
Environmental	Easily replaceable parts	Fulfilled
	Non-specific mechanical design	Fulfilled
	Modular and user-changeable software	Fulfilled

Table 4.3: Demand specification status

audience. This can help identify any issues or challenges that users may face when using the product and gather feedback on the product's overall usability.

Test design for usability testing of the Daisy Dub:

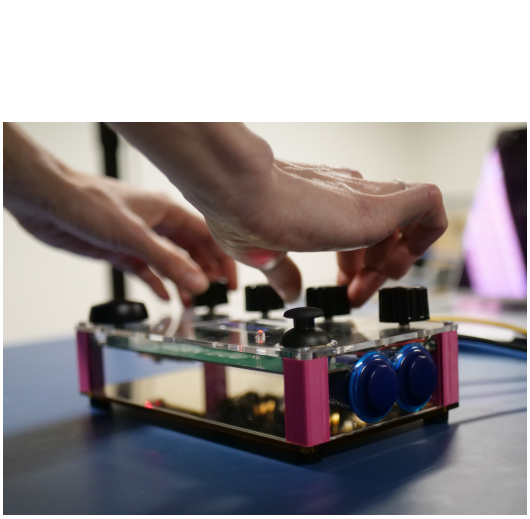
1. Test objective: The objective of this usability test is to gather feedback on the usability of Daisy Dub, and to identify any issues or challenges that users may face when using the product.
2. Test subjects: The target audience for this usability test will be music producers, performers, and DJs, who are representative of the target audience for the product.
3. Test environment: The usability test will be conducted in a controlled environment, such as a dedicated testing lab or a quiet room, where users can comfortably and easily use the product.
4. Test tasks: Users will be asked to complete a series of tasks using the Daisy Dub, such as setting up the effect, adjusting parameters, and applying the effect to audio signals.
5. Test measures: The usability test will be conducted using a combination of quantitative and qualitative measures, including user performance on the test tasks, user satisfaction with the product, and user feedback on the product's overall usability.
6. Test participants: A sample of 5 test subjects will be recruited to participate in the usability test.
7. Test materials: Daisy Dub, a computer with a sound card outputting audio to Daisy Seed, and any necessary documentation or instructions will be provided.
8. Test debrief: Following the usability test, users will be asked to provide feedback on their experience using the product and to suggest any improvements or recommendations for future development.

Just before testing, it was noticed that four resistors were missing on the PCB due to an error in the bill of materials document. The missing resistors were R49 through R52. It is possible to get the Daisy Dub running without these resistors, but in order for it to function properly these four parts are fairly important. Luckily it was possible to get them on in time for the testing, and we would like to extend our gratitude to Peer Klausen from Gadget Group for helping us with this issue.

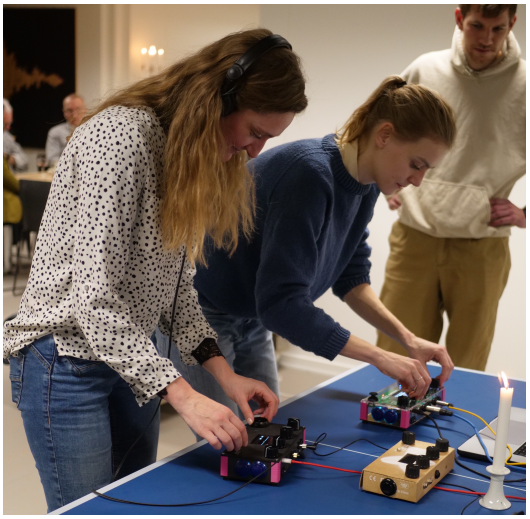
4.6.1 Active listening test - Sound Hub Denmark

An active listening test was conducted at Sound Hub Denmark. The test involved 20 participants who took turns playing with one of two set-up devices (see figure 4.47). One device was connected to speakers, while the other was connected to headphones. Both units received audio from an individual source and consisted of dub music, house music, dance music, and dub techno. The participants represented diverse individuals, including entrepreneurs, developers, engineers, administration, and business/economy professionals. The test aimed to gather participant feedback and suggestions on the device's performance and sonic quality. The participants were allowed to experiment with the device, exploring its various parameters and controls. They were then asked to provide feedback on their experience, highlighting what they liked and disliked about the device.

The participants provided positive feedback on the Daisy Dub device. They enjoyed its performance, sonic quality, and versatility, noting that it was easy to use and produced exciting and unique effects. Some participants suggested additional features, such as saving and recalling presets. The test demonstrated that the Daisy Dub device is well-suited for various applications, from creative music production to live performance. Its intuitive interface and versatile sound capabilities make it a valuable tool for musicians and audio professionals. The active listening test for the Daisy Dub device was successful in gathering valuable feedback from a diverse group of participants. The positive feedback received highlights the device's strengths, while suggestions for additional features and controls provide opportunities for further development and refinement.



(a) Daisy Dub prototype, see-through top plate and sides, connected to a single hifi audio speaker at Sound Hub Denmark



(b) Test setup with two device prototypes, each with their own sound source of various types of music

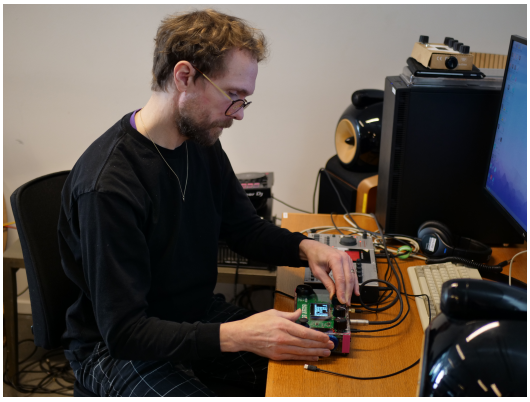
Figure 4.47: Listening test setup at Sound Hub Denmark during a gathering of Beyond Beta incubator participants, employees of Sound Hub Denmark and visitors from Danish Sound Cluster. One device is connected to headphones, and the other is connected to a single hifi audio speaker.

4.6.2 Active listening test - ME-Lab, Sound & Music Computing, Aalborg University

A similar active listening test was performed with a single Daisy Dub at Aalborg University's Sound & Music Computing (SMC) Multi-sensory Experience LAB (ME-Lab)[10] during the networking event SMC By Night (see figure 4.48). The aim of the test was to evaluate the performance and user experience of the device, particularly in the context of live electronic music performance. The test involved a group of music production, music computing and programming, medialogy students and alumni; some were familiar with the Elektron Machinedrum and its capabilities, but most were not. The Machinedrum served as the primary sound source for the first hour of testing. Afterwards, the sound source was changed to a smartphone playing classical and choral music into Daisy Dub. The device was connected to hifi stereo audio speakers.

The participants were given the opportunity to experiment with the device, exploring its various parameters and controls, using the device to process and manipulate the sound of the Machinedrum or the music. The test aimed to evaluate the device's performance and its impact on the participants' creative process and overall musical expression. During the test, the experimenters took notes and engaged in dialogue with the test subjects. The test demonstrated that the device effectively enhanced the users' creative output and provided a unique sound experience. Participants praised the device's intuitive interface and the flexibility of its effects and processing options. Remarks were made on the clipping functions of the device, as well as the overly sensitive joystick XY-axes. The test highlighted the device's potential as a valuable tool for live electronic music performance. It provided important insights into the user experience and performance of the device in a real-world context. In conclusion, the test performed at Aalborg University's Sound and Music Computing programme successfully evaluated the performance and user experience of a device designed for

[10] `https://melcph.create.aau.dk/`



(a) Daisy Dub prototype, see-through top plate and sides, connected to stereo hifi audio speakers at ME-Lab. Audio input from Elektron Machinedrum. Casually being tested and evaluated by Danish electronic music pioneer and legend, Bjørn Svin.

(b) Test setup with one prototype device; first hour sound source was an Elektron Machine drum connected directly to Daisy Dub. Afterwards, the sound source was changed to classical and choral music being fed from a smartphone.

Figure 4.48: Listening test setup at Multisensory Experience LAB, Aalborg University Copenhagen during SMC By Night networking event. The Daisy Dub prototype device is connected to stereo hifi speakers and receives audio from either an Elektron Machinedrum sequencer or stereo audio as classical or choral music being fed from a smartphone.

live electronic music performance and production. The test provided essential insights into the user experience and performance of the device, which can be used to inform future development and refinement. The authors send a big thanks to Bjørn Svin[11] for providing valuable feedback, praise and critique to the device.

[11] `https://seismograf.org/profiler/bjoern-svin`

Chapter 5

Evaluation

To evaluate the newest prototype of the Daisy Dub, a usability test was conducted, similar to the one conducted at the start of the project, in section 2.7.1. The focus of this test is to see whether the new hardware of the Daisy Dub is still nice to use, and with the improvements in the menu, the hope is that one of the biggest complaints from the first test is gone.

This test was conducted at the Aalborg University campus in Copenhagen, in the Multisensory Experience LAB, before and during SMC By Night. The test subject was seated at a table where the test conductor, and the note taker could easily see what they were doing during the test.

For this evaluation 5 participants were used. This is because [17] argues that any more gives a poor return on investment when looking for usability issues with a product. The test participants ranged in age from 27 to 36, and all had varying expertise in audio effects, scoring on either end of a subjective scale given at the start of the test.

5.1 Procedure

Before commencing the test, the test subjects were given a consent form to sign, the template for this form can be found in appendix E. After this, the script for the testing was followed by the test conductor. This script includes a small introduction to the project and then asks the participant to state their age, as well as give an estimate of their knowledge of audio effects. They are asked about their knowledge of audio effects mainly to ascertain if some terms might require a bit more explanation during the testing, as someone unfamiliar with audio effects may not know the term self-oscillation for instance. After this the test subject is given the opportunity to familiarise themselves with the Daisy Dub, this simply means that they are given time to use the device without any goal in mind. This is done so that they will have some idea of which controls do what during the test, so that doesn't have to be explained, which should lessen any potential confusion. When the test subject feels familiar enough with the Daisy Dub, the test begins. It consists of 9 tasks that are given one at a time. As the knobs are not labeled, some of these tasks require some audio effects knowledge to complete. Task 2 for instance asks that the subject uses the filter to remove mid and high-range frequencies, and if one is not familiar with basic music terminology this may be confusing. As the test subjects are mostly students of sound- and music computing this should not be a problem, however. Lastly, after the testing is done an exit interview is performed, the questions for which can be seen here:

- On a scale of 1 to 10, where 1 is completely unexpected and 10 is exactly as expected, did the Daisy Dub react as you expected?

- On a scale of 1 to 10, where 1 is very hard and 10 is very easy, how difficult was it to find the right setting?
- On a scale of 1 to 10, where 1 is very hard and 10 is very easy, how well were you able to change a desired parameter?
- On a scale of 1 to 10, where 1 is not at all, and 10 is completely, how much did you feel like you were in control of what happened to the sound?
- On a scale of 1 to 10, where 1 is very bad, and 10 is very good, how is the sound quality of the Daisy Dub?
- Was there any interaction that you found inconvenient or would like to work differently?
- On a scale of 1 to 10, where 1 is not at all and 10 is definitely, how likely would you be to use a product like this?
- On a scale of 1 to 10, where 1 is not at all, and 10 is definitely, how likely would you be to spend money on a product like this? And can you give an estimate of how much you would be willing to pay?

The full script for the test can be found in appendix E.

5.2 Results of usability testing

During the user testing of the Daisy Dub, it became very clear that this is not a tool for inexperienced musicians, as some knowledge of the music terminology seem necessary to fully navigate the menu of the Daisy Dub. One participant scored their knowledge of audio effects as a 2 out of 10, and needed most terminology used during the testing explained. They were not able to translate the task of turning an effect on or off to adjust the dry/wet of that effect. This struggle with the most basic functionality also comes out in their SUS evaluation, as they gave the product a score of 27.5. Another test subject scored their knowledge a 4 out of 10, and with that small perceived improvement, the SUS evaluation jumps to 77.5, which highlights that the biggest hurdle to using the Daisy Dub is the knowledge of music terminology. The collective SUS score for all of the test subjects is 67.5, heavily dragged down by the one subject who scored it low. If that test subject is removed from the calculations the combined score jumps to 81. While there is no basis for excluding them from the final results, this big difference shows that with this test it is still not sure whether the Daisy Dub lives up to the qualitative demand of reaching a SUS score of at least 79.5.

Chapter 6

Discussion

In this section, the future of Daisy Dub will be explored. This includes the notes gathered from user testing, but also other thoughts for expanding the uses that the Daisy Dub may fill.

6.1 Improvements

During the evaluation of the Daisy Dub, the test subjects shared a lot of thoughts that should be taken into account when the next prototype of the Daisy Dub is developed. While the feedback overall from the test subjects was very positive, it was clear that certain aspects needed to be improved upon in order to ensure better user interaction.

6.1.1 Menu

Although improvements were made to the menu following the initial user test, all of the participants struggled with certain elements in the menu. Where it was a more general issue in the initial test, this time it seems like it was specific options that were confusing. When prompted to adjust the delay time modulation every single subject adjusted the delay time multiplier instead. Due to the limited space on the OLED screen and the phrasing of the task, it is clear that this mistake is happening because the modulation is noted as "mod_amount", and the multiplier is listed as "dlytime_mul". Although this may just be an unexpected consequence of the task phrasing, some more dedicated testing of the menu could be done to try and minimize any confusion.

6.1.2 Software

For the possible improvements to the software, it seems to become more subjective as to what could be added. The intention to have the Daisy Dub be a modular and customizable effect does mean that a lot of signifiers or labels that might help the usability in its current state would become counterproductive if an end user decides to map the controls differently. There was however still some improvements suggested during the testing, that seem like good additions to the platform.

Dry/wet control
One subject suggested adding toggles for some of the effects in order to quickly turn them on and

off. Something similar was suggested in the initial test, where some of the subjects wanted one of the potentiometers to control the master dry/wet. With the plan of making the different parameters mappable to any control, it should fix the issue as it is easy to quickly turn the potentiometer all the way to either fully dry or wet when needed.

Encoder speed
One subject suggests an acceleration of the encoder, as it is rather slow to gradually turn up the values of parameters. Adding acceleration so that the rate of change is much higher if the encoder is turned faster, would still allow for finer control so this should be implemented into the Daisy Dub.

Parameter scaling
Some of the test subjects expressed a want to change the scale of some of the parameters. One subject thought it would be better to have more resolution between 0 and 100% feedback, and then only go up to around 120% at the max. Another subject expressed their interest in configuring scaling for easier use in a performance setting. Allowing the user to scale the parameters within certain limitations would make the Daisy Dub useful for more people, and also goes well with the mentality of customization.

6.2 DIY workshop for Daisy Dub

Hosting a workshop like the Daisy Dub event is an excellent way to test the mechanical design and repairability of the unit, as well as to gather valuable feedback from the target user group. First, the workshop provides an opportunity to observe and evaluate the assembly process in real-time. This allows for the identification of any issues or challenges that may arise during the assembly process, and for the development of strategies to address these issues. By watching participants assemble the unit, workshop facilitators can identify areas where the instructions may be unclear, or where the mechanical design may be problematic. This feedback can then be used to refine and improve the design of the unit, making it easier and more intuitive to assemble. Additionally, hosting a workshop allows for the gathering of valuable feedback from the target user group. By interacting with participants and gathering their insights and opinions, workshop facilitators can gain a deeper understanding of the needs and preferences of music producers and performers. This feedback can then be used to shape and refine the design of the unit, ensuring that it meets the needs and expectations of the target user group. Finally, hosting a workshop like the Daisy Dub event is an excellent way to build community and create a sense of engagement and ownership among participants. By assembling the unit themselves and interacting with workshop facilitators and other participants, workshop attendees can feel more connected to the product and invested in its success. This sense of community and engagement can be an invaluable resource for the development and refinement of the unit.

The workshop

The Daisy Dub DIY workshop is planned to be held at the Electronic Music School Rumkraft in Copenhagen, Denmark in cooperation with Copenhagen Music Maker Space. The workshop will have a participation fee, and participants will be given the opportunity to assemble their own Daisy Dub unit as part of the workshop. The workshop will be structured as a series of interactive sessions, each focusing on a different aspect of Daisy Dub. Participants will be invited to assemble the unit themselves, following detailed instructions from the workshop facilitators. Along the way, participants will be encouraged to ask questions and seek guidance as needed. Throughout the

workshop, participants will be able to explore the various features and functions of the Daisy Dub and try out different effects and settings. The workshop facilitators will be on hand to provide assistance and facilitate discussions about the unit and its capabilities. This will allow participants to provide valuable feedback and insights that will be used to refine and improve the unit as it moves toward final production. In addition to the hands-on sessions, the workshop will include a series of surveys and interviews designed to gather participant feedback and insights. These surveys and interviews will be conducted by the workshop facilitators and will be designed to gather detailed and nuanced responses about Daisy Dub and its usability. The Daisy Dub DIY workshop is expected to be a highly successful and engaging event, providing a valuable opportunity for participants to explore the unit's capabilities and share their experiences and insights with the rest of the group. The feedback and insights gathered from participants will be invaluable in refining and improving the unit in the future. See Appendix F for more info on the workshop.

6.3 Firmware development and web-based update utility

A key feature of the Daisy Seed platform is the ability to update the firmware, which allows users to add new features, change existing ones or fix bugs. This fits perfectly with the wish to make the Daisy Dub customizable for the end user. While the user will be able to flash new code onto the Daisy Dub by simply putting it in bootloader mode and using the same update utility provided by Electrosmith, a more focused solution would be ideal for future versions. During the development, an update to the firmware of the Daisy Seed was released that accidentally removed a vital JSON file, and by creating a specialized web-based solution for the Daisy Dub, potential mistakes like this could be prevented. Developing the update utility was out of the scope of the thesis, but will be explored as future work.

High-level overview:

To implement the firmware update utility, the following high-level steps will be followed:

- Implement the client-side program using a programming language such as JavaScript that is compatible with web browsers. This client-side program includes a user interface for selecting and uploading the firmware file, as well as functions to communicate with the Daisy Seed development platform and update the firmware.
- Host the client-side program on a web server or cloud platform so that it can be accessed from a web browser.
- Implement code validation, checks, and error reporting.
- Configure the development platform to accept firmware updates over USB or via SD card or USB thumb drive.
- Test the update utility to ensure that it is working correctly and is able to update the firmware.

Low-level implementation details:

Here are some of the key implementation details of the firmware update utility:

- Client-side program: The client-side program will be implemented using JavaScript and is hosted on a web server. It includes a form for the user to select and upload the firmware file, and a function to send a request to the web server to update the firmware on the development board. The function uses the fetch API to send a POST request with the firmware file as the body of the request. If the request is successful, it displays a success message to the user.

- Web server program: The web server program will be implemented using Python and is hosted on a server or cloud platform. It includes a route for the firmware update web page that handles the POST request from the client-side program and a function to update the firmware on the microcontroller using the libdaisy library. The libdaisy library provides a Python interface for communicating with the Daisy Seed development board over a USB connection.
- Microcontroller configuration: To accept firmware updates over USB, the device must be configured to accept updates. This will be done using the libdaisy library and our custom bootloader.
- Testing: To ensure that the firmware update utility is working correctly, it will be tested on machines running Linux, Windows, macOS and ChromeOS. This includes testing firmware compiled directly from Oopsy and firmware compiled from Visual Studio Code or a similar integrated development environment (IDE).

In this section, a future process of implementing a firmware update utility for the Daisy Seed platform that runs in the browser was described. The firmware update utility allows users to update the firmware of the device from a web browser, without requiring specialized software. By following the high-level steps outlined in this chapter and implementing the low-level details described, it should be possible to create a firmware update utility that is convenient, reliable, and secure and it will greatly improve the usability, appeal, longevity, and hackability of the device.

6.4 Daisy Dub and STEAM education

STEAM education, an acronym for Science, Technology, Engineering, Arts, and Mathematics, is a multi-disciplinary approach to learning that focuses on the integration of these subjects in order to foster critical thinking, problem-solving, and creativity in students. In recent years, there has been a growing emphasis on the importance of STEAM education in preparing students for the increasingly technical and globalized world of the 21st century. In this chapter, we will explore the potential benefits of incorporating Daisy Dub into STEAM education, specifically in the final years of K-12 education and at the bachelor level. The Daisy Dub is a unique and versatile product that offers numerous benefits for STEAM education. One of the key features that makes it ideal for educational purposes is its modular and customizable design. Allowing students to assemble and program the unit themselves encourages hands-on learning and fosters creativity and critical thinking skills. Additionally, the fact that the unit can be made using easily accessible materials such as cardboard or bamboo with a laser cutter, and can be programmed using open-source platforms such as Arduino, PureData, Rust and C++, means that it is an affordable and accessible option for educational settings. This makes it an excellent tool for introducing students to the principles of engineering, programming, and digital audio processing in a fun and engaging way. In the final years of K12 education, Daisy Dub can provide students with a hands-on approach to learning about audio processing and programming. Its modular design allows students to experiment with different configurations and better understand the principles behind audio effects. Additionally, the programmable microprocessor opens up the opportunity for students to learn how to code, using platforms such as Max/Gen, PureData, Arduino, C++ and Rust. At the bachelor level, Daisy Dub can serve as an excellent learning tool for students interested in audio engineering, music production, and programming. Its updateable firmware and open-source nature encourage students to constantly improve upon their designs and develop new audio effects. Furthermore, Daisy Dub's programmability allows students to explore and experiment with different programming languages and approaches to real-time audio processing. By allowing students to assemble and program the unit themselves, it encourages a deeper understanding of engineering and programming concepts, as well as fostering creativity and

critical thinking skills. This makes it an excellent tool for introducing students to a range of STEAM subjects and providing a bridge between these subjects in a real-world context. Furthermore, Daisy Dub is a highly customizable product, with the ability to be programmed using a variety of platforms, including open-source alternatives such as Arduino, PureData, Rust and C++. This allows students to tailor the unit to their specific needs and interests, and to explore a range of programming languages and approaches.

The Daisy Dub is an innovative and educational product that has the potential to inspire and educate students in their final years of K-12 education and at the bachelor level in a wide range of STEAM subjects from programming and engineering to music production, and design. Its modular and customizable design, versatility, and accessibility make it an excellent choice for experiential education and real-world problem solving in the classroom and higher education settings.

Chapter 7

Conclusion

7.1 Summary of findings and contributions

In this thesis, we have explored the design and implementation of the Daisy Dub, a modular and updateable real-time audio effect for music producers and performers. Through a state of the art analysis, we have identified the key features and characteristics of contemporary modular real-time audio effects, with a particular focus on delay units. We have also introduced the Daisy Seed platform and Gen by Cycling '74 as key technologies for the development of Daisy Dub. In the Design and implementation chapter, we have outlined the mechanical, electrical, and software design considerations for Daisy Dub. We have also proposed a firmware update utility and hosting solutions for this utility. Daisy Dub is a versatile and user-friendly tool for music production and performance and has the potential to be a valuable addition to the arsenal of modular real-time audio effects available to musicians and producers. We hope that this research will inspire further exploration and development in the field of modular real-time audio effects.

7.2 Limitations and future work

The limitations of Daisy Dub include the fact that it is still in the prototyping phase and has not yet undergone thorough testing or mass production. As such, there may be unforeseen issues or problems that arise once the device is more widely used. Additionally, the current design is focused on music production and performance, and may not be suitable for other applications or users. In terms of future work, there are several areas that could be explored in order to improve and expand upon the capabilities of Daisy Dub. For example, further research could be conducted into optimizing the signal processing algorithms or adding additional features such as pitch shifting or reverb. It could also be interesting to investigate the potential for integrating the device with other modular systems or incorporating it into a larger ecosystem of musical instruments and effects. Additionally, further user testing and evaluation could be conducted to identify areas for improvement and gather feedback from a wider range of users.

7.3 Implications and applications

The Implications and applications chapter aims to explore the potential impact and practical applications of Daisy Dub. In the context of this research paper, implications refer to the consequences or effects that the research and development of the Daisy Dub unit may have on the field of modular real-time audio effects for music production and performance. These implications could be practical, such as the potential for the unit to become a popular choice among music producers and performers, or theoretical, such as the contribution the unit and its development process may have on our understanding of audio signal processing and user-centered design. Additionally, the implications of this research could extend beyond the specific field of audio effects, as the techniques and methodologies used in the development of the Daisy Dub could potentially be applied to other areas of technology development and design. One potential application of Daisy Dub is in the realm of music production and performance. The compact size and customizable nature of the unit make it a useful tool for both studio and live settings. Its programmable microprocessor and ability to process audio on a per-sample basis allow for a wide range of delay effects, from traditional echoes to more experimental and creative sounds. This versatility makes the Daisy Dub a valuable tool for music producers and performers looking to expand their sonic palette. Another potential application of the Daisy Dub is in education. Its modular design and the fact that it can be assembled from readily available components make it an ideal platform for hands-on learning in STEAM education. Its programmable microprocessor allows for a range of programming languages to be used, making it accessible to students with varying levels of programming experience. Daisy Dub's open-source nature also allows for collaboration and community involvement in its development, providing students with the opportunity to contribute to a real-world project.

In conclusion, Daisy Dub has the potential to make a significant impact in the field of music production and performance, as well as in education. Its modular and updateable design makes it a versatile and valuable tool for a wide range of users, and its open-source nature allows for community involvement and collaboration in its development.

Acknowledgements

We would like to express our deepest gratitude to Professor Stefania Serafin for her invaluable guidance, support, and encouragement throughout the process of developing this research paper.

Professor Serafin has been an invaluable resource, providing us with valuable insights, feedback, and direction at every stage of the research process. Her expertise in the field of audio effects and signal processing has been invaluable, and her guidance has helped us to focus our research and to develop a clear and coherent direction for our work.

We are deeply grateful for Professor Serafin's tireless support and encouragement, and we are proud to have had the opportunity to work under her supervision. We are confident that this research would not have been possible without her guidance and support, and we are deeply thankful for everything that she has done for us.

We would also like to thank the rest of our research team, friends and colleagues, who have contributed invaluable insights, feedback, and support throughout the process of developing this research paper. Especially Razvan Paisa, Peter and Jesper from Aalborg University Copenhagen E-lab whose contributions have been invaluable.

We would also like to thank Sound Hub Denmark and the Beyond Beta program for providing us with the resources, support, and opportunities that have been essential to the development of this research paper. Sound Hub Denmark has been an invaluable resource, providing us with access to state of the art facilities, equipment, and expertise that have been essential to our research. The support and guidance of the Sound Hub Denmark team have been invaluable, and we are deeply grateful for everything that they have done for us. The Beyond Beta program has also been an invaluable resource, providing us with funding, support, and opportunities that have been essential to the development of this research paper. The support and guidance of the Beyond Beta team have been invaluable, and we are deeply grateful for everything that they have done for us. We are deeply grateful to Sound Hub Denmark and the Beyond Beta program for their invaluable support and guidance, and we are proud to have had the opportunity to be a part of their community.

Finally, we would like to thank the Electronic Music School Rumkraft for providing us with invaluable resources, including access to their studio facilities, the opportunity to borrow equipment, and the use of their network and connections. We are thankful for being a part of their community.

We would also like to thank our family and friends for their unwavering support and encouragement throughout this process. Their love and support have been a constant source of strength and motivation, and we are deeply grateful for everything that they have done for us.

Bibliography

[1] 1010music. *Blackbox User Manual*. https://1010music.com/wp-content/uploads/2021/03/blackbox-user-manual-1.7c.pdf. Accessed on March 5th, 2023. 2021.

[2] 1010music. *Bluebox User Manual*. https://1010music.com/wp-content/uploads/2021/07/blueboxusermanual-1-1-2-a-clean.pdf. Accessed on March 5th, 2023. 2021.

[3] Cycling '74. *Max/MSP Manual*. Documentation for the Max/MSP visual programming language. 2022.

[4] Christopher L. Bennett. *Digital audio theory: a practical guide*. Abingdon, Oxon ; New York, NY: Routledge, 2021. ISBN: 978-0-367-27655-3 978-0-367-27653-9.

[5] Lloyd Bradley. *This is reggae music: the story of Jamaica's music*. New York: Grove Press, 2001. 572 pp. ISBN: 978-0-8021-3828-6.

[6] Alessandro Cipriani and Maurizio Giri. *Electronic music and sound design. Volume 1*. eng. Trans. by David Stutz. Fourth edition. Rome: ConTempoNet, 2019. ISBN: 978-88-99212-10-0.

[7] Alessandro Cipriani and Maurizio Giri. *Electronic music and sound design. Volume 2*. eng. Trans. by David Stutz. Third edition. Rome: ConTempoNet, 2020. ISBN: 978-88-99212-14-8.

[8] Arne Eigenfeldt and Marcelo M. Wanderley, eds. *A NIME Reader: Fifteen Years of New Interfaces for Musical Expression*. New York: Springer, 2015.

[9] *Electrosmith GitHub*. GitHub. URL: https://github.com/electro-smith.

[10] Tom Erbe. "Building the Erbe-Verb: Extending the Feedback Delay Network Reverb for Modular Synthesizer Use". In: (2015), p. 4.

[11] Tom Erbe. *Tom Erbe / Soundhack "Designing the Make Noise Erbe-Verb" Reverb Design Lecture*. https://youtu.be/Il_qdtQKnqk?t=660. 2019.

[12] Rasmus Kjærbo. "Daisy Dub - The Playable Delay". English. In: *NordicSMC 2021* (2021). URL: https://nordicsmc.create.aau.dk/wp-content/NordicSMC/Nordic_SMC_2021_paper_29.pdf.

[13] Rasmus Kjærbo. "Daisy Dub - The Playable Delay". In: *Sound and Music Computing, Aalborg University, Denmark* (2022).

[14] Paul D Lehrman. *What is MIDI?* https://mma.pages.tufts.edu/emid/WhatIsMIDI2017.pdf. 2017.

[15] V. J. Manzo. *Max/MSP/Jitter for music: a practical guide to developing interactive music systems for education and more*. Second edition. Oxford ; New York: Oxford University Press, 2016. ISBN: 978-0-19-024374-6.

[16] Johann Merrich. *A short history of electronic music and its women protagonists*. eng. Trans. by Barbara Beatrice Lavitola and Mattia Brundo. 1 edizione. OCLC: 1268146837. Rome, Italy: Arcana, 2021. ISBN: 978-88-927-7067-6.

[17] Jakob Nielsen. *How Many Test Users in a Usability Study?* Visited: November 7th 2022. https://www.nngroup.com/articles/how-many-test-users/, 2012.

[18] Dan Overholt. "The Musical Interface Technology Design Space". In: *Organised Sound* 14.2 (Aug. 2009), pp. 217–226. ISSN: 1355-7718, 1469-8153. DOI: 10.1017/S1355771809000326. URL: https://www.cambridge.org/core/product/identifier/S1355771809000326/type/journal_article (visited on 02/26/2020).

[19] Daniel James Overholt. "Musical Interface Technology: Multimodal Control of Multidimensional Parameter Spaces for Electroacoustic Music Performance". In: (2007).

[20] William C. Pirkle. *Designing audio effect plugins in C++: for AAX, AU, and VST3 with DSP theory*. Second edition. New York, NY: Routledge, 2019. ISBN: 978-1-138-59189-9 978-1-138-59193-6.

[21] Miller Puckette. "The Theory and Technique of Electronic Music.pdf". In: *World Scientific Publishing Co. Pte. Ltd.* (2006).

[22] RME Audio GmbH. *Fireface UCX Manual*. 2012. URL: https://www.rme-audio.de/downloads/fface_ucx_e.pdf.

[23] Curtis Roads. *Composing electronic music: a new aesthetic*. Oxford ; New York: Oxford University Press, 2015. ISBN: 9780195373233.

[24] Andrew Rogers. *Introduction to USB Type-C*. English. Microchip Technology Inc. 21 pp. URL: https://ww1.microchip.com/downloads/en/Appnotes/00001953A.pdf. Published and archived online.

[25] Roland. *Digital Delay SDE-2500 Owner's Manual*. English. Roland. 32 pp. Published and archived online.

[26] Kenneth Steiglitz. *A digital signal processing primer: with applications to digital audio and computer music*. Dover Edition. Mineola, New York: Dover Publications, Inc, 2020. ISBN: 978-0-486-84583-8.

[27] Strymon. *El Capistan dTape Echo User Manual*. Version 1. Accessed on March 5th, 2023. Strymon. Westlake Village, CA, 2010. URL: https://www.strymon.net/wp-content/uploads/2014/07/El-Capistan-Manual-v1.0.pdf.

[28] usability.gov. *System Usability Scale (SUS)*. https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html. [26/02-2023].

[29] Graham Wakefield and Gregory Taylor. *Generating sound & organizing time: Thinking with gen~ Book 1*. eng. OCLC: 1350453643. San Francisco, California: Cycling '74, 2022. ISBN: 978-1-73259-031-1.

[30] Udo Zölzer, ed. *DAFX: Digital Audio Effects: Zölzer/DAFX: Digital Audio Effects*. Chichester, UK: John Wiley & Sons, Ltd, Mar. 11, 2011. ISBN: 978-1-119-99129-8 978-0-470-66599-2. DOI: 10.1002/9781119991298. URL: http://doi.wiley.com/10.1002/9781119991298 (visited on 09/30/2021).

Appendix A

Gen patchers and C++ code

Code, figures and files are available on our GitLab repository[1].

[1] `https://gitlab.com/northernstructuresaudio/daisy-dub`

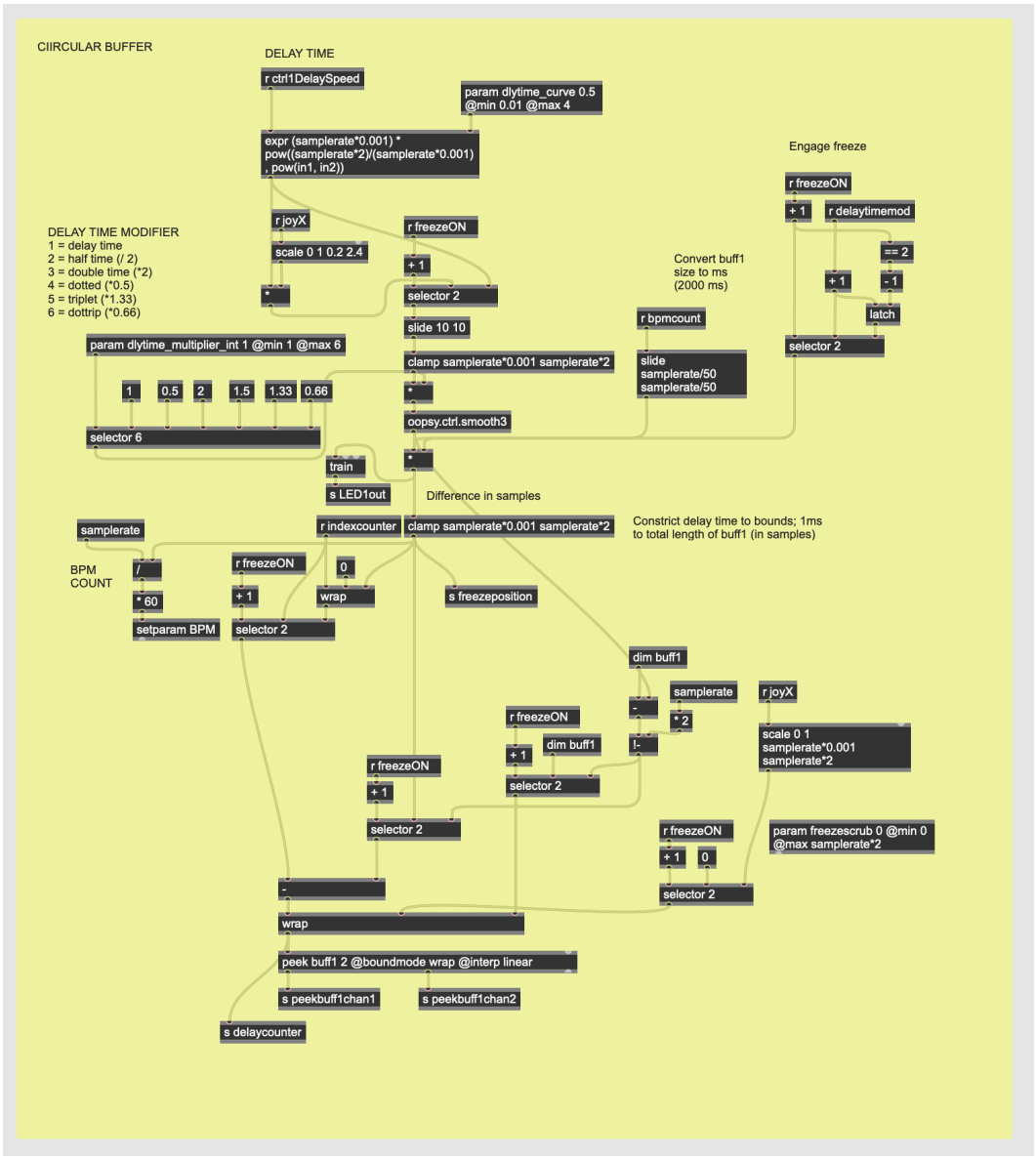


Figure A.1: gen-circular-buffer

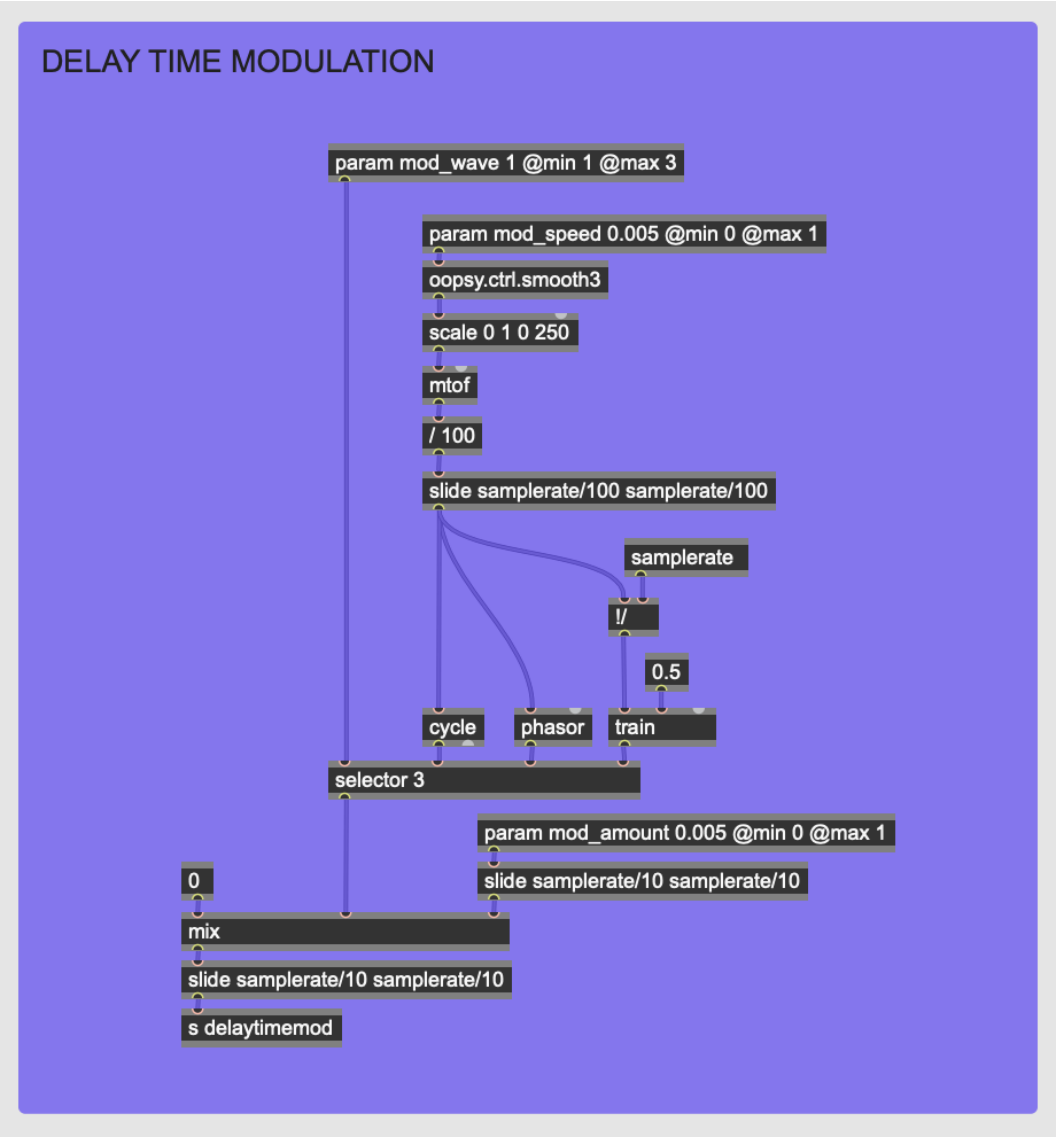


Figure A.2: gen-delaytime-modulation

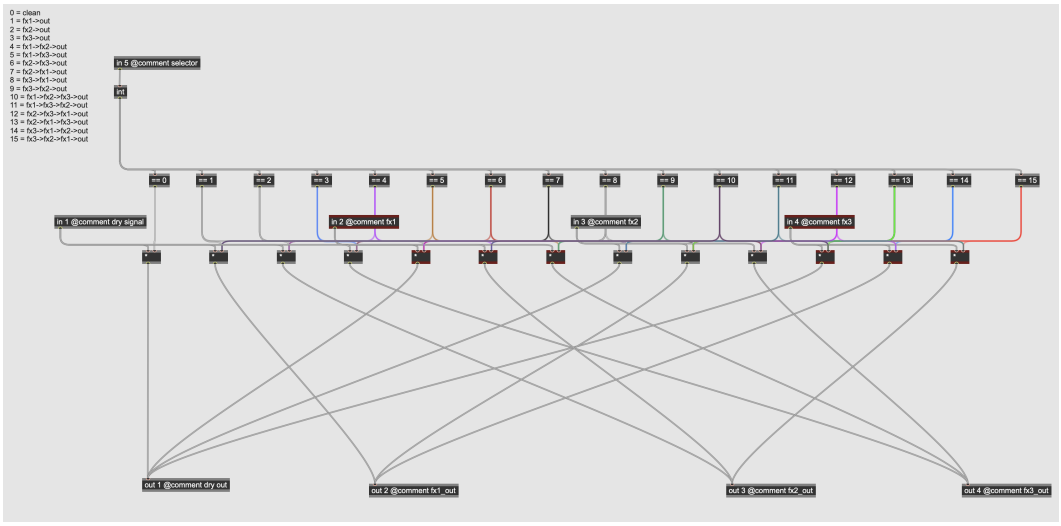


Figure A.3: gen-fx-router-gendsp

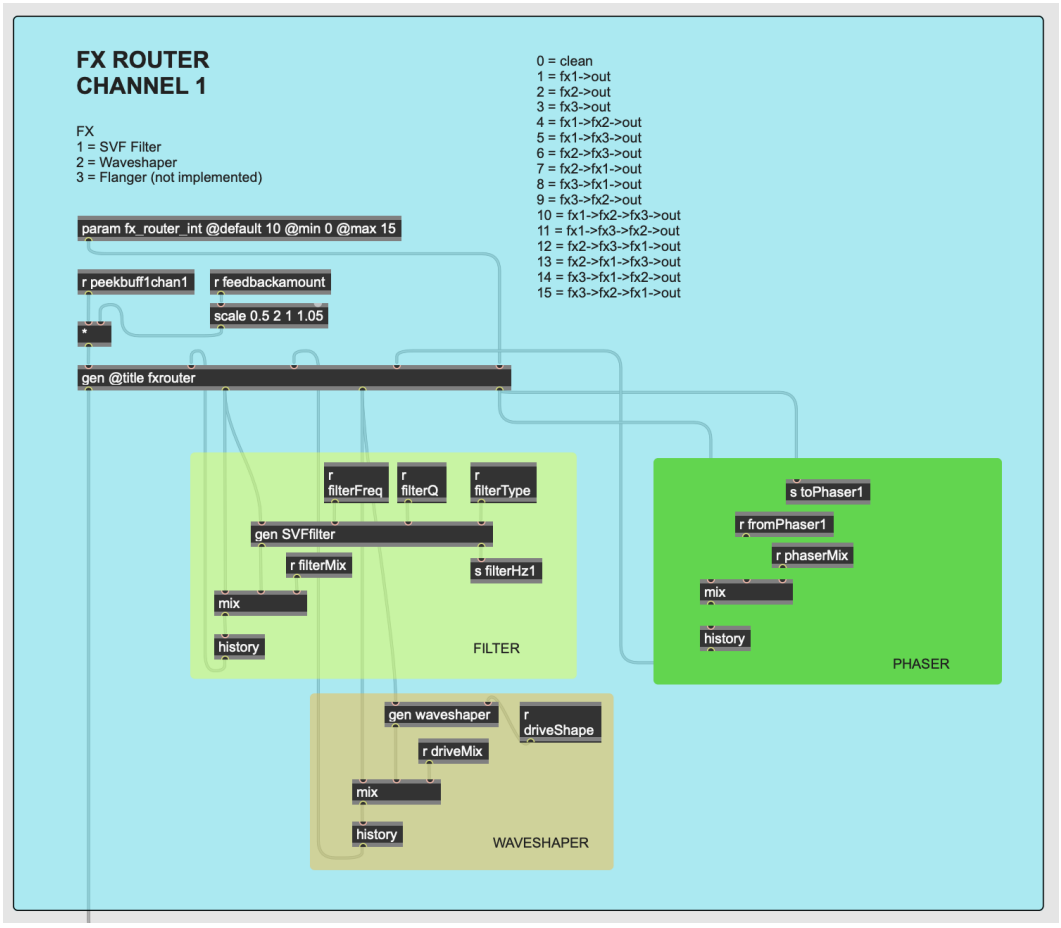


Figure A.4: gen-fx-router-overview

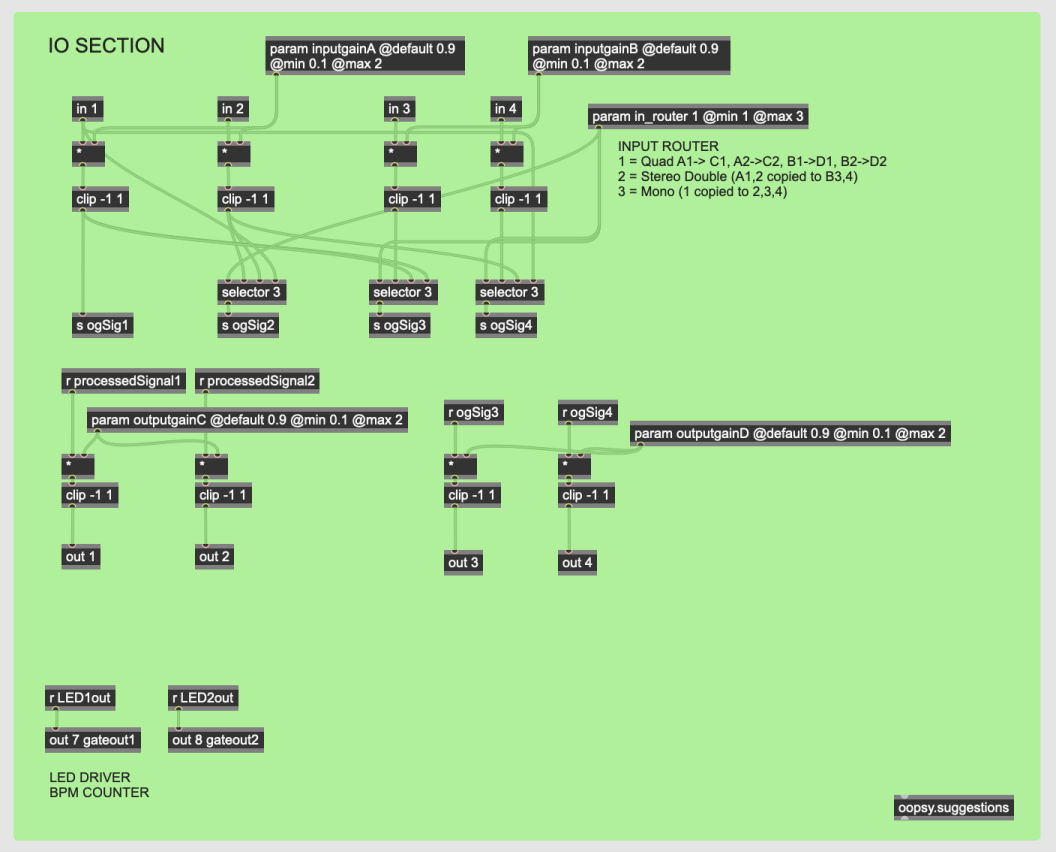


Figure A.5: gen-io-section

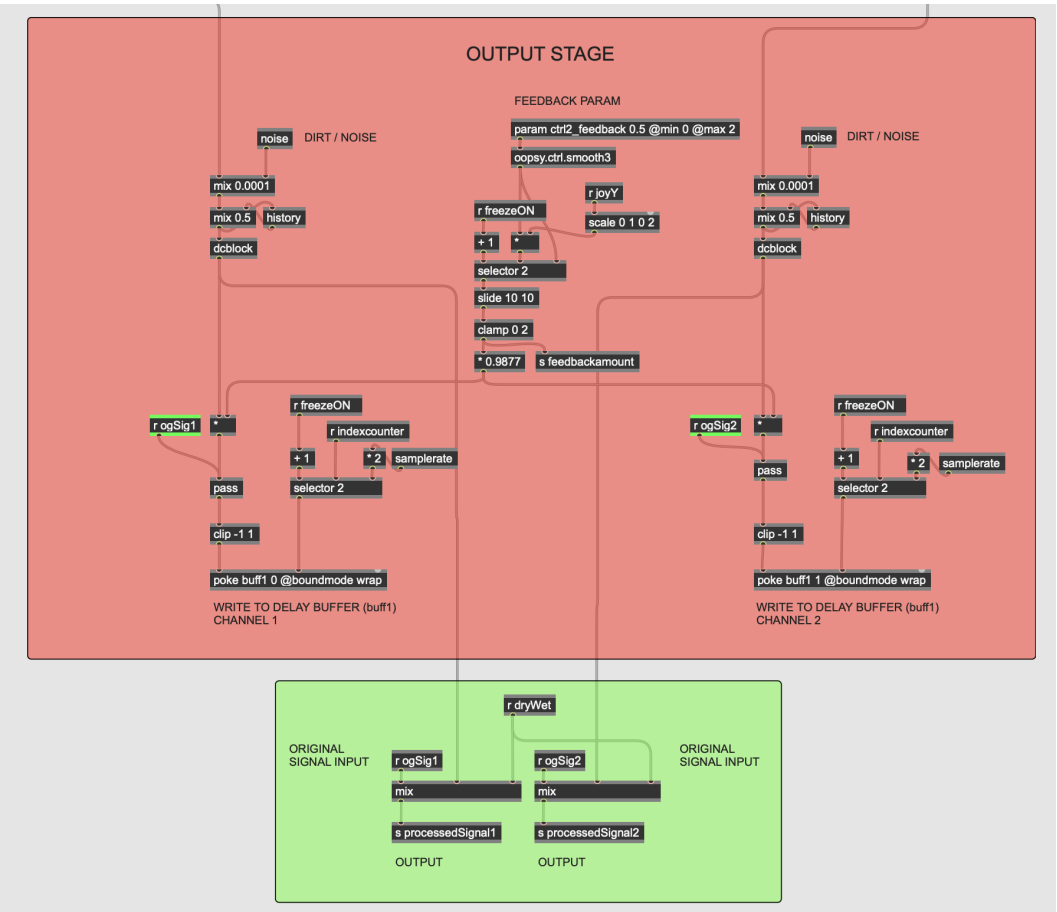


Figure A.6: gen-output-stage

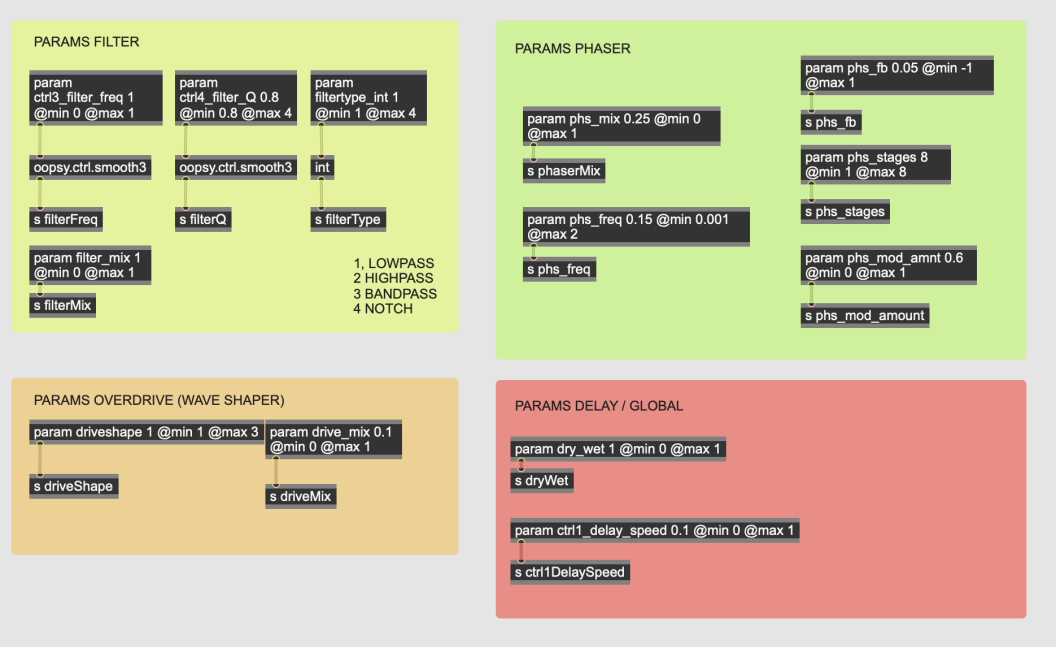


Figure A.7: gen-parameters

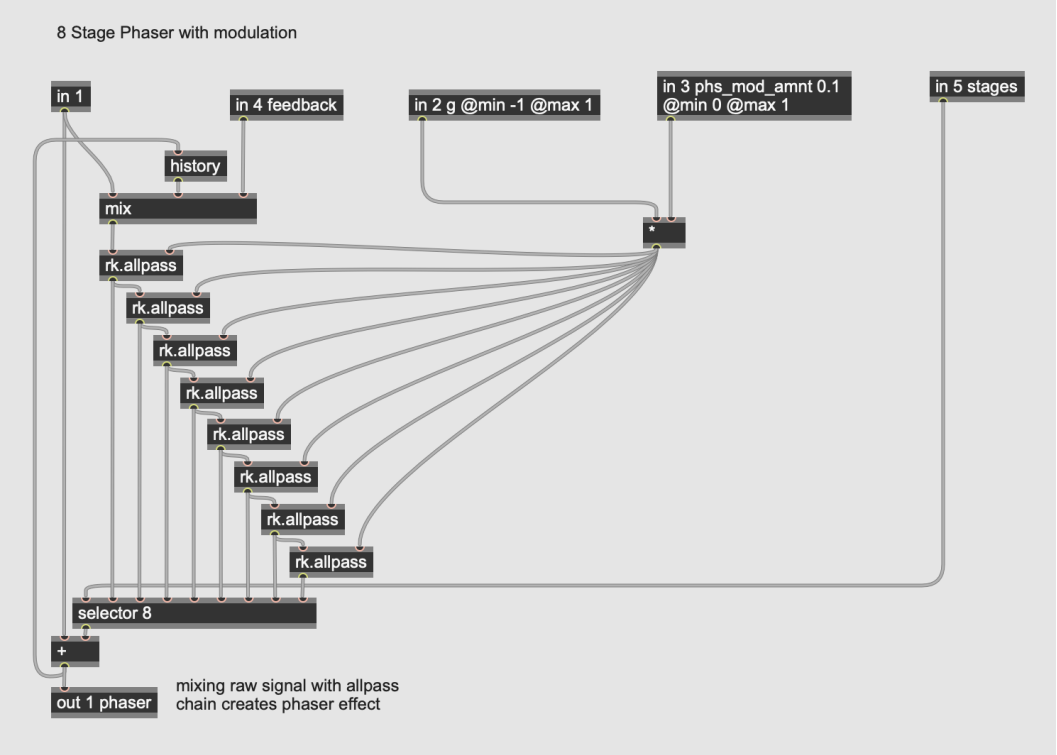


Figure A.8: gen-phaser-gendsp

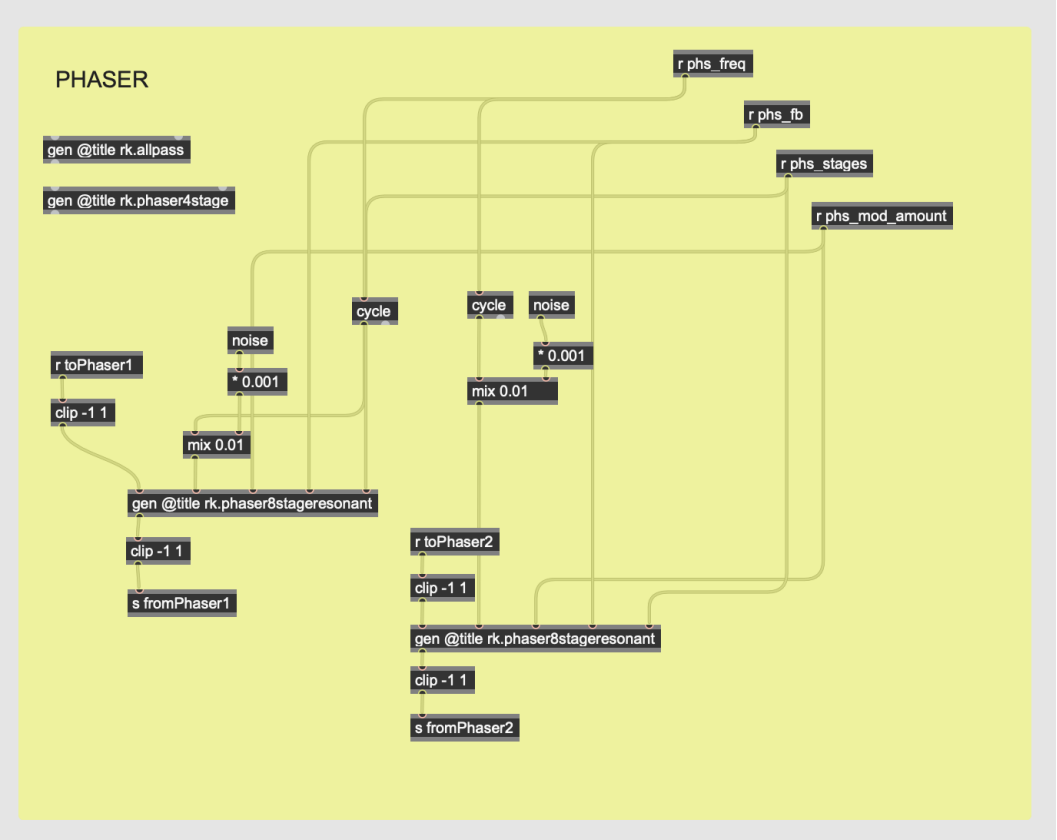


Figure A.9: gen-phaser-stereo

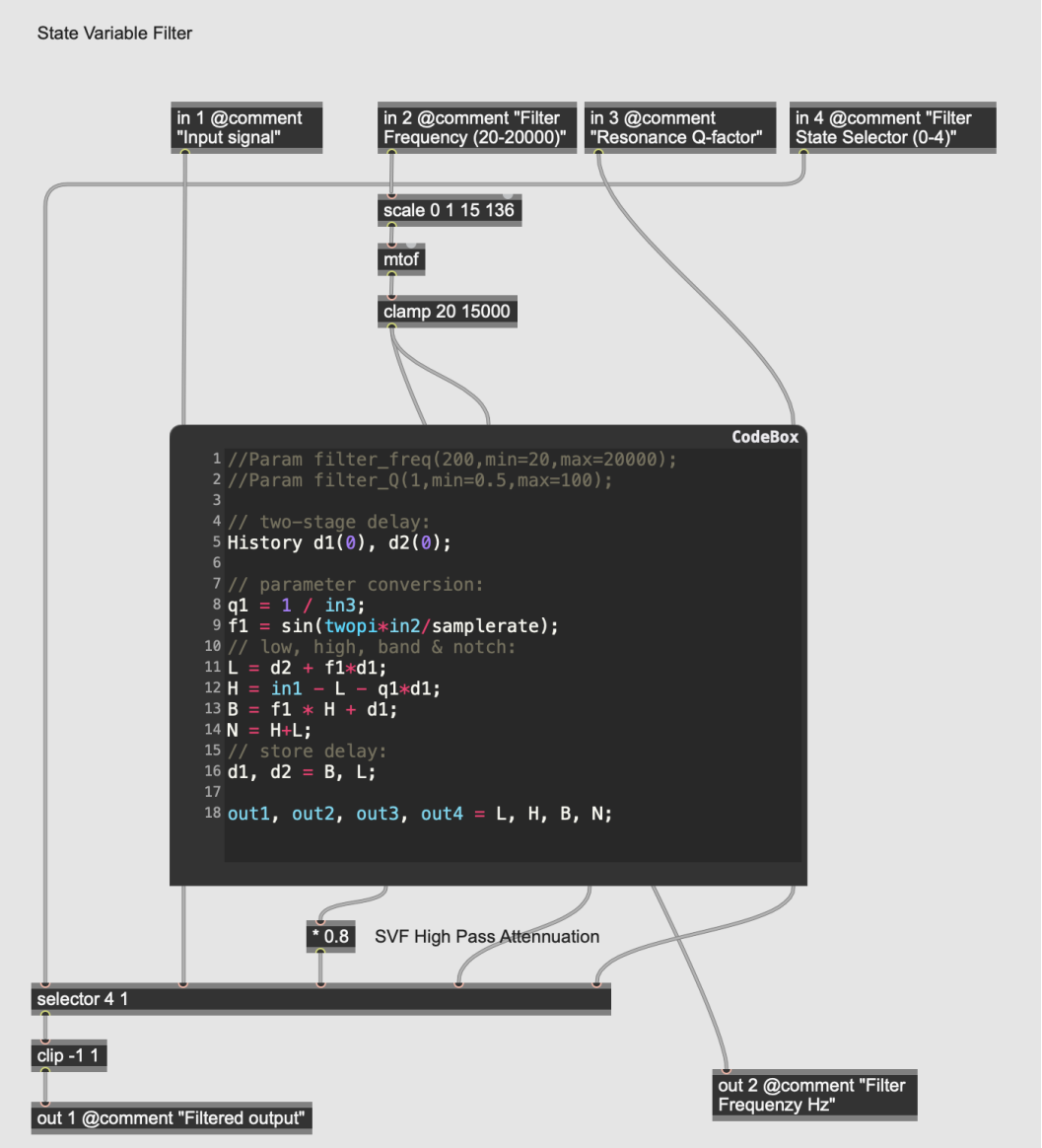


Figure A.10: gen-svf-filter

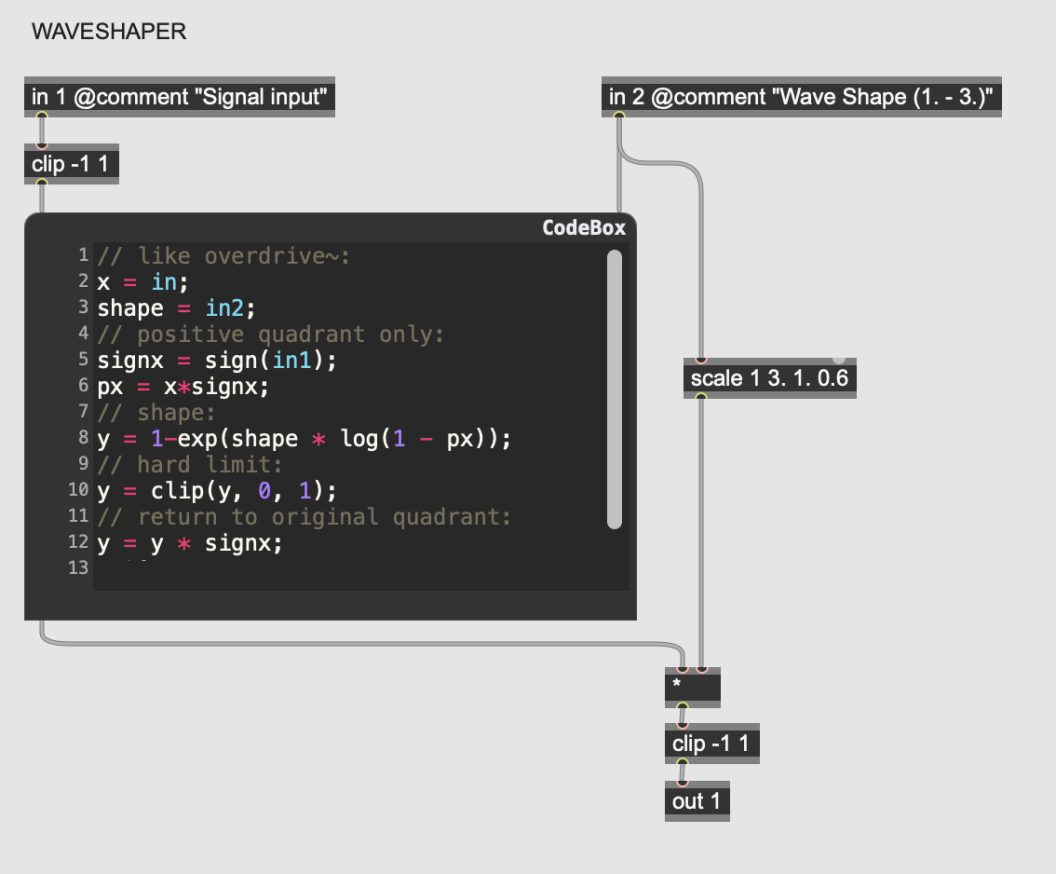


Figure A.11: gen-waveshaper

Appendix B

DSP evaluation Matlab code

Code, figures and files are available on our GitLab repository[1].

B.1 Latency test Matlab code

```
%% Latency test from algorithm 0 (fx_router = 0). Only the delay is active with 0 feedback and delay
%% at minimum (96 samples), FX bypassed. This is consistent for both tests, so the difference
%% in processing is 0.
% Load the audio signals, both samples are 96.000 Hz sample rate
[signal1, fs] = audioread('/impulseclick.aiff');
[signal2, fs] = audioread('/algo0impulse.aif');

% Convert stereo to mono
signal1 = mean(signal1, 2);

% Find and index maximum values of each signal
[maxValue1, maxIndex1] = max(abs(signal1));
[maxValue2, maxIndex2] = max(abs(signal2));

% Compute the latency in samples from the index of the maximum value found
latency_samples = maxIndex2 - maxIndex1;
latency_ms = (latency_samples / fs) * 1000;

disp(['Samplerate = ', num2str(fs)]);
disp(['Latency in samples = ', num2str(latency_samples)]);
disp(['Latency in ms = ', num2str(latency_ms), ' ms']);


% Estimate from Ableton Live to compare results
latency_samples_ableton = 6 * 96000 / 1000;

disp(['Latency in samples from Ableton = ', num2str(latency_samples_ableton)]);
```

[1] https://gitlab.com/northernstructuresaudio/daisy-dub

```
error = latency_samples_ableton - latency_samples;
disp(['Error between calculations = ', num2str(error), ' samples']);

% Set the plot limits
limit = 600;

% Trim the longer vector to the size of the shorter vector
n = min(length(signal1), length(signal2));
signal1_trim = signal1(1:n);
signal2_trim = signal2(1:n);

% Create a time vector in samples
t = (0:length(n)-1);

% Plot the time-domain representation of the signal
figure;

subplot(2, 1, 1);
plot(t, signal1_trim);
ylim([-1 1]);
xlabel('Time (samples)');
ylabel('Amplitude');
title('Signal 2');

subplot(2, 1, 2);
plot(t, signal2_trim);
ylim([-1 1]);
xlabel('Time (samples)');
ylabel('Amplitude');
title('Signal 2');
```

B.2 Spectrogram comparison Matlab code

```
%% SPECTROGRAM COMPARISON
%Read in the two audio files
% load audio files
filename_test = 'SINE WAVE 96kHz 32bit.wav';
[pathstr,name,ext] = fileparts(filename_test);
[x1, fs1] = audioread(filename_test);
x1 = x1(:, 1);

filename_recorded = 'algo0sine420Hz.aif';
[pathstr,name,ext] = fileparts(filename_recorded);
[x2, fs2] = audioread(filename_recorded);
x2 = x2(:, 1);                          %mono signal

% Set the segment start time (in seconds) and duration (in seconds)
segment_start_time = 1;
segment_duration = 1;

% Extract the specified segment from each audio file
segment1 = x1(round(segment_start_time*fs1) : round((segment_start_time+segment_duration)*fs1));
segment2 = x2(round(segment_start_time*fs2) : round((segment_start_time+segment_duration)*fs2));

% Compute the spectrograms of the two segments
nfft = 2048;
window = hann(nfft,'periodic');
overlap = round(nfft*0.5);
[s1,f,t] = spectrogram(segment1,window,overlap,nfft,fs1);
[s2,~,~] = spectrogram(segment2,window,overlap,nfft,fs2);

% Plot the spectrograms side-by-side
freq_range = [20 10000];                % for plots
figure;
subplot(1,2,1);
imagesc(t,f,20*log10(abs(s1))); axis xy; colormap('jet'); colorbar;
ylim(freq_range);
xlabel('Time (s)'); ylabel('Frequency (Hz)');
title('Spectrogram of Test Signal');

subplot(1,2,2);
imagesc(t,f,20*log10(abs(s2))); axis xy; colormap('jet'); colorbar;
ylim(freq_range);
xlabel('Time (s)'); ylabel('Frequency (Hz)');
title('Spectrogram of Processed Signal');

set(gcf, 'Position', [100 100 800 400])
```

B.3 Error plots Matlab code - matlab/errorplots.m

```
%% Error plots
% Set variables
signal_type      = 1;                       %sine 420 Hz = 1, saw 1kHz = 2
normalize        = 0;
phase_invert     = 0;
computed_latency = 1;

%limit            = 1000;                    %plot limit in samples
%mov_avg_window   = limit/2;                 %windows size for moving average error

print_figure     = 1;

% Load audio signals and set analysis parameters
switch signal_type
    case 1
        signal = 'Sinewave 420 Hz'
        disp(signal)
        filename_test           = 'SINE WAVE 96kHz 32bit.wav';
        filename_rec            = 'algo0sine420Hz.aif';
        [test_signal, fs]       = audioread(filename_test);
        [recorded_signal, fs]   = audioread(filename_rec);
        limit                   = 1000;     %plot limit in samples
        segment_start           = 4000;     %analysis start, samples
        fsine                   = 420;      %frequency of test signal in Hz
        T = 1/fsine;
        latency_samples         = 600;      %manually set for sine wave, phase inverted
        figure(1);
        set(gcf, 'Name', signal);
    case 2
        signal = 'Sawtooth wave 1 kHz'
        disp(signal)
        filename_test           = 'saw1000hz96kHz.aiff';
        filename_rec            = 'algo0saw1kHz.aif';
        [test_signal, fs]       = audioread(filename_test);
        [recorded_signal, fs]   = audioread(filename_rec);
        limit                   = 400;      %plot limit in samples
        segment_start           = 3800;     %analysis start, samples
        fsaw                    = 1000;     %frequency of test signal in Hz
        T = 1/fsaw;
        latency_samples         = 544;      %manually set for sawtooth wave, phase inverted
        figure(2);
        set(gcf, 'Name', signal);
    otherwise
        disp('Invalid case')
end

if normalize == 1
    disp('Signals normalized')
```

```
    test_signal = test_signal/max(abs(test_signal)); %normalize signal
    recorded_signal = recorded_signal/max(abs(recorded_signal)); %normalize signal
end

test_signal_mono = test_signal(:, 1);    %select one channel for mono

% Manually adjust latency compensation
if computed_latency == 1
    latency_samples = 546;               %calculated from impulse latency test
end
%latency_samples = 486;                  %manually set for sine wave, without phase inversion

% Pad recorded signal with zeros at the beginning
recorded_signal = padarray(recorded_signal, [latency_samples, 0], 0, 'pre');

% Set analysis start and duration
segment_duration = limit;
segment_end = segment_start+segment_duration;
test_signal_mono = test_signal_mono(segment_start: segment_end);
recorded_signal = recorded_signal(segment_start: segment_end);

% Invert phase if selected
if phase_invert == 1
    disp('Phase inverted')
    recorded_signal = recorded_signal*-1; %invert phase of recorded signal
end

% Create a time vector in samples for plotting and calculate samples per cycle
t = (0:limit-1);
samples_per_cycle = T * fs;

% Plot the time-domain representation of the signals
subplot(3, 1, 1);
plot(t, test_signal_mono(1:limit), 'b');
hold on;
plot(t, recorded_signal(1:limit), 'r');
hold off;
ylim([-1 1]);
xlabel('Time (samples)');
ylabel('Amplitude');
legend('Test Signal', 'Recorded Signal');
title(sprintf('Time-domain Representation, %.2f samples per cycle', samples_per_cycle));

% Compute the error between the signals
error_signal = test_signal_mono(1:limit) - recorded_signal(1:limit);
mov_avg_window   = limit/2;                %windows size for moving average error


% Scale error signal if it exceeds range of [-1, 1]
if max(abs(error_signal)) > 1
    error_signal = error_signal/max(abs(error_signal)); %normalize signal when clippping
```

```
end

% Plot the error between the signals
subplot(3, 1, 2);
plot(t, error_signal, 'k');
ylim([-1 1]);
xlabel('Time (samples)');
ylabel('Error');
title(sprintf('Error Between Signals, %d samples latency correction', latency_samples));

moving_average_error = movmean(error_signal, mov_avg_window);
%figure(2);
subplot(3, 1, 3);
plot(t, moving_average_error);
ylim([-0.25 0.25]);
xlabel('Time (samples)');
ylabel('Error');
title(sprintf('Moving average error %.5f, window size %d samples', ...
max(abs(moving_average_error)), mov_avg_window));

% Save the plot as a PNG file
% Set string variables for filename
if normalize == 1
    normalize_attribute = 'normalized';
else
    normalize_attribute = 'NOTnormalized';
end

if phase_invert == 1
    phase_invert_attribute = 'phaseinverted';
else
    phase_invert_attribute = 'NOTphaseinverted';
end

[~, name, ~] = fileparts(filename_rec);

% Print figure to PNG
if print_figure == 1
    filename_print = sprintf('time-domain-analysis-%s_%dsamplesshift_%s_%s',name, ...
    latency_samples,normalize_attribute,phase_invert_attribute);
    disp(sprintf('Printing figure %d to %s.png', signal_type, filename_print));
    %print(signal_type,filename_print,'-depsc', '-tiff', '-r300');
    print(signal_type,filename_print,'-dpng', '-r300');
end
```

Appendix C

DSP evaluation Matlab and iZotope Insight figures

Code, figures and files are available on our GitLab repository[1].

C.1 Matlab spectrograms

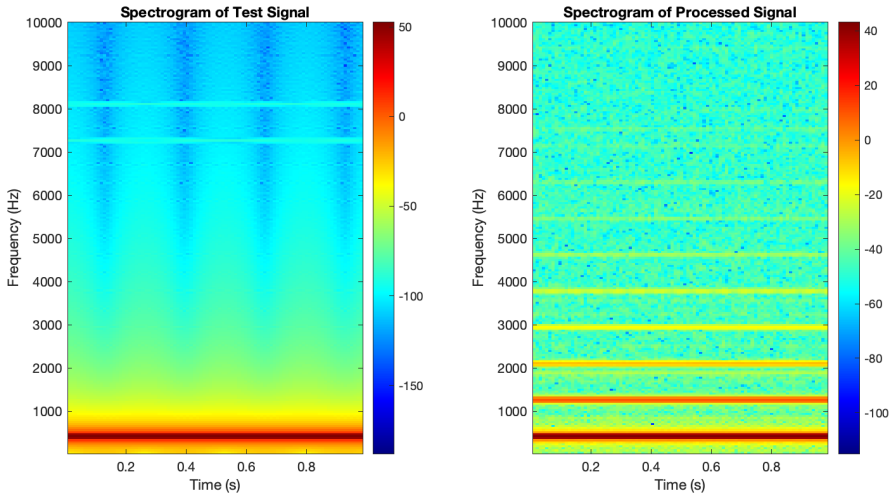


Figure C.1: spectrogram comparison algo2 sine 420 Hz Overdrive drive mix 0.5 shape2 window10000

[1] `https://gitlab.com/northernstructuresaudio/daisy-dub`

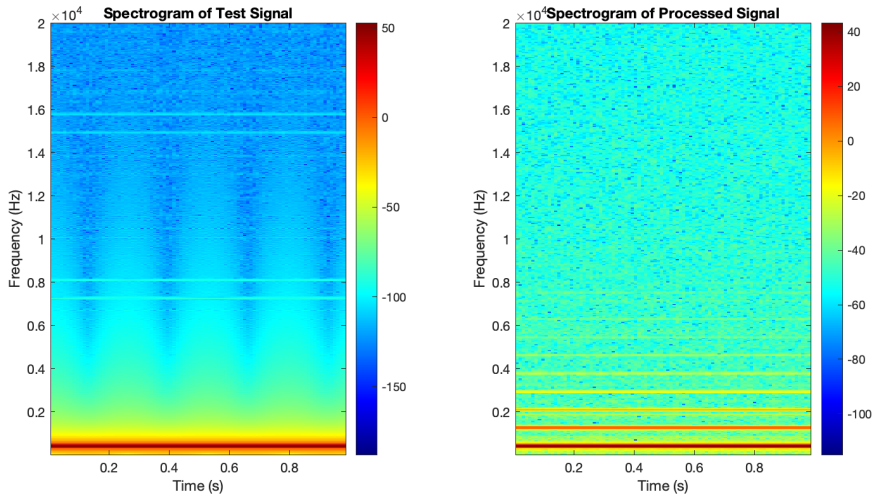


Figure C.2: spectrogram comparison algo2 sine 420 Hz Overdrive drive mix 0.5 shape2

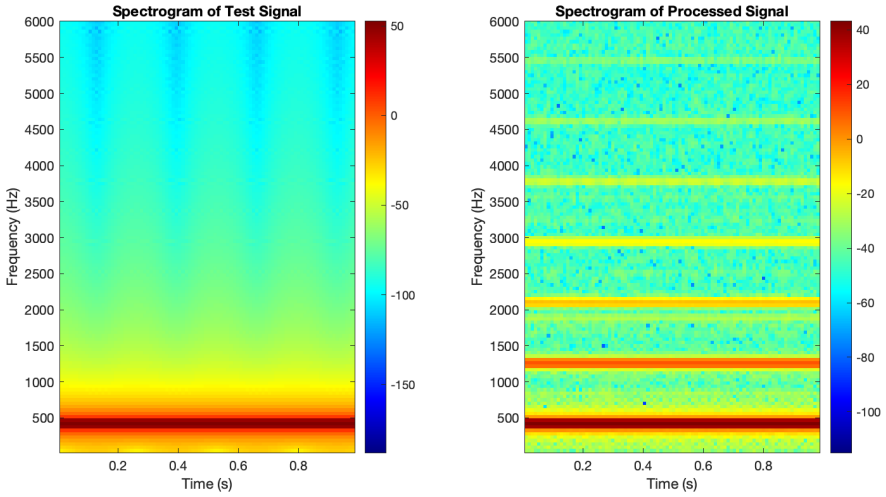


Figure C.3: spectrogram comparison algo2 sine 420 Hz Overdrive drive mix 0.5 shape2 6000 Hz window

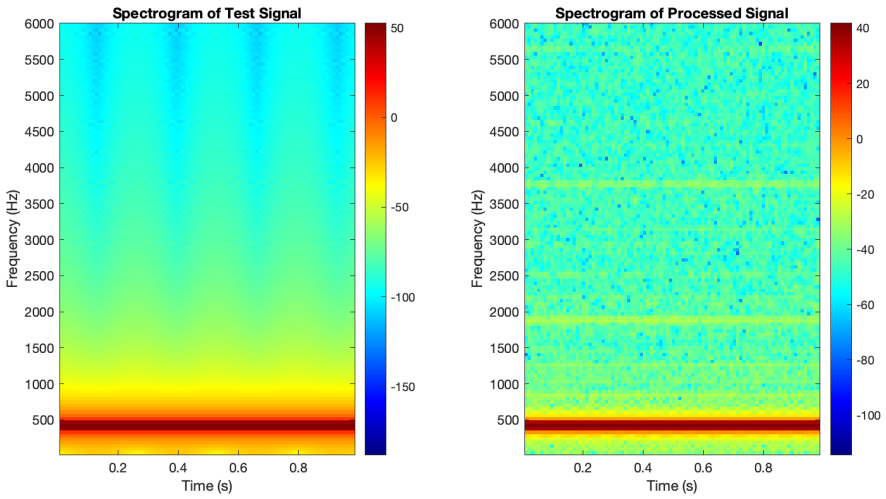


Figure C.4: spectrogram comparison algo2 sine 420 Hz Overdrive 6000 Hz window

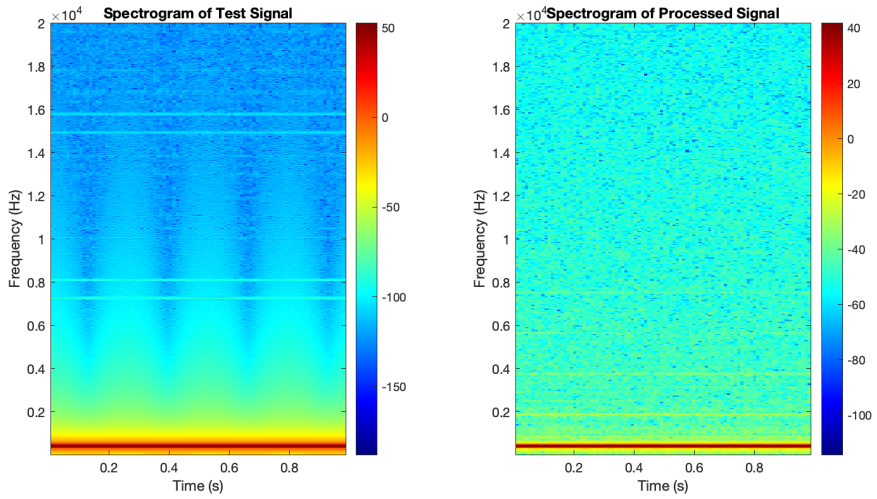


Figure C.5: spectrogram comparison algo2 sine 420 Hz overdrive

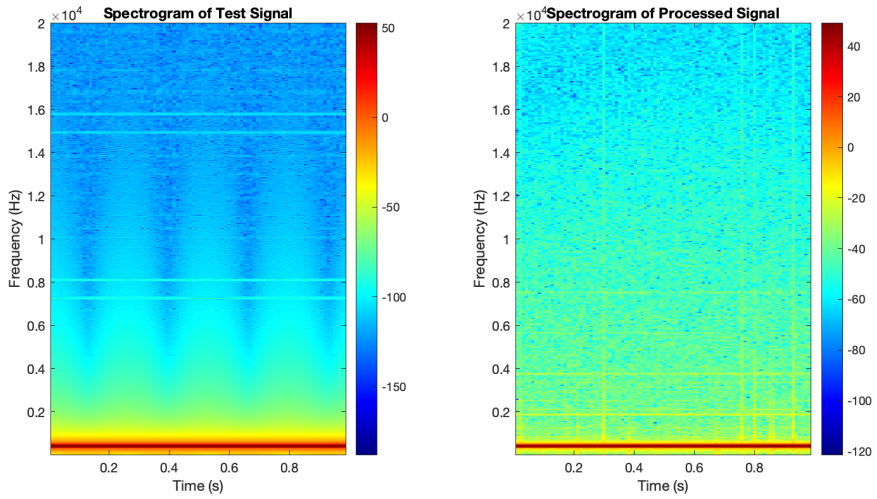


Figure C.6: spectrogram comparison algo3 sine 420 Hz Phaser more of everything

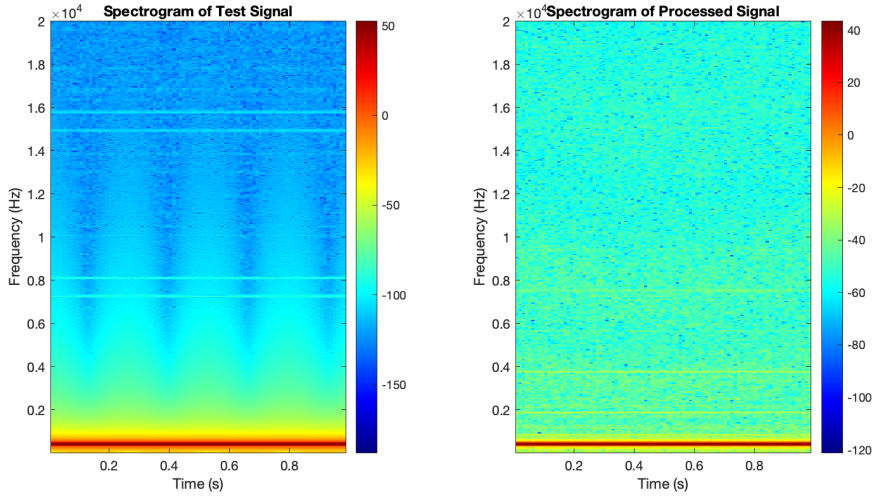


Figure C.7: spectrogram comparison algo3 sine 420 Hz Phaser fb0.05 freq0.150 mix0.25 mod amn0.6 stages8

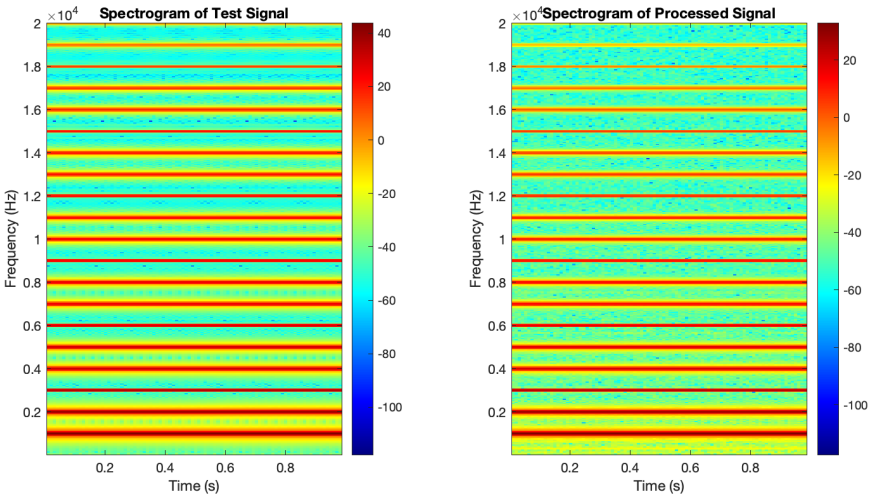


Figure C.8: spectrogram comparison algo0 saw 1 kHz

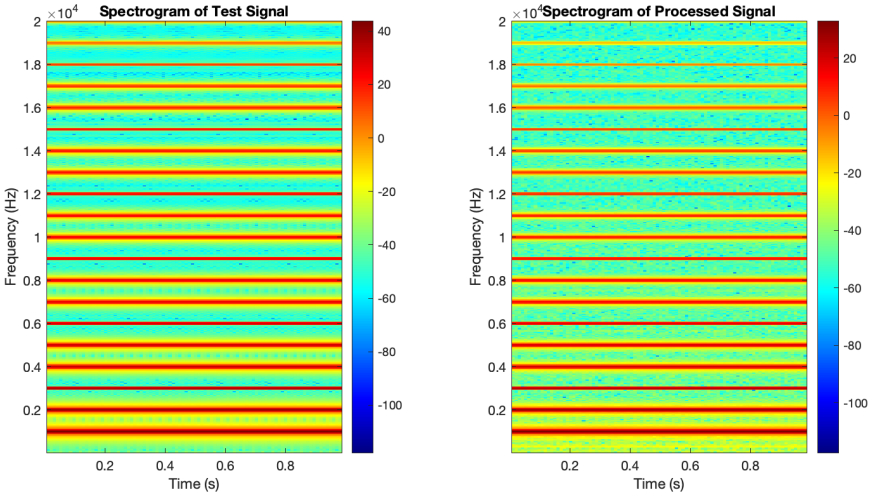


Figure C.9: spectrogram comparison algo1 saw 1 kHz

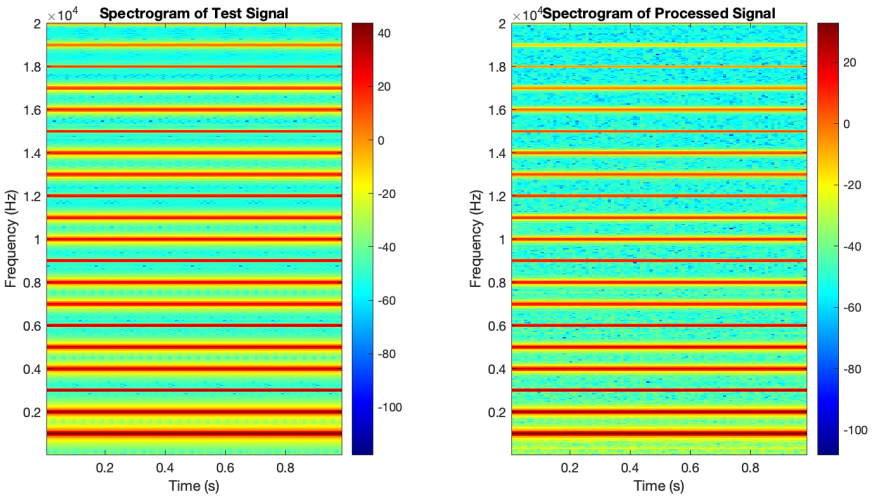


Figure C.10: spectrogram comparison algo2 saw 1 kHz Overdrive

C.2 iZotope Insight test signals

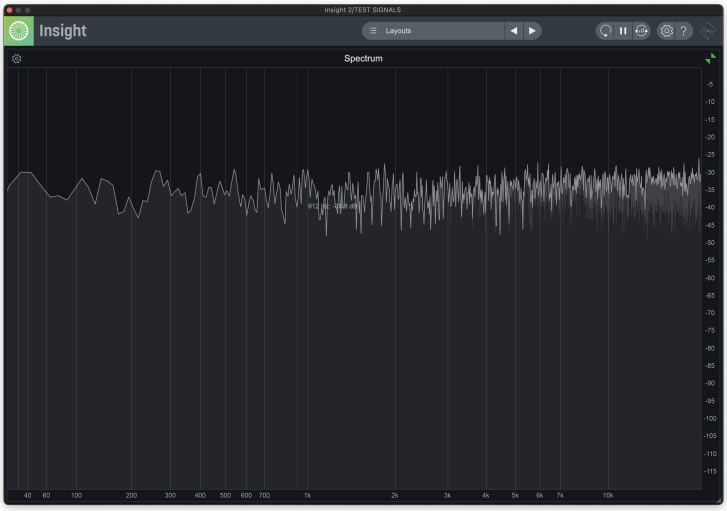


Figure C.11: Test signal white gaussian noise spectrum

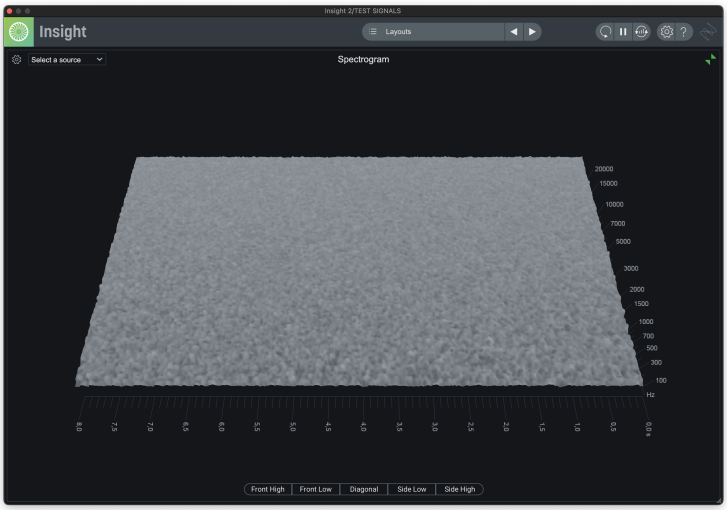


Figure C.12: Test signal white gaussion noise spectrogram

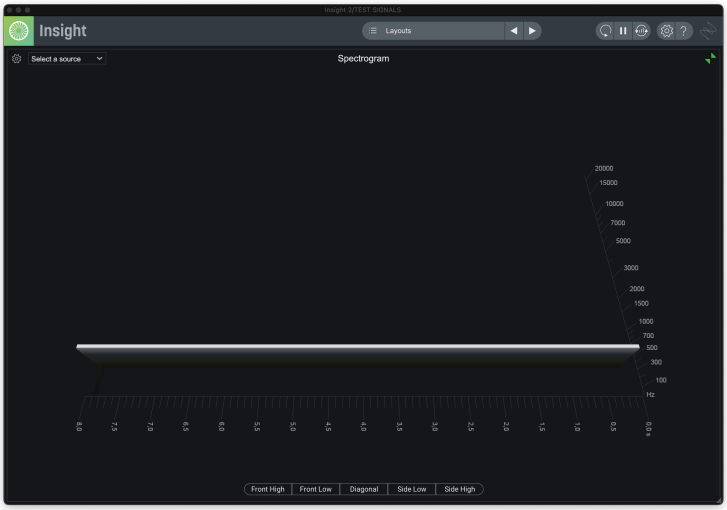


Figure C.13: Test signal sine wave 420 Hz spectrogram

Figure C.14: Test signal sawtooth was 1kHz spectrogram

C.3 iZotope Insight recorded signals

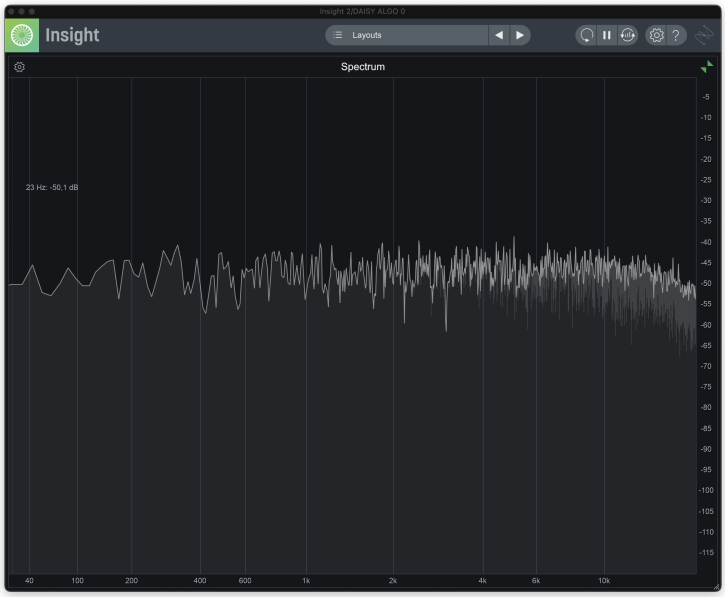


Figure C.15: Recorded signal white gaussian noise spectrogram

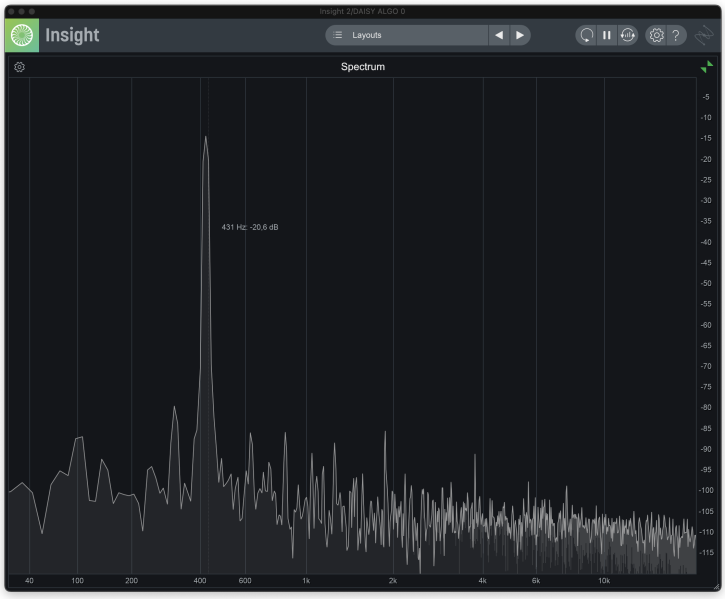


Figure C.16: Recorded signal sine 420 Hz spectrogram

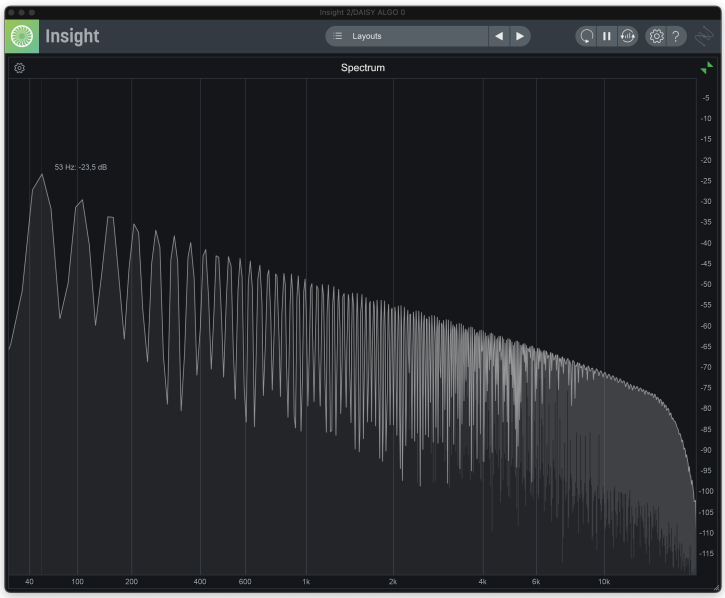


Figure C.17: Recorded signal sawtooth wave 53 Hz spectrogram

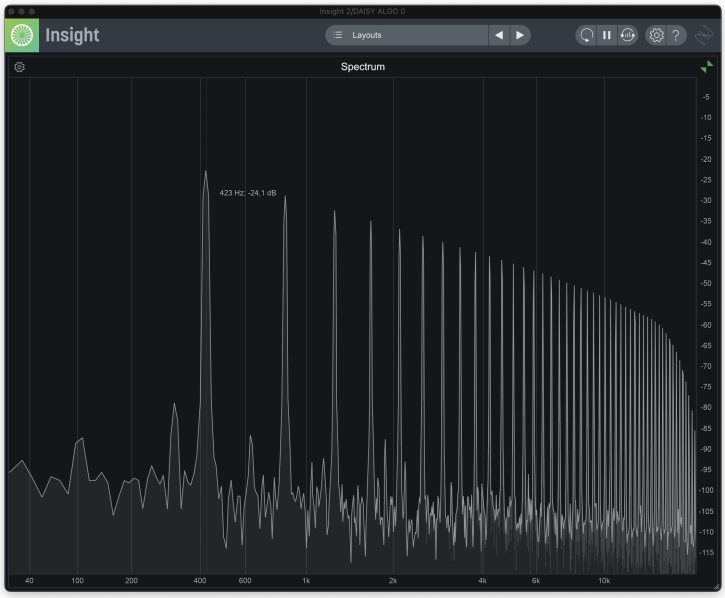


Figure C.18: Recorded signal sawtooth wave 420 Hz spectrogram

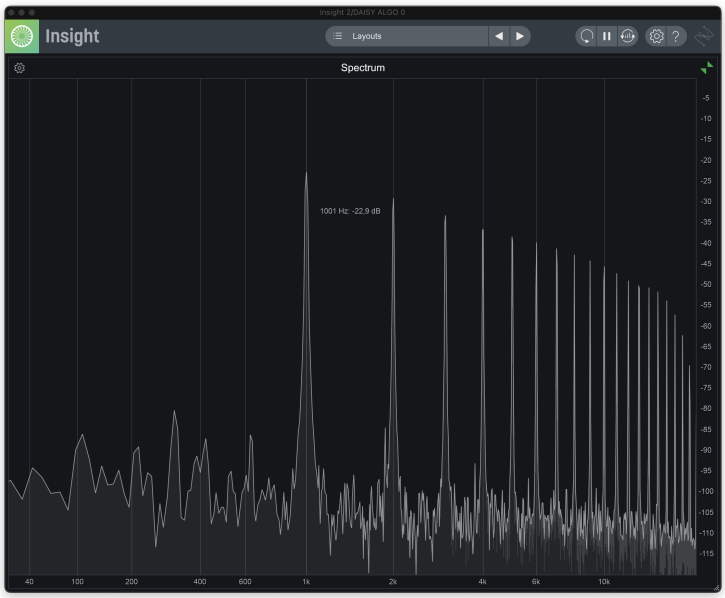


Figure C.19: Recorded signal sawtooth wave 1 kHz spectrogram

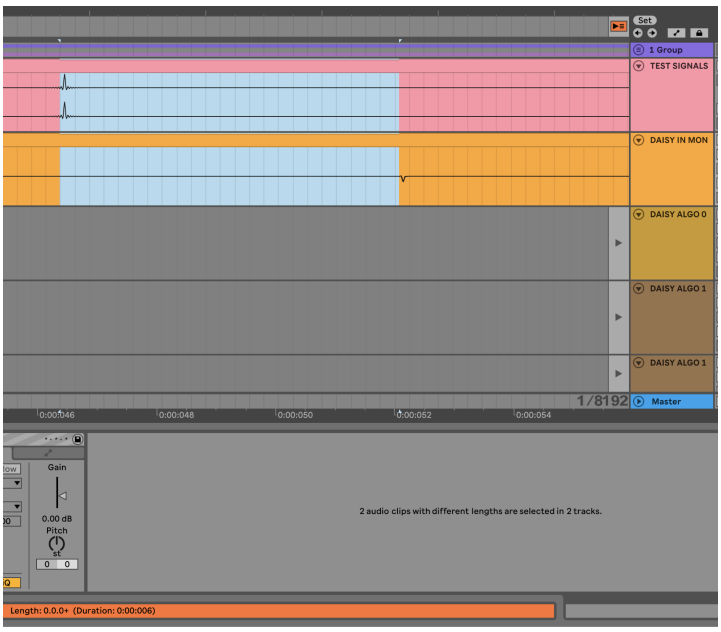


Figure C.20: Crude latency test measurement in Ableton Live

Appendix D

KiCad electrical schematic, gerber files and bill of materials

The full electricalschematic is available on our GitLab repository[1], while the bill of materials spreadsheet can be found in the external appendix by following this path: Appendix → Bill Of Materials
.

[1] https://gitlab.com/northernstructuresaudio/daisy-dub/-/blob/main/schematics/DaisyDubShematicPDF.pdf

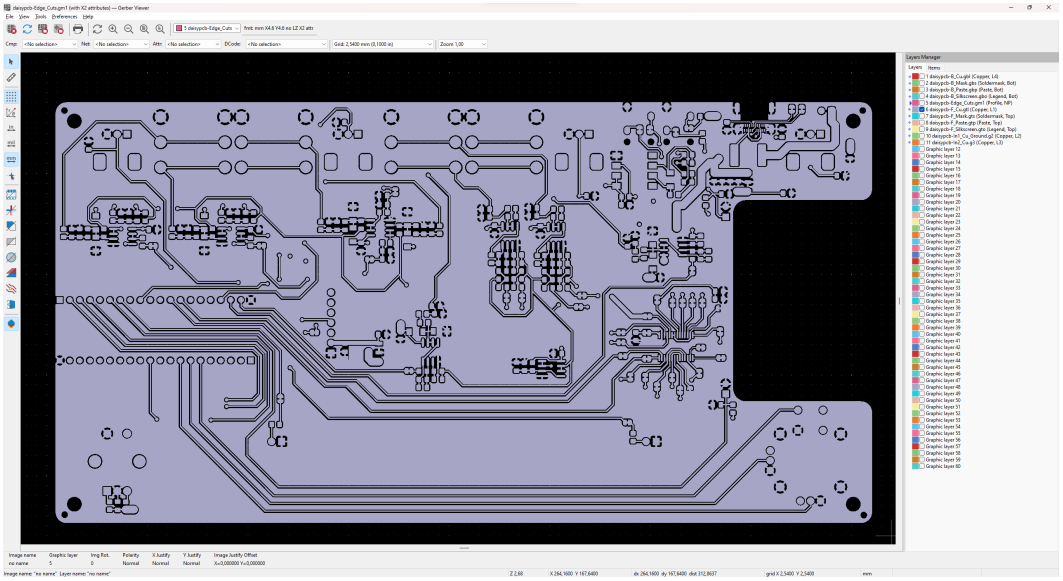


Figure D.1: Front copper layer

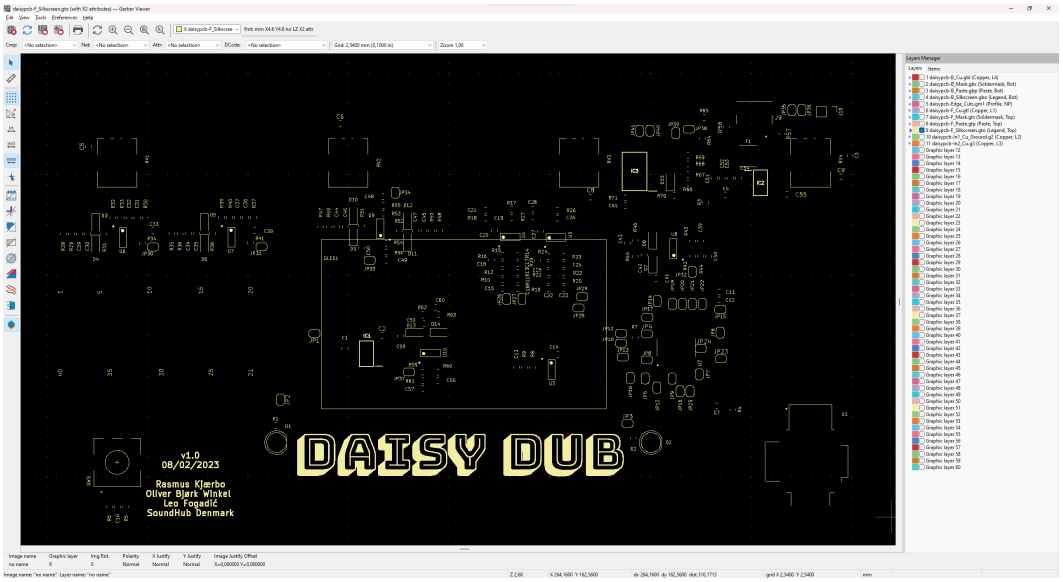


Figure D.2: Front silkscreen layer

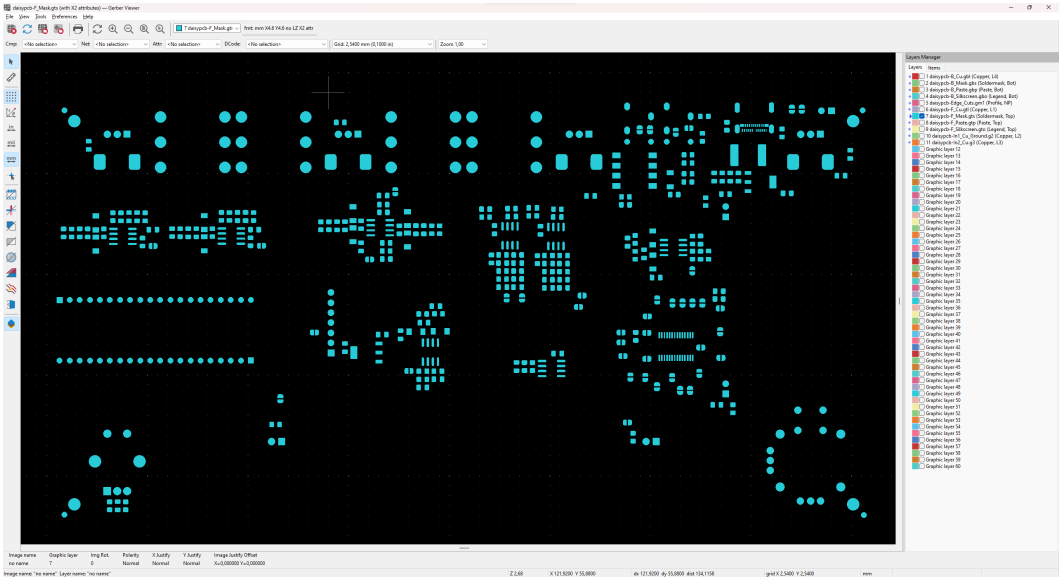


Figure D.3: Front solder mask layer

Figure D.4: Front paste mask layer

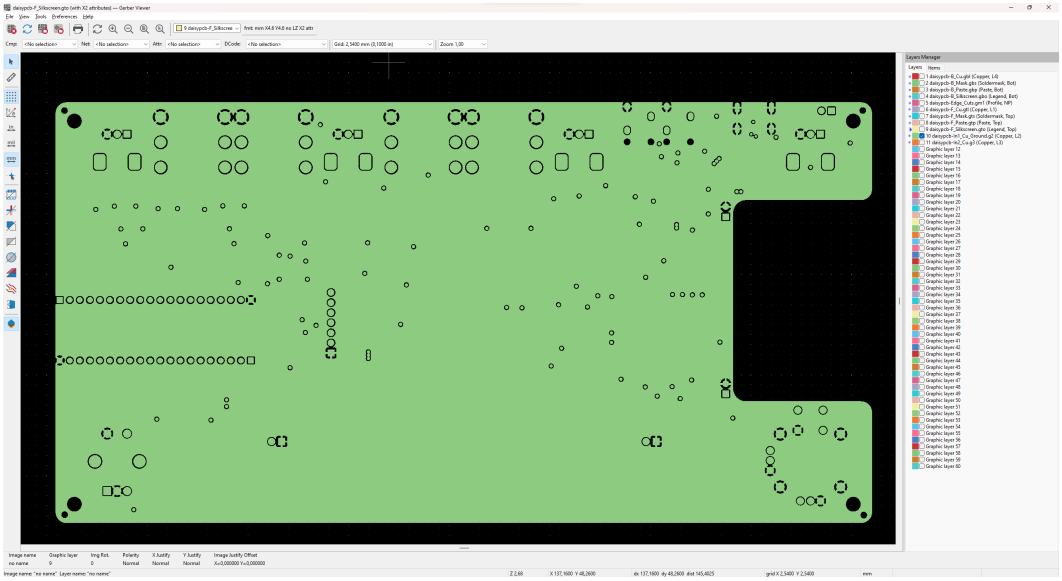


Figure D.5: First inner copper layer (ground plane)

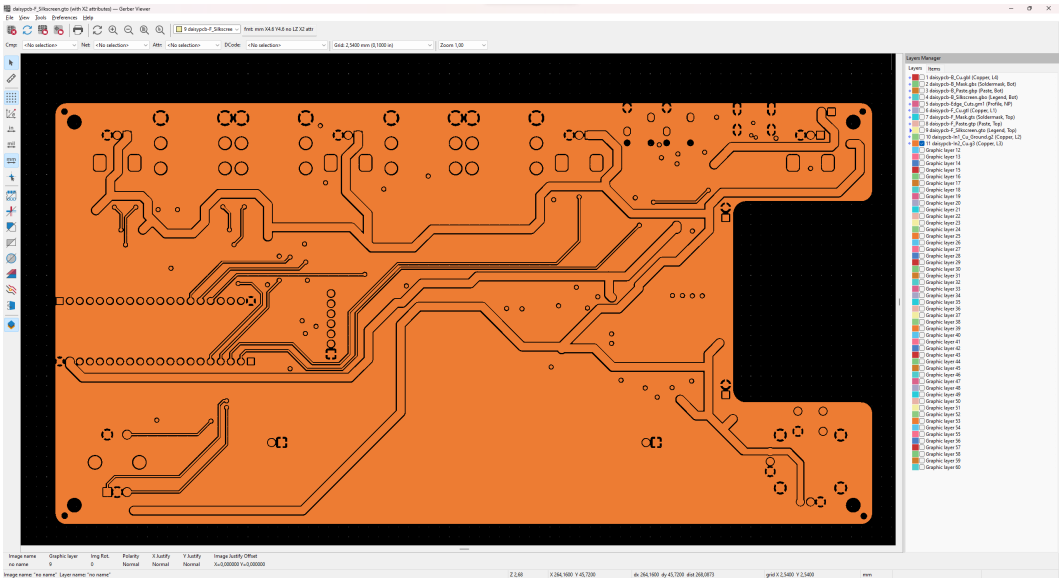


Figure D.6: Second inner copper layer

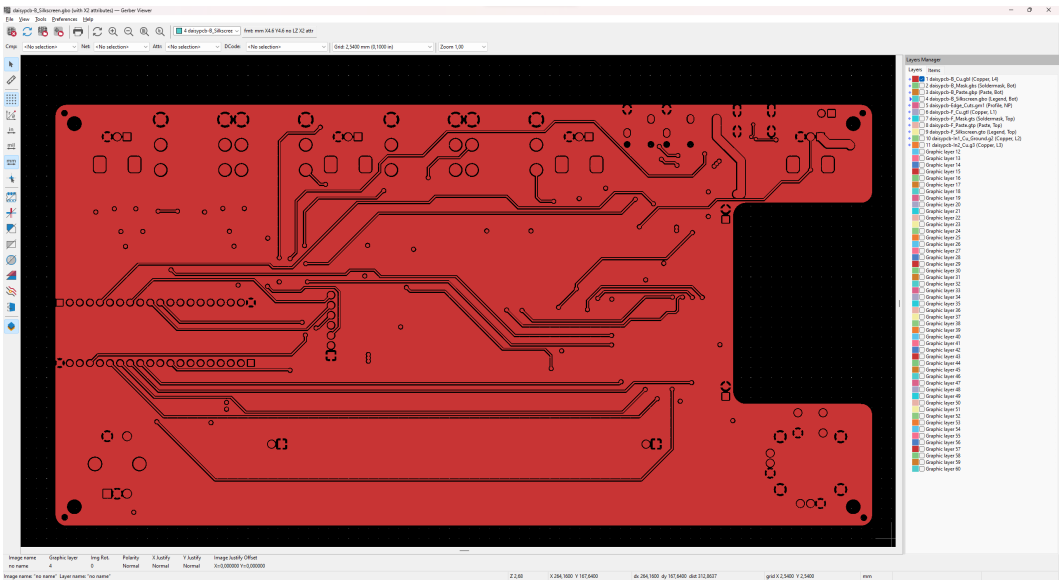


Figure D.7: Back copper layer

Figure D.8: Back silkscreen layer

Figure D.9: Back solder mask layer

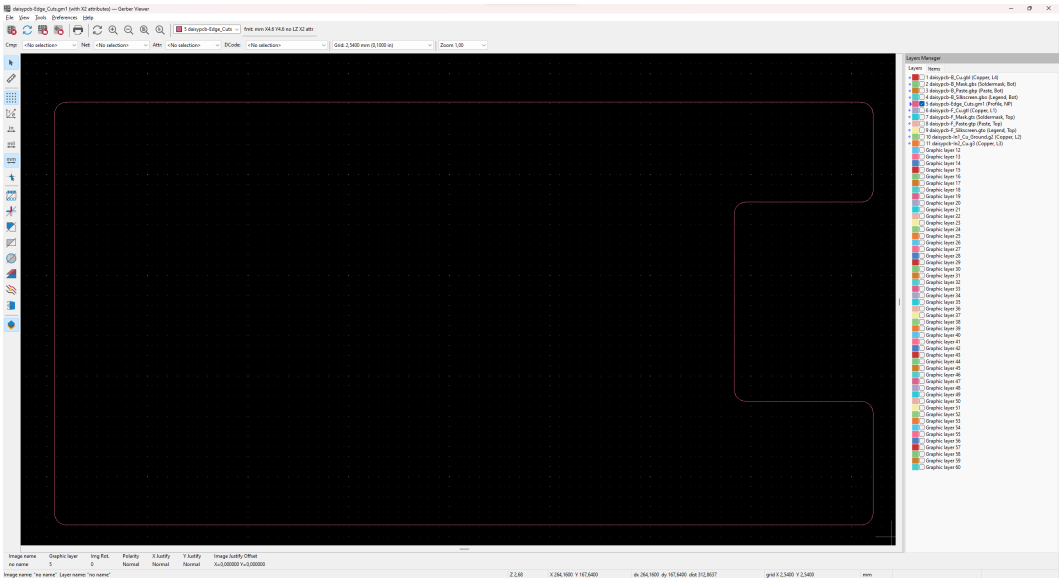


Figure D.10: Edge cuts

Appendix E

User testing documents

This appendix contains the documents used during the user testing. All of these will be available in the external appendix by following this path:
Appendix → User testing
This appendix contains the documents for the consent form, the script used by the test conductor, and the System Usability Scale template.

Appendix F

Daisy Dub DIY workshop mockup

Mock up of workshop instructions and introduction

Welcome to Daisy Dub DIY workshop!

We are so excited to have you here and to have you join us in the development of this innovative modular and updateable real-time audio effect.

As you know, Daisy Dub is still in its early stages of development and your participation in this workshop is critical to its success. By assembling the unit yourself, you are not only learning about the inner workings of this amazing piece of equipment, but you are also providing valuable feedback and insights to our team of researchers. Your participation and contributions will help shape the final product and make it the best it can be.

We are confident that with your help, Daisy Dub will become an invaluable tool for music producers and performers around the world. So, let's get started! Below, you will find a step-by-step guide to assembling your very own Daisy Dub Delay. We encourage you to take your time and be careful, but most importantly, have fun! We are here to support you every step of the way, so please don't hesitate to ask any questions or seek assistance if you need it.

Thank you again for your participation and support. We can't wait to see what you will create with your new Daisy Dub Delay!

Instruction palette

1. First, gather all the necessary components for Daisy Dub unit. This should include the PCB, OLED display, joystick, arcade buttons, encoder, knobs, LEDs, and audio jacks.
2. Begin by attaching the OLED display to the PCB using the provided connectors. Make sure to align the display correctly and securely fasten it in place.
3. Next, attach the joystick and arcade buttons to the PCB using the provided connectors. Again, make sure to align everything correctly and to securely fasten the components in place.
4. Install the encoder, knobs, and LEDs on the PCB in the designated locations. Make sure to firmly press these components into place, and to ensure that they are aligned correctly.
5. Connect the audio jacks to the PCB using the provided connectors.

6. Once all the components are in place, connect the USB Type-C cable to the PCB. This will provide power to the unit and allow it to function.
7. Finally, test the unit to ensure that everything is working properly. Adjust the settings and try out different effects to get a feel for the capabilities of Daisy Dub.
8. If any issues arise during the assembly process, don't hesitate to seek guidance from the workshop facilitators. They will be happy to help you troubleshoot and resolve any problems you may encounter.
9. Once the unit is fully assembled and functioning properly, you can begin exploring the full range of capabilities and features offered by Daisy Dub. Have fun, and don't be afraid to experiment and try out different settings and configurations.