Review: Cakewalk SONAR 2 XL
by Rick Paul - *4th June 2002* -

Overview

Almost exactly one year after Cakewalk introduced its flagship SONAR product, SONAR went into its second generation with the April 2002 deliveries of SONAR 2.0 and SONAR 2.0 XL.

SONAR represented a significant enhancement over Cakewalk's earlier flagship Pro Audio 9, bringing such enhancements as non-destructive editing, softsynth integration, and support for ACID-style looping. However, the 1.0 release left a lot to be desired when it came to actually using the product for real work, owing to a number of performance and user interface design issues. Cakewalk issued some early patches, but it wasn't until the (free to registered users) 1.31 patch, which also featured some significant enhancements, especially on the user interface side, that SONAR truly came into its own.

Of course, popular wisdom with software is to avoid anything ending in ".0", especially if it starts with "1". SONAR users no longer need to worry about the 1.0 syndrome, but the real question for SONAR 1.31 users considering an upgrade to 2.0, and for non-SONAR users just considering jumping into SONAR, is whether it is safe to dive in with SONAR 2.0. And, of course, existing SONAR users, or users who have thus far avoided SONAR due to missing functionality for their needs, will want to know what else SONAR 2 delivers.

We'll take a more detailed look at both questions below, but let's cut to the chase right now: Yes, SONAR 2.0 is an unusual ".0" release in that it actually improves upon the stability of SONAR 1.31, which was itself highly stable for most users. With that out of the way, let's move on to looking at the new features delivered in the 2.0 release.

What's New?

SONAR 1.31 was already fairly fully-featured digital audio workstation (DAW) software. While the 2.0 designation makes it a major release, and there certainly are a number of substantial new features provided, SONAR 2 largely takes the approach of enhancing and expanding upon what is there, rather than introducing earth-shaking surprises.

In some cases, the enhancements just make using SONAR a smoother experience. In other cases, an enhancement may appear to be a small addition on the surface, but it may bring with it the potential for dramatically improving your working processes. In still other cases, the enhancement may only be of use to users with fairly specific other software or hardware.

The Cakewalk web site lists the following enhancements over SONAR 1:

- CYCLONE DXi 16-part groove sampler.
- Dedicated support for the CM Labs MotorMix and Tascam US-428, plus a Global Control panel with learn-mode to quickly integrate any MIDI-compatible control surface.
- Multi-port drum editing with grid & pattern based enhancements and custom drum mapping.
- Enhanced soft synth integration: Synth Rack, multi-output synths, DXi 2.0 support, more.
- ReWire 2.0 support: integrate Reason, ReBirth and other ReWire-compatible synths with SONAR.
- Advanced project management: easy-to-use file management tools, per-project audio folders, intelligent file naming, more.
- Yamaha OPT Panels Support.
- Export ACID-format WAV files for use in other projects and applications.

We'll get to the details of those below, but SONAR 2 brings more to the table than just that list of features.

SONAR 2 has added a number of very subtle, sometimes almost hidden, enhancements. Some of these, such as the Audio Engine's running more smoothly, come for free. That is, they just work better by default. Others, such as the ability to modify the layout of audio and MIDI track properties and do other tweaking that affects Audio Engine and/or Input Monitoring performance, require the user to do a little work under the hood to gain maximum benefit, though the default behavior is at least reasonable for most users.

Besides the basic SONAR 2 package, Cakewalk offers an additional value package called SONAR 2 XL. These XL package are purely marketing bundles, and the value of the premium package will generally depend on the user's style of music and what he or she already has. However, Cakewalk has done a particularly good job of making the XL "extras" appealing to a wide base of users this time, including high quality compressor and equalizer plug-ins from Sonic TimeWorks and a very flexible drum machine softsynth from FXpansion.

Let's dive in and take a look, starting with the features Cakewalk highlights.

Get in the Groove

SONAR 1 introduced ACID-style loop arrangement and playback, complete with time and pitch shifting. But what about the user who wanted even more control than making a loop conform to a new tempo or chord change? What if the user wanted to change one of the sounds in a drum groove, shift the groove around, or even play elements of the groove in real-time from a MIDI controller? Enter Cyclone DXi, which is billed as a "16-part groove sampler".

Now, I have to confess I don't make a lot of loop-based music, and it took me awhile to figure out how to use ACID, and even longer to get around to even trying to use loops in SONAR. In fact, my first loop-based production in SONAR was done about a month ago in SONAR 2. (Those who are thinking trance, house, techno, or something similar will probably be amused, or perhaps horrified, to learn that the song was actually fairly traditional country, and the loops I used were for pedal steel and electric guitar, off Cakewalk's and Messer Music's "Loops of Hazzard" collection.)

Thus, when I first heard SONAR 2 would include a "groove sampler" I really didn't know what to think. "Why would I want to sample a groove? And what would I do with it once I sampled it?" You can stop laughing now -- hey, I play my own grooves!

I must further confess that, even after playing with Cyclone for a couple of hours, including running through a tutorial on the Cakewalk web site, where the results I was seeing on my screen were very different from the ones they were showing, I still hadn't fully come to grips with Cyclone. However, after a bit more experimentation, I think I've at least got a somewhat better sense of what it can do, and maybe even a few thoughts on how even I may eventually be able to use it.

There is a lot happening in the Cyclone screen (see screen shot above). In the upper left corner is the loop bin, where WAV files, ACIDized or not, can be dragged for use in grooves. Just to the right of that is the pad group area, where "pad" level parameters, such as mute, solo, whether the pad is synchronized to SONAR tempo and pitch changes, whether the pad should be looped, and a few other parameters can be set.

What is a pad, you ask? We'll get to that shortly, but, in the interim, to the right of the pad group area is the pad inspector area, which allows tweaking additional pad-specific parameters, for a single pad at a time (the pad group area allows tweaking all 16 pads in parallel). In the pad inspector area, you can tweak such parameters as the MIDI velocity and note ranges responded to by a pad, the root pitch of the pad for use with SONAR's ACID-style pitch shifting, etc. For pitched pads, playing the "unity key" for a pad on a MIDI keyboard (or via a MIDI sequencer) will result in the pad's playing back in its base key. Playing another note within the note range specified for that pad will result in the pad's being transposed by the appropriate amount from the unity key or base key, depending on the offset of the note played from the unity key note.

"But what is a pad, and why would I want to transpose it?" We're getting there.

Just below the section of the Cyclone screen we just talked about is an area that displays either the current WAV file, ACIDized or otherwise, or a picture of a MIDI keyboard. In the case of an ACIDized loop, you see not only the audio wave form, but also the segment

boundaries set up in the ACIDizing (is that a word?) process. We'll talk about why that is useful shortly. If the display is a MIDI keyboard, we seek the unity key and note range for the currently selected pad. The user can toggle between displays quickly by clicking on a tab control.

At the bottom right of the screen is the pad editor, where you can position the "slices" (huh?) within each "track", where each track corresponds to one of the 16 pads. And, finally, to the left of that (i.e. the bottom left of the screen) is the slice inspector, which allows viewing and setting properties (pitch offset, gain, and pan) for the currently selected slice.

Okay, if you're not sufficiently confused by now, you're much "groovier" (sorry, I couldn't resist) than I am. Let's talk turkey -- er, pads and slices.

Remember I mentioned that an ACIDized loop will show its individual segment boundaries in the loop display. Well, the portion of the wave between two segment boundaries is a slice, and you can construct and deconstruct grooves by positioning slices within the track area of a pad. As for what a pad is, as best I can explain it, it consists of a track, which may have looping properties or not, and a set of properties, or behavioral traits, associated with that track, such as some of the notions I've alluded to above with respect to MIDI note response and more. Let's try and look at a few practical examples, though, and maybe things will become a bit clearer.

Let's say we have an ACIDized drum groove, and it sounds just great for most of the areas within the song where we want to use it. However, there is one point where we want to just take it down to the kick drum and snare, and we want to tweak the timing of those elements slightly in those sections of the song. A quick way to start in Cyclone is to read the ACID loop for the groove into a pad. Since it is ACIDized, Cyclone will detect this and automatically assign the looped and synchronized properties to the pad. It will also read all the slices of the loop into individual, ordered slices within the pad's track area.

At this point, there is lots of tweaking you can do, but probably the first thing we'll want to do is click the mouse on various individual slices in the pad so we can see how they correspond to the ACIDized loop display, then figure out which ones we want to delete to leave only the kick and snare hits (and anything else that happens to be hitting in the same slice of the ACIDized loop -- Cyclone is powerful, but it has no magic to deal with separating out concurrent hits of different instruments).

After deleting the unneeded slices, playing back the loop sounds kind of odd -- pretty much like we just took some slices out of an audio groove, in fact. Perhaps that may work for some cases, but it is decidedly not what I had in mind here. I wanted a more natural sounding, albeit thinner, groove. No problem, though, enabling the "tails" property for the pad in the pad inspector magically extends the tail, or decay, of the slice so things sound quite natural again, and I can slide the slices around to my heart's content for tweaking the feel of the groove.

Here's another, related, application: How many times have you been frustrated by a great groove sample, but not having the tools to build the fill you want while still sounding like the same drum kit. With Cyclone, if you can isolate the individual drum and cymbal types within slices of a loop, you can put them on their own individual pads, not looping them, so you can play the one shot samples from your keyboard.

Another possibility that isn't related might be if you have a pitched groove lick, such as a bass guitar part, that you'd like to use repeatedly, but where the lick basically needs to change at different parts of the song. Of course, you could use the ACID-style pitch changing along with SONAR's pitch maps to deal with that, but Cyclone provides a somewhat more musician-friendly way. Simply read the ACIDized groove lick into the pad, set up the relevant pitch information, then give the pad a note range that at least covers the key changes you'll want to use. Now just hit the relevant key on the keyboard when you want to trigger it in a particular key. And, of course, by using different pads with different licks and different key ranges, you can "play" a more complex part in real time by interchanging grooves via the note ranges you are using. Record that performance to a MIDI track in SONAR, and you can play it back at will and even edit it later. Somehow that sounds like it might be a bit more enjoyable than just painting groove loops.

All in all, Cyclone DXi looks like it could be a very interesting tool, even for less groove-oriented types like me. One of my big objections to groove loops has been the difficulty in making them conform to what I want them to do and the lack of playability. Cyclone seems to address both areas to a degree, with the biggest limitation likely being its only having 16 pads. Then again, you can always use multiple instances of Cyclone to get more.

For the Control Freak

When I think of control surfaces, my mind immediately pictures those small, or sometimes large, boxes that resemble a traditional mixer -- for example, Cakewalk's StudioMix, the Peavey 1600, the CM Labs Motormix, Tascam's US-428, and so on. I think mainly about mixing in a relatively traditional style, moving real faders in parallel while using the ears as the primary judge of what is working and what isn't.

SONAR 2's enhanced control surface support features allow using a number of these control surfaces right out of the box. Just pick the appropriate presets and go to town.

While that's great for people who own one of these boxes, and a number of DAW vendors have implemented support for one or more of these boxes, what about the user (e.g. me) who doesn't have this sort of traditional control surface, or maybe doesn't have one of the specific ones for which support has been preconfigured? This is where SONAR 2's new generic control surface support comes in.

The generic control surface support allows specifying up to 16 track control parameters (e.g. volume, pan, mute, solo, record arm, aux send levels, etc.) for one or more consecutive tracks. In addition, it provides the ability to configure global control parameters (e.g. play, stop, record, jog forward/backward, etc.). For each global or track-specific parameter, you simply select the parameter, twiddle the slider (or button, knob, etc.) you want to use to automate it, then hit a "Learn" button, and SONAR takes care of the rest.

In my case, I didn't have any traditional control surfaces, but my trusty Roland Rhodes MK-80 keyboard has 4 sliders on it that I thought might be useful as track faders. Sure enough, once I figured out how to make the sliders send their MIDI data (thankfully, we're not reviewing old Roland manuals here), it was extremely simple to tell SONAR I wanted to control 4 consecutive tracks of volume, then to select the first parameter on each track (which defaulted to a volume control anyway), twiddle the relevant slider, press the Learn button, then move on to the next track's parameter. Once all were finished, I saved a generic control surface preset which I called "MK-80 4-track volume", and specified the first track I wanted to control, in this case track 5, thus making my preset control tracks 5-8,

and wiggle the sliders to my hearts content, with the relevant track volume faders on SONAR's screen responding in kind.

Easy enough. What wasn't so obvious, though, was why I couldn't get any softsynths to play after having done that. It turns out that a MIDI device that is in use as a control surface can't also be used concurrently as a keyboard MIDI controller. That makes sense, of course, and wouldn't affect the user with a dedicated control surface anyway. However, for the truly limited user with just a keyboard with some sliders (i.e. me again), having to continually enable and disable the control surface to switch between tracking MIDI parts and controlling faders and such limits the practical benefit of the feature. But it's an added bonus that it works at all for such users, and the lucky ones with real control surfaces don't have to depend on Cakewalk or any other vendor to provide nice integration between their choice of control surface and SONAR.

I've Got Rhythm

SONAR has always had the ability to display drum note names in its piano roll view, and its Session Drummer MIDI plug-in provided some MIDI loop-based drum part construction capabilities. Still, the rhythm programmer who needed much better control and efficiency was left wanting more.

SONAR 2 adds three key facilities to help the rhythm programmer. First, the Drum Map Manager allows mapping drum names to not only MIDI note names, but also MIDI ports and channels and more. Next comes a dedicated Drum Grid Editor pane within the piano roll view. That may seem relatively insignificant given the older capabilities within the piano roll view, but, as we'll see in a moment, there is more to it than there might appear to be on initial inspection. Finally, the new Pattern Brush facility allows painting rhythmic patterns, and not necessarily just drum patterns.

In getting ready to program a rhythm part, the first thing a user will likely want to do is set up a drum map, either by creating a new one or calling up an existing one. Existing drum maps can come from presets supplied by Cakewalk for a number of popular hardware and software drum machines or from previous drum maps the users have created for their own equipment or software-based drum machines.

It is probably obvious from the name that the new Drum Map Manager allows mapping drum note names (e.g. "kick", "snare", "closed hat", etc.) to MIDI notes. But what if you want to use cymbals from a software-based sampler, drum kicks from an analog-style drum machine, latin percussion parts from another drum machine, and FX-type sounds from a softsynth? This is where the more advanced capabilities of the drum map manager come into play, because, in addition to MIDI notes, each drum sound name can be mapped to a MIDI output port (hard or soft) and a MIDI channel. For each MIDI port and channel combination, the user can specify what bank and program should be used. For each MIDI input note (i.e. what is played or programmed by the drummer), the drum map manager can dictate what actual MIDI note is sent to the relevant soft or hard MIDI module, and it can also tweak the velocity sent by adding to or subtracting from the input velocity and/or scaling the input velocity by a percentage.

While it is fairly straightforward to set up a new drum map, or edit an existing one once you've done it a time or two, it can be time consuming to set up an involved set with lots of potential sounds. The good news, is you can save the map as a preset for future use if it is something you are likely to use on multiple projects.

Once you have a drum map set up and selected for a given MIDI track (or set of tracks), any time you call up the piano roll view for that track, the Drum Grid Editor will appear as a separate pane in the piano roll view. If only drum tracks are selected in the piano roll view, then only the Drum Grid Editor will appear so that space isn't wasted.

Cosmetically, the Drum Grid Editor looks fairly similar to the traditional piano roll with drum note names along the left side, except that only drum notes mapped by the Drum Map Manager for the relevant tracks are included. That can represent a significant vertical space savings if you are using a relatively small number of rhythmic sounds. Additionally, if you'd like to have the sounds appear in some order other than by their input MIDI note number to make programming rhythms more logical visually, it is as easy as assigning the appropriate order of input notes within the Drum Map Manager.

For example, I've never found it very helpful that the open, foot-closed, and closed hi-hat sounds were split up by various tom tom sounds in the MIDI note range. It's not so much that the notes used are a problem -- in fact, I kind of like playing the hi-hat parts on three consecutive black keys while keeping the toms on the white keys. Rather, when I am editing the parts later, or programming them from scratch, I find it disconcerting to not be able to look at the hi-hat part as one entity due to its being broken up by tom sounds. It means too much looking back and forth at the note names, trying to "draw a straight line with the mouse" and so on. With SONAR 2's Drum Map Manager, I just place the hi-hat notes together in the map, and they appear together in the Drum Grid Editor. I still play the same notes I always did on my keyboard.

Besides keeping vertical space wastage to a minimum and allowing more logical grouping of rhythmic parts, the Drum Grid Editor introduces new, more rhythmic looking drum note displays (you can still use the old style piano roll graphs with durations if you like, but most drum sounds ignore durations anyway, so the newer option helps make the rhythm look more obvious). In addition, it provides the ability to show and edit "velocity tails", which are kind of like vertical bar graphs, shown in a set of horizontal lines to make it obvious which parts are notes and which parts are for velocity. This is especially useful when programming a part manually, which usually means with fixed velocity for rapid note entry, then wanting to tweak the feel by altering the velocity of various notes.

The combination of drum maps and the Drum Grid Editor can make programming and editing rhythm parts much more efficient than in the past. For those who want to speed things up even further by using preprogrammed MIDI patterns, whether original, provided as part of the SONAR package, or purchased from a third party, SONAR 2 also provides a new Pattern Brush facility within the piano roll view and Drum Grid Editor.
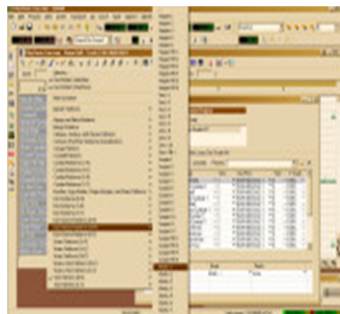
With the Pattern Brush, you simply select a rhythmic pattern from a large menu of presets (and/or patterns you create yourself or purchase), then "paint" the pattern where you want it to go within the Drum Grid Editor.

Cakewalk provides a large number of preset patterns as a starting point, generally divided up by types of patterns, such as kick and snare patterns, cowbell patterns, cymbal patterns, and so on. It isn't strictly necessary that you use, for example, preset hi-hat rhythms for hi-hat parts -- you can paint them wherever you want. However, if you'd prefer they go where they were intended to go, SONAR makes it easy to do that by letting you tell it to "use pattern polyphony", which makes each note in the pattern go to the same note in the part you are painting, regardless of the user's mouse coordination skills.

Mind you, the Pattern Brush doesn't have to be used only for rhythm programming. If you have a library of MIDI groove licks, you can paint pitched patterns just as easily, using the traditional piano roll view.

My biggest wish in this area is that the Pattern Brush facility provided some sort of previewing capability, such as right clicking on a pattern name to hear the pattern. There are just too many patterns for me to remember the differences between, say, "Hoppin' 1", "Hoppin' 2", ... and "Hoppin' 7", not to mention all the other varieties of patterns, and painting and undoing, while quick enough, can be a hassle if you need to audition multiple patterns, especially for a section that needs to extend over a large number of measures.

A secondary wish is that there were some way to make the Pattern Brush menu "stick" to make it quicker to toggle between patterns when doing heavy programming. Having to call up a large, multi-level menu to switch between, say, "Hoppin' 1" and "Hoppin' 2", then to "Hoppin' 3" and back to "Hoppin 1", gets tedious pretty quickly. Of course, you can work around this by painting all your "Hoppin' 1" parts, even though they're not sequential, then doing all the "Hoppin' 2" ones, and so on, but I tend to work sequentially, and this is obviously a workaround for a facility that is good, but which could be even better.

Taken together, the new rhythm programming oriented facilities can provide a big leap in productivity over what was available previously, and for those who need or want to use multiple sound modules within a single "drum set", the ability to do that with the powerful new drum maps represents a significant advance.

Going Soft

I used to have a rack of hardware MIDI modules. Turning each one on, making sure it was enabled in the mixer, making sure it was set to the right mode (i.e. for those that didn't remember my chosen mode across powerdowns), configuring the appropriate channels in Cakewalk Pro Audio, and so on was a pain, and dampened creativity to a degree in that there was almost an obligatory 5-10 minute wait between getting an idea and being ready to start recording it.

When SONAR came along, the DXi softsynths, and VSTi softsynths via a third party adapter, helped immensely. No more did I have to power up a bunch of modules in a hardware rack just because the one I wanted to get at was the last one in the MIDI chain. Just go into SONAR, right click in the FX bin (huh?) of an audio track and pick the module I want, then configure a MIDI track for it, set up some channels, pick a bank and patch, and I was ready to go. What might have taken 5-10 minutes now took a minute or two tops. It was a big improvement, and soon led to most of my hardware modules' getting lonely from non-use.

Still, there was lots of room for improvement. The whole paradigm of putting a softsynth in an FX bin was unintuitive at best, having to solo both an audio track and a MIDI track (and keep track of which MIDI tracks were connected to which audio tracks to make it possible to do that) was a pain, and multitimbral softsynths did not provide the same sort of routing flexibility that multitimbral hardware synths provided. And I have to admit there was something comforting about being able to walk over to a rack of synths and see which ones were turned on, how they were set, and so on.

SONAR 2 builds on the foundation laid in SONAR 1, adding a dedicated Synth Rack, supporting multi-output-capable DXi2 softsynths (and VSTi softsynths via updated adapters), and generally streamlining softsynth routing and operations. For the heavy softsynth user, this means a significant improvement in convenience and efficiency.

Despite my hardware synth background, when I first heard SONAR 2 would have a dedicated synth rack view, my first thoughts were that it was just a cosmetic gimmick that would have little practical value. Pretty much everything that could be done in the synth rack could also be done elsewhere in SONAR, so what was the big deal? As it turns out, once I got used to the presence of the synth rack, and adapted some of my own working processes slightly, its presence really did make a difference in my productivity.

In SONAR 1, when you wanted a new softsynth part, you would create an audio track, put a softsynth plug-in in the FX bin of the track, possibly assign some effects plug-ins to go later in the audio chain, then set up a MIDI track to drive the softsynth. Finally you could get around to playing it, recording, it or whatever. Once you'd tracked a part, you might temporarily hide the two tracks while tracking other parts, but later on you'd need to bring them back into view because playing around with things during mixing would require having both the MIDI and audio track available in case of any need to mute or solo the track, and you'd need both to be soloed and selected if you were bouncing it down to an actual audio track, too. As a result, my most frequent way of working was to just use the softsynth track for tracking, then immediately bounce to audio so I could hide the original MIDI and audio tracks thereafter, only having the bounced audio track to work with subsequently rather than the two original tracks.

In SONAR 2, things become much more flexible, right from the start. If you want to set up a new softsynth track, where the softsynth resides in the synth rack, you can just bring up the synth rack, and use the Insert Synth button, then it will give you options on what gets inserted. In the case of a single softsynth where I will have one MIDI track going to a single stereo audio output, it is convenient to just have this operation create both the MIDI track and the first audio output, which will be connected and functionally linked to a degree by SONAR. However, if you have a multi-output synth, you can even have it create an instance of the audio track for each of the softsynth's outputs. It will still only create one MIDI track to start out, but then you can just insert additional MIDI tracks later, choosing from the appropriate MIDI ports to get the output of the MIDI track. At this point, unless you want to keep the softsynth's audio track(s) around for playing with effects plug-ins, you can hide the audio tracks. Most of the rest of what you'll need to do can be done with just the MIDI track, and possibly also the synth rack window in a few cases, until it comes time to bounce the track to audio, at which point you will need the audio tracks again to allow selecting the relevant range for the bounce.

Once the MIDI to audio track connection is made on a softsynth, double clicking on the MIDI output port in the MIDI track will bring up the softsynth's user interface for any tweaking you'd like to do there. Soloing the MIDI track will solo both the MIDI track and the

connected audio track. If you want to mute all MIDI tracks connected to a given softsynth, just mute the softsynth in the synth rack (or mute its audio track, but the point here is that you don't need the audio track to be visible to do this in SONAR 2). And so on -- that is, things just work very conveniently and logically.

Perhaps the one point of confusion on the front end is coming to grips with the input and output routing. In particular, if the softsynth is in the synth rack (i.e. as opposed to the FX bin of an audio track), then the audio track corresponding to the specific stereo output of the softsynth will have its input as the softsynth's specific output, and the output of the audio track will be one of SONAR's virtual mains. However, if you think about the traditional mixer scenario, this is pretty much exactly how it work. That is, you would run a cable between an output of a synth or other instrument to the input of your mixer, which just so happens to be SONAR in this case. And no cables are needed in the softsynth case.

The bottom line is this enhanced softsynth architecture simply makes softsynth use, well, simpler. While you can still use the old FX bin approach if you like -- I assume the main reason that is still there is for compatibility with older projects as I can't see why anyone would actually want to think of a true softsynth as an effect (and, yes, I do realize that there are cases, such as using Native Instruments' B4 for its Leslie effect only, where a softsynth could actually be used as an effect, and that there are also some effects, such as Alien Connections' ReValver, that are implemented as DXi's) -- putting the softsynths in the synth rack just seems more straightforward. It also provides more routing flexibility. Want to split the output of a softsynth to two different tracks (e.g. for different parallel processing)? Just create an extra audio track to use the same softsynth output as an input. And the intelligent dealing with soloing and muting streamlines operations significantly.

In the same way the 2.0 release makes softsynths come of age for SONAR users, the introduction of and support for DXi2 softsynths makes DXi softsynths come of age. Perhaps the first thing softsynth users will notice is that DXi2 softsynths can support multiple outputs. Want separate processing on the snare and kick sounds from a drum module? If the drum module is a DXi2 softsynth that supports multiple outputs, just route the snare and kick sound to separate outputs from the rest, then route those outputs to their own audio tracks so you can use different insert effects on each track. Or, with a multi-timbral sound module, such as the included Edirol Virtual Sound Canvas, different instruments can go to different outputs, at least until you run out of outputs (the VSC only has 4).

Multiple outputs aren't the only enhancement brought by DXi2, though. For example, DXi2 softsynths can also send MIDI information to SONAR, thus allowing the automation of the softsynth's user interface. For example, if you want to move a filter in real time, that can be done with the mouse in a softsynth that supports it and transmits MIDI information, like the included Dreamstation DXi2 does. And DXi2 also allows softsynths to access SONAR's pitch markers, which is critical, for example, for the ACID-style pitch shifting performed by Cyclone DXi (see above). For the complete list of DXi2 enhancements check out The DirectX Files: Developers. Note that not all DXi2 softsynths will implement all features, and not all DXi softsynths have been updated with the new DXi2 capabilities.

While SONAR does not directly support VSTi softsynths, it is worth noting that the DXi2 enhancements, in conjunction with updates to third party VST adapters, do allow taking advantage of VSTi softsynth capabilities, such as multiple outputs, that could not be used in SONAR 1. Based on the experimentation I did with SampleTank Free, which supports up to 4 outputs, and FXpansion's VST Adapter 4.0 public beta, after configuring plug-ins, the use of VSTi softsynths was as smooth as using DXi2 softsynths.

Between the more efficient interfaces SONAR 2 offers softsynth users and the enhancements that DXi2 brings to the table for softsynths that have been updated, things are pretty good in softsynth land for SONAR users. They're not perfect, though. While it is more a matter of SONAR's architecture and how MIDI track focus is handled than anything specific to softsynths, SONAR's inability to allow MIDI Thru data to go to multiple output ports, or even to multiple channels within a single MIDI output port, is highly limiting. For example, if you want to use multiple keyboards, or a single split keyboard (i.e. that can send on different MIDI channels from the different sides of the split) to track an organ part where one keyboard (or side of the split) is used for the upper manual and the other is used for the lower manual of the organ, SONAR will allow you to record the data on multiple MIDI channels, but it can only send MIDI Thru information to one of those channels. While this could be an issue with hardware synths, too, at least you could work around the problem by routing physical MIDI cables to allow monitoring before the data got to SONAR. With softsynths, the MIDI modules are under the control of SONAR, so you'll only hear whichever MIDI track has the focus, and that means you'll only hear the notes being played on one keyboard (or one side of the split).

Are You ReWire'd?

ReWire is technology for transferring data between software applications, much like an audio cable links a synthesizer to a mixer or tape recorder. It provides real-time audio streaming between applications, sample accurate synchronization, and common transport controls. That sounds interesting in theory, but what does it mean to SONAR users, and how is that different from using DXi2 softsynths?

When ReWire-compatible applications, such as Propellerhead Software's Reason or ReBirth are installed on your system, SONAR 2 allows inserting ReWire devices in much the same way a DXi synth is inserted, including providing multiple audio outputs -- up to 64 with Reason.

Once you have SONAR and your ReWire-compatible application linked up like this, hitting Play in the transport controls of either will start both, and they will be synchronized together. Since I don't use any ReWire-compatible applications myself, I tried loading up the Reason demo, inserting it via the synth rack in SONAR. The default demo file in Reason has a 16 bar loop set, and one of the first things I noticed was that SONAR's loop marker was set to the same 16 bars. Moving the loop markers in either Reason or SONAR resulted in the applicable loop markers' being moved in the other application if playback was running, or the next time playback was started if playback was not running when the marker was moved.

This tight synchronization between, say, Reason and SONAR would allow building up patterns in Reason to be played back in tight sync with live audio tracks recorded in SONAR. But the ReWire support in SONAR is about more than just synchronizing two applications.

Any audio-producing device in Reason can be routed to a ReWire output in Reason, and the "virtual audio cable" then allows bring that into SONAR as an audio input, just like a DXi softsynth can be used in SONAR. Then SONAR can be used for adding DirectX effects, mixing, and other audio processing.

SONAR can also access MIDI modules in Reason just like it would access DXi softsynths. So, for example, a Subtractor (analog-style synthesis) module could be set up in Reason, and a MIDI track in SONAR could be routed to it, then the output of Subtractor could be routed back to a ReWire output in Reason, possibly after going through processing by other Reason rack modules, for bringing back into SONAR as a ReWire input for further processing and mixing.

I'm not a Reason or ReBirth user, but I have to admit that I had a fair amount of fun playing around with Reason's analog-style pattern sequencer, just drawing different shapes and seeing what came out note-wise. That would have little to nothing to do with any music I could imagine myself making, but I also had some fun just playing Subtractor like I might play any DXi softsynth.

My gut feel with the ReWire support is that current Reason and ReBirth users who also use SONAR for recording will now have an easy but powerful way to hook things up. While the support probably won't entice users who don't already have Reason or ReBirth to go out and get those if they're not already considering them -- e.g. all things being equal, I'd rather have a simple DXi softsynth for something like Subtractor than to have to have something as complex as Reason just to use as a sound module (or set of sound modules) -- anyone who is sitting on the fence just might get enough to help push them over.

I Can Manage

Perhaps the biggest historical complaint users who came to Cakewalk Pro Audio or SONAR from other DAW platforms have had has been the lack of control users were given over where their project audio files were stored and, as a result, how inaccessible they were for such routine operations as interim project backups. For some users, too, the design in this area could result in disk space management issues, not due to a lack of available disk space, but rather due not having the extra space where Pro Audio or SONAR wanted it to be.

The issue was that SONAR, and Pro Audio before it, used a single global audio directory, under which it transparently managed all audio files belong to every project on the system. Cakewalk argued that this was actually much more efficient for the user than other schemes where the user had to name audio files at various points along the way, rather than simply hitting Record and letting the system worry about what went on under the hood. Additionally, the transparent file management made it possible for SONAR (and Pro Audio) to share audio files between projects where applicable, for example when one project was another iteration of an earlier project and some specific audio files within the project had not been edited.

Of course, they were right, at least to some degree, but still users wanted more control. Also, there were inconveniences with this system, such as the difficulty in making interim backups. With the old style of audio file management, all audio files were stored in a flat directory, and the user couldn't easily tell which files were applicable to which project. Thus, the only convenient way to backup a project was to save the project as a bundle file, which involved writing all project and audio data into a single, monolithic file. Besides the

inconvenience of backing up a whole project even if one small audio file had changed, it was not unheard of for bundle files to exceed 1GB in size, and this meant the project would be too large to backup on a single CD-R or CD-RW, perhaps the most common backup media by the time SONAR 1 arrived.

Some enterprising users came up with workarounds, such as changing the global audio directory for every single project. However, not only was this a hassle, but it also invited user errors, such as opening up a project with the wrong audio directory set. Other users just made the best of the system, using bundle files for backup, and working around the size constraints as best they could. For example, my main workaround here was to split projects up into logical entities once I was ready to archive a project that had exceed a CD-R's capacity. For interim backups, I just wrote the bundle file onto another hard disk on my system, effectively betting that both of my hard disks wouldn't crash before I got around to finishing the project and archiving its files. That scheme itself had evolved as the result of my having once lost the bet inherent in my previous scheme, which was that my audio hard disk would not crash prior to finishing and archiving the project.

At long last, Cakewalk has seen fit to provide SONAR 2 users with the option of per-project audio file management. Users who are comfortable just using the old global audio file management scheme can continue to do so. In fact, global audio file management is the default for SONAR 2. But users who want more control can now have it.

Electing per-project audio file management is a user preference-level option. When it is set, creating a new project prompts for the project name, and default project directory and audio file subdirectory names are calculated based on that. This treads a fine line between making things efficient in terms of the need for information from the user and giving the user sufficient control.

Once the project is set up, audio track naming also becomes fairly intelligent, with a newly recorded audio file's name starting with the name of the project, then the name of the track, along with some additional information to separate the fields and make sure the file names are unique. Thus, it is a good idea to name tracks early, before hitting the Record button, to make file names somewhat self-documenting. Even when global audio file management is used, the new naming conventions for audio files are used, and this allows the user to get a better sense of to which project a given audio file belongs, though the user still should avoid deleting files in the global audio directory to avoid the possibility of lost audio clips in a project.

SONAR 2 also makes it easy to find the audio files associated with a project or clip. At the project level, there is a new Project Audio Files command in the File menu that lists the names and locations of all audio files associated with the project. At the clip level, the Clip Properties dialog now has an Audio Files tab, which shows the file name and directory path.

In general, the audio files for a project will all be stored in the same directory, whether that is the global audio directory or the per-project audio directory. However, when using a library of sounds, such as from a sound effects or loop library, SONAR can be told not to copy and manage imported files, in which case any imported audio files will be left in their original location. To deal with the prospect of archiving a project that uses this facility, SONAR provides a "Consolidate Project Audio" command, which will make a backup of all audio files in use by a project in a single sub-directory underneath the project's audio folder (or the global audio directory if per-project audio is not on use on the current project). Beware, though, that this feature will double the active audio storage being used by a project.

Okay, all this flexibility is nice, but what happens when the inevitable user error occurs, and a user moves a project file to the wrong directory, renames directories, or does something else that results in a disconnect between the project and its audio files? SONAR 2 provides a few levels of protection. If, for example, all you've done is rename a top level project directory, but not the project (which will have the same name by default), no problem, SONAR adjusts without complaint since it is just looking for an Audio subdirectory. If, on the other hand, you rename or move the Audio subdirectory, or move the project to another directory that does not have an Audio subdirectory, SONAR will give you the option of creating a new Audio subdirectory, and will then bring up a dialog box to let you search for missing files, which it can copy to the new directory, move to the new directory, or reference in place. Once you've found the first file, assuming the other files are in the same directory as the first file, it will find the rest without asking you where to look again. If you don't have it create a new Audio directory, it will use the global audio directory for any audio that is not referenced in place for that project. If you've moved the project to another directory that has an Audio subdirectory, but not the right one, SONAR will also let you look for the missing files and copy them, move them, or reference them in place once you find them.

Now, I'm sure there are ways to mess things up so SONAR won't let you recover, but all these safeguards, and the amount of flexibility they provide, should cover most non-disastrous user errors. Of course, if you've deleted the Audio directory and all its files, you're on your own, at least unless you've taken advantage of the Consolidate Project Audio feature to make a fairly recent offline backup of the files, or have just been backing up changes to your project audio directory on a fairly regular basis. And, because the per-project audio management makes it easy to just backup the few files that have changed recently, the odds get better that users will make regular interim backups -- at least if they've ever been seriously burned by disk crashes or losing files due to their own errors in the past.

Another Plug-in OPT?

Yamaha's Open Plug-in Technology (OPT) was announced in early April of 2002 as a "new open plug-in format for the control of MIDI devices from within music software and sequencing products." According to Yamaha's press release OPT is intended to allow "seamless integration of external hardware devices and control surfaces with synthesizer editors, enhanced editing views and other MIDI processing tools." The basic intent seems to be to make music hardware, such as MIDI modules and control surfaces, just as controllable and usable as softsynths from within a software environment such as SONAR.

The philosophy espoused in a quote in the press release suggests that the ultimate goal of OPT is to integrate hardware and software environments into a seamless music production

environment. That sounds great and noble and all that sort of thing, but what is actually deliverable right now?

For the moment, the answer is, "not much." I did a web search, and about the only references I came up with were from Yamaha and Cakewalk, and the only actual product references I found were SONAR 2's support for OPT panels, which are only one of three levels of OPT integration defined by Yamaha, and a free voice editor from Yamaha for their Motif 6, 7, and 8 synthesizers. A Yamaha OPT page also indicated that Sonic Foundry is planning to support OPT, but I was unable to find any specific product references.

For those SONAR users who do have a Yamaha Motif synthesizer, though, installing the editor results in an "OP Panel" sub-menu's being added to SONAR's View menu. Clicking on that, brings up a list of any OPT panels installed, in this case the above-mentioned Motif voice editor, and selecting the panel brings up a fairly generic-looking editor with a list of patches, a list of drum sets, and three lists of plug-in settings. Kind of boring thus far, but double clicking on a specific patch, drum set, or plug-in name spices things up a bit.

The result is to bring up a very nice-looking (see above) visual editor that allows editing the applicable patch in a very user-friendly way, reminiscent of editing on a highly visual synth's front panel. Unfortunately, I didn't have a Motif synth to hear the results of my playing around with the editor screen, but it looks like this editor would be quite nice for the SONAR user who does have one of the applicable synths and enjoys tweaking sounds.

The net is that Yamaha Motif synth users should enjoy this feature now. Whether it will ever benefit other SONAR users remains to be seen, and will depend to a large degree on how successful Yamaha is in promoting this new plug-in "standard" with other vendors, including Yamaha's direct competitors. If anyone comes across an OPT panel for a Roland Rhodes MK-80 keyboard or JV-1080 module, though, be sure to let me know!

Doing More ACID

SONAR 1 was the first fully featured DAW application to include ACID style looping capabilities. In addition to the ability to play back ACID format loops, complete with time and pitch stretching, SONAR 1 provided the ability to create loops and edit existing loops. For some strange reason, though, SONAR 1 did not provide the ability to save those newly created or edited loops in ACID format.

For SONAR users, that seeming oversight was an inconvenience in that it would at least be possible to copy a loop from one project and paste it in another -- not something I'd want to do a lot of, but it might do in a pinch. It made it even more difficult to share original loops with other SONAR users -- you'd have to create a project bundle (CWB) file to send to the other user since the loop itself couldn't be exported in ACID format. But it was an absolute showstopper for the user who wanted to author ACID-style loops to make available to ACID

---

users or as a loop library for other SONAR users. SONAR 2 remedies this by adding a Save button in the Loop Construction view.

This seemingly trivial enhancement now means SONAR users can create loop libraries for private or public consumption, and the loops they create can be used in SONAR, Cyclone DXi, ACID, and any other applications which support ACID format loops. Given how easy SONAR's Loop Construction view is to use for creating ACID-style loops, along with all SONAR's other audio processing capabilities, not to mention the new loop slicing and dicing capabilities of Cyclone DXi, this new ability to actually save loops should make SONAR 2 a very nice platform for loop authoring.

Gentlemen, Tune Your Engines

Enhancements can be nice, but a fairly fully featured application like SONAR 1 could go a long way toward satisfying many users just as it was. Not everyone needs ReWire support, ACID loop authoring, or even drum programming. In fact, I know it's hard to believe for those of you who, like me, have made the dramatic move from a hardware rack to softsynths, but some SONAR users don't even use softsynths! So what does SONAR 2 bring the user who was, by and large, happy with SONAR 1?

Of course, there are the inevitable bug fixes, but, perhaps more significantly, SONAR 2 just seems to run better -- i.e. more smoothly, and, in some cases, with better latency. Cakewalk doesn't seem to publicize what they've done under the hood, but I've noticed, and I've also heard other users mention, that SONAR 2's audio engine seems to drop out a lot less than SONAR 1's did, especially when getting up high on the CPU usage meter. It can seemingly go quite a bit higher without dropping out when a periodic spike comes along, and this is without any special tweaking. For those who don't mind getting their hands dirty, though, read on....

For Tweakers Only

If you are the type of user who doesn't mind editing a .INI file if it can make a significant difference in performance and/or usability, SONAR 2 has a few fairly hush hush (as in "not listed in the marketing hype, but you can find them in the ReadMe file if you know what to look for") enhancements that may be worth the price of a SONAR 2 upgrade in and of themselves. For example:

There are two new variables, KSUseInputEvent and ExtraWaveOutBuffers, in AUD.INI that affect input monitoring latency. Whether they'll help or not is sound card-dependent, but, if they do work for your particular sound card, they can get input monitoring down from the "barely acceptable" level to the "emminently usable" level. In my case, tweaking KSUseInputEvent did seem to help input monitoring latency with my MOTU 2408 mkII.

There is another AUD.INI variable, StopIfStarved, that can tell the audio engine to keep going even across a condition that would normally cause a dropout. Users who use softsynths a lot, and have been frustrated when a dropout occurs when they're just playing the softsynth or tracking a MIDI part, may want to set this flag to keep on going since the audio engine glitch isn't critical to what they are doing at that point, but an audio engine dropout might break their creative flow. A dropout while tracking audio is more significant, but some users may still prefer to set this flag to keep on going, preferring to just keep the session going, even though the portion of the take with the glitch may need to be overdubbed or edited out.

There are some new, optional sections in CAKEWALK.INI that allow the user to rearrange the order of controls, which Cakewalk calls "widgets", in the track properties section of SONAR's track view. The layout of controls in the track strips for audio tracks, MIDI tracks, Auxes, and Virtual Mains can be adjusted. Different users have different styles of working, and there is no satisfying everyone with a single layout for each of thes controls. It would be great if SONAR allowed dragging the controls in any order we wanted them, but it doesn't. At least we tweakers can get under the hood to customize the layout for our personal working styles. I can't overstate how big a productivity boon having the track control order suit your personal working style can be, especially as track sizes get made narrower to fit more tracks on the screen.

The XL Difference

Everything I've described thus far is available to all SONAR 2 users. As has been Cakewalk's practice in the past, there is a premium bundle available for SONAR 2, called SONAR 2 XL, that provides a few extra goodies. In SONAR 1, the value of the XL package primarily depended on your feelings on the value of the Tassman softsynth, which got decidedly mixed reviews from SONAR users.

This time around, Cakewalk has done an XL-ent (excuse the bad pun) job of providing value that should appeal to a wide variety of users. Of course, SONAR 2 XL users get all that is in the basic SONAR 2 package. In addition, they get the Sonic TimeWorks' CompressorX and Equalizer plug-ins and FXpansion's DR-008 soft drum module.

CompressorX is an analog-style compressor/limiter that is best suited to track insert uses, though it can also be used on the mix and in mastering. While its user interface leaves a bit to be desired in the readability department, its sound quality is in the same league as the best software-based compressors on the market. It is also fully automatable in SONAR.

The TimeWorks Equalizer can operate in "clean" and "vintage modes", depending on whether the user is looking for fairly clinical processing or that analog-style feel. While there are other plug-in EQs on the market that provide the vintage sound possibilities, where the TimeWorks Equalizer sets itself apart from the pack is in providing a graphic mode where the user can see an FFT-style, real-time bar graph to show the spectrum of either the sound being processed or the processed sound, setting EQ values on the same graphic display. While the legibility of the display also leaves much to be desired, due largely to small fonts and a poor choice of colors, the feedback from the FFT meters more than compensates, and makes it perhaps the easiest EQ there is to use for users, like me, who might not have golden ears but who can be aided greatly by visual feedback. While the Equalizer can be automated, the automation is, unfortunately, only applicable to its default (i.e. non-graphic, "hardware look") mode. Thus, my tendency is to use the TimeWorks Equalizer when I can "set it and forget it", and use SONAR's built-in FxEq (from DSP-FX) when automation is required.

The FXpansion DR-008 is one deep softsynth. It is equal parts drum synthesizer and drum/percussion sample playback module. Playing with the analog-style drum machine side of it brought back memories of my mid-80s Roland TR-909, at least on the sound manipulation side. But the sample playback side also allows setting up realistic drum kits, including defining interaction between drum pads, for example, to program realistic hi-hat behavior with different notes serving as open, closed, and half open or foot-closed behavior. Suffice it to say, there is a significant amount of power for building most any kind of drum kit imaginable, but that same power can make it daunting trying to figure out where to start

with the DR-008. Luckily, a few canned kits are provided, though I do wish more of them had General MIDI compatible layouts for the subset of sounds they offer, and that there were one full General MIDI kit there for "starter" purposes.

All in all, the SONAR 2 XL package brings significant extra value to the SONAR user who doesn't already have its components. In fact, some users indicated they could justify the entire SONAR 2 XL upgrade over SONAR 1 simply due to the savings on the XL package components.

Final Thoughts

When I first saw the feature list for SONAR 2, I was somewhat underwhelmed. I didn't need ReWire support, Yamaha OPT panels, or control surface support. I didn't even know what a "groove sampler" was, no less how I might use it, and I don't make my own audio loops. The project management features were of interest, though I'd also gotten comfortable enough living without them. I thought the synth rack was just a cosmetic change, and none of the softsynths I used frequently supported multiple outputs. The most interesting part was the drum editor, but was that enough to justify an upgrade? What most strongly sold me on the upgrade from the feature list perspective was actually the TimeWorks plug-ins in the SONAR 2 XL package, which more than justified the cost of the entire SONAR 1 XL to SONAR 2 XL upgrade such that it really didn't matter much to me if the rest of the SONAR 2 updates were fairly "minor".

After using SONAR 2 for almost two months now, though, I have to say there is a lot there that isn't obvious from a reading of SONAR 2's new features. For example, being a heavy softsynth user, the improved handling of mutes and solos and synth insertion gives me major productivity improvements. The improved engine stability and performance, especially with some of the tweaks applied for my work patterns and audio hardware, means I am running back to the computer much less while arranging and tracking MIDI parts. And being able to customize the track strip controls for my working style means a lot less resizing of tracks. And, yes, I do use the new per-project management on most of my projects, and find it makes managing my disk and audio files much simpler. The TimeWorks Equalizer in my XL package has become my first call EQ plug-in (CompressorX is in my top 3 compressor plug-ins, too), and I find myself reaching for the DR-008 increasingly for drum tracks.

I guess, if I were called upon to summarize the value of the SONAR 2 or SONAR 2 XL upgrade, I'd probably say, "it's the little things." Perhaps there is no one feature I might single out as compelling, but, when you add it all up, the SONAR 2 upgrade kicks serious butt! Oh yeah, and, while SONAR 1.31 was quite stable for me, SONAR 2.0 is even moreso, so no need to worry about that ".0" release stuff.

Have you upgraded to SONAR2 yet? If so, what features made you decide to upgrade? Which features most pleasantly surprised you after using SONAR 2 for awhile? What are your most pressing additional needs for a future version of SONAR?

*Rick Paul is a songwriter living in Southern California. You can contact him at http://www.RickPaul.info.