DimensionalTips

“Tips for using Project 5's Amazing Dimension Synth”

Text authored by René Ceballos (creator of Dimension)
on the Cakewalk Project5 User Forum

Compiled into this format by techead (Bob Hammond)
of the Cakewalk Project5 User Forum
Updated November 7, 2005

Table of Contents

DimensionalTips: Dimensional Synthesis

Each element in Dimension can make sound in three different ways: sample-playback, wavetable oscillation, or waveguide. Which might result confusion at first is that the three synthesis types require a sample to play.

Sample Playback

I guess the first one is very well known: just drop a wav/ogg/aif/aiff/sfz file into Dimension, and presto. Any samplerate, any bit-depth, any format (mono/stereo), with any amount of loops. It will follow all the mappings in any sfz format file, with all the bells and whistles.

Wavetable Oscillator

In any synthesizer like Pentagon I or [[z3ta+]], you'll see a 'waveform selector', which selects the waveform for the oscillator. Common choices are Sine, Square and Sawtooth, with some others called Partials, etc. That waveform is a mathematical representation of a single cycle of the oscillator, which be repeated over and over to generate a tone.

One might think that this is equivalent to 'loop' a single wave file, but it's not. To make a transparent, pristine oscillator, a process called 'bandlimiting' needs to be performed. This is, removing some harmonics in the spectra that would generate a nasty distortion when transposed.

Dimension features all those common oscillator waveforms, and more: it allows you to convert *any* wave file in the oscillator representation. To indicate Dimension that we want to turn the element into a wavetable oscillator, there are two methods we can use:

Method 1: Inside the sfz file

If you create an sfz file with the following:

```
 <region> sample=MyWaveform.wav oscillator=on
```

The MyWaveform.wav sample will be bandlimited and converted on load, and the element will behave as oscillator. When using this method, the file can be any size. This method offers the advantage of allowing you to map different waveforms on the keyboard, layer them, etc.

Method 2: Direct sample-loading with less than 3000 samples

If you load a single file with less than 3000 samples, Dimension will say 'ok, it's a waveform definition'. This is the method used in all the factory wavetables. You can find them in the '00 - Wavetables' folder. Note that this works only for direct-loading: if you load the same file inside a .sfz file, you need to specify the opcode in (1). Note that the size of the file doesn't matter, as a 32-bit file will be four times bigger than a 8-bit file, both with 3000 samples.

So, when your neighbor asks "how many waveforms does Dimension feature?", you now know what to answer: "oh, just all".

Waveguide Synthesis

A waveguide is a bidirectional delay line, at some wave impedance. In easy terms, you can think of it as a complex resonator, where if you make some initial short or burst noise, the resonator will continue 'vibrating' for a while, with a natural decay. This synthesis method has been introduced in the early nineties as a fresh way to synthesize plucked attacks and plucked strings. A waveguide is the most common component used in Physical Modelling.

In this method, we give Dimension a sample to provide the initial noise burst, or 'impulse'. This impulse has nothing to do with room/hall impulse responses used in convolution processors, but using those as bursts result in interesting sounds. This is the sfz syntax to turn an element into a waveguide:

```
 <region> sample=MySample.wav waveguide=on
```

Beware of the length of the impulse you use: it might end in DC, with no sound produced. This is a normal consequence on waveguide synthesis. If you're up to investigate, just take a look at the stock impulses in 20 - Waveguide. If you're after a sample of how waveguides can sound, check the PM Clav patch, a physical modelled Wonder-like clav sound.

Hope this brings some light.

-René

DimensionalTips: Shifting The Piano

Hola friendus,

Back in the early 90s, the best available piano sound in a synthesizer was in the fantastic Roland Sound Canvas. Noone was using it of course, SC didn't qualify as 'pro' unfortunately, how one might imagine anyone who would consider him/herself professional using such a consumer thing. It was funny when Keyboard mag rated it higher than anything, right there with the bosen in the Korg 01W-ProX. Not always the best things are labeled 'pro'

Hmm... what was I about to write anyways? ah! shifting the piano.
It happens very often when you have only one piano sound in your synth/sampler, that it sounds great, but... you get bored of it. You turn it on every day of the week, and it happens to sound always identical. Temperature and humidity doesn't seem to affect samples as much as the real thing. It never goes out of tune, and you can't use duct tape on samples. It also happens with other sounds, but the piano is... well... the piano.

Here's a trick you can use on Dimension pianos to have a new one anytime you want it. It's a trick some synthesizers have used for a while, so it's not really an invention, but very handy.

As you surely know, when you load a multisample in Dimension you're actually loading many wav files. Those wav files were recorded for several keys in the keyboard, then mapped so when you play say G3, the real sample taken from G3 from the original piano plays.

Dimension features two parameters to define the pitch of a multisample: Transpose and Shift, both in the main display. Transpose moves the whole multisample up or down, while Shift changes the pitch of the notes without changing the mapping.

The trick of 'shifting' the piano consists of transposing the multisample, and shifting the multisample map in the opposite direction. This might sound chinese but it's very simple: Just set Transpose to 4, and Shift to -4.

When you transpose a multisample down, and shift the pitch high, the result is a multisample playing at the same pitch but the tonal characteristics are affected: the original G3 sample is now playing B3, but tuned to G3, for instance.

I don't think it's that easy to understand till you experience it. Therefore, I've prepared a small p5p project featuring a quick piano improvisation in a shifted to 'dark' piano. Just open the Dimension instance and tweak Transpose and Shift so you can observe how it works.

http://www.rgcstuff.com/External/Tips_ShiftingThePiano.p5p

Here's a mp3 for those who didn't receive their P5II yet. And for those who didn't buy it yet, if such a thing exist. This is one of the smallest pianos in Dimension (14Mb), no effects or processing whatsoever. No mistake-fixing nor quantizing either, sorry. Natural is natural

http://www.rgcstuff.com/External/Tips_ShiftingThePiano.mp3

Hope you like it. Shift the world!

-René

DimensionalTips: Groovah!

You might have already seen that we have a couple of cathegories called 'Drum Grooves' and 'Musical Grooves' in Dimension's program browser (did I mention you can resize the browser?). Those cathegories are synchronized loops, and here's how Dimension plays them:

1- Load any groove, from 05 - Drum Grooves for instance.
2- Now play C3.

That's it. You're now hearing the groove, synchronized to P5II tempo. It wasn't that hard, was it? you can now draw a single note in the editor, and you have your groove going on.

All dsp processing in Dimension can be applied to it, and that's the most interesting part. There're a few examples under 05 - Drum Grooves\A few processing examples.

The examples show how to apply pitch changes, filter freaking, pan drive-to-insanity random movement and other nice tricks. Don't miss those.

Anyways, the C3 trick will make it for many uses. There's an 'extension' to C3, which works by using the mod wheel on it (CC1). Once you activate the mod wheel, the loop plays each slice backwards. Nice to combine extreme pitchbend ranges with backward playback.

If you're a preset-player-only, you can stop reading now. I'm pretty sure you have enough to keep you entertained making strange sounds which will increase your popularity in the neigborhood for weeks. Too bad you won't know what others will, but well, it was great to have you here.

Ok, for the rest. The fun begins when you:

1- Open Dimension editor
2- Click in any place with no controls (i.e. the logo), and without releasing the mouse button...
3- Drag into the Dimension track in Project 5 arranger.

Dimension will create a MIDI file for the currently selected groove and P5II will openly receive it. It will play each slice when it's time to. Like if the arranger would be playing it for you. Some of you using RMX or a few other proggies might be familiar with this.

Slices start on C4, and go up in one-semitone steps. Now, it's your turn of being creative. Some options this feature open are:

- Identify a couple of single-hit slices and use those as kick and drum, leaving the

rest of the groove unused.
- Repeat notes
- Slide notes to create a different groove feel
- Rearrange notes
- Combine two grooves using alternate notes
- Record or automate CC1 to have some slices backwards and others forward

But most likely you tweakers will find more interesting tricks to play with it

-René

DimensionalTips: ModWheel

This is a very easy one, perhaps obvious. However, I think it's worthy to mention.

You surely have seen many patches in the browser where explicit use of the ModWheel is indicated. For instance, the E175 Gibson guitar used in 'Elevator Radio' performs the slide-up gesture when notes are pressed while the mod wheel is activated.

But what might not be so evident, is that when you load a patch from '01 - Dimensions' or '08 - Pads', you're not loading a program. You're actually activating a powerful brain disturbing, mind boggling machinery with unspecified long-term effects. That machinery has an 'intensity' control, which regulates how much your perception is affected and your senses are overriden by the machine.

Engineers have tried to explain the phenomena saying that actually, mod wheel is attached to several parameters: LFO depth and speed, cutoff, volume and others, in the four elements differently and individually, with different depths and different smoothing factors which control the time of application the effect, resulting in complex sound animations. They have also said that there's no real magic but only tricks the sound designer played. I don't believe that at all, and I think the 'mind control intensity' term describes what really happens best.

Here's a small project I prepared with a program called 'Enigmantic Voices'. I recorded a clip, then copied it. Then in the second instance I recorded some modwheel automation on top.

http://www.rgcstuff.com/External/Tips_ModWheel.zip

So now, get every Dimension and Pad and use the modwheel on it. I've heard that soundnerds enjoy having the MIDI Matrix window visible and excersize their geekdom trying to predict what will happen when they move it.

-René

DimensionalTips: The Loop Wave

Standard Windows wave files are one of the sample types Dimension eats. As you most likely know, a wave file has audio inside, and it can include some extra information as well.

A particularly useful component of that extra info is the loop information. Samplers have used looping for decades, in order to allow long sustains without spending big amounts of disk and memory. In few words, the trick consists in finding two 'equivalent' points within the sample, and loop that part: once the player reaches the loop end point, it would start on the loop start point, over and over.

However, a not-so-exposed aspect of loop information is that the wave headers allow standard wav files to include -multiple- loop definitions. It also provides tags for each loop to have its own repeat count and direction (forward, backward, alternat), besides the start and end points. This opens many doors for creative options.

The traditional approach for most sample-playback devices is to play only the first loop, forward. Dimension can read up to 16 loops from any wav file, and play those as defined. Most wave editors including Sound Forge and Audition have tools to define those loops, including repeat count and direction.

Here's a sample wav I've prepared, with several loops defined. First, please play it in Winamp (or whatever you use to play wav files), then drag it into Dimension's gui and play middle C. You'll hear all loop definitions, with different lengths, repeat counts and directions.

http://www.rgcstuff.com/External/ZetaArp.wav

Using the sfz format, each loop can also be tuned, and some other neat tricks can be performed. That'll be for some future tip.

-René

DimensionalTips: Naturally Released

This tip is about amplitude release. You most likely have experienced the situation where you've been tweaking the release of the Amplitude Envelope for a while, and even after making it really long, it still doesn't sound natural.

The default shape for the envelope curves in Dimension is variable-power. This means, you'll see curves based in the formula x^k, where k is variable (using the shape control). When k=1 (which happens when you double-click on the shape to reset it), then the envelope segment is linear, the most common envelope type used in synthesizers.

Variable power envelopes are great for most uses: they offer symmetry in the shape manipulation, they're very easy to turn back and forth into linear and they make it easy to draw complex shapes using multiple nodes. When applied on release times, they're great CPU savers, as the release segment goes quickly to zero, ideal when we're making sounds where we'll apply heavy delay or reverb, therefore getting an exponential tail anyways.

OTOH, they just don't sound excellent when applied to solo, exposed instruments where we pretend a natural, room-like decay. There we need segments following the k^x expression.

Fortunately, Dimension features both variable-power and exponential curves. So, how do we turn a release segment into exponential? Here's how:

1- Click on the envelope window to give it focus
2- Highlight the ending node for the segment, by hovering on it.
3- Press the 'N' key (as in 'natural').

The segment will turn into dark blue, meaning it's now exponential. Couldn't be easier right?
Here's two suggestions:

1- For truly natural releases, make sure the last node goes well down to zero. The Master Envelope will take care of the final decay from the latest node to zero, so the curve effect might be lost.
2- Adjust the shape as usual. If the latest node goes down to zero, you can imagine the envelope segment as if it were 'continuing' its natural decay. This can also be imagined as adjusting the release time for the exponential curve.

Here's a small video describing the procedure. Too bad it doesn't show the 'S' to mark the sustain point nor the 'N' to mark the natural release setting, but I'm sure you'll get it anyways.

http://www.rgcstuff.com/External/NaturalRelease.avi

And this is the 'bonus' for those who *do not* want natural releases. Yes you know I'm looking at you. The N segments can also be used in the middle of the envelope train, and adjusting the shape you get a nice 'jump', great to... uhm... to... well, just great

-René

DimensionalTips: Master of the Amplitude Envelopes

I'm sure you've seen or created an amplitude envelope curve following the User Guide directions, and wondered "hmm... if my last node doesn't go down to zero... will I hear my sound day and night... forever?"

I have a very straignt answer for that question: no idea.
Err... <cough> no idea why I didn't mention this before, I mean.

As having multi-segment, multi-point, multiple and adjustable shape, loopable envelope where each segment can be adjusted for velocity and keyboard tracking individually would have clearly insufficient you tweaker souls, we thought that we could give you some more by making the Amplitude envelope actually composed of two envelopes, interacting in an out-of-this-world fashion.

Correcto. There's two envelopes for each Amplitude envelope: a 'Master' envelope which does the hard and boring work, and a 'Creative' envelope, which is the one you can edit with the Flexible eg. Here's how the combo works:

When the Amplitude envelope is turned off, only the Master envelope is working. This takes care of moving the amplitude to zero when you release the keys you're playing, and nothing else. Once you turn the Amplitude envelope on, then the Flexible envelope generator works obeying your carefully sketched path, till the last node or till the Sustain Node reaches (the one you mark with S). Then it remains there for a while, till you release the key.

Once you release the key, then the Envelope passes thru all segments you have created *after* sustain point, till last node is reached. If the last node is zero, then we're all set and both envelopes stop working and go to bed.

However, if the last node is not set to zero, someone has to do the dirty work of moving the sound to zero. Then the Master Envelope takes over the situation and does the job, while the Flexible eg goes partying.

That's the most important job for the Master Envelope. The 'other most important job' it does, is offering a MIDI-controllable Amplitude Envelope, which is applied *on top* of the Flexible Eg. So yes, you can have both working all the time for all four elements.

Yes, we can control the Main Attack using CC73, and the Main Release time using CC72, as in the MIDI spec. Those might be handy and cool to write down somewhere 😊

Hasta la vista.

-René

DimensionalTips: sfz format 101

Ok. As you have surely heard, the main format for Dimension is the sfz format. I thought it would be good to start writing a few lines about the sfz format, as I feel that many of you might be scratching your head wondering what it is, what it could do for you, and how it's used. So I think the top questions/answers are:

What is a sfz file?
It's a text file, containing information on how a group of samples have to be performed.

What can I use to write that text file?
Notepad.

What do I have to write inside the file?
A simple sequence of opcodes, plus the filenames of the samples, using a special syntax. Most of that syntax opcodes is detailed in the sfz format definition page (not all). The sfz format has been greatly expanded and enhanced for Dimension, so keep an eye in that page and this forum as more information will eventually appear.

Ouch, so there's no graphical editor?
No, there isn't a graphical editor for the sfz format yet. Graphical editors have advantages and disadvantages, as a text-based format does. If you really 'need' a graphical editor then this is a good time to stop reading this tip, and try to get the job done in DS864, or some of the major samplers like HALion, VSampler, Kontakt or Gigastudio.

Ok, if there's no graphical editor, then what's interesting about the sfz format?
Many things, I'll mention just a few:For heavy-weight content producers, who keep sample collections organized after recording, using graphical editors to map the same structure over and over again is a huge productivity loss. With the sfz format, it's just a search... replace operation.

Example: user A records samples from a synthesizer, every minor third. He carefully writes all wav files using the same convention (i.e. "Solina-C3.wav, Solina-D#3.wav, etc.". He got 20 samples, and assembled the .sfz file. Now he records a Moog, with the same mapping, resulting in a similar wave collection (i.e. "Moog-C3.wav, "Moog-D#3.wav, etc.". Creating the second sfz file is just find...replace all 'Solina' instances with 'Moog'. No need to use a graphical mapper to do the job again. Let's say we got a 143 synths samples... ok you got the point.

The sfz format also allows people to publish mappings and original arrangements of the whole myriad sample contents available for free over the internet, without actually distributing the samples. This avoids any copyright infringement, distribution rights dispute, requirement for written permissions, and other evil legal tricks in the sampling industry.

Example: user A downloads some free samples S from site X. However, as *most* of the free samples in the internet, his license says he doesn't have rights to redistribute the samples (this is very logical, as most free samples are offered as promotion) so user A cannot show to the world how fantastic his contribution, based on modifying the samples is. The sfz format allows user A to publish a simple, small and universally readable text file, then point his friends to the samples.
User B could take the work user A did, and modify it again, then publish another .sfz file. Then user C and D, and more letters. With other formats, (like Akai, SoundFont or Gigastudio), this would force users to download the total sample contents over and over again, even when all the changes would have kept the original samples unchanged.

The sfz format simply offers more sample manipulation capabilities than all the major samples in the industry together. Some of its features are unlimited keyboard and velocity regions, sample trigger based on midi continuous controllers and special controllers, round-robin, random, exclusive zones (choke groups), detection of legato playback, playback in-sync with host tempo, synchronized loop playback, etc.

Ok, I might give it a try. How do I get started?
That's great to hear, here we go!

An sfz format file is arranged in regions. A region is a statement of WHEN and HOW a specific SAMPLE will play (the three uppercase words are the keys to the sfz format). So to start a region, there's a specific opcode:

```
<region>
```

That means 'The region definition starts here', and it will extend till the next <region> is found. It doesn't matter if you use lower/upper/mixed case, or how many spaces you put in the middle of the file. I'll use lower case in the examples for simplicity. There's one very important rule: there should *not* be spaces at the sides of the '=' sign.

A region has three parts, all of them optional:

1- A sample opcode: 'sample=' (indicating what SAMPLE to play).
2- A set of Input Controls (WHEN the sample will play).
3- A set of Performance Parameters (HOW the sample will play).

Great. Now let's see a real example. Let's suppose we have a sample called 'Solina-C3.wav'. We'll create a sfz file using it. It goes as follows:

```
<region> sample=Solina-C3.wav
```

I've prepared the wav file and the .sfz file for you so you can check it out without writing anything. Just extract the zip anywhere, and then drag the .sfz file into Dimension's gui (keep the .sfz and the .wav file together).

That sample sfz file maps the sample Solina-C3.wav to the whole keyboard. Pretty uninteresting, as we can just drop the wav into the element for the same result right? Ok, but we had to start somewhere. Now just for fun, add this line to the file yourself:

```
<region> sample=Solina-C2.wav
```

So the whole file reads:

```
<region> sample=Solina-C3.wav
<region> sample=Solina-C2.wav
```

I've shipped the other .wav file in the same zip. So just write, save, drag the .sfz into Dimension's GUI and check what you got: a nice stack of the two samples layered into the whole keyboard!

I now seem to hear that voice in your head repeating "this is somehow interesting... how many samples I could stack?". As many as you want. There's simply no limit.

That was too easy. Now let's try to make a split example. Here is how it would look:

```
//----------------------------------------------------------------------------------
// this is a four-regions keyboard split example
//
//
//----------------------------------------------------------------------------------
<region> sample=Solina-C2.wav lokey=0 hikey=b2 pitch_keycenter=c2
<region> sample=Solina-C3.wav lokey=c3 hikey=b3 pitch_keycenter=c3
<region> sample=Solina-C4.wav lokey=c4 hikey=b4 pitch_keycenter=c4
<region> sample=Solina-C5.wav lokey=c5 hikey=127 pitch_keycenter=c5
```

That's it. You might have noticed that I used some lines starting with '//'. Those are comments, and they'll be ignored by the player, but they help to keep things clear.

First region goes from 0-b2, second from c3-b3, then third from c4-b4 and finally fourth from c5-127. You can check how the splitted map sounds now. As a homework, I would like to have a dual-layer, octaved and splitted map together, combining both previous examples. Candy for the first who gets it. Using the same samples.

So in the example above we see several opcodes. the 'sample=' opcode is the Sample Definition opcode, and signals what sample the region will play. The 'lokey=' and 'hikey=' opcodes define what are the lowest and highest key for that region: when any key inside the lokey-hikey range is played, that region will play.

Finally, we see the 'pitch_keycenter=' opcode. This one tells Dimension that what is the key that should play the sample *untransposed*, as the sample has that note recorded inside.

You might wonder why the wavefiles never stop when you play a chord. That is because the waveforms have been looped, and contain loop information tags inside. For more info about looping, check "DimensionalTips: The Loop Wave".

Nice. Something else for today?
Yes. we'll finish with two more opcodes. With the ones you got so far, and the next two ones you'll be able to do 95% of the tricks you've seen in any hardware synthesizer or sampler. So even if you don't plan to become an sfz guru, these are the ones you need to know.

So far we know 'lokey=' and 'hikey='. We also have 'lovel=' and 'hivel=', to indicate what the minimum and maximum velocity are for the region. Using a previous sample:

```
<region> sample=Solina-C3.wav lovel=64 hivel=127
<region> sample=Solina-C2.wav lovel=0 hivel=63
```

Now, the Solina-C3.wav sample will only play when velocities above 64 are played (play hard in the keyboard), and C2 will sound when you play soft. Combining lokey/hikey and lovel/hivel is possible to create keyboard splits and velocity splits. You might have heard that some formats allow 8, 16 or 32 keyboard/velocity splits, and I can hear that voice inside your head again: "how many keyboard/velocity splits can I create?". Same answer as before: no limits.

Just as a reference, the whole set of Dimension multisamples is in the sfz format, and was crafted using Notepad.

Those are the real basics, I hope that something in the sfz format will seduce you.

-René

DimensionalTips: Simple Gate

This tip is mostly for P5/Dimension beginners, and shows how to 'assemble' a very simple gate effect, as used in almost every trance/dance song.

Dimension doesn't feature a built-in trance gate editor, but that's just because it doesn't need one. Project 5 has the most powerful gate editor one might imagine, so we'll take advantage of it. The procedure is as follows:

1- Select the Dimension program you'd like the gate effect applied to.
2- Open the MIDI Matrix in Dimension
3- Add a single line with the following values:

SOURCE: CC25
DESTINATION: Volume All
DEPTH: -96
SMOOTH: 0

This instructs Dimension to control the volume of all elements using CC25, with a depth of 96dB. This means that when the value we send is maximum, the volume will be -96dB, so we won't hear it.

4- Open P5 editor.
5- Select the Automation Tool, select MIDI CC 25
6- Draw your gate effect as desired.

Presto. Now the pattern can be saved as 'Trance Gate 1', and we can apply it to other Dimension programs, after adding the MIDI Matrix line.

I've created a small test example to show

http://www.rgcstuff.com/External/Tips_SimpleGate.p5p

In that example, I created yet another MIDI Matrix line to automate cutoff with another pattern. Hope you find it useful!

-René

DimensionalTips: Stack Yourself

This one is intended for tweakers and advanced users, or for anyone wanting a powerful oscillator stack in Dimension. If you're up to follow it, I suggest you get a good cup of Colombian coffee at this point.

In 'sfz 101' we covered the basics on how to map samples in the keyboard in a very simple way. A few samples included how to create keyboard or velocity switching, and how to stack and layer different samples. Now we'll cover the same subject, but applied to wavetable oscillators instead of samples.

As you surely know from reading Dimension's Users Guide, if we load a wave/aiff file directly in Dimension, and the file has less than 3000 samples, instead of treating it as a transient sample Dimension will use the sample contents to create a wavetable oscillator on-the-fly.

Dimension will assume that the sample loaded contains the definition for a single-cycle of the desired oscillator waveform, and will resynthesize that waveform to create a full-spectrum oscillator with zero aliasing based on that cycle.

However, if that transient definition is included as a sample inside a .sfz file, Dimension will treat it as a sample, unless we instruct it otherwise. Here's how we say that we want the sample to be treated as an oscillator single-cycle definition:

```
<region> sample=saw mini.wav oscillator=on
```

That means that the sample 'saw mini.wav' will be considered as a single-cycle waveform, just like if we'd load it directly (and it has less than 3000 samples).

Ok. Now that we know that, we can use the same keyboard/velocity switching and stack/layer tricks we've learned in 'sfz 101', but with oscillators. Using those, we can assign different oscillators to different key ranges, velocity ranges, we can stack oscillators, and we can stack or switch oscillators with samples. Cool innit?

Let's create a simple example to showcase the oscillators stacking majesty. The idea is to create a powerful stack of multiple saw waveforms, like the 'multisaw' or 'supersaw' stacks found in other synthesizers, with adjustable detune.

We'll start by creating a new text file, inside your $\Multisamples\Wavetables\ folder, so we can use an existing wavetable to create our stack. Rename the new text file to 'multisaw.sfz', and then add the following lines to it:

```
<group>
sample=saw mini.wav oscillator=on oscillator_phase=-1

<region> pitch_oncc140=-20
```

```
<region> pitch_oncc140=-10
<region> pitch_oncc140=10
<region> pitch_oncc140=20
```

Then save it, and we're done. Now we have a powerful stack of four saw waveforms, and we've assigned the detune control of each layer to the unused-in-this-patch BITRED knob in the interface. If you've followed the instructions carefully so far, you should be saying 'Whot?' by now while scratching your head. Ok, please stay with me. Let's examine the sfz definition piece by piece.

First we have the <group> opcode. A <group> opcode is a way of simplifying the definition file, by saying 'what I say now goes into every region that follows'. Therefore, anything we write after <group> will be applied to all regions following, till the end of the file or till the next <group> opcode arrives.

Then we have the well known 'sample=' opcode, which tells Dimension that we want to create our oscillators based on the 'saw mini.wav' single-cycle definition, which is in your \Wavetables folder by default. Then we have the above mentioned 'oscillator=on', which tells the player we want an oscillator, not a sample.

Then the first unknown opcode appears: we have an 'oscillator_phase=' thing, (which is new for 2.0.1, requires the patch applied). This opcode tells Dimension what the initial phase in the cycle should be for the oscillator. For z3ta+rians, there's nothing new here: it's the definition equivalent to the PHASE slider, other synth users might also find this familiar. What it is new here though, is the fact that if we specify -1 as value (as we do in this example), the phase will be randomized for every new note. This allows us to escape from the huge pop on note-on that would cause four saws stacked in-phase.

Then we've got those four <region> lines. We've defined the main opcodes within the <group> header, so those will be applied to every new region we create. Therefore, we only need to specify those opcodes which are particular for each region now.

In each region, we've used the simple 'pitch_onccXXX=' opcode to 'link' the pitch of the oscillator to a physical controller. In this case, you just move your MIDI CC 140, and you'll be controlling the oscillator's pitch, in the specified depth in cents. Oh right, CC 140 doesn't exist... does it?. We're almost there.

All dimension knobs are mapped as if they were MIDI controls, with values that the midi spec can't reach. This allow us to 'recycle' the interface knobs for specific purposes within an sfz file, when a particular knob is unused. Bitred knob is 140, and that's the one we've used in this example. So what we've done is linking the pitch of each oscillator to the bitred knob, with a different amount in each layer (Note that the knob will work even when the button is in the 'off' position. You should only turn it on if you want the bitred effect applied at the same time).

That's it. If you're too lazy to create the sfz file yourself, you can still try it. I've created the file for you, you can get it from here:

http://www.rgcstuff.com/External/Multisaw.zip

There're two files inside the zip: 'multisaw.sfz', intended to go in your $\Multisamples\Wavetables\ folder, and 'multisaw test.prog', which you can place anywhere. The latter is a test file program with a trance-oriented patch.

Simple experiments could include modifying the sfz file to get seven saws instead of four, making the detune amounts more extreme, etc. Hope you find this tip useful.

-René

DimensionalTips: DefaultRegion

This is another tip aimed to advanced tweakers, and for those who want to dive into Dimension's architecture to take advantage of every bit of it when designing sounds, or just for fun.

If you take a look into your \Dimension folder (most likely under c:\Program Files\Cakewalk\Shared DXi\, but depending on where you installed it), you'll see there're a few files inside. Let's focus in one file called 'DefaultRegion.sfz'.

The 'DefaultRegion.sfz' is a partial sfz definition file. It includes several opcodes, and every opcode inside it will be added to the definition of *every* region in any file you load into Dimension. As factory-shipped, this file includes all standard MIDI CC, Interface, EG and LFO bindings.

Let's examine a few lines of it:

```
/ lofi
bitred_oncc140=100 / 140 = bitred control
decim_oncc141=100 / 141 = decim control
```

This is the line which links the interface knobs to the bit reduction and decimation depth controls respectively. The 100 means '100 % of depth', and the _oncc140 and _oncc141 are respectively the bitred and decim knobs in the interface.

Ok so, what this file is useful for? As anything you write as definition inside DefaultRegion.sfz will apply to every region defined, you can use it to 'preset' your MIDI setup, with specific purposes.

For instance, let's say that I would like to have a dedicated MIDI control (cc20) assigned to decim, and which would work in every program having the decimator on. Then we would just add:

```
bitred_oncc20=100
```

anywhere in the file, and we're done.   If you examine the file to the end, you'll see the links for all LFO definitions as well. We're using 5 LFOs in Dimension, 'connected' to the main parameters (pitch, cutoff, reso, pan and amplitude). However, there're more LFOs available, well hidden in the dark. Possibly you'll find a way to use them.

WARNING: If you decide to edit this file, please make sure you have a backup copy of it in a handy and safe place. Mistakes on editing this file might break some functionality in Dimension.

-René

DimensionalTips: CPU under control

You might have noticed that we got a nice CPU usage reduction in 2.01, which will make every project run smoother. The bad part of it is that your room won't stay as warm as it used to.

Anyways, as projects are more complex everyday, here's a list of tips to preserve CPU, when using dimension in a multiple-instance context. Note that I won't include here the system-level tips (like getting a SSE/SSE2 capable CPU, audio drivers, latency, Windows settings, etc.).

- Know your programs

As in every softsynth, every factory preset was tweaked extensively, including several control routings which are commonly used on modulation wheel, aftertouch or other gestures.

Sometimes there're sound layers which are almost silent, but which will become alive when modwheel is activated. There're chances that you have a few of those programs in use, while you don't have plans to apply modwheel in that project, as the sound is useful without it.

Result: several 'hidden' layers of sound being calculated, with minimum or no impact in the final sound result.
Cure: Solo every track, open Dimension editor and turn off the elements one at the time (using the four on/off buttons in the mixer). If you don't hear a difference once the button is of, then keep it off.
If you think you'll be using that preset in a future song, save a new program. The same name, with a '-eco' suffix will do it.

- Bus your effects

Many programs in Dimension make an extensive use of effects, as every patch is intended to be 'finalized'. When you mix several programs, you might find that they don't mix too well. Most likely, that's because they have different effect settings, especially the reverb.
Additionally, if you happen to insert a new program which has a reverb running, you'll soon run out of CPU.

Result: Bad and expensive mix.
Cure: Turn off the reverb in all programs, and use a common reverb in a bus. Same for other effects.

- Automate ENABLE

There're chances that you have a few instances of Dimension being used only in the intro, while others are used only in the bridge, and others in the ending.

Result: the idle load of all instances is hitting the CPU at all times.
Cure: Use DX Parameter automation on the ENABLE parameter, to turn off the unused instance when they're not supposed to play.

- [tweakers] Lower interpolation order on realtime

Given that Project 5 features a superb freeze implementation, this one will hardly be needed. But in any case, it's good you know about its availability. It will also mean lowering the realtime playback quality output of Dimension. The 'lowered' quality is due linear interpolation will be used for sample-playback (it won't affect wavetable oscillators).

As a reference, many well-know major samplers use that kind of interpolation, so it might still be good for most purposes. If you tick the 'Use sinc interpolation... ' option, it will not affect freeze/render, which will still use high-order sinc interpolation.

Add the following registry key under

```
HKLM\Software\Cakewalk Music Software\Dimension
String: "Realtime Quality"
Value: either "1" (linear interpolation) or "2" (smart interpolation, default)
```

The most important part of the idle CPU usage on modern softsynths is due the onboard effects. If you turn off all onboard effects, you should see a very small CPU usage, mostly used for the mixer stage. That should be very small (1-2%, depending on the number of outputs of the synth).

However, almost every synth nowadays has onboard effects. There has been a never-ending discussion about this fact: many 'purists' think that synths should not include any effect, just plain sound. "After all", they say, "I could apply the effects I want to the synth output to get the same result". However, in many synths (like Dimension), this is not possible due most effects are 'insert effects', and are connected in points of the signal flow thar are not externally accessible. Additionally, getting the auto-pan when selecting the rhodes, or the drive when selecting the lead, or the delay when selecting the riff patch automagically is comfortable and handy. Anyways, I digress.

From a design point of view, effects do not know if they have a silent input or not, so they have to be processing all the time. Many effects need to generate output *even* with a silent input for some time (i.e. delays, reverbs), so that's an added reason.

That said, There're ways to 'detect' if the input is silent, and one might think in auto-off effects. If the input has been silent for x seconds, then I turn myself off. However, the mechanisms to detect silent inputs take some CPU themselves, and sometimes even more CPU than the effect itself. For a simple delay stage, for instance, it doesn't make any sense, as the detector takes approx. the same CPU as the delay stage itself. That's the reason behind most plugins not implementing any of those mechanisms.

At host level, there're some interesting opportunities though, which can be performed extending the plugin/host integration. 'Smart-freeze' is up there in our discussions, so it might see the light one day.

-René

DimensionalTips: Vinyl Noise

Ok, an easy tip and a more advanced one together.

Easy

Let's say you want to add some Vinyl Noise to a groove, or to the track background. Then just:

1- Insert a Dimension
2- Click on the main display in any element, to load a new multisample
3- Select "97 - Effects"
4- Select "Vinyl Noise (volume).sfz"

Presto. We now have a nice background noise. The volume can be controlled using Mod Wheel (CC1) without any entry in the MIDI Matrix, or with the element volume. The other 'Vinyl Noise.sfz' is the same thing, but element volume won't affect the noise, only mod wheel. This one is good if you want to copy/paste the effect so when you load some groove it brings the noise with it.

Advanced

The vinyl noise is generated by something called 'static generator'. A static generator object is created using the special <effect> header in the sfz format. Here's how it's defined for our vinyl noise:

```
// static generator
// tight static noise
//
// cc233= noise level

<effect>
type=static
bus=main

static_level=-30
static_level_oncc233=30
static_filter=hpf_1p
static_tone=20
static_cyclic_level=50
static_cyclic_time=1.82 // 33 rpm
static_random_level=2
static_random_mintime=0
static_random_maxtime=0.001
```

The static generator is composed by two generators: a Cyclic generator (it generates a cyclic 'pop' or 'click', like those caused by scratches in the vinyl surface) and a Random generator, which generates random clicks.

static_cyclic_level= and static_random_level= set the output level for each one. For the cyclic, you can specify the period of appearance with static_cyclic_time=. For the random, it is possible to specify the miinimum and maximum time before a click appears. There's also a filter in the output, in this case a high-pass filter.

Tweaking those parameters you can create a huge variety of noises and other effects. Actually, in the vinyl we used there are two static generators together.

Enjoy your noise!

-René

DimensionalTips: Attack with Velocity

As you all perfectly know, each envelope has a control which allows to modify the times of all segments in the envelope with the note-on velocity of the incoming MIDI message. It's called "Vel->Tim", and it's in the EG section. I'll take that the User Manual is in all your nightly reads.

Now I'll mention some even more advanced: Dimension Envelope Generators, offer the ability to adjust with the velocity of the incoming note-on message *any* segment in the envelope, and all segments can have a different adjustment.

How to do this? Ok. First, draw your envelope till you're happy with how it works in the 'standard' velocity, which is 100. Then click on the Envelope Generator window, and press V.

Then drag between the two nodes containing the segment you want to adjust. An orange bar will appear: when it goes down, it means that higher velocities will result in shorter times.

This is very used to create those strings which will attack slowly when you play piano, and will attack *schwazang* when you play forte. Here's a small patch to show it:

http://www.rgcstuff.com/External/VelocityToAttackTime.prog

Envelope time of *each segment* can also be adjusted according to the keyboard position. This means, you might make higher notes having a faster attack than lower notes, or vice-versa.

To edit the keyboard tracking on each segment, repeat above procedure but now press 'K'. The bars will be blue, and I guess you get how the effect will be.

-René

DimensionalTips: Sign Your Samples

Let's say that you have crafted a very fine piece of sfz. One which you'd like to share with friends.

Most likely you would compress the samples and place them in some kind of electronic distribution media, so your friends can get it. Something like an ftp server, website, etc.

But when your first friend grabs it, he complains about a click in the D#3 sample. Nasty, as it naturally doesn't behave that way in your box. Then you think, "corrupted sample?".

The sfz format allows to digitally sign every sample, and to verify it on loading. A digital signature is a string which represents the whole content of the sample in a small size.

It works as follows: when you write the sfz file, you specify the digital signature of your samples. Then, when Dimension loads the samples, it calculates the digital signature using the same algorithm (MD5), and compares it with the specified string. If they don't match, it warns you. Here's an example:

```
<region>
sample=HBj1slF#4H-S.wav
md5=32a9430da770954074a612951699dd72
```

If I explained this well, you should now be about to ask "and how da hell I get that md5= string?". Right. There're many utilities available to do it, but the easiest one is to use... Dimension. Just write this:

```
<region>
sample=HBj1slF#4H-S.wav
md5=a
```

and load the sfz file. Dimension will popup the expected md5 string, which you can copy/paste.
I use this method also to make sure I don't edit the wav files in an sfz file after they're final... which is a common accident when working with samples. Hope you find this useful as well.

-René

DimensionalTips: Piano High Notes Undamped vs Damped

Real grand pianos have a mechanism called 'damper', which is encharged of stopping the string vibration once the keys are released. When you hit a key, the damper 'releases' the string so the hammer can hit it, and the vibration will last as long as you have the key down, following the natural grand piano decay.

Regardless of how soon you release the key, the hammer will return to its default position (usually it's held in an intermediate position during sustain to facilitate repeated strikes response), and the damper mutes the string vibration.

Some notes in the piano are produced with only one string, some other use two or three strings. All notes have a damper, shaped according to how many strings it has to damp. There's an exception to this: notes above A7 don't have a damper, so the vibration on those will last longer, and will die of 'natural causes' after the decay expiration.

DPianoDampers.jpg
DPiano.mp3

A few hardware ROM-based pianos behave this way, and some sample libraries too. All Dimension's Grand Piano patches have this feature, enabled by default.

However, there're many early piano patches in synths which do not behave like that. In pop or other contemporary styles, a 'synthy' piano might fit the track even better than a superrealistic piano.

In other words, if a phrase moves from, say, D7 to B7 in a real piano, the damped/undamped transition will be very evident, and in a classical piano context the audience is very used to it. However, in a ballad or pop tune that might sound a bit strange, as many of those were recorded using rom pianos. The singer might look at you strangely, with a 'stop playing with that pedal' -like expression.

Ok. Here's a replacement multisample set for all Grand Piano multisamples in Dimension, which assign the Damped/Undamped for high notes to CC3. If you want the notes damped, then send a CC3=0 (or do nothing, as it's the default). If you want high notes undamped, send CC3=127. In this way, you can have both possibilities, and use them according to context.

GrandPiano_UpperDampersOnCC3.zip

Those need to be extracted in your \Multisamples\01 - Grand Piano folder, to replace existing ones.

Body/Damper simulator
All Grand Piano patches with 'sim' as part of the title have the Body/Damper simulator in action. This one will add string resonances to the sound, as part of a

model for sympathetic string resonance. The level for the simulator output can be adjusted using CC1 (mod wheel).

Here's the same snippet with and without the simulator (level a bit exaggerated to demonstrate it). Please pay attention to how the last note makes the neighbor strings vibrate when the simulator is on.

Just to remind a few other options for the Grand Piano patches:

DPiano_SimulatorOff.mp3
DPiano_SimulatorOn.mp3

Sympathetic Resonance
Several classical piano composers have used the sympathetic resonance trick. The trick consists in pressing a key, or a chord in a 'silent' way (veeery softly, so it won't make any sound), and then play other notes, related to the ones in the chord in octave or fifth intervals (others will also work, softer). This causes the strings in the 'silent' notes to start vibrating, as their dampers are off, due an accoustic phenomena called 'sympathy'.

The main part of the Body/Damper simulator in Dimension is activated when the player activates the sustain pedal. However, there's a special circuitry in the simulator dedicated to generate sympathetic resonances on notes being active even without the sustain pedal. Here's a snippet to demonstrate it:

DPiano_Sympathetic.mp3

<trick-for-extreme-tweakers-and-piano-fans-only> By default, the simulator is configured to emulate 36 strings. The number of strings can be changed using this line in the .sfz files with 'sim' in the name:

strings_number=36

You can try adding more, for uberresonant megapianos. Adjust level with CC1 to match. Get your own Klavins!

</trick>

Hammer noise on Release Trigger
When you release a key in a real piano, the hammer and damper mechanisms make an accoustic noise. That noise, depending on how you mic the piano, can be very evident in recordings. Also, the strings being damped generate a 'mute' tone, which muffles the sound a bit for a few milliseconds before it dies away.

All Grand Pianos with 'rt' in the title feature a sample of that release transition. The level of the release triggers can be adjusted with CC2 (breath). Those noises add a lot to realism in solo piano, but they can be unwanted in a pop piano tune. Here's a snippet of three notes without and with release trigger.

DPiano_HammerRelTrigger.mp3

All the effects can be turned on or off simply loading the specific programs, with 'sim' and 'rt'. If you don't plan to use the simulator or sympathetic effect, it's better not to load the 'sim' patches, as the simulator adds to the CPU usage. Similarly, the 'rt' patches add a bit to the memory load.

The level of all effects can be adjusted, and this allows to keep the same piano for different parts of the track. For instance, you might want to have the intro with a resonant, release triggered solo piano sound, and then just lower the simulator and release trigger levels for the rest of the track.

Enjoy your piano playing!

-René

DimensionalTips: Evolving Textures and Soundscapes

One of my favorite tricks to get evolving textures and soundscapes when looking for swirling, strange stuff consists of:

1. Play a couple of chords in any synth, in sequence
2. Export to wav.
3. Create a patch that loads the wav in two layers, with one of the layers backwards.
4. Assign a controller to crossfade between both layers.

Then play a note and move the controller to death.

And here's the sfz magic, in case anyone wants to try it (for mod wheel):

```
<group> sample=wave_name_here.wav xf_cccurve=gain
<region> xfin_locc1=0 xfin_hicc1=127 direction=reverse
<region> xfout_locc1=0 xfout_hicc1=127
```

-René

DimensionalTips: Dimension 16-Pole Filter

Here is a trick to add a master 16-pole filter to any sfz file. Just add the following lines:

```
// filter r
<effect>
directtomain=100
type=filter
filter_type=lpf_4p
filter_cutoff_oncc2=12700
filter_resonance_oncc3=10

// filter o
<effect>
directtomain=100
type=filter
filter_type=lpf_4p
filter_cutoff_oncc2=12700
filter_resonance_oncc3=10

// filter c
<effect>
directtomain=100
type=filter
filter_type=lpf_4p
filter_cutoff_oncc2=12700
filter_resonance_oncc3=10

// filter k
<effect>
directtomain=100
type=filter
filter_type=lpf_4p
filter_cutoff_oncc2=12700
filter_resonance_oncc3=10
```

Those will give you a 16-pole filter, with cutoff controlled by CC2 and resonance by CC3. Replace those numbers with your favorite controls.

Here's a full sfz file, which needs to go into your 00 - Wavetables folder:

```
<group>
sample=saw mini.wav oscillator=on oscillator_phase=-1

<region> pitch_oncc140=-30 pan=100
<region> pitch_oncc140=-20 pan=50
<region> pitch_oncc140=-10
<region> pitch_oncc140=10
<region> pitch_oncc140=20 pan=50
<region> pitch_oncc140=30 pan=100

// filter 1
<effect>
directtomain=100
type=filter
filter_type=lpf_4p
```

```
filter_cutoff_oncc2=12700
filter_resonance_oncc3=10

// filter 2
<effect>
directtomain=100
type=filter
filter_type=lpf_4p
filter_cutoff_oncc2=12700
filter_resonance_oncc3=10

// filter 3
<effect>
directtomain=100
type=filter
filter_type=lpf_4p
filter_cutoff_oncc2=12700
filter_resonance_oncc3=10

// filter 4
<effect>
directtomain=100
type=filter
filter_type=lpf_4p
filter_cutoff_oncc2=12700
filter_resonance_oncc3=10
```

That's a detuned saw, and as you already know you need to tweak the detune factor with the bitred knob. Then you got the 16-pole filter on it, composed by 4 cascaded 4-pole filters. Now please, *make sure you have the limiter turned on*. Resonance on this filter can be extreme.

It's very interesting to try different cutoff ranges (in cents), as alternative to the 12700 (127 notes * 100 cents per note). Then you control cutoff, and the peaks move in strange ways. By adding the static 'filter_cutoff=12700' then a 'filter_cutoff_oncc2=-12700' you can reverse the control. Make sure you experiment with filters going in opposite ways with the same control.

-René

DimensionalTips: Dimension MIDI Matrix Parameters

The 'Reserved' word was used for parameters which a) can be automated thru recording, but can't or shouldn't be automated by the use of envelopes, MIDI remote or host widgets, either because they can generate weird behaviors or just because they're meaningless, or b) are program-loading-time only parameters.

For instance, one of those parameters is the Layer Counter. Yes, it is possible to automate the laýer counter, but that is... well.. useless :-)

I take that you have already discovered what the reserved values are for the MIDI Matrix parameters. I see you don't firmly believe in that 'ignorance is bliss' thing, so here it is a slightly more comprehensive list. There are a few of those which are still 'Reserved', but I can't talk about those, I apologize

-René

```
Enabled
Quality
Reserved
Reserved
Reserved Multisample name, element 1.
Reserved Multisample name, element 2.
Reserved Multisample name, element 3.
Reserved Multisample name, element 4.
LoKey 01
LoKey 02
LoKey 03
LoKey 04
HiKey 01
HiKey 02
HiKey 03
HiKey 04
LoVel 01
LoVel 02
LoVel 03
LoVel 04
HiVel 01
HiVel 02
HiVel 03
HiVel 04
Transpose 01
Transpose 02
Transpose 03
Transpose 04
Tune 01
Tune 02
Tune 03
Tune 04
Pitch KeyTrack 01
Pitch KeyTrack 02
Pitch KeyTrack 03
Pitch KeyTrack 04
Shift 01
Shift 02
Shift 03
Shift 04
Bend Up 01
Bend Up 02
Bend Up 03
Bend Up 04
Bend Down 01
Bend Down 02
Bend Down 03
Bend Down 04
Polyphony 01
Polyphony 02
Polyphony 03
Polyphony 04
Reserved Layer counter 01
Reserved Layer counter 02
Reserved Layer counter 03
Reserved Layer counter 04
Bitred On/Off 01
Bitred On/Off 02
Bitred On/Off 03
Bitred On/Off 04
Bitred Depth 01
Bitred Depth 02
Bitred Depth 03
Bitred Depth 04
Decim On/Off 01
Decim On/Off 02
Decim On/Off 03
Decim On/Off 04
Decim Depth 01
Decim Depth 02
Decim Depth 03
Decim Depth 04
Filter Type 01
Filter Type 02
Filter Type 03
Filter Type 04
Cutoff 01
Cutoff 02
Cutoff 03
Cutoff 04
Resonance 01
Resonance 02
Resonance 03
Resonance 04
Drive Mode 01
Drive Mode 02
Drive Mode 03
Drive Mode 04
Drive Shape 01
Drive Shape 02
Drive Shape 03
Drive Shape 04
Drive Tone 01
Drive Tone 02
Drive Tone 03
Drive Tone 04
Eq1 On/Off 01
Eq1 On/Off 02
```

```
Eq1 On/Off 03
Eq1 On/Off 04
Eq1 Type 01
Eq1 Type 02
Eq1 Type 03
Eq1 Type 04
Eq1 Gain 01
Eq1 Gain 02
Eq1 Gain 03
Eq1 Gain 04
Eq1 Freq 01
Eq1 Freq 02
Eq1 Freq 03
Eq1 Freq 04
Eq1 Q 01
Eq1 Q 02
Eq1 Q 03
Eq1 Q 04
Eq2 On/Off 01
Eq2 On/Off 02
Eq2 On/Off 03
Eq2 On/Off 04
Eq2 Type 01
Eq2 Type 02
Eq2 Type 03
Eq2 Type 04
Eq2 Gain 01
Eq2 Gain 02
Eq2 Gain 03
Eq2 Gain 04
Eq2 Freq 01
Eq2 Freq 02
Eq2 Freq 03
Eq2 Freq 04
Eq2 Q 01
Eq2 Q 02
Eq2 Q 03
Eq2 Q 04
Eq3 On/Off 01
Eq3 On/Off 02
Eq3 On/Off 03
Eq3 On/Off 04
Eq3 Type 01
Eq3 Type 02
Eq3 Type 03
Eq3 Type 04
Eq3 Gain 01
Eq3 Gain 02
Eq3 Gain 03
Eq3 Gain 04
Eq3 Freq 01
Eq3 Freq 02
Eq3 Freq 03
Eq3 Freq 04
Eq3 Q 01
Eq3 Q 02
Eq3 Q 03
Eq3 Q 04
Fx Mode 01
Fx Mode 02
Fx Mode 03
Fx Mode 04
Fx Filter Type 01
Fx Filter Type 02
Fx Filter Type 03
Fx Filter Type 04
Fx Time L 01
Fx Time L 02
Fx Time L 03
Fx Time L 04
Fx Time C 01
Fx Time C 02
Fx Time C 03
Fx Time C 04
Fx Time R 01
Fx Time R 02
Fx Time R 03
Fx Time R 04
Fx Feedback 01
Fx Feedback 02
Fx Feedback 03
Fx Feedback 04
Fx Cutoff 01
Fx Cutoff 02
Fx Cutoff 03
Fx Cutoff 04
Fx Lfo Freq 01
Fx Lfo Freq 02
Fx Lfo Freq 03
Fx Lfo Freq 04
Fx Lfo Depth 01
Fx Lfo Depth 02
Fx Lfo Depth 03
Fx Lfo Depth 04
Fx Resonance 01
Fx Resonance 02
Fx Resonance 03
Fx Resonance 04
Fx Input 01
Fx Input 02
Fx Input 03
Fx Input 04
Fx Dry/Wet 01
Fx Dry/Wet 02
Fx Dry/Wet 03
Fx Dry/Wet 04
Reserved EG1 Dirty 01
Reserved EG1 Dirty 02
Reserved EG1 Dirty 03
Reserved EG1 Dirty 04
Reserved EG2 Dirty 01
Reserved EG2 Dirty 02
Reserved EG2 Dirty 03
Reserved EG2 Dirty 04
Reserved EG3 Dirty 01
Reserved EG3 Dirty 02
Reserved EG3 Dirty 03
Reserved EG3 Dirty 04
Reserved EG4 Dirty 01
Reserved EG4 Dirty 02
Reserved EG4 Dirty 03
Reserved EG4 Dirty 04
Reserved EG5 Dirty 01
Reserved EG5 Dirty 02
Reserved EG5 Dirty 03
Reserved EG5 Dirty 04
Pitch Eg On/Off 01
Pitch Eg On/Off 02
Pitch Eg On/Off 03
Pitch Eg On/Off 04
Cutoff Eg On/Off 01
Cutoff Eg On/Off 02
Cutoff Eg On/Off 03
Cutoff Eg On/Off 04
Reso Eg On/Off 01
Reso Eg On/Off 02
Reso Eg On/Off 03
Reso Eg On/Off 04
Pan Eg On/Off 01
Pan Eg On/Off 02
Pan Eg On/Off 03
Pan Eg On/Off 04
Amp Eg On/Off 01
Amp Eg On/Off 02
Amp Eg On/Off 03
Amp Eg On/Off 04
Pitch Eg Depth 01
Pitch Eg Depth 02
Pitch Eg Depth 03
Pitch Eg Depth 04
Cutoff Eg Depth 01
Cutoff Eg Depth 02
Cutoff Eg Depth 03
Cutoff Eg Depth 04
```

```
Reso Eg Depth 01
Reso Eg Depth 02
Reso Eg Depth 03
Reso Eg Depth 04
Pan Eg Depth 01
Pan Eg Depth 02
Pan Eg Depth 03
Pan Eg Depth 04
Amp Eg Depth 01
Amp Eg Depth 02
Amp Eg Depth 03
Amp Eg Depth 04
Pitch Eg Velo 01
Pitch Eg Velo 02
Pitch Eg Velo 03
Pitch Eg Velo 04
Cutoff Eg Velo 01
Cutoff Eg Velo 02
Cutoff Eg Velo 03
Cutoff Eg Velo 04
Reso Eg Velo 01
Reso Eg Velo 02
Reso Eg Velo 03
Reso Eg Velo 04
Pan Eg Velo 01
Pan Eg Velo 02
Pan Eg Velo 03
Pan Eg Velo 04
Amp Eg Velo 01
Amp Eg Velo 02
Amp Eg Velo 03
Amp Eg Velo 04
Pitch Eg VeloTim 01
Pitch Eg VeloTim 02
Pitch Eg VeloTim 03
Pitch Eg VeloTim 04
Cutoff Eg VeloTim 01
Cutoff Eg VeloTim 02
Cutoff Eg VeloTim 03
Cutoff Eg VeloTim 04
Reso Eg VeloTim 01
Reso Eg VeloTim 02
Reso Eg VeloTim 03
Reso Eg VeloTim 04
Pan Eg VeloTim 01
Pan Eg VeloTim 02
Pan Eg VeloTim 03
Pan Eg VeloTim 04
Amp Eg VeloTim 01
Amp Eg VeloTim 02
Amp Eg VeloTim 03
Amp Eg VeloTim 04
Pitch VelTrack 01
Pitch VelTrack 02
Pitch VelTrack 03
Pitch VelTrack 04
Cutoff VelTrack 01
Cutoff VelTrack 02
Cutoff VelTrack 03
Cutoff VelTrack 04
Reso VelTrack 01
Reso VelTrack 02
Reso VelTrack 03
Reso VelTrack 04
Pan VelTrack 01
Pan VelTrack 02
Pan VelTrack 03
Pan VelTrack 04
Vol VelTrack 01
Vol VelTrack 02
Vol VelTrack 03
Vol VelTrack 04
Pitch Lfo On/Off 01
Pitch Lfo On/Off 02
Pitch Lfo On/Off 03
Pitch Lfo On/Off 04
Cutoff Lfo On/Off 01
Cutoff Lfo On/Off 02
Cutoff Lfo On/Off 03
Cutoff Lfo On/Off 04
Reso Lfo On/Off 01
Reso Lfo On/Off 02
Reso Lfo On/Off 03
Reso Lfo On/Off 04
Pan Lfo On/Off 01
Pan Lfo On/Off 02
Pan Lfo On/Off 03
Pan Lfo On/Off 04
Vol Lfo On/Off 01
Vol Lfo On/Off 02
Vol Lfo On/Off 03
Vol Lfo On/Off 04
Pitch Lfo Wave 01
Pitch Lfo Wave 02
Pitch Lfo Wave 03
Pitch Lfo Wave 04
Cutoff Lfo Wave 01
Cutoff Lfo Wave 02
Cutoff Lfo Wave 03
Cutoff Lfo Wave 04
Reso Lfo Wave 01
Reso Lfo Wave 02
Reso Lfo Wave 03
Reso Lfo Wave 04
Pan Lfo Wave 01
Pan Lfo Wave 02
Pan Lfo Wave 03
Pan Lfo Wave 04
Vol Lfo Wave 01
Vol Lfo Wave 02
Vol Lfo Wave 03
Vol Lfo Wave 04
Pitch Lfo Phase 01
Pitch Lfo Phase 02
Pitch Lfo Phase 03
Pitch Lfo Phase 04
Cutoff Lfo Phase 01
Cutoff Lfo Phase 02
Cutoff Lfo Phase 03
Cutoff Lfo Phase 04
Reso Lfo Phase 01
Reso Lfo Phase 02
Reso Lfo Phase 03
Reso Lfo Phase 04
Pan Lfo Phase 01
Pan Lfo Phase 02
Pan Lfo Phase 03
Pan Lfo Phase 04
Vol Lfo Phase 01
Vol Lfo Phase 02
Vol Lfo Phase 03
Vol Lfo Phase 04
Pitch Lfo Offset 01
Pitch Lfo Offset 02
Pitch Lfo Offset 03
Pitch Lfo Offset 04
Cutoff Lfo Offset 01
Cutoff Lfo Offset 02
Cutoff Lfo Offset 03
Cutoff Lfo Offset 04
Reso Lfo Offset 01
Reso Lfo Offset 02
Reso Lfo Offset 03
Reso Lfo Offset 04
Pan Lfo Offset 01
Pan Lfo Offset 02
Pan Lfo Offset 03
Pan Lfo Offset 04
Vol Lfo Offset 01
Vol Lfo Offset 02
Vol Lfo Offset 03
Vol Lfo Offset 04
Pitch Lfo Freq 01
Pitch Lfo Freq 02
Pitch Lfo Freq 03
Pitch Lfo Freq 04
Cutoff Lfo Freq 01
Cutoff Lfo Freq 02
Cutoff Lfo Freq 03
Cutoff Lfo Freq 04
Reso Lfo Freq 01
Reso Lfo Freq 02
Reso Lfo Freq 03
Reso Lfo Freq 04
Pan Lfo Freq 01
Pan Lfo Freq 02
Pan Lfo Freq 03
Pan Lfo Freq 04
Vol Lfo Freq 01
Vol Lfo Freq 02
Vol Lfo Freq 03
Vol Lfo Freq 04
Pitch Lfo Sync 01
Pitch Lfo Sync 02
Pitch Lfo Sync 03
Pitch Lfo Sync 04
Cutoff Lfo Sync 01
Cutoff Lfo Sync 02
Cutoff Lfo Sync 03
Cutoff Lfo Sync 04
Reso Lfo Sync 01
Reso Lfo Sync 02
Reso Lfo Sync 03
Reso Lfo Sync 04
Pan Lfo Sync 01
Pan Lfo Sync 02
Pan Lfo Sync 03
Pan Lfo Sync 04
Vol Lfo Sync 01
Vol Lfo Sync 02
Vol Lfo Sync 03
Vol Lfo Sync 04
Pitch Lfo Delay 01
Pitch Lfo Delay 02
Pitch Lfo Delay 03
Pitch Lfo Delay 04
Cutoff Lfo Delay 01
Cutoff Lfo Delay 02
Cutoff Lfo Delay 03
Cutoff Lfo Delay 04
Reso Lfo Delay 01
Reso Lfo Delay 02
Reso Lfo Delay 03
Reso Lfo Delay 04
Pan Lfo Delay 01
Pan Lfo Delay 02
Pan Lfo Delay 03
Pan Lfo Delay 04
Vol Lfo Delay 01
Vol Lfo Delay 02
Vol Lfo Delay 03
Vol Lfo Delay 04
Pitch Lfo Fade 01
Pitch Lfo Fade 02
Pitch Lfo Fade 03
Pitch Lfo Fade 04
Cutoff Lfo Fade 01
Cutoff Lfo Fade 02
Cutoff Lfo Fade 03
Cutoff Lfo Fade 04
Reso Lfo Fade 01
Reso Lfo Fade 02
Reso Lfo Fade 03
Reso Lfo Fade 04
Pan Lfo Fade 01
Pan Lfo Fade 02
Pan Lfo Fade 03
Pan Lfo Fade 04
Vol Lfo Fade 01
Vol Lfo Fade 02
Vol Lfo Fade 03
Vol Lfo Fade 04
Pitch Lfo Depth 01
Pitch Lfo Depth 02
Pitch Lfo Depth 03
Pitch Lfo Depth 04
Cutoff Lfo Depth 01
Cutoff Lfo Depth 02
Cutoff Lfo Depth 03
Cutoff Lfo Depth 04
Reso Lfo Depth 01
Reso Lfo Depth 02
Reso Lfo Depth 03
Reso Lfo Depth 04
Pan Lfo Depth 01
Pan Lfo Depth 02
Pan Lfo Depth 03
Pan Lfo Depth 04
Vol Lfo Depth 01
Vol Lfo Depth 02
Vol Lfo Depth 03
Vol Lfo Depth 04
Reserved P Keytrack 01
Reserved P Keytrack 02
Reserved P Keytrack 03
Reserved P Keytrack 04
Reserved C Keytrack 01
Reserved C Keytrack 02
Reserved C Keytrack 03
Reserved C Keytrack 04
Reserved R Keytrack 01
Reserved R Keytrack 02
Reserved R Keytrack 03
Reserved R Keytrack 04
Reserved P Keytrack 01
Reserved P Keytrack 02
Reserved P Keytrack 03
Reserved P Keytrack 04
Reserved A Keytrack 02
Reserved A Keytrack 03
Reserved A Keytrack 04
Reserved A Keytrack 05
Element On/Off 01
Element On/Off 02
Element On/Off 03
Element On/Off 04
Element Fx1 01
Element Fx1 02
Element Fx1 03
Element Fx1 04
Element Fx2 01
Element Fx2 02
Element Fx2 03
Element Fx2 04
Element Pan 01
Element Pan 02
Element Pan 03
Element Pan 04
Element Volume 01
Element Volume 02
Element Volume 03
Element Volume 04
Mod Fx Type
Mod Fx Freq
Mod Fx Delay
Mod Fx Depth
Mod Fx Feedb
Mod Fx DryWet
Reverb Fx Type
Reverb Fx Predelay
Reverb Fx Size
Reverb Fx Damp
Reverb Fx Tone
Reverb Fx Dry/Wet
Vector Mixer X
Vector Mixer Y
Vector Mixer Desaccel
Reserved
```

Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved
Reserved	Reserved

```
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved Reset LFOs phase.
Reserved Multitimbral mode.
Element Chain 01
Element Chain 02
Element Chain 03
Element Chain 04
Reserved Loading Mode element 1
Reserved Loading Mode element 2
Reserved Loading Mode element 3
Reserved Loading Mode element 4
Sustain On/Off 01
Sustain On/Off 02
Sustain On/Off 03
Sustain On/Off 04
Sost On/Off 01
Sost On/Off 02
Sost On/Off 03
Sost On/Off 04
Reserved MIDI Matrix Depth 01
Reserved MIDI Matrix Depth 02
Reserved MIDI Matrix Depth 03
Reserved MIDI Matrix Depth 04
Reserved MIDI Matrix Depth 05
Reserved MIDI Matrix Depth 06
Reserved MIDI Matrix Depth 07
Reserved MIDI Matrix Depth 08
Reserved MIDI Matrix Depth 09
Reserved MIDI Matrix Depth 10
Reserved MIDI Matrix Depth 11
Reserved MIDI Matrix Depth 12
Reserved MIDI Matrix Depth 13
Reserved MIDI Matrix Depth 14
Reserved MIDI Matrix Depth 15
Reserved MIDI Matrix Depth 16
Reserved MIDI Matrix Destination 01
Reserved MIDI Matrix Destination 02
Reserved MIDI Matrix Destination 03
Reserved MIDI Matrix Destination 04
Reserved MIDI Matrix Destination 05
Reserved MIDI Matrix Destination 06
Reserved MIDI Matrix Destination 07
Reserved MIDI Matrix Destination 08
Reserved MIDI Matrix Destination 09
Reserved MIDI Matrix Destination 10
Reserved MIDI Matrix Destination 11
Reserved MIDI Matrix Destination 12
Reserved MIDI Matrix Destination 13
Reserved MIDI Matrix Destination 14
Reserved MIDI Matrix Destination 15
Reserved MIDI Matrix Destination 16
Reserved MIDI Matrix Depth 01
Reserved MIDI Matrix Depth 02
Reserved MIDI Matrix Depth 03
Reserved MIDI Matrix Depth 04
Reserved MIDI Matrix Depth 05
Reserved MIDI Matrix Depth 06
Reserved MIDI Matrix Depth 07
Reserved MIDI Matrix Depth 08
Reserved MIDI Matrix Depth 09
Reserved MIDI Matrix Depth 10
Reserved MIDI Matrix Depth 11
Reserved MIDI Matrix Depth 12
Reserved MIDI Matrix Depth 13
Reserved MIDI Matrix Depth 14
Reserved MIDI Matrix Depth 15
Reserved MIDI Matrix Depth 16
Reserved MIDI Matrix Smooth 01
Reserved MIDI Matrix Smooth 02
Reserved MIDI Matrix Smooth 03
Reserved MIDI Matrix Smooth 04
Reserved MIDI Matrix Smooth 05
Reserved MIDI Matrix Smooth 06
Reserved MIDI Matrix Smooth 07
Reserved MIDI Matrix Smooth 08
Reserved MIDI Matrix Smooth 09
Reserved MIDI Matrix Smooth 10
Reserved MIDI Matrix Smooth 11
Reserved MIDI Matrix Smooth 12
Reserved MIDI Matrix Smooth 13
Reserved MIDI Matrix Smooth 14
Reserved MIDI Matrix Smooth 15
Reserved MIDI Matrix Smooth 16
Limiter On/Off
```