The Omnipressor® Redux-The Anatomy of a "Preset"

What are Presets?

The original effects processors of the '70s and early '80s didn't have "presets." They had knobs, buttons, switches, and blinking lights just as the gods of rack-mount equipment intended them to have. A delay line had a knob for adjusting the delay. An equalizer had a few knobs or maybe a group of slide faders for adjusting the level at various frequencies. Then things got complicated. Unlike the original processors that used shift registers for delay or even *circuitry* for frequency adjustment, they now all have DSP chips and microprocessors that do *everything*. You can't connect a knob to an address line or a delay-tap switch to a data bus, so you rely on software that talks to you and to the DSP. This software tells the chip how to create the effect you want, and allows you to "tweak" the effect by the adjustment of parameters.

The utility of an effects processor is, in large part, determined by this software and by the creativity and initiative of the people who tell it how to create the effect. Simple effects still have their uses, of course, but things start getting interesting when you combine, for example, a number of delays with gain and frequency-dependent processes, and then automatically pan them among outputs. The person who purchases a processor for use in his studio is rarely an expert in DSP programming and/or effect design-he is interested in getting the right sound for his mix. Because this is true in most cases, manufacturers of effects equipment concentrate on creating "presets," combinations of DSP processing program fragments that combine to create something musically (or eccentrically) useful. Because these presets are composed of several elements, each of which may be adjustable over a musically useful range, the "controls" of these elements, such as amount of delay, feedback gain, or whatever, are attached by the software program to user adjustments-knobs, faders, buttons, etc. These are the adjustments referred to as "parameters." Most commercial effects processors use this strategy. Folks at the factory put together presets, give them whimsical names, screen them on the panel or put them in software menus, and then flog the product citing the number of presets. For the majority of customers, this is fine. But what if they want more?

Beyond Presets

By "more" I mean the ability to create sounds "limited only by their imagination?" Back in the days of modular synthesizers it was possible to do this. You bought a number of VCAs, VCOs, mixing modules, ring modulators, etc., and then "patched" them together with pieces of wire. There were limitations: Modules were expensive. Modules came in a limited variety. The modules were analog and so were a bit unstable and somewhat noisy. And, they worked largely with their own signals and couldn't process external inputs in a very useful way.

Even so, some very creative and very interesting material came from the old Moog, Arp, and Buchla hardware. So what if it filled the room and required enough patch cords to circle the Vampire Stoat building? Now that it is possible to create these (and many, many more!) modules digitally, anyone can have "more."

This article is being written with three purposes in mind:

1: To show how a current generation signal processor (The Eventide Orville and the earlier DSP4000 series) can be used to create sounds "limited only by ones imagination."

2: As a practical example of this, to provide a mini-tutorial on how at least one vintage product-the Eventide Omnipressor-has been re-created.

3: To justify the time that I spent recreating the Omnipressor, a really neat box which otherwise would remain a footnote in audio history and unavailable for experimentation.

Hopefully others will be encouraged by this example to experiment with their Eventide DSP units to create their own "signature" effects and perhaps even other product recreations.

The Omnipressor

The Eventide Omnipressor was one of our earliest products. It existed in two incarnations, the 2826 (white meter) and 2830 (black meter) versions and was built in limited numbers between the early '70s and early '80s. Unlike the LA2A limiter and other vintage products, it has not achieved such "cult" status that it's worth recreating in metal and silicon. (And, since we have the great good fortune of still being in business, I guess nobody feels entitled to steal the design or the name.) Even so, it was a popular and, to some, vital product. To this day we get letters…

The Omnipressor is a "dynamics" controller (expander, gate, compressor) with an unusually large gain control range and highly-variable control law. It's distinguished by a large "Function" knob calibrated smoothly from extreme expansion (gating) to extreme compression. The compression is so extreme that you can achieve what we call "dynamic reversal" or the condition where a loud signal comes out soft and a soft signal comes out loud! If you are familiar with a normal compressor in which the output signal is fed back to the gain control stage, you will realize that the gain of the feedback must increase without limit as one gets closer to "infinite" compression-the state at which the output level remains the same no matter how great or small the input. Dynamic reversal using this scheme is theoretically impossible because it would make the gain control loop unstable. The Omnipressor achieves dynamic reversal by using the input signal rather than the output to determine the amount of gain change. Since there is no feedback, there is no potential instability. Dynamic reversal has a very interesting sound. It is particularly applicable to percussion instruments with a decay trail such as snare drums and cymbals. In addition to the signature Function control the Omnipressor has a number of additional adjustments (parameters, if you will) that allow you to tame the otherwise extreme level changes that can result with unbridled usage.

Omnipressor Preset Creation Overview

While the Omnipressor itself is no longer being manufactured, we can recreate all of its functions electronically in the context of a general purpose signal processor. Eventide makes a line of DSP products that give the owner the ability to create presets by "patching" together standard modules. These modules are described in the product manual, and fall into various categories, such as audio processing, control signal processing, and user interface. There is also available a "freeware" program called "VSIGfile" on the Eventide web site. This program, in essence, gets a "database" of modules from the processor and stores it in a PC. It then allows you to manipulate (either in text form or graphically) these modules to perform a useful function, and sends the instructions for the function back to the processor to be executed. Using VSIGfile is an interactive exercise. Unlike the "build" process for complex software development, which can take minutes to hours, each module "build" consumes only a few seconds. You can make a quick change or try an experiment and immediately have the processor execute it to see how it sounds.

There are three general methods of patching together modules. One requires nothing but the Harmonizer brand processor itself. By using the internal "patch editor" you can do anything in the unit that you can

do with the external program. This is very convenient for modifying existing patches, or creating simple presets, but the large screen of a general purpose PC is much better for presets of moderate or greater complexity. The second and third methods are complements of each other: VSIGfile-created patches come in graphic or text form, and you can switch your "view" of the preset between them. The two are equivalent, and I find the graphic form more intuitive.

Before you create a preset, you should have a good idea of what you want to do. In this case, the preset was defined by the product to be emulated. The Omnipressor has six knobs and three sets of switches on the front panel, not including the power and IN/OUT switches (which are also features of the Harmonizer models). Each of these controls interacts with the user and with the circuitry in a defined way. In order to reproduce their effect, one must analyze how they work in the original unit and how the user adjusts them, and then put together a set of modules and connections as necessary to achieve the same effect. I assembled the following items for this project:

1: Orville™ Harmonizer brand effects unit. (The DSP4000/4500 series can be used with a MIDI connection instead of a serial connection.)
2: PC (not a Macintosh) with serial port cable.
3: Omnipressor model 2830 and its instruction manual.
4: Audio oscillator with step attenuator to provide input signal for both.
5: Microphone and preamp for sound comparison
6: Earphones. (The Omnipressor can have up to 50dB gain, so you will probably have feedback problems if you use speakers.)

Programming Style

Professional programmers, of which group I consider myself a non-member, are keen on analysis. Before writing a line of code they will define, analyze, and plan. I prefer to code, test, and code. Every little chunk that works is encouragement to proceed. In order to give myself that encouragement, I decided to start out with the Omnipressor controls, a simpler issue than the gain computation. Since I wanted this to be a functional recreation of the original product, the control portion was pre-defined for my convenience. The controls are used to adjust the characteristics of the heart of the Omnipressor, the level detection and gain control "circuitry." Consonant with this style, I shall simply move along, and discuss the various simulation issues as they come up.

I have placed scanned pages of the original Omnipressor manual on the web at (URL HERE). Feel free to refer to the manual to supplement any discussion of the schematic, control functions, etc that may seem sketchy. The VSIGfile program is available on the Eventide web site. You can download it to follow along with this description, or even to recreate the preset yourself.

The Omnipressor Panel and User Interface

Here are the controls that must be provided:

Knob modules (standard linear potentiometers in the Omnipressor)

1:	Threshold	-25 to +15
2:	Attack Time	.1ms to 100ms
3:	Release Time	1ms to 1sec
4:	Function	10 to -.1
5:	Atten Limit	-30 to 0
6:	Gain Limit	+30 to 0

Switches

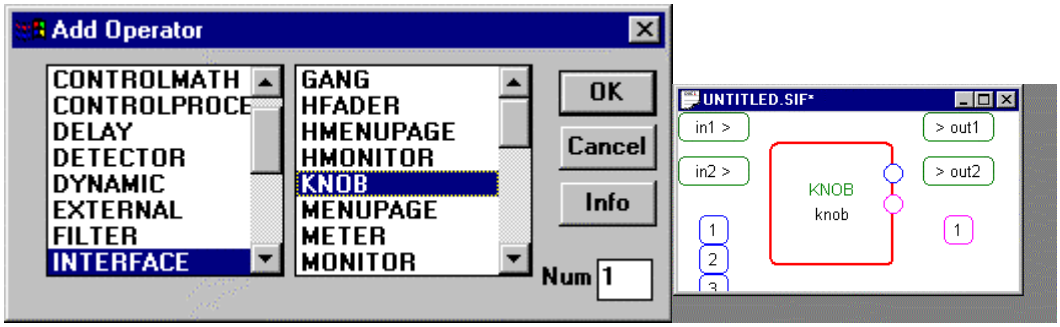
7: Bass NORM/CUT
8: Meter Funct: INPUT/GAIN/OUTPUT
9: Output Cal 0Db/+10Db/+20Db (Misspelling of dB on original panel!)

The most natural way emulate a knob on the Omnipressor panel is to use the Knob module on the Harmonizer panel. Since there is only one Knob on the Harmonizer, there must also be a way to select which Omnipressor control is associated with the Knob at any time. We use softkeys and cursor keys to highlight the active linkage.

The Threshold Knob and other controls

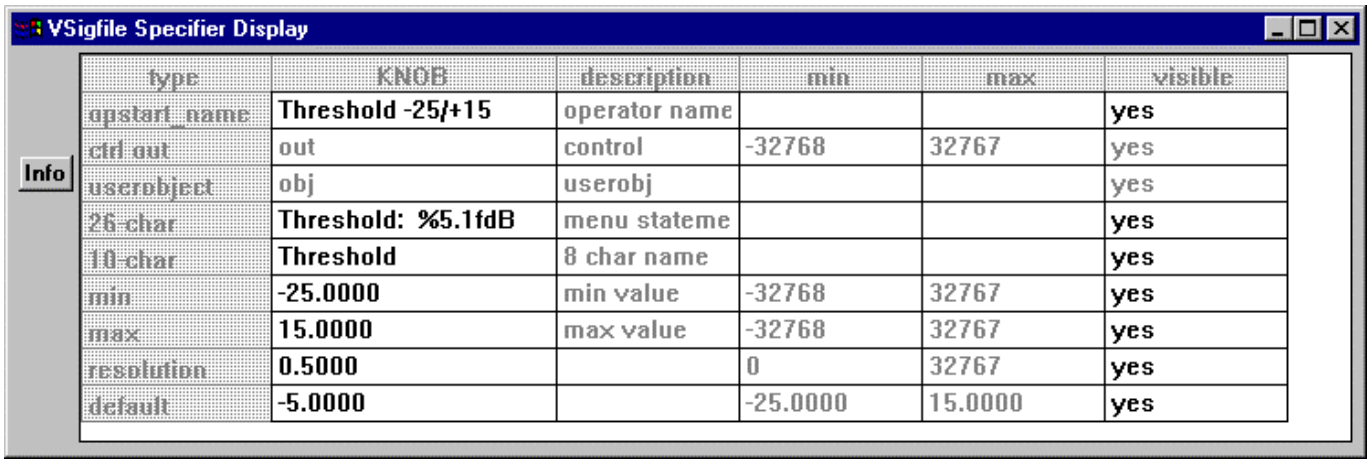
Our goal is to create a "knob" whose value will emulate the Threshold control on the Omnipressor. Looking at the front panel of the Omnipressor itself we see a control calibrated from –25 to +15 with marks every 10. The "units" of calibration are, in fact, decibels, although that isn't shown. Since this is a continuously rotating control, it clearly has better resolution than every 10 steps. We are faced with a decision, then, as to how to "calibrate" the simulated knob. The Harmonizer module will allow you to create a control with an arbitrary range between –32768 to +32767 that has resolution steps as small as .0001. A compromise is necessary since you don't want to spend the rest of your natural life trying to get the threshold from –5dB to +5dB. In this case it's pretty easy. We set the upper limit of the control at +15 and the lower at –25. Since we know the calibration is in dB, a logical setting of the resolution of this control could be anywhere from 1dB per step to perhaps .1dB per step. I selected .5dB per step as a good compromise between how many turns of the Harmonizer Knob are required to select an arbitrary value and the desire for better resolution.

Having made this decision, it must be implemented. (Following along in VSIGfile if you wish.) Click on Edit/Add Module and select Interface with the mouse. This puts a list of interface modules in the "Add Operator" window. Double click on "KNOB." The window goes away and a KNOB module appears in your workspace.



We need to do two things with the KNOB module. We have to implement our decision about what it will do, and we have to label it and locate it conveniently for later on. Double clicking on the module anywhere but on the bottom instance of its name will bring up the "Specifier Display." This is a window in which we can enter the knob control characteristics. In the case of a KNOB, we can select its minimum and maximum values (-25, +15), its resolution (.5) and a default value. The default must, of course, be between the minimum and maximum. In this case I selected a midrange value, -5, for the default threshold. The rest of the specifier concerns how the knob information appears on the Harmonizer screen. This formatting option allows you to select how many digits of a knob's characteristic are displayed and what it will be called. Commensurate with the minimum and maximum values and the resolution, we will

select a format that displays “Threshold xx.xdB” where the “x” stands for a digit. Since the number can be positive or negative we reserve a space for the minus sign. You can also name the module in the Specifier Window. In very simple programs you might not bother but since we will be using a handful of knobs let’s call this one simply “Threshold.” (You can also change the name of a module by double clicking on the bottom instance of its name in the workspace.) Finally, the module can be located conveniently in the workspace. In designing the program I put the controls on the bottom of the screen, left to right and in (mostly) the same order as they appear on the Omnipressor panel. Click and drag the knob module, now labeled “Threshold,” to the lower-left corner of the workspace.



type	KNOB	description	min	max	visible
opstart_name	Threshold -25/+15	operator name			yes
ctrl out	out	control	-32768	32767	yes
userobject	obj	userobj			yes
26-char	Threshold: %5.1fdB	menu stateme			yes
10-char	Threshold	8 char name			yes
min	-25.0000	min value	-32768	32767	yes
max	15.0000	max value	-32768	32767	yes
resolution	0.5000		0	32767	yes
default	-5.0000		-25.0000	15.0000	yes

Having gone through this process in painful detail for the Threshold control, I’ll leave it to you to create the rest of the Knob controls. Using the table above, set the minimum and maximum as shown and make a reasonable guess for the resolution. Follow the instructions in the Harmonizer manual for setting the display format, which is less intuitive than the rest. Label the controls and select the defaults. I recommend setting the default for the Attack and Release time at their minima, the Function control to zero, i.e., no compression or expansion, and the two Limit controls to their maximum, i.e., the least restrictive setting which is at their fully CCW rotation.

Recreating the switch functions with “Text Knobs”

There are two more user interface issues: the meter display and the two switch controls. Let’s deal with the switches first. On the Omnipressor itself there is a “Bass Cut” control. The “Cut” position provides a factory selected roll-off in the frequency response of the level detector channel. While it would be trivial to improve on this function by making the frequency a variable one, our intention is to reproduce the Omnipressor. To do this, we need to somehow provide a control that has two positions, “NOR” and “CUT.” By adding a “TEXTKNOB” control we can display and select from two (or more) non-numerical items. Again, we use the Edit/Add Module function and add two TEXTKNOBS to our workspace. One will perform the CUT/NOR function, the second will control the Output Cal function by adding 0, 10, or 20 dB to the output level. The TEXTKNOB modules provide the user display and also an internal numerical interface to the signal processing modules. Selecting 0/10/20 on the Output Cal control generates a number of 0/1/2 corresponding to its “position.” In order to convert this to a gain of 0/10/20 dB, we will connect the “output” of the control to a module that multiplies by 10, and then apply the output of the multiplier to a module that scales the amplitude of the audio signal.

Adding Meter(s) to the Omnipressor

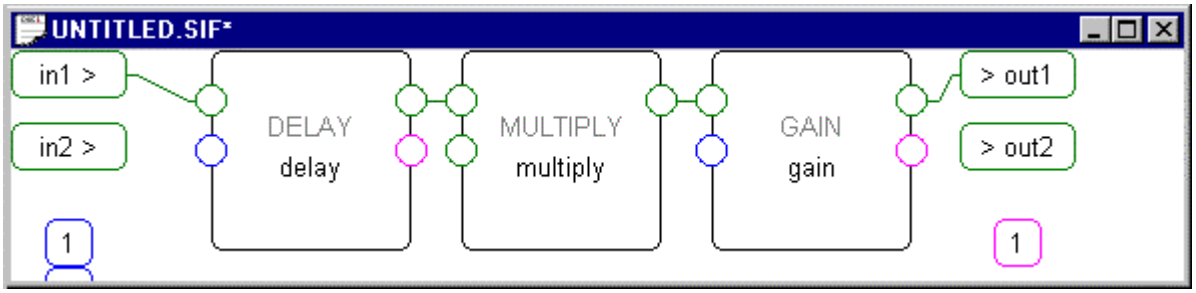
Finally we come to the meter. The meter on the Omnipressor is vital for setup and to see what’s going on with the audio signal. Physical meters are expensive and large and delicate, so there was no question of

putting more than one on the original product. However, being able to monitor Input AND Gain AND Output simultaneously would have been a significant boon. It is actually possible to display up to four "meters" or as many as eight bar graph level indicators on the Harmonizer. Rather than dogmatically restrict ourselves to a single meter and switch, I decided to eliminate the switch and put 3 meter displays on the Omnipressor simulation. As with most modules, the Specifier box allows wide and arbitrary ranges for the meter displays. Since the Omnipressor uses a single scale calibrated from –30 to +30, I simply created three meters that read from –30 to +30. Since these are electronic meters, the "pointer" can move as fast or as slowly as desired. The basic meter module display changes almost instantaneously, so I added a filter to emulate the time constant of the mechanical meter on the Omnipressor.

Before we move on to the next section, note that that these meters are calibrated from –30 to +30. These are arbitrary units since the internal "levels" in the Omnipressor preset will be quite different. Just as the physical meter movement (0-100 microamps) has a mathematical correspondence to the printed meter scale (a multiplier plus an offset), the DSP equivalent of this multiplier and offset must be applied to get the meter to read correctly. Since I mentioned that the meters could have arbitrary calibration themselves, you might wonder why bother with scale and offset modules instead of just calibrating the meters to the correct values to begin with. It's a good question! I did it to help with the thought process involved in designing the preset. In fact, several modules can be eliminated in this way, but it's more important right now to understand the process than to save a few machine cycles or a bit of graphics on the computer screen.

The Audio Path

All the controls described above serve a fundamentally very simple purpose. They all affect in one way or another a single voltage value (in the Omnipressor) or a single number (in the emulation). That value/number represents the instantaneous gain of the audio path. To take a break from the complexity above and following, let's note that the audio path itself goes directly from the input to the output, being interrupted only by three modules: a Delay module (of a fraction of a millisecond-another "improvement" over the original), a Multiply module whose other input comes from the gain control value and which dynamically controls the Omnipressor gain, and a following Gain module, which statically adds 0, 10, or 20dB of additional gain depending upon the setting of the Output Cal control.



DSP and Control Resources

That's the whole audio path! Since this section is so brief, let's introduce another important concept. Control modules *vs*. Audio modules. All computational products have limited "resources." Just as your 450MHz Pentium can do more work in a given time period than can your old 486, so can a fast DSP processor do more than a slow control CPU. It is always the goal of equipment designers to use resources most efficiently, and that concept enters into preset design. The Orville Harmonizer has much faster DSP chips and a much faster control processor, and so can do more processing than can the DSP 4000/4500 series. Even so, it is always a good idea to conserve resources as you might want to add additional functions or emulate four Omnipressors (or more!) at once. To allow optimization, modules that work at

different speeds are provided. While the audio path (by definition) must work at the unit's sampling rate, modules connected to manually operated control knobs can operate orders of magnitude more slowly. These control modules run on the slower control processor and don't consume valuable DSP resources. To allow the control and audio modules to communicate, "bridge" modules are provided. A third, intermediate speed, function (not used in this preset) does use DSP resources but performs its function once every four audio samples instead of every sample. This is more than fast enough for high speed operations such as modulation, sweeping, etc., and yet makes modest demands on the DSP.

The major challenge in getting the Omnipressor preset to function correctly was to keep the entire gain control chain from the input of the preset to the input of the gain control multiplier in the fast "audio" speed realm. The actual Omnipressor had an attack time adjustable down to about 100 microseconds, and putting any slower "control" modules in the direct path would degrade this capability. On the other hand, using "control" modules in quasi-static functions such as the human interface is desirable.

The Gain Control "Circuit"

We are now at the heart of the Omnipressor. Here's how the heart functions:

1: An audio signal comes in. It's level (amplitude) can typically vary from clipping to inaudible. We are concerned with signals typically ranging from +15 to –25dBm, i.e., above and below the adjustable setting of the threshold control.

2: We detect the level and convert it to a rapidly varying number. We apply an "attack" and "release" time to this number so that it doesn't vary too rapidly-otherwise the signal would vary at an audio rate instead of the typically much slower rate desired, especially for "release." We optionally gently roll off the low frequency component so that very low frequency, high amplitude signals (e.g., bass drum) don't have as great an effect on the detected level as they otherwise might.

3: We take this detected signal representing the level of the input and modify it in several ways:

A: We set a Threshold as a reference, so that the level signal, when subtracted from the Threshold, will take on a negative value below the threshold and a positive one above it.

B: We multiply the resulting signal by a variable controlled by the Function control that, in effect, goes from +1 to –1. When multiplying by +1, the signal representing the level remains the same as it did before; when multiplying by –1, this signal *decreases* as the signal input increases. When the variable goes through 0, the level signal remains at a fixed value regardless of the input signal .

C: Optionally restrict the positive and negative excursion from B: so that (again, in effect) the signal may only go from, say -.5 to +.7. If we restrict its positive excursion, it reduces the maximum gain available; the negative excursion represents attenuation. These restrictions (the Atten and Gain Limit controls) operate after the Function control so that whether the signal representing Level is normal or inverted, the limits always refer to the appropriate function, i.e., attenuation and amplification respectively.

This conceptual description will be complete after one important addendum: For convenience in operation and calculation, our control signals are all linear. However, decibels are a logarithmic measurement. To attenuate a signal by a number of decibels, you must *multiply* it by a fractional number. In order to make this project work, we manipulate the logarithm of the level signal, and so convert our circuitry from the multiplicative domain of normal audio gain and attenuation and convert it to the additive world of control signals. Here is an explanation of how gain control functions are realized.

Putting together the Gain Control modules

The audio input is connected to a switch module that either routes it through or around a filter module. Although this is a general purpose filter that could be high-pass, band-pass, low-pass, or notch with a flick of the wrist, all of its control inputs are set to specific values that make it act like a low-Q, high-pass filter with a corner frequency of about 100Hz. The Omnipressor itself simply uses a capacitor in series with the signal that is shorted with a switch. The filter module characteristics were determined empirically, by matching the meter readings on the Omnipressor with the "meter" on the Harmonizer. The input then goes to a "log" module, which calculates the (bipolar) logarithm of the instantaneous signal. A conventional logarithm calculation excludes zero, which is mathematically undefined; the module used here is optimized for this practical application.

The log of the instantaneous level is applied to a Peak Detector module. To this module we attach the Knob controls created earlier. In this way the Attack and Release times are under user control. The output is still bipolar, but we want a unipolar signal that is representative of the signal level. Therefore the output of the Peak Detector is routed through the ABSolute value module. A little aside here: In the analog domain of the hardware Omnipressor a single capacitor did the job of the rather complicated filter used in the digital version. In this case, however, a very simple DSP operation-converting a negative signal to its positive counterpart, takes the place of what would be a complicated precision rectifier circuit. There are further examples of this duality in the Omnipressor emulation.

A detour to Meter the input signal

From the ABS module we have a signal that represents the level of the audio input, and the appropriate attack and release time constants have been applied to it to use it, after further processing, to control the Omnipressor gain. But there is also something we must do with this signal just as it is. Recall that one of the metering functions is to show the input level. We can apply this signal to as many module inputs as we choose. (Think of the DSP modules as having zero output impedance and infinite input impedance.) In order to measure the input level, the signal must be connected to a "Meter" module. Further recall that we set up three meters earlier, each calibrated from –30 to +30. The output signal from our ABS module is a (potentially) high-speed signal typically in the range from 0 to 1. We want to convert it, therefore, to a smoothed signal in the range –30 to +30. To do this, we connect the output of the ABS module first to an "A to C" module, which takes the high-speed, resource-intensive audio signal and converts it to a low-speed control signal. This control signal output goes to an adder, whose other input is a constant, which we dial in as appropriate so that the output from the adder is zero when the input signal also reads zero on the Omnipressor that we are using as a comparison. (For conceptual clarity I'm not referring to the exact numbers, which are determined empirically. A listing of the "preset" has all the precise details.) We have now set the meter "offset," i.e., a zero reading on the hardware and on the simulation will occur at the same input signal level. To adjust the "scale" we put the offset signal through a "C Multiply" module. Again, this is a lower-speed, non-resource-hogging version of the DSP Multiply module. Since we want a fixed multiplier, we dial it into the module directly instead of using a Knob control that would let the user adjust the meter scale factor. Finally, the signal is applied to a "C Smooth" module which has, in effect, a "speed control" that we also set manually to make the meter "movement" look the same as that on the physical Omnipressor. As mentioned above, the meter module also has a scale and offset control built in, so it would be possible to eliminate the Constant, Multiply, and Adder module from the meter circuit if we wanted to. We will be using the –30 to +30 signal later, so we'll leave it alone here. When optimizing presets it is a good idea to look around to see which modules can be telescoped to free valuable resources.

Back to the gain control

Reverting to the level signal from the ABS module, we now want to combine it with the Threshold control so that it is "zero" when the input audio is at the threshold. In a manner the inverse of how we handled the metering circuit, we *divide* the large range of the Threshold knob by an appropriate number so that it is commensurate with the lower level representing the signal level. We then apply this scaled signal to an offset with a "C Adder." To simplify things just a bit, we don't bother using a "Constant" module but rather dial the offset right into the Adder input. Since the offset-and-scaled Threshold is a slow-moving control signal and we need to subtract the fast level signal, we take the Adder output and use a Control to Audio module followed by an audio-speed Subtract module. The output from the Subtract module, given the right constants and scale factors, should now be zero when the signal input is at the threshold level.

The final steps before using this signal to control the Omnipressor gain are to adjust its polarity (to compress or expand), it's gain (gentle compression/expansion or dynamic reversal/gating) and its bounds (wild gain swings or just a few dB.) Polarity and gain first:

To smoothly adjust polarity and gain we use the "Scale" module. This multiplies a fast (audio) input by the slower moving control input from the threshold knob. As the threshold knob varies from +1 through 0 to –1, the output from the module varies from full in-phase level to zero to full but out-of-phase level. Because it is desirable to have a finer control of the gain control law in the "gentle compression/expansion" region, a "taper" is applied to the Function control. Here's another example of the analog/digital complexity dichotomy: In the Omnipressor hardware, both ends of a linear potentiometer are driven with low-impedance, inverted signals, and the control signal is taken from the wiper. Simply by connecting a carefully-selected resistor from the wiper to ground we obtained a parabolic control adjustment and achieved a good control feel and calibration. After playing around with this problem in the digital emulation, I settled on using two multiplier modules connected together to cube the signal. Squaring the signal might also have sufficed, but this would have eliminated the necessary negative values from the output. The output from the Scale module, with the proper polarity and "control feel" is now available to be restricted by the two Limit controls.

Since we wanted to calibrate the controls as on the Omnipressor (0 to 30), we have another situation where a signal level and a control level are incommensurate. Two Divide modules, one for each Limit control are added to fit the range of the adjustment to the range of the signal. Before the Bound module is a Gain module whose purpose is to amplify the level signal enough to give it *more* than enough range to control the gain. This is necessary to get a greater-than-unity compression or expansion ratio. The Bound module, if it were an analog circuit, would be ideal diodes connected to variable supply rails, one negative and one positive. The signal can do whatever it desires as long as it doesn't exceed either rail voltage. The output from the Bound module is our almost-fully-conditioned gain control signal. Only two more steps to go…

Next, an Adder adds a fixed (non-user-controllable) offset to the level signal. This is an empirical adjustment so that the simulation matches the hardware. The output of the Adder is the final gain control signal after it has been acted on by all the user controls and adjustments and all the constants and scale factors necessary for the simulation to be accurate. In order to control the audio level coming out of the simulation, one final module is necessary, the EXPonential module. Recall that all of the level control signals we've been processing have been derived from the logarithm of the input, to enable us to adjust them linearly in decibels. It's now time to convert the control signal back to something that can control the audio. The EXP module takes a signal that varies over a small range (conceptually zero to one volt) and outputs an exponential curve whose value is represents a number raised to the power of the input. This output varies over many orders of magnitude, and is used as one input to the multiplier whose other input is the actual input audio signal that we will be listening to. (Remember the original audio signal? It's finally here again!)

One of the main design challenges in the hardware Omnipressor was to obtain accurate and complementary LOG and EXP functions. When realized in semiconductor circuitry the wide dynamic range of these functions makes them very temperature sensitive and quite temperamental. DSP circuitry, however, calculates them as precise complements-no drift, no errors. So there it is! A complete Omnipressor, realized digitally, within the confines of a Harmonizer DSP effects processor. There are still a few minor unexplicated items to wrap up…

Gain and Output Metering

The input meter circuit was described as part of the above section. Two additional "meters" show the Omnipressor gain and the output level. The gain is controlled by the signal that comes out of the Bound module. Since we went to some trouble to make this level "zero" when the gain through the unit was unity, this is one case in which an offset doesn't have to be added to make the Gain meter read correctly. We do need to use a multiplier to get the scale factor correct. This multiplier should be (and is) very close to the reciprocal of the divider that we applied after the two Limit controls. The slight difference was again determined experimentally from watching the real meter and the simulated one.

The Output of the Omnipressor is easy to meter. The output is simply the sum of the input, the gain, and the Output Cal control. Therefore we make a three-input adder by adding a third input to the standard adder module, connect the three signals to it, and connect the adder output to the smoothing circuit and thence to the meter itself. Since the signals have been previously scaled and offset to make the Input and Gain read correctly, no further adjustment is necessary for the Output meter.

Menu and Meter displays

In order to display multiple items (meters, adjustments, etc.) on the Harmonizer display screen, you have to connect the "display object" from each module to the display itself. For more control over how things are displayed, you can use a "Menupage" object that collects the display objects from several modules and puts them on the screen in the desired manner. This is explained in detail in the program instructions, of course, and needn't be discussed here. I decided to create three menus, to be selected by the Harmonizer softkeys: One contains all the controls, one contains all three meters, and a third combines both metering and the most important controls, Function and Threshold. VSIGfile allows you to create "repeating fields" which, for the Menupage, are the display inputs. For each Menupage, a softkey is assigned in the Harmonizer display area. So if you press the leftmost softkey, you get the MAIN menu which has the meters and controls. The middle-left softkey displays the meters only; the middle-right gives access to all the controls but no metering.

Since the standard Meter module (which looks like a traditional meter with a moving pointer) consumes one fourth of the display, placing three on the display leaves no room for the Knob controls, which require one half the width of the display themselves. The Hmonitor module provides the same information as a Meter module but requires only one of four possible horizontal lines on one half of the display. These modules were used to permit both metering and Knobs to occupy the same screen.

Improvements (now and prospective)

A lot of effort was taken to create a preset that would behave as much like the Omnipressor as possible. With a bit more effort I could have added a bit of noise and distortion to make it even more realistic, but that is left as an exercise for the masochist. In addition to adding full metering, there was one addition I couldn't resist. When the original Omnipressor was designed, a delay line with transparent quality was an awesome and awesomely expensive thing. Adding a Delay module to the Omnipressor simulation is both

trivial and free. One problem with compressors is that, even with an "zero" attack time, nasty-sounding transients can occur when the input is hit with a fast attack. By delaying the signal path, even for a fraction of a millisecond, the compressor attack can precede the arrival of the signal! Problem solved. I put a tiny delay in the signal path. You don't like it? YOU take it out.

The Function control, in this preset, gives a number that corresponds to the compression/expansion ratio. The original Omnipressor was calibrated less accurately but more usefully with approximate compression/expansion ratios. Using the "Textknob" feature it would be easy to make a stepped or interpolated control that would do the same thing as the Function knob and give a description of the function on the display as well.

Several application notes in the original Omnipressor manual show how multiple units can be connected for stereo or quad compression. The Harmonizer has more than enough DSP power to emulate four cross-connected Omnipressors. This preset can be enhanced in other ways; it can also be deconstructed. The main feature of the Omnipressor is its ability to control and to reverse dynamics. I hope the explanation has been clear enough to allow you to extract this feature from the morass of controls and to incorporate it in other presets that you develop yourself.

Supplementary Material

A scanned version of the original Omnipressor manual is available on the web. VSIGfile is also available, along with appropriate help and tutorial information. If you have access to an actual Eventide DSP4000 family or Orville Harmonizer effects processor you can use VSIGfile to load the preset and try it out. Two copies of the preset are available: OMNIPRES.SIG is the final version shipped in the Orville model Harmonizer and has an info page. OMNIORIG.SIF is the version I created for this article and is easier to use as an example since the graphical layout is preserved.

Hopefully I have accomplished the first two of my aims in writing this, so that you now want to go out and create vintage or modern presets yourself. I know I have achieved the third aim-I now have a digital Omnipressor to play with!