The Use of Digital Signal Processing Techniques in Audio Effects

By Robert Chidlaw, Chief Scientist, Source Audio LLC

Introduction

I was asked to become a founding member of the Source Audio team, because they knew how important it is to create guitar effects that actually sound good to serious musicians. The other team members came from a big chip company and while they knew electrical engineering, they did not have the experience of being in the industry. Through my 20 years of experience as Chief Scientist at Kurzweil and 40 overall years of tinkering and building guitar effects and amps, I have learned a few things about how to make things sound good. I will describe in this paper some of the key things required to make effects circuits (especially digital ones) sound great.

Hardware

The first thing we need is a processor with sufficient computational horsepower to get the job done right. It also has to be inexpensive enough such that we can sell the finished product at a reasonable price. Our SA601 chip meets these goals. It is expressly designed for the efficient processing of audio signals.

Digital vs analog

I have built these kinds of signal processing circuits using analog hardware during the 1970's and 80's. In many ways, it's so much easier in digital. There are no issues with component tolerances, no stray capacitance between wires, and no hum. All active components (tubes and transistors (either singly or integrated into an op amp)) have some intrinsic noise level. There are noise sources in digital processing, but we can choose to arrange the details of the computation such that the final resultant noise is as small as we wish.

Control - variable resistors vs multiplication

In analog, controls are often implemented with potentiometers, which are variable resistors that provide a hand adjustable amount of signal attenuation. This attenuation can be used to control the frequency of an analog filter circuit. If this control is to be electronically varied, then a multiplication must be done

between two signals, the control signal and the audio signal. This is quite a challenge to do in analog without a lot of cost, noise, or distortion – you can't minimize all three simultaneously.

In the DSP world, the multiplication operation is just the same as what we learned in elementary school. It's just done a little faster. Our SA601 chip can do 48 million multiplications in a second.

Numerical precision

Now we get to the specifics of the DSP arithmetic. Word size is of critical importance. Every time a multiplication is done, the word length would have to double to preserve the exact answer. For a practical implementation with a fixed word length, we are constantly discarding data from the least significant portion of each multiplication result, and thus adding noise. Although the basic word size in the SA601 is 28 bits, support is provided for double precision operations such that the number of extra instructions required is not a problem. Using such a 56 bit accumulator where needed means that the unavoidable noise appears at such a low level as to be negligible.

The SA601 uses fixed point arithmetic, where the binary point appears five bits down from the most significant bit. Fixed point arithmetic is cheaper to implement in silicon than floating point. With floating point, part of the data word is used to specify a scale factor, which means that the same amount of precision is always available, for small numbers or large numbers. Fixed point programming is more difficult to write, as care must be taken that the computational values do not get too small (and lose precision) or too large (and clip.) But in the end it helps to force an understanding of exactly what is happening during each computation.

Division and table look-up

If you think back to grade school, you probably remember that division seemed a lot harder to do than multiplication. Well, that's also true for DSP chips. There is no division instruction in the SA601. If a division cannot be avoided, then a table of reciprocals is used. Interpolation is used to get adequate resolution from a coarse table. At times, a cubic interpolation polynomial is used to meet the required accuracy.

Table look up is used to evaluate trigonometric, power law, and exponential functions as well. Direct computation of these kinds of functions is quite out of the question. It also permits creation of empirically derived functions, whose exact shape is tailored to produce good sounding results.

Algorithms

Filters

Some of the filter algorithms are built using the digital state variable filter configuration. The various filter parameters have been adjusted by ear to sound interesting and be useful. The multi-peak algorithms, with up to three peaks or dips, have an awful lot of parameters. I can't claim to have explored the entire range of possible sounds, but certainly a good-sized chunk. That was one of the more interesting and creative parts of the project.

To evaluate these algorithms, there must be testing by musicians. Algorithms must be judged to sound good and feel and play right. It is a totally subjective analysis. During development, we had a lot of musicians come in and try out various algorithms, and rate them according to how much they liked them. A number of algorithms got both the lowest possible and the highest possible scores. It was very difficult to make the final choices. We tried to come up with a set that has a good variety of useful sounds.

The "classic" wah, on the other hand, is a digital implementation of the exact frequency response of the original wah circuit. It was more an exercise in mathematics than a question of what sounds right.

Transitions

All transitions from one sound to another are smoothed. When moving from one preset to another, or adjusting a front panel control, the resultant changes in the numbers that govern the computation process ramp smoothly from their current value to their new value. This avoids clicks and pops that might otherwise occur.

Control feel

The accelerometer signals must be mapped into control of the filter parameters in a way that provides a good natural

feel to the musician. This did not turn out to be all that difficult, but one is faced with interpreting comments by musicians attempting to describe what they like and dislike, and mapping those comments into changes in the code. This is where it is best if the designer is also a player.

Objective measures

Well, OK, there are some objective measures of performance. Noise is almost always bad. (Although at my previous job we put a noise source in a flanger algorithm to simulate the old noisy analog delay lines. Personally, I don't use it.)

Unwanted distortion is bad. In analog, higher order distortion components generally drop off rapidly. Digital distortion tends to have only very slowly decreasing amounts of high order distortion components, unless a lot of care is taken in how the distortion is simulated.

Aliasing distortion is particularly bad in digital, since it has no counterpart in analog. The harmonic distortion components that digital distortion generates march right up to half the sampling frequency and beyond. The frequencies beyond half the sampling rate are then reflected back down again, to lower frequencies that have no audible relationship to what went in.

There are still open questions as to the audibility and desirability of the inevitable small amounts of distortion always present in analog circuits. That's getting to be a lot of computation, to add analog-like distortion everywhere throughout a digital algorithm. At this point, we get back to subjective evaluations of whether such a high degree of precision in a model is worth the extra cost of the computations.

Summary

I don't feel that digital audio processing needs to have compromises today. If implemented correctly, it should be able to meet and exceed all the specs of analog gear. In addition, there are many things, with many more coming, that can only be done digitally.