COSC 4P98

Ring Modulator Technical Report

Marcus Bowden (2958221)

April 30, 2008

Introduction

The Virtual Studio Technology (VST) standard, developed by Steinberg, is a way in which programmers can create different tools for digital signal processing. These tools are considered plug-ins which can be cross-platform and link into a number of host programs, such as Ableton Live, Adobe Audition and & Fruity Loops, just to name a few. VST was developed to replace large analog hardware with small highly customizable software that could achieve the same effects. There are 3 main types of VST: Effects, Instruments and Midi Effects. Instruments create an emulation of a hardware instrument such as a piano or violin. Midi effects are used to change and manipulate a midi signal. The standard effects are used to take a pre-existing sound and manipulate it in countless ways. All of these plug-ins implement a GUI that can be programmer customizable so as to look as close to the original hardware as possible.

For my project, I have chosen to create a VST Effect called 'Narkotik RingMod'. This plug-in emulates the effect created by a hardware Ring Modulator which we will discuss further in the next section. I chose this effect because it was a simple way to demonstrate the functionality of a VST Plug-in while maintaining relevance to the topics of this course.

Ring Modulation

What is Ring Modulation? Ring Modulation, is simply the combination of two waveforms mixed together to create a new waveform. The name 'Ring Modulator' comes from the way the original Ring Modulators were created. They consisted of a ring of diodes were in the shape of a circle, or ring.

Ring Modulation takes the original signal, or 'Carrier', and multiplies it with some other signal, called the 'Modulator'. The Modulator can be any signal although it is usually some sort of oscillated signal such as a saw-tooth wave, sine wave, square wave or triangle wave, just to name a few.

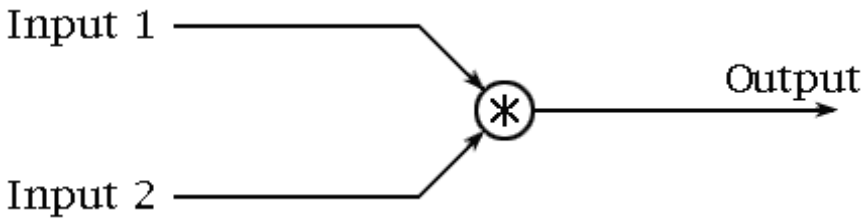

Fig 1 – Diagram of how Ring Modulation Processes Signals. (Harmony Central – Ring Modulation) (http://www.harmony-central.com/Effects/Articles/Ring_Modulation/ring-modulation-f1.gif)

With most Ring Modulators the Carrier is usually left constant while changes occur to the Modulator such as phase shifting, amplitude scaling (volume of Modulator) and increasing/decreasing the amount of Modulator signal to mix with the Carrier. The creation of a hardware analog ring modulator is quite complex and can be implemented in many ways. One way consists of a ring of four diodes and a few transformers to multiply the signals together. However, we are definitely more concerned with the digital implementation of a Ring Modulator. The digital implementation is very simple and only consists of actually multiplying the Carrier samples with the Modulator samples. The complex part is creating the Modulator wave so that it is accurate and doesn't completely overwhelm the Carrier since multiplying two signals will almost certainly create some clipping and fold over.

Narkotik RingMod Design

Narkotik RingMod is a simple Ring Modulator that uses the input from a host program as its Carrier and uses a customizable Sine wave as its modulator. This plug-in utilizes four parameters, three of which actually modify the sine wave. They are: Output Gain, Phase, Dry/Wet and Wave Amplitude. Output Gain is strait forward in that is controls the volume of the output of the plug-in. The Phase parameter modifies the phase of the oscillation of the Sine wave. Increasing and decreasing this

parameter affects the pitch of the sine wave. Dry/Wet determines how much of the Modulator is mixed with the original Carrier. Completely 'Dry' would mean there is no modulation occurring where completely 'Wet' would mean the output is a complete mix of the Carrier and Modulator. Wave Amplitude determines the volume of the sine wave. Similar to the Dry/Wet parameter, when the wave amplitude is brought all the way to zero, there is no modulation since the sine wave is just a muted wave. Conversely, when the sine wave is maximized the modulation effect is more pronounced. This is mainly used to reduce fold over and clipping.

Fig 2 – A screenshot of Narkotik RingMod in Fruity Loops Studio 8

The design of the plug-in is also very simple and is as follows:

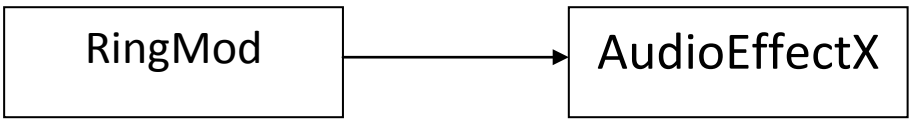

Fig 3 – Class Diagram

This class diagram shows the dependencies between the main classes. The AudioEffectX class is the base class that enables a programmer to implement a VST plug-in. In order to use it you have to create your own class that extends AudioEffectX. RingMod.h and RingMod.cpp inherit base methods from AudioEffectX and edits them accordingly in order to provide the effect implementation.

Narkotik RingMod Algorithm

The main method in a VST plug-in that actually returns the output back to the host is the processReplacing() method and the processDoubleReplacing() method. With Narkotik RingMod the double precision processDoubleReplacing() method was omitted. Here is the processReplacing() Method:

```
void RingMod::processReplacing (float** inputs, float** outputs, VstInt32
sampleFrames)
{
	float* in1  =  inputs[0];
				//in and out buffers for L and R
	float* in2  =  inputs[1];
	float* out1 = outputs[0];
	float* out2 = outputs[1];

	float tempOutR = 0.0f;
			//temp vars to hold changes
	float tempOutL = 0.0f;
	float tempInR = 0.0f;
	float tempInL = 0.0f;
	int position = 0;

	fBuffer = createSineWave(sampleFrames);
			//create the sine wave as a modulator

	while (--sampleFrames >= 0)
	{
		tempInL = (*in1++);
			//hold temp vars for inputs
		tempInR = (*in2++);

		tempOutL = ((tempInL) + ((fBuffer[position] * tempInL)));
			//multiply carrier(original) and modulator(dirty sine wave)
		tempOutR = ((tempInR) + ((fBuffer[position] * tempInR)));
		position++;
			//increment modulator wave position

		(*out1++) = tempOutR * fGain;
			//output the results with user determined gain
		(*out2++) = tempOutL * fGain;
	}
}
```

Fig 4 – RingMod::processReplacing() Method

First we see that the method parameters consist of pointers to the respective input and output buffers as well as a VstInt32 called sampleFrames which is just a 32-bit integer telling us how many samples there are currently in the input buffer. The next thing to note is the line:

```
fBuffer = createSineWave(sampleFrames);
```

This method createSineWave() creates a wave table which is filled with a sine wave of length sampleFrames. In the while loop, we see that the wave (fBuffer) is mixed with the tempInL and tempInR variables which hold the temporary values of the input buffer. This modulated value is then summed with the original value to create the mix. These left and right values are then multiplied by out gain parameter and then placed back into the output buffer.

Since the processReplacing() method simply multiplies the signals together so we should also note the createSineWave() method which actually creates and modifies the Modulator wave.

```
/*
 * createSineWave - creates a nasty modified sine wave with variable
 * amplitude and frequency equal that of the sample buffer
 *
 */
float* RingMod::createSineWave(VstInt32 numSamples)
{
	float* resultWave;
	float phase = 0.0;
	bool increasing = true;
	resultWave = new float[numSamples];

	for(int i = 0; i< numSamples; i++)
	{
		resultWave[i] = (fAmp)*sin(((phase+(PI/2)))*fDryWet);
		if(increasing){
			phase += fPhase;
			if (phase >(2*PI))
				increasing = false;
		}
		else if (!increasing){
			phase -= fPhase;
			if (phase <0)
				increasing = true;
		}
	}
	return resultWave;
}
```

Here we see that createSineWave will return a wave table (float array) that takes in numSamples as a parameter and bound for the wave table. The float pointer resultWave will hold the resulting wave table and the Boolean variable increasing will determine if the wave is on the way up or down. In my for loop, from zero to the number of samples, the result wave is equal to the Wave Amplitude parameter multiplied by the product of sine(phase+(∏/2)) and the Dry/Wet parameter. Again, the wave amplitude controls the volume of the sine wave and the Dry/Wet parameter determines how much of the signal is modulated into the carrier. We can also note the phase+(∏/2) part. Phase changes as the wave is moves up and down and is modified by our Phase parameter. If the wave is on the way up, phase will equal itself summed with our phase parameter. If the wave is on the way down, phase will equal itself summed with the negative of our phase parameter.

User Manual

Installing the plug-in is very simple. For our example we will show how to install and operate our plug-in with Fruity Loops Studio 8. The only file you need is RingMod.dll.

Steps:

1. Download RingMod.dll from www.marcusbowden.com/vst/download.html
2. Locate your Plug-in directory for your host. For FL Studio 8 the path should be: "C:\Program Files\Image-Line\FL Studio 8\Plugins\VST\"
3. Place RingMod.dll into the VST folder and open the program.
4. With Fruity Loops open. Locate your "Mixer – Master" window.
5. Click on the "Master" channel, or any channel you would like to apply the plug-in too.
6. On the right of the mixer, underneath the word "In", find an empty slot and click on the small down arrow.
7. With the effect menu open, scroll to the top and click "More…"
8. With the new effect window open look under "VST Plugins" and find "RingMod".
9. Double click "RingMod". The plug-in will come up and is now all set to use.

Note: Similar to these instructions, for most other hosts simply place the plug-in DLL in the associated plug-in directory and find that effect through the program.

Conclusions

Narkotik RingMod demonstrates the basic use of Ring Modulation and the use of Virtual Studio Technology by showing how a Carrier wave can be modulated by a Modulator. Using different parameters we can control how this modulator affects the original Carrier and just how much effect signal is to be processed with the Carrier. This project has just definitely inspired me to continue my work with VST plug-ins and other studio technology, hopefully as a 4F90 next school year. I hope this report has been informative. Enjoy Narkotik RingMod!

Sources

Ring Modulation – Wikipedia
http://en.wikipedia.org/wiki/Ring_modulation

Ring Modulation – Harmony Central
http://www.harmony-central.com/Effects/Articles/Ring_Modulation/

Virtual Studio Technology – Wikipedia
http://en.wikipedia.org/wiki/Virtual_Studio_Technology