

The Modular Synth

Overview And Ground Rules

Audio Input / Output

Oscillators

Filters

Envelopes

Graphical Envelopes

LFOs

Mix And Level

Switches

Effects

MIDI

Special

More About Esync

Connections

[Return To Main Table Of Contents](#)

Overview And Ground Rules

There's a lot to read here – don't get discouraged!

Loading The Modular Synth

Like other Pulsar Synths, the basic Modular Synth is located in the **..\Pulsar\Devices** folder. It appears in the File Browser under the name **Empty Modular.mdl** (make sure the **Mdl** button in the browser is pushed in, otherwise it *won't* appear!) and can be dragged from there into the Project window in the usual way.

The Empty Modular is a bare-bones framework suitable for creating new Modular setups from scratch. This makes it quite useful. But since it's "empty", it does need to be worked on a bit before it will even begin to make sounds.

If you're new to the Modular Synth, a highly recommended alternative is to drag in one of the **Modular Synth examples** found in **..\Pulsar\Devices\Modular Patches** (these are also .mdl files, by the way). With any of these (as with the Empty Modular as well), you'll still need to go into the Project window to make MIDI in and audio out connections to the Modular, but once you've taken care of that, you'll have a ready-to-play setup in front of you, complete with presets, which offers a concrete example of what can be done with the Modular and how to do it, and which can also serve as a starting point for your own forays into Pulsar Modular synthesis.

About The Modular Window

The Modular window is a special-purpose device surface into which Modular Synth *modules* can be dragged and then arranged and interconnected as desired (more about the modules later – *much* more). You get into this window via the usual methods for opening a device surface: open its module menu by right-clicking on the Modular device in the Project window, and then select *Open Modular Window* – or simply **double-click on this device or on the Modular Synth icon** which appears at the bottom of the screen when you load the Modular Synth into Pulsar.

The window itself is a standard movable, adjustable Pulsar window with only a few features. At the top are four buttons. Two of these are the usual ones found on every device surface. The **Preset** button displays or hides the Preset List for the current Modular setup. The small unlabeled **Minimize** button in the upper right corner shrinks the Modular window to an icon.

The **Routing** button toggles the display of cables in the Modular window on and off. Naturally you need to see the cables when modifying a Modular setup – but the cabling for a full-blown setup can get to be quite a rat's nest! When you're merely *using* a setup, you can hide the whole mess and make it much easier to see and do things. The **Save** button is discussed in the next section.

Saving Modular Synth Setups (Important!)

It's important to be aware that the Modular Synth requires a bit of special handling: **You have to save your Modular setups!** A brief explanation:

The structure of the typical Pulsar synth is stable – it's the same in every project. Pulsar projects therefore normally save only synth *references*. A given synth is loaded from the same device file whenever you load *any* project which uses it. This file can be shared by any number of projects. The synth itself doesn't need saving.

The Modular, on the other hand, has no single fixed architecture. It may contain any combination of modules, which can be cabled together in a huge variety of ways. It's likely to look quite different from one project to the next (or even within a single project, since you *can* have more than one Modular in a project). Think of having not just one but *many* unique and separate Modular synths.

This shape-shifting aspect of the Modular makes the technique of references to a single common device file unworkable. Therefore, **when you save a project which includes a Modular synth, the complete Modular setup is saved directly within the project.** This ensures that when you reload the project, the included Modular setup will be the same as it was when you saved the project.

Alternatively, the **Save** button at the top of the Modular window lets you **save a Modular setup as a *separate module (.mdl)* file**, in effect exporting it from its project. This Modular setup can then be used in other projects. Here you should be aware that: **1)** the setup thus saved retains ***no connection*** to the setup in the project – they're now entirely separate, and **2)** when you use the new copy in another project and save the project, the setup is saved again *in the new project*, and this setup is likewise now **independent of the copy in the .mdl file**. Remember the basic principle of many *unique and separate* Modulares.

In both cases, **the saved setup includes the Modular preset list**. By now, you can probably guess why: since Modular setups are so flexible, the **presets created for one particular setup are usable *only* with that setup**. You *can* export a preset list via the Preset List control panel, but this list will be usable only with the original setup or an *unmodified* copy. (Also note that the Preset List *Save Device* function does not apply to the Modular.)

The bottom line: to retain any changes you make to a Modular setup, save the project or save the setup. Otherwise, those changes are gone for good. We also heartily recommend, for all of the above reasons, that you give meaningful, descriptive or at least distinctive names to the Modular setups you save. Don't be lazy – use those **long filenames** that G— (Gates) gave you!

Adding, Removing and Copying Modules In A Setup

To **add new modules** to a setup, locate them using the File Browser in one of the module library folders (by default under `..\Pulsar\Devices\Modular` – remember that you need to have the browser *Mdl* button pushed in). Click and drag on the name of the desired module, move the cursor over the Modular window and let go. The new module appears in the window after a brief loading delay.

To **delete a module**, select it by clicking on it, and then hit the Delete key on your keyboard. All connections to the module will likewise be deleted.

You can **copy modules** to save time and mouse-clicks when a module of the type you want to add is already present in a setup. Just click on it, then do the usual: hit Ctrl-C (copy) and then Ctrl-V (paste). Or, simply hit Ctrl-D (duplicate). The mouse cursor changes shape to show that it's ready to drop a new copy of the module wherever you next click. Existing connections to the original module are *not* duplicated – the new module shows up "naked".

Arranging Modules In The Modular Window

Modules occupy **positions on an invisible grid**, to which they automatically "**snap**" when dragged around in the window. Any connections you've already made to a

module hang on tight and follow the module around when you move it. This makes the creation and maintenance of compact and manageable module arrangements both easy and quick.

Because of the grid, modules won't move until you drag them far enough. Don't be confused **if a module seems to be stuck in place**, especially when you try to drag it sideways – the grid spacings are wider in this direction. Keep dragging and it'll eventually snap. Also make sure you haven't grabbed the module by one of its active elements – a knob, switch or jack, etc. This will adjust the control or begin the making or breaking of a connection, but won't move the module.

The Modular won't let you arrange modules such that a module is completely hidden behind another.

With very large setups, screen response may become slow. To counter this, you can: **1)** Shrink the window and use the scroll bars to focus on the area of interest – for example, make the window tall and narrow when moving modules vertically, or short and wide for shifting them sideways. Where possible, dragging the window partway offscreen gets the same result. **2)** In the **Pulsar Settings** control panel, select *Standard cursor*. It doesn't look as cool as the Pulsar cursors, but it is updated much more quickly, even when the graphics workload is quite heavy.

Making Connections In The Modular Window

In the Modular, you **make a connection** by simply clicking on an input or output jack and then on the jack you want to connect it to. To **disconnect** two connected jacks, likewise click on one and then on the other – or, click on the cable itself and then hit the Delete key. (By the way, clicking on a cable highlights it and brings it to the front. This can help you to trace its path through the rat's nest.)

Note that there are a few different **types of jack** in the Modular which carry **different signal types**. A jack can only be connected to other jacks of the same type, since the different signal types are not compatible with one another. However, each jack type has its own color, so you don't have to strain your brain too much over this. Besides, Pulsar won't let you make an "inappropriate" connection. This means it also won't let you connect two outputs to the same input or to one another (just like in the "real" world), although you *can* connect multiple inputs to the same output. Here's a list of the different jack types you'll find:

Audio	- sound and control signals (LFOs etc)
MIDI	- MIDI messages into / out of the Modular
Gate	- envelope triggering (from note events etc.)
Freq	- oscillator frequency control
Esync	- polyphonic voice control feedback

This Stuff, You Gotta Have

There are a few **mandatory modules** which are present in every Modular setup and can't be deleted. A handful of specific connections to these modules must also be made before a modular setup can begin to function. The **Empty Modular** setup is a good basis for creating new setups, as it includes precisely these modules.

The **Audio Out** module lets you send audio out of the Modular Synth via any or all of its four outputs and into the Pulsar environment at large for connection to the Mixer, the Analog Outputs, etc. Likewise, audio fed to the Modular's two audio inputs is available via the **Audio In** module.

MIDI messages which are routed to the MIDI input of the Modular Synth from the Pulsar card MIDI input or from a sequencer program are "pipelined" into the Modular setup via the **MIDI In** module and appear at its output jack. From here they can be cabled as desired in the Modular setup.

Each setup must have at least one **MVC** (MIDI Voice Control) module, whose MIDI In jack must be connected to the MIDI In module. It's possible to have more than one MVC in a Modular setup. Among other things, this permits independent voicing control using separate MIDI channels within a single Modular setup.

The MVC module is the foundation of MIDI note response and voice control activity in the Modular Synth, as described in the next few paragraphs:

The **Freq** (frequency) output of the MVC module must be connected to the Freq input of the oscillator modules it is to control in order to have those oscillators – such as the Multi Oscillator – produce pitches which track, in the "usual" well-tempered way, the incoming MIDI notes.

The **Gate** output of the MVC must be connected to the Gate inputs of any envelope generator (ADSR) modules which are to be controlled by MIDI Note events. Typically this includes both an amp (volume) envelope and a filter cutoff envelope. Gate signals have other uses as well, such as retriggering of LFOs or synchronous starting of FM operators. Also note that gate signals can come from other sources, such as the MIDI Clock module.

The **Esync** input of the MVC **facilitates proper polyphonic voice control**. It accepts feedback from the envelope generator (ADSR) module(s) used in a setup. If a setup has only one of these, its Esync output must be connected to this input. The Esync outputs of multiple envelope generators can be combined via one or more Esync Adder modules into a single signal which can be fed to this input. For more information about this, refer to the "*More About Esync*" section of this manual.

In "connection" with polyphony, the **Poly Out** module is important to be aware of. This module is not mandatory (and is not part of the Empty Modular setup), but it *is necessary for polyphonic output from the Modular Synth* (achieved by setting the *Voices* parameter in the Modular's Preset List to a value higher than 1). Simply connect it last in line before the Audio Out module when using the Modular in a polyphonic mode. Or, if you're using a mono-in/stereo-out effect such as the Chorus Mono To Stereo on the Modular output, connect the Poly Out module in line directly *ahead* of this effect.

If any of the details described here aren't entirely meaningful on the first reading – don't ponder them too deeply. Treat them as a cookbook recipe and refer back here as often as needed until you get it down. Again, the example setups in Modular Patches are a great place to start – each of them demonstrates the use of all of the "must-have" items described here.

The rest of this chapter

.... is devoted to descriptions of the modules provided with the Modular Synth. To help you find the ones you're interested in, the subchapter titles correspond to the folders in the module library (..\Pulsar\Devices\Modular).

Play around a lot and have fun!

Audio Input / Output

The Modular Synth has two inputs and four outputs for audio connections in a Pulsar setup. These are accessible inside the Modular window via the Audio In and Audio Out modules.

The **Audio In** module is included in the Empty Modular setup and comes pre-wired to the Modular Synth audio inputs. Audio signals sent to these inputs are available at the module output jacks for connection as desired in a Modular setup. The Audio In module can't be deleted, ensuring that its connections to the outside can't be lost.

The four jacks of the **Audio Out** module provide a direct line to the Modular Synth outputs. Signals generated in a Modular setup go out to the Pulsar system via these jacks. As with the Audio In module, this module is part of the Empty Modular setup and cannot be deleted, in order to ensure that these connections remain intact.



Oscillators

The Oscillators category includes **single-waveform** and **multi-waveform conventional oscillator** modules, simple and advanced **noise generator** modules, **FM operator** modules and a **sample player** module.

This group contains a large *number* of modules. There are several different single-waveform oscillator modules. Each of these, as well as the multi-waveform oscillator, comes in multiple versions with various **external pitch modulation options**, allowing non-keyboard position related pitch control arrangements, and each variety of multi-waveform oscillator is furthermore available with or without a noise waveform. This systematic assortment lets you choose exactly the oscillator you want, without wasting DSP capacity or Modular window "real estate".

Pitch Modulation Input Options

Oscillator module name suffixes indicate the pitch mod options they are equipped with:

M1 (for example, Sine Oscillator M1) indicates a **single** pitch mod input with an **exponential** control characteristic.

M2 (e.g., Sine Oscillator M2) indicates **two** pitch mod inputs, one of each type: **exponential and linear**.

" " (the third option – i.e., nothing – e.g., Sine Oscillator) indicates **no** pitch modulation inputs.

Pitch Modulation Characteristics

The **exponential** characteristic is the more "musical" one. A specific mod input change always produces a corresponding **pitch** change. For example, if a given input change raises oscillator pitch by a whole tone, then a change which is twice as large will raise the pitch by two whole tones, etc. This is the best input for pitch-bend effects. The numerical setting on the control indicates directly in semitones the maximum pitch change (produced by full-scale *positive* mod input).

The **linear** characteristic, by contrast, relates mod input changes to oscillator **frequency**. For example, if a specific input change increases oscillator frequency by 100 Hz, then a 2x change will raise it by 200 Hz, etc. This input is good for vibrato, FM and other non-pitch related effects.

Pitch mod inputs accept **bipolar signals** (both positive and negative modulation is possible) and have controls which adjust the **modulation depth**. On exponential inputs, these controls have their zero point at center, permitting inverted modulation with left-of-center settings. LFOs, envelope generators, MVC MIDI control outputs or even other oscillators can serve as control sources.

The various oscillator modules are described in detail in the following pages.

Multi Oscillator – General

This is the workhorse oscillator. Select one of the basic unit's **five waveforms** (Sine, Triangle, Sawtooth Up, Sawtooth Down, and Pulse) by grabbing the blue highlight on the waveform selection bar and dragging it left or right.

Freq In *must* be connected to the Freq Out signal of an MVC (MIDI Voice Control) module for proper operation of an oscillator module. This causes oscillator pitch to track keyboard position in the usual way.

Coarse and **Fine** tune the oscillator, in semitones and cents, up to four octaves up/down from the center setting of zero which corresponds to "standard" keyboard pitch.

The **Initial Pulse Width** control sets the base pulse width of the Pulse waveform (50% at full counterclockwise, almost zero at full clockwise). This can be modulated via the **PWM** (Pulse Width Mod) input. PWM depth and polarity are adjustable via the control above this input.

The Multi Oscillator consumes more DSP capacity (and window area) than a single-waveform oscillator. Use it when you need it, or for trying various waveforms when working on a setup. If you end up using only one of the waveforms on a Multi Oscillator, you can "economize" by replacing it with the appropriate single-waveform module.



Multi Oscillator – Available Variations

The Multi Oscillator is also provided in versions which additionally include an **exponential** modulation input (**M1**) or both **exponential and linear** modulation inputs (**M2**), permitting non-keyboard related pitch control setups. Refer to the first page of the *Oscillators* section for details.

Each of the three varieties of Multi Oscillator described above comes in an alternate version which adds a **noise generator** as a sixth selectable waveform on the waveform selection bar. These versions consume slightly more DSP capacity than their noise-less counterparts.

The noise waveform is furthermore not affected by *any* of the module inputs or controls. This makes one of the simple **noise generator modules** a better choice in spots where the other oscillator waveforms will not be used.

Single-Waveform Oscillators

These modules are largely **functionally identical to the Multi Oscillator**, except that each one produces only a **single waveform**. These modules offer a simple advantage: they are more sparing in their use of both DSP capacity and window "real estate". They're the best choice if you know that you'll need only one specific waveform at a particular point in a setup – or as a replacement for a Multi Oscillator module, if you're actually using only one of its waveforms.

Each of the waveforms available on a Multi Oscillator (except for triangle wave) is also available in a single-waveform oscillator module. As with the Multi Oscillator, each member of the single-waveform series comes in "**regular**", **M1** and **M2 versions**, each of which offers a different configuration of pitch modulation inputs (as described on the first page of the *Oscillators* section).

The **Pulse Oscillator** differs from the Multi Oscillator in the single detail that its Initial Pulse Width control produces 50% (square wave) pulse width at the *center* position (with almost zero or almost 100% at the extremes), rather than at full counterclockwise.



The **Square Oscillators** are equivalent to Pulse Oscillators with pulse width fixed at 50%, and no PWM inputs or controls. They take up less screen space than the corresponding Pulse Oscillators and consume less DSP capacity as well. They should be used instead of Pulse Oscillators when a simple square wave is desired.

The simple **Noise Generators** are extremely space-efficient – they feature outputs only, and *no* controls. They provide pink as well as white noise via either separate modules or a single combination module.



Super Noise

The **Super Noise** module is a deluxe noise generator with multi-mode filter and envelope generator which offers generous mode and modulation possibilities.

The **Filter Mode** can be set for low/high/band-pass (or "by-pass" with the Thru setting). The **Cutoff** control sets the filter cutoff or peak frequency. The **Resonance** control is active in all modes.

The filter cutoff can be made to track keyboard position by connecting the **Note** input to the corresponding MVC (MIDI Voice Control) module output. The modulation depth and polarity can be trimmed via the **Kbd Track** control.

Use of the **Envelope** requires a **Gate In** connection from the MVC module or another gate signal source. The filter cutoff or the amplifier or both at once can be modulated by the envelope. **Decay** and **Release** times share a single control (there is no Sustain control). The **Filter Env** control adjusts the amount of envelope effect on filter cutoff.

Velocity modulation effects require a connection of the **Vel In** jack to the Vel jack on the MVC module. The velocity input operates directly on the envelope level. This means that velocity modulation of filter cutoff is possible only when envelope modulation of the filter is selected.



FM Operator / FM Operator FB

FM synthesis techniques can also be used in the Modular, extending all the way to construction of a multi-operator FM synth. These modules are essentially sine oscillator modules with *linear* pitch modulation inputs and other features to optimize them for FM synthesis.

The **Coarse** and **Fine** controls adjust a single *linear frequency* setting which is better suited for FM than the usual semitone/cents settings. The base setting of 1.000 corresponds to "standard" keyboard pitch (equivalent to a setting of 0 semitones on other oscillator modules).

The **Coarse** control sets the frequency in *harmonic* (rather than semitone) steps over whole-number values 1 through 32 (up five octaves) as well as 0.5 (one octave down). The **Fine** control gradually *scales* this setting over a range of 1.999:1 (*almost* one full octave up). The complete 7-octave span sits a good deal higher than that of most other oscillators, as you'll no doubt quickly notice.

The **Detune** control permits a still-finer shift of +/- 20 cents.

The **Freq** input *must* be connected to the Freq Out signal of an MVC (MIDI Voice Control) module for proper operation of the internal oscillator. This causes oscillator pitch to track keyboard position in the usual way.



EG In is the operator's gain control input. It's intended to accept the output of a DC (unipolar) **envelope generator**. Other control sources (e.g., LFOs) can be used with some success. In any case, you must connect *something* here in order to get any output from the operator.

The two **Mod** inputs are identical. Each has its own modulation amount control. FM happens here. Typical input signals are outputs from oscillators (classically sine), from other FM operators, or even from the *same* operator.

The **Output Level** control is useful for fine-tuning the interaction between operators in a multi-operator setup.

Gate In is used with the remaining controls, which are not essential for basic use of the FM Operator. They permit **precise phase control of the internal sine oscillator**,

which can be quite useful for tweaking the timbre in multi-operator FM setups. A gate signal from the MVC or other gate signal source must be applied to the **Gate In** jack in order to activate these features.

The **same gate signal must be applied in common** to all FM Operator modules which are to have their oscillator phases coordinated, in order for these features to produce their intended result.

The **Free Run / Retrigger** button controls whether the internal sine oscillator produces a continuous, uninterrupted waveform at all times or is reset to a specific phase point whenever a gate event arrives via the Gate In jack.

The **Phase** control adjusts the phase point to which the oscillator is reset in Retrigger mode.

FM Operator FB (FeedBack)

This module is largely identical to the "standard" FM Operator. The FM Operator description up to this point applies equally to both. Additional internal functions enable the FB version to complete an **operator modulation loop**. In a two-operator loop, for example, the output of each operator is routed to a modulation input of the other, producing a form of modulation feedback.



Such a configuration is a common component of FM algorithms. However, this is not possible using only the standard FM operator. To "close" a loop, whether it includes one, two, three or more FM operator modules, **at least one must be of the FB type**. Which one is not critical, since the loop has no beginning or end.

The FB module is also capable of fixed-frequency operation, another common FM component. **Fixed Freq** dials the frequency in directly (0.001 Hz - 9772.000 Hz). A non-zero setting here overrides the Coarse/Fine tuning setting and disables the Freq input (mod inputs still work).

A thorough discussion of FM synthesis is well beyond the scope of this manual. However, you can find some information on this topic in the *FM One* chapter, which describes the Pulsar FM Synth.

WAV Oscillator (Sample Player)

This module produces sounds by way of sample playback. As with the Sample Player devices, Akai S1000 format samples (but here, **single samples only**, not programs) are **dragged from the File Browser and dropped onto the window of this module** to load it. Otherwise, this module can be used just like any other oscillator module, with a few differences and restrictions as described below.

The WAV Oscillator **must be used with the MIDI Voice Control Sampler** module. The standard MVC won't work. (The MVC Sampler *can* control conventional oscillators.)

The **Freq** input of the WAV Oscillator must be connected to the **Freq Off** (offset) output of the MVC Sampler, and **not** to its Freq Out jack. (If you're layering conventional oscillators on top of the WAV Oscillator, *these* can be connected to the Freq Out jack of the MVC Sampler.)

Since the WAV Oscillator is sample-based and not a simple oscillator, it features **Note** and **Gate** inputs which must be connected to the corresponding MVC Sampler outputs.

The **Note** input provides MIDI note numbers which facilitate the WAV Oscillator's key-mapping functions (these will be familiar to any experienced sampler user).



The **Gate** input lets the WAV Oscillator know when a new note comes in, so it can trigger sample playback. For the same reason, **envelope generators** used in conjunction with the WAV Oscillator should obtain their Gate signal from the **To EG** output of the WAV Oscillator, instead of directly from the MVC module, as is normally the case. This allows the WAV Oscillator to ensure that the triggering of envelope generators is properly synchronized with the triggering of samples. (Esync connections between the envelope generators and the MVC Sampler are made in the usual way.)

Once you've made these connections – plus of course an audio Out connection and optionally a Pitch Mod connection – *and* loaded a sample (don't forget!), you

can use the key-mapping controls of the MVC to put the sample where you want it on the keyboard, and the tuning controls to further adjust its tuning.

CenterKey specifies the MIDI note number on which the sample will be played at its original or "native" pitch – that is, assuming that the tuning controls on the WAV Oscillator *and* on the MVC Sampler are all set to nominal. **LowKey** and **HighKey** set the lower and upper limits of the keyboard range over which the sample plays. Notes outside of this range produce no response.

The CenterKey value is *not* restricted to the LowKey-HighKey range. This may not seem entirely logical at first. However, all it means is that on every key in the key range in which the module responds, the sample is played back transposed from its original pitch.

Alternatively, if the **Fixed Freq** mode is activated, the sample will play back at its original pitch on all keys in the LowKey-HighKey range (again, disregarding tuning adjustments).

Fixed Freq mode is useful for creating percussion maps – as it allows the keyboard layout to be arranged without affecting sample tuning – for layering a fixed-pitch sample over a different sound (whose pitch may or may not be



fixed), and for mapping a sample to multiple keyboard notes to make it easier to play or to conform to preexisting MIDI layouts (GM drum maps, sound effects, etc.).

As to the tuning controls – note that **Coarse** (semitone) and **Fine** (cents) controls appear both on the WAV Oscillator and on the MVC Sampler module. If you're using multiple WAV Oscillators connected to a single MVC Sampler, the MVC Sampler tuning controls affect all of the oscillators in common – thus serving as master tuning controls – while the tuning controls on each oscillator of course affect only that oscillator, permitting individual sample tuning corrections as well as the creation of detune effects or intervals.

Filters

Operation of the filter modules is very similar from one type to the next.

Each one has an audio input and output, a **cutoff** control which adjusts the filter's cutoff or passband-peak frequency, and three **modulation inputs**, each with its own depth control. Most also have a **resonance** control which adjusts the amount of signal boost or "ringiness" around the filter cutoff / peak frequency, and which can bring the filter itself right to the edge of oscillation.

The **4-Pole Low Pass** filter lets low frequencies pass and reduces higher-frequency signal content by 24 dB per octave above the cutoff frequency.

The **4-Pole High Pass** filter does the converse, letting high frequencies pass and reducing lower-frequency signal content by 24 dB per octave below the cutoff frequency.

The **2-Pole Multimode** filter provides simultaneous low-pass, high-pass and band-pass filtering via separate outputs. The low-pass and high-pass functions cut off at 12 dB per octave beyond the cutoff frequency, while the band-pass function slopes off at 6 dB per octave on either side of the peak frequency.



The **8-Pole Band Pass** filter allows a narrow band of frequencies around the passband peak frequency (adjusted via the Cutoff control) to go through, rather like an extremely exaggerated wah-wah pedal.



The **18dB Low Pass** filter has a more gentle cutoff slope than the 4-Pole Low Pass. In addition, this filter is not a resonant filter and thus has no resonance control. It's useful for more subtle high-frequency rolloff applications where the "synth-filter" sound is not desired.



Envelopes

Overview

Envelope Generator (**EG**) modules are triggerable, "one-shot" generators of control signals which follow a defined course. They're usually used to control amplitude and filter cutoff but can be connected to virtually any control input to produce pitch envelopes, pan envelopes, etc.

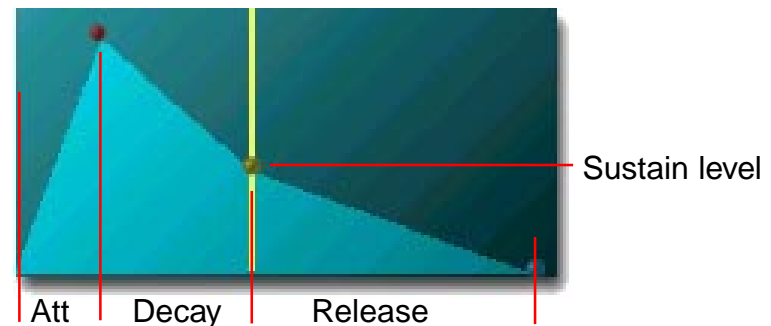
EGs are triggered via their **Gate** inputs by **gate events** (gate-on and gate-off). Gate events are usually derived from **MIDI note-on and note-off events** via the **MVC (MIDI Voice Control) module**. The Gate input of an EG is normally connected to the Gate Out of this module.

Gate events can also come from other sources such as the **MIDI Clock module**, which continuously generates **repetitive gate events** on its own (either in free-run mode or synchronized to an incoming MIDI clock) for automatic rhythmic envelope triggering.

The **Esync output** is a feature of most Pulsar EGs. EGs which have this output will function normally without any connection to it. However, for **proper polyphonic voice control**, it *must* be connected to the **Esync input of the MVC module** (either directly or via an Esync Adder module). This is discussed further below. (For more details, refer also to the *"More About Esync"* section of this chapter, or to the description of the MVC module.)

Envelope Generator Types

Pulsar Modular Synth EGs have two main distinguishing points: Each one is either an **ADSR or Multisegment** EG, and each one produces either **unipolar or bipolar** outputs. These points are explained below.



ADSR (Attack-Decay-Sustain-Release) Envelopes

The **Attack** phase begins when the envelope is triggered by a gate-on event. In this phase the envelope level rises from zero up to its (fixed) **peak level** at a rate determined by the Attack control.

When the peak level is reached, the **Decay** phase begins. The Decay knob sets the speed at which the envelope drops from its peak level down to the level set by the **Sustain** control. The amount of time this takes depends on the Sustain level setting. If the Sustain level is set fairly

high, it is reached relatively quickly. If Sustain is set to maximum, this occurs virtually instantaneously and the Decay phase is essentially skipped.

When the Sustain level is reached, the **Sustain** phase begins. The envelope holds at this level until a gate-off event occurs. Naturally, this phase has no specific duration.

The **Release** phase begins when a gate-off event occurs. The envelope drops from its current level back towards zero at the rate set by the Release control. Release phase can also begin during Attack or Decay phase, if a gate-off occurs during one of these phases.

ADSR Module Common Features

The **Gate** input and **Esync** output are discussed in the Overview section on the preceding page.

Each of these modules features the basic **Attack**, **Decay**, **Sustain** and **Release** controls described just above.

The **TKF** (Time Key Follow) input is intended for connection to the Note output of the MVC module. This allows all envelope times to be lengthened or shortened in tandem in response to keyboard position. An associated control adjusts the depth and direction of this modulation.

The **Level Vel** input is intended for connection to the Vel output of the MVC module. This connection causes the envelope peak and sustain levels to vary in response to note velocity.



ADSR with Amp

The **ASDR / Amp** combines ADSR and control amplifier (two separate control amps in the **ASDR / Stereo Amp**) within a single module – very handy, since amplitude envelopes find use in almost every Modular setup.

The envelope generator output is available directly via the **DC** and **AC** control outputs (the single output on the stereo module is DC). The DC output is for use with modulation inputs (such as on the VCA module) which accept only positive control values.

The **Check** button triggers the envelope directly on the module itself. This is convenient for testing envelopes without playing notes, but also opens up the performance option of (re)triggering individual envelopes independently of note events.

The Time Key Follow input and control discussed above are joined by a **Time Vel** input and control for adjustable modulation of envelope timings in response to note velocity info (from the Vel output of the MVC module).

The modulation assortment is rounded out by the addition of a **mod depth control for the Level Vel** input.



ADSR Simple Unipolar

The special feature of this module is its **Slope** control, which affects the the shape of the decay and release portions of the envelope. It provides a continuous adjustment between **linear** (full counterclockwise) and **exponential** (full clockwise) curve shapes. Linear curves

are the norm for Modular envelope generators. The exponential curves tend to have a smoother and more natural sound when **used to control amplitude**, especially during slow, extended decays and releases.

In fact, this module is mainly intended for amplitude control (via the VCA module). Therefore, its single output produces a DC signal (hence the name Unipolar), and it also includes an **Esync** output for connection to the MVC module (which coordinates polyphonic voice assignment on the basis of amplitude envelopes).



ADSR Simple Bipolar

This module is for **control of parameters other than amplitude**. It outputs simultaneous positive- and negative-going envelopes via two outputs. These envelopes are inverted versions of one another – both of them start and end at zero – and have linear curve shapes. In keeping with its intended usage, the module has **no Esync output**.



ADSR Vintage

This module expands upon the ADSR Simple Unipolar module by adding an **attack slope control** which varies the shape of the attack curve smoothly between linear and exponential. This is thus somewhat of an **all-around ADSR** – it offers linear curve shapes and an AC envelope output for control of pitch, filter frequency, etc., along with exponential curve shapes, a DC envelope output and an Esync output for amplitude control use.

Multisegment Envelope Unipolar And Bipolar

These are complex envelope generators permitting point-by-point envelope design. They are described in detail over the next few pages. The Unipolar, for amp control, has points with control values in the range 0-127. The Bipolar, for non-amp uses, has points with values -63 to 63, and correspondingly omits the Esync output.



Graphical Envelopes

The graphical envelopes found in various Pulsar synths are all basically alike. The description here applies to all of them.

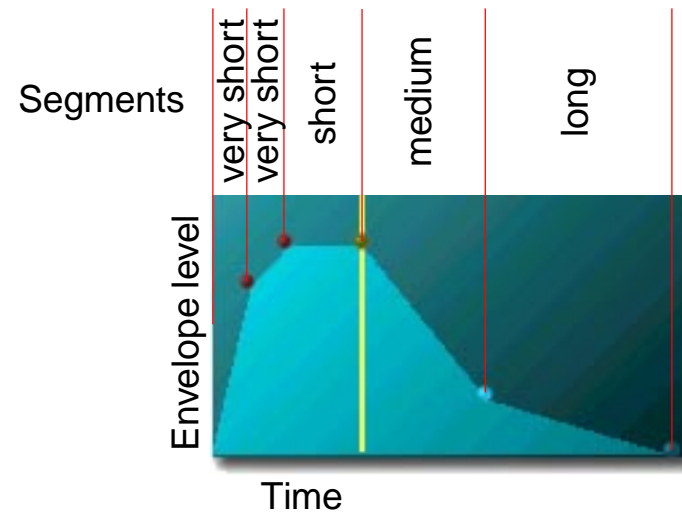
Segments And Points

A graphical envelope consists of a sequence of envelope **segments**. Envelope segments are defined using envelope **points**. Each point has a **value** (level) between 0 and 127 in a *unipolar* envelope. *Bipolar* envelopes also permit negative point values – the full range is -63 to 63.

A point's vertical position graphically indicates its value. In operation, the envelope begins at the left and rides up and down along the lines which are drawn between consecutive points. In the diagram, each envelope point sits at the right-hand end of its segment, since the point is where the segment ends.

Each segment has a **duration** (time). This is indicated graphically by the segment's width – the horizontal spacing between two points. Longer-duration segments appear wider than segments with shorter durations.

However, because **segment times can range anywhere from tiny fractions of a second up to ten seconds**, the horizontal time scale is adjusted individually for each segment, so that shorter segments are displayed



proportionately wider (and longer segments proportionately narrower) than they would be if the same scale was used for all of them. This per-segment time scale "compression" yields a more usable overview with envelopes containing a wide range of segment times.

Edit both point value and segment time by dragging points with the mouse, or select a point and edit its level or time via knobs or direct keyboard entry of numerical values. The values/times of other point are not affected. **Add a new point** between two existing points by double-clicking in the space between them. You can have **up to 99** points. **Delete an existing point** by double-clicking on it. This deletes the associated segment and correspondingly cuts the segment's time out of the envelope as well.

Point Modes

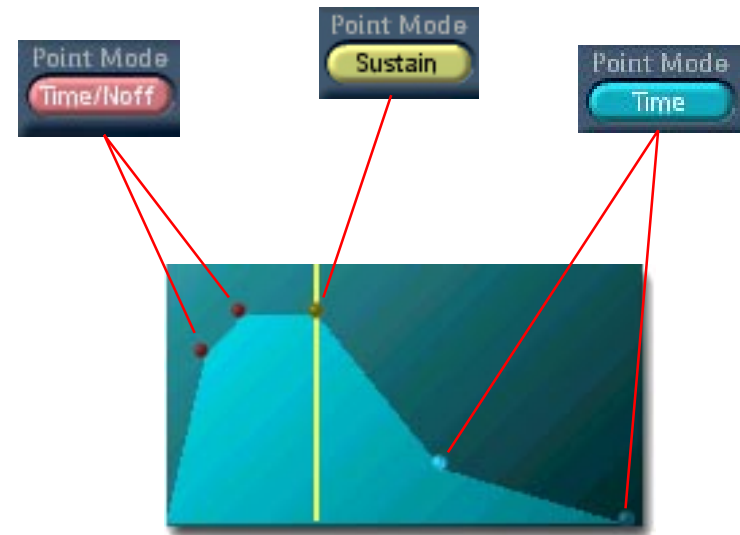
Clicking on the Point Mode button switches the selected envelope point (and its segment) between **three modes** which allow for versatile envelope construction:

Time mode: The simplest mode. The envelope always goes all the way to the segment endpoint. When the segment time is up – and only then – the envelope proceeds into the next segment.

Time/Note-Off mode: Same as Time mode, except that segments in this mode are cut short by a Note-Off (key release) event. Note-Offs also cause any Time/Note-Off segments which have not yet begun to be skipped.

Sustain mode: The envelope "pauses" at a Sustain point if no Note-Off event has yet occurred, and remains there indefinitely until the Note-Off occurs. Otherwise identical to Time/Note-Off mode.

In the display, the **mode** of each envelope point is **indicated by its color**. In addition, the Sustain point is visually highlighted by a vertical line running through it.



Using Point Modes

Note-Off events cause an envelope to skip "straight across" – without a level change – **to the first Time mode** segment (unless it is already in one) **and continue** from there.

Time mode is thus useful in the release phase of an envelope (following a Sustain point) or for the last segment of any envelope (to ensure a non-zero release time and avoid note-off clicks). Envelopes consisting *entirely* of Time mode points are not affected by note-event duration – percussion envelopes are created in this way.

Time/Note-Off mode is normally used for attacks, decays and anything else coming *before* a Sustain point. It can also be used for envelopes with no Sustain point, such as piano/guitar-type envelopes, which decay steadily towards zero for as long as a note is held and cut off quickly if/when the note is released.

Envelope Loops

Envelope loops can be set up by clicking on the **Loop Point Set** button and then on two envelope points, which become the loop **start point** and **end point**. This is



indicated graphically as shown above. An envelope can have only one loop. Creating a new loop clears any existing loop.

Any number of envelope points can be included in a loop. However, **a loop cannot include a Sustain point**. A loop must therefore be either completely before or completely after the Sustain point, if the envelope contains one.

The **Loops** control lets you dial in a specific repeat count 1-255 ("0" = infinite repeats). The loop repeats this number of times, or until note-off, if it contains Time/Note-Off segments. A loop containing *only* Time mode segments *always* repeats the specified number of times.

Loops are played by jumping from the end point back to the start point (forward-only looping). The time setting of the start point is applied to the end-to-start transition.

A loop can be removed by clicking on the **Loop Point Delete** button and then on one of the loop points. This merely clears the loop – the points which were included in the loop are not affected.

LFOs

Multi LFO / Multi LFO (Fast) M1 / Multi LFO b

These deluxe **LFOs** have six different waveforms (Sine, Square, Sawtooth Up, Sawtooth Down, Triangle and Random). These are selected by grabbing the blue slide switch and dragging it left or right.

The **Rate** control sets the base LFO frequency or speed (0.00-400.00 Hz for the Fast M1, 0.00-50.29 Hz for the others). The **Rate Mod Lin** input (M1 and Fast M1 only) permits external modulation of the LFO rate with a linear control characteristic. The associated **Rate Mod** control adjusts the modulation amount and polarity.

The normally free-running **LFO waveform can be reset** by gate-on events appearing at the **Retrg** (retrigger) input. The switch above this jack must also be set to On – set it to Off to use the LFO envelope (see below) with LFO retriggering disabled. The **Initial Phase** control sets the phase point to which the LFO is reset. Connect Gate Out of the MVC (MIDI Voice Control) module to this input to cause the LFO to be reset at the start of every note.

The Multi LFOs (excluding Multi LFO b) include a simple **envelope** which is likewise triggered via the Retrg input and operates only when there is a connection to this input. The **Delay Time** control adjusts the delay between the trigger (gate-on) event and the start of the LFO attack.



The **Attack** control sets the fade-in time of the LFO following this delay. The **Release** control sets the fade-out time following gate-off.

The LFO has two **MIDI clock sync** options which produce a one-cycle per beat (24 MIDI clocks) LFO speed with the **Freq** input connected to the Freq output of the MIDI clock module. For sync to an external MIDI Clock, the switch above this jack must be set to **Ext**. Setting it to **Int** allows sync to the Modular's internal MIDI clock.

Finally, the overall output level of the LFO is adjustable via the **Output Level** control.

Sine LFO

This module is a good choice when the utmost simplest, continuous-running, unsynchronized, unmodulated sinewave LFO is all that is needed. It takes up less space onscreen, requires only a single connection and uses up less DSP power as well. Its controls include **Rate** for the LFO speed and **Output Level**.



Triangle LFO

This is another fairly compact single-waveform LFO module with a mix of selected features from the Multi LFOs. It has an LFO envelope with **Attack** time and pre-attack **Delay Time** adjustments (the envelope has no release phase), plus waveform retriggering capability with **Initial Phase** control. A **Retrigger** input is provided to actuate both envelope and retriggering, along with a switch which disables retriggering only, allowing use of the envelope alone.



Mix And Level

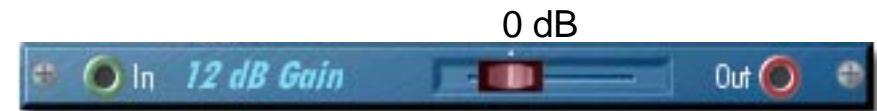
This is a set of simple modules for modifying signal levels. A small mixer is included in this group.

The **6dB Gain** and **12dB Gain** modules contain amplifiers providing a signal boost of x2 (+6dB) and x4 (+12dB) respectively. Each one has a slider which allows the output to be adjusted continuously all the way down to zero. These modules are often useful for boosting the output of a resonant filter module, as these filters reduce their gain internally when their resonance controls are turned up to avoid internal overloading.

The **Attenuator**, like the above gain modules, includes a slider for adjustment of output level, but contains no amplifier. Thus, its output at the maximum slider setting is exactly equal to the input.

The **4-Input Mixer** provides a simple mixing function for up to four signals, such as multiple oscillators. It has a **Gain** (level) control for each input and a **Master Gain** control for adjusting the output level of the mix.

The **VCA** module is an attenuator whose gain is controlled by an external signal instead of a slider. It's intended mainly for amp envelope control. Its **DC Mod In** jack accepts the output of a **unipolar control signal** source such as the ADSR Simple Unipolar or the Multisegment Envelope Unipolar envelope generators.



Switches

Switch modules allow the **selection of various destinations or sources for a signal**. This lets you build various configurations into a Modular setup which can be accessed quickly and repeatably without the need for recabling or for trying to remember "where that cable used to go?" or "what was connected here before?".

In addition to this basic convenience factor, switch modules also permit **signal routing variations** within an otherwise fixed setup to be **stored as presets**, since a Modular preset includes the switch positions of any switch module in a setup.

As a rule, both **normal audio signals and control signals** (LFOs etc.) can be routed using switch modules. These modules do not handle MIDI and Gate signals.

The **1x4 Switch** routes a single signal to one of four different destinations. For example, it could be used to apply an audio signal from the Audio In module to one of four completely different audio processing chains.

The **4x1 Switch with Gain** selects a signal from one of its four inputs. This module also has individual **Gain** controls on each of its inputs, letting you balance possibly unmatched levels on the different inputs to one another. This module could be used, for example, to select one of the three outputs of a Multi-Mode filter or the filter input

signal instead (i.e., bypassing the filter) – or to select one output from a set of different Multisegment envelope generators, etc.

The **4x1 Switch** module is the same as the 4x1 Switch with Gain, but without the gain controls.



Effects

Pulsar Modular effects fall into two main categories.

Polyphonic effects are able to process the signals of multiple voices separately and preserve their "separateness" when used in a polyphonic setup. Behind the faceplate, polyphonic effects are actually **multiple copies – one per voice**. Therefore, they can be used anywhere in a polyphonic setup, but can be relatively "expensive" (in terms of DSP capacity consumed).

Monophonic effects (green faceplates) load only a single copy of an effect, but **defeat polyphonic voice control** beyond the point where they are used. In a polyphonic setup, they are best used following the Poly Out module. Most Pulsar Modular effects are of this type.



The **Distortion** module is available in both varieties and illustrates some of their differences. It produces soft- and hard-distorted versions of its input, with distortion amount adjustable via the slider.

The **green monophonic Distortion** module has an effect like that of an amp on a guitar – the *mix* of notes is distorted. They get mashed together into a single thick-sounding grunge whose texture is affected by each note. The **blue polyphonic Distortion** module distorts each voice separately, more like a group of lead guitarists each playing through their own amplifiers – the notes don't interact with each other at all and remain sonically distinct.

The remaining effects described here are **all of the monophonic type** unless otherwise indicated and should be used at the end of the line, *after* the Poly Out module.



The **Chorus** produces stereo output from mono sources. It's thus best used as the very last module before the outputs (i.e., *after* the Poly Out module!). The **Rate** control adjusts the speed of the internal sweep LFO. The **Depth** control adjusts the sweep amount. Each of two separately swept delay lines is mixed to one of the outputs. The amount and phase of delayed signal going to left and right outputs is adjusted by **Mix 1** and **Mix 2**, respectively.

The **Auto Pan** module pans a mono source signal between its two outputs under control of an internal LFO. **Rate** and **Depth** controls for the LFO are provided. As with the Chorus, it yields **stereo output from a mono source** and **should be used after the Poly Out module**.

The **Auto Pan M1** module is like the Auto Pan module, but adds an input for **external control of pan depth**, along with an associated amount/polarity control.

The **Pan** module has no LFO. **Initial Pan** can be set manually for static pan control. An input for **external pan modulation** is provided, with an associated depth/polarity adjustment.

The **Audio Modulator** is similar to a VCA. The **Offset** control sets the initial gain, so that an adjustable amount of the input signal between zero and 100% is always fed to the output. The **Mod 1** and **Mod 2** inputs accept *bipolar* control signals which can then **increase or decrease the base gain** set by the Offset control. Each Mod input has an associated depth/polarity control. This module appears in both monophonic and polyphonic categories.

The **Fixed Filter Bank** is a set of boost/cut filters. Each can produce a gentle boost or cut within its own fixed frequency band, without strongly affecting other frequencies. **Gain** adjusts the overall output level.



MIDI

MVC (MIDI Voice Control)

The **MVC** (MIDI Voice Control) module is the first line of processing for incoming MIDI messages. It manages MIDI-controlled voicing in the Modular Synth. Its **MIDI In** jack must be connected to the output of the MIDI In module.

NOTE: The MVC module is for use with *standard* oscillator modules. **When using the WAV Oscillator** (sample playback) module, **you must use the MIDI Voice Control Sampler** module (described following) in place of the MVC. Use of both types of MVC in one setup is possible.

The **MIDI Channel** control can be left on "Omni" (MVC responds on all MIDI channels) or set to confine MIDI response to a specific channel between 1 and 16.

Two or more MVC modules can be used within a single Modular setup to control separate voicing circuits. These will respond independently of one another if each MVC is set to its own MIDI channel. The MIDI channel setting can also be used to isolate the Modular from other Pulsar synths running in the same Pulsar setup.

Coarse Tune adjusts the pitch of controlled voices 4 octaves up or down in semitone steps. **Fine Tune** adjusts pitch in cents over a range of one semitone up or down.



The **Pitch Wheel Range** control sets the maximum pitch change of the controlled voices in response to incoming MIDI Pitch Bend messages.

The **Port/Gliss Time** (Portamento/Glissando) control adjusts the speed of up/down pitch glide from one note to the next which is activated when the **Port/Gliss Mode** switch is moved away from position 1 ("Off"). Both **Portamento** (smooth glide) and **Glissando** (glide in semitone steps) are available, each in two modes. In "plain" mode, the pitch glide between two notes always occurs. In **Fingered** mode, the glide occurs only when a new key is played before the previous key is released.

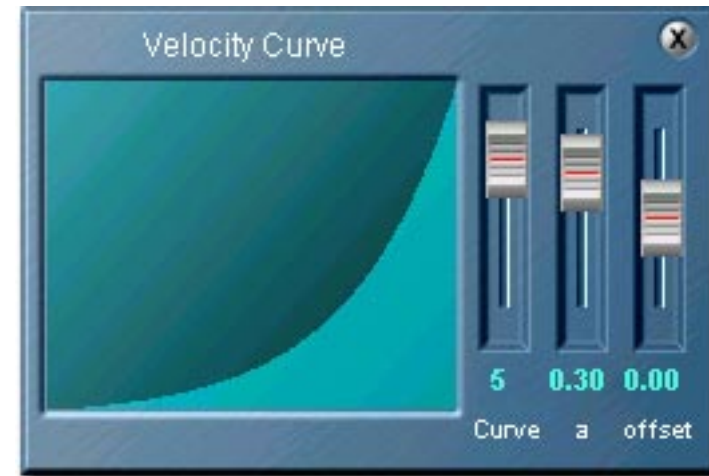
The **Gate Out** jack must be connected to Gate In of all envelope generators and LFOs which are to be triggered by MIDI note events.

Esync is a functional necessity for polyphonic voice handling (see "*More About Esync*" later in this chapter for more info). It must be connected to the Esync output of an envelope generator, typically the one which generates the primary voice amplitude envelope.

Freq Out is a control signal specifically intended for oscillator pitch control. It is derived from the combination of MIDI note number and pitch bend values, interpreted with respect to the current MVC settings for Coarse/Fine Tuning and Pitch Wheel Range, with possible MVC Portamento/Glissando effects thrown in on top. It must be connected to the Freq input of an oscillator in order to cause its pitch to respond to all of these factors in the usual way.

The **Note**, **Vel** and **AT** jacks send out control signals derived from MIDI note event Note Number and Velocity values and MIDI Aftertouch messages, respectively. These are in general intended for connection to correspondingly-named modulation inputs on other modules for keyboard tracking, velocity modulation etc.

The **Vel Curve** button opens a window for editing of a velocity curve which allows contouring of the response characteristic of the **Vel** output. The **AT Curve** button likewise allows editing of an aftertouch response curve for the **AT** output.



Editing of these curves is fairly simple. The **Curve** control at left selects one of six basic curve types, the **a** control at center produces variations on the chosen curve and the **Offset** control at right dials in a minimum value for the curve.

MIDI Voice Control Sampler

This module is similar to the "standard" MVC module, but has special features to support the **WAV Oscillator** sample playback module. This module *must* be used to control the WAV Oscillator – the standard MVC won't work.

Most aspects of the MVC Sampler are identical to those of the standard MVC. Only the differences between the MVC Sampler and the standard MVC are described here.

Naturally, the **MIDI In** jack must be connected to the output of the MIDI In module. This might seem too obvious to mention, but it's easily overlooked, since the MIDI In connection to the standard MVC is already made for you in the Empty Modular.

The **Freq Off** (frequency offset) output handles pitch control of the WAV Oscillator. The **Freq** input of the WAV Oscillator must be connected to *this* output, rather than to the accustomed Freq Out signal.

(The MVC Sampler has both **Freq Off** *and* **Freq Out** outputs, enabling it to **also control standard oscillators**. In fact, when using the WAV Oscillator, even if you wish to use standard oscillators along with it, you can delete the standard MVC altogether, if the other oscillators are



to be layered onto the same MIDI channel as the WAV Oscillator, *and* if you don't need the Portamento/Glissando functions, which the WAV Oscillator does not support.)

The **Note** connection from the MVC Sampler to the WAV Oscillator is required for correct keyboard pitch response (unlike with the standard MVC, where a Note connection is optional). **Gate Out** must go to the Gate input of the WAV Oscillator (which uses it to trigger sample playback), and *not* to an envelope generator.

Finally, as alluded to just above, the **MVC Sampler does not provide the Portamento/Glissando functions** found on the standard MVC.

For information regarding the WAV Oscillator, refer to the *Oscillators* section of this chapter of the manual.

MIDI In

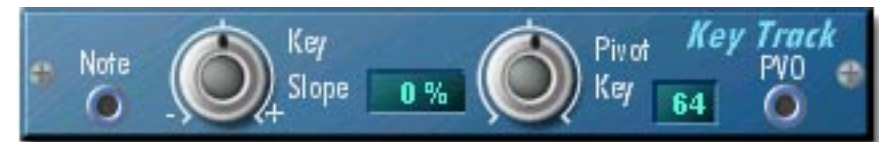
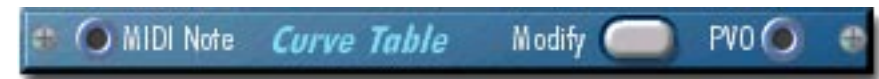
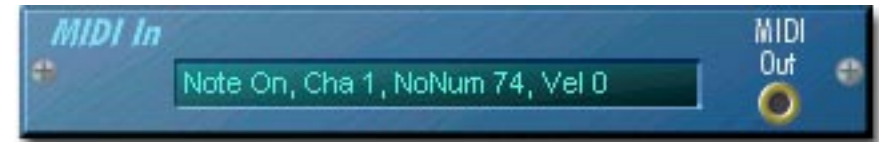
The **MIDI In** module provides MIDI input to the Modular. MIDI messages which are routed to the MIDI input of the Modular Synth from the Pulsar card MIDI input or from a sequencer program are "pipelined" into the Modular setup via this module and appear at its output jack, in addition to being displayed in its text window. (This can come in very handy as a diagnostic tool for MIDI problems.) This module is mandatory in a Modular setup and cannot be deleted.

Curve Table

This module generates a control signal from **MIDI note numbers** (or other MIDI numbers such as controllers) by way of an editable transform curve similar to those offered by the MVC module for Vel and AT.

Key Track

A simplified version of the Curve Table with a linear-only curve. **Pivot Key** sets the note number at which the module output is zero. The output rises steadily with increasing key number and drops steadily with decreasing key number (or vice versa), which is useful for such things as panning based on keyboard position. Rate and direction of the change are set by the **Key Slope** control.



Key Zone

The **Key Zone** module filters MIDI note messages fed into its **MIDI In** jack. Only those messages whose note numbers are within the range set by the **Low Key** and **High Key** controls are passed through to the **MIDI Out** jack. The **Root Key** control allows for simultaneous transposition of the messages which get sent through, effectively shifting them up or down on the keyboard.

MIDI Controller

This module allows the use of **MIDI control change** messages as modulation sources in a Modular setup. Simply **assign a controller number** via the text fader, and then connect the output to modulation inputs as desired. The output can also be passed first through the Curve Table module to shape the control characteristic.





MIDI Clock

This module **translates external MIDI clock messages** received by the Modular into **tempo and gate messages** usable within the Modular setup. This requires that the output of the MIDI In module be connected to the MIDI In jack on this module).

Flipping the switch from **External** to **Internal** converts the module to a standalone MIDI clock source with tempo (**Rate**) adjustable via the text windows in whole and hundredth **BPM** (Beats Per Minute) according to the MIDI standard of 24 PPQN (clocks or pulses per quarter note).

In both modes, the **Gate** output produces one **gate on/off event per clock** for use with the Gate Multiplier module (see below). The **Frequency** output produces a control signal proportional to the tempo of the generated or detected MIDI clock stream. This is intended for connection to the Freq input of an LFO, in order to control its speed and thus **synchronize the LFO waveform to the MIDI clock** tempo. The **Start/Stop** output controls the Gate Multiplier module in response to MIDI Start and Stop commands from an external clock source.



Gate Multiplier / Freq Multiplier

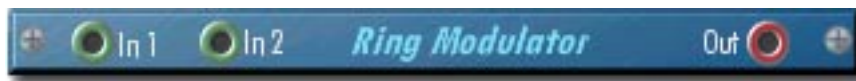
The **Gate Multiplier** module accepts a gate event stream from the MIDI Clock module and **divides it into slower streams** for synchronized envelope triggering, LFO retriggering, etc. It has two outputs with independent frequency settings. The **value to the right of the slash is adjustable** and sets the **number of input gate pulses** which are counted **for each output gate pulse**. With a 24 PPQN input from the MIDI Clock module, **setting this value to 24 yields one gate event per beat**. This module can also be controlled by the MIDI Clock module via the **Start/Stop** input in response to external MIDI Start and Stop commands.

The **Frequency Multiplier** module performs a similar function with respect to the Frequency output of the MIDI Clock module, permitting **four different simultaneous LFO speeds with waveforms synchronized to the same clock source**.

Special



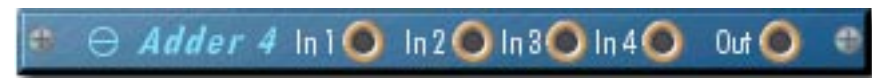
The **Control Mixer** allows amplitude modulation of control signals. For example, use it to apply an envelope to an LFO. The **Offset** control sets the base gain, thus controlling the amount of input signal fed to the output in the absence of modulation. The **Mod 1** and **Mod 2** inputs accept *bipolar* control signals which can then **increase or decrease the base gain** set by the Offset control. Each Mod input has an associated depth/polarity control.



The **Ring Modulator** is a no-frills amplitude modulator effect which produces its output by simply multiplying its two (audio) input signals by one another – rather like a VCA whose control input is also a bipolar audio signal instead of an envelope or a steady DC level. Since there's no built-in gain offset, the Ring Modulator produces output only when there are signals present on *both* inputs.



The **Poly Out** module is a special-purpose module which is **required for polyphonic output from the Modular**. Its function is to properly deliver the signals stemming from multiple voices to the Modular output. Connect it inline **directly before the Audio Out module**. The **fader** can reduce the level of the combined multi-voice signal to **avoid digital overload**. This is often necessary, since each of the voices is otherwise individually capable of using up the full headroom of the Modular audio output. The more voices you use, the more you'll need to decrease the level.



Esync Adder 2 and **Esync Adder 4** permit **merging of Esync signals from multiple envelope generators** for return to the Esync input of the MVC module. Use these in particular when your setup includes **multiple amp envelopes**, to ensure that polyphonic voicing is correctly managed by the MVC. (For details about Esync, refer to the "More About Esync" section of this chapter.)

More About Esync

Here's an explanation of the Esync (Envelope Sync) connection, for those who wish to know more about why it's there. If you're content simply to use it, feel free to skip over this part (but be sure to read the instructions regarding Esync in the "Overview And Ground Rules" section of this manual and elsewhere).

The Basic Issue

An important function of the MVC module is the coordination of voice reassignment or "stealing". Every synth has a specific number of voices as determined by its design. This limits the number of sounds it can make at one time. (This is actually not an issue with *real* modular synths, which are inherently monophonic anyway.) Pulsar synths let you specify how many voices they have, but are likewise limited by this setting. When all voices are in use and a new note is played, one voice must be "stolen" away from what it is doing to play the new note. Which one to steal? Ideally, the one least likely to be missed.

How Pulsar Synths Handle This

Pulsar synths resolve this question by scanning the amp envelopes of all voices to find the momentarily quietest voice. However, this often turns out to be one which has just been started and not yet heard (since notes often arrive in tight groups – e.g., in chords, on downbeats,

etc). A preferable choice for stealing is a voice whose envelopes are already well along in the release phase, even if this voice is not absolutely the quietest one.

Why The Modular Synth Needs Help From You

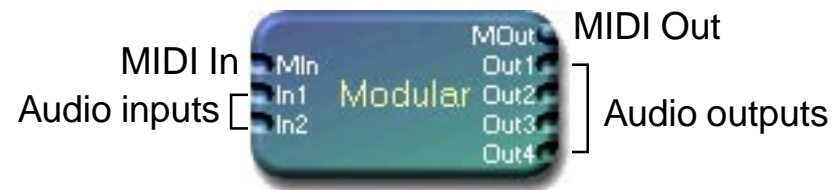
Most Pulsar synths are "hard-wired" – their structure is fixed. They use internal Esync connections to route the necessary amp envelope status info back to the internal equivalent of an MVC module.

The Modular has no fixed structure. A Modular setup may contain any number of envelopes, not all of which affect voice volume. Pulsar is not clever enough (not *yet*, anyway) to figure out which or how many ADSR modules should have Esync connections, so this is left up to the user to decide.

Can You Foul This Up Somehow?

What if you pick the "wrong" ADSR(s) for Esync hookup? Not to worry. You can't do any damage and you *may* not even notice a difference. Given the flexibility of modular setups, there could in some cases not even be a truly "right" choice. Most important is that *something* is connected to the MVC Esync input (even if only the MVC's own Gate output!) – otherwise, *nothing* at all will happen.

Connections



Project window (module) representation



Minimized (icon) representation