Implementation of digital audio effects for electric guitar on DSP platform

Miroslav Malko, Jelena Kovacevic, *Member, IEEE,* Robert Peckai-Kovac, Marko Gajic, Marija Jovanovic

***Abstract* — Starting from mathematical model, following electric guitar effects are implemented: delay, chorus and flanger. Implementation is done on a commercial fixed point single core DSP processor, which has total 24k words data memory, 2k program memory and processor speed of 150 MIPS.**

***Keywords* — chorus, delay, dsp, electric guitar effects, flanger, implementation**

I. Introduction

ELECTRIC GUITAR effects in general, like any other audio effects, change how electric guitar sounds like. They can be used at home, at live performances and professional recording studios. Guitar effects mostly appear in three forms: guitar processors, stompboxes and rack mounted effects. Also, there are some guitar amplifiers that have built in guitar effects (such as distortion and reverb which are most common built in effects).

Most common types of guitar effects can be divided into 7 categories:

1) Distortion
2) Dynamics
3) Filters
4) Modulation
5) Pitch/frequency
6) Time-based
7) Feedback/sustain

Although most of these effects are originally invented and implemented in analog domain, by fast advancing technology of DSP (Digital Signal Processing) chips it is now possible to implement electric guitar effects in digital domain at almost equal quality but much lower price. Also, implementation in digital domain is more reliable and robust than it is case with analog effects [1]. One more advantage of digital implementation over analog is that on one DSP chip many effects can be implemented and run in same time and also in many different combinations, which is not case with analog pedals. State-of-the-art multiple guitar effects processors today have hundreds of presets, almost all types of effects and are packed in one compact case, which is very easy to use and carry with. Rack mounted effects are also DSP based but are mainly used in professional studios, unlike guitar processors, which are mainly used at gigs.

Even with all those advantages of DSP based guitar effects, some guitarists still decide to use analog effects rather than DSP based multiple effects processors and rack mounted effects. Mostly, guitarists rather prefer tube amps than transistor based amps. Therefore, some guitar manufacturers, such as BOSS, Digitech, Vox, Line 6 and other offer hybrid solutions, and embed tubes in multi-effect electric guitar processors, in order to match sound of tube guitar amps. There is no doubt that as technology advances, tube technology in music will be abandoned and DSP technology will become dominant.

DSP implementation was chosen because of its many advantages, such as availability to be implemented in one chip, cheaply and reliably. Their possible application could be in multiple-effect guitar processors and rack mounted effect processors. The chip that was chosen is CS48560 audio DSP chip, made by Cirrus Logic. It is a 32-bit fixed point audio DSP processor, which has total of 24k words of data memory, 2k words of program memory and processor power of 150 MIPS (Million Instructions Per Second). This chip is successfully applied in many embedded devices, such as digital TV, car amplifiers, portable audio devices, digital speakers etc.

Effects implemented in this work are delay, chorus and flanger effects. Those are some of most popular electric guitar effects, used in almost every music genre.

Testing of these three effects has been done in two ways: listening tests and bit-exact tests, using set of 20 test vectors, which are clean (unprocessed) electric guitar streams.

II. Algorithms

A. Delay effect

Delay effect is time based electric guitar effect.

This work was partially supported by the Ministry of Education and Science of the Republic of Serbia under the project No. 32034, year 2011.

Miroslav Malko, Faculty of Technical Sciences, Trg Dositeja Obradovica 6, 21000 Novi Sad, Serbia (e-mail: miroslav.malko@rt-rk.com)

Jelena Kovacevic, Faculty of Technical Sciences, Trg Dositeja Obradovica 6, 21000 Novi Sad, Serbia (e-mail: jelena.kovacevic@rt-rk.com)

Robert Peckai-Kovac, RT-RK, Narodnog Fronta 23a, 21000 Novi Sad, Serbia (e-mail: robert.peckai-kovac@rt-rk.com)

Marko Gajic, RT-RK, Narodnog Fronta 23a, 21000 Novi Sad, Serbia; (e-mail: marko.gajic@rt-rk.com)

Marija Jovanovic, RT-RK, Narodnog Fronta 23a, 21000 Novi Sad, Serbia; (e-mail: marija.jovanovic@rt-rk.com)

978-1-4577-1500-6/11/$26.00 ©2011 IEEE

Delay/echo units produce an echo effect by adding a duplicate instrument-to-amplifier electrical signal to the original signal at a slight time-delay. Delay effect schematic is shown in Fig. 1.

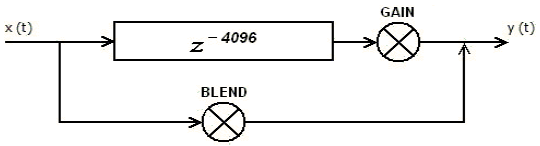

Fig. 1. Delay effect schematic

Output signal is formed as a combination of input signal and delayed input signal. The network that simulates a single delay is called the FIR comb filter. The input signal is delayed by a given time duration. The effect will be audible only when the processed signal is combined (added) to the input signal [2].

Original input signal is delayed by 4096 samples (at sample rate of 48kHz). The length of a delay-line is selected mostly based on listening tests (more in chapter testing). Also, in schematic there are two parameters, BLEND and GAIN. They adjust ratio of original input signal to delayed signal and are 0.6 and 0.4 respectively. However, they can be changed by taste. Only restriction is that their sum must be equal to one. This is necessary for two reasons: the first is, if their sum is greater than one, amplification of the signal and clipping is possible, which must be avoided. The second is - if their sum is less than one, signal on output can be weaker comparing to input and that shouldn't happen either. This is the reason that sum of these two parameters should be equal to one.

B. Chorus and Flanger effects

Chorus and flanger are very similar by their implementation. Chorus pedals mimic the effect choirs and string orchestras produce naturally by mixing sounds with slight differences in timbre and pitch. A flanger creates a "jet plane" or "spaceship" sound, simulating a studio effect produced by recording a track on two synchronized tapes and periodically slowing one tape by pressing the edge of its reel (the “flange”). To understand how these two effects work, it is useful to know how the heart of these two effects – the vibrato effect works. Vibrato effect schematic is shown in Fig. 2.

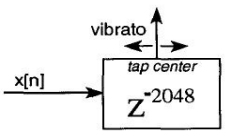

Fig. 2. Vibrato effect

Vibrato effect generates its output by dynamically delaying input signal in time. This dynamical delay is changing in time by sine function around tap center. Amplitude of this sine is defined with CHORUS WIDTH parameter, whose value is different for every effect (defined later in table 2.). However, vibrato output is not only time varying delayed input signal. Output from vibrato effect is generated by delay-line interpolation. The technique of delay-line interpolation is used when it is desired to delay a signal by some number of samples expressible as a whole plus some fractional part of a sample [3]. Delay-line interpolation is shown in Fig. 3.

In Fig. 3 v[n] is actually vibrato effect output. VoiceL[i] is input signal delayed by *i* samples (x[n-i]). Variable *frac* is calculated as:

$$frac \equiv i.frac - i$$

where *i.frac* is given with:

$$i.frac = NOMINALDELAY + CHORUSWIDTH \cdot y[n]$$

and y[n]:

$$y[n] = \sin(\Omega_\varepsilon nT),\ \Omega_\varepsilon = 0.15\text{Hz}$$

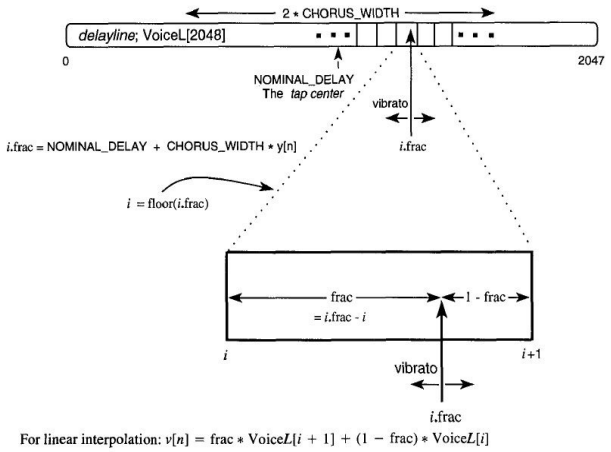

Fig. 3. Delay-line interpolation

Now, when vibrato effect is explained, way that chorus and flanger effects work can be explained as well. As they are very similar, their schematic is the same, only some parameters differ. Chorus/flanger schematic is shown in Fig. 4.

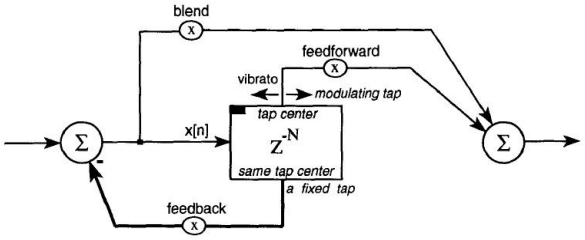

Fig. 4. Chorus/flanger effect schematic

As previously mentioned, core of both chorus and flanger effects is vibrato effect and this can be seen from Fig. 4. These two effects use fixed delay sample as well as varying delay sample for creating their output, which is difference to the vibrato effect. This fixed delay sample is forming negative feedback with input signal. Let’s say this is signal A[n]. This together is accepted into the left-hand side of the delay line. Varying delayed sample is then forming positive feedback with A[n] and that is the output of both chorus and flanger effects. Notice three parameters – blend, feedback and feedforward. Values for these three parameters for different guitar effects are given in table 1.

TABLE 1: PARAMETERS FOR CHORUS, FLANGER AND OTHER EFFECTS

Effect	Blend	Feedforward	Feedback
Vibrato	0.0	1.0	0.0
Flanger	0.7071	0.7071	−0 7071
Industry standard chorus	1.0	0.7071	0.0
White chorus	0.7071	1.0	0.7071
Doubling	0.7071	0.7071	0.0
Echo[82]	1.0	≤1.0	<1.0

This schematic is very versatile and not only chorus and flanger effects are implemented with this schematic – but also vibrato, doubling and echo effects. It is only necessary

to change parameters. This is shown in table 1., where different parameters are given for all above mentioned effects.

Only two things left to be defined are NOMINAL DELAY and CHORUS WIDTH parameters. They represent tap center of vibrato effect and amplitude of sine function varying delay. For chorus effect they are given as 400 and 350 samples respectively (for sample rate of 48kHz). For other effects they are given in table 2. and are given relatively, in milliseconds.

TABLE 2: PARAMETERS FOR TAP CENTER AND SINE AMPLITUDE FOR VARIOUS EFFECTS

Effect	Onset	Nominal	Range End
Vibrato	0	Minimal	5
Flange	0	1	10
Chorus	1	5	30
Doubling	10	20	100
Echo	50	80	∞

III. SYSTEM OUTLINE

In Fig. 5 is shown typical use scenario of guitar effects with guitar, effects and guitar amp.

Fig. 5. Typical use case scenario of guitar effects

As it can be seen from Fig. 5, guitar effects in most cases go between guitar and amplifier. This is not only the case for DSP–based effects – but for all effects, both analog and digital ones. In that chain there can be either one effect or many of them. It is guitarists choice based on type of music he plays, his musical taste etc.

IV. IMPLEMENTATION

By doing some research, analyzing both speed and efficiency of implementation, from the very beginning until final product, one specific methodology was chosen for this work. It proposes series of steps, through which work implementation is done. The methodology was successfully applied to several audio applications and their implementation to Cirrus Logic Coyote DSP family. Experience with those applications shows that this methodology greatly shortens time to market for DSP firmware product [4]. This step-based implementation methodology is shown in Fig. 6.

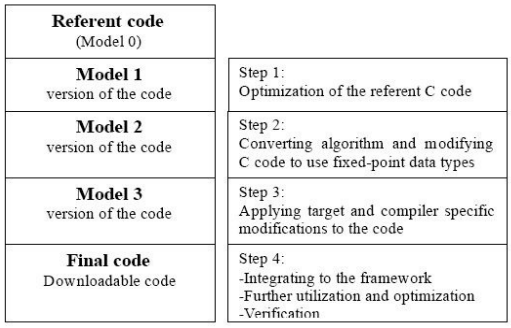

Fig. 6. Suggested implementation flow

First stage of implementation of these effects includes making a solid reference, which will be used for estimating the quality of final solution (Model 0). Therefore, algorithms described in previous chapter are implemented in C language, on PC architecture. Reference code is done in floating point, as that is standard for PC architectures. Then, when reference code is done, clean (unprocessed) electric guitar streams are tested with these effects and compared to commercial solutions, to make sure that desired effect is present. When satisfied with reference code, preparations for implementing on DSP are done. First, optimizations of reference code are made (Model 1). Later, instead floating point arithmetic, emulation libraries of fixed point arithmetic (as targeted commercial audio DSP processor uses this fixed point arithmetic) are included into C projects (Model 2). Important thing to mention is, when implementing algorithms on PC, practically there are not any limitations regarding MIPS and memory usage, which is not case when implementing on DSP. When finishing this stage of implementation (crossing from floating point to fixed point arithmetic), it is necessary to compare results with reference code. Afterward, code that is changed from floating point to fixed point arithmetic, written originally in C language is compiled into assembly language using CCC (Cirrus Logic C Compiler). This is Model 3. Finally, when this phase of implementation is finished, Model 3 code is adapted to be run on DSP processor (Final code). As mentioned, targeted DSP processor is a commercial audio DSP processor which has one core and processing power of 150 MIPS. It has 24k words of data memory and 2k program memory, reserved for code. In order to process in real time, all of implemented effects must consume less than 150 MIPS. They of course also have to fit into memory.

A. Delay effect

Delay effect schematic is shown in Fig. 1. This schematic is not complicated to implement (in terms of resource usage and algorithm complexity), as delay effect contains only one buffer of 4096 samples. Its functionality is described in chapter Algorithms. Buffer of 4096 sample is chosen. At sample rate of 48kHz, 4096 samples is approximately 85ms of delay which is enough for the effect to be audible. On the other side, target DSP processor has hardware support for circular buffer sizes that are power of 2 and number 4096 is power of 2. Resource usage for this implementation of delay effect is shown in table 3.

TABLE 3: RESOURCE USAGE OF DELAY EFFECT

MIPS consumption	2,784 MIPS
Data memory consumption	4296 words
Program memory consumption	396 words

From table 3. it can be seen that delay effect fits in real time limitations of DSP processor. Also, memory consumption is far lower than maximum allowed for any commercially available DSP.

B. Chorus and flanger effects

Core of chorus and flanger effects is vibrato effect, which uses time varying delay to create its output. Their

functionality is explained in chapter Algorithms. There is explained that variable delay is generated by sine function. For implementation, sine could be generated with LFO (low frequency oscillator) or sine table. Second approach was selected, in order to minimize MIPS usage. In first stages of implementation, where we did not have problems with memory limitations, sine table contained sine value for every sample. Because sine function is periodic function and its values are repeated in time, table contains values for only one period. This however is impossible to implement on given DSP processor (because of memory limitations), and also not necessary. For that reason, sine table size is reduced and sine values are calculated by linear interpolating between values from sine table. With this compromise we fit in memory limitations, but increase MIPS consumption. However, this is not a significant increase, as linear interpolation is simple to implement on a DSP. In table 4. are shown resource consumptions for both chorus and flanger effects.

TABLE 4: RESOURCE USAGE OF CHORUS AND FLANGER EFFECTS

MIPS consumption	15,45 MIPS
Data memory consumption	3578 words
Program memory consumption	654 words

Table 4. shows that both chorus and flanger effects are fitting in real time limitations of DSP processor. Also, memory consumption is far lower than maximum allowed for any commercially available DSP.

V. TESTING

Testing of complete implementation has been done in two phases:

- Reference code testing - listening tests
- Model 1-to-Final code testing - bit exact tests

Test suite included set of 20 test vectors, clean electric guitar streams, with which all three implemented effects were tested.

A. Reference code testing - listening tests

Listening tests have been done in every phase of implementation. For verifying reference code, listening tests are the only way to prove its functionality. Gaining access to high-quality professional digital audio algorithms is not easy. Intense competition among music industry companies results in virtually all professional audio DSP algorithms to remain closely guarded trade secrets [5]. Thus, for quality estimation of reference code were used some commercial products of same type, made by some of the famous electric guitar effects manufacturers (way that they sound like).

Listening tests were also done in next phases of implementation. There are very important, especially when changing from floating point to fixed point arithmetic, where it is very difficult to achieve bit exactness between the two stages. Results that are obtained in every phase of implementation are that implemented effects resemble commercial solutions and produce desired effects.

B. Model 1-to-Final code testing - bit exact tests

Bit-exact tests, as well as listening tests, have to be done in order to successfully verify all effects implemented. When implementing an algorithm on board, in order to test bit exactness it is necessary to align outputs in order to successfully and properly test implementation. When doing bit-exact tests on PC (models 0 to 2), previously recorded clean electric guitar streams are passed through effects and therefore we don't have problem aligning outputs, as they are already aligned. First phases of implementation were tested this way between themselves, floating point and fixed point arithmetic implementation on PC (emulation of fixed point arithmetic). In this case, bit-exactness wasn't achieved, however, we didn't expect that as the two arithmetics are different. Maximum bit error for all implemented effects was 3 bits, which is acceptable.

Final phase of implementation was to implement all effects on DSP. As mentioned, in order to successfully test bit exactness, board output must be aligned with output from second phase of implementation. It can be done manually or using tool that does this automatically, named RT EXECUTOR. RT EXECUTOR is a sophisticated tool for control, development and execution of BBT (Black Box Testing) automated test solutions [6]. Results that are obtained are that model 3 and board outputs differ in 1 bit for all tested effects with all test streams. With these results, we successfully verified all effects.

VI. CONCLUSION

In this work we implemented three popular electric guitar effects – delay, flanger and chorus on an audio DSP processor. Targeted DSP processor is CS48560, a commercial audio DSP processor with fixed point arithmetic, made by Cirrus Logic. Maximum consumption of resources were 33.68 MIPS, 11452 words of data memory and 1050 words of program memory. This is the case when all three effects are running at the same time, which is a very rare combination for these effects. After and during work development, all three of these effects were tested and verified using two methods – listening tests and bit-exact tests with set of 20 test vectors.

Future work would include implementing some other popular electric guitar effects, such as reverb, equalizer, distortion etc. in order to cover full spectrum of guitar effects and make an electric guitar multiple effects processor.

REFERENCES

[1] Vicerut Nonzee, Piya Poongbunkor, “DSP Audio Effects”, ECE 320 Final Project Paper , May 3, 2001

[2] Udo Zoltzer, “DAFX: Digital Audio Effects”, John Wiley & Sons, pp. 63-75, April 2005

[3] J. Dattorro, "Effect Design: Part 2 Delay-Line Modulation and Chorus", J. Audio Eng. Soc., vol. 45, pp. 764-788, Oct. 1997.

[4] M. Djukic, N. Cetic, J.Kovacevic, M. Popovic, “A C compiler based methodology for implementing audio DSP applications on a class of embedded systems “, IEEE International symposium on consumer electronics, 2008

[5] M. J. Caputi, “Developing Real-Time Digital Audio Effects For Electric Guitar in an Introductory Digital Signal Processing Class”, IEEE Trans. Educ., vol. 41, no. 4, Nov. 1998

[6] M. Popovic, J. Kovacevic, “A Statistical Approach to Model-Based Robustness Testing”, 14th Annual IEEE International Conference and Workshop on Engineering of Computer Based Systems, 2007.