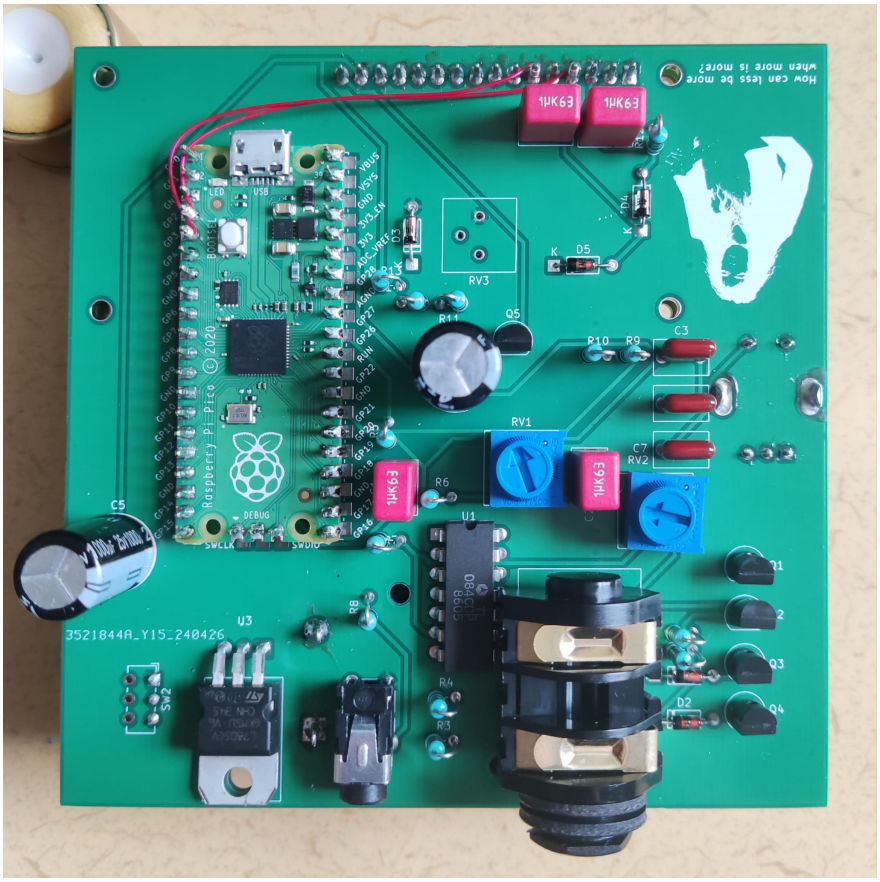
CHALMERS
UNIVERSITY OF TECHNOLOGY



Signal processing unit for bone conduction microphone and speaker

An implementation of delay, pitch shifting and echo cancellation

Master's thesis in Embedded electronic system design

Daniel Eliasson
Lucien Stauffer-Kee

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2024
www.chalmers.se

Master's thesis 2024

Signal processing unit for bone conduction microphone and speaker

An implementation of delay, pitch shifting and echo cancellation

Daniel Eliasson
Lucien Stauffer-Kee

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Division of Signal processing and Biomedical engineering
Unit of Biomedical Signals and Systems
Chalmers University of Technology
Gothenburg, Sweden 2024

Signal processing unit for bone conduction microphone and speaker
An implementation of delay, pitch shifting and echo cancellation
Daniel Eliasson
Lucien Stauffer-Kee

Supervisor: Karl-Johan Fredén Jansson, Signal processing and Biomedical engineering
Examiner: Sabine Reinfeldt, Signal processing and Biomedical engineering

Master's Thesis 2024
Department of Electrical Engineering
Division of Signal processing and Biomedical engineering
Unit of Biomedical Signals and Systems
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Printed circuit board of an electronic fluency device.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2024

Signal processing unit for bone conduction microphone and speaker
An implementation of delay, pitch shifting and echo cancellation
Daniel Eliasson & Lucien Stauffer-Kee
Department of Electrical engineering
Chalmers University of Technology

Abstract

This paper introduces the development of a signal processing unit for bone conduction microphone and speaker generating real-time adjustable delayed auditory feedback (DAF) and frequency altered feedback (FAF). These types of feedbacks are known to effectively increase fluency in people who stutter. The incidence of stuttering is about 1% of the world population. The effects of DAF have been noted since the 1950s. The user's voice is captured by a microphone, processed, and then relayed back through a speaker. DAF introduces a delay of 50 to 200 milliseconds, while FAF alters the pitch by a quarter to a full octave. A notable challenge for bone conduction devices is the potential for strong feedback paths between speaker and microphone, which can result in echo or oscillation.

The signal processing algorithms, including DAF, FAF, and echo cancellation, were developed and tested using MATLAB®. Additionally, an analog chain was constructed and evaluated on a breadboard, featuring variable amplification and power amplifier to directly drive a passive speaker. To achieve a standalone device, the algorithms were ported to a microcontroller, which was further enhanced with a user-friendly interface, including a rotary encoder and LCD, allowing adjustments of the algorithms without programming expertise. The entire system was then integrated onto a custom-designed printed circuit board (PCB), combining both analog and digital circuitry.

The MATLAB® script successfully implements all algorithms; DAF, FAF and echo cancellation. It can be used either with sound files like wav or mp3, or through the use of a audio interface the MATLAB® script can be used in real-time for live application such as a test with a real person.

The hardware implemented design on PCB has a working and tested DAF, and a untested implementation of echo-cancellation. Due to limitation in floating point performance of the microcontroller a pitch shifting algorithm remains incomplete. The hardware device has sufficient audio quality with a total harmonic distortion of about 3% adhering to IEC 60645-1. This work lays the groundwork for future enhancements, particularly in refining the pitch-shifting capability.

Keywords: Signal processing, stuttering, DAF, FAF, echo-cancellation, Raspberry Pi Pico, audio.

Acknowledgements

We would like to thank our kind supervisor Karl-Johan Fredén Jansson who have provided facility, components and expertise. The characterization of the device would not have been possible to such a degree without you.

Daniel Eliasson & Lucien Stauffer-Kee, Gothenburg, June 2024

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

AAF	Altered Auditory Feedback
ADC	Analog to Digital Converter
DAC	Digital to Analog converter
DAF	Delayed Auditory Feedback
FAF	Frequency Altered Feedback
FIR	Finite Impulse Response
GPIO	General Purpose Input Output
LCD	Liquid Crystal Display
LED	Light Emitting Diode
LMS	Least Mean Square

Contents

1
Introduction

1.1 Background

Devices based on skin microphones are currently researched to be used as bone conductive stethoscope for hearing aids among other application. A similar device to hearing aids can be used for altered auditory feedback (AAF). Today an approximate one percent of our population are experiencing experiencing some amount of stuttering [1]. Both delayed auditory feedback (DAF) and frequency altered feedback (FAF) have been already identified to inhibit stutter to a large degree. Research in inhibiting and reducing stuttering have yielded between 60-100% stutter inhibition by altering the auditory feedback [2]. Conclusion of these findings suggest that the means of altering the audio, primarily DAF and FAF, could be used in clinical application.

Research in bone conduction microphone and transducers for use within hearing aids have recently seen great progress. One of the more elusive problems of objectively measuring the audibility of bone conductive devices have been solved by using skin microphones [3]. It is therefore possible to verify the operation of a bone conduction device for audio playback. Bone conduction devices have also been successfully used as implants with no serious adverse effects [4], it is therefore of great interest for making better, smaller and seemingly invisible hearing aids among other things. Previous study have determined the optimal placement for bone conductive microphone for speech intelligibility which have been understood to provide better audio clarity in noisy environments compared to traditional air microphones [5].

Because bone conduction devices has seen very recent advancements, it has become ever more interesting to apply the technology to new fields such as stutter inhibition. Since there's many system similarities between hearing aids and the devices proposed to inhibit stutter, there's also great potential for near-term product development and thus patient life quality improvement.

1.2 Aim

For the purpose of continuing the research in stutter inhibition and evaluate if skin microphones are suitable, a suitable signal processing unit needs to be constructed. Since the best combination of settings for inhibiting stutter can vary between individuals among other things, a large degree of customization is needed.

1.3 Limitations

Despite the testing of the device on the participants of the project, no clinical studies will be conducted during the course of the master thesis.
Although many of the functions will be adjustable, it will not be investigated how they perform while being adjusted in real-time. We assume a steady state of these adjustable parameters and any artifacts created during parameter change will be disregarded.

1.4 Specification of the issue being investigated

A complete signal processing system needs to be designed. The system is expected to receive an analog signal and using a combination of analog and digital circuitry, apply an adjustable delay and a pitch shift. In addition the microphone signal may be enhanced with echo-cancellation as this is a issue more prevalent to bone conducting speakers and microphones. When the signal processing is complete the system should output the audio as an analog signal to a speaker.

2

Theory

In this chapter we will go through some of the more niche knowledge applicable to this report. It is however assumed that knowledge of electronics as well as signals and signal processing at a basic level is already known. Therefore no discussion about the differences in time-domain and frequency-domain or how one signal may be transformed from one to the other domain. It is also assumed knowledge of the difference between continuous signals and discrete signals, both in time and amplitude.

2.1 Digital delay line

Digital delay line is a discrete element in signal processing which allows a signal to be delayed by certain number of samples, see figure 2.1. The delay line may be realized in various ways, one of those is a circular buffer.

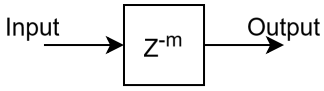
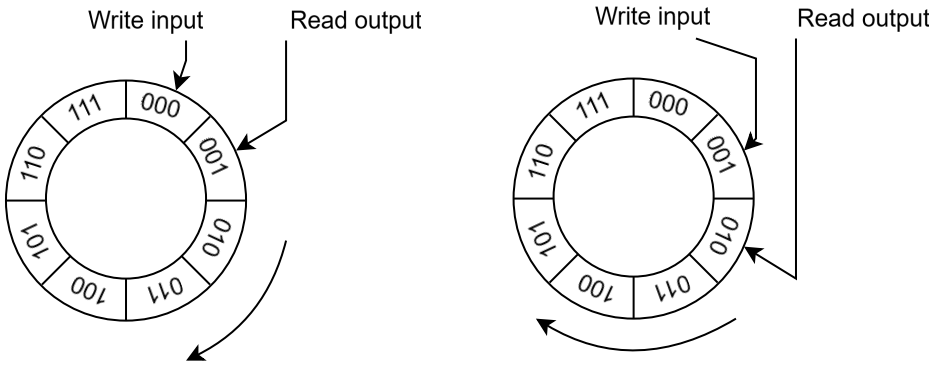


Figure 2.1: Delay line with signal delay m

The circular buffer uses a fixed length of memory and through changing the position of read and write you can decide when to read back an old value, see figure 2.2. The delay is not fixed to the buffer size instead a change in the relative position of the read output to the write input [6] will have different delay. This is because it effectively creates a part of the buffer which has been written but has not been outputted, in essence delaying it from being output. The largest spacing that can exist is the same as the size of the buffer. A delay larger than the buffer size will otherwise overwrite input-data before ever being read out.



(a) buffer at position 000 **(b)** buffer at position 001

Figure 2.2: The working principle of circular buffer

2.2 Fast Fourier Transform

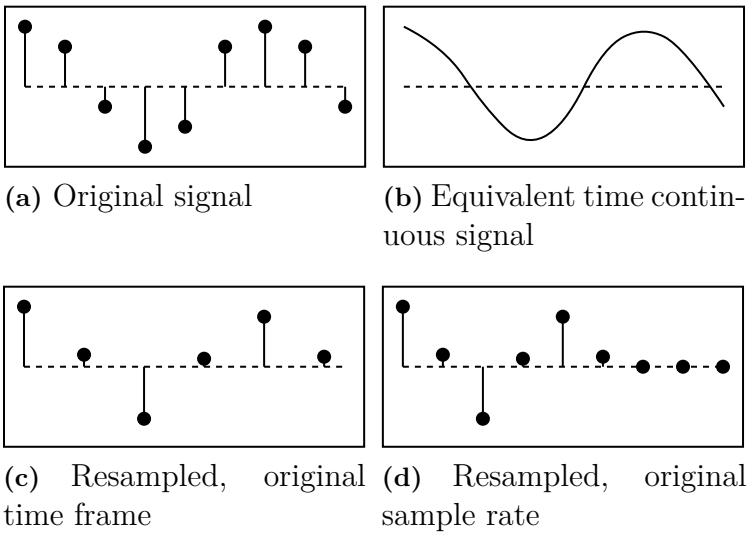
The Fast Fourier Transform (FFT) is a set of algorithms which aims to transform a signal from the time domain in to the frequency domain. In most applications it is equivalent to discrete Fourier Transform (DFT) with the big advantage of being faster. FFT is not just one algorithm, instead it is a name entitling all algorithms equalling DFT in function but being computationally faster. The most famous and widely used varient is the Cooley–Tukey algorithm and is the one we will discuss. The Cooley–Tukey algorithm aims to break down the computational task in to smaller ones. In practice this is done by breaking a N long DFT to two lighter DFTs of length N/2. Through the use of some lighter mathematical manipulation the results will become the same. This being recursive means that in the ideal case DFT with the length of a 2 power, it only needs to calculate the trivial DFT of length 1 and the rest is lighter manipulation [7]. Whenever a move to the frequency domain is done some consideration about the window function should be done. The default rectangular window may give substantial artifact, instead it is common to use a Hanning window.

2.3 Pitch shifting

The process of pitch shifting aims to alter the signal in frequency domain to shift it up or down. One of the core principles is to alter the playback speed. However altering the playback speed is not possible in real-time applications as having playback faster/slower than recording speed means either running out of samples to play or an ever increasing remainder to play neither of which is desirable. There are many solutions to overcome this issue, one of these is the phase vocoder.
The two fundamental parts of the phase vocoder is resampling and phase realignment. To achieve a playback speed different from the recording speed resampling is used, which effectively changes the pitch. The other part, phase realignment makes sure to keep phase coherence, preventing audible artifacts [8].

2.3.1 Resampling

Resampling is the process of changing the sample amount. This means that the signals still occupies the same window of time but now has more or less amount of samples, see figure 2.3 for a visual representation. Playback after resampling with the original sample rate changes the time window of the signal instead. In turn changing the size of the time frame a signal occurs means it plays faster or slower in time, in other words you have shifted the signal in frequency domain [9].



(a) Original signal

(b) Equivalent time continuous signal

(c) Resampled, original time frame

(d) Resampled, original sample rate

Figure 2.3: Resampling of a signal

When resampling it is important to know the general frequency spectrum of the signal as the signal may not be supported by the new sample rate resulting in aliasing artifacts.
In practice it is separate in to two different methods, one for increasing sample rate and one for decreasing. To increase sample rate insertion of zeros in between samples is done, which is followed with a low-pass filter designed according to the maximum supported frequency allowed by the original sample rate. This low pass

filter is sometimes called a reconstruction filter. To decrease the sample rate is done by first making sure there is no frequency content not supported by the new sample rate. Which is done by filtering the signal with a low-pass filter, sometimes called an anti-aliasing filter. After this is done, exclusion of samples at an even spacing trims away to desired sample rate. These two methods by themselves will only allow changes to the sample rate as a multiple or division by an integer factor. To change the sample rate to any rational number both methods needs to be done in succession, first increase the sample rate and later decrease it [10].

2.3.2 Time framing

In a real time application an extra step is needed to be able to perform multi sample processing. One function fulfilling this is to store multiple samples in a buffer representing a frame of time, see figure 2.4.

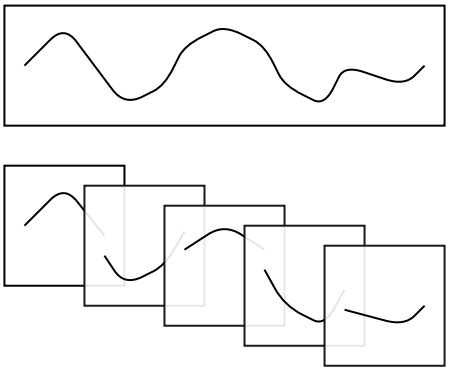
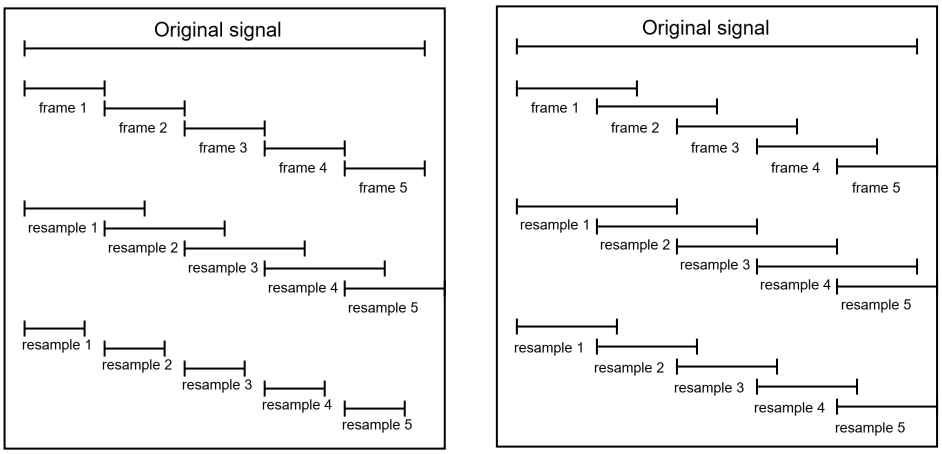


Figure 2.4: Signal divided in to frames

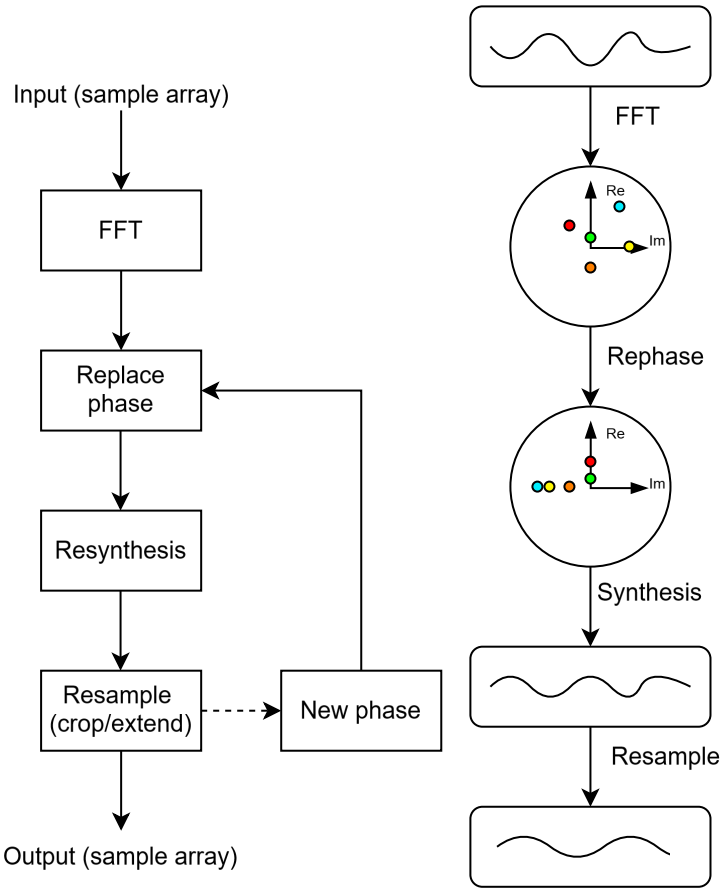
In the context of resampling each frame to achieve pitch shift there is two substantial issues, visualized in figure 2.5. When each frame is stretched out there is an overlap, in which case there is two signals that needs to be combined. In the second case there is a far worse problem. When the frame is shrunk it results in spaces of time in which there is no signal to play.



(a) Segmenting without overlap **(b)** Segmenting with overlap

Figure 2.5: Effect of resampling on time frame signal

A solution to the issue of having no signal to play it to create the initial frames with an overlap. Thus there's a margin of time which the frames can be shrunk. To handle overlapping time frames a window function may be applied to each frame and then it is possible to simply add them together [11].
One issue will however remain, phase misalignment, the solution of which have given the phase vocoder algorithm its name. Through the use of FFT, each small time frame can be transformed into the frequency domain. Samples will then represents magnitude and phase at a specific frequency instead of an amplitude at a given time. This leads to the opportunity to manipulate the phase while keeping the frequency content the same. Inverse FFT can then be applied to resynthesise each time frame back to time domain [12]. The end algorithm is summarized in figure 2.6.



(a) block diagram **(b)** signal diagram

Figure 2.6: Block diagram and the corresponding changes in signal it makes for the pitch shifting algorithm

2.4 Echo cancellation

Echo cancellation is the process of removing echoes due to an unwanted feedback path by first estimating the feedback path, secondly reproduce the echoed signal using the estimated feedback path to then thirdly, remove it from the signal, see figure 2.7.

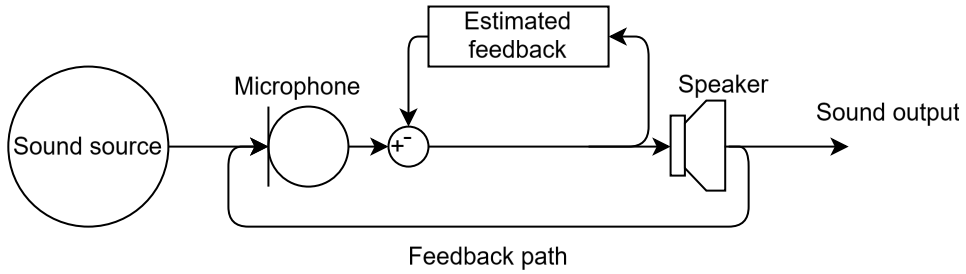


Figure 2.7: Block diagram of echo cancellation

The estimation of the feedback path is commonly done using a finite impulse response (FIR) filter. This is an adequate solution for most naturally occurring feedback as it encompass both signal delay and frequency alteration such as low-pass or high-pass. It will however not work for signal deterioration like clipping or distortion as the FIR filter cannot approximate these.
For the echo cancellation to work, there needs to be an accurate estimation of the feedback path and in order to do so, the filter requires to be trained. To train the FIR filter it's possible to play a known sound from the speaker and use the microphone to listen to it. For better performance, it's common to use a sound with a wide frequency spectra and so it is typical to play white noise. Then, comparing the result from the real feedback path that's captured through the microphone to the estimated result, the difference is the error and is called e. By using the least mean square (LMS) algorithm the filter can be incremented for as low as possible error e [13]. Ideally this will converge giving you a good approximation.

3
Methodology

The aim of this project is to develop a tool to be used for investigations of delayed auditory feedback for rehabilitation of patients who stutter. In more detail, the goal is to develop a signal processing prototype with the functions in figure 3.1. The device will have an analog and a digital section. The analog section main purpose is to amplify the signal. Meanwhile the digital section contains the different audio alterations like delay and pitch shifting.

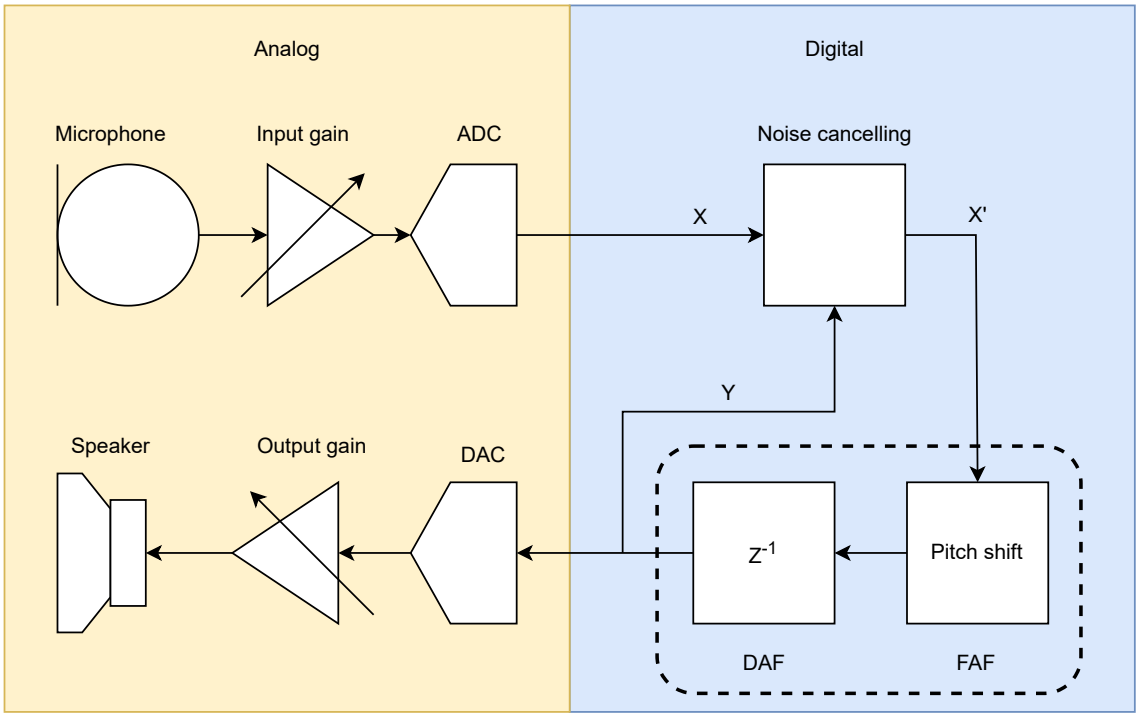


Figure 3.1: Block diagram

We began working in MATLAB® creating scripts of the different algorithms. The scripts were first tested on audio clips confirming each algorithms function. These scripts were then combined into a complete model able to process audio clips or real-time audio through an audio interface.
The learning's from the MATLAB® experimenting was then used to create a stand alone unit based around the Raspberry Pi Pico microcontroller. First assembled and tested on breadboard where alterations on hardware could easily be done. Analog section was partly simulated in LTSpice and then tested with components on breadboard. Features not implemented in the MATLAB® scripts such as an interface to

change settings was also added. We then moved to a printed circuit board (PCB) when we achieved a stable hardware solution. The PCB was designed in KiCad. The firmware running on the Raspberry Pi Pico was written in C using the Raspberry Pi Pico version of Visual Studio Code. The algorithms were initially written heavily based on previous work in MATLAB®. A lot of work was however put in to optimizing these algorithms for the specific platform. See figure 3.2 for a block representation of above.

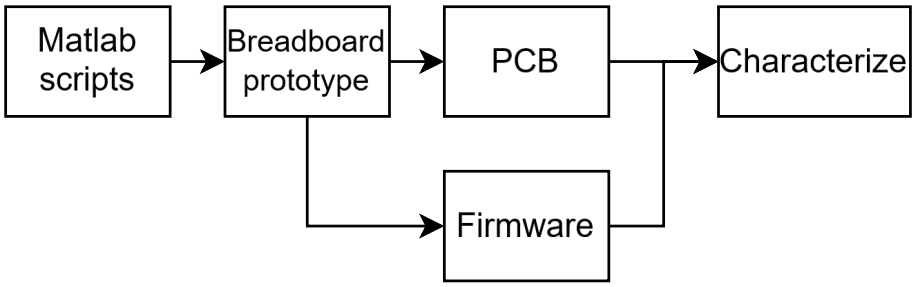
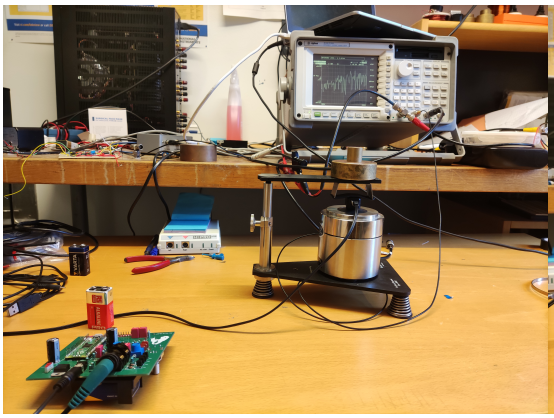


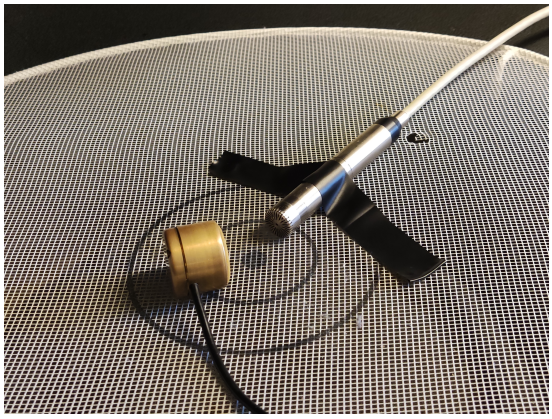
Figure 3.2: Overview of task schedule

Once the standalone unit was finished work on measuring performance commenced. To characterize the device a set of measurements have been taken using a dynamic signal analyzer connected by a multitude of methods. Directly connected, meaning electrically stimulating the input of the device and simultaneously measuring the direct output voltage. We've also measured the device connected to a bone-conducting microphone and bone vibrator to characterize the behavior in a close to intended application. The microphone is being stimulated by a speaker inside the anechoic test chamber BK 4222. The bone vibrators force is being measured through the artificial mastoid BK 4930, see figure 3.3. All tests have been done with maximum input and output gain. The measured data collected includes frequency response, bone vibrator force, total harmonic distortion, noise and input saturation point.
The bone vibrators tested were the Radioear B71w, B81 and B250.



(a) overview

(b) artificial mastoid



(c) microphone in anechoic chamber setup

Figure 3.3: Measurement setup

4

Results

We designed two different solutions. Firstly, a software unit consisting of a MATLAB® prototype running algorithms and using an audio interface for input and output. Secondly, a custom and completely standalone hardware unit.

4.1 Software unit

The software unit consists of MATLAB® scripts which runs on a computer. The script may either run using previously recorded data in the form of audio files or using real time audio. Recorded data is convenient to verify performance of the FAF. In order to run the script with real time audio, an audio interface is to be used, see figure 4.1. To be able to drive a speaker directly from the audio interface we used an additional power amplifier, the same as described later in 4.2.1. For the FAF, two different methods were implemented, pitch-shifting and frequency shifting.

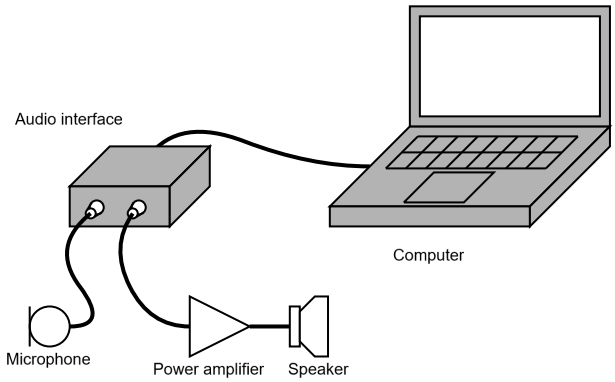


Figure 4.1: Software unit with real-time sampling of audio

4.1.1 Pitch-shifting

In order to produce FAF we need to alter the sampled audio in frequency domain. One way we did this was to implement a pitch-shifter using a phase vocoder as described in 2.3. The implementation in MATLAB® follows the phase vocoder without any major alterations. The only notable difference is how we have chosen to implement resampling. For improved computational efficiency a simplification has been done, by instead to use linear interpolation between the two closest samples for resampling. This will introduce some unwanted harmonics but they are of fairly low

amplitude, see fig 4.2. The interpolation method is computationally the same effort independent to the resampling ratio, which is not true for conventional resampling.

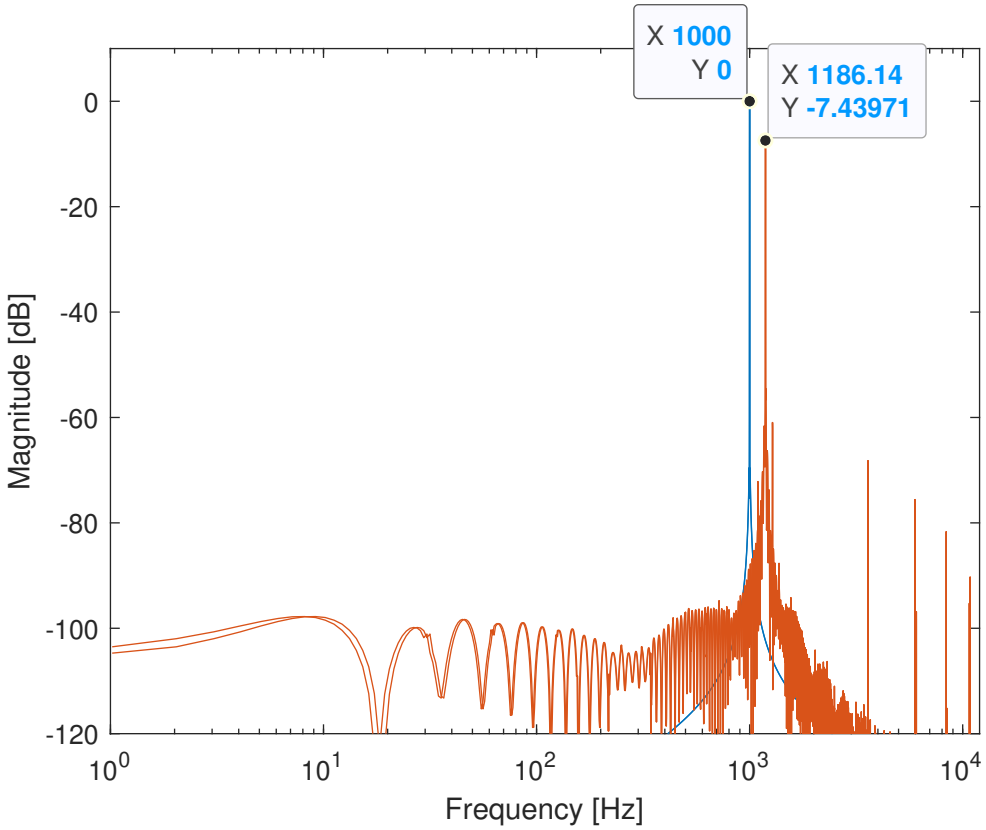


Figure 4.2: Pitch shifting using interpolation resampling of 1.2 times speed increase. Original signal is a sinewave at 1 kHz and is drawn blue, altered signal is orange

4.1.2 Frequency shifting

Frequency shifting can be used to create a Frequency Altered Audio. It is inherently different to pitch shifting. The principle we used was to transform the audio in to frequency domain using FFT. Then the values of amplitude of frequency bins were shifted up or down a predefined integer. Finally the frequency spectrum is re-synthesized to time-domain. The resulting sound files differ in timber rather than sounding pitch-shifted because the frequency spectrum is translated sideways rather than linearly compressed or expanded. This means that harmonics are not multiples of the fundamental frequency anymore.
Specifically the algorithm divides the signal into frames. FFT is calculated for each frame. In the present rendition, a frame is 16000 samples and the sampling frequency is 16 kHz. The frequency spectrum of each frame is then shifted. Figure 4.3 illustrates the principle with a shift of the frequency spectrum of a single frame 300 bins higher. Then an inverse FFT of the modified spectrum is calculated to have an audio signal of the frame. See figure 4.4 for a block diagram describing this process. This solution is inherently less complex than the pitch shifter as it foregoes

the resampling step.

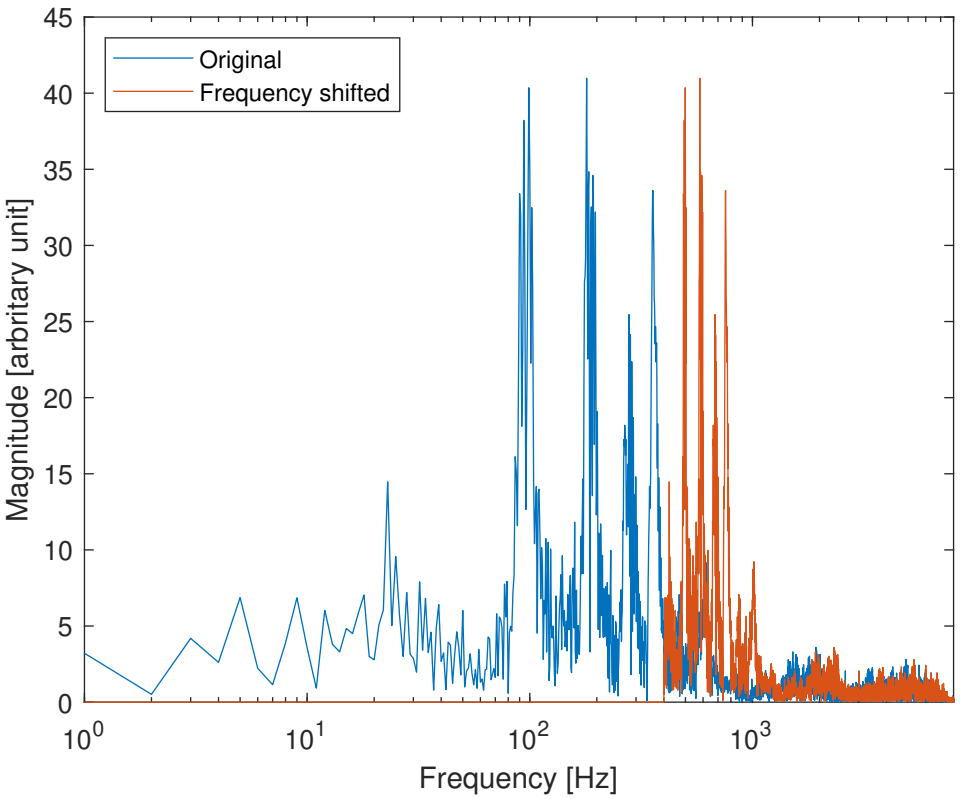


Figure 4.3: Frequency spectra of a frame before and after a shift of +300 bins

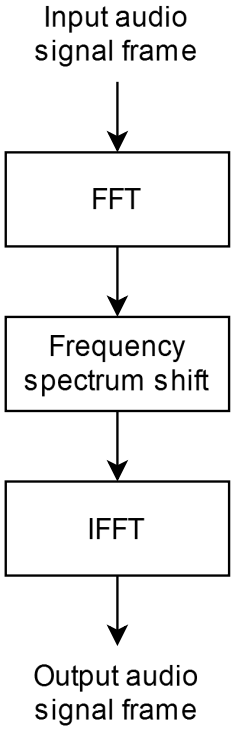


Figure 4.4: Block diagram of the frequency shift signal path

4.2 Hardware unit

The hardware unit was constructed in two stages, firstly a breadboard prototype, see figure 4.5 and then, final PCB version, see figure 4.6. A system overview in the form of a block diagram can be seen in figure 4.7. The signal path starts by sending the audio through a variable gain amplifier. This allows the user to set the input level in order to utilize most of the dynamic range and thus also resolution of the ADC without clipping. The ADC converts the audio signal from analog to digital in order to further process it in the digital domain by a processor. After processing it is converted back in to analog form by a DAC. A variable gain amplifier allows the user to set the output level to a comfortable level. Finally a output power amplifier allows driving a passive speaker directly from the device.

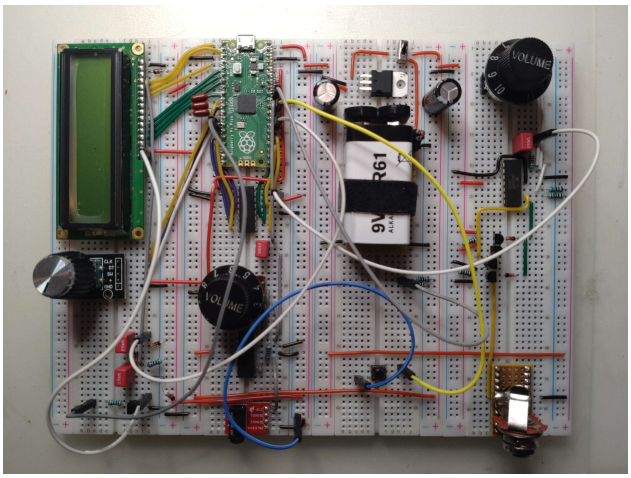
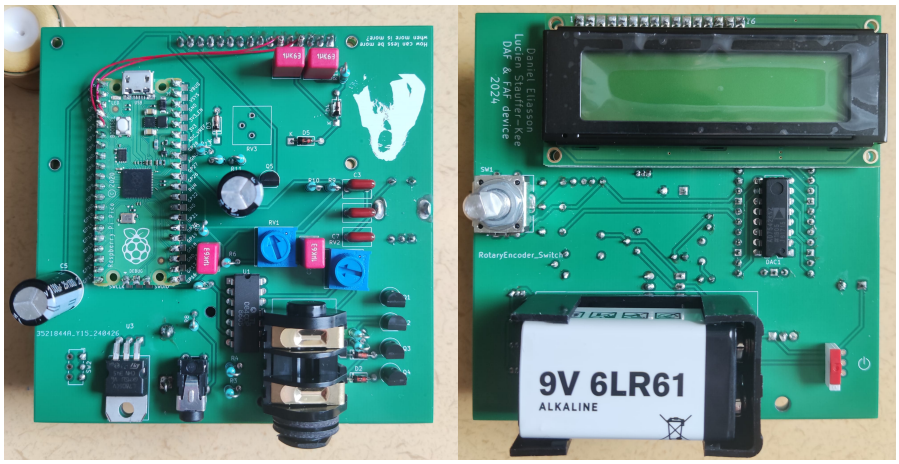


Figure 4.5: Breadboard version of the hardware unit



(a) top **(b)** bottom

Figure 4.6: PCB version of the hardware unit

4.2.1 Circuit design

In more detail the variable gain amplifier consists of an opamp, a resistor and a potentiometer, see figure 4.8. The circuit is a common inverting amplifier with the feedback resistor replaced with a potentiometer thus allowing a variable gain. As the potentiometer is in one of its extreme position the feedback will assume a short and the output will be zero. In the other extreme position it will assume a value approximately 10 times larger than the fixed resistor resulting in a good amount of gain. The fixed resistor could be discarded if we instead feed the input signal through the unconnected pin on the potentiometer, this would however result in a very large gain swing from -infinite to +infinite, making the potentiometer very sensitive.

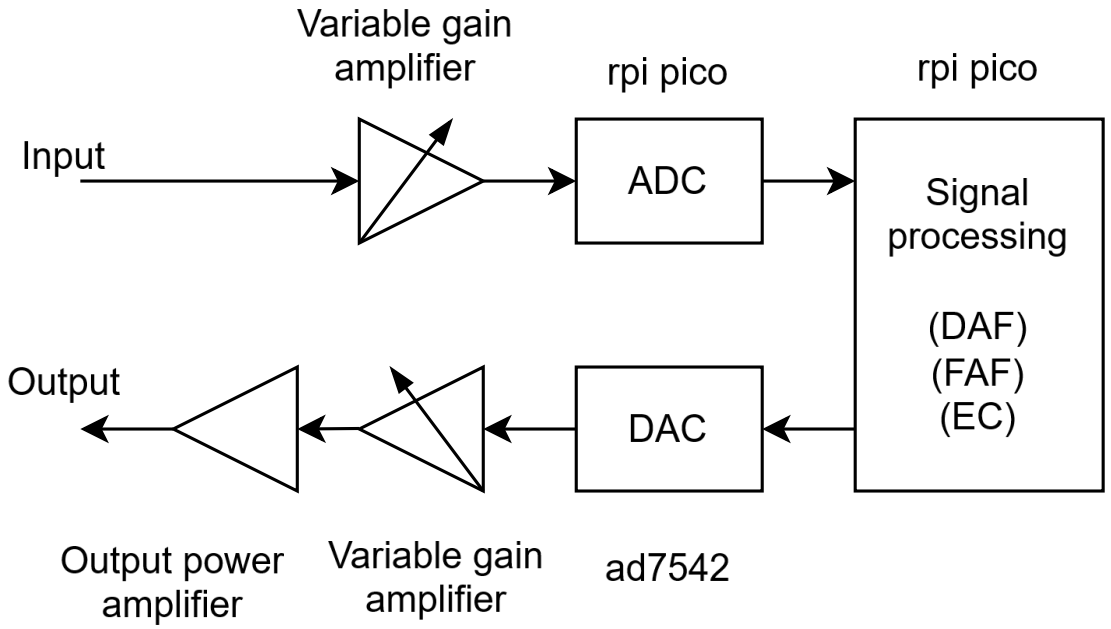


Figure 4.7: Block diagram of the signal path

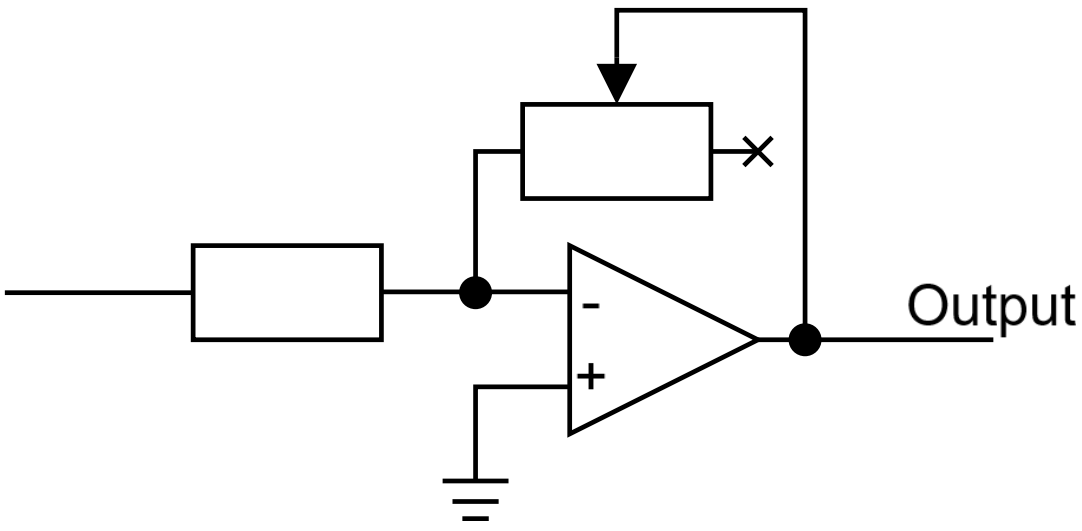


Figure 4.8: Schematic view of variable gain amplifier

The power amplifier is a op-amp driven class AB bipolar transistor amplifier. When using a single transistor pair we could observe severe current saturation on the op-amp preventing a good output level. Therefore, to reduce current draw from the op-amp and in effect increase output level we've chosen transistors in a Darlington configuration, thus increasing amplification factor of the transistor stage.

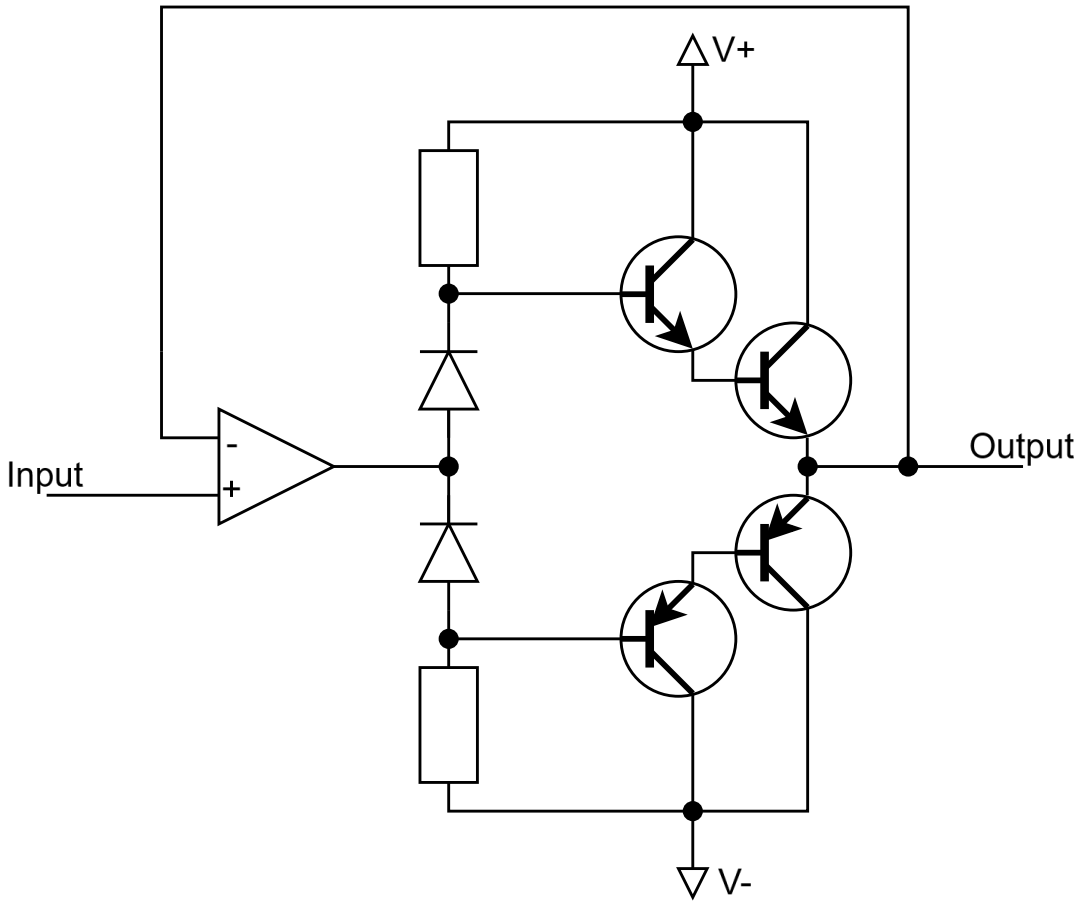


Figure 4.9: Schematic view of output power amplifier

To control various aspects of the signal processing we needed some visual and tactile interface to the device. We chose to add a character LCD and a rotary encoder in order to implement a simple menu system in which the user can change certain parameters. The raspberry pi pico has a built in LED, this was used to indicate the input level being high and close to clipping.

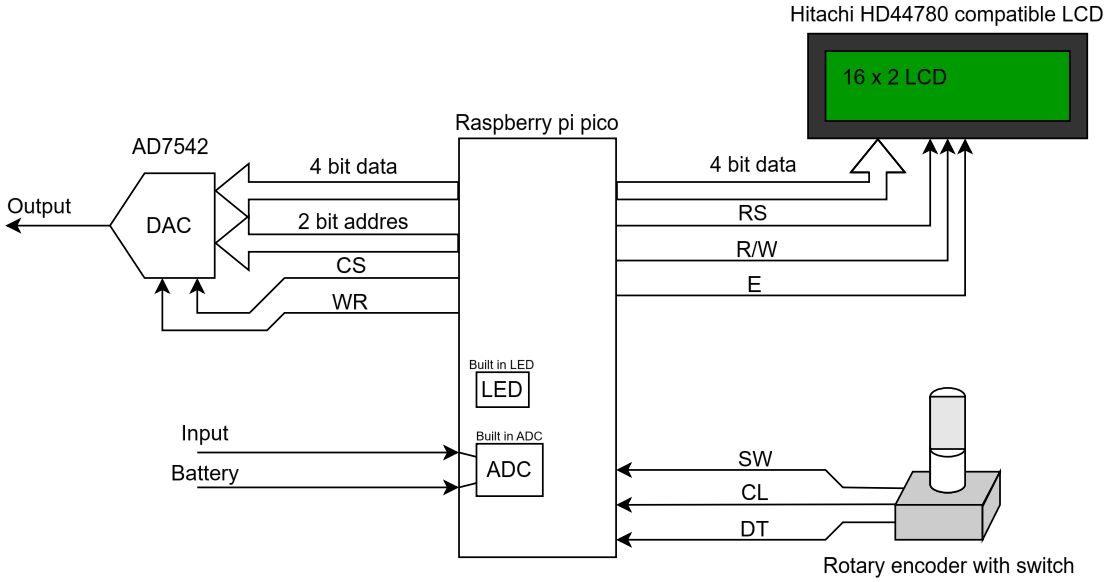


Figure 4.10: Block diagram of digital interfaces

In addition to the digital interfaces, one gpio is set up as to control the contrast level of the LCD by acting as an analog negative supply. This is achieved through some additional circuitry, shown in fig 4.11. The GPIO outputs a pulse train. First an inline capacitor isolates and removes the dc-bias. Then diodes will only allow the negative voltage to be passed on to LCD VEE. Lastly an additional capacitor and resistor smooths the voltage and adds a small load to the output.

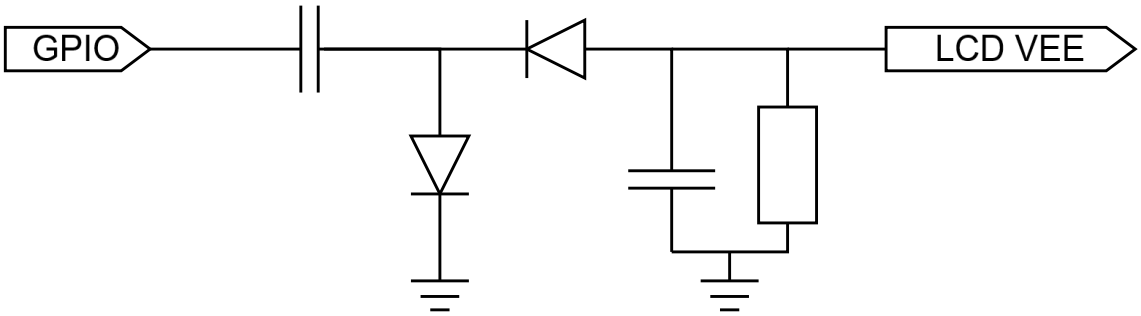


Figure 4.11: Block diagram of digital interfaces

4.2.2 Firmware

The firmware running on the raspberry pi pico has multiple functions to tend to, see figure 4.12 for a complete overview. The most important one being reading and writing to the DAC. It's very important that it is handled consistently for the echo cancellation and potential pitch shifting to work because these assumes a fixed and known sampling rate. Even the DAF may suffer severe quality deterioration, albeit at a lesser degree, if used with a varying sample rate as this could lead to a sample being played at different rate than it was recorded (due to the delay). For the best repeatability, we've implemented the signal chain through a repeating interrupt (technically a hardware alarm) thus halting whatever process is running to guarantee a consistent sampling-rate is fulfilled. However, it's still important that we set the repeating time to be longer than the execution time. Not doing so would cause one interrupt to be called whilst the previous is still running, thereafter pushing the old interrupt to the stack, which after some time would lead to a stack overflow and therefor corrupting the memory. Even though we use a hardware alarm function which would not cause a stack overflow, it would however still malfunction as the subsequent samples will be delayed, lowering the sample-rate to a unknown value. See [14] for a complete hardware reference.
We've also used interrupts to handle the main input, the rotary encoder. The three signals, SW, CL and DT, are grouped to one hardware interrupt. The interrupt function deciphers which signals changed and runs the appropriate function according to the menu system followed by requesting an LCD update by setting a flag. Holding the button down allows the user to reset the device to initial settings.
Finally the main program starts by initialize the hardware and sets the corresponding GPIO. It sets the proper interrupt mask to be ready for input and it redraws the LCD. The main loop checks whether the LCD flag is set and if so it then redraws the LCD.

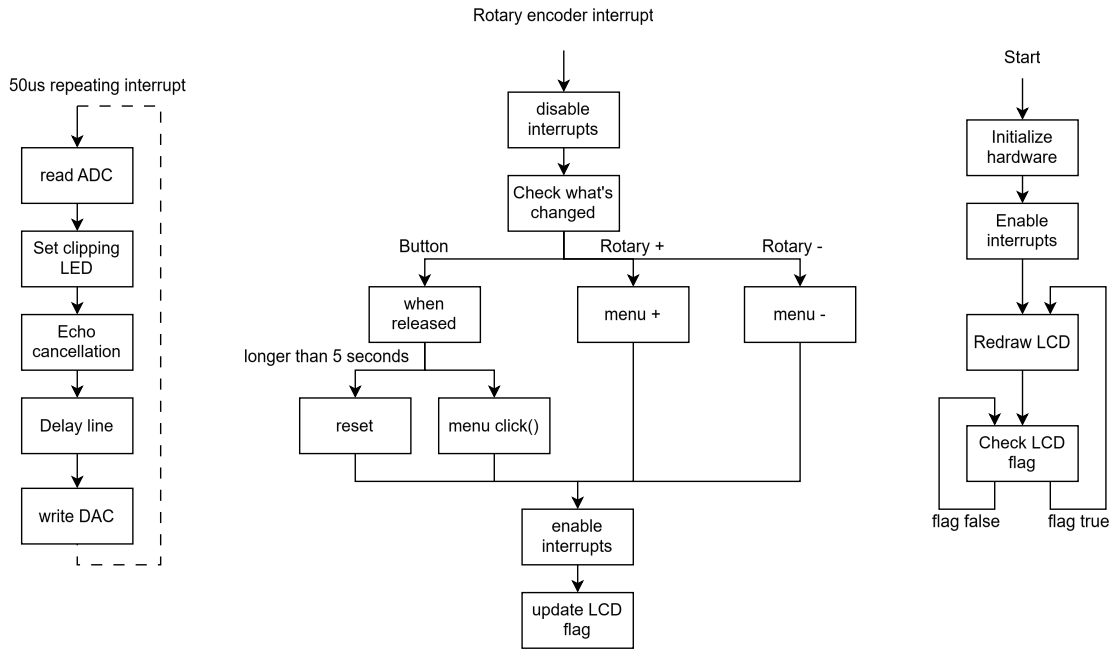


Figure 4.12: Block diagram of the firmware

The menu system is fairly concise and consist of a scroll-able subset of functions using the rotary encoder. Depressing the button of the rotary encoder allows the user to run or change a parameter of the currently displayed function, see figure 4.13.

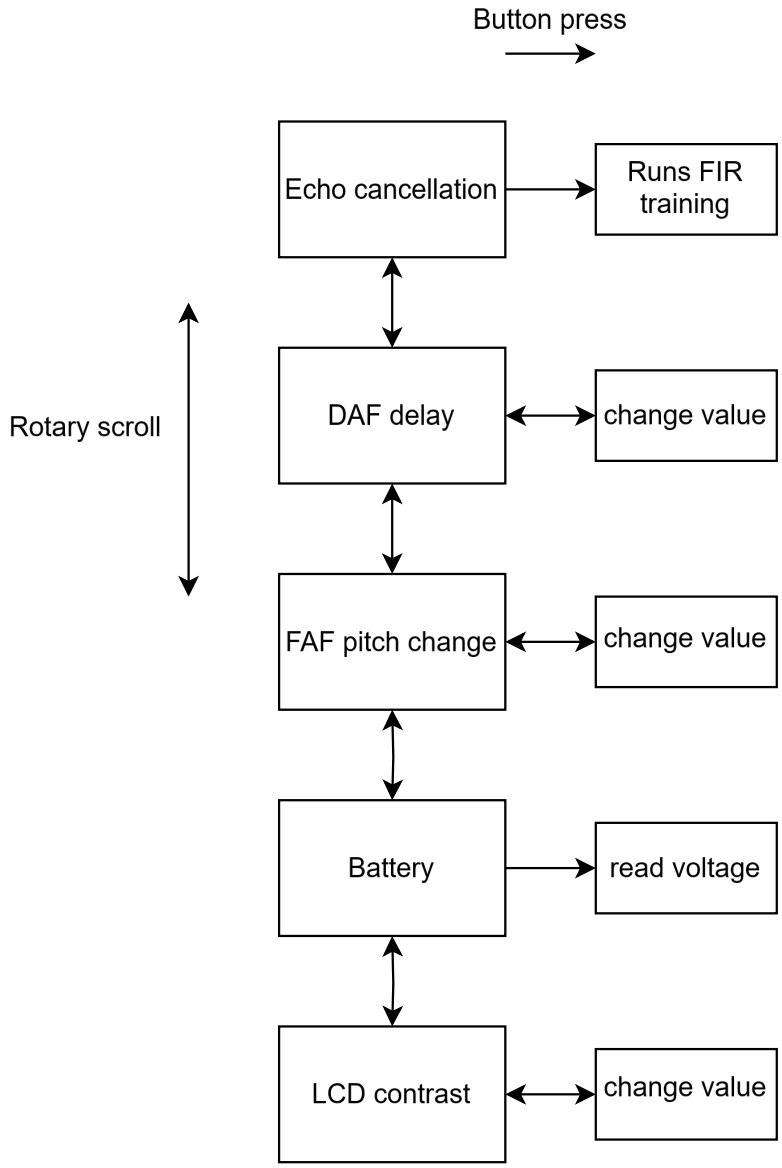


Figure 4.13: Block diagram of the menu system

4.2.3 Optimization

It shall be emphasized that the execution-time has been identified as the limiting factor for the sample-rate. We could measure this using the built in hardware timers of the pico, see figure 4.1. Training of the FIR also necessitate some extra time as it needs to compute not only the feedback filter but also the remaining error. We've settled on a sample-time of 35μs (approximately 28kHz) , which results in a decent margin for consistent sample-rate.

When implementing the FAF algorithms we stumbled upon a big hurdle, creating a performant FFT algorithm. Following the Cooley-Tukey approach it would appear the Raspberry Pi Pico isn't fast enough for the high enough sampling rates and so we've therefore tried improving it in a multitude ways. Some of these improvements are general across the Pico platform and is implemented for all algorithms, whilst

Table 4.1: Execution time for sampling, echo cancellation and playback

	execution time (μs)
normal operation	21
training echo cancellation	28

others are algorithm specific.
First of all, we changed the variable type used inside of the firmware. It is common to use floats in this application and many libraries does so, for instance KISS FFT. However, due to the lack of a dedicated floating point unit within the Pico, it is exceptionally slow for this particular variable type. We eventually determined through testing that 16 bit signed integers are instead the preferred variable type.
Secondly, we choose to implement the FFT in radix 4 mode instead of radix 2 which is more common. Radix 2 and 4 refers to how many parts each iteration divides the signal in and we saw a considerable performance increase whilst using radix 4. Depending on hardware and code compilation this isn't necessarily the optimal solution since the amount of iteration and therefore also the amount of function calls is far greater in a radix 2 implementation compared to radix 4. Because each function call results in a jump instruction and possible some push and pull on the stack memory it could instead lead to severe slow downs.
Thirdly, we removed the functions input variables in order to prevent unnecessarily pushing and pulling on the stack. This lead to a big performance increase for the echo-cancellation algorithm from this small change.
Lastly, we overclocked the microcontroller to 250MHz from it's default value of 125MHz. The Pico has generally been proven to be reliable even at these speeds, however, since the FAF could not be implemented with an acceptable sample rate even when overclocked, it was later returned to default for higher reliability and lower power consumption.

4.3 Measurements

For the first test, by directly measuring the device, we characterize the frequency response of the device over four different input amplitudes, see figure 4.14. By observing the graphs, we note a very flat response over most of the frequency spectra with a gentle taper which indicates a slight low pass behavior. The low pass behavior is believed to be inherent to the specific DAC that we have chosen since there is mentioning of such behavior in the data-sheet. We can also observe that the gap between lines diminish with increasing amplitude as we reach saturation.

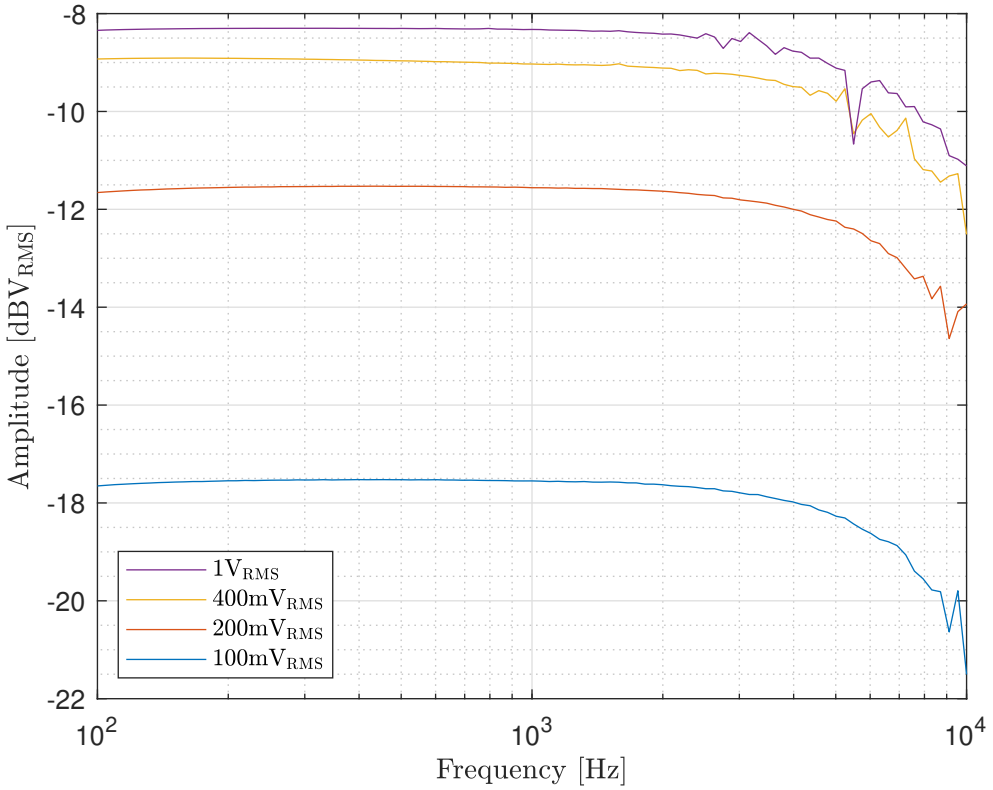
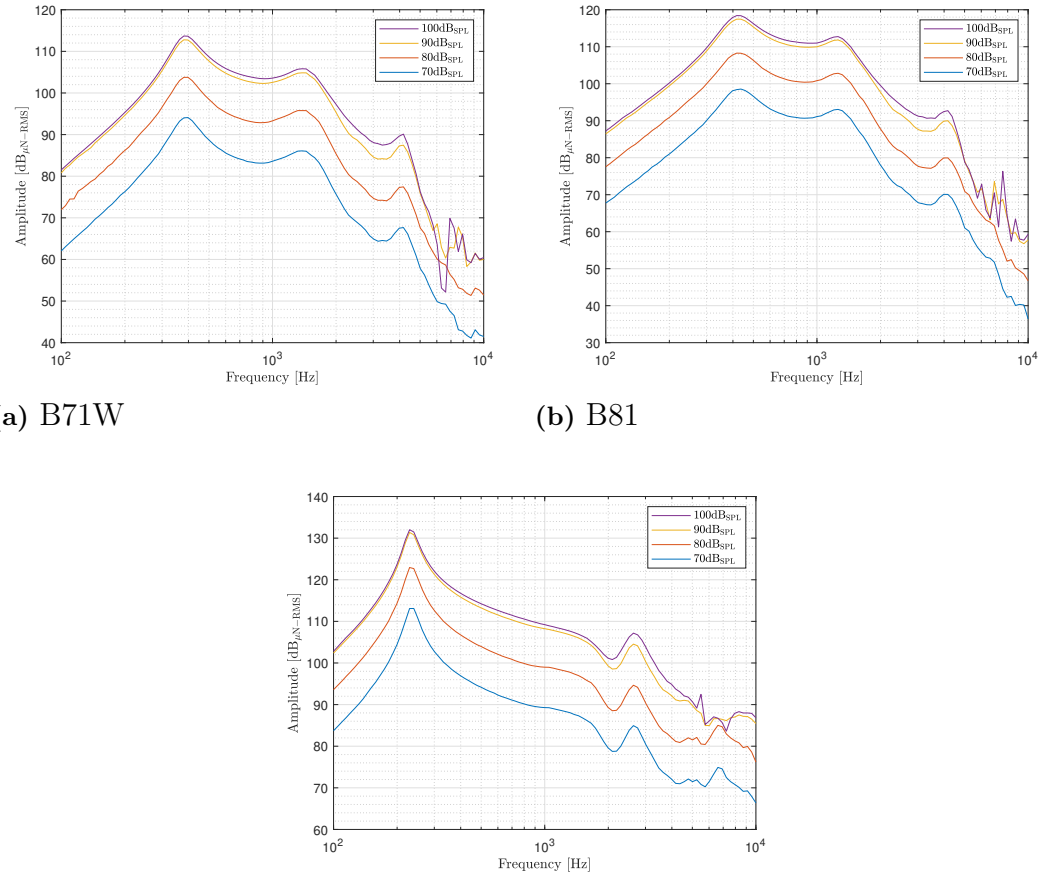


Figure 4.14: Frequency response with direct electric stimuli of different amplitude

After connecting vibrator and microphone to the device, we've then measured the frequency response using three different bone vibrators, see figure 4.15. The response curves are similar to those of the bone vibrator alone from their respective datasheet, therefore confirming that the system does not substantially alter the sound characteristics. We can also observe that the peak force being fairly high across the board, which indicates a good power amplifier design.



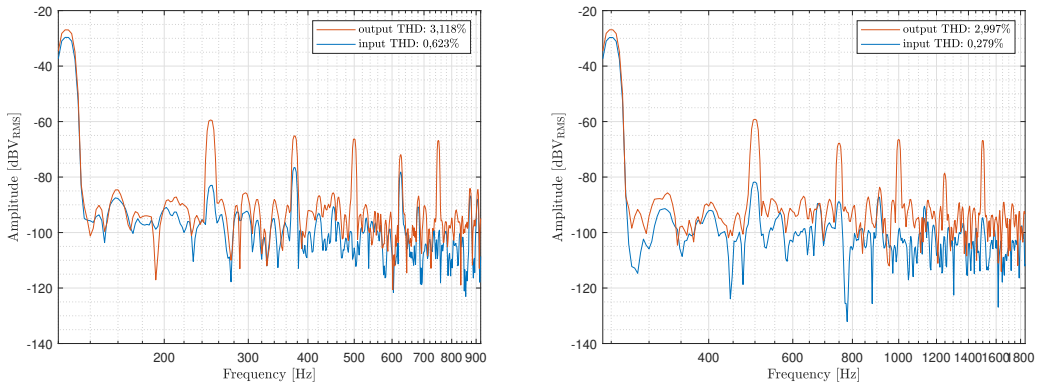
(a) B71W

(b) B81

(c) B250

Figure 4.15: Force frequency response for various input amplitudes and bone vibrators

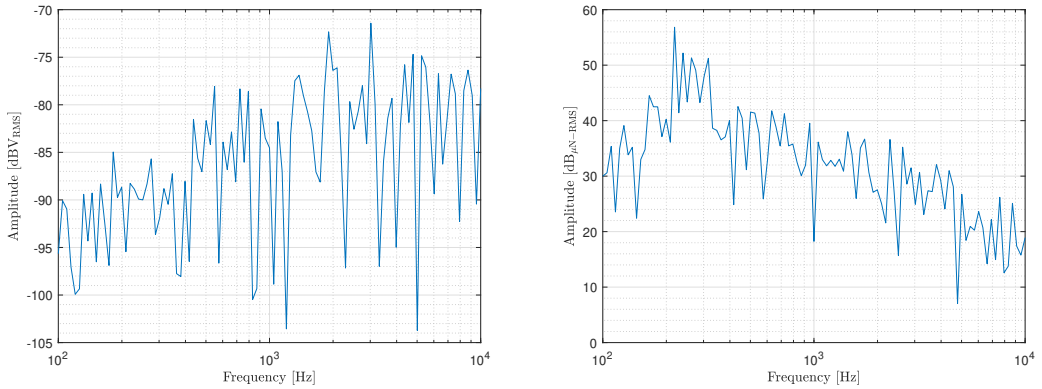
Aside from frequency response, we also intended to characterize distortion levels. We measured the total harmonic distortion when feeding the device a sine wave tone. Measurements are done with a bone conducting microphone at 70dB$_{\text{SPL}}$, see figure 4.16. The input level is measured between the microphone and the device and is noted to have a very low distortion at less than 1%. The output is measured directly without a bone vibrator connected and whilst we've observed that the device is causing some distortion, it is still fairly low at about 3% adhering to IEC 60645-1 (less than 6%).



(a) 125Hz **(b)** 250Hz

Figure 4.16: Total harmonic distortion measured with two different input signals, one 125Hz sine, the other 250Hz sine

Finally we made noise measurements by leaving the input disconnected and whilst measuring the output, both directly and whilst connected to a B250 bone vibrator, see figure 4.17. We can establish that the noise is very low and inaudible in most applications.



(a) direct measure **(b)** Bone-vibrator measure

Figure 4.17: Noise characteristic without input stimuli measured directly or force from bone-vibrator

5
Discussion

Whilst the software version implemented in Matlab is function complete and encompass DAF, FAF and echo cancellation successfully, a standalone hardware unit can still be preferable. The primary drawback of using a normal computer is that the buffering of audio input causes an inherent delay and thus limits the practical range for DAF. For this reason, it is of some interest to further develop the hardware unit to encompass all of the same features within a single device.

Since we have established that it is interesting to have a working FAF implementation on hardware, we can ascertain a few options for future work. A conceptionally simple solution is to change the digital platform either by using a faster microcontroller or by using something more niche and purpose built, like a dedicated digital signal processor (DSP) or a field programmable gate array (FPGA). The current software is only using one thread and because the Raspberry Pi Pico has two cores, another option is to rewrite the code for multi threading which could yield a doubling in performance. A third option is to lower the frequency criteria and thus the requirement on sample-rate which would decrease computational load. Whilst lowering the frequency criteria could come at a cost of audio quality, in accordance with frequency response of the bone conducting vibrators, we can expect a decent margin due to its poor high frequency behaviour.

Even though a multitude of tests performed on the PCB design seems to indicate that the design has been successful, we have achieved low distortion values and a very good spectral performance, the DAF works and can be adjusted in different increments, from 0 to more than 500ms of delay, there is still a point of contention. Even though the echo cancellation is also implemented, it has not yet been extensively tested due to it will likely require a very complicated setup to replicate the intended use.

Whilst radix 4 was determined to be a better option than radix 2 when implementing FFT, further testing can be done to find the optimal radix for the given hardware. Although, due to the considerable amount of new code needed to evaluate any radix over 4 whilst also expected to have diminishing improvements, instead, investigated the performance of non-recursive FFT is likely of higher interest.

We think that the frequency shift is a promising solution to achieve FAF at a reduced computational cost as it achieves the core function of altering the sound making it brighter or darker in tone without resampling. It is however not yet proven if it has the same effect in real patients as the pitch shifting do. Even further deviation could be made from the standard pitch shifting, for instance chorus effect is common frequency type effect which alters between pitching audio up and down, something which may be similar enough to the proven methods that it works too.

During testing, the power consumption was noted to be poor by the fact that batteries had to be changed more frequently than desired. It stands to reason that the power amplifier circuit is likely poor performing in terms of power efficiency. Whilst not confirmed, the Darlington pair configuration is suspected to worsen the already poor efficiency of the class AB amplifier circuit. Due to the strong need for it early in the hardware prototype phase, the power circuit design was primarily guided by our familiarity to its implementation and component availability. For future improvement, we believe that a class D amplifier circuit should be investigated due to its superior power efficiency.
Another point for reduction in power consumption is voltage regulation and battery management. A 9V battery solution was chosen due to its convenience and due to the preference for a replaceable battery which can lead to continuous operation. However, since both the LCD interface and processing parts are powered by 5V, a linear regulator is used. It would be preferable to use a switched DC to DC power supply such as a buck converter for voltage regulation since its efficiency is generally higher.
Lastly, LCD interface is a likely contestant for optimisation in terms of power efficiency. We've already omitted LCD backlight for the PCB design due to its large contribution to power consumption. The LCD used is an old and widely adopted model but for future work a more modern display with less power consumption could be investigated.

6
Conclusion

The main goal of this project was to provide a means of introducing delayed auditory feedback, an effect known to induce fluency in people who stutter. A MATLAB® script and a standalone hardware unit have been produced, with the addition of frequency altered feedback and echo-cancelling features. The device has been made to function with bone conduction microphone and transducers. While some electronic fluency devices are commercially available, further research can be carried out about the use of bone conduction devices in the field of stutter inhibition. Future enhancements include refinement of the pitch-shifting feature, miniaturisation, energy consumption.

Bibliography

[1] O. Bloodstein, N. B. Ratner, and S. B. Brundage, *A Handbook on Stuttering*. Plural publishing Inc, 2021, ch. 2. [Online]. Available: https://www.google.com/books/edition/A_Handbook_on_Stuttering_Seventh_Edition/Abw0EAAAQBAJ

[2] D. Hudock and J. Kalinowski, "Stuttering inhibition via altered auditory feedback during scripted telephone conversations," *Int J Lang Commun Disord*, 2013, pMID: 24372890. [Online]. Available: https://doi.org/10.1111/1460-6984.12053

[3] A.-C. Persson, B. Håkansson, M. C. Mechanda, W. B. Hodgetts, K.-J. F. Jansson, M. Eeg-Olofsson, and S. Reinfeldt, "A novel method for objective in-situ measurement of audibility in bone conduction hearing devices – a pilot study using a skin drive bcd," *International Journal of Audiology*, vol. 62, no. 4, pp. 357–361, 2023, pMID: 35238713. [Online]. Available: https://doi.org/10.1080/14992027.2022.2041739

[4] S. Reinfeldt, M. Eeg-Olofsson, K.-J. Fredén Jansson, A.-C. Persson, and B. Håkansson, "Long-term follow-up and review of the bone conduction implant," *Hearing Research*, vol. 421, p. 108503, 2022, acoustic Implant Technology. [Online]. Available: https://doi.org/10.1016/j.heares.2022.108503

[5] M. McBride, P. Tran, T. Letowski, and R. Patrick, "The effect of bone conduction microphone locations on speech intelligibility and sound quality," *Applied Ergonomics*, vol. 42, no. 3, pp. 495–502, 2011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0003687010001523

[6] S. Chandrasekaran. (2014) Implementing circular buffer in c. Accessed : 2024. [Online]. Available: https://embedjournal.com/implementing-circular-buffer-embedded-c/

[7] A. W. M. van den Enden and N. A. M. Verhoeckx, *Discrete-time signal processing : an introduction*. Prentice Hall, 1989, ch. The DFT and the FFT, pp. 141-148. [Online]. Available: https://archive.org/details/discretetimesign0000ende

[8] W. A. Sethares. A phase vocoder in matlab. Accessed : 2024. [Online]. Available: https://sethares.engr.wisc.edu/vocoders/phasevocoder.html

[9] F. Grondin. (2009) Guitar pitch shifter. Ch. 2 Pitch shifting. [Online]. Available: https://www.guitarpitchshifter.com/pitchshifting.html

[10] A. W. M. van den Enden and N. A. M. Verhoeckx, *Discrete-time signal processing : an introduction*. Prentice Hall, 1989, ch. Multirate systems, pp. 235-251. [Online]. Available: https://archive.org/details/discretetimesign0000ende

[11] F. Grondin. (2009) Guitar pitch shifter. Ch. 3.1 - 3.2. [Online]. Available: https://www.guitarpitchshifter.com/algorithm.html
[12] ——. (2009) Guitar pitch shifter. Ch. 3.3 - 3.4. [Online]. Available: https://www.guitarpitchshifter.com/algorithm.html
[13] B. Widrow, J. Glover, J. McCool, J. Kaunitz, C. Williams, R. Hearn, J. Zeidler, J. Eugene Dong, and R. Goodlin, "Adaptive noise cancelling: Principles and applications," *Proceedings of the IEEE*, vol. 63, no. 12, pp. 1692–1716, 1975.
[14] Hardware apis. Raspberry Pi Ltd. Accessed : 2024. [Online]. Available: https://www.raspberrypi.com/documentation/pico-sdk/hardware.html
cleardoublepage appendix

7
Appendix

Operation directions

Startup

Whilst the device is off, make sure the microphone, speaker and battery is plugged in. Then, the device can be turned on using the red switch on the front side of the device. Be mindful there's no loud feedback which can cause discomfort, if present, immediately turn the device off and turn to the section about setting input and output gain.

Echo cancellation

After startup, whenever the microphone or speaker are re-positioned and when gain is changed, the FIR filter needs to be retrained. If FIR training is not performed after startup, the device will operate without echo cancellation. Using the rotary dial on the front of the device, navigate the menu until the LCD display says "Echo suppression", then press on the rotary dial to begin FIR training. During training, the wearer will hear a white noise. The FIR training will continue until the rotary dial is pressed again. Adequate time for training is expected to be roughly 20s but will vary by changing training parameters within the firmware. If the trained FIR doesn't yield a desirable result, consider changing the training parameters within the firmware.

DAF delay

The DAF delay is running by default. To change the delay value, navigate first to "DAF" using the rotary dial, then, press down on the rotary dial. The rotary dial is now used to control the delay time and is immediately updated. In order to turn DAF off, change the delay to 0. When the desired delay value is achieved, pressing the rotary dial will return the rotary dial to control the menu.

FAF pitch change

Since FAF is not implemented it will appear as a placeholder menu item. Pressing the rotary dial whilst navigated to "FAF" will let you adjust the pitch change value however this is also a placeholder menu item and won't affect the operation of the device. Press the rotary dial again to return the rotary dial to control the menu.

Battery

Low battery voltage can lead to reduced volume output and distorted sound. In order to diagnose if the battery is at fault, navigate to "Battery" using the rotary dial. Then a voltage measurement can be done by pressing the rotary dial once. For ideal performance battery voltage should be kept above 8V.

LCD contrast

If the LCD is hard to read then the contrast should be adjusted. This is done by first navigating to "LCD" using the rotary dial. Press the dial once and it will now be used to adjust the contrast value. Press the rotary dial again to return the rotary dial to control the menu.

Setting input and output gain

Two blue potentiometers are located on the back of the device. The two controls can be adjusted using a flat blade screwdriver. In order to calibrate these controls, first the gain should be set to its minimum. This is achieved by turning the controls fully anti-clockwise. The input gain should then be increased by turning clockwise until slight activity is shown by the clipping LED during normal speech. Too high input gain will result in severe clipping and deteriorate the sound quality. The output gain can be set as high as possible accordingly with the comfort of the user. Now the input and output gain is calibrated.

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY