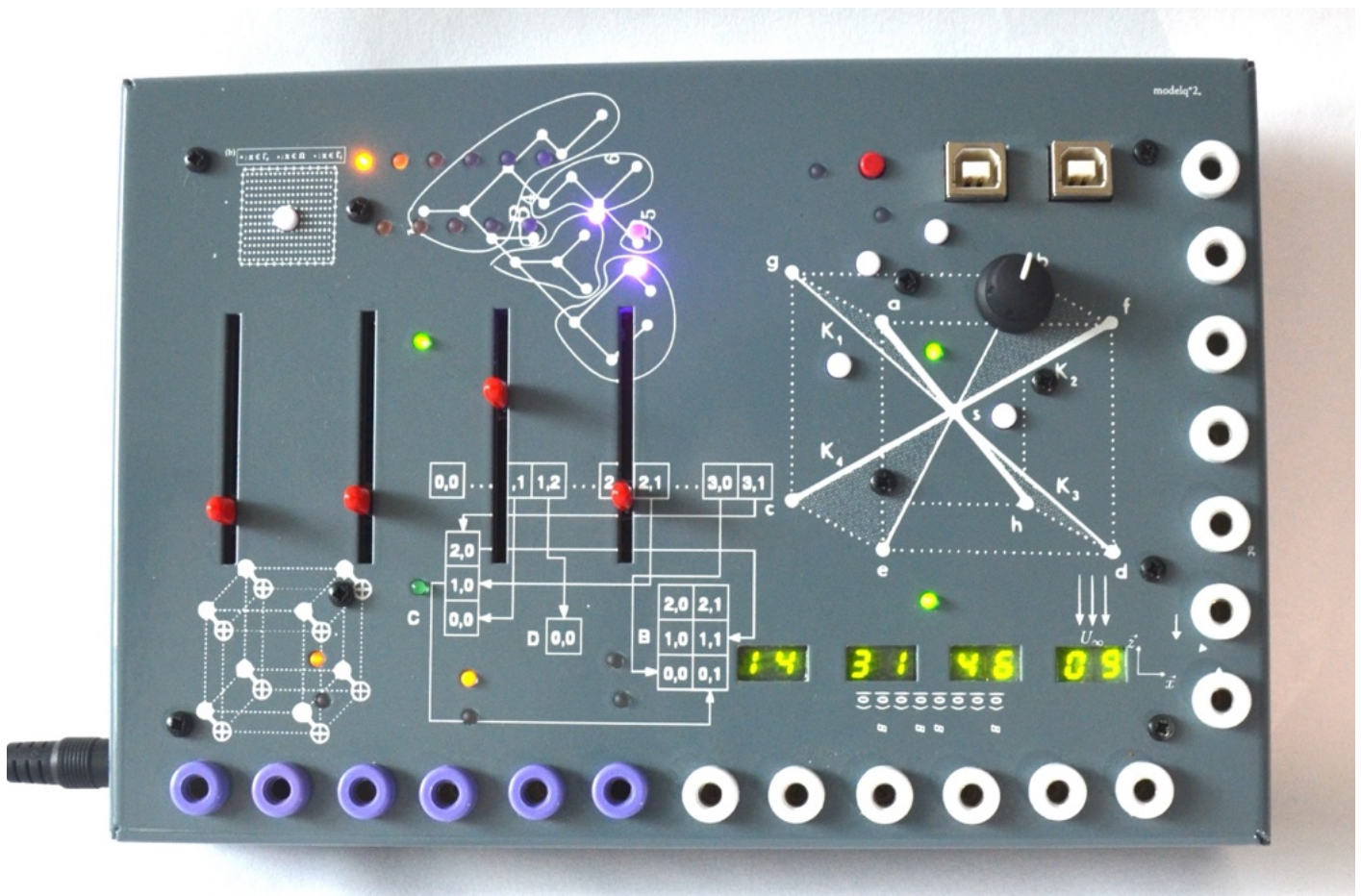
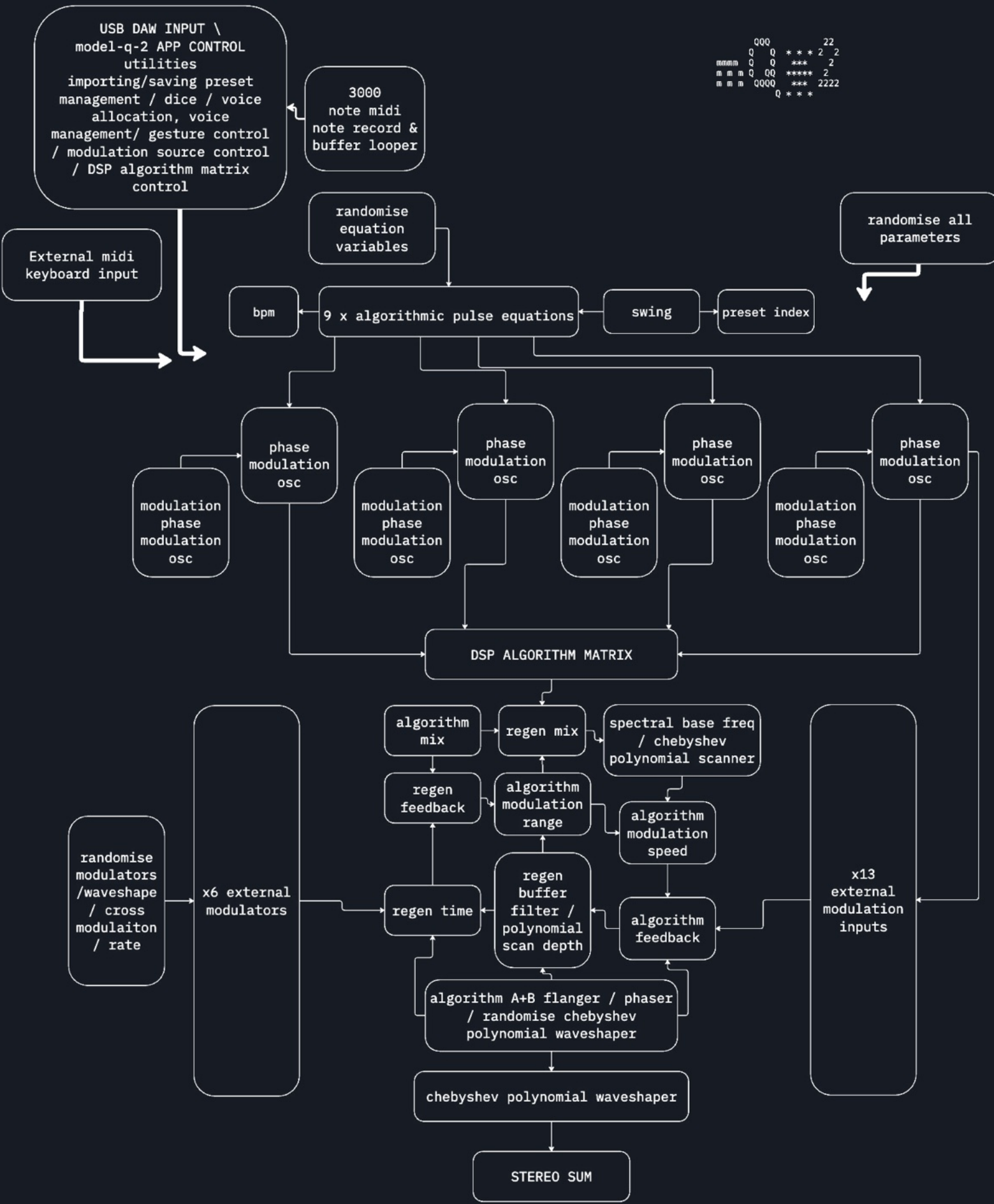


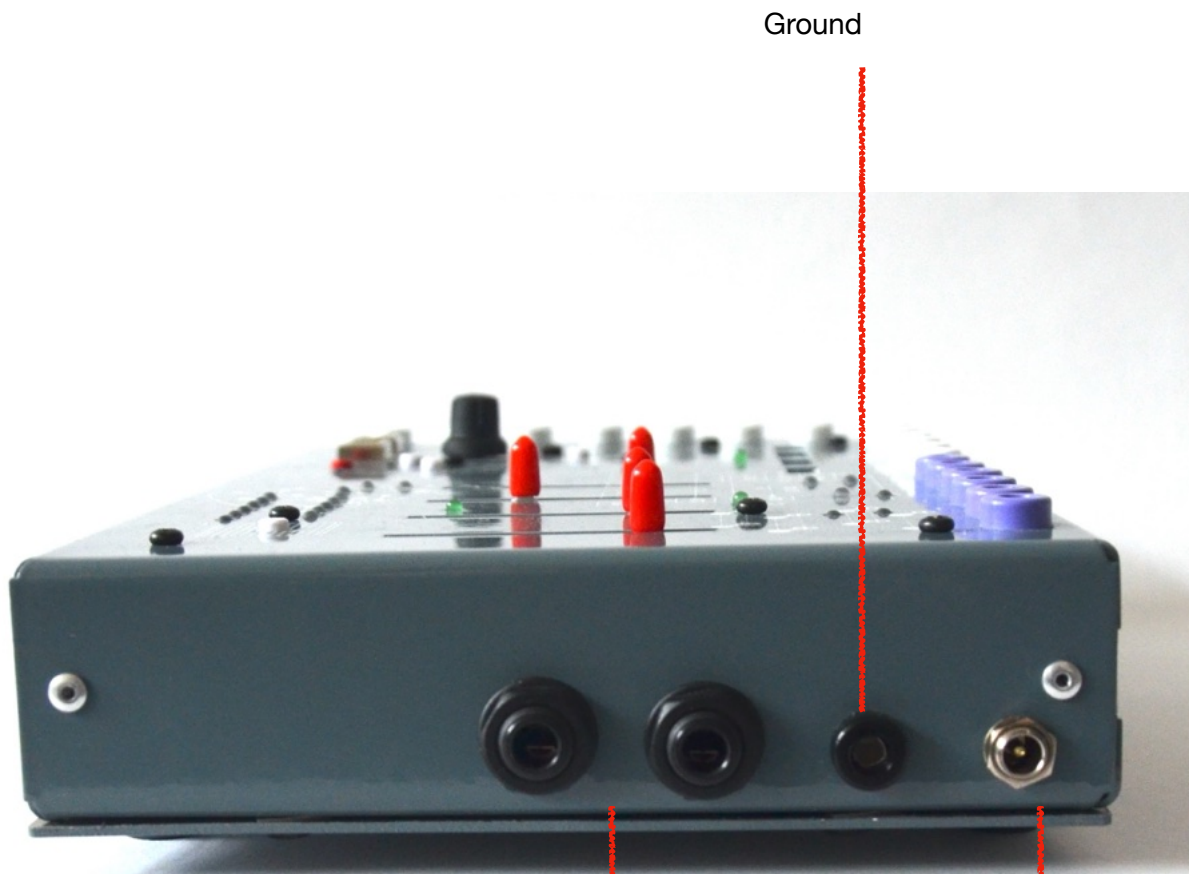
Model-q-2

Model-q-2 is a small complex system built on 10 algorithmic equations capable of infinite variable polyrhythmic pulses, 8 phase modulation oscillators, a DSP algorithmic matrix including app and gesture control, a chebyshev polynomial waveshaper of the 2nd kind and 6 external complex modulation sources.



- Signal Architecture
- Quick hardware overview
- Pulse Algorithms, variables, bpm, swing.
- Phase modulation Oscillators
- DSP algorithm matrix
- Modulators and external inputs / outputs
- Chebyshev polynomial waveshapers
- OSX App and gesture
- Saving / preset index





Ground

Left / right out

12v/2A/2.1mm centre
positive psu.

Whilst all parameters on model-q-2 are editable via the app the hardware allows all functionality.

Quick hardware overview:

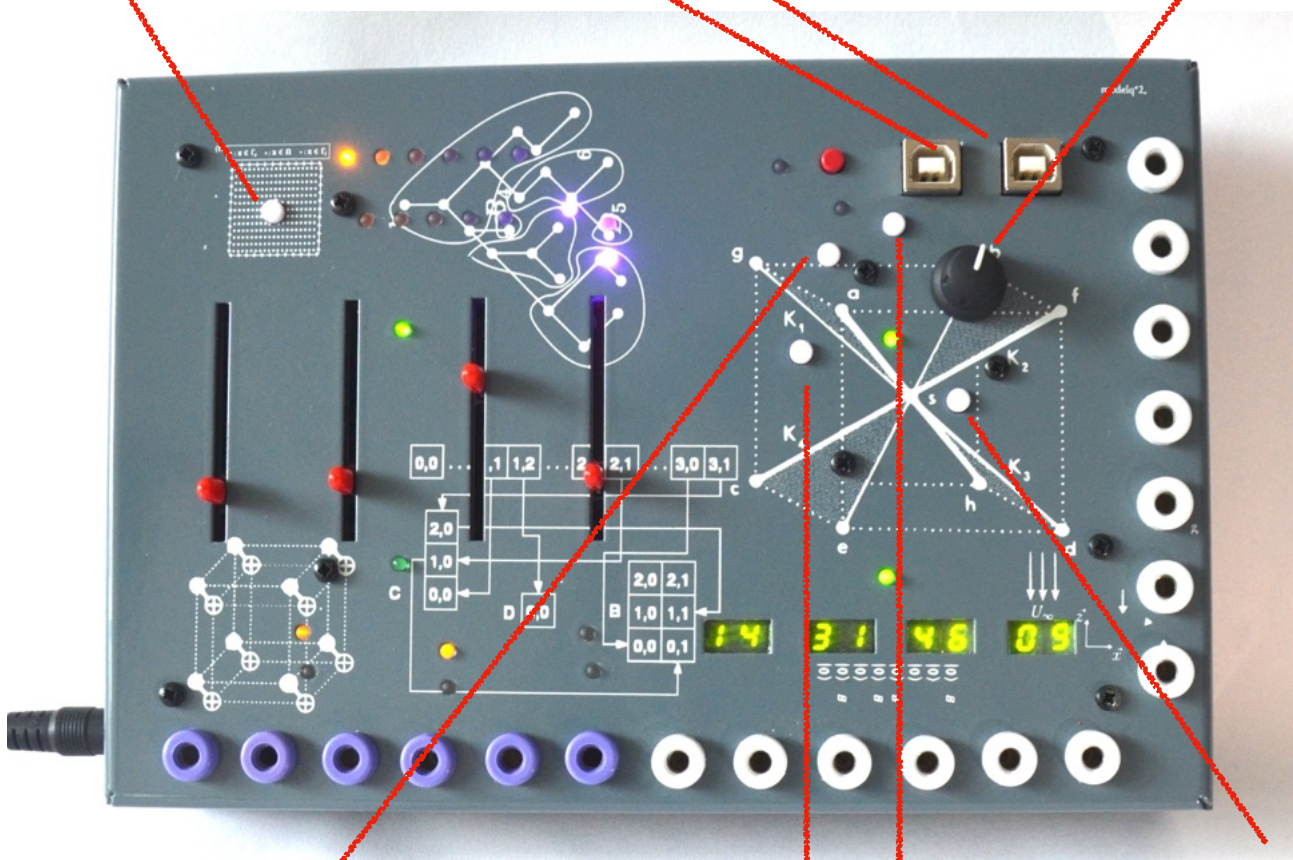
This switch alternates between allowing the encoder to control scrolling through the bank of parameters for slider manipulation or allowing the encoder to control its core logic. BPM>PULSE
ALGORITHM>SWING>PRESET INDEX.

The slider bank parameter list is noted below.

When in the encoders core logic role pressing down on the encoder will cycle through BPM>PULSE
ALGORITHM>SWING>PRESET INDEX.

Scrolling the encoder left or right whilst in any of these modes will edit that modes parameter.

Left usb : DAW integration
Right usb: external midi keyboard



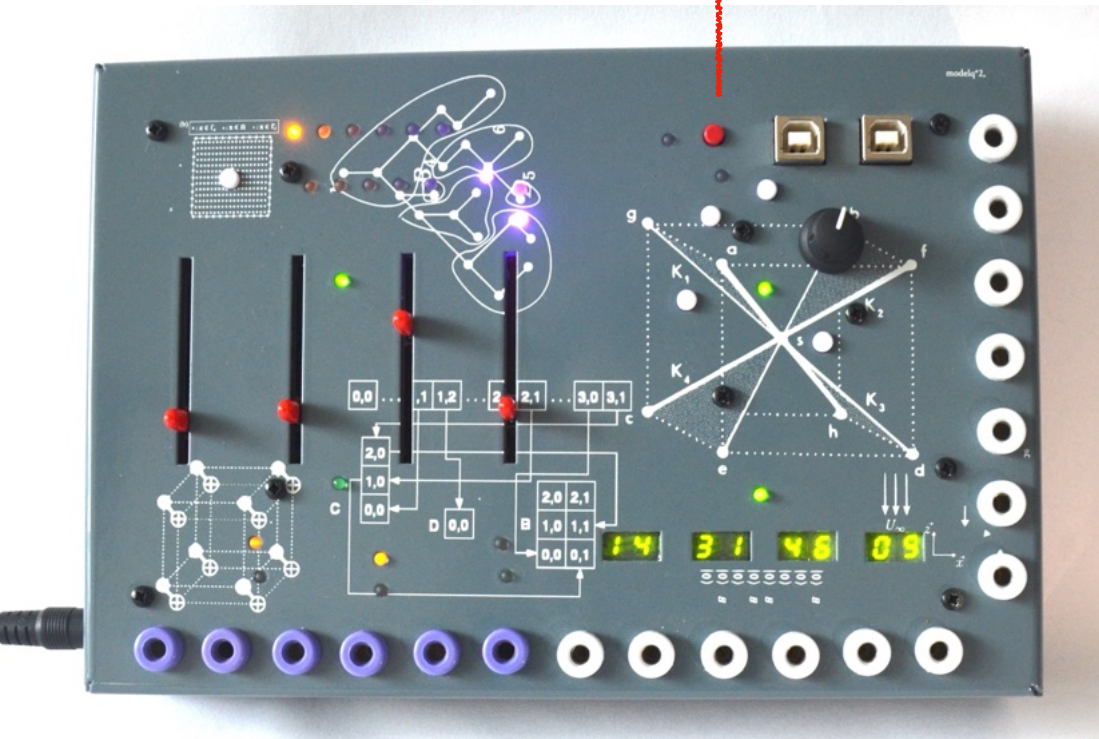
Switch between flanger / phaser algorithm in dsp matrix and randomly choose one of 40 pre defined chebyshev polynomial wave shape outputs

Triple tap to stop start the pulse equation, press once to randomise all variables of the chosen pulse algorithm.

Randomise all parameters / load preset when in PRESET INDEX MODE.

Save patch to preset index.

When using app / daw integration, pressing this once will activate the midi buffer, once a note is triggered or external pulse registered the midi looper will begin recording, press again to disarm and loop will automatically begin looping. Press again to clear loop.



Slider bank parameter list

slider 1

slider 2

slider 3

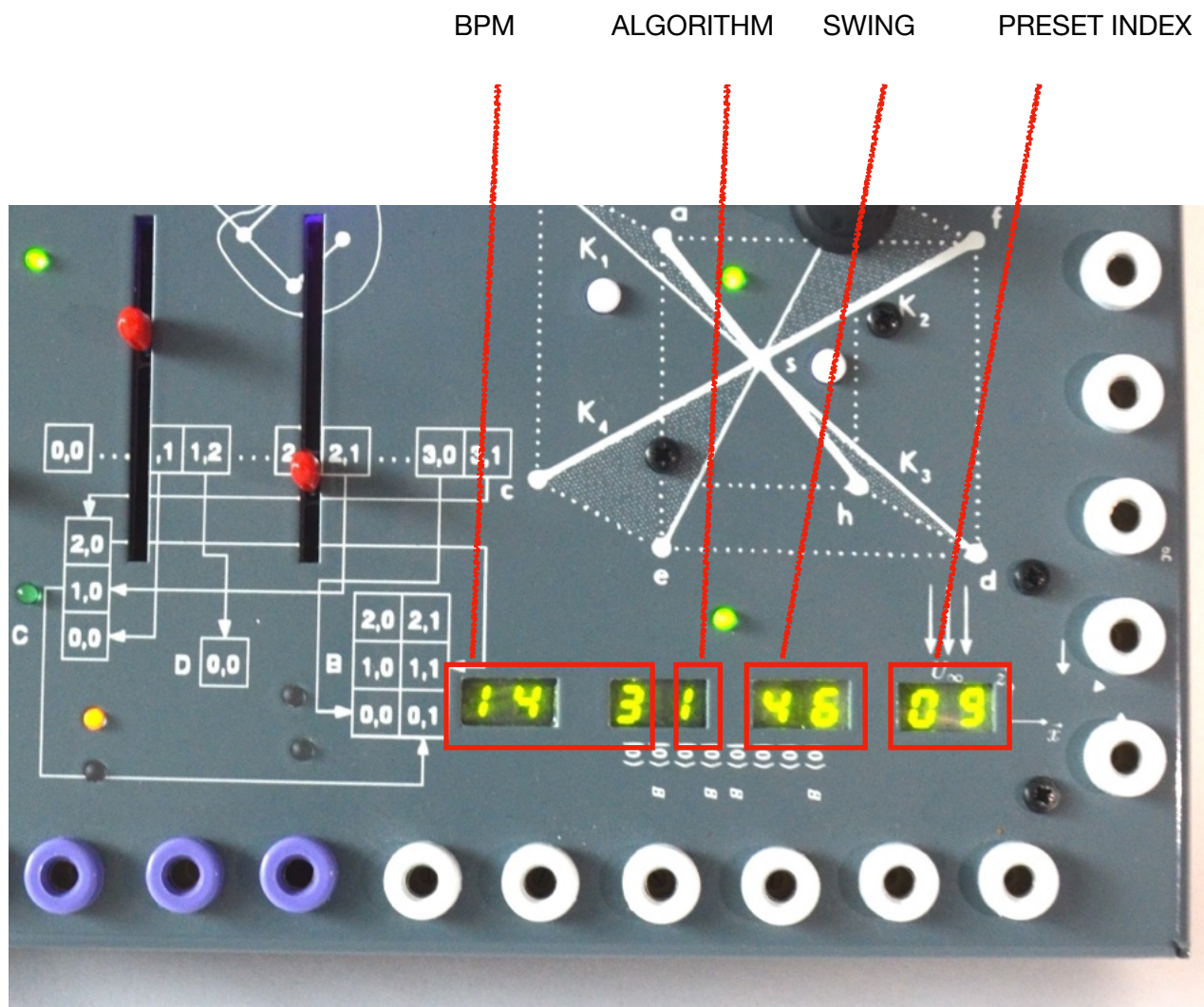
slider 4

Pm modulation depth A	Pm modulation frequency A	Pm frequency A	Decay A
Pm modulation depth B	Pm modulation depth B	Pm frequency B	Decay B
Pm modulation depth C	Pm modulation depth C	Pm frequency C	Decay C
Pm modulation depth D	Pm modulation depth D	Pm frequency D	Decay D
Algorithm mix	Regen mix	Spectral base frequency / chebyshev polynomial scanner	Regen feedback
Algorithm modulation range	Algorithm modulation speed	Regen time	Regen buffer filter / polynomial scan depth
Algorithm feedback	Mod A rate	Mod A shape	Mod A depth
Mod B rate	A X B MOD	Mod B depth	Mod C rate
Mod C shape	Mod C depth	Mod D rate	C X D MOD
Mod D depth	Mod E rate	Mod E shape	Mod E depth
Mod F rate	E X F MOD	Mod F depth	

- **Pulse Algorithms, variables, bpm, swing.**

Modelq2 features **10** base pulse algorithms that users can choose to derive complex and non linear pulse results. The key feature being that users can achieve near infinite rhythmic patterns, with in any of the 10 algorithms using the random variables button with in a user defined **algorithm**, **BPM** and **Swing**.

This also shines when contributing melodic notes to any given pulse sequence.



- i is the variable representing the index in your pulse pattern.
- bpm is a parameter that needs to be defined externally.
- `rand()` represents a call to a random number generator.
- Mathematical constants and functions like π , $\exp$, $\sin$, $\cos$, $\log$, $\tanh$, and $\sqrt{}$ follow their usual definitions.
- The output is binary, determined by whether the expression exceeds a threshold.

The Pulse Algorithms are as follows.

$$f(i) = \text{random}(2)$$

This represents a function where the value at position i is either 0 or 1, randomly chosen.

$$f(i) = (i \bmod 5 = 0)$$

This describes a pattern where the output is HIGH if i modulo 5 is 0; otherwise, it is LOW.

$$f(i) = (i \bmod 7 = 0)$$

This represents a pattern where the output is HIGH if i modulo 7 is 0; otherwise, it is LOW.

$$f(i) = (\text{random}(100) < 50)$$

This function represents a probabilistic distribution where the output is HIGH if a random number between 0 and 99 is less than 50, effectively, ↓ 50% chance of being HIGH or LOW.

$$\begin{cases} \text{HIGH} & \text{if } i = 0 \\ (\text{random}(100) < (\sin(\frac{i\pi}{10}) + 1) \times 50) \text{ and } ((i > 1 ? f(i-2) : \text{LOW}) \oplus (f(i-1) \vee (i > 2 ? f(i-3) : \text{LOW}))) & \text{otherwise} \end{cases}$$

$$f(i) = \left(\frac{\exp(\sin(i))}{\log(i+1)} > 1 \right)$$

This equation evaluates the ratio of the exponential of the sine of i to the logarithm of $i + 1$. If this ratio is greater than 1, the output is HIGH; otherwise, it is LOW.

$$f(i) = (\text{random}(100) < (\sin(\frac{i\pi}{16}) + 1) \times 50 \times (1 + 0.1 \exp(-0.05i)))$$

This expression modulates a probability based on a sine wave, then adjusts this probability with an exponential decay factor to determine if the output is HIGH or LOW.

$$\begin{cases} \text{random}(2) & \text{if } i = 0 \\ (\text{random}(100) < 50) ? f(i-1) : \neg f(i-1) & \text{if } i = 1 \\ (\text{random}(100) < 75) ? f(i-1) : \neg f(i-1) & \text{if } f(i-1) = f(i-2) \\ (i \bmod 5 = 0) ? \neg f(i-1) : f(i-1) & \text{otherwise} \end{cases}$$

This case uses a complex decision pattern based on the state of the previous outputs, with probabilistic and periodic modifications.

$$f(i) = (\text{random}(100) < 30)$$

This simple probability check determines if the output is HIGH (30% chance) or LOW (70% chance).

↓

$$\begin{cases} \text{HIGH} \\ \left((\text{random}(100) < \left(\sin\left(\frac{i\pi}{10 \times \frac{60}{\text{bpm}}}\right) + 1\right) \times 50) \text{ and } ((i > \text{beatOffset}) ? f(i - \text{beatOffset}) : \right. \\ \left. : \text{LOW}) \oplus (f(i-1) \vee (i > 2 \times \text{beatOffset}) ? f(i - 2 \times \text{beatOffset}) : \text{LOW}) \right) \text{ otherwise} \end{cases}$$

This case dynamically adjusts cellular automaton rules and modulation factors based on BPM, integrating musical tempo into the probabilistic model.

$$f(i) = (\text{random}(150) < \left(\sin\left(\frac{i\pi}{16}\right) + \cos\left(\frac{i\pi}{12}\right) + 1\right) \times 50 \times \left(1.0 - 0.05 \frac{(i \bmod 20)^2}{400}\right) \times (1 + 0.1 \sin\left(\frac{i\pi}{4}\right) \exp(-0.05(i \bmod 10)))$$

This complex case combines several trigonometric functions with polynomial decay and exponential adjustments to modulate the probability of HIGH output, using a random threshold check against an increased range.

$$\begin{cases} \text{HIGH} & \text{if } \frac{\exp\left(\sin\left(\frac{i\pi}{8 \times \frac{\text{bpm}}{120}}\right) \times \cos\left(\frac{i\pi}{12 \times \frac{\text{bpm}}{120}}\right) \times \tan\left(\frac{i\pi}{16 \times \frac{\text{bpm}}{120}}\right)\right)}{\log(i+1+0.1 \times (i \bmod 10)^2)} > \left(1.0 + 0.5 \times \sin\left(\frac{i\pi}{20 \times \frac{\text{bpm}}{120}}\right)\right) \times (f \bmod(i, 5) + 1) \\ \text{LOW} & \text{otherwise} \end{cases}$$

This complex expression integrates various trigonometric interactions modulated by BPM, applies a dynamic threshold influenced by BPM, and employs a rapid modulation pattern for decision-making regarding HIGH or LOW output based on the calculated "complexity value."

$$\begin{cases} \text{LOW} & \text{if } (i \bmod 16) \geq 4 \\ \text{HIGH} & \text{if } \frac{\exp\left(\sin\left(\frac{(i \bmod 16)\pi}{16}\right) + 0.5 \times \cos\left(\frac{(i \bmod 16)\pi}{8}\right)\right)}{\log((i \bmod 16) + 1 + 0.1 \times ((i \bmod 16))^2)} > 1.0 + 0.5 \times \sin\left(\frac{(i \bmod 16)\pi}{8}\right) \\ \text{LOW} & \text{otherwise} \end{cases}$$

This case involves a repetitive gating pattern within a loop of 16 steps, incorporating oscillatory components that reset within the loop length. The output HIGH or LOW is dynamically determined based on an oscillation-driven complexity value, compared against a dynamically adjusted threshold, but only when the gate is open; otherwise, the output is forcibly set to LOW.

$$f(i) = \left(\frac{\exp(\sin(\frac{i\pi}{1680}))}{\log(i+1)^2} + 0.5 \cos\left(\frac{i\pi}{\text{bpm}}\right) > 9.5 \right)$$

$$f(i) = \left(\frac{(\exp(\sin(i) + \cos(\frac{i}{3})))^{\text{bpm}}}{\log(i+1) \cdot \log(i+2)} > 9.2 \right)$$

$$f(i) = \left(\frac{(\exp(\sin(i) \cdot \tanh(i)))^{\sqrt{\cos(i)^2}}}{\log(i+1) + \left| \sin\left(\frac{i - \text{rand}() \% 100}{3.14}\right) \right|} > (\text{rand}() \% 2 + 1) \right)$$

Phase modulation Oscillators

Of the 8 phase modulation oscillators 4 are primary oscillators with each having a secondary oscillator for phase modulation. Each oscillator has 4 primary functions.

Phase modulation depth, Phase modulation frequency, Primary Frequency, decay.

DSP algorithm matrix.

The algorithm matrix compromises a deep set of effects for sound design chains.

Algorithm mix:

overall mix of dsp matrix

Regen mix:

delay and regeneration overall mix

Spectral base frequency / chebyshev polynomial scanner:

this sets the base spectral frequency for the flanger and phaser algorithms individually. This also scans and interpolates the 40 predefined chebyshev polynomials.

Regen Feedback:

delay regeneration feedback

Algorithm modulation range:

Depth of algorithm modulation source

Algorithm modulation speed:

Speed of modulation source

Regen time:

Delay and regeneration time

Regen buffer filter / polynomial scan depth :

If regen mix is high this slider will filter the buffer from lowpass to high pass. This slider will also adjust polynomial scan depth. Smooth interpolation to static.

Algorithm feedback:

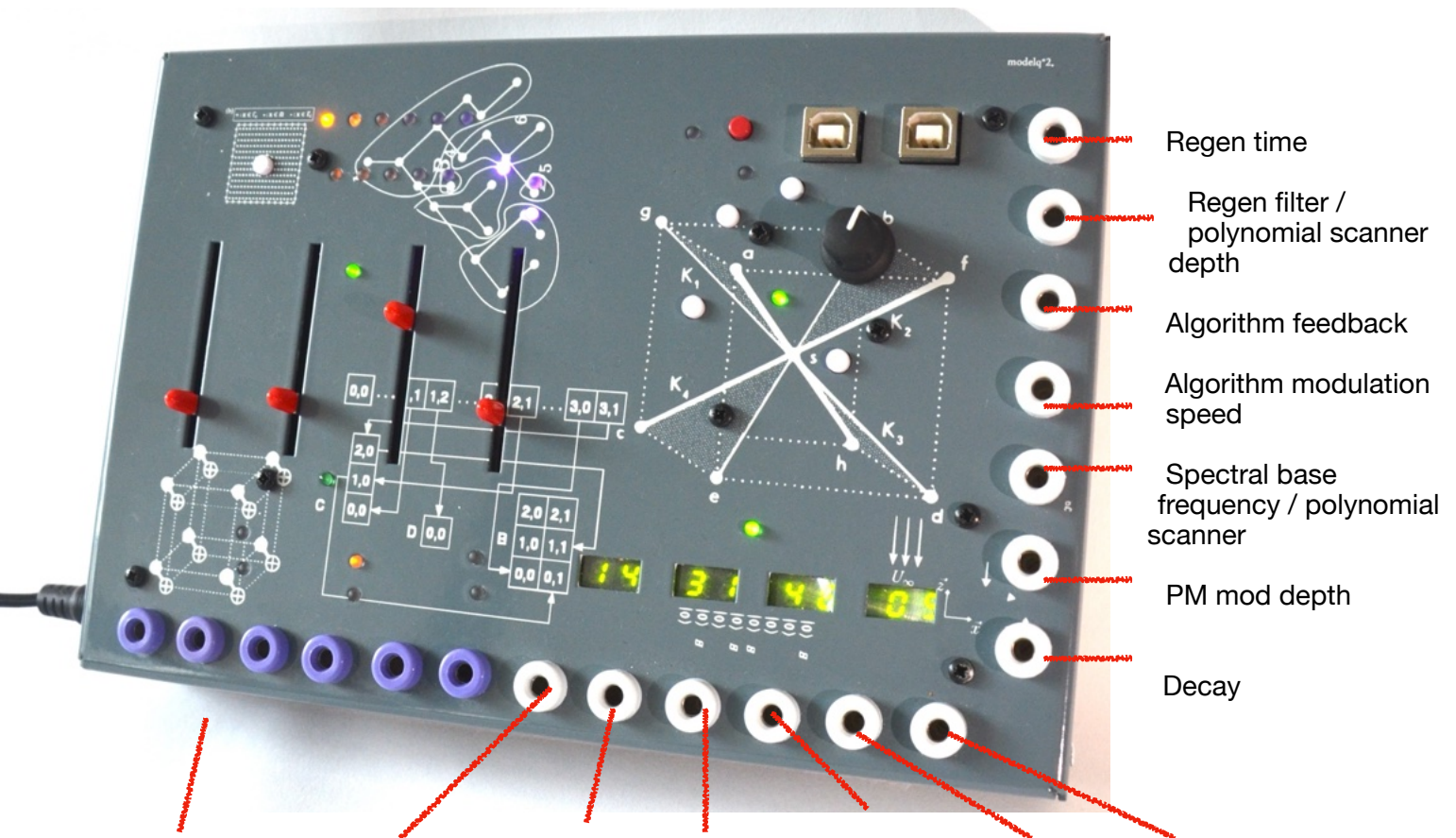
Feedback of flanger and phaser algorithms

Algorithm A & B button:

This button will switch between experimental phaser and flanger algorithms. It will also randomly choose one of the 40 predefined chebyshev polynomials for the waveshaper output.

Modulators and external inputs / outputs

Modelq2 has 6 external modulators that can be used to modulate many of the sources.
Each modulator has modulation rate, waveshape and depth capabilities.
Users can also cross modulate sources with each other for more complex outcomes.

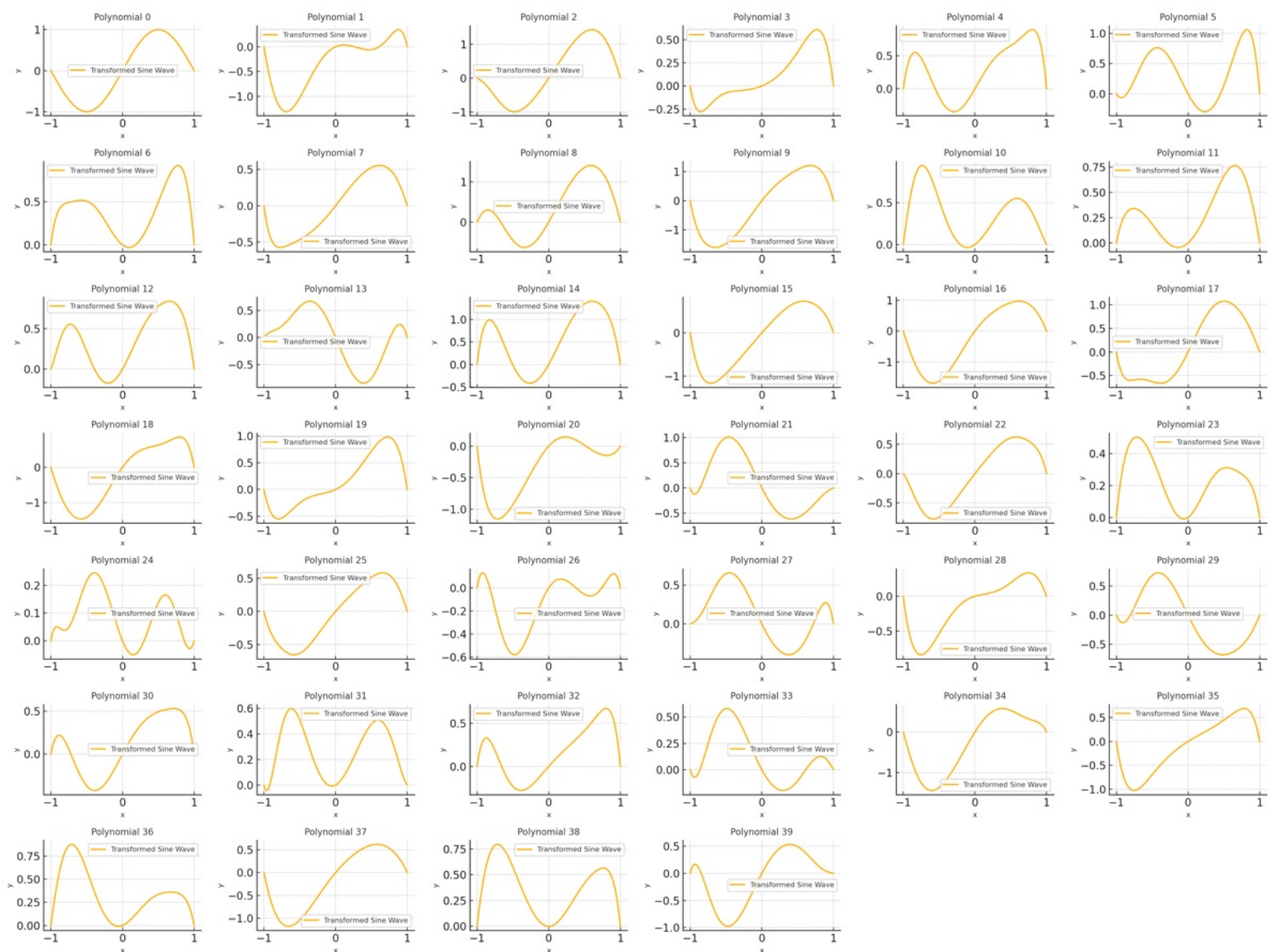


MODULATORS X6.

Decay - PM mod depth - decay - PM mod depth - decay - PM mod depth

Chebyshev polynomial waveshapers

Post output from the **DSP matrix modelq2** has stereo waveshapers that utilise chebyshev polynomials of the 2nd kind. 40 predefined polynomials each with 10 harmonics. These polynomials can be smoothly interpolated with each other and provide a unique harmonic contribution to the phase modulation cores.



OSX App and gesture

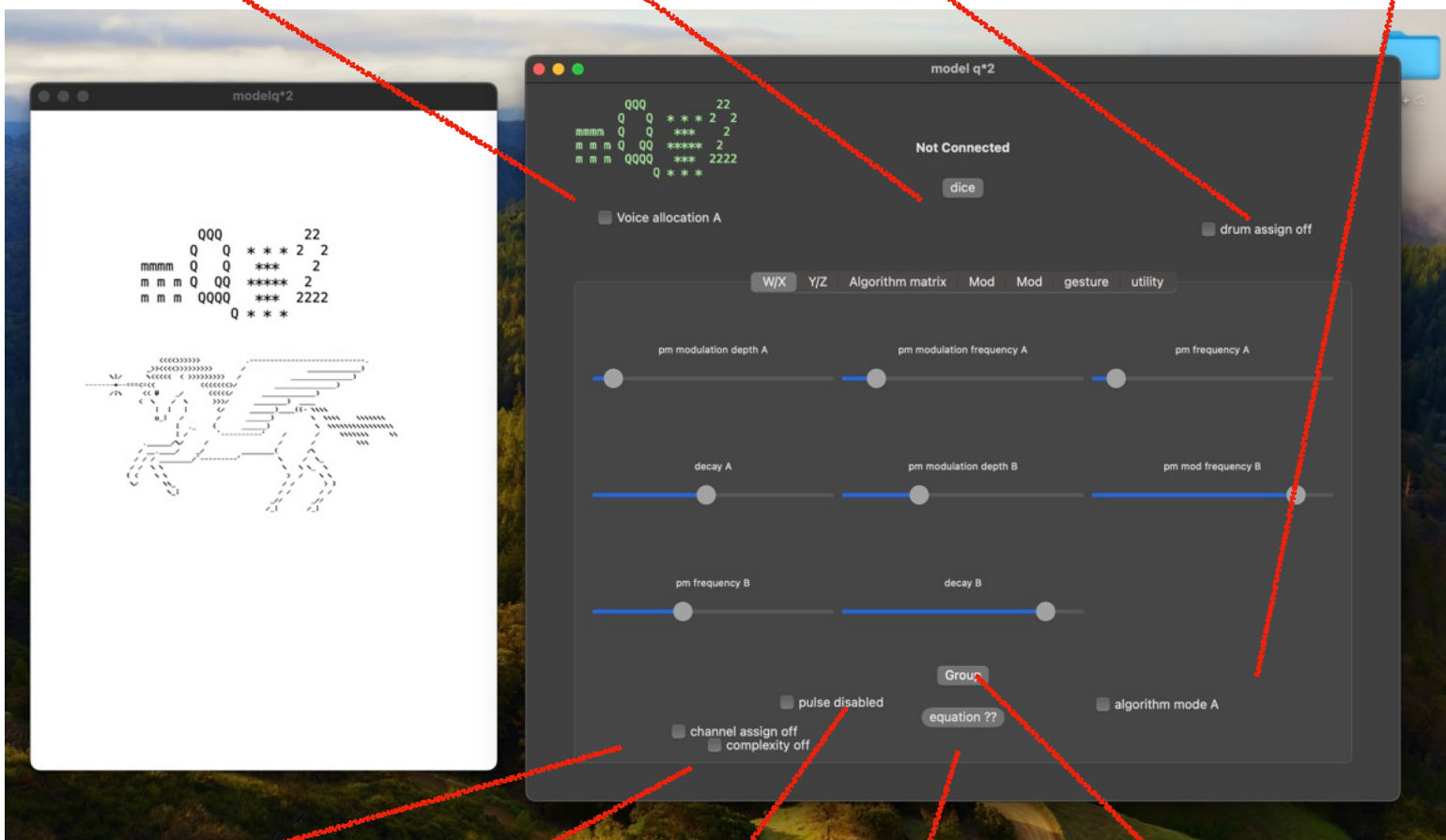
Modelq2 ships with an **osx** app designed to share full functionality with the hardware including preset saving, dice / random functionality, gesture control and more.

Switch between voice stack or round robin

Randomise all parameters

Drum mode on / off
Voice 1: c0
Voice 2: c#0
Voice 3: d0

Switch between DSP algorithm A & B



Channel assign on
assigns each voice a midi channel so can play each voice independently.

Complexity on / off
enhances pulse complexity.

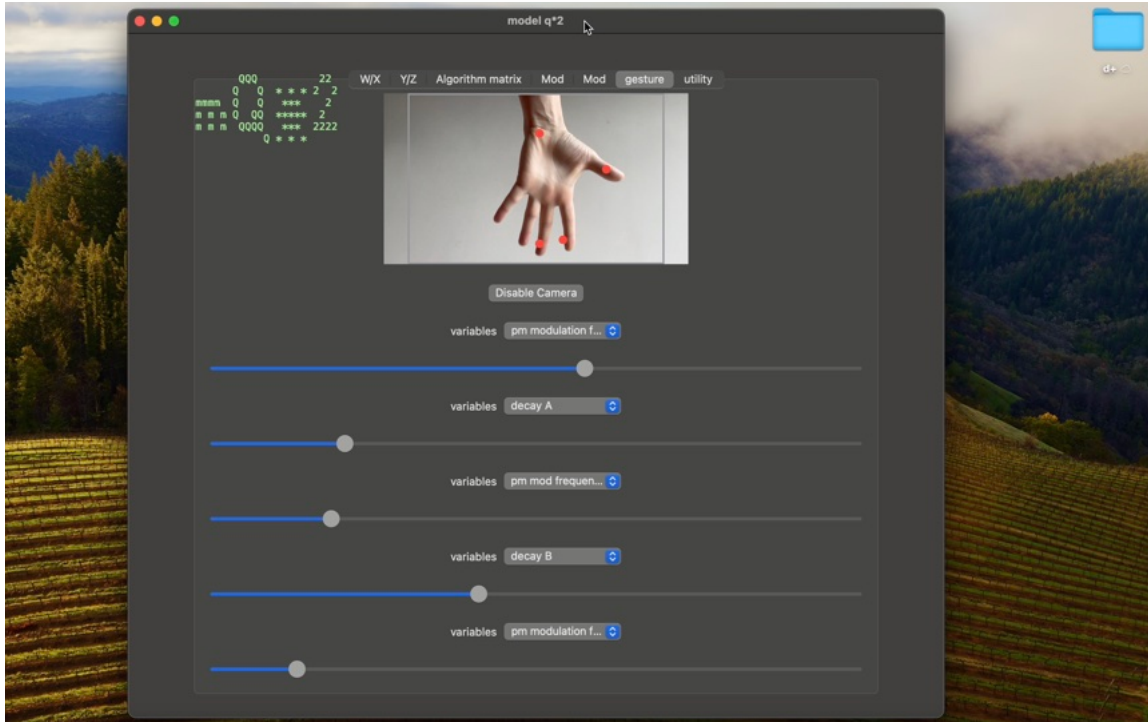
Pulse on / off

Randomise equation variables

Grouping voices together can link them making parameter changes uniform, including dice. The first voice to initiate the grouping will be the master voice and should be treated as such

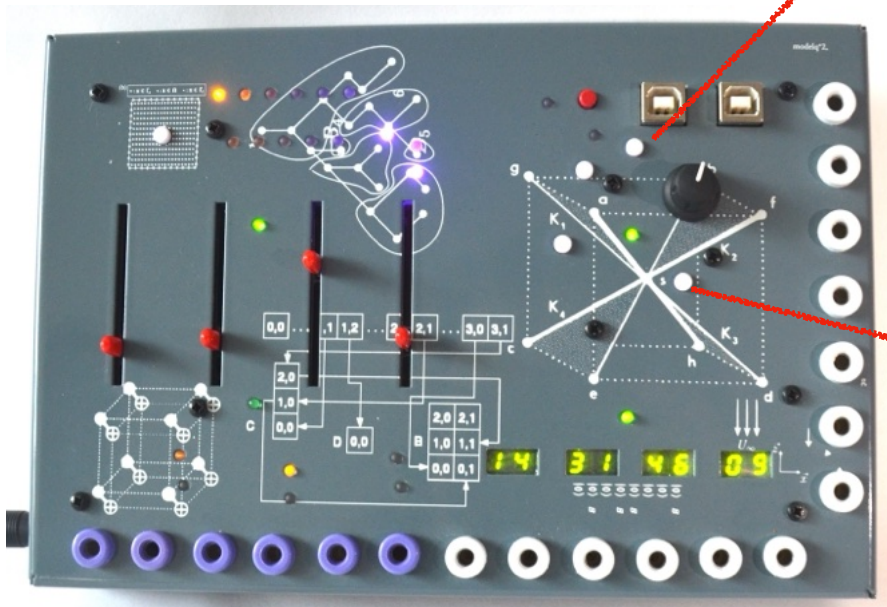
Gesture

The Modelq2 app is designed to facilitate gesture control. Allowing variables to be assigned to the base of the palm, the thumb, the index finger and the middle finger. The tracking inbuilt to follow these targets is precise and can be the basis for interesting integration with Modelq2 hardware.



Saving / preset index

Saving any patch in any state will overwrite the current preset index



The load preset button is only available when the encoder logic core is set to preset index. In any other mode this button is randomise all parameters