Spring 2023

On the Fly Audio Processing for the Vocal Conditioning Unit

Tim Lester
University of Maine - Main, tfrancislester@gmail.com

ON THE FLY AUDIO PROCESSING

FOR THE VOCAL CONDITIONING UNIT

by

Tim Lester

A Thesis Submitted in Partial Fulfillment
of the Requirements for a Degree with Honors
(Computer Engineering)

The Honors College

University of Maine

May 2023

Advisory Committee:

Vincent M. Weaver, PhD., Associate Professor of Electrical and Computer Engineering, Advisor
Nuri W. Emanetoglu, PhD., Associate Professor of Electrical and Computer Engineering
Andrew K. Sheaff

ABSTRACT

The Vocal Conditioning Unit was a device designed, constructed, and programmed as a senior design project in Electrical and Computer Engineering by Tim Lester and Grady White. The device's intended goal was to perform a role similar to Auto-Tune, but as a standalone device similar to effects pedals used by guitarists and other musicians on stage. On-the-fly audio processing, however, was deprioritized in the design of the original device due to other design considerations. In this thesis project, the original design of the Vocal Conditioning Unit is analyzed, and critical functionalities of the device are identified. Then, the device is redesigned to perform on-the-fly audio processing using a lower end microprocessor. Finally, comparisons are made between the original design and the new design, with consideration given to the issues still existing with both designs. Several possible solutions to on-the-fly processing for the Vocal Conditioning Unit were found, and all solutions technically perform pitch detection and adjustment. However, audio quality issues exist in all solutions.

ACKNOWLEDGEMENTS

Grady White – For his contributions to the Vocal Conditioning Unit project, as well as his help throughout our time as undergraduates.

Edmund Paquin – His enthusiasm for music and tomfoolery inspired the Vocal Conditioning Unit.

STMicroelectronics – For providing the low-level hardware interfaces for their microprocessors and microprocessor kits.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

The Vocal Conditioning Unit (VCU) was a senior design project undertaken by Tim Lester and Grady White in the department of Electrical and Computer Engineering at the University of Maine, beginning in the Spring of 2022. The device was intended to be used by a singer for experiencing pitch correction of their voice in a similar manner to Auto-Tune, but with a standalone gadget. During the research phase of the project, several assumptions and decisions were made which significantly limited the design of the VCU. The most notable limitation, which is the primary subject of this thesis project, was the lack of the ability to achieve on-the-fly audio processing for the VCU. Due to decisions which will be discussed in depth in the methodology section, the first design for the VCU only allows the singer to hear their pitch corrected performance asynchronously. That is, instead of acting like an effects pedal such as might be used during a live performance, the VCU acts more like a tool one might use in audio editing software such as a digital audio workstation. Thus, the aim of this thesis project is to redesign aspects of the VCU to allow for on-the-fly audio processing. Redesigning such a device requires enumeration and analysis of the critical functionalities of the original device, determining which critical functionalities to preserve and which new functionalities to introduce in satisfaction of the new design requirements, then redesigning, building, and testing the new device. Finally, the two designs will be analyzed against each other for similarities, differences, and drawbacks which affect both devices.

The original design of the VCU included two major sections, the pitch correcting hardware and software, which was an STM32H7B3I discovery kit with custom software, and an audio compressor which would act an input to the microprocessor. For the purposes of this thesis project, only the pitch correcting hardware and software is being redesigned as most of the design for the compressor was conducted by Grady White and is not relevant to on-the-fly audio processing. Further discussion of this will occur in the methodology section in regard to the critical functionalities of the VCU.

This thesis paper is divided into several sections. First, literature relevant to the VCU will be reviewed and discussed to provide important technical background and context. Then, methods used in redesigning the VCU for on-the-fly processing will be discussed. Results from the redesign process will be presented and analyzed. Finally, some concluding remarks will be given to provide a thorough summary of the VCU project overall.

LITERATURE REVIEW

As early as 1966, signal processing researchers have been investigating methods and designing systems for vocal audio manipulation. In their fundamental work on the phase vocoder, researchers J. L. Flanagan and R. M. Golden of Bell Labs produced some of the first synthetic audio signals using time-frequency analysis and synthesis [1]. Utilizing the principle that a speech signal does not degrade when passed through contiguous bandpass filters and recombined, Flanagan and Golden were able to modify the contents of each of these filtered signals using short term phase and amplitude information. The modifications made to speech signals by these researchers were principally time-scale modifications, the speeding up and slowing down of speech. The VCU fundamentally uses the same techniques as the original phase vocoder, but generally avoids time-scale modification of signals due to the degradation in quality caused when considering the purpose of pitch correction in vocal performance. Further research conducted by Bell Labs researcher Jont Allen showed that some aspects of Flanagan and Golden's phase vocoder technique were lacking. Particularly, Allen showed that using contiguous bandpass filters is not sufficient for signal synthesis. Instead, to avoid introducing unexpected distortions into the synthesized signal, the frequency bands analyzed must overlap by some amount [2]. In other words, Allen showed that using contiguous frequency bands (referred to as 0% overlap in the context of this paper) leaves the signal undersampled in both the frequency and time domains.

The works of Flanagan and Golden were synthesized by future researchers in their explanations and investigations of short time Fourier transforms, also known as STFTs for short. In order to understand design decisions made involving the STFTs and phase

vocoder techniques used in this paper, the following discussion of STFTs are provided with reference to the works of J. Allen [2], E. Jacobson and R. Lyons [3] as well as J. Laroche and M. Dolson [4]. As given in [2], the following is a continuous-time definition of the STFT:

Equation 1. Continuous-time short time Fourier transform.

$$X(f,t) = \int_{-\infty}^{\infty} w(t-\tau)x(\tau)e^{j2\pi f\tau}d\tau$$

The function w(t-τ) is the shifted windowing function and x(τ) is the real audio signal. For digital signal processing applications like the VCU, both the windowing function and real audio signal will be sampled discrete-time functions. Further, while Equation 1 describes the Fourier transform of x(τ), discrete-time signals would be analyzed using the discrete Fourier transform (DFT), implemented as a fast Fourier transform (FFT). Given a particular windowing function, one can determine the appropriate sampling parameters for the STFT. For example, J. Allen finds that the amount of time domain samples to take while using a Hamming window is exactly equal to the time-limit T for the window, that is the amount of time for which the window is non-zero. Further, taking the Fourier transform of the windowing function and determining its approximate bandwidth F *in relation to the window's time-limit T* we can find an appropriate overlap percentage between adjacent STFT frames. Again, using the example of a Hamming window, its bandwidth F is approximately 4/T. The reciprocal of this bandwidth, T/4, gives the distance in samples in the time domain between STFT frames, which leads to the conclusion that while using a Hamming window, each frame should have a 75% overlap (T – T/4 = 3*T/4, i.e., 75% of the samples from the previous window). J. Allen also gives

the following as a discrete-time definition for the STFT, where *n* are indices in time and *m* are indices in frequency:

Equation 2. Discrete-time definition for the short time Fourier transform.

$$X_{nm} = X\left(\frac{m}{T}, \frac{n}{F}\right) = \sum_{k=0}^{T-1} w\left(\frac{n}{F} - k\right) x(k) e^{j2km/T}$$

E. Jacobson and R. Lyons describe a generalized case of the STFT which they refer to as the sliding discrete Fourier transform (SDFT), which assumes an overlap of all but one sample between frames [3]. The VCU does not use the advanced techniques presented in this article. Instead, the VCU specifically uses techniques identified as relatively computationally inefficient by Jacobson and Lyons. Although performing an SDFT provides incredible time resolution (frames are separated by as little as the sampling period) and is computationally efficient for long sequences of data, obtaining sufficient frequency resolution for the task of pitch detection and correction is space inefficient with these techniques. Given a sampling frequency F and FFT length N, the width of each frequency bin in the resulting SDFT (or STFT for that matter) is F/N. With a low sampling rate of 8820 samples per second, an FFT length of 2048, and signal time of 10 seconds, that resolution would be 4.31 Hz, but the amount of data in the SDFT would be (88200 samples * 2048 bins/sample) = 180,633,600. With 16 bits per sample, this SDFT would require 361 Megabytes of space! Given that a major constraint on the original VCU design was memory, it seems reasonable to avoid excessively overlapping FFT frames despite the computational advantages offered. Instead, the STFTs implemented in the VCU must recalculate the entire FFT for each frame as they come rather than taking the previous FFT and modifying it for the next frame using the circular shifting property described in [3].

Jean Laroche and Mark Dolson completed several papers on the topic of the phase vocoder in the late 1990s, greatly expanding on the work of Flanagan and Golden. Of note is a 1999 paper in which the pair describe solutions to a well-known problem with the results of the phase vocoder [4]. They refer to this well-known problem as "phasiness," which is a distortion often described as "robotic" and is likely the same artifact which people associate with pitch correction devices. Laroche and Dolson prove the existence of this artifact and then describe how to correct it when performing frequency-domain modification of audio signals using the phase vocoder. Though there are circumstances in which the so-called phasiness can be avoided without such modifications, the VCU does not encounter these exceptions. Additionally, the VCU does not attempt phase unwrapping or any other techniques to correct the phasiness. Hence, the quality of the resulting audio is subjectively poor at times, for the reasons shown in [4]. Another paper from Laroche and Dolson in 1999 describes the importance of the STFT frame overlap percentage with regards to pitch adjustment in particular, reiterating the inevitability of phasiness when performing such a task [5]. As the VCU was designed to perform pitch adjustment, one might expect its algorithms to be able to adjust the pitch of a performed note to an arbitrary frequency under the Nyquist frequency of the system. Laroche and Dolson show that adjustments by integer factors, such as doubling or tripling the frequency, or halving the frequency, can be performed using a phase vocoder with as little as 50% overlap between STFT frames. From a music theory perspective, it could be said that accurate pitch adjustments under an octave are impossible with 50% overlap. Considering the VCU is meant for vocalists who are only slightly out of tune, pitch adjustments under an octave are essential. Fortunately, Laroche and Dolson showed that a

75% overlap, just as described by J. Allen, allows for what they call “fractional shifts.” That being said, it appears that these fractional shifts introduce more phasiness into the resultant signal, which the VCU does not correct for.

While the phase vocoder is a method for adjusting the pitch of notes given an audio signal’s STFT, it requires further input to function. Importantly, accurate pitch adjustment requires accurate pitch detection. Several researchers have constructed and analyzed systems and algorithms for pitch detection of speech signals [6, 7, 8]. Many of the earliest techniques, such as the auto-correlation function and the average magnitude difference function, use time-domain methods to calculate the likely frequency of a speech signal. Specifically, many early techniques utilize the fact that periodic signals will periodically cross through zero to determine the frequency [6]. However, it has been shown that time domain techniques are not only susceptible to octave errors and noise [7], they are also computationally inefficient compared to certain frequency domain techniques [8]. The harmonic product spectrum (HPS) detects pitch in speech signals by taking advantage of the fact that most pitched sounds include several harmonics which are spaced predictably in the frequency domain. The HPS is found by taking a short window of an audio signal, finding its DFT, then downsampling the DFT at least once by a factor of 2. Then, the full DFT should be truncated to the length of the downsampled DFT. After this, the two are multiplied elementwise. The highest magnitude value of the resulting array (i.e., the argmax of the array) should be at the frequency of the pitch performed in that window. It should be noted that the simplest implementations of HPS work on monophonic (single voice) audio only. Fortunately, the VCU was designed for a single performer so monophonic techniques are acceptable.

By far the most widely known pitch correction tool today is Auto-Tune, produced by Antares Audio Technology. While the specifics of the pitch detection and correction techniques used by Auto-Tune are mostly unknown to the public, how it is used and its impact are well understood. Originally, Auto-Tune was primarily used to fix slight tuning issues of recorded vocals post-recording to improve the quality of vocals for album releases and for radio. At times, use of these techniques has been considered controversial [9] as some view pitch correction as a crutch for poor vocal performances. Regardless of this controversy, Auto-Tune has stayed in continuous use since 1997. Modern versions of the application allow for correction of live performances on-the-fly, much like the intended goal of this project for the VCU. It is worth noting, however, that Auto-Tune is typically used in a digital audio workstation on computationally powerful desktop computers with multiple processor cores. The VCU uses a less powerful microprocessor and makes use of only one processing core. The distinctions between Auto-Tune and the VCU have implications for the feasibility and resulting quality of the on-the-fly algorithms for the VCU, which will be discussed in depth in the results section.

METHODOLOGY

Redesigning systems like the VCU requires several steps. First are the steps which help an engineer understand their original design. In the context of the VCU, this includes enumeration and analysis of the critical functionalities of the device. In order to successfully redesign the VCU, it is important to understand what the original device's purpose was and which design decisions led to the successful (or perhaps unsuccessful) execution of that purpose. Next are the steps which help the engineer understand how the original design will relate to the new design. These steps include the enumeration and analysis of the critical functionalities of the new device, as well as some analysis of how to preserve or discard functionalities from the original design. Then, the new device should be built and tested to verify that it achieves its purpose. Finally, the two designs should be compared against each other to verify the design rationale for each device as well as their similarities, differences and any advantages or disadvantages one design has over the other.

Analysis of the VCU

Some external documents will be referenced in order to enumerate the critical functionalities of the original VCU design. The original contract for the VCU drafted in ECE 405 (the first class of the design project sequence for Computer Engineering majors) will be referenced, as well as the software code associated with the device. According to the original contract, the pitch adjustment section of the VCU was expected to "receive audio from an analog input, [perform pitch detection and adjustment on] it and output an analog audio signal with corrected pitches." It is also indicated that the device should be capable of storing the corrected audio signal in a digital format. Finally, the user should

be able to control the device and should expect feedback from the device in the form of indicator LEDs for recording status.

The stated requirements for the original VCU design can be analyzed as a set of critical functionalities which can be connected to decisions made when designing the VCU. The simplest of such decisions include determining that an analog-to-digital converter (ADC), digital-to-analog converter (DAC) and general-purpose input/output ports (GPIO) were all necessary on the selected microprocessor for the VCU due to the requirement of receiving analog audio, transmitting analog audio, and handling user input and output. It is common for microprocessors to have all of these peripherals on board anyway, and an ADC is the best way to sample an analog signal for signal processing purposes. Further, while other options such as pulse-width modulation allow for transmitting an analog audio signal from a digital format, a DAC is preferred due to the relative quality of the resulting signal.

Requiring the VCU to digitally store the corrected audio signal introduced some complexity into the design. Several options exist for storing digital data, but the first decision made was to avoid storing the audio on any media directly attached to the microprocessor running the VCU software. While many microprocessors can have devices such as Secure Digital (SD) cards or even hard disk drives (HDDs) attached for storing digital data, the VCU team wanted to avoid working with files on a system lacking an operating system. This leaves two major options for storing the data, both of which require media on a remote device: over some sort of networking protocol, or over a dedicated digital transfer protocol. Networking protocols were decided against as this may require a complex networking stack on the VCU processor, which is beyond the

scope of the VCU. Ultimately, universal asynchronous receiver/transmitter (UART) was chosen to store the corrected audio signal on a desktop computer physically connected to the VCU. One reason for this decision was that the team already had a software program developed for desktop computers to receive serial data such as that transmitted with UART. However, selecting the method for storing the digital data was not the only complexity introduced with digital storage. If the VCU is required to store, for example, 10 seconds of audio sampled at 44.1 ksps, assuming a bit-depth of 32 bits for each sample, this would require at least 1.7 Megabytes of storage for the corrected signal. Since the decision was made to not attach hard storage to the VCU, the only other place these samples can be stored is in memory. While some microprocessors have an excess of 1.7 MBs of RAM on board, it is more likely that an external memory unit of some sort is necessary to hold on to this many samples at once. This leads to the final hardware requirement introduced by the critical functionalities of the VCU, which is the need for some sort of external memory controller and external memory module. Table 1 provides a summary of the critical functionalities of the original VCU, and the hardware required due to these functions.

Table 1. Enumerated critical functionalities of the original VCU.

CRITICAL FUNCTIONALITY	REQUIRED HARDWARE	NOTES
PITCH DETECTION AND ADJUSTMENT	Floating-point unit (FPU) External memory & controller	External memory required to store audio signal awaiting processing.
RECEIVE ANALOG AUDIO	Analog to digital converter (ADC)	
OUTPUT ANALOG AUDIO	Digital to analog converter (DAC)	
DIGITAL STORAGE OF CORRECTED AUDIO SIGNAL	Some sort of connectivity (universal asynchronous receiver/transmitter was chosen) External memory & controller	External memory required to store corrected audio awaiting digital transfer.
USER INPUT/OUTPUT	General purpose input/output (GPIO)	

The design and operation of the algorithms used to acquire audio samples and perform pitch detection and correction were impacted by the requirements introduced by the previously enumerated critical functionalities. Due to the fact that digital storage already required holding several seconds of contiguous audio samples in memory, there was no need to investigate on-the-fly processing in the original VCU design. Instead, once the desired number of samples were collected, the entire audio signal was processed, and the processed signal also stored in memory for streaming or storing digitally. The following is a description of the operation of the original VCU:

1. Upon start up, perform hardware and software initialization.
2. Wait for user input. Upon user input, DMA is enabled for the ADC, filling a temporary buffer of length ADC_BUF_LEN at the sampling rate and moving those samples into another buffer of a length which is an integer multiple of ADC_BUF_LEN. Continue until the larger buffer is filled.

3. Wait for user input. Upon user input, the large buffer of audio samples is processed. First, an STFT is performed, then passed to the phase vocoder pitch detection and adjustment algorithm for modification, which is then passed to the inverse STFT.
4. Wait for user input. Depending on user input, either enable DMA for the DAC and stream the large buffer out, or attempt to transmit the buffer via UART to an attached desktop computer. Return to step 2 when finished regardless.

While processing occurs quickly, the corrected audio is not calculated as samples are being taken. Instead, parameters for the algorithms are calculated based on the full length of audio. This means that the resultant corrected audio signal is contiguous and only suffers from energy loss at the beginning and end of the signal, which is not particularly noticeable due to the signal changing rapidly at the start and end anyway. In alignment with the findings in [2, 4, 5], STFTs in the VCU were performed with 75% overlap between windows, and signals were windowed with a Blackman window to allow for relatively little distortion when performing fractional pitch shifts. While a sample rate of 44.1 ksps was initially chosen due to its standard use and usefulness for sampling musical audio, the actual sampling rate for the VCU was 8820 samples per second. Since FFT resolution is given by the quotient of the sampling rate and the FFT size and the largest FFT size which can be performed with ARM's digital signal processing library is 2048, a frequency resolution of 4.31 Hz is possible. This is barely acceptable for vocal audio since notes in the 0^{th}, 1^{st} and part of the 2^{nd} standard octaves are separated by less than 4.31 Hz.

Redesigning the VCU

Design decisions for the new VCU design will be discussed. First, hardware considerations will be discussed with a focus on discarding external memory as well as selecting a cheaper microprocessor board. Then, several possible solutions will be considered for on-the-fly audio processing. These solutions include simply processing small buffers of samples as they come without any correction steps, reducing the overlap between STFT windows to eliminate energy losses, and processing audio slightly longer than the output buffer by setting aside some portion of the previous input sequence and including it in processing. The solutions will be evaluated for accuracy and quality in the results section.

Hardware

Hardware for the new VCU design was selected in accordance with the intended functionalities of the new device. While the old VCU design required digital storage which relied on abundant external memory, the new design does not. Thus, the microprocessor board chosen for the new design has far fewer overall features and is considerably cheaper than the original board. The Nucleo-G474 was selected for the new design. While both the Nucleo-G474 and the STM32H7B3I-DK have several ADCs, DACs, and GPIO pins, as well DMA, the Nucleo-G474 does not use an external memory controller to control attached external memory. Instead of using up to 16 Megabytes of memory for the recorded audio alone, only 128 kilobytes of internal memory was available on the Nucleo-G474 for all of the tasks performed by the VCU. Also of note when considering the hardware of the original and new VCU designs is the fact that both projects were developed on prototyping boards. The original design was developed on a so-called “discovery kit,” which offered few customizable input and output pins due to

the sheer number of features that STM intended to demonstrate with the board. The Nucleo-G474 on the other hand has 64 pins available, most of which can be used for several tasks. The implications of the style of prototyping board chosen, as well as the choice to use a prototyping board in the first place, will be discussed in further detail in the results section.

Algorithms

The primary motivation in redesigning the pitch detection and correction algorithms of the VCU is performing on-the-fly pitch detection and correction. As described previously, the underlying algorithm for pitch detection and correction in the VCU is the STFT. In order to perform accurate pitch detection, the FFTs taken for the STFT must have sufficient resolution to differentiate pitches. Simultaneously, accurately reconstructing a time-domain signal from an STFT using an inverse STFT necessitates supplementing the first and last windows of the STFT due to the loss in energy in those windows. Regardless, it should be possible to perform on-the-fly pitch detection and adjustment using STFTs by adjusting the sample acquisition process and other parameters to the algorithms. The first and simplest approach involves disregarding the fact that energy losses occur at the ends of signals being processed with an STFT. The following steps describe the simple approach:

1. Acquire K samples where K is some integer multiple of the FFT size.
2. Process K samples using a 75% overlap, immediately placing the samples into the output buffer to be streamed out.

The expected result for this approach is an audio signal where the pitch has been adjusted accurately, but the first and last (FFT size)/2 samples in every K sample block will be inaccurate, having lower magnitudes than they should. This should result in a less than

enjoyable listening experience since the periodic drops in energy are expected to produce a periodic beating in the resulting signal.

Another approach disregards the findings of J. Allen [2] and Laroche and Dolson [5] by choosing an overlap percentage of 0%. While Laroche and Dolson found that the only way to perform highly accurate and undistorted fractional pitch shifts in a phase vocoder required at least 75% overlap between windows, this does not mean fractional pitch shifts are impossible with no overlap. Further, the only reason energy losses can be expected at the ends of signals analyzed with an STFT is due to the overlap between adjacent windows. Since some parts of the signal show up in fewer windows than others, they will have a lesser impact on the resulting signal. However, with an overlap of 0%, every sample in the original signal is only found in one frame of the STFT, eliminating the expected periodic beating described for the previous solution. The expectation for this solution is that the signal will be contiguous with no periodic losses in energy, but it may not perform pitch adjustment with great accuracy. Further, the "phasiness" described by Laroche and Dolson should be evident in the resulting signal.

The final approach is the most complex and seeks to utilize a non-zero overlap percentage while still avoiding the periodic losses in energy associated with STFTs. By keeping a number of samples from time steps in the past and introducing a delay of a few hundred samples, processing can be performed such that every output buffer produced has several samples which were already streamed out in the previous iteration of processing. Streaming from part way through the output buffer should avoid the energy losses from the STFT. This solution does waste some computational power since some number of samples in each processing block were already processed before. The expected

result from this approach is a signal without periodic losses in energy and with greater accuracy in pitch adjustment than the previous approach. Further, it should subjectively sound better than the previous approach due to the overlap between windows reducing distortions introduced by the STFT analysis and synthesis process.

Testing the New Algorithm Designs

To test the various solutions to on-the-fly audio processing for the VCU, a simple testing procedure was developed. The testing procedure needed to be easily repeatable and compare the VCU's response to known signals between the various solutions. The procedure should also work on both the original VCU design and the new design, with some modifications due to differences between the two devices. While the original device did not perform on-the-fly processing, it is important to compare the output of the various on-the-fly algorithms to the output of the original device, as successful on-the-fly processing should produce roughly the same output as the original device. The primary measurement equipment used in testing is the Digilent Analog Discovery 2 (DAD2), a portable multitool for testing analog and digital systems alike. Important for this testing procedure is the availability of oscilloscopes and arbitrary function generators on the DAD2. Interfacing with the DAD2 was performed using the WaveForms software developed by Digilent, which communicates with the DAD2 over USB. The following steps describe the testing procedure:

1. Attach the output of the DAD2's wave generator to the input of the VCU's ADC.
2. Attach the output of the VCU's DAC to oscilloscope channel 1 on the DAD2. To hear the audio during testing, the output of the DAC should also be routed to speakers with a 3.5mm jack breakout or another connector.

3. The VCU program which allows for switching between the on-the-fly implementations using the user button is compiled and the VCU board is programmed with this using the STLINK interface.
4. Load pre-recorded vocal audio files into WaveForms, which can then be played into the VCU's ADC using the DAD2's wave generator channel 1. Ensure appropriate voltage levels (between 0V and 3V) for the audio, as audio which is too loud or too quiet will not process properly.
5. Channel 1 of the oscilloscope provides the output of the VCU. Perform a spectrogram over 5 seconds on the output using WaveForms spectrogram feature. These spectrograms will be compared to spectrograms of the original audio.
6. To obtain spectrograms of the original audio, attach oscilloscope channel 1 to wave generator channel 1 and perform a spectrogram as in step 5.
7. Next, attach oscilloscope channel 1 to the first digital timing pin on the VCU, and oscilloscope channel 2 to the second digital timing pin.
8. The first digital timing pin will go high when the first call to the sample acquisition function occurs for each solution, and low after that call has returned.
9. The second digital timing pin will go high when processing has begun (i.e., when the STFT function is called) and low when processing has finished (i.e., after the inverse STFT function has returned.)

Steps 1-6 of these procedures were repeated for a 203 Hz sung tone as well as a 250 Hz square wave for all 3 on-the-fly solutions as well as on the original VCU. The first signal

was chosen for its relevance to the project. After all, the design is for the *vocal* conditioning unit, so it is only appropriate to perform testing on vocal audio. A square wave was also chosen due to its relative simplicity yet high susceptibility to sound dramatically different when its spectra are modified slightly. Some of the spectrograms as well as other measurements will be presented in the results section to compare the several solutions.

RESULTS

The testing procedures discussed previously were performed on the original VCU as well as the three solutions in the new design of the VCU. Figure 1 shows an oscilloscope trace of the original 203 Hz sung audio signal being used to test the VCU solutions. The audio signal is roughly 2.5 seconds long and repeats indefinitely while the wave generator is enabled.

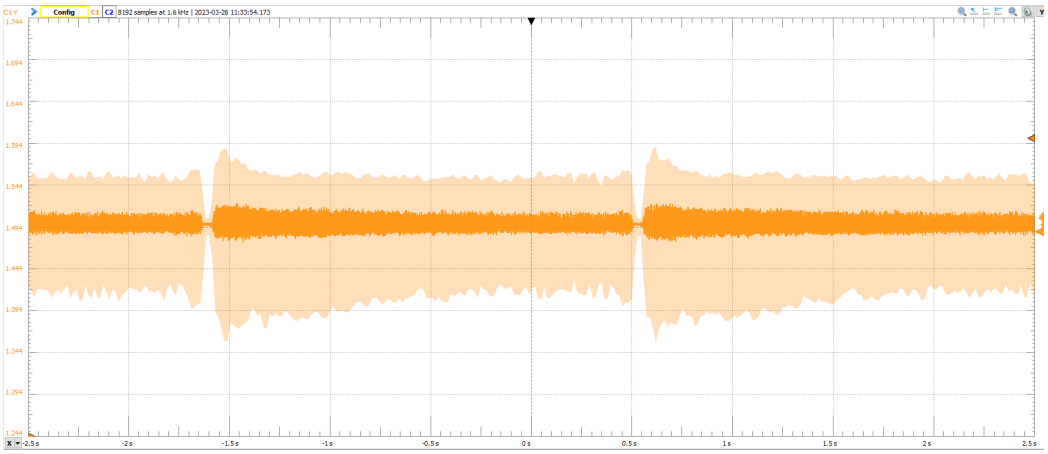

Figure 1. Oscilloscope trace of the 203 Hz sung tone, output by the DAD2's wave generator.

A spectrogram of the same audio signal is provided in Figure 2. Note that time in each figure is not equivalent. Since this is a 203 Hz sung tone, it is expected that some energy is contained around 200 Hz for most of the time the signal is active. As well, substantial energy exists around 400 Hz, which would be the first major harmonic beyond the fundamental frequency of the note. Since this is a human voice and is imperfect (i.e., inharmonic), lots of energy is also present at frequencies near the performed frequency.

Since the VCU's primary purpose is to adjust the pitch of the performed audio, only a narrow band of frequencies near the performed pitch are shown in spectrograms in this section. Preferably, high frequency harmonics would also be adjusted properly by the VCU, but issues with this will be discussed later in this section.

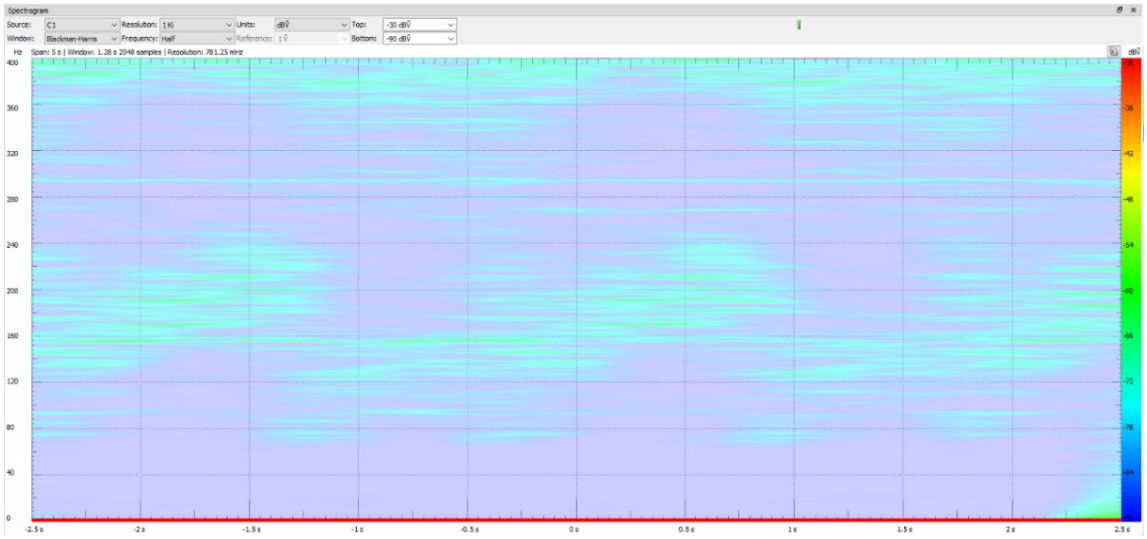

Figure 2. Spectrogram of the 203 Hz sung tone, taken with the DAD2's spectrogram function.

Testing on Vocal Audio

Overall, the measurable performance of the two simpler solutions in the new design was better than expected. The subjective quality of solution 3 was worse than expected, but solution 2 was much better than expected. Table 2 summarizes the performance of the various solutions to on-the-fly processing for the VCU. Note that two frequencies listed for one solution in the "post-processing frequency" row means that the spectrogram indicated there was substantial amounts of energy at both frequencies over the entire spectrogram window.

*Table 2. Comparison of VCU solutions on a 203 Hz sung tone. Expected post-processed frequency is 207.65 Hz (G#3). *The original VCU collected samples in a drastically different way than the new design, so these times are misleading.*

	Original VCU	Solution 1 (naïve)	Solution 2 (No overlap)	Solution 3 (Just in time)
Approximate Post-processing frequency (Hz)	206	208	206 and 210	206
Time between first sample acquisitions (msec)	>10000*	58.133	261.39	173.86
Processing time (msec)	131.160	51.199	59.053	64.448
Percentage of total time spent processing	<2%*	88.1%	22.6%	37.1%
Relative subjective quality ranking (1-4)	3	4	1	2

Table 2 shows that for vocal audio, the VCU generally shifts the fundamental frequency of the performed audio towards the nearest 12-TET note, regardless of the processing algorithm architecture being used. However, it is worth noting that solution 1 has incredibly poor quality overall. The oscilloscope trace in Figure 3 shows the audio output from the VCU using solution 1. Note that for just under half of the time, the output voltage is at 0V and does not change over time as the input signal does.

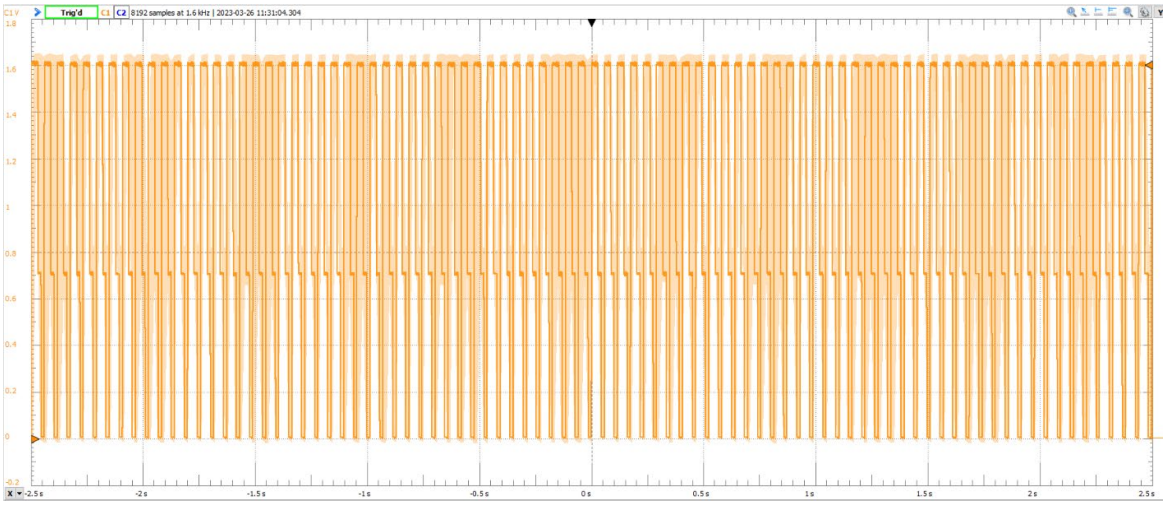

Figure 3. Oscilloscope trace of solution 1 output (203 Hz sung tone).

The time domain audio signal for solution 1 clearly illustrates the issues with this solution. This is reinforced further by the spectrogram of this audio shown in Figure 4. The small magnitude content indicated with blue-green is the modified sung tone, and has clearly moved to about 208 Hz. However, most of the signal content shown in this spectrogram can be attributed to the fact that the signal is nearly a square wave due to half of the samples being zero instead of corresponding to the original signal in some way.

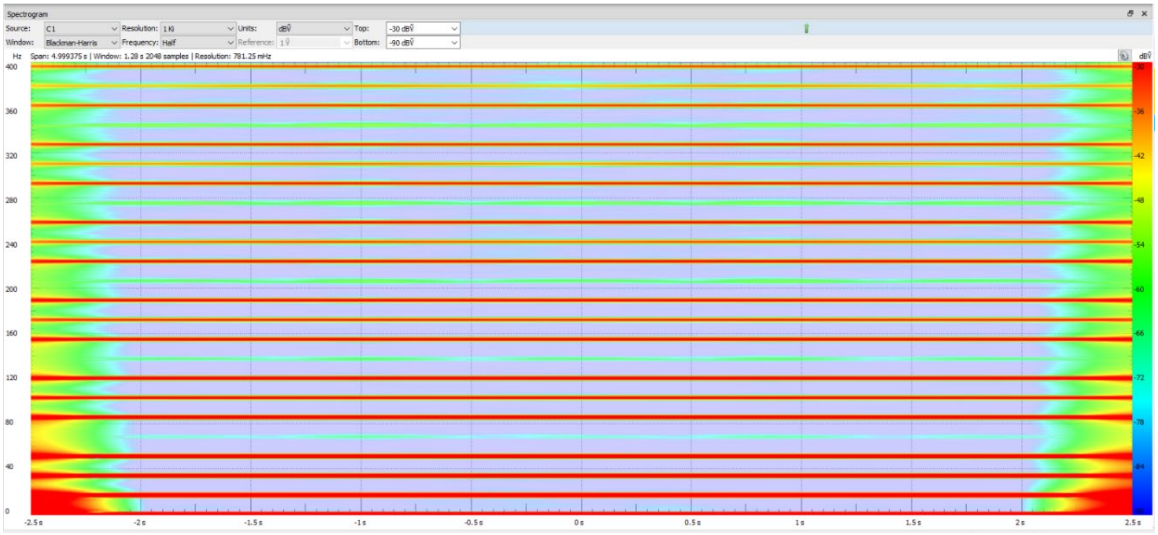

Figure 4. Spectrogram of solution 1 output (203 Hz sung tone.)

While solution 2 also successfully moved the fundamental frequency closer to the nearest 12-TET note, it appears to have done so inconsistently. The spectrogram shown in Figure 5 shows that the resulting audio signal from solution 2 has substantial energy around both 206 Hz and 210 Hz for all times where there is any energy near 200 Hz. It is also notable that, compared to the spectrogram of the original audio given in Figure 2 there seems to be a periodic attenuation of the signal. That is, instead of the signal remaining at roughly the same magnitude throughout the duration of the 5 second window, the signal has a low magnitude from -1.5 seconds to -0.5 seconds as well as

from 0.5 seconds to 1.5 seconds. Note that the inharmonic content introduced due to the nature of the human voice can still be observed in this spectrogram, though like the rest of the signal content its frequency has been modified by the vocoder algorithm.

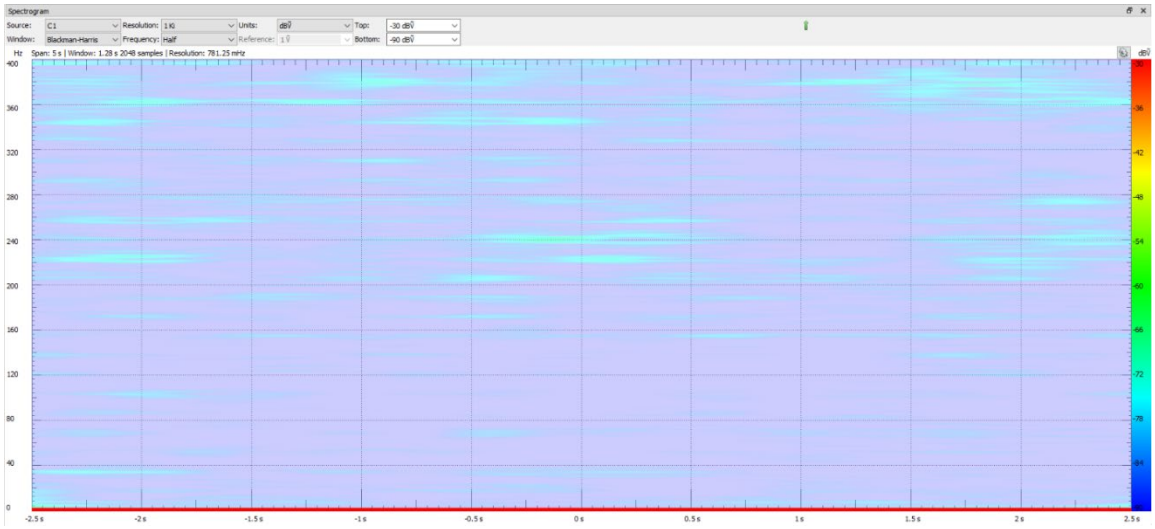

Figure 5. Spectrogram of solution 2 output (203 Hz sung tone).

Solution 3 also successfully moves the fundamental frequency towards the nearest 12-TET note. It was expected to have the best subjective quality of the solutions, due to the fact that it not only guarantees no missing samples in the output sequence, but also uses some amount of overlap to increase the accuracy of pitch adjustment. However, some issues exist with the audio produced by solution 3, which impact its quality. Figure 6 shows the spectrogram produced by solution 3 on the 203 Hz sung tone audio. While the signal at around 206 Hz for all time in this spectrogram can be attributed to accurate pitch adjustment of the original audio by the VCU, the signal content at harmonics of roughly 27 Hz cannot be directly attributed to any operation performed by the VCU. Since the magnitude of this erroneous signal content is around the same as the corrected content, it is audible within the output audio, decreasing the subject quality of this solution.

Figure 6. Spectrogram of solution 3 output (203 Hz sung tone).

Since the original VCU is the frame of reference for the performance of the various solutions to on-the-fly processing, it is important to briefly analyze the spectrogram output of the original VCU on this audio as well. Figure 7 is the spectrogram of the output of the original VCU on the 203 Hz sung tone audio in a similar frequency range to the figures for the other solutions. Just like the three on-the-fly solutions, the original VCU correctly moves the frequency content of the original audio signal towards the nearest 12-TET note. Comparing this spectrogram to each of the solutions, it clearly does not have the same square-wave-based distortions that solution 1 has. In terms of overall performance, it is most similar to solution 2 in that it correctly moves the fundamental frequency of the performed audio without introducing many distortions in the depicted frequency range. Solution 3 includes distortions at harmonics of roughly 27 Hz, which are entirely absent in the original VCU. The original VCU has a large amount of low intensity noise in the frequency ranges depicted, which differs from the on-the-fly solutions. Finally, Figure 8 depicts a slightly larger frequency range of this same spectrogram. When listening to the audio produced by the original VCU, a relatively high

frequency sinusoidal signal is obvious. Figure 8 indicates this signal is intense at 2.2 kHz and 3.6 kHz. Like the distortions at harmonics of 27 Hz in solution 3, there is no operation in the original VCU which this can be directly attributed to.

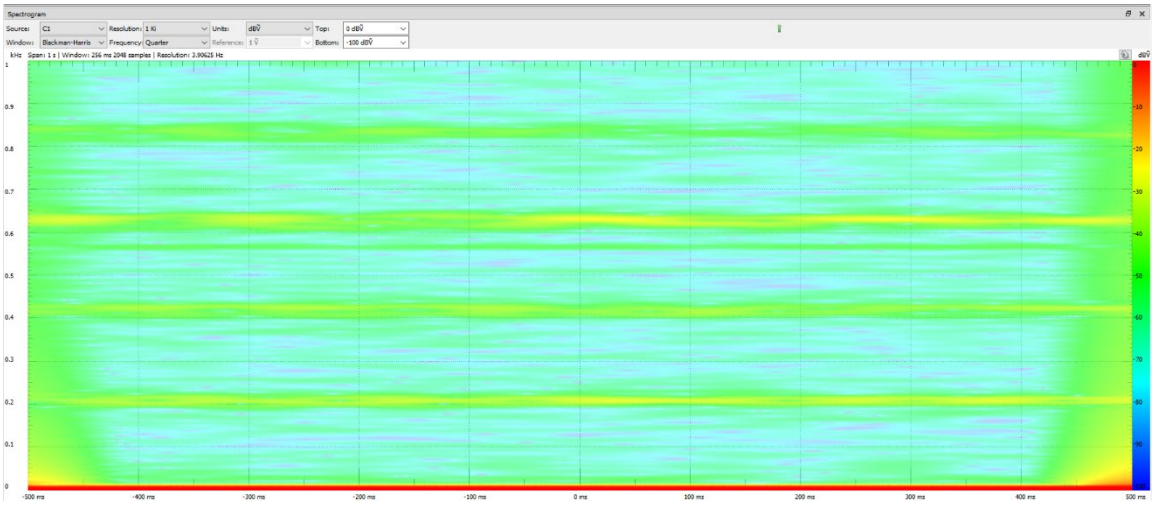

Figure 7. Spectrogram output of the original VCU, 203 Hz sung tone.

Testing on a Synthesized Square Wave

Table 3 summarizes the performance of the various solutions and the original VCU on a 250 Hz square wave. Timing analysis was not performed for this signal as computation time is mostly independent of the signal content. Unlike with the vocal audio, solution 2 failed to move the fundamental frequency of the signal towards the nearest 12-TET note, instead going in the opposite direction. However, it still adjusted the fundamental frequency to a frequency near a valid 12-TET note. What this likely indicates is an error detecting pitch in the HPS algorithm, which is expected occasionally.

Table 3. Comparison of VCU solutions on a 250 Hz square wave. Expected post-processed frequency is 246.9 Hz (B3).

	Original VCU	**Solution 1 (naïve)**	**Solution 2 (No overlap)**	**Solution 3 (Just in time)**
Approximate Post-processing frequency (Hz)	245	242	260	245

Spectrograms of the VCU output on the 250 Hz square wave audio for the various solutions are contained in the appendix. A spectrogram of the original square wave is provided for reference. The results of processing the square wave audio with the VCU solutions generally match the results found when processing vocal audio. Solution 1 correctly moves the fundamental frequency of the square wave towards the nearest 12-TET note, but due to roughly half of the samples in the output not correlating to the input, the audio quality is poor, and the spectrogram indicates strong signal content at other frequencies. Solution 2 moved the fundamental frequency towards the next highest 12-TET note, which is slightly further away than the next lowest. However, the resulting audio sounds like a square wave at that frequency with moderate distortions. That being said, comparing the original square wave's spectrogram in Figure 9 to the output of solution 2 in Figure 11, solution 2 did much more than move the signal content from 250 Hz to 260 Hz. Similar to how solution 3 introduced distortions into the vocal audio at harmonics of some frequency in the range of 20 to 30 Hz, so did solution 2 on the square wave. A pure square wave is expected to have signal content at odd harmonics only. For a square wave at the nearest 12-TET frequency of 246.9 Hz, this means signal content should appear at 246.9 Hz, 740.7 Hz, 1.23 kHz, etc. These distortions clearly indicate that the VCU's pitch adjustment algorithm struggles to preserve the quality of incoming signals. The spectrogram for solution 3 is quite similar to that for solution 2, more so than is the case for the vocal audio. While the strongest signal content is clearly around the nearest 12-TET frequency, lots of additional content appears at harmonics of a frequency in the 20 to 30 Hz range.

Analysis of Results & Discussion

Solutions 1 and 2 to on-the-fly audio processing for the Vocal Conditioning Unit generally met the expected outcomes. The overall audio quality of solution 1 is worse than expected, but it was not expected to be good either way. This solution definitely moved the fundamental frequency of the audio signal towards the nearest 12-TET note, at least whenever the signal was actually present in the output audio. Solution 2 also moved the fundamental frequency towards the nearest 12-TET note, though it suffered from occasional pitch detection errors. The audio quality of solution 2 was better than expected, as the research by Laroche and Dolson seemed to imply that no window-to-window overlap causes great distortions when attempting fractional pitch shifts. Solution 3 performed worse than expected overall. Despite properly moving the fundamental frequency of the audio signal and using a 50% window-to-window overlap, the audio quality was worse than solution 2. Assuming a proper implementation of window-to-window overlapping, this seems to be in conflict with the conclusions reached by Laroche and Dolson. This conflict could be explained by errors with the sample acquisition performed for solution 3, or possibly other underlying issues with the implementation of the vocoder pitch adjustment algorithm.

Several issues exist with the various solutions to on-the-fly audio processing for the VCU. These issues have further revealed issues with the design of the VCU overall, particularly with regards to sample acquisition methods and the implementation of the pitch adjustment algorithm. While each version of the VCU software technically adjusts the pitch of incoming audio, it is easy to conclude that the impact the VCU has on the quality of processed audio makes the device generally unusable. Unfortunately, it is unlikely that the source of the degradation of the processed audio is singular. It is likely

that both the method of acquiring samples for processing as well as the pitch adjustment step itself contribute to the poor audio quality. This conclusion can be reached by comparing the design of the original VCU to solutions 2 and 3 for on-the-fly processing. The original VCU's sample acquisition process involves collecting five seconds of contiguous samples, followed by processing all five seconds of samples. Despite the fact that the samples are guaranteed to be contiguous in time, the resulting audio is still moderately distorted and generally sounds bad. Because this sample acquisition process avoids any issues with STFT continuity, the pitch adjustment algorithm must be partly at fault. Solution 1, on the other hand, is obviously greatly affected by its sample acquisition method. Using the naïve method of processing blocks of samples as they come will introduce the periodic losses in energy associated with the STFT, causing almost half of the samples in the resulting output to have no correlation to the input signal. Solutions 2 and 3, similar to the original VCU, seem to avoid the periodic STFT energy losses, but still produce audio of poor quality. As stated previously, while the pitch adjustment algorithm is likely partly at fault, it is also likely that the sample acquisition methods are contributing. While the sample acquisition method in the original VCU avoids the common digital signal processing problem of sample overrun since processing does not begin until all samples are acquired, the on-the-fly solutions must contend with sample overrun. Unfortunately, none of these solutions avoid sample overrun entirely due to the performance of the pitch detection and adjustment algorithms. This means that while the sample acquisition methods for solutions 2 and 3 were designed with the intent of guaranteeing contiguous samples, some subset of samples will be missing anyway. Missing samples in the input means that the input audio signal as seen by the pitch

detection and adjustment algorithms is not the same signal as performed by the user of the VCU.

As noted, solutions 2 and 3 as well as the original VCU have some distortions in the frequency domain which cannot be attributed directly to any one operation in the VCU. For solution 3, both the vocal audio and square wave audio had distortions around harmonics of 27 Hz. These were also present in the spectrogram of solution 2 for the square wave. Determining the specific cause of these distortions would be tedious at best, but the literature on vocoders offers some clues. In the discussion of the literature, requirements around window overlap percentage as well as windowing techniques were discussed repeatedly. Recall that Laroche and Dolson found that a window-to-window overlap of at least 50% was necessary for integer pitch shifts, and at least 75% for fractional pitch shifts. Solution 2 has a clear problem: it uses a window-to-window overlap of 0%. While this is beneficial for avoiding the other issues with STFTs, it is likely inappropriate for musical audio. Moving the frequency of a signal from 203 Hz to 207.65 Hz is certainly a fractional pitch shift, so it is possible any distortions in solution 2 came from this. Solution 3 uses a 50% overlap between windows, which could cause the same problem. The high frequency distortions in the original VCU's audio output appear to be some artifact of the aliasing of a high frequency signal unintentionally injected into the audio signal. The most likely explanation is that the step taken just before the STFT, known as windowing, is being handled improperly. While the literature suggests that it is important to use a Blackman window for 75% overlap, it also indicates that a complementary "synthesis window" be used to window the inverse STFT output. Neither the original VCU nor any of the on-the-fly solutions used this. In fact, the on-the-fly

solutions skipped the use of a Blackman analysis window as well. Skipping that step on the original VCU caused even more extreme and unpredictable distortions which can be attributed to spectral leakage between adjacent frequency bins. It seems that leaving out the analysis window removed the high frequency, strong signal distortions. It is unclear why spectral leakage does not cause extreme distortions in the on-the-fly solutions.

Both the original VCU device as well as the redesigned device use extensive libraries written by STMicroelectronics for their respective microprocessors. These libraries, commonly known as the hardware abstraction layer (HAL), made working with the microprocessor kits relatively easy. However, as with any situation where a software developer uses another's software code, this presents some substantial drawbacks in each design. Since the libraries provided are extensive and complicated, there is not enough time in the context of a senior design project or even an honors thesis to analyze all of this code and ensure it is not interfering with the design task at all. Particularly on the original VCU's discovery kit, it is important yet time consuming to consider how the hardware present on the device might affect performance. Both of these considerations led to some oversights in both the design of the original VCU and the on-the-fly solutions. An example from the original VCU would be the wireless module present on the device. It was not noticed until shortly before the design project presentation for the original VCU that the wireless module was still receiving power and possible operating on the original VCU. While unlikely, it is entirely possible that radio frequency signals associated with this module interfered with the incoming or outgoing audio from the device. For both devices, the sample acquisition code relies heavily on low level code for setting the sampling rate, moving the samples into working buffers, and of course for

driving the sample acquisition hardware. Any issues or inefficiencies in this code would carry over to the VCU's operation as well.

CONCLUSION

The Vocal Conditioning Unit was a device intended to offer the experience of pitch adjustment to a vocal performer with a standalone gadget, without the need for expensive software such as Auto-Tune or the digital audio workstations it is often used with. Realistically, the VCU does not offer this due to the issues with audio quality previously discussed. Pitch adjustment is a complex, computationally expensive process. This thesis project has affirmed that it is possible to implement on-the-fly pitch adjustment with a low-end microprocessor, although many improvements could be made to the overall design of the device to improve the resulting audio quality and the experience of using the device. Auto-Tune can provide some helpful insights. Today, Auto-Tune is used almost exclusively on powerful multi-processing devices like the MacBook Pro. At the very least, it should be possible to take advantage of having multiple processing cores to avoid issues like sample overrun. As microprocessors become more and more powerful, cheap multicore microprocessors become more common. The RP2040 by the Raspberry Pi Foundation is a great example of such a multicore microprocessor.

In the original design of the VCU, it could be argued that too much focus was placed on using familiar tools, hardware, and techniques. For example, the choice to use STMicroelectronics processors on discovery kits and Nucleos was entirely due to the author's previous use of this hardware and its associated software in microprocessing courses. The processor on the discovery kit used for the original VCU has two processing cores, but only one was used as the author was not familiar with multi-processing techniques, especially not in the context of a microprocessor. The biggest design concern

for the original VCU, which was discussed briefly in this paper, was the need for an external memory module. The author sought out a discovery kit which solved this design problem by having an external memory module and existing software for interfacing with it. Not only was this discovery kit difficult to work with due to the complexity of the driver software STM provides with it, but it also offered little flexibility for inputs and outputs. Instead, the original VCU team could have designed and laid out a printed circuit board with, for example, an RP2040 interfacing with some sort of external memory module. The hardware design process for the original VCU could have been much more thorough to allow for on-the-fly processing by including more powerful hardware as well as much more precise and accurate pitch adjustment by using an appropriate sampling rate for musical audio as well as larger FFTs. While using an RP2040 would not remove the issue of using other people's code which cannot be efficiently understood or analyzed, it would reduce the amount of external code relied upon.

Overall, the VCU project presented many challenges and did not offer many satisfying results. The fact of the matter is, while the pitch detection algorithms are complicated and require a thorough understanding of digital signal processing techniques to implement in depth on a microprocessor, a higher quality result can be achieved easily on a powerful desktop computer. Using existing libraries in Python, it is possible to implement on-the-fly pitch correction in under an hour. This statement is not meant to trivialize the task of implementing such a project on a microprocessor, but to put into perspective the usefulness of such a project. Many aspects of the project would benefit from a renewed approach or could be interesting and difficult design projects in and of themselves. Previously discussed was the possibility of using a multicore system to fix

some issues in the existing project. Another, lower-level project could involve avoiding the use of libraries such as STMicroelectronic's hardware abstraction layer to make a simple digital effects pedal for effects such as reverb, level equalization, and others. Finally, experimenting with vocoders and implementing some of the "exotic effects" discussed by Laroche and Dolson in [5] could be an interesting project for developing a deep understanding of vocoders.

REFERENCES

[1] J. L. Flanagan and R. M. Golden, "Phase vocoder," The Bell System Technical Journal, vol. 45, no. 9, pp. 1493-1509, 1966.

[2] J. Allen, "Short term spectral analysis, synthesis, and modification by discrete Fourier transform," IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 25, no. 3, pp. 235-238, 1977.

[3] E. Jacobson and R. Lyons, "The sliding DFT," IEEE Signal Processing Magazine, vol. 20, no. 2, pp. 74-80, 2003.

[4] J. Laroche and M. Dolson, "Improved phase vocoder time-scale modification of Audio," IEEE Transactions on Speech and Audio Processing, vol. 7, no. 3, pp. 323-332, 1999.

[5] J. Laroche and M. Dolson, "New phase-vocoder techniques for pitch-shifting, harmonizing and other exotic effects," Proceedings of the 1999 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics. WASPAA'99 (Cat. No.99TH8452), 1999.

[6] L. Rabiner, M. Cheng, A. Rosenberg and C. McGonegal, "A comparative performance study of several pitch detection algorithms," IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 24, no. 5, pp. 399-418, 1976.

[7] N. Sripriya and T. Nagarajan, "Pitch estimation using harmonic producted spectrum derived from DCT," 2013 IEEE International Conference of IEEE Region 10 (TENCON 2013), 2013.

[8] P. de la Cuadra, A. Master and C. Sapp, "Efficient Pitch Detection Techniques for Interactive Music," in International Computer Music Conference, Havana, 2001.

[9] D. Hughes, "Technological pitch correction: controversy, contexts, and considerations," Journal of Singing, vol. 71, no. 5, 2015.

[10] T. Lester and G. White, "Vocal Conditioning Unit Engineering Design Report," 2023.

APPENDIX: ADDITIONAL RESULTS

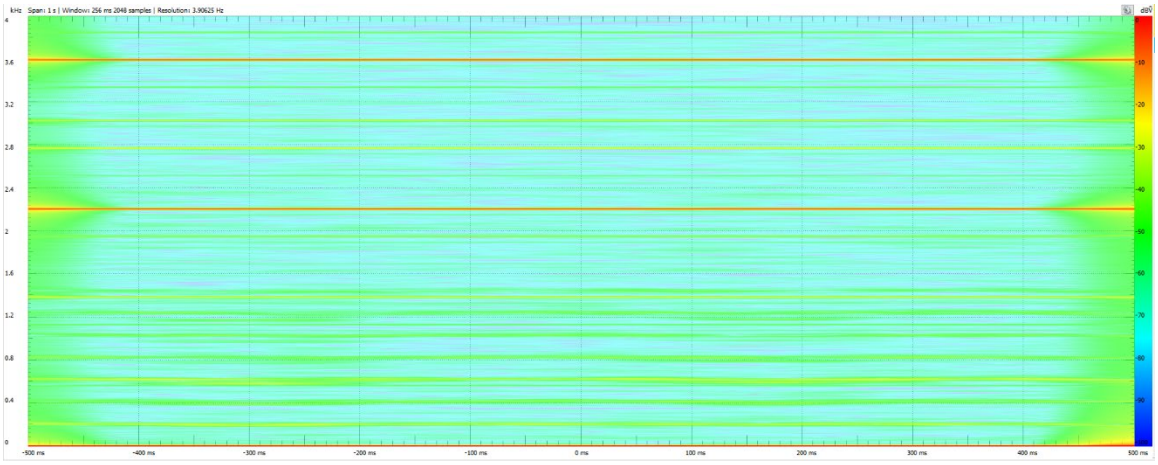

Figure 8. Spectrogram of original VCU output, 203 Hz sung tone, up to 4 kHz.

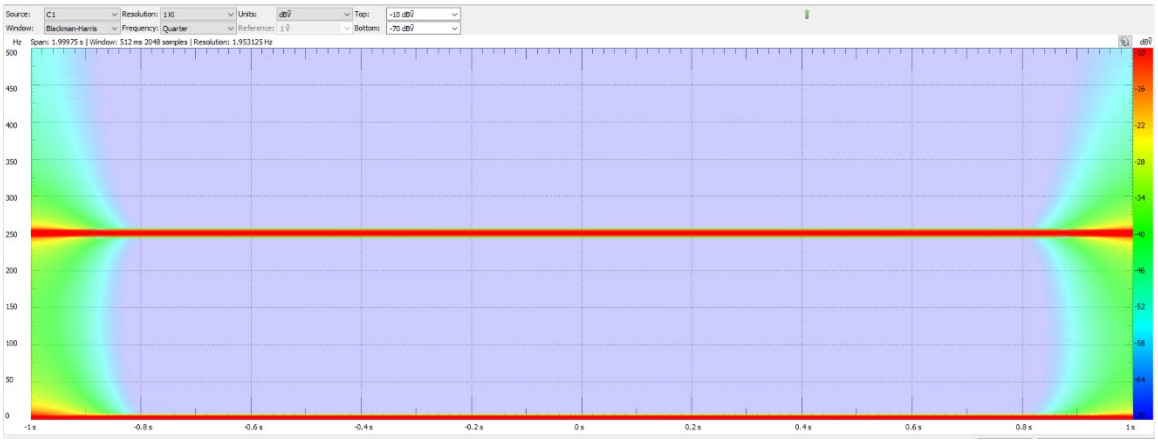

Figure 9. Spectrogram of an unmodified square wave at 250 Hz.

Figure 10. Spectrogram of solution 1 output, 250 Hz square wave.

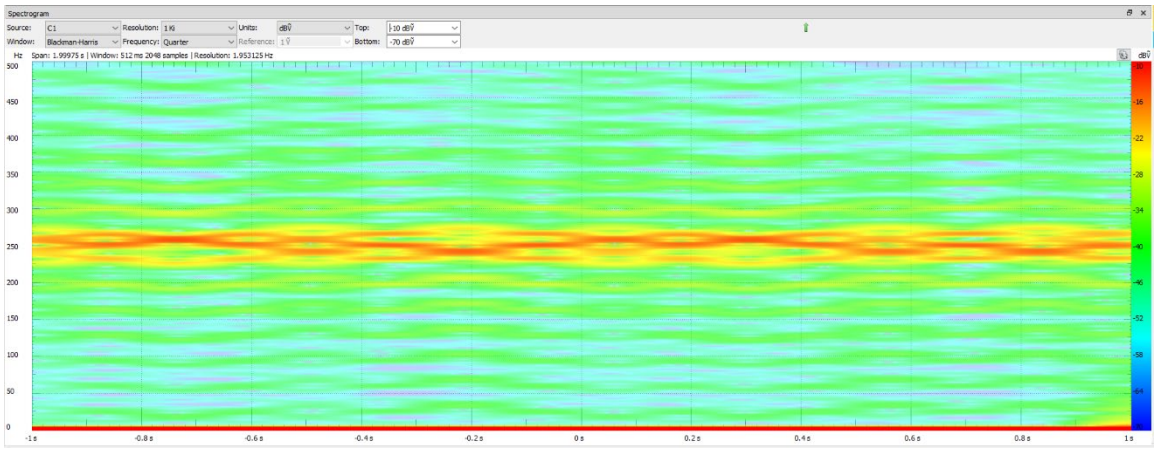

Figure 11. Spectrogram of solution 2 output, 250 Hz square wave.

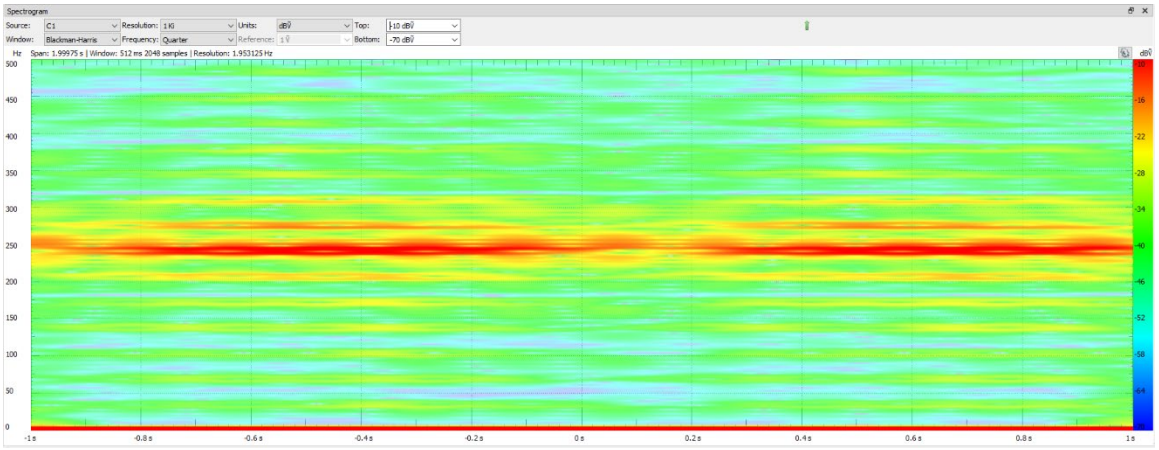

Figure 12. Spectrogram of solution 3 output, 250 Hz square wave.

AUTHOR BIOGRAPHY

Tim F. Lester was born in Rock Island, Illinois on October 30, 2000. They spent most of their childhood in Cumberland, Maine and graduated from Greely High School in 2019. They have majored in computer engineering and minored in mathematics. Tim is a student member of the Institute for Electrical and Electronics Engineers, IEEE, and a member of the IEEE Eta Kappa Nu honors society.

After graduating, Tim plans to start their career in computer engineering, likely in the field of embedded software development or firmware engineering.