Algorithms in Signal Processors

ETIN80

Instrument Multi effects Processor

Authors:
Erik Nilsson, Wang Yihe,
Wenbo Ye

Abstract

The purpose of this project was to implement a multi effects processor for an electric instrument or vocals. Some classic effects were chosen to implement on the SHARC ADSP-2126 DSP. These effects include chorus, flanger, delay and reverb. In order to implement a real time effects processor, constraints associated with embedded systems had to be taken into consideration. These constraints typically involved memory management. The project resulted in a functional multi effects process with potential for further development.

March 5, 2014

Contents

1 Introduction

Effects units that alter the output sound from a musical instrument or other sound sources have been largely available for consumer use since electrical instruments began to win in popularity.

Effects units were originally created by exclusively using analog components to manipulate the input signal of the instrument. Digital technology has had an important role in reducing cost and creating new possibilities for "exotic" effects.

Today effects units comes in a variety of form factors ranging from small stomp boxes designed to have one specific effect to rack mounted multi effects units relying on multiple DSPs.

The purpose of this project was to learn about the implementation of some well-established classic effects in a modern DSP and to understand how the DSP is well suited for the application of manipulating incoming signals in real time.

1.1 Delay

Perhaps the most classic and historically used effect is the basic delay. As one would assume from the name, it takes the input signal from a fixed time ago and outputs it along with the most current signal. Parameters for the user to control may typically include the delay time and gain of the delay. In more advanced implementations of the delay effect, the user may be able to manipulate the decay of the delayed signal and if the effect is in stereo; the stereo separation.

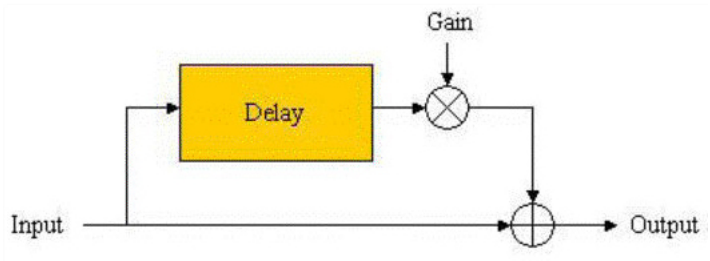

Figure 1: The Delay-effect

1.2 Chorus

Another classic effect is chorus. A chorus effect makes the output signal seem thicker and gives it a type of electronic shimmer. The implementation of the chorus effect is much similar to the delay effect. The main difference is the modulation of the delay time. This is achieved by using a LFO (Low Frequency Oscillator).

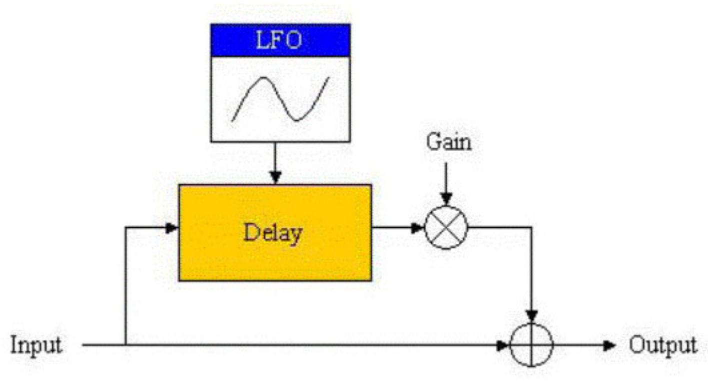

Figure 2: The Chours-effect

1.3 Flanger

The flanging effect is produced by mixing the input signal with a delayed input where the delay modulates by using a LFO. Part of the output signal is fed back to the input to enhance the effect. This feedback is the main difference from the chorus effect. The name "flanger" originates from when the effect was produced with tape recorders, one would put a finger on the flange or rim of the tape reel to slow down the speed, making it out of sync with the other identical recording.

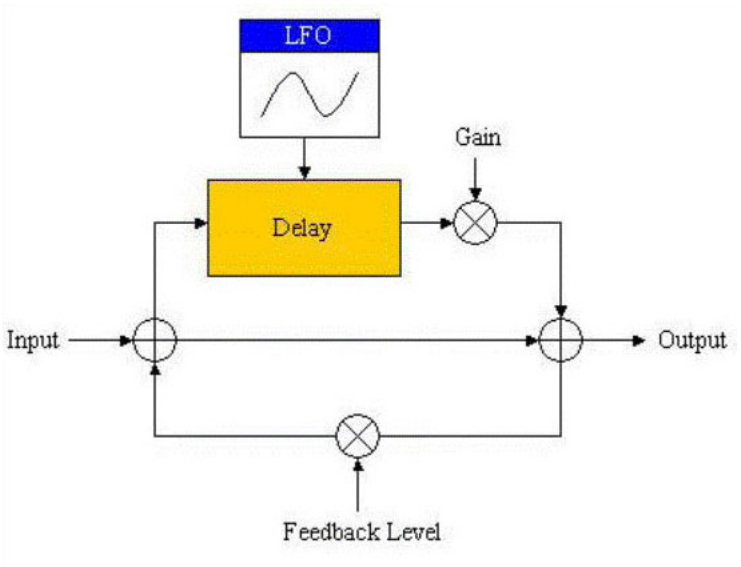

Figure 3: The Flanger-effect

1.4 Reverb

Reverberation is the persistence of sound in a space where the original sound was produced. A simulated reverberation can be produced by using multiple feedback delays to create a large series of decaying echoes. This is the method used in this project since the process of creating a realistic approximation of reverberation is fairly complicated and the time budget did not allow for more elaborate implementations.

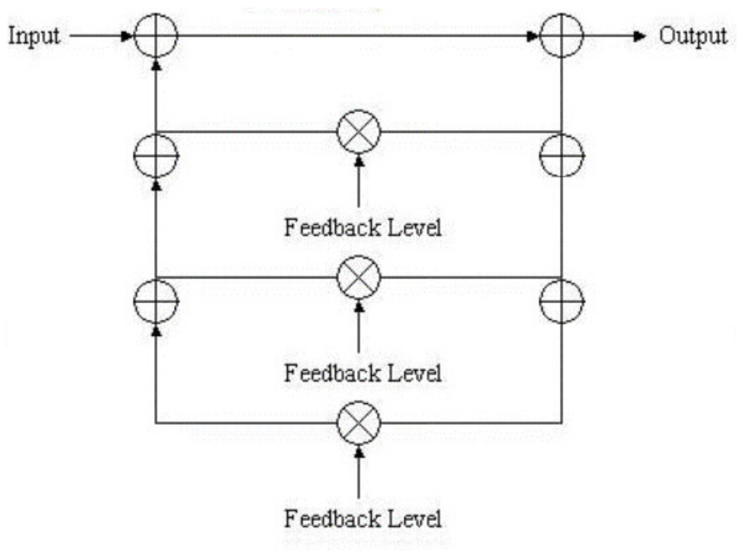

Figure 4: The Reverb-effect

2 Theory

All the effects described above relates to each other in the sense that they somehow need to make use of signals from the past that has been stored away in a buffer. The very basics of a delay effect are to take the stored signal from a specific time ago and mix it with the untreated input signal.

In the chorus and the flanger effect the delay is modulated between a minimum and a maximum delay value by using a LFO. The LFO consists of a waveform that may take the form of different shapes (Sine, Square, Triangle, and Sawtooth); in this project a basic sine wave function was chosen.

The basic formula for modulating the delay in the chorus and flanger effect:

$$D = Fs * ([maxDelay - minDelay] * [1/2sin(ft) + 1/2] + minDelay) \quad (1)$$

Where Fs are the sampling rate and f is the LFO frequency. In order to implement an approximation of the reverberation effect the output signal was fed back to the beginning of the buffer several times with different delay values. This feedback needs to be performed for every sample in the current block.

3 Implementation

3.1 Buffering of samples

In order to test the algorithms for the effects, they were implemented in Matlab. Since there are no constraints on time or memory when using Matlab it was fairly straight forward to get a working implementation of the delay, chorus and flanger. The reverb was not so easy to implement since it relies on multiple feedback loops to give an approximation of realistic reverberation.

When implementing the effects on the DSP consideration of real time results and memory constraint was the main issues. The SHARC ADSP-2126 DSP has 2 Mibit (Mebibit) RAM which makes memory somewhat scarce for buffering purpose. The DSP samples at a fixed rate, every samples contains two channels for stereo. In order to conserve memory and maximize the buffer size, only one channel was used. To be able to store one second of delay at 16000 samples per second, the same amount of samples needs to be stored.

If the global buffer was allocated and initialized in stack memory it could hold a maximum of 15000 integer values for one channel. This equals almost a full second of delay and is considered enough for this project.

The effects function in the code is registered as a hardware interrupt that arrives every time a block of samples is streamed in to the DSP. The block size is user defined just like the sample rate. In this project the block size was left at the default size of 32 samples.

In order to continuously write and read samples to and from the buffer, a FIFO circular queue structure was needed. The most straightforward way of achieving this buffer could be summed up as:

1. Shifting the entire buffer minus the last block to the left, thereby overwriting the first block in the buffer.
2. Write the last block to the end of the buffer.

This assures that the latest sample is always at the end of the buffer, making the read operating easy since one would only have to go T*Fs steps back in the buffer to get the delay of T seconds. This algorithm is however somewhat costly since almost the entire buffer has to be shifted for every block. Since the DSP runs at 200 MHz this is however not a big problem, to optimize the algorithm one would use pointers to keep track of the latest sample. The less costly algorithm then becomes:

1. Store the latest value at the global end-pointer index.
2. Update the end pointer using modulus to cycle the buffer.

This approach requires that the application checks that no negative index is produced when trying to access the buffer when the end-pointer refers to a location at the beginning of the buffer. This check may possibly be performed by the hardware to further optimize the implementation.

3.2 Calculating output

The effects function is executed every time a block of samples is streamed to the DSP. All the calculations are performed in a loop that goes through every sample in the block. In order to calculate the time for the LFO function a static counter is defined that keeps track of the current block. Knowing the sample rate and the block size enables the calculation from "block time" to time in seconds by using the formula:

```
double SECONDS = (BLOCK / DSP_SAMPLE_RATE) * DSP_BLOCK_SIZE;
```

that updates for every block. In addition with

```
double t = SECONDS + (n/DSP_SAMPLE_RATE);
```

that gets calculated for every sample in the block the time parameter of the LFO is calculated.

3.3 Cycle effects with keypad

The board were the DSP is located also provides a four button keypad. A push on a key generates a hardware interrupt in order to tell the application that a key has been pressed. The user is able to cycle through the effects by pressing the two buttons nearest the I/O. The two buttons located nearest the power input were used to cycle preset parameter values for the various effects. A basic switch-case block was used to handle an incoming key press. The application starts with no effect applied to the input signal.

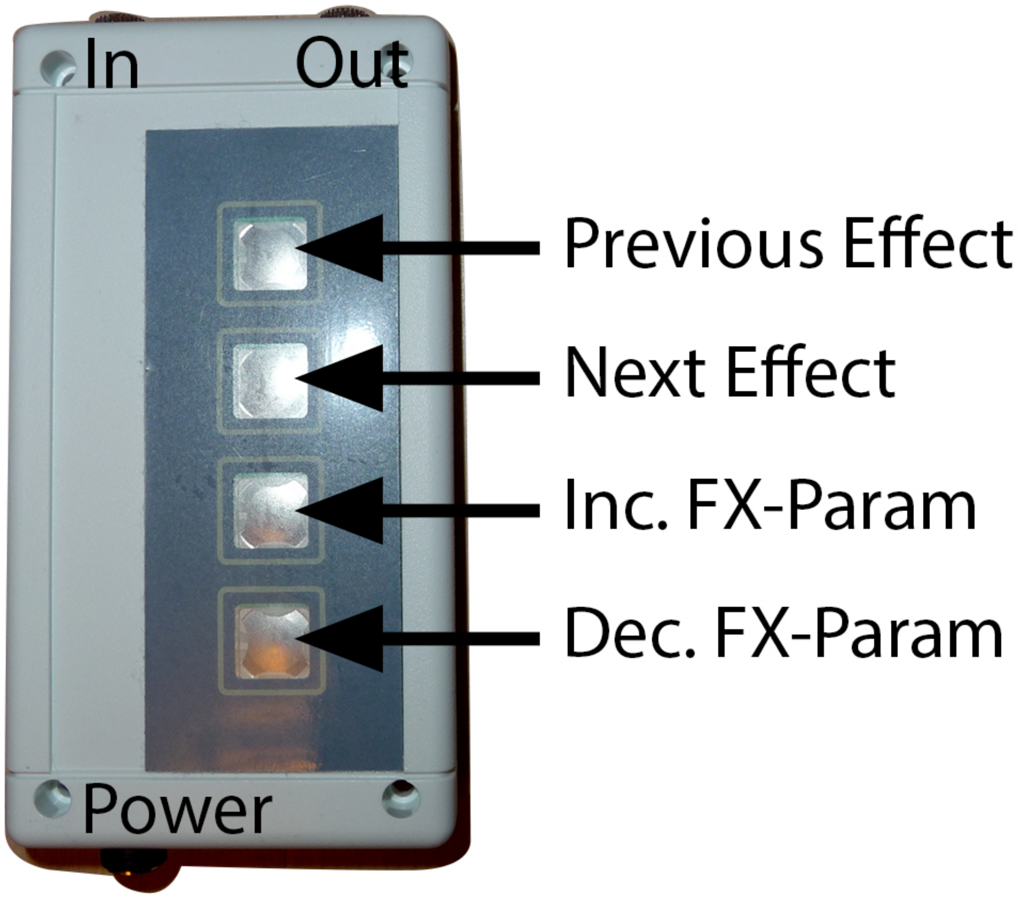

Figure 5: Button assignment

4 Result

The project resulted in a functional multi effects processor for vocals or an electrical instrument. The user is able to change the effects by pushing the two highest keys, and change parameters by pressing the two lower keys. The parameters are set at a few different fixed values and include delay time and oscillator frequency.

The perceived quality of the effects much depends on what type of instrument that is played and in what style. For an instance, an acoustic guitar sounds great with some chorus applied, but for a sharp clean electric guitar the sound may clip if too much chorus is applied. If the modulating frequency is too high, the sound is perceived as out of tune.

5 Discussion and Conclusion

This project has been very awarding since it provided hands on experience in manipulating signals in real time combined with overcoming the challenge of embedded systems hardware constraints. The project also contributed to a deeper understanding and how to practically apply theory involving Z-transformation and sampling of analog signals.

This project could easily be expanded by adding other similar effects or improving on the current implementation. The purpose of the project was mainly to learn about the inner workings of a DSP and the project could be viewed as a proof of concept.

6 References

[1] Chorus Effect Page `http://www.donreiman.com/Chorus/Chorus.htm`
Visited: 2014-03-05
[2] Chorus effect `http://en.wikipedia.org/wiki/Chorus_effect`
Visited: 2014-03-05
[3] Flanger Effect Page `http://www.donreiman.com/Flanger/Flanger.htm`
Visited: 2014-03-05
[4] Flanging `http://en.wikipedia.org/wiki/Flanger`
Visited: 2014-03-05
[5] Delay Effect Page `http://www.donreiman.com/Delay/Delay.htm`
Visited: 2014-03-05
[6] Reverb Effect Page `http://www.donreiman.com/Reverb/Reverb.htm`
Visited: 2014-03-05
[7] Reverberation `http://en.wikipedia.org/wiki/Reverberation`
Visited: 2014-03-05