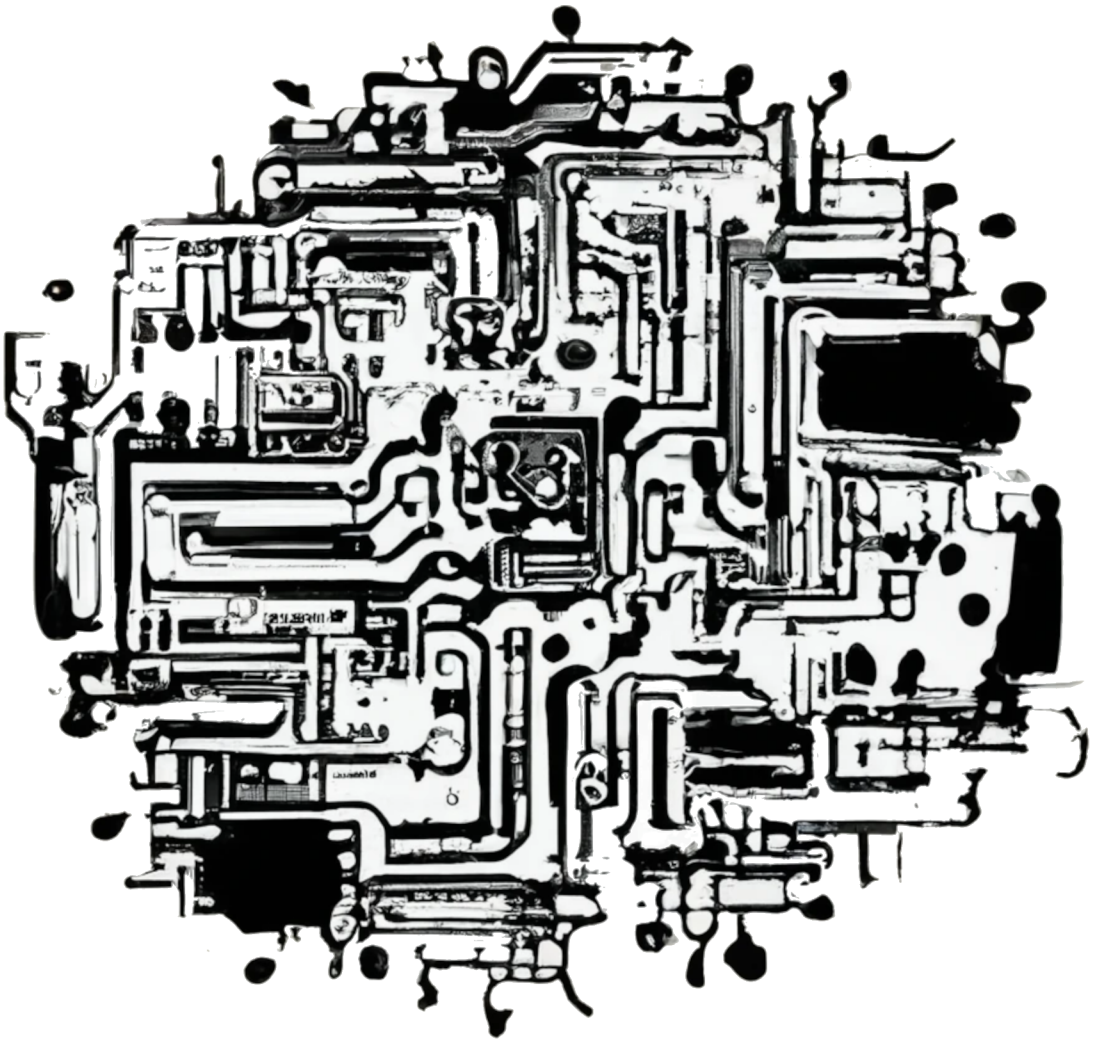
Phazerville 1.71

ORNAMENT & CRIME



ORNAMENT & CRIME

PHAZERVILLE 1.71

INDEX

GENERAL INFO

FULL SCREEN APPS (DEFAULT)

FULL SCREEN APPS (OTHER)

APPLETS

New to Ornament and Crime?

Ornament and Crime is a Polymorphic CV Generator — a swiss army knife of modular control voltage. It can perform as an *excellent* sequencer, envelope generator, quantizer, MIDI-to-CV and/or CV-to-MIDI interface, and much more via its many apps and applets.

o_C is a **collaborative open-source project** originated by Patrick Dowling, mxmxmx, and Tim Churches, and extended by many contributors (special thanks to Chysn for the original Hemisphere Suite). The ***Phazerville*** firmware optimizes the code base so that more apps can fit on the original hardware (which come in many flavours, for Eurorack and otherwise — *See Build Choices*), and paves the way for the next generation of hardware on the Teensy 4.1 platform.

Each of the full screen apps takes advantage of all inputs and outputs in their own way, which is usually configurable. ***Hemisphere*** splits the screen into two halves: each side available to load any one of a long list of applets. On o_C hardware with inputs and outputs arranged in 3 rows of 4 columns (i.e. most 8hp units), the I/O corresponding to an applet should be in line with that half of the display. If you're coming from any of the other Hemisphere forks, note that many of the applets have been upgraded for additional flexibility and functionality, and several are brand new.

*Note: Some apps, Hemisphere applets, and parameter editing contexts use special behaviour for the various encoders, buttons, and encoder buttons, which **should** be noted somewhere in these docs. If your encoders don't rotate the way you expect, you can flip the behaviour of one, the other, or both as part of the Setup / About calibration routine. See Hemisphere Gestures for all button combos within Hemisphere.*

Development Philosophy

Jump to bottom

bowlneudel edited this page 2 weeks ago · 1 revision

Allow me to explain my motives behind this project...

A Brief History

As a youth, I first learned the BASIC programming language on an old Apple IIc - a very procedural programming paradigm. After highschool, I studied Computer Science in undergraduate programs from 2005 to 2009, when the focus was on Object-Oriented paradigms using languages like Java. I became increasingly disenchanted with the state of the industry and job prospects, and financial troubles prevented me from completing my degree. For many years I gave up on software development entirely.

Renewed Interest

Sometime after 2020, I encountered the notion that **Object-Oriented Programming is GARBAGE** - the spark of enlightenment necessary to restore my hope for good software design. It validated my previous disenchantment with industry practices in a profound way. Maybe my intuitions were right all along? Inheritance and encapsulation always felt excessively complicated, and the whole mess violated my desires for simplicity and efficiency. I'd be happy to never touch a piece of Java code ever again!

I discovered a renewed interest in coding for the sake of building confidence in my programming intuition, and the O_C firmware became my hyperfixation.

Goals

Henceforth, my development efforts are driven by a few core desires:

- Simplify & Optimize by using *Functional* design paradigms instead of OOP
- Explore ways of using Git collaboratively
  - Exercise social coordination, encourage team efforts
  - Good communication - code readability, atomic commits, effective documentation
- Learn advanced C++ language features
- Gain a deeper understanding of Music Theory and Sound Synthesis fundamentals through software implementation
- Explore ways to improve UI/UX design
  - Never blame users for misunderstanding an interface! If something is not naturally intuitive, change it.
- "What's the worst that could happen?"
  - Don't be afraid to try stuff, especially if it might bother some people... Open Source is all about flexibility and choices.
  - A bug or broken feature might just bother someone else enough to join in and fix it - this one could be wishful thinking ;)

I hope to educate and inspire others along the way!

Cheers, ~Nick

Build Your Own Firmware!

I've set up an automated GitHub Action to handle Custom Firmware Build requests for Teensy 3.2

To initiate it, leave a comment starting with `/buildthis` followed by a list of the Apps and features you want. Use spaces or commas as delimiters. Anything that doesn't look like a valid keyword is ignored.

Hemisphere - and all of the applets - are included by default, with 4 preset slots.
If you don't want Hemisphere, use keyword **NO_HEMI** or **NOHEMI** to exclude it and all applets, leaving more space for other Apps.

Below is a list of the features, and approximately how much of the remaining flash storage space they consume. If you go over 100% it will fail! There is also very limited EEPROM space for app settings... More rough calculations here. Try to practice moderation, and keep things focused by picking just a handful of tools and you'll be fine. Good luck!

Feature keywords:

- **VOR** - for Plum Audio units
- **FLIP** - upside-down orientation
- **Grids2** - alternate patterns for DrumMap
- **Buchla** - hardware support for 1.2V/Oct scaling and unipolar 0-10V outputs
- **PEWPEWPEW** or say "**Pew**" 3 times - 8 Presets in Hemisphere + Phazerville easter eggs
  - Double presets mean double the EEPROM consumption (uses 256 of 1024 bytes)
- **MOAR_PRESETS** - 16 Presets in Hemisphere (uses 512 out of 1024 bytes)
  - (consumes about half of EEPROM - limits room for other apps!)

App keywords:

- **NO_HEM** or **NOHEMI** - Exclude Hemisphere and all applets, which frees up about 50% of total Flash storage
- Calibr8or (5.8%)
- Scenes (4.7%)
- MIDI aka Captain (12.8%)
- Pong (1.3%)
- Enigma (15.5%)
  - (will not save TM library if Acid Curds is included, see Acid Curds chord storage vs. Enigma TM storage #75)
- **Darkest** Timeline (5.1%)
- Neural Net (14%)
- CopierMaschine (14.8%)
- Harrington 1200 (10.5%)
- Automatonnetz (7.9%)
- Quantermain (17.9%)
- Meta-Q (14.2%)
- Quadraturia (13%)
- LORENZ aka Low-rents (3.3%)
- Piqued (22.2%)
- Sequins (32%)
- BBGEN aka **Dialectic** (6%)
- BYTE aka Viznutcracker (7.1%)
- **Acid** Curds (14.9%)
  - (overrides Enigma TM storage)
- Passencore (12.5%)
- References aka REFS (???)

After about a minute, check the status here: 

If successful, you can download the Artifact containing a .hex file. (Use a browser, not the GitHub mobile app!)

Improvements to the comment parser script are welcome - Python is not my specialty ;)

Ornaments & Crimes is free, open-source software (firmware) for the Ornament + Crime module. The licenses under which the software is released permit anyone to freely install and use the firmware on copies of the module, to modify it, and to provide copies to others. Third-party module manufacturers who re-use portions of the O+C software in their modules or devices should ensure that they meet the obligations imposed by the licenses under which the O+C source code is released - details are here (/licensing/).

Firmware upload methods

NB/FAQ: just updating the firmware (on a calibrated module) ***doesn't*** *require re-calibration — the calibration values are* ***not*** *overwritten when you install new versions of the firmware. Newly built modules, on the other hand,* ***must be calibrated*** *in order to function properly.*

There are **two ways** of getting the firmware onto your module:

- Method A: upload a pre-compiled HEX file. this is easy and quick. or
- Method B: install the Arduino IDE and the Teensyduino add-on and compile the code yourself. this option is required if you wanted to hack the code, etc.
- either way, you'll need: a **micro-usb cable** (make sure this isn't for charging only, but data transfer).
- NB: the following steps assume that you have cut the **usb trace**. see here (/build-it/#cut_trace).

method A: uploading the HEX file

step 1): install the Teensy Loader program

- the Teensy Loader is available at https://www.pjrc.com/teensy/loader.html (https://www.pjrc.com/teensy/loader.html). v1.33 of the Teensy Loader or later is required.

step 2): download the binary HEX file

- download the latest released version of the Ornaments and Crimes firmware image file (.hex) from https://github.com/mxmxmx/O_C/releases (https://github.com/mxmxmx/O_C/releases)

step 3): open the HEX file in the Teensy Loader

- open the HEX file in the Teensy Loader application
- make sure a USB cable is connected to the Teensy, and that the O+C module is powered up
- press the program push switch on the Teensy board (on the back of the O+C module)
- click the Program icon, or choose Program from the Operation menu in Teensy Loader
- (you should briefly see a progress bar as the firmware is uploaded)
- click the reboot icon or choose **Operation > Reboot**
- your O+C should now run the updated firmware (resp. come to life, if newly built)
- (**note**: if the loader app complains about the .hex file being "too large", try to load this hex (https://drive.google.com/uc?export=download&id=0ByUu4ePJDmVnd3U2RFdqYzJfbVE) first, so the loader can detect the MCU, then **re-try** with the OC firmware hex; or proceed to **method B**.)
- if this is a **newly built** module, proceed to Calibration (/calibration/) (just updating the firmware doesn't require re-calibration)

method B: compiling the firmware

step 1): get the IDE + teensyduino add-on

- if you don't have it already, you need to install the Arduino IDE as well as the Teensyduino v1.35 add-on. Note that you **MUST** use specific versions of these! At the time of writing, that's **Arduino IDE 1.8.1** (available from https://www.arduino.cc/en/Main/OldSoftwareReleases - just scroll down and it's in the table) and **Teensyduino v1.35** ([Mac (https://www.pjrc.com/teensy/td_135/TeensyduinoInstall.dmg)] [Windows (https://www.pjrc.com/teensy/td_135/TeensyduinoInstall.exe)] [Linux (https://www.pjrc.com/teensy

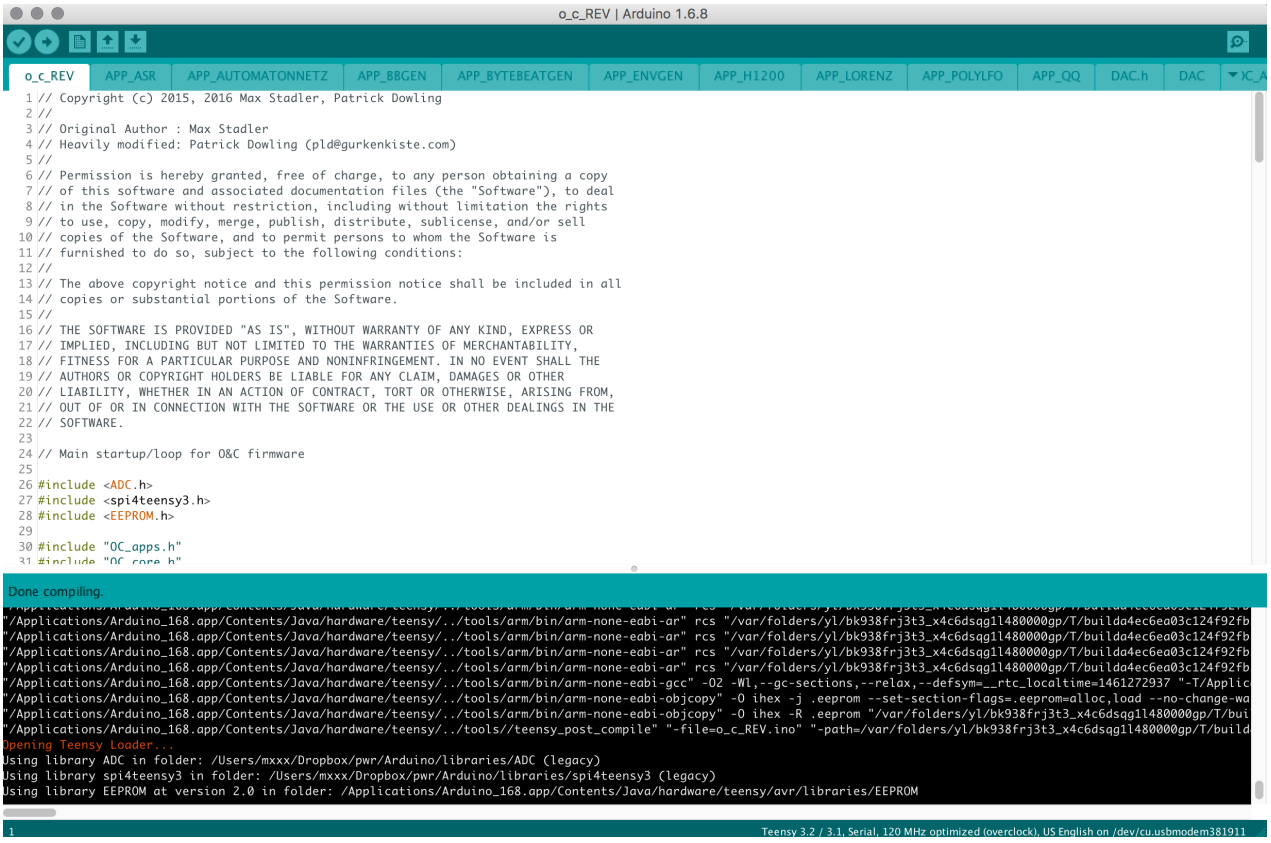
/td_135/)]), respectively; at this point, later versions of Teensyduino (which use gcc 5.4 instead of gcc 4.8) **do not work**, so please don't use them (they will cause screen corruptions). **Only use Teensyduino v1.35!**

step 2): clone or download the firmware source code repository to your computer

- clone from https://github.com/mxmxmx/O_C (https://github.com/mxmxmx/O_C) or download the .zip (https://github.com/mxmxmx/O_C/archive/master.zip) file.
- use the master branch (which is the default) - the master branch contains the latest "production" released code
- the develop and other branches contain bleeding-edge code which may or may not contain bugs

step 3): compile

- once the libraries and the source code for Ornaments and Crimes are in place, you should be able to **compile** the Ornaments & Crimes firmware. Open the file called **o_c_REV.ino**. now make sure you:
    - **1)** select **teensy 3.2/3.1** in `Tools > Board` . and
    - **2)** select **No USB** in `Tools > USB Type` (don't worry, you will still be able to use the USB port to upload the firmware!) . and
    - **3)** select **120MHz optimized (overclock)** in `Tools > CPU Speed` . and
    - **4)** select **Faster** (= o2) in `Tools > Optimize` (teensyduino 1.34 and 1.35)
- things should look somewhat like this:



- compile. and download to your board (since you've cut the usb trace, the module needs to be powered from your eurorack PSU): the display should come to life now.

step 4): calibrate

- if this is a **newly built** module, proceed to Calibration (/calibration/)
- just updating the firmware doesn't require re-calibration - the calibration values are **not** overwritten when you install new versions of the firmware.

ornament & crime is developed and maintained by mxmxmx, Patrick Dowling and Tim Churches.

Build Choices

Jump to bottom

bowlneudel edited this page 2 weeks ago · 8 revisions

By default, all builds include **Hemisphere, Calibr8or, Scenes, Enigma, Pong, Piqued, Quadraturia, Passencore, Viznutcracker, Dialectic Ping Pong**

Or you can roll your own! Generate a custom firmware with your choice of Apps! (Read and leave a comment on the Discussion, and a robot will build it for you.)

Your choices of default builds

Flipped Operation

Files with "_flipped" are for running the module upside-down.

VOR

Files with "**+VOR**" are only for **Plum Audio / 4ROBOTS** hardware variants equipped with a *Variable Output Range* circuit - O&C Plus, 1uO_c, & OCP X.

Standard 8HP uO_C, After Later uO_C 1U, or full-size 14HP O_C modules should be installed *without* VOR - your outputs will behave strangely if you do!

Other options

Buchla models and Teensy 4.0 have their own default builds on the Releases page

Saving State

Jump to bottom

bowlneudel edited this page 2 weeks ago · 14 revisions

Your App / applet state will not be remembered between power cycles unless you:

- (A) Manually save to EEPROM *(Long-press RIGHT encoder to escape to main menu, long-press RIGHT again to save)*
- (B) Store the current state of Hemisphere to a preset
- (C) Turn on Auto Save

To Save/Load presets or toggle Auto Saving in Hemisphere, long-press the DOWN button to open the config menu, and (if necessary) rotate the LEFT encoder to paginate to the floating preset menu.

When storing a Preset, it immediately triggers an EEPROM Save (with a potential 2ms interruption, fyi) so there is no need to also long-press-save on the main menu.

Troubleshooting

Jump to bottom

bowlneudel edited this page 2 weeks ago · 10 revisions

If applets like Stairs, Carpeggio or Shredder appear unresponsive, you might need to calibrate your ADC to eliminate an offset on the inputs. This can be done without a fancy multimeter - go to the **Setup / About** app and just skip all the other Calibration steps. Simply unplug all CV/Trigger inputs and outputs, enter the Calibration routine and ensure that all ADC values are around 0v (-1v ≀ 1v).

If your encoders don't rotate the way you expect, you can flip the behaviour of one, the other, or both as part of the Setup / About calibration routine.

After selecting "Calibrate" in Select / About (short press of LEFT encoder), press either the UP and DOWN buttons to choose your encoder reversal: L, R, both (LR), or neither (normal) — press the RIGHT encoder to accept.

If you want to use non-default calibration, you will need to scroll through the entire calibration routine to save the encoder reversal setting (rotate LEFT encoder to the last page, press RIGHT encoder to save).

Hemisphere Gestures

Jump to bottom

bowlneudel edited this page 2 weeks ago · 9 revisions

Operating Hemispheres makes use of button combinations to perform various actions. See below for a complete list.

Please note that certain applet (or parameter editing) contexts may override the default behaviour of the UP and DOWN buttons &/or the LEFT and RIGHT encoders.

Action	Gesture	Notes
Change left Hemisphere applet	Short press UP button to highlight hemisphere, scroll applets with LEFT encoder	
Change right Hemisphere applet	Short press DOWN button to highlight hemisphere, scroll applets with RIGHT encoder	
Open applet help screen	Double press UP button for left hemisphere, double press DOWN button for right hemisphere	
Open Clock / Trigger Setup	Press both UP + DOWN buttons together	Adjust internal BPM, swing, external sync, and per-trigger clock mult/div; remap trigger inputs (also available within the Config); and manually perform triggers.
Cycle Internal Clock state	Long-press LEFT Encoder	Stop -> Arm *(i.e. play on next input trigger)* -> Start
Open Hemisphere Config menu	Long-press DOWN button (scroll pages with LEFT encoder, scroll options with RIGHT encoder)	Load/save presets; adjust trigger length; select screensaver and cursor mode; toggle auto-MIDI out; trigger and CV input mapping; quantizer settings; applet filtering
AuxButton	***Highlight parameter for editing***, press select button (UP or DOWN, depending on Hemisphere)	This gesture is only enabled in certain applets for secondary functions. Use it to mute/unmute steps in DivSeq, SequenceX, Seq32, etc.; re-randomize sequences in Shredder; and transpose octaves in the pop-up quantizer menu
Invoke screensaver	Long-press UP button	Global
Return to main menu	Long-press RIGHT encoder	Global, execution continues in the background.
Save Settings to EEPROM	Long-press RIGHT encoder *again* while on main menu	Why not use presets instead?
Easter Egg	Press the LEFT encoder while on the main menu to find the secret Debug Screen!	

VOR Gestures

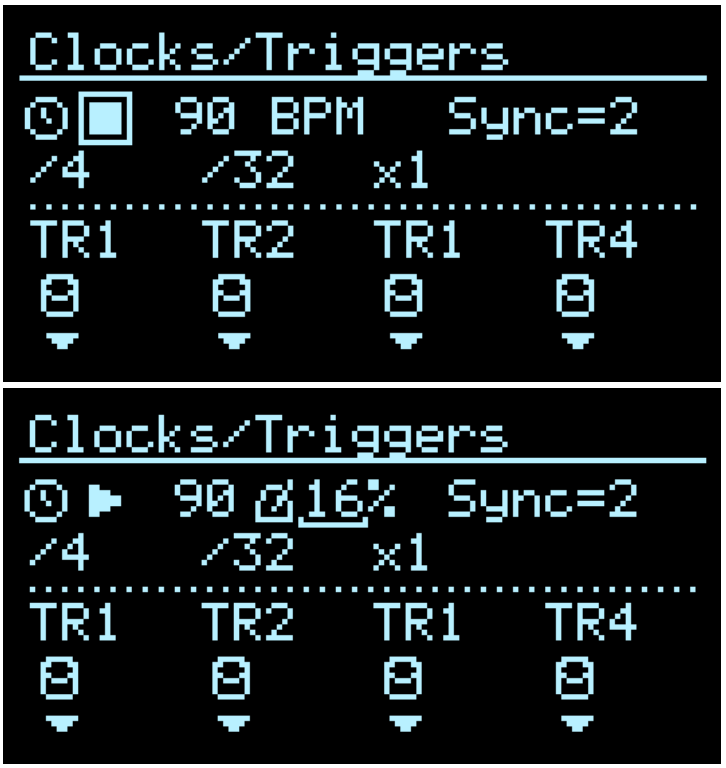
Variable Output Range compatible hardware only (i.e. Plum Audio 4robots, OCP, and OCP x)

- **Start/ stop clock:** press VOR button
- **Cycle VBias offset (-5V, -3V, 0V):** Press LEFT and RIGHT encoders together

Clock Setup

Jump to bottom

Nicholas Michalek edited this page last week · 10 revisions



The **Clocks/Triggers** screen, fka Clock Setup, is accessed by dual-pressing both *UP+DOWN* buttons. The top half of the screen provides internal clock controls for standalone operation. The multipliers determine how internally generated clock triggers are fed to the applets. The clock will sync to external pulses on TR1 while it's running. The bottom half of the screen has settings for remapping the physical trigger inputs (to avoid the need for mults) as well as manual triggers for convenience.

Parameters

- **Play/Stop/Pause** (toggle) - indicated by a Clock icon followed by a Pause, Play or Stop icon
  - Can also be cycled with *Left Encoder Long-Press* on any screen within Hemisphere
  - When Paused, the clock is armed - a single pulse on TR1 will immediately start the clock
- **Tempo** - pretty self-explanatory :P
  - Manual Tap-Tempo - press the encoder button 4 times on this parameter to detect and set the BPM accordingly
- **Swing/Shuffle** - delays every other clock pulse within a beat
- **Sync** - aka PPQN; expected resolution of external clock pulses on TR1
  - set to 0 to ignore incoming pulses
- **Multipliers/Dividers** - virtual triggers generated for each channel in relation to Tempo
  - set to 0 to disable internal clock on that channel; physical triggers will pass thru instead
- **Trigger Input Remapping** - reroute physical trigger inputs to the applets
  - only applies when the internal clock is not running, or if multiplier is set to "x0" for that channel
  - to mimic legacy "Clock Forwarding" function, simply set channel 3 to "TR1" instead of "TR3" (saved settings will automatically load like this)
- **Manual Triggers** - The four Button icons at the bottom enable you to manually fire each trigger
  - handy for advancing or resetting sequencers, etc.

Notes

- Outgoing MIDI messages for Start, Stop, & Clock (at 24ppqn) are sent via USB automatically. An option to disable this is planned.
- Incoming MIDI Start/Stop/Clock messages intuitively start the internal clock and feed it sync pulses at 2ppqn. The clock speed can then be reinterpreted differently with the `Sync=` parameter.
- The internal Clock is available in **Hemisphere** as well as the Calibr8or app. However, Clock settings are only stored in the active Preset in Hemisphere.

Hemisphere Config

Jump to bottom

bowlneudel edited this page 2 weeks ago · 30 revisions

Long-press DOWN button to enter the Hemisphere Configuration menu. Short-press UP or DOWN button to exit.

As of v1.7.1, the config menu is split into several pages. Rotate LEFT encoder to jump to different pages. Use RIGHT encoder to move the cursor and push to edit.

- **Floating Menu:** Presets
- **Page 1:** General Settings
- **Page 2** Quantizer Set-Up
- **Page 3:** Input Mapping
- **Page 4:** Applet Filtering

See also: Clock-Setup

Hemisphere Presets

Jump to bottom

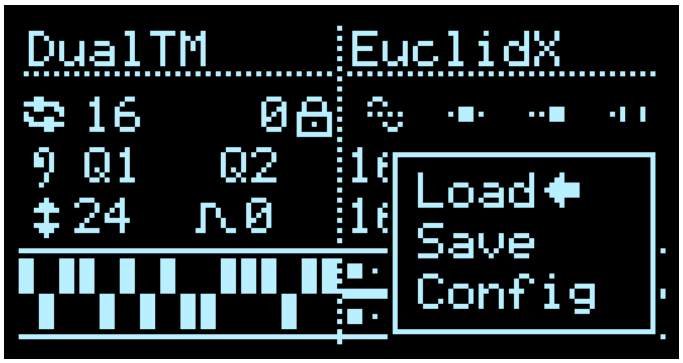
bowlneudel edited this page 2 weeks ago · 12 revisions

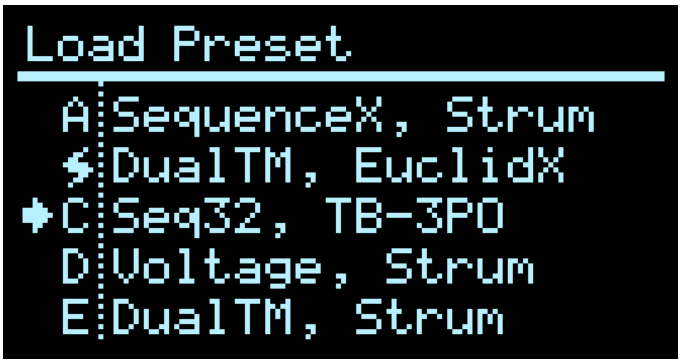
Floating menu: Presets

The first three options in the config menu are **Load**, **Save**, and **(auto)**

Rotate either encoder to select, push to enter. Press UP or DOWN buttons to cancel. Both applets (plus Clock Setup) with their state are recalled, along with the rest of the Config options on this page.

Presets are saved with the name of the applets in each hemisphere. Note that default builds have 4 preset slots, but you can enable 8 or 16 with a custom build





Load Presets via MIDI PC

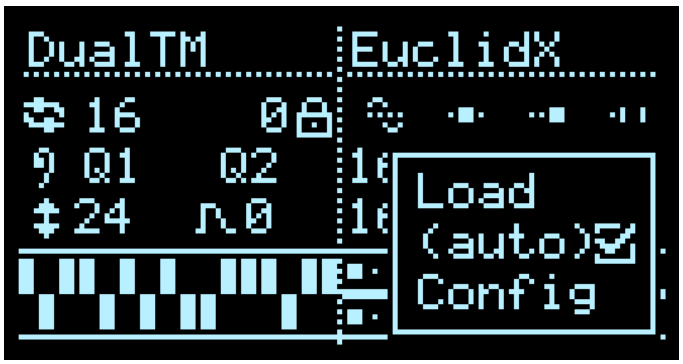
Sending MIDI Program Change messages to Channel 1 will load the corresponding preset

- Value 0: Preset A
- Value 1: Preset B
- Value 2: Preset C
- Value 3: Preset D
- ...Etcetera (up to however many presets are enabled in your build: 4, 8, or 16)

Auto-save

New in v1.6.999

If enabled, settings are automatically stored in the last loaded Preset when the screensaver is invoked, or when loading the main App menu via Right Encoder Long-press. (You can set the screensaver timeout as low as 1 minute in Calibration)



As of v1.6.999, when storing a Preset, settings are immediately written to EEPROM - no need to manually do an EEPROM Save, unless you need to save changes to global patterns, etc. Combined with Auto-save, this can result in frequent EEPROM writes... (only 100,000 write cycles are guaranteed, so use at your own risk!)

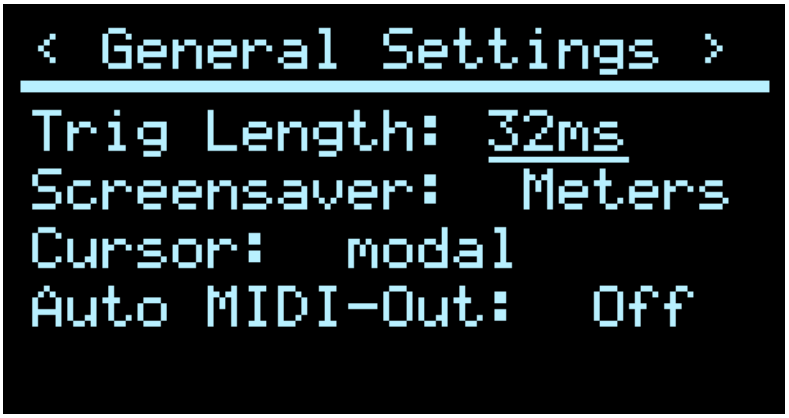
Hemisphere General Settings

Jump to bottom

bowlneudel edited this page 4 days ago · 10 revisions

Page 1: General Settings

The first full-screen page in the config menu (after the floating presets menu) is for General Settings



Trigger Length

This sets the pulse width (in milliseconds, approximate) for applets that generate simple triggers, such as **EuclidX** or **TrigSeq**. The old default was close to 3ms, but some modules may require longer pulses.

Screensaver

Options are:

- [blank]
- Meters
- Scope
- Zips (or "Stars" in Teensy 4.x builds)

Cursor

As of v1.7, the Legacy cursor mode (press encoder to step to next parameter, rotate to edit) has been removed, and been replaced by a **modal** cursor: **rotate encoder to move cursor, push to toggle editing, rotate to edit parameter, push again to untoggle editing**

Note: Some applets (eg. Button2) override the default cursor behaviour, usually to enable different functionality when scrolling CW versus CCW.

Setting the cursor configuration to **"modal + wrap"** will allow infinite looping scrolling: the cursor will wrap from the last parameter to the first, and vice versa.

Setting the cursor to simply **"modal"** will prevent looped scrolling, terminating at the beginning and end of the parameter list.

Auto MIDI Output

(Experimental) When enabled, MIDI messages are sent automatically based on applet outputs. By default, the Left Hemisphere outputs on Channel 1, and the Right Hemisphere on Channel 2 (configurable with the MIDI-Out applet). Outputs A/C are interpreted as Note values, and B/D as gates for NoteOn/NoteOff.

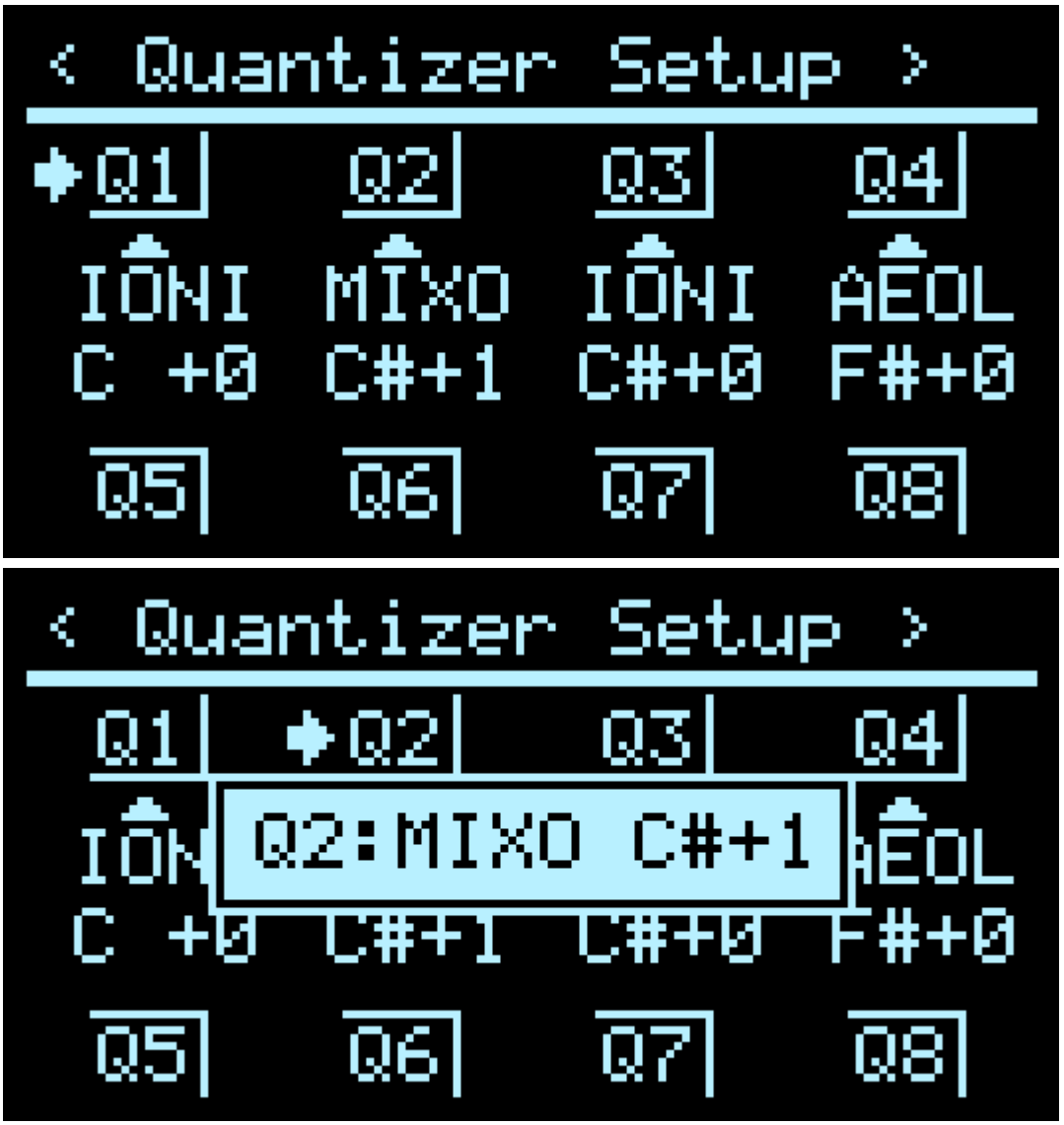
Hemisphere Quantizer Setup

Jump to bottom

bowlneudel edited this page 2 weeks ago · 1 revision

Page 2: Quantizer Setup

The second full-screen page in the config menu is for configuring 8 separate quantizer engines, which may be quickly selected & edited in any applet that uses quantization.



There are 8 global quantizer engines, each with a **Scale**, **Root Note**, and **Octave** setting. An encoder push will open the popup editor. Use LEFT encoder to change Scale, RIGHT encoder for Root Note, and UP/DOWN select buttons to jump octaves. Push either encoder again to exit. These gestures are the same when editing quantization settings in-applet.

Currently, settings for the first 4 quantizers can be reliably saved with Calibr8or. Some applets may also save/restore quantizer settings, but this may change.

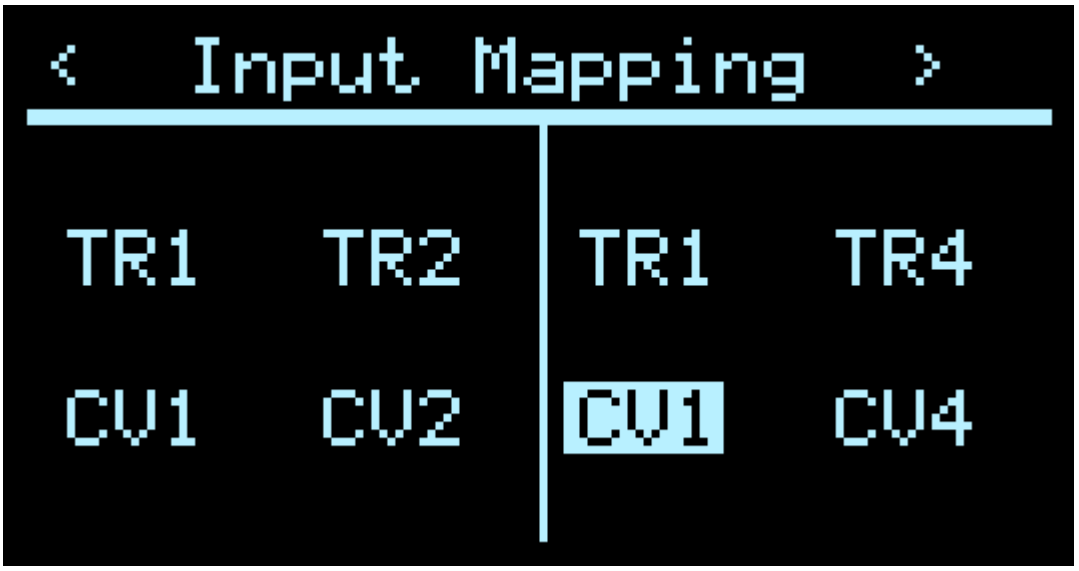
Hemisphere Input Mapping

Jump to bottom

bowlneudel edited this page 2 weeks ago · 1 revision

Page 3: Input Mapping

The third full-screen page in the config menu is for remapping the software inputs to any of the physical jacks or any of the outputs A-D



Every applet has 2 trigger inputs and 2 CV inputs. Here, you can remap which physical inputs are routed to the applet's logical inputs. The trigger map settings also appear in Clock Setup.

Hemisphere Applet Filtering

Jump to bottom

bowlneudel edited this page 2 weeks ago · 2 revisions

Page 4: Applet Filtering

The fourth and final full-screen page in the config menu is for Applet Filtering.

Are you tired of scrolling past applets you never use? Now you can hide them with this filter!

This feature is an experimental Proof-of-concept. Settings are not saved and only first 64 applets are actually listed.

This is a full list of all the applets. RIGHT encoder push will toggle visibility of the selected applet. LEFT encoder push inverts the entire list. When an applet is hidden (unchecked), it is simply skipped when switching applets on the main view.

Full Screen Apps

Full screen apps in Phazerville are mostly from the original Ornament and Crime firmware, with a few notable additions (Calibr8or, Scenes, and Passencore). Each of the full screen apps takes advantage of all inputs and outputs in their own way, which is usually configurable.

Not all the apps can fit at once on Teensy 3.2 hardware, but you can use the default set or choose your own selection with a custom build.

- Hemisphere - 2 Applets at a time
- Calibr8or - Quad performance quantizer with pitch tracking calibration
- Scenes - Macro CV switch / crossfader
- Captain MIDI - Configurable CV-to-MIDI and MIDI-to-CV interface
- Pong - It's Pong!
- Enigma – Sequencer of shift registers (Turing Machines)
- The Darkest Timeline - Parallel universe sequencer
- Neural Net - 6 Neuron logic processor
- Scale Editor - Edit and save microtonal scales
- Waveform Editor - Edit and save vector waveforms (for LFOs, envelopes, one-shots, and phase scrubbing)
- CopierMaschine - Quantizing Analogue Shift Register
- Harrington 1200 - Neo-Riemannian transformations for triad chord progressions
- Automatonnetz - Neo-Riemannian transformations on a 5x5 matrix sequence!
- Quantermain - Quad quantizer
- Meta-Q - Dual sequenced quantizer
- Quadraturia - Quadrature wavetable LFO
- Low-rents - Lorenz attractor
- Piqued - Quad envelope generator
- Sequins - Basic dual-channel sequencer
- Dialectic Ping Pong - Quad bouncing ball envelopes
- Viznutcracker, sweet! - Quad Bytebeat generator
- Acid Curds - Quad 8-step chord progression sequencer
- References - Tuning utility
- Passencore - Generate a chord progression from LFOs
- Backup / Restore - Transfer app and calibration data as SysEx
- Setup / About - Check your version, change encoder directions, adjust display/DAC/ADC, screen off time

Hemisphere Applets

Hemisphere splits the screen into two halves: each side available to load any one of a long list of applets. On o_C hardware with inputs and outputs arranged in 3 rows of 4 columns (i.e. most 8hp units), the I/O corresponding to an applet should be in line with that half of the display (i.e. paired into 1+2/A+B and 3+4/C+D).

If you're coming from any of the other Hemisphere forks, note that many of the applets have been upgraded for additional flexibility and functionality, and several are brand new.

- ADSR - Dual attack / decay / sustain / release envelope
- AD EG - Attack / decay envelope
- ASR - Analog Shift Register
- AttenOff - Attenu-vert, Offset, and Mix inputs (now with +/-200% range, mix control)
- Binary Counter - 1 bit per input, output as voltage
- BootsNCat - Noisy percussion
- Brancher - Bernoulli gate
- BugCrack - Sick drums, don't bug out Mutable Instruments: PEAKS, drums with CV mod feature
- Burst - Rapid trigger / ratchet generator
- Button2 - 2 simple triggers or latching gates. Press the button!
- Calculate - Dual Min, Max, Sum, Diff, Mean, Random, S&H
- Calibr8 - 2-channel, mini Calibr8or for v/Oct correction
- Carpeggio - X-Y table of pitches from a scale/chord
- Chordinate - Quantizer with scale mask, outputs root + scale degree (from qiemem)
- ClockDivider - Dual complex clock pulse multiplier / divider.
- ClockSkip - Randomly skip pulses
- Compare - Basic comparator
- Cumulus - Bit accumulator, inspired by Schlappi Nibbler
- CVRec - Record / smooth / playback CV up to 384 steps on 2 tracks
- DivSeq - Two sequences of clock dividers
- DrumMap - Clone of Mutable Instruments Grids
- DualQuant - Basic 2-channel quantizer with sample and hold
- DualTM - Highly configurable pair of Turing Machine shift registers (replacement for ShiftReg/TM)
- Ebb & LFO - clone of Mutable Instruments Tides; oscillator / LFO with CV-controllable waveshape, slope, V/Oct, folding
- EnigmaJr - compact player of curated shift registers
- EnvFollow - follows or ducks based on incoming audio
  - added Speed control
- EuclidX - Euclidean pattern generator (replacement for AnnularFusion)
- GameOfLife - experimental cellular automaton modulation source
- GateDelay - simple gate delay
- GatedVCA - simple VCA
- Dr. LoFi - super crunchy PCM delay line with bitcrushing and rate reduction
  - based on LoFi Tape and Dr. Crusher

- Logic - AND / OR / XOR / NAND / NOR / XNOR
- LowerRenz - orbiting particles, chaotic modulation source
- Metronome - internal clock tempo control + multiplier output
- MIDI In - from USB to CV
- MIDI Out - from CV to USB
- MixerBal - basic CV mixer
- MultiScale - like ScaleDuet, but with 4 scale masks
- Palimpsest - accent sequencer
- Pigeons - dual Fibonacci-style melody generator
- PolyDiv - four concurrent clock dividers with assignable outputs
- ProbDiv - stochastic trigger generator
- ProbMeloD - stochastic melody generator
- ResetClk - rapidly advance a sequencer to the desired step (from pkyme)
- RndWalk - clocked random walk CV generator (from adegani)
- RunglBook - chaotic shift-register modulation
- ScaleDuet - 2 quantizers with independent scale masks
- Schmitt - Dual comparator with low and high threshold
- Scope - tiny CV scope / voltmeter / BPM counter
  - expanded with X-Y view
- Seq32 - compact 32-step sequencer using Sequins pattern storage
- SequenceX - up to 8 steps of CV, quantized to semitones
- ShiftGate - dual shift register-based gate/trigger sequencer
- Shredder - clone of Mimetic Digitalis
- Shuffle - it don't mean a thing if it ain't got that swing
  - triplets added on 2nd output
- Slew - Dual channel slew: one linear, the other exponential
- Squanch - advanced quantizer with transpose
- Stairs - stepped CV
- Strum - the ultimate arpeggiator (pairs well with Rings)
- Switch - CV switch & toggle
- SwitchSeq - multiple Seq32 patterns running in parallel
- TB-3PO - a brilliant 303-style sequencer
- TL Neuron - clever logic gate
- Trending - rising / falling / moving / steady / state change / value change
- TrigSeq - two 8-step trigger sequences
- TrigSeq16 - one 16-step trigger sequence
- Tuner - oscillator frequency detector
- VectorEG - Dual envelopes from a library of bipolar and unipolar shapes (customizable with the Waveform Editor)
- VectorLFO - Dual LFOs from a library of bipolar and unipolar shapes
- VectorMod - Dual One-shots from a library of bipolar and unipolar shapes
- VectorMorph - Dual (or linked) phase scrubbing along a library of bipolar and unipolar shapes
- Voltage - static output CV

Apps and Applets by Function

Function	Hemisphere Applets	Full Screen Apps
Accent Sequencer	Palimpsest	
Analog Logic	Calculate	
Clock Modulator	ClockDivider, ClockSkip, DivSeq, Metronome, PolyDiv, ProbDiv, ResetClk, Shuffle	
CV Recorder	ASR, CVRec	
Digital Logic	Binary Counter, Compare, Cumulus, Logic, Schmitt, TL Neuron, Trending	Neural Net
Delay	GateDelay	
Drums / Synth Voice	BootsNCat, BugCrack	Viznutcracker, sweet!
Effect	Dr. LoFi	
Envelope Follower	EnvFollow, Slew	
Envelope Generator	ADSR, AD EG, VectorEG	Piqued, Dialectic Ping Pong
LFO	Ebb & LFO, LowerRenz, VectorLFO	Quadraturia
MIDI	MIDI In, MIDI Out *(See also: Auto MIDI Output)*	Captain MIDI
Mixer	MixerBal	
Modulation Source	GameOfLife, Stairs, VectorMod, VectorMorph	Low-rents, Pong
Performance Utility	Button2	Scenes
Pitch Sequencer	Carpeggio, DualTM, EnigmaJr, Pigeons, ProbMeloD, Seq32, SequenceX, Shredder, Strum, SwitchSeq, TB-3PO	Enigma, The Darkest Timeline, Automatonnetz, Sequins, Acid Curds, Passencore
Quantizer	Calibr8, Chordinate, DualQuant, MultiScale, ScaleDuet, Squanch	Calibr8or, Harrington 1200, Quantermain, Meta-Q
Random / Chaos	Brancher, LowerRenz, ProbDiv, ProbMeloD, RndWalk, Shredder	Low-rents
Shift Register	ASR, DualTM, EnigmaJr, RunglBook, ShiftGate	Enigma, CopierMaschine
Switch	Switch, SwitchSeq	Scenes
Trigger / Gate Sequencer	DivSeq, DrumMap, EuclidX, PolyDiv, ProbDiv, Seq32, ShiftGate, TrigSeq, TrigSeq16	
VCA	GatedVCA	
Voltage Utility	AttenOff, Calculate, Calibr8, Scope, Slew, Stairs, Switch, Tuner, Trending, Voltage	Calibr8or, References

Calibr8or

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 7 revisions

Read the full write-up from Chris Meyer.

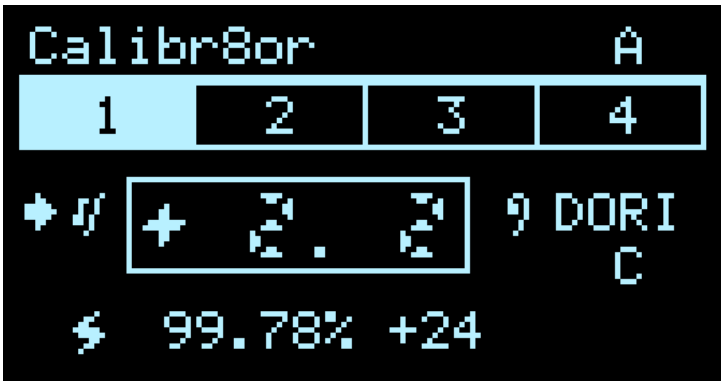
Quad Performance Quantizer + Pitch Fine-Tuning App

Commissioned by Chris Meyer (Alias Zone / Learning Modular) - this is a **4-channel Pitch CV fine-tuning tool** with performance-oriented transpose controls. Each channel can be clocked for latching of the *Transpose* value or full S&H of the input. There are 4 preset banks to store and quickly switch configurations.

The Fine Tune controls at the bottom of the main view apply voltage *Scaling* and *Bias Offset* to the output stage, per channel. Scaling is in increments of ±0.01% and Bias is effectively ±1/128th of a semitone.

Accurate performance will depend on the calibration of your o_C module itself.

UI Controls



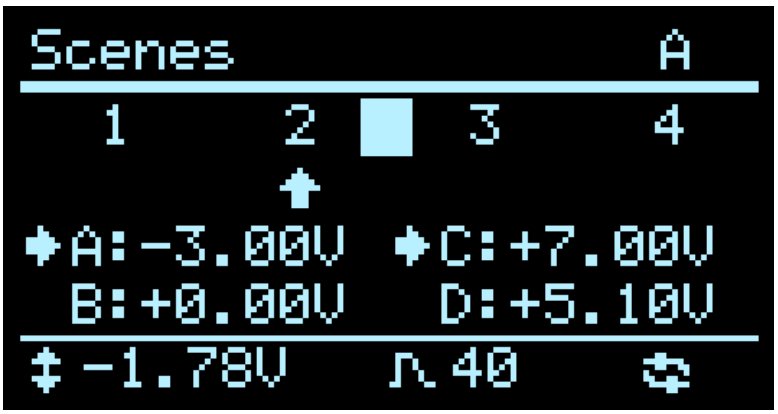
- UP / DOWN buttons - switch channels
- Quick-press Left Encoder - switch between Transpose and Fine-Tune controls
- Quick-press Right Encoder - toggle Scale & Root Note selection
- Long-press Left Encoder with cursor on Transpose - cycle Clocked Transpose / S&H modes per channel
- (new in v1.6.777) Long-press Left Encoder with cursor on Fine-Tune controls - Auto-tuner closed loop DAC calibration routine, ported from References
- Long-press DOWN button - preset switching
  - turn or press Left Encoder to toggle Save/Load
  - use Right Encoder to select
  - UP or DOWN to cancel
- Dual-press UP+DOWN for Clock Setup (shared with Hemispheres)

As of v1.6.6, storing a preset will automatically save App Data to EEPROM in the background.

Scenes

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 9 revisions



Scenes is a macro CV Switcher/Crossfader inspired by **Traffic** by Jasmine & Olive Trees. This app offers 4 "scenes" with 4 CV output values each; there are also 4 preset banks, using the same mechanism as Calibr8or and **Hemisphere** (long-press DOWN button to access).

As of v1.7, storing to a Preset automatically saves to EEPROM.

Controls

	Left Encoder	Right Encoder
TURN		
PRESS	Toggle between editing A or B output for selected scene	Toggle between editing C or D output for selected scene
LONG	Toggle Trig Sum mode on output D	

	Up Button	Down Button
PRESS	Move edit cursor between the 4 scenes.	Move edit cursor between the 4 scenes.
LONG PRESS	Invoke screensaver view	Go to Preset Menu

Inputs

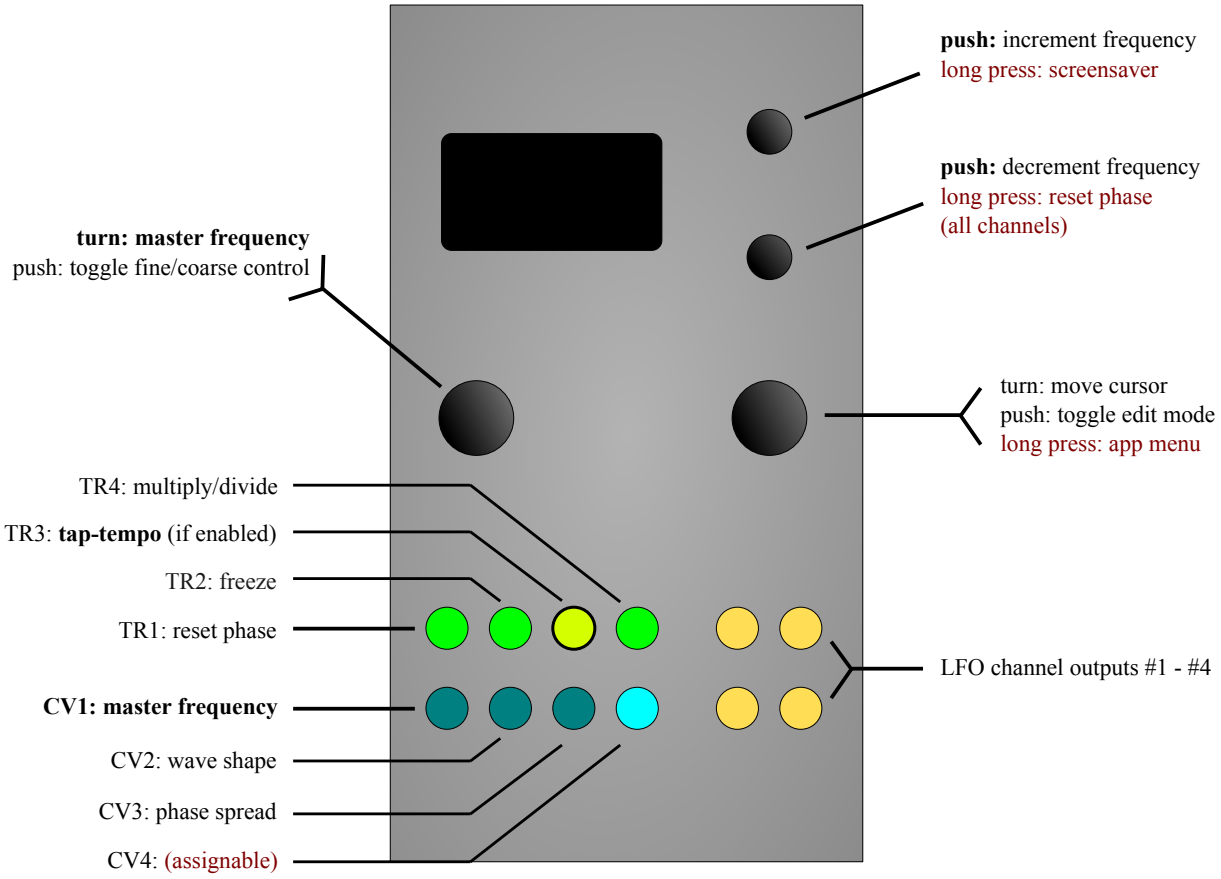
	1	2	3	4
TR	Jumps to corresponding Scene. Prioritized so TR1/Scene1 will take precedence if multiple inputs are gated simultaneously.	Jumps to corresponding Scene. (v1.6.5)	Jumps to corresponding Scene. (v1.6.6)	Jumps to corresponding Scene.
CV	bipolar smooth crossfade between scenes, centered on the last triggered scene. 1 Volt == 1 Scene. 4 Volts loops back around.	bias offset for all values	modulates slew/smoothing factor for all outputs	when gated (>2V), enables a random 16-step sequence on Scene 4 using a shuffled combination of all 16 CV values. TR4 will advance the sequence. A new sequence is generated every time the gate goes high
OUT				

Quadraturia

Quadraturia is a port of of the quadrature wavetable LFO that is available as the “Easter egg” in the Mutable Instruments Frames (https://github.com/pichenettes/eurorack/tree/master/frames) module. However, *Quadraturia* adds voltage control over three of the four LFO parameters (as well as CV control of frequency/rate of the LFO, as in Frames). Because the background ISR of o_C runs at 16.7KHz instead of 32KHz as in Frames, the behaviour of *Quadraturia* may not be identical to the Frames Easter egg, but it should be very close, and is nonetheless a very useful and flexible source of modulation voltages. For more details of how the Frames Easter egg LFO behaves, see the relevant section of the Frames Manual (http://mutable-instruments.net/modules/frames/manual).

Quadraturia also incorporates the “predictive” tap-tempo capability from the Tap LFO mode in the Mutable Peaks module. Details are given in the **Tap tempo** section below.

Internally, there are four LFOs (LFO1 to LFO4), with LFO2 to 4 running at some ratio of the (master) frequency of LFO1 (by default, that ratio is one, so all the LFOs run at the same frequency). The wave shape, phase of frequency modulation of LFO2 to 4 can be changed, relative to those parameters for LFO1, using the Shape spread, phase/frequency spread and coupling controls (see table below).



I/O	**Function**
TR1	Reset phase (all LFOs)
TR2	Freeze (all LFOs stop in their tracks and hold current value while TR2 is high)
TR3	Tap tempo lock or trigger/gate input when used in **Tap tempo** mode.
TR4	Divide/Multiply frequency (while TR4 is high — see `TR4: MULT`)
CV1	Master frequency
CV2	Wave shape
CV3	Phase/frequency spread

I/O	Function
CV4	(mappable) : coupling, shape spread, range, offset, amplitude modulation (a > b, b > c, or c > d)
A, B, C, D	LFO channel outputs

Controls

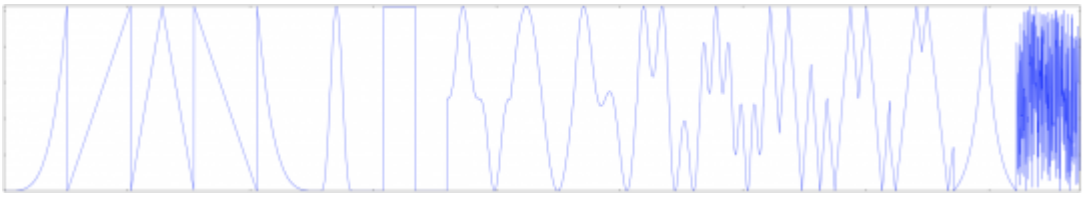
Control	Function
Left encoder (turn)	Increase or decrease frequency of master LFO
Left encoder (press)	Toggle frequency control between Coarse and Fine control.
Right encoder (turn)	Navigation mode: move up and down through the menu items. Edit mode: increase or decrease the value being edited.
Right encoder (press)	Toggle between menu navigation (selection) mode and value editing mode
Right encoder (long press)	App selection menu
Up button	Increment frequency by 32
Up button long press	screen saver short cut
Down button	Decrement frequency by 32
Down button long press	reset waveform phase to zero on all channels

Settings

Setting	Meaning
`Coarse` or `Fine` (frequency)	When in normal mode (**not tap tempo mode**), frequency is adjused with the left encoder. The frequency is displayed above the line in the menu. Clicking on the left encoder toggles between (C)oarse and (F)ine frequency adjustments. There are 256 steps for coarse adjustment, and each unit of the coarse adjustment is divided into 256 steps for fine adjustment. If **tap tempo** mode is enabled, the frequency adjustment using the left encoder is disabled.
`Tap tempo`	enables or disables **tap tempo** mode. The tap tempo trigger/clock input is TR3.
`Shape`	controls the shape of the primary waveform (LFO1), as shown in the preview waveform. See below for a diagram of all the waveforms in the wavetable.
`Shape spread`	sets the difference in wavetable position between each channel (and thus the differences in the wave shape for chennels B, C and D.
`Phase/frq spread`	sets the phase or frequency offset between each channel. Values greater than zero cause the phase to be offset incrementally in channels B, C and D with respect to (wrt) channel A. When `Phase/frq spread` is set to its maximum value of 127, the phase shift is 90 degrees on channel B, 180 degrees on channel C and 270 degrees on channel D. Values for `Phase/frq spread` of less than zero cause a progressive frequency shift (detune) across channels B, C and D wrt channel A, rather than a phase shift. **Note:** when frequency division or multiplication is set on channels B, C or D (see `B freq ratio` etc below), only 90 degrees of phase shift is available on each of channels B, C and D.
`Coupling`	sets the degree of phase-modulation “bleed” between each successive channel.
`Output range`	sets the overall output range for all channels. The range of this settings goes from 0 (no output) to 230, which equates to a nominal output range from about -3.5V to +6V. By default, with the `Offset` setting (see below) at zero, the output range is asymmetric. By reducing the `Output range` and then setting a positive `Offset` , the output can be shifted so that it is unipolar or otherwise offset at the level desired.

Setting	Meaning
Offset	shifts the output on all channels up or down by up to several volts. The range is -128 to 127, with a default of zero. Note that this is an internal offset and the absolute output voltage range is constrained by the hardware to about -3.5V to +6V. Nonetheless, by using it in conjunction with reduced settings of the Output range parameter, it is possible to make the output entirely unipolar, or even entirely negatively unipolar, if desired. Note that the Output value is added directly to the value sent to the DAC, and therefore positive or negative offsets (that is, non-zero offsets) without any reduction in the Output range setting will result in the 16-bit values sent to the DAC overflowing and wrapping around. This will cause waveform deformations, which may be useful or interesting. To remove any such wrap-around deformation, reduce the Output range setting when using non-zero offsets. If Output range and Offset are left at their defaults of 230 and zero respectively, no waveform deformation will occur.
Freq range	sets the frequency range for the quadrature LFOs, with self-explanatory settings of cosm (cosmological), geol (geological), glacial (glacial), snail , sloth , lazy (very lazy), lazy , vslow (very slow), slow , med (medium), fast and vfast (very fast). The faster settings extend into audio range. The slowest period for one cycle of the LFO on the cosm setting exceeds 18 hours. Note that Quadraturia does not, and is not intended to track 1V/octave.
B freq ratio	sets the frequency ratio at which LFO2 (channel B) runs with respect to the master LFO (LFO1, channel A). The default is unity , so that it runs at the same frequency as channel A. Available ratios are 16/1, 15/1, 14/1, 13/1, 12/1, 11/1, 10/1, 9/1, 8/1, 7/1, 6/1, 5/1, 4/1, 3/1, 5/2, 2/1, 5/3, 3/2, 5/4, 1/1 (unity), 4/5, 2/3, 3/5, 1/2, 2/5, 1/3, 1/4, 1/5, 1/6, 1/7, 1/8, 1/9, 1/10, 1/11, 1/12, 1/13, 1/14, 1/15 and 1/16. 16/1 means the the frequency is multiplied by 16. 1/16 means the frequency is divided by 16, and so on.
C freq ratio	does the same as B freq ratio , except for the channel C LFO.
D freq ratio	*likewise*
B XOR A	if enabled causes the 16 bits of the channel B LFO current value to be bit-wise XORed with the bits of the channel A LFO current value before being output on channel B. Available settings are Off , or 1 to 8, where the number refers to the number of bit of right-shift that are carried out before bit-XORing. bit-XORing can create digital noise at audio rates, but at slower modulation rates, all sorts of interesting “toothed” or other geometric patterns may emerge - but with very sharp transitions in amplitude. Try feeding it into the cut-off of a LPF (a technique shown in a Bastl video somewhere).
C XOR A	if enabled causes the 16 bits of the channel C LFO current value to be bit-wise XORed with the bits of the channel A LFO current value.
D XOR A	if enabled causes the 16 bits of the channel D LFO current value to be bit-wise XORed with the 16 bits of the channel A LFO current value.
B AM by A	set the level of cross-channel amplitude modulation (AM), deaulting to none (zero), up to 127. Note that the amplitude modulation is inverted, so that higher values of the waveform in channel A result in lower amplitudes of the waveform in channel B. If the relative frequency of channel B is set at unity or lower, then a type of waveshaping occurs.
C AM by B	as for B AM by A except that the amplitude of channel C is modulated by the current output value of channel B.
D AM by B	as for B AM by A except that the amplitude of channel D is modulated by the current output value of channel C.
CV4: DEST	CV4 destination: cplg (coupling), sprd (shape spread), rng (range), offs (offset), a -> b (B AM by A), b -> c (C AM by B), or c -> d (D AM by B)
TR4: MULT	gated frequency division/multiplication factor (TR4): /8 , /4 , /2 , x2 , x4 , x8

Waveforms in the wavetable



Graphic courtesy of Mutable Instruments (https://mutable-instruments.net/modules/frames/manual/).

Tap tempo mode

The “predictive” tap tempo facility used in the Tap LFO mode in the Mutable Peaks (https://mutable-instruments.net/modules/peaks/) module has been added to *Quadraturia* in v1.3. This allows the period of the LFO waveform output on channel A to be locked to the period of “taps” or pulses (or, in fact, the rising edge of any regular input signal of more than about 1V amplitude) input on TR3. This will synchronise to both regular clock inputs as well as regular rhythms with unequal spacing of the taps. However, it **cannot** predict the future, and thus it will **not** be able to sync to irregular rhythms or randomly-timed pulses.

The frequency/period of the output on channels B, C and D will be the same as channel A if the frequency ratio setting for those channels is 1.0 (unity). Otherwise, it will be some multiple or division of the channel A frequency. In other words, the frquency ratio settings for channels B, C and D are honoured in **tap tempo** mode as well as in normal mode.

Screensaver display

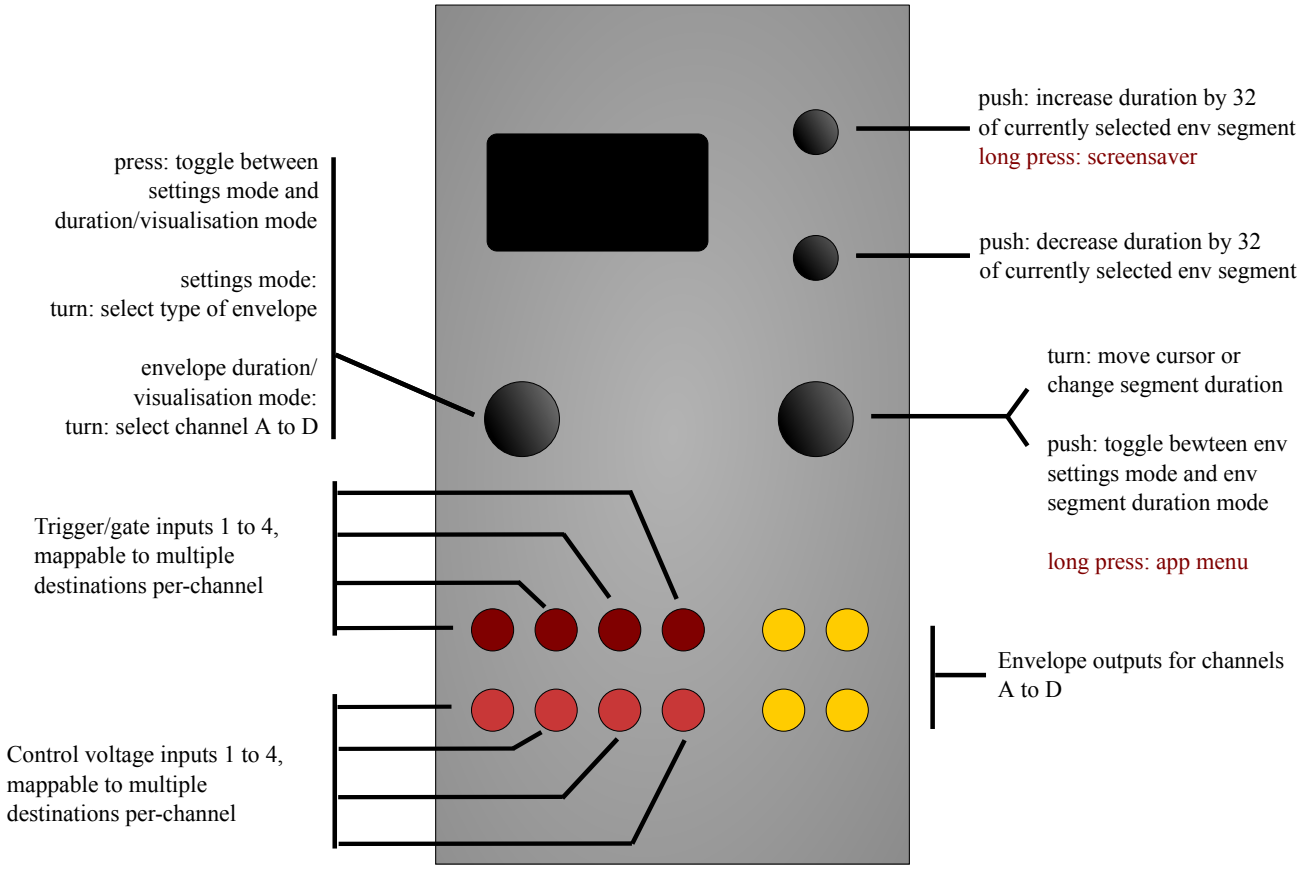
The screen is divided into quadrants, each showing a rolling display of the output values on each of channels A to D.

Tips

- To achieve the classic quadrature LFO patch (incremental 90° phase offset on outputs), set the phase/frq spread to +127.
- Try mixing the some of the outputs with a DC-coupled mixer (such as the Mutable Instruments Shades or Links modules) in order to create even more complex waveforms.

Piqued

Piqued is a a port of the envelope generator function from the open-source Mutable Instruments Peaks (http://mutable-instruments.net/modules/peaks) module. *Piqued* provides four independently-triggerable envelopes on output channels A to D, with independently mappable voltage-control via the CV1 to CV4 inputs over envelope duration and/or other parameters for each segment of each envelope. Triggers for each of the four envelopes can also be mapped from any of the four trigger inputs (TR1 to TR4). Segment shape (curves) can be set for each segment of each of the four envelopes. A variety of envelope types are available, also independently settable for each envelope, including repeating (looping) envelope types. The shape of each envelope can be visualised while setting parameters. There is also a “Euclidean trigger filter” included, which turns the *Piqued* app into a quad-channel Euclidean polyrhythm generator, that can output envelopes, not just gate/trigger signals. See the o_C videos page (/videos/) for some demonstrations of *Piqued*.



Controls

Piqued presents quite a rich UI (user interface), which is harder to describe in words than it is to use. The following explanations should make sense as soon as you see the UI in action. Once experienced, the interface becomes quite intuitive, we think.

Control	Function
Left encoder (turn)	In menu settings mode, elect the type of envelope. In envelope visualisation mode, select channel A to D to edit (all channels always active)
Left encoder (press)	Toggle between menu settings and envelope visualisation and segment duration settings.

Control	Function
Right encoder (turn)	Navigation mode: move up and down through the menu items (when in menu setting mode), or move back and forth between envelope segments (envelope visualisation/segment duration setting mode). Edit mode: increase or decrease the value being edited (segment durations when in envelope visualisation mode).
Right encoder (press)	Toggle between menu navigation (selection)/envelope segment selection mode and value editing mode
Right encoder (long press)	App selection menu
Up button (long press)	invoke screensave
Up button (press)	Increase duration for currently selected envelope segment for currently displayed envelope generator (A to D) by 32 (when in envelope visualisation mode only)
Down button (press)	Decrease duration for currently selected envelope segment for currently displayed envelope generator (A to D) by 32 (when in envelope visualisation mode only)

Available settings (per-channel)

Setting	Meaning
(envelope type)	the type of envelope (segment nomenclature is standard, with additions: A=attack, D=decay, S=sustain (level), R=release, L=loop point, with the number indication how many enevelope segments immediately before the L will be looped over). Available envelope types are described in the table below.
`Trigger input`	specifies which trigger/gate input, TR1 to TR4, is used to trigger or gate the envelope on the channel currently displayed. In addition, the end-of-cycle (EOC) of an envelope in one of the three other channels in *Piqued* can be specified as a trigger source. Thus, if `Trigger input` for channel B is set to `A EOC`, then chanel B will trigger as soon as the channel A envelope reaches it's end-of-cycle. In this way, envelopes across the channels can be chained or sequenced, in potentially complex patterns. By such internal cross-channel triggering of an envelope with a gate shape at the end-of-cycle of another envelope, EOC gate outputs can be emulated.
`Tr delay mode`	sets the mode for the trigger delay. See notes below.If enabled, the trigger delay will postpone the "firing" of the envelope (that is, the commencement of the attack segment of the envelope) for the time set by `Tr delay msecs` and `Tr delay secs` (see below). Available trigger delay modes are `Off`, `Queue` and `Ring`. `Queue` means that subsequent triggers received while a delay period is active are added to a queue for later action, up to a maximum queue depth set by the `Tr delay count` setting (maximum 32). Further triggers during the delay period are ignored until the number of queued triggers falls below the value set by `Tr delay count`. `Ring` is similar except that triggers received after the queue is full will replace the final trigger in the queue.
`Tr delay count`	sets the number of trigger delays that will be stored or buffered for later processing. The maximum setting is 24 triggers that can be delayed.
`Tr delay msecs`	trigger delay in milliseconds (range 0 to 999 milliseconds). If you set a trigger delay of greater than zero, then the envelope for that channel will not "fire" (commence its attack segment) until the specified delay has elapsed. The delay in milliseconds and the delay in seconds are added together, allowing very fine control over the delay. The "countdown" time for the delayed trigger is shown as a fall bar on the righthand side of the trigger indicator for that channel (at the top of the display).
`Tr delay secs`	trigger delay in seconds (range 0 to 64 seconds) — see `Tr delay msecs` above.

Setting	Meaning
`Eucl length`	sets the length of the Euclidean pattern (`Off` or range 2 to 32 in “beats”, where each beat is a received trigger/gate pulse) used to filter triggers for that channel. For a detailed explanation of Euclidean patterns and their use in rhythm generation, see this paper (http://cgm.cs.mcgill.ca /~godfried/publications/banff.pdf) by Godfried Toussaint, or for a brief explanation, see this presentation (http://www.maths.usyd.edu.au/u/joachimw/talk2.pdf). `Eucl length` defaults to `Off` , which means there is no filtering of triggers.
Euclidean pattern	is visible if `Eucl length` is not set to `off` . This displays the currently set Euclidean pattern. The current step is indicated by a small moving dash under the dashed line (one dash per step in the pattern, with the number of steps set by `Eucl length`). Pressing the R encoder invokes an “Euclidean editor”. When this editor is displayed, rotating the L encoder sets the fill setting for the pattern, and rotating the R encoder sets the offset or rotation of the patter. Both encoders can be used simultaneously. The fill setting is the number of beats in the pattern that let triggers through to “fire” the envelope. If the fill number is equal to or greater than the Euclidean pattern length number, then every incoming trigger will pass the Euclidean filter and fire the envelope for that channel. If the fill number is zero, then none shall pass, to quote Gandalf. The combination of pattern length (`Eucl length`) and number of active beats (the fill setting) within that pattern length uniquely determines the Euclidean pattern, using the Bjorklund algorithm. For example, if `Eucl length` is set to 8 and fill is set to 5, and the offset parameter is set to the default of 0, then the pattern will be 10110110, where 1 is an active beat (triggers are allowed to pass) and 0 is inactive (triggers are blocked). By setting offset to 1, the pattern becomes 01101101, if set to 2 the pattern becomes 11011010 and so on - that is, the pattern is rotated.
`Eucl reset`	defaults to `-` (off), but when set to a trigger input (TR1 to TR4), a trigger or gate on that input will cause the current position in the Euclidean pattern to skip back to the first position ie, it resets the step in the Euclidean pattern to 1. **Note:** if you set `Eucl reset` to the same trigger input used to trigger the enevlope for a particular channel (i.e. the `Trigger input` setting), and if `Eucl reset div` (see below) is set to the deafult of 1, then the envelope will either never fire or will always fire, because you will be constantly resetiting the Euclidean pattern to step 1 each time the envelope receives a trigger – and step 1 will be either 1 (fire) or 0 (don’t fire). In general, use a different trigger input for `Eucl reset` , and/or set `Eucl reset div` to greater than 1, or even better, to a value greater than the length of the Euclidean pattern.
`Eucl reset div`	is a divider for `Eucl reset` . For example, if set to 7, then the current step position in the Euclidean pattern is only reset to 1 after 7 triggers or gates have been received on the input specified by `Eucl reset` . It defaults to 1.
`CV1 ->`	sets the mapping from the CV1 input. The value on CV1 can be ignored (`None`) or sent to control the duration of one of: `Att` (ack), `Dec` (ay), `Sus` (tain) (level, not duration), `Rel` (ease), or `ADR` which affects the duration of all three of attack, decay and release simultaneously (†), or it can be set to control the Euclidean trigger filter parameters: `Eleng` (Euclidean pattern length), `Efill` (Euclidean pattern fill, `Eoffs` (Euclidean pattern offset/rotation), or the trigger delay time (`Delay`), or it can be set to control overall envelope amplitude (`Ampl`), or it can be set to `Loops` which controls the maximum number of enevelope loops (automatic re-triggers) for the looping envelope types. Note that when set to `Eleng` or `Efill` , negative voltages can be used to block all triggers. The input CV values are added to whatever is set for the duration/level or Euclidean parameters or trigger delay time or overall amplitude level or loop number via the respective menu settings.
`CV2 ->`	same as `CV1 ->` , but for the CV2 input
`CV3 ->`	same as `CV1 ->` , but for the CV3 input
`CV4 ->`	same as `CV1 ->` , but for the CV4 input

Setting	Meaning
Attack reset	specifies the behaviour when a new trigger or gate (i.e. rising edge of the gate) is received while the attack segment of an envelope is still in progress. The available values are `None` (new triggers are ignored while the attack segment is active), `SP` (reset Segment and Phase, which resets the segment to attack, which is was anyway, and restarts the segment at phase zero, but retains the current envelope level when the trigger was received, thus avoiding sudden jumps in envelope level and consequent audible clicks or pops, as far as possible), `SLP` (reset Segment, Level and Phase — this also resets the level back to zero — this may result in clicks or pops, as the change in level is instantaneous), `SL` (reset Segment and Level - this resets the level but not the phase - this also results in sudden level jumps, but of a different type), `P` (reset Phase only, which for the attack segment is the same as resetting both segment to attack and resetting phase). The default is `None`, and in general, the expected behaviour will probably result by setting it to `None` or `SP`. The other options are offered to permit experimentation. A dual-channel oscilloscope which allows simultaneous visualisation of the trigger pulses and the resulting envelope shapes is very useful in working out the exact behaviour, which can be complex when the period of a train of triggers is about the same as the period for the envelope being triggered. See also the `Att fall gt` setting, immediately below, which also affects the behaviour of the envelope types with a sustain segment when triggered by short-duration triggers or pulses.
Att fall gt	specifies the behaviour for the `ASR`, `ADSR` and `ADSAR` envelope types when a trigger or gate signal falls (i.e. when a falling edge is detected) during the attack segment of the envelope. The choices are: `Ignor` (ignore), the default - the falling edge of the trigger/gate is ignored, and the attack segment continues; or `Honor` - the envelope proceeds immediately to the segment immediately following the sustain segment (i.e. the release segment for `ASR` and `ADSR` types, and the second attack segment for the `ADSAR` type). `Honor` was the hard-coded behaviour prior to v1.3.
DecRel reset	specifies the behaviour when a new trigger or gate (i.e. rising edge of the gate) is received while the decay or release segments of an envelope are still in progress. The available values are `None` (new triggers are ignored while the decay or release segment is active), `SP` (reset Segment and Phase, which resets the segment to attack and restarts the segment at phase zero, but retains the current envelope level when the trigger was received, thus avoiding sudden jumps in envelope level and consequent audible clicks or pops, as far as possible), `SLP` (reset Segment, Level and Phase - this also resets the level back to zero - this may result in clicks or pops, as the change in level is instantaneous), `SL` (reset Segment and Level - this resets the level but not the phase - this also results in sudden level jumps, but of a different type), `P` (reset Phase only, which doesn't restart a new attack segment, instead, the current decay or release segment is restarted at phase 0 but at the current level, effectively extending it). The default is `SP`, and in general, the expected behaviour will probably result by setting it to `None` or `SP`. The other options are offered to permit experimentation. A dual-channel oscilloscope which allows simultaneous visualisation of the trigger pulses and the resulting envelope shapes is very useful in working out the exact behaviour, which can be complex when the period of a train of triggers is about the same as the period for the envelope being triggered.
Gate high	when set to `Yes`, forces the trigger/gate to high. This is useful when using the looping envelope types, which then just loop continuously, regardless of what is happen on their trigger/gate input, and thus they act as LFOs.
Attack shape	sets the shape of the attack segment for the currently displayed envelope. Available shapes are listed in the table below.
Decay shape	sets the shape of the decay segment for the currently displayed envelope. Available shapes are listed in the table below.
Release shape	sets the shape of the release segment for the currently displayed envelope. Available shapes are listed in the table below.
Attack mult	sets the duration multiplier for the attack segment(s). Range is 1 to 8192, which allows for very, very slow envelopes if desired.
Decay mult	sets the duration multiplier for the decay segment. Range is 1 to 8192. Note that you can still have a short attack but a very, very long decay or release, if desired, by setting different duration multipliers for each of attack, decay and release.

Setting	Meaning
`Release mult`	sets the duration multiplier for the release segment. Range is 1 to 8192.
`Amplitude`	sets the overall amplitude for the envelope, from 0 (no amplitude) to 127 (full range). Defaults to 127. Use it in conjunction with voltage control overall envelope amplitude to dynamically vary the envelope amplitude (that is, the maximum envelope level). See also the related `Sampled Ampl` setting, immediately below.
`Sampled Ampl`	toggles whether the overall amplitude value for the envelope is set continuously, or whether the overall amplitude value is sampled when a new trigger/gate is received and then remains at that sampled value until a new trigger/gate is received. It defaults to `off` .
`Max loops`	sets the maximum number of loops (automatic re-triggers) for the looping envelope types, range 0 to 127. A setting of 0 (zero) means “loop forever” (or at least while the gate input is high - see above), while numbers greater than zero mean loop for that number of times, then stop (at least until a new gate signal is received. New gate signals reset the loop counter. Note that there is settable CV control over this parameter. This allows these looping envelope modes to become CVable burst generators, with CV control of the number of repetitions in each burst of envelopes, as well as the shape, duration and amplitude of the envelopes that make up the burst. By self-patching, one or more other channels can be used to dynamically vary the nature of the burst (eg declining amplitude and increasing duration of each repetition in the burst, simulating an echo if used to drive a VCA etc).
`Inverted`	values are `Yes` or `No` , and `Yes` inverts the envelope output. The output voltage is shifted so that the inverted envelope drops below zero volts (to about -0.5V) at it’s peak (or rather, at it’s trough, or nadir), but this can be trimmed using the Amplitude setting so that it drops to exactly 0V if desired.

- (†) When the `ADR` CV input target is used with looping enevlope modes, each channel of Piqued effectively becomes a voltage-controlled LFO (but not with 1V/octave scaling), with settable waveshapes (via the envelope segment shape settings).

Envelope type	Description
`AD`	Attack-Decay: the attack segment commences on receipt of a trigger or on the rising edge of a gate signal, and the decay segment follows immediately after the attack segment has reached its peak, regardless of whether the gate or trigger signal is still high.
`ADSR`	Attack-Decay-Sustain-Release: just like every other ADSR envelope
`ADR`	Attack-Decay-Release: the attack segment commences on receipt of a trigger or on the rising edge of a gate signal, and the decay segment follows immediately after the attack segment has reached its peak. The sustain level is an inflection point for the decay - when the decay reaches the sustain level, the release segment immediately commences, regardless of whether the gate or trigger signal is still high. See the *Tips* section below on how to use the ADR mode as an AHR/AHD (attack-hold-release or attack-hold-decay) envelope generator with trigger signals.
`ASR`	Attack-Sustain-Release: the attack segment commences on receipt of a trigger or on the rising edge of a gate signal, and the envelope stays at maximum level for as long as the gate input for it remains high (sustain), and then commences the release segment as soon as the gate signal goes low.
`ADSAR`	is like an ADSR envelope, except that the attack segment re-triggers as soon as the sustain segment is finished, before proceeding to the release segment.
`ADAR`	is like the ADR envelope, except that the attack segment re-triggers as soon as the decay segment has finished, before going into the release segment.
`ADL2`	is like the AD envelope, except that it automatically re-triggers (i.e loops) the entire envelope (i.e. both the A and D segments) for as long as the trigger/gate input for it is high (see also the `Gate high` and `Max loops` settings)
`ADRL3`	is like the ADR envelope, except that it automatically re-triggers (i.e loops) the entire envelope (i.e. the A, D and R segments) for as long as the trigger/gate input for it is high (see also the `Gate high` and `Max loops` settings)
`ADL2R`	is like the ADR envelope, except that it automatically re-triggers (i.e loops) the A and D segments for as long as the trigger/gate input for it is high (see also the `Gate high` and `Max loops` settings), before proceeding to the R segment

Envelope type	Description
`ADAL2R`	is like the ADAR envelope, except that it automatically re-triggers (i.e loops) the D and second A segments for as long as the trigger/gate input for it is high (see also the `Gate high` and `Max loops` settings), before proceeding to the R segment
`ADARL4`	is like the ADAR envelope, except that it automatically re-triggers (i.e loops) the entire envelope for as long as the trigger/gate input for it is high (see also the `Gate high` and `Max loops` settings)

Illustrations of these envelope types can be found here (/envelopes/).

Segment shape	Description
`Lin`	Linear (a straight line, equation $x = t$ where t is time)
`Exp`	Exponential (equation $x = 1 - e^{-4t}$)
`Quart`	Quartic (equation $x = t^{3.32}$)
`Sine`	half a sine wave (equation $x = \sin(8 * \pi * t)$)
`Ledge`	almost a square wave, but with rounded corners, when used for attack, gives an immediate (punchy) rise, then a plateau. When used for decay or release, it has a plateau before falling.
`Cliff`	similar to `Ledge` , but when used for attack, has a delay before rising, when used for decay or release, it falls immediately.
`Gate` (v1.1)	is used when gate outputs are desired. The value rises immediately to maximum in the attack segment, and the value falls immediately to minimum in the decay and release segments. In other words, a pulse is output.
`BgDip`	Big dipper - has one large bump on the way up or down.
`MeDip`	Medium dipper - has a medium sized dip on the way up or down.
`LtDip`	Little dipper - has a little dip (more a ledge) on the way up or down.
`Wiggl`	Wiggles - lots of wiggles on the way up or down.

Inputs and outputs

Trigger input and CV1 to CV4 are mappable per-channel via the menu, as described above. Outputs for envelopes A to D appear on outputs A to D respectively.

Screensaver display

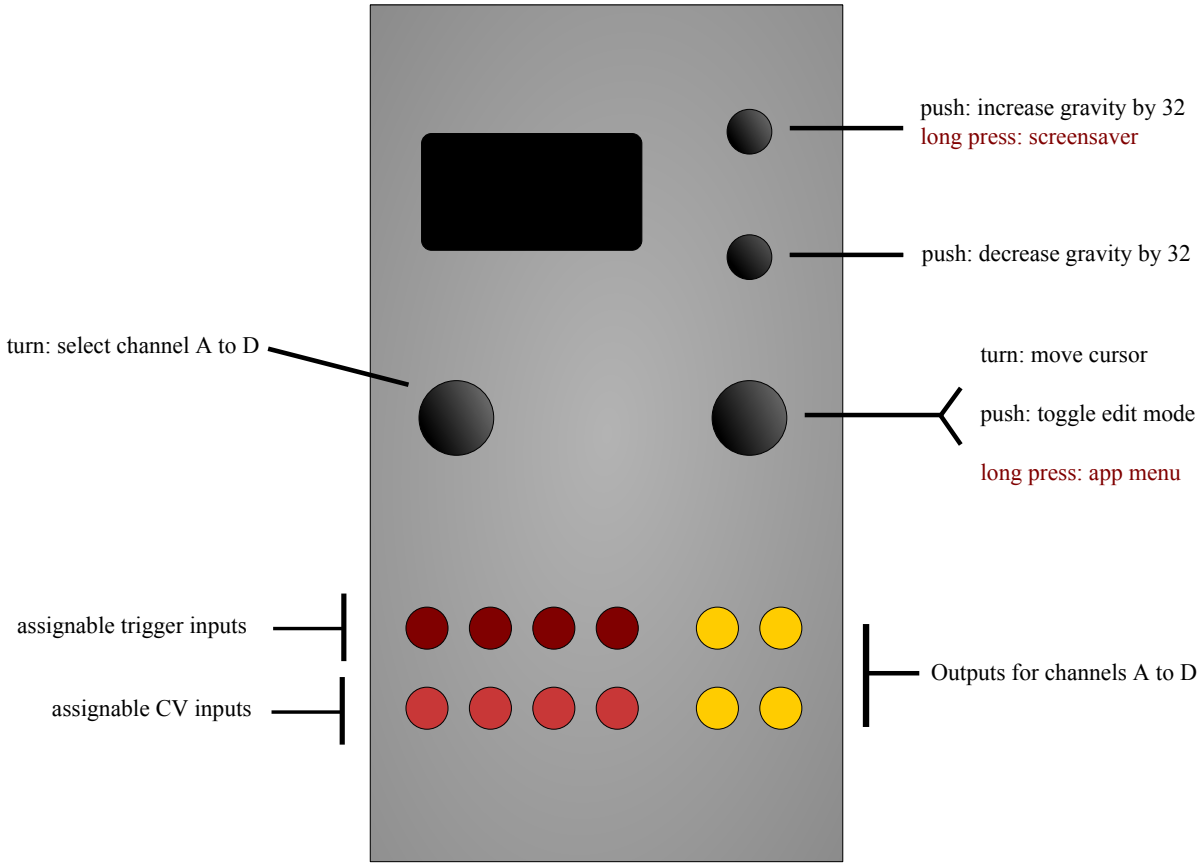
The screen is divided into quadrants, each showing a rolling display of the output values on each of channels A to D. Superimposed on this rolling value line is a representation of the envelope for that channel, as it progresses through its segments.

Tips and tricks

- the ADR mode can also be used as an AHR (attack-hold-release, sometimes also called AHD (attack-hold-decay)) envelope generator. AHR envelope are useful when you want to generate an envelope with a flat sustain period, which usually requires a gate input with some duration. If you only have trigger signals, that is, short pulses, then you can create an AHR envelope by choosing ADR mode, and setting the sustain inflection point to 255 (maximum). By doing that, the decay segment falls from maximum value to… maximum value - in other words, it's flat! The decay segment time/duration then sets the hold duration after a trigger is received. Also try setting the sustain level in ADR mode to something a little bit less than 255 (say 230), and set the decay shape to Wiggle. Now you have an AHR envelope with a wiggly and slightly downsloping plateau segment. Several variations on this theme are possible.

Dialectic Ping Pong

Dialectic Ping Pong is a port of the bouncing ball envelope generators from the Mutable Instruments Peaks module source code (these are not exposed in the official Peaks firmware, but are available on Peaks with the Dead Man's Catch (https://github.com/timchurches/Mutated-Mutables/releases) alternative firmware installed). These generators implement a simple but effective simulation of the physics of a ball that is thrown into the air with a certain velocity, from a certain height, and which then returns to Earth (or a planet of your choice) under the influence of (configurable) gravity, and then bounces (with a settable "bounce loss" simulating how hard the ball is pumped up, if it is a basketball), before being pulled back to Earth and bouncing again, and so on.



Controls

Control	Function
Left encoder (turn)	Select channel A to D to edit (all channels always active)
Right encoder (turn)	Navigation mode: move up and down through the menu items. Edit mode: increase or decrease the value being edited.
Right encoder (press)	Toggle between menu navigation (selection) mode and value editing mode
Right encoder (long press)	App selection menu
Up button (long press)	invoke screensaver
Up button	Increase gravity by 32
Down button	Decrease gravity by 32

Available settings (per-channel)

Setting	Meaning
`Gravity`	*g*, the acceleration due to gravity, from 0 (no gravity) to 255 (gravity on a massive alien planet)
`Bounce loss`	The amount of energy lost by the ball at each bounce (0 to 255). Higher values act like a deflated basketball that goes “splat!” instead of bouncing.
`Amplitude`	Initial amplitude (height) of the ball when the envelope is triggered, 0 to 255.
`Velocity`	Initial velocity of the ball when the envelope is triggered - the size of the kick or impulse imparted to the ball, if you like. Note that high values will cause the ball to bounce off the roof of the miniature gymnasium inside the module.
`Trigger input`	Trigger input source (TR1 to TR4) for the current channel
`Retrigger`	retriggers the ball after *x* bounces, where *x* is the value of this setting. There is configurable CV control over this parameter. This retriggering after *x* bounces makes the bouncing ball into a strange sort of LFO…
`CV1 ->`	Mapping of the CV1 input to a parameter for the selected channel. Values are “off”, “grav” (gravity), “bnce” (bounce loss), “ampl” (initial amplitude), “vel” (initial velocity) and `retr` (number of bounces before retriggering).
`CV2 ->`	As for `CV1->` , but for CV2.
`CV3 ->`	As for `CV1->` , but for CV3.
`CV4 ->`	As for `CV1->` , but for CV4.
`Hard reset`	If set to on, the envelope will instantly restart at the currently set initial amplitude, rather than starting from the height that the ball happens to be at the time when the trigger is received.

Inputs and outputs

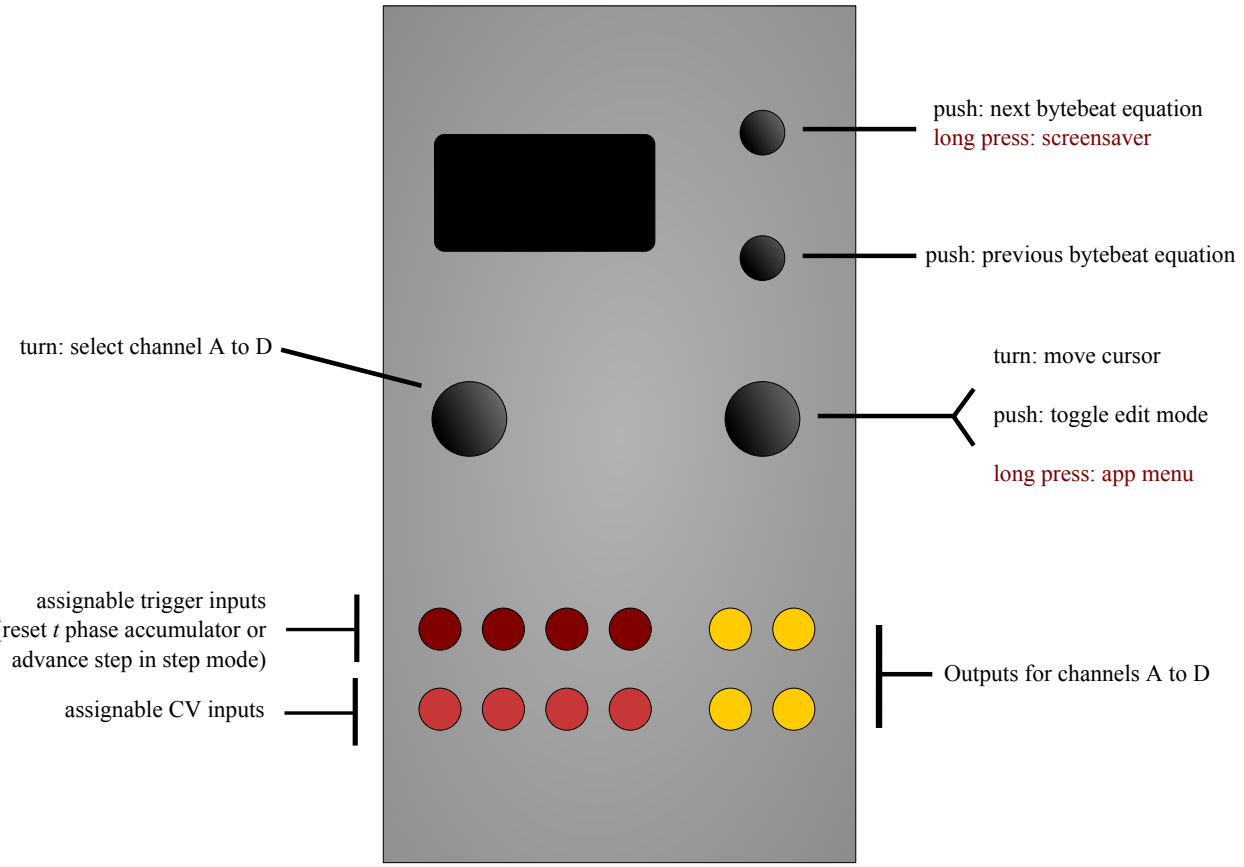
Trigger inputs and CV1 to CV4 are mappable per-channel via the menu. Outputs for channels A to D appear on outputs A to D respectively.

Screensaver display

The screen is divided into quadrants, each showing a rolling display of the output values on each of channels A to D.

Viznutcracker, sweet!

This is a experimental implementation of several bytebeats signal generators. “bytebeats” are equations (actually, recursive functions), expressed usually as a single line of programme code, typically involving various bit-level operators, which when evaluated with an incrementing phase value at audio rates produce all manner of harsh digital noises, some of which sound musical, or at least, interesting. bytebeats were first described (http://countercomplex.blogspot.com/2011/10/algorithmic-symphonies-from-one-line-of.html) in 2011 by viznut (aka Ville-Matias Heikkilä).



The output, if used as an audio signal, usually needs to be fairly heavily filtered through a low-pass filter to remove at least some of the unpleasant digital “screech” due to high-frequency aliasing and other effects which are characteristics of bytebeats. This digital aliasing is a fundamental characteristic of the way bytebeats work, and isn’t due to any hardware limitations of the o_C module.

The *Viznutcracker, sweet!* app current provides access to 16 different byte beat equations. The app provides four independent byte beat generators, on channels A to D, which all run independently. The equation, speed/frequency and three equation parameter values (p0, p1 and p2) can be set via the menus and/or voltage-controlled for each generator via mappable CV inputs.

Perhaps uniquely amongst byte beat generator modules, the *Viznutcracker, sweet!* apps permits the byte beat generators to be run at very slow rates, and because the o_C module outputs are DC-coupled, they can therefore be used as sources of stepped control voltages. For example, the outputs can be fed into a quantiser (such as another o_C module) to create potentially interesting pitch sequences (possibly even melodies…). Furthermore, the app allows each byte beat equation to be incremented by an external clock/trigger input, so that these stepped voltages can be generated in synchrony with other external processes.

Controls

Control	Function
Left encoder (turn)	Select channel A to D to edit (all channels always active)
Right encoder (turn)	Navigation mode: move up and down through the menu items. Edit mode: increase or decrease the value being edited.
Right encoder (press)	Toggle between menu navigation (selection) mode and value editing mode
Right encoder (long press)	App selection menu
Up button (long press)	invoke screensaver
Up button	Next bytebeat equation
Down button	Previous bytebeat equation

Available settings (per-channel)

Setting	Meaning
`Equation`	Byte beat equation (see table below)
`Speed`	0 to 255. 255 equates to a 16.7 kHz sample rate (i.e. the phase accumulator advances 16,666 times per second). There is a rough quadratic scaling of lower rates, meaning that 0 is quite slow.
`Pitch`	1 to 255. This is a very rough pitch setting, which behaves differently with each equation. Increasing values of `Pitch` will increase the pitch (frequency) of some elements of the equation output, but not necessarily all components, so it may not sound like a normal pitch increase or decrease. However, its effect does sound different to the `Speed` parameter. (shrugs shoulders)
`Parameter 0`	The first adjustable parameter in the chosen equation. Range 0 to 255, but some parameter settings in some equations do not produce any output, or do not produce output for all values of the phase accumulator (be patient!)
`Parameter 1`	ditto for the second adjustable parameter in the equation
`Parameter 2`	ditto for the third adjustable parameter in the equation
`Loop mode`	Enables loop mode, in which the phase accumulator is constrained to loop between specific start and end values, instead of between 0 and 4,294,967,296.
`Loop begin ++`	Coarse loop begin point, range 0-255
`Loop begin +`	Medium loop begin point, range 0-255
`Loop begin`	Fine loop begin point, range 0 - 255
`Loop end ++`	Coarse loop end point, range 0-255
`Loop end +`	Medium loop end point, range 0-255
`Loop end`	Fine loop end point, range 0 - 255
`Trigger input`	Specified which of the 4 trigger inputs (TR1 to TR4) is used for the trigger input for `Step mode`, or when `Step Mode` is off, which trigger input is used to reset the phase accumulator for that channel.

Setting	Meaning
`Step mode`	When set to on, the phase accumulator is incremented when a trigger or clock pulse is received on the digital (trigger) input specified by the `Trigger input` setting. When set to off, a trigger (or rising edge of a pulse or clock) received on the trigger input specified by the `Trigger Input` setting will reset the phase accumulator (the *t* variable in byte beat equations), which has the effect of resetting the byte beats "tune" or "melody" back to its beginning, or back to the start of the loop start point if loop mode is enabled.
`CV1 ->`	specifies which parameter CV1 is mapped to for this channel. Choices are `off` , `eqn` , `spd` , `p0` , `p1` , `p2` , `beg++` , `beg` , `end++` , `end` and `pitch` .
`CV2 ->`	ditto for CV2
`CV3 ->`	ditto for CV3
`CV4 ->`	ditto for CV4

Equation name	Source of equation
`hope`	"atmospheric, hopeful" via royal paw (http://royal-paw.com/2012/01/bytebeats-in-c-and-python-generative-symphonies-from-extremely-small-programs/)
`love`	the equation by stephth via here (https://www.youtube.com/watch?v=tCRPUv8V22o) at 3:38
`life`	the second equation listed here (http://xifeng.weebly.com/bytebeats.html)
`age`	"Arp rotator" via Microbe Modular (http://microbemodular.com/products/equation-composer/overview) Equation Composer Ptah bank (https://github.com/clone45/EquationComposer/blob/master/EquationBankPtah.cpp)
`clysm`	"BitWiz Transplant" via Microbe Modular (http://microbemodular.com/products/equation-composer/overview) Equation Composer Ptah bank (https://github.com/clone45/EquationComposer/blob/master/EquationBankPtah.cpp)
`monk`	"Vocaliser" via Microbe Modular (http://microbemodular.com/products/equation-composer/overview) Equation Composer Khepri bank (https://github.com/clone45/EquationComposer/blob/master/EquationBankKhepri.cpp)
`NERV`	"Chewie" via Microbe Modular (http://microbemodular.com/products/equation-composer/overview) Equation Composer Khepri bank (https://github.com/clone45/EquationComposer/blob/master/EquationBankKhepri.cpp)
`Trurl`	"Tinbot" via Microbe Modular (http://microbemodular.com/products/equation-composer/overview) Equation Composer Sobek bank (https://github.com/clone45/EquationComposer/blob/master/EquationBankSobek.cpp)
`Pirx`	"My Loud Friend" via Microbe Modular (http://microbemodular.com/products/equation-composer/overview) Equation Composer Ptah bank (https://github.com/clone45/EquationComposer/blob/master/EquationBankPtah.cpp)
`Snaut`	""A bit high-frequency, but keeper anyhow" via Microbe Modular (http://microbemodular.com/products/equation-composer/overview) Equation Composer Khepri bank (https://github.com/clone45/EquationComposer/blob/master/EquationBankKhepri.cpp)
`Hari`	"The Signs" via Microbe Modular (http://microbemodular.com/products/equation-composer/overview) Equation Composer Ptah bank (https://github.com/clone45/EquationComposer/blob/master/EquationBankPtah.cpp)
`Kris`	"Light Reactor" via Microbe Modular (http://microbemodular.com/products/equation-composer/overview) Equation Composer Ptah bank (https://github.com/clone45/EquationComposer/blob/master/EquationBankPtah.cpp)
`Tichy`	"Alpha" via Microbe Modular (http://microbemodular.com/products/equation-composer/overview) Equation Composer Khepri bank (https://github.com/clone45/EquationComposer/blob/master/EquationBankKhepri.cpp)
`Bregg`	"Hooks" via Microbe Modular (http://microbemodular.com/products/equation-composer/overview) Equation Composer Khepri bank (https://github.com/clone45/EquationComposer/blob/master/EquationBankKhepri.cpp)

Equation name	Source of equation
Avon	“Widerange” via Microbe Modular (http://microbemodular.com/products/equation-composer/overview) Equation Composer Khepri bank (https://github.com/clone45/EquationComposer/blob/master/EquationBankKhepri.cpp)
Orac	“Abducted” via Microbe Modular (http://microbemodular.com/products/equation-composer/overview) Equation Composer Ptah bank (https://github.com/clone45/EquationComposer/blob/master/EquationBankPtah.cpp)

Inputs and outputs

Trigger inputs and CV1 to CV4 are mappable per-channel via the menu. Outputs for channels A to D appear on outputs A to D respectively.

Screensaver display

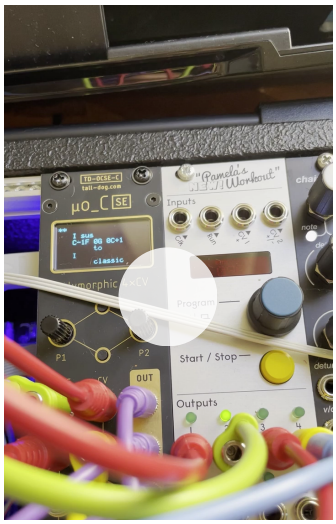
Tips

- as noted above, use the byte beat generators at very slow rates to generate DC control voltages. Feed these to a quantiser to produced pitch sequences, or pass them through a portamento or slew-limiting module to create interesting smooth modulation signals, or use them as-is, to modulate filters etc.
- process the byte beats outputs running at audio rates through a low-pass or shelf filter to remove some of the high-pitched screech
- process the output through band-pass filters (such as the Mutable Instruments Shelves (http://mutable-instruments.net/modules/shelves) filter), or other complex filtering arrangements subject to slow modulation (by another o_C module, perhaps)
- process the output through a VCA and/or filter with a percussive envelope to produce interesting, well, percussive sounds. Use the same trigger or gate signal used to fire the envelopes also reset the phase on the byte beat generator. Use slow modulation to slowly vary the frequency/rate, and/or to “scrub” loop start and end points in order to vary the nature of the percussive sound.

Passencore

Turn an LFOs into a chord progression? Harmonize a bass line that doesn't just play roots, with an eye to functional harmony and keeping voices within a range? Just want something in the lines `library` category for a platform not yet represented?

(this clip is my test patch right now, rather than an attempt at a "piece of music" so it's nothing special, but maybe it gives an idea)



0:00 / 0:21

Patch notes: Pam's triangle LFO into Passencore `tension` input, TR1 set to "sample" and "passing" and TR2 set to "target", both from Pam's at different times. Passencore 4x CV into Stages configured as a sequential switch into rings 4x polyphonic to make chords

Passencore is an Ornament & Crime app that replaces Meta-Q in an alternate firmware. It's a chord sequencer without the sequence, where you don't (have to) pick the chords. Instead, you patch in an LFO or two and some triggers for when you want it to pick a new chord. One of your trigger options is for a "passing chord", to lead you from whatever chord you're playing now to your target chord. I have imbued it with just enough music theory to be dangerous, because that's how much music theory I know. It knows:

- basic voice leading, but dgaf about parallel fifths because idgaf about parallel fifths
- functional harmony, ish, with the ability to substitute chords
- dominants are pretty cool
- sus chords are even cooler

The inspiration is somewhere between Acid Curds (for the s&h chord workflow), MI Grids (for how it's repeatable, but not exactly *predictable*), and some kind of "synf-onion" that I have only heard about but never seen.

Requirements

Ornament & Crime, Teensy bootloader, uUSB cable, willingness to try something new.

Documentation

Installation

Follow the **Ornament & Crime firmware installation instructions** with the firmware linked below.

Basic use

Patch a slow 0to5v LFO of some kind to CV1. This is “tension”.

Scroll down in the settings to make sure “sample trigger” and “target trigger” are both set to TR1 (that’s the default I set, but I’m somewhat new to o_C development, and sometimes I find my defaults set wrong, so ymmv.) Anyway, patch a trigger to TR1 that goes more than once per cycle of your LFO. Tune some oscillators to each other, and give all four outputs to them, or maybe give them to one oscillator with a sequential switch.

Your fancy trigger inputs

All assignable, so you can duplicate “sample” and “target” or “sample” and “passing” or whatever.

- `sample`: Read the values of *tension* and *color* and whatever other constraints and pick the next target chord based on them. Don’t necessarily play it yet, but do play it yet if `target` is also triggered.
- `target`: Play the target chord.
- `passing`: Play a passing chord to get to the target chord. If you’re already at the target chord, play a passing chord to get back there anyway. The algorithm currently likes sus chords and secondary dominants, but chromatic or stepwise passing chords are also possible.
- `reset`: Pick the next target chord with a root voicing. Useful for if you find yourself with a pattern that kind of wanders around based on the voice leading rules without repeating in as short a time as you like. Since the algorithm is deterministic (even if it’s complicated) `reset` can be a forced repetition point.

Your trusty CV inputs

CV1 *tension*, roughly equivalent to “function” in functional harmony. Goes from 0v is roughly “tonic”, 5v is roughly “dominant”, and intermediate voltages do predominants and other chords and stuff.
[include instructions inline, or provide link to external documentation, or attach a PDF/etc.]

CV2 *color*, roughly equivalent to how consonant the chord is going to be on its own. Goes from “power” (roots and fifths) through “classic” (triads) to “interesting” and “ext/subst” (which start adding in 7th and sus chords, and have weightings that allow for more functional substitutions, ex vi for I or iii for V) to “iono jazz?”. You can set the “base color” in settings, and CV2 adds to that. Default base color is “classic” for triads and the occasional seventh chord.

CV3 & CV4 Assignable. For now options that constrain the target chords we’re picking, but not the passing chords.

- `-`: nuthin'
- `include`: Input a note. It gets quantized to your active scale, and the resulting *target* chord must include that note.
- `root`: Input a note. It gets quantized to your active scale, and the resulting *target* chord has that root. Any inversion remains possible.
- `bass`: Input a note. It gets quantized to your active scale, and the resulting *target* chord has that note in output A.

Other business

- Settings for the center of each voice's range. Controls for octave and half-steps up or down from that.
- In certain cases Passencore wants to borrow a chord from another mode — chromatic passing chords and dominants or secondary dominants, usually. Allow or disallow, as you please.

How does it pick chords?

It does a few rounds of "add some chords to a buffer, evaluate them on some criteria, narrow down to just a few".

- Round 1: Add all triads, narrow by function and by required root note
- Round 2: Add variations like sus chords, sevenths, and ninths, narrow by function, color, required included note.
- Round 3: Add different voicings, narrow by all criteria above plus voice leading rules and required bass notes.

Pick the top-rated chord.

It does something similar for passing chords, but simpler, and populates with secondary dominants, sus chords, and chords rooted in between the notes it's trying to get between.

Bonus

Passencore works with all the modes of the diatonic scale and the harmonic minor scale; the harmonic minor and its modes are now in the global scale list.

NB I'm not actually that sure how applicable functional harmony is to the ultralocrian or whatever, but whatever my code does with it sounds interesting at least.

Roadmap

Features I want to add:

- CV assignments that let you modulate key & scale
- Ability to limit chord output to 3 outputs, and reserve one for a melody quantizer channel.
- Given a melody quantizer channel, options to let it quantize to the home scale or an "appropriate" pentatonic for the chord.

Enigma

Jump to bottom

bowlneudel edited this page 4 days ago · 3 revisions

Enigma is a Shift Register (a.k.a. "Turing Machine") Workstation, designed to enable (nearly) repeatable full compositions by curating a Turing Machine library and chaining Turing Machines into songs. Enigma is in Hemisphere Suite starting with v1.5.

Workflow

Enigma's workflow involves four screens:

- **Library** contains 40 Turing Machines (hereafter called "Registers"). As you audition a Register in the Library, you may change its length, probability, and rotation. You may "lock" a Register by designating it as a Favorite.
- **Assign** allows you to choose the function of each of the Ornament and Crime's four outputs (A, B, C, D). Each output may be assigned to a source track, a type (note, modulation, gate, or trigger), a scale (for note types), and a MIDI channel (1-16, or Off). On the Assign screen, you may choose whether you're auditioning the Library or the current Song.
- **Song** allows you to chain Registers together on up to four tracks. Each track may contain up to 99 steps, for a total of 396 steps. Each step specifies a Register, the number of times that Register repeats, the probability of a bit flip at each step (see "How a Turing Machine Works," below), and the transposition of notes played.
- **Play** allows you to control the playback of the song by choosing clock divide and loop per track, resetting and starting the song. It also allows you to visually monitor the song's position.

Navigation and Control Overview

The **LEFT encoder** chooses screens and screen-level selection. Each press of the LEFT encoder button moves to the next screen (in the order Library -> Assign -> Song -> Play). Turning the LEFT encoder selects items as follows:

- Library Screen: The Register (A-1 through E-8)
- Assign Screen: The output (A, B, C, D)
- Song/Play: The track (Track 1 through Track 4)

The **RIGHT encoder** chooses parameters and values on each screen. Each press of the **RIGHT encoder** button moves to the next parameter from left-to-right, top-to-bottom. Turing the RIGHT encoder changes the selected value.

The **UP and DOWN buttons** have different functions on each screen. Each screen will display the function of these buttons for a few seconds. You can dismiss this help by moving any control. During each session, Enigma will pay attention to how long you view the help text and will adjust the time accordingly. If you dismiss a help screen within one second, Enigma will stop showing them to you.

Long-pressing the DOWN button allows you to erase the current song and start over. You will be asked to confirm this action. On the confirmation screen, press the RIGHT encoder to erase the song, and the LEFT encoder to cancel.

Long-pressing the LEFT encoder button will send data related to the screen via system exclusive (SysEx) dump. Library, output assignment, and song data can be sent and received separately. Long-pressing the RIGHT encoder will return you to the main menu, and send all data via SysEx dump.

How a Turing Machine Works

The original Turing Machine circuit, by Music Thing's Tom Whitwell, works like this: A random sixteen-bit number is generated. On an incoming clock signal, the binary representation of that number is shifted to the left by one bit. A probability check determines whether the last bit (bit 15) of the pre-shifted number is moved back to the beginning (bit 0) during this rotation, or it changes its value. Then, an output value is determined based on the new sixteen-bit number. Originally, this was the bottom eight bits scaled to a voltage output.

How Enigma is Different

Enigma uses Turing Machine-like registers as its source material for composition. In a traditional Turing Machine or Turing Machine-like system, the starting register is different with each session. So if your Turing Machine hits upon a melody or modulation pattern that you like, you need to record it somehow, either with an audio recorder or CV recorder. The philosophy of Enigma is that Turing Machines can be durable and re-usable. When you find something you like, you can lock it in place and save it for later, and use it as part of a larger composition.

Library

The Library is where you manage your Registers. Forty registers are arranged in five banks (A-E) of eight registers (1-8). Use the LEFT encoder to select a register. When you first select a register, it starts with a random sixteen-bit number.

Registers are stored with three pieces of information: The register value itself, the length, and the Favorite status of the Register. Two other controls in the Library (probability and rotation) are only used for shaping the Register.

Note: The Library cannot be accessed or auditioned while a song is playing. If you're trying to get to the Library, but can't, go to the Play screen and stop the playback.

Auditioning the Register

A clock signal into Digital 1 will advance the register and calculate a change based on the probability value. The CV at Outputs A-D will be updated. The voltage present at the outputs is determined by the output assignments (see the Assign screen). By default, the assignments are as follows:

- Output A: A note quantized to a semitone scale
- Output B: A modulation amount from 0V-5V
- Output C: A gate (goes high when bit 0 is on, and low when bit 0 is off, capable of spanning multiple advances)
- Output D: A trigger (fired when bit 0 is on)

Favorites

If you find a melody, modulation pattern, or trigger pattern that you want to keep, press the UP button. This will mark the pattern as a Favorite (a heart icon will indicate this status for each pattern). When a pattern is a Favorite, the following things apply:

- Probability will no longer alter the pattern
- The pattern cannot be overwritten by single-Register SysEx dumps
- The pattern may now be reset using the DOWN button
- The pattern may now be rotated with the RIGHT encoder when the "rotate" icons are shown

When a Register is a Favorite, you may turn off Favorite status by pressing the UP button again. When you do this,

the probability will always start out at 0%.

Assign

The Assign screen is used to specify how the Ornament and Crime's outputs are used. You select the output with the LEFT encoder, and choose and change parameters with the RIGHT encoder. The parameters on the Assign screen, from top to bottom, are as follows:

- **Source:** Specifies which Track is routed to the output. A Track may be assigned to any number of outputs.
- **Type:** Specifies the type of CV that will be generated by the output. See the Types section below.
- **Scale:** Available only for note Types, chooses the scale to which the output is quantized
- MIDI Channel: Specifies the MIDI channel for the output, or Off

While you are on the Assign screen, you may audition from two sets of data, the Library or the Song. Press the UP button to audition from the Library (default), and the DOWN button to audition from the Song. If the song isn't playing, all CV outputs will be at 0V.

Types

The following types of CV output are available, based on the current state of the Register that's being auditioned:

- **Note:** There are five depths of note data, from 3 bits to 7 bits. At 3-bit depth, melodies will be constrained to 8 notes of the selected scale. At 4-bit depth, 16 notes; at 5-bit depth, 32 notes; at 6-bit depth, 64 notes; and, at 7-bit depth, 128 notes (or the entire MIDI note range) is available. The note types look at the least-significant bits of the current Register.
- **Modulation/Expression:** The least-significant 8 bits are used to generate a modulation value that's scaled to 0-5 volts. Modulation and Expression are identical for CV, but generate different MIDI messages (see below).
- **Gate:** When bit 0 is 1, the assigned output emits 5V until the next time bit 0 is 0. That is, a high gate will span multiple rotations of the Register
- **Trigger:** When bit 0 is 1, a trigger pulse is emitted at the assigned output

When the MIDI channel is set (that is, it's not Off), the Type also determines the MIDI event that's generated:

- **Note:** Sends a Note Off for the output's previous note (if any), and a Note On message based on the specified bit depth. For depths of 3-6, the note is above Middle C (that is, if a three-bit value is 5, the MIDI note number is 65 (Middle C, 60, plus 5). For a depth of 7, the seven-bit value determines the raw MIDI note number. If you want to play notes in rhythm, then set the MIDI Channel of the output to "Off" and specify a MIDI Channel for the next Gate or Trigger output (see below). Note that MIDI output does not observe the Scale setting used by the CV output.
- **Modulation/Expression:** The least-significant 8 bits are scaled down to a seven-bit number (0-127) for sending a MIDI modulation or expression event. Modulation is a continuous controller number of 1, and Expression is a continuous controller number of 11, which is often used for relative volume changes.
- **Gate/Trigger:** If a Note value was calculated in a previous output, but the MIDI Channel is Off, then that note is considered "deferred." A deferred note will be played if a MIDI Channel is assigned to a subsequent Gate or Trigger output. For MIDI output, Gate and Trigger do the same thing.

Song

Song mode allows you to chain Registers together into long (or short) compositions of up to four Tracks. This is done by creating a series of "Steps" on each Track.

Use the LEFT encoder to choose the Track (Track 1 - Track 4). The RIGHT encoder will cycle through the following

settings:

- **Step Number:** The Step number cursor is a blinking highlighted number showing the current Step number. Use the RIGHT encoder to move through the existing steps.
- **Register:** Choose a Register (from A-1 to E-8). When the cursor is over the Register selection, the Favorite status and length of the Register will be shown to the right.
- **Probability:** Probability is checked each time the Register is advanced, allowing the Register to randomly change over time. Note that this only changes the Register on playback, and not in the Library, regardless of the Register's Favorite status. When a new step starts, each Register is returned to its original form in the Library.
- **Number of Plays:** A Register can be played between 1 and 99 times for a single Step.
- **Transposition:** For each Step, notes played from the Track can be transposed over an eight-octave range, from -48 semitones to +48 semitones. Transposition is in semitones, regardless of the scale used, allowing you to use this value to specify a diatonic root.

Adding and Deleting Steps

A Track starts out with no Steps. To add a Step, press the UP button. This will insert a step *after* the selected step. The default Register of your new step will be randomly selected from among your Favorite Registers. If you have no Favorites, the default will be A-1.

To delete a Step, press the DOWN button. This will delete the currently-selected step, and the next Step up will now be selected.

Play

Songs may be played from the Assign or Song screens, but the Play screen provides some extra setup and playback control.

As with the Song screen, the RIGHT encoder is used to select the Track. This screen is formatted a bit differently, as a table with the following information:

- **Track Number**
- **Track Step Location:** Shows the Step number, and the number of times the Register has started playing during this step, after the semicolon
- **Clock Divide value:** Specifies how many clock pulses are required to advance the Register on this track
- **One Shot/Loop:** Specifies whether the Track plays once and stops, or loops. You can set up a short looping triggered rhythm that plays alongside a longer melody; or, a looping melodic ostinato that goes on throughout a composition.
- **Play Status:** The play icon, solid, indicates that the Track is playing. The play icon, blinking, indicates that the Track is paused, but will continue playback when the UP button is pressed. The stop icon indicates that the Track has run out of material to play.

Pressing the RIGHT encoder button alternates between the Clock Divide and the Loop setting on the selected channel.

On the Play screen, the UP button toggles between playback and pause. The current status is shown with an icon in the upper-right corner of the screen. The DOWN button resets the song to the beginning.

CV Control

The Digital and CV inputs have the same function whenever a song is being played. These controls are as follows:

Digital 1: The master clock. Advances all Tracks, observing the Clock Division of each Track. *This is the only CV control that works in the Library* **Digital 2:** Reset the song **Digital 3:** Reset the song and start playback **Digital 4:**

control that works in the Library **Digital 2:** Reset the song **Digital 3:** Reset the song and start playback **Digital 4:** Toggle between playback and pause **CV 1:** Gate Song End. When a high signal is sent to this input, the song will stop playing when all non-looping Tracks have ended. This can be used to end a song for recording, without having to fade out. **CV 2:** Gate Song Repeat: When a high signal is sent to this input, the song will start over from the beginning when all non-looping Tracks have ended. *When both CV 1 and CV 2 are un-gated, the default behavior is for looping Tracks to just continue looping until otherwise stopped.*

Storage

Internal Storage

As with other Ornament and Crime applications, the state of the Ornament and Crime can be stored by long-pressing the RIGHT encoder button, and then long-pressing it a second time at the main menu. This will store the following data for Enigma:

- All 40 Registers in the Register Library
- The current output assignments (as seen on the Assign screen)
- 32 Steps of the current Song
- The Track settings, Clock Division and One-Shot/Loop (as seen on the Play screen)

The Ornament and Crime's internal EEPROM is too small to save all 396 possible song Steps, so it only saves the first 32 Steps. If the Song is longer than that, then only 32 Steps will be restored when the module is powered up again. To help determine whether your entire song will be saved, there is a percentage counter in the upper-right corner of the Song screen. When your song exceeds 32 steps in length, the percentage will be highlighted. To save the entire song, you'll need to use System Exclusive storage (see below).

System Exclusive Storage

As with all Hemisphere Suite applications, returning to the main menu with a long-press of the RIGHT encoder will initiate a SysEx dump of all app data. For Enigma, this includes the entire Register Library, the current output assignments, all Steps of the current song, and the Track settings. If you save this data with a SysEx Librarian, you can dump it back any time Enigma is running to restore the entire app state.

You may initiate dumps of subsets of information by long-pressing the LEFT encoder button. In the Library, this will dump all 40 Registers; on the Assign screen, it will dump the output assignments; on the Song or Play screen, this will dump the Song's steps and Track settings.

When you mark a Register as a Favorite in the Library, Enigma will send SysEx for the single Register. If you send this SysEx back while in the Library, it will be saved to the current Register location, provided that is not marked as a Favorite. If it is, the incoming dump will be ignored.

A single Register dump received outside the Library will be stored in its original location and--again--only if the current occupant of that location is not a Favorite.

Credits

Pong

Jump to bottom

bowlneudel edited this page 4 days ago · 1 revision

Like twenty years ago, I owned a Kurzweil K2000. It had what we call today--but didn't call back then--an Easter Egg. It was a Pong game that you could play from the panel, and it generated MIDI notes when the ball bounced off a wall.

This is the Pong we all know and love, with a few twists. As a ball bounces its way across the screen, the player defends the left side of the screen with a "paddle," and the module defends the right side. It's an unfair game, though, because the module can't lose. As you return the ball and level up, the ball gets faster, and your paddle gets smaller and closer to your opponent. The odds are not in your favor!

Controls

- Up/Down Buttons: Move the paddle up and down. This is really to illustrate the use of the buttons' event handler, and you really don't want to play the game with these things.
- Encoders: Both encoders move the paddle up and down.
- CV Input 1: Negative values move the paddle up, and positive values move the paddle down. There's a "center detent," a small range that doesn't move the paddle at all. This is to compensate for noise that gets into the ADC.
- Output A: When the ball bounces off your paddle, a short 5V trigger is sent to Output A.
- Output B: When the ball bounces off anything else, a short 5V trigger is sent to Output B.
- Output C: Sends 0 to 4-ish volts, based on the Y position of the ball. 0V is the top of the screen.
- Output D: Sends 0 to 4-ish volts, based on the Y position of the player paddle. 0V is the top of the screen.

Exercises

1. Create a patch that can be played by this game
2. Create a patch that can play this game

My patch's high score is 24, using Maths and Distro. Update 8/31/2018: 27 with just Maths!

Credits

Backup and Restore

Jump to bottom

Chysn edited this page on Nov 30, 2018 · 18 revisions

Backup/Restore is a utility that allows you to transfer complete sets of app and/or calibration data to and from your module. Possible uses include

- Transferring app data to a new O_C module,
- Moving calibration to a replacement Teensy for the same O_C module,
- Changing settings between pieces during live shows,
- Intercontinental internet-based collaboration with other Hemisphere Suite users, or
- Simply preparing for inevitable disaster.

Backing Up Your Module

Backup/Restore backs up data as it exists in the EEPROM. So if you want to take a backup of the O_C's *current* state, save the module's state by long-pressing the right button from the main menu.

Connect your computer or tablet to the module and enter the Backup/Restore app on your module. Turn either encoder to choose "Data" or "Calibration." Enable recording on your SysEx Librarian software, and then press the right encoder button ([BACKUP]). The size of the data dump will be 172 bytes for Calibration, and about 2.5K for Data.

Restoring Your Module

Connect your computer or tablet to the module and enter the Backup/Restore app on your module. Press the left encoder button ([RESTORE]). Backup/Restore will indicate that it is listening for SysEx data. From your SysEx Librarian software, initiate the dump. The module's screen should indicate a progress bar. When the restore is successful, the module returns to the saved state.

If you use Backup/Restore to restore calibration settings, you'll need to cycle the power on your module for the restored calibration settings to become active.

Setup About

Jump to bottom

bowlneudel edited this page 2 weeks ago · 7 revisions

Press the LEFT encoder to enter the Calibration routine

Press the RIGHT encoder to reset calibration / settings to defaults - press RIGHT again to confirm (LEFT to cancel)

On the calibration confirmation screen, you can toggle user calibration / settings or default

- Rotate RIGHT encoder to change
- To edit user calibration, select "No", and press the RIGHT encoder (OK)
- To accept the defaults, select "Yes", and press the RIGHT encoder (OK)
  - *Note: this will accept the displayed encoder reversal setting*
- To cancel, press the LEFT encoder (Cancel)

Encoder scrolling direction reversal

There is no cursor wrapping in the calibration routine, so note the displayed list of encoder scrolling reversals at the calibration confirmation screen.

In the calibration confirmation page OR last page, press either of the UP and DOWN buttons to select encoder reversal for L, R, both (LR), or neither (normal).

If you want to use non-default calibration, you will need to scroll through the entire calibration routine to save the encoder reversal setting (you will have another opportunity to edit encoder settings at the last page).

Press the RIGHT encoder to accept the listed encoder reversal settings.

Calibration routine

To change pages:

- Rotate the LEFT encoder to paginate, or
- Press the RIGHT encoder to go to the next page
- Press the LEFT encoder to go the previous page

To edit the current page parameter:

- Rotate the RIGHT encoder

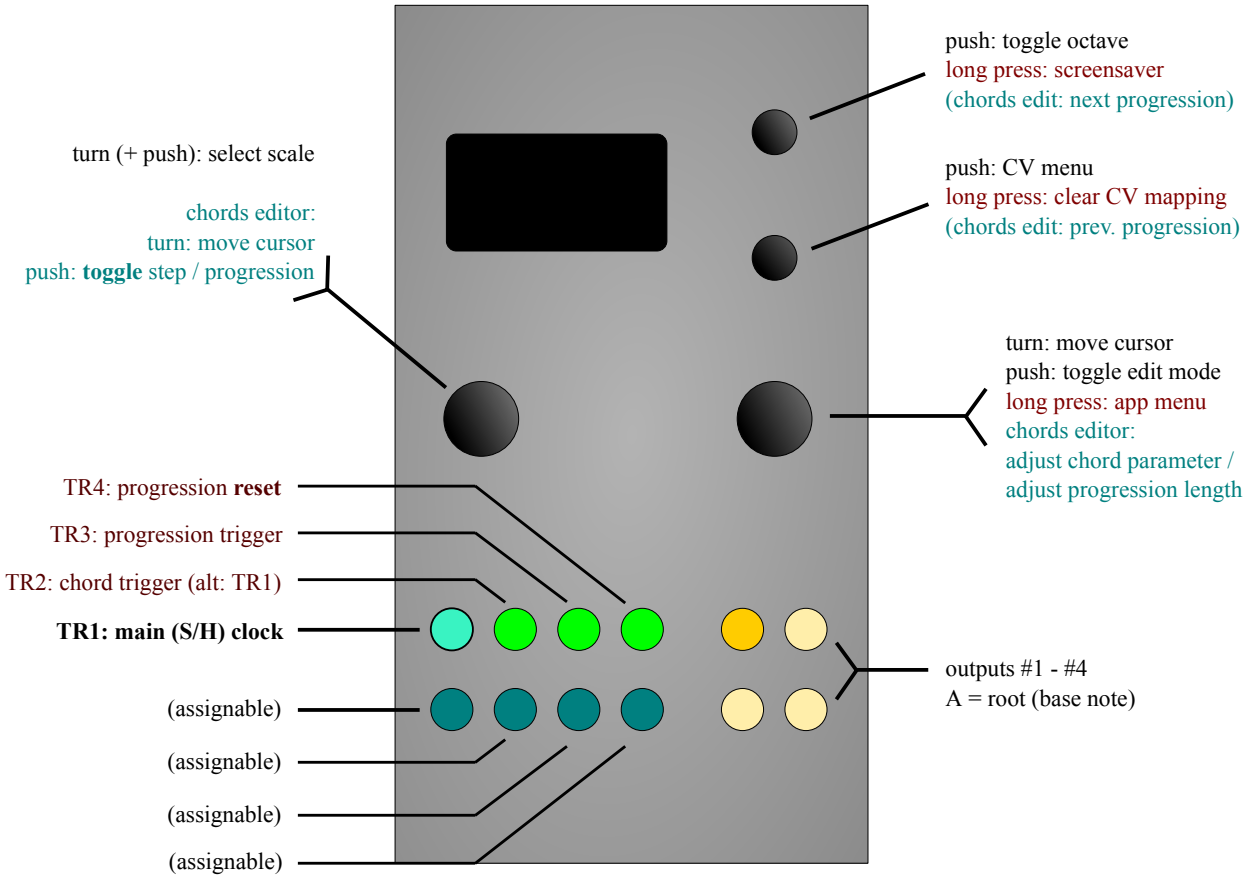
Pages:

- (1) Center Display horizontal pixel offset
- (2-11) DAC A -3v through 5v
- (12-21) DAC B -3v through 5v
- (22-31) DAC C -3v through 5v
- (32-41) DAC D -3v through 5v
- (42) ADC CV1
  - *All ADC values should be around 0v (-1v ≀ 1v)*
- (43) ADC CV2
- (44) ADC CV3
- (45) ADC CV4
- (46-47) CV Scaling 1v and 3v
- (48) Screen blank time
- (49) Save and adjust encoder direction
  - Press UP and DOWN buttons to select encoder reversal for L, R, both (LR), or neither (normal)
  - Select "Yes" to save and "No" to cancel, and press the RIGHT encoder to exit

Acid Curds

Acid Curds is a basic chord sequencer. The app provides **four 8-step chord progressions** (in total), and CV-control over various parameters, including chord type, progression length, direction, voicing, inversion, etc. The four chord progressions can be **chained** in various ways, providing progressions from **1 to up to 32 chords**. The chords/sequence data is stored along with the other settings (whenever settings are saved).

- *Acid Curds* can be used as either a basic quantizer / sample-and-hold type thing, in which case the chords will be formed on the basis of the incoming root CV, or it can be used as a step-sequencer, in which case the chord base-notes and chord properties will be fixed notes (chosen from a given scale) respectively features (inversion, voicing, etc); it's also possible to combine sequencer and S+H type behaviors. Here is the basic i/o mapping; the CV inputs are freely assignable:



- the **chords editor** works much like the scale and sequence editors in Copiermaschine, Sequins, etc:
  - in the main menu, select which progression (#1-#4) to edit by adjusting the `progression` parameter.
  - then right-click on the item `chords -->` to **open the chords-editor**:
    - adjust the progression **length** by pointing the cursor to the far right (using the left encoder), then turn the right encoder.
    - turn the **right** encoder to change the selected chord-parameter (highlighted w/ white background).
    - turn the left encoder to either select which chord (step) to edit, or to select which step-feature to edit.
    - pushing the left encoder **toggles** between chord-select and feature-select.
    - push the right encoder to close the editor again.
- a chord (or step) consists of five features: quality/type (`Q`), voicing (`V`), inversion (`I`), base note (`B`),

and the register/octave (`0`). most of the parameter values should be fairly self-explanatory.

- the 'base note' (or root note) of a chord can be set to either `CV` , in which case a chord will be formed based on the voltage present at input CV1 (TR1 in that case is used as the S+H clock, the chords progression will advance depending on the `chords trg src` setting); or, it can be set to a fixed value: `#1` , `#2` , `#3` , etc (these values are given in scale degrees, e.g. choosing `#3` = the third note of a given scale).

- the `playmode` and `direction` settings offer various ways of moving through the sequence(s):

    - `-` (default): advance by trigger, using the chosen direction settings.
    - `SEQ+1 - SEQ+3` : ditto, but **cycling** through 2, 3, or 4 adjacent progressions (= chain up to 32 chords).
    - `TR3+1 - TR3+3` : ditto, but **jumps** to the next progression only *if/when* a trigger is received at the **TR3** trigger inputs.
    - `S+H#1 - S+H#4` : CV-address, triggered by **TR3** (= **sample and hold**).
    - `CV#1 - CV#4` : CV-address, **free-running**.

Inputs and outputs

I/O	Function	-
TR1	main clock / S+H input	-
TR2	chord advance trigger (if selected via `chords trg src`)	-
TR3	playmode trigger `TR3+1 - TR3+3` , CV-address trigger (`S+H#1 - S+H#4`)	-
TR4	progression reset	-
CV1	(mappable)	-
CV2	(mappable)	-
CV3	(mappable)	-
CV4	(mappable)	-
A, B, C, Dchord outputs		A is the base note

Available settings

Setting	Meaning
`--> scale`	edit current scale
`root`	scale root
`progression`	select progression `#1 - #4`
`chords -->`	edit progression/chords
`playmode`	chain sequences (`SEQ+x`), advance by TR3 (`TR3+x`), or CV-address
`direction`	forward, reverse, pendulum1, pendulum2 (repeat first/last), random, brownian
`transpose`	transpose (in scale-degrees)
`octave`	transpose in octaves
`CV source`	CV source (`CV1` , `CV2` , `CV3` or `CV4`
`chords trg src`	chords-trigger: `TR1` or `TR2`
`TR1 delay`	TR1 trigger-to-processing latency - see the discussion of the `Trigger delay` setting in the *quantermain* app for more details

Controls

main menu

Control	Function
Left encoder (turn)	select scale
Left encoder (press)	activate scale

Control	Function
Left encoder (long press)	-
Right encoder (turn)	Navigation mode: move up and down through the menu items. Edit mode: increase or decrease the value being edited
Right encoder (press)	Toggle between menu navigation (selection) mode and value editing mode
Right encoder (long press)	app selection menu
Up button	toggle up/down by one octave
Down button	go to CV menu
Up button (long press)	screensaver
Down button (long press)	clear CV mapping

CV menu

- **enter** the CV menu by holding down the **down** button.
  - use the right encoder to assign CV input channels 1-4 to a channel parameter (currently available parameters are: `root` (scale/global), `scale mask`, `transpose`, `octave`, `voicing`, `inversion`, `progression #`, `direction`, and `progression length`.
  - **return** to the main menu by either pressing the **up** or **down** buttons.
  - **clear** all mappings (per channel) by long-pressing the down button.

chords editor

Control	Function
Left encoder (turn)	select chord-step, or select chord-feature
Left encoder (press)	toggle step-select / feature-select
Right encoder (turn)	1) adjust chord feature; 2) adjust progression **length** by pointing the cursor to the far right, then turn
Right encoder (press)	exit editor
Up button	go to next progression (edit 'offline')
Down button	go to previous progression (edit 'offline')
Left encoder (long press)	-
Down button long press	-
Right encoder (long press)	– (app selection menu)
Up button long press	– (screensaver)

Tips

- usually what trips people up is that by default there's two clocks involved:
  - TR1 = S+H
  - TR2 = "chord trigger"
- a pulse into TR1 will make the module sample the CV inputs (which won't do anything if you choose a static 'base' note)
- a pulse into TR2 will actually step to the next chord in the progression (if any)
- you can map both types of behaviour to TR1, see the `chords trg src` parameter above

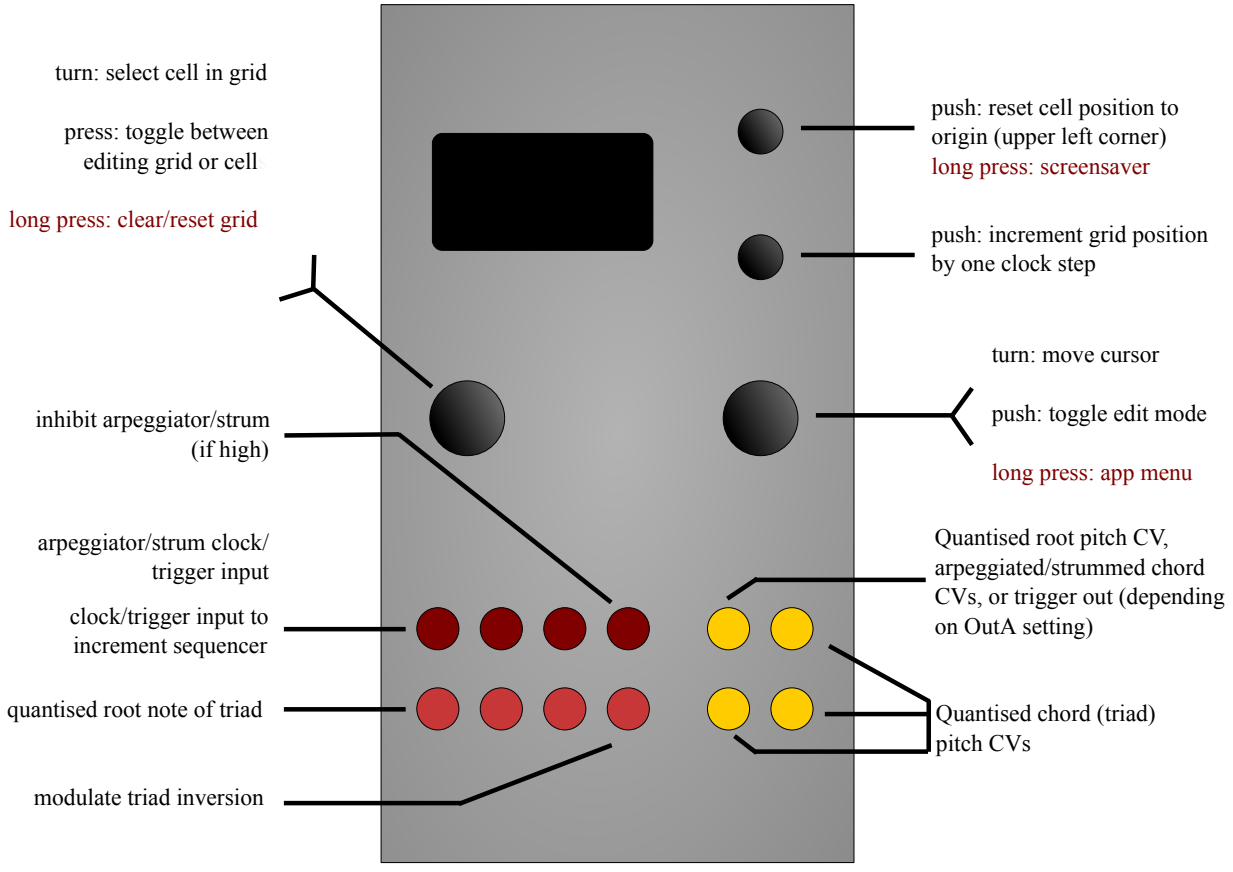
Automatonnetz

This app uses the neo-Riemannian transformations implemented in the Harrington 1200 app (see above), but with the sequence of transformations determined not through triggers for each type of transformation, but rather by navigating a 5x5 grid of cells. On each clock input the `dx` (delta x) and `dy` (delta y) — hence "vector" sequencer — values are added to the current position on the grid to determine the next cell. The position simply wraps around when it reaches the edge of the grid, and "backwards" motion is also possible. The position and movement can also be fractional, allowing for clock divisions and all kinds of patterns.

Each cell of the grid can contain a neo-Riemannian transformation, or a reset, as well as other parameters. In this app, there are three additional transforms available, which can be represented as a combination of the basic three neo-Riemannian transforms:

- S (Slide): *LPR*, example S(Cmaj) = C#min
- H (Hexatonic): *LPL*, example H(Cmaj) = A-min
- N (Nebenverwandt): *RLP*, example N(CMaj) = Fmin

The implementation computes these in a single transform step however, not sequentially.



Controls

Control	Function
Left encoder (turn)	Select cell in the grid — the 25 cells are accessed sequentially, row-wise.
Left encoder (press)	Toggle between editing overall grid settings or current cell settings
Left encoder (long)	Clear/reset grid (the results of this action depends on `Clr` setting)
Right encoder (turn)	Navigation mode: move up and down through the menu items. Edit mode: increase or decrease the value being edited.

Control	Function
Right encoder (press)	Toggle between menu navigation (selection) mode and value editing mode
Right encoder (long press)	App selection menu
Up button	Reset current cell position to grid origin (top left-hand corner of grid)
Down button	Increment grid position by one clock step

Grid settings

Setting	Meaning
`dx`	Amount of movement along x-axis (horizontal) per clock input
`dy`	Amount of movement along y-axis (vertical) per clock input
`Mode`	Musicological mode of root triad, either `maj` or `min`
`Oct`	Move outputs up/down in octave steps
`TrDly`	sets the TR1-input-to-processing latency (for details see the `Trigger Delay` setting in *Quantermain* below)
`OutA`	Switch output mode of channel A: `root` outputs root note, `trig` outputs a trigger whenever the triad output on B, C and D is transformed, `arp` arpeggiates the current triad, `strm` (strum) arpeggiates the triad once only as soon as the triad transformation has taken place (tip: very useful with the Mutable Instruments Elements or Rings modules, or Mutable Instruments Braids in “PLUK” mode)
`Clr`	Sets how the grid is cleared on a long-press of the left encoder. `zero` clears the grid, `rT` fills with random transforms, `rTev` sets each cell’s event to `randT`

Per-cell settings

Setting	Meaning
`Trfm`	Determines the transform which is applied when this cell is active; special values are `@` (reset) and `*` (no transform)
`Offs`	Offset in semitones applied while this cell is active
`Inv`	Inversion of the transformed triad
`Muta`	Mutation event that is applied when the cell is left (i.e. on the next clock after the cell’s transform is applied). Valid values are shown in the table below. Note that this setting makes the grid self-modifying as the current cell traverses it!

`Muta` setting	Action
`none`	nothing happens
`rT__`	The transformation for this cell is set to a random value.
`r_O_`	The transposition for this cell is set to a random value.
`rTO_`	The transformation and the transposition for this cell is set to a random value.
`r__I`	The inversion for this cell is set to a random value.
`r_OI`	The transposition and the inversion for this cell is set to a random value.
`rTOI`	The transformation, the transposition and the inversion for this cell is set to a random value.

Input/output assignment

I/O	Function
TR1	Clock/trigger input to increment steps in the sequencer
TR2	Arpeggiator clock (if mode is `arp` or `stem`)
TR3	
TR4	If high, inhibits arpeggiator clock
CV1	The voltage on this input is quantised to the root note of triad (before transform) - that is, it provides external voltage control of the root note (same as Harrington 1200)

I/O	Function
CV2	
CV3	
CV4	Modulate triad inversion (same as Harrington 1200)
A	Depending on the `OutA` setting: pitch CVs for quantised root note, arppegio/strum, or trigger out
B, C, D	Pitch CVs for the triad after transformation

Screensaver display

Similar to Harrington 1200 app, the current triad (output as pitch CVs on the B, C and D sockets) is shown graphically on a pitch circle on the left. On the right, the last few vector moves are shown as a “snake”. The current output triad is also displayed.

Tips

If you wish to use the vector sequencer to play melodies, then set the TRFM (transform) value for every cell in the grid to * (null transform), and set the `Offs` value for each cell to a specific note offset from the root note that you want to appear in your sequence. Then, as the current cell is moved around the grid by clock/trigger inputs on TR1, the note defined for that cell will be output on output B (with transpositions of the same note sequence on outputs C and D).

Captain MIDI

Jump to bottom

bowlneudel edited this page 4 days ago · 5 revisions

Captain MIDI is a highly-configurable CV-to-MIDI and MIDI-to-CV interface. It supports four outputs for MIDI-to-CV and four inputs for CV-to-MIDI. It features multi-channel operation, polyphonic note distribution (up to four notes), transposition, and note-range for layers and/or splits. It keeps a log of the last 100 MIDI messages. Up to four complete Setups can be saved in memory. Setups can be copied from the panel, or saved and retrieved via MIDI system exclusive dump.

Videos

- https://youtu.be/PN6EEVkcqJ8
- https://youtu.be/l1i4FIAcubw
- https://youtu.be/pzHHOJ3jrCk (Uhost)

References

Blog post on Captain MIDI with SuperCollider: https://madskjeldgaard.dk/posts/hemisphere/

Connection

The back of the Ornament and Crime module has a micro USB port. This is what you use to make a MIDI connection to a USB host, usually a Windows, macOS, or Linux computer. Hemisphere Suite uses a class-compliant MIDI interface, which should be recognized as "Hemisphere" by your operating system and DAW/MIDI software.

Basic Controls and Navigation

- RIGHT encoder: Selects parameter or value. Pushing the RIGHT encoder switches between parameter selection and value editing.
- LEFT encoder: Selects Setup screen (Assign, Channel, Transpose, Range High, Range Low). Pushing the LEFT encoder toggles between the Setup screen and the Log Display. When the log is displayed, the LEFT encoder scrolls through the last 100 MIDI events sent and/or received. A LEFT encoder button long-press (held for about 2 seconds, and released) initiates a Panic! function: Captain MIDI sends Note Off for all notes and on all channels.
- UP/DOWN buttons: Selects which Setup (from 1-4) is active. If the DOWN button is long-pressed, the active Setup may be copied to another Setup.

Setup Screen Basics

Each of the Setup screens has eight parameters. The input or output (I/O) name is shown on the left, and the value is on the right. The parameters are arranged in groups of four, with the MIDI-to-CV parameters and then CV-to-MIDI parameters:

- A name like "MIDI > ?" indicates that a MIDI message of the assigned type, on the assigned channel, is routed to the corresponding letter output.
- A name like "? > MIDI" indicates that a MIDI message of the assigned type, on the assigned channel, is sent when the voltage at the corresponding CV input changes or, in the case of notes, when a gate is received at the corresponding digital input. If the assigned type is Note or Legato, and a MIDI channel is set, an eighth-note icon appears next to the parameter name. This indicates that note-specific parameters (transpose, range) apply to the assignment.

Each parameter line also displays a MIDI indicator icon when its assignment is sending or receiving MIDI data. For note assignments, a note name is displayed instead of a MIDI icon.

Assign Setup Screen

On this screen, you determine which MIDI messages are sent or received.

MIDI-to-CV (MIDI In) Types

- **Note:** When a Note On message is received on the specified channel, the assigned output sends a quantized pitch value corresponding to the message's MIDI note number. Note may be assigned to any number of outputs. If Note is assigned to multiple outputs on the same channel, reception will be polyphonic; each new simultaneous note will be picked up by a different output until no more Note outputs are available. You may have up to four notes of polyphony by assigning all four outputs on the same MIDI channel; but you'll have to find another way to control your VCAs, since you won't have any outputs for gates.
- **Gate:** When a Note On message is received on the specified channel, the assigned output sends a high (approx. 5 volt) signal, which remains high until the corresponding Note Off message is received.
- **Trig:** When a Note On message is received on the specified channel, a trigger is sent on the assigned output.
- **Veloc:** When a Note On message is received on the specified channel, the assigned output sends CV between 0 and 5 volts proportional to the velocity of the Note On message. When the corresponding Note Off message is received, the assigned output goes to 0V.
- **Mod:** When a continuous controller message is received for CC#1 (modulation wheel) on the assigned MIDI channel, the assigned output sends CV between 0 and 5 volts proportional to the CC value.
- **Aft: **When aftertouch is received on the assigned MIDI channel, the assigned output sends CV between 0 and 5 volts proportional to the aftertouch value.
- **Bend:** When pitch bend is received on the assigned MIDI channel, the assigned output sends CV between about -3V and 3V proportional to the pitch bend value.
- **Expr, Pan, Brth: **Similar to Mod, but with different controller numbers (Expression: #11, Pan: #10, Breath: #2). These output values are maintained until the controller changes.
- **Hold: **When a hold pedal controller message is received on the assigned MIDI channel, the assigned output sends 5V until the pedal is released.
- **yAxis:** Similar to Mod, with with controller number #74. This is used as the Y-Axis controller for three-directional controllers that support MPE.
- **Clock:** There are four clock settings, which send triggers to the assigned CV output at various divisions of an incoming MIDI beat clock. The settings are: Qtr (which triggers every 24 clocks), 8th (12 clocks), 16th (6 clocks), and 24ppq (sends a trigger with every clock). Note that clock is a real-time message, and doesn't require a MIDI channel to be assigned. A clock indicator on the parameter line moves every eighth note.

CV-to-MIDI (MIDI Out) Types

Note: When Note or Leg. (Legato) is assigned to an input, the digital input and the CV input are used at the same time. When the digital input goes high, the voltage at the CV input is quantized into a MIDI note number, and a Note On message is sent on the specified channel. When the digital input goes low, a Note Off message is sent.

- **Leg. (Legato):** Works like Note, except that Captain MIDI watches the CV input for pitch changes. If the pitch changes by a semitone or more, a Note Off is sent (regardless of whether the gate has gone low) and a new Note On is sent for the new pitch. This is a more natural setting for humans playing CV controllers (like Pressure Points, Tetrapad, Keystep) because it doesn't require a player to completely disengage the controller before playing another note. The Note setting is more appropriate for sequenced melodies.
- **Veloc:** Usually, a Note On message uses a default velocity of 100. However, if Veloc is assigned to the same channel as a Note or Leg. assignment, the Note On velocity will be proportional to the voltage at the assigned input. Note that if two Veloc assignments are made on the same channel, only the lowest-numbered assignment will be used for Note On velocity.
- **Mod:** A change in positive voltage at the assigned input will cause a MIDI controller change for CC#1 (modulation wheel) to be sent on the specified channel.

- **Aft:** A change in positive voltage at the assigned input will cause a MIDI channel aftertouch message to be sent on the specified channel.
- **Bend:** A change in bi-polar voltage at the assigned input will cause a MIDI pitch bend message to be sent on the specified channel. Positive voltage is positive bend, and negative voltage is negative bend. The range is about -3V to +3V.
- **Hold: **Voltage over around 2.5 volts at the assigned input will cause a MIDI Hold "on" (127) controller message to be sent. When the assigned input goes to 0 volts, a MIDI Hold "off" (0) message is sent. Expr, Pan, Brth, yAxis: Similar to Mod, but with different controller numbers (Expression: #11, Pan: #10, Breath: #2, Y-Axis: #74).

MIDI Channel Setup Screen

This screen sets the MIDI channel for each assignment. For the MIDI-to-CV ("MIDI > ?") assignments, this determines which channel Captain MIDI is listening on. For the CV-to-MIDI ("? > MIDI") assignments, this determines which channel Captain MIDI is transmitting on.

When the channel is set to "Off" the assignment is effectively muted; no MIDI data will be received or sent.

Transpose Setup Screen

The transpose screen sets transpose values over a four-octave (-24 to +24 semitone) range. The transpose value is added to outgoing voltage for MIDI-to-CV assignments, and is added to MIDI note numbers for CV-to-MIDI assignments.

Transpose is a note-specific screen, and only assignments set to Note or Legato, and have a MIDI channel set, will be shown.

Note Range Low/High Setup Screens

These screens set the low and high values transmitted or recognized by the assignment, between the lowest MIDI note (C -1) and the highest (G9). Values outside the range will be ignored. Range checking is applied after transposition. So if a Note On is out of range, but transposition takes it into range, the note will be played; if a Note On is in range, but transposition would take it out of range, the note will be ignored.

Range screens are note-specific screens, and only assignments set to Note or Legato, and have a MIDI channel set, will be shown.

Setups

Captain MIDI has four independent Setups. A Setup consists of information for assignments, channels, transposition, and note range for all eight inputs and outputs. Use the UP and DOWN buttons to change the active Setup. The active Setup number is shown at the top of all of the Setup screens.

Copying Setups

To copy the information for the active Setup to another Setup, long-press the DOWN button. When you release the button, a Copy screen will open. Choose the copy's destination with the UP and DOWN buttons, and then push the RIGHT encoder ("[Copy]") to execute the copy. To leave the Copy screen without changing any data, push the LEFT encoder ("[Cancel]") or long-press the DOWN button again.

If you select a Setup as its own copy destination, the display will change to a SysEx dump screen. The RIGHT encoder option becomes "[Dump]". If you push the RIGHT encoder, Captain MIDI will send a system exclusive file containing the data for the active Setup.

Saving Setups

There are two ways to save Setup data.

Save all four Setups in the module: Long-press the LEFT encoder to get to the main menu, then long-press the LEFT encoder

again to save data for all apps. A bar indicator will confirm the save operation. The Ornament and Crime module will recall the saved Setups on power-up.

Save using MIDI system exclusive: Follow the SysEx dump procedure from the Copying Setups section above, and capture the system exclusive dump with SysEx librarian software. To restore a Setup from a SysEx dump, simply send the SysEx file back to the module whenever Captain MIDI is running.

Shortcut: When you long-press the RIGHT encoder to go to the main menu, the current Setup's data also is dumped via SysEx.

Notes about System Exclusive:

- A Setup from a received SysEx dump will always be placed in the active Setup, regardless of the Setup's original location.
- You can use system exclusive to change Captain MIDI's handling of MIDI data in an automated way during a performance.
- Data received from a SysEx dump will overwrite Setups in memory, but not data saved via the first save method; that is, unless you long-press to save the new Setup data, Captain MIDI will return to its previously-saved state at the next power-up.

Log Display

Captain MIDI logs the most recent 100 MIDI messages, in and out. Push the LEFT encoder button to see the MIDI log. When the log is being displayed, use the LEFT encoder to scroll through the events. Push the LEFT encoder button again to return to the Setup screens.

Captain MIDI can be used to identify the applications associated with Hemisphere Suite system exclusive data. Open the log and send the system exclusive file to Captain MIDI. If it's a Hemisphere Suite file, Captain MIDI will identify it. Note that SysEx messages are logged only while you are viewing the Log Display.

Panic! Function

Long-press the LEFT encoder if notes get hung. Captain MIDI will send Note Off messages on all 16 MIDI channels for all MIDI notes.

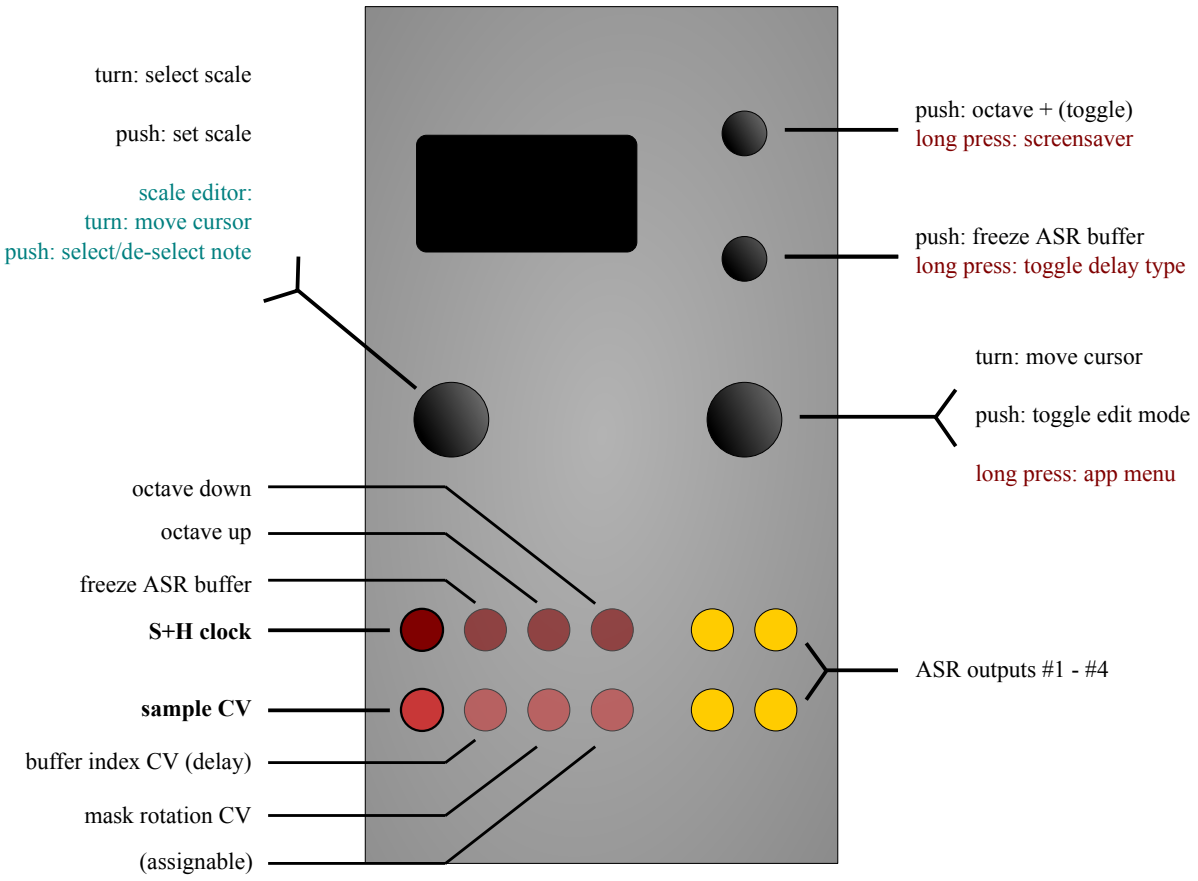
Credits

CopierMaschine

This app works much the same as the original o_C / quantising **ASR** firmware, except that it now incorporates some of the new/improved quantiser features, including a larger selection of **(editable) preset scales** as well as advanced user-scale-edit and (non-1V/oct) tuning options (for details, see the Quantermain “app” below and the info re: custom scales/tunings above).

In essence, then, ASR mode works as a cascaded, four-stage sample-and-hold module (see here (http://www.cyndustries.com/synapse/synapse.cfm?pc=35&folder=sept1976&pic=19) for an exposition on the classic ASR implementation):

- Feed a pulse (clock, trigger) into the left-most digital input (**TR1**), and a CV signal (LFO, ADSR, etc) into the leftmost CV input (**CV1**): on receiving a/the clock, the DAC outputs will be updated, **ASR-style**: the sampled value will be present at output A, the previous sample values shifted down the remaining outputs B, C, and D.



- The ASR mode features additional parameters, including
    - a **delay** (= controlled via the `buf. index` parameter (CV2))
    - **scale ‘mask’** rotation (CV3)
    - **hold** (which “freezes” the sample buffer) (TR2, down button)
    - CV over **transposition**, **scale mask rotation**, **root**, **buffer size** (assignable via CV4)
- Please also see the discussion of the `Trigger delay` menu selection in the *Quantermain* app documentation below — the same considerations apply to the `trigger delay` setting in *CopierMaschine*.

buffer index (delay)

- The `index` parameter works as a delay, sort of: internally, the ASR is a ring-buffer (buffer size = 256), and (to simplify things) by default outputs the sampled values S[x] stored at the buffer locations **index * output-stage**, ie A = S[i * 1], B = S[i * 2], C = S[i * 3], and D = S[i * 4].
- The default index setting (`buf. index`) is 0 (internally i = 1), in which case things boil down to standard ASR behaviour:

  A = S[1], **B = S[2]**, **C = S[3]**, and **D = S[4]**
- If the index parameter was instead set to, say, i = 8, the ASR in that case would output the values stored in the buffer at locations S[8], S[16], S[24], and S[32], thus delaying output A by 8 clocks, B by 16 clocks, and so on. Thus, modulating the `index` parameter doesn't just delay the output on channels B to D, but also allows different patterns to be created (based on the contents of the buffer).
- **Alternatively**, you can change the way the delay behaves by long-pressing the down-button once. (Two little dots will appear next to the clock indicator at the top of the menu, on the right). In this case, the value of `buf. index` will simply be **added** to the buffer locations, ie A = S[1 + i], B = S[2 + i], C = S[3 + i], and D = S[4 + i]. The resulting behaviour is that of a regular delay line. Another long-press on the down button will toggle back to the first `index` mode.

hold ('freeze')

- Pressing the **down button** will toggle 'freeze' mode. Freeze mode is also activated while the TR2 (= hold) input is held **high** (using a gate or the like). In freeze mode, **no further samples will be acquired**; when clocked, the four outputs then simply cycle through what's already the buffer. If 'freeze' is engaged, two little dots will appear next to the scale name, at the top of the menu, indicating the buffer is currently frozen.
- The size of the hold buffer is determined by the `hold (buflen)` parameter, which goes from 4 to 63. While the buffer is 'frozen', all the various modulation options (buffer length, transposition, scale change, etc) are still available / operate on the frozen buffer contents.
- 'Freeze' mode effectively turns *CopierMaschine* into a clocked, quantised CV recorder, with four output taps and variable playback parameters.

Inputs and outputs

I/O	Function
TR1	Clock input
TR2	Hold (= freeze ring buffer)
TR3	Transpose: Octave up, when high
TR4	Transpose: Octave down, when high (overridden by TR3)
CV1	Sample in
CV2	Index: ring buffer index (= "delay")
CV3	Mask: rotate scale mask
CV4	assignable: octave, root, transpose (by scale-degrees), buffer-length, or CV input scaling

A, B, C, DASR outputs 1-4

Controls

Control	Function
Left encoder (turn)	Select scale in main menu; move cursor in scale edit menu
Left encoder (press)	Activate scale in main menu; add/remove note in scale edit menu

Control	Function
Right encoder (turn)	Navigation mode: move up and down through the menu items. Edit mode: increase or decrease the value being edited; move scale “mask” in scale edit menu
Right encoder (press)	Toggle between menu navigation (selection) mode and value editing mode
Right encoder (long press)	App selection menu
Up button	toggle octave up
Up button (long press)	screensaver shortcut
Down button	freeze ASR input buffer
Down button (long press)	toggle delay type: multiplying / basic

Available settings

Setting	Meaning
`scale`	Current scale (active scale labelled with a little dot)
`root`	Root note for scale
`mask`	“scale mask” / active note pattern in the selected scale
`octave`	octave
`buf. index`	ring buffer index (= “delay”) amount
`hold (buflen)`	length of buffer (when freezing the ASR): `4 - 63`
`trigger delay`	sets the TR1-input-to-processing latency (for details see *Quantermain* below)
`input gain`	CV “gain”, multiplies incoming CV1 sample value by selected value (range: 0.05 - 2.00 in steps of 0.05). Note that this setting also affects the range of the ‘internal’ CV sources (LFSR, bytebeats and integer sequences, see below).
`CV4 dest. -->`	parameter assigned to CV4 input
`CV source`	sets the source of the sample voltage, either an external voltage (CV1), or an ‘internal’ source (details see below)

Scale edit:

see here (user-scales are shared across apps).

Screensaver display

Four little “Arabesque” patterns, representing the pitch CV output on each of the four channels.

CV sources

Four settings of the `CV source` parameter are available:

- `CV1` — voltages on the CV1 input are quantised, according to the `scale` and `Active notes` settings.
- `LFSR` — “Linear Feedback Shift register” (also referred to as a “Turing Machine” or “TM” elsewhere in this documentation). This is the same as the `Turing` source in the *Quantermain* app - please see the discussion of the Turing Machine source in the *Quantermain* section for further details of

operation. The available settings for the Turing Machine source in *CopierMaschine* are shown in the table below.

- `ByteB` — “bytebeat” equations used to generate semi-fractal note values, rather than audio signals (which is what byte beat equations are usually used for). Please see the *Viznutcracker, sweet!* app for details of the byte beat equations available.
- `IntSq` — integer sequences - several classes of random and fractal integer sequences, used as note values.

LFSR source settings in *CopierMaschine*

Note: `LFSR` is used equivalently to “Turing Machine” here. (see *Quantermain* for details).

Setting	Meaning
`LFSR length`	Length of the linear feedback shift register, in bits, range 4 to 32
`LFSR p`	Probability that the least significant bit will be flipped when it to copied, range 0 to 255 (0 means *p*= 0, 255 means *p*= 1)
`LFSR CV1`	The Turing Machine parameter to which any voltage input on CV1 will be directed. Choices are `rng`, `len` and `p` (ie, range, length, and probability)

Note that the `LFSR range` setting in previous versions has been subsumed by the `input gain` setting, which now also affects the ‘internal’ CV sources such as LFSR, bytebeats and integer sequences.

Byte beats source settings in *CopierMaschine*

Setting	Meaning
`BB eqn`	sets the byte beat equation used as the source. See the *Viznutcracker, sweet!* app documentation for more details of the currently available equations.
`BB P0`	Parameter 0 for the byte beat equation - see See the *Viznutcracker, sweet!* app documentation for more details.
`BB P1`	Parameter 1 for the byte beat equation - see See the *Viznutcracker, sweet!* app documentation for more details.
`BB P2`	Parameter 2 for the byte beat equation - see See the *Viznutcracker, sweet!* app documentation for more details.
`BB CV1`	The byte beat parameter to which the input on CV1 is directed. Possible destinations are “igain” (input gain), “eqn” (equation), “P0”, “P1”, “P2”. See `LFSR CV1` above for details of input voltage ranges.

Note that compared to the bytebeat source in *Quantermain*, the `Bytebeat range` parameter is missing from *CopierMaschine*. The reason is that the `input gain` setting has the same effect in *CopierMaschine* when the `ByteB` source is used as the `Bytebeat range` setting does in *Quantermain*.

Integer sequence source settings in *CopierMaschine*

Setting	Meaning
`IntSeq`	sets the integer sequence used as the source. See below for a list of available integer sequences and their characteristics.
`IntSeq modul`	sets the modulus for the integer sequence. The value of the integer from the integer sequence is divided by the modulus and the remainder is used. For example, if the modulus is 8 and the current integer value from the sequence is 19, then the remainder of 19 - (2 x 8) i.e. 3 is used as the value. In other words, values “wrap around” at the modulus setting — it sets a maximum note range for the integer sequence, similarly to the `M/A` setting, but `M/A` compresses of expands the range of notes for a given integer value from the sequence, whereas the modulus wraps the values around.

Setting	Meaning
`IntSeq start`	sets the start point in the stored integer sequence. The stored sequences are 128 steps long, and the maximum start point is 126 to ensure a minimum sequence length of 2.
`IntSeq len`	sets the length of the integer sequence. Thus a length of 16 will use just 16 values from the stored 128 step sequence, starting at the step specified by `IntSeq start`.
`IntSeq dir`	sets whether the integer sequence loops back to the beginning when it gets to the end, or whether it swings back like a pendulum and plays in reverse when it reaches the end. The “end” is the last step in the sequence, as defined by the sequence start plus the sequence length settings.
`Fractal stride`	Several of the sequences are fractal or semi-fractal in nature (i.e. they are self-similar), and the “stride” setting sets how many steps are advanced on each trigger input. This also works well with the non-fractal sequences and provides additional variation, particularly if the stride is not an exact divisor of the sequence length.
`IntSeq CV1`	The integer sequence parameter to which the input on CV1 is directed. Possible destinations are (parameter names in brackets): `M/A` (mult/att), `seq` (IntSeq), `strt` (IntSeq start), `len` (IntSeq len), `strd` (Fractal stride) and `mod` (IntSeq modul). See `LFSR CV1` above for details of input voltage ranges.

Integer sequences available in *CopierMaschine* and *Quantermain*

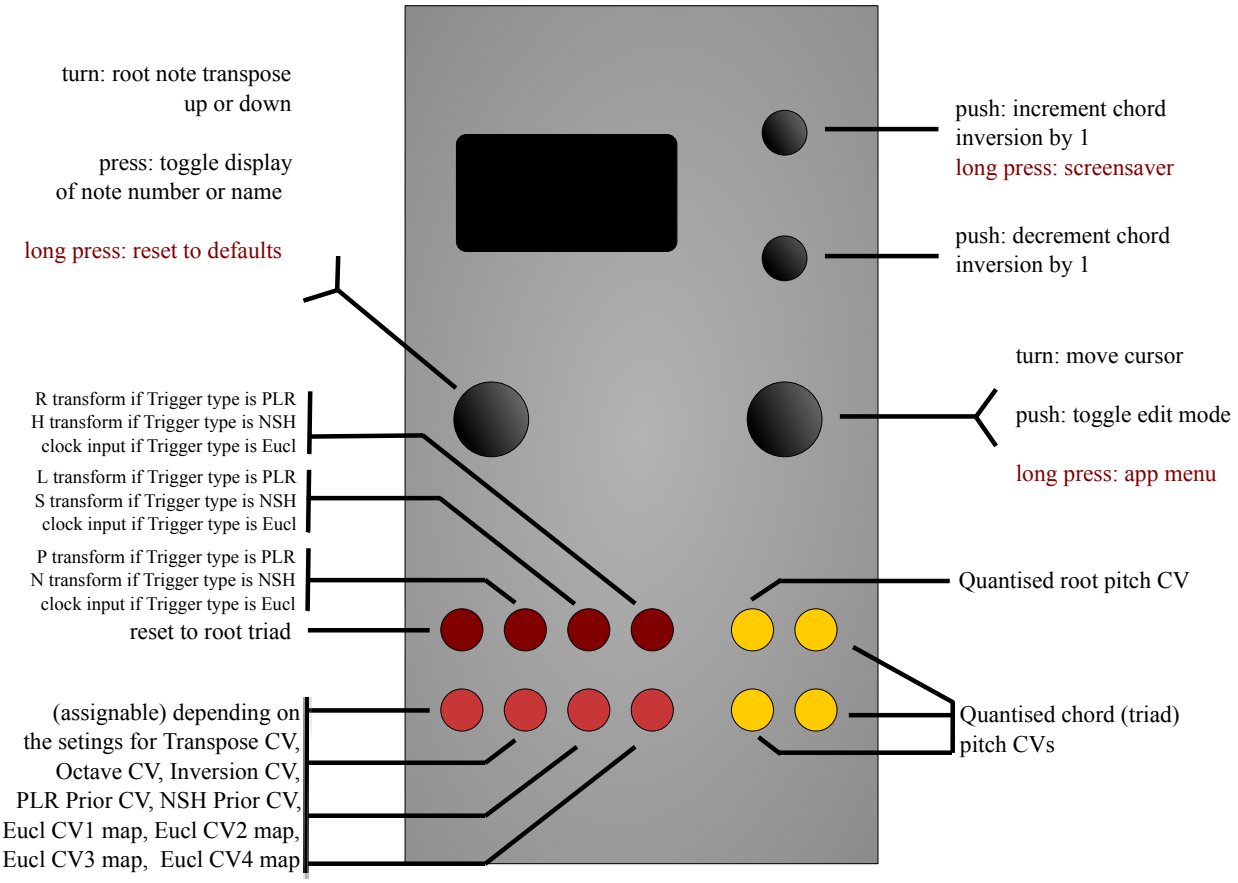
Menu name	Description
`pi`	The first 128 digits of π (https://en.wikipedia.org/wiki/Pi)
`vnEck`	The first 128 integers in van Eck’s sequence (https://oeis.org/A181391).
`ssdn`	The first 128 integers in the sequence of the sum of squares of the digits of *n* (https://oeis.org/A003132).
`Dress`	Dress’s sequence (https://oeis.org/A001316).
`PNinf`	Per Nørgård’s infinity series (http://www.pernoergaard.dk/eng/strukturer/uendelig/ukonstruktion03.html)
`Dsum`	Digital sum (i.e. sum of digits) of *n*; also called digsum(n) (https://oeis.org/A007953).
`Dsum4`	Digital sum (i.e. sum of digits) of *n* written in base 4 (https://oeis.org/A053737).
`Dsum5`	Digital sum (i.e. sum of digits) of *n* written in base 5 (https://oeis.org/A053824).
`CDn2`	Fractal sequence: count down by 2’s from successive integers (https://oeis.org/A122196).
`Frcti`	Fractal sequence of the interspersion A163253 (https://oeis.org/A163256).

See the acknowledgements section for additional references for some of these integer sequences.

Note that there isn’t really anything magical or mystical about the digits of transcendental numbers such as π — they are just a convenient source of sequences of digits. However, if you use them to make music, then you can post videos like this (https://www.youtube.com/watch?v=YOQb_mtkEEE), or this (https://www.youtube.com/watch?v=OMq9he-5HUU). The fractal and semi-fractal sequences can produce some magical melodies, however.

Harrington 1200

This is a relatively straight-forward implementation of neo-Riemannian transformations for generating triad (three note chord) progressions (see the acknowledgements (http://ornament-and-cri.me/acknowledgements/) section for more details on neo-Riemannian music theory).



- In settings mode, the top line of the display shows, from left to right:
  - the current **root note** for the root (initial) chord
  - the **musicological mode** (major or minor) for the root chord
  - the three notes comprising the **current triad** being output. Clicking on the left encoder toggles between note display and display of semitone offset from the root note.
- The root note can be changed using the `Transpose` setting in the menu. The left encoder can also be used to change this setting at any time (such as during live performance). The voltage input on CV1 also changes the root note (i.e. the transposition).
- The musicological mode of the root chord is set by `Root mode` in the menu to either major or minor. Chord inversion is similarly set using the `Inversion` menu item. Note that these settings are not immediate - they will take effect when the next transformation trigger input is received. The top line of the display indicates the current triad, not the current menu settings for these parameters. Note that the voltage input on CV4 also changes the inversion.
- The pitch voltage (scaled to 1V/octave) for the root note appears on output A. The pitch voltages for the three notes of the current triad appear on outputs B, C and D. Thus, to pproduce chords, you should feed the B, C and D outputs into the 1V/octave pitch inputs of three VCOs.
- Trigger inputs TR2, TR3 and TR4 are used to apply the atomic P, L or R transformations, or the compound N, S or H transformations (see below), depending on the `Trigger type` setting. Trigger input TR1 resets the current triad back to the root chord for all settings of `Trigger type1` .

- If multiple triggers are received, the reset input (TR1) always has priority, then all triggered transforms are applied. The order in which they are applied can be set in the menu by the `PLR Priority` and `NHS Priority` settings.
- The neo-Riemannian transformations themselves are quite simple, and "reversible" i.e. applying them twice returns the original triad. The following basic (atomic) tranformations are provided:
    - P (Parallel): Moves the third up or down a semitone, thus P(Cmaj) = Cmin, P(Cmin) = Cmaj.
    - L (Leittonwechsel): Converts a major triad to a minor by shifting the root down a semitone and making the third the root, or from minor to major by moving the fifth up a semitone to become the root.
    - R (Relativ): Converts a major triad to its relative minor, or a minor triad to its relative major.
- Alternatively, secondary transformations can be used:
    - N (Nebenverwandt): Exchanges a major triad for its minor subdominant, and a minor triad for its major dominant (e.g. C major and F minor). The "N" transformation is the same as applying R, L, and P successively.
    - S (Slide): Exchanges two triads that share a third (e.g. C major and C♯ minor); it is the same as applying L, P, and R successively, in that order.
    - H (Hexatonic): Exchanges a triad for its hexatonic pole (e.g. C major and A♭ minor); it is the same as applying L, P and L transformations successively.
- The implementation computes these N, S and H transformations in a single step however, not sequentially.
- Note that in trigger modes `PLR` and `NSH`, only PLR or NSH transformations can be applied, but not mixtures of both. However, in `Eucl` (Euclidean) trigger mode (see below), mixtures of both PLR atomic transformations and NSH secondary or compound transformations can be applied.
- Internally, the triad is stored in a neutral form (basically just offsets), thus the chord voicing is preserved and can be shifted easily to the quantised root note, and inversions created on-the-fly. For an alternate way of implementing these transformations, see the documentation of the Tonnetz Sequent (http://www.noiseengineering.us/tonnetz-sequent/).

Screensaver display

The current triad (output as pitch CVs on the B, C and D sockets) is shown graphically on the left. The circumference of the circle has 12 points on it, one point for each semitone in an octave, with C at the 12 o'clock position. The current triad is indicated by the three dots, joined to form a triangle. The middle section of the screensaver display shows the last four transformations applied, with the most recent one at the top. The rightmost section indicated the current triad being output, with numbers showing the octave for each note.

Tip: for a pretty display, patch the output of an LFO into CV1 in order to rapidly rotate the root note up and down.

Settings

Setting	Meaning
`Transpose`	Shift root/triad in semitones, range -24 to 24 (i.e. two octaves up or down)
`Transpose CV`	Selects CV input used to shift root/triad in semitones, up (positive voltages up to about 6.5V) or down (negative voltages down to about -3.5V). Transposition should be scaled at 1V/octave. Choices are `None`, `CV1`, `CV2`, `CV3`, `CV4`.
`Octave`	Shift root/triad in octaves, range -3 to 3.
`Octave CV`	Selects CV input used to shift root/triad in semitones, up (positive voltages up to about 6.5V) or down (negative voltages down to about -3.5V). Scaling is **not** 1V/octave. Choices are `None`, `CV1`, `CV2`, `CV3`, `CV4`.

Setting	Meaning
`Root mode`	Mode of root triad, either `maj` or `min`
`Inversion`	Chord inversion, range is from -3 to 3.
`Inversion CV`	Selects CV input used to shift the chord inversion, up (positive voltages up to about 6.5V) or down (negative voltages down to about -3.5V). Choices are `None`, `CV1`, `CV2`, `CV3`, `CV4`.
`PLR Priority`	Order in which the P, L and R transforms are applied if multiple triggers on TR2, TR3 and TR4 are received simultaneously
`PLR Prior CV`	Selects CV input used to shift the PLR priority amongst the available choices, up (positive voltages up to about 6.5V) or down (negative voltages down to about -3.5V). Choices are `None`, `CV1`, `CV2`, `CV3`, `CV4`.
`NSH Priority`	Order in which the N, S and H transforms are applied if multiple triggers on TR2, TR3 and TR4 are received simultaneously
`NSH Prior CV`	Selects CV input used to shift the NSH priority amongst the available choices, up (positive voltages up to about 6.5V) or down (negative voltages down to about -3.5V). Choices are `None`, `CV1`, `CV2`, `CV3`, `CV4`.
`CV sampling`	Selects how sampling of CV1, CV2, CV3 and CV4 inputs is done: either continuously (`Cont`) or on a sample-and-hold basis (`Trig`) which is triggered by a trigger on any or all of the four trigger inputs.
`Output mode`	Output mode, with the default being `chord`, or use `tune` to output the quantised root on all four output channels
`Trigger type`	Sets the trigger behaviour, either P. L and R transforms triggered by trigger inputs on TR2, TR3 and TR4 respectively, or N, S or H transforms, also triggered bt TR2, TR3 or TR4 respectively, or `Eucl`, which enable Euclidean trigger masks - see below for details.
`Trigger delay`	sets the TR1-input-to-processing latency (for details see the setting with the same name in *Quantermain* below)

Controls

Control	Function
Left encoder (turn)	Root note transpose up or down
Left encoder (press)	Toggle display of note numbers (semitone offsets from the root note) or names (note: the names are the simplest possible mapping, thus there are no enharmonic substitutions)
Left encoder (long press)	Reset to defaults
Right encoder (turn)	Navigation mode: move up and down through the menu items. Edit mode: increase or decrease the value being edited.
Right encoder (press)	Toggle between menu navigation (selection) mode and value editing mode
Right encoder (long press)	App selection menu
Up button (long press)	Invoke screensaver display
Up button (press)	Increment chord inversion by 1
Down button (press)	Decrement chord inversion by 1

Inputs and outputs

I/O	Function
TR1	Reset to root triad

I/O	Function
TR2	P transform (if `Trigger type` is `PLR`), or N transform (if `Trigger type` is `NSH`), or clock input (if `Trigger type` is `Eucl`)
TR3	L transform (if `Trigger type` is `PLR`), or S transform (if `Trigger type` is `NSH`), or clock input (if `Trigger type` is `Eucl`)
TR4	R transform (if `Trigger type` is `PLR`), or H transform (if `Trigger type` is `NSH`), or clock input (if `Trigger type` is `Eucl`)
CV1	Mapped according to `Transpose CV`, `Octave CV`, `Inversion CV`, `PLR Prior CV` and `NSH Prior CV` settings (see table above), as well as according to the `Eucl CV1 map`, `Eucl CV2 map`, `Eucl CV3 map` and `Eucl CV4 map` settings (see below) - each CV input can be used for multiple internal destinations.
CV2	Ditto
CV3	Ditto
CV4	Ditto
A	Quantised root
B, C, D	Transformed & inverted triad (also quantised)

Euclidean trigger masks

Euclidean trigger mask mode in the *Harrington 1200* app is enabled by setting the `Trigger type` setting to `Eucl`. Twenty-two additional settings will appear in the menu when the `Eucl` trigger type is selected. In this mode, a simple, single, regular clock input can be used (into TR2, TR3 or TR4) to trigger P, L, R, N, S or H chord transformations. Each of the six transformation types has its own “Euclidean mask” or “Euclidean filter” — by varying the parameters on each of these for each transformation type, very complex “polyrhythms” of chord transformations can be derived from as single regular clock input. Furthermore, all of the Euclidean trigger mask parameters, for each transformation type, can be placed under external voltage control. This permits external voltages to influence complex, evolving patterns of chord transformations driven by a single external clock.

Each type of transformation (P, L, R, N, S and H) has three settings controlling the Euclidean trigger mask (Euclidean filter) for it (thus 18 controls all together): the length of the Euclidean pattern (named `X EuLeng`, where X is the type of transformation, one of P, L, R, N, S or H), the fill amount (`X EuFill`) for that length of pattern, and the offset or rotation (`X EuOffs`) of the pattern. These all operate in the same way as the Euclidean trigger filters for the envelope generators in the *Piqued* app operate - see the discussion in the *Piqued* section for further details.

The settings `Eucl CV1 map` through to `Eucl CV4 map` permit the four external voltage inputs to be mapped to four of the 18 available Euclidean trigger mask settings. The same input voltages can also be used to simultaneous control transposition, octave, inversion and trigger priorities (see above), if desired.

Tips

Try using a rhythm generator, such as the Mutable Instruments Grids (http://mutable-instruments.net/modules/grids) module, or the ALM/Busy Circuits Pamela’s Workout (http://busycircuits.com/alm001/) module, or the Rebel Technology Stoicheia (http://www.rebeltech.org/products/stoicheia/) module, or some combination of clock division and logic modules, or even a second o_C module running the *Piqued* app with Euclidean trigger filters engaged, to trigger the P, L and R transformations (via trigger inputs TR2, TR3 and TR4) in varying patterns to create a variety of chord progressions that may or may not be musically pleasing, or interesting, or horrifying.

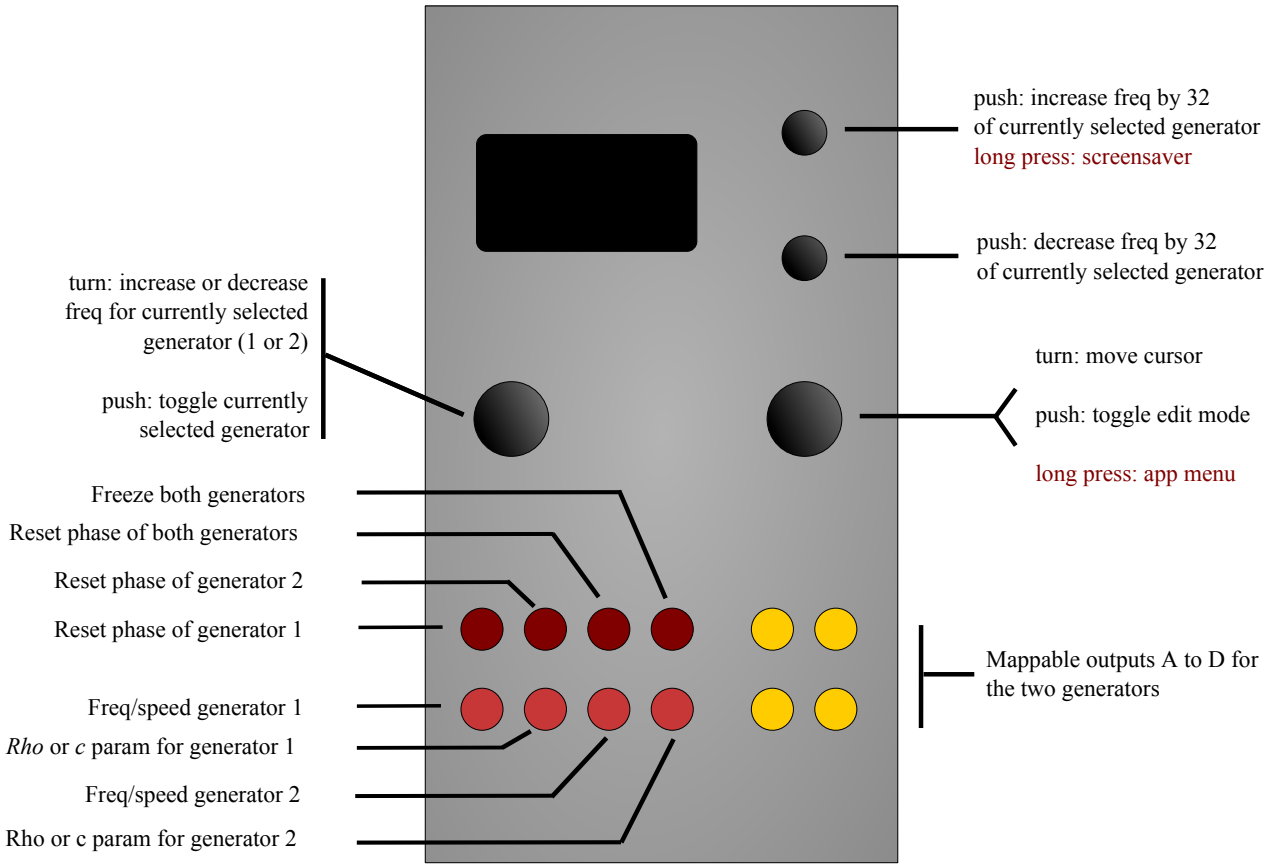
Update: in v1.2, you can use the built-in Euclidean trigger masks to achieve the same thing, and more, with just a single external regular clock input (into TR2, TR3 or TR4).

Low-rents

Low-rents is a port of the Lorenz attractor (https://en.wikipedia.org/wiki/Lorenz_system) modulation generator from the “Easter egg” in the Mutable Instruments Streams (http://mutable-instruments.net /modules/streams) module, to which Rössler attractors (https://en.wikipedia.org/wiki/Rössler_attractor) have been added.

Two independent function generators are provided (referred to here as Generator 1 and Generator 2), with each generator calculating *both* the Lorenz and Rössler functions simultaneously, using the same phase accumulator, but with the rate/speed of each generator independently settable. Both the Lorenz and the Rössler functions output three values (*x*, *y* & *z*), and various combinations of these can be mapped to the four output channels. The chaotic strange attractors work best as slow modulation functions.

Note that the output voltage range of the o_C module is asymmetrical (about -3V to +6V) — because it was designed to process pitch CVs. Therefore the output of the *Low-rents* app is not centred about 0V.



I/O	Function
TR1	Reset phase of generator 1
TR2	Reset phase of generator 2
TR3	Rest phase of both generators
TR4	Freeze (both generators stop in their tracks and hold current value while TR4 is high)
CV1	Frequency/speed of generator 1
CV2	Rho or c parameter for generator 1
CV3	Frequency/speed of generator 2
CV4	Rho or c parameter for generator 2

A, B, C, DMappable outputs from the two generators (see table below)

Controls

Control	Function
Left encoder (turn)	Increase or decrease frequency of currently selected generator (1 or 2)
Left encoder (press)	Toggle currently selected generator frequency control
Right encoder (turn)	Navigation mode: move up and down through the menu items. Edit mode: increase or decrease the value being edited.
Right encoder (press)	Toggle between menu navigation (selection) mode and value editing mode
Right encoder (long press)	App selection menu
Up button (long press)	invoke screensaver
Up button (press)	Increment frequency of currently selected generator by 32
Down button (press)	Decrement frequency of currently selected generator by 32

Settings

Setting	Meaning
`Freq 1`	Frequency/speed of generator 1, range is 0-255
`Freq 2`	Frequency/speed of generator 2, range is 0-255
`Rho/c 1`	Rho (for Lorenz attractor) or c (for the Rössler attractor) parameters for generator 1
`Rho/c 2`	Rho (for Lorenz attractor) or c (for the Rössler attractor) parameters for generator 2
`LFreq 1 Rng`	frequency/speed range of Lorenz and Rössler generator 1, ranges from ‘sloth’ to ‘fast’.
`LFreq 2 Rng`	frequency/speed range of Lorenz and Rössler generator 2, ranges from ‘sloth’ to ‘fast’.
`out A`	output mapping for output A. Available choices shown in the table below.
`out B`	output mapping for output B. Available choices shown in the table below.
`out C`	output mapping for output C. Available choices shown in the table below.
`out D`	output mapping for output D. Available choices shown in the table below.

Output mapping value	Meaning
`Lx1`	Generator 1 Lorenz attractor *x* value
`Ly1`	Generator 1 Lorenz attractor *y* value
`Lz1`	Generator 1 Lorenz attractor *z* value
`Lx2`	Generator 2 Lorenz attractor *x* value
`Ly2`	Generator 2 Lorenz attractor *y* value
`Lz2`	Generator 2 Lorenz attractor *z* value
`Rx1`	Generator 1 Rössler attractor *x* value
`Ry1`	Generator 1 Rössler attractor *y* value
`Rz1`	Generator 1 Rössler attractor *z* value
`Rx2`	Generator 2 Rössler attractor *x* value
`Ry2`	Generator 2 Rössler attractor *y* value
`Rz2`	Generator 2 Rössler attractor *z* value
`Lx1+Rx1`	Sum of Generator 1 Lorenz attractor *x* value and Generator 1 Rössler attractor *x* value
`Lx1+Rz1`	Sum of Generator 1 Lorenz attractor *x* value and Generator 1 Rössler attractor *z* value
`Lx1+Ly2`	Sum of Generator 1 Lorenz attractor *x* value and Generator 2 Lorenz attractor *y* value
`Lx1+Lz2`	Sum of Generator 1 Lorenz attractor *x* value and Generator 2 Lorenz attractor *z* value
`Lx1+Rx2`	Sum of Generator 1 Lorenz attractor *x* value and Generator 2 Rössler attractor *x* value

Output mapping value	Meaning
`Lx1+Rz2`	Sum of Generator 1 Lorenz attractor *x* value and Generator 2 Rössler attractor *z* value
`Lx1xLy1`	Bit-wise XOR of Generator 1 Lorenz attractor *x* value and Generator 1 Lorenz attractor *y* value
`Lx1xLx2`	Bit-wise XOR of Generator 1 Lorenz attractor *x* value and Generator 2 Lorenz attractor *x* value
`Lx1xRx1`	Bit-wise XOR of Generator 1 Lorenz attractor *x* value and Generator 1 Rössler attractor *x* value
`Lx1xRx2`	Bit-wise XOR of Generator 1 Lorenz attractor *x* value and Generator 2 Rössler attractor *x* value

The Rho and c parameters for the Lorenz and Rössler attractors respectively determine the degree of variability in the chaotic generator system. Note that the values have been constrained so that the functions do not collapse, but some combinations of extreme settings may cause the generator functions to collapse completely. If this happens, change the Rho/c setting and send a rest pulse to the relevant trigger input to reset the function generator.

Screensaver display

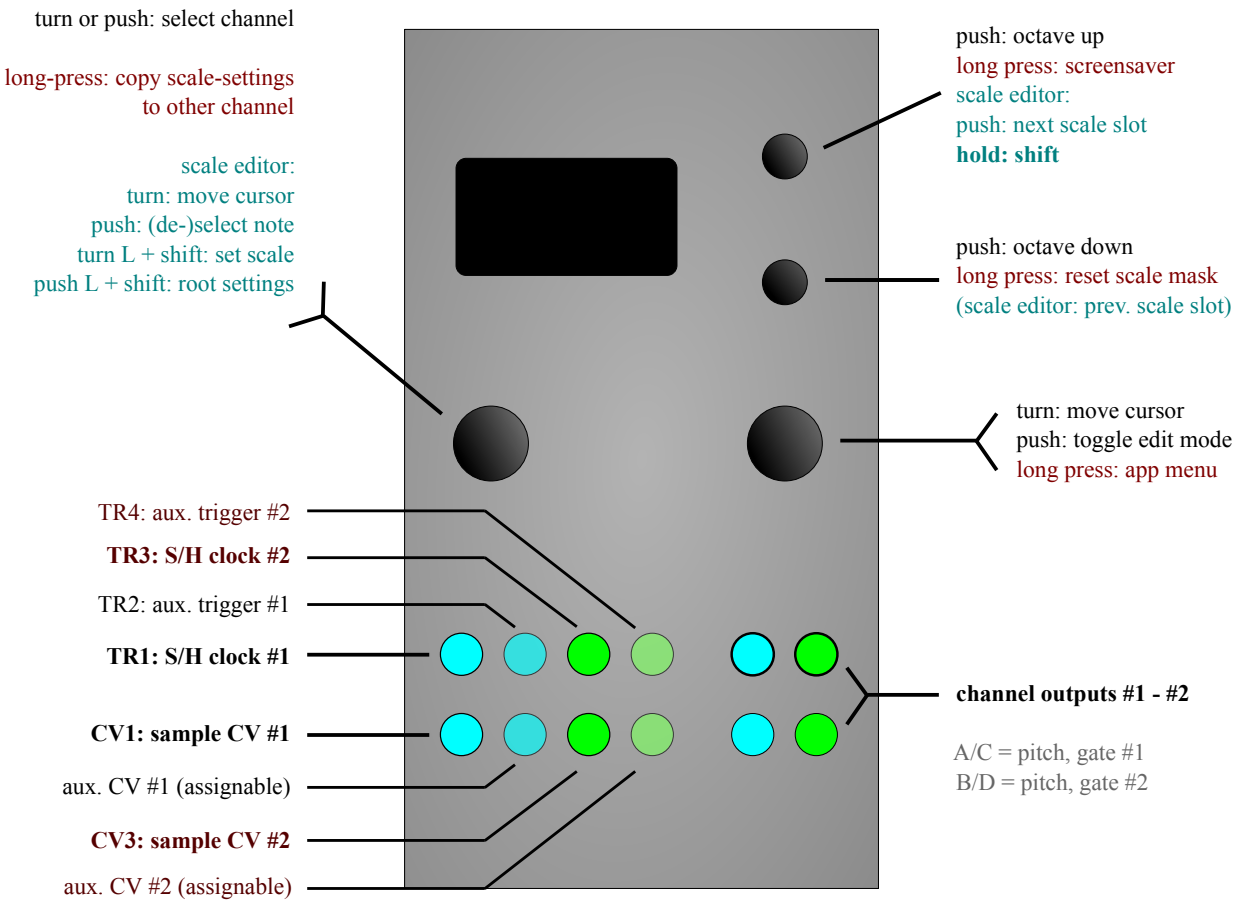
The screensaver show the A and B outputs in a vectorscope (X/Y) display on the left half of the screen, and the C and D outputs as a vectorscope display on the right half of the screen.

Tips

If you have an oscilloscope capable of displaying X/Y (vectorscope) signals, try patching pairs of the *x*, *y* and *z* outputs from either type of generator into it to observe the classic strange attractor patterns.

Meta-Q

Meta-Q is a dual-channel quantiser that's basically similar to *Quantermain*, except, of course, there are only two channels. It has fewer internal CV sources (currently, only LFSR. See *Copiermaschine*/*Quantermain* for details), but comes with four 'scale slots' per channel — a package of scale, mask, root and transpose values —, which themselves can be **sequenced** to provide a lot of additional, structured variation to the input CV sequences (or the LFSR internal CV source).



scale 'slots' / seq mode:

- Each scale 'slot' (= `scale #`) can be mapped to any of the available preset and user scales and/or scale masks (via the main menu or scale editor). The scale slots can also be assigned their own root and transpose settings.
- The scale editor in *Meta-Q* thus is slightly more complex than in the other modes: it allows you to edit the four scales-slots (not just one scale) from within the editor, ie both scale as well as the root and transpose values. In basic use, it works much the same as in *Quantermain* or *Copiermaschine*; notably, the `up` button behaves slighlty differently, however: in the *Meta-Q* scale editor it assumes a 'shift' functionality to access the (slot-specific) root and (diatonic) transpose settings (details see below).
- What is more, said four scale-slots can be sequenced (or just toggled) by using a clock signal and/or by modulating the slot-parameter with a control voltage:
    - Use the auxiliary trigger inputs **TR2** resp. **TR4** to cycle through either 2, 3, or 4 *adjacent* scale slots: see the `seq mode` setting. Available settings are **TR+1** (= plus next scale), **TR+2** (= plus next two scales), **TR+3** (= plus next three scales). Things wrap around slot #4. Thus, for example, when selecting `scale #` = 3 and `seq mode` = TR+2 (= scale #3 plus next two scales), when clocked via the aux. trigger input, the scale slots will step through the following sequence: #3, #4, #1, #3, #4, #1, #3, #4, #1, #3, #4, #1, #3 …

- Setting the `CV aux.` parameter to `scl#` (= scale #) allows modulation of the slot-parameter with a CV signal applied to CV2 (channel #1) or CV4 (channel #2). (Alternatively, **CV aux.** can be routed to root, mask (= scale mask), trns (= transpose/scale degrees), or oct (transpose/octaves)).
- **NB**: inputs TR2 and CV2 are hardcoded to service channel #1, TR4 and CV4 to service channel #2.

aux. outputs (C, D):

- when `trigger source` = TR1 - TR4, the **aux. outputs** (C, D) simply pass through the main trigger signal, with adjustable pulse-width (`--> pw`); when set to continuous (`cnt`), the aux. output goes high if/when the note **changes** (= trigger-on-note-change).
- the aux. outputs can alternatively output a transposed copy of the main channel CV (aux.output = `copy`), or output said CV, delayed by one clock (aux.output = `asr`).

Inputs and outputs

I/O	Function	-
TR1	clock input #1	-
TR2	aux. / scale sequencer clock #1	-
TR3	clock input #2	-
TR4	aux. / scale sequencer clock #2	-
CV1	sample in # 1	-
CV2	(mappable)	-
CV3	sample in # 2	-
CV4	(mappable)	-
A, B	CV outputs #1, #2	-
C, D	aux outputs #1, #2 (default to gate output)	-

Available settings (per-channel)

Setting	Meaning
`scale`	selected scale
`--> edit`	edit scale mask (for details see *Quantermain* and below)
`seq mode`	number of slots (-, 2, 3, or 4) to advance scale sequencer before reset (via `TR2` resp. `TR4`)
`scale #`	selected scale slot (`#1 - #4`)
`root #n`	root note for scale slot n
`transpose #n`	offset (in scale degrees) for scale slot n
`octave`	offset in octaves (for all scale slots)
`CV source`	sample input, internal/external (`CV1 - CV4` , `LFSR`)
`CV aux.`	auxiliary CV input destination: `scale #` , `root` , `transpose` , `octave` , `mask`
`trigger source`	main trigger source: `TR1 - TR4` , `cnt+` , `cnt-` (†)
`--> latency`	trigger-to-quantization latency (default: `0` = 60 us). Note: this is equivalent to the `Trigger delay` setting in several of the other apps, please see the *Quantermain* app documentation for a discussion of this setting.
`aux.output`	aux channel output: `gate` , `copy` , `asr` (††)
`--> pw`	pulse-width of triggers at C/D outputs (`gate` mode)
`--> aux +/-`	pitch offset at C/D outputs (in octaves) (`copy` and `asr` modes)
`> LFSR length`	length of shift register (when in LFSR mode)
`> LFSR p`	probability of flipping (when in LFSR mode)
`> LFSR range`	output range (when in LFSR mode)
`> LFSR CV`	destination of `CV1` resp. `CV3` (when in LFSR mode)

Setting	Meaning
`> LFSR TRIG`	aux channel output (when in LFSR mode): `echo` , `lsb` , `chng` : clock through, track LSB, and output on note-change

- notes:
  - (†) `cnt+` and `cnt-` = continuous quantization; in this case, a gate applied to `TR1` resp. `TR3` will shift the pitch up (`cnt+`) or down (`cnt-`) by one octave.
  - (††) `copy` simply duplicates the main channel output to the aux channel output. `asr` does the same, but delayed by one clock.

Controls:

main menu

Control	Function
Left encoder (turn)	select channel
Left encoder (press)	toggle channel
Left encoder (long press)	copy selected scale/mask to *all* channels/slots
Right encoder (turn)	Navigation mode: move up and down through the menu items. Edit mode: increase or decrease the value being edited
Right encoder (press)	Toggle between menu navigation (selection) mode and value editing mode
Right encoder (long press)	app selection menu
Up button	Transpose up: one octave (parameter menu)
Down button	Transpose down: one octave (parameter menu)
Up button (long press)	invoke screensaver
Down button (long press)	reset scale mask

scale editor

- works **mostly** the same (as in the other quantising modes). Notably, the `up` and `down` buttons behave slighlty differently, however: pushing `up` will point the editor to the next scale-slot, pushing `down` will point it to the previous scale-slot. This way, you can edit all the scale-slots more easily from within the editor.
- In the editor, you can also change the scale-type, root, and transpose settings: to do so, **hold** the `up` button (= 'shift') while turning the **left** encoder: that'll change the scale, just like in the regular menu. to edit the root and transpose values, **hold** the `up` button, then push the left encoder (= 'shift' + push left): that'll open a new window, displaying the slot's root and transpose values. you can select which parameter to edit by using the left encoder; adjust the value by turning the right encoder; or use the up / down buttons to advance to a different slot. push the left encoder again to return to the basic scale editor.

Control	Function
Left encoder (turn)	select note
Left encoder (push)	activate/de-active note
Right encoder (turn)	rotate mask
Right encoder (push)	exit editor
Up button (push)	**go to next scale slot**
Down button (push)	**go to previous scale slot**
Up button (hold) + Left encoder (turn)	**shift: select scale**
Up button (hold) + Left encoder (push)	**shift: toggle root/transpose view**
Right encoder (long press)	– (app selection menu)
Up button long press	– (screensaver)

Neural Net

Jump to bottom

Chysn edited this page on Mar 18, 2020 · 14 revisions

Neural Net is a highly-configurable logic processor with six *Neurons*, each of which can contain one of 11 different logic gates, including a Threshold Logic Neuron. Sources for each logic gate can be any of the eight inputs, any of the six Neurons, or a consistent ON/OFF value. Four Setups can be saved for later use, and Setups can be saved and loaded via MIDI system exclusive.

Basic Controls and Navigation

Neural Net has two basic setup screens, the Selection Screen (which shows all six Neurons), and the Edit Screen (which allows editing and monitoring of a single Neuron). The controls work exactly the same way on both screens, but some changes are only visible from the Edit Screen.

- Left Encoder: Specifies what is being edited. It can be any of the six Neurons, or the Output assignments.
- Left Encoder Press: Alternates between the Selection Screen and the Edit Screen.
- Left Encoder Long-Press: Toggles the All Connections display.
- Right Encoder: Changes the value at the cursor.
- Right Encoder Press: Advances the cursor.
- Up/Down Buttons: Selects which Setup (from 1-4) is active. If the down button is long-pressed, the active Setup may be copied to another Setup.

Selection Screen

The Selection Screen shows an overview of all six Neurons, all inputs, and all outputs. Use the left encoder to choose a different Neuron, or the outputs.

The inputs are on the left side, with the Digital inputs (1-4) under the D, and the Control Voltage inputs (1-4) under the V. When an input is high, its corresponding number's display is reversed.

The outputs (A-D) are on the right side. When an output is high, its corresponding letter's display is reversed. If the outputs are selected, a blinking line will appear to the left of the outputs.

The six Neurons are in the center, with Neurons #1, #3, and #5 at the top, and Neurons #2, #4, and #6 at the bottom. Each Neuron shows the symbol of its selected logic gate. If a Neuron is selected, the logic gate type's name will blink, and lines from the Neuron's source(s) and dotted lines to assigned outputs (if any) will be displayed.

Note: When a new Neuron is selected, the Neuron's editing cursor is reset to the logic gate type. This means that you can use the right encoder to set the gate type without having to go to the Edit screen.

Another Note: Actually, you never *have* to go to the edit screen. Since the controls work the same way on both screens, you can cursor through the parameters and change everything right from the Selection Screen. But it's hard to do that blind, and you'll probably want to go to the Edit Screen.

Long-press the left encoder to turn on All Connections. This will show source and assignment lines for all Neurons and outputs. Long-press the left encoder again to turn off All Connections.

Edit Screen

Press the left encoder button to alternate between the Selection Screen and the Edit Screen.

The Edit Screen is split into two halves. The left half of the screen shows the logic gate type and the Neuron's parameters. The right half of the screen shows the logic gate symbol, along with its inputs and output state. For the Threshold Logic Neuron, the right half of the screen also allows editing of weights and threshold.

Press the right encoder button to advance the cursor, and turn the right encoder to change the value. The following sources are available: **Dig 1, Dig 2, Dig 3, Dig 4, CV 1, CV 2, CV 3, CV 4, Neuron 1, Neuron 2, Neuron 3, Neuron 4, ON, and OFF**.

Edit Screen Parameter Names

The meanings of the parameters you'll see on the Edit Screen are below. See Logic Gate Reference for details about what the settings actually do.

- **Opd**: Source of an operand of a logical operation (NOT, AND, OR, XOR, etc.). All operands in a logical gate are "commutative," meaning that the resulting state will be the same regardless of the order of the operands.
- **Data**: Source of the Data input for the Data Flip-Flop.
- **Toggl**: Source of the Toggle input for the Toggle Flip-Flop.
- **Clock**: Source of the Clock input for the Flip-Flop.
- **Set**: Source of the Set input for the Set-Reset Latch.
- **Reset**: Source of the Reset input for the Set-Reset Latch.
- **Den 1-3**: Source of a Dendrite input for the Threshold Logic Neuron.

Output Assignment

When the outputs are selected, the Edit Screen shows Outputs A-D, with a Neuron assigned to each. The state of the selected Neurons will be sent to the specified outputs.

Logic Gate Reference

- **NOT** is a unary (single-input) gate whose state is the opposite of its operand.
- **AND** is a binary (two-input) gate whose state is ON if and only if both of its operands are ON.
- **OR** is a binary gate whose state is ON if and only if at least one of its operands are ON.
- **XOR** is a binary gate whose state is ON if and only if exactly one of its operands is ON.
- **NAND** is a binary gate whose state is ON if and only if at least one of its operands are OFF.
- **NOR** is a binary gate whose state is ON if and only if both of its operands are OFF.
- **XNOR** is a binary gate whose state is ON if and only if both of its operands are the same value.
- **D-FF** is a *Data Flip-Flop*. When its Clock input goes high, its state becomes the value of its Data input.
- **T-FF** is a *Toggle Flip-Flop*. When its Clock input goes high, its state flips if the value of its Toggle input is ON.
- **Latch** is a *Set-Reset NOR Latch*. When its Set input goes ON, its state becomes ON. When its Reset input goes ON, its state becomes OFF.
- **TL Neuron** is a Threshold Logic Neuron, a ternary (triple-input) gate whose state is ON if and only if the

sum of the weights of ON inputs (dendrites) exceeds its specified threshold. See here for more information about Threshold Logic Neurons.

Setups

Neural Net has four independent Setups. A Setup consists of information for Neuron logic gate assignments, source assignments, and output assignments. Use the Up and Down buttons to change the active Setup. The active Setup number is shown at the top of all of the Setup screens.

Copying Setups

To copy the information for the active Setup to another Setup, long-press the Down button. When you release the button, a Copy screen will open. Choose the copy's destination with the Up and Down buttons, and then push the right encoder ("[Copy]") to execute the copy. To leave the Copy screen without changing any data, push the left encoder ("[Cancel]") or long-press the Down button again.

If you select a Setup as its own copy destination, the display will change to a SysEx dump screen. The right encoder option becomes "[Dump]". If you push the right encoder, Neural Net will send a system exclusive file containing the data for the active Setup.

Saving Setups

There are two ways to save Setup data.

Save all four Setups in the module: Long-press the left encoder to get to the main menu, then long-press the left encoder again to save data for all apps. A solid rectangle will expand from the middle of the screen to confirm the save operation. The Ornament and Crime module will recall the saved Setups on power-up.

Save using MIDI system exclusive: Follow the SysEx dump procedure from the Copying Setups section above, and capture the system exclusive dump with SysEx librarian software. The restore a Setup from a SysEx dump, simply send the SysEx file back to the module whenever Neural Net is running.

Shortcut: When you long-press the right encoder to go to the main menu, the current Setup's data also is dumped via SysEx.

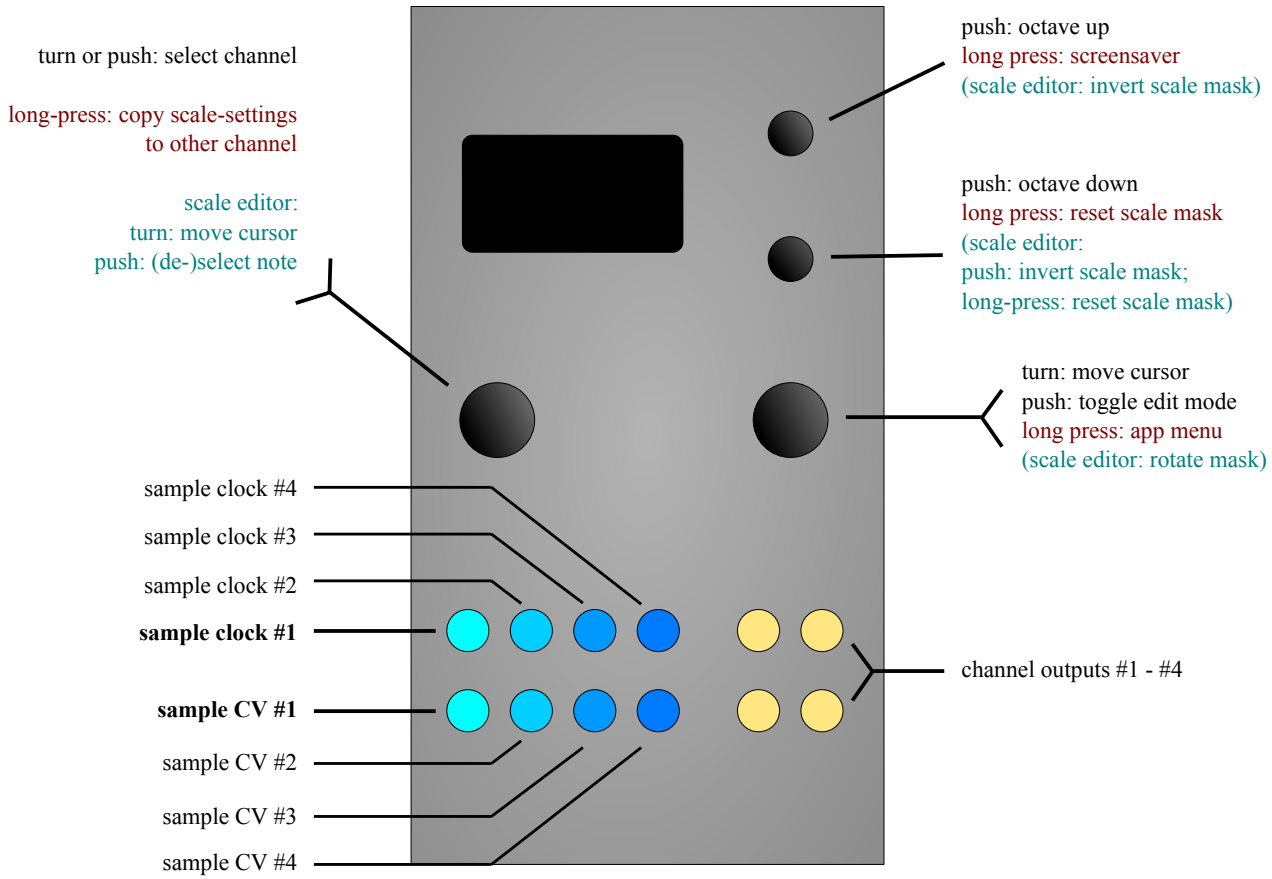
Notes about System Exclusive:

- A Setup from a received SysEx dump will always be placed in the active Setup, *regardless of the Setup's original location.*
- You can use system exclusive to change Neural Net in an automated way during a performance.
- Data received from a SysEx dump will overwrite Setups in memory, but not data saved via the first save method; that is, unless you long-press to save the new Setup data, Neural Net will return to its previously-saved state at the next power-up.

Quantermain

4 channel quantizer, integrates the quantiser from the Mutable Instruments Braids (https://github.com/pichenettes/eurorack/blob/master/braids/quantizer.h) module, but expands it with **interactive scale-edit** functionalities (99 pre-defined scales (editable in the source code - see instructions here (/custom-scales/) - are included, plus 4 fully user-definable scales (with a maximum of 16 notes per octave and additional finetune/microtonal edit options)).

- for further information re **custom scales, microtonal scales and tunings** see here (http://ornament-and-cri.me/custom-scales/) and here.
- if properly, calibrated, the **output accuracy** should be fairly excellent (is fairly excellent) due to the use of the precision TI 16-bit DAC; the **trigger-to-quantised-output latency** is also very decent: < 100 microseconds.
- the four channels are fully independent, but can be slaved to the same clock and/or track the same CV sources, if desired. here's the default mapping:



Input/output assignment

I/O	Function
TR1, TR2, TR3 & TR4	Mappable as the trigger input for each channel A to D, independently (except when `Trigger source` is set to `cnt+` or `cnt-` — see below)
CV1, CV2, CV3 & CV4	Mappable as the external CV input to be quantised for each channel A to D, independently, or can be mapped to control various parameters when internal CV "sources" (e.g. built-in Turing Machines) are used.
A	Output voltage for channel A (quantised if quantisation is enabled)

I/O	Function
B	Output voltage for channel B (quantised if quantisation is enabled)
C	Output voltage for channel C (quantised if quantisation is enabled)
D	Output voltage for channel D (quantised if quantisation is enabled)

Controls

Control	Function
Left encoder (turn)	Select channel, or note in scale in scale edit mode
Left encoder (press)	Increment channel by one
Right encoder (turn)	Navigation mode: move up and down through the menu items. Edit mode: increase or decrease the value being edited.
Right encoder (press)	Toggle between menu navigation (selection) mode and value editing mode
Right encoder (long press)	App selection menu
Up button	Transpose up one octave for currently displayed channel
Down button	Transpose down one octave for current displayed channel
Up button long	screen saver short cut
Down button down	reset scale mask

Available settings (per-channel)

Setting	Meaning
`scale`	Current scale
`Root`	Root note for scale
`Active notes`	“scale mask” / active note pattern in the selected scale
`CV Source`	external CV source (CV1 to CV4) or internal value sources (Turing Machine or Lgstc (logistic map) or `ByteB` (byte beats) or `IntSq` (integer sequences) for the current channel
`CV aux >`	auxiliary CV destination: `root` , `oct` , `trns` , or `mask` (enabled only when using a **non-default** CV setting for `CV Source` is used, e.g. if `CV2` is selected as the source for channel A, then `CV1` can be re-mapped using `CV aux` . `CV aux` is not enabled if any of the internal pseudo-CV sources (Turing Machine/LFSR, byte beats etc) are selected).)
`Trigger source`	Trigger input source (TR1 to TR4, or `cnt+` or `cnt-` for continuous quantisation, see below for discussion of continuous quantisation) for the current channel
`Clock div`	Clock divider for the trigger input selected for the current channel (note: the same trigger source may have different clock dividers set in each channel)
`Trigger delay`	sets the delay between receiving a trigger or gate (rising-edge) on the chosen trigger input for the current channel, and the input voltage being sampled. It defaults to Off (no delay beyond the normal latency, which is less than 100 microseconds, typically about 50 microseconds), but available values are 120us, 240us, 360us, 480us, 1ms, 2ms and 4ms. (†)
`Transpose`	Transpose up or down of output CV for current channel (in scale-degrees)
`Fine`	Fine tuning control, adjusts the pitch CV for the current channel up or down in tiny increments.

- (†) o_C is very fast, processing signals in the audiorate, which (in triggered/clocked quantisation mode) can easily produce a **mismatch** between the voltage that you *intend* to sample and the voltage that **actually** is being sampled (that is, used as an input to the quantizer). For example, if a sequencer or some other stepped voltage source is being sent to the quantizer, then that voltage may still be slewing to its new intended voltage when o_C acquires the samples; or if this source is digital, there may be a few tens of microseconds delay before the voltage source reacts to the trigger signal (assuming it is the same trigger signal that has been sent to o_C) and starts to slew to the new voltage. If this is the case, the *Quantermain*, *CopierMaschine*, *Harrington 1200*, *Automatonnetz*, *Meta-Q*, *Sequins* or *Acid Curds*

apps will be sampling/quantizing the **previous** note / step, or they may be sampling the CV as it is slewing to the new value. In such cases, increase the `Trigger delay` setting to compensate for these differences in timing. NB: These considerations only apply when using *Quantermain* in trigger mode and to the *CopierMaschine* app, which can only be used with an external trigger) — they do not apply when `Trigger Source` is set to `cnt+` or `cnt-` (= continuous quantization) in *Quantermain*.

Method of operation

- When used as a voltage quantiser, for each of the four independent channels, the Quantermain app reads an input voltage from the CV source set for that channel and quantises it to the scale and notes set for that channel, and then applies any post-quantisation transpositions that have been specified. Voltages are read an a new quantised voltage is output when the trigger source specified for that channel fires (rising edge of the trigger, clock or gate signal), unless `Trigger source` is set to `cnt+` or `cnt-` , in which case quantisation is performed continuously on the input voltage. The latency from receipt of a trigger to output of a newly quantised note is under 100 microseconds, and is typically about 50 microseconds. In continuous quantisation mode, the input voltage is read at an effective rate of about 1 kHz, thus quantisation of the input occurs approximately once every millisecond.
- When `CV source` is set to an input channel **other** than the default/nominal input (e.g. channel #2 is set to `CV source = CV4`), the default input can be re-mapped to a different channel parameter. See `CV aux >` .
- When the `CV source` is set to `Turing` , `Lgstc` , `ByteB` or `IntSq` , instead of reading a voltage on one of the CV inputs (`CV1` to `CV4`), an internally generated value is used instead (for that channel - you can use the internal sources on some channels and external CV sources on others if you wish). Details of each of these internal sources follow.

Note: the Turing Machine, Logistic Map, byte beat and integer sequences sources all require external clock inputs to make then step, and thus they do not operate when `Trigger source` is set to `cnt+` or `cnt-` .

The internal Turing Machine source

The `Turing` source uses a linear-feedback shift register (LFSR) randomly seeded with habit pattern, which is right-shifted by one bit at each clock step (read from the `Trigger source` for that channel). There is a settable probability that the least-significant bit is randomly each time the pattern is shifted by one. This arrangement was popularised in modular synthesis as the Richter Noise Ring and the Music Thing Turing Machine. The name of the latter module has been borrowed here (see acknowledgements (http://ornament-and-cri.me/acknowledgements/)).

If a Turing Machine has been selected as as the `CV Source` , then the following additional settings are made available:

Setting	Meaning
`LFSR length`	length of the linear feedback shift register, in bits, range 2 to 32
`LFSR modulus`	sets the modulus for the integer value output from the LFSR. The value of the integer from the LFSR is divided by the modulus and the remainder is used. For example, if the modulus is 8 and the current integer value from the LFSR is 19, then the remainder of 19 - (2 x 8) i.e. 3 is used as the value. In other words, values “wrap around” at the modulus setting value — it sets a maximum note range for the LFSR, similarly to the `LFSR range` setting, but `LFSR range` compresses or expands the range of notes for a given integer value from the sequence, whereas the modulus wraps the values around. **Note**: the `LFSR modulus` setting and any external voltage input as set by `LFSR mod CV src` have **no effect** on the LFSR is no `scale` is selected.
`LFSR range`	the range or span of notes available to the LFSR (Turing Machine) from which a note is selected

Setting	Meaning
`LFSR prb`	probability that the least significant bit will be flipped when it to copied
`LFSR p CV src`	(v1.1 or later only) The control voltage source (none, or CV1, CV2, CV3 or CV4 inputs) used to control the probability. The CV value is added to the probability value set via the `LFSR p` menu item (see above).
`LFSR mod CV src`	(v1.2 or later only) The control voltage source (none, or CV1, CV2, CV3 or CV4 inputs) used to control the modulus of the values output by the LFSR (Turing Machine) from which a note is selected. The CV value is added to the LFSR modulus value set via the `LFSR modulus` menu item (see above).
`LFSR rng CV src`	(v1.1 or later only) The control voltage source (none, or CV1, CV2, CV3 or CV4 inputs) used to control the range or span of notes available to the LFSR (Turing Machine) from which a note is selected. The CV value is added to the range value set via the `LFSR range` menu item (see above).

The internal Logistic Map source

If a Logistic Map has been selected as as the `CV Source`, then the following additional settings are made available:

Setting	Meaning
`Logistic r`	The ***r*** coefficient for the logistic map equation (https://en.wikipedia.org/wiki/Logistic_map)
`Logistic range`	The range or span of notes available to the Logistic Map from which a note is selected.
`Log r CV src`	(v1.1 or later only) The control voltage source (none, or CV1, CV2, CV3 or CV4 inputs) used to control the r coefficient for the logistic map equation (https://en.wikipedia.org/wiki/Logistic_map). The CV value is added to the ***r*** value set via the `Logistic r` menu item (see above).
`Log rng CV src`	(v1.1 or later only) The control voltage source (none, or CV1, CV2, CV3 or CV4 inputs) used to control the range or span of notes available to the Logistic Map from which a note is selected. The CV value is added to the range value set via the `Logistic range` menu item (see above).
`Logistic seed`	The seed value to initialise the logistic map (has very little effect, but different values result in different sequences)

The internal Byte Beats source

If `ByteB` (byte beats) has been selected as as the `CV Source`, then the following additional settings are made available:

Setting	Meaning
`Bytebeat eqn`	sets the byte beat equation used as the source. See the *Viznutcracker, sweet!* app documentation for more details of the currently available equations.
`Bytebeat range`	The range or span of notes available to the byte beat equation source from which a note is selected.
`Bytebeat P0`	Parameter 0 for the byte beat equation - see the *Viznutcracker, sweet!* app documentation for more details.
`Bytebeat P1`	Parameter 1 for the byte beat equation - see the *Viznutcracker, sweet!* app documentation for more details.
`Bytebeat P2`	Parameter 2 for the byte beat equation - see the *Viznutcracker, sweet!* app documentation for more details.
`Bb eqn CV src`	The CV input source to vary the byte beat equation used.
`Bb rng CV src`	The CV input source to vary the byte beat range.
`Bb P0 CV src`	The CV input source to vary byte beat equation parameter 0.

Setting	Meaning
`Bb P1 CV src`	The CV input source to vary byte beat equation parameter 1.
`Bb P2 CV src`	The CV input source to vary byte beat equation parameter 2.

The internal Integer Sequences source

The `IntSq` source in *Quantermain* behaves similarly to the `IntSq` source in *CopierMaschine* — full details are given in the *CopierMaschine* section above. The same integer sequences are available, but in *Quantermain*, up to four channels of them can be used independently and simultaneously.

Setting	Meaning
`IntSeq`	sets the integer sequence used as the source. See the *CopierMaschine* section for a list of available integer sequences and their characteristics.
`IntSeq modul`	sets the modulus for the integer sequence. The value of the integer from the integer sequence is divided by the modulus and the remainder is used. For example, if the modulus is 8 and the current integer value from the sequence is 19, then the remainder of 19 - (2 x 8) i.e. 3 is used as the value. In other words, values “wrap around” at the modulus setting — it sets a maximum note range for the integer sequence, similarly to the `IntSeq range` setting (see next row of this table), but `IntSeq range` compresses or expands the range of notes for a given integer value from the sequence, whereas the modulus wraps the values around.
`IntSeq range`	sets the span or range of notes created by the integer sequence. It essentially compresses or expands the mapping from integer values to notes. Unlike `IntSeq modul`, it does cause the note values to “wrap around”. Both `IntSeq range` and `IntSeq modul` can be used together.
`IntSeq dir`	sets whether the integer sequence loops back to the beginning when it gets to the end, or whether it swings back like a pendulum and plays in reverse when it reaches the end. The “end” is the last step in the sequence, as defined by the sequence start plus the sequence length settings.
`IntSeq start`	sets the start point in the stored integer sequence. The stored sequences are 128 steps long, and the maximum start point is 126 to ensure a minimum sequence length of 2.
`IntSeq len`	sets the length of the integer sequence. Thus a length of 16 will use just 16 values from the stored 128 step sequence, starting at the step specified by `IntSeq start`.
`IntSeq FS prob`	sets the probability (0 means *p*=0, 255 means *p*=1) that the `Fractal stride` value will be randomly altered at the next step. See `Fractal stride` in this table.
`IntSeq FS rng`	sets the range of the probabilistic shift in the fractal stride controlled by `IntSeq FS prob`. The shift range is from 0 to 5.
`Fractal stride`	Two of the sequences are fractal in nature, and the “stride” setting sets how many steps are advanced on each trigger input. This also works with the non-fractal sequences and provides additional variation, particularly if the stride is not an exact divisor of the sequence length.
`IntSeq CV`	configures external CV control over the integer sequence used. The available choices are `None`, `CV1`, `CV2`, `CV3` or `CV4`.
`IntSeq mod CV`	configures external CV control over the integer sequence modulus. The available choices are `None`, `CV1`, `CV2`, `CV3` or `CV4`.
`IntSeq rng`	configures external CV control over the integer sequence range. The available choices are `None`, `CV1`, `CV2`, `CV3` or `CV4`.
`Frctl stride CV`	configures external CV control over the integer sequence fractal stride. The available choices are `None`, `CV1`, `CV2`, `CV3` or `CV4`.
`IntSeq reset`	configures external trigger control over reset of integer sequence pointer. The available choices are `None`, `TR1`, `TR2`, `TR3` or `TR4`. A positive trigger (or rising edge) received on the selected trigger input will cause the integer sequence counter to return to the start position of the sequence (as determined by `IntSeq start`).

Active note (scale mask) and scale editing

To invoke the active note and scale editor, click the right encoder when the *Active notes* menu option is selected.

- turn the left encoder to move the note cursor, click the left encoder to toggle currently selected note on or off
- Up/Down buttons invert the current note mask
- turning the right encoder note-shifts the pattern left or right (= rotate scale mask)
- click the right encoder to exit

Using the note editor (user-scales 1-4):

- use the right encoder to scroll to the top of the menu, and click the right encoder to select the scale. Choose one of the 4 user-editable scales (`USER1` , `USER2` , etc) at the beginning of the list of scales. Click the right encoder again to return to menu scrolling mode.
- Scroll down to *Active notes* and click the right encoder to invoke the note editor.
- turn the left encoder to move to the note you wish to edit. **Hold down the left encoder, and use the right encoder to edit the pitch value for that note**.
- **the length of the scale** can be set by navigating to the final note in the scale, and turning the right encoder to shorten or lengthen the scale. In this way, scales with **4 up to 16 microtonal notes** are possible.
- click the right encoder to exit the note editor
- user-defined scales are saved along with all the other settings, when the save settings procedure is used

Note

The user-defined scales only save the note values in a scale - that is, the number of notes in the scale and the pitch value for each of those notes. The user scales don't save the note masks applied to those scales. Note masks **are** stored when you save settings, but they are not stored with the user-defined scales because note masks also apply to the pre-defined scales.

The user-defined scales were really intended for defining microtonal or non-equally-tempered tunings, where the notes are not spaced 100 cents apart (or at multiples of 100 cents).

However, you can also use the user-defined scales for defining any subset of, say, the "normal" 12-tone equally-tempered (12-TET) chromatic scale. It is a bit fiddly, but doable. In the note mask editor, with a user scale selected, first note down with a pen and paper the pitch values for the 12-TET notes you want in your scale. Note that the values are not in cents, they are in 1/1536 divisions of the octave. Then adjust the number of notes in the user-defined scale using the lozenge at the extreme right of the scale in the editor. Then go to each note in turn in your user-defined scale and adjust the pitch value for it to match the values you wrote down. The notes must be in ascending order! Exit the note mask editor, save you settings and then the user-defined scale should be that subset of notes you just defined. You can still apply note masks on top of that to further subset the notes in the scale if you wish.

Continuous (non-triggered) quantisation modes

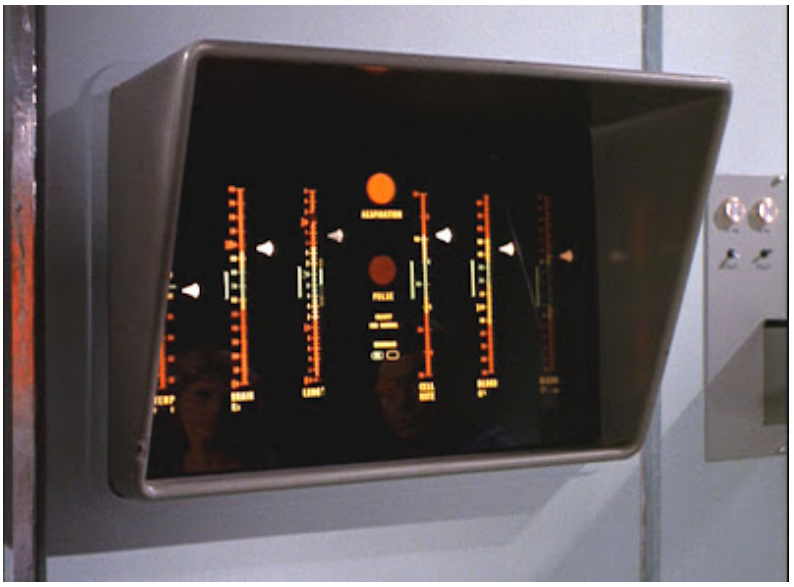
When `Trigger source` for a channel is set to `cnt+` or `cnt-` , quantisation is performed on that channel continuously (well, about 16,667 times per second). In addition, the trigger input which ordinally corresponds to that channel (i.e. TR1 for channel A, TR2 for channel B, TR3 for channel C and TR4 for channel D) can then be used as a gate to transpose the output for that channel up by one octave (if `cnt+` is set) or down by one octave if `cnt-` is set. In other words, for channel B, if the `Trigger source` for it is set to `cnt+` , then if TR2 input is high, the output for that channel will be transposed up by one full octave, while the TR2 input remains high. Likewise, if `Trigger source` is set to `cnt-` , then channel B output will be transposed down one octave while TR2 is held high. Note, however, that the octave transpositions only take effect when there is a

note change i.e. when the CV input for that channel has changed sufficiently that it quantises to a new note value. That way, octave transpositions co-incide with note changes, which sounds much better.

Note also that although the trigger inputs used by each channel can be mapped independently, the `cnt+` and `cnt-` octave transposition behaviour cannot be mapped - it is hard-coded, so that only TR1 can be used for this purpose for channel A, only TR2 for channel B and so on. Note that TR1 can still also be used as a trigger for any other channel, but not for the `cnt+` and `cnt-` octave transposition behaviour on other channels.

Screensaver display

The screensaver for *Quantermain* is inspired, possibly on an unconscious level, by the Sick Bay vital signs display in the original *Star Trek* series.



There are four “lanes”, one for each channel. In each lane, short lines representing the quantised pitch (on a semitone scale) scroll leftwards. The small triangles to their right move up and down to indicate the octave for that channel. Triggers (a vertical bar) and input voltages (a horizontal bar, left-going for negative voltages, right-going for positive) are indicated above each channel lane (replaced by a bit-pattern display when the sources are set to LFSR (Turing Machine) or logistic map, rather than external CV).

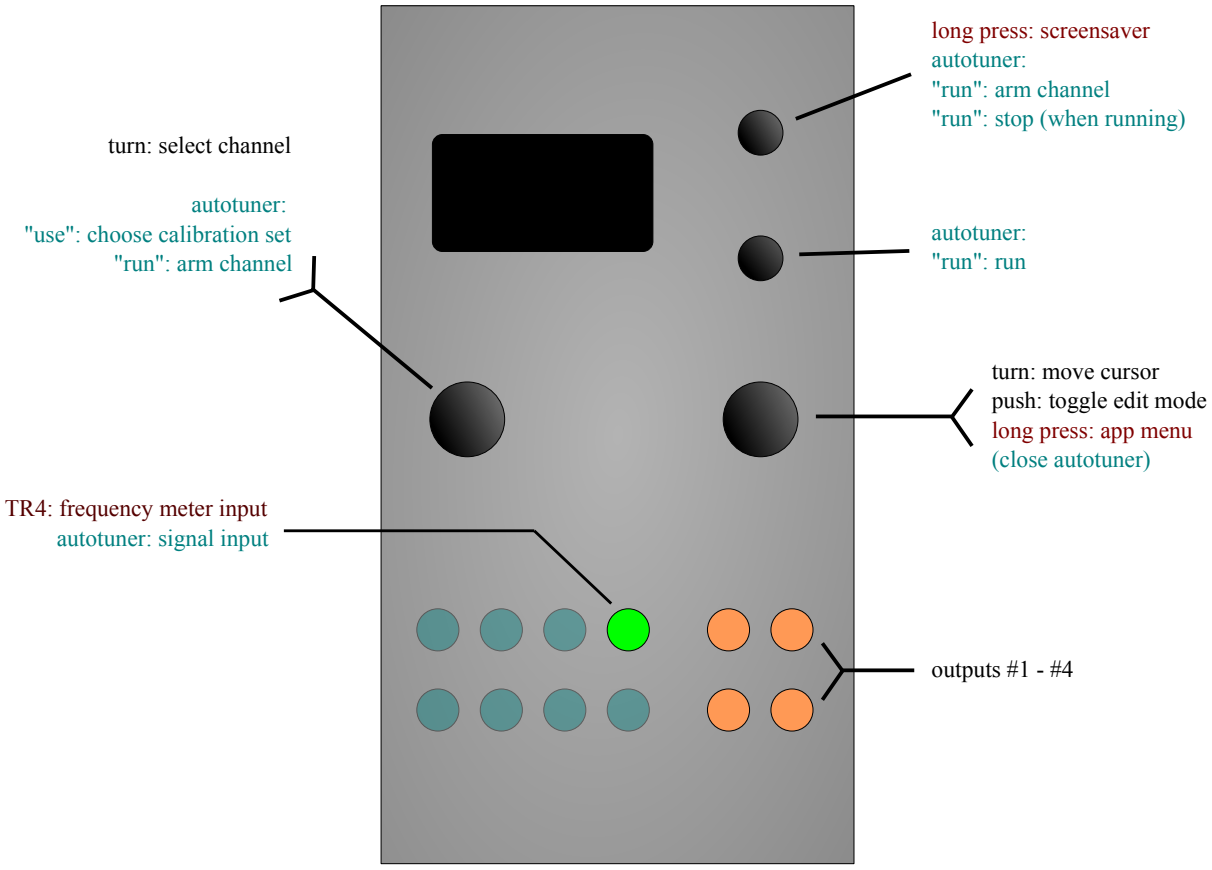
Tips

You can disable quantisation on each channel by setting the scale to `Off`. When in clocked mode (that is, `Trigger source` is set to TR1, TR2, TR3 or TR4), then that channel acts as a traditional sample-and-hold (S&H), without any voltage droop, of course. Thus, Quantermain can act as a quad S&H if you want. The only limitation is that the effective maximum sample rate is about 1 kHz, even if the clock/trigger signal you give it is at a higher frequency. This is because the ADCs are only read at an effective rate of about 1 kHz.

Disabling quantisation when using a Turing Machine, Logistic map, byte beats or integer sequence internal source allows the output for that channel to act as a semi-random modulation source.

References

References is a tuning-utility app. It comes with a frequency/BPM meter (via the TR4 input, or TR1 if the module is being used in upside-down mode) and a **closed-loop calibration** mode, as well as outputs fixed **reference voltages**. These are handy when calibrating other modules, such as VCOs.



- Most of these functions should be fairly self-explanatory.
    - To use the frequency/BPM meter, simply patch a signal into **TR4** (see note below) and long-press the up-button to invoke the screensaver, where the measurement results will be displayed:
        - the frequency of the audio or clock signal input into TR4 is displayed in Hertz, to three decimal places (and it really is accurate to at least three decimal places!).
        - the `Note/BMP` menu setting determines whether notes tuning or BPM is displayed below the frequency reading
    - To use the closed-loop calibration:
        - 1) patch the output channel (A-D) that you want to calibrate into the V/oct input of your VCO.
        - 2) patch output of the VCO (ideally, a square wave) into **TR4**.
        - 3) then right-click on `autotune -->`, and turn right encoder to select `run -->`.
        - 4) press the up-button to "arm" the tuner. you should see a number displaying the frequency (at 0V); this shouldn't jump around erratically (in which case the procedure is bound to fail); tune the initial frequency to somewhere in the 50Hz-200Hz range.
        - 5) then push the down-button to set off the procedure, which will take a couple of seconds. If everything goes well, it'll save the calibration data into the slot `use --> .auto`, which can be selected/de-selected via the menu item `use -->`.

Note: the frequency meter/BPM and closed-loop calibration input is **TR1** if the module is being used in

upside-down mode.

Controls

Control	Function
Left encoder (turn)	Select channel A to D to edit (all channels always active)
Right encoder (turn)	Navigation mode: move up and down through the menu items. Edit mode: increase or decrease the value being edited.
Right encoder (press)	Toggle between menu navigation (selection) mode and value editing mode
Right encoder (long press)	App selection menu
Up button (long press)	invokes the screensaver display immediately
Down button	

Available settings (per-channel)

Setting	Meaning
`Octave`	sets the octave range. Note that the octave numbering is such that C for octave 0 is 0V, C for octave 1 is 1V and so on.
`Semitone`	sets the semitone offset within each octave range. One semitone increment is 83.33mV.
`Mod range oct`	sets the number of octaves to automatically jump up or down. This can be useful when adjusting trimpots on VCOs.
`Mod rate (s)`	sets the rate (as a period in seconds) at which the reference voltage jumps up or down by the number of octaves set by `Mod range oct`.
`autotune -->`	opens the autotuner-submenu
`Notes/BPM`	toggle displayed values (for frequency meter) - frequency in Hertz or beats per minute (bpm)
`A above mid C`	Sets the frequency in Hertz (Hz) of the A above middle C used for the tuner calculations (range is 400 Hz to 480 Hz). It defaults to 440 Hz, which is now standard Western concert pitch, but it can be set to other values.
`> mantissa`	sets the fractional part of the `A above mid C` value, in tenths of a Hertz (0.1 Hz) increments.
`> ppqn`	sets the number of pulses per quarter note (ppqn) used for the bpm display (available settings are 1, 2, 4, 8, 16, 24, 48, 64 and 96 ppqn)

Closed-Loop Calibration

Control	Function
Right encoder (turn)	Navigation mode: move up and down through the menu items.
Left encoder (turn)	item: `use -->` : choose calibration-set for selected channel (when options are available): default calibration autotuned
Up button	item: `run-->` : arm autotuner (TR4, or TR1 if upside-down), or stop procedure if already running
Down button	item: `run-->` : run autotuner
Right encoder (press)	close
Right encoder (long press)	App selection menu

Inputs and outputs

TR4: frequency-meter input, inputs TR1-TR3 are unused. Reference voltages for channels A to D appear on

outputs A to D respectively.

Note: the reference voltages output by the References app are only as accurate as the calibration of the module. However, the module is capable of a precision of better than +/- 0.5mV, if calibrated with a precision multimeter. Furthermore, the `autotune -->` setting for each channel must be set to `default`, not `auto` in order for the output voltages output to reflect the indicated voltage. If set to `auto`, then correction factors for specific VCOs will be applied, and the output voltage will not match the indicated voltage.

Note: the indicated output voltages are only valid if 1V/oct output scaling (the default) is selected for each channel. If a dotted line appears under the menu, then an alternative output scaling is active. You may need to set this back to 1V/oct before using the References app. As at v1.3, you can only access the alternative output scaling settings via the scale editor in the *CopierMaschine*, *Quantermain*, *meta-Q*, *Sequins* or *Acid Curds* apps. We apologise for this slight inconvenience.

Screensaver display

The screen is divided into four channel lanes (columns), with the same layout as the *Quantermain* screensaver, except that the output voltage (rounded to the nearest millivolt) is shown as the top of each lane (column). Note that the minus signs for negative voltages are quite small due to the limited display space, and can be quite hard to see.

Below the channel lanes, the TR4 (or TR1 if run upside-down) frequency measurement is displayed. This will blank itself out ofter about a minute if no signal is being received on TR4 (or TR1 if flipped). Below that is the note/tuner display, or the BPM (beats per minute) display, depending on the `Notes/BPM` setting. Note that the note tuner only operates down to about 16 Hz.

Scale Editor

Jump to bottom

Chysn edited this page on Aug 27, 2018 · 4 revisions

https://www.youtube.com/watch?v=UKX79rkSdIQ

Scale Editor is a standalone application for editing and managing user-defined microtonal scales, and importing scales from the internal scale library, or via system exclusive. A web-based Scala-to-Hemisphere Suite converter is available (http://www.beigemaze.com/scala), which can generate system exclusive files from Scala documents.

Basic Controls and Navigation

- CV Input 1/Output A: CV 1 is input for unquantized pitch CV. The quantized pitch appears at Output A, using the currently-selected user scale. Pitch is quantized in real time
- CV Output B: Quantized pitch for the currently-selected user scale and the selected note.
- Right Encoder: Selects note or length. Push the right encoder button to toggle between those two functions. For Output B, note selection can be used to set the octave of the currently-selected note's output. On the Import screen, the right encoder chooses the scale, and the right encoder button executes the import.
- Left Encoder: Changes the value of the currently-selected note by as little as 1.28 cents sharp or flat. Pressing the left encoder undoes the most recent note value change. Long-pressing the left encoder sends a system exclusive dump of the current scale data.
- Up/Down: Chooses the user scale. Long-pressing the Down button opens the Import screen.

Choosing a User Scale

Press the Up and Down buttons to choose a scale (from USER1 to USER4).

Selecting and Editing Notes

Use the right encoder to choose which note you're currently editing. The value of the note, in cents from the root, is shown at the bottom of the display.

Turn the left encoder to change the value by as little as 1.28 cents. Turn clockwise for sharp, and counterclockwise for flat. Note values cannot overlap or exceed adjacent nodes, so the range of each note will be constrained based on the settings of the next note.

Undo

If you make a mistake, you don't need to remember the old value of the note. Press the left encoder button to undo a note value change. You can undo a change until you select a different note.

Monitoring

You can hear the changes you're making in real time with two different monitoring methods. Both methods update pitch in real time as you make changes:

- Full Scale Monitoring: Patch unquantized CV into Input 1, and patch Output A into an oscillator. Output A will continuously quantize Input 1 according to the currently-selected scale. The octave played will correspond to the input.
- Selected Note Monitoring: Output B will continuously output the quantized value of the currently-selected note. You can change the note's octave (within a 5-octave range) by turning the right encoder beyond the first or last note.

Changing Scale Length

User scales may contain between 4 and 16 notes. Press the right encoder button to enter the Scale Length screen. The display will show the number of notes. Now, use the right encoder to choose the number of notes. When you're done, press the right encoder to return to the Note Edit screen.

Importing Scales

The Import screen allows you to copy any scale in the scale library to the currently-selected user scale. Long-press the Down button to enter the Import screen. Then use the right encoder to choose a scale. To import the specified scale into the current user scale, press the right encoder ("[IMPORT]"). If you change your mind, press the left encoder ("[CANCEL]").

System Exclusive

You may initiate a MIDI system exclusive dump of the current scale by long-pressing the left encoder, or by long-pressing the right encoder to return to the main menu. You may capture the system exclusive dump with a sysex librarian program on a connected computer.

To restore a scale from a sysex dump, simply initiate the dump from the computer while Scale Editor is running. The received scale will be placed into the currently-selected scale.

Scala Format Import

With the help of a web-based tool (http://www.beigemaze.com/scala), you can bring Scala scales into Scale Editor. Upload (or paste) your Scala into the tool, and download the SysEx file. See "System Exclusive" (above) for how to send the file to Scale Editor.

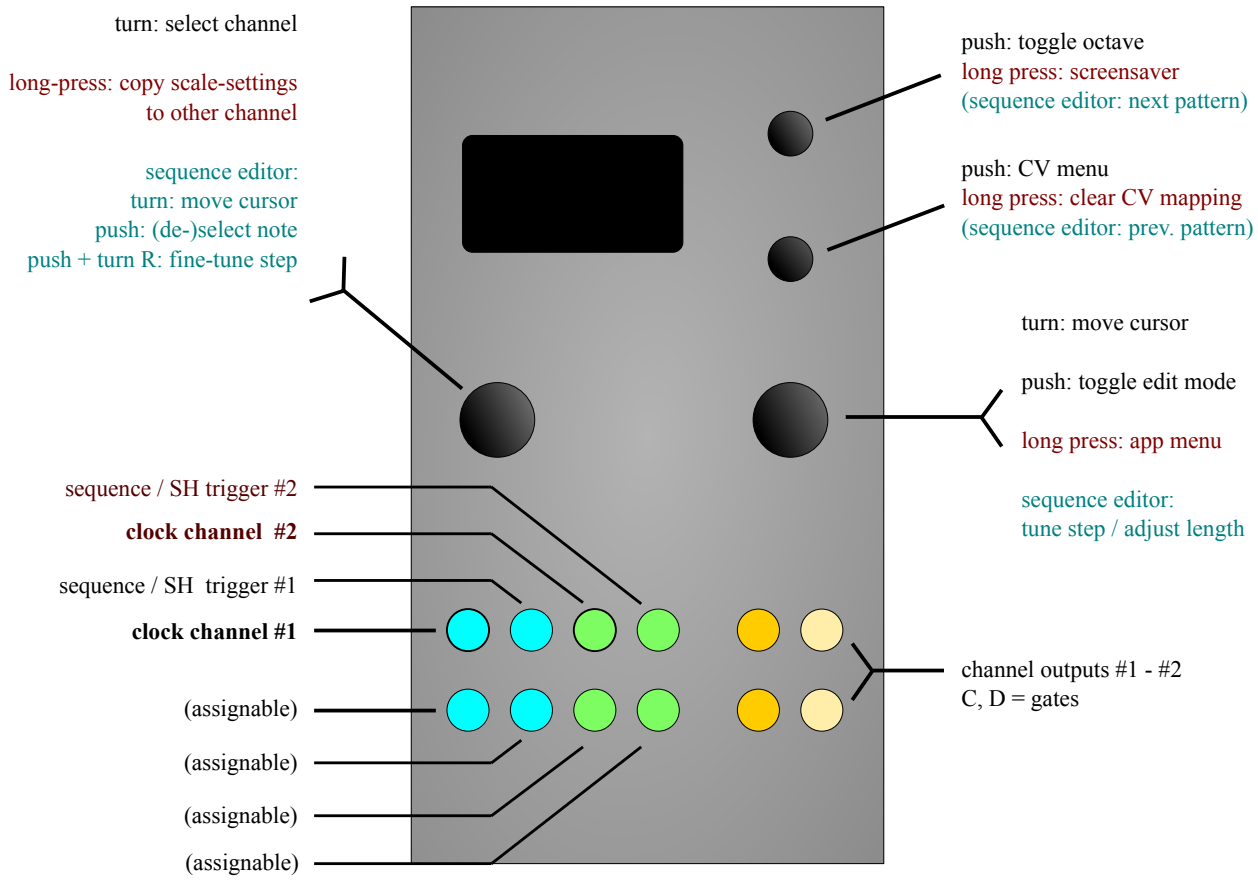
For your convenience, sysex files for about 3000 scales from the Scala Scale Archive are available here, along with the original Scala files and instructions:

http://beigemaze.com/downloads/HS_Scala_Archive.zip

These scales are distributed with the kind permission of Manuel Op de Coul

Sequins

Sequins is a basic, **dual-channel sequencer**. The app provides four 16-step sequences per channel, and CV-control over various channel parameters, including scale mask, octave, pulsewidth, clock multiplication and division. The four sequences can be **chained** in various ways, providing sequences from **4 to up to 64 notes**. The sequence data is stored along with the other settings (whenever settings are saved).



- the **sequence editor** works similarly to the scale editor in the quantizer modes:
  - in the main menu, select which sequence to edit by adjusting the `sequence #` parameter.
  - then right-click on the item displayed right below (`--> edit`) to **open the sequence-editor**.
  - **mute/unmute** notes by clicking the left encoder: the step value is displayed in a format x+y.yy, where x is the octave (above or below 0.00V) and y.yy gives the semitone offset. for instance `+2+2.23` = 2.23 semitones above octave #2 (=~ 2.000V).
  - adjust the sequence **length** by pointing the cursor to the far right, then turn the right encoder.
  - (**simultaneously** push down the *left* encoder to also **mute/clear** the steps that are being added while expanding the sequence).
  - to adjust the **pitch**, simply move the cursor to a given slot, then turn the right encoder: this will increase or decrease the pitch in **semitones** (= step size: `1.00`). note these values are **pre-quantization**, so the effect this has will depend on the chosen scale.
  - to **fine-tune** the pitch (= step size: ~ `0.01`), hold down the left encoder **while** turning the right encoder.
  - **note**: the number displayed when adjusting the note values is the actual (pre-quantization) DAC code (that's far from ideal, i know …). details on what those numbers mean can be found here (/custom-scales/).
  - in the sequence-editor, you can use the **up/down** buttons to select which sequence to edit

(`#1 - #4`). (this makes it possible to edit sequences 'offline', ie make adjustments to, say, sequence #2 while sequence #1 is playing).
    - **copy+paste**: in the sequence-editor, long-press the left encoder to **copy** the selected sequence. select a different sequence slot, then long-press the down button to **paste** it into that slot. copy+paste works across channels. (note: once copied (or pasted), sequence data will be stored in a temporary buffer for about 15 seconds, after which it'll expire.)
- the `playmode` and `direction` settings offer various ways of moving through the sequence(s):
    - `-` (default): advance by trigger, using the chosen sequence and direction settings.
    - `SEQ+1 - SEQ+3` : ditto, but **cycling** through 2, 3, or 4 adjacent sequences (= allows up to 64 notes).
    - `TR+1 - TR+3` : ditto, but **jumps** to the next sequence only *if/when* a trigger is received at the **aux.** trigger inputs.
    - `ARP` : **arpeggiate** the sequence.
    - `S+H#1 - S+H#4` : CV-address, when triggered (**sample and hold**).
    - `CV#1 - CV#4` : CV-address, **free-running**.
- note: when in CV-address mode, the `CV adr. range` setting adjusts the resolution of the chosen input (CV1-CV4), relative to the length of the sequence (ie, you'll either need `5V` or `10V` to move through the entire span of the sequence).
- the secondary/aux. clock inputs (TR2, TR4) can be used to either **cycle** through the sequences (see above), or they can be selected as a **reset/mute** signal (`reset/mute`). the options are: `RST2` , `RST4` (reset) and `=HI2` , `=LO2` , `=HI4` , and `=LO4` (mute when aux. clock goes high, respectively low.). NB: When using a clock divider or the like (or anything that will introduce latency relative to the main clock inputs TR1/TR3), it can/will make sense to increase the trigger-to-processing latency (see the `Trigger delay` setting). This will make sure the auxiliary trigger (or control voltage) is processed within the current update window. (Also see *Quantermain* above for an extended discussion of the trigger delay setting).

Inputs and outputs

I/O	Function	-
TR1	clock input #1	-
TR2	aux clock input #1 / reset/mute #1	-
TR3	clock input #2	-
TR4	aux clock input #2 / reset/mute #2	-
CV1	(mappable)	-
CV2	(mappable)	-
CV3	(mappable)	-
CV4	(mappable)	-
A, B	CV outputs #1, #2	-
C, D	aux outputs #1, #2 (default to gate output)	-

Available settings (per-channel)

Setting	Meaning
`scale`	current scale
`--> edit`	edit scale mask (for details see here)
`sequence #`	select sequence `#1 - #4`
`--> edit`	edit sequence
`playmode`	chain sequences (`SEQ+x`), advance by trigger (`TR+x`), or CV-address
`direction`	forward, reverse, pendulum1, pendulum2 (repeat first/last), random, Brownian

Setting	Meaning
`-->brown prob`	(if `direction` = `brwn` (Brownian)) probability (*p*) that sequence direction (up or down will reverse on the next step, 0 means *p*=0, 255 means *p*=1)
`mult/div`	set clock multiplier / divider
`octave`	offset octave
`aux. mode`	aux channel output: `gate` (see note below), `copy` , `AD` , `ADR` , `ADSR` (the last three are envelopes, see below for details)
`CV adr. range`	toggle `5V/10V` (in CV address mode)
`--> pw`	(if `aux. mode` = `gate`) pulse-width of triggers at C/D outputs
`--> aux +/-`	(if `aux. mode` = `copy`) offset at C/D outputs (in octaves)
`--> att dur`	(if `aux. mode` = `AD` , `ADR` or `ADSR`) duration of the attack segment of the envelope output on the auxiliary channels (C or D outputs)
`--> att shape`	(if `aux. mode` = `AD` , `ADR` or `ADSR`) shape of the attack segment of the envelope output on the auxiliary channels (C or D outputs). See the *Piqued* app for details of envelope segment shapes.
`--> dec dur`	(if `aux. mode` = `AD` , `ADR` or `ADSR`) duration of the decay segment of the envelope output on the auxiliary channels (C or D outputs)
`--> dec shape`	(if `aux. mode` = `AD` , `ADR` or `ADSR`) shape of the decay segment of the envelope output on the auxiliary channels (C or D outputs). See the *Piqued* app for details of envelope segment shapes.
`--> sus dur`	(if `aux. mode` = `ADSR`) duration of the sustain segment of the envelope output on the auxiliary channels (C or D outputs)
`--> sus level`	(if `aux. mode` = `ADR` or `ADSR`) level of the sustain segment of the envelope output (for `ADSR` envelopes), or the level at which the the decay segment transitions into the release segment (for `ADR` envelopes) on the auxiliary channels (C or D outputs).
`--> rel dur`	(if `aux. mode` = `ADR` or `ADSR`) duration of the release segment of the envelope output on the auxiliary channels (C or D outputs)
`--> rel shape`	(if `aux. mode` = `ADR` or `ADSR`) shape of the release segment of the envelope output on the auxiliary channels (C or D outputs)
`--> loops`	(if `aux. mode` = `AD` or `ADR` or `ADSR`) sets the number of loops for each envelope output on the auxiliary channels (C or D outputs), defaults to 1 (see also the Tip below)
`reset/mute`	select reset/mute source (mute being more like ‘pause’)
`clock src`	choose channel clock source (`TR1` or `TR3`)
`trigger delay`	trigger-to-processing latency (accessed via CV menu): details see *Quantermain* above

Note: There is a bug in the v1.3.3 firmware which prevents the gate auxiliary output from working unless the `mult/div` for that channel is set to `x2` or above. This bug was fixed in v1.3.3b firmware.

Controls

main menu

Control	Function
Left encoder (turn)	select channel
Left encoder (press)	re-sync channels (reset)
Left encoder (long press)	copy selected scale to other channel
Right encoder (turn)	Navigation mode: move up and down through the menu items. Edit mode: increase or decrease the value being edited

Control	Function
Right encoder (press)	Toggle between menu navigation (selection) mode and value editing mode
Right encoder (long press)	app selection menu
Up button	toggle octave up (parameter menu)
Down button	enter CV menu
Up button (long press)	invoke screensaver
Down button (long press)	toggle sequence-select behaviour (instant update vs. update at end-of-sequence)

CV menu

- **enter** the CV menu by pushing the down button:
  - use the right encoder to assign CV input channels 1-4 to a channel parameter (currently available parameters are: `transposition` (in octaves), `scale mask`, `sequence number` (1-4), `sequence length`, `direction`, `multiplier/divisor` and `pulsewidth`. The quantiser/sampling `trigger delay` (see notes for `Trigger delay` in *Quantermain*) and the `clock src` (clock source) can also be specified in this sub-menu, as well as CV input mappings to these envelope parameters: `att dur`, `dec dur`, `sus lvl`, `rel dur` and `env loops`. By using CV to control `env loops`, a sort of psuedo-ratchetting effect can be obtained. Other envelope behaviours can also be specified (see the *Piqued* app for details): `att rest`, `att fall gt` and `dec/rel reset`.
  - **return** to the main menu by either pressing the **up** or **down** buttons, or by moving the cursor to an empty field and then pressing the right encoder button.
  - **clear** all mappings (per channel) by long-pressing the down button (in CV menu)

Control	Function
Left encoder (turn)	select channel
Left encoder (press)	re-sync channels (reset)
Left encoder (long press)	copy selected scale to other channel
Right encoder (turn)	Navigation mode: move up and down through the menu items. Edit mode: increase or decrease the value being edited
Right encoder (press)	toggle editing mode, if applicable; return to main menu otherwise
Right encoder (long press)	app selection menu
Up button	return to main menu
Down button	return to main menu
Up button (long press)	screensaver
Down button (long press)	clear CV mappings

sequence editor

Control	Function
Left encoder (turn)	select step

Control	Function
Left encoder (press)	activate/de-active step
Right encoder (turn)	1) adjust pitch in semitones; **push left encoder switch** while turning to **fine-tune**. 2) adjust sequence **length** by pointing the cursor to the far right, then turn; **push left encoder switch** while turning to also **clear** the sequence mask (when expanding the pattern)
Right encoder (press)	exit editor
Up button	go to next sequence (edit ‘offline’)
Down button	go to previous sequence (edit ‘offline’)
Left encoder (long press)	**copy** selected pattern
Down button long press	**paste** previously copied sequence
Down button **and** Left encoder long press	**clear** sequence (including pitch data)
Right encoder (long press)	– (app selection menu)
Up button (long press)	– (screensaver)

Tips:

- if identical sequences are set up on both sequencer channels, and the same clock input is used to drive both channels, then Steve Reich “Piano Phase” (https://en.wikipedia.org/wiki/Piano_Phase) type effcts can be acheived by setting the `mult/div` setting for one channel to, say, `/32` (divide by 32), and to `/31` on the other channel, and then using a fast clock. The two sequences will slowly move out of phase. By varying the division up and done in one channel, the two sequences can be moved in and out of phase as required.
- If you set the `aux. mode` to one of the envelope types (as an alternative to just gate or trigger out), then an additional setting `--> loops` will appear in the menu. This setting defaults to 1, but you can increase that, and then each time the envelope triggers, it will loop that set number of times. Tthat gives you a fixed ratchet-like effect, which might be useful in some circumstances.
    - However, in the CV assignment menu (see above), you can assign an input CV to the loops parameter using the `env loops ->` setting. If you do that, and also set `--> loops` to 1 in the main menu and you can then use an external voltage (which could be a gate signal, suitably attenuated) to vary the number of loops for each new step in the sequence between 1 and 127 (usually you would want between 1 and 3 or 4). Of course, the voltage to do that can come from the other channel of Sequins, so you can set the number of envelope loops at each step in channel A by a sequence in channel B, clocked from the same source, and cross-patched to one of the CV inputs configured to affect the loops parameter for channel A. Or you can use external voltages, of course, including controllers such as joysticks etc.
    - You can do something similar in the Piqued app, because there is a Loops parameter for the looping envelope types, and you can put that under external CV control.

The Darkest Timeline 2.0

Jump to bottom

Chysn edited this page on Jun 10, 2022 · 7 revisions

The Darkest Timeline is a CV/MIDI sequencer application for Ornament and Crime. It is included in the Hemisphere Suite firmware from version 1.4.

Functional Concepts

Timelines

Two timelines, each of up to 32 steps, run simultaneously. The timeline at the top of the screen is the Pitch Timeline. The timeline at the bottom of the screen is the Probability Timeline.

Pitch Timeline

The Pitch Timeline controls pitch data. The CV output can be quantized to a specific scale and root note. The MIDI output sends Note On and Note Off messages whose notes are based on the Pitch Timeline. The Pitch Timeline is shown on the top half of the screen, and each step on the timeline is shown as a vertical bar, with taller bars corresponding to higher pitches (or voltages).

Probability Timeline

The Probability Timeline controls the probability of trigger CV firing and of MIDI Note On events being sent. For MIDI notes, the Probability Timeline also determines the velocity of the Note On messages. The Probability Timeline is shown on the bottom half of the screen, and each step on the timeline is shown as a vertical bar, with taller bars corresponding to greater probability of the event being fired (from 0% to 100%).

Length

Set the sequence length by turning the left encoder.

The sequence has a single length value, from 1 to 32 steps. If the sequencer is advanced past the last step, it will return to the first step of the sequence (see "Index" below). If the sequencer is running in reverse, and is reversed past the first step, it will return to the last step of the sequence.

Index

To set the index, press the right encoder to enable the index cursor (a flashing vertical line), and then turn the right encoder to change the index. The index can also be set by providing positive voltage to CV Input 3

Most sequencers have adjustable sequence lengths, but the sequence usually starts at the first step. The Darkest Timeline allows you to choose the sequence's first step, or index, either from the panel or via CV. When the sequence is reset, it will go back to the index. When the sequence is advanced past its last step, it will go back to the index.

The index is shown as a dashed vertical line. Pressing the right encoder button will toggle the right encoder's function between scrubbing through the sequence and setting the index.

Outputs

Pitch Timeline Universe

Outputs A and B are connected to the Pitch Timeline. Output A is the pitch of the left-most step on the display. This is the "Normal Universe" output.

Output B is the "Parallel Universe," which is the corresponding step in the set of steps *after* the last step in the sequence. Okay, that sounds confusing, so here's an example. Let's say you have a length of 8, and your index is step 1. So your Normal Universe is steps 1, 2, 3, 4, 5, 6, 7, and 8. Meanwhile, Output B will play steps 9, 10, 11, 12, 13, 14, 15, 16.

If the length is greater than 16, the Normal and Parallel Universes will share steps. If the length is 32, the Normal and Parallel Universes will be identical.

Probability Timeline Universes

Outputs C and D are connected to the Probability Timeline. When the sequencer is advanced to a step, a probability is calculated based on the value in the Probability Timeline. Output C emits a trigger based on this probability; the higher the bar, the higher the probability of C emitting a trigger. At its maximum, the trigger will always fire. At its minimum, the trigger will never fire. This is the "Normal Universe" probability output.

Output D is the "Alternate Universe," which is a trigger based on the *complementary* probability of the Probability Timeline; if a trigger has a 70% chance of firing in the Normal Universe, it has a 30% chance of firing in the Alternate Universe. The probabilities for both universes are calculated independently, so it's possible for both outputs to fire, or both outputs to not fire on the same step.

MIDI Output Universes

To set MIDI channels, press the left encoder button to enter the Setup screen, and keep pressing the button until you get to the MIDI settings. Then use the left encoder to change the channel.

The Darkest Timeline may send notes independently on up to two MIDI channels. The first of these is the Normal Universe, which is a Note On message that may (or may not) be sent, depending on the probability in the Probability Timeline.

The other MIDI out channel is the Alternate Universe, which uses the Parallel Universe's Pitch Timeline values and the Alternate Universe's Probability Timeline values.

In both cases, the velocity of the Note Out message is proportional to the Probability Timeline value used for the calculation.

Setup Screen

To access the Setup screen, press the left encoder button.

The Setup screen allows you to set the tuning and MIDI settings. Use the left encoder button to move through the settings (and back to the main screen), and use the left encoder to change the selected value. After a short period of inactivity on the Setup screen, you'll be returned to the main screen. Available settings are:

- **Scale**: Chooses one of the User or built-in scales. This will be used to quantize the CV outputs. It does not affect MIDI output.
- **Root**: Sets the diatonic root note based on the selected scale. This will be used to quantize the CV outputs. It does not affect MIDI output.
- **MIDI Out**: Sets the MIDI channel for the Normal Universe MIDI output, as described above. If this is set to Off, there will be no MIDI notes sent from that universe.
- **MIDI Out Alt**: Sets the MIDI channel for the Alternate Universe MIDI output, as described above. This this is set to Off, there will be no MIDI notes sent from that universe.
- **MIDI In**: Determines whether the recording source is CV or MIDI. If you're recording from CV, this should be set to "Off." Otherwise, input for recording will be expected on the specified channel.
- **Gate/Trigger**: Determines whether the Probability Timeline (outputs C and D) will send a 6ms trigger, or a gate that's a percentage of the clock period.

Playback Controls

CV Control of Playback

Inputs:

- Digital 1: A trigger will advance the sequencer forward (by default) or backward
- Digital 2: When gated, a trigger at Digital 1 will move the sequencer backward
- Digital 3: Reset the sequencer to index
- Digital 4: When gated, all probabilities are 100%
- CV 1: CV record value for Pitch Timeline
- CV 2: CV record value for Probability Timeline
- CV 3: Set the index (0V ~ 5V)
- CV 4: Transpose by adding positive or negative voltage to the Pitch timeline

Outputs:

- Output A: CV: Pitch Timeline, Normal Universe
- Output B: CV: Pitch Timeline, Parallel Universe
- Output C: Trigger: Probability Timeline, Normal Universe
- Output D: Trigger: Probability Timeline, Alternate Universe (complementary probability of Output C)

Panel Control of Playback

- Left Encoder: Change sequence length (clockwise to decrease, counterclockwise to increase) from 1 to 32 steps
- Press Right Encoder: Toggle Index editing
- Right Encoder with Index editing off: Scrub through the sequence. If the sequencer is not being clocked with CV 1, calculate probabilities and potentially fire triggers.
- Right Encoder with Index editing on: Change the index. If the sequencer is not being clocked with CV 1, calculate probabilities and potentially fire triggers.

Recording

Recording is enabled independently for each timeline. Press the Up button to enable recording for the Pitch Timeline, and the Down button to enable recording for the Probability Timeline. Pressing the button a second time will disable recording on the corresponding timeline.

CV Recording

If the MIDI In channel is not set on the Setup screen, recording is done via CV. Connect CV sources to CV 1 (for recording into the Pitch Timeline) and/or CV 2 (for recording into the Probability Timeline), and enable recording on one or two timelines, as specified above. When recording is enabled for a timeline, the leftmost step will display a flashing cursor.

When recording is enabled, and set to record CV, a small "CV" icon will appear at the top of the screen.

To record data to the cursor point(s), set the CV at the corresponding input(s), and then advance the sequencer. You can advance the sequencer with a trigger to Digital 1, or by turning the right encoder.

MIDI Recording

If the MIDI In channel is set on the Setup screen, recording is done via MIDI. Connect your O_C module to a MIDI host (usually a computer), choose "Hemisphere" as the MIDI device, and set the output channel to correspond to The Darkest Timeline's MIDI In channel. When recording is enabled for a timeline, the leftmost step will display a flashing cursor.

When recording is enabled, and set to record MIDI, a small MIDI DIN icon will appear at the top of the screen.

To record data to the cursor point(s), play a note. A value proportional to the note number will be recorded to the Pitch Timeline (if recording is enabled), and a value proportional to the velocity will be recorded to the Probability Timeline (if recording is enabled). The sequencer will advance automatically with each note.

Please note that a trigger at Digital 1 will also advance the sequencer, so you probably want to turn off these triggers while recording MIDI data.

Saving The Darkest Timeline Data

The Darkest Timeline's data can be saved in two ways:

(1) **With the built-in Ornament and Crime data storage system** Return to the main menu by long-pressing the right encoder. Then, long-press the right encoder again. This will save the state of all applications in the module.

(2) **System Exclusive Librarian** Your current sequence (both Timelines, length, index, scale, and root) can be dumped to a system exclusive librarian program on a computer for (somewhat) permanent storage. When you long-press the right button to return to the main menu, the sysex dump will be initiated. When setting up your librarian software, select Hemisphere as the MIDI device.

The Darkest Timeline sends your sequence as five sysex messages, so make sure that your librarian captures all of them and stores them in the same file.

To restore your sequence, simply send the dump back any time The Darkest Timeline is running.

Compatibility Note: The Darkest Timeline 2.0 is fully-compatible with sysex files from the previous version. The only difference is that scale and root will not be changed.

The only difference is that scale and root will not be changed.

Misc. Controls

- Long-Press Left Encoder: Randomize both timelines.
- Long-Press Down Button: Clear both timelines.

Waveform Editor

Jump to bottom

Chysn edited this page on Nov 21, 2020 · 4 revisions

Waveform Editor is an application for creating, editing, and saving Vector Oscillator waveforms.

What is a Vector Oscillator Waveform?

Waveforms are described by as few as two, and up to twelve segments. Each segment has a bi-polar *level* (between -128 and 127) and a *time* (between 0 and 9). The level and time are not fixed values, but depend on the application and its settings.

Vector Oscillators are used in four Hemisphere applets:

- VectorLFO, as a cycling bi-polar modulation source
- VectorEG, as a non-cycling uni-polar gated envelope generator with sustain and release
- VectorMod, as a bi-polar triggered modulation source that can be either cycling or non-cycling
- Boots 'n Cats, as an audio oscillator and drum envelope generators

Level

The Level (-128 to 127) is scaled to the specific application. For example, the VectorLFO's range is -3V to 3V. When a Vector Oscillator is started, its signal level is set to the Level of the *last* segment, and immediately starts moving to the Level of the first segment, based on the segment's Time (see below).

Time

The Time of a segment specifies time relative to other segments in the waveform. Two segments with the same Time value will take the same time to complete, regardless of what that value actually is. The total time that it takes a segment to reach its Level will depend upon the Time value (specifically, its ratio to the sum of all Time values in the waveform) and the waveform's frequency (see below).

Frequency

Each Vector Oscillator application provides one or more ways to affect an oscillator's frequency. In most cases, a value in Hertz (cycles per second) is provided, which expresses how fast the oscillator will cycle (if it is cycling). Each segment in the waveform will be scaled based on its Time and the current frequency. Note that Hz values are approximate, due to hardware and software numeric precision limits.

Basic Controls and Navigation

- Left Encoder: Turn to move through the waveform's segments. The selected segment will be a solid line. Press the left encoder button to insert a new segment after the selected segment. Long-press the left encoder button to delete the selected segment. **Note**: You cannot delete a segment if it would leave the waveform with fewer than 2 segments, or if it would leave the waveform with a total Time of 0.
- Right Encoder: Press to switch the cursor between editing of Level and Time. Turn the right encoder to set the selected segment's Level between -128 and 128, or the selected segment's Time between 0 and 9.
- Up/Down Buttons: Press to select a waveform. Long-press the Down button to activate a page to add a new waveform, or delete a waveform. **Note**: You cannot delete the last waveform, and you cannot add a waveform if there are fewer than 3 segments remaining.
- For jack I/O functions, see **Monitoring** below.

Segment Memory

Hemisphere Suite allocates 63 segments of waveform memory. The upper-right corner of the screen shows how many segments are remaining. You may add as many waveforms as you like, with the following limitations:

- At least 3 segments must be available to create a new waveform. This is because each waveform uses one segment as a table-of-contents entry, and each waveform must have at least two Level/Time segments.
- Each waveform may use up to 12 Level/Time segments

Choosing a Waveform

Press the Up and Down buttons to choose a waveform.

Selecting and Editing Segments

Use the left encoder to move through the segments. The selected segment will be shown as a solid line.

Press the right encoder button to alternate the cursor between Level and Time. Turn the right encoder to set that value for the selected segment.

Adding and Deleting Segments

To add a new segment, press the left encoder button. A new segment will be inserted after the selected segment, and the newly-created segment will become selected. You cannot add a segment if the waveform already has 12 segments.

To delete a segment, long-press the left encoder button. The selected segment will be deleted, and the next segment will become selected. You cannot delete a segment if doing so would leave only one segment in the waveform. You also cannot delete a segment if doing so would leave the waveform with a total Time of 0.

Monitoring

You may monitor the waveform you're editing in four different ways, one for each output:

- Out A: Bi-polar LFO (default 1Hz) - *Modulated by CV 1*
- Out B: Bi-polar audio rate oscillator (default 440Hz) - *Modulated by CV 2*
- Out C: Uni-polar gated EG with sustain and release - *Gated by Digital 3*
- Out D: Bi-polar triggered one-shot modulation - *Triggered by Digital 4*

Saving Waveforms

Waveform data can be saved in two ways:

(1) **With the built-in Ornament and Crime storage system** Return to the main menu by long-pressing the right encoder. Then, long-press the right encoder again. This will save the state of all applications in the module.

(2) **System Exclusive Librarian** Your current waveform library can be dumped to a system exclusive librarian program on a computer for permanent storage. When you long-press the right button to return to the main menu, the sysex dump will be initiated. When setting up your librarian software, select Hemisphere as the MIDI device.

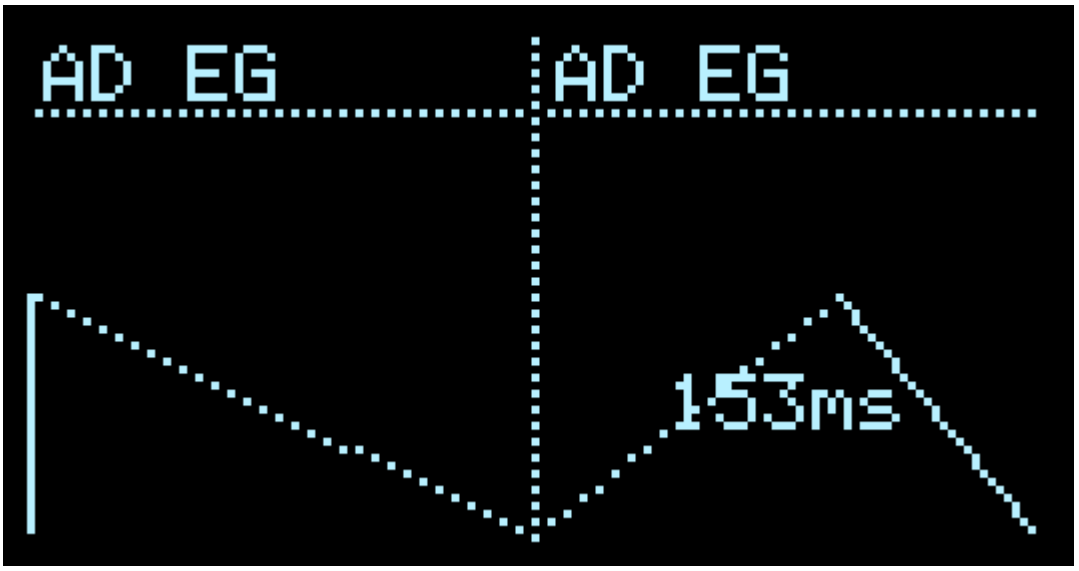
Waveform Editor sends your waveform library as four sysex messages, so make sure that your librarian captures all of them and stores them in the same file.

To restore your waveforms, simply send the dump back any time Waveform Editor is running.

AD EG

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 5 revisions



AD-EG is an attack/decay envelope generator with forward and reverse trigger inputs, and end-of-cycle trigger output.

I/O

	1/3	2/4
TRIGs	Triggers the AD envelope	Triggers the envelope in reverse (DA)
CV INs	Attack modulation (bipolar)	Decay modulation (bipolar)
OUTs	Envelope	End of Cycle trigger

Note that triggering the envelope in reverse does not change the CV modulation controls, meaning that CV 1 controls the effective "attack" segment regardless of direction.

UI Parameters

- Attack duration
- Decay Duration

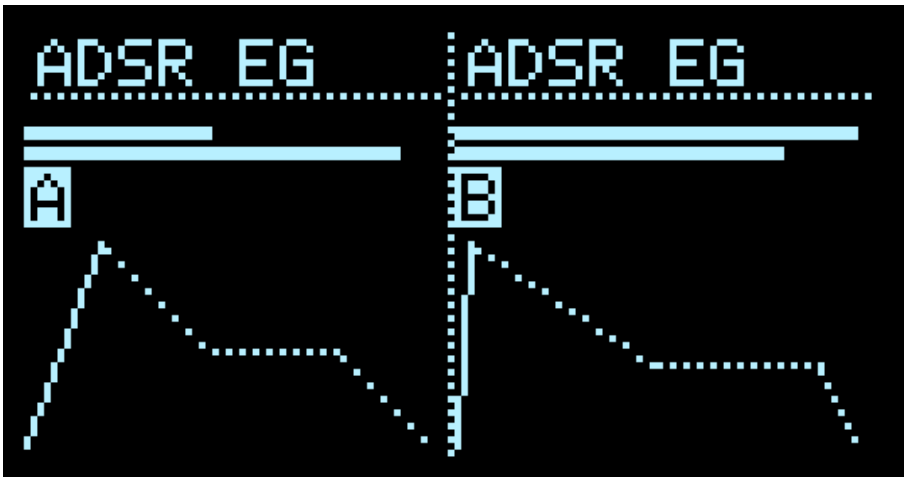
Credits

Adapted from AD EG © 2018-2022, Jason Justian and Beige Maze Laboratories.

ADSR EG

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 4 revisions



ADSR EG is a linear envelope generator with two independent channels.

I/O

	1/3	2/4
TRIG	Channel A Gate	Channel B Gate
CV INs	Env. A Release	Env. B Release
OUTs	Envelope A	Envelope B

Note that CV inputs modulate the release stage over a range of about -2.5 volts to about 2.5 volts. There is a small center detent in the middle of the range, at which point no modification will be made.

UI Parameters

Push encoder to advance cursor; turn to adjust.

- Channel A: Attack duration
- Channel A: Decay duration
- Channel A: Sustain level
- Channel A: Release duration
- Channel B: Attack duration
- Channel B: Decay duration
- Channel B: Sustain level
- Channel B: Release duration

Credits

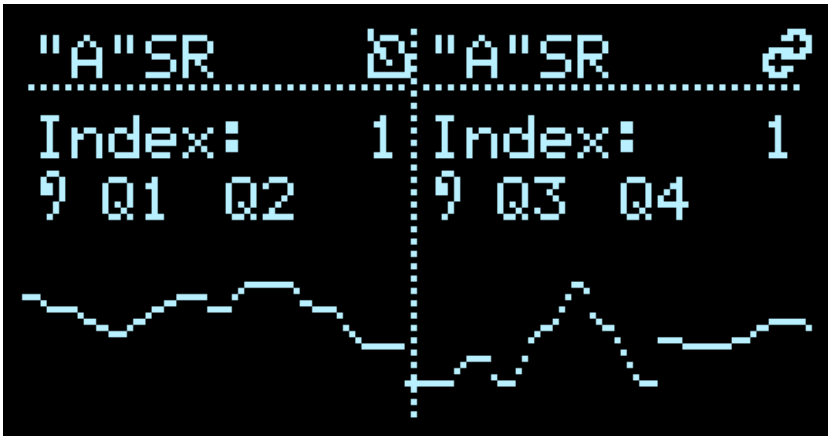
Adapted from the original ADSR EG by Chysn, with modifications by ghostils.

Future improvements may be derived from the subsequent ADSREG_PLUS applet by ghostils.

ASR

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 4 revisions



"A"SR is an "Analog" Shift Register simulation with a 256-step ring buffer.

When selected in one hemisphere (Unlinked Mode), it has two indexed outputs. When selected in both hemispheres (Linked Mode), the same buffer is used across all four outputs and a link icon will appear in the upper right corner of the screen.

I/O (Unlinked Mode)

	1/3	2/4
TRIG	Clock (Advance buffer)	Gate freeze writing buffer
CV INs	Signal to be sampled	Index (n) modulation (bipolar between -32 and 32 steps)
OUTs	Most recent value in buffer	Buffer value n steps back

I/O (Linked Mode)

When linked, the scale may be different for each hemisphere, but the base index will be the same (and may be modulated independently).

	1	2	3	4
Trig (Digital)	Clock (Advance buffer)	No effect	No effect	No effect
CV In	Signal to be sampled	Index (n) modulation	No effect	Index (m) modulation
Output	n steps back in buffer	2n steps back in buffer	3m steps back in buffer	4m steps back in buffer

Note: At an index of 0, all outputs will be the same. When the index value is modulated by CV, a small CV indicator displays below the value.

UI Parameters

- Index
- Edit Quantizer settings, per output

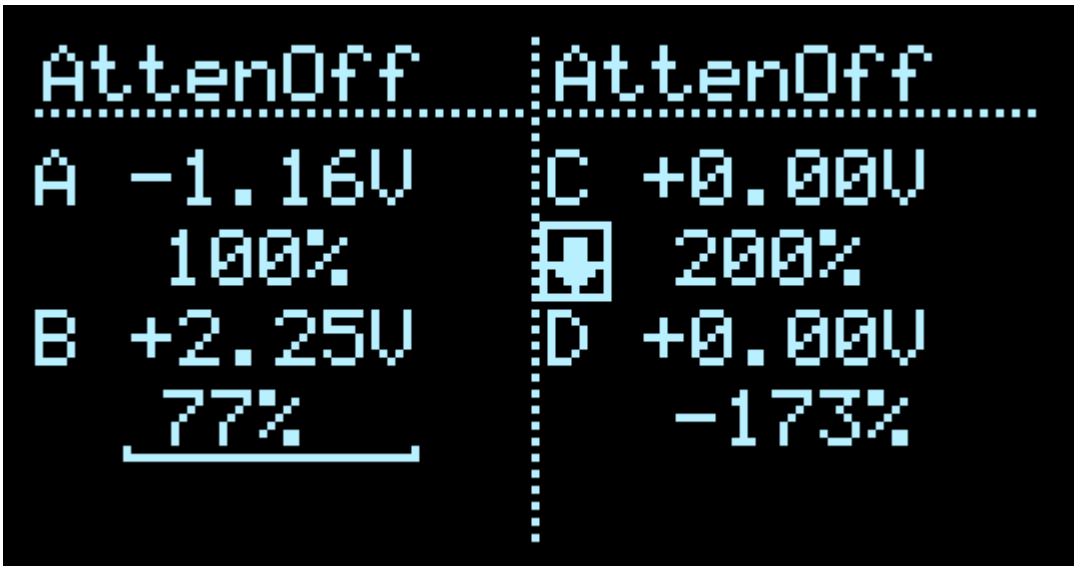
Credits

Adapted from "A"SR © 2018-2022, Jason Justian and Beige Maze Laboratories.

AttenOff

Jump to bottom

bowlneudel edited this page 2 weeks ago · 9 revisions



AttenOff is a dual attenuverter with a -200% to +200% range plus an offset voltage. Offsets are selectable in increments of one semitone, approximately 0.08V.

I/O

	1/3	2/4
TRIG	No effect	Mix Ch 1 into Ch 2 (Gate)
CV INs	Channel 1 Input	Channel 2 Input
OUTs	Attenuverted Ch 1 input + offset	Attenuverted Ch 2 input + offset

UI Parameters

- Ch 1 Offset voltage
- Ch 1 Attenuversion
- Ch 2 Offset voltage
- Ch 2 Attenuversion
- Toggle summing Ch 1 into Ch 2

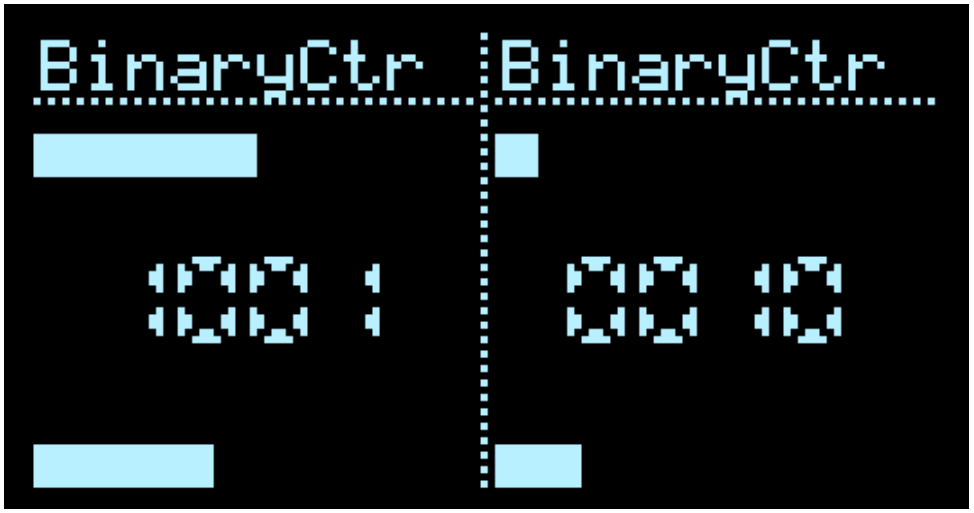
Credits

Adapted from AttenOff © 2018-2022, Jason Justian and Beige Maze Laboratories.

Binary Counter

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 4 revisions



Binary Counter is weird. It takes four logic inputs (high or low) and provides a binary-summed output and a count output.

I/O

	1/3	2/4
TRIG	Bit 3	Bit 2
CV INs	Bit 1	Bit 0
OUTs	Binary sum of inputs	Count of inputs

Binary Sum (Outputs A/C): each bit contributes a value based on its bit number. Approximate voltages are as follows:

- Bit 0 contributes .33V
- Bit 1 contributes .67V
- Bit 2 contributes 1.33V
- Bit 3 contributes 2.67V

Count (Output B/D): Each bit contributes the same value (approximately 1.25V).

UI Parameters

- None

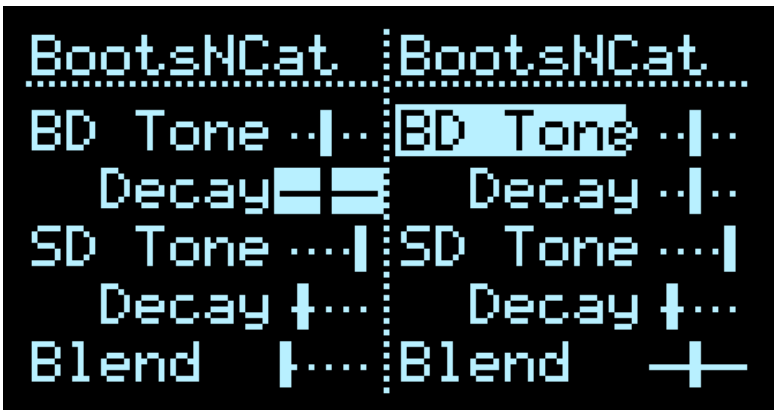
Credits

Adapted from Binary Counter © 2018-2022, Jason Justian and Beige Maze Laboratories.

BootsNCat

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 5 revisions



Boots 'n Cats (BootsNCat) is a simple bass/snare drum synthesizer with stereo audio outputs.

I/O

	1/3	2/4
TRIG	Bass drum trigger	Snare drum trigger
CV INs	Bass drum attenuation	Snare drum attenuation
OUTs	Bass drum (or mix)	Snare drum (or mix)

Note that triggering the envelope in reverse does not change the CV modulation controls, meaning that CV 1 controls the effective "attack" segment regardless of direction.

UI Parameters

- Bass drum tone
- Base drum decay
- Snare drum tone
- Snare drum decay
- Blend

Tone

- Each drum has a Tone control that sets the basic sound of the drum. For the bass drum, Tone controls the frequency, with higher values indicating a higher frequency. For the snare drum, the Tone control is more like a low-pass filter, with high values indicating a higher cutoff frequency.

Decay

- Each drum has a Decay control that sets the decay of the drum between about 1/10 of a second and 1 second, with higher values indicating longer decay.

Blend

- At a Blend level of 0, the bass drum and snare drum are totally isolated on Outputs A/C and B/D, respectively. As the Blend is increased, some of the signal from the opposite channel will be fed into each channel. A Blend level of 63 results in a 50/50 mix on both outputs.

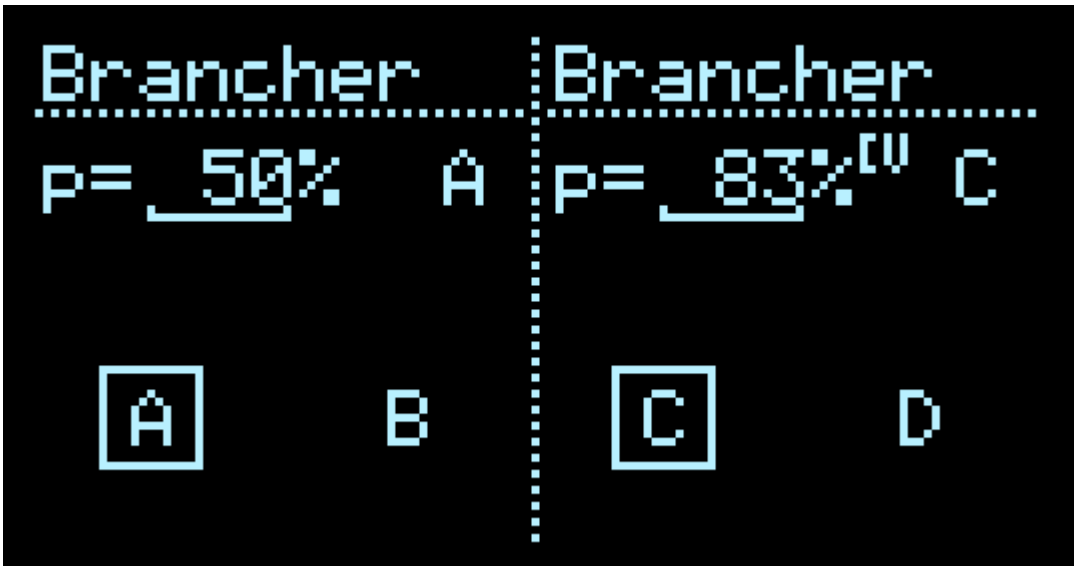
Credits

Adapted from BootsNCat © 2018-2022, Jason Justian and Beige Maze Laboratories.

Brancher

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 4 revisions



Brancher is a Bernoulli Gate, inspired by Mutable Instruments Branches. It sends incoming gate or clock to one of two outputs, based on a selected probability.

I/O

	1/3	2/4
TRIG	Signal (clock or gate) for routing	No effect
CV INs	Probability modulation (bipolar)	No effect
OUTs	Output A/C	Output B/D

UI Parameters

- Probability
- Press encoder to override probability-selected output and switch to the other one

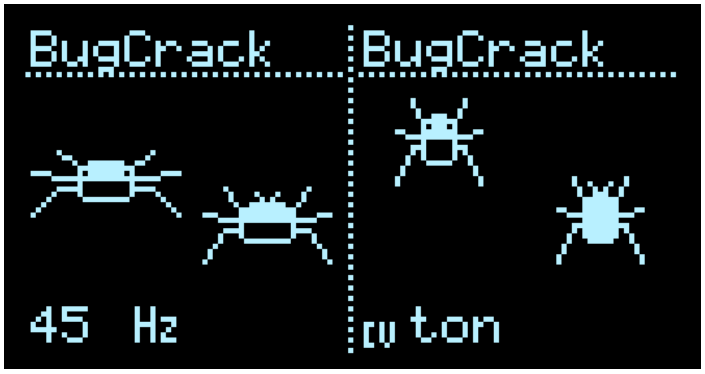
Credits

Adapted from Brancher © 2018-2022, Jason Justian and Beige Maze Laboratories.

BugCrack

Jump to bottom

Nicholas Michalek edited this page on Jun 13 · 5 revisions



Bug Crack is a hi-fi replacement for BootsNCat. It has a nice and boomy kick, as well as a versatile snare/percussion channel. The snare is inspired by the FM drum voice in Peaks, by Mutable Instruments, and can blend between a sine wave oscillator and noise source.

Hear it in action: https://youtu.be/ToZaxJ_7NpM

I/O

	1/3	2/4
TRIG	Kick drum trigger	Snare drum trigger
CV INs	Kick drum modulation (assignable)	Snare drum modulation (assignable)
OUTs	Kick drum	Snare drum

UI Parameters

- Kick tone (frequency)
- Kick decay
- Kick punch
- Kick drop
- CV1 (Kick) assignment (attenuation, tone, decay, FM, drop)
- Snare tone (frequency)
- Snare decay
- Snare snap
- Snare blend
- CV2 (Snare) assignment (attenuation, tone, decay, FM, blend)

Frequency (ton) - Kick and Snare The base frequency of the drum. Kick has a range of 30-60Hz, snare has a range of 100-700Hz.

Decay (dec) - Kick and Snare The time in takes for the sounds to fade out. On the snare, this parameter also affects the Snap time.

Punch (FM) - Kick Punch causes a quick pitch drop at the beginning of the sound that's common in kick drums. Punch controls how high up the pitch drop starts from.

Drop (dro) - Kick Drop controls the time it takes to get from the Punch value to the Frequency value.

Snap (FM) - Snare Snap is equivalent to the Punch parameter. It causes a pitch drop at the beginning of the sound, at the higher frequencies of the snare range it can create a nice snapping sound.

Blend (bln) - Snare Blend between the oscillator and noise.

CV Assignment In addition to all of the parameters above, the CV inputs can also be assigned to attenuation (atn). A positive voltage increases the attenuation, meaning it gets more quiet. This is opposite to what a "volume" parameter would do, but is required due to constraints of the o_C.

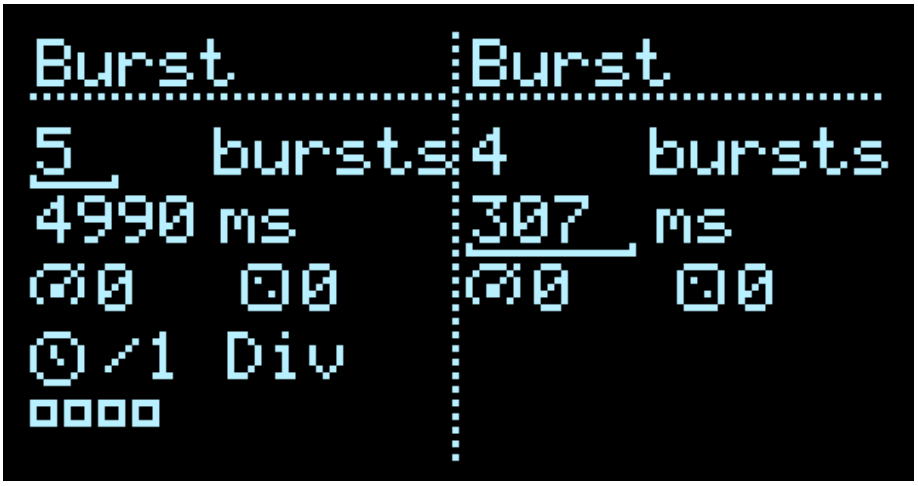
Credits

Written by benirose & herrkami, with mods by djphazer. Adapted from BugCrack

Burst

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 6 revisions



Burst is a burst generator based loosely on the Ladik S-075.

I/O

	1/3	2/4
TRIG	Clock (optional)	Start burst sequence
CV INs	Number of triggers per burst	Spacing (bipolar)
OUTs	Burst triggers	Burst gate (high until end of burst)

UI Parameters

- Triggers per burst (1-12)
- Spacing (8ms-500ms) [Clock input overrides]
- Acceleration/deceleration (positive values space later triggers closer together)
- Randomization (humanizes trigger timing)
- Clock input multiplication/division

Notes

If Burst receives another trigger at Digital 2 before the burst sequence has finished, a new sequence will start.

When CV is received at CV 1, the Number setting at the panel is overridden. There is a center detent around 0V, so slightly more than 0V is required to change the Number.

When CV is received at CV 2, the Spacing setting at the panel is modulated. There is a center detent around 0V, so slightly more than 0V is required to change the Spacing.

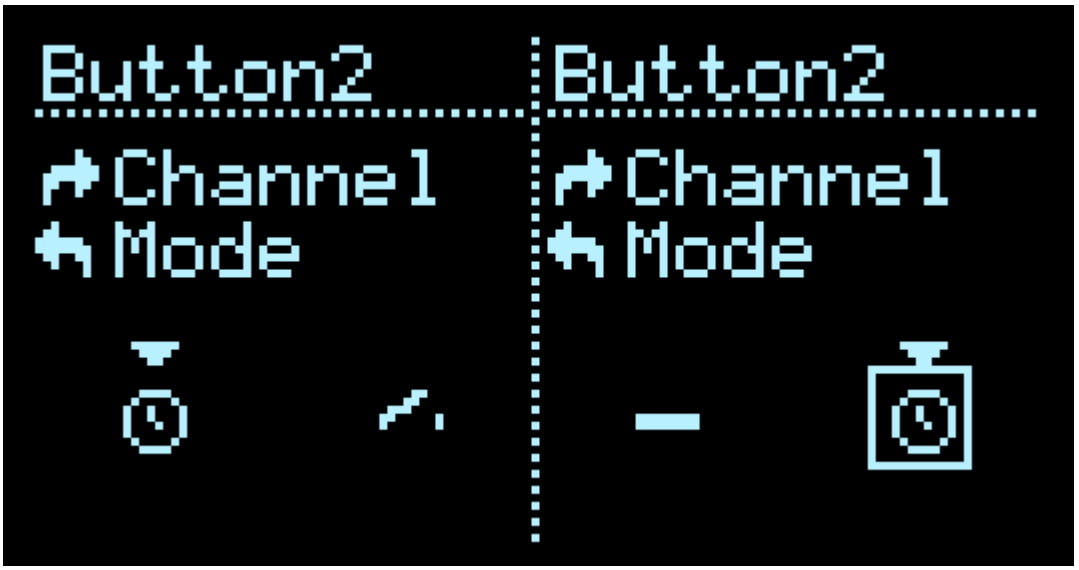
Credits

Adapted from Burst © 2018-2022, Jason Justian and Beige Maze Laboratories.

Button2

Jump to bottom

Nicholas Michalek edited this page on Jun 13 · 6 revisions



Ever just need a button, to trigger something or toggle something? Me too! Button functions as a dual trigger and/or latching toggle (both 5V when high).

I/O

	1/3	2/4
TRIG	Press button 1	Press button 2
CV INs	No effect	No effect
OUTs	Trigger/Gate 1	Trigger/Gate 2

UI Parameters

- Channel selection (scroll clockwise)
- Button mode: trigger, latch (scroll counterclockwise)
- Press selected button (push encoder)

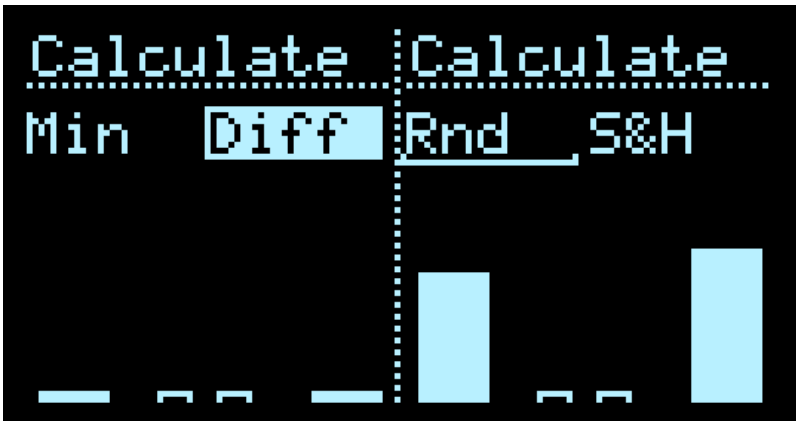
Note: Button2 overrides your Hemisphere cursor mode. In this applet, scrolling clockwise switches between selecting button channels, scrolling counterclockwise switches the selected channel between a trigger mode (clock icon) and a latching toggle mode (switch icon)

Adapted from Button © 2018-2022, Jason Justian and Beige Maze Laboratories.

Calculate

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 7 revisions



Calculate can perform two separate arithmetical operations on two CV inputs: each outputting either the minimum voltage, the maximum, the sum, the difference, or the average. It can also function as a basic Sample & Hold or a Random CV noise source.

I/O

	1/3	2/4
TRIG	Ch1 Clock (for Rnd or S&H modes)	Ch2 Clock (for Rnd or S&H modes)
CV INs	Operand A	Operand B
OUTs	Channel 1 calculation results	Channel 2 calculation results

UI Parameters

- Operator A
- Operator B

The following operators are available:

- **Min:** The result is the lower voltage of the two operands
- **Max:** The result is the higher voltage of the two operands
- **Sum:** The result is the sum of the two operands, up to 5 volts
- **Diff:** The result is the absolute value of the difference between the two operands
- **Mean:** The result is the mean (average) of the two operands (that is, Sum divided by 2)
- **S&H:** When the channel is clocked, the output is held at the voltage of the corresponding input.
  - *Note: the input will still provide an unclocked operand for the other calculation*
- **Rnd:** A random value between 0 and 5 volts appears at the corresponding output.
  - When the channel is clocked, a random value is held at the corresponding output until the next clock. To reset the input (that is, to turn off clocked operation), place the cursor on the Rnd operator and turn the encoder. If you turn the encoder clockwise, clocked operation will be turned off without changing away from the Rnd operator. When Rnd is clocked, a clock icon will appear to the right of the operator selector.
  - When "Rnd" is selected for both channels, Clock on channel 1 is normalized to channel 2.
  - *Note: the corresponding input will have no effect on the random output, but will still provide an unclocked operand for the other calculation*

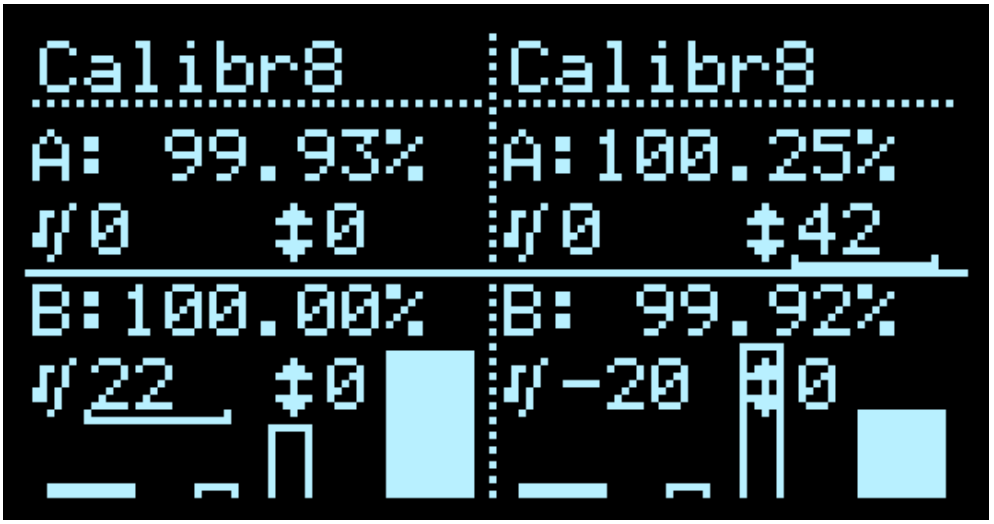
Credits

Adapted from Calculate © 2018-2022, Jason Justian and Beige Maze Laboratories.

Calibr8

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 6 revisions



Calibr8 is a 2 channel pitch calibration tool to compensate for imperfect v/Oct tracking in your oscillators. For a 4 channel version with many more features (flexible quantization, sample and hold, auto-calibration), use the full Calibr8or app.

I/O

	1/3	2/4
TRIG	Transposition clock	No effect
CV INs	Pitch CV input 1	Pitch CV input 2
OUTs	Calibrated pitch CV 1	Calibrated pitch CV 2

Note: the outputs of Calibr8 are quantized to semitones

UI Parameters

- Pitch CV1 scaling factor
- Pitch CV1 transposition
- Pitch CV1 offset (detune)
- Pitch CV2 scaling factor
- Pitch CV2 transposition
- Pitch CV2 offset (detune)

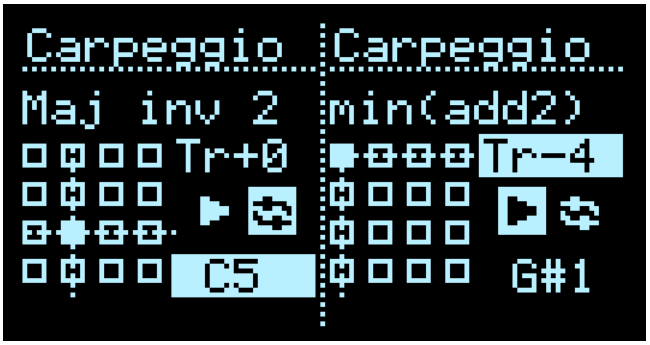
Credits

Authored by djphazer, as a proof-of-concept for Calibr8or

Carpeggio

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 6 revisions



Carpeggio is an arpeggiator/sequencer that uses coordinates on a 4x4 Cartesian plane to select notes. Notes can be assigned to the plane from among 55 chord patterns. Carpeggio can also be clocked and reset, to function as a more traditional arpeggiator. Additionally, any of the 16 notes may be edited on the fly, allowing Carpeggio to function as a Cartesian or traditional 16-step sequencer.

I/O

	1/3	2/4
TRIG	Advance to the next step (x, then y)	Reset to (1,1)
CV INs	X position	Y position
OUTs	Quantized pitch CV	Modulation output proportional to x*y

UI Parameters

- Chord
- Transposition
- Toggle random/linear note order
- Edit current note

Note: edited notes are shuffled when random note order is toggled. However, returning to linear note order erases any note edits.

Cartesian Operation

Provide x and y CV (0-5V) to the CV inputs. The position will be displayed with cross hairs on the grid. To select the note at the current position for playing, provide a clock signal to Digital 1.

Linear Operation

To use Carpeggio as a more traditional linear arpeggiator or sequencer, disconnect the x and y CV (or set them both to 0V). A clock to Digital 1 will play the note at the current position, and then advance to the next position, from left to right, and top to bottom.

Note Editing

When the cursor is flashing under the note number near the bottom of the screen, the note at the current step may be edited with the encoder. Such edits will be lost on power-down, or when the chord is changed.

Changing Chords

When the cursor is flashing under the chord name, select a chord with the encoder. If you change the chord and push the encoder, the sixteen steps will be filled with notes from that chord, removing any note edits you've made. When you're changing chords, a checkmark will appear next to the currently-selected chord. If you press the encoder button at this chord, your note changes will not be overwritten.

Transposition

Carpeggio has a four octave transposition range, 24 semitones in either direction.

Credits

Thanks to Roel Das for writing the chord pattern code, and generously allowing it to be used here.

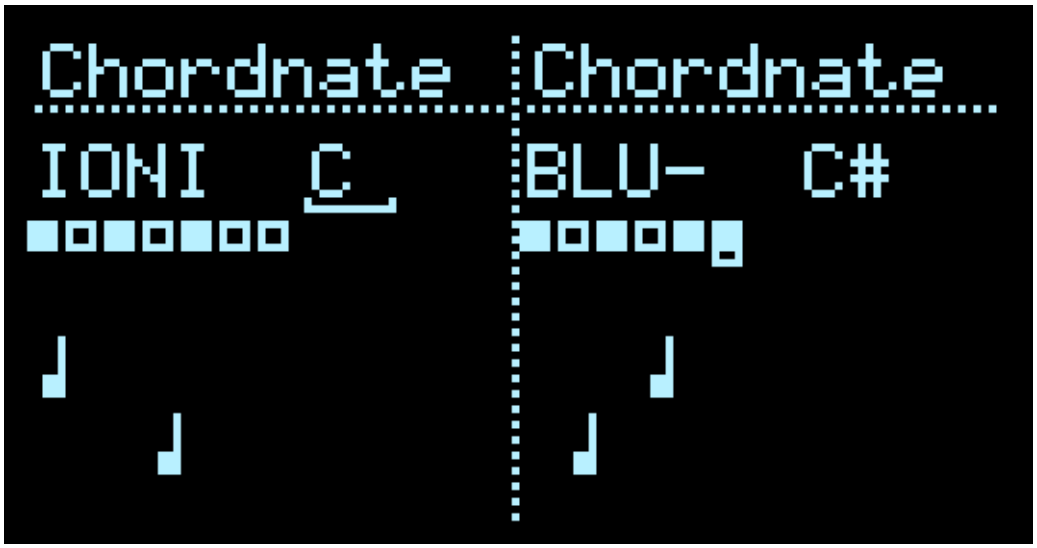
Adapted from Carpeggio © 2018-2022, Jason Justian and Beige Maze Laboratories.

Shuffled operation added by benirose

Chordinate

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 8 revisions



Chordinate is a quantizer with a scale mask: outputting a basis pitch (i.e. the root note, notwithstanding any transposition) and a scale degree offset. The scale mask allows only user-determined intervals relative to the root note to be a valid output on CV 2/4.

I/O

	1/3	2/4
TRIG	Sample & hold root note (CV 1)	Sample & hold scale degree (CV2)
CV INs	Basis pitch	Scale degree offset (bipolar)
OUTs	Basis pitch (thru)	Scale degree pitch

Note that sending a trigger to either digital input will initiate a clocked S&H mode, which will persist until power cycling. Root note and scale degree may be held independently, allowing for syncopated voicing changes

UI Parameters

- Quantizer scale
- Root note (affects output 2/4 only)
- Scale mask

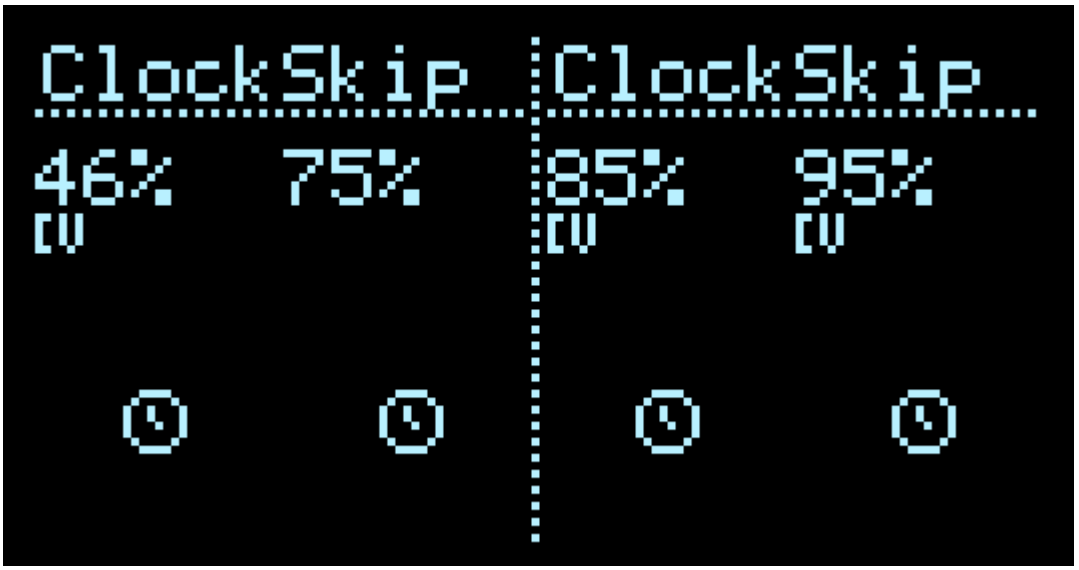
Credits

Chordinate is authored by qiemem

Clock Skipper

Jump to bottom

Nicholas Michalek edited this page on Jun 13 · 4 revisions



Clock Skipper is a dual probability skipper, with independent inputs.

When an incoming clock is received at the digital input, that clock may or may not be sent to the corresponding output, depending on the probability. A probability of 100% means that the gate will always be passed along. Probability can be modified with voltage to the CV inputs. Negative voltage decreases probability, and positive voltage increases probability.

I/O

	1/3	2/4
TRIG	Channel 1 Clock	Channel 2 Clock
CV INs	Ch 1 Probability modulation (bipolar)	Ch 2 Probability modulation (bipolar)
OUTs	Ch 1 Clock	Ch 2 Clock

UI Parameters

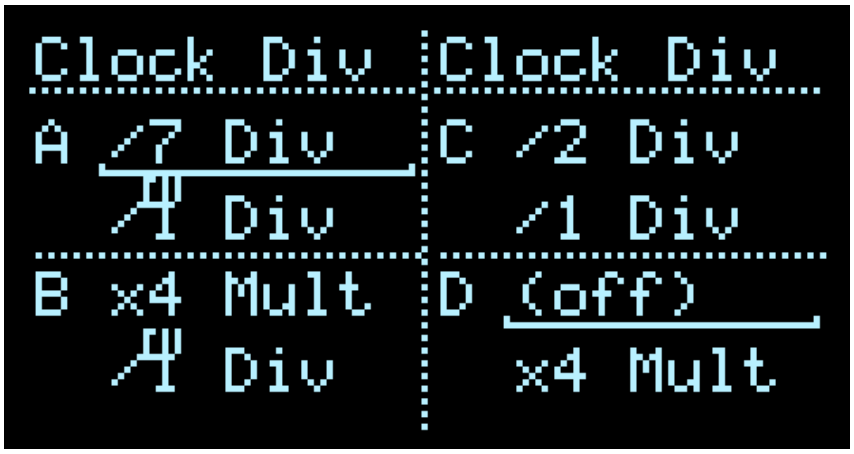
- Ch 1 Probability
- Ch 2 Probability

Adapted from Clock Skipper © 2018-2022, Jason Justian and Beige Maze Laboratories.

ClockDivider

Jump to bottom

Nicholas Michalek edited this page on Jun 13 · 10 revisions



Complex clock divisions / multiplications made easy! **ClockDiv** is a dual, complex clock divider and multiplier, using a single clock source. Each stage can divide from 1:1 to 1:32 (for a maximum of 1:1024), or multiply from 1:1 to 32:1 (for a theoretical maximum of 1024:1 — although in practice you're unlikely to exceed 180:1 without strange behaviour). Voltage control is available for the first stage of each channel (displaying a CV icon when engaged).

I/O

	1/3	2/4
TRIG	Clock input	Reset
CV INs	Ch 1 div/mult factor 1 modulation (bipolar)	Ch 2 div/mult factor 1 modulation (bipolar)
OUTs	Ch 1 div/mult result	Ch 2 div/mult result

UI Parameters

- Ch 1 div/mult factor 1
- Ch 1 div/mult factor 2
- Ch 2 div/mult factor 1
- Ch 2 div/mult factor 2

Notes

Division works by counting clocks and sending a trigger on the _n_th clock. When the encoder is turned, the counter is reset.

Multiplication works by counting interrupt cycles (c) between the last two clock inputs, and sending a trigger every c/n cycles.

CV inputs are bipolar, with a range of about -2.5 volts to about 2.5 volts. Positive values set clock division, and negative values set clock multiplication. There's a small center detent CV range, over which range control is only set from the panel. This is done because O_C can't sense whether a jack is patched.

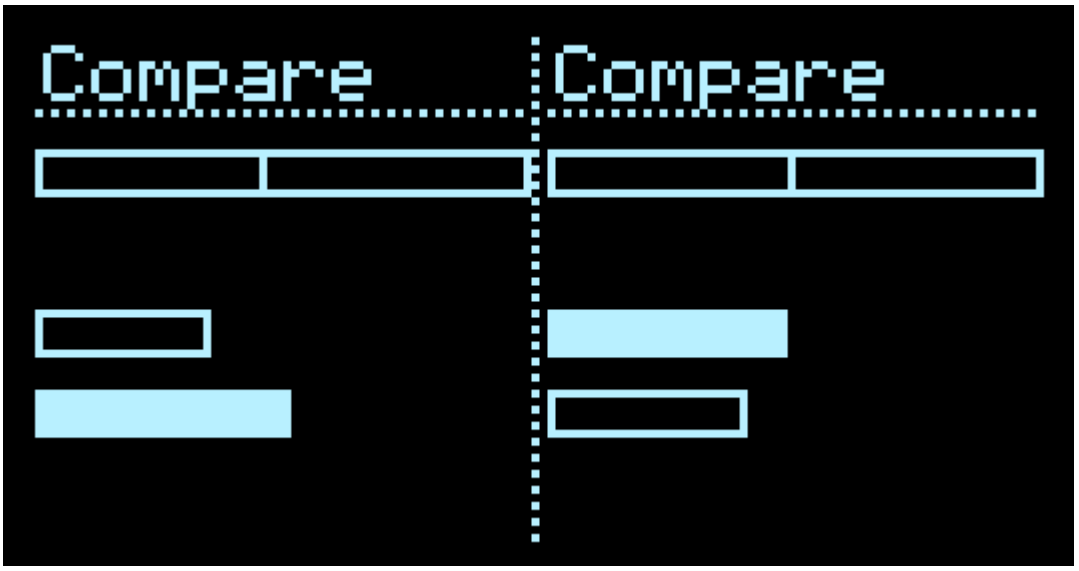
Credits

Adapted from Clock-Divider-Multiplier © 2018-2022, Jason Justian and Beige Maze Laboratories...

Compare

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 5 revisions



Compare is a comparator applet with complementary gate outputs, indicating when an input voltage is greater or less than a set (or modulated) threshold.

I/O

	1/3	2/4
TRIG	No effect	No effect
CV INs	Input signal (A)	Comparison voltage level (B) modulation (bipolar)
OUTs	Gate if A>B	Gate if A≤B

UI Parameters

- Comparison voltage level (B)

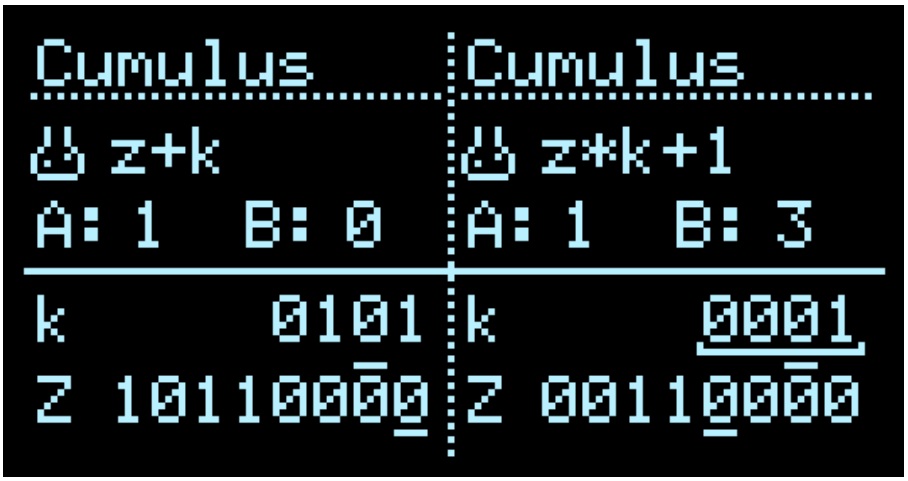
Credits

Adapted from Compare © 2018-2022, Jason Justian and Beige Maze Laboratories.

Cumulus

Jump to bottom

Nicholas Michalek edited this page on Jun 13 · 3 revisions



Cumulus is a 8-Bit operation machine inspired by Nibbler from Schlappi Engineering.

Whenever the a gate goes high at 1/3 the applet performs an operations of the output accumulator (Z) and with the constant (k), the result is saved back into the accumulator (Z). The constant value can either be set by encoder using the menu oand/or modulated by the CV in 2/4. The value of A and B determine which bit of the accumulator is routed to the outputs. The value of A and B are set using the menu and A is modulated using CV 1/3.

I/O

	1/3	2/4
Trig (Digital)	Execute Instruction	Randomize Accumulator
CV In	Modulation A location	Modulation k Value
Output	Bit at A's position	Bit at B's position

UI Parameters

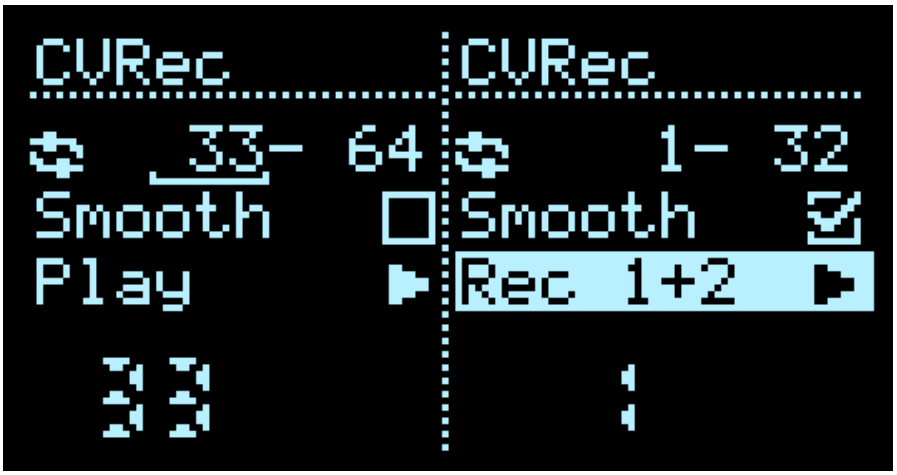
- Instruction
  - `z+k`
  - `z-k`
  - `z*k+1`
  - `(z^k)<<1` (z xor k) and rotate left by 1
  - `(z-k)>>2` (z - k) and rotate right by 2
  - constant value k
- A location
- B location
- k value

Authored by Zerbian. In Phazerville from v1.7.1

CV Recorder

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 4 revisions



CV Recorder is a two-track 384-step CV recorder with smoothing (linear interpolation) and adjustable start/end points.

I/O

	1/3	2/4
TRIG	Advance sequencer	Reset
CV INs	Track 1 Input	Track 2 Input
OUTs	Track 1	Track 2

UI Parameters

- Start point
- End point
- Toggle smoothing on/off
- Transport control
  - Play
  - Record Track 1
  - Record Track 2
  - Record Track 1 + 2

Recording To start recording, choose a length by setting the Start and End points. Advance the cursor down to the transport control (which will usually just say "Play"). Turn the encoder to choose which track or tracks you wish to record (1, 2, 1+2). When you press the encoder button, recording will begin. An indicator bar will display on top of the transport control line to indicate remaining steps. Recording will automatically stop once the end point step has been reached, and looping play mode will resume.

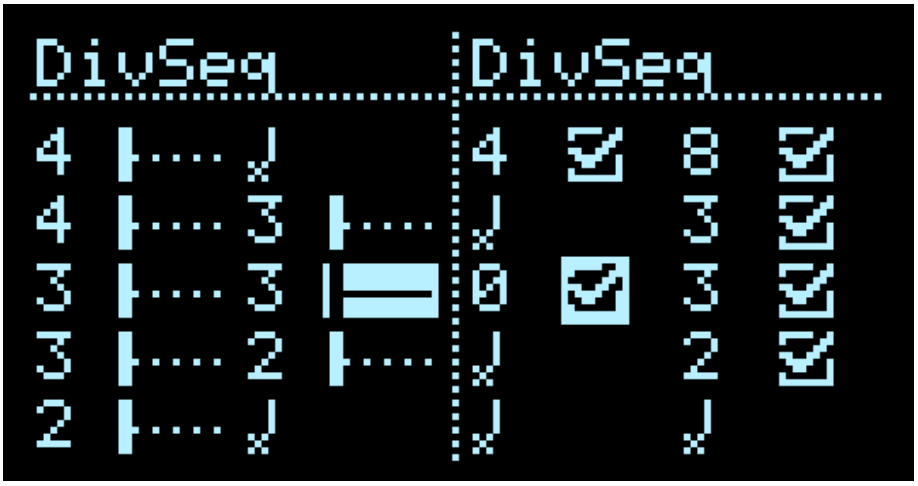
Credits

Adapted from CV Recorder © 2018-2022, Jason Justian and Beige Maze Laboratories.

DivSeq

Jump to bottom

Nicholas Michalek edited this page on Jun 13 · 9 revisions



DivSeq is a dual sequential clock divider with a single input clock. Each of the two channels is composed of a sequence of up to 5 clock dividers. Under normal operation, a step "n" triggers 1 clock pulse, skips n-1 pulses, and then steps to the next clock divider.

Via the CV inputs, each channel may also be in inverted mode (skip 1, trigger n-1 pulses, step) with positive voltage, or cross-channel XOR mode with negative voltage. There is a virtual detent around 0v for normal mode.

Performance mutes may be manually toggled for each divider step. Each step may be set to a divider of 0 (off) up to 63.

See it in action: https://youtu.be/J1OH-oomvMA?si=Z97wJ3HXe0HocaBa&t=357

I/O

	1/3	2/4
TRIG	Clock	Reset
CV INs	Ch 1 mode: inverted (positive) XOR (negative)	Ch 2 mode: inverted XOR
OUTs	Ch 1 Trigger sequence	Ch 2 Trigger sequence

UI Parameters

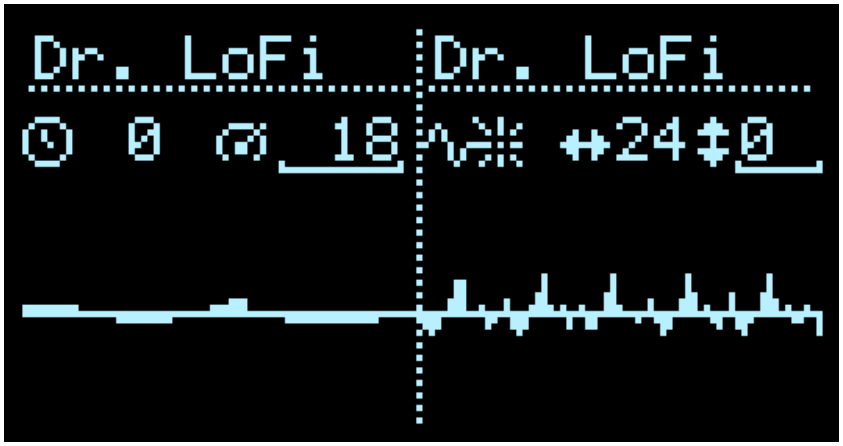
- Ch 1 divider value steps 1-5
- Ch 2 divider value steps 1-5
- Ch 1 performance mutes steps 1-5
- Ch 2 performance mutes steps 1-5

With a step highlighted for editing, the AuxButton action will also toggle mute

Dr. LoFi

Jump to bottom

Nicholas Michalek edited this page on Jun 13 · 2 revisions



(Formerly called LoFi Echo)

Dr. LoFi is a friend of Dr. Crusher, who built a digital delay line using LoFi Tape. It still provides sample rate and bit depth reduction for CV or audio signals, along with a feedback loop with configurable delay and depth.

I/O

	1/3	2/4
TRIG	Pause/Freeze	Feedback=100
CV INs	Signal Input	Rate mod
OUTs	Signal Output	Reverse buffer output

UI Parameters

- Delay Time - set to 0 for passthru with digital distortion
- Feedback
- Rate Reduction - 1 is fastest (16.6khz); higher is slower / lower sampling rate
  - some smoothing is applied as rate reduction increases
- Bit Depth Reduction - 0 is full-range (sampled at 12-bit); higher truncates more bits

Notes

There is one global 8-bit PCM buffer shared by both Hemispheres, which has strange and experimental implications when running the applet on both sides simultaneously.

Out A/C is the main output; Out B/D is the buffer played in reverse (weird but cool)

Note that some "crushing" will always be applied, no matter what. This is a digital module processing signals at 16.6khz, you're going to get aliasing and noize!

If you're not getting any signal at lower bit resolutions, increase the input amplitude.

Credits

Adapted from LoFi Tape by Chysn. Reworked by djphazer, based on a digital delay line idea by armandvedel.

DrumMap

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 4 revisions

DrumMap is a 2 channel port of the popular rhythm sequencer Grids, from Mutable Instruments. You are able to select any of the 3 parts to assign to each channel. In addition, the 2nd channel can be used as the accent channel of the 1st channel. Traverse a map of rhythms by X and Y coordinates!

DrumMap expects a 32nd note clock (8ppqn) to operate normally, but can use other clock speeds to advance the sequencer at slower or faster rates.

If you are unfamiliar with the premise of Grids, watch this video for an excellent explanation: https://www.youtube.com/watch?v=I5yN0N6aTws

Thanks to Émilie Gillet for making the code for Mutable Instruments modules open source!

See it in action: https://youtu.be/yLr3vQJm3wM

I/O

	1/3	2/4
TRIG	Clock	Reset
CV INs	Assignable (Fill A, X position,)	Assignable (Fill B, Y Position, Chaos)
OUTs	Ch 1 Trigger sequence	Ch 2 Trigger sequence

Note that alternate rhythm patterns are available with the build flag "DRUMMAP_GRIDS2"

UI Parameters

- Ch 1 Drum (Kick, snare, high hat)
- Ch 2 Drum (Kick, snare, high hat, accent)
- Ch 1 Fill
- Ch 2 Fill
- X position
- Y postion
- Chaos
- CV assignment (X/Y, Fill A/B, Fill A/Chaos)

In order to make parameter changes more performable using the encoder, knob acceleration has been implemented for the "Fill", "X", "Y" and "Chaos" parameters. This means if you turn the knob quickly it will traverse the whole range, but if you turn it slowly it will increment by a single value.

Additionally, DrumMap will auto-reset after ~2 seconds of no clock signal.

Assignable CV Inputs

You can change the CV Input assignment from “Part A Fill and Part B Fill”, “X and Y coordinates” and “Part A Fill and Chaos” (the last is useful when you’re using accent on channel 2, since it has no fill). CV is added to the parameter value set via the encoder, and voltages from -3 to +5 are supported. This means when the parameter is set in the middle, you should be able to use a +/-5 volt CV signal to reach both ends of the parameter.

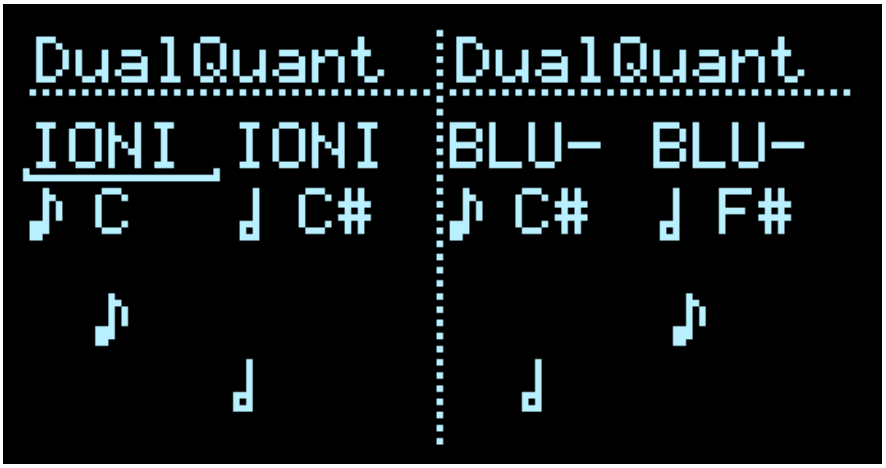
Credits

Adapted from DrumMap by benirose. Alternate "Grids2" patterns collected by KittenVillage.

Dual Quantizer

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 5 revisions



Dual Quantizer converts 2 independent CV inputs to pitches of specific musical scales. Clocked (sample & hold) operation is available for each channel.

See it in action: https://youtu.be/GkV-rB4ntho

I/O

	1/3	2/4
TRIG	Ch 1 Clock	Ch 2 Clock
CV INs	Ch 1 Unquantized CV	Ch 2 Unquantized CV
OUTs	Ch 1 Quantized CV	Ch 2 Quantized CV

Note: Each channel begins in continuous operation. That is, the incoming CV is quantized at a rate of about 16667 times per second. This might result in undesirable slippage between notes, so clocked operation is available on a per-channel basis.

To enter clocked operation, send a clock signal to a digital input. A clock icon will appear next to the selector for the corresponding channel to indicate that that quantizer is in clocked mode.

To return to continuous operation, stop sending clock to the channel, and then change the scale. As long as no additional clock signals are received at the channel, the quantizer will remain in continuous operation.

UI Parameters

- Ch 1 Scale
- Ch 1 Root
- Ch 2 Scale
- Ch 2 Root

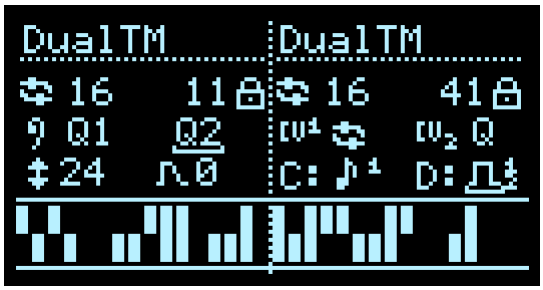
Credits

Adapted from Dual Quantizer © 2018-2022, Jason Justian and Beige Maze Laboratories.

DualTM

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 10 revisions



Adapted from the original ShiftReg applet, this pair of 32-bit shift registers is designed to be the ultimate source of generative sequences. Digital inputs 1 & 2 are still *Clock* & *p-gate*, respectively. The CV inputs and outputs are assignable.

I/O

	1/3	2/4
TRIG	Clock	p Gate
CV INs	Assignable (Slew, Length, Probability, Quantizer, Range, Transpose, Crossfade)	Assignable
OUTs	Assignable (Pitch, Mod, Trig, Gate)	Assignable

The **Slew** parameter allows extreme smoothing on the output stage for portamento and gentle modulation. It acts as a Decay tail on the Trigger output modes. With CV input modulation of Slew, you can modulate the smoothing - CV control over decay envelopes, or variable portamento.

Parameters:

- Length - how many bits are looped in the registers
- p=Probability (%) - when unlocked with the cursor or a gate input on TR2, how likely the current bit will be flipped
- Quantizer select (v1.7.1) - for pitch quantization
- Range - how many discrete notes, in scale degrees (only applies to pitches)
- Slew - smoothing parameter (hybrid linear/logarithmic function)
- Input modes CV1 and CV2 - see below
- Output modes A: and B: - see below

Output Modes

Each output can be assigned to one of the following:

- Pitch 1 - derived from 8 bits of the FIRST register
- Pitch 2 - derived from 8 bits of the SECOND register
- Pitch 1+2 - a blend of the two pitches, optionally crossfaded via CV
  - in 1.5.x, it was actually the SUM of the two pitches; changed in v1.6
- Mod 1 - unquantized bipolar modulation from FIRST register
- Mod 2 - unquantized bipolar modulation from SECOND register
- Trigger 1 - output trigger pulse when current bit is 1 on the FIRST register
- Trigger 2 - output trigger pulse when current bit is 1 on the SECOND register
- Gate 1 - hold high if current bit is 1 on FIRST register
- Gate 2 - hold high if current bit is 1 on SECOND register
- Gate 1+2 - two-level gate output from the sum of the current bits

Input Modes

Each CV input can be assigned to modulate (bi-polar) one of the following:

- Slew
- Length
- p - probability
- Q - Quantizer select
- Range
- Transpose 1 - offset Pitch 1 (basic voltage adder)
- Transpose 2 - offset Pitch 2
- Crossfade 1+2 - Affects the combo Pitch 1+2 Output mode only
  - at 0V or unpatched, the two pitches are averaged
  - Positive voltage fades toward Pitch 2; negative voltage toward Pitch 1 (i think, lol, somebody should check my math)

Credits

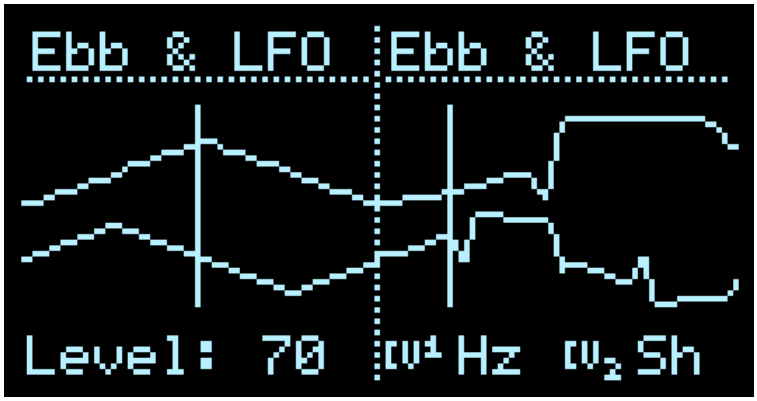
Authored by djphazer. Adapted from ShiftReg by Chysn, with mods by benirose.

Inspired by the original Turing Machine by Music Thing Modular / Tom Whitwell.

Ebb & LFO

Jump to bottom

Nicholas Michalek edited this page on Jun 13 · 4 revisions



This is a single Tides-like oscillator with two outputs. Both inputs and outputs are configurable. The outputs are visualized on the screen.

It can function like a looping envelope generator or an audio-rate oscillator, with modulatable parameters for morphing the contour. Triggers to input 1/3 may function as tap tempo. "One Shot" mode will output a single waveform cycle and stop, much like the AD envelope mode on Tides (use Reset to retrigger).

I/O

	1/3	2/4
TRIG	Clock	Reset
CV INs	Assignable (Freq / Slope / Shape / Fold)	Assignable (Amp / Slope / Shape / Fold)
OUTs	Waveform A	Waveform B

CV Inputs & UI Parameters

These are also the corresponding on-screen parameters. Each input is assigned to one of:

- Level / **Am**plitude (CV2 only)
- Frequency (**Hz**) (CV1 only) - tracks V/Oct for audio-rate operation
- **Sl**ope - skews the waveform (much like the deprecated SkewedLFO applet)
- **Sh**ape - morphs between various curved segment combos
- **Fo**ld - wavefolder!

Lastly, there is a UI toggle for *One Shot* mode.

Outputs

Each output can be one of:

- **Un**ipolar
- **Bi**polar
- Gate **Hi**gh - starts low, goes high at the end of the Attack phase
- Gate **Lo**w - starts high, goes low at the end of the Attack phase

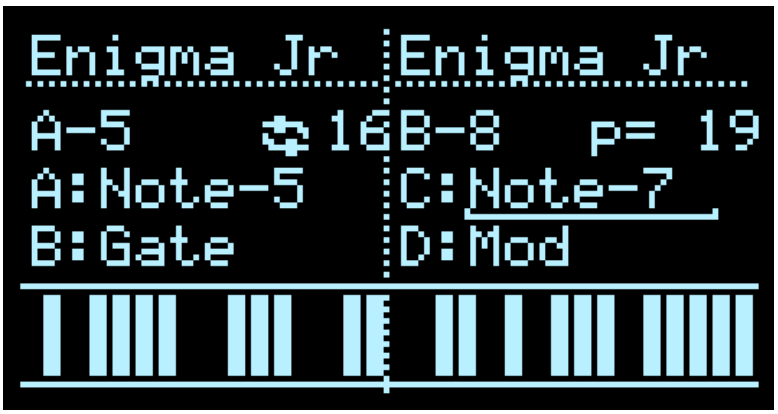
Credits

The original applet and backend *tideslite* algorithm were written by **qiemem** (Bryan Head). In Phazerville from v1.6

Enigma Jr.

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 3 revisions



While Enigma imposes a track-song workflow, **Enigma, Jr.** allows you to organize your Turing Machines via CV. It allows playback and temporary manipulation of registers in your Enigma Register Library.

In many respects, Enigma, Jr. is similar to ShiftReg. But instead of generating a random sixteen-bit register, Enigma, Jr. uses a Register from the Enigma Register Library. Enigma may be used to shape and curate this Library (see https://github.com/Chysn/O_C-HemisphereSuite/wiki/Enigma).

I/O

	1/3	2/4
TRIG	Clock	No effect
CV INs	Transpose (on step)	Organize
OUTs	Assignable	Assignable

UI Parameters

- Register selection
- Probability
- Output Assignment (Channel A and Channel B)
  - **Note-3:** 3-bit depth, 8 notes of selected scale
  - **Note-4:** 4-bit depth, 16 notes of selected scale
  - **Note-5:** 5-bit depth, 32 notes of selected scale
  - **Note-6:** 6-bit depth, 64 notes of selected scale
  - **Note-7:** 7-bit depth, 128 notes of selected scale
  - **Modulation / Expression:** The least-significant 8 bits are used to generate a modulation value that's scaled to 0-5 volts. Modulation and Expression are identical for CV
  - **Gate:** When bit 0 is 1, the assigned output emits 5V until the next time bit 0 is 0. That is, a high gate will span multiple rotations of the Register
  - **Trigger**: When bit 0 is 1, a trigger pulse is emitted at the assigned output

Organize

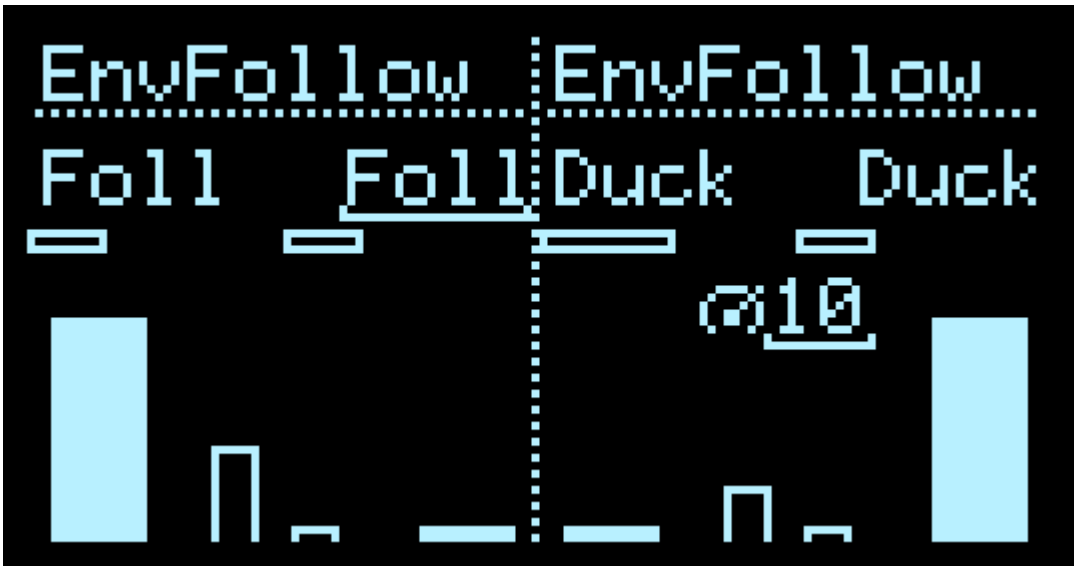
CV 2 is the Organize control. When a Register has advanced to the end of its length, CV 2 is read, and a new Register is loaded, based on the CV 2 modulation. If your Library has Registers marked as Favorites, the CV will choose from among your Favorites. If you have no Favorites selected, the CV will choose at random from among all 40 Registers in the Library.

Enigma, Jr. is in Hemisphere Suite from v1.5.

Envelope Follower

Jump to bottom

Nicholas Michalek edited this page on Jun 13 · 3 revisions



EnvFollow is a dual envelope follower / ducker, with adjustable gain and response speed

I/O

	1/3	2/4
TRIG	No effect	No effect
CV INs	Channel 1 Input	Channel 2 Input
OUTs	Envelope 1	Envelope 2

UI Parameters

- Mode Selection
  - Follow: Envelope output matches input signal amplitude
  - Duck: Envelope output inverse of input signal amplitude
- Gain (higher values produce an envelope that is more sensitive to the input signal), up to 31x
- Response speed (both channels)

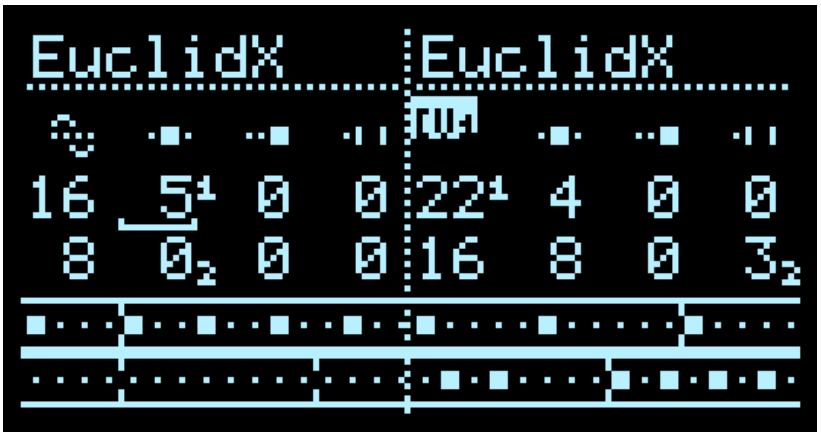
Credits

Adapted from EnvFollow by Chysn, with mods by djphaze...

EuclidX

Jump to bottom

Nicholas Michalek edited this page on Jun 13 · 4 revisions



A deluxe Euclidean pattern generator, two channels, up to 32 steps each. Using UI code from qiemem; named after adegani's version of the applet with the flexible CV Input routing. The extra zero-padding parameter makes it worthy of the *X* ;-)

I/O

	1/3	2/4
TRIG	Clock	Reset
CV INs	Assignable	Assignable
OUTs	Ch 1 Triggers	Ch 2 Trigger

UI Parameters

- Pattern length
- Pattern fill (hits)
- Offset (right shift)
- Padding (empty steps added to the pattern length)
- CV Input assignments

CV Inputs

The CV inputs will modulate whatever parameter the tiny *1* or *2* is sitting next to.

Outputs

Each channel outputs a standard trigger pulse when clocked. Global Pulse Length can be adjusted in Hemisphere Config.

Future ambitions

- Expand the pattern generator to 64 bits...
- Some kind of UI tweaks to better visualize patterns longer than 16.

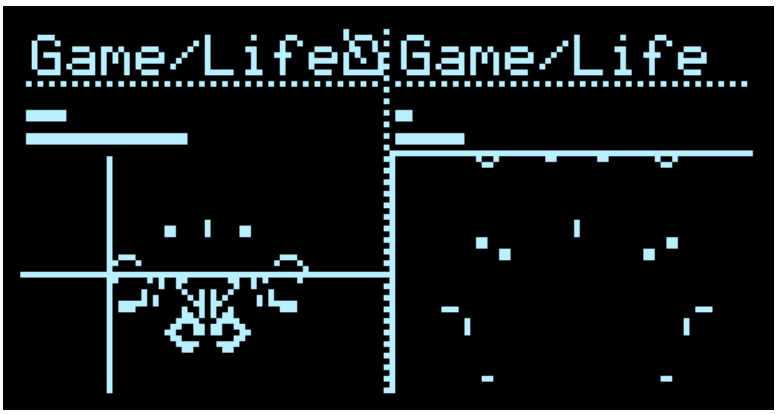
Credits

This applet is a mashup of code from qiemem, adegani, djphazer, and Chysn, with icons by ph-xyz

GameOfLife

Jump to bottom

Nicholas Michalek edited this page on Jun 13 · 2 revisions



Game of Life has been restored in Phazerville Suite v1.7!

Original wiki text from Chysn follows:

Conway's Game of Life

Note: Game of Life was an experimental applet, and was retired from Hemisphere Suite after v1.3. It had only tangential musical utility, and (more importantly) used a relatively large amount of the O_C's limited volatile memory, which will be of more value to upcoming projects.

Game of Life is a modulation source based on John Conway's 1970 cellular automaton Game of Life.

Controls

- Digital Ins: Digital 1 is a clock that generates the next generation. Digital 2 is a gate that draws a point at the current Traveler location.
- CV Ins: Input 1 sets the X position of the Traveler, and Input 2 sets the Y position of the Traveler
- Outputs: A/C is the Global Density, and B/C is the Local Density
- Encoder Push: Clears the play field
- Encoder Turn: Increases or decreases the effect of cells on the CV output

With each clock, Game of Life evaluates each cell (or pixel) the play field based on four rules:

Rule 1: If a live cell has fewer than two live neighbors ("neighbor" here meaning an adjacent cell in one of the eight principal directions), it dies. **Rule 2**: If a live cell has more than three live neighbors, it dies. **Rule 3**: If a live cell has two or three live neighbors, it continues to live. **Rule 4**: If a dead cell has exactly three live neighbors, it comes to life.

Additionally, cells can be written onto the screen with the "Traveler," a point on the screen described by an X and Y value, and controlled via CV. The CV inputs control the X value (with 0 volts being on the left side the display, and 5 volts being on the right) and Y value (with 0 volts being on the top and 5 volts being on the bottom). A gate to input 2 draws a new cell at the position of the Traveler.

There are two CV outputs. The first output is Global Density, which is a CV value proportional to the total number of live cells on the play field. The more populous your board is, the higher the Global Density value will be.

The second output is Local Density, which is a CV value proportional to the number of live cells in the vicinity of the Traveler. If you move the Traveler around, even with a static or slowly-changing play field, you'll get different CV values from the Local Density output.

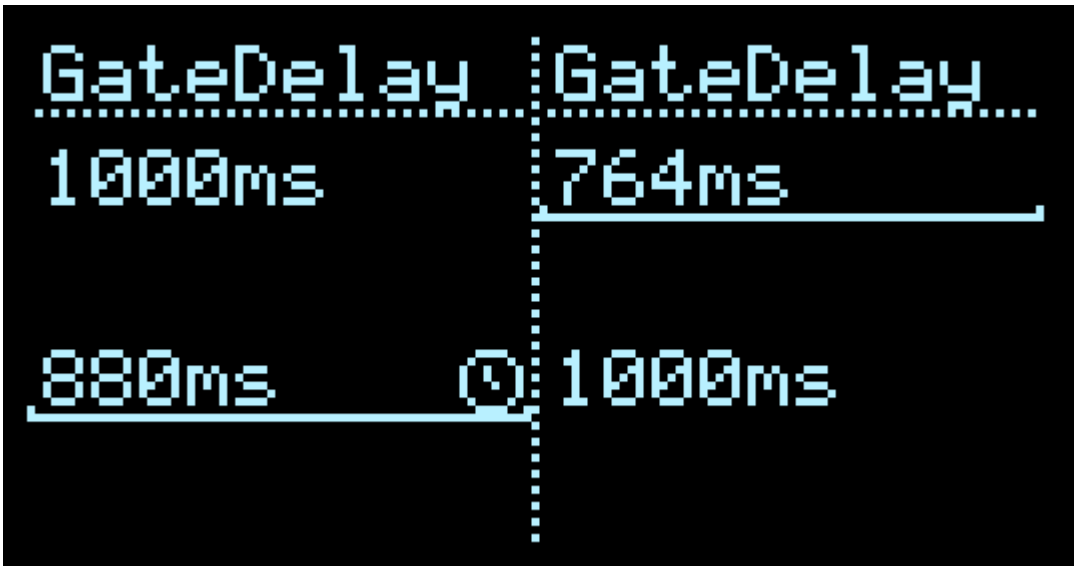
Credits

Copied/Adapted from Conway's Game of Life by Chysn

Gate Delay

Jump to bottom

bowlneudel edited this page 3 weeks ago · 4 revisions



Gate Delay is a dual trigger/gate delay loosely based on the Ladik S-189. Gate Delay has a 1ms resolution and a 2 second recording time. Modulation is bi-polar and can change the time by up to 1000ms.

I/O

	1/3	2/4
TRIG	Clock Ch1	Clock Ch2
CV INs	Time mod Ch1	Time mod Ch2
OUTs	Delayed Trigger Ch1	Delayed Trigger Ch2

UI Controls

- Per channel: Set delay time, from 0ms to 2000ms

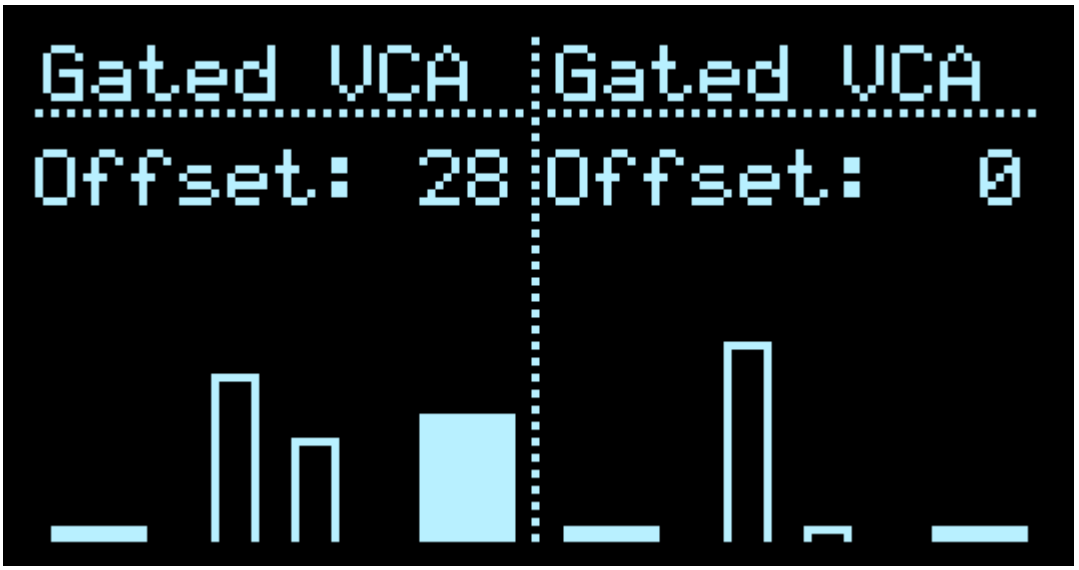
Credits

Adapted from Gate Delay by Chysn.

Gated VCA

Jump to bottom

bowlneudel edited this page 3 weeks ago · 6 revisions



Gated VCA is a VCA with the ability to gate the first output, and independently mute the second output. It can sort of be used for audio, but works best as an extra CV VCA. The display visualizes outputs A and B as the solid bars on the far left and right, respectively. The input signal is visualized as the hollow bar on the centre left, and the amplification CV on the centre right.

I/O

	1/3	2/4
TRIG	Ch A Gate	Ch B Mute (reverse gate)
CV INs	Input signal	Amplification CV
OUTs	Ch A output	Ch B output

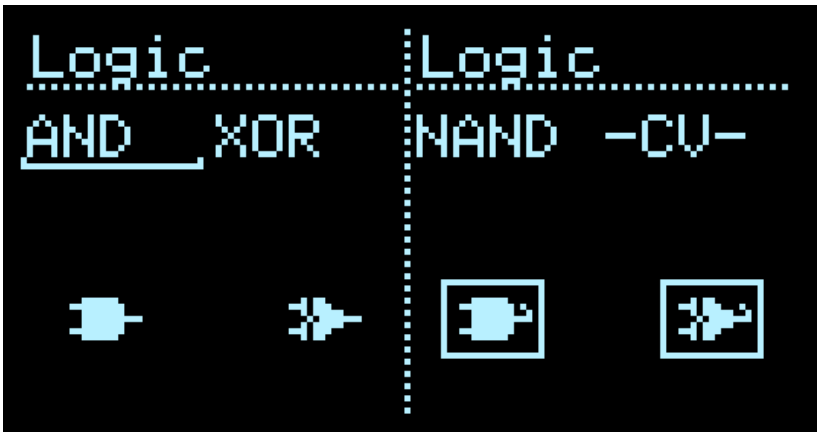
UI Parameters

- CV offset

Logic

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 3 revisions



https://youtu.be/mHZCaMH_Dgk

Logic is a two-input logic module that performs two logical operations at once.

I/O

	1/3	2/4
TRIG	Operand 1	Operand 2
CV INs	Gate type select for "-CV-" mode	Gate type for "-CV-" mode
OUTs	Result Ch1 (high/low)	Result Ch2 (high/low)

UI Parameters

- Logical gate type or "-CV-" for each channel

Notes

The Digital Inputs act as two logical operands, in the form of gate signals with high being True and low being False. CV Inputs 1 and 2 set the logical gate when the gate selected for the corresponding channel is "-CV-". The results of the logical operations are output as a high (5 volt, when True) or low (0 volt, when False) signal.

Available gate types are:

- **AND**: True when both inputs are True
- **OR**: True when either or both inputs are True
- **XOR**: True when both inputs are different than the other input
- **NAND**: True when either input is False
- **NOR**: True when both inputs are False
- **XNOR**: True when both inputs are the same

Another option, "-CV-" is available. When set to "-CV-" the logical gate type will be set via CV using the corresponding CV input.

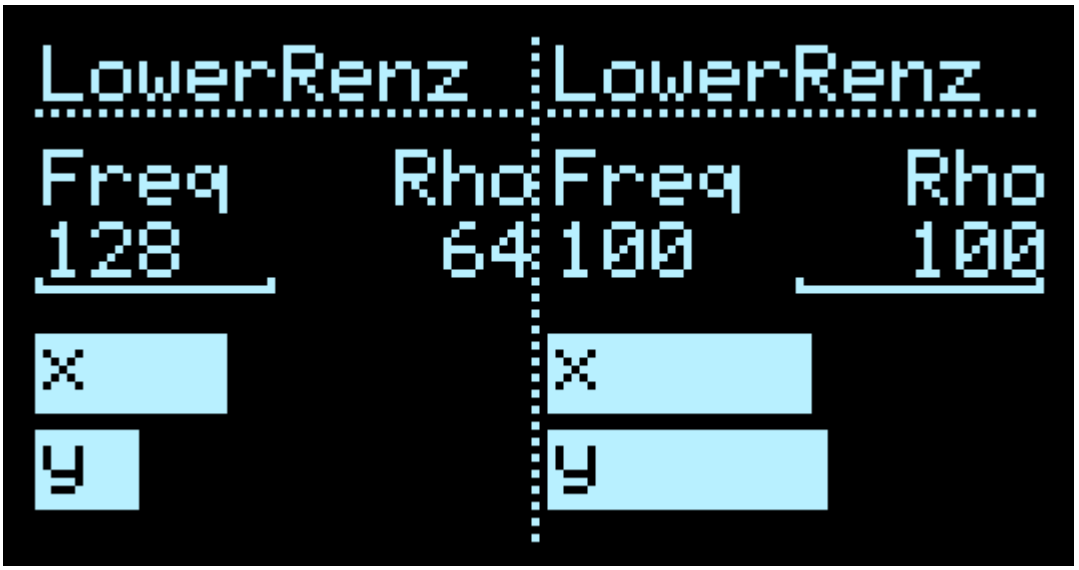
Credits

Adapted from Logic by Chysn.

LowerRenz

Jump to bottom

bowlneudel edited this page 3 weeks ago · 5 revisions



LowerRenz is a single Lorenz-only modulation generator based on the O_C's own Low-Rentz Dual Lorenz/Rössler Generator, which is itself based on an Easter Egg from Mutable Instruments Streams. It is a model of a 2-dimensional "strange attractor", practically functioning as a random LFO which orbits 2 poles (the low and high ends of the voltage range), and occasionally flipping between them. Changing the value of "Rho" will have subtle effects on the stability of the output.

I/O

	1/3	2/4
TRIG	Reset generator	Freeze outputs (gate)
CV INs	Frequency modulation (bipolar)	Rho modulation (bipolar)
OUTs	Lorenz X value	Lorenz Y value

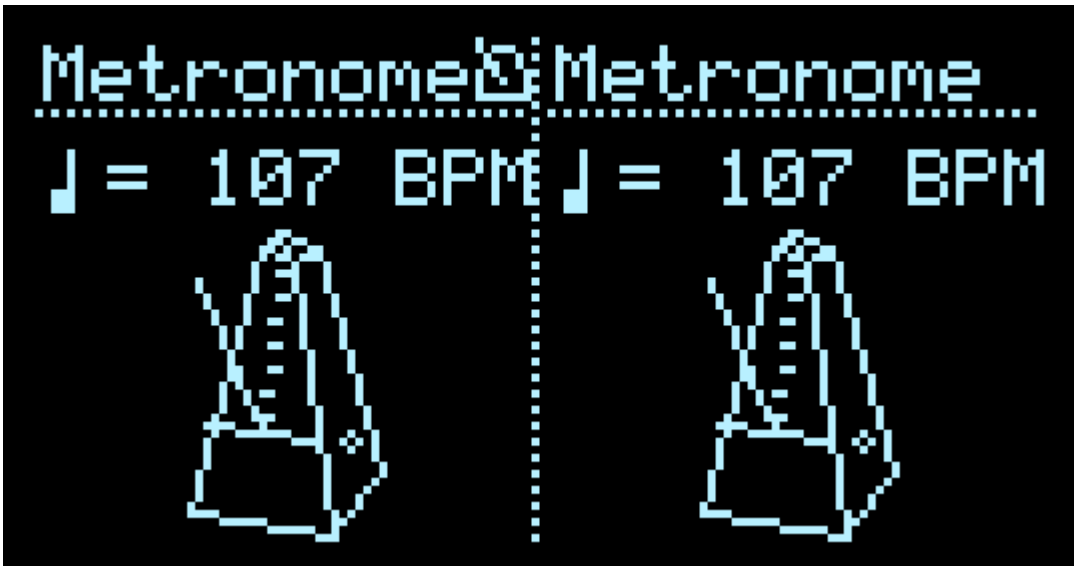
UI Parameters

- Frequency
- Rho

Metronome

Jump to bottom

bowlneudel edited this page 3 weeks ago · 3 revisions



Metronome is a simple tempo control & visualizer for the Hemisphere internal clock. The output is subject to any multiplication or division set in the Clock Setup menu (press UP + DOWN buttons simultaneously) to Trigger input 1/3.

If the clock is running, you'll see a small metronome icon on the main screen header. To arm, start or stop the clock, long-hold the left encoder button. This functionality is global to Hemispheres, and does not require the metronome applet to be loaded.

I/O

	1/3	2/4
TRIG	N/A	N/A
CV INs	N/A	N/A
OUTs	Multiplied clock (while internal clock is running)	None

UI Controls

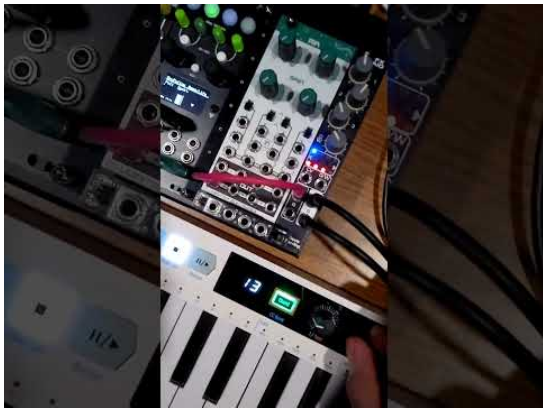
- Encoder: Sets the tempo of the internal clock (before any multiplication or division is applied)

MIDI Input

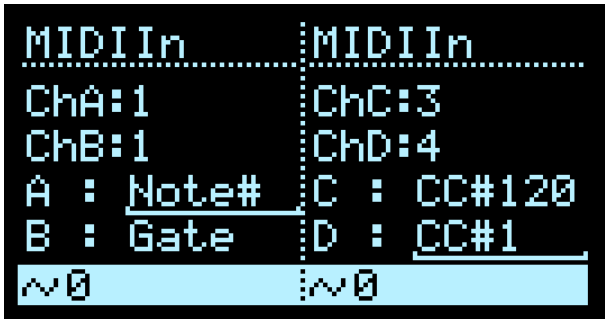
Jump to bottom

Nicholas Michalek edited this page on Jun 13 · 7 revisions

Video



Advanced MIDI-to-CV For Hemisphere



Phazerville greatly expands on the capabilities of the original **MIDI In** applet. MIDI messages coming in via USB are parsed and handled at a high level; the applet acts as a configuration UI, and also passes signals to the outputs.

If you switch to a different applet, the configured incoming MIDI messages are rerouted to the *inputs* of the selected applet and are *combined with the physical CV or trigger input*. This allows things like modulating parameters via MIDI CC or Pitch Bend, quantizing MIDI Notes to a scale, or triggering sequencer applets with MIDI Note-On. You can use **AttenOff** to scale and offset MIDI CC values. You can transpose **TB-3PO** patterns via MIDI Note and modulate Density with the Velocity, or Aftertouch, etc.

MIDI *Clock*, *Start*, and *Stop* messages are also handled automatically by the internal **Clock Setup** applet. Incoming MIDI Clock is divided from 24 PPQN to 2 PPQN internally.

Channel Settings

By default, all channels are set to "None". A pair of the MIDI In applets can be saved as a Preset to quickly recall settings.

Each channel filters and translates incoming MIDI messages as output CV from **MIDI In** applet, or to a corresponding logical input for other applets.

- **MIDI Channel** (independent for all 4 slots)
- **Mode**

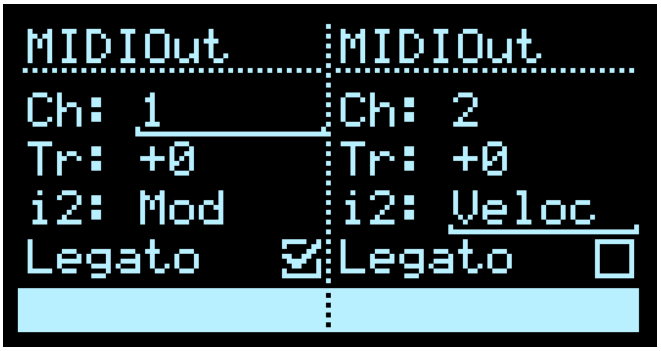
The available modes are:

- **None** - disabled
- **Note#** - semitone-quantized pitch CV
- **Trig** - standard trigger pulse from *NoteOn*
  - generates Clock pulses at corresponding trigger input
- **Gate** - held high at *NoteOn*, respecting polyphony; goes low when all Notes are Off
  - holds trigger input high, but does not generate a Clock pulse
- **Veloc** - Positive CV from the *Velocity* of the most recent *NoteOn*
- **CC#** (auto-learn) - Positive CV from assigned *CC#*
  - when selected, it will display `CC#-1` until a MIDI CC message on the selected MIDI Channel is received, at which point it latches onto the CC# of the message. Basically, select CC mode and wiggle a knob to auto-learn!
- **Aft** - Positive CV from *Aftertouch*
- **Bend** - Bipolar CV from *Pitch Bend*
- **Clock** - standard trigger pulse from *MIDI Clock* (divided internally to 2 PPQN)
- **Start** - standard trigger pulse from *MIDI Start*

MIDI Out

Jump to bottom

bowlneudel edited this page 2 weeks ago · 5 revisions



https://youtu.be/cVnJ3RqdbJU

MIDI Out is a monophonic CV-to-MIDI interface that uses the USB port on the back of the Ornament and Crime module. The MIDI interface will likely appear in your computer as "Teensy MIDI".

I/O

	1/3	2/4
TRIG	Gate	N/A
CV INs	Pitch Input	Assignable
OUTs	N/A	N/A

UI Parameters

- MIDI Out channel
- CV 2 function
- Transposition
- Legato
- Display MIDI Log

Notes

Digital 1 is a gate. A new gate causes MIDI Out to send a new Note On message, with the note based on CV 1. CV 1 will be quantized to a MIDI note number, which will be sent when Digital 1 goes high. CV 2 is assignable.

The CV 2 input may be assigned to one of the following MIDI messages:

- **Mod**: Incoming CV will be converted to a CC#1 (modulation wheel) coarse value message
- **Aft**: Incoming CV will be converted to an aftertouch message
- **Bend**: Incoming CV will be converted a pitch bend message. Pitch bend can be positive or negative, so MIDI Out expects a bi-polar voltage
- **Veloc**: Incoming CV will be used to set the velocity value of outgoing Note On messages. If Veloc is not assigned to CV 2, Note On messages will have velocity of 100.

Transposition

Transposition has a range of -24 to +24 semitones, and this number is simply added to the computed note number.

Legato

When Legato is Off, a Note On message is only sent when the gate goes from low to high. In other words, one note needs to be released before the next note can be sent. This option is useful for playing MIDI Out with sequencers, or when using things like LFOs for pitch.

When Legato is On, once the gate goes high, a Note On message is sent. If the note changes, then a Note Off message is sent for the previous note, and a new Note On message is sent for the new note. This option is useful for playing MIDI Out with a CV controller (Tetrapad, Pressure Points, KeyStep, etc.).

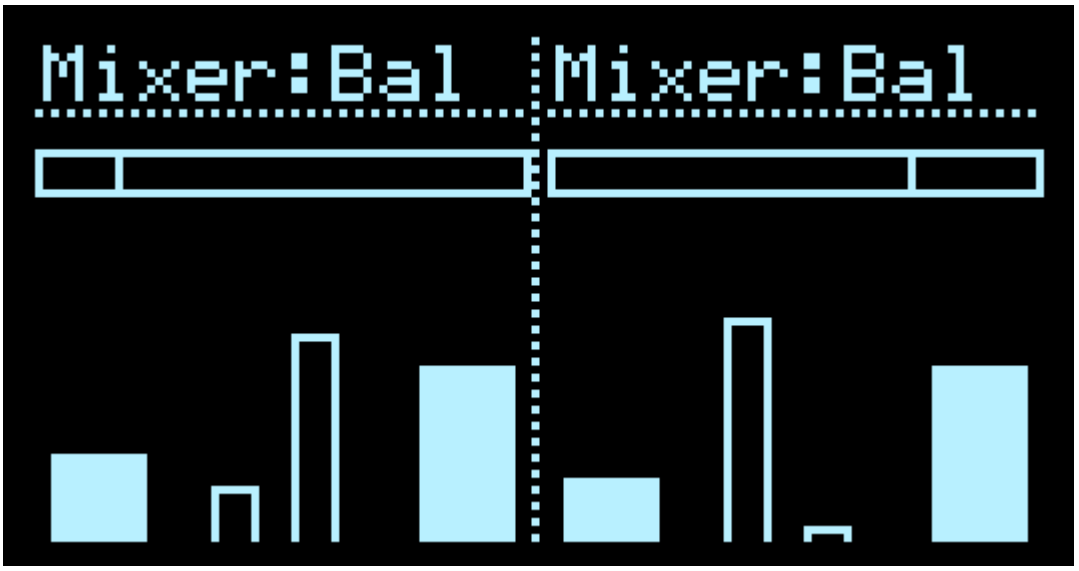
ADC Lag

There is a ~3 millisecond delay between the time the gate goes high and the note value is read from CV 1. This is because Ornament and Crime has some latency in the ADCs which causes the digital inputs to register first; so some delay is required to give us the best chance of getting the right note.

Mixer Balance

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 2 revisions



Mixer:Balance is a two-input CV mixer with a balance control and two complementary outputs.

I/O

	1/3	2/4
TRIG		
CV INs	Input 1	Input 2
OUTs	Mixed Output 1	Mixed Output 2

UI Control

- Encoder: Adjusts the balance between CV 1 and CV 2

The balance indicator, controlled by the encoder, shows the relationship between the CV 1 and CV 2 signals' relative levels at Output A/C. With the indicator all the way to the left, only CV 1's signal appears at A/C; all the way to the right, only CV 2's signal appears at A/C.

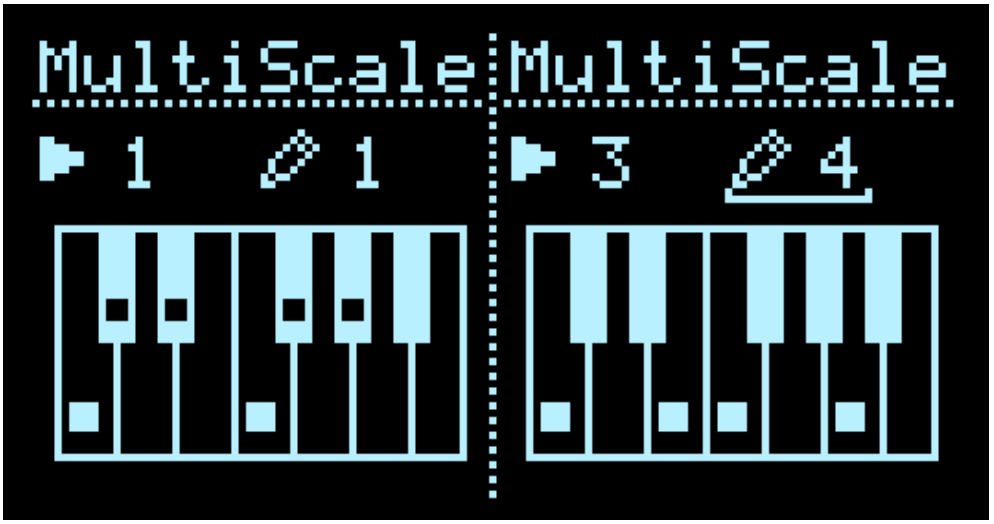
Output B/D is the complement of A/C. That is, the relative levels of CV 1 and CV 2 are the opposite, so that with the balance indicator at the left, only CV 2's signal appears at B/D, etc.

In Mixer:Balance, all signals are attenuated by 50% before being sent to the mix.

MultiScale

Jump to bottom

bowlneudel edited this page 3 weeks ago · 2 revisions



Similar to Scale Duet Quantizer, **MultiScale** provides 4 separate editable scale masks, with CV control to switch between them. Unlike ScaleDuet, this applet outputs a trigger on B/D when the active scale changes. Eventually, the functionality of the two applets will be merged.

The number next to the Play icon indicates the scale mask currently playing (selectable by CV input 2/4). The number next to the Pencil icon indicates the scale mask currently being edited and displayed in the UI.

Operation is continuous by default, until a Clock is received.

I/O

	1/3	2/4
TRIG	Clock	Unclock (switch back to continuous)
CV INs	Input Signal	Scale select
OUTs	Quantized Pitch	Trigger when Scale select changes

UI Parameters

- Scale selection for edit
- Enable/disable notes in currently editable mask

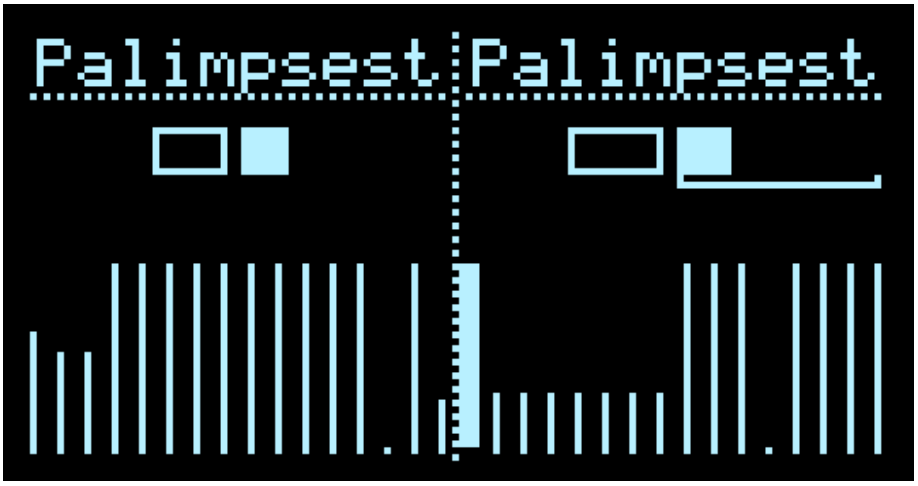
Credits

Authored by zerbian

Palimpsest Accent Sequencer

Jump to bottom

bowlneudel edited this page 3 weeks ago · 4 revisions



Palimpsest is an accent sequencer that composes a pattern by way of a repeated sequence of trigger impressions. A/C is the accent output, and B/D is a trigger output sent when the level of the composed step is around 3V

Palimpsest for Ornament and Crime is a port of this developer's alternate firmware for Mutable Instruments Peaks.

The idea behind Palimpsest is to write sequences gradually, using a pair of trigger signals. One of the triggers (Digital 1) clocks the sequencer. The other trigger (Digital 2) is a "Brush," which adds a CV value ("Compose") to the current step. If the sequencer is clocked without a brush trigger having arrived during that step, a CV value ("Decompose") is *subtracted* from that step.

The results can be subtle, with a small Compose value and small or zero Decompose value. Or more dramatic shifts can be made, with larger values.

The sequence length can be between 2 and 16 steps. *When the cursor is on the Length setting, the sequence is locked, and incoming triggers will not affect it.*

I/O

	1/3	2/4
TRIG	Clock	Brush (gate)
CV INs	Decompose	Compose
OUTs	Accent	Trigger

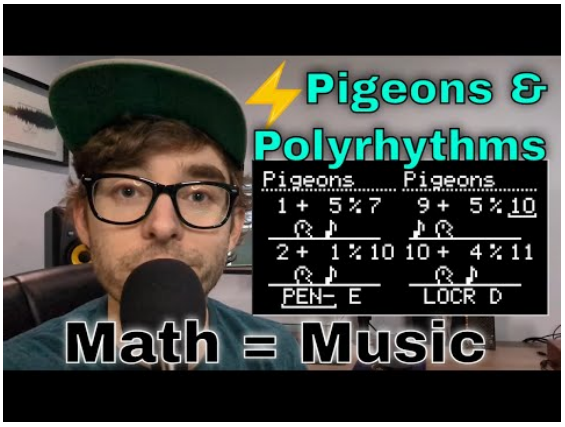
UI Parameters

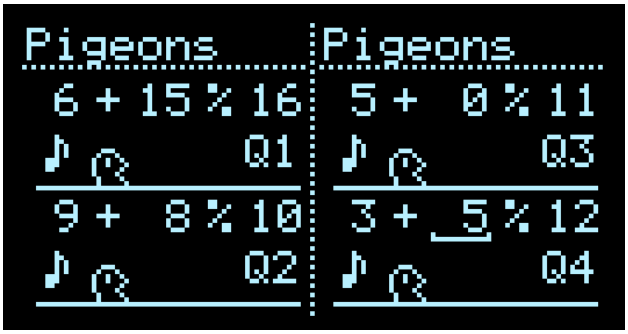
- Compose
- Decompose
- Length

Pigeons

Jump to bottom

bowlneudel edited this page 3 weeks ago · 12 revisions





I/O

	1/3	2/4
TRIG	Poke Pigeon 1 (top)	Poke Pigeon 2 (bottom)
CV INs	Modulo of Pigeon 1	Modulo of Pigeon 2
OUTs	Pitch CV from Pigeon 1	Pitch CV from Pigeon 2

("Poke" == Clock == Trigger)

UI Parameters

- Seed value 1
- Seed value 2
- Modulus
- Quantizer selection/edit

What is the Pigeonhole Principle?

The core concept is related to the Fibonacci Sequence, based on **this most excellent video** by Marc Evanstein. Something about the way numbers behave in modular arithmetic being akin to a limited number of pigeonholes for an infinite number of pigeons...

How does the applet work?

This applet generates two Pitch CV values at the Outputs, each quantized using given Scale and Root Note settings.

There are two **Pigeons** (channels), independently triggered. They are each continuously singing a *note* (a number representing scale degree) and have a *modulus* that limits their range. When triggered, the current and previous *notes* are added together, divided by the *modulus* value, and the remainder used for the next *note* value.

Each Pigeon has a certain number of holes they can visit (the *modulus*). The pair of notes represents the coordinates of the current hole; your Pigeon is guaranteed to revisit the same holes eventually.

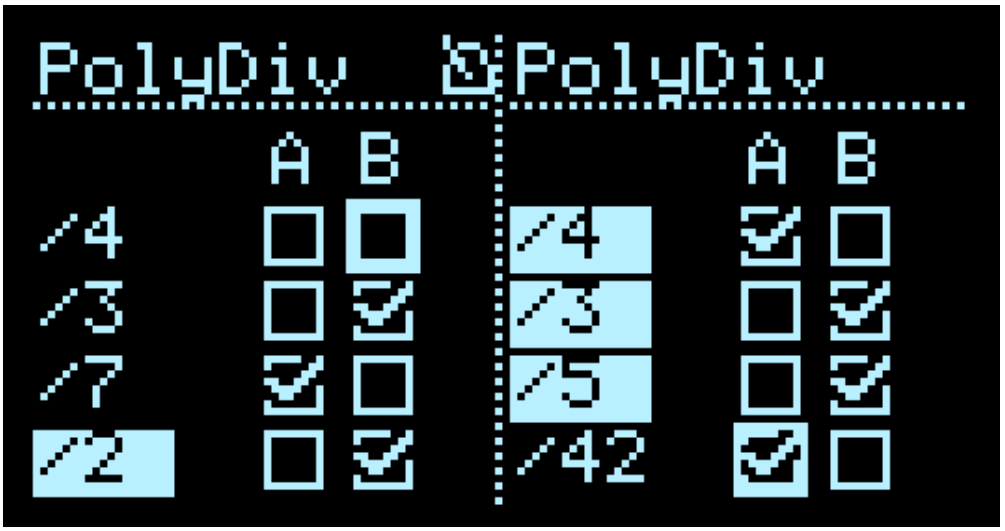
Pigeons are easily triggered - by the physical trigger inputs, internal clock pulses, or neighboring trigger sequencer applets (like ProbDiv, or DivSeq).

The **CV inputs** change the *modulus* value for each channel, affecting the range of the generated melodic sequence. It is possible to cause both note values to drop to 0 (the root note), and your Pigeon will take a nap there. If that happens, you'll have to nudge it with the encoder, or maybe load a Preset with a MIDI PC message...

PolyDiv

Jump to bottom

Nicholas Michalek edited this page on Jun 13 · 3 revisions



PolyDiv is a set of 4 clock dividers running in parallel, driven by a single Clock and Reset. A combination of them can be routed to each output with a 2x4 matrix.

The CV inputs can be used to toggle **XOR** mode per channel - a positive voltage switches from the default logical **OR** operation to **XOR**. This means two simultaneous pulses from the selected clock dividers will cancel each other out, leading to more curious polyrhythmic patterns.

See it in action: https://www.youtube.com/watch?v=J1OH-oomvMA&t=305s

I/O

	1/3	2/4
TRIG	Clock	Reset
CV INs	Ch 1 mode: XOR (positive) OR (zero)	Ch 2 mode: XOR OR
OUTs	Ch 1 Trigger sequence	Ch 2 Trigger sequence

UI Parameters

- Division value for each of the four clock dividers
- Checkboxes to route pulses to outputs

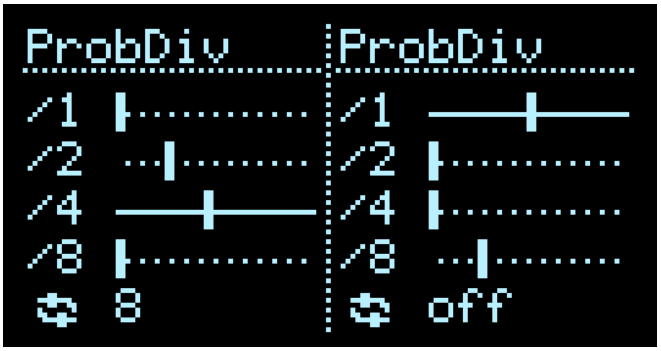
Credits

Authored by djphazer, loosely based on the Moog Subharmonicon sequencer

ProbDiv

Jump to bottom

bowlneudel edited this page 3 weeks ago · 4 revisions



ProbDiv is a stochastic rhythm generator where you can assign probability to different clock division settings. It takes inspiration from the rhythm section of the Stochastic Inspiration Generator, except it uses clock divisions instead of note divisions. You can also capture a loop based on the probability settings with anywhere between 1 and 32 steps. It can be used by itself or with ProbMeloD, which it will automatically link to.

At the heart of ProbDiv is a clock divider. There are four clock divisions that can be given probability of being selected, `/1`, `/2`, `/4`, and `/8`. On the first clock input received at Digital 1, a new division is selected based on the probability settings for each division. Once that division is reached, a trigger is sent to Output A/C, another division is selected, and the process starts over. If all divisions have a probability of 0, nothing will be selected and no clocks will be output.

See it in action: https://www.youtube.com/watch?v=uR8pLUVNDjI

I/O

	1/3	2/4
TRIG	Clock	Reset
CV INs	Loop length	Reseed loop using current probabilities (when > 2.5v)
OUTs	Divided clock	Skipped pulses (when a division `/1` is active)

UI Parameters

- Probability per division
- Enable / disable loop
- Loop length

Probabilities

Probabilities are defined as weights from 0-10 for choosing a specific division. For example, if you only have a probability set for `/1` it will always be selected no matter the if the probability is 1 or 10. Or if you have a probability of 10 for both `/1` and `/2`, it will be a 50/50 chance of either division being selected. Probability value can be thought of as the number of raffle tickets entered for each division setting when it's time to pick a new one, which is any time the current division is reached.

Looping

Loop can be enabled by setting the loop length > 0. Once loop has been activated, an entire 32 step sequence of divisions will be generated based on the current settings. The length will use a subset of this sequence. If the final division in the sequence goes beyond 32 steps it will get cut off.

Changing any of the probabilities while loop is enabled will generate a new loop using the new values. Loops are non-deterministic, so changing a probability and then changing it back will still result in a new loop. Changing loop length will not result in a new loop, unless you disable the loop by changing loop length to 0. A voltage of 2.5 or greater on CV2 will reseed the loop (if enabled) and a trigger on Digital 1 will reset the loop (if enabled)

Pairing with ProbMeloD

If ProbDiv and ProbMeloD are loaded in each hemisphere, they will automatically link. Clock division outputs from ProbDiv will automatically trigger ProbMeloD, and ProbMeloD will capture a loop when ProbDiv is looping as well. A new loop is generated in both applets when any parameter in ProbDiv or ProbMeloD are changed. When used together they can be treated as a whole probabilistic sequencer!

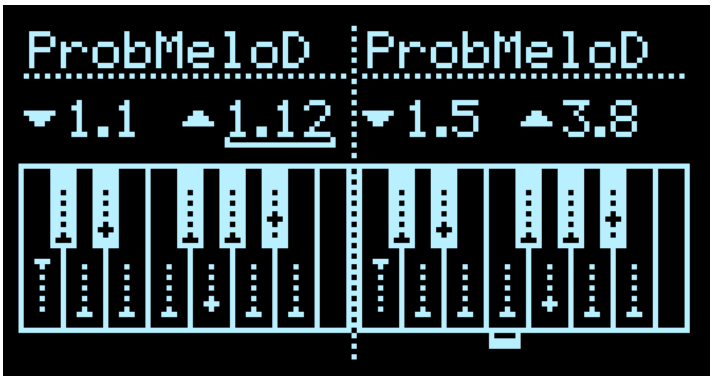
Credits

Copied/Adapted from ProbDiv by benirose

ProbMeloD

Jump to bottom

bowlneudel edited this page 3 weeks ago · 4 revisions



ProbMeloD is a stochastic melody generator inspired by the Melodicer and Stochastic Inspiration Generator. It allows you to assign probability to each note in a chromatic scale, as well as set the range of octaves to pick notes from. It can be used by itself or with ProbDiv, which it will automatically link to.

ProbMeloD has 2 channels, which can output independently clocked pitch values based on the same note ranges and probabilities.

See it in action: https://www.youtube.com/watch?v=uR8pLUVNDjI

I/O

	1/3	2/4
TRIG	Clock A	Clock B
CV INs	Low range	High range
OUTs	Pitch A	Pitch B

UI Parameters

- Low range
- High range
- Probability per note

Probabilities

Probabilities are defined as weights from 0-10 for choosing a specific note. For example, if you only have a probability set for C it will always be selected no matter the probability. Or if you have a probability 10 for both C and D, it will be a 50/50 chance of either note being selected. Probability value can be thought of as the number of raffle tickets entered for each note when it's time to pick a new one, which is any time a trigger is received on Digital 1.

Range

ProbMeloD has a range of 5 octaves. Lower and upper range are displays in an *octave.semitone* notation, so a range of `1.1` to `1.12` will cover the entire first octave, or C through B if your oscillator is tuned to C.

Pairing with ProbDiv and Looping

ProbMeloD does not have any looping functionality on its own, but can loop when paired with ProbDiv. Like ProbDiv, a new loop is generated in both applets when any parameter in ProbDiv or ProbMeloD are changed. If ProbDiv and ProbMeloD are loaded in each hemisphere, they will automatically link. Clock division outputs from ProbDiv will automatically trigger ProbMeloD, and ProbMeloD will capture a loop when ProbDiv is looping as well. When used together they can be treated as a whole probabilistic sequencer!

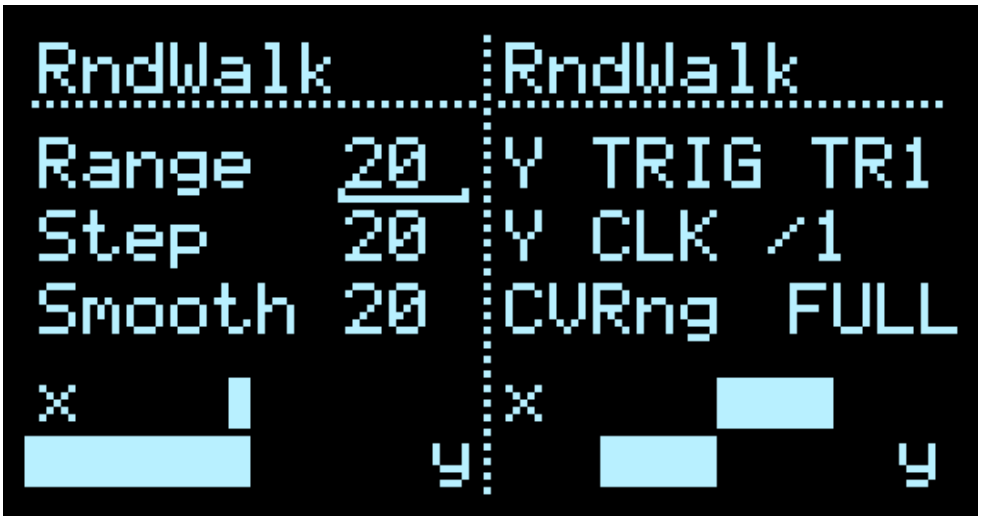
Credits

Copied/Adapted from ProbMeloD by benirose

Random Walk

Jump to bottom

bowlneudel edited this page 3 weeks ago · 2 revisions



RndWalk is a dual-channel stepped random bipolar CV generator. Range sets the total "distance" available to explore, and Step sets the maximum travel of each random step. The output voltage range may be scaled to full, 1 octave, 1 semitone, or 0.5 semitones.

Output Y may be clocked using either Trigger input 1/3 or 2/4, and subject to a clock division up to `/32`

I/O

	1/3	2/4
TRIG	X Clock (also assignable to Y clock)	Assignable to Y clock
CV INs	Range	Step
OUTs	Random Walk X	Random walk Y

UI Parameters

- Range
- Step
- Smooth
- Y Trigger input
- Y Trigger division
- CV Range

Credits

Authored by adegani

Reset Clock

Jump to bottom

bowlneudel edited this page 3 weeks ago · 4 revisions

ResetClk is a tool to assist with sequencers that can only advance forward (like the DFAM). It keeps track of the current step position and emulates a "reset" by quickly sending trigger pulses to loop back to the desired step. The first CV input modulates the position offset, and the applet keeps the actual sequencer position synchronized.

See it in action: https://youtu.be/i1xU6-oPwfA

I/O

	1/3	2/4
TRIG	Clock	Reset
CV INs	Offset (actively corrects position)	N/A
OUTs	Sequence Advance Trigger	Final Trigger

- The first output is the primary one to use for advancing another sequencer.
- The second output is a trigger that waits to fire until the rapid bursts are finished and the step position has settled. Use this one for triggering envelopes.

UI Parameters

- Length
- Offset - Reset jumps back to this position; changing with encoder actively corrects current position
- Spacing between pulses
- Current position - to calibrate display with actual sequencer position

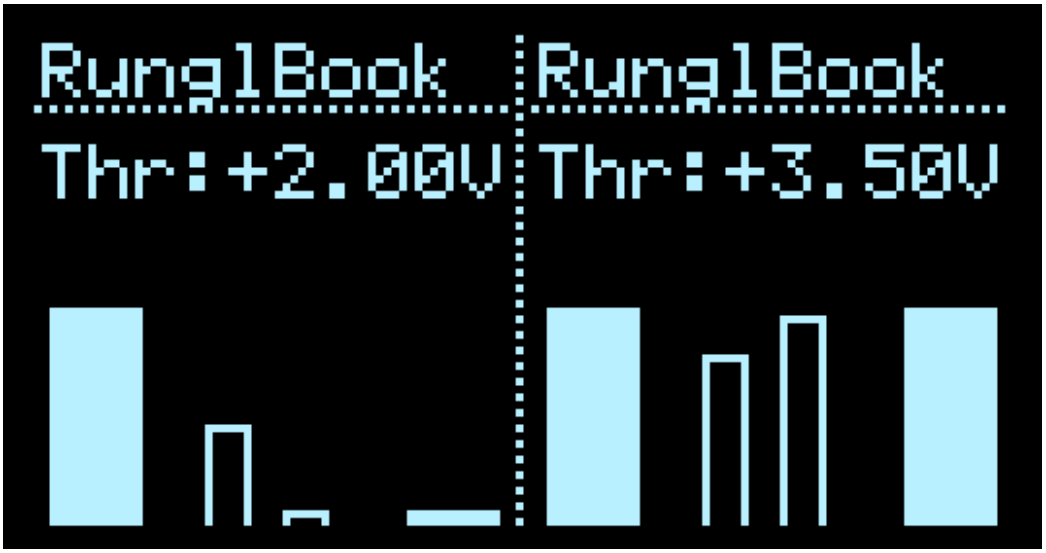
Credits

Copied/Adapted from Peter Kyme

RunglBook

Jump to bottom

bowlneudel edited this page 3 weeks ago · 4 revisions



RunglBook is a chaotic shift-register modulation based on Rob Hordijk's rungler circuit.

I/O

	1/3	2/4
TRIG	Clock	Freeze
CV INs	Signal	N/A
OUTs	Rungle output (low three bits)	Rungle output (high three bits)

UI Parameters

- Threshold (semitone increments)

When Digital 1 is clocked, the register is shifted one bit to the left, and the signal is read. If the signal exceeds the voltage threshold, bit low bit of the shift register is set to 1. Otherwise, it is set to 0.

If Digital 2 is receiving a gate at the time the clock is received, the register is shifted, but no read is done; instead, the high bit is moved to bit 0.

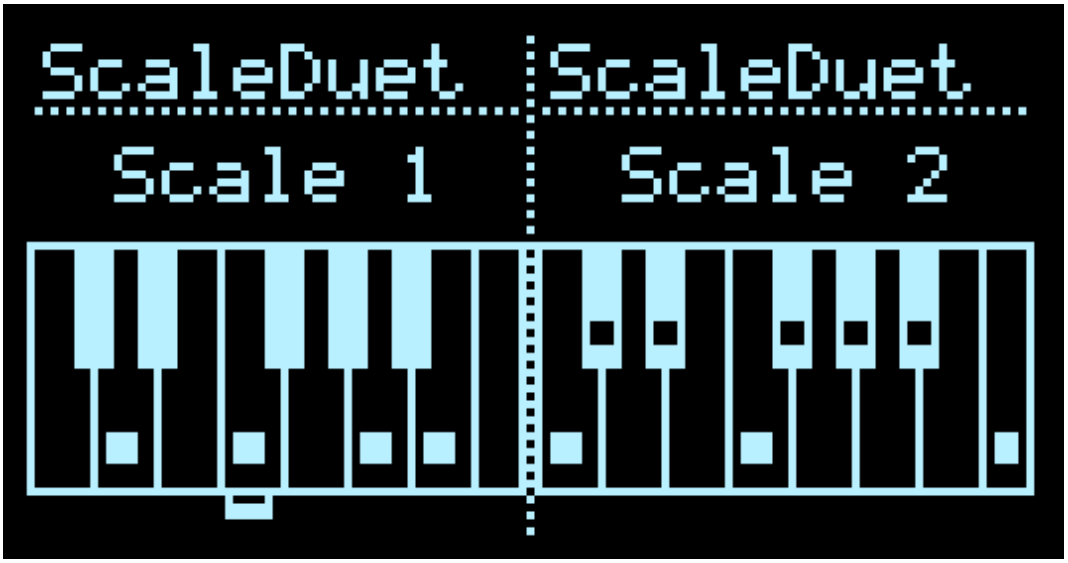
Three bits of the eight-bit register are scaled to 8 possible values between 0V and 5V, and sent to the outputs. These are the Rungle outputs.

RunglBook is in Hemisphere Suite starting with v1.5.

Scale Duet Quantizer

Jump to bottom

Nicholas Michalek edited this page on Jun 13 · 4 revisions



Scale Duet is a single-channel quantizer that allows you to switch between two user-defined scales. The scales are edited with an on-screen keyboard.

Quantization is continuous by default, unless a Clock pulse is received. Sending a positive voltage to CV2 will restore continuous operation.

I/O

	1/3	2/4
TRIG	Clock	Scale 1 (low) or Scale 2 (high)
CV INs	Input Signal	Unclock (switch back to continuous)
OUTs	Quantized Pitch	Auto-trigger when Pitch changes

UI Controls

- Encoder Push: Toggles the note above the cursor ON or OFF. A small square will appear on the keyboard for notes that will be played when that scale is selected.
- Encoder Turn: Moves through the notes for each scale, and between Scales 1 and 2.

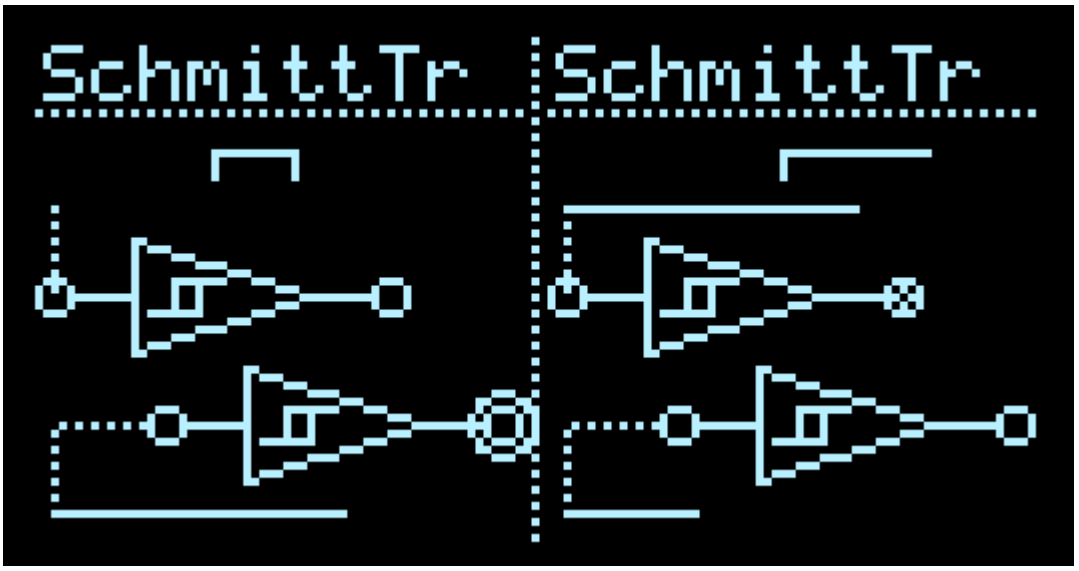
Scale Duet differs from Dual Quantizer in a few important ways:

- Scale Duet plays from two user-defined scales, rather than pre-programmed scales
- It quantizes only one value at a time

Schmitt Trigger

Jump to bottom

bowlneudel edited this page 3 weeks ago · 5 revisions



This applet is a dual Schmitt Trigger with a programmable threshold range.

A Schmitt Trigger is a type of comparator that provides hysteresis in a modular patch. Each Schmitt Trigger's output goes high when its input crosses the High threshold (default of 2.6V), and stays high until the input goes back below the Low threshold (default of 2.1V).

I/O

	1/3	2/4
TRIG	N/A	N/A
CV INs	Signal 1	Signal 2
OUTs	Gate A (if Signal 1 between high and low threshold)	Gate B (if Signal 2 between high and low threshold)

UI Parameters

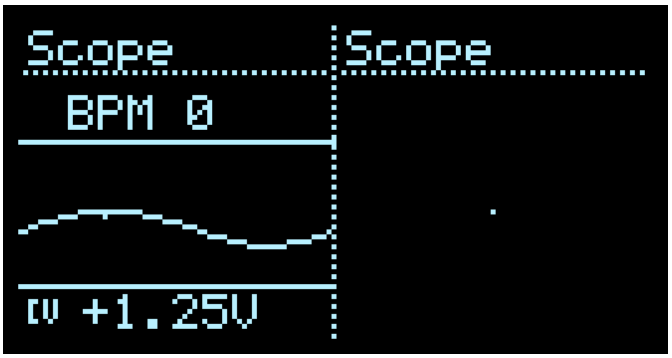
- Low threshold
- High threshold

Schmitt Trigger is in Hemisphere Suite from v1.4.

Scope

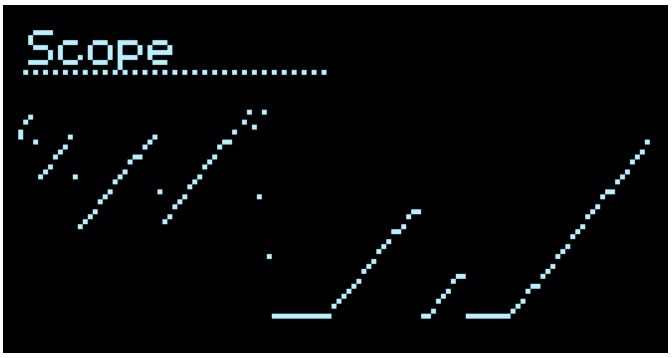
Jump to bottom

bowlneudel edited this page 3 weeks ago · 14 revisions



Scope is a simple CV and clock monitoring tool.

Phazerville Suite has updated it with more options and an X-Y view mode that plots both inputs in 2 dimensions. In v1.7.1, the help screen (double press UP button for left hemisphere, double press DOWN button for right hemisphere) has also been replaced with a full-screen view for maximum resolution.



I/O

	1/3	2/4
TRIG	Clock for BPM counter	Sync pulse for display wavelength
CV INs	Signal 1	Signal 2
OUTs	Pass-thru	Pass-thru

UI Parameters

- Mode
  - 1+1 (Scope signal 1, Report signal 1)
  - 1+2 (Scope signal 1, Report signal 2)
  - 2+1 (Scope signal 2, Report signal 1)
  - 2+2 (Scope signal 2, Report signal 2)
  - 1,2 (Scope XY)
- Rate (see notes)
- Freeze (toggle)

Notes

The "Rate" control is the number of 60µs "ticks" that will elapse between samples. 64 samples will be shown on the display at any time. The display is bi-polar. You can use the encoder to find the best rate at which to view a waveform. *Starting at Hemisphere Suite 1.1: If you send a clock to Digital 2 with the same period as the signal to CV 1 (for example, EOR trigger of Maths while viewing a waveform from Maths), Scope will automatically adjust the sample rate for the best view.*

Note: Everything's approximate, including the CV passthru.

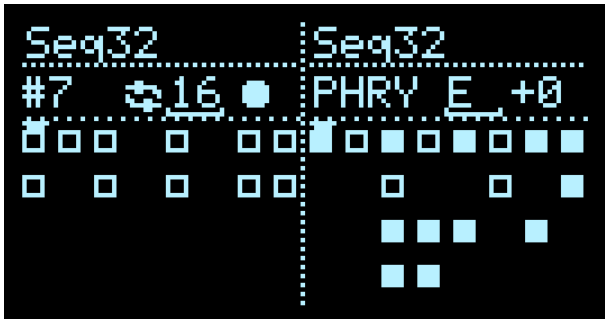
Credits

Originally authored by Chysn (Jason Justian) with modifications by Tom Waters & djphazer

Seq32

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 4 revisions



The **Seq32** applet reinterprets the 8 pattern storage slots from the original **Sequins** app as 32-step note sequences. (This will corrupt your Sequins patterns, if you have any!)

Each step has a bipolar range of -32 to +31 (scale degrees) and can be Muted or Accented. Accent currently means a longer gate + glide. Mute means no gate is sent and the Pitch output doesn't change.

Important: In addition to storing applet settings in a Preset, a manual EEPROM Save is necessary to save changes to the global patterns!

I/O

	1/3	2/4
TRIG	Clock	Reset
CV INs	Sequence Select	Transpose
OUTs	Pitch	Gate/Trigger

UI Parameters

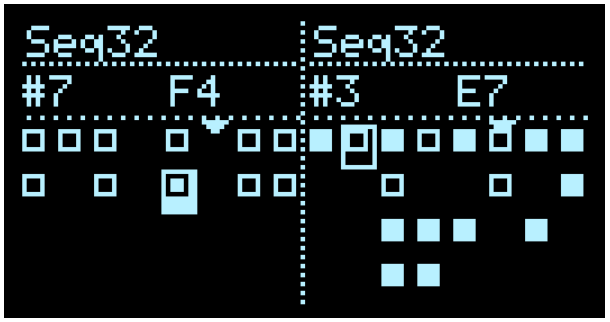
- Pattern select #1-8
- Length, per pattern
- Record mode
- Quantizer scale settings (popup editor)
- Transpose
- Edit notes

AuxButton Actions

With certain parameters highlighted for editing, the select button will execute a secondary action:

- Pattern# - Clear pattern
- Length - Randomize current pattern completely
- Transpose - Randomize pitches of current pattern, using transpose as the range

Step Editing



Use the encoder to move the cursor to a step, and push to toggle editing the note value. While editing, use the AuxButton to toggle Mute. Double-click to toggle Accent, indicated by a solid square.

Record mode

When the small record icon is engaged, the cursor is locked and a crude CV recording mechanism is active. When a clock is received, the sequencer advances, CV1 is captured as pitch, and CV2 is measured as Mute/Unmute/Accent (threshold at 0.5v for Unmute, and 2v for Accent).

It works, but it's a little clumsy because recording pitch/gate with the inputs conflicts with normal operation. This might get reworked in the future.

SequenceX

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 3 revisions

https://youtu.be/zsqAbNRgHJI

SequenceX is an 8-step semitone-quantized sequencer, adapted from the original Sequence5 applet. It now features bipolar output values and CV-triggered randomization.

I/O

	1/3	2/4
TRIG	Clock	Reset
CV INs	Transpose	>1V = Randomize sequence
OUTs	Pitch	BOC Trigger

UI Controls

- Encoder: select/edit each step; scroll further for mute toggles
- AuxButton: mute/unmute selected step

Each step is a semitone increment over a bipolar 5-octave (60-note) range. To change the length of the sequence, you can mute one or more steps by moving the slider all the way down until the slider handle disappears, or using the mute toggles. The clock will skip muted steps.

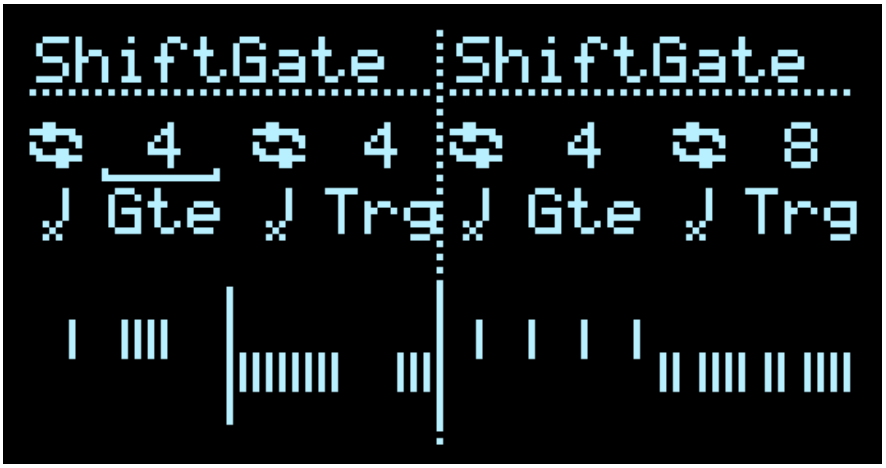
Credits

Adapted from Sequence5 by Chysn

ShiftGate

Jump to bottom

bowlneudel edited this page 3 weeks ago · 4 revisions



ShiftGate is a dual shift register-based gate/trigger sequencer for creating aleatoric rhythm patterns.

I/O

	1/3	2/4
TRIG	Clock	Freeze
CV INs	Flip bit 0 (Ch A)	Flip bit 0 (Ch B)
OUTs	Gate/Trigger Ch A	Gate/Trigger Ch B

UI Parameters

- Length
- Output type

ShiftGate has two channels controlled by a single clock input. Each channel starts with a random 16-bit register*. When a clock is received at Digital 1, the following things happen:

1. The register is shifted to the left.
2. If Digital 2 is high, the register is frozen; the high bit (based on length) of the previous value is moved back to the beginning (bit 0), and skip to 4.
3. If the CV input for the corresponding channel is low, the high bit (based on length) of the previous value is moved back to the beginning (bit 0). If the CV input is high, the high bit is flipped, and that value is put at the beginning. This is known as an XOR operation.
4. The value of bit 0 is examined. If the channel's Type is set to "Trig," then a trigger will be sent from the channel's output if the value is 1. If the channel's type is set to "Gate," the gate state is high if the value is 1. The gate will then *remain* high until the next time the register's bit 0 is 0.

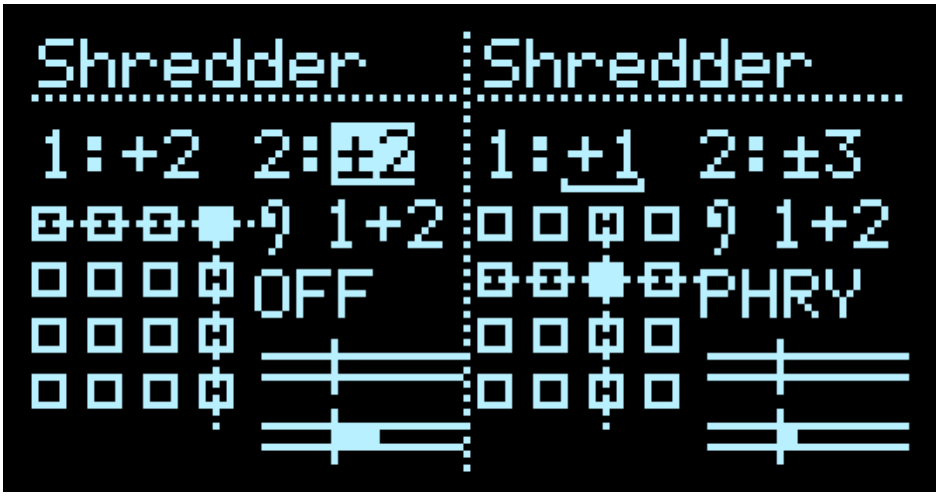
ShiftGate is in Hemisphere Suite from v1.5.

** When ShiftGate's state is saved via SysEx or system save, Output A's register is saved, and Output B's register is randomized.*

Shredder

Jump to bottom

bowlneudel edited this page 3 weeks ago · 6 revisions



Shredder is a 2-channel, 16-step CV sequencer on a 4x4 grid, inspired by the Noise Engineering Mimetic Digitalis. Each channel can have random values "shred" to the sequence within a specified range. Ranges can be from 1 to 5 volts unipolar, or 1 to 3 volts bi-polar. Additionally, one or both channels can run through the quantizer.

See it in action: https://youtu.be/yLr3vQJm3wM

I/O

	1/3	2/4
TRIG	Clock	Reset
CV INs	X position	Y position
OUTs	Pitch Ch 1	Pitch Ch 1

UI Parameters

- Octave range
- Quantized channels
- Quantizer edit
- Auxbutton: Shred selected channel (Highlight channel octave range and press the select button — UP or DOWN, depending on Hemisphere)

Note: Due to memory limitations, generated sequences cannot be saved. However, all parameters will be saved and new sequences will be generated from those settings on power up.

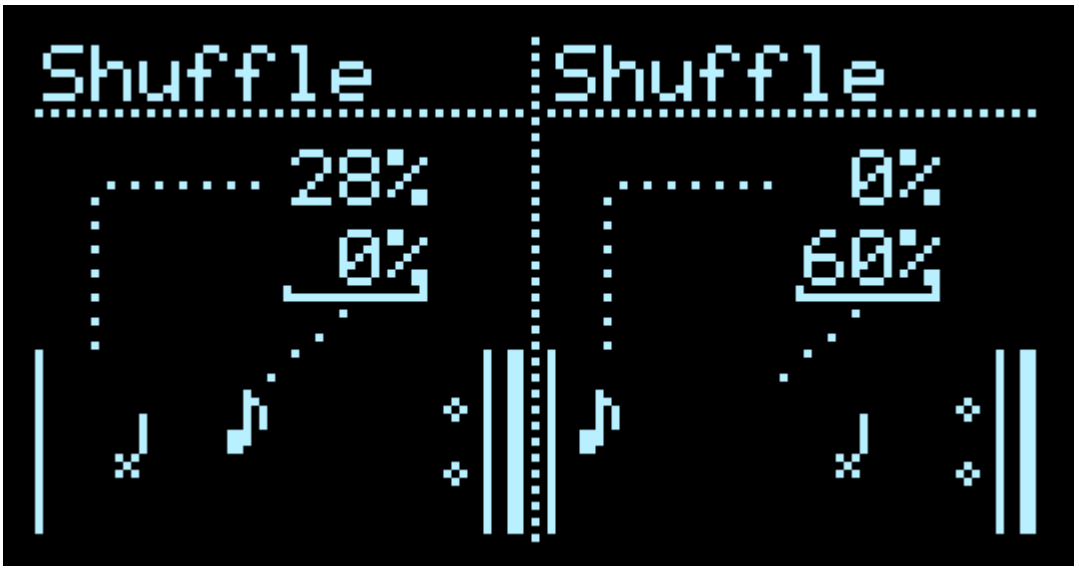
Credits

Copied/Adapted from Shredder by benirose

Shuffle

Jump to bottom

bowlneudel edited this page 3 weeks ago · 4 revisions



Shuffle is a two-step clock offset. Each step can be delayed by between 0% and 99% of the incoming clock tempo.

I/O

	1/3	2/4
TRIG	Clock	Reset
CV INs	% Delay for 1st step (bipolar)	% Delay for 2nd step (bipolar)
OUTs	Shuffled Clock	Triplet Clock

UI Parameters

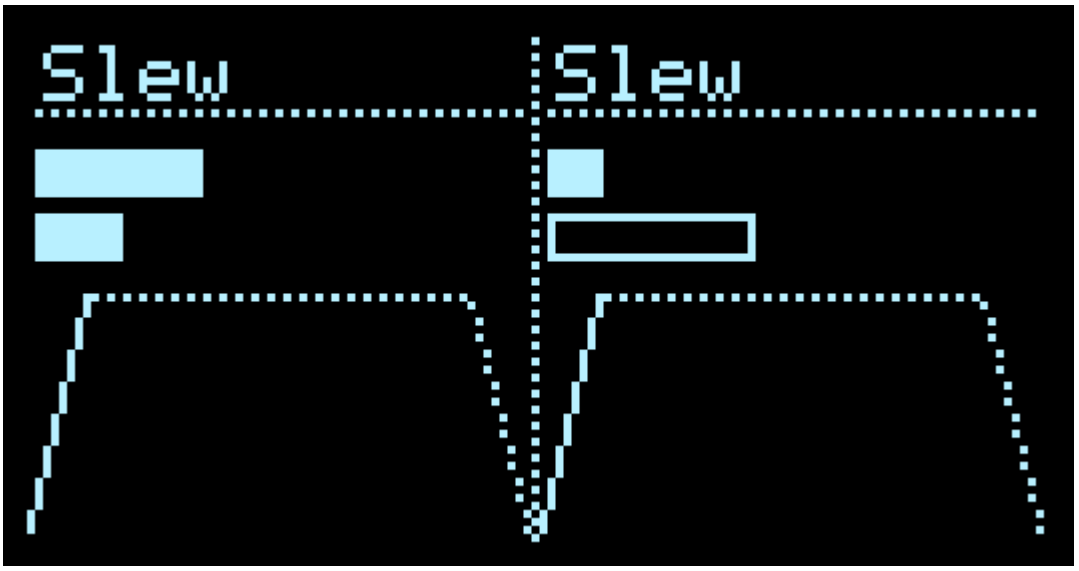
- Percentage delay of each step

Note that reset does not, itself, trigger a clock. It just brings the step back to the beginning so that another clock can trigger the first step.

Slew

Jump to bottom

bowlneudel edited this page 3 weeks ago · 4 revisions



Slew is a simple slew (or lag) processor. Two independent channels share the same settings. Channel 1's output is linear, and Channel 2's output is exponential.

I/O

	1/3	2/4
TRIG	Defeat Ch1	Defeat Ch2
CV INs	Input Ch1	Input Ch2
OUTs	Linear Ch1 output	Exponential Ch2 output

UI Parameters

- Rise time
- Fall time

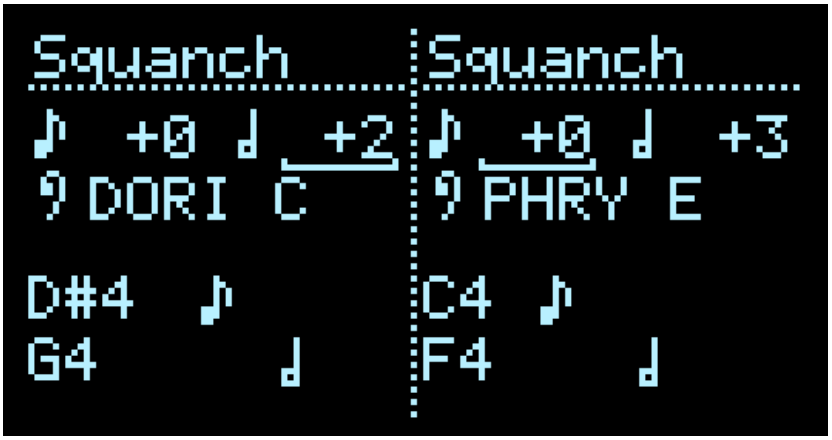
When Rise or Fall values are changed, a time (in ms) will briefly appear on the display. This indicates the approximate time that it would take for the linear signal to rise or fall 5 volts. The time between actual voltages will be proportionate to this time, and the exponential signal will take less time.

When a high gate is present at either digital input, the corresponding channel's slew will be defeated. That is, the input will pass through to the output.

Squanch

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 3 revisions



Squanch is a pitch-shifting quantizer with a single input and two pitch-shifted outputs, useful for producing harmonic intervals. It can be used as a voltage adder.

I/O

	1/3	2/4
TRIG	Clock	+1 Octave for Pitch 1
CV INs	Main Signal Input	Transpose/Shift for Pitch 2
OUTs	Pitch CV 1	Pitch CV 2

UI Parameters

- Pitch shift for each output, in scale degrees
- Scale
- Root

Notes

A clock pulse at Digital 1 causes Squanch to sample the signal at CV 1 and quantize it. A gate at Digital 2 adds one octave to the output at A/C.

CV 1 is the signal to be quantized. CV 2 is a bi-polar shift input that adds (or subtracts, if negative) voltage to or from the output at B/D.

Each channel begins in continuous operation. That is, the incoming CV is quantized at a rate of about 16667 times per second. This might result in undesirable slippage between notes, so clocked operation is available.

To enter clocked operation, send a clock signal to Digital 1. A clock icon will appear next to the selector for the scale to indicate that that quantizer is in clocked mode.

To return to continuous operation, stop sending clock to the quantizer, and then change the scale. As long as no additional clock signals are received, the quantizer will remain in continuous operation.

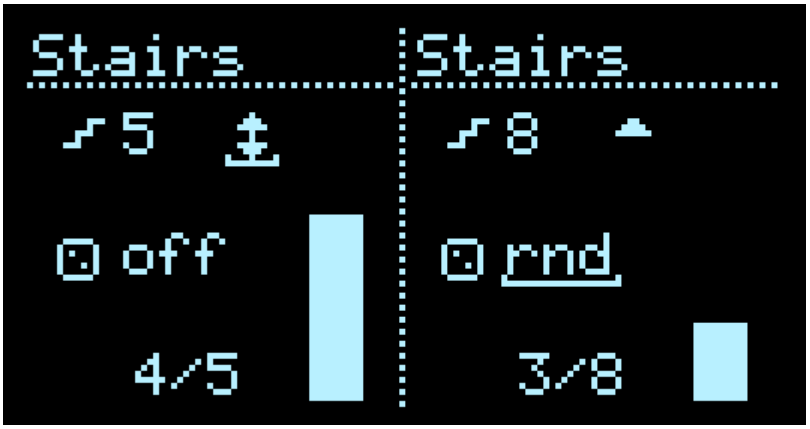
Credits

Adapted from Squanch by Chysn; in Hemisphere Suite starting with v1.5.

Stairs

Jump to bottom

bowlneudel edited this page 3 weeks ago · 4 revisions



Stairs is a stepped, clocked voltage generator based on the Noise Engineering Clep Diaz's 'step' and 'rand' modes. On each input clock pulse, the output voltage advances to the next 'step,' where the first step is always 0v and the last always 5v, with even voltage divisions on the intermediate steps.

I/O

	1/3	2/4
TRIG	Clock	Reset (can be held)
CV INs	Number of Steps	Current Step (overrides Clock if > 0v)
OUTs	Stepped CV output 0-5v	Beginning-of-Cycle trigger

UI Parameters

- Steps
- Direction (up, down, pendulum)
- Random (toggle)

Steps

Controls the total number of steps taken from 0 to 5v or 5v to 0 volts.

Direction

Sets the direction the voltage changes in on each step: up, up+down, or down. This effectively gives up ramp, triangle, and down ramp shapes. The step count is effectively doubled in up+down mode, except that the bottom and top steps are not repeated when changing directions.

Random

Turning random ON makes each step's voltage deviate a little bit from where it would be, but the overall direction of movement is preserved.

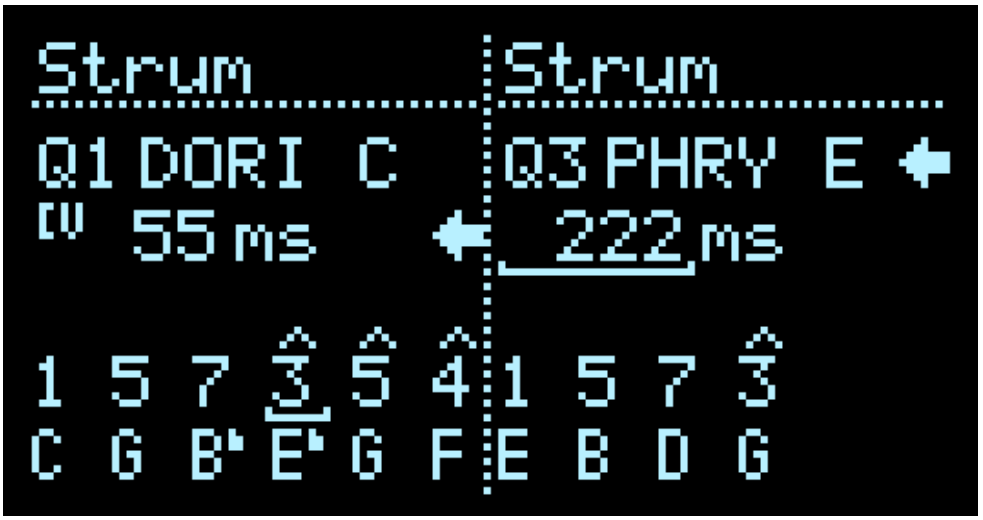
Credits

Authored by Logarhythm1

Strum

Jump to bottom

Nicholas Michalek edited this page on Jun 13 · 2 revisions



Strum is an arpeggiator. It generates fast sequences of up to six pitches from a selected scale, along with trigger pulses. This is ideal for achieving chord tones with voices with round-robin polyphony, like Rings, but also nice for slower arp sequences - spacing goes up to 500ms.

Incoming triggers intuitively run through the sequence either forward or backward. If retriggered in the middle of the sequence, it will immediately jump to the next step, allowing unique grooves to emerge from tempo-synced pulses.

I/O

	1/3	2/4
TRIG	Strum Up	Strum Down
CV INs	Chord Root Transpose	Spacing (default) / Quantizer select
OUTs	Pitch CV	Trigger Pulse

UI Parameters

- Quantizer select - AuxButton to edit
- Spacing - AuxButton to switch CV2 modulation destination
- Length
- Edit Intervals 1-6

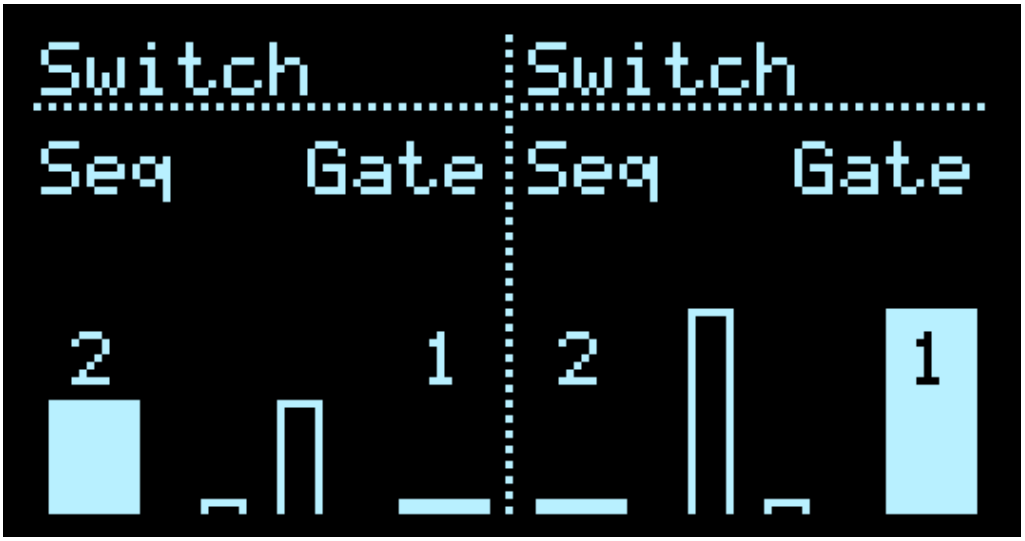
Credits

Authored by qiemem, with mods by djphazer

Switch

Jump to bottom

bowlneudel edited this page 3 weeks ago · 4 revisions



https://youtu.be/juu65pJyXlY

Switch is a two-channel switch with two switching methods: sequential and gated.

I/O

	1/3	2/4
TRIG	Flip/flop (Ch A)	Gate (Ch B)
CV INs	Signal 1	Signal 2
OUTs	Channel A	Channel B

UI Parameters

- None

The solid meter on the far left represents the state of output Channel A (noting the active input signal), likewise the solid meter on the far right represents output Channel B. The pair of meters in the middle represent input signals 1 and 2.

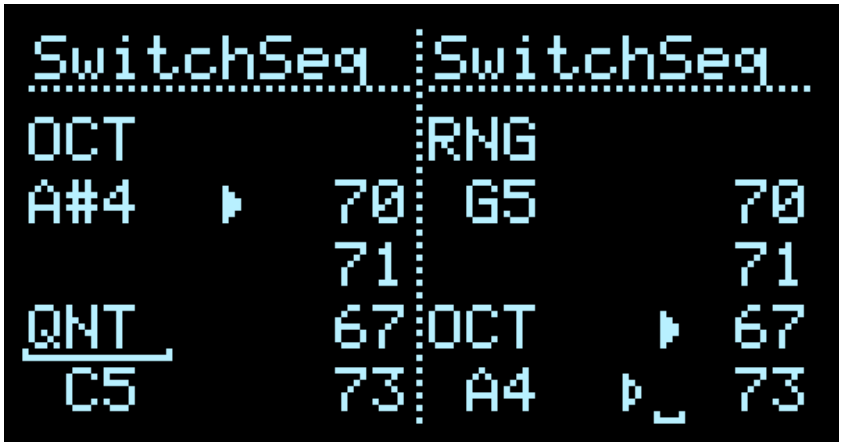
The sequential output (Ch A) alternates between Signal 1 and Signal 2, and the gated output (CH B) sends Signal 1 when digital Digital 2 is low, and Signal 2 when high.

Note: *The Ornament and Crime's circuitry is not made for precision 1:1 reproduction of voltages. The outputs will not exactly match the inputs, so Switch is not suitable for pitch CV. This is inherent to the O_C's design and does not indicate poor calibration.*

Switch Seq

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 1 revision



SwitchSeq is a flexible player for Seq32 patterns, providing various ways to switch between four sequences running in parallel. The global quantizers Q1-Q4 are used for outputs A-D, respectively. (Use Hemisphere Config to adjust Quantizer Scale settings)

I/O

	1/3	2/4
TRIG	Clock	Reset
CV INs	Octave / Seq select / Unquantized Input (depends on Mode)	(depends on Mode)
OUTs	Ch1 Pitch CV	Ch2 Pitch CV

Usage

4 sequences are played in parallel. The patterns themselves can be edited with Seq32, and can have various lengths. Currently, *Accent* and *Mute* have no effect in SwitchSeq.

The CV inputs can manipulate the sequences in a variety of ways.

There are two channels, with a shared Clock and Reset. Each channel has a few different Modes, controllable with the encoder.

Modes:

- OCT: This mode is active when the sequence arrow is solid. The encoder will select the active sequence, with CV controlling the octave (by 1 or 2 octaves).
- PCK: The CV input will pick which sequence is playing. Sampled after each clock.
- RNG: A sequence is chosen at random when clocked. CV is ignored.
- QNT: The internal sequences are ignored and the CV input will be quantized using the global Quantizer settings.

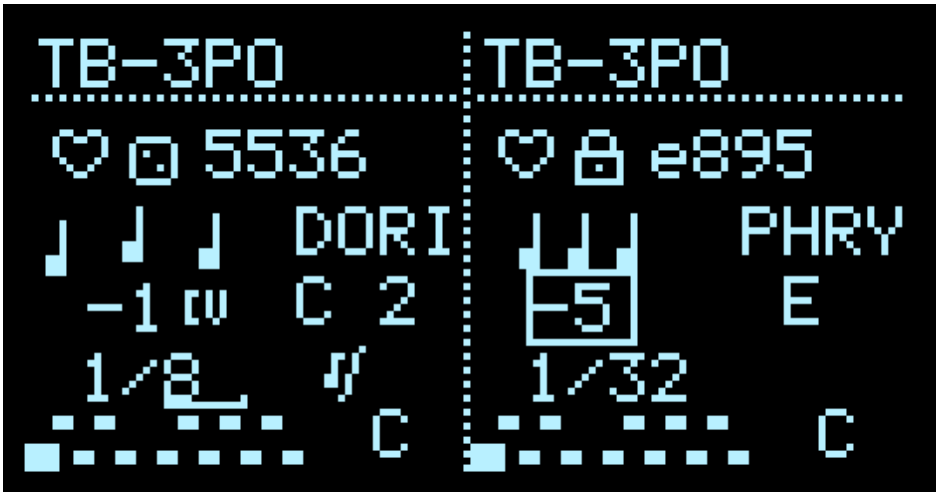
Credits

Authored by Nick Beirne

TB 3PO

Jump to bottom

Nicholas Michalek edited this page on Jun 14 · 5 revisions



TB-3PO is a TB-303 style, pitch CV and gate pattern generator robot, capable of fixed-time, exponential slides on the pitch CV for that secret TB sauce. It will do mono-pitched style 303 lines on one end, or full-range, Turing Machine style stuff on the other, all designed and tuned for musicality and calls/responses in live wiggling.

I/O

	1/3	2/4
TRIG	Clock	Reset / Regenerate
CV INs	Transpose	Density
OUTs	Pitch CV (with glide)	Gate (3v normal, 5v accent)

- Video overview: https://www.youtube.com/watch?v=AHC2zWnkjJY
- Synth Dad Tutorial: https://youtu.be/m08NMHuhFnE

UI Controls

- Sets the seed as unlocked or locked
- Edits unlocked seed value
- Note/pitch Density (-7 to +7)
  - AuxButton enables automation of encoder wiggling
- Quantizer Scale
  - AuxButton for popup editor
- Root note + octave
- Pattern Length

Seed

The seed parameter controls the random pattern generation. This is done deterministically, which means for the same seed you'll get the same patterns, based on the other controls.

Locking and Unlocking

By default the seed is unlocked (die icon) and will randomly change on every reset input pulse. Turning the encoder to the right will lock the seed (lock icon) and prevent it from changing when reset pulses restart the pattern.

Manual reset

When the seed die icon is selected, turning the encoder to the left once will reset the pattern after randomly choosing an entirely new seed. When the seed has the lock icon (turned right,) turning right once more will reset the pattern but leave the seed unchanged.

Editing the seed

When the seed is locked, pressing the encoder advances to each of the four seed hex digits in turn. In this way you can experiment or return to past favorite seeds.

Density

A bi-directional 'density' control specifies how many of the pattern's steps are likely to be gated, as well as the number of pitches that will be selected from. It ranges from -7 to +7:

Value / Gate Count / Pitches

- -7 / High / Root Only
- -6 / High / Root and +1 note in scale
- -5 / Med / Some notes in scale available
- ...
- -1 / Low / Most notes in scale available
- 0 / Low / All Notes in scale Available
- 3 / Med / All Notes in scale Available
- +7 / High / All Notes in scale Available

Density changes are applied on every clock, but if the seed is locked, specific values of density will give the same subset of patterns. In this way you can break down from busy to sparse and back.

Density CV

When the cursor is on Density, it can be changed directly by turning the encoder. CV Input 2 applies offsets to this centerpoint value. CV is bipolar, with every +-2.5 volts corresponding to +-7 density steps. Due to the O&C's CV input range of approximately -3v to +6v, this means that there is an effective CV range of about -8 to +15 density values from the encoder-set center point. Turning the encoder while nonzero CV is applied will

show the encoder's value alongside a "knob" icon momentarily, so the offset can be edited precisely even while CV is modulating it. Note that if the encoder value is set to -7 then the full range of density values is available by applying CV 0-5v.

Scale

Sets the quantization scale to use for the pattern, and makes only those notes available to the pattern generator. It affects the current output CV immediately and applies to the pattern on the next clock.

Root Note

Sets the root note from C to B, transposing the entire pattern. The octave can also be changed by scrolling beyond these notes. Due to space restrictions, the root note is saved but its octave is not.

Pattern Length

Sets the length of the pattern from 1-32 steps. This doesn't alter the pattern apart from setting the loop point.

Step Settings

Like on the TB-303, each step can have Gate, Accent, Glide, +octave or -octave set on it. TB-3PO chooses these values randomly based on the seed and some 303-like rules. Gates emulate the TB-303 by going high for 50% of the detected clock rate, unless they have slide set. In this case they stay high through the next step, when an exponential, fixed-time glide to the next step's pitch is engaged. Accent steps output the gate at 5v instead of 3v, which may be useful with a secondary VCA to punch them a bit.

Visualization

- The heart icon will beat whenever the pattern is reset
- The seed die icon will hop whenever a new seed is randomly picked
- A CV icon next to density indicates that a +- offset is being applied to the encoder-set density value.
- A Knob icon next to density indicates that the encoder-set density value is being shown momentarily.
- A single octave keyboard shows the current playing step's pitch, quantized to semitones
- Icons to the keyboard's right:
  - "!" Indicates that the current step has an Accent and the gate cv will be 5v instead of 3v
  - A note-with-arrow icon indicates a step that has Slide active
  - A wiggly waveform icon shows that an active exponential pitch bend is occurring to reach the current step's pitch
  - UP and DOWN arrows indicate if the current step is transposed up or down by one octave, respectively

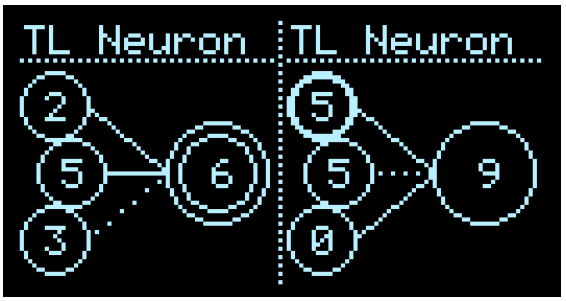
Credits

Authored by Logarhythm1, with modifications by djphazer

Threshold Logic Neuron

Jump to bottom

bowlneudel edited this page 3 weeks ago · 4 revisions



https://youtu.be/NdHY-eDipkY

Threshold Logic Neuron is a three-input programmable logic gate.

I/O

	1/3	2/4
TRIG	Dendrite 1 (True if high)	Dendrite 2 (True if high)
CV INs	Dendrite 3 (True if > 5v)	None
OUTs	Gate if sum of Dendrite weights > Axon threshold	Same

UI Parameters

- Weights (dendrites 1, 2, 3)
- Axon threshold

Threshold Logic Neuron is a three-input/single-output logic gate. Each of three inputs ("Dendrites") is given a weight (in the case of Hemisphere, a weight of between -9 and 9). The output ("Axon") is given a threshold (in Hemisphere, between -27 and 27). When the sum of the weights of high inputs exceeds the threshhold, the output goes high.

The neuron can be used to reproduce common logic gates; for example:

- Dendrite 1: w=3
- Dendrite 2: w=3
- Dendrite 3: w=3
- Axon Output: t=8

This reproduces and AND gate, because all Dendrites need to go high (for a sum of 9) to exceed the threshold of 8.

- Dendrite 1: w=3
- Dendrite 2: w=3
- Dendrite 3: w=3
- Axon Output: t=2

This reproduces an OR gate, because only one Dendrite needs to go high to exceed the threshold of 2.

The values can also be negative. So

- Dendrite 1: w=-3
- Dendrite 2: w=-3
- Axon Output: t=-5

This is a two-input NAND gate, because the output is high (because 0>-5) unless both Dendrite 1 and 2 are high, which brings the sum below the threshold.

But you're not limited to reproducing standard Boolean operations. You can create your own logical operations. For example, you can do this:

- Dendrite 1: w=2
- Dendrite 2: w=2
- Dendrite 3: w=5
- Axon Output: t=3, and mult the output into Dendrite 3

This creates a type of state memory: when Dendrites 1 and 2 go high, the output goes high (2+2>3), and the output is sent back to Dendrite 3, which forces the output to remain high, whatever happens to Dendrites 1 and 2 later.

A variation of that theme is

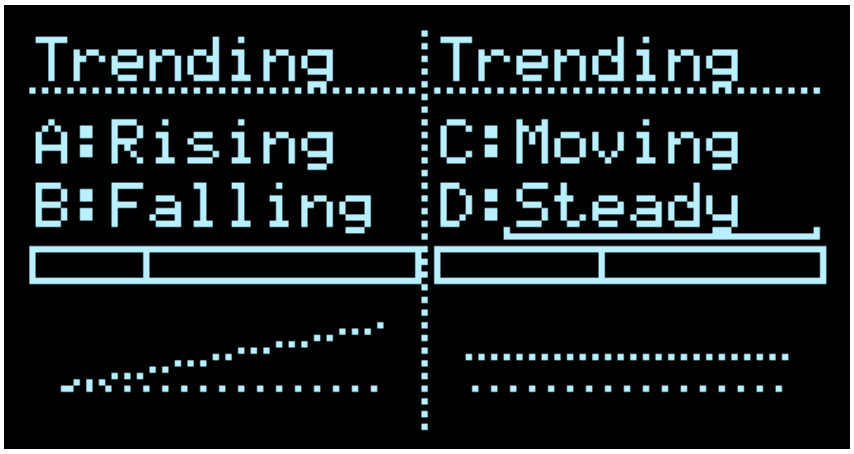
- Dendrite 1: w=2
- Dendrite 2: w=-2
- Dendrite 3: w=2
- Axon Output: t=1, and mult the output into Dendrite 3

Now, Dendrite 1 fires the axon, which feeds back into itself. Dendrite 1 can go low again and the output stays high. But a high signal at Dendrite 2 will reset the memory unless Dendrite 1 is still on.

Trending

Jump to bottom

bowlneudel edited this page 3 weeks ago · 4 revisions



Trending is a dual slope detector with assignable outputs.

I/O

	1/3	2/4
TRIG	N/A	N/A
CV INs	Signal 1	Signal 2
OUTs	Gate / trigger Ch A (corresponding to Signal 1)	Gate / trigger Ch B (corresponding to Signal 1)

UI Parameters

- Channel A mode
- Channel B mode
- Sensitivity

Output modes:

- Rising: The assigned output is a gate, which is high when the signal is rising
- Falling: The assigned output is a gate, which is high when the signal is falling
- Steady: The assigned output is a gate, which is high when the signal is steady
- Moving: The assigned output is a gate, which is high when the signal is rising or falling
- ChgState: The assigned output emits a trigger when the signal changes from one state to another (e.g., rising to steady, steady to falling, falling to rising, etc.)
- ChgValue: The assigned output emits a trigger when the signal changes its value by more than 1/4 semitone (or about .02V)

Sensitivity

The sensitivity control can be used to fine-tune the response of the slope detector. At lower settings, the detector will respond more slowly, but will be more consistent. At higher settings, the detector will respond faster, but may change direction more erratically.

Trigger Sequencer

Jump to bottom

bowlneudel edited this page 3 weeks ago · 4 revisions

https://youtu.be/qPRspJApd1Y

TrigSeq is a dual eight-step trigger sequencer with shared Clock and Reset.

I/O

	1/3	2/4
TRIG	Clock	Reset
CV INs	Swap Channels if > 3v	Offset position for Reset (square outline)
OUTs	Ch A Trigger	Ch B Trigger

UI Parameters

- Edit steps (cursor selects 4 steps at a time)
  - Ch 1 1st half
  - Ch 1 2nd half
  - Ch 2 1st half
  - Ch 2 2nd half
- Ch 1 length
- Ch 2 length

Step Editing

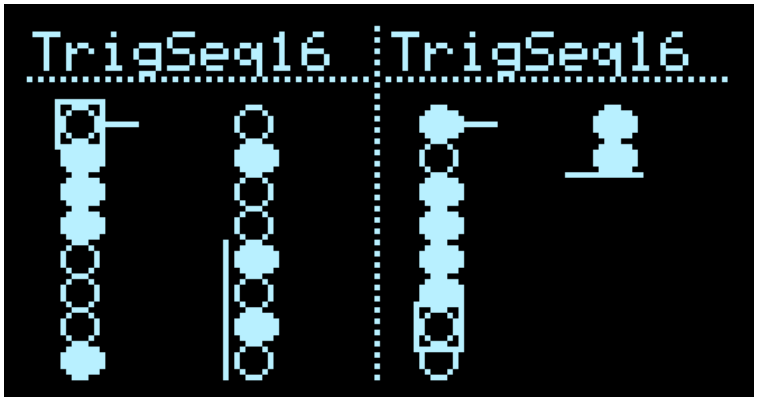
The cursor appears over four steps at a time. Turning the encoder selects a binary representation of the bit pattern. Each half of the sequence has sixteen possible values. That is, turning the encoder clockwise will cycle through

- (silence)
- ---X
- --X-
- --XX
- -X--
- -X-X
- -XX-
- -XXX
- X---
- X--X
- X-X-
- X-XX
- XX--
- XX-X
- XXX-
- XXXX

Trigger Sequencer 16

Jump to bottom

bowlneudel edited this page 3 weeks ago · 4 revisions



TrigSeq16 is a 16-step trigger sequencer. It's visually and functionally similar to Trigger Sequencer, except it's 1x16 instead of 2x8. Solid steps send triggers on Out A/C; empty steps send triggers on Out B/D.

I/O

	1/3	2/4
TRIG	Clock	Reset
CV INs	Swap channels if > 3v	Offset position for Reset (square outline)
OUTs	Trigger output	NOT Trigger output

UI Parameters

- Edit sequence (cursor selects 4 steps)
- Length

Step Editing

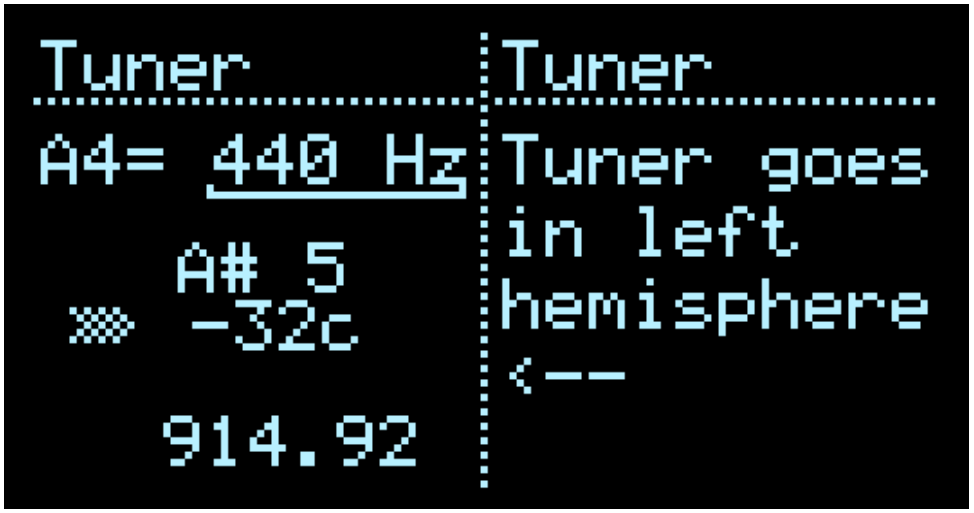
The cursor appears over four steps at a time. Turning the encoder selects a binary representation of the bit pattern. Each quarter of the sequence has sixteen possible values. That is, turning the encoder clockwise will cycle through

- (silence)
- ---X
- --X-
- --XX
- -X--
- -X-X
- -XX-
- -XXX
- X---
- X--X
- X-X-
- X-XX
- XX--
- XX-X
- XXX-
- XXXX

Tuner

Jump to bottom

bowlneudel edited this page 3 weeks ago · 5 revisions



Tuner is a chromatic tuner with adjustable A4 setting.

Important: Tuner can only measure frequencies on the physical TR4 input on Teensy 3.2-based hardware. It will remind you which side it needs to be on. ;)
(pictured above running in a FLIP_180 build, upside-down orientation)

The underlying FreqMeasure driver is currently unimplemented on Teensy 4.x...

I/O

	1/3	2/4
TRIG	Signal input	N/A
CV INs	N/A	N/A
OUTs	N/A	N/A

UI Parameters

- Set A4 frequency (press to reset to A=440hz)

The A4 frequency has a range of 400-500Hz. The closest note is displayed under the frequency. Under that, it shows how far off from that note in cents the input is. When the note is in tune, it will become highlighted.

If Tuner seems to stop responding, push the encoder button to reset it.

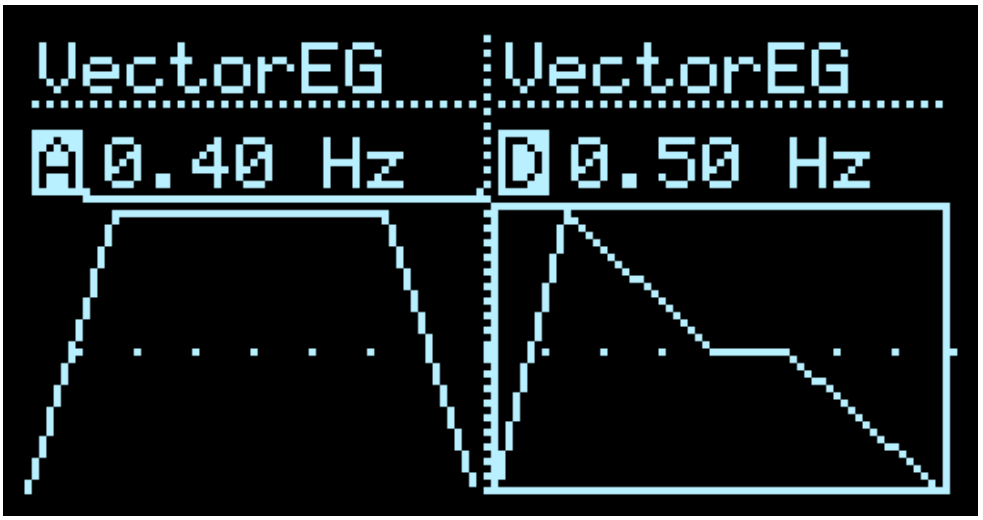
Yeah, that's about it. It's a tuner.

Tuner is available from Hemisphere Suite v1.4.

VectorEG

Jump to bottom

bowlneudel edited this page 3 weeks ago · 5 revisions



VectorEG is a dual AR envelope generator based on Vector Oscillator waveforms. There is a variety of built-in waveforms from which to choose, or you can create your own with the Waveform Editor.

I/O

	1/3	2/4
TRIG	Gate Ch1	Gate Ch2
CV INs	N/A	N/A
OUTs	Ch1 EG	Ch2 EG

Controls

- Channel 1 EG Frequency
- Channel 1 waveform selection
- Channel 2 EG Frequency
- Channel 2 waveform selection

The EG runs freely until it gets to the second-to-last segment, and sustains at that point. When released, it proceeds to the level of the final segment at the speed specified by the final segment.

Unlike a traditional envelope generator, the speed of the envelope is determined with a single frequency control. The higher the frequency, the faster the envelope will run.

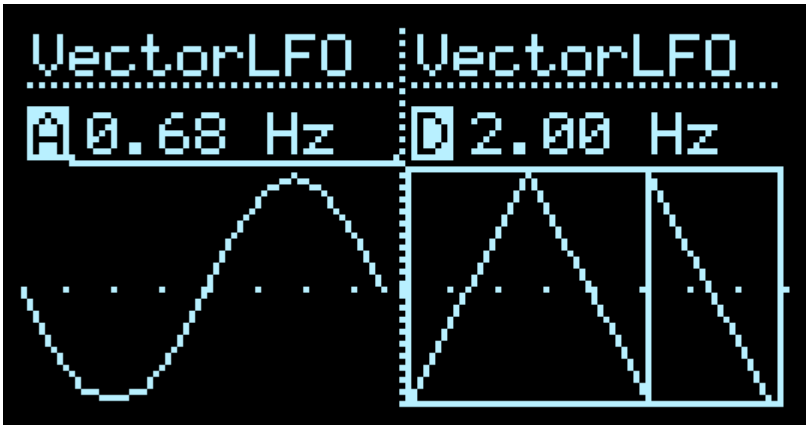
The outputs are uni-polar, and with positive offset, so that a level of -128 in the Waveform Editor corresponds to 0V, and a level of 127 is 5V.

VectorEG is in Hemisphere Suite starting with v1.6.

VectorLFO

Jump to bottom

bowlneudel edited this page 3 weeks ago · 4 revisions



Vector LFO is a dual low-frequency oscillator based on Vector Oscillator waveforms. There is a variety of built-in waveforms from which to choose, or you can create your own with the Waveform Editor.

I/O

	1/3	2/4
TRIG	Sync Clock Ch1	Sync Clock Ch2
CV INs	Freq mod Ch1	Mix Ch1 into Ch2
OUTs	LFO Ch1	LFO Ch2 + Ch1 mix

UI Parameters

- LFO 1 frequency
- LFO 1 waveform select
- LFO 2 frequency
- LFO 2 waveform select

The LFO range is from .10Hz to 999Hz.

Output A/C and LFO 1

Output A/C is a the output of only LFO 1. LFO 1 can be synchronized with an external clock at Digital 1, and its frequency can be modulated over a 30Hz bi-polar range with CV 1.

Output B/D and LFO 2

Output B/D is a mixed output. By default, it is 100% LFO 2. Increasing voltage to CV 2 mixes in the LFO 1 signal up to a max of 50%/50% of each.

Manual Oscillator Reset

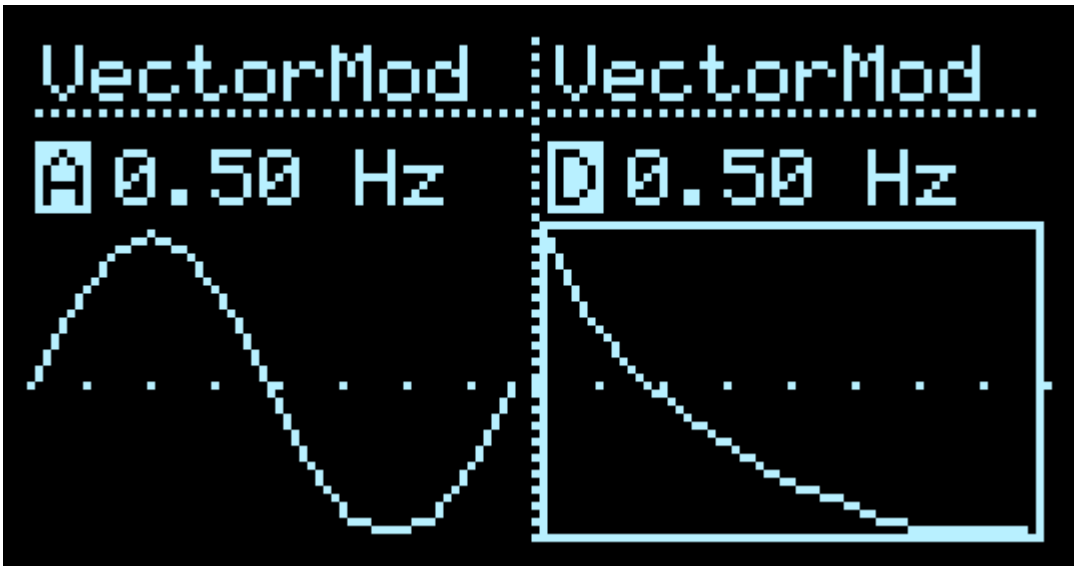
When a new waveshape is selected from the panel, both LFO 1 and LFO 2 will be reset to the beginning of their respective cycles.

VectorLFO is in Hemisphere Suite starting with v1.6.

VectorMod

Jump to bottom

bowlneudel edited this page 3 weeks ago · 4 revisions



VectorMod is a dual triggered one-shot or cycling modulation source based on Vector Oscillator waveforms. There is a variety of built-in waveforms from which to choose, or you can create your own with the Waveform Editor.

I/O

	1/3	2/4
TRIG	Trigger Ch 1	Trigger Ch 2
CV INs	Cycle Ch 1 > 3v	Cycle Ch 2 > 3v
OUTs	Ch 1	Ch 2

Controls

- Mod 1 frequency
- Mod 1 waveform select
- Mod 2 frequency
- Mod 2 waveform select

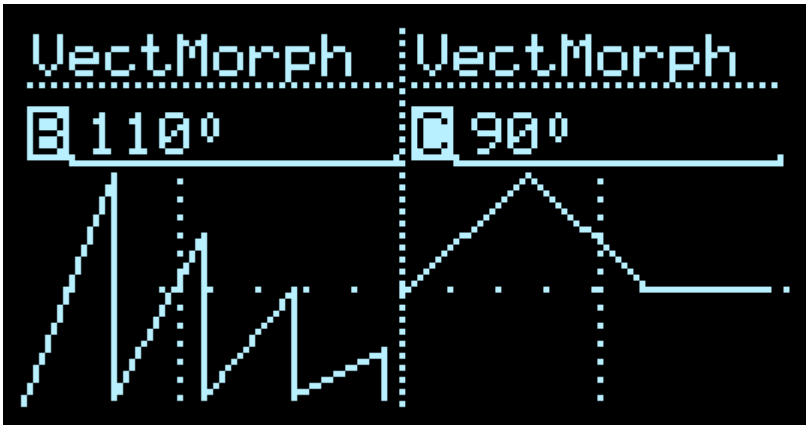
The speed of the Modulator is determined with a single frequency control. The higher the frequency, the faster the Modulator will run.

VectorMod is in Hemisphere Suite starting with v1.6.

VectorMorph

Jump to bottom

bowlneudel edited this page 3 weeks ago · 4 revisions



https://youtu.be/WrENiAQHhq0

VectMorph is a morph controller inspired by the Doepfer A-144, and based on Vector Oscillator waveforms. There is a variety of built-in waveforms from which to choose, or you can create your own with the Waveform Editor.

I/O

	1/3	2/4
TRIG	N/A	N/A
CV INs	Phase Ch 1 (and Ch 2 when linked)	Phase Ch 2
OUTs	Ch 1	Ch 2

UI Parameters

- Channel 1 Phase Offset
- Channel 1 Waveform selection
- Channel 2 Phase Offset
- Channel 2 Waveform selection

The Vector Morph Controller (VectMorph) is a modulation source for sending phase-offset output to one or more destinations. A typical use case would involve sending overlapping phased triangular waves to control a voltage-controlled mixer.

Each channel can be assigned a Phase Offset, in degrees. This offset is added to the voltage-determined phase to determine the current amplitude of the channel's output.

Linked Mode

By default, CV 1 controls the phase of both channels. This is Linked Mode. You can use two instances of the Vector Morph Controller, one in each hemisphere, to control up to four things. To use Linked Mode in both hemispheres, mult one hemisphere's CV 1 input to the other hemisphere's CV 1 input (as demonstrated in the video).

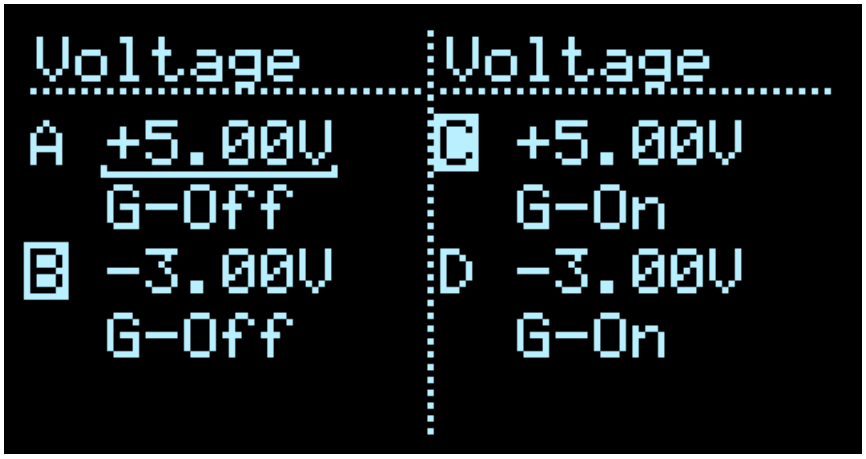
If you wish to control the phase of each channel independently, just send some voltage to CV 2. This will disable Linked Mode, and the Link icon will disappear. To return to Linked Mode, stop sending voltage to CV 2, and use the encoder to change the waveform of either channel.

Vector Morph Controller is in Hemisphere Suite starting with v1.6.

Voltage

Jump to bottom

bowlneudel edited this page 3 weeks ago · 5 revisions



Voltage is a dual gate-activated fixed-voltage emitter.

I/O

	1/3	2/4
TRIG	Gate Ch1	Gate Ch2
CV INs	N/A	N/A
OUTs	Voltage Ch1	Voltage Ch2

UI Parameters

- Ch1 Voltage
- Ch1 Gate behaviour
- Ch2 Voltage
- Ch2 Gate behaviour

Notes

The voltage per channel is selectable in 1 semitone (approx. .08V) increments. The available range is the min/max your hardware supports. On VOR-capable units, this depends on the current Vbias setting (default -5V to +5V).

For output, there are two gate states available:

- **G-On**: When the gate of the corresponding digital input is high, the output is the specified voltage. Otherwise, it's 0V.
- **G-Off**: When the gate of the corresponding digital input is low, the output is the specified voltage. Otherwise, it's 0V.

The indicator over the output name (A,B,C,D) will display when the corresponding output is emitting non-zero voltage.

Credits

Adapted from Voltage by Chysn; available in Hemisphere Suite from v1.4.