

# The FM One

FM Synthesis

Graphical Envelopes

LFOs and Key Follow

Global Parameters

Operator Switch (Algorithm Edit)

Individual Operator Parameters

Common Operator Parameters

Connections



Return  
To Main  
Table Of  
Contents

# FM Synthesis

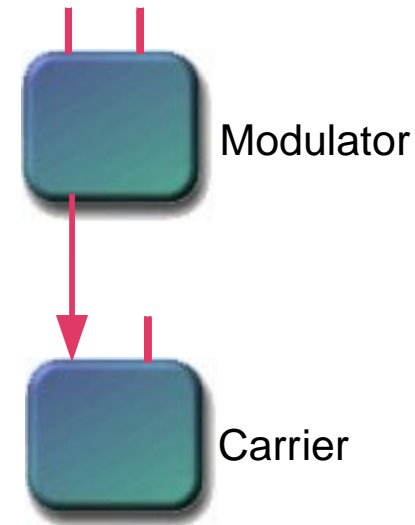
Prof. Dr. John M. Chowning discovered in 1967 that modulation of the frequency of a simple sine waveform by another sine wave results in complex sound spectra. FM means precisely Frequency Modulation – but what does *that* mean?

Quite simply, a sinewave oscillator produces a sinusoidal tone with a specific frequency. Changing the frequency by manually adjusting the oscillator's frequency control is a basic form of frequency modulation.

If this oscillator is outfitted with an amplitude envelope and one or more frequency modulation inputs, it becomes an FM operator. An operator can modulate other operators as well as be modulated.

A DX7 has six such operators which can be connected together in 32 different configurations or algorithms. This set of possibilities allows a large library of sounds to be created.

FM synthesis involves a large number of parameters, but the creation of an FM sound can be explained on a basic level according to a few simple principles.



An operator can be used either as a carrier or as a modulator. The simplest configuration – not FM at all – is a single carrier (with no modulator), producing a simple sine tone corresponding to the note played on a keyboard.

The simple FM circuit in the above diagram can create complex sounds via frequency modulation of the carrier.

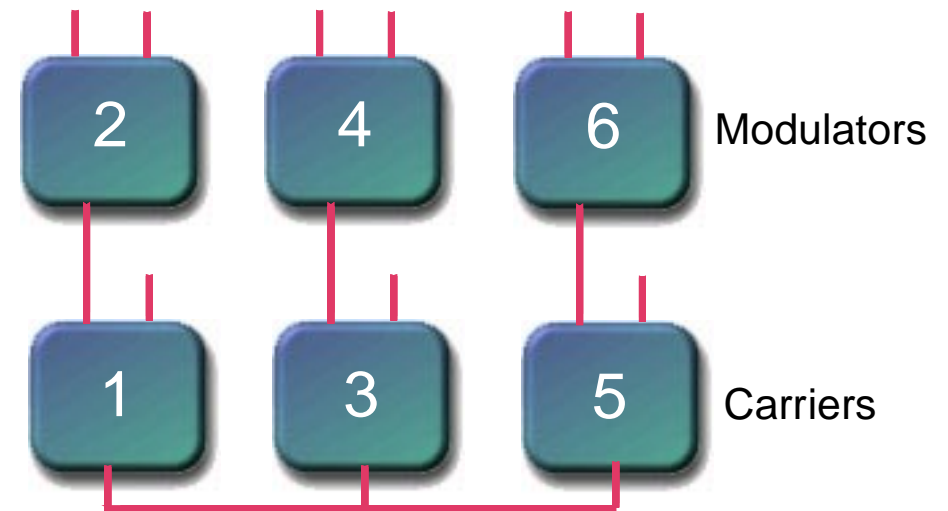
If an amplitude envelope is applied to the carrier, it affects the overall level of the generated sound. On the other hand, applying an amplitude envelope to the modulator causes the tone of the sound to vary with time. This is the basic essence of FM synthesis.

Actually, in FM synthesis there are few truly dependable statements which can be made regarding the effects of parameter changes on the sound. The results of changes in the amplitude or pitch of an operator depend upon the frequency relationship between the modulator and the carrier. Practically the only thing which can be said with any certainty is that the amplitude of a carrier directly affects the level of the composite sound.

In the six-operator circuit shown at right, each of the three carriers (operators 1, 3 and 5) determines the level of the sound generated by its branch of the algorithm.

This algorithm is actually three separate branches, each containing a single modulator/carrier pair, whose outputs are combined. Each branch can independently generate its own sound. For example, the first branch could generate a key-strike sound, the second a sustain-phase sound and the third a release-phase sound. The amplitude envelopes of each carrier are adjusted to create the transition from one phase to the next.

Another possibility is to generate similar sounds in each branch but slightly detune the branches from one another via the carrier frequency to result in a "fatter" sound.



Algorithm 42

# Graphical Envelopes

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

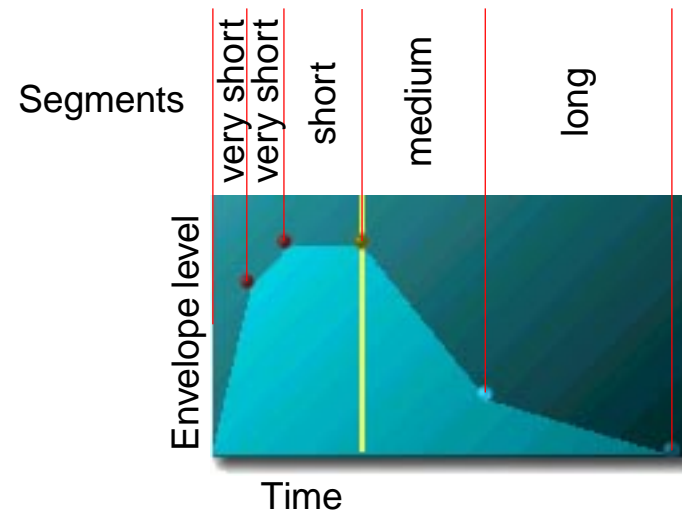
## Segments And Points

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

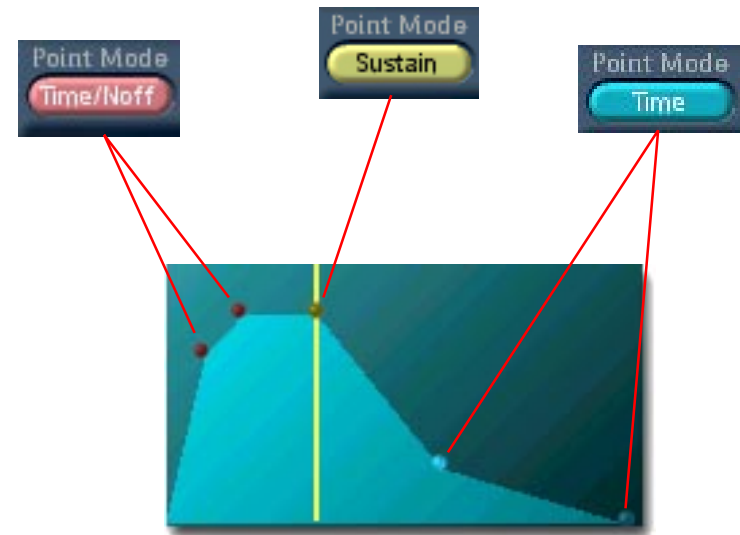
Clicking on the Point Mode button switches the selected envelope point (and its segment) between **three modes** which allow for versatile envelope construction:

**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.



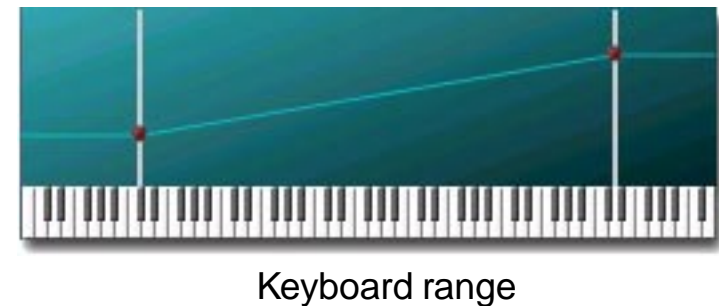
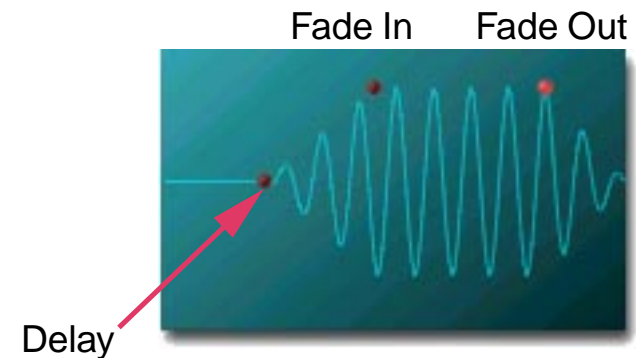
# LFOs and Key Follow

A **graphically-editable LFO** has four parameters which are adjusted via three points.

The first point sets the LFO **delay time**. The LFO waits this amount of time following a Note-On event before beginning its fade-in. The second point simultaneously sets both the **final LFO level** (after the fade-in) and the **fade-in time**. The third point sets the duration of the LFO **fade-out** which begins when a Note-Off event occurs. This point can also be used to set the final LFO level.

**Key follow**, or keyboard tracking, causes a parameter to vary depending upon which key on a keyboard is played. The key follow curve at right, applied to a low-pass filter cutoff, would cause the filter to close far down when low notes are played and be open fairly wide with high notes. Notes played on other keys would produce a gradual variation between these two extremes.

The vertical lines are curve *breakpoints* which can be moved side to side on the keyboard. As shown in the diagram, the curve slopes linearly between the red points which ride up and down on the breakpoints. Above or below the breakpoints, the curve is flat. The red points can be raised or lowered to set the *slope* of the curve between the breakpoints. The two points are coupled – raising one lowers the other by the same amount – making them both effectively part of a single slope setting.



# Global Parameters

The **Global Parameters** (located in a drawer at the top of the **Operator surface**) include settings affecting the FM synth as a whole. Included here are **MIDI channel** and **Pitch Wheel Range** settings, coarse and fine tuning (**Transpose Semitones** and **Cents**), and **Portamento** controls.

The **Portamento Time** (Portamento/Glissando) control adjusts the speed of up/down pitch glide from one note to the next which is activated when the **Portamento Type** is set to a value other than 0 ("Off"). Both **Portamento** (smooth glide) and **Glissando** (glide in semitone steps) are available, each in two modes. In "plain" mode (settings 1-2), the pitch glide between two notes always occurs. In **Fingered** mode (settings 3-4), the glide occurs only when a new key is played *before* the previous key is released.

In the **Aftertouch** and **Velocity** drawers, you can tweak the curves which adjust the response of the FM synth to the corresponding MIDI values. Editing of these curves is fairly simple. The **Type** control at left selects one of six basic curve types (e.g. linear, logarithmic, etc.), the **Angle** control at center produces variations on the chosen curve type and the **Offset** control at right produces an overall shift the curve. (If that didn't completely make sense to you – never mind, just go ahead and try it. It's much easier to use than to explain.)





# Operator Switch (Algorithm Edit)

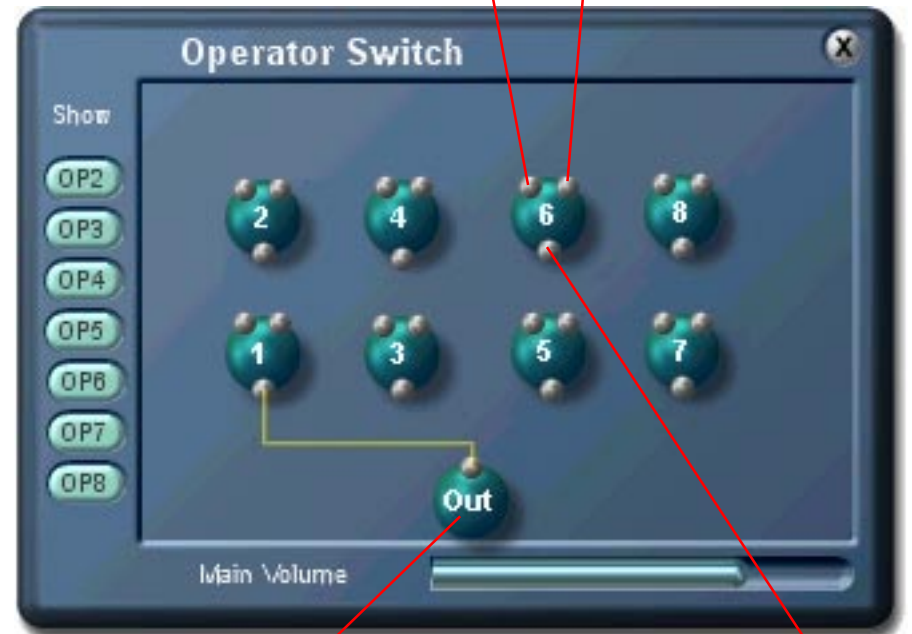
As discussed at the beginning of this chapter, the operators of an FM synth can be variously interconnected to construct FM *algorithms*. In the FM One, the eight operators can be interconnected more or less without restrictions, resulting in a very large number of possible algorithms.

The **Algorithm** button at upper left in the *Operator Surface* (a picture appears in the next section) opens the **Operator Switch** control panel, in which you can view and edit the algorithms of the FM One.

Each operator is depicted as a small sphere. In turn, each of these small spheres has three tiny spheres on its surface. The upper two represent the operator modulation inputs, while the one at bottom represents the operator output. These are connection points similar to module connection pads in the Project window and are used in the same way. By clicking on them, you can make or break point-to-point connections to custom-construct an algorithm. The connection from operator 1 to the **algorithm output** is a fixed connection which cannot be altered or removed.

The **Show** buttons let you switch individual operators on or off. Only *unconnected* operators can be switched off (hence no switch for operator 1, which is always

Operator modulation inputs 1 and 2



Algorithm output

Operator output

connected). Switching off an unconnected operator is a good idea – it avoids unnecessary consumption of DSP capacity.

The **Main Volume** slider controls the output level of the algorithm (and thus the overall output level of the FM One, when using this algorithm).

To clear up the picture, the operator spheres can be dragged around as desired within the control panel.

# Individual Operator Parameters

All sound settings for the FM synth are carried out via the **Operator** surface. On this surface, some parameters apply only to the selected operator, while other apply to all operators in common. In this section, the first group is described.

The FM synth has eight operators. Settings for a specific operator can be accessed by selecting the operator via the small window at the top of the Operator surface.

**Coarse** and **Fine** are used to adjust operator pitch. *Coarse* allows wide-ranging settings in integer steps between 0.5 and 32, where a factor of two change (from 1 to 2, or from 4 to 8) corresponds to an octave. *Fine* can dial in one more octave on top of the *Coarse* setting, in extremely fine steps. **Detune** allows a still-finer tuning adjustment of +/- 20 cents.

In conjunction with the key number of a note, these controls determine the pitch generated by an operator – as long as *fixed frequency* mode is not active. Setting the **Fix Freq** setting to a value other than zero activates this mode, in which operator pitch is set directly in Hz and is not affected by the key number of a note. In this mode, the *Coarse*, *Fine* and *Detune* settings have no effect.



The starting **phase angle** of the operator is adjustable via the fader directly below the small waveform display window.

**Ret** (retrigger) controls whether an oscillator is restarted upon each new note (according to the *Phase* setting) or runs continuously. This option applies only to fixed frequency mode.

The **Env D** and **LFO D** (depth) controls allow the effect of the **Common Pitch Envelope** and **Common Pitch LFO** to be adjusted for each operator.

Each operator has two modulation inputs, whose sensitivity is adjusted via **Gain 1** and **Gain 2**. Below these are the operator **amplitude envelope** controls.

For detailed information about **editing of the envelope** itself, including the use of the **Point Mode** and **Loop Point** controls, refer to the *Graphical Envelopes* section earlier in this chapter.

The **Time** and **Level** windows display the values for the currently selected envelope point. Values can be entered directly into these windows from the PC keyboard.

If there is a loop in the amp envelope, the **Loops** control can be used to specify how many times the loop should repeat.

The **Pressure** and **LFO D** (depth) knobs control the amount of effect which MIDI aftertouch and the *Common Amp LFO*, respectively, have upon operator **amplitude**.

The **Out** control sets the operator output level. The **output level** can also be affected by **key position**. This is adjustable in the **Output Key Scaling** drawer (next page).

The amp envelope can additionally be modified via *key follow* and *note velocity*.



In the **Amp Envelope Key Follow** drawer (see next page), the influence of **keyboard position** upon amp envelope **times** can be adjusted.

The **Time** control in the **Velocity** drawer allows the effect of **note velocity** upon amp envelope **times** to be adjusted. The **Level** control sets the amount of effect which **note velocity** has upon the overall amp envelope **level**.

**Output Key Scaling** allows you to specify a curve to control the variation of an operator's output level based on keyboard position. Among other things, this is useful for creating keyboard split or crossfade patches in which two or more distinct sounds appear across the keyboard, or for controlling changes in the timbre of a sound across the keyboard.



**Amp Envelope Key Follow** lets you control how the amplitude envelope times of an operator will vary with keyboard position. An obvious use for this (of course there are other uses) is to mimic the typical characteristic of acoustic percussion and plucked-string instruments in which higher-pitched notes decay more quickly.



# Common Operator Parameters

This group of settings affects all operators in common.

The **Common Pitch LFO** is graphically editable (refer to the *LFOs and Key Follow* section earlier in this chapter). In addition, LFO **frequency** and starting **phase** can be edited via the sliders below the LFO edit window.

The **Options** drawer contains a few more LFO controls:

The tiny **Wave** window allows LFO waveform selection. Left-click and hold here and move the mouse up and down to change the selection.

The **Velocity** control sets the amount of effect which note velocity has upon LFO **depth**.

The **Retrigger** button controls whether the LFO is retriggered (per the specified *starting phase*) with each new note or continuously free-runs.

The **Common Amp LFO** permits modulation of voice output level. This LFO has controls identical to those of the *Common Pitch LFO* described above.





The **Common Pan Modulation** drawer contains options allowing modulation of voice pan position:

The **Common Pan Envelope** is a *bipolar* envelope with controls largely similar to those of the operator amp envelope, including the options for velocity modulation of envelope levels and times.

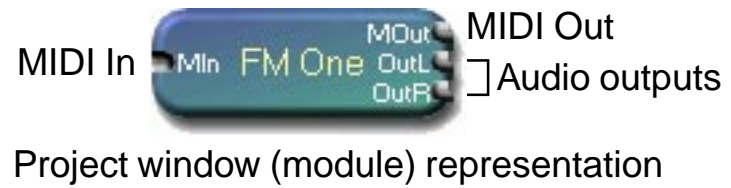
Alternatively, pan modulation is possible via **Pan Key Follow**, which allows a sound to be spread left-to-right based on key position.

The **Mod Source** switch determines which of these two pan modulation sources is active.

The options in the **Common Pitch Modulation** drawer affect the pitch of all operators in common. These options are similar to those of the *Common Pan Modulation* drawer described above, except for the absence of the Mod Source switch (both envelope and key follow modulation of pitch are available simultaneously).



# Connections



Minimized (icon) representation