Guitar Effects Processor Using DSP

Alexander J. Czubak
Gorav Raheja

Advisor: Dr. Thomas L. Stewart

Acknowledgements

We would like to thank the Bradley University Electrical and Computer Engineering Department for allowing us to do this project, Mr. Christopher Mattus for providing the software to interface with the DSP board and allowing us to use one of the lab computers for the Student Expo, our fellow students for having to listen to us in lab with real-time testing, our friends and families for dealing with our time management to get the project done and for supporting us in everything we do, Dr. In Soo Ahn for providing the DSP board used for the project, and especially Dr. Thomas L. Stewart for advising us and giving us guidance when we hit difficult hurdles to overcome.

Abstract

Digital signal processing has been used in the audio field for some time. One of its uses is to modify specific audio signals to generate effects not inherent in the signal. The digital domain provides quicker and easier methods of audio processing, especially with increased speeds and decreased costs of microprocessors and digital signal processors. The processing of guitar effects is just one of many applications. This project focuses on the design of a guitar multi-effects processor using a digital signal processor, development board. An electric guitar sends an analog signal to the board, and the board converts it to a digital signal and makes the necessary modifications and adjustments to it to provide the desired effects. The user controls the selection of effects and their intensities through a graphical user interface, or GUI. Once the signal is processed, it is converted back to an analog signal and sent to a guitar amplifier for the user and others to hear.

Table of Contents

I. Project Overview

A. Introduction

This project involves creating a system for real-time audio effects applications. An electric guitar sends a signal to a digital signal processing development kit. The primary components of the board for this project are the audio input port, analog-to-digital converter, digital signal processor, digital-to-analog converter, and audio output port. The converters run at a sampling rate of 48000 samples per second. The processor houses eight digital filters designed in Simulink. Each filter generates a specific effect onto the signal. The eight designed effects are Distortion, Volume Envelope, Octaver, Flanger, Phase Shifter, Chorus, Delay/Echo, and Reverberation. A graphical user interface, or GUI, allows the user to select which effects are active and which are not. The GUI also allows the user to determine the intensity of each effect.

B. Why It Was Chosen

Audio effects are commonplace in the music industry and with music hobbyists. Software exists that can create effects similar to their analog counterpart. Recording programs allow the user to choose what effects he or she wants to add to the audio. Effects pedals have become predominantly digital over time because of ease of production versus the analog parallel. However, many of these hobbyists know do not know how these effects are created outside the click of a button and the effort it takes to develop the effects for practical use.

What this project allows is an in-depth study of how audio effects, specifically effects for electric guitars, are created in the digital domain. There are a few reasons for pursuance of this project. First, there is the already-mentioned notion of understanding how these effects are created. This understanding leads to the next reason: a practical application of classroom study. Many theoretical concepts have been discussed in electrical engineering classes. However, some of these concepts, especially in digital signal processing, were not implemented through hands-on activities and exercises. This project not only provides practical experience but also provides development for assignments and projects involving audio and/or digital signal processing.

The third reason for pursuance of this project is the marriage of hobbies and interests with the study of electrical engineering. A number of students have interests in music and/or audio. Although the effects are designed with a guitar in mind, other instruments could be implemented with this effects processor, essentially allowing hobbyist musicians ways to alter their sound without changing the instruments. This link of class study with common interests allows this project to be enjoyable overall.

Because effects are so common, it is difficult to see how this project provides something outside of the engineering field. The GUI allows the project to have practicality outside of the field. In live performances effects are chosen and levels are set before the actual performance. In recording situations effects are either added after the parts are recorded or preset before recording on the pedals or mixing board. Being able to change the effects on the fly using a GUI allows easier presetting and real-time modifications for both live and recording situations. The GUI also allows portability for live shows if placed on a laptop. The approach is similar to keyboard and synthesizer players such as Tony Banks of Genesis and Nick Rhodes of Duran Duran having laptops with their instruments for automated preset selection.

II. System Description

A. Overall System

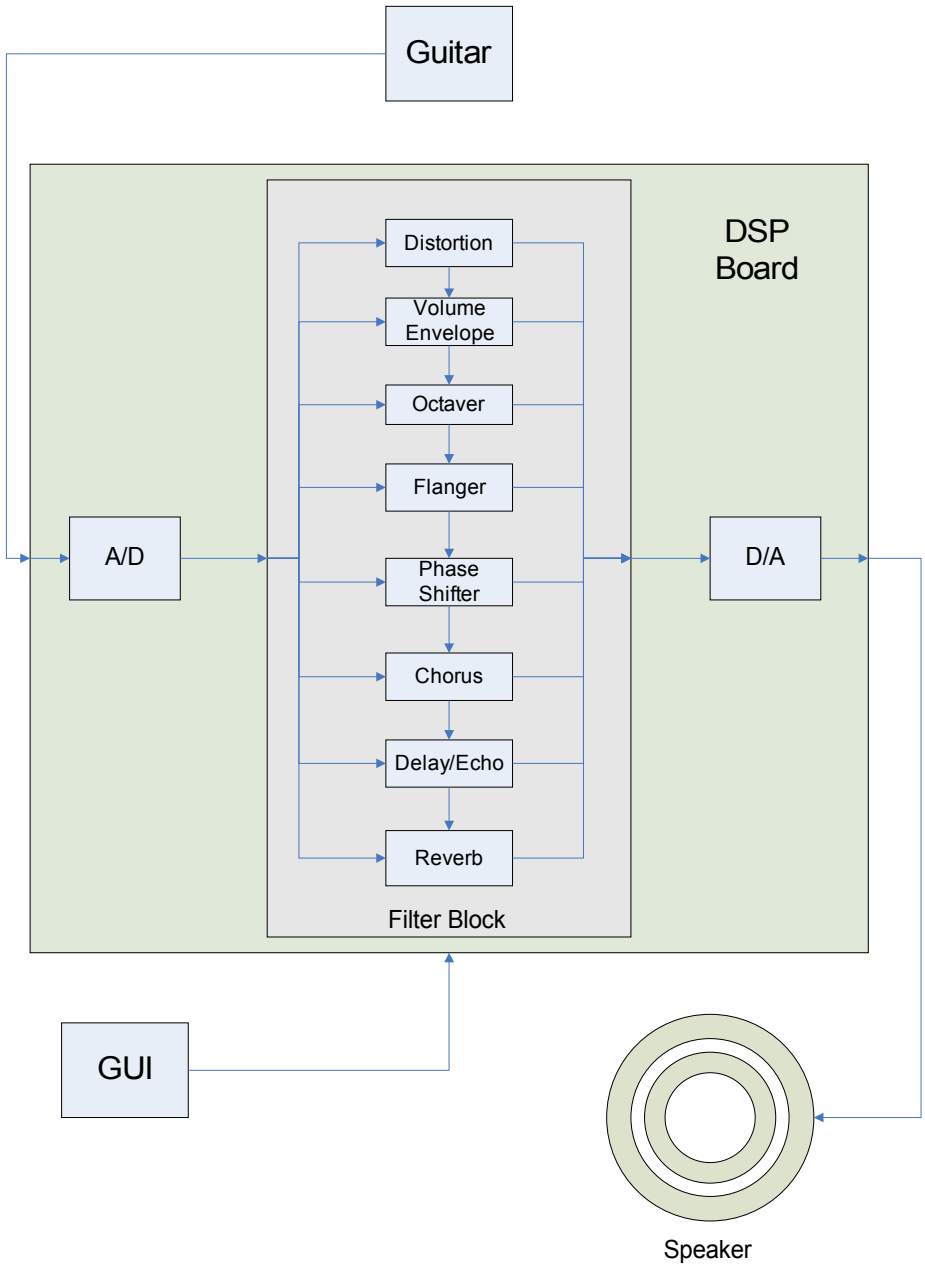

Figure 1: Overall Block Diagram of Project

Figure 1 shows how the project is laid out. For this project, two guitars were used – a Squier Stratocaster and a Squier Telecaster Custom. The DSP board utilized was a Spectrum Digital TMS320C6713 development starter kit. The speaker used was a Fender Frontman 15R guitar amplifier. Both the filters and GUI were designed in MATLAB and Simulink. The filter block is shown to be in series and in parallel to illustrate that the user can decide on which effects are on or off without worrying about the signal being stopped.

B. Guitar Description

:: Close Window

Model Name	Strat® (Rosewood)
Model Number	031-0600-(Color#)
Series	Affinity Series
Colors	(506) Black, (525) Metallic Red, (595) Metallic Blue, (Polyurethane Finish)
Body	Alder
Neck	Maple
Fingerboard	Rosewood, 9.5” Radius (241 mm)
No. of Frets	21 Medium Jumbo
Pickups	3 Single-Coil Pickups
Controls	Master Volume, Tone 1. (Neck Pickup), Tone 2. (Middle Pickup)
Pickup Switching	5-Position Blade: Position 1. Bridge Pickup Position 2. Bridge and Middle Pickup Position 3. Middle Pickup Position 4. Middle and Neck Pickup Position 5. Neck Pickup
Bridge	Synchronous Tremolo
Machine Heads	Standard Die-Cast Tuners
Hardware	Chrome
Pickguard	1-Ply White
Scale Length	25.5” (648 mm)
Width at Nut	1.61” (41 mm)
Unique Features	Large Headstock ‘60s Style Headstock, White Plastic Parts, Black Silkscreen Logo, Dot Position Inlays
Strings	Fender Super 250L, (.009 to .042) Nickel Plated Steel p/n 073-0250-003
Accessories	None
Introduced	1/2001
Notice	Product Prices, Features, Specifications and Availability Are Subject To Change Without Notice

:: Close Window

Figure 2: Squier Stratocaster Specifications

:: Close Window

Model Name	Vintage Modified Telecaster® Custom
Model Number	032-7502-(Color#)
Series	Vintage Modified Series
Colors	(506) Black (Polyester Finish)
Body	Agathis
Neck	Maple, C-Shape, (Gloss Polyurethane Finish)
Fingerboard	Maple, 7.25” Radius (184 mm)
No. of Frets	22 Medium Jumbo Frets
Pickups	2 Chrome Covered Humbucking Pickups (Neck/Bridge)
Controls	Volume 1. (Neck Pickup), Tone 1. (Neck Pickup), Volume 2. (Bridge Pickup), Tone 2. (Bridge Pickup)
Pickup Switching	3-Position Toggle: Position 1. Bridge Pickup Position 2. Bridge and Neck Pickups Position 3. Neck Pickup
Bridge	6-Saddle Strings-Thru-Body Tele Bridge
Machine Heads	Standard Die-Cast Tuners

Hardware	Chrome
Pickguard	3-Ply Black
Scale Length	25.5” (648 mm)
Width at Nut	1.650” (42 mm)
Unique Features	Dot Inlays
Strings	Fender Super 250L, Nickel Plated Steel, (.009 to .042), p/n 073-0250-003
Accessories	None
Introduced	7/2003
Notice	Product Prices, Features, Specifications and Availability Are Subject To Change Without Notice

:: Close Window

Figure 3: Squier Telecaster Custom Specifications

Figures 2 and 3 describe the guitars used for the project. The importance of each guitar lies in their pickups. The Squier Stratocaster contains single-coil pickups; this pickup is prone to what is known as “60 Cycle Hum.” This hum is generated by the magnetic coils picking up the 60-Hz frequency inherent in the national power system. When objects such as lights and televisions are close to the pickups, the 60-Hz signal in those devices interferes with the magnetic coils, adding a hum to the signal whether or not a note is played. The Squier Telecaster Custom contains what is known as “humbucker” pickups. These pickups are essentially two single-coil pickups wound in opposing directions to cancel out the hum. The sound from these pickups is generally louder and more expansive because of each pickup have two coils instead of one.

C. DSP Board Description

The board in question is a Spectrum Digital TMS320C6713 development starter kit. The digital signal processor used is a Texas Instruments C6713 DSP that runs at 225 megahertz. The board houses 16 megabytes of Synchronous Dynamic Random Access Memory as well as 512 kilobytes of flash memory. There are two inputs, Line In and Microphone, as well as two outputs, Line Out and Headphone, for audio. This board also contains an A/D converter and a D/A converter. These converters only handle amplitudes of two volts before sustaining damage.

Another key aspect of the board is that it contains the AIC23 Stereo Codec allowing it to have sampling rates from eight kilohertz to 96 kilohertz. Typical audio cards and systems have sampling rates of 44.1 kilohertz or 48 kilohertz for monaural audio, and 48 kilohertz is an easier number to use in calculations, so that was the chosen sampling rate for the system.

D. Speaker Description

:: Close Window

Model Name	Frontman™ 15R
Model Number	023-1501-000
Series	Frontman Series
Type	Solid State
Output	15 watts into 8 ohms
Ohms	8 ohms
Speakers	1-8" Fender® Special Design Speaker, p/n 0025421000
Channels	Dual Selectable Channels (Normal and Drive)
Features	Reverb, 3-Band EQ, Headphone Jack, Auxilliary Input for CD, Tape or Drum Machine, Closed Back, Blackface Styling with Silver Grille Cloth
Controls	Volume Normal Channel, Gain, Drive Select Switch, Volume Drive Channel, Treble,

	Mid, Bass, Reverb
Covering	Black Textured Vinyl with Silver Grille Cloth
Weight	15 lbs. (6.80 kg)
Dimensions	Height: 12.5" (31.8 cm), Width: 13.25" (33.65 cm), Depth: 7.25" (18.41 cm)
Power Handling	N/A
Tube Complement	N/A
Cover	None
Accessories	None
Footswitch	None
Introduced	1/2004
Notice	Product Prices, Features, Specifications and Availability Are Subject To Change Without Notice

Figure 4: Fender Frontman 15R Amplifier Specifications

Figure 4 presents the unit used for audio output. The Fender Frontman 15R amplifier contains two effects that had to be bypassed for this project – distortion and reverberation. The distortion comes from the overdrive/distortion channel of the amplifier. To bypass it, the clean channel was selected. The reverberation effect on the amplifier is controlled by a potentiometer, so setting it to the lowest value allows that effect to be bypassed. The other potentiometers control the volume of each channel as well as the frequency response of the signal.

III. Filter Design

A. Summary of Design Approach

A Creative SoundBlaster X-Fi Platinum sound card recorded signals for testing of the filters. The filters themselves were designed in MATLAB and Simulink. MATLAB uses a coding scheme called "M-code" that allows for easier function creation. This was the first part of the filter design. Simulink provides faster calculations and visual representation of the filter design. Components of the filters are represented by blocks that contain the necessary calculations. There is also the capability to use M-code in Simulink. This was the second part of the filter design. The final part of the design process was testing the design in real-time. The DSP Board came with software called Code Composer 3.1; this software links up with Simulink to automatically generate the proper C code for board programming, allowing for real-time testing and demonstration.

B. Distortion Design

Figure 5: Simulink Distortion Model Used for Project

Distortion works similar to an operational amplifier when the output voltage is at saturation. The signal is boosted by a specific gain value, and then it is clipped by a saturation block, making the waveform contain more harmonic components depending on how similar the wave is to a square wave at the same frequency. Setting the gain value at one leaves the signal unchanged, while setting it at 50, this filter's highest gain, boosts the signal so much that the waveform is almost identical to a square wave. Different audio clips were used to hear the effect in Simulink, and the effect worked each time.

This effect worked in real-time as well, making this the first fully functional effect of the project

C. Volume Envelope Design

The concept of this effect comes from two sources – songs featuring backwards guitar solos such as The Beatles' "I'm Only Sleeping" and songs featuring guitars with volume pedal such as Genesis' "Ripples." This effect begins the signal at a gain of zero, taking away the attack or pluck of the guitar at the very beginning. Then, the gain slowly increases to one and the signal approaches its original amplitude. The design is still in the first part. It only works at the beginning of each audio file; it does not do this for each plucked note. An attempt to make a function in Simulink for this effect did not work; the function kept resetting the gain value to zero because the function commands repeated every sample. A way to define the changing gain outside of the function block was not found because of time constraints. A method that could be implemented in the first part that resets the gain to zero when a certain value is reached was not tested in time for the demonstration, but it could be used for further study and testing if the effect is needed in a system.

D. Octaver Design

A typical 6-string or 12-string guitar has its lowest frequency at approximately 80 hertz. Bass guitars, usually a 4-string or 5-string design, reach frequencies lower than that. If a band or group has no bass, then a guitar with an octave-down, or frequency halving, effect can fill in the void. The original approach of the octaver was to have an octave-up effect, but research concluded that an octave-down effect would be more practical for musicians. Most octave-down effects add a waveform, usually a sine or sawtooth wave, that is an octave lower than the note played to the signal. The effect in this project attempts to modify the actual guitar signal instead of just adding a signal along with it.

Two approaches were attempted to modify the signal – doubling the samples at each value and upsampling the signal by two but still outputting it at a 48-kilohertz sampling rate. Both approaches worked in M-code and Simulink with one caveat – the signal was twice as long as the original. This is caused by a fundamental property of waves – to maintain velocity, decreasing the frequency by half means the wavelength must be doubled. Because of the doubling length, notes were not played at the same time as they were plucked outside of the first note. An approach was discussed that was similar to the volume envelope method in which the signal resets what sample to start the upsampling or sample doubling when it hits a certain value. The approach needs more research and study before implementation, and time constraints prevented this and further development from happening.

E. Flanger Design

Analog flanging occurs during the recording process. One tape records accurately, while another tape has its motor changing speed from faster to slower by small amounts to distort what is recorded. When the shifting tape and original recording is mastered together, the resulting sound contains a "whooshing" effect behind the recorded sound. This in-and-out sound occurs because of construction and destruction of the combined wave. Digital flanging takes this approach and allows real-time capability.

Digital flanging design contains a signal line that is split. The split is passed through a variable-time delay, simulating the changing motor speed of a recording reel. The modified signal is added back into the original signal, creating the flanging effect. Variable-time delay in the digital domain poses a problem in that accurate variable-time delay calls for delays to occur at fractions of a sample, something that is impossible to do. A method to overcome this is interpolation, a way to average the value of the sample in-between two samples. Linear interpolation is the easiest scheme as it is an

averaging calculation.

$$e = a \cdot c + b \cdot d \quad (1)$$

where *c* and *d* represent values of two corresponding samples, *a* representing the percentage the fractional sample is toward the first sample, and *b* representing the denominator of that fraction.

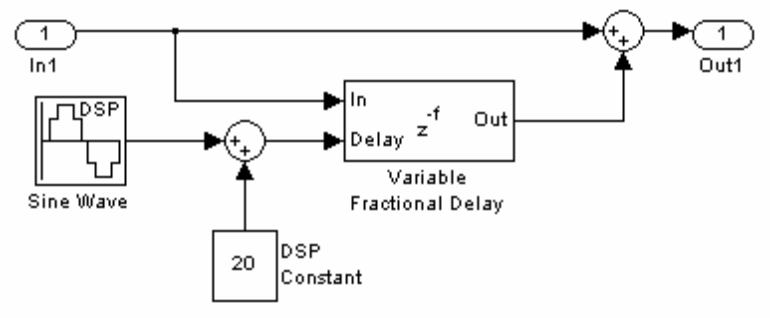

Figure 6: Simulink Flanger Design Used for Project

A sine wave of a frequency less than or equal to one hertz is used to control the variation value. The maximum value of delay is ten milliseconds. The amplitude of the sine wave is fifteen units, and the constant value is added to the wave in order to make the delay vary more intensely. Without the constant, there is little or no effect present. The effect worked in Simulink, but it was not tested in real-time as its cousin effect, chorus, presented a problem with the variable fractional delay in its real-time test. This is discussed later in the chorus effect design section.

F. Phase Shifter Design

Phase shifter works essentially like the flanger, but it has one key difference. Instead of a straight variable-time delay, it is a series of all-pass filters that each have variable-time delay.

$$H(z) = \frac{a + z^{-n}}{az^{-n} + 1} \quad (2)$$

Two all-pass filters create one notch in the overall magnitude frequency response. Selecting the location of the poles and zeros allows nonlinear spacing of the notches in the magnitude frequency response when the altered signal is added back into the original. These notches have to be within the audible range of frequencies – between 20 hertz and 20 kilohertz – but far enough away from the actual notes of the guitar to keep it from affecting the notes. The sweeping nonlinearly spaced notches generates an effect similar to flanger but with unique audible characteristics that make it sound more otherworldly.

The design included eight filters in series to create four notches. However, the filter did not function properly in the second part of design. The Simulink model created for the effect did not generate any audible differences to the original signal. The development of the filter occurred later in the project cycle, so fine-tuning and debugging could not commence on the filter design. The first test that could be done is to set all of the filters at the same poles and zeroes to see what notches, if any, are created. After the effect is tested and functioning, creating a signal other than a sine wave to drive all of the variable-time delays would add to the otherworldly presence of the signal.

G. Chorus Design

Some guitarists want to have a guitar with a fuller sound, a sound that makes it sound like a group of guitars played at the same time. This is where the chorus effect comes into play. Similar to the flanger, a signal is split and the split signal is then passed through a variable-time delay. In the case of the chorus effect, the signal is split at least once and theoretically up to an infinite number of times; each split has a variable time delay with different maximums and rates of change. The maximum delay is 50 ms.

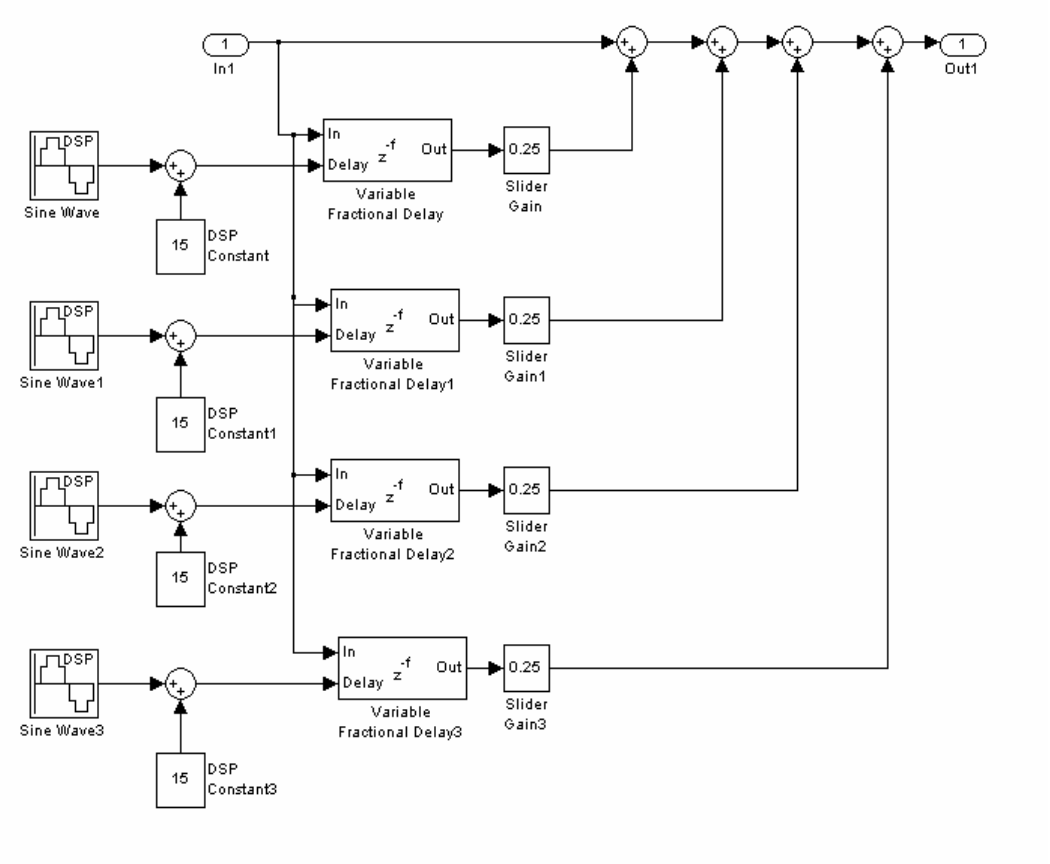

Figure 7: Simulink Chorus Design Used for Project

This project called for a chorus design that had four splits. Each has a variable-time delay that is driven by independent sine waves. Also, each split has attenuation so that the result of the filter is a signal closer to the original values. This design made it to real-time testing, but these tests presented a common flaw with variable fractional delay – popping sounds whenever the delay changed. This popping occurs because of discontinuities brought on by the delay. As mentioned in the flanger design, fractional delay is difficult to manage in the digital domain because there is no fraction of a sample. Interpolation is used to alleviate the issue, but what worked in Simulink did not in real-time. It is possible that the interpolation scheme did not translate over to the C code for the DSP, but time constraints hindered a deep analysis of the code. Also, the hardware was not set up to do real-time analysis in Simulink, so checking the Fast Fourier Transform, or FFT, of the signal was not an option. If the FFT is analyzed, there may be a way to filter out the popping, but this is an assumption and not a guaranteed solution.

H. Delay/Echo Design

Echoing a signal makes it sound like it is bouncing off of the walls, creating an illusion of more space in varying sizes of rooms. Delay is a direct relative in that it repeats the signal but it only does it once or twice and at typically around the same volume level as the original. Delay adds a repeating effect if it is at a small time delay like in the guitar solo of "Time" by Pink Floyd or allows a player to play against himself/herself if it is at a large delay, four to 60 seconds, like the Frippertronics and Soundscapes of Robert Fripp. These effects make the solos sound more complicated than what is actually played.

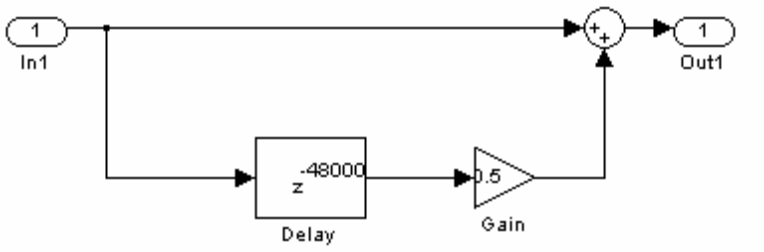

Figure 8: Simulink Delay/Echo Design Used for Project

The design is a creation of a fixed delay line. The delayed signal is passed through an attenuator to make the second appearance of the signal less prominent. The delayed signal is added back into the original signal for the effect to appear. For a more robust echo design, different delay lines at factors of one of the delays with varying values of attenuation would be the best approach. To hear a repeat of the signal, a delay of at least 100 milliseconds would suffice. This design worked in all three parts of the design, and it was suitable for real-time application. This was the second fully functional real-time effect.

I. Reverberation Design

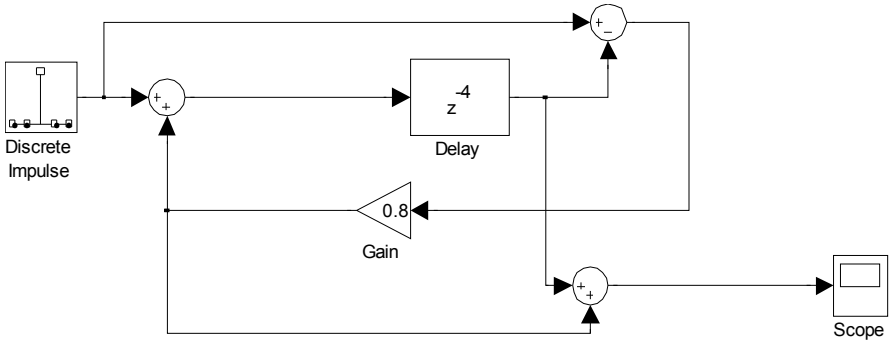

Figure 9: Simulink Reverberation Design Used for Project

Audio signals very rarely die out instantaneously. Acoustics of a room determines how fast audio dies out. Digital reverberation simulates this by a delayed attenuated signal after the initial signal reaches zero. In order for this effect to work, the magnitude response must be one for all frequencies, so the design is an all-pass filter. Figure 9 is the design used for the reverb filter. The feedback loop has attenuation so that the delayed samples slowly approach zero when the signal input is cut.

This filter was one studied in a digital signal processing class, so it was not created in MATLAB initially. The Simulink model was first tested with an impulse signal to see if the signal slowly dies down. Recorded signals that were tested showed the filter working with practical signals. In real-time tests the filter generated the necessary reverberation. In fact, the real-time effect worked better than the Simulink test. This was the third and final fully functional real-time effect.

IV. GUI Design

A. Summary of Design Approach

Since the filters were being designed in MATLAB and Simulink, it was decided to program the GUI in MATLAB as well. Having the two portions of the project designed in the same program would make interfacing easier than if they were developed in independent programs and languages. The programming environment in MATLAB called GUI Design Environment, or GUIDE, was not familiar territory, so time and practice was needed to learn the environment and any features that would aid in the design process. The next step is designing an interface that is easy for all users to understand and connecting it to the filter designs. The final step is making the GUI work with real-time effects.

B. Learning the Programming Environment

A GUI in MATLAB allows for basic graphics, animation, and handling graphic objects. Basic graphics such as 2-D plotting, figure windowing, data statistics, subplots, specializing plotting routines, 3-D plotting, and images can be used. When a GUI is created in GUIDE, it is stored in two files which are generated the first time the GUI is saved or ran. One of the files contains a complete description of the GUI figure layout and the components of the GUI. Changes to this file are made in the layout editor of the GUI. An additional file contains the M code that controls the GUI. A user can program the callbacks in this file using an editor in MATLAB.

GUIDE contains choices of effects like push buttons, sliders, radio buttons, check boxes, text

boxes, pop-up menu's, list boxes, toggle buttons, axis, etc. All of the graphics can be manipulated in size and color. GUIDE also allows a designed GUI to be used independently from MATLAB and on any computer by being compiled in the computer's low-level language.

The first test in understanding GUIDE was a basic calculator that displayed what number was pressed and performed addition. This test was a success in that it showed how text boxes work and how the push buttons functioned. The next test was to add more functions to the calculator such as subtraction, division, multiplication, and exponents. After the success of this test, further analysis of other GUI functions such as sliders and drop-down menus was done in preparation of the GUI design for the project.

C. GUI Design

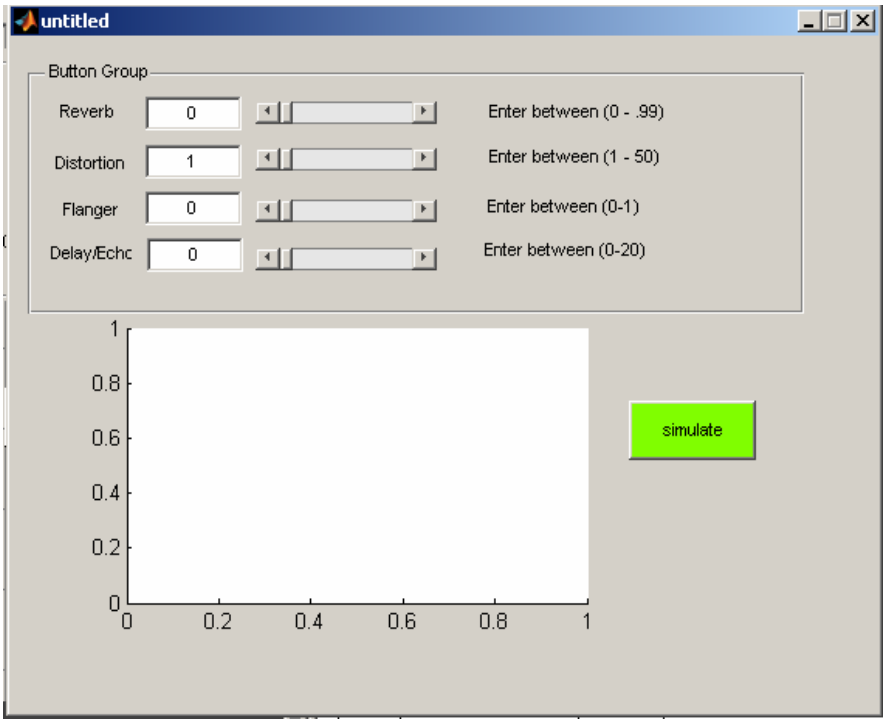

Figure 10: GUI Design for Project at initial values

The original design called for a drop-down menu to select an effect, a check box for each effect to activate it, and slider bars and text boxes to determine the intensity of each. Implementation of a drop-down menu could be done as easily as was necessary, so each effect and its settings appear simultaneously in the GUI as it does in Figure 10. A plot would show the waveform for the user to see how the signal looks after passing through the effects. Four of the eight effects were implemented in the GUI; three were incomplete and a format was not agreed upon for the chorus effect.

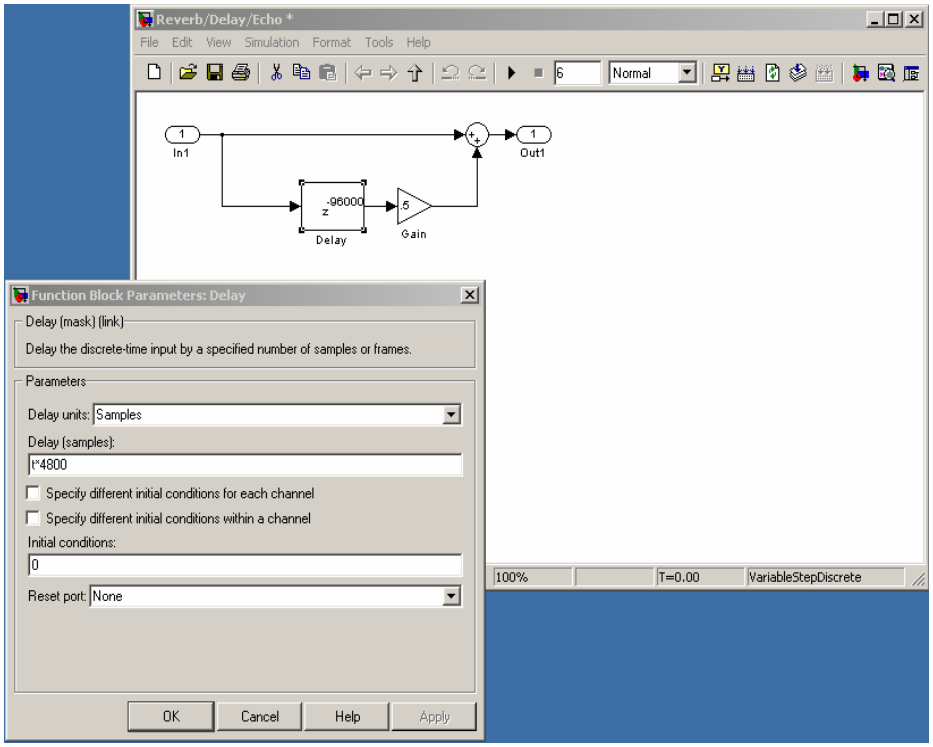

Figure 11: Declaring variables for Simulink Delay/Echo Filter to connect with the GUI

To link the GUI with Simulink, variables need to be declared in specific locations such as the delay or gain like in Figure 11. The GUI controls what values go into the variables, allowing each effect's intensity to be controlled. This method made for seamless integration of the GUI and Simulink models.

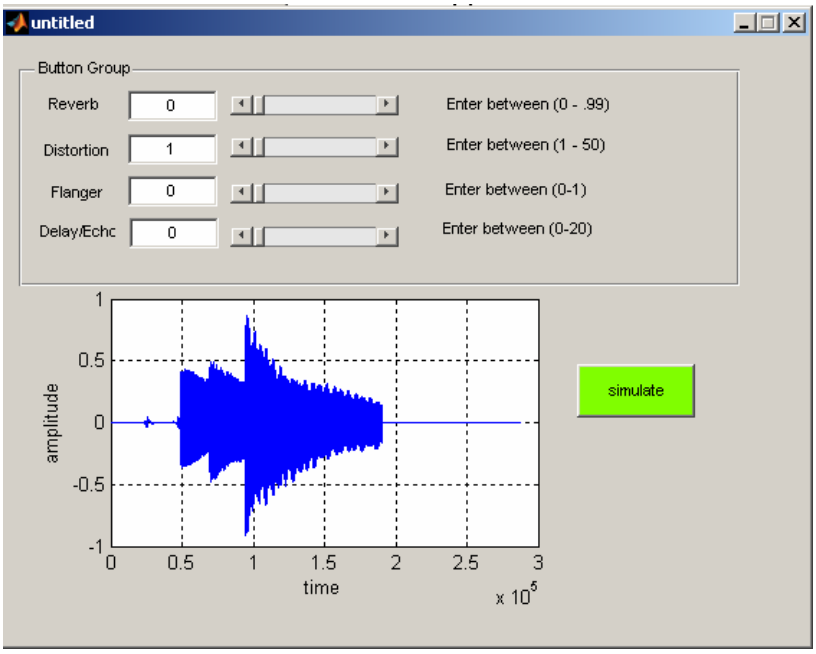

Figure 12: GUI with unprocessed recorded signal

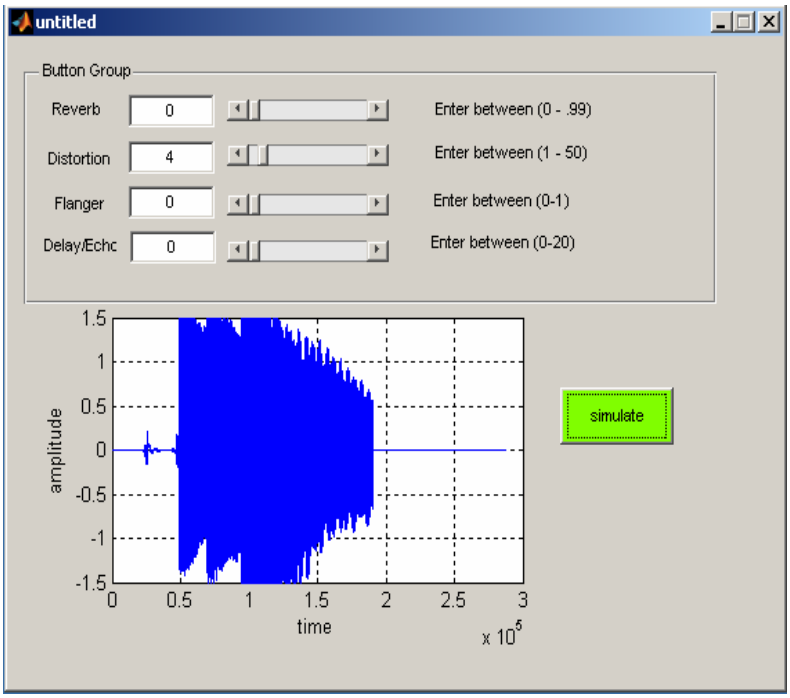

Figure 13: GUI testing recorded signal through the Distortion Filter

Figures 12 and 13 show how the GUI works with recorded signals. In this instance, the gain of the Distortion Filter is set to four, and the signal is clipped. The GUI works with recorded signals, but it could not be tested in real-time. In order for the GUI to function with the DSP board, a port called RTDX is required. This port allows real-time interaction with the DSP board on the computer. By the time the DSP board came into the project, there was no time to research and learn how to use the RTDX port in Simulink.

V. Results and Conclusion

A. Results

At the end of the project period, three effects were fully functional in real-time – distortion, delay/echo, and reverb – and two more effects were completed up to real-time implementation – chorus and flanger. The variable-time delay prevented chorus and flanger from proper real-time functionality. The last three effects – octaver, volume envelope, and phase shifter – need more tuning to get them working up to real-time. Octaver worked but slowed the signal, rendering it useless in practical real-time applications. Volume envelope only worked at the very beginning of the signal, and phase shifter did not produce any effect as of now. The three that worked in real-time were effective and satisfied the goal of getting effects working in real-time on the DSP board.

The GUI could not be implemented in real-time. However, it was able to interface with the Simulink models, controlling aspects of functioning effects and was a step before real-time integration. Slider bars as well as text boxes gave the user direct control over the effects. A plot in the GUI showed the output of the signal. That plot may have use beyond visually representing the signal for the user such as providing visual effects behind the performer in a live venue. The two halves of the project never merged through the DSP board, but they did function as they should, providing a foundation for connecting the two in real-time.

B. Conclusions

This project allows effective use of digital signal processing with audio. Full real-time implementation is a concept easy to understand but not as easy to design. The design approach, one side working on the filters and the other side working on the GUI, allows people of different expertises to work together. Design difficulties came about in the filter portion because of the variable-time delay. Difficulties with the GUI appeared because of developing using a proprietary language, M-code, instead of one in common use such as C or C++. Although the real-time aspect was not fully completed, the three effects working in real-time as well as the GUI working seamlessly with the Simulink models made this project modest success and one that should be used as a base for future projects or as a small part of a larger project.

C. Future Plans

Finalizing all of the effects is probably the first and most important thing that could be done in the future. The three variable-time delay effects could provide a separate project on developing interpolation schemes to make these effects work in real-time. The addition of more effects would add to the understanding of more complicated effects filters as well as making the system more robust. For example, adding a automatic “wah-wah” effect similar to the manual effect in the Issac Hayes song “Theme from 'Shaft'” or a “talk box” effect like the one used by Peter Frampton in many of his songs would be projects unto themselves. Another future addition would be adding pedal functionality to the board so that effects can be chosen on the fly while the GUI still allows the values for each filter to be changed. To add portability to the system, designing it as a USB drive or as a PC Card for a laptop, making the functionality similar to the keyboard players mentioned in the project overview. For the GUI portion of the project, getting it to function in real-time using the RTDX port would be a great starting point for future development. Adding more details in the GUI to control the effects may take away some of the simplicity of the interface, but it would allow the effects to be more dynamic.

References

"http://www.squierguitars.com - Strat (Rosewood)." Squier Guitars by Fender. 10 Dec. 2007 < http://www.squierguitars.com/products / view_specs.ph p? full_partno=0310600&name=Strat %26reg%3B+%28Rosewood%29>.

"http://www.squierguitars.com - Vintage Modified Telecaster Custom." Squier Guitars by Fender. 10 Dec. 2007 < http://www.squierguitars.com/products/ view_specs.php ? full_partno=0327502&name=Vintage+Modified+Telecaster%26reg%3B+Custom>.

"http://www.fender.com - Frontman 15R." Fender.com. 10 Dec. 2007 <http://www.fender.com/products//view_specs.php?full_partno=0231501000&name=Frontman %26trade%3B+15R>.

Oboril, David, Miroslav Balik, et al. Proceedings of the COST G-6 Conference on Digital Audio Effects. "Modelling Digital Musical Effects for Signal Processors, Based on Real Effect Manifestation Analysis." December 7-9, 2000: Verona, Italy.

Fernandez-Cid, Pablo and Javier Casajus-Quiros. IEEE. "Multiband Approach to Digital Audio FX." 2000: Madrid, Spain.

Verfaille, Vincent, Udo Zolzer and Daniel Arfib. IEEE Transactions on Audio, Speech, and Lauguage Processing. "Adaptive Digital Audio Effects (A-DAFx): A New Class of Sound Transformations." Volume 14, Number 5. September 2006.

Keen, R.G. Guitar Effects FAQ. May 20, 2000. <http://www.geofex.com/effxfaq/fxfaq.html>

Caputi, Mauro J. IEEE. "Developing Real-Time Digital Audio Effects for Electric Guitar in an Introductory Digital Signal Processing Class." 1998. <http://www.ewh.ieee.org/soc/es/Nov1998/01/BEGIN.HTM>

Stewart, Dr. Thomas L. Bradley University. Professor and Advisor. 18 Oct. 2007.

Lehman, Scott. "Phase Shifting (Phasing)". Harmony Central. 10 Dec. 2007. <http://www.harmony-central.com/Effects/Articles/Phase_Shifting/>

Lehman, Scott. "Flanging". Harmony Central. 10 Dec. 2007. <http://www.harmony-central.com/Effects/Articles/Flanging/>

Lehman, Scott. "Chorus". Harmony Central. 10 Dec. 2007. <http://www.harmony-central.com/Effects/Articles/Chorus/>