Patchman

You are facing a maze of twisty little passages, all different... .

1 Finally. Here we are. This is where it all begins to come together. This is what, in the original 2600 manual, was chapter five: a summary of patch templates, methods, and fun things to do.

In one way this is a lot easier today: with its unlimited patch storage capacity, the timewARP2600 can do without lots of detailed and tedious diagrams. Instead of having to set up every new patch starting from a bare front panel, you can load it from storage. And you'll know that the patch you loaded is exactly the one we're commenting on in this text.

But there's a downside to this: there's a temptation to just load patches, and not modify them or learn anything from them. Two decades of digital synthesizers and samplers have turned many musicians completely away from the real flexibility of analog synthesis.

With the TimewARP2600, you can step out of that crowd and free yourself from that way of thinking.

Look at it this way. Playing with analog modules involves

- setting up signal and control paths
- adjusting continuously-variable parameter settings

1.2 Now the signal and control paths set the logic of a patch; that is, they determine what modules are part of the patch and which ones aren't, and they determine some of the major features of the patch. I think of the configuration as a determining a landscape.

1.3 The parameter settings – all of the continuously variable sliders – provide you with a way to travel around in that landscape.

2 The first part of this patch manual is a tutorial on the fundamentals of audio synthesis. In order to get the most out of each of these, you'll want to identify – logically – what parameters each configuration gives you, and experiment thoroughly and systematically with those parameters.

Take note of the landscape, and then travel around in it.

2.1 Silence P_001

This patch removes all patch cords and shuts down all sliders. It produces no output signal. It's the starting line for building new configurations.

2.2 Noise – out P_002

The first source you want to listen to is noise. You may load the patch, or simply connect a patchcord from the noise generator output to one of the mixer inputs.

Here are some things to listen for and take note of:

Open the output signal level with the right-hand slider. Vary the spectral balance with the left-hand slider.

2.3 Noise – low pass filter P_003

Since a noise spectrum is distributed across the whole range of hearing, you'll be able to hear immediately if any part of that spectrum is modified. So noise is actually more useful than a pitched sound for learning about filters. (This is also why stereo systems are tested with noise signals as well as with music.)

As loaded, this patch starts with the filter wide open, i.e. the cutoff frequency is up around 10KHz, so the output signal is almost unchanged from the input. (Confirm this by using the two mixer sliders to compare the direct noise signal with the filtered one. You should be able to hear the effect of the filter – even when it's wide open – as a slight loss at the high end.) Now listen to the filtered signal as you slowly lower the cutoff frequency.

Note that you can always hear changes in the noise spectrum – from white to pink to red – no matter where the filter cutoff frequency is set. The VCF is a 24dB cutoff slope, but the differences in noise color are much shallower, at 6dB/octave.

2.4 Noise – filter (with voltage control of cutoff frequency) P_004a

Here we'll use a low-frequency sine wave to sweep the filter cutoff frequency up and down. As loaded, the sine wave is slow – on a period of maybe one cycle every five seconds – and the depth of modulation is about an octave peak-to-peak.

Even though this is still a very simple configuration, it has more performing variables. They are:

2.4.1 the baseline (before modulation) cutoff frequency of the filter

2.4.2 the filter resonance level (manually set, not voltage controllable)

2.4.3 the spectral balance of the noise

2.4.4 the modulating sine wave frequency

2.4.5 the modulating sine wave depth (this is set by the FM input slider at the filter)

2.4.6 Also, there are attenuators in the audio signal path at three different points in the configuration (the noise generator output, the filter audio input, and the mixer input). But all of them affect the signal identically (you should fiddle with each one to prove to yourself this is true).

2.4.6.1 In general, attenuators in the audio signal path should always be set wide open, except when you need to set the relative levels of signals that you are mixing.

2.4.6.2 Control signal levels, on the other hand, determine depth of modulation, and will usually need to be set with great precision, very carefully.

For example (P_004b), substitute a square wave for the sine – using a patchcord – and note that even very tiny changes in the modulation depth are very audible.

2.5 Noise – high-q filter P_005

The filter, in this patch, is tweaked to a very high resonance, just short of the point at which it would oscillate.

You should learn to do this yourself (tweak it, I mean, not oscillate). Set the filter Fc around 1KHz, start from low-Q, and move the resonance slider slowly to the right until the filter screams at you like a PA system in feedback. Then back off on the resonance until the filter stops screeching.

Now open the filter's noise input. It'll probably remind of you of wind in the trees; that's what most people – myself included – first think of. The filter Fc plays a heavy role in this; fiddle around – without changing the resonance. You'll get reminders of many different naturally occurring noises.

Note that even in this situation there is a very audible difference between white noise and pink and red. Confirm this for yourself with the NG spectral balance slider.

2.6 Sawtooth harmonics – highQ filter P_006

This is identical to the preceding configuration except that you've substituted a sawtooth for the noise signal. With the filter highly resonant, it peaks sharply at each harmonic in the spectrum. This should give you a pretty solid idea of what we mean when we say that noise has a distributed spectrum, while periodic signals such as this sawtooth have discrete spectral components.

Sweep the filter cutoff frequency slowly up from left to right through the frequency range. As loaded, the sawtooth is set to around 250Hz – approximately middle C in the pitch range – and so you'll encounter a harmonic every 250Hz above that. The first note you hear will be the fundamental – the base frequency of the spectrum. The next one will be one octave up, at 500Hz, then a musical fifth up from there at 750Hz, and so on.

2.6.1 You can, of course, automate this filter sweep, using a slow (subaudio) sawtooth to control the filter Fc. (P_006b) Patch a sawtooth – use VCO1, for example – into one of the filter FM (control) inputs, switch VCO1 to LF range, and open the slider for that filter input (how far? Whatever seems to work best after a little experimentation.)

2.7 Spectral analysis of other periodic waveforms: square (P_007a)

Just in case you got lost with the preceding configuration, we've provided one that uses a 50% pulse (VCO3 square wave) for its audio, and has the sawtooth control signal already patched into a filter control input.

Notice that the square wave spectrum is missing all of the even-numbered members of the harmonic series. From 250Hz, the next spectral component you hear is the one at 750Hz, an octave and a fifth above the fundamental. And then at 1250Hz, two octaves and a third up.

2.7.1 Triangle (P_007b)

Triangle waves have the same spectral components, but their amplitude falls off more rapidly with increasing frequency. Patch in a triangle instead of the square wave and confirm this for yourself.

2.7.2 Pulse (P_007c)

Pulse waves have spectral null points at multiples of the pulsewidth as a fraction. For example, a pulse wave with a 1/3 duty cycle will be missing every third member of the harmonic series. With a ¼ duty cycle, it will be without every fourth member. You can use the present patch to confirm this; roughly, your strategy might be to tune the Fc to the third harmonic of a sawtooth; then switch to the pulse output of the same oscillator, and then adjust the pulse width so as to minimize the filter output level.

2.8 Noise – VCA [linear and exponential control] P_008a,b

Here's a chance to hear the effective difference between linear changes in amplitude, and exponential changes. You've got a slow sawtooth signal coming in to both VCA gain-control inputs. So you can hear what linear control sounds like, and then what exponential control sounds like, just by opening the control sliders alternately.

Each cycle of the controlling sawtooth drives the VCA gain smoothly from minimum to maximum. But because of how signal amplitude relates to perceived loudness, the linear-response control input will give you most of the volume gain in the early part of the cycle, while the exponential control input will spread the volume gain equally through the entire cycle. Try it.

2.9 Noise – vca [AR envelope] P_009

Up to this point, you've been listening to essentially steady-state signals, getting to know their spectral characteristics. You've got a pretty clear idea of what we mean when we talk about “distributed spectra” of noise, and “discrete spectra” of periodic waveforms.

Now you will begin learning to carve these continuous signals – both noise and pitched tones – into auditory events.

This configuration introduces voltage control of the VCA by the Attack/Release envelope generator. We'll feed some noise into the VCA; but we'll leave the VCA initial gain completely closed, and use an envelope signal to control the VCA gain. The result is that we'll get an audio output from the VCA only when we provide a gate signal to the envelope generator.

Click on the little red gate button in the envelope-generators module. While the button is down, you've got noise. When the button-press ends, so does the sound.

2.9.1 Things to play with and take note of:

2.9.1.1 The atttack time and release time sliders on the AR generator govern the corresponding parameters of the generator output, which – in this patch – governs the moment-by-moment gain of the VCA

2.9.1.2 Setting the VCA initial gain to anything but minimum allows signal to “leak” through to the output. In some patches that might be the behavior you want, but it's more common to not want that.

2.9.1.3 The two gain-control inputs to the VCA, because they have different reponse curves (one linear, the other exponential) produce different auditory event shapes even from the same envelope. Prove this for yourself with P_009b, by feeding the AR envelope output alternately to the left (linear) input and then the right (exponential) input.

2.9.1.4 (P_009c) Now use the gate-source selector switch – in its lower position – to patch in the Electronic Clock as a gate source. That, at least, is the default connection.

2.9.1.5 (P_009d) Switch VCO1 to LF range, and patch the VCO1 square wave in as the gate source; now the rate of audio event generation is governed by the VCO1 frequency slider.

2.9.1.6 Notice how the gating frequency (from VCO1) interacts with your attack and release times; uptempo gating, with long AR times, gives almost continuous (and very legato) output, while short AR times at the same tempo give very distinct staccato events.

2.10 Noise – VCA [adsr envelope] P_010a

You have two further parameters to experiment with here. Listen for:

2.10.1 Setting the Sustain level to maximum gives you what is effectively an AR generator. (With maximum Sustain, the Decay setting doesn't matter.)

2.10.2 Set Sustain to zero, Decay to anything shorter than Release, and experiment with the differing results you get from short gates and long gates.

2.10.3 (P_010b) Use an oscillator gate source – make it VCO3 this time – and play with the pulse width. This lets you vary the duration of the gates without changing their frequency. This is an absolutely simple, bare-bones patch, but it has a lot of variables:

2.10.3.1 Envelope Generator parameters:

2.10.3.1.1 Attack time

2.10.3.1.2 Initial Decay time

2.10.3.1.3 Sustain Level

2.10.3.1.4 Final Release time

2.10.3.2 VCO3 parameters

2.10.3.2.1 the gating frequency (VCO3 frequency)

2.10.3.2.2 gate duration (VCO3 pulse width)

2.10.3.3 Noise Generator spectral balance

Strictly speaking, you've also got the VCA initial gain, and the gain modulation depth but mostly you will leave these at their minimum and maximum levels respectively.

You may lose yourself in playing among these seven parameters. Almost certainly, you will find dozens or hundreds of configurations that remind you of everyday sounds; most people do. Save these, if you like. You never know when they might come in handy.

2.11 Noise – filter[adsr[gates]] P_011

Here we are really just substituting the VCF for the VCA. If you saved any patches from the preceding exercise, call them back and reroute the noise signal through the filter instead of through the VCA. Thanks to the TimewARP2600 default signal routing, you can do this by merely flicking a few sliders around.

With this signal routing, the envelope signal is governing your noise spectrum, not just its amplitude. In nature, most things that vibrate when you bang on them lose their high-frequency energy quicker than their lower frequencies; and that produces the same sort of spectral behavior that the VCF gives you in this configuration. Once again, you will almost certainly hear many sounds here that remind you of naturally-occuring events. Save them if you like.

Things to take note of:

2.11.1 like the VCA, the VCF can shut down a signal completely, to open up only under control of an envelope or other signal. This can be useful in patches where you want to do something else with the VCA.

There is one small difference, but it is important in some patches: unlike the VCA, the VCF is a frequency-dependent function, and just because you're not hearing anything coming out of it doesn't mean that no signal is passing through. A spectrum that has heavy subsonic content (this might be intentional but it can happen by accident too) may have all of its audio content removed but not the subsonic stuff.

2.11.2 (P_011b) To hear this difference in its rawest, most basic form, use a low-frequency sawtooth again, to sweep both the VCF and the VCA, and use the two mixer inputs to listen to the VCF and VCA alternately. You used a similar configuration to examine the difference between the two gain-control inputs on the VCA.

2.11.3 (P_011c) A little resonance goes a long way. A very long way.

One of the commonest cliches of audio synthesis is the resonant filter, swept by an adsr envelope. You'll go through a phase where it sounds wicked cool, and then – I hope – you'll get over it. The present patch is your first official opportunity to discover this cliché. You might as well get on with it. But don't say I didn't warn you.

2.12 A basic keyboard configuration P_012

This is a bare-bones version of what has been the prevailing audio-synthesis model for three decades of commercial keyboard instruments. Essentially, you take one or more oscillators, mix them, feed them through a VCF and a VCA in that order, and provide some envelopes to control the VCF and VCA. The VCO's are controlled by a keyboard signal (for setting their pitch), and the envelopes by a keyboard gate. So, when you press a key, the synthesizer modules create an audio event at some appropriate pitch.

Now this is not by any means the only useful way to configure your TimewARP2600 modules. But it is certainly one useful way. Here in the tutorial, you're going to spend some time with it, becoming familiar with every nook and cranny of the landscape this configuration creates for you.

2.12.1 The complete set of variables that this module configuration makes available to you:

2.12.1.1 the selection and relative strengths (mixing) of oscillator signals

2.12.1.2 the base cutoff frequency of the filter (before any modulation by an envelope)

2.12.1.3 the filter resonance

2.12.1.4 the depth of Fc modulation by an envelope signal

2.12.1.5 the adsr or ar parameters of the vcf-modulating envelope

2.12.1.6 the initial VCA gain (before any modulation by an envelope)

2.12.1.7 the depth of modulation by an envelope signal

2.12.1.8 the adsr or ar parameters of the vca-modulating envelope

This amounts to at least 14 continuously-variable parameters, even without counting things like pulse-width or pwm (supposing that one of the oscillator signals is a pulse), or any keyboard parameters such as portamento, trigger mode, or depth of Fc modulation by the keyboard pitch-control signal. (This latter, by the way, is an extremely important parameter in most keyboard patches. We'll get to it in due time.)

2.12.2 These 14+ variables give you access to a lot of territory. It's a huge landscape of hills and valleys and little hidden places that maybe nobody else knows about. We'll only be able to give you an overview. Here are some of the places you'll want to visit:

2.12.2.1 finding the right spectrum

One very simple, basic way to use this standard voice configuration is to build some sort of a steady-state spectrum from your input waveforms – mixing or modulation them together as you please – and then chopping that into separate events with some combination of envelopes on the VCF cutoff frequency and the VCA gain.

The following patches illustrate differ only in the steady-state spectrum that they are feeding into the VCF. All other parameters – envelope settings, modulation depth, and so on – remain constant throughout the entire set of patches.

2.12.2.1.1 P_013a through P_013g mixing and tuning

P_013a mixes three saws, at three different pitches. The bottom one, from VCO1, is inverted in the mix.

P_013b mistunes the topmost oscillator (VCO3)

P_013c drops the middle oscillator, so you get just two signals two octaves or so apart. The upper one would generally be at a reduced level; things seem to sound better that way, except when they don't.

P_013d adds some FM of the VCF, from VCO2. VCO2 is not in the audio signal mix.

P_013e uses two uninverted saws, from VCO1 and VCO2, an octave apart.

P_013f is two saws an octave and a fifth apart.

P_013g is three saws, an octave and two octaves apart.

2.12.2.1.2 P_014a through P_014f illustrate just a couple of instances of ring modulation. The spectra you generate this way consist of sidebands: sum-and-difference frequencies from the components of the signals fed into the ring modulator. Therefore, what you hear depends on the interval between the source oscillators.

For some real fun, add a small amount of ADSR modulation to either one of the oscillators.

2.12.2.1.4 P_015a through P_015g frequency modulation

In most of these patches, you are listening to just a single oscillator; but the spectrum you hear is complex and fascinating because it consists of sidebands generated by frequency modulation. If the modulating oscillator (usually VCO2 in these patches) is not tuned to a simple harmonic relationship with the modulated oscillator, the sidebands will be enharmonic – they don't form a harmonic series. Instead, they may remind you of a gong, tubular bells, or some other metallic percussion instrument.

2.12.2.2 signal mixing

A couple of things to bear in mind, always:

mixing raw oscillator waveforms is not at all like mixing recorded or sampled sounds, from acoustical instruments, at a recording console.

mixing geometrically perfect waveforms can make them disappear;

P_016

- mixing perfect waveforms can transform them into other waveforms;

P_017

- mixing filtered or modulated waveforms may lead to goofy results; but if you think carefully enough, the results will be predictable enough so that your listeners will be surprised but you won't be.

P_018

2.12.2.3 easy on the filter, boys

This is an often overlooked family of patches. Since the VCA can chop your audio into separate events, the VCF doesn't need to; and that allows you to do all sorts of things with the VCF that you otherwise couldn't. Here's one simple example:

P_019a,b

2.12.2.4 a little noise never hurt anybody

In keyboard patches – where mostly you intend to generate pitched events for melodic use – noise can still have many uses. Specifically, there are opportunities to

use a little noise to modulate your oscillators

P_020a,b

or to modulate the VCF cutoff frequency

P_021a,b

You might also find it useful or interesting to sample your noise (triggered either directly by the keyboard or by a free-running LFO) and apply a minuscule amount of the resulting random steps to control any of the above points. In either case, you might want to filter the sampled noise signal using your lag processor.

P_022a,b

2.12.2.5 cooperating envelopes

You can use a longer VCA risetime to mask a shorter VCF risetime or – more interesting – short risetime on a pitch or modulation envelope. In the latter situation, combined with a legato keyboard play, the final-release state can become – if you want it to – a kind of all-hell-breaks-loose situation.

P_023

2.12.2.6 voicing an instrument

Whatever your basic parameter settings in a keyboard patch, the (imaginary) instrument that you're building is not immune from the general facts about human hearing: the frequency-dependent sensitivity, the absolute limits in both dynamics and frequency range, the insensitivity to phase difference, and so on. Consequently, once you've centered in on the kind of sound you want, you'll want to tune it across the keyboard and pitch range in which you want to use it. This is “voicing” the instrument; all instrument builders do this.

P_024

2.12.2.7 playing technique

Finally, the spectral and amplitude contours you arrive at will interact with your keyboard playing technique. For example, relatively long attack/decay/release parameters may turn detached gates into a legato melodic line:

P_025

Another example: zero sustain combined with a final release that's longer than the initial decay, can make a useful “marimba” keyboard technique:

P_026

envelope control of pulse width makes good bass patches. Because of the physics involved in picking a string, you generally want to set the baseline width at 50%, and use the adsr envelope to offset this to 90% or so at the beginning of the audio event.

P_027

3 Inventing a patch

The default trigger source for the S&H module is the Internal Clock. But that can get pretty boring. If we used a pulse wave instead, at something other than 50% duty cycle, we might be able to get triggers in rhythmic pairs.

We'll start with something simple here, and in each patch we'll add something simple to what we already have. That's how patches grow. Have fun with this one.

(P_3_01) Triggering the S&H from a subaudio VCO3 pulse starts us off. We're using the pulse from VCO3 to trigger the S&H. All of the signal attenuators in the noise patch are wide open. To monitor the output from the S&H, we've routed it to one of the FM inputs of VCO2, which we then listen to through the VCF.

(P_3_02) We shift the pulse width from 50% to something more interesting.

(P_3_03) We select the Internal Clock as gate source for the envelopes, and reconfigure the VCF to be gated from the ADSR envelope. Oops: what we meant to do was get our envelope synchronized with new pitches from VCO2; but instead, our envelopes are gated from the Internal Clock.

(P_3_04) We fix that by rerouting VCO3 through the multiple, so that we can send it to both the S&H trigger input, and the envelope-generator gating input.

(P_3_05) Listening to this, we realize that the S&H is triggering on both the rising and the falling edges of the pulse. We're getting two pitch intervals per auditory event; the second one is less accented, because it comes just at the final release cycle. So we experiment with changing the length of the VCO3 pulse; if we set it to about 70%, the weak beat sounds more like a lead-in to the strong beat.

Have some fun playing with the ADSR envelope parameters here. A lot of different effects arise out of different a,d,s,r settings.

(P_3_06) Of course, we needn't be sampling a random noise signal. If we sample a periodic waveform, we'll get regular patterns in the S&H output. VCO1 isn't busy; let's use the sawtooth from there. But now why do the pitches go so high?

(P_3_07) Oh, because the TimesARP2600 sawtooth waves are positive-going rather than balanced across zero. We can fix that with the VP1, using the -10vV default at the first slider. With this setup, we've inverted the VCO1 sawtooth and added about 5vV to it, so that it's balanced across zero, just like the noise signal. So the sample levels output from the S&H module are also balanced, and therefore the pitch range of VCO2, under the control of these sampled levels, is balanced both above and below whatever the baseline pitch was that we began with.

(P_3_08) Could we get some kind of a rhythm accompaniment? Let's shut down the VCF signal path for a while and experiment with the Ring Modulator. The AC coupling at the Ring Modulator inputs (which is enabled when you switch to “audio” mode) works like a high-pass filter on low-frequency signals coming in. They can't “hold their level”, but instead always leak away back to zero.

In fact, they decay just like little envelopes. We can take advantage of that fact by running some noise into one input, and our S&H levels into the other.

(P_3_09) Here we mix the melodic sequence back into the rhythmic one.

(P_3_10) We add some further rhythmic interest, by partially gating the VCA from the Internal Clock. The general idea is that, since the IC is free-running and not synchronized with the primary gating from VCO3, it can provide an independent series of pulses which we will use to make the VCA alternately louder and softer. How successful this is depends strongly on the tempo of the Internal Clock compared to VCO3. Try changing this from the loaded patch. P_3_10a is a simple variation.

(P_3_11) With no change in the rhythmic fundamentals of the patch, we try for some timbral variation. Our audio signal is no longer the VCO2 pulse; we have substituted VCO2 sine, with FM from the VCO1 sine. Because we haven't tuned the two S&H FM inputs to these oscillators, they aren't moving in perfect parallel; and therefore the spectrum varies widely from one sample step to the next. P_3_11a is a variation in the baseline oscillator tuning, to raise the carrier signal frequency (VCO2).

(P_3_12) Here we go back to sampling the periodic output of VCO1 instead of noise. Notice that there is feedback from the S&H to one of the FM inputs at VCO1. The sequence of sampled levels is not random, but it's not really predictable either.

(P_3_13) Here we are set up for keyboard control of the event tempo. Going up an octave will double the tempo; going up a fifth yields a triplet, and an octave and fifth a double-tempo triplet. These are the frequency ratios of the harmonic series.

(P_3_14) Basically similar to the preceding, with a slight adjustment of the envelope parameters to simplify the event contours. Same keyboard control.

(P_3_15) Reroute the Internal Clock through the Lag Processor, so as to avoid the uselessly sharp “attack” at the VCA gain control.

3.1 Finally, let's take a break and summarize for ourselves what's going on here, and what control we have over it:

3.1.1 Your keyboard governs the event tempo. Each octave doubles the tempo; fifths go up by a 3:2 ratio (triplets), fourths by a 4:3, and so on, just as you would expect from the frequencies that make up a harmonic series.

3.1.2 The ADSR sustain level governs whether or not you get the little upbeat note just before the main beat.

3.1.3 The Internal Clock governs – well, there's no real name for this – the “continuity” of the noise rhythm. This is easier to hear than it is to explain; with the IC at a very slow rate, the noise events are silenced during the low part of the IC output cycle. At its fastest rate, because we're using the lag processor as a kind of envelope generator with long attack/decay, we get effectively continuous output from the VCA – there's no audible contribution from the IC at all. (Interestingly, at this maximum IC rate, cutting the lag-processor time to minimum provides a little AM buzz in the VCA processing of the noise events.)

3.1.4 Fiddling with the source oscillator signal routings can of course give you timbral variety. We've already experimented with VCO2 pulse, and with VCO2 sine under FM from the VCO1 sine, and with mixing VCO1 into the audio path.

3.1.5 Sampling VCO1 (through our offset-compensation setup at VP1) yields various repetitive patterns in the pitch sequence, depending on the ratio of VCO1 – the sample source – to VCO3, the triggering source. As long as there's no feedback into VCO1 from the S&H output, these patterns will be controllable and predictable. So of course, in a performance, you would open the feedback slider to destroy a pattern, and close it down again to restore the pattern. What's important to understand here is that you need only one single patchcord (at the input to the S&H module) to switch between random noise and periodic patterns.

3.1.6 VCO3 pulse width adjusts the “swing” of the rhythm.

3.1.7 And, of course, VCO3 frequency is the fundamental tempo. Note that in order to get control of this with the keyboard, we had to patch an explicit connection; normally, switching an oscillator to subaudio operation disconnects keyboard control.

3.1.8 Noise-Generator spectral balance has the usual effect.

3.1.9 The S&H depth slider at VCO2 governs the pitch range of the “tune” that's being played on VCO2. Note that, when we decided to set up keyboard control of the tempo, we disconnected keyboard pitch-control of VCO1 and VCO2. That's a matter of taste.

4 Some Instruments

Before the era of digital sampling, it was a big deal to be able to emulate natural acoustical instruments with an analog patch. Today, of course, most of us would simply reach for a sampler product, either in hardware or in software.

But there are still good reasons for investigating these make-believe natural instruments, even though they are like Bambi compared to a real fawn. All instruments, even imaginary ones, eventually must tickle the same ears and the same listening activity: human ones. So they are all subject to the same constraints.

But probably the most important and intrinsically interesting reason is psychological. (It's one that the Disney animators have studied for well over half a century now.) Given that you're creating a Bambi – that is, a fake baby deer – how do you show people it's a fake deer and not a collie? In other words, what are the clues that tell your listeners they're hearing “something like” a flute rather than “something like” a trumpet?

4.1 Some flutes.

P_4_01 is possibly the simplest musically useful (for keyboard performance) patch possible. Pure sines don't usually make good instrument sources because they don't have good pitch definition – they don't have any overtones. Sawtooth and pulse waves, as musical instrument sources, have too many overtones, so they absolutely must be filtered. (Of course this is a matter of taste on which you're welcome to disagree with me and with each other.)

Triangles have overtones, but the spectral falloff rate is much steeper than for saw or pulse waves; so it's possible to listen to a raw triangle without any pain, and in fact with a good deal of pleasure.

P_4_02 routes our triangle through the VCF so we can take control of the spectrum if we want to. And we want to.

As stored, this patch uses no keyboard control of the VCF. That's a natural place to start from in voicing your instrument. “Voicing” the instrument is adjusting, in an integrated way...

- the initial spectrum
- the filter Fc
- the depth of keyboard control of Fc, and
- the Fc envelope depth (if any)

...so that your instrument “sounds right” across its intended playing range. To give you an idea of what you're trying to do, let's listen to a few versions of this flute that are badly voiced:

P_4_03 is too dull at the top end. The VCF is cutting out almost all of the spectrum.

P_4_04 is too bright. We have opened up the keyboard control of Fc, and the filter is opening up in parallel with the changing source frequency. The result is that as the source spectrum climbs up into the region of greatest sensitivity for human hearing (around 2KHz), the sound gets shriller and brighter compared to what it sounds like at middle C and below.

P_4_05 is just about right for keyboard balance. Once you've got this, for any particular patch, you can continue to adjust the overall brightness using the VCF initial Fc slider, occasionally bringing the keyboard control level up or down a little according to what your ears tell you.

Since triangles have only odd-numbered members of the harmonic series, they usually remind people more of classical wooden flutes (recorders) than of metal transverse flutes. If that's what you're after, you should start with a sawtooth rather than a triangle.

P_4_06 does that. Setting the initial Fc is a delicate balance; too high, and your flute transforms itself into a french horn; too low, and you can't hear it at all.

There are other problems. One of them is common to a lot of synthesized imitations: we have too much fundamental. It makes flutes, brasses, and strings sounds tubby and tanky, not light and bouyant as they should.

The problem begins with the raw spectral characteristics of sawtooth and square waves. There are a lot of ways to alter the spectral balance to de-emphasize the bottom end. Here are the ones that are available to you on the TimewARP2600:

- cobble together a 6dB/octave highpass from the inverted lag processor;
- use a narrow pulse instead of a sawtooth for your source spectrum; a narrow pulse, up to its first spectral null, has an essentially flat spectrum rather than a -6dB/octave slope like a sawtooth.
- Use feedback around a sine to do direct waveshaping, and then route that through the highpass cobble; or even through a 24dB/octave built from the inverted VCF. The thing is, in order for this to work, you have to be starting from a spectrum that doesn't have the high-frequency buzz that a sawtooth or pulse does.

P_4_07 uses another, simpler method. It depends on the fact that we are not driving the filter Fc with an envelope. What we've done is just to increase the filter resonance by a little. This attenuates frequencies below Fc. Play with it yourself to get familiar with the effect and with its usefulness. You may find the patch tipping over into oboe country.

P_4_08 uses a highpass built from the inverted lag processor.

4.2 Some brasses

What's critical in brasses is getting the right ADSR parameters, and a surprisingly shallow depth of filter modulation. People often err towards too great a depth of Fc modulation by the envelope.

brass_01 is a baseline for further work: we use a sawtooth and a small depth of ADSR. There is a tiny amount of resonance in the VCF, to attenuate the bottom end of the spectrum.

Brass_02 uses the inverted lag processor as a highpass filter.

Brass_03 mixes, experimentally, an inverted narrow pulse. The lag processor here is NOT functioning as a highpass filter, since the two signals being mixed together at the VCF inputs were not identical to begin with.

Brass_04 uses raw waveshaping, in the form of FM feedback around VCO3. The spectrum resembles that of a pulse wave up to its first null; the difference here is that, unlike the pulse, the shaped sine has no spectral components beyond that first null. Since it doesn't have a fuzz of upper harmonics, it really doesn't need to be shaved with a lowpass filter. Sounds pretty good, to my ears.

Brass_05 makes a rhetorical point: you have to be careful with this FM feedback. At extreme modulation depths, it can drive you off-pitch. Throughout most of its range, brass_05 is completely yummy, warm, crunchy, just what a brass should be; but up at the top of its range it goes radically out of tune.

4.3 Some reeds

sax_01 is an accidental discovery. I can see what's going on, but not why it sounds like a tenor sax. It's exactly like brass_05, with the addition of a narrow pulse off the same VCO3 that's doing the shaped sine.

What's been difficult about saxophone is that the natural instrument is extraordinarily flexible. It morphs, under the control of a good performer, through a very wide range of timbres, attacks, and event contours. No subtractive audio synthesis can begin to emulate this range; the trick is, just as with Bambi, to get enough of the details right so that your audience will supply the rest out of their own imagination.

Double_reed_01 is a simple, baseline member of the oboe/bassoon family. It's a narrow pulse, and a mildly resonant filter at a constant cutoff frequency. If your studio setup includes some parametric EQ, you can use that as a part of your synthesis patches in contexts like this. After all, the VCF here is not varying at all – there's nothing “voltage-controlled” about what it's doing for you.

Double_reed_02 is exactly like 01, but we've inserted the highpass construction into the patch. To my ears, this sounds a little better.

Double_reed_03 does a little voicing trick: using keyboard voltage to shift the pulse-width from narrow to wide as you go from left to right across the playable pitch range of the instrument.

Clarinet_01 and 02 are variations on the double_reeds, with the pulse width centered around 50% instead of 10%.

4.4 Plucked-string patches

Here things get difficult but also exciting. Difficult, because all acoustical string instruments, whether bowed or plucked, have big resonators. The body of a guitar, or a violin, is effectively an extremely complex filter/amplifier for the thin and colorless sound of the stretched strings that are bowed or plucked to generate your source signals. Neither the TimewARP2600 nor any other analog synthesizer has the filtering resources to emulate that wooden resonator. Not even close. For that reason, emulation of string instruments – even the Bambi imitations – is pretty much a no-go unless you have an external parametric EQ or something capable of playing the role of the instrument body.

On the other hand, almost any envelope with a sharp attack and a longer decay can persuade my ears that I'm hearing something plucked or percussive, even if it's only an imaginary instrument. I think that this is because plucked and percussive events are intrinsically more dynamic than the relatively steady-state events we get from the wind instruments. They can be perceived as exciting, whether they resemble something we've heard before or not.

Generic_picked_01 and 02 illustrate this. They don't imitate anything in particular, but they're perfectly usable in a wide range of musical contexts, using whatever parameter variations sound good to you.

Bass_01 illustrates using pulse-width modulation to do a little physical modelling. When you pick a stretched string near one end of its length, the initial spectrum is approximately that of a pulse with a duty cycle proportional to the point where you picked the string. As the vibration decays, the point of greatest amplitude settles down to the middle of the string. We can model this (very approximately) by starting with a narrow pulse and letting it decay towards 50%.

Bass_02 is a straight variation on this, with a more pronounced envelope.

Generic_03 is a pulse, with the highpass subpatch, PWM from the ADSR. This configuration is like the basses we've been looking at, but the operating parameters are adjusted slightly for an alto/soprano performance range.

Generic_04 adds sine-wave FM to the mix going into the VCF.

Generic_05 has increased filter resonance, to cut somewhat the bottom end of the spectrum.

Generic_06 adds some modulation of the VCF cutoff frequency from VCO2. Since that's the same oscillator that's our primary audio source, the FM won't produce any sidebands of interest; but it does have a pronounced effect on the overall spectral balance.

5 Noises

Noise is the only signal that is really interesting in its own right. It is the sum of all possible periodic signals; it contains within itself all oscillators, all musical instruments, all melodies, every song that has ever been sung or symphony performed. Try listening to a raw sawtooth for some length of time – say, a half hour. You'll go mad before ten minutes are up. Now listen to noise. It's soothing. You can take a nap.

Surf_01 is a simple baseline patch, running pink noise through the VCF, with a few of the standard controls for variety.

Surf_02 has a few more tweaks, for additional interest. We're running a second independent time series from the Internal Clock, and using it to generate a small fluctuation in the VCF frequency. Also, a small S&H modulation of VCO1 makes it slightly irregular in timing.

Noises_01 constructs an unpredictable sequence of noise events from the Internal Clock and VCO3 sawtooth. The latter, inverted, works like the first two cycles of an ADSR envelope: attack/decay. Listen to how the two control signals work together.

Noises_02 is another of these. It's built on top of the 01 patch.

It can be fun, constructing unpredictable control sequences this way. I've been listening to noises_02 for about 15 minutes as I write this, and I'll swear I hear a rhythm in it.