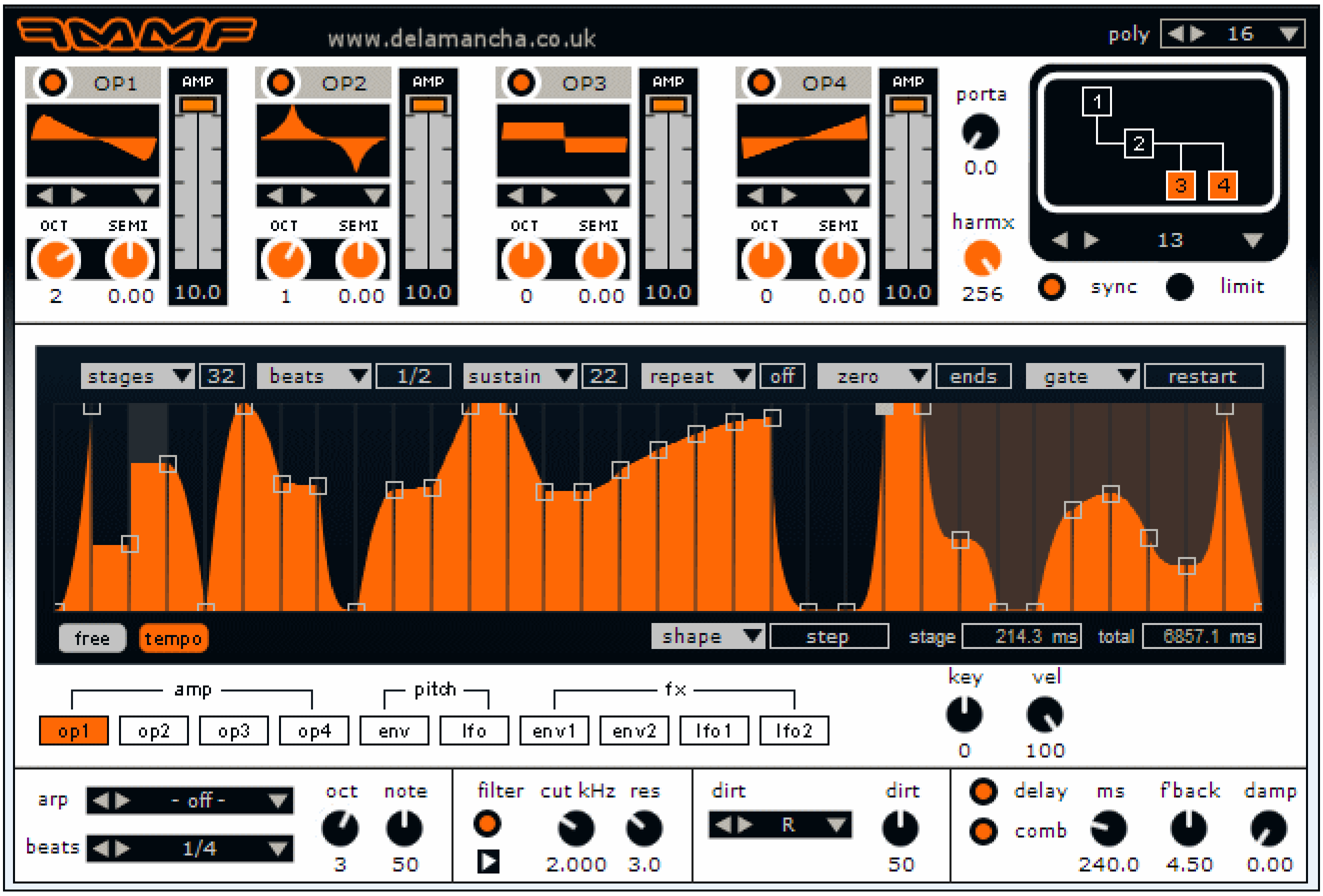


FMMF was developed by **de la Mancha** with presets by **brian botkiller**
for the **KVR Developer Challenge 2009**
It is a FM Synthesizer in VST plug-in format for Microsoft Windows based hosts.

:: Introduction

FMMF is a 4 operator FM (Frequency Modulation) synthesizer in VST plug-in format for Windows based hosts. It was written as an entry into the **KVR Developer Challenge 2009**, which is an audio freeware competition. If you have enjoyed using **FMMF** or another contest entry, please consider voting here:
http://www.kvraudio.com/developer_challenge_2009.php

FMMF has 2 primary aims

- Make FM synthesis as accessible and user-friendly as possible
- Use multi-stage envelopes and LFOs to make give lots of movement and modulation

The waveforms, algorithms and envelopes are all shown graphically, with the envelopes allowing dragging of nodes and clicking to change contour. Special attention has been paid to the envelopes to allow you to customise them in many ways, such as the number of stages, the contour of each stage, sustain and repeat points and tempo-sync. There are envelopes for the volume of each operator, the pitch, and the effects section (filter, distortion and delay) all with velocity and key tracking. Further modulation comes in the form of LFO's for the pitch and effects section, with many waveforms including randomising.

For further sound design potential you have a low pass filter, a multi-mode distortion and a flexible delay (with comb delay option). An arpeggiator let's you add even more movement to create rhythmic sequences.

The default preset bank was created by sound designer **brian botkiller** and shows off the potential of FMMF to make atmospheric pads and leads with lots of movement, big fat bass sounds, funky arps and oddball sound fx.

We hope you enjoy using FMMF as much as we enjoyed developing it!

:: Features

- 4 Operator FM Synthesis instrument with 17 FM algorithms
- 11 different waveforms per operator, with note sync option
- 7 Multi-segment Envelopes (up to 32 stages) for amp, pitch and effects modulation
- 3 LFO's for pitch and effects modulation
- All envelopes can be free or tempo-sync, each stage can have it's own contour
- Envelopes can have user defined sustain and repeat points with flexible locking and zeroing options
- LFO's can be free or tempo sync, with 20 waveforms and phase & note sync options
- Each LFO has it's own ADSR envelope
- Arpeggiator with 6 modes and adjustable tempo-sync, note length and octave range
- Resonant low pass filter with key and velocity tracking, can be modulated by envelopes or LFOs
- Distortion effect with 18 flavours, can be modulated by envelopes or LFOs
- Delay with comb delay option, size, feedback and damping, modulated by envelope or LFO
- Harmonics can be adjusted from 1 to 256, with limit on frequency beyond Nyquist
- Polyphony adjustable from 1 to 16 voices
- Portamento time
- 64 presets by brian botkiller covering many styles

:: Installation

Installation is simple, just extract *FMMF.dll* from the zip file and copy it into your VST directory. Install and load in your host program as you would any other VST instrument

To uninstall, simply delete the *FMMF.dll* file and the associated *FMMF* folder from your VST directory

:: FM Synthesis

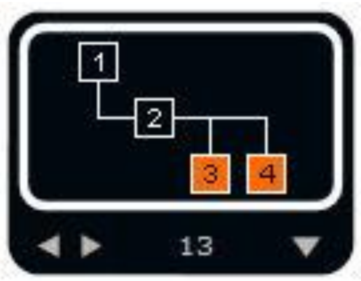
FM Synthesis has a reputation as being difficult to get to grips with, which probably comes from the painful process of programming 1980's hardware synths such as the Yamaha DX7 and TX81Z. Trying to program through a 2 line LCD screen with many layers of abbreviated menus using just a handful of multifunction buttons isn't exactly simple.

However, the actual theory behind FM isn't too complex and the software world allows much friendlier interfaces for sound design, which FMMF takes advantage of. There are many articles on FM Synthesis on the internet, so I won't try to explain the origins, technical detail or life enhancing properties here. Instead, I will try to explain in simple terms what FM is and how it works in this particular synth.

So, FM is Frequency Modulation, which simply means changing the pitch of the oscillator. That's it.

Well OK, there's more. Usually in subtractive synthesis, pitch is modulated quite slowly, by envelope, LFO, mod wheel etc. You can hear the pitch rise and fall to create sweeps, plucks or bends. In FM synthesis, the pitch of the oscillator is modulated very quickly by the audio output of another oscillator. This modulation is at audio rate, so it's too quick to hear it as a pitch bend or sweep, but it does change the tonal quality of the sound.

Each oscillator has it's own amp envelope, and together they are commonly referred to as an *operator*. **FMMF** has 4 operators, referred to as OP1 to OP4. An operator that modulates another is called a *modulator* (figures). An operator that is being modulated is called a *carrier*. Any operator can be both a modulator and a carrier at the same time, and can be routed so the output goes to audio. Any given routing set-up is called an *algorithm*. Sounds complicated? It's easier to see than to read, so like most FM synths, **FMMF** has preset algorithms, which are selectable as diagrams on the GUI.



In the example, algorithm 13 has been selected. You can see that OP1 is modulating OP2 which in turn modulates OP3 & OP4, which are both output to audio (highlighted orange)

OP1 is a modulator
OP2 is both a carrier and a modulator
OP3 is a carrier
OP4 is a carrier

These names don't really matter, what is important is that you can see which OP is modulating which and which ones are output to audio. More on this later.

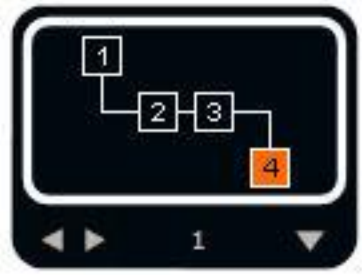
By changing the pitch of the modulator, a different tone is achieved in the carrier. Cleaner tones result from changing pitch by whole octaves, where semitone shifts give a rougher tone. The envelope of a modulator means that the amplitude of the modulation will change, resulting in movement of the carrier tone. An LFO can also modulate the pitch of the modulator so that the resulting tone of the carrier is modulated further.

That wasn't too painful was it? Your next step is to start fiddling with FMMF and see how the controls allow you to change the tonal output. But if you are feeling geeky and want to know more about FM synthesis, further detail and links on FM can be found by visiting Wikipedia: http://en.wikipedia.org/wiki/Frequency_modulation_synthesis

:: Controls

So, you have FMMF installed and loaded, you have an idea what FM synthesis is about and you heard the awesome presets, but now you want to tweak. Let's start with algorithms then, since we already mentioned them.

Algorithms



Here is a typical FM algorithm, number 1 out of the 17 different algorithms that FMMF has. To change algorithm, you have a number of options available;

- Click the left arrow to go down one
- Click the right arrow to go up one
- Click the down arrow or diagram to choose from a menu

The menu has the algorithm written in a descriptive way, but the diagram is much more informative. This algorithm for example is written as 1>2>3>4 which means that OP1 modulates OP2 which modulates OP3 which modulates OP4 which outputs to audio

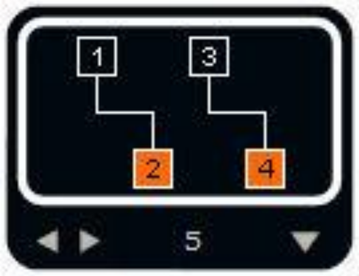
The algorithm display follows the following conventions

- Operators always modulate operators with higher numbers (OP1 can modulate OP2, but not the other way round)
- Operators highlighted in orange are output to audio

The algorithm description follows the following conventions

- ">" indicates the first operator modulating the second eg 1>2 means OP1 modulates OP2
- "+" indicates a parallel modulation or audio path eg 1+2 means OP1 and OP2 in parallel
- "(..)" is used as per mathematical formulas to indicate the order of the algorithm. The part between brackets should be carried out first before the part outside the bracket

The written description part is not so easy to decipher and it is generally better to scroll through the algorithm diagrams to find the one that meets what you are after (or randomly sounds good!)

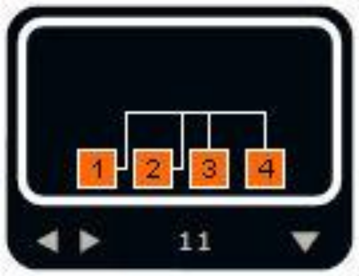


Meet algorithm number 5

You can see from the diagram that

- OP1 modulates OP2 which goes to audio
- OP3 modulates OP4 which goes to audio

The menu description is (1>2)+(3>4)



and now I present to you algo 11
(He's cool and likes to be called an algo rather than algorithm)

Here you can see that all 4 operators are output to audio and that

- OP1 and OP2 both modulate OP3
- OP1 and OP2 both modulate OP4

The rather cryptic description is ((1+2)>3)+((1+2)>4)

Operators

So, now we can see how the operators modulate each other, it's time to tweak each operator

Each operator has the same set of controls.

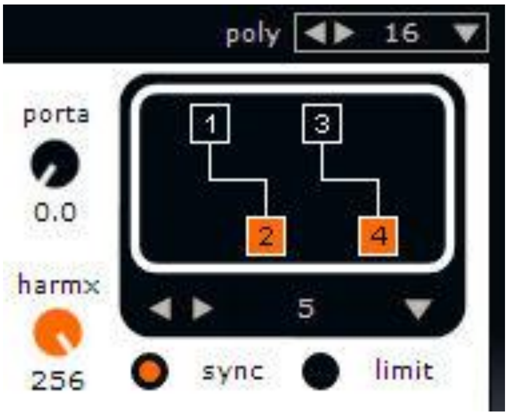
The round toggle switches the operator on/off. It can be useful for sound design or to switch off an operator that isn't needed.

In this example, the chosen waveform is sine, but there are 11 to chose from. Click on the waveform itself or the down arrow to see the list, or click left/right arrow to scroll through them

OCT and SEMI allow you to detune the pitch of the operator from the played midi note, in whole octaves and semitones, down to hundredths of a semitone. The range of the OCT knob is –5 to +5 octaves. The range of the SEMI knob is –12.00 to +12.00 semitones. You can double click the SEMI knob to reset it to zero. If the operator output goes to audio (highlighted orange in the algorithm diagram) then you will hear the pitch of note change from the played note. If the operator is a modulator of another operator, then changing the pitch in relation to the carrier will affect the tonal character of the carrier

AMP changes the amplitude of the operator waveform. If the operator output goes to audio (highlighted orange in the algorithm diagram) then this will change the volume of the audio. If the operator is a modulator of another operator, then the AMP slider will control the depth of the modulation.

Global controls



There are a few controls that affect all operators, regardless of the algorithm chosen.

Poly – (1 to 16) is the amount of polyphony, or the number of voices that can be played simultaneously. If set to 1, FMMF becomes monophonic, which means it can only play one note at a time. If two notes overlap, the first will stop playing when the second one begins. This can be desirable for some bass/sub-bass sounds for example, where you don't want chords to play.

Porta – is the portamento or glide time. When two notes overlap, this controls the amount of time for the pitch to glide from the pitch of the first note to the pitch of the second, creating a bending effect.

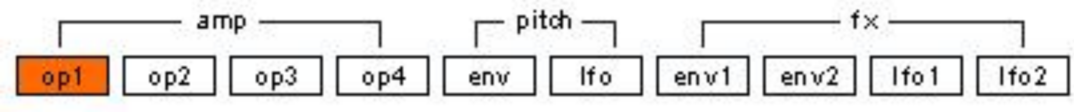
Harmx – (1-256) is the number of harmonics in the audio output of each operator. Usually you would want all harmonics for a fuller sound, but you can reduce the number as a sound design option.

Sync – ensures that the waveform for each operator starts from zero at each note-on message. When off, the operators are free running, and so may start inconsistently within the waveform at each note-on. Sync is useful for a consistent sound, especially at low frequencies like sub-bass or with percussive sounds with a short attack phase.

Limit – removes the root harmonics of operators 2, 3 and 4 if they exceed 20kHz. With more extreme frequency modulation, you can find the carrier frequencies are taken outside of the Nyquist range and can cause artefacts and aliasing, which are usually undesirable, but then again some people like that kind of thing.

Envelopes and LFOs

The envelope and LFO display take up a large part of the GUI, for a couple of reasons. These modulation options are the main focus of the FMMF sound design, and 32 stage envelopes need quite a lot of space to display properly. You can select which envelope or LFO is displayed using the row of buttons below the display area



Operator amp envelopes

Each operator has an amp (amplitude) envelope. This envelope will modulate the waveform amplitude each time a note-on message is received.

The envelopes can have from 1 to 32 stages, which can be free or tempo-sync in length. You can drag the nodes up/down and left/right to change levels and lengths, plus you can click any stage to change the contour. Let's take a look at some examples to see what the many and different options do.

This example is a standard ADSR envelope. This doesn't demonstrate the full power of FMMF's envelopes, but it is a familiar place to start from.

First thing to note is that it is a **free** envelope rather than a **tempo** envelope.

- Free means that you can freely drag the nodes so that each stage is a different length and is the most common form of envelopes found on synths.
- Tempo means that each stage is exactly the same length, expressed in beats, in sync with the host tempo

Here's a tempo sync envelope for example. You can see that beats has been set to ½, so each stage is exactly ½ beat long and the stage lengths cannot be different from each other. More on this later…

So, back to our ADSR example.

Along the top of the envelope, you will see a set of drop-down options.

- Stages – (1 to 32). In our example, there are 3. This is because the sustain stage is not counted as a stage with a length, just a node.
- Sustain – This is the node that will be held until the note is released. Here it is set to stage 2 (out of 3). So when a note is played, the envelope follows stages 1, then 2, then holds and when the note is released it follows stage 3. You can see the sustain node is filled in where the other nodes are outlines. Also the following stage is highlighted in a muted brown colour
- Repeat – sets the repeat loop that is followed during the sustained or held note. In our ADSR example it is off, but you can see in the previous tempo-sync example the repeat is set to stage 8, sustain as stage 15 and the loop from 8-15 is highlighted in the muted brown colour. When a note is held, instead of simply holding at the sustain level of node 15, it repeats stages 8-15 in a loop until the note is released
- Lock – Here you can lock the nodes in either axis or the total length of the envelope
    - Off – no lock active
    - Levels – locks the level of each node, allowing you to only change the stage lengths
    - Times – locks the length of each stage, allowing you to only change the node level
    - Length – locks the total length of the envelope, allowing you to only drag nodes within the total length
- Zero – forces certain nodes to be zero level
    - Ends – first and last node will be zero, usually the case for amp envelopes to avoid clicks on new notes
    - Last – forces the last node to be zero, ensures a release to zero, but a start at any level
    - None – all nodes can be any level
- Gate – determines the behaviour of the envelope when a new note is played before the last note has finished the release stage
    - Restart – the envelope restarts from the level of the first node
    - Pick-up – the envelope restarts from the level it was at in the release stage

  The pick-up method can avoid clicks, but is not useful for sounds which depend on the attack starting at zero

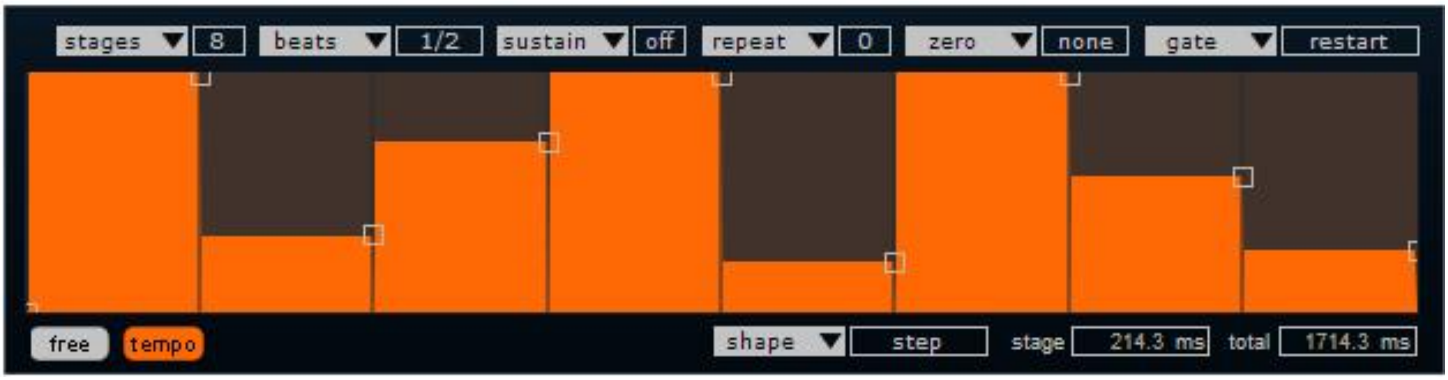
Along the bottom of the envelope, there are options for changing the contour and length of the selected stage as well as a display of the envelope total length. When you click on any stage of the envelope, it will highlight a dark grey colour and you can then change the shape and length in these boxes

- Shape – click the drop-down to change the contour curve of the stage. There are 14 different contours to choose from, including linear, exponential and S-curve. You can also left-click any stage on the envelope display to cycle through the shapes with the name of the shape updating in this box
- Stage – (ms) is the length of the stage in ms. You can type a precise number into this box to change the length of the highlighted stage, or drag the node left/right and this box will update
- Total – (ms) is the total of all the stages added together. You cannot type into this box, it is a display only for information. Obviously this does not include the sustain stage while the note is held, as this is only determined by the length of time the midi note is on and not the envelope itself

Right, well that's a simple ADSR envelope out of the way, now for some envelopes that show off a bit more…

So this one is a little more complex, with 7 stages. 4 before the sustain and 3 release stages. Note the lengths are locked so you can only change the levels of each stage. Of course you can unlock if you wish and re-lock again after

Here's a massive 32 stage beast, tempo-sync so every stage is a ¼ beat with a repeat sustain loop from stages 19 to 24. You can see the use of different shapes used for each stage

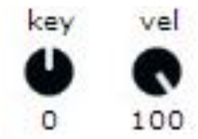


By using the shape **step**, you can make a step sequencer. Note there is no sustain and it repeats from 0, so the whole sequence just repeats over until the note is released.

Here is a simple decay envelope for percussive sounds. Only 1 stage and no sustain, so the note decays to zero over 45ms each hit. The S-curve gives a rounded sound, lovely! Note also the levels are locked and only the last node is zero

Key Tracking & Velocity Tracking

You may have noticed 2 small knobs just below the **amp envelope** display area



These knobs control the key and velocity tracking of the amp envelopes. They are expressed in percentages, from –100% to +100% and they both work in a similar way however slightly differently. You can double click the knobs to reset to zero.

Key tracking means that the depth of envelope modulation depends on the pitch of the played midi note.

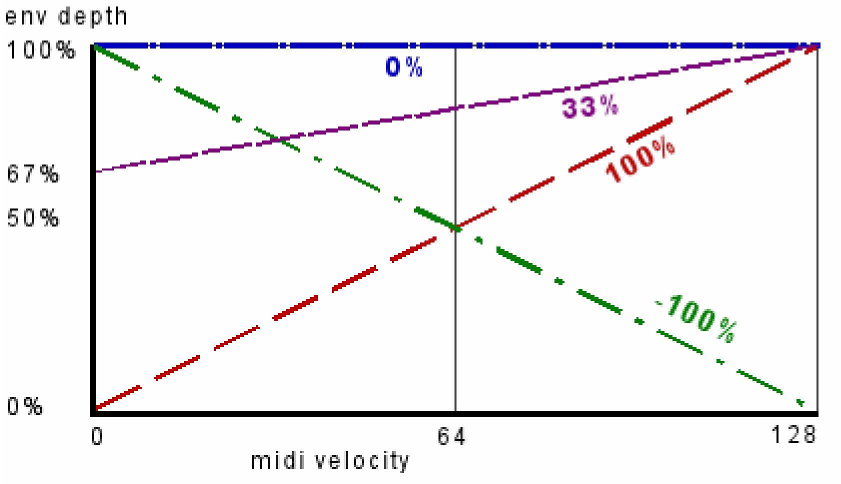
Velocity tracking means that the depth of envelope modulation depends on the velocity of the played midi note.

In the simple default case, **key = 0** means that there is no key tracking and **vel = 100** means that the harder you hit the note, the greater the envelope depth, which gives you standard behaviour for an amplitude envelope, soft note velocity gives quiet volume, hard strike = high volume.

Here's how they work in a bit more detail.

<u>Velocity Tracking</u>

We'll start with this one first as it is slightly simpler and more common (although usually fixed at 100%)



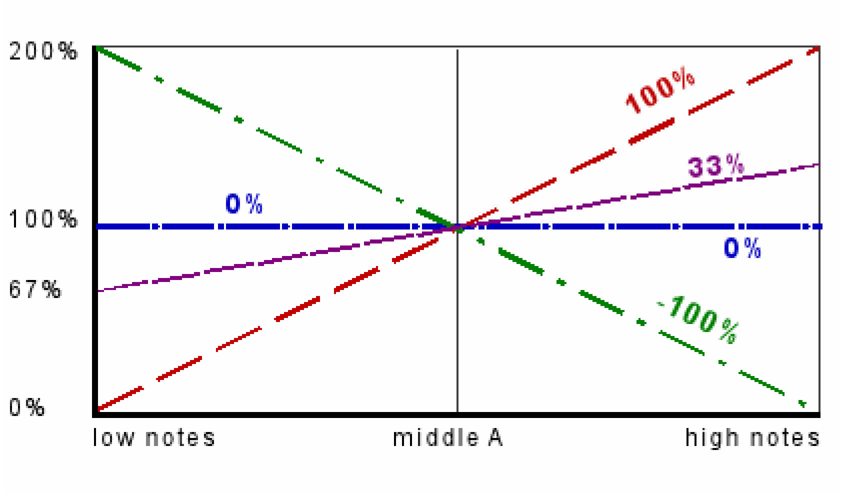
At vel = 0% (blue line), there is no velocity tracking, all midi note velocities result in an envelope depth of 100% This is usually the case for envelopes that do not need to be velocity sensitive, like for an operator that is only a modulator and does not output to audio,.

The more usual case is vel = 100% (red line), and is often fixed in many synths. This gives full velocity sensitivity and means that envelope depth is a linear relation to midi note velocity. At zero midi velocity, the audio amplitude is zero, at full midi velocity the amplitude is 100%

If you wanted a less drastic relationship, let's say you didn't want audio volume to be zero at zero midi velocity, an example is shown of vel = 33% (purple line). Here, the envelope depth still depends on midi note velocity, but it is never below 67% amplitude.

You can also have negative values, and an example is shown as vel = -100% (green line). This gives an inverse relationship, where the quiet midi velocities give high output and low midi velocities give high output. It would be unusual to use this for an amplitude envelope, but then I'm no stranger to unusual.

Key Tracking



In key tracking, the envelope depth can go as high as 200% (ie double the envelope value). Middle A is always at 100% envelope depth, so no matter what % key tracking you have, middle A always plays at 100% depth.

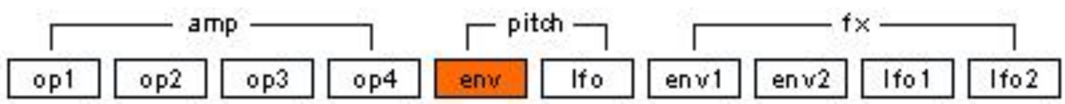
If key = 0% (blue line) then there is no key tracking and all notes will generate an envelope depth of 100% This is fairly standard envelope behaviour, but you might want higher notes to be louder or have a different filter cut-off.

If key = 100% (red line) then the lowest note will have zero envelope depth and the highest note will have 200% envelope depth. This would make high notes louder and low notes quieter on an amplitude envelope. Used on a filter cut-off envelope, it would make low notes have a low cut-off (duller) and high notes a higher cut-off (brighter).

A gentler slope is key = 33% (purple line) where the difference between the low and high notes is not as large.

An inverse slope is key = -100% (green line) where high notes would be quieter and low notes louder for an amp envelope. I know at least one person who would do this just because they can.

Pitch envelope
Further along the button menu, you'll find the **pitch envelope**



It is pretty much the same principle and controls and the amp envelope, with a few minor differences

Firstly, you can turn it off to save CPU if you aren't using it.

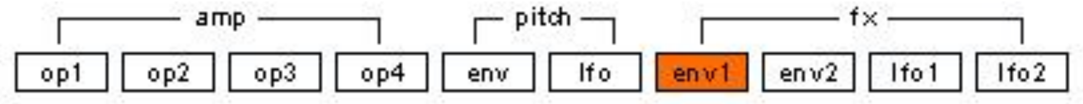
Click this little fella and when it is orange, the envelope is on and will display brightly. When off, the button is grey and the envelope display goes dim

Secondly, the one envelope controls the pitch of all 4 operators, but allows you to change the depth of each individually. This allows you to modulate the pitch of only 1 or 2 operators, or have each modulated by different depths. You can double click the amount knobs to reset to zero.

In this example, when the envelope is at 100% depth, OP1 is modulated up by 6 semitones, OP2 down by 12 semitones, OP3 up by 24 semitones and OP4 is not modulated at all.

Fx envelopes

There are also 2 **fx envelopes**, both are identical, allowing you to set up complex modulation of the same effect or 2 different envelopes for 2 (or more) effects.

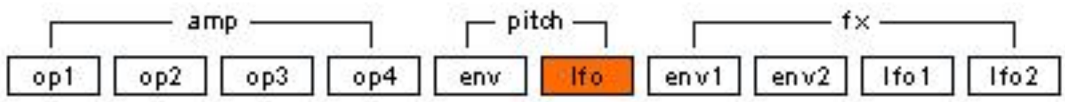


It is exactly like the pitch envelope, except the 4 amount knobs define the modulation of 4 different effect parameters

In the above set-up, when the envelope depth is at 100%, the filter cut-off is modulated up by 3 kHz, the filter resonance is not modulated at all, the dirt is modulated down by 50% and the delay feedback is modulated up by 2 units. We will cover the effects section in more detail later. You can double-click these knobs to reset to zero (seeing a pattern here?)

Pitch LFO

Under the pitch section, you'll see an **LFO** button next the the env button



An LFO is a Low Frequency Oscillator and is used to modulate parameters in a rhythmic manner (compared to envelopes which tend to modulate once per note or arhythmically)

Click the on/off toggle to switch the LFO on, or off to save CPU when not needed. When it is orange, the LFO is on and will display brightly. When off, the button is grey and the display goes dim

There are a few options to make some quite different LFO set-ups, let's take a look

This LFO has a triangle waveform, which is sync'd to start from zero with every new note and has a frequency that is tempo-sync to the host with one cycle every ¼ beat.

Each drop-down let's you change the way the LFO works

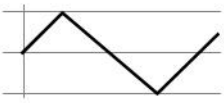
LFO – choose from 20 different waveforms, including random

Sync – when on, starts the LFO from zero each time a new note is pressed, for consistent modulation at the start of each note

Freq – choose from tempo or free.
When **tempo** is chosen, you select the length of one cycle in beats, from 1/32 beats (super fast) to 32 beats (super slow)

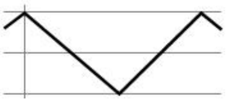
When **free** is chosen, a **freq Hz** knob appears so you can set the number of cycles per second.

The **phase** knob is shown for most (but not all) of the waveforms and is expressed from –100% to +100%
The phase determines the start point of the LFO along the waveform
The phase length for 1 cycle is 360 degrees, so 50% is 180 degrees.



Phase = 0%, waveform starts at 0 degrees

Phase = +50% means it starts at +180 degrees (180 degrees late)



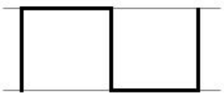
Phase = -25% means it starts at –90 degrees (90 degrees early)

You can double-click the phase button to reset to zero

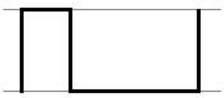
The phase knob is replaced by another parameter for 2 of the waveforms

- Pulse – knob becomes a **width** knob, in %
- Rnd glide – knob becomes a **glide** knob in %

Width determines the width of the pulse. A **square** waveform is the same as pulse with 0% setting for width. If you move the pulse width to negative values, the low part of the pulse becomes longer and the high part becomes shorter. If you move the pulse width to positive values, the low part becomes shorter and the high part becomes longer.



Width = 0%, Square wave with equal high and low parts



Width = -50%, Pulse wave with short high part

Width = +60%, Pulse wave with long high part

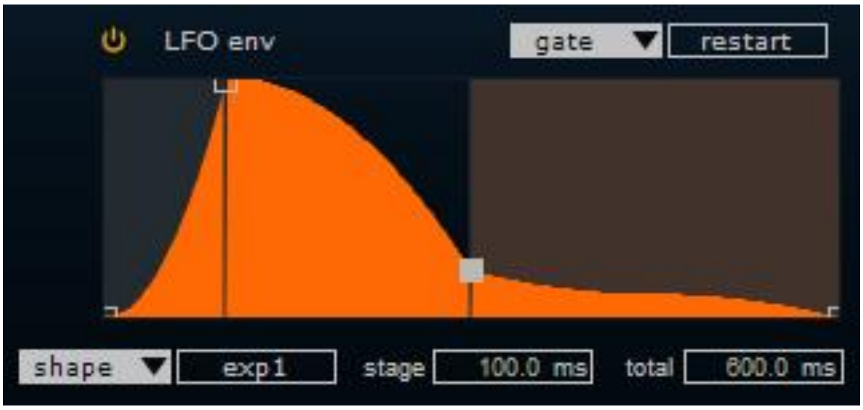
Glide determines how smoothly the LFO moves from one random value to another.

The waveform **random** will move from one value to another instantly, which can be desired but sometimes you just want it to wander aimlessly, not jerk around. This is where **rnd glide** comes in. Increase the glide value to make the transistion from one value to another smoother. Around 90% is about as far as you want to go, otherwise it becomes sooo smoooooth that it doesn't actually change. Oh, and negative numbers don't make it rougher, they just do the same a s positive numbers. Don't ask.

You can double click the width or glide knob to reset to zero (I bet you knew that by now though)

LFO envelopes

Each LFO has it’s own envelope to modulate the depth of the LFO for each new note. This can allow you to make the LFO subtle during the attack and kick in during sustain, or be full-on at attack and fade out during decay, or whatever crazy scheme your mind can imagine.



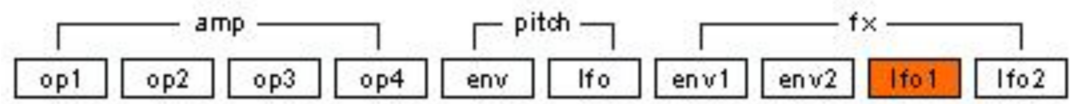
The LFO envelope is a much simplified version of the envelopes described earlier in the manual. Click the on/off toggle to switch it on and the display brightens. Click off to dim the display and save CPU if not needed.

The LFO envelope is an ADSR envelope, drag the nodes to change level and time, click the stage to change shape. For more detail, see the previous section of the manual on envelopes.

FX LFO’s

Just as the pitch LFO, but the **fx LFO** modulates the same 4 fx parameters as the **fx envelopes**

Select either fx LFO in the button menu below the display



You can determine the depth of modulation the LFO performs on each parameter by setting the knob values. The modulation is +/- the knob value, so a negative knob value is 180 degrees out of phase with a positive one

In this example, the LFO modulates the filter cut-off by -/+ 3 kHz and +/-25% dirt. Filter resonance and delay feedback are not modulated. As one of the knobs is negative and the other is positive, they are out of phase with each other. As the LFO starts to modulate, the cut-off moves down as the dirt moves up. When the LFO changes direction cut-off moves up and the dir moves down. This can be interesting to phase 2 different effects in and out.

You may have guessed that you can double-click these knobs to reset them to zero.

Both fx LFO’s are identical and allow you more flexibility in modulating different parameters.

Effects section

There are 4 effects available to add to whatever you might have cooked up already.

- Arpeggiator
- Low pass filter
- Dirt
- Delay

The latter 3 can all be modulated by their own envelopes and LFO's, the first however is a rebel and refuses to be influenced by others.

Arpeggiator

Known to it's friends as **arp**, this fiddles with the incoming midi signal to generate tempo-sync'd midi sequences across several octaves. Often used for *tarnce* music, but can be used to make interesting sequences and 8-bit game sounds too.

You have 6 different arp modes to select in the drop-down list (or use the left/right arrows to cycle through). You can simply make your arp go up or down, do both or get tarncey or fancy if you please. Or choose **- off -** if you aren't in the mood tonight.

The arp is tempo-sync'd to the host. The **beats** setting determines how often a new note is played by the arp (yes, that's right, you don't need to add a tiresome set of midi notes to your host piano roll, FMMF will play gazillions of arpeggiated notes all by its ownsome). Use the drop-down or left/right arrows to choose from 1/32 beat (über-fast) to 4 beats (oh-so slow).

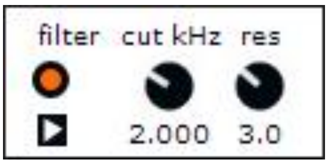
Use the **oct** knob to select the number of octaves to arpeggiate over (1-4). 1 will give you a gated on/off effect, 2 gives you that 8 bit game sound, 3 and 4 are for perverts.

Note is the length of each arpeggiated note, from 0% to 100%. This is a percentage of the total time between notes, so 50% would get a nice balance, 80% would get longer notes, 25% would get you quick blips. 100% gets no gaps between notes and 0% gets you no notes between gaps (a.k.a. silence)

Filter

Lots of 80's hardware FM synths didn't have filters at all. FMMF isn't an emulation so I threw one in there. So sue me.

The FMMF filter is a Low pass filter. As the name suggests, it lets low frequencies below the **cut-off** pass through and cuts out the higher ones. It is a resonant filter, so it boosts the frequencies around the cut-off, depending on the **res** knob setting.



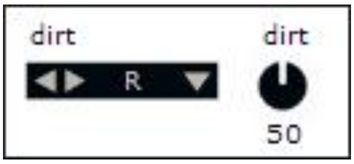
Click the toggle to switch the filter on (and off to save CPU when not needed). Adjust the **cut-off** to only let frequencies below the cut-off pass through and the **res**onance to boost the frequencies around the cut-off frequency.

But wait, what's that little arrow doing there? (To be honest, it's because this feature was added after the GUI size and layout was too far gone to change it, but let's pretend it's some kind of *expert* setting and we'll both feel good about using it)

Click it and you'll see it reveals the filter key tracking and velocity tracking knobs. Please refer to pages 9 and 10 of the manual where this is explained in more detail for the amp envelopes. Instead of *envelope depth*, replace with *cut-off value* and you should be good. Basically it allows the filter cut-off to be greater or lesser than the knob value, depending on the note pitch and/or velocity.

Dirt

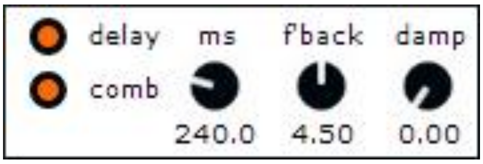
The dirt section is a multi-flavour distortion effect, with 18 different types from subtle saturation through even harmonics and overdrive to evil ear-bludgeoning. Choose your weapon carefully.



Use the drop-down to select which flavour of distortion you want (unhelpfully labelled A to R but I really don't have 18 useful names). Dial-in from 0 to 100% dirt to give anything from subtle to dangerous.

Delay

The delay-lay-ay-y effect gives you echoes of the sound you play.



Click the toggle to turn on/off and also to select the comb delay option. The comb option adds a delayed version of itself, which causes some peaks and troughs in the spectrum (resembling a comb). Easier to hear than to describe, so switch it on to hear what it does.

The **ms** knob determines how long between echoes in ms, short for quick repeats-s-s-s and fattening up, long for big, cave-like echoes-choes-choes-oes-oes-s-s. The **feedback** knob determines how strong and long (or how many) echoes you will hear and the **damp** knob will damp down the number of echoes, like a big, fluffy cushion.

This part of the page is unintentionally blank. You can use it to write notes on the different KVR DC09 plugins you are testing before voting, write a list of top 5 synth manuals to post later on KVR or just draw a picture of a dinosaur attacking a suburban town centre.

:: End User Licence Agreement (EULA)

:: Credits

Many thanks to **brian botkiller** for the awesome presets that show **FMMF** off to its full potential. Thanks also to **Jeff McClintock** for creating SynthEdit and to the 3rd party SE module developers, without which this plug-in wouldn't exist.

Links	
brian botkiller	http://brianbotkiller.com/
SynthEdit	http://www.synthedit.com/
Dave Haupt Modules	http://www.dehaupt.com/SynthEdit/semodules.htm
Chris Kerry Modules	http://www.chriskerry.f9.co.uk/
K D Lynch	http://www.rubyhex.com/synthedit/
Scoofster Audio	http://scp.web.elte.hu/synthedit/modules.html
Daz Disley	http://www.roughdiamondproductions.com/SE/

:: About the team

de la Mancha lives, eats, dreams and breathes VST plugins, seeking to bring randomization and modulation to the masses. He is also a producer of odd-skool breakbeat, downtempo glitchy beats and other assorted bleeps and noises. You can find his music at www.papadodo.co.uk www.3x0.co.uk and www.mono-log.co.uk

Sign up for the newsletter if you want to be kept in touch about plug-in releases and updates: www.delamancha.co.uk/contact.htm

brian botkiller is an electronic musician, sound designer, producer and performer from Albuquerque, NM, USA. Info and music can be found here http://brianbotkiller.com/ and here http://www.myspace.com/brianbotkiller