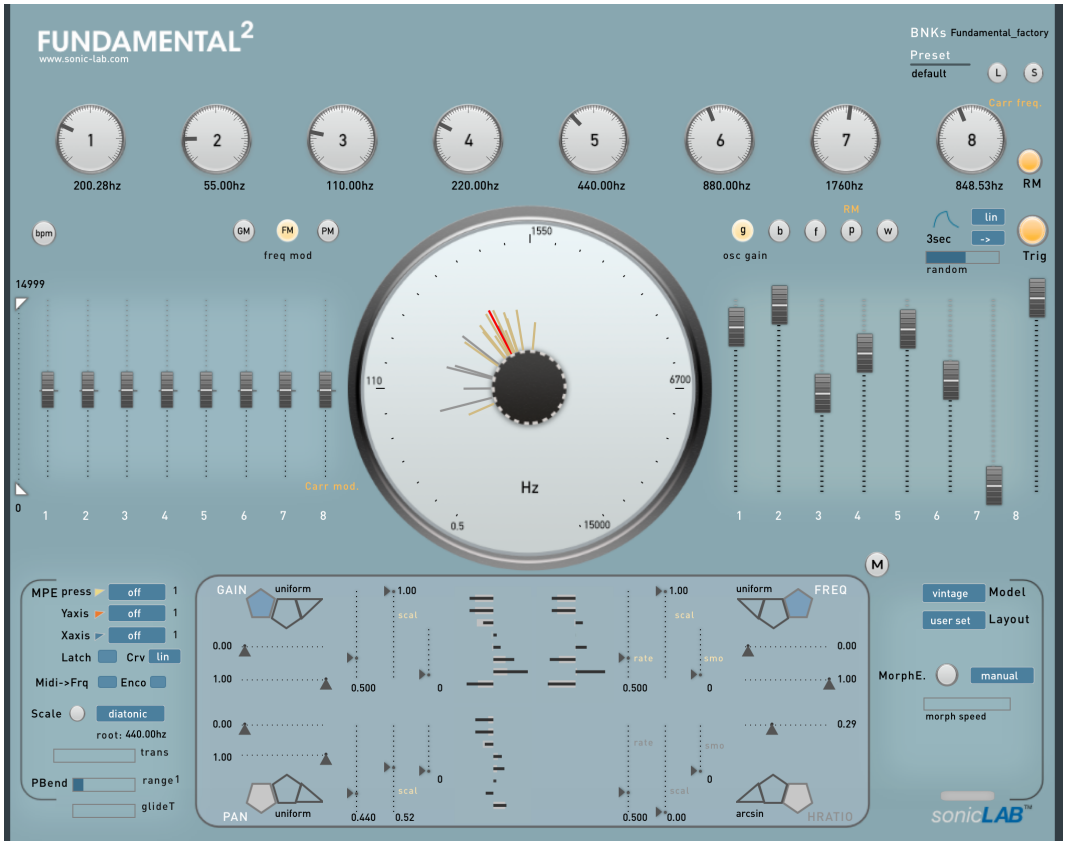


Fundamental2 iOS is designed&programmed by Sinan Bökesoy.

www.sonic-lab.com

installation of Fundamental2 iOS

Fundamental2 iOS comes as a standalone and a AUv3 app for your iPAD only, which you can purchase and download on the AppStore.

If you own the Fundamental v1 app, it is strongly recommended to backup your custom banks created there. You will be able to use them on Fundamental2 as well.

We recommend at least an A12 processor iPAD and the small screen iPAD's can be challenging for the usage of the app. (such as the iPAD mini)

Fundamental2 uses the resources for best sonic quality. The audio engine calculations are done continuously and even partly if there is no audio output. sonicLAB audio engine design does exclude optimisation methods applied to favour the compatibility of old iOS hardware while degrading the audio quality.

Fundamental2 uses the full screen height automatically on the standalone version. The AUv3 version also tries to use the full window height offered by the hosting app.

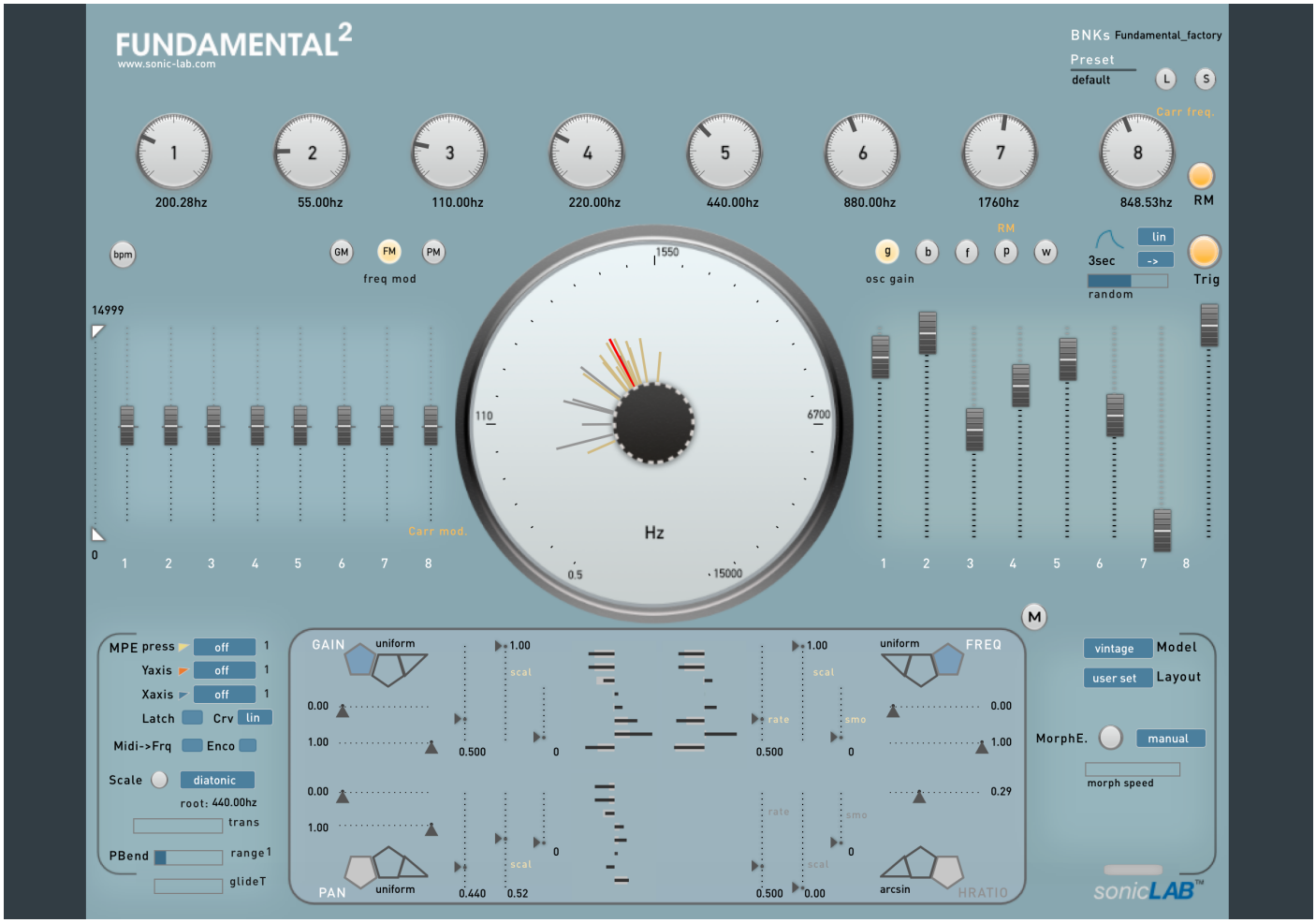
Fundamental2 creates and uses the following directories in the app application domain: **PresetBanks**, and **scl** .

Deleting the app will also delete these directories, so you should create backups of your custom banks and export them to another place using a file handling utility on your iPAD. More info you can find on the preset / bank handling chapter.

some important issues :

- Fundamental uses sample level precision needed to control in real time all these parameters. Internal operations are normalized for 96kHZ. Likewise recommended sampling rate (if available) to use for best sound quality is 96kHz.

- Around 3800Hz there will be a transition to mathematical sine wave from the vintage sine wave. The explanation you can find at the Appendix section.



What is Fundamental ?

sonicLAB commitment in audio development is combining art, science and craftsmanship in software form. We believe that the Fundamental2 connects the present with the past and the future of sound design tools. That significant role leads exactly in how we should describe it.

Let’s retouch and scratch a bit the surface of electronic music history. EM in the second half of the 20th century has been spread at different hands, centers and directions beginning with the 50's. Serving the material to composers requirements and that being limited within the technology available of those times, has certainly influenced also the genres following these efforts. There has been cross disciplines and paths which have justified that there can be no monopoles, neither a single methodology nor a certain technology providing all the needs of producing electronic music.

Much of the electronic music technology has been actually a migration/transformation of electronic equipment development from other disciplines and practices but the composers/researchers have created their labs in order to incorporate their multidisciplinary approach, knowledge to produce a new sonic universe by using these tools.

For instance, the famous GRM / Radio France studio through the direction of Pierre Schaeffer was handling concrete recorded sonic material with tapes machines, introducing analogue processing methods to combine and recompose them into a structured body after a tedious work flow. Magnetic tapes, tone mixers were the analog equivalent of basic tasks of today's DAWs. Many composers of that time including Stockhausen and Xenakis (who have moved on in different directions later) have visited the GRM.

Around the same times, the studio for electronic music WestDeutscheRundfunk (WDR) has been established by Herbert Eimert and later with Karlheinz Stockhausen (invited composer and later the director) has brought the need of sine-wave generators / beat-frequency oscillators from the testing and calibration department (as well noise generators, band pass filters and tone mixers and oscilloscopes for visualization) for the purpose of using sine waves to artificially build a custom sonic spectra to constitute the basic sonic elements for Stockhausen.

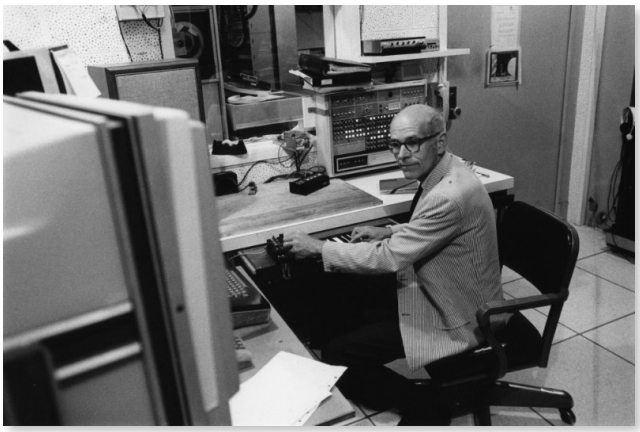
Stockhausen, the best student of Messiaen along with Pierre Boulez has been the leading electronic serial music composer, where not just the pitch duration and onset time has been structured with serial approach but also the timbral composition. By using the available test equipment tone generators he could juxtapose his preferred serial combination of sine wave frequencies as distinctive bodies put in series in his electronic music studies. In Studie II, he has created clusters of 5 sine waves each by calculating by the formula 25âˆš5 to create a 81 tone scale.

100	=	100	→	100 Hz
100 x 25√5	=	106.649494220837	→	107 Hz
106.649494220837 x 25√5	=	113.74114617560345	→	114 Hz
113.74114617560345 x 25√5	=	121.30435711726397	→	121 Hz
121.30435711726397 x 25√5	=	129.37048333339991	→	129 Hz
129.37048333339991 x 25√5	=	137.97296614612284	→	138 Hz

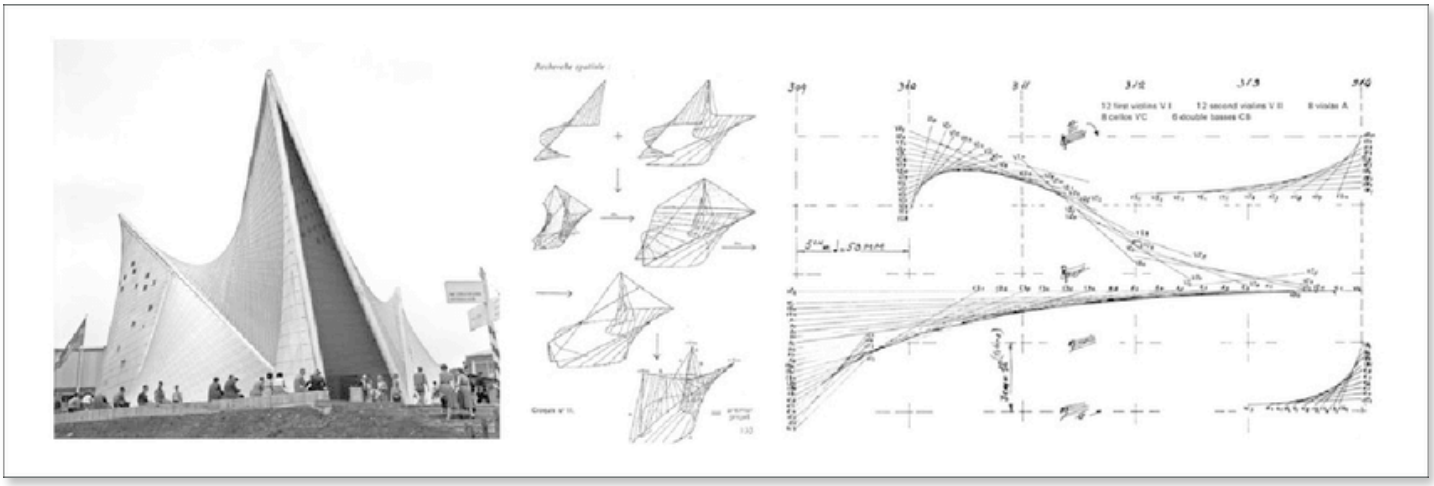
Not surprisingly, the test equipment delivered in use of compositional purposes came also with certain limitations. For instance Stockhausen could only handle integer numbers of sine wave frequencies. As you can see above the calculated numbers have been rounded and this has certainly changed the beating character between the waves.

Furthermore the test equipment have no modulation capabilities (the amplitude envelope has been in fact a subjective handling of Stockhausen). Later other composers have used these tools within other musical approaches, also Stockhausen has liberated himself from serial music in his later electronic music compositions by combining other tools.

Accidentally it was also around the same times of the first computer systems and the first experiments at Bell Labs, where the first programming languages for sound generation by Max Matthews and the composer Jean Claude Risset as a co-worker of him has been conceived; the foundations of computer music. Although being the very early stages of digital computing, this gave a big boost in study of sound and the science of it. Two decades later Ircam in Paris would play a primary role for developing the software tools especially for spectral processing, real time processing by inviting regularly the (now legendary) computer music scientists from the US in order to collaborate with the contemporary composers of that time. Jean Claude Risset has also been the lead of the computer department at that time.

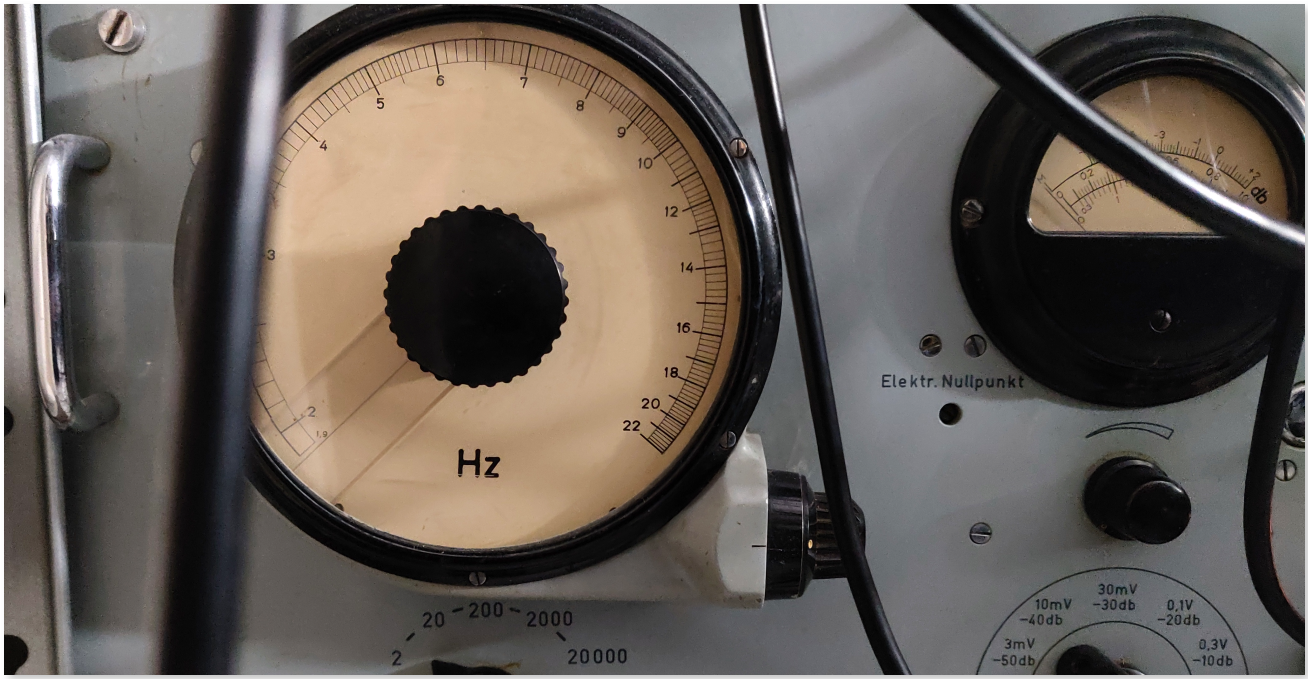


A true pioneer of "computed" music in the 20th century, Xenakis has developed with his self-efforts a remarkable combination of mathematics, sound physics, composition and architecture. Beginning late in the 1950s he proposed the use of stochastic methods driven by pure mathematical distribution functions for the constitution of many musical components. Hand calculated results were mapped to structural components of the music (notation, instrumentation, gestural information, sonic emergence through the mass vertical/ horizontal movement of layers) were inviting in a sense the forces of nature and its complexity between forms of order and disorder. This multi disciplinary approach has taken him a full career of inspiration which has followed later also on the digital hardware / software domains of applications.



Back to the question : ***What is Fundamental ?***

Fundamental is a sound synthesis software tool combining the different roots of electronic music history. Fundamental wave engine is a faithful sonic recreation of a vintage Rohde&Schwartz tone generator (having been widely used at WDR) and leaving behind the limitations of the original device as such proposing complex modulation possibilities by mapping the sonic parameters like frequency , gain, stereo panning etc. in continuum, which has not been applicable until now.



A test equipment sound is not pure technically. The gain stage has a vacuum tube and also the frequencies below 50hz do sound anything but a sine wave. All these are the signature of a vintage sonic richness. The Fundamental software presents you 8 instances of this tone generator with all the artifacts and richness of a vintage equipment. The combination is we believe something you should experience.

The grounds of this project have been founded during my visit to Hainbach's studio in Berlin in 2020, who is well known for incorporating these vintage tools of the past to modern studio environment. Basically after sonicLAB VOLBot and PaSSBot, it was time for an oscillator.

The idea was basically recreating a test tone equipment sound in software form but with the approach and expertise on generative design which sonicLAB has followed on its previous products. A test equipment itself can be rather a tone generator but not a musical instrument to perform with.

sonicLAB incorporates also Xenakis's methodology and has successfully implemented it already in its current products of sonic creation. The perspective derived from Xenakis follows that these tools are not designed just to deliver static results but in contrary calculated and animated in continuous motion which is not possible to do by hand. Now imagine the results of implementing the signature modulation scheme on 8 parallel instances of a vintage sine wave generator in continuum.

Upon the various recordings of his vintage tone generator delivered by Hainbach, sonicLAB has developed a genuine poly-morphing wavetable oscillator algorithm to faithfully regenerate the vintage tone with infinitesimal value steps of possible gain and frequency combinations. (please check the **Appendix** section for details.) Additionally one can choose different tune scales or import custom ones to quantize the pitch of these oscillators.

4 different continuous stochastic modulations (GENs) and stochastic AttackRelease envelopes , tuning scales (also reminiscent of sonicLAB PaSSBot) can drive each voice instance and put them in constant motion. sonicLAB design gives you a dynamic structure which will take you to unheard sonic territories.

Furthermore, sonicLAB implemented the direct interaction of an MPE controller, and delivered a user interface combining the past and present trends to deliver you fast and intuitive results.

Finally Fundamental2 comes with a the audio rate parameter morphing tool derived and adapted from sonicLAB Cosmos*ƒ* Saturn. Furthermore it has now very useful direct standard midi mappings for sequencing purposes. But maybe it is the first time a synthesizer software sees a morse code translator embedded as a synthesis modulation tool. Combined with Fundamental which is a complex sine wave oscillator, it can create very interesting generative patterns.

The latest version includes a stylized ring modulator, which we have tried to bring with a musical balance, and not just mathematical representation.

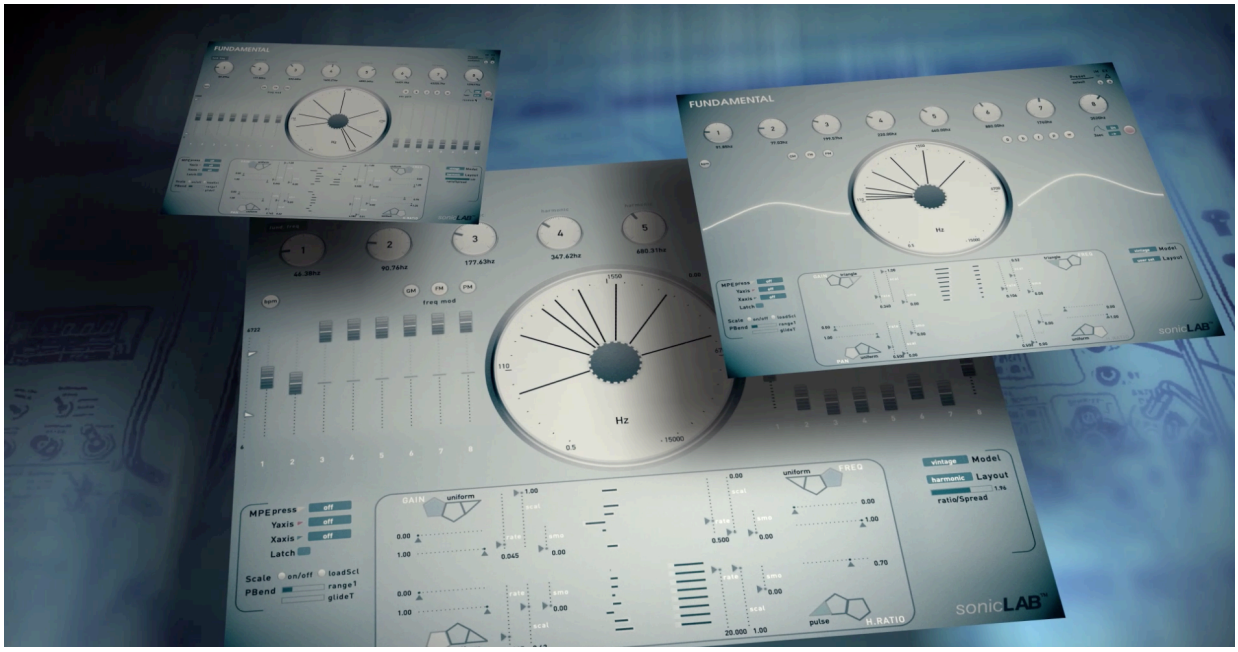
The purpose of Fundamental is not recreating an experience of some nostalgia with the exact representations of that relevant gear used back then. But with the expertise of sonicLAB in unique instrument design, Fundamental creates a bridge with the past and its compositional/ hand made practical experiences and let's it reach today's modern computer software by calculating and renders compositionally meaningful parameters, modulations and generative soundscapes, which cannot be hand made.

Here again I salute all the composers / researchers having founded all these roots for us. I find myself lucky to be able to reach this technological progress letting us develop, creating and producing freely and I am extremely happy to deliver you this important electronic music tool.

Dr. Sinan Bökesoy August 2021.

chapters:

Pages

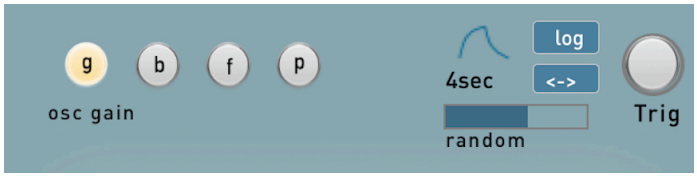


Integrating and running the Fundamental2 in your workspace

After the launch, Fundamental will load the default preset. When you are using the plugin version, all the settings and parameter changes which you apply on the Fundamental will be saved along with your DAW project and also recalled back when you open that project.

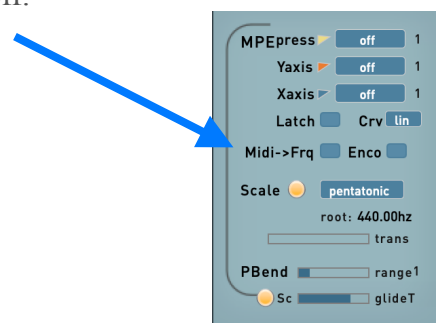
Triggering Fundamental :

- When a preset is loaded , Fundamental2 expects a trigger to run the audio output. Simply it is this button.



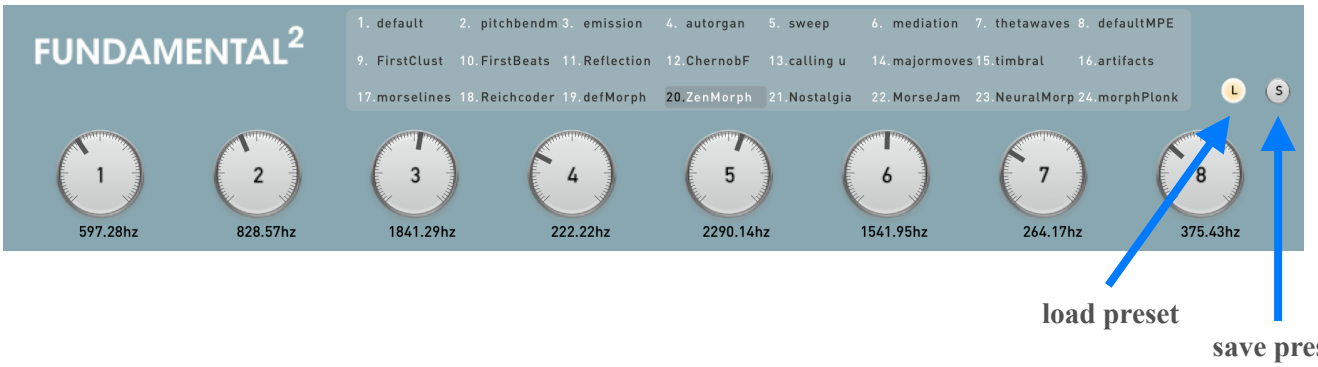
trigger the output state, also with MIDI note C5.

- You can also trigger with a **C5** Midi key (**72**) **(Midi->Freq switch must be off).** This happens like a switch again, press the key to turn it on, and then press later again to turn the trig off.
- You can also use the TrigAR plugin automation on your DAW to draw the on / off instances along the track.
- Triggering Fundamental happens through a AttackRelease envelope , which has by default a “gate” state, meaning also as pass thru.
- When the **Midi2Freq switch is on**, you can perform each Fundamental oscillator with midi note on input on respective midi channel. This will also turn on the Trig state on/off.

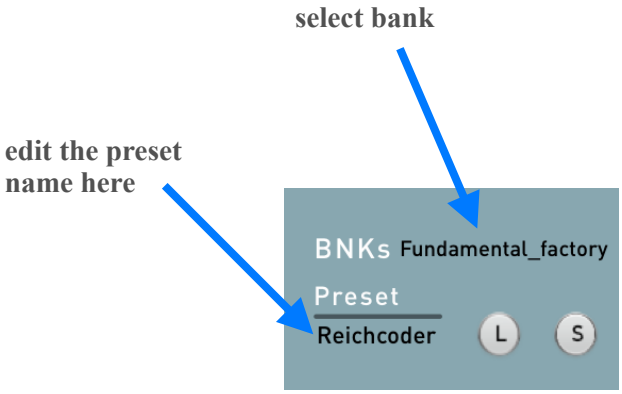


Important : You can save the Trig state along with the preset, so that when recalled it will keep the last state. Likewise you can navigate between presets and create transitions without acting on the Trig button again and again.

Presets : Fundamental2 offers banks with 24 preset slots. You can load from these slots , save onto them, import and export the banks. Fundamental comes with a number of preset banks. To load and save presets press the respective button as shown below and the current work bank will open as 24 slots.



For both operations, just press on the desired preset slot. When saving on a slot, a confirmation dialogue will open. The Load button will remain lit and the preset loading window will remain floating unless you turn the load button back off. This will give you the possibility to navigate between your presets quickly during a live performance.



Important notes: The default bank when the Fundamental launches will be the Fundamental_factory.xml which sits inside the Fundamental PresetBanks directory (explained above).

It is this bank being loaded by default each time you launch the Fundamental. (other than that you have saved the last plugin state along with your DAW project) and the first preset will be loaded.

All the preset save operations you do apply on a slot will replace the content of this particular bank slot. So it is a destructive operation. However when you reopen the application, the present changes which you have applied on the working bank remains.

Therefore we suggest that you keep a backup of the factory bank and when finished working on a custom bank with new presets then create a copy of this bank later with a different name. Likewise you can create easily backup copies of your banks in the Fundamental work directories.

Alternative preset selection : When not in **Midi->Freq** mode **,** you can use the **midi notes** 21-37 (A0-C2) to select a preset between slot 1-16.

preset bank handling on Fundamental2 iOS

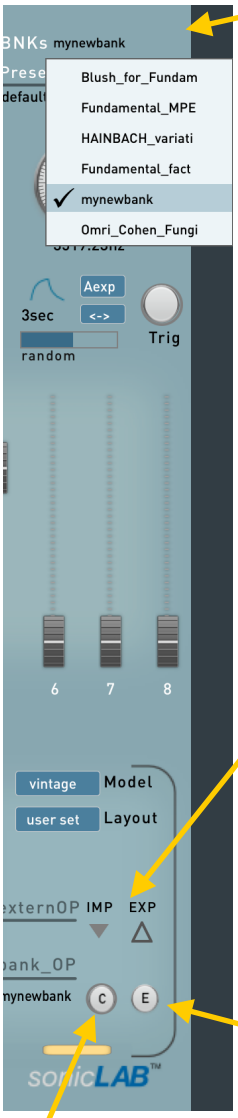
Both the standalone iOS and AUv3 version share the AppSandBox file space where the Fundamental files are stored and shared. But it is only the standalone version which is authorized to import/export from outside to the Fundamental2 preset bank folder or vice versa. You can import preset banks to the Fundamental2/PresetBanks folder or export your custom ones from there to outside.

You cannot directly access inside this folder ! but you can see its content on the preset bank pop-up menu.

You can use a file utility software to access the exported preset banks or bring other ones to your Fundamental2 folder and import them to your PresetBank folder.

You are welcome to check Fundamental tutorial videos on our YouTube channel.

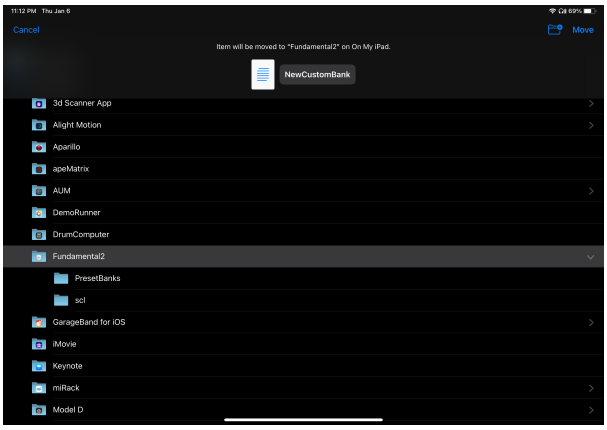
Fundamental iOS standalone



The preset banks in the App sandbox are available through this pop-up menu. You can import banks into this menu from external environment only on the standalone iOS version.

Here the current selected bank can be exported. We suggest to do the export inside the Fundamental2 folder but outside the Fundamental2/PresetBanks directory. Likewise you can easily access it with a file utility on your iOS device and send it elsewhere, for example load it to the OSX version of Fundamental2. Below, on the screenshot, we move the NewCustomBank to Fundamental2 app folder.

only available on the standalone version



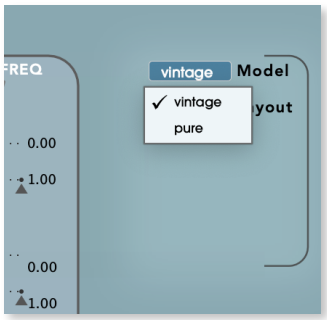
exporting the current selected bank. - select the destination location as Fundamental2/ PresetBanks and give it a name and choose “move”.

erase the selected bank in the app sandbox.

create a copy of the selected bank in the app sandbox but with a given custom name.

Importing a bank to the Fundamental2/ PresetBanks directory. The imported bank can be seen now on the PresetBanks pop-up menu.

Structure and the controls on the user interface



Fundamental consists of a bank of sine-wave of oscillators, in total 8 parallel voices. They deliver you highest quality generated wavetables of a **vintage vacuum tube Rohde&Schwarz** sine tone generator with no compromise of any data content on any frequency range. They can also be switched to **mathematical pure sine wave** generator as an optional mode.

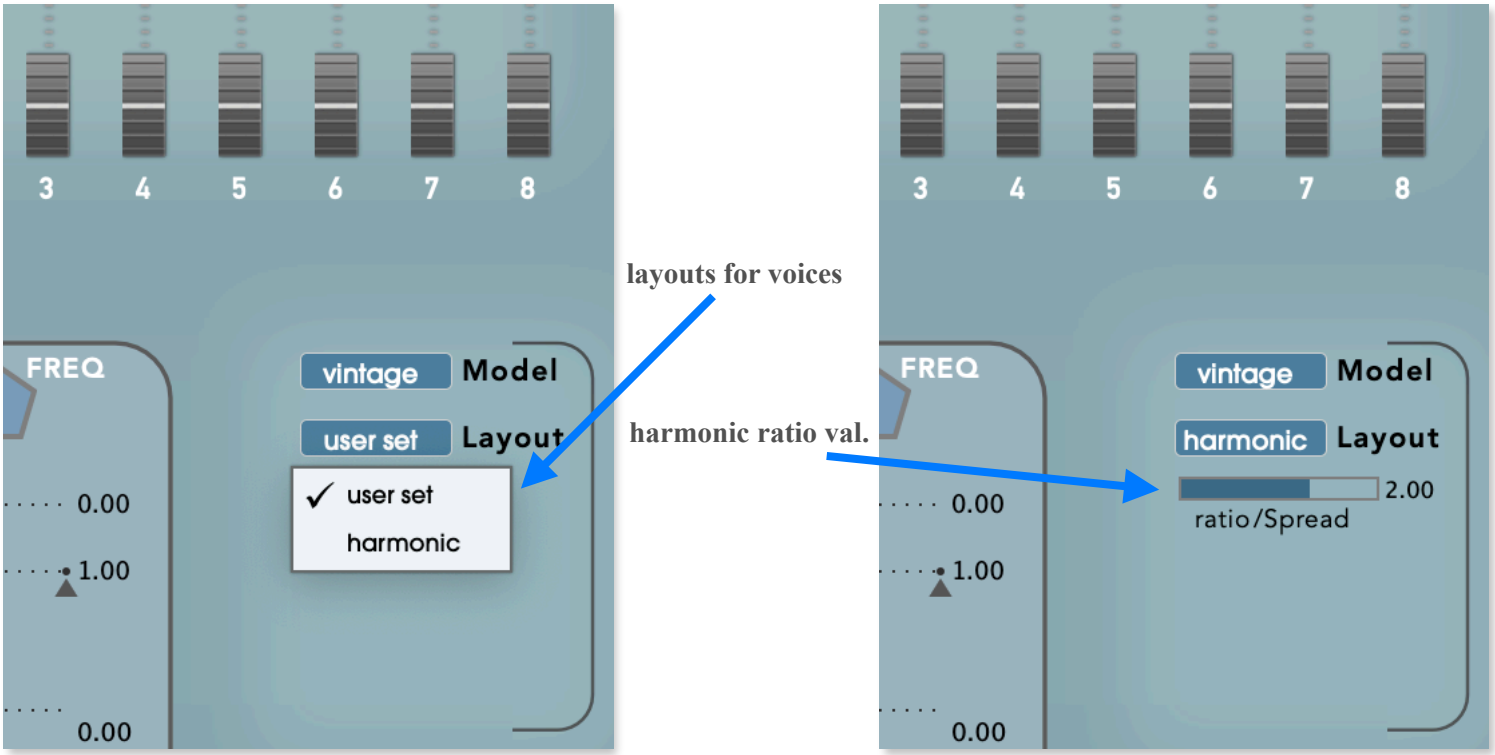
You can control the frequency, gain, voice volume boost (up to 9db) , panning and apply various modulations to these parameters dedicated on each voice.

You can trigger the system in several ways; assign each voice frequency by midi keys, and also perform multiple parameters directly with your MPE controller in real time.

The organizational layout of voices consist of 2 modes: User set layout, harmonic distribution layout.

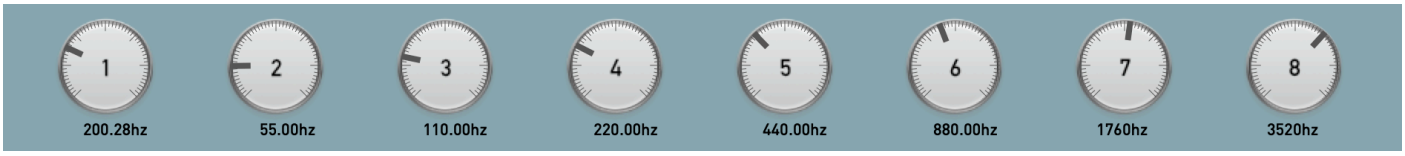
User set layout : Each voice operates independently with relevant parameter values assigned to it.

harmonic distributed layout : The first voice behaves as a **fundamental** tone, and the others are harmonics of this fundamental distributed with frequencies set with harmonic ratio value (between 0 - 3). Harmonic ratio can be set also with Midi CC31 message.



The harmonic ratio val. (ratio / Spread) defines the frequency relation between the fundamental and the next harmonic. 1.harmonic / fundamental = 2 harmonic / 1. harmonic = … = ratio / Spread value.

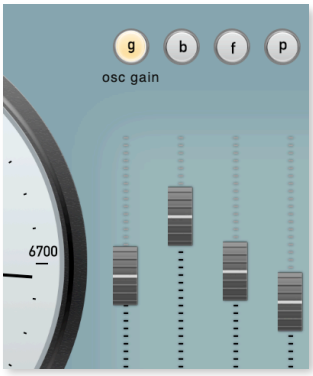
By default it is set to 2.0 (octave relation) , you can also apply to it a stochastic modulator (H.Ratio) which will change it for every harmonic individually.



Above, the layout of voices where the fundamental is at 220Hz and the harmonics are automatically calculated and set with a ratio of 2.0 in this case. Whenever you change the fundamental frequency the harmonics will adapt to this change.

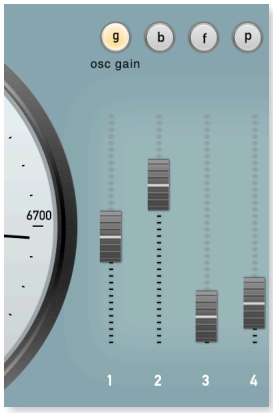
You can edit the frequency of a voice oscillator with several tools.

- Dial the rotary slider of each oscillator to assign the frequency.
- Type in the number in **hz** exactly on the value of each rotary slider.
- Use a fine tune slider to change the frequency around the current rotary freq setting.



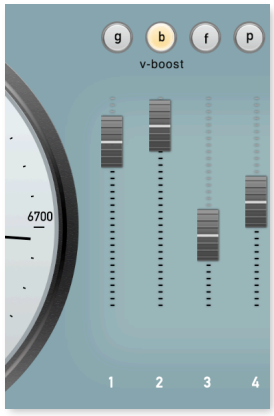
You can set the gain of a voice oscillator as following;

- with the gain sliders dedicated to each voice. Note that, the gain slider also plays a role on the character of the oscillator as on the vintage sine tone oscillators. So this is not a simple signal multiplication operation but calling the exact behavior of the vacuum tube tone generator which changes the timbre according to its gain setting.
- By using the v-boost slider dedicated to each voice. This slider simply adds signal strength for the relevant voice up to 9db. You may want to create a custom balance between the voices in favor of the combined timbre you would like to create. This is where to do it.



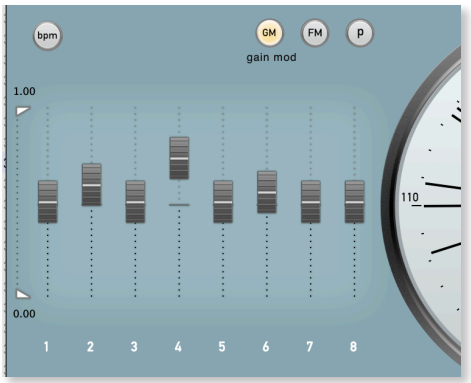
You reach these setups by selecting the appropriate buttons above the sliders

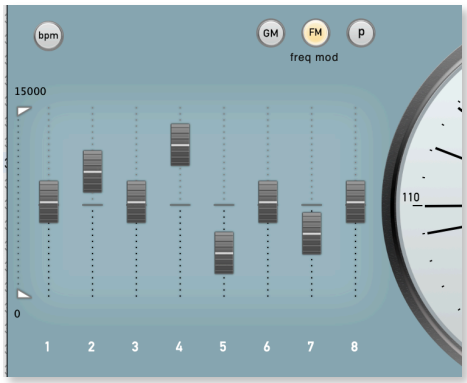
When you double click on the sliders, they will move to their default position.

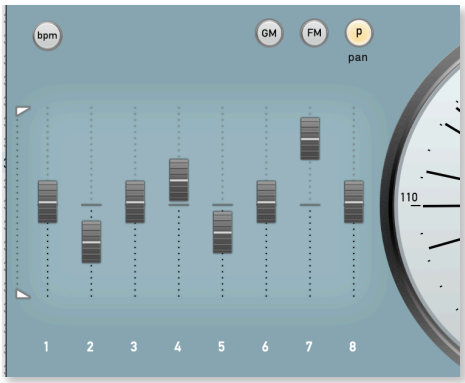


The left bank of the sliders are dedicated to modulation depth assignment for each voices gain modulation, frequency modulation and panner. When they are in middle position, they cancel the modulation absolutely.

These faders apply bipolar depth to the modulation depending on their position relative to the middle position. For the pan fader set, the up position is right and below position is left oriented.







Technically the stochastic modulator output value of a voice is multiplied with the relevant modulation depth fader value belonging to that voice.

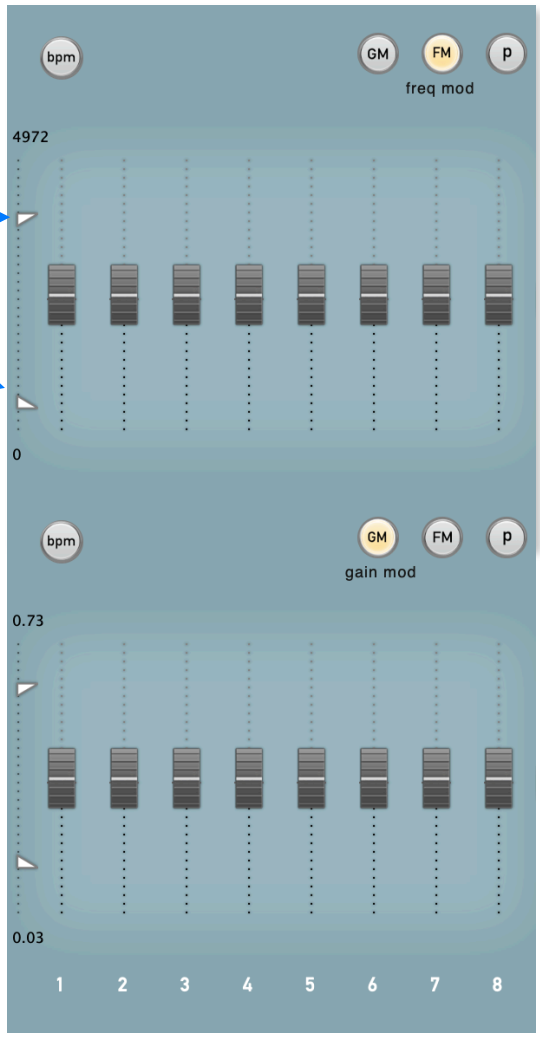
High & Low Barriers :

One can limit the frequency and gain output values (output of set value + modulation applied to it) between Low and High barriers.

The values for the barriers can be set with the left most slider of the left bank. If you are on **FM** modulation interface, then the barrier slider is for the frequency barriers.

If you are on GM modulation interface, then the barrier slider is for the gain barriers.

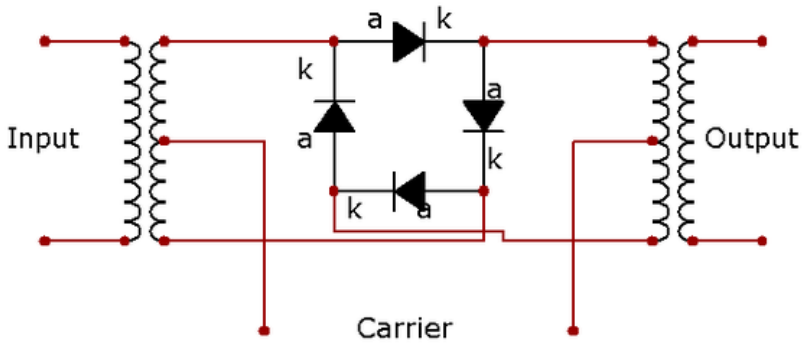
Technically any value exceeding the low and high settings will be bounced back inside the barriers range.



The Ring Modulator on Fundamental2 (new in v2.3)

Some background : In electronics, the **ring modulation** is a signal processing function, in which two frequencies are combined to generate an output signal. One signal, called the **carrier** is typically a sine wave or another simple waveform and the other is called the **modulator** signal and can be more complicated serving as an input signal.

The name derives from the fact that the analog circuit of diodes originally used to implement this technique takes the shape of the ring: a diode ring.



Ring modulation in a way modulates the amplitude of the signal, both modulator and carrier signal is being multiplied on the same time scale. If the modulator consists of a sine wave of frequency **f2** and the carrier is a sine wave of frequency **f1**, then mathematically it can be shown that the output signal consists two sine waves with frequencies **f1 + f2** and **f1 - f2**. These two output frequencies are known as sidebands. If one of the input signals has significant overtones, the output will sound quite different, since each harmonic of this input signal will generate its own pair of sidebands that do not have to be harmonically related.

The ring modulator has been used in electronic music as earliest experiments for synthesizing new tones and also as an effects unit. Namely Werner Meyer-Eppler has used extensively the earliest musical instrument utilizing a ring modulator (*Melochord* 1947 built by Harald Bode) in his early days of Bonn Uni. electronic music studio.

Meyer-Eppler's student Karlheinz Stockhausen, a prominent figure in 20th century music, has used the ring modulation in many of his compositions starting with *Gesang der Junglinge* , 1956. He has experimented with ring modulation both for synthetic sound generation and also processing live acoustic instruments directly with it.

Ring modulation has also captured great interest in sci-fi movie sound design as major component such as on *Forbidden Planet* 1956, and *Doctor Who*, 1963.

Harald Bode designed the Bode Ring Modulator in 1961 and also the Bode Frequency Shifter (reducing one side band from the output signal) in 1964 as commercially available products for sound synthesis and then later has been licenced to companies like Moog , Buchla etc. One of the very famous analog synthesizers having the ring modulation as a distinctive technique is the Yamaha CS80.

For the simplest scenario, one has to turn on the RM switch on Fundamental2 located above the Trig switch to activate the ring modulation. The 8th oscillator of Fundamental will act as the carrier signal generator and all the 7 other oscillators will become the modulator signal generators.

The figure above shows the gain settings of the 1st osc and also the 8th osc (carrier signal).

For the ring modulation, the carrier oscillator has to have a gain setting other than zero. The frequency position of the carrier signal and modulator signals can be edited with the frequency dials.

The circular display of Fundamental2 shows the carrier signal position (red line) and the produced side band frequencies (with yellow/orange color lines) . For the case above we use one modulator signal (osc1) and the carrier signal (osc8) to produce the side band signals (244 hz + 690 hz) and (244 hz - 690 hz).

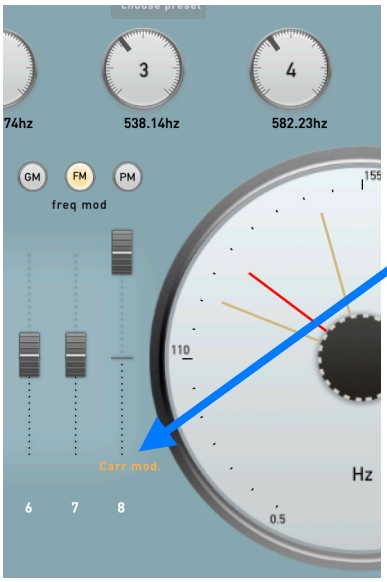
Note that the carrier signal is not directly in the output but only shown as visual reference on this circular display panel.

According this principle, Fundamental2 can deliver 7 unique ring modulators at the same time by pairing each modulator oscillator (1 -7) with the carrier signal oscillator.

Each RM **modulator** oscillator has its own gain / frequency / boost and panning setting. And the gain / frequency and pan values can be also modulated individually with the powerful GEN stochastic modulators of the Fundamental.

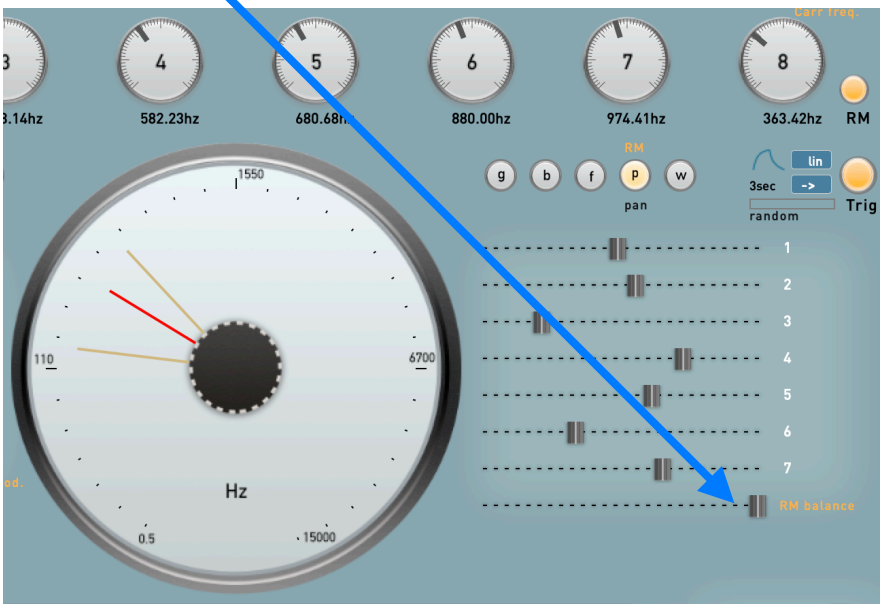
The carrier oscillator (8th osc of Fundamental) has its own gain / frequency / boost setting. Panning is irrelevant for this oscillator. The gain and frequency value of this oscillator can be also modulated individually with the powerful GEN stochastic modulators of the Fundamental.

The carrier oscillator will not be affected by the envelope playback modes, hence its gain setting will remain constant.

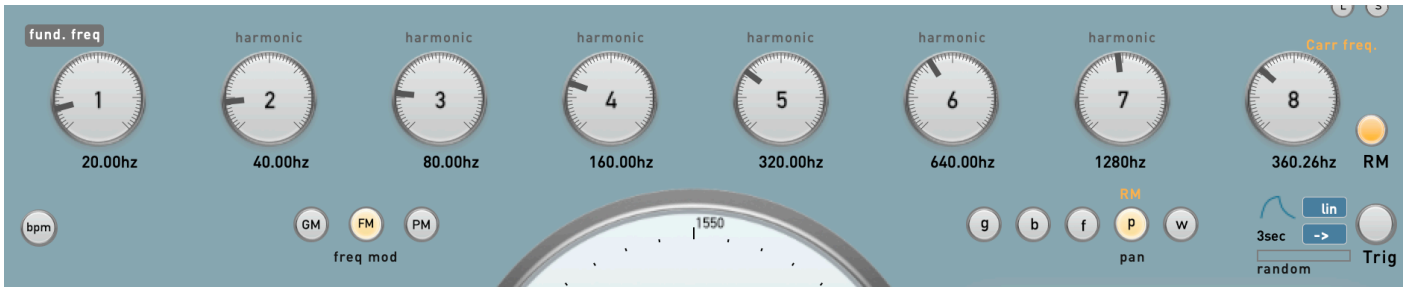


You will see helper labels which show the functions dedicated to the carrier osc on the user interface.

We have mentioned that the Pan setting is irrelevant for the carrier oscillator. But we have used the pan setting slider (see below) dedicated to the 8th osc of Fundamental to gradually morph between full Ring modulation to non Ring modulation state of the oscillators. This unique sonic processing can be handled by the continuous movement of this slider serving as a RM balance.



When the harmonic distribution mode is being used together with the Ring modulation, the 7 oscillators will line up harmonically according the ratio/spread setting. But the 8th oscillator (the carrier oscillator) will have its free dedicated frequency setting. On the example below, the first 7 oscillators are harmonically related with a fundamental frequency of 360.26hz.



Notice that, no additional slider, mode button etc. has been added to the existing Fundamental2 layout for the Ring Modulation operations. (except the RM switch)

Likewise all existing the external Midi features , MPE and morphing engine functioning can be used as before for the Ring Modulation mode activated as well.

<u>about the sonicLAB implementation of RingModulation</u>

sonicLAB is dedicated for best audio quality and the implementation of the RM on Fundamental2 was not a quick process by the book.

Fundamental offers two type of oscillators, one mode simulates the R&S test equipment oscillator and the other mode is a pure mathematical sine wave.

The test equipment oscillator generates rich harmonics especially when operating in low frequencies and these will create harmonic / inharmonic sidebands for each component found in the modulator and carrier signal. sonicLAB has spent quite some time to come up with a stylized ring modulation version in order to contain the beauty of the grit and also deliver a pristine quality without the digital artifacts. Of course this comes with a price of computational power needed.

For a perfect transition between the no-ring modulation mode to full ring modulation mode, we had to re-calculate the balancing of the side bands and mod signals dynamically.

You can compare the ring modulation effect results by using the both modes of the oscillators and decide what suits best to your particular use case.

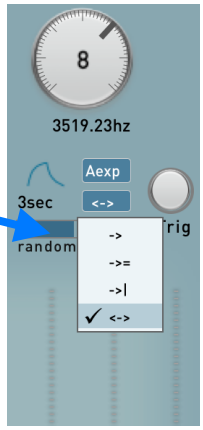
A new preset bank called “**Intro2RM**” has been packaged with Fundamental2 dedicated to the use of RM. Therefore checking its presets will be a good practice as well.

The AR envelope on Fundamental

Fundamental offers you another gain modulation mechanism as attack release envelope mapping to gain sliders.

AttackRelease envelopes are applied on the current settings of the gain sliders of each voice and a variety of them are preset:

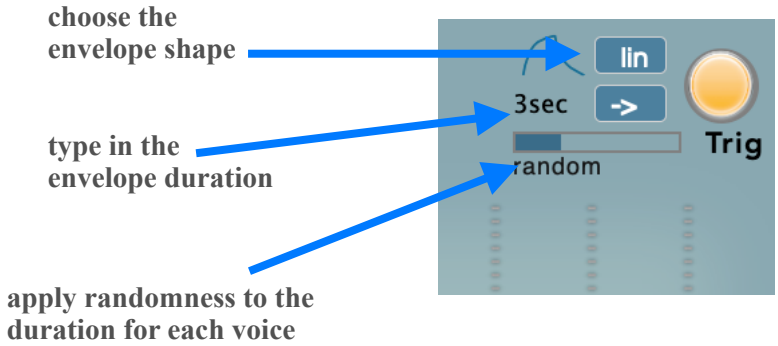
-> gate, **->= stop at sustain, ->| full cycle , <-> looping types**.



That said, the **gate** mode is no envelope but for the others the current voice gain setting becomes the Sustain point of the envelope. When you activate the Trig button, an attack envelope from zero point up to sustain level will be applied for the defined duration, and the same from sustain to release point.

Depending on the randomness setting, the faders will move up to the sustain point and back to release point on different times.

The available envelope shapes are linear, exponential and logarithmic.



The envelope curves are : **lin** (linear), exp (exponential) , **log** (logarithmic) , **Aexp** (very fast attack and exponential when decay) and **ERnd** (randomly alternating exponential fast / slow attack envelopes). **ERnd** envelope curvature is also changing randomly at each envelope start.

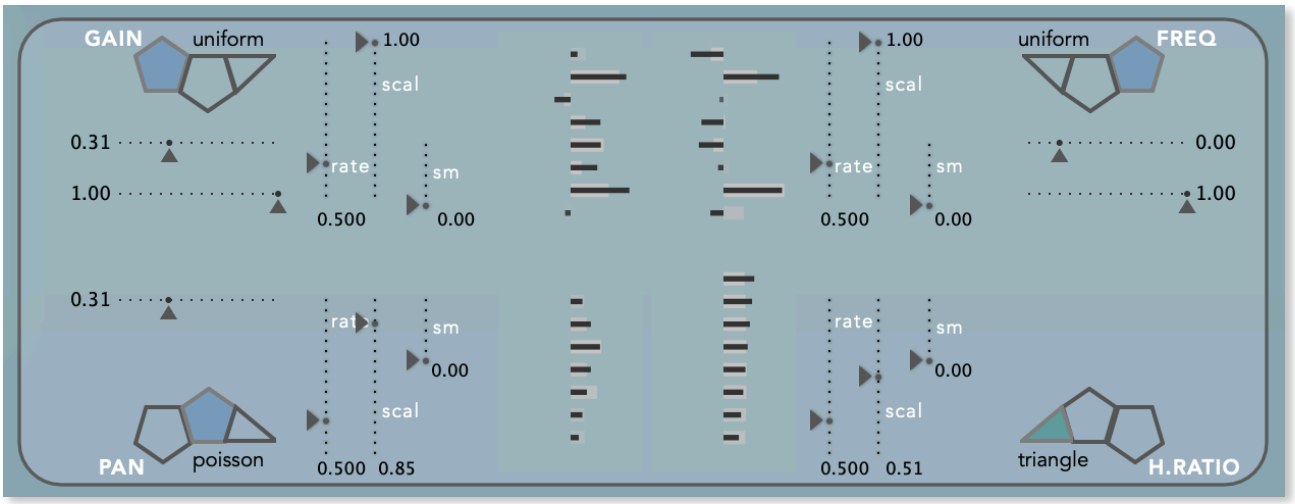
I. **->= mode :** The Trig button will activate the Attack part of the envelope and then it will stay on the sustain point until another Trig event has been achieved which will activate the Release part of the envelope.

II. **->| mode :** The Trig button will activate the Attack part of the envelope and when the envelope reaches the sustain point, the release part will be activated automatically so that the envelope runs a full cycle by itself.

III. **<-> mode** : When the Trig button is activated the attack - release - attack - release ….. will go on in a loop. And when you set the random slider value each attack - release will be set with random times.

All these actions can be followed with the animation of the gain sliders. Again this gain level manipulation preserves all the oscillator timbre interaction according to the gain level.

Fundamental GEN modulation generators

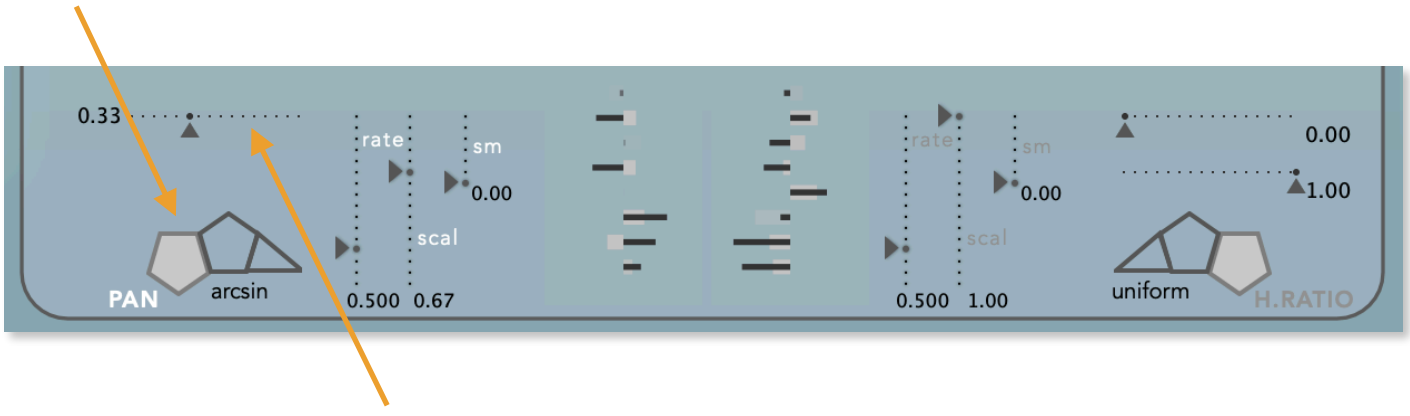


Fundamental incorporates 4 continuous modulation generators for each voice with rich stochastic function possibilities called as **GEN,** and their destination is directly labeled on each of them. GAIN, FREQ, PAN, H.RATIO.

Each GEN offers modulation sources with continuous probabilistic distributions, discrete probabilistic distributions and continuous standard waveform functions as listed below. *

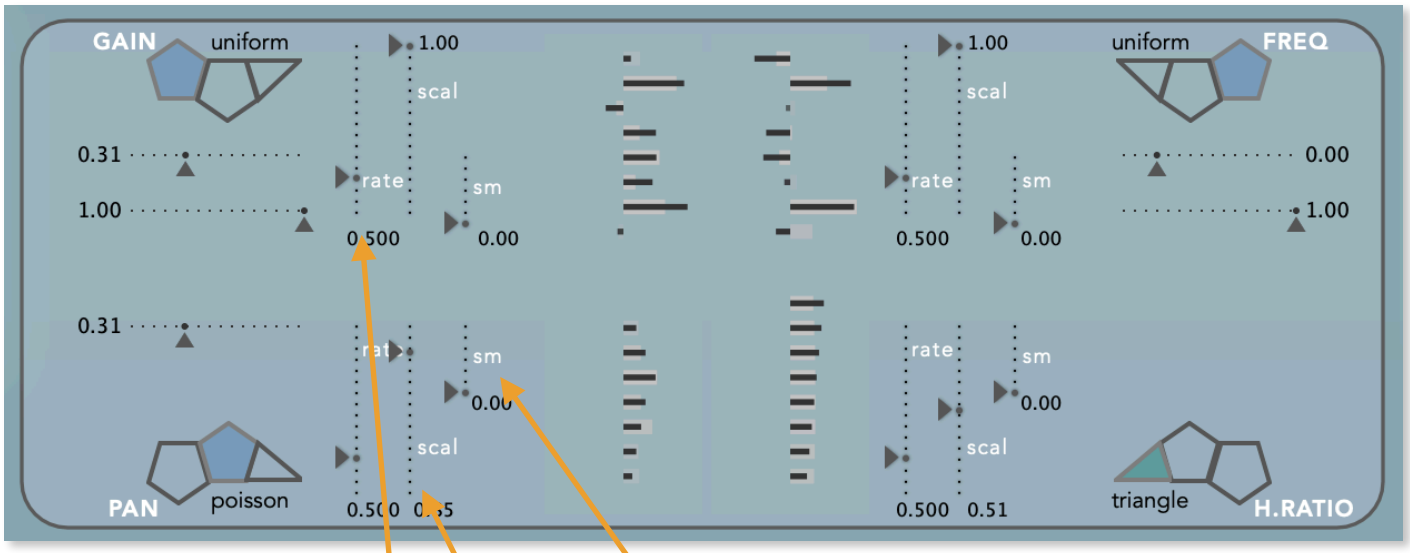
uniform	**poisson**	**triangle**
gausssian	**bernoulli**	**sine**
chauchy	**binomial**	**sawtooth**
exponential	**pascal**	**pulse**
weibull	**geometric**	**exponent**
arcsine		**morse**
lognormal		
chisquare		
gamma		

Each GEN has relevant button switches to select the desired function category as listed above.

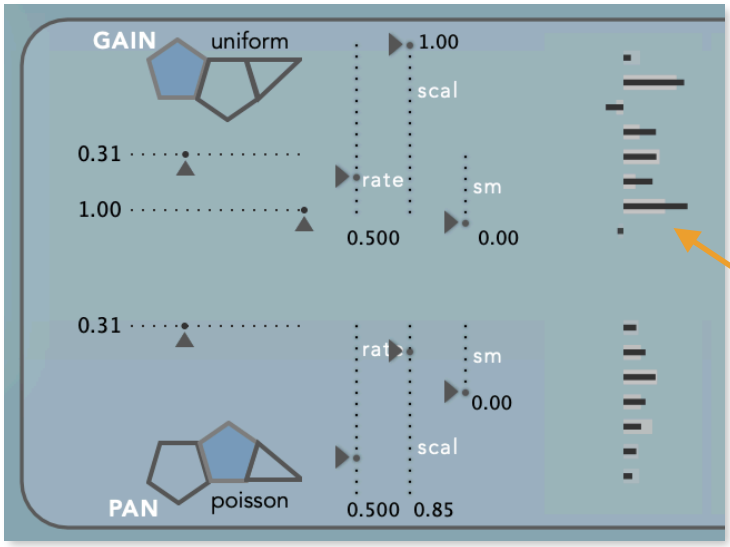


For each function selection the relevant parameter sliders will show up. Some of the functions have one parameter, some two parameters, and some none. Experiment with them !

* What are stochastic distributions ? The ending page of this section answers this question.



Each GEN generator has **rate**, **scaling**, and **smoothing** rotary sliders. The **rate** slider defines the speed of the generator (in Hertz) , the scaling slider defines the amplitude of the generator and the **smoothing** slider sets the degree of softening the rapid changes of the generator function. When the **BPM lock** mode is on, the **rate** slider shows the speed as note duration values.



The value display of each GEN represents the unique value calculated for each filter. The display updates accordingly to the filter count. The center point represents value 0.

(New in Fundamental2)

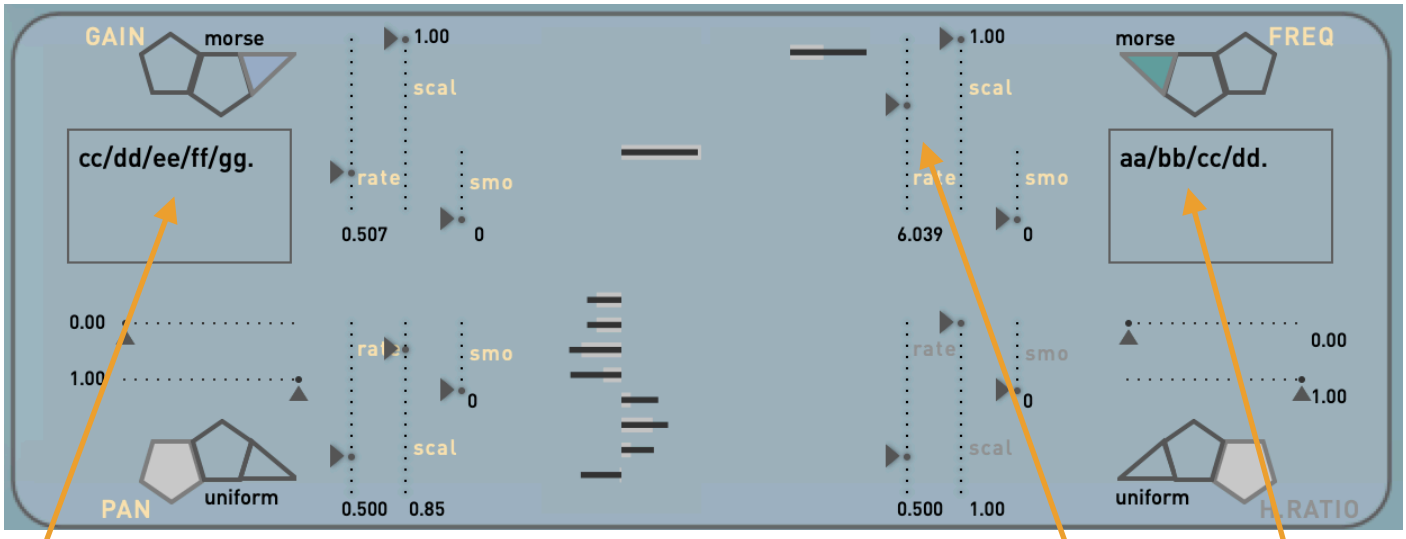
Introducing the more code translator as a modulation source :

If a test equipment can become a tone generator, then a morse code generator can be considered as a sine wave rhythmic pattern generator.! Likewise Fundamental2 incorporates a full morse code translator, which converts letters to its code patterns to modulates the Fundamental oscillator amplitudes, frequencies etc. All the 8 oscillators can have different patterns assigned and this will deliver rich poly-rhythmic behavior. Add to that the stochastic modulations etc. to reach some unheard versions of the morse world easily.

This pattern consists of one type of short / long signal combinations, such as note lengths.

A ·–	J ·–––	S ···	1 ·————
B –···	K –·–	T –	2 ··–––
C –·–·	L ·–··	U ··–	3 ···––
D –··	M ––	V ···–	4 ····–
E ·	N –·	W ·––	5 ·····
F ··–·	O –––	X –··–	6 –····
G ––·	P ·––·	Y –·––	7 ––···
H ····	Q ––·–	Z ––··	8 –––··
I ··	R ·–·	0 —————	9 ————·

The define such patterns for each Fundamental oscillator, we will use again letter combinations. As you know each oscillator can have 4 dedicated GENs and the letter combinations are being typed in directly on the GEN fields explained as below.



When you select “morse” as modulation source on any GEN, a rectangle area appears where you can type in the morse letters to be translated. By separating the letters with a ‘/‘ , you can assign them to corresponding oscillators.

Hence here the first oscillator of Fundamental will be gain modulated with a morse code of “**cc**” , the second one “**dd**” and the next oscillators with “**ee**”, “**ff**” and “**gg**”. The point “**.**” defines the end of the code.

The GEN parameters “rate”, “scal” and “smo” act on the morse code generator like they act on other modulation sources. Above, the morse code modulates the frequency of the oscillators 1-4. And the “scal” parameter will define the range of the freq modulation.

Your input patterns will be saved along with the preset. Practically you are not limited with two letters as on the above example, try much longer patterns. But you can only use the letters and numbers listed above.

Since different morse patterns can be set active at the same time on multiple Fundamental oscillators, one can get easily creative with rich polyrhythmic combinations. Apply tune scales , barriers limiting gain / frequency, and add stochastic modulations , the simple morse code generator becomes a true generative musical tool.

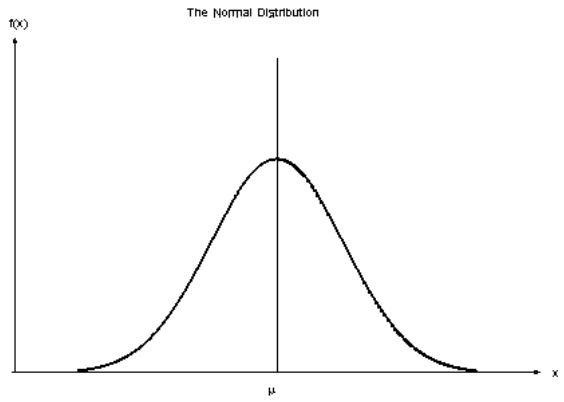
What is a stochastic function / probabilistic distribution ?

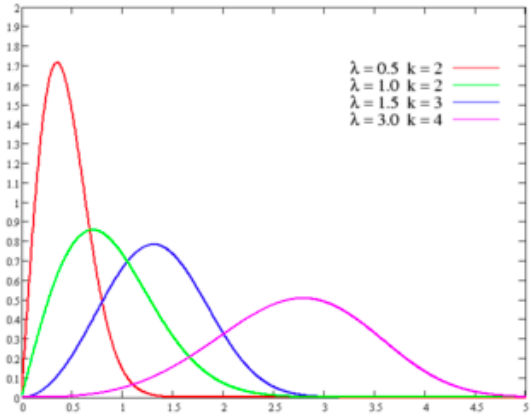
Stochastic processes are widely used as mathematical models of systems and phenomena that appear to vary in a random manner. It is a field of statistics, and the term random **function** is also used to refer to a **stochastic** or random process, because a **stochastic** process can also be interpreted as a random element in a **function** space. A type of random function can be modeled with a probability distribution function showing distinctive behavior when distributing the values as its output.

If you flip a coin, both sides have the same probability of showing up, so this can be modeled with a uniform random distribution. For example, if we have a light bulb which goes on and off randomly due to a cable connection failure / electric spark, this is not a uniform random distribution but can be modeled with a poisson distribution. The movement of a mass of molecules in gaseous spaces can be modeled with brownian motion (a random walk).

Most of the stochastic functions have parameters, a value of certain range to change the behavior of the stochastic function. The reason that the **Fundamental** offers many types of stochastic functions is to present you different types of 'distribution shapes', which give a distinctive character to the process being used for.

Below, you can see the gaussian / normal distribution and the Weibull distribution shapes (which varies according to its parameters). As you can see, with the normal distribution of events, an x event has a higher possibility to happen towards the centers of the bell shaped cure.



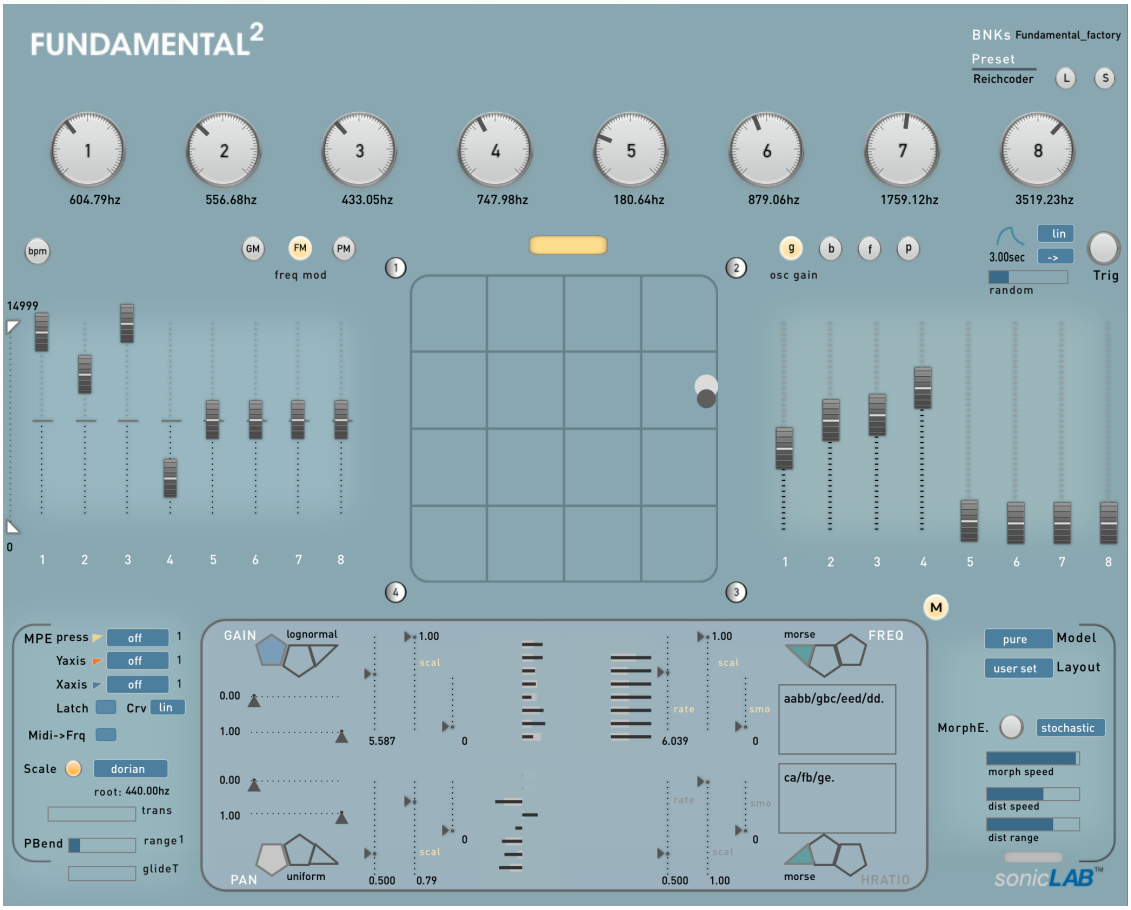


Music is a distribution of events in organized manner and stochastic functions let us control the randomness between order and total disorder. This is applied heavily in 20th century contemporary music and computer music by the composer Iannis Xenakis *.

sonicLAB products offer similar mechanisms and inspiration in form of modern tools to todays composer / sound designers. We can assign these distribution characteristics to various musical parameters and sound synthesis parameters at various levels. The GEN modulation sources are themselves LFO's which modulate various parameters of the sonicLAB PaSSBot LFO with a rich palette of stochastic distributions.

* Iannis Xenakis "Formalized Music" Pendragon Press.

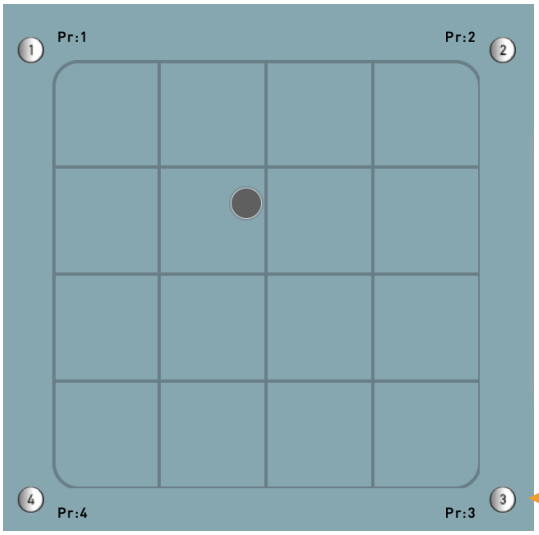
The Morphing Engine on Fundamental2



The morphing engine has the purpose of gradually morphing the parameters between 4 unique Fundamental instances. It has been derived from the sonicLAB Cosmosf Saturn, which has an audio rate parameter space morphing engine, highest precision smooth passages along with visual projection.

These 4 instances can be custom assigned variations of the current work preset of Fundamental, or can be also chosen among the presets of the loaded preset bank.

On the example shot below the preset parameters instances are chosen from Preset1, 2, 3, and 4 as you can see them at the corners of the morph panel. The small filled circle inside the panel area represents the morph state, which you can drag with mouse or sending CC messages respectively.

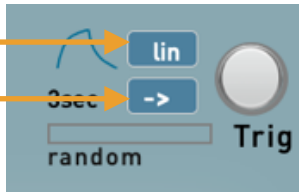


The distance of this circle from the corners defines how much each instance contributes to the current morph state. The closer the circle to an instance corner, to more that instance contributes to the morph state.

Morphing works best on continuously altered parameters where one can follow a gradual smooth change. *Osc pitch, gain, freq mod*. are example parameters. However changing the *envelope type* introduces abrupt change in audio for example. Therefore some Fundamental parameters have been excluded from the morphing engine. Below we explain each of them.

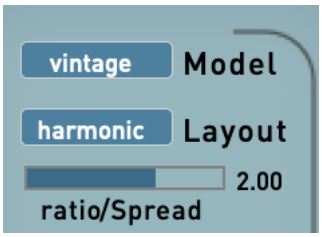
A Fundamental instance on the 3rd slot. Each slot represents a unique Fundamental parameter space.

Some parameters on the envelope section such as (envelope type, and the playmode) are excluded from the morphing engine, as their change introduce abrupt jumps on the audio process.



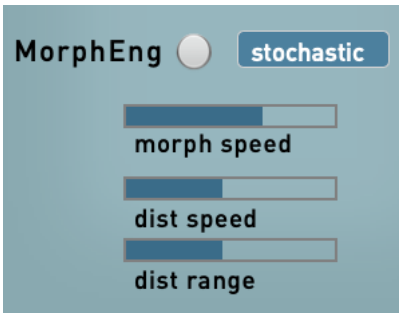
All MPE / midi parameters are excluded as well from the moprhing engine parameter space.

The sine oscillator model and the oscillator layout parameters states are excluded as well because there happens no gradual change on sonic quality when you change them, they are structural high level parameters.



As you would expect. all MorphEng parameters are excluded from the process.

Likewise you have now an idea about the logic behind this organizational decision.



What happens when you change an excluded parameter, for example the envelope type ?

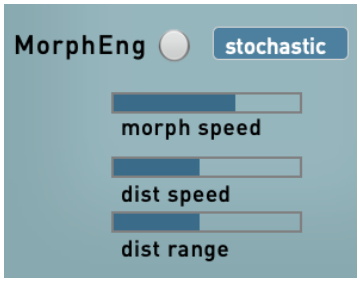
This change will be effective on all the 4 instances used by the morphing engine. For example when you load a preset to an instance slot , that preset envelope type will be effective on the other instances as well. We will give more examples about that.

Operating the Morph engine

To activate the morph engine , click on its button and automatically the morph panel visual will pop up. If you haven't assigned any instances to the morph slots before, the application will load the Preset1, 2, 3 and 4 to the slots respectively.

You can switch between the morph panel visual and the Fundamental circle by pressing the **M** button located above the right upper corner of the GEN panel, unless the MorphEng (morph engine) is off.

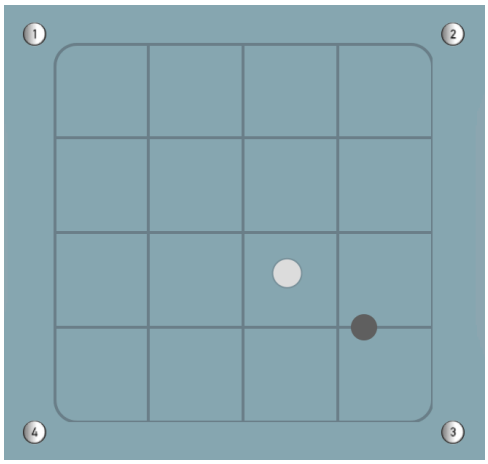
There are two types of morph operation : *manual or stochastic*. The stochastic type has additional parameters like distribution speed and distribution range.

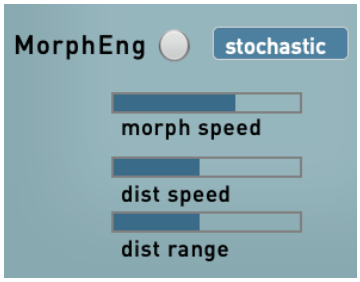


In both cases of morph operation, if the morph speed is not at maximum you will see a white circle following the lead dark circle.

Now the morph engine is smoothing out the abrupt changes on the dark circle position which you apply. At max morph speed, the change is identical but it can be quite slow at low speeds according to your needs. In fact, it is the white circle what you are listening to.

When the *stochastic morph mode* is selected, a random change will be applied to both x and y positions for the dark circle. Its speed is defined by the *dist speed* slider and the randomness range is defined with the *dist range* slider.





Morph Engine user interface behavior

When the Morph mode is turned on, all the user interface control is overridden by the morph engine (except the parameters which are excluded from the morph process as listed above)

When you change the location of the morph circle, the current morph state will be calculated and this parameter state will be projected on the user interface elements. (you will see them moving / changing by themselves.

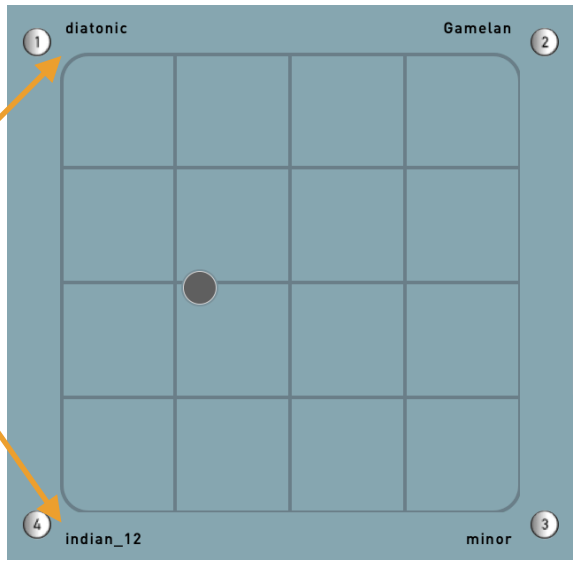
When you turn the Morph engine off, you will gain access to the UI controls back again. Turning on the morph engine automatically switches to the relevant user interface.

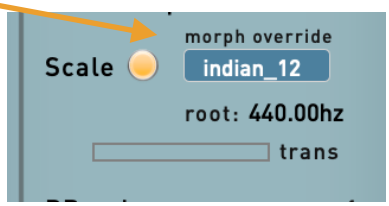
Morphing between tuning scales

When you turn on the tuning scale while the Morph Engine is on, then the engine will continuously morph between the tuning scale settings stored on each morph slot. These settings will be shown next to each slot automatically.

For a smooth change, we suggest that you keep the morph speed less than the maximum level.

Also you will see a notification that the preset tuning scale setting will overridden by the morph engine.





- load a preset to a morph slot

To import a preset to a morph slot, first turn off the morph engine. Then click on one of the slot buttons and the preset bank window will pop up. Your selection will load the continuous parameters of the preset to be integrated in the morph process.

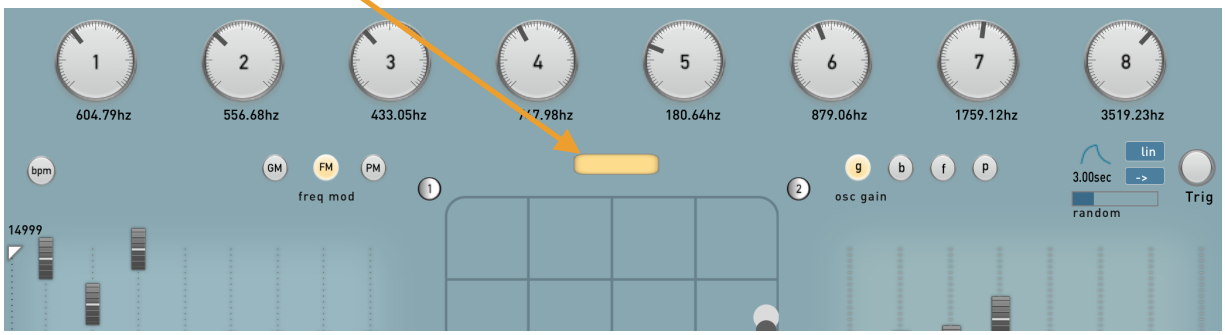
If you wish to cancel the load process just click again on the slot button.

If you wish to load a preset from different bank, first import the bank with the “import bank button” and then click on the morph slot to load the preset from the new bank.

- export the current state to a morph slot

You can assign the current parameter state of Fundamental2 to a morph slot. This can be your departure preset with altered parameters or the current morph state which you would like to assign to a specific morph slot.

To do that, press on the bar button above the morph panel, it will turn yellow and next choose a slot. This will save the current state to that slot to be integrated in the morph process.



For exporting the current state to a morph slot, you don’t need to turn off the morph engine. (with the exception of editing morse code for different slots) The idea is that you can change the slot content on the fly and have a continuous experience.

When your morph setup is using tuning scales and you turn off the morph engine, then the effective tuning scale will be your presets last set tuning scale as indicated on its menu item. This can be different from the calculated morph engine scale, no surprises !

- Saving your preset with morph settings

All the morph slot contents and the morph engine settings will be saved along with your preset and can be imported back. Just use the usual “preset save” button to export your work preset to a preset slot on the last imported preset bank.

Performing the Fundamental

We have designed the **Fundamental** as a different instrument than the standard direct to midi key mapped instruments.

First of all, to achieve the deep sonic offerings of the Fundamental and possessing the frequency resolution of it, performing the Fundamental just with standard Midi scales is not enough. Otherwise it would easily become an organ like sounding instrument playing usual chord structures, which we don't prefer to happen.

Fundamental offers full MPE support and also can receive standard MIDI note and defined MIDI CC messages. Below we will explain these cases.

Fundamental and MPE input

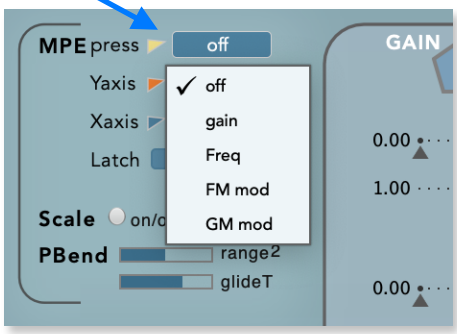
Fundamental makes full use of MPE controllers within following structure :

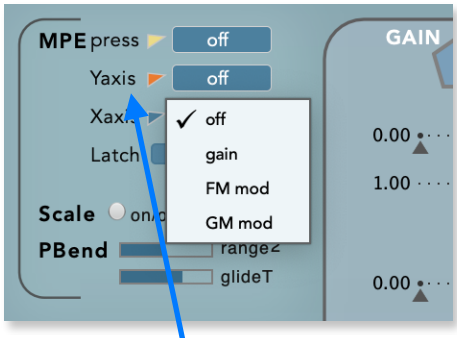
Each voice is hardwired to a defined MIDI note;

1. Voice : Midi Note **57** (**A3**)
2. Voice : Midi Note **59** (**B3**)
3. Voice : Midi Note **60** (**C4**)
4. Voice : Midi Note **62** (**D4**)
5. Voice : Midi Note **64** (**E4**)
6. Voice : Midi Note **65** (**F4**)
7. Voice : Midi Note **67** (**G4**)
8. Voice : Midi Note **69** (**A4**)

for MPE, this should be off.

MPE pressure mapping





MPE Y axis mapping

Midi Note **(C5)** acts like a **Trig** switch, each time you press the key,

The well known gestures of an MPE controller is as following:

- Pressure gesture on a midi key (after a midi note on event)
- Glide along the Y axis of the pressed key.
- Glide along the X axis of the pressed key (bending)

You can map these gestures to desired parameters acting on each voice of the Fundamental. The voice number is related to the pressed midi key as shown on the table above.

The available gesture mapping to parameter listing is likewise :

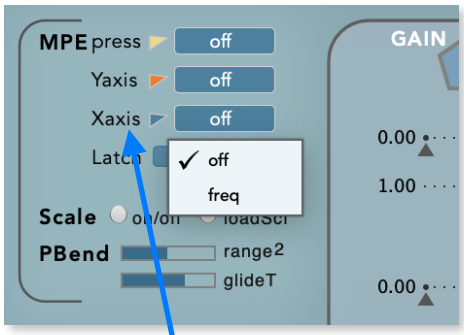
MPE pressure : off, gain, Freq, FM Mod, GM mod

MPE Y axis : off, gain, FM Mod, GM mod

MPE X axis : off, freq, FM Mod

Attention : The MPEx is hard wired as a polyphonic pitch bender. When it is set to off, both this and the standard midi pitchbend effect will be set off.

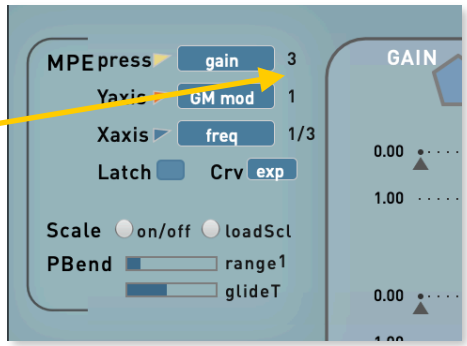
Just note the color coding used for each gesture ; **yellow**, **orange** and **blue**. The modulation which these gestures apply will be shown in real time below the left and right bank sliders depending their mapping.



MPE X axis mapping

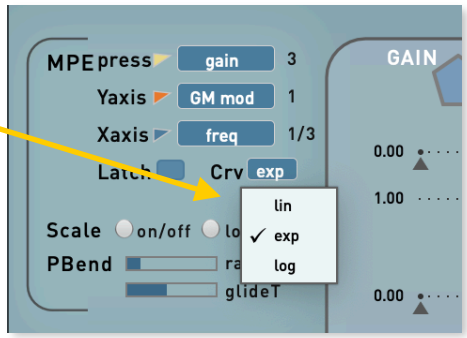
For each MPE component a scale factor can be set by using the MIDI controllers 28, 29 and 30 (value 0 -> 127 triggers). This will help you scale your performance input easily.

possible scale factors are : 0, 1, 2, 3, 1/3, 1/2.



One can also map the performance input with curve functions (like the velocity curves on standard midi keyboards) You can choose between linear, exponential and logarithmic curves.

The curve is being applied on Pressure and MPE Y component.



The **Latch** mode :

This mode provides a sustain state during the performance. When it is on, the last state of MPE pressure values will be kept even if the midi keys are off. How do we turn on / off the Latch mode ?

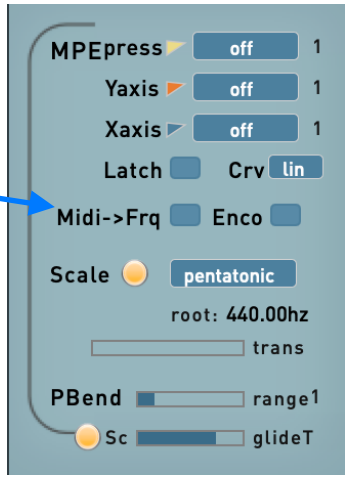
The Latch mode can be activated ;

- with a midi controller message (64) which is in general used for the sustain pedal control.
- by pressing the midi key 53 (F3). Note On turns is on, and note off turns also the button to off position.

Attention !

For using MPE and the Latch mode, the **Midi->Frq** switch should be off. Otherwise direct midi note-on input will take over.

Fundamental and Midi->Freq mode



Besides the MPE mode, Fundamental2 offers a new mode,
where one can address each Fundamental oscillator pitch with Midi Note On messages on unique midi channels.

- Turn on this mode with its Midi->Freq switch, and this will bypass the MPE actions. Then for each oscillator send midi note on messages to define the oscillator pitch on the relevant midi channel. (**use midi channels 1-8 to map oscillators 1-8**)

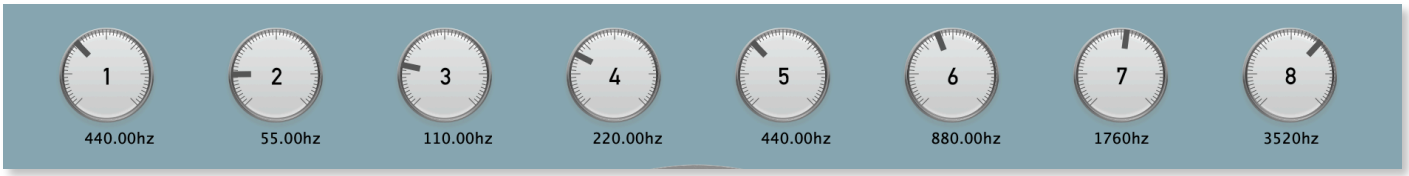
- This is excellent to address the midi sequences mapped directly to selected Fundamental oscillators.

- Midi note-off will not be translated to gate off, just the midi note-on will be interpreted. However if at any current state, there are no midi note events played, then the Trig switch will be turned off.

- If at any state the Fundamental Trig switch is not on, then the next midi on message in the Midi->Freq mode will turn on the Trig switch.

- As expected, the received note-on pitch values will be filtered through the Fundamental tuning scale quantizer.

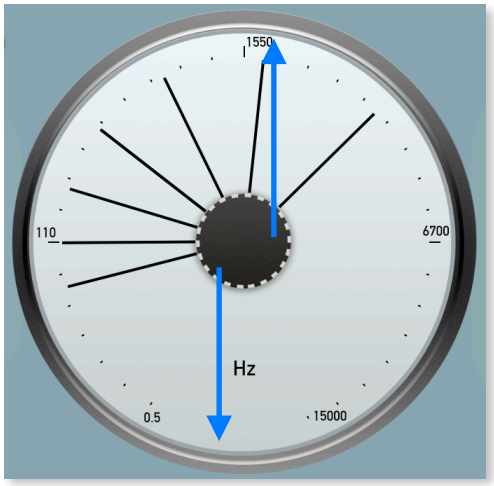
Here is another method to assign midi input chromatic scale notes to the voice frequencies.



You need to go with your mouse pointer on top of any voice freq rotary slider, you will see that it will be highlighted. Then press any midi key on your keyboard and this will be converted to frequency and assigned to that voice. (This action can be replicated on the iOS version by touching the rotary slider and pressing a midi key.

Further performance gestures..

Another performance gesture one could apply on Fundamental is the **gliding all the voice pitches together** like a pitch bend.



You do that by dragging your mouse on the Fundamental circle. From the center towards the upper edge or lower edge for a negative glide or positive glide. When you drag inside the circle the offset bend value will be shown as corresponding number.

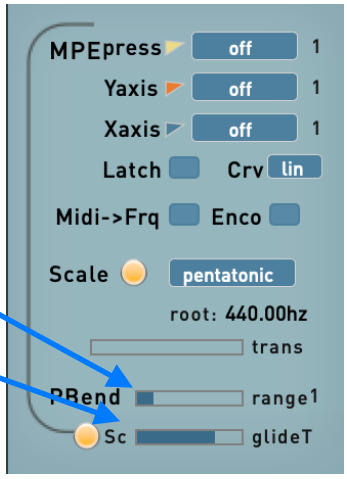
Then when you click inside the circle, the frequency values will go back to original settings.

You can also use **Midi pitch bend** for the same action, and likewise record your gesture on a midi track.

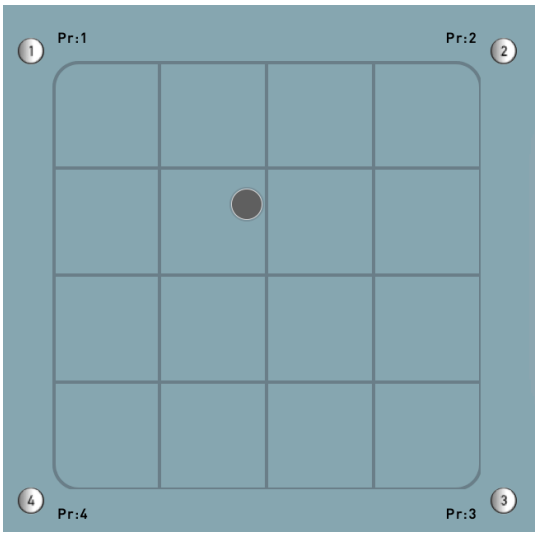
The portamento glide acts like a frequency shifter on all oscillator frequencies. However, if Fundamental is on harmonic mode, the fundamental frequency will follow the glide and the harmonics will be locked to it.

Or course one can assign the maximum glide range and the glide speed with these sliders.

When the **Sc** button is on, the glide will happen between the tune scaled oscillator frequency changes. By changing the **glideT** setting you can achieve very long glissandi.



There are some hard coded Midi Controller Messages addressing parameters of Fundamental2 :



- **Midi CC 14 :** X position of the Morph Circle.
- **Midi CC 15 :** Y position of the Morph Circle.
- **Midi CC 13 :** sets the scale transition slider value.
- **Midi CC 31 :** harmonic ratio slider value.
- **Midi CC 7 (Volume):** controls oscillator final volume. Use and send on channels 1 - 8 to address the respective oscillator.
- **Midi CC 10 (Pan):** controls oscillator final pan. Send on channels 1 - 8 to address the respective oscillator pan.

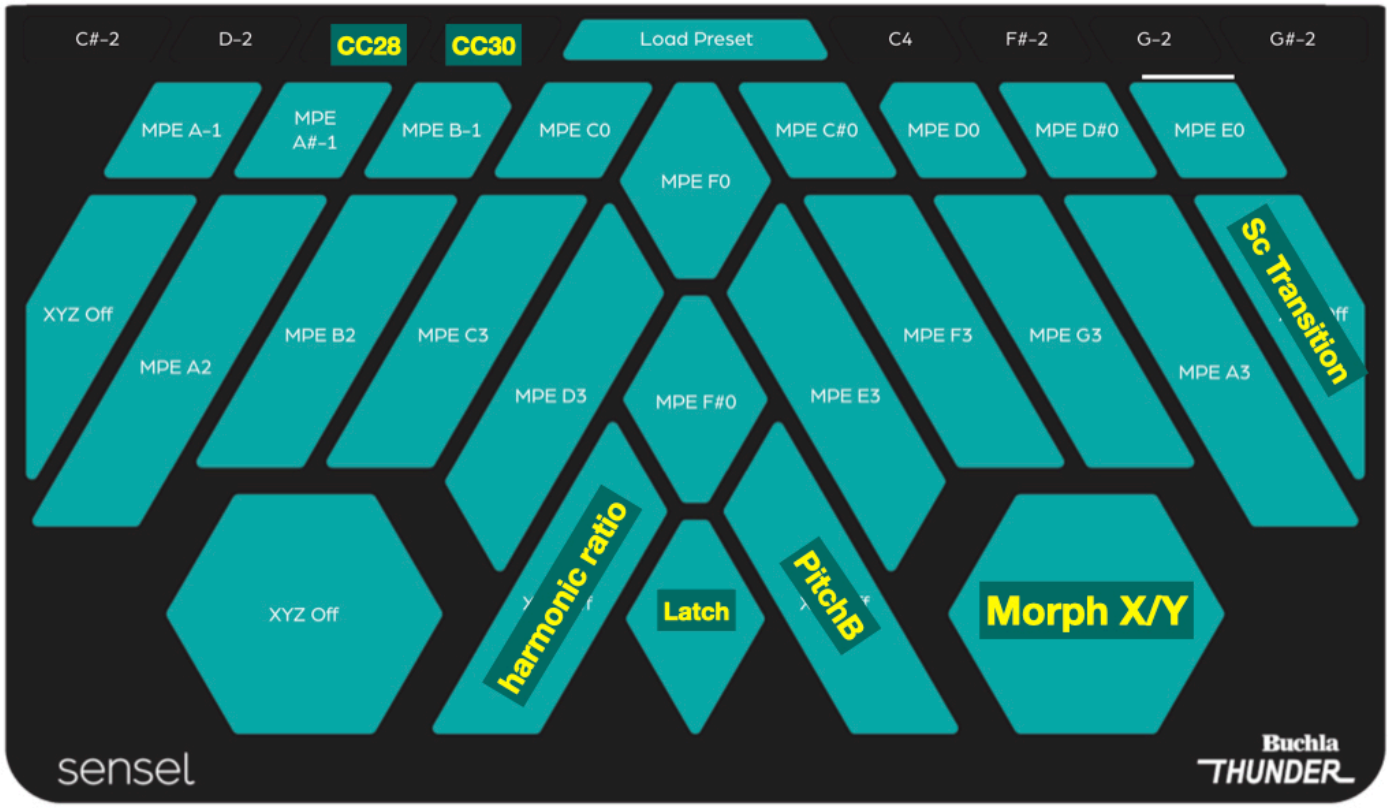
MPE controller mapping for Fundamental

We have collaborated with Sensel, to create a matching touch controller mapping system for the Fundamental synthesiser.

All you need is a Sensel Morph controller with Thunder Overlay. And you can download the map here.

Upload this map via the SenselApp to your Sensel Morph. The related mapping is on Preset 9, so please select it when using with the Fundamental.

Below you can see the functions mapped on the Thunder overlay addressing directly the Fundamental. You will see the midi keys assigned to each oscillator (MPE A2, MPE B2 , MPE C3 ….) and slider and button functions with custom CC mapping. However you can change these locations anywhere you like, this is just a template map.



enjoy!

Tuning scales on Fundamental

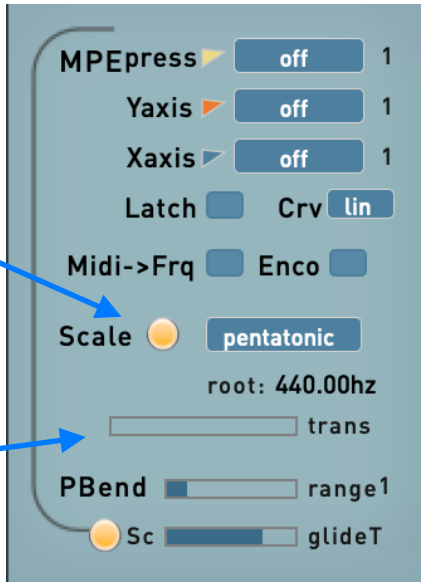
Scala format tuning scales can be imported to Fundamental and applied as pitch quantizers. There are already some example useful scales coming within the Fundamental package. The existing scala files are located at sonicLAB/Fundamental2/scl/ directory , and you can put new ones there and the list will update accordingly.

In order to make the tuning scale active, turn it on with its button. Also you need to choose a scale. A pop-up menu will let you choose from the existing scales of Fundamental. When you save a preset, these settings will be saved along with.

The reference frequency (called root on the app but the scale can go both upwards and downwards between (2hz and 14999hz) based on that reference value) can be typed directly on the interface.

By default, this is 440hz and your custom setting will be saved along with each preset.

Fundamental2 introduces a **scale transition** slider where you can gradually go from a defined scale quantization to non-scale state. Left end position of the slider is full scale state and the right end position of the slider is a no-scale state.



Technically the scales act like a pitch quantizers so the frequency calculated with the user setting + the freq. modulation being applied.

High precision Midi control on Fundamental2

Fundamental2 incorporates special handling of encoder style Rotary Midi CC knobs, and therefore it can map the incremental motion of this type of knob to high resolution oscillator frequency and gain values.

A Rotary encoder is an electro-mechanical device that converts the angular position or motion of its shaft to analog or digital output signals. Rotary encoders are used in a wide range of applications that require monitoring or control of industrial systems including robotics, computer input devices (mouse / trackball), many types of rotating platforms such as wind turbines etc.

There are two main types of rotary encoder: absolute and incremental. The output of an absolute encoder indicates the current shaft position, making it an angle transducer. For instance it gives an absolute value based on its angular position. (a value of CC 0-127 for example in MIDI case.)

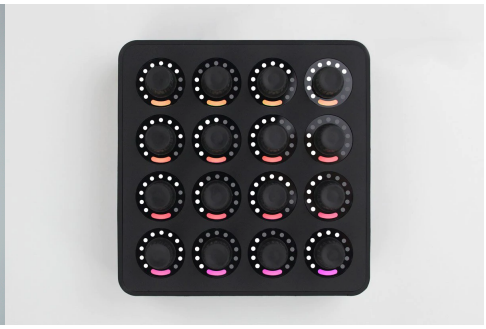
The output of an incremental encoder provides information about the motion of the shaft, which typically is processed elsewhere into information such as position, speed and distance. Therefore it produces a certain value immidiately when moved clockwise or anti-clockwise by stepwise interpretation of the movement.

Rotary encoder equipped MIDI controllers exist since a while on the market. For example, DAW controllers and desktop digital mixing consoles have incorporated this type of encoders well for modulating parameters such as track level, panning etc. The photo on the right shows a Mackie DAW controller with 32 Rotary Encoders, and each knob has a led display around it showing the absolute value of its dedicated parameter destination.

The advantage of these endless rotary knobs is that they act relative to the current state (or last recalled preset) of the parameter which they do control. The motion of the knob creates a relative value based on the stepwise incremental / decremental motion of the knob and this adds up to the last recalled state of the parameter.

This is a convenient non-destructive editing method, which is not possible with an absolute standard 0-127 Midi CC controller.

Today, on the market one can find several rotary encoder style midi controller units not just dedicated as DAW controller but for general purpose usage. To name a few the Intech EN16, Midi Fighter Twister, Behringer BCR2000 etc.



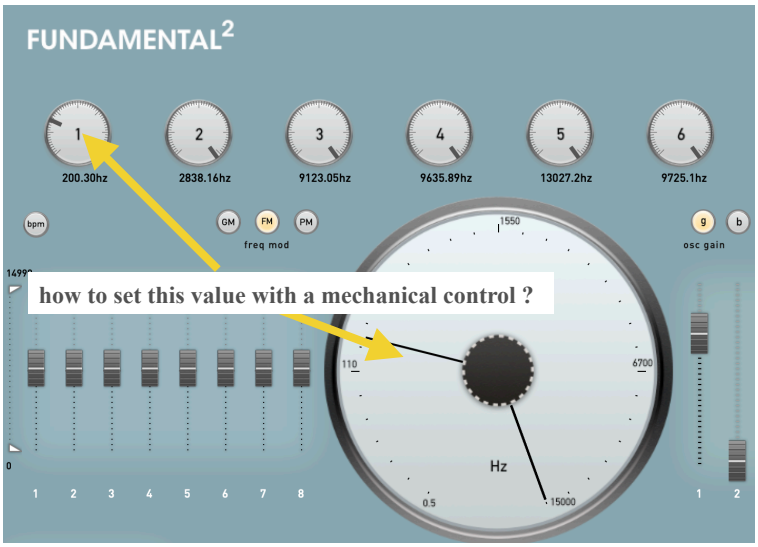
* We are not affiliated business-wise with any Midi Controller device producer. There are different units in price / quality on the market.

Why choose a Rotary Encoder control for Fundamental2 ?

Fundamental2 is a semi-generative synthesizer of which sound source is oscillators which model a Rohde&Schwartz test tone sine wave generator.

The sonic quality of the oscillator varies according its frequency and also the gain settings.

On its user interface one can set the frequency of an oscillator with a rotary dial knob, or type in the exact value as text or assign a midi controller on the DAW to control this parameter.



Standard Midi continuous controller data ranges between 0-127 integer values (7bit resolution) There is no way to address the frequency value range of the Fundamental2 oscillator (1hz - 15000hz) with this range of modulation data.

Standard Midi controller access is destructive. Imagine a Fundamental2 preset; all the frequency and gain settings of each oscillators matters to achieve its beautiful interference between sine waves and the emergent sonic quality of the generative engine. But when you touch a midi controller, its current absolute value will jump the current setting on the Fundamental oscillator, which is not what we ideally prefer.

It is not feasable the invent a custom communication method and build a dedicated hardware for this, and yet we need to still use Midi and existing hardware to achieve this.

What did sonicLAB achieve ?

- Therefore sonicLAB has decided to create a high resolution frequency mapper which utilizes the data coming from on an incremental endless Rotary encoder knob. And we succeeded reaching the entire frequency range of the Fundamental2 oscillators with one knob motion and with a **1hz** resolution. !
- There remained another problem, since each product on the market can send different Midi CC messages based on the direction of the knob motion. For example one product sends Midi CC 64 when the knob is being turned clockwise on each step, and Midi CC63 when anti-clockwise. Another product sends Midi CC127 and Midi CC0 for the same motion directions.
- What sonicLAB has achieved is a behavior agnostic to these values, it adapts itself with a simple twist. You just need to move the knob shortly in both directions and that's all, the message of the knob is captured.

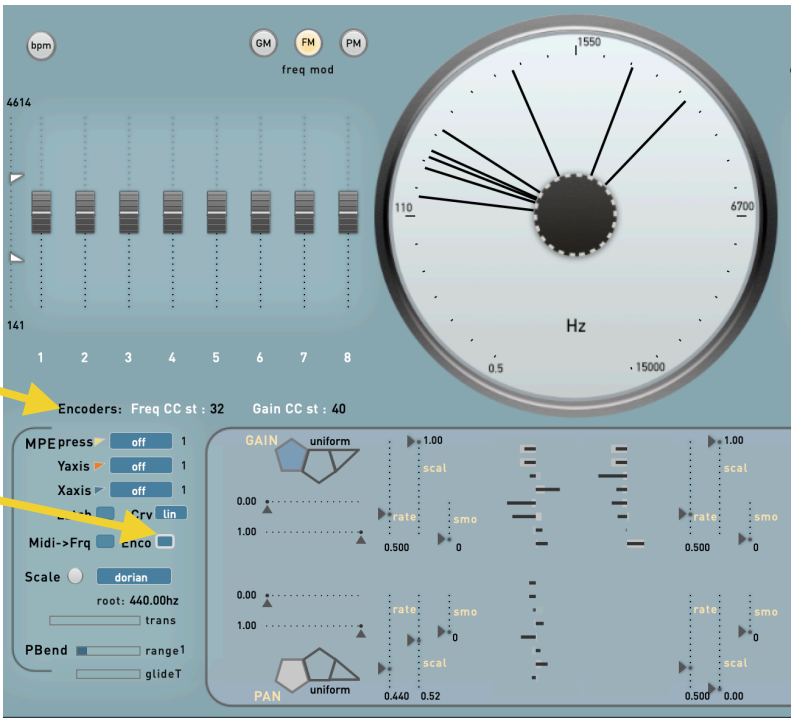
implementation on Fundamental2 desktop and iOS versions

You have purchased a rotary encoder Midi controller and connected it to your computer / iPAD.

Be sure that Fundamental2 receives it either from the DAW channel or your controller is selected as the input device on the standalone Fundamental2.

cllick on the **Enco** button and see that the Encoders CC numbers appear on the interface.

As mentioned before yon can control the frequency and the gain values of the 8 oscillators on the Fundamental2 with these Rotary encoders. Ideally you will need 16 of them to access all these parameters instantly.



You can specify the starting CC number for the frequency parameter control and gain parameter control on the interface.

For example the default Midi CC start number for the frequency control is 32. Therefore a Midi CC32 will address the frequency of the Oscillator1 and Midi CC33 for the freq. of the Oscillator 2. When we set the start as 32, the Midi CC's 32-40 are allocated automatically for the frequency control of the oscillators.

The default Midi CC number for the gain control starts at 40. Likewise the Midi CC40 controls the gain of the Oscillator1.

You can customize these starting numbers the way it fits to your controller setup or you can choose to change the Midi CC number of your Rotary encoders respectively.

Remarks on the use of Rotary Encoder control on Fundamental2

- Each step (incremental or decremental) of a Rotary Encoder movement will achieve a **1hz** relative change on the Oscillator frequency and an amount of **0.01** change on its gain level. This kind of frequency resolution is never achievable with a standard CC0-127 control.
- We advice to avoid rapid jumps on the Rotary Encoder movement. This implementation is meant to focus on fine movements and fine adjustments of the destination parameter. sonicLAB has implemented adaptive **motion -> freq scale mapping** and higher rotation speeds also maps to faster step intervals in an intelligent way. However the Rotary knobs mechanically won't comply to same results when rotated fast. This depends on the hardware quality and there is not much to do on the software side to fill these gaps.
- A useful analogy would be the use of mechanical mouse / trackballs during 90's (before touch pads) where one could not precisely know the advance of the mouse pointer when moved fast. Slow movements were always more predictable. Therefore when you rotate the Rotary knobs fast, the jump on the destination value will vary.
- sonicLAB implementation aims to adapt to any Rotary encoder whatever CC value it sends for its incremental and decremental change. However there is the logical restriction that the CC value of an incremental change (a clockwise rotation step) has to be higher then the value sent when there is a decremental change (a anti-clockwise rotation step) . When this condition is fulfilled, any Rotary Encoder will work within our implementation.
- First test the standalone version of Fundamental2 with your Rotary encoder controller. When confirming that it works, you can test it in your DAW. Every DAW has different midi port input / output handlings , and we cannot know and test each DAW. This is why that it is important your test first outside a DAW with the standalone version.
- First time you touch a Rotary encoder knob to control a parameter on Fundamental2, apply a little twist back and forth on the knob so that the CC values for the incremental and decremental motion becomes evident to the software.

APPENDIX

The sonic resources for Fundamental have been a vintage Rohde&Schwartz Sine Wave Generator.

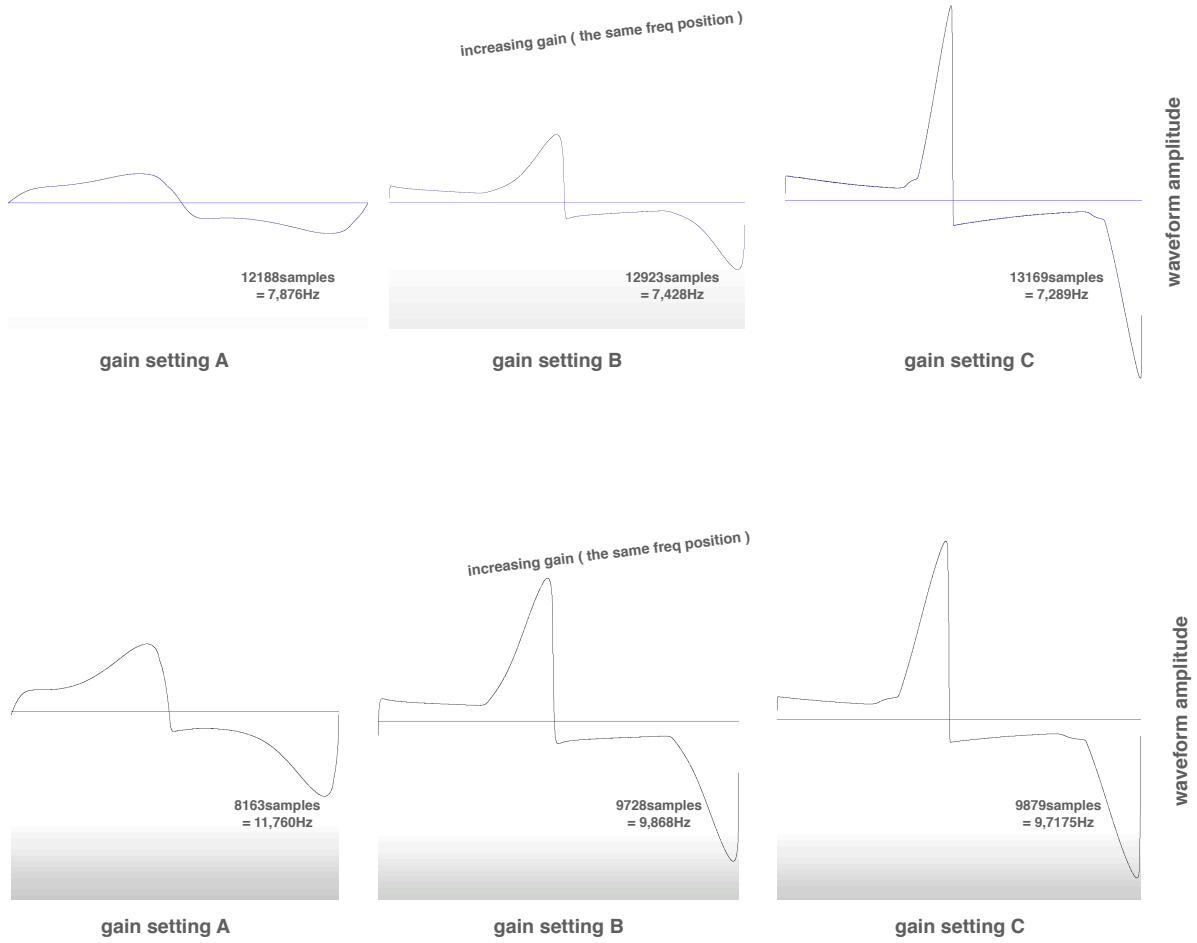
As its name suggests, the unit gives us sine waveforms and by setting a frequency value and also the gain setting for the signal strength. Furthermore these controls open the doors to many other waveforms depending on the frequency setting and the gain setting, a timbral palette unique to this test waveform generator.

It is exactly this nonlinear aspect and tone color palette on which we are interested in, and not just a pure sine wave output. So the work has begun with recording many frequency sections and by continuously changing the gain setting on each of them.

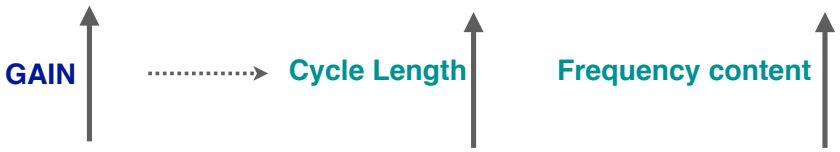
This means that we have potentially captured 1000's of waveforms with their unique frequency and gain setting as a result of simple gestural actions on the device controls. Then it was time to pass the task to a computer and a custom software to analyze these recordings, data mining, for what a computer is known the best.

The figures below show the interesting dynamics of the vintage tone generator in terms of waveform output related to its control settings. Especially below 50Hz the unit generates anything but a pure sine wave.

timbral dynamics of a vintage Rohde&Schwartz Sine Wave generator



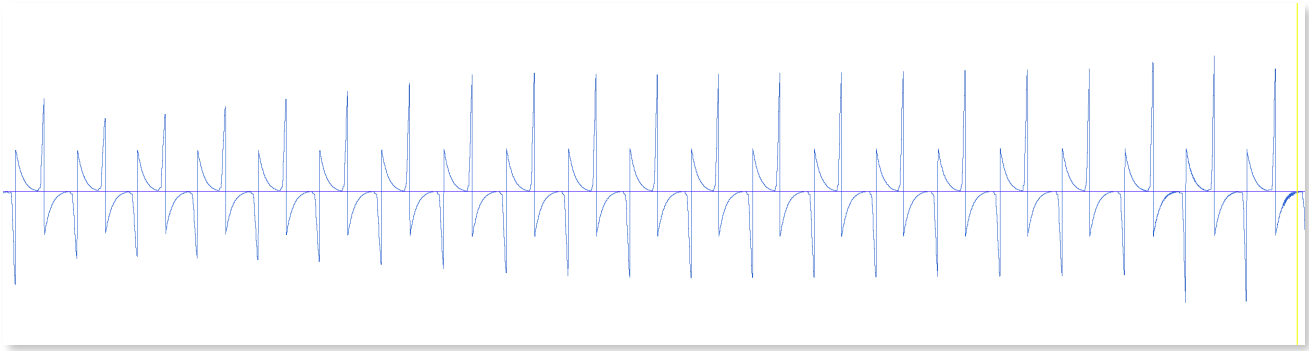
more gain —> higher frequency content, sharper transitions
—> increasing waveform length



All these recordings have been analyzed , and corresponding wavetables have been created, categorized and stored with a custom helper software developed by sonicLAB. 100's of wavetables are labeled and precisely kept without any compression.

The wavetable sizes vary from 57118 to 17 samples depending on the frequency setting of the R&S unit. The wave shape is changing according to the gain setting, and also there is dependency between the wavetable size and the gain setting even when the frequency setting is being kept unchanged. Sounds like such hard work was waiting for us !!

Generic wavetable synthesis suggests that the waveforms being used by a wavetable lookup oscillator share the same wavetable size (mostly a number powered by two, e.g. 2048) While this approach benefits a great amount of memory and CPU optimization, the amount of data lost is unacceptable for us to maintain the timbral quality of the vintage tone generator especially on its lower registers.



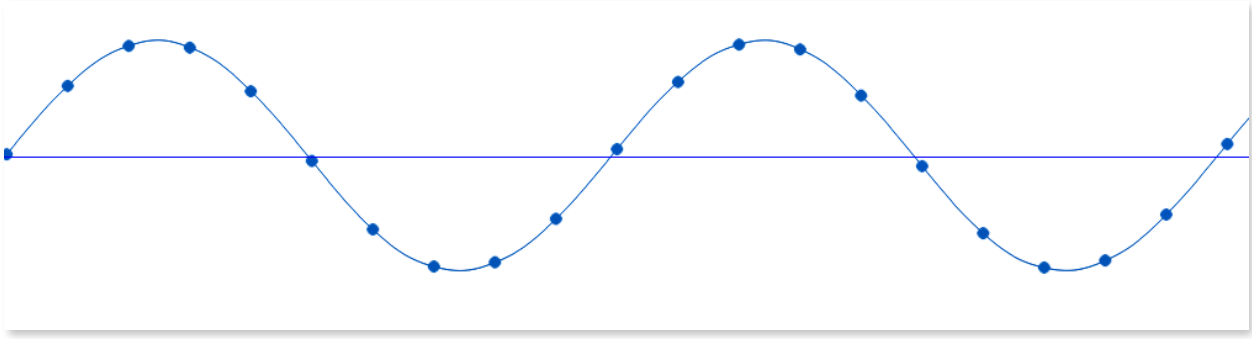
The purpose of Fundamental application is to deliver all possible frequency output with the intermediate gain steps in full continuum, not even possible to obtain on the vintage tone generator controls.

To achieve this, an accurate poly-morphing wavetable synthesis algorithm has been developed. This algorithm, taking a magnificent amount of depth in C++ coding, takes in account all the available data captured from the vintage unit (no loss for optimizations , no compromise) and dynamically calculates the desired waveform sample to be output at that frequency and gain value.

The frequency and gain values can be changed with sample precision and within infinitesimal steps; you will hear no glitch, no jumps. And just reminding that how things can get difficult and dangerous when dealing with sine waves, no mercy at all ! There are 4 oscillators running in background sharing the job to calculate that sample in question and in real time. Let's imagine this as a poly network of oscillators each contributing to the rendered waveform. Thanks to this algorithm, continuity in performance by keeping the rich palette of the original vintage tone generator has been achieved. The continuous rendering of the oscillators don't just answer to abrupt changes of user manipulation of parameters but also to complex LFO modulation applied on gain and frequency settings driven by stochastic and other mathematical functions.

The singularity/limits of the algorithm is that, after some frequency (chosen here as 3800hz after empirical tests) we have decided to morph to a computed sine wave for several reasons. Sampling of a sine wave at high frequencies of the R&S unit becomes meaningless as the amount of data lacks a faithful representation of the original waveform.Â

See below, a sine wave has been approximated artificially (drawn) from a limited amount of sampled data. Otherwise there are round off errors, timing quantizations distorting the frequency image and other things leading to aliasing problems, artifacts of the digital world.



Therefore we present you a faithful and super continuum representation of a vintage tone generator between 1hz - 3800hz and after that point the algorithm switches to a pure mathematical sine wave.

Sinan Bökesoy, May2020