An Introduction to Audio Signal Processing

Basic Functions of Audio Production

James Aaron Jordan

Department of Computing Sciences, Coastal Carolina University, Conway, SC, US, jajorda3@coastal.edu

ABSTRACT

Audio processing is a rich topic, both in the analog and the digital domain. Within this project, some common processing functions are implemented and investigated on an Intel DE-10 Standard FPGA. The discussion focuses on frequency spectrum analysis of music, white noise, pink noise, and a 440Hz test tone after application of processing to modulate the sample rate and apply audio filters to the given files. The report opens with a basic description of all concepts necessary to understand the analysis of the processed samples, which is followed by a brief description of the advantages of the hardware used for this experiment. Section three details the experimental procedure and plots the data collected, followed by analysis of these findings in section four. The report closes with a section covering final remarks, including future research opportunities and reflections on the experiment, as well as a review of the experiment in comparison to a flagship machine in the field of digital signal processing. Prerequisite knowledge in the field of audio signal processing is useful, but section one details all technical terms and concepts used in the report.

KEYWORDS

Audio processing, signal processing, DSP, sample rate, filtering, high pass, low pass, filter, Quartus, Intel, DE10, FPGA, pink noise, white noise, Verilog, VHDL

1 Introduction: The Foundations of Digital Sound

The simple idea of taking a sound from one place to another has long been a topic of discussion in engineering fields, particularly electrical and computer engineering. The iterative improvement from wax cylinder to streaming internet radio has been a long process of asking questions like, "could this sound better?" and "could this be more convenient?" The field and its accompanying debates have covered vast ground over the years, from analog vs. digital mixing and distribution to the loudness wars of the 2000s, displaying clearly the richness and variety of the equipment and the problems facing audio system design. Over time, fundamentals of audio processing have been developed and built upon to reach the sophisticated level of understanding that exists within the field today.

1.1 Noise and Human Hearing

To digitally reproduce sound, it is necessary to be able to measure it. As such, several standards were developed to define the most important attributes of sound. The focus of this research revolves around understanding two of these characteristics: volume and pitch.

Volume, loudness, or amplitude can be used interchangeably in many contexts to refer to the level of a sound, from nearly silent sounds like the ticking of a watch to the loudest common sounds, like an airplane taking off. This attribute, sound pressure level, is most commonly measured in decibels [dB], which is a logarithmic unit. An 80dB room will be twice as loud as a 70dB room, and each 10dB gain from there will be twice as loud as the previous step.

In the digital domain, this metering operates a little differently. Sounds are instead measured by how much quieter they are than the loudest possible sound, which is anchored to 0dBFS (decibels full-scale). To draw similarity to the previous example, a -20dBFS sound will be half as loud as a -10dBFS sound. A program or file which attempts to write a value over 0dBFS will need to be cut back down, or clipped, to 0dBFS. Finding ways to work with audio freely despite this technical limitation was a major hurdle in the development of digital sound processing. On an unrelated note, the full-scale designation is typically assumed digital in audio, so it is dropped and—perhaps confusingly—digital signals are also noted to be measured in plain dB.

The second important characteristic to this research is pitch, which is marginally less complicated and refers to the frequency of a sound. Pitch can range from low, like the hum of an A/C unit or thump of a kick drum hit, to high, like cymbals crashing or Styrofoam scratching against another surface. The exact frequency or specific pitch which is most prominent in the given sound can be measured in Hz to be put into relation with other sounds, though sounds mix together to form a blend of all the frequencies. Human hearing allows perception of sounds from around 20Hz to 20,000Hz. Due to this, many audio metering tools (such as those used in this project) are calibrated to give their most accurate results on this range.

Also, apparent volume is logarithmic with respect to pitch. If two pitches are played one octave apart (such as 220Hz and 440Hz), but at the same measured volume, the high pitch will be perceived as significantly louder. For this reason, there are two major types of noise used in audio development: white and pink. White noise can be thought of as “technically” uniform noise, where the volume of any pitch is the same as every other pitch. Pink noise, in turn, can be thought of as “realistically” uniform noise, as it is logarithmic, and therefore more closely matches the spectrum of human hearing.

1.2 Common Methods of Processing Audio

Audio processing in general has built on a few fundamentals for many years. Most processes can be broken into one of two categories or defined as a combination of the two.

1.2.1 Filtering and Equalization

Equalization (or just EQ) can be thought of as manipulating the many pitches within a sound to change it. This is done in this research intentionally through filtering, which can be thought of as a basic type of EQ. Most filters remove frequencies beyond a certain point, known as the filter’s cutoff frequency. For instance, if we have typical a high-cut filter with a cutoff frequency of 3200Hz, it will cut 3dB from the sound at 3200Hz and progressively cut more for higher pitches within the sound. This type of filter could be used to give the impression of hearing from underwater.

Filters may be described in two ways: as a “pass” or a “cut/reject” filter, where one is simply the inverse of the other. Above, the filter was defined as a high-cut, which is simpler to explain but could have just as easily been defined as a lowpass filter, as it allows the low frequencies to pass through the filter unchanged. The remainder of this report will use “pass” naming to describe filters.

1.2.2 Compression and Limiting

Compression and limiting are used to manipulate the overall loudness of a sound. Compression can be used to make loud sounds softer and soft sounds louder, leading to a more consistent volume level. Limiting follows the same concept but generally has emphasis on making the whole sound louder, while compression is commonly used to move volume in either direction. Clipping, as mentioned above, is the most basic example of limiting. Within the scope of this project, these functions are avoided as much as possible to yield consistent data for measurable, comparable results.

1.3 Additional Concerns in the Digital Domain

While the processes in section 1.2 apply to all audio manipulation, these topics are most relevant to digital processing. To understand the significance of this section, it is important to know that speakers work by moving back and forth to displace and vibrate air, which carries sound through the world. Digital audio files map the movement that speakers make to reproduce the desired audio.

1.3.1 Sample Rate

Sample rate is a variable used to specify part of the resolution of a sound. It refers to the number of times within one second that a measurement is taken of where the speakers should be. Some common, modern sample rates are 44.1kHz, which is CD-quality; 48kHz, which is DVD quality; and 96kHz, which is Blu-ray disc quality. Sample rate is related to the range of frequencies which can be reproduced through a file of that resolution by the Nyquist rate. In sum, this rate defines that the highest possible frequency that can be stored at any sample rate is at most half the chosen sample rate. As the range of human hearing runs up to 20kHz, the advantage of sampling beyond 40kHz quickly diminishes in a most cases.

1.3.2 Bit Depth

Bit depth is used to refer to the number of possible values available for any specific sample to read. Throughout this project, the CD quality standard of 16 bits is used.

2 The FPGA Approach

2.1 What is an FPGA?

An FPGA, such as the DE-10 used in this project is a field-programmable gate array. This type of device gives the programmer the ability to instruct the system to run its gates and processing elements in any desired order. This allows the device to operate differently from a normal computer program or microcontroller because operations can happen simultaneously. In a very simple case, a user could program the unit to connect each switch on the device to an LED elsewhere on the board. The inputs could then be toggled independently and operate on the assigned LED instantly, whereas a microcontroller performing the same functionality would have to wait on the processor to check every input on each clock cycle, delaying the output to the LEDs. In this case, the time taken could be trivial, and a microcontroller may yield about the same performance as an FPGA, but as designs become more complex, the advantages of the FPGA become more apparent.

2.2 Why use an FPGA for Audio Processing?

One such complex design is a live audio processing engine. While any set of functionalities may be implemented to augment an audio signal, as explained in section 1.2, it is critical that all live processing be completed before that signal needs to be sent to the speakers. This requires incredibly rapid hardware. At a sample rate of 96kHz, the system must complete all necessary processing for each sample and be ready to output it within 10.4µs. While a microcontroller or software solution could handle this task, it is unlikely that it will be able to handle it as reliably as a hardware solution, such as testing on an FPGA.

3 The FPGA Experiment

3.1 Equipment Used for this Project

There are many pieces in the signal flow of this project. The keystone, of course, is the Intel-Terasic DE-10 Standard FPGA. This board did all intended manipulation of the audio involved in the project, aside from a clean, digital boost of the output audio from the board by 21dB to aid in comparison and compensate for the low output volume from the DE-10.
The full signal path for white noise, pink noise, and the 440Hz tone is as follows:

1. Sounds were generated by the SigGen (noise generator) section on an Allen&Heath dLive C3500 mixing console at a 96kHz sample rate mixing console and stored to a USB key.
2. These sounds were loaded into an iPhone 7 Plus for playback through the Google Drive app.
3. The playback was carried over a 3.5mm auxiliary cable at full volume into the line-in on the DE-10.
4. Processing was handled within the DE-10 and output through the headphone jack on the FPGA.
5. The left channel of the output from the FPGA flowed into the mono FX return on an M-Audio M-Track audio interface, also believed to be set at a 96kHz sample rate.
6. The audio was recorded into Audacity, a digital audio workstation running in a Windows environment.
7. Files were exported as a sequence of 16bit PCM .WAV files and moved into a macOS environment.
8. Files were inspected in Logic Pro X through utilization of Waves Audio's PAZ-Analyzer plugin.

Signal path for music was very similar, only differing in steps 1 through 3, where the alternative path took these steps:

1. Music was played through the Spotify app on an iPhone 7 Plus at full volume over a 3.5mm auxiliary cable into the line-in on the DE-10 FPGA.

2. Continue with step 4 above.

Also note that the same section of the songs chosen for phase 1 and phase 2 were used for fair analysis. As labeled in section 3.3, these songs were Real Friends' "Unconditional Love" in phase 1 and The Maine's "Black Butterflies and Déjà Vu" in phase 2 of the project.

3.2 Experimental Procedure

While the general procedure follows the signal path demonstrated above, specifics to each phase of the experiment are listed in this section.

3.2.1 Phase 1: Sample Rate Comparison

In phase one, all known toggles for the M-Track interface and involved digital audio workstations were set to 96kHz so that the only downsampling that could occur would be within the FPGA circuit. The System CD example project DE10_Standard_Audio was used for this phase of the experiment. Following this pre-requisite, data was collected in this sequence:

1. Music at the following sample rates: 96kHz, 48kHz, 44.1kHz, 32kHz, and 8kHz.
2. White noise at the following sample rates: 96kHz, 48kHz, 44.1kHz, 32kHz, and 8kHz.

3.2.2 Phase 2: Filtering

In the second phase of the experiment, the sample rate of the M-Track and digital audio workstations were set to CD quality, 44.1kHz. A GitHub repository with a project set up for the DE2-115 was adapted in Quartus to work with the pins available on the DE-10 for this phase of the project. After setup, data was collected by gathering baselines, then applying filters in this order:

1. Music with the following filters: none (bypass), bandpass, lowpass, highpass.
2. Pink noise with the following filters: none (bypass), bandpass, lowpass, highpass.
3. White noise with the following filters: none (bypass), bandpass, lowpass, highpass.
4. A 440Hz test tone with the following filters: none (bypass), bandpass, lowpass, highpass.

3.3 Data Collected

This section showcases charts of selected data[1] collected as part of this experiment. Line properties are consistent within each phase of the experiment.

[1] A solid "Source" line is provided on relevant charts, but this data is not level-matched to the baseline "Bypass" line. "Source" is only included because its differentiation from the "96kHz"/"Bypass" line indicates an error in the measurement of frequency information near and above 17.5kHz due to a technical error detailed later in this document.

3.3.1 Charts for Phase 1

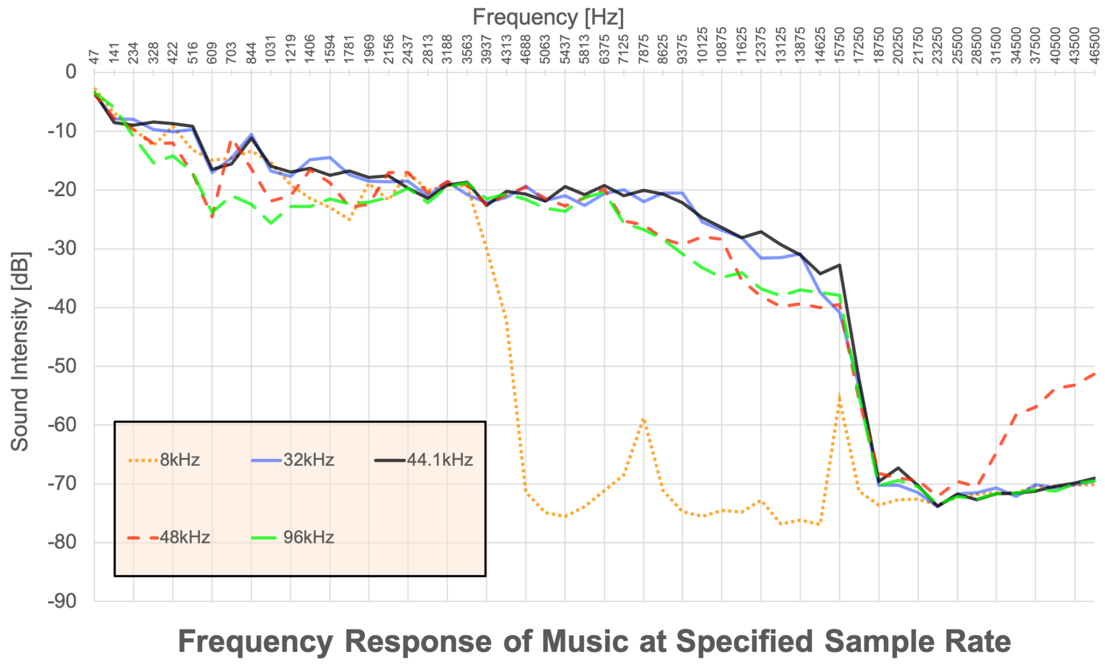

Fig. 1: Frequency response of Real Friends' "Unconditional Love" at varied sample rates

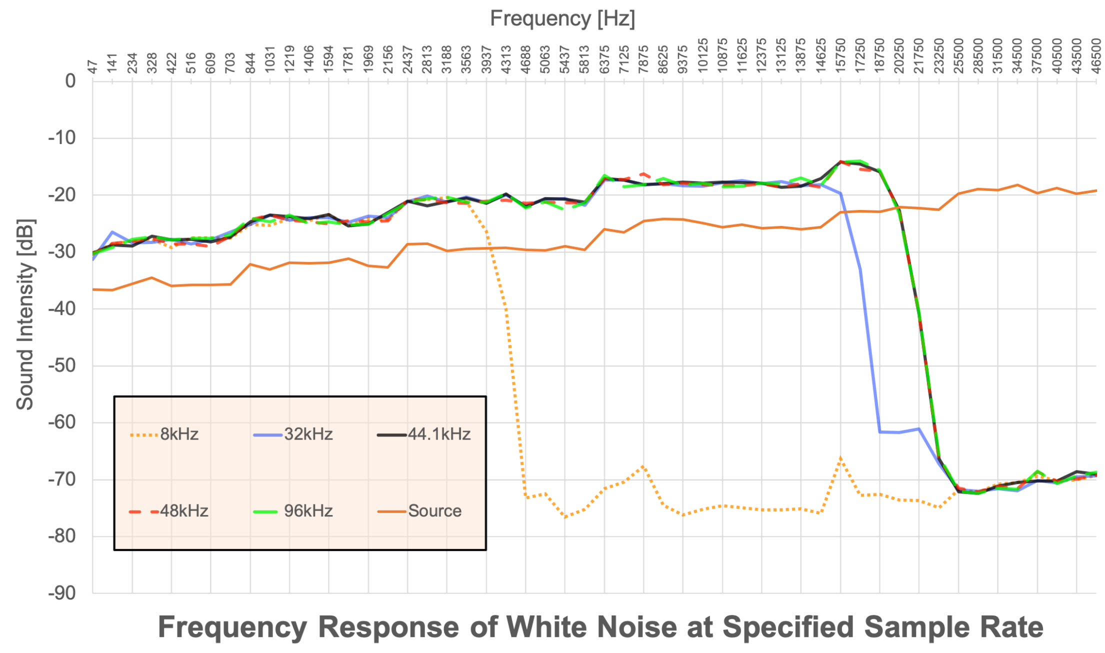

Fig. 2: Frequency response of white noise at varied sample rates

3.3.2 Charts for Phase 2

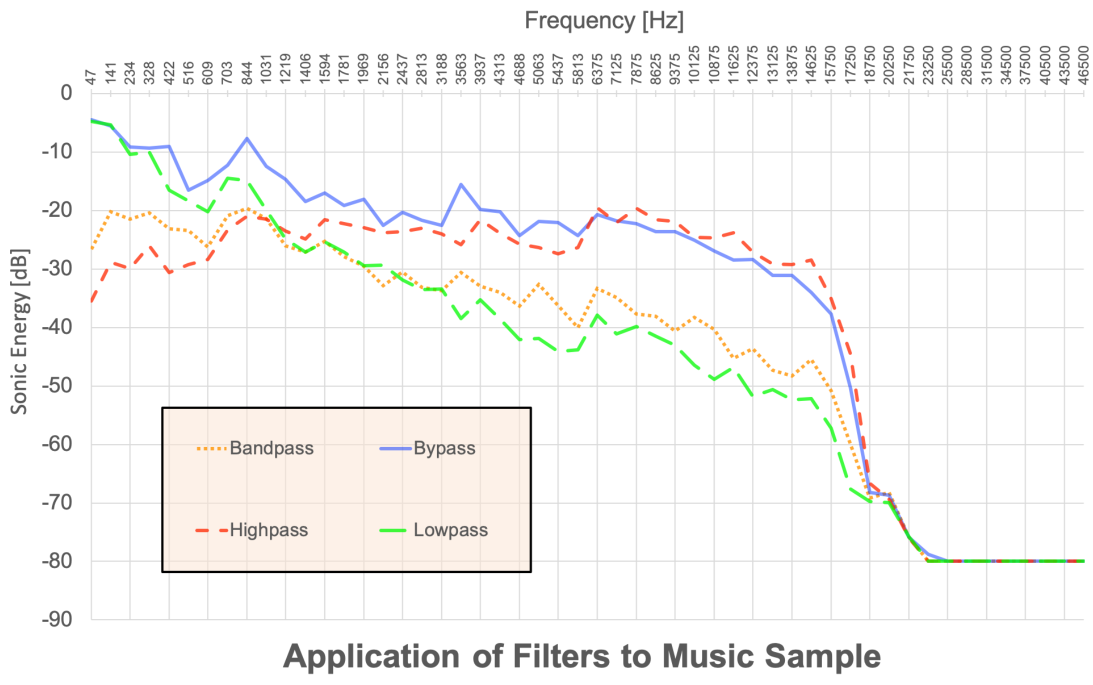

Fig. 3: Frequency response of The Maine's "Black Butterflies and Déjà Vu" with varied filters applied

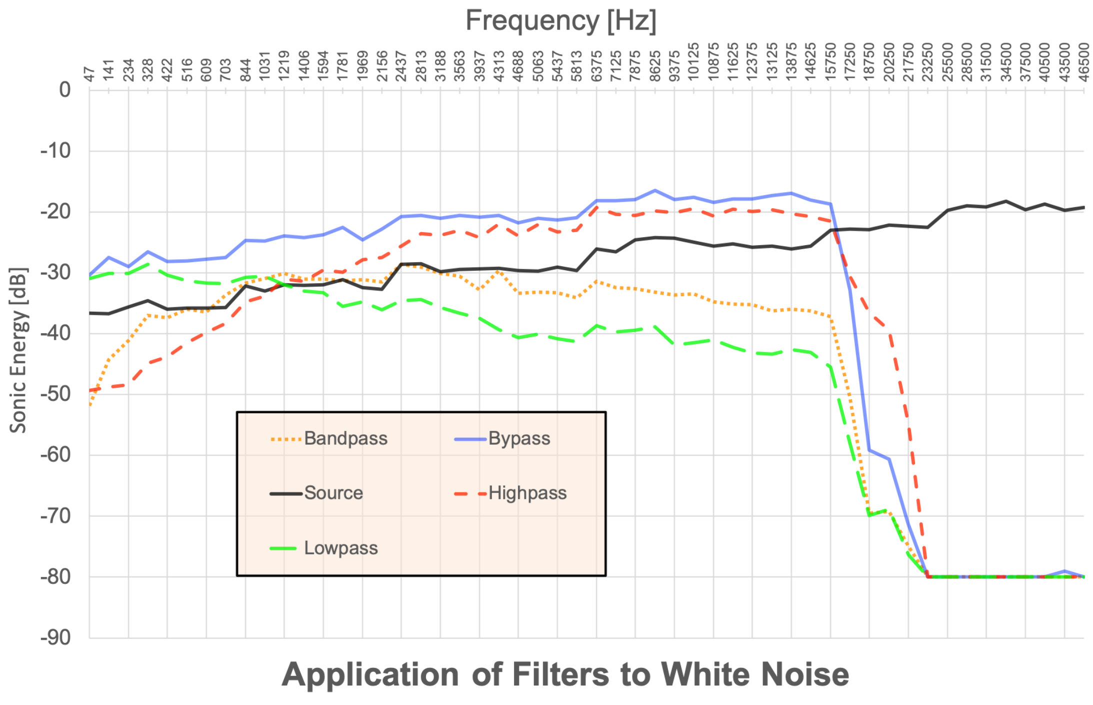

Fig. 4: Frequency response of white noise with varied filters applied

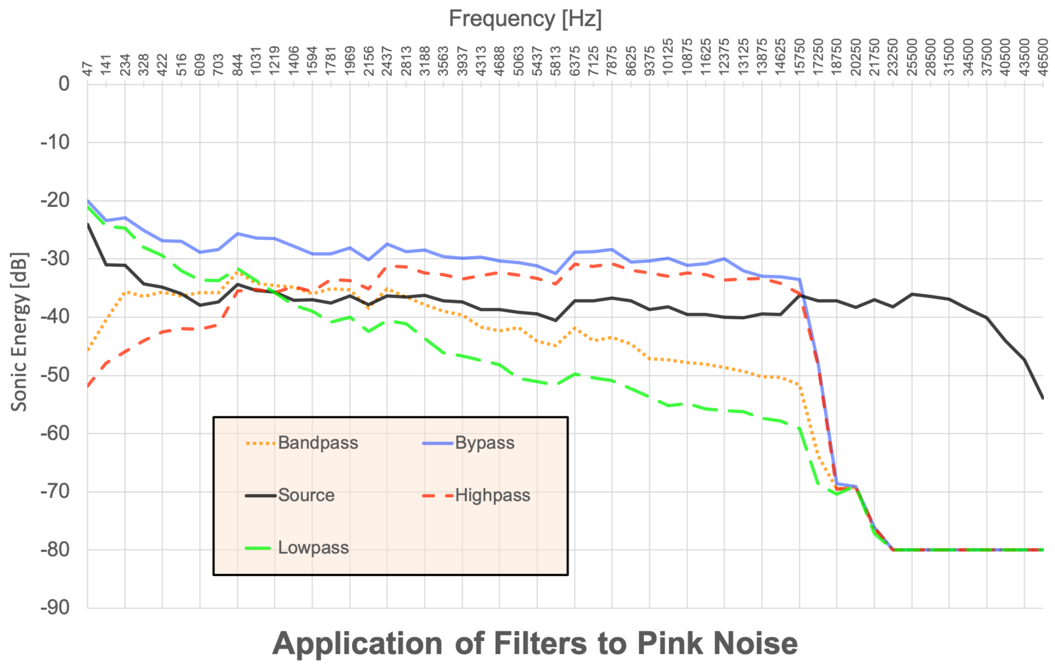

Fig. 5: Frequency response of pink noise with varied filters applied

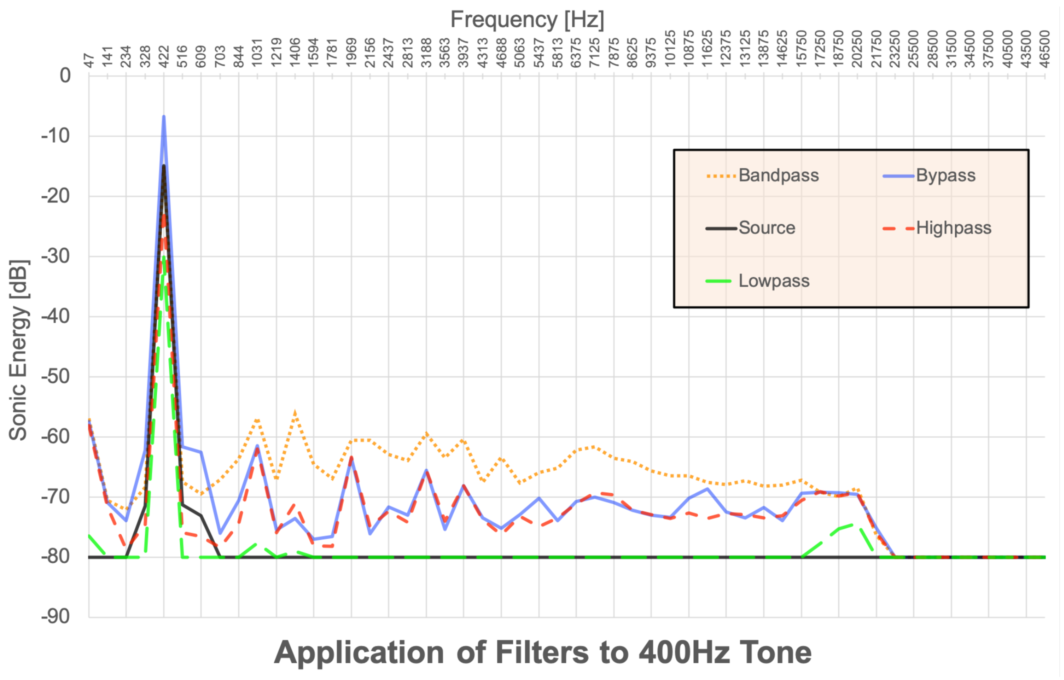

Fig. 6: Frequency response of a 440Hz test tone with varied filters applied

4 Analysis of Experiment

4.1 Preface: One Major, Noteworthy Error

As is exceptionally apparent in Fig. 2 of this document, there is significant falloff of frequency response data near and above 17.5kHz across all sections of the data. A retrospective observation of the involved recording equipment revealed that the Windows 10 operating system forced the audio interface used for recording to operate in 44.1kHz mode, though it is capable of operating at up to a 96kHz sample rate if configured properly. The option to run the session in 96kHz mode was selected in the digital audio workstation (Audacity), but it is clear from the data that this did not override the system setting for the device. This bottleneck is the reason for inclusion of the "Source" line, which was used to check that the source file was generated correctly at 96kHz by the dLive console in all three instances of its use. Inclusion of this line also isolated the error to one possible source: the Windows operating system setting. Once again, this line is not amplitude-matched to the baseline "Bypass" line in section 3.3.2. In an ideal case, the "96kHz"/"Bypass" and "Source" lines would be congruent. Implications of this error in the sections detailing the phases of the research conducted are ignored: only frequency information below 17.5kHz is regarded as reliable for the following analysis.

4.2 Phase 1: Sample Rate Comparison

Within phase 1 of research, sample rates natively accessible with example code available within the "Demonstration" section of the system CD for the DE-10 board were compared with respect to their frequency response. The findings, both with music and white noise recorded through the system corroborated the accuracy of the Nyquist rate as an upper limit to the accuracy of sample rates. A clear falloff in frequency response occurs near 4kHz when the 8kHz sample rate is active, and again around 16kHz when the 32kHz sample rate is active. However, this also occurs around 19kHz when the 44.1kHz sample rate is active, which is a lower frequency than the ideal expected falloff at 22.05kHz. This is allowed within the definition of the Nyquist rate.

White noise is best to use for readability of this comparison, as its attribute of having fixed energy at any frequency makes it easier to compare and generalize with than the music sample, which more closely follows the logarithmic nature of pink noise. Waves' frequency analyzer used for this experiment uses varied Q (essentially frequency width) in its detection, which is the reason that the white noise "Source" line trends upward, rather than directly across the graph. This more closely emulates human hearing, which is useful in music production, but a minor inconvenience in this setting.

In practice, except for the 8kHz sample rate, the audio quality across sample rates was incredibly similar, in part due to the technical error mentioned in 4.1. In fact, the difference between a sample rate of 32kHz and 44.1kHz was subtle enough that it encouraged confirmation bias This lead to the belief that the samples at higher rates were even more subtly different, though after analysis, it was clear that they are not. The 8kHz sample rate stood in stark contrast to the other samples, causing the music to lack the clarity of its top-end frequencies. This was especially pronounced in reproduction of sibilance, the sound of air moving to form sounds like the "s" in sound, "f" in wildflower, or "v" in evening.

4.3 Phase 2: Filtering

In phase 2, the earlier-specified audio filter implementations on the DE-10 FPGA board were explored. Analysis of these filters highlighted the basic qualities of each type of filter and some anomalies of this implementation.

The README file for the repository on GitHub mentioned that the bandpass filter includes a volume boost on the FPGA. This seemed odd, as a traditional Butterworth or Bessel filter would have a configurable cutoff frequency, which would generally be set in such a way that this type of boost should not be necessary. For example, one could configure a Butterworth lowpass filter at 3kHz and a highpass filter around 600Hz to isolate the midrange frequencies, resulting in a telephone-like effect. However, though this program did not have an apparent user-configurable filter frequency, it did follow traditional behavior of a common filter.

Analysis of the data reflects a first-order Butterworth filter. For instance, the highpass filter on pink noise (Fig. 5) drops amplitude by 3dB near 3kHz compared to the bypass signal, making this a candidate for the cutoff frequency of the filter. Around a decade away on the graph, close to 350Hz, the frequency response is at -20dB compared to the baseline. In addition, at 1500Hz, one octave below the estimated crossover, the frequency response falls near 6-7dB below the baseline curve. Both assertions are examples of the expected behavior for a first order filter.

Similarly, the lowpass filter of the white noise curve (Fig. 4) has an apparent cutoff frequency just over 500Hz. One octave above, the cut should be -6dB compared to the baseline, which occurs around 1100Hz. Also, above 500Hz by one decade is 5kHz, which should be (and is) near the -20dB mark. Once again, it seems the programmer has implemented a first-order Butterworth filter.

It is also noted in the README that the bandpass is configured by running audio through both filters before output. Based on this, it seems obvious that the reason so much signal is lost in the bandpass filter is the placement of the two cutoff frequencies. By setting the cutoff for the highpass to 350Hz, a lower frequency than the lowpass around 500Hz, at least 6dB of gain reduction is guaranteed at any frequency in the signal! This gain application is apparent in the higher noise floor of the bandpass-filtered data in the 440Hz test tone (Fig. 6). On the range (800Hz, 16kHz) where the bypass and highpass signal[2] have a noise floor near -72dB, the bandpass signal has a noise floor closer to -65dB, suggesting the chosen boost is close to 7dB. The "Source" line is also included in this figure to check the possibility that this noise floor was baked into the test tone file. Based on the results of that analysis, it appears that all measured noise in the signal was generated between the output jack on the iPhone and the boost occurring in Audacity.

5 Conclusions Drawn

This research demonstrates some of the fundamentals of audio signal processing. The Nyquist rate is supported by the sample rate comparison data collected, and the filtering data demonstrates a common filtering algorithm. Implementation of this circuitry on an FPGA proved a versatile choice compared to a microcontroller, such as an Arduino, which was initially believed to be a very similar machine.

One major challenge when working on this project was understanding how to interface with the DE-10. After multiple unsuccessful attempts to work with the unit through Elementary OS, a Linux distribution, the switch was made to work in a Windows 10 environment. This, unfortunately, led to the mistake outlined in section 4.1 due to experimenter unfamiliarity with the non-Unix ecosystem.

Some follow-up research based on this project could be extension of the starter audio filtering GitHub repository to include a few new features, such as a user-controllable cutoff frequency, higher-order Butterworth filters, or different filtering curves entirely. It could also be beneficial to repeat phase 1 of the project in a more familiar software ecosystem or with a dedicated audio recorder to avoid the 44100Hz downsampling issue, though the result of such a procedure would be somewhat predictable.

5.1 Implications on Commercial Systems

Commercial mixing consoles, such as the dLive C3500 used to generate the test signals for this project would be likely to have hundreds of circuits dedicated to the functions implemented during the experiment on the DE-10. While a system like the DE-10 could be used to configure the layout for an individual channel strip or bus processor, commercial machines have far more niche processing power, dedicated circuitry, and I/O than the FPGA used in this project. For example, the block diagram (Fig. 7) of the C3500 console features 449 filter units, which can each either operate as a bandpass, lowpass, or highpass with four different options for filter shape and completely independent cutoff frequencies. As such, it appears that a console like that takes these and more fundamentals, then repeats them ad nauseum.

2 It appears there may have been an error with the analysis of the 440Hz lowpass signal. It would be expected that the same noise floor would apply to the highpass, lowpass, and bypass data.

18. Block diagram

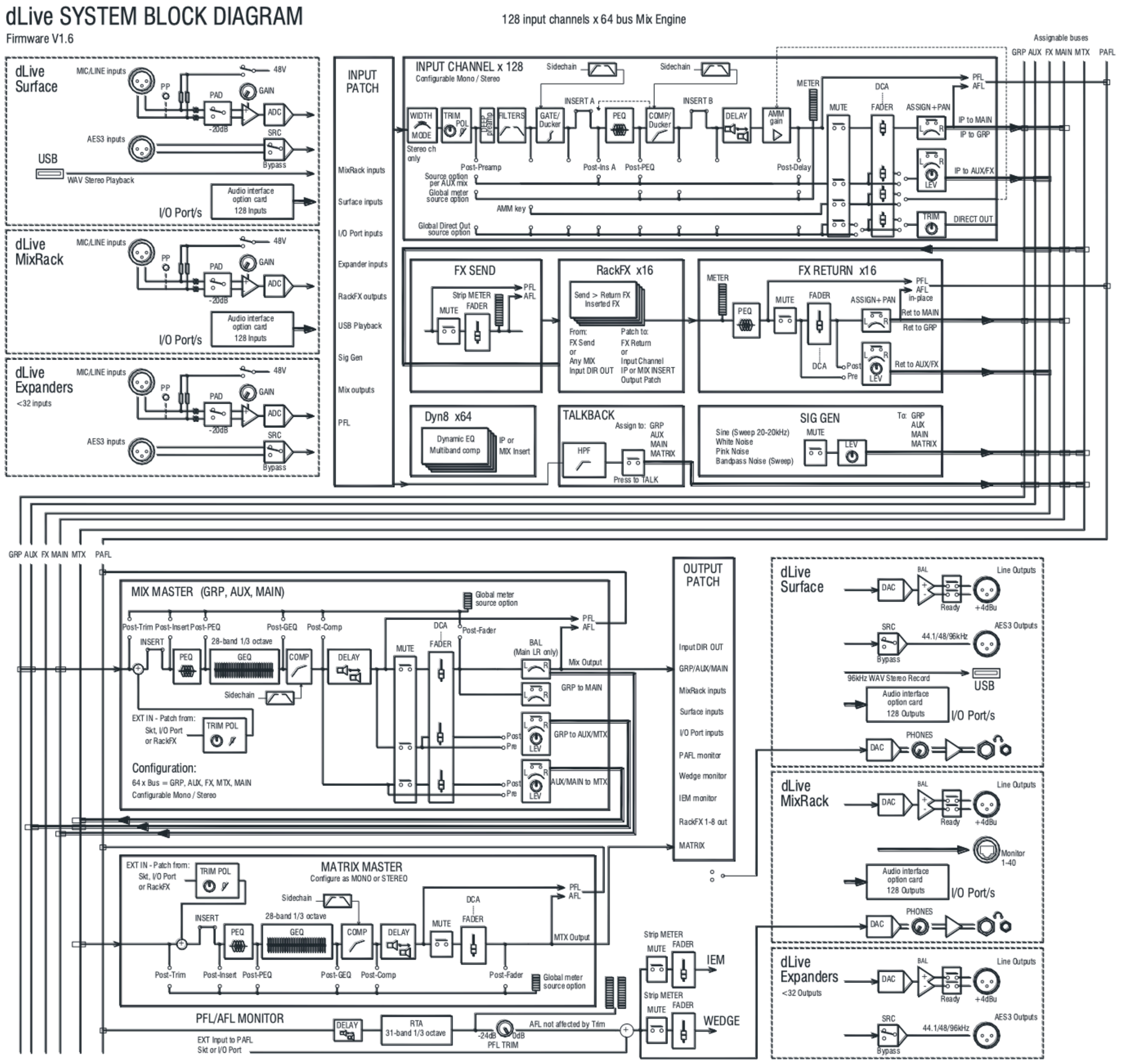

Fig.7: Allen&Heath dLive C3500 Block Diagram

ACKNOWLEDGMENTS

Major thanks are due to Dr. William Jones for lending the FPGA used in this research and providing direction in its foundational stages.

REFERENCES

[1] Terasic Inc. 2003, *DE10-Standard User Manual*.
[2] Austyn Larkin. 2018, *Digital Filters and Audio Effects on the DE2-115 FPGA*. https://github.com/Reenforcements/VerilogDE2115AudioFilters
[3] Allen & Heath. 2019, *dLive Firmware Reference Guide*. https://www.allen-heath.com/media/dLive-Firmware-Reference-Guide-V1.8.pdf