expert sleepers disting NT

User manual

Version 1.17

And you may ask yourself, "How do I work this?"

- *Talking Heads, “Once in a Lifetime”*

Table of Contents

Introduction....8
Preamble....8
What does it do?....9
What is it not?....9
Getting started....10
Example presets....10
Installation....10
Controls....16
Navigation....16
Inputs and outputs....20
USB....21
MIDI connections....21
MicroSD card....22
Signal flow....29
Algorithm specifications....30
Sample rate and buffer size....31
CPU usage/overload....31
Fault handler....32
Preset editor/routing analyser....33
NT Helper....34
Performance page....35
Menu reference....38
Algorithms menu....39
Presets menu....40
Mappings menu....43
Performance page menu....47
Settings menu....48
Misc(ellaneous) menu....54
UI Scripts menu....57
Algorithm-specific menu....57
Common algorithm features....58
Common polysynth features....60

Plug-ins..... 67
Algorithm reference..... 70
Accent sweep..... 71
Arpeggiator..... 73
Attenuverter..... 79
Audio recorder..... 80
Augustus Loop..... 82
Auto-calibrator..... 89
Auto-sampler..... 92
Bit Crusher..... 95
Chaos..... 97
Chorus/Flange..... 99
Chorus (Vintage)..... 101
Clock..... 102
Clock Divider..... 106
Clock Multiplier..... 108
Compressor..... 109
Convolver..... 111
Crossfader..... 114
Debouncer..... 116
Delay (Mono)..... 117
Delay (Stereo)..... 119
Delay (Tape)..... 121
Delayed Function..... 123
Display Awakener..... 126
DJ Filter..... 127
Dream Machine..... 129
EQ Parametric..... 133
Envelope (AR/AD)..... 135
Envelope (DAHDSR)..... 138
Envelope Follower..... 142
Envelope Sequencer..... 143
ES-5 Encoder..... 147

ESX-8CV Combiner..... 148
Euclidean Patterns..... 150
Feedback Receive/Send..... 153
Filter bank..... 155
Frequency Shifter..... 158
Granulator..... 160
Kirbinator..... 169
Knob Recorders..... 175
LFO..... 177
Linear/Exponential..... 180
Little Spacey..... 182
Logic..... 186
Looper..... 189
Looper (MicroSD)..... 201
Lua Expression..... 203
Lua Script..... 205
Macro Oscillator..... 207
Macro Oscillator 2..... 210
MIDI CC Generator..... 213
MIDI Filter..... 215
MIDI Player..... 217
MIDI Triggers..... 221
Minimum/maximum..... 223
Minky Starshine..... 224
Mixer Mono..... 232
Mixer Stereo..... 234
More Like Space..... 236
Multi-Switch..... 238
Noise gate..... 241
Noise generator..... 243
Notes..... 244
Oscilloscope..... 245
Phase Shifter..... 247

Phaser....249
Pitch reference....251
Pitch Shifter....252
Poly CV....254
Poly FM....258
Poly FM (Editable)....261
Poly Macro Oscillator 2....263
Poly Multisample....267
Poly Multisample (legacy)....272
Poly Resonator....275
Poly Wavetable....279
Quadraphonic Mixer....286
Quantizer....290
Rectifier....296
Resonator....297
Reverb....300
Reverb (Clouds)....302
Rotary....304
Sample and Hold....306
Sample Player....307
Sample Players....310
Sample Player (Clocked)....317
Sample Player (Scrub)....320
Sample Player (Sliced)....322
Sample Player (Speed)....326
Saturation....327
Seaside Jawari....328
Shift Register Random....331
Slew rate limiter....334
Slope Detector....336
Spectral Conquest....338
Spectral Freeze....343
Spectral Vocoder....347

Spectrum Analyzer....349
SSB Modulator....350
Step Sequencer....352
Step Sequencer Head....359
Stopwatch....362
Temporal Freeze....364
Temporal MIDI Quantizer....370
Texture Synthesizer....372
Three Pot....374
Tracker....378
Transient Detector....382
Tuner (fancy)....383
Tuner (simple)....385
USB audio (from host)....386
USB audio (to host)....387
VCA/Multiplier....388
VCF (State Variable)....389
VCF (24dB/oct LP)....392
VCO - Pulsar....394
VCO - Waveshaping....397
VCO - Wavetable....400
Vocoder....402
Vowel Filter....404
Waveform Animator....406
Waveshaper....408
Wavetable Waveshaper....410
UI Scripts....412
MIDI SysEx reference....413
I2C reference....424
Updating the firmware....427
Acknowledgments....433

Introduction

Congratulations on your purchase of an Expert Sleepers disting NT. Please read this user manual before operating your new module.

A note on navigating this manual

When one part of the manual refers to another, it may say something like “see Settings, below”. In such cases the word 'below' (or 'above') is a hyperlink, and can be clicked on. Try it.

It seems very technical

This is intended as a reference manual, not a tutorial. Please visit our YouTube channel for some friendlier demos, or just ask for help.

Where to get help

Email, forum, and social media links can be found at the bottom of every page on our website[1]. We also have a Discord server[2].

We prefer support requests to be made through the forum, if possible – failing that, by email.

If you’re lucky enough to have a local bricks and mortar modular store, please visit them, try things out, and get help there. And most of all, buy your modules from them too.

Preamble

The disting NT is the latest in the line of modules from Expert Sleepers that started with the original disting in 2014, which performed one of 16 very different functions at the turn of a knob. This concept has evolved through the disting mk3 (2015) – which added the possibility of firmware updates, and so a continually improving selection of functions – and the disting mk4 (2017) – which added a display, and brought the MicroSD card slot onto the front of the module, emphasising its increased importance for functions such as sample playback – to the disting NT’s predecessor, the disting EX (2020). The EX (finally) added an OLED display, and a much more powerful CPU platform, taking the functions it could perform to the next level.

Ever since the mk3 was released, the modules’ firmwares have been continuously updated with new features, making them even more useful and better value.[3]

We aim to continue this legacy with the disting NT.

Whereas the disting EX built directly on the selection of functions in the mk4, and then added some of its own, with the disting NT we’ve taken everything back to square one. We anticipate (eventually)

1 https://www.expert-sleepers.co.uk/

2 https://modwiggler.com/forum/viewtopic.php?t=287914

3 Some stats – the mk3 had 10 major updates in 2 years; the mk4 has (so far) had 64 major updates in 7 years; the EX has (so far) had 25 major updates in 4 years.

adding every function ever offered by the previous distings, but every algorithm has been reconsidered and reappraised in the light of the possibilities offered by the new hardware platform. Often things have been streamlined and simplified – for example, a disting EX algorithm that offered a polysynth, a chorus, and a delay would be implemented on the NT as three separate functions, which you could combine as you choose – each function therefore being smaller and easier to deal with.

While the exact future of this module is yet to be written, one thing is certain – it will be driven by user feedback. We have an active user community on the ModWiggler forum[4], and suggestions (and we'll admit, bug reports) left there have often made it into the firmware. So we strongly encourage you to get in touch! Let us know what the disting NT can be for you.

What does it do?

The module takes audio and/or CV input, runs a bunch of software processing, and generates audio and/or CV output. It does this continuously.

The chunks of software processing are called *algorithms*. You can think of them much as you would think of plug-ins in your DAW. The module has a number of algorithms to choose from, and you can add the ones you want in a big list and the module will run them all.

A collection of algorithms, their various parameters, and associated data such as MIDI mapping, together make up a *preset*.

Like plug-ins in a DAW, we use 'algorithm' to mean both the abstract *type* of processing you want to run, and a particular *instance* of that algorithm in a preset. For example, you might say "I'll load a reverb plug-in on this track" meaning the *type* of plug-in you want to load, and also "I changed the decay time on the reverb plug-in" to refer to the particular *instance* of the reverb plug-in.

What is it not?

The disting NT is not a "virtual modular". There are no virtual cables flying around. It simply collects together a variety of functions that are useful in the context of a modular synthesizer and makes it possible to use a number of them at once.

It is not a "patch player". It is not the intention that you do all the complex setup on a computer and then copy it over to the module. The module is designed to be used *in a modular*, with a UI that makes it fluid to do everything in the module itself.

It is not only useful via MIDI. The large number of inputs (and outputs) makes it possible to use it entirely via CV, interfacing with the rest of your modules in a natural way. (That said, it does also make a really good MIDI polysynth.)

4 https://www.modwiggler.com/forum/viewforum.php?f=35

Getting started

First, carefully read the installation instructions below.

Then, at least skim the ‘Navigation’ section.

This will enable you to load some of the example presets from the MicroSD card.

You might like to watch this quickstart video[5] before going much further, which was made to help get new users up and running.

Please also check if you have the latest firmware, which can be found on our website here[6].

Example presets

Some example presets are included on the MicroSD card supplied with the module, but by the time you read this there will almost certainly be more available on our GitHub[7]. Please look there for the latest examples, and for documentation of how the presets are put together, which can be quite informative.

Installation

House the module in a Eurorack case of your choosing. The power connector is 16-pin Doepfer standard[8]. If using the power cable supplied with the module, the red edge of the cable is furthest from the top edge of the PCB, and carries -12V. ("-12V" is marked on the PCB itself next to this end of the connector.) Be sure to connect the other end of the power cable correctly, again so -12V corresponds to the red stripe on the cable.

Please observe ESD (electro-static discharge) precautions when handling the module (and indeed, any module).

5 https://www.youtube.com/watch?v=EfkrNHkTKF0

6 https://expert-sleepers.co.uk/distingNTfirmwareupdates.html

7 https://github.com/expertsleepersltd/distingNT/tree/main/presets

8 http://www.doepfer.de/a100_man/a100t_e.htm

Please do not operate the module without it being securely fastened in a case.

Physical dimensions

The disting NT is 22HP wide and 25mm deep (including the depth added by the included power cable).

Power requirements

The disting NT draws 211mA on the +12V rail, and 96mA on the -12V rail, when idling. The actual power consumption is heavily dependent on how many input and output socket LEDs are lit. Each fully illuminated socket draws about 4mA (from the +12V rail or -12V rail depending on the polarity), so you could in theory pull another 72mA, but that is highly unlikely to occur in actual use – you'd have to somehow feed a steady 10V into every input socket and be running algorithms that were outputting a steady 10V from every output. When budgeting for case power it's probably safe to split that 72mA between the two rails and say it draws 247mA on the +12V rail and 132mA on the -12V rail. But really, if that 36mA is enough to put your PSU over the edge, you need a bigger PSU anyway.

The disting NT does not use the 5V rail.

Connecting expansion modules

Turn off the power before connecting or disconnecting expansion modules.

ES-5

Connect an ES-5[9] module via the header on the back of the disting NT marked J11 ES-5, using the 10-way cable provided with the ES-5. The red stripe should be oriented down on both modules, as shown in the photo below, and in the ES-5 user manual[10]. At the ES-5 end, connect the cable to the header marked "GT7/To ES-3".

The disting NT can use an attached ES-5 to output a considerable number of extra CVs and gates,

9 https://www.expert-sleepers.co.uk/es5.html

10 https://expert-sleepers.co.uk/es5usermanual.html

from the ES-5 itself and from attached ESX-8GT[11] and ESX-8CV[12] expanders. Some algorithms may support direct use of the expanders, but in general any signals can be passed to the expanders via the ES-5 Encoder and ESX-8CV Combiner algorithms (below).

NTX-8CV

Connect an NTX-8CV[13] module via the header on the back of the disting NT marked J11 ES-5, using the cable provided with the NTX-8CV. The red stripe should be oriented down on both modules, as shown in the photo above, and in the NTX-8CV user manual. At the NTX-8CV end, connect to the header marked J4 “IN from host”.

Up to eight NTX-8CV modules can be daisy-chained, each one adding eight CVs outputs.

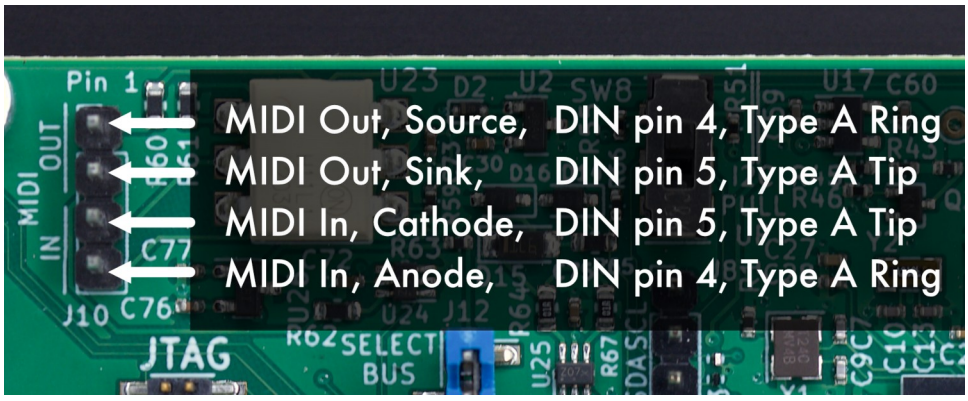
The NTX-8CVs need to be enabled, via the Engine settings menu, below.

MIDI

Connect a MIDI breakout to the header on the back of the disting NT marked J10 MIDI.

MIDI can be used to remotely control the algorithm parameters, to share clocks, and to play notes in the synthesizer algorithms etc. The disting NT can also send MIDI to other devices, for example to sequence them, or simply to provide parameter feedback to a MIDI controller.

The image below shows how the header is wired, including pin numbers for 5-pin DIN connections, and tip/ring indications for a Type A TRS MIDI jack.



See below for more details on hooking up MIDI using the included breakout module.

I2C

A device or module that communicates via I2C can be connected via the header marked J8. The SDA and SCL pins are marked; the third pin is ground.

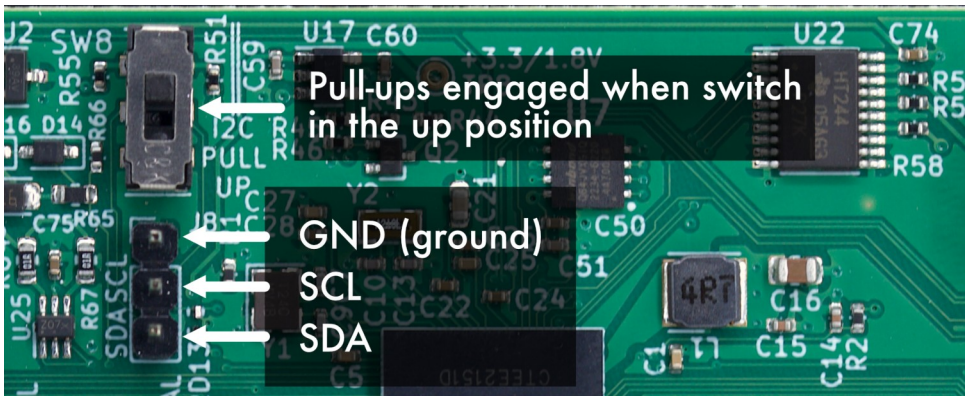
Like MIDI, I2C can be used for remotely controlling algorithm parameters and playing notes. While

11 https://expert-sleepers.co.uk/esx8gt.html

12 https://expert-sleepers.co.uk/esx8cv.html

13 https://expert-sleepers.co.uk/ntx8cv.html

less common than MIDI, it is the control mechanism of choice for the whole Teletype[14] module ecosystem.



CVM-8

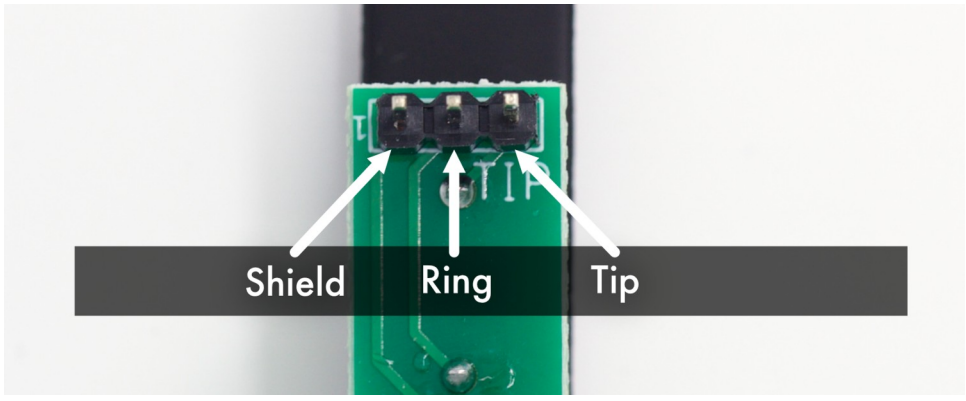
If you feel the need for more CV inputs on the disting NT, please take a look at the CVM-8[15], which has eight inputs and can connect via MIDI, I2C, or the Select Bus.

Using the included breakout module

The disting NT is supplied with a “Tiny MIDI Breakout” (aka TMB), a pre-existing Expert Sleepers product, which is simply six TRS jack sockets on a 2HP panel. The product page for the TMB is here[16]. In the box are also some jump cables (aka “DuPont cables” or “Arduino cables”) which you can use to connect the two.

If you need more, or longer, cables they are easily found online.

Each of the six TRS sockets is wired to a 3-pin header on the back of the PCB, like so:



Use the jump cables to connect the relevant pins of the TRS sockets to the pins of the breakout headers on the disting NT that you would like to use.

14 https://monome.org/docs/teletype/

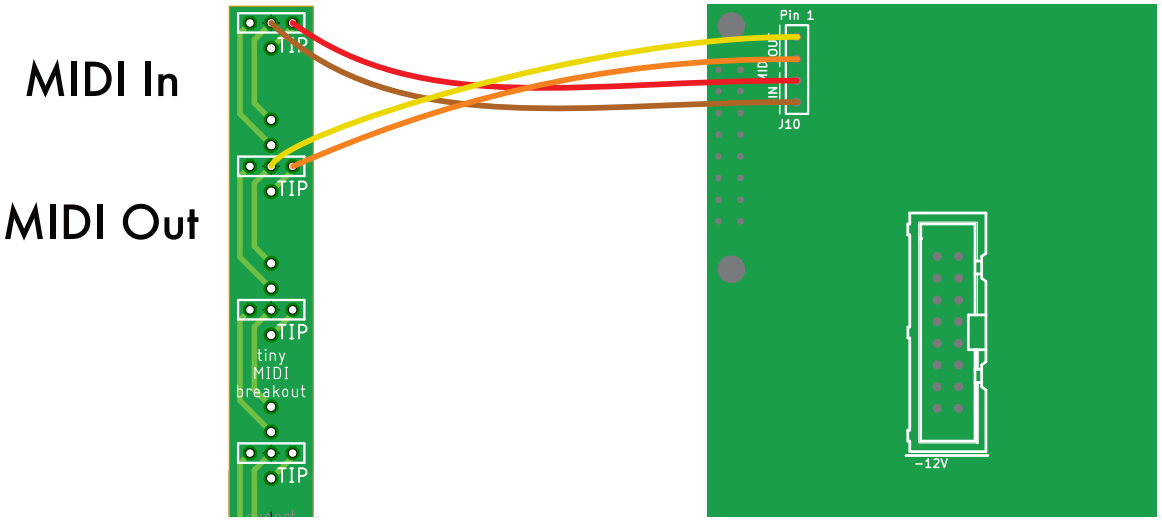
15 https://expert-sleepers.co.uk/cvm8.html

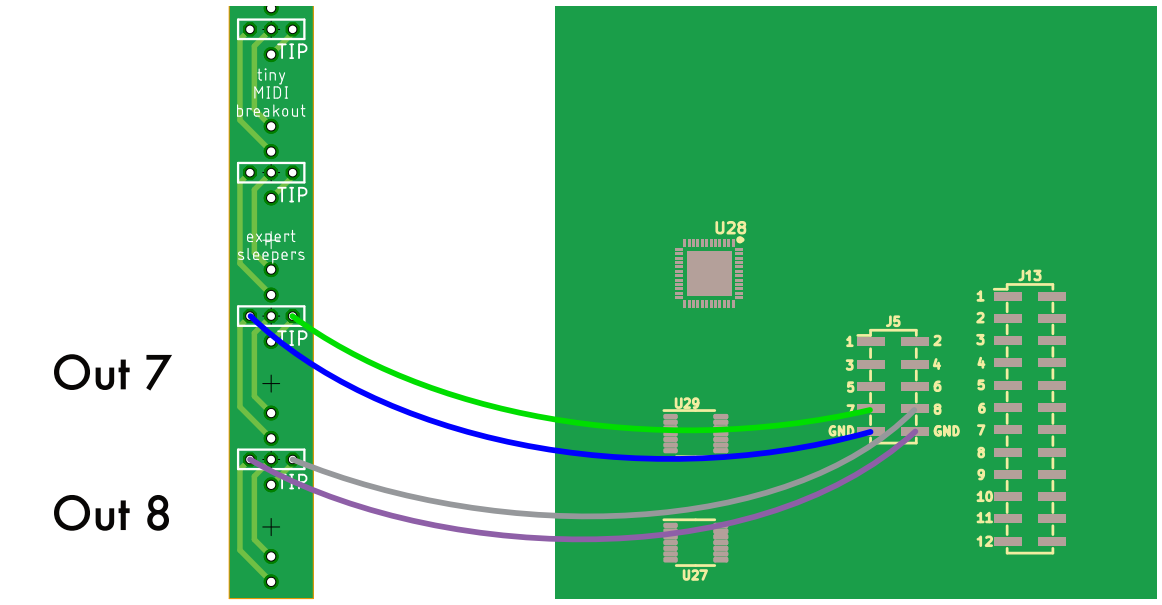
16 https://expert-sleepers.co.uk/tinymidibreakout.html

For example, to use one of the sockets as a TRS MIDI input, use two cables to connect the Tip and Ring of the socket to the MIDI In Tip & Ring as shown above.

There is a video showing how to connect the breakout here[17].

The diagrams below shows typical connections when using the breakout for MIDI and audio. The wire colours match those shown in the video.

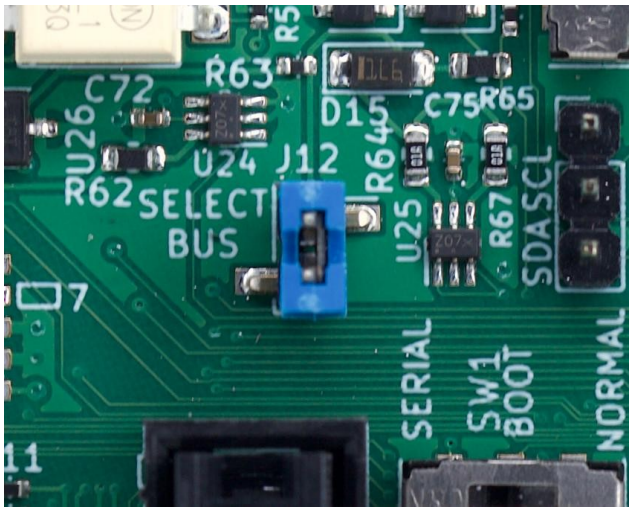




17 https://www.youtube.com/watch?v=NxiFBvlkGgo

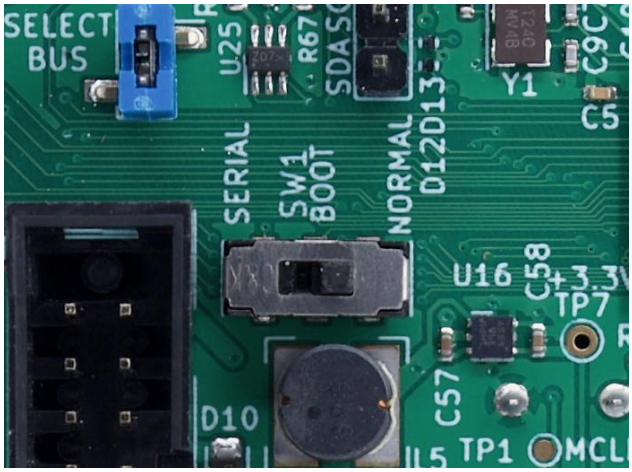
Jumpers/switches

The jumper on the board labelled J12 SELECT BUS connects the module's Select Bus circuitry to the power bus's CV line. Fit the jumper if you want to use the Select Bus; remove it if you use the CV bus for its originally intended purpose.



Use the switch marked SW8 I2C PULL UP to enable or disable the I2C pull-up resistors (image above). The pull-ups are enabled when the switch is in the 'up' position (nearest the top of the PCB). At least one device on the I2C bus needs to have pull-up resistors enabled. It may work fine with more than one set of pull-ups enabled, or it may not.

The switch marked SW1 BOOT can be used to force the module into serial bootloader mode. Normally you would leave the switch in the "Normal" position (to the right) and use the module's menu to enter bootloader mode. If however this is not possible, put the switch into the "Serial" position (to the left) to force it into bootloader mode. Note though that you will then have to return it to "Normal" to boot normally after installing new firmware. See below for a description of the usual firmware update procedure.

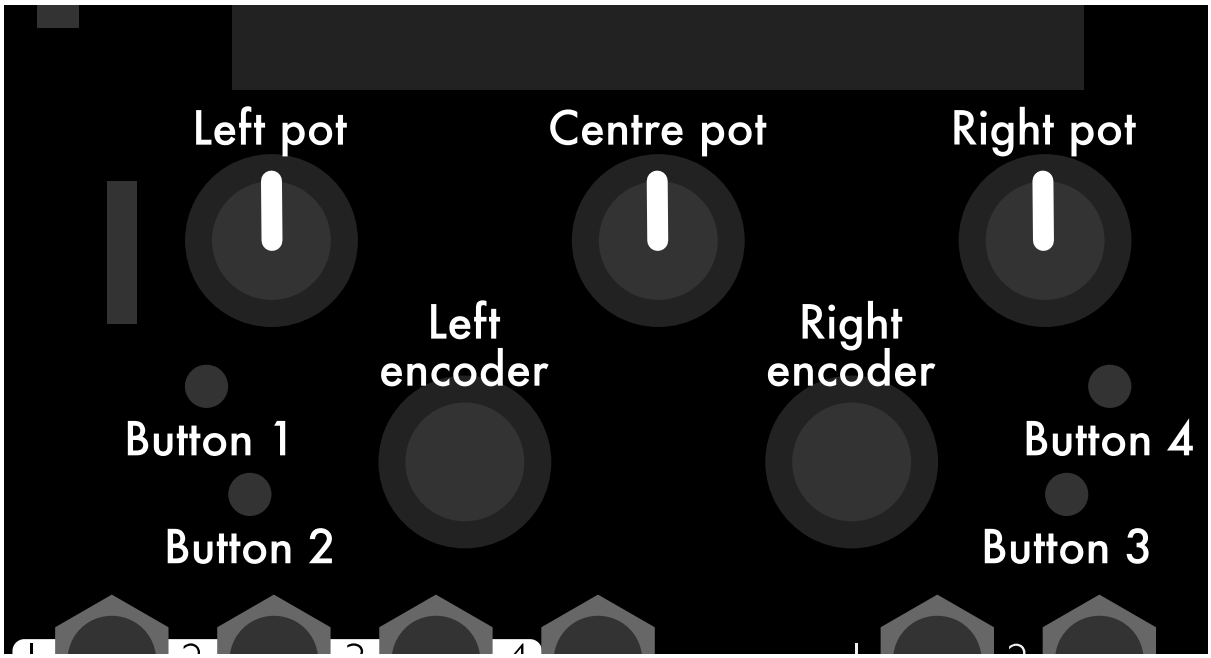


Controls

The disting NT has three pots, two encoders, and four pushbuttons. The pots and encoders also have a pushbutton action.

The controls are unlabelled, on the basis that their function can be and is completely redefined in software. Popup help can be displayed on the display to help you keep track.

In this manual, we will simply refer to “the left pot”, “the centre pot”, and “the right pot”; similarly, “the left encoder” and “the right encoder”. The pushbuttons are referred to as buttons 1-4, ordered from left to right.



There is a setting you can change to order the buttons in the opposite direction i.e. from right to left. See below.

Navigation

There are two states that the module UI will be in most of the time: the algorithm view, in which you navigate the various active algorithms and interact with their parameters; or the menu system.

Pop-up help is available, indicating the function of each control in a small area at the bottom of the screen. If you’ve disabled it in the Settings, you can still view the help at any time by holding down button 3.

In the pop-up help, words at the extreme left and right sides refer to the pushbuttons. Other help is, as far as space will allow, lined up above the encoder or pot it refers to. In the image above, ‘Menu’,

‘Params’, ‘Help’, and ‘Shift’ refer to pushbuttons 1-4 respectively. ‘Turn:Select’ and ‘Push:Algorithm’ both refer to the left encoder.

Menus

You can easily tell when you’re in the menu system because there will be a dotted line border around the screen.

Button 1 enters the menu system; it also immediately exits the menu system if pressed while you’re anywhere in the menus.

The left encoder navigates the menu: turn it to select a submenu or menu item, and press it to descend one level. Button 2 is ‘back’ i.e. it goes up one menu level. If you keep pressing button 2 you will eventually leave the menu entirely.

Where a menu allows you to change a value, or select from a list of items, you can in general do this either by turning the right encoder, or by pressing the left encoder to descend one more menu level and then turning the left encoder.

There is a setting to swap the function of the two encoders, if you find it preferable to primarily drive the menu with the right encoder. See below.

File browser

Some menus, for example the menu for loading presets, require you to browse for a file on the MicroSD card.

In the file browser, use the left encoder to move up and down the list of files, to select items, and to enter folders. Press the right encoder to leave a folder and go up one level of the folder hierarchy.

Editing text

Some menus, for example the menu to set the preset name, allow you to edit text.

- Turn the left encoder to move the cursor, and the right encoder to change the character at the cursor position.
- Pressing the right encoder advances to the next character in the list ‘ ’, ‘0’, ‘A’, ‘a’.
- Pressing the right pot inserts a space at the cursor position and moves following characters to the right.
- Pressing the left pot deletes the character at the cursor position and moves following characters to the left.

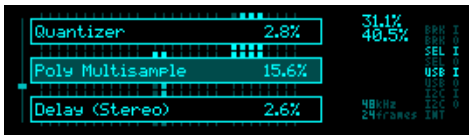
- Press the left encoder to accept the text and return to the menu above.

Algorithms view

This is where you'll spend most of your time interacting with the module. There are two primary display modes – the algorithms overview, where you can see all the running algorithms and the signals flowing between them, and the single algorithm view, where you edit the parameters of an algorithm.

Algorithms overview

Here you can see all the algorithms running, in a list starting with the module inputs at the top, and flowing down to the module outputs at the bottom.



Turning the left encoder or the left pot scrolls through the algorithm list.

The 'current algorithm' is the one highlighted, in the centre of the screen. This is the algorithm that will be the target of any menu commands e.g. Move up/down. It is also the algorithm that will be live for editing when you switch to the single algorithm view.

There is a 'scroll bar' graphic on the left side of the screen showing where you are in the list.

Holding button 4 and turning the left encoder moves the current algorithm up and down in the list, exactly as if you'd used the 'Move algorithm up/down' menu.

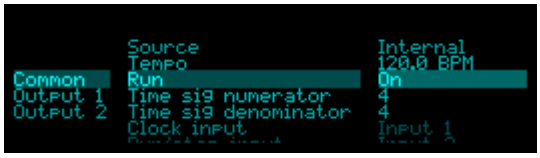
Holding button 4 and pressing the right encoder toggles the Bypass parameter of the current algorithm.

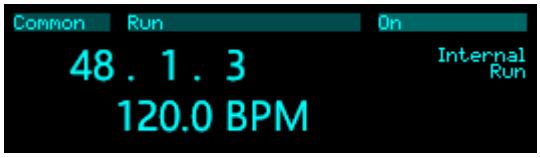
Press the right encoder to switch to a different view showing the input and output signals of the current algorithm. Turning the encoder or pots still scrolls through the list as before.

Press the left encoder or left pot to switch to the single algorithm view.

Single algorithm view

This is where you edit the parameters of an algorithm. There are two modes for this view: one which shows only parameters, arranged in pages, and one which shows a single parameter and devotes the rest of the screen to a display specific to the algorithm in question.





In either mode, navigation is exactly the same.

Turn the left pot to select the page of parameters to view.

Turn the centre pot or the left encoder to select a parameter within the current page.

Turn the right pot or the right encoder to change the parameter value.

The pot uses a form of ‘soft takeover’ rather than jumping to the value that corresponds directly to the knob position. When a new parameter is selected for editing, turning the pot clockwise will always increase the value, and turning it anticlockwise will always decrease the value, wherever the pot happens to be.

Pressing the right pot causes the parameter value to jump directly to the position of the knob, bypassing the soft takeover. This can be useful, say, if you want to set a number of parameters to a similar value. For example, if you wanted to set a number of mixer channel gains to the same value, you could turn the right pot to set the first value, then step through the other channels with the centre pot and press the right pot to jump the values to the right area, without having to turn the pot a considerable distance for each parameter.

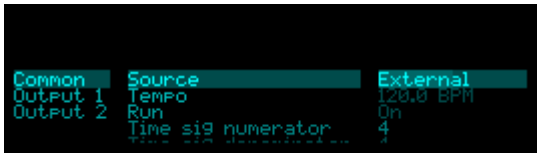
Pressing the right encoder sets the parameter to its default value, with some exceptions:

- For parameters with a ‘Confirm’ function, for example the wavetable selection of the Poly Wavetable algorithm, pressing the right encoder confirms the new value.
- For string parameters, pressing the right encoder switches to a view where you can edit the string. See the notes on editing text, above.
- For parameters specified in dB, pressing the right encoder sets the parameter to its default, or if the parameter is already at minimum (usually -∞dB), to 0dB.
- For parameters specified in %, if the parameter is at 0%, pressing the right encoder sets it to 100%, and if it’s at 100%, to 0%.

Pressing button 4 is a shortcut to the Mappings menu. If you prefer, you can change this in the Settings to go directly to one of the CV, MIDI, or I2C Mappings submenus instead.

Press the left encoder or the left pot to switch to the algorithm overview. You can also hold down the left encoder or left pot and turn it while held to select a different algorithm, in which case releasing the encoder or pot returns you to the parameters view. This is a shortcut to what would otherwise be a press to switch view, a turn, and another press to switch the view back.

Sometimes you may see parameters “greyed out”. This indicates that the parameter is currently unused, most likely because of some other parameter selection. For example, the tempo parameter of the Clock algorithm is greyed out if the algorithm’s internal clock is not being used.



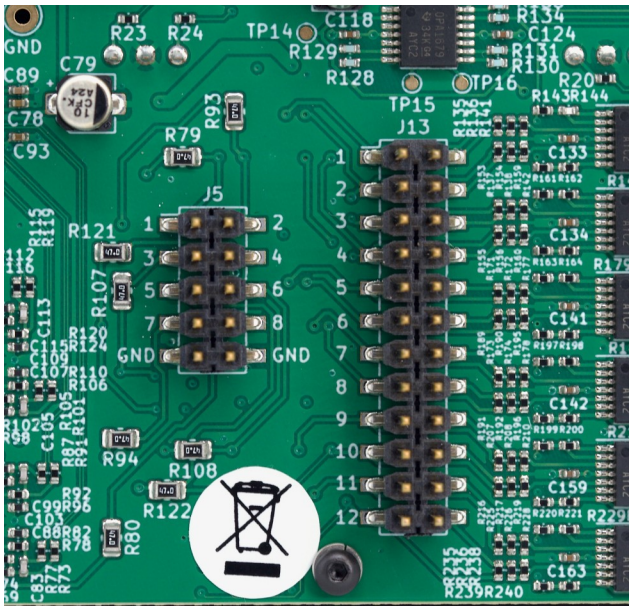
Inputs and outputs

The disting NT's input and output jack sockets are illuminated, lighting red for positive voltage and blue for negative voltage. (Audio-rate signals appear purple, since you see a rapid alternation of positive and negative.)

The sockets are 3.5mm TS jacks. Do not use TRS cables.

The inputs and outputs are all DC coupled.

All of the inputs and outputs are also available on headers on the rear of the module. The outputs are on J5 and the inputs on J13:



Inputs

The twelve sockets on the left of the module (numbered in black on white) are the inputs.

The inputs can operate at either modular or line level, selected by jumpers. See the image above. The left pins on the J13 header correspond to the front panel sockets, and operate at modular level. The right pins are line level. If you fit a jumper across a pair of pins, this shorts the input socket to the line level input i.e. it makes the front panel socket a line level input.

Mode	**Maximum voltage before clipping**	**Input impedance**
Modular	±12V	101kΩ

Line	±3.5V (+10dBu)	29kΩ

Outputs

The six sockets on the right of the module (numbered in white on black) are the outputs.

The outputs have a maximum range of ±11V.

There are an additional two outputs available via the J5 header on the rear of the module. See the image above. The pins of the header are numbered according to the output numbers; pins 1-6 correspond directly to the front panel output sockets, while pins 7-8 are the two additional outputs. Ground ('GND') connections are also exposed on this header.

One possibility is to connect the additional two outputs to two sockets on the included breakout module - or even to connect them to a single socket, as a stereo TRS output.

USB

The module has a type C USB socket to the left of the display. The disting NT is a High Speed USB 2.0 device. It is not a USB host.

The module does not draw power from the USB connection.

If you are planning on using the USB socket more than occasionally, we recommend buying a cable with a right-angled plug, to keep the cable out of the way of the display.

The disting NT is a MIDI Class, Audio 2.0 Class, and Video Class compliant device. As such it needs no drivers when used with macOS, iOS, and Linux. An audio driver for Windows is available via our website[18].

The USB socket can also be used to mount the module's MicroSD card as a removable drive on your computer – see below.

MIDI connections

The disting NT can send and receive MIDI via a breakout header, via the Select Bus, and via USB.

MIDI breakout

The disting NT's MIDI breakout header is a four pin connection, exactly the same as on all of our other modules with MIDI – at the time of writing, these are the disting mk4 and EX, the FH-2, the ES-9, and the CVM-8. The header can be directly connected to DIN or TRS sockets – for example, our own MIDI breakout modules, or the sockets built into certain Eurorack cases.

18 https://expert-sleepers.co.uk/downloads.html

The wiring is as follows:

Breakout pin	MIDI DIN pin
1	OUT pin 4
2	OUT pin 5
3	IN pin 5
4	IN pin 4

Pin numbers refer to the 5-pin DIN socket as in the MIDI specification[19].

As well as connecting the module to a breakout, you can also use the header to connect directly to another of our modules. In this case, connect pin 1 on one module to pin 4 on the other, pin 2 to pin 3, and so on, so that the outputs connect to inputs and inputs connect to outputs.

Select Bus

The Select Bus is a means of inter-module communication currently supported by a handful of modules from various manufacturers, including the Malekko Varigate 8+, Macro Machines Storage Strip and the Make Noise Tempi & René. It is essentially a MIDI connection piggybacked on what was originally conceived as the CV bus, which is one of the lines on the power bus that connects all modules in the system.

The ‘Select Bus protocol’ specifies certain messages to be used for preset saving and recall, which are combinations of MIDI program change and CC messages. The disting NT currently does not implement this protocol at all – it simply uses the Select Bus as a MIDI connection. It is particularly convenient since no additional cables are required – the modules are connected simply by virtue of being on the same power bus.

USB MIDI

The disting NT is a MIDI Class compliant device, as noted above.

MicroSD card

The card slot is just to the left of the left pot. Insert cards with the exposed contacts facing right. It is safe to insert a card after the module is powered up.

Do not remove the card while the module is writing to it, or data corruption could result.

If you do remove the card and re-insert it while the module is powered on, you will need to manually re-mount it via the menu. See the Misc menu, below.

19 https://midi.org/5-pin-din-electrical-specs

Card format

The disting NT requires the card to be formatted as FAT32 or exFAT. If you need to format a card, we recommend that you use the formatting tool provided by the SD Association, which you can download here[20].

Card content

Nothing on the card is actually required for the module to operate; you can insert a completely blank card (or not insert a card at all) and the module will function.

Note though that, in the current firmware, a card is required for storing presets, so you will probably want to keep a card installed at all times.

The card provided with the module is preloaded with some content created by ourselves and our partners, including

- Spitfire Audio[21] - multisamples, including the iconic Soft Piano.
- Goldbaby[22] - samples, mostly drums.
- Adventure Kid[23] - wavetables.

Card layout

Data on the card is organised into a number of top level folders to keep things tidy. While some functions (e.g. preset loading) allow you to select files from anywhere on the card, many of the core algorithm functions require data to be in a specific place – e.g. the algorithms that play samples require all the samples to be in the ‘samples’ folder.

The standard folders, as found on the card supplied with the module, are as follows:

FMSYX	FM banks (SysEx format, usually .syx)
kbm	Scala keyboard map files (.kbm)
MIDI	Subfolders, each containing MIDI files (.mid)
MTS	MIDI Tuning Standard SysEx dumps (.syx)
presets	The default folder into which presets are saved. Can be changed in the Settings menu.
programs	Subfolders contain programs for specific algorithms.
programs/lua	Lua scripts for the ‘Lua Script’ algorithm (.lua)

20 https://www.sdcard.org/downloads/formatter/

21 https://www.spitfireaudio.com

22 https://www.goldbaby.co.nz

23 https://www.adventurekid.se

programs/plug-ins	C++ plug-ins (.o)
programs/three_pot	Programs for the ‘Three Pot’ algorithm (.3pot)
recordings	The default folder for audio recordings made by the Audio Recorder algorithm. Can be changed in the Settings menu.
samples	Subfolders, each containing audio files (.wav)
scl	Scala scale files (.scl)
ui_scripts	Scripts for custom UIs (.lua)
wavetables	Wavetables, either as single file wavetables, or subfolders of single cycle waves (.wav, either way)

FM banks

The ‘FMSYX’ folder on the card contains Yamaha DX7 format voice banks in SysEx format. Such files usually have the extension ‘.syx’ and will be exactly 4104 bytes in size. Literally hundreds, if not thousands, of voice banks can readily be found online, or you can make your own with various software tools designed as DX7 patch editors.

In the current firmware, the FM banks are primarily used by the Poly FM algorithm, and up to 100 banks can be used at once.

Scala files

Many algorithms on the disting NT support microtuning via the common Scala[24] file formats. Scale files (.scl format) go in the ‘scl’ folder and keyboard map files (.kbm) format go in the ‘kbm’ folder.

Keyboard map files are not required – the algorithms can apply microtuning from a .scl file only – but are supported for more detailed control over tone mapping if desired.

Scala files can be generated by the Scala application, by another tool that writes Scala-format files (for example, Scale Workshop[25]), or even written by hand in a text editor.

In the current firmware, you can use up to 1000 .scl files and 1000 .kbm files.

MTS (MIDI Tuning Standard)

MTS can be used live, over a MIDI connection, or via SysEx dump files. The disting NT supports reading MTS bulk SysEx dumps from the card. Such files will be exactly 408 bytes in size, and should be put in the ‘MTS’ folder.

In the current firmware, you can use up to 1000 MTS SysEx dumps at once.

24 https://huygens-fokker.org/scala/

25 https://sevish.com/scaleworkshop

MIDI song files

Some algorithms, notably the MIDI Player, support MIDI song files. Such files typically have the extension '.mid'.

Song files should be put in subfolders of the top level 'MIDI' folder (i.e. not directly within the 'MIDI' folder itself).

A good technical description of the MIDI file format is here[26]. Most DAWs export MIDI files in some form – for example, Ableton Live exports MIDI clips as single track (format 0) MIDI files, while Logic Pro will export multi-track (format 1) files. And of course, the internet is full of sites where you can download complete songs or looping patterns in MIDI format.

Those so inclined might find it useful to create MIDI files programmatically, which can be easily done, for example, in Python using libraries such as mido[27].

The current firmware supports up to 1000 folders of 1000 files each.

Presets

The disting NT uses JSON[28] format for presets. As such they are human readable, and if necessary can be manipulated using a simple text editor, though this is not something we expect most users to do.

Presets can be loaded from anywhere on the card, so you can if you wish organise them into folders. However they are always saved to a location that is chosen in the Settings, which defaults to a top level folder named 'presets'.

Samples

Algorithms that play samples look for them in the 'samples' folder, which should contain subfolders, each of which contains audio files in standard WAV format. Subfolders may be nested.

Mono or stereo files are supported, at any sample rate. Supported bit depths are 8 bit, 16 bit, 24 bit, and 32 bit float.

The current firmware supports up to 1000 sample folders and a total of 10,000 samples, with a maximum of 1000 files per folder.

Q: Why is there a maximum at all? Can't I just load a file from the card and play it?

A: Scanning the card and pre-preparing lists of samples means that sample selection can be mapped to a CV (or MIDI CC etc.). One of the design goals of the disting series has always been to integrate well into a modular synth, not simply to be physically a Eurorack module.

The card is scanned for samples at power on, and most of the salient data cached back onto the card, so a slow scan of each folder is only done when it's added or changed.

26 http://www.music.mcgill.ca/~ich/classes/mumt306/StandardMIDIfileformat.html

27 https://mido.readthedocs.io/en/stable/

28 https://en.wikipedia.org/wiki/JSON

Avoid turning off the module while the scan is in progress, since it's potentially writing to the MicroSD card.

Sample naming

The sample naming convention is the same as for our disting EX[29] module, so if you have libraries prepared for that module they can be easily shared by the disting NT. Look at the samples on the card provided with the module for some examples. The filenames are examined for various patterns, all starting with an underscore ('_'), to set various properties of the files. These are as follows:

Key	Example	Description
_<note name>	piano_A2.wav piano_C#5.wav	Sets the natural note pitch, that is, the pitch of the audio in the sample as recorded. Use naturals and sharps ('#') only, not flats.
_SW<note number>	piano_A2_SW40.wav	Sets the sample's switch point when used in multisample algorithms. The 'note number' is the MIDI note number, in decimal. The switch point specifies the lowest pitch that will use the file.
_RR<number>	snare_RR1.wav snare_RR2.wav	Links files as round-robin variants of the same sample.
_V<number>	snare_V1.wav snare_V2.wav	Sets up a velocity switched sample. The lowest-numbered _V is used for the lowest velocity.

If round-robins and velocity switches are combined, the velocity switch comes first e.g.

snare_V1_RR1.wav, snare_V1_RR2.wav
snare_V2_RR1.wav, snare_V2_RR2.wav

In other words, each velocity layer can have round-robins.

Automatic 'natural' values

If sample files are not assigned a natural pitch (for example, a folder of drum samples), they are assigned values from 48 upwards, so playing a keyboard chromatically from C3 will play a new sample on each semitone.

Automatic 'switch' calculation

If the switch point for multisample files is not explicitly specified, the disting NT calculates sensible defaults, as follows.

If the gap between neighbouring samples is at most 3 semitones, the switch is set so that the higher sample is pitched down within the gap.

For larger gaps, the lower sample is pitched up over half the remaining range.

29 https://expert-sleepers.co.uk/distingEX.html

For example:

Gap between samples (semitones)	Behaviour
1	Higher sample stretched down over 1
2	Higher sample stretched down over 2
3	Higher sample stretched down over 3
4	Higher sample stretched down over 3; lower sample stretched up over 1
5	Higher sample stretched down over 4; lower sample stretched up over 1
6	Higher sample stretched down over 4; lower sample stretched up over 2
	Etc.

Converting from other sample library formats

There are tools in our Github repository[30] for converting the output of Logic Pro's autosampler and of DiscoDSP's Bliss to a format readable by the disting EX (and by extension the disting NT).

We also have a tool for converting SoundFont®s which you can find here[31].

Our format is supported by the very capable ConvertWithMoss[32], which supports many other formats.

Creating your own samples

Creating new sample sets is simply a matter of collecting WAV files in a folder on the MicroSD card and naming them appropriately, if required (see above). Non-pitched samples e.g. drums need no special naming.

There are tools available to automate the sampling of existing instruments, for example DiscoDSP's Bliss as already mentioned.

The disting NT's own Auto-sampler algorithm will automatically sample any instrument it can trigger via CV/gate or MIDI. The files it records to the MicroSD card are named correctly for use by the multisample playback algorithms.

Loop markers in WAV files

The disting NT supports reading loop information embedded in the WAV file. If this information is not present, any playback that loops the sample simply loops the whole file.

The disting NT looks for 'cue ' chunks and 'smpl' chunks. Which of these your files contain will

30 https://github.com/expertsleepersltd/distingEX_tools

31 https://github.com/expertsleepersltd/sf2_to_dex

32 https://mossgrabers.de/Software/ConvertWithMoss/ConvertWithMoss.html

depend on the authoring software.

When interpreting a 'cue ' chunk, loops are inferred either from markers or regions, as follows:

1 marker point in file	Marker is assumed to be loop start; loop is from the marker to the end of the sample.
2 marker points in file	Markers are used as loop start and end.
3 or more marker points in file	First marker is ignored (assumed to be playback start point); second and third markers used as loop points. Remaining markers ignored.
1 or more regions in file	First region is used as the loop; other regions and markers ignored.

When interpreting a 'smpl' chunk, the first loop in the chunk is used, and any others are ignored.

There are some example multisamples with loop points in our GitHub here[33].

Round robin mode

Algorithms that have a 'Round robin mode' parameter allow you to choose how round-robin variants of the same sample are chosen. The options are as follows:

Sequential	The round-robins are played in numerical order.
Random	The round-robins are played in a random order.
Random2	The round-robins are played in a random order, except that the same round-robin cannot be played twice in a row.
Random3	The round-robins are played in a random permutation, then another random permutation, and so on.

Wavetables

A number of algorithms (e.g. Dream Machine and Poly Wavetable) include wavetable-based oscillators. These look for wavetables inside the ‘wavetables’ folder.

Within the 'wavetables' folder, wavetables can take one of two forms: a single WAV file containing all the waveforms concatenated, or a folder of WAV files, one per waveform. All of the examples included on the card use the latter format.

Waveform WAVs can use any supported bit depth. The sample rate is unimportant, since the file is assumed to contain exactly one cycle and so can be pitched arbitrarily.

When using a folder of single waveform WAV files, each file should be the same length. For example, in the ‘AKWF_0001’ wavetable on the MicroSD card each WAV file is 600 samples long.

33 https://github.com/expertsleepersltd/distingNT/tree/main/samples

When using a single concatenated WAV file, the disting needs to know how many frames in the file make up one waveform. It does this as follows:

- If the filename ends in a dash followed by a number, that number is taken to be the number of frames in one waveform. For example, if the filename is “awesome-256.wav”, then the disting will interpret the file as having 256 frames per waveform.
- If the number of frames in the file is a multiple of 2048, then the disting assumes 2048 frames per waveform. This is a convenience to support the popular ‘Serum’ format of wavetable, since those always have 2048 frames per waveform. This means you can simply download one of thousands of Serum wavetables from the internet and drop them straight onto the disting NT’s MicroSD card.
- If neither of the above apply, the disting NT assumes 600 frames per waveform.

If you have a wavetable that you’d like to use but you’re not sure of the correct number of frames per waveform, open the file in a sample editor, or use the waveform view in your DAW. It should be fairly clear where the waveforms lie just from looking at it. Measure the number of frames in one waveform, or count the number of waves in the file and divide that into the total length of the file.

The current firmware supports up to 1000 wavetables on the MicroSD card. The maximum size of each individual wavetable depends on the specific algorithm using it.

Signal flow

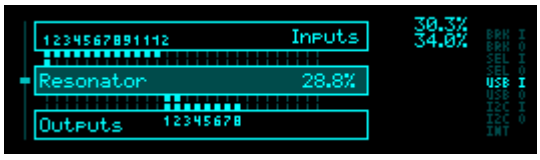
The disting NT implements a very straightforward scheme for passing signals around the system.

All algorithms process data from, and to, a selection of ‘busses’[34], of which there are 64. The algorithms get to work on this data one at a time, in sequence, from top to bottom as seen in the algorithm overview.

At the top of the chain, the first 12 busses are driven from the module’s 12 inputs. The remaining 52 busses are empty/silent.

The next 8 busses feed the module’s 8 outputs, at the end of the chain. The remaining 44 busses are ‘aux’ busses, provided for more routing flexibility between algorithms. Note that only the first 8 aux busses are shown in the display; the full 44 can be visualised in the preset editor tool (see below).

In the algorithm overview, each algorithm’s block has a graphical representation of which busses it is using for input and output:



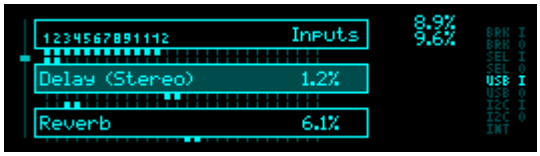
For example, in the image above, the Resonator is taking input from the “Input 1” bus and producing output on the “Output 1” & “Output 2” busses.

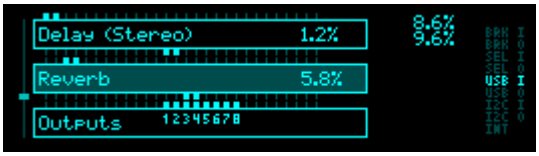
34 We’re using the term ‘bus’ as commonly used when describing audio mixers: see https://en.wikipedia.org/wiki/Audio_bus

Algorithms can be independent

The module is not restricted to running a single signal chain, from inputs to outputs. Implicit in the above bus scheme is the fact that algorithms can work on any bus, no matter where they are in the stack.

To take a simple example, the first algorithm could take inputs from physical inputs 1-2 and output to physical outputs 1-2, while the second algorithm could take inputs from physical inputs 3-4 and output to physical outputs 3-4 – thus operating as two completely independent processes; two separate virtual modules, if you like.





Output mode

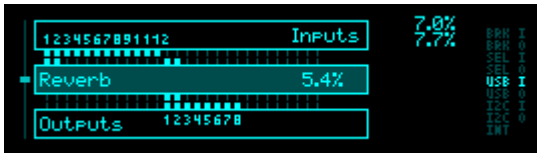
Many algorithms have a parameter (or parameters) named ‘Output mode’, which can either be ‘Add’ or ‘Replace’. This refers to whether the algorithm output in question will be summed onto the output bus, or will replace the bus contents entirely. Both are useful in different situations.

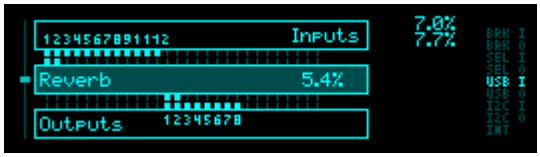
For example, algorithms which are instruments will typically add their outputs to the bus, so if you have multiple instruments feeding the same bus, their outputs will all be audible together.

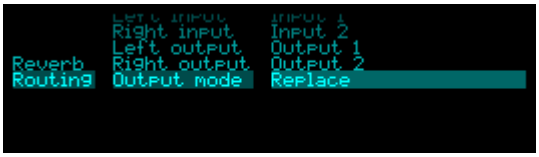
On the other hand, an EQ will typically want to replace the signal on the bus with the EQ’d version.

For other algorithms, the two modes are both useful in different usages. For example, a delay might want to add its output to a master mix bus when used as a send effect, but might want to replace the bus signal when used as an insert (then perhaps using its own wet/dry mix to control the signal balance).

In the algorithm overview, outputs which are using ‘Add’ mode will also show as active inputs to the block:







Algorithm specifications

Algorithms have the concept of ‘specifications’. These are in general either

- Things that affect the resource (chiefly memory) usage of the algorithm, for example, the maximum delay time of an echo effect.
- Things that affect the number of parameters an algorithm has, for example, the number of

audio channels of a mixer.

Some algorithms have no specifications; currently, the highest number of specifications that any algorithm has is three.

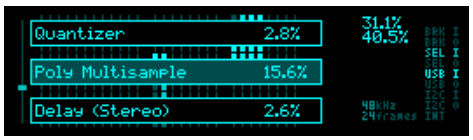
Specifications are set when initially adding an algorithm. If necessary, they can be changed later via the ‘respecify’ menu.

Changing specifications interrupts audio, as it’s essentially causing the entire preset to be rebuilt.

Sample rate and buffer size

By default, the module runs at a sample rate of 48kHz, with an internal buffer size of 24 frames. This means that all signal processing into, within, and out of the module happens at 48kHz, and that “control rate” processing (which includes parameter mapping, and MIDI handling) happens every 24 frames i.e. at 2kHz.

The current sample rate and buffer size are shown in the bottom right of the algorithm overview:



You can change the sample rate and buffer size (from a fixed selection) via the Settings menu (below). In general a higher sample rate or smaller buffer size increases CPU load.

When using USB audio, you can choose (via the Settings) to allow the USB host to control the sample rate (for example, via your DAW’s audio preferences).

CPU usage/overload

The CPU usage of each algorithm is shown in its box in the algorithm overview. The total CPU usage for all algorithms is shown top right. You should aim to keep this below about 90% for reliable operation.

Below this figure is another percentage, which is an estimate of the CPU usage of the entire module, which takes into account MicroSD card access and other background processes. In some cases, you may find this goes significantly higher than the algorithm CPU usage.

If you add too many complex algorithms, or if you adjust algorithm parameters in certain ways, you may end up in a situation where the module’s CPU can no longer keep up. This will likely result in crackly audio and the UI may run slowly or even freeze.

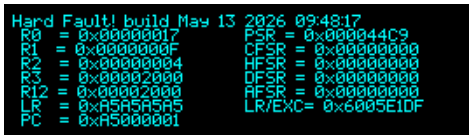
If it detects this situation, the module will suspend all algorithm processing and mute the outputs. The algorithm overview will show the message “CPU OVERLOAD !MUTED!”.

The module will stay muted until you do something which is likely to modify the CPU usage, which includes:

- Removing an algorithm.
- Respecifying an algorithm.
- Bypassing an algorithm.
- Loading a preset or simply doing ‘New preset’.

Fault handler

In the unlikely event that the module crashes, it will attempt to display a fault handler screen:

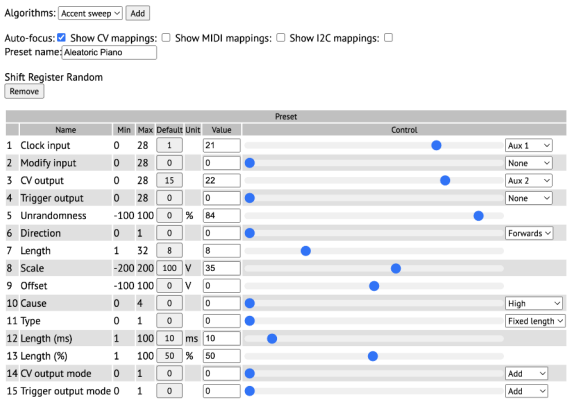


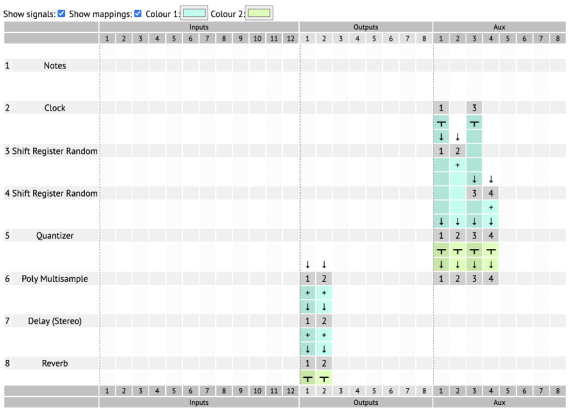
If you see such a screen, please take a photo and include it in a bug report. It contains enough information for us to determine the cause of the crash, and so will usually enable us to fix it.

When in this state, pressing both encoders together will reboot the module.

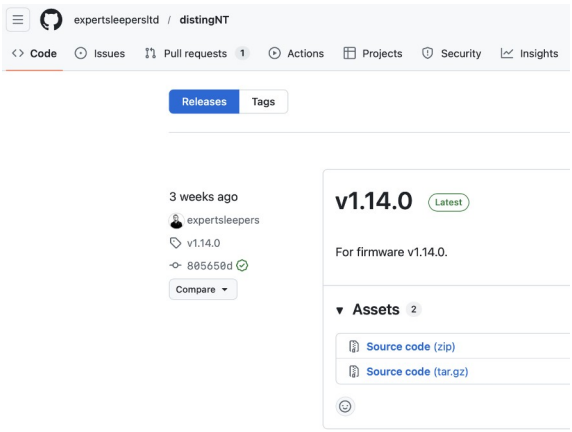
Preset editor/routing analyser

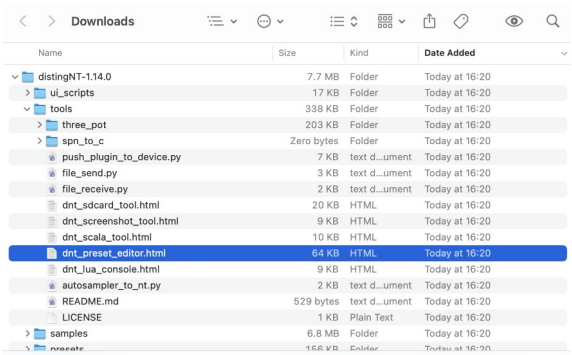
There is a web-based tool in our GitHub repository (here[35]) for editing presets. This is very much a work-in-progress, but useful nonetheless. As well as allowing you to view/edit algorithm parameters, it will show you an analysis of the routing within the preset.





You can install the tool by visiting the releases page of the GitHub repository here[36]. Find the latest release (or the one that corresponds to the firmware version you're using) and click on the zip file to download it. In the zip you'll find a 'tools' folder, and in there 'dnt_preset_editor.html'. Drag this into Google Chrome (it has to be Chrome as most browsers don't support the required Web MIDI API).





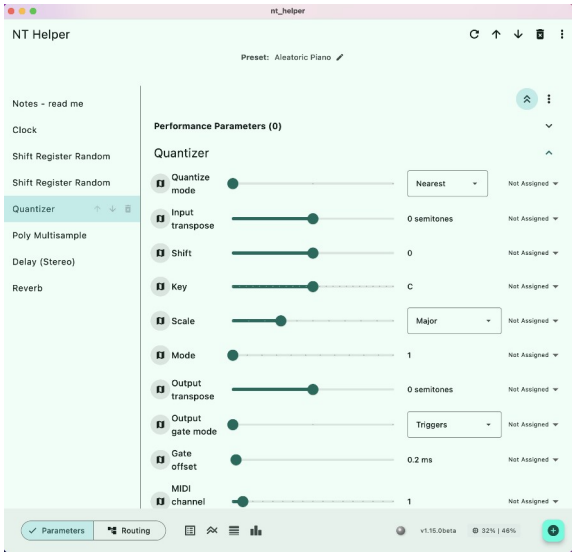
35 https://github.com/expertsleepersltd/distingNT

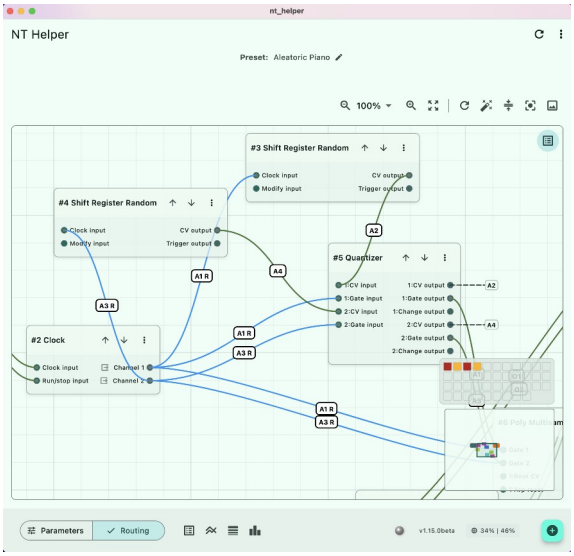
36 https://github.com/expertsleepersltd/distingNT/releases

NT Helper

In addition to the tool above, there is also a third party tool called “NT Helper”, which was written by a user of the module. It runs on Android, Windows, Linux, macOS, and iOS, and can be found here[37].

Please note that this tool is supported by its author, not by Expert Sleepers. You can get support from the author via the Expert Sleepers Discord server[38], in the #nt-helper-app channel.





37 https://nosuch.dev/nt-helper/

38 https://modwiggler.com/forum/viewtopic.php?t=287914

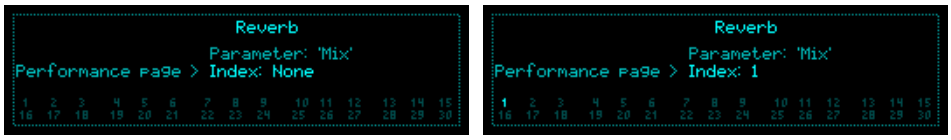
Performance page

The ‘performance page’ is designed to allow quick access to key parameters within a preset from a single location – allowing you to tweak and perform with your preset without navigating around between algorithms.

The performance page also includes knob recorders for instant gestural looping.

Adding parameters to the performance page

To make a parameter appear in the performance page, assign it an index via the Mappings menu (below).



The numbers across the bottom of the screen show the possible indices, with the ones in use highlighted.

You can also do the same thing via the Performance page/Edit menu (below):

As soon as any parameter in the preset has a performance page index, extra lines will appear in the overview screen at the very top and bottom:

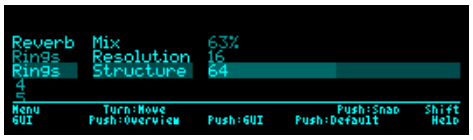
These two are the same, and are duplicated simply to increase the ease of getting to one or the other quickly.

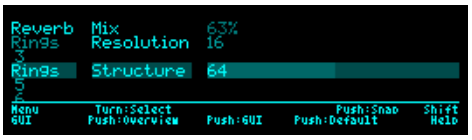
The interface for navigating into and within a performance page is very close to that of an algorithm – e.g. press the left encoder to enter the page.

The left column shows the algorithm name; the middle column shows the parameter name; and the right column shows the parameter value. Scrolling between parameters and editing the parameter value all use the same controls as the usual algorithm parameter interface.

Reordering performance page items

Items can be moved around in the performance page either via the menu (below) or by holding down button 4 (‘Shift’) and turning the left encoder. (This is the same method that is used to reorder algorithms on the overview screen.)





Custom performance page UI

Pressing button 2 (the same button that switches an algorithm’s view to its custom UI) switches to a view where the performance page parameters are arranged in pages of 3, with each assigned to its own knob:

In this view, the three pots control the three parameters on the page, and the left encoder switches between pages.

Because there is limited space, the fields showing the algorithm and parameter names may be truncated. Remember that you can rename algorithms via the Algorithms menu (below). You can also set completely custom text to appear instead of the algorithm and parameter names.

Knob recorders

In the custom view just described, with the parameters in groups of three, each pot also has a knob recorder function. This is very much inspired by the simple yet effective knob recorder on the disting mk4.

To use the knob recorders, you must have an instance of the ‘Knob Recorders’ algorithm (below) in your preset. This allows you to manage the memory allocated to each knob, and to set up clocks to which to synchronise the recordings.

Press a pot and hold it in while turning to activate the knob recorder function. The display will show ‘Record’ and an incrementing timer below the value:

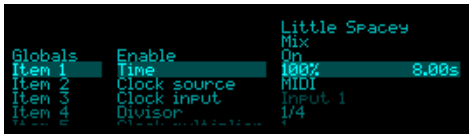
When you release the pot, it will immediately start playing back the knob movement you recorded, in a continuous loop (the display shows ‘Play’):

To stop playback and regain manual control, simply turn the pot.

Note that you can switch to another page of the performance page, or even leave the performance page altogether, and any playing knob recorders will continue to play.

Knob recorder synchronisation

Using the Knob Recorders algorithm, each item’s recorder can be synchronised to a clock signal or to MIDI clock.



When you push the knob to begin recording, or release the knob to end recording, the recorder will wait until the next clock pulse before taking the action.

Menu reference

The pages that follow describe the disting NT’s menus.

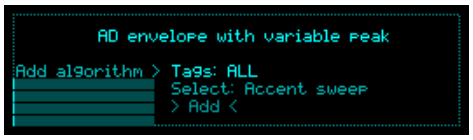
Algorithms menu

The Algorithms menu is where you interact with the list of algorithms that makes up a preset – adding/removing algorithms, reordering them etc.

In general the operations here apply to the ‘current’ algorithm, that is to say, the one highlighted in the algorithm overview.

Add algorithm

Allows you to select an algorithm, which will be added to the preset.

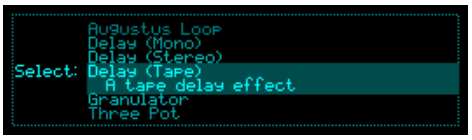


The “Tags” submenu allows you to filter the available algorithms, to show only certain categories e.g. instruments, effects, or utilities.

You can turn the right encoder to select the algorithm directly, and press the right encoder to add it.

Adding an algorithm with the right encoder, or with the left encoder on the ‘> Add <’ menu, adds the algorithm to the end of the preset. Pressing the centre pot instead inserts the algorithm in front of the current algorithm.

You can also press the left encoder to go down to a more detailed view, which shows a description of each algorithm as well as its name.



If the algorithm has specifications, you can adjust these here before adding it.

To the left of the display you’ll see four memory usage gauges, showing the impact that the new algorithm will have on the four memory domains. The memory usage of the algorithm may, or may not, be affected by its specifications. If the algorithm exceeds the available memory, the message ‘INSUFFICIENT MEMORY’ will be shown, and you won’t be able to add it.

If you hold button 4 (‘Shift’), you’ll see that ‘Add’ changes to ‘Add bypassed’. This adds the new algorithm with its Bypass parameter already enabled.

You can add a total of 40 algorithms to a preset. This limit is completely arbitrary. If you feel it

should be higher, please let us know – and we want to see that massive preset you want to make!

Respecify algorithm

Allows you to change an algorithm's specifications, while maintaining its parameters and position in the list.

As in the 'Add algorithm' menu, a memory gauge is shown. You cannot respecify an algorithm if doing so will cause its memory usage to exceed that available.

Respecifying an algorithm may add or remove parameters (for example, respecifying the mixer will add or remove parameters to reflect the change in the number of mixer channels). Any added parameters will assume their default values.

Remove algorithm

Removes the current algorithm from the list.

Move algorithm up

Moves the current algorithm up one position in the list.

Move algorithm down

Moves the current algorithm down one position in the list.

Customise name

Allows you to change the algorithm's name. By default, each algorithm takes the name of the algorithm of which it is an instance e.g. if you add a 'Reverb' the algorithm in the list will be called 'Reverb', but you can change this to anything you like. This can be especially useful if you have more than one instance of the same algorithm, so you can quickly tell them apart.

Duplicate algorithm

Duplicates the current algorithm, if available memory permits. The duplicate algorithm is positioned immediately after the original in the list.

Presets menu

The Presets menu is where you load and save presets, and perform related functions.

A preset contains the full state of the module – the algorithms, their parameters, and any mappings. Presets are stored on the MicroSD card (in this firmware version, at least – future versions may offer the possibility of storing presets in flash memory).

Load preset

Lets you choose a preset file from the MicroSD card to load.

Name preset

Lets you set the preset name. This will be used as the filename when the preset is saved. See the notes on editing text, above.

Generate random preset name

Sets the preset name to a randomly generated phrase.

Save preset

Saves the current preset in memory to the MicroSD card, using the name set in ‘Name preset’ and to the folder chosen in ‘Set save location’.

New preset

Removes all algorithms and sets the preset name back to the default ‘Init’.

Split/combine

This submenu contains a number of functions to allow you to combine presets, or to split parts of presets out for reuse elsewhere.

Append preset

Like “Load preset”, but the chosen preset file is added to the current state, instead of replacing it – that is, the algorithms in the preset file are appended to the list of running algorithms (if possible, that is, if memory permits).

Insert preset

Like “Append preset”, but the file is inserted in the existing preset before the current algorithm.

Prepend preset

Like “Append preset”, but the file is inserted at the top of the existing preset instead of at the bottom.

Save single algorithm preset

Saves the current algorithm only to a preset file. Useful in combination with ‘Append preset’ to copy a single algorithm from one preset to another, or simply to save a library of presets for a particular algorithm, independently of whatever else is going on in the preset.

The preset file is named using the algorithm’s default name and its custom name as set via the ‘Customise name’ menu. For example, if you have an instance of the ‘Reverb’ algorithm and you’ve

customised its name to ‘Hall’, the preset will be saved as ‘Reverb - Hall.json’.

Save range

Similar to ‘Save single algorithm preset’, but saves a range of algorithms from the current preset to a new file.

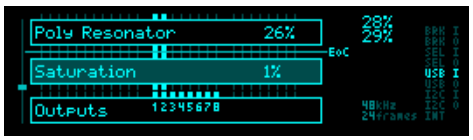
The saved file is named using the current preset name, plus numeric suffices to indicate the range of saved algorithms. For example, if your preset is named ‘elegiac’ and you save algorithms 3 thru 5, the filename will be ‘elegiac-3-5.json’.

End of chain

The disting NT supports the concept of an “end of chain” (hereafter EoC) preset. This is a notional split in the algorithm list, where everything after a certain point stays the same when presets are loaded. This is designed to be particularly useful for setting up final bus compression, EQ etc. so you could for example arrive at your gig, tweak the EQ for the venue, and then have that stay the same as you load the various presets used during your set.

Note that there’s nothing special or different about an EoC preset file – it’s just a regular preset file containing a list of algorithms and their parameters.

The EoC marker is shown in the overview page as a line with “EoC” at the right side:



The various EoC submenu functions are as follows.

Load EoC

Lets you choose a preset file to load as an EoC preset. If there is no existing EoC preset, the chosen file is appended to the preset, and the EoC marker is set. If there is an existing EoC preset, everything below the EoC marker will be removed and replaced by the loaded file.

Name EoC

As ‘Name preset’, but used when saving the EoC preset.

Generate random name

Sets the EoC preset name to a randomly generated phrase.

Save EoC

Saves a preset file using the name set under ‘Name EoC’, containing everything below the EoC marker.

Remove EoC

Removes all algorithms below the EoC marker, and resets the marker so there is no EoC section.

Adjust

Lets you move the EoC marker up and down in the algorithm list. The bottom of the screen shows the name of the algorithm that the marker will be above.

Set save location

Lets you choose the folder on the MicroSD card where presets will be saved. Note that presets can be loaded from anywhere – this only affects the save operation. You're free to take the MicroSD card over to a computer to rearrange, rename, or archive the saved preset files.

Load disting EX preset

Lets you choose a disting EX preset file and attempt to load it as a disting NT preset. At the time of writing, this will only succeed for disting EX 'Poly Wavetable' presets. If you have disting EX presets for other algorithms that you particularly would like to load into the disting NT, please get in touch!

Author name

Lets you set a name that will be saved in presets as the "author".

Mappings menu

Video[39]

The Mappings menu is where you set up remote control of algorithm parameters by CVs, MIDI, and I2C. Any parameter can be controlled by any combination of these. Also, any control source can control as many algorithm parameters as you like.

The mappings shown refer to the current algorithm, highlighted in the algorithm overview.

When you enter the menu, the various mapping submenus will reflect the current parameter that was being edited in the single algorithm view. Likewise, if you select a different parameter while setting up the mappings, that parameter will be the current one when you leave the menu.

Mappings are part of the preset, and are saved and loaded with it.

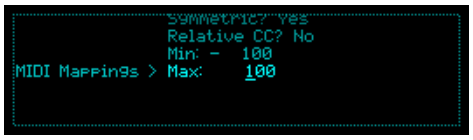
39 https://www.youtube.com/watch?v=utpF6fIP_zM

Editing mappings values

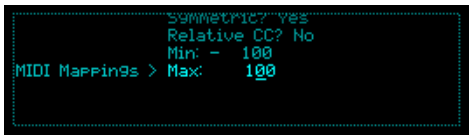
The mappings UI includes a unique method of adjusting values which would otherwise be very fiddly to adjust using the standard knob/encoder UI used elsewhere in the module.

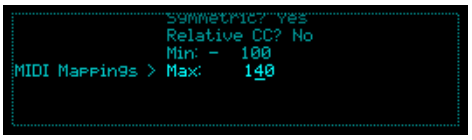
The values in question are the Delta for CV mappings, and the Min & Max for MIDI and I2C mappings. (See below for details of what these values actually mean.)

Taking the MIDI mappings Max as an example:



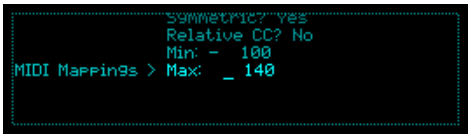
You will see that there is a cursor under the value, which is used to select a digit to adjust:

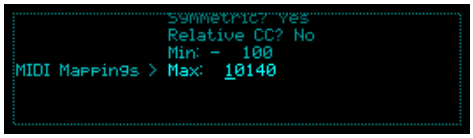




Move the cursor with the middle pot; adjust the digit with the right pot.

You can position the cursor on an empty digit and turn the right pot, to jump directly to a large value:





You can also use the right encoder as normal to adjust the value.

CV Mappings

This is where you set up a mapping from a CV source. This will commonly be an external CV applied to one of the module's inputs, but can also be any signal on any bus within the system – so you can for example run an LFO algorithm and map the CV that it generates to control a parameter on another algorithm.

Unless the mapping is defined as a 'Gate', the mapping *adds* to the base parameter value set in the preset – it does not *replace* it. To put it another way, a CV of 0V, or a disconnected cable, will always give you the parameter value as if it wasn't mapped.

There are three choices for where the CVs to be used for mapping come from.

The first, and default, is 'Own inputs'. This means that the algorithm will look at state of the busses at its own inputs for the mapping CVs.

The second is 'Module inputs' – the mapping CVs are taken from the physical inputs on the module.

The third option is that the mapping CVs are taken directly from the outputs of another algorithm in

the preset.

There is some redundancy here, inasmuch as if you choose the source as the algorithm immediately before the mapped algorithm in the preset, the effect is the same as if the source was ‘Own inputs’. However, these will differ in terms of what happens if the preset is reordered or algorithms are deleted.

The items that can be set up per mapping are as follows:

Parameter	Chooses the parameter to map.
Source	Chooses the CV source – ‘Own inputs’, ‘Module inputs’, or the outputs of one of the algorithms in the preset.
Bus	Chooses the CV source bus.
Unipolar?	If set, the CV is clamped to 0V (i.e. negative voltages are treated as 0V).
Gate?	If set, the CV can only toggle the parameter between its minimum and maximum values.
Volts, Delta	These two together determine the relationship between CV voltage and parameter value. The value added to the parameter is equal to (CV voltage) x (Delta) ÷ (Volts) To put it another way, a CV of (Volts) is required to change the parameter value by (Delta).

MIDI Mappings

This is where you set up mappings from MIDI CCs (Continuous Controllers), channel pressure (aka aftertouch), pitch bend, and notes.

Unlike CV mappings, MIDI mappings directly set the algorithm parameters – they do not add to them.

While in the MIDI Mappings menu, pressing the right encoder enters “Learn” mode. While this is active, the module listens for MIDI CCs and notes, and the first one received will be set as the source for the current mapping. Press the right encoder again to leave Learn mode without altering the mapping. Note that there is a setting for whether setting up a mapping via Learn also disables any other mappings that were already using the CC.

14 bit CC pairs are supported. This is a convention used by some controllers where a high resolution value is split across two CCs: a ‘high’ or ‘most significant’ byte using a CC in the range 0-31, and a ‘low’ or ‘least significant’ byte using the CC 32 higher (i.e. in the range 32-63). Unfortunately nobody can agree which should be transmitted first, so we support both. When using 14 bit CCs, specify the CC in the range 0-31 for the CC number for the mapping.

The items that can be set up per mapping are as follows:

Parameter	Chooses the parameter to map.
Channel	Sets the MIDI channel for the mapping controller.
Type	One of • ‘CC’ • ‘Note – momentary’ (note on sets the Max value; note off sets the Min value) • ‘Note – toggle’ (note on toggles between Min and Max; note off is ignored) • ‘14 bit CC - low first’ (a 14 bit CC where the low byte is sent first) • ‘14 bit CC - high first’ (a 14 bit CC where the high byte is sent first) • ‘Pitch bend’ • ‘Channel pressure’ (aka aftertouch)
CC/note	Sets the MIDI CC or note number (0-127).
Enabled?	Sets whether the mapping is enabled or disabled.
Symmetric?	Allows you to specify a mapping as symmetric. For a normal CC, the range 0-127 is mapped across the range set by the min & max settings below. For a symmetric CC, the value 64 is mapped to the midpoint of the range, and values above and below that are scaled by half the total range. A symmetric mapping is appropriate for a parameter which has a bipolar range around zero (for example, a pan position), where you want to be sure that a MIDI value of 64 gives you exactly zero in the middle.
Relative CC?	If no, the CC is a standard 0-127 value that directly sets the parameter. If yes, the CC is an up/down relative movement. Some MIDI controllers support this scheme as an alternative to soft takeover (see below).
Min	Sets the minimum parameter value that the mapping will set.
Max	Sets the maximum parameter value that the mapping will set.
View change?	Sets whether the mapping will change the display view, if the global ‘CC changes view’ setting is enabled.

I2C Mappings

This is where you set up mappings from I2C controllers.

Like MIDI mappings, I2C mappings directly set the algorithm parameters.

While in the I2C Mappings menu, pressing the right encoder enters “Learn” mode. While this is active, the module listens for I2C controllers and the first one received will be set as the controller for the current mapping. Press the right encoder again to leave Learn mode without altering the mapping. Note that there is a setting for whether setting up a mapping via Learn also disables any other mappings that were already using the controller.

The items that can be set up per mapping are as follows:

Parameter	Chooses the parameter to map.
Controller	Sets the I2C controller number.
Enabled?	Sets whether the mapping is enabled or disabled.
Symmetric?	Allows you to specify a mapping as symmetric. For a normal mapping, the range 0-16383 is mapped across the range set by the min & max settings below. For a symmetric mapping, the value 8192 is mapped to the midpoint of the range, and values above and below that are scaled by half the total range. A symmetric mapping is appropriate for a parameter which has a bipolar range around zero (for example, a pan position), where you want to be sure that a controller value of 8192 gives you exactly zero in the middle.
Min	Sets the minimum parameter value that the mapping will set.
Max	Sets the maximum parameter value that the mapping will set.

Performance page

This is where you assign an index within the performance page to a parameter. See the section on the performance page above.

Performance page menu

This menu contains functions relating to the performance page (above).

Edit

Allows you to directly edit the contents of each performance page slot. It is equivalent to doing the same thing in the mappings menu. Which you use is a matter of approach – either “here’s a parameter, let me put that into the performance page”, or “here’s a performance page slot, let me put something in it”.

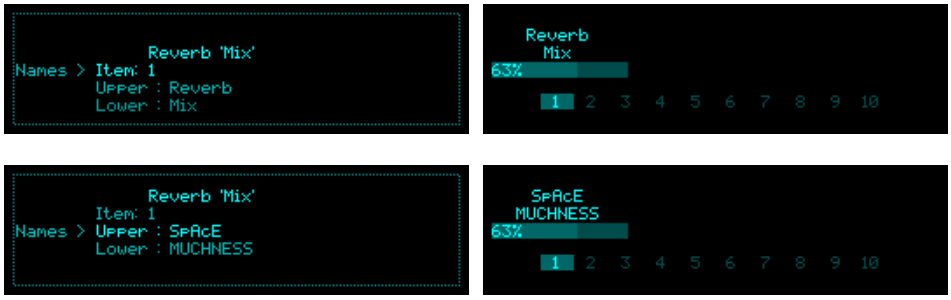
Min/max

Allows you to restrict the range of values that the performance page will set to a parameter:

The menu items here use the custom UI for editing large values, described above.

Names

Allows you to customise the text that is used for a parameter in the performance page.



Move up/down

These items move the current performance page item up or down in the list. You can also do this in a more immediate manner using the Shift button as described above.

Settings menu

The Settings menu lets you change various global options. Settings are stored internally, in flash memory, not on the MicroSD card.

Display

This submenu contains options for the module's screen.

Setting	**Description**
Contrast	Sets the display contrast (brightness).
Master contrast	Sets the display contrast (brightness) in coarse steps.
Flip	Allows you to flip the display and so use the module upside down.
Screensaver (minutes)	Sets the number of minutes after which the screensaver will kick in.
Screensaver style	Sets whether the screensaver is an animated graphic, a blank screen, or actually powers off the screen.
Wake on MIDI/I2C	If set, the display is woken from the screensaver when a mapped MIDI or I2C message is received.
Enter screensaver now	Immediately activates the screensaver.
Support USB Video Class	Sets whether the module advertises USB Video Class support in its USB descriptor. Some USB MIDI hosts are known to fail if the descriptor includes this, so the option is provided to turn it off.

CPU load display	Sets the update rate of the CPU load figures on the overview page. The options are “Responsive” and “Calm”.
Bypass	This submenu allows you to edit the strings that are displayed for the ‘Bypass’ parameter that is shared by every algorithm. The defaults are ‘Off’ and ‘On’, reflecting whether bypass is off or on. Some users have found this confusing, especially in the Performance Page, where seeing ‘On’ can suggest that the algorithm is ‘on’, the opposite of what is actually the case.
Performance page	This submenu contains some options that apply to the Performance Page. ‘Show CPU load’ displays the CPU usage (as usually shown on the overview page). ‘Show Stopwatch’ shows the time on a Stopwatch algorithm (see below) if one is present in the preset. ‘CPU on list view’ sets whether the CPU and stopwatch time are shown in the Performance Page parameter list UI as well as in its custom UI.
Expert settings	This submenu lets you mess with very low level settings on the display driver chip. Most users will not want to change these. Just in case you do, the default values are given in parentheses in each menu name.

UI (User Interface)

This submenu contains options for how you interact with the module.

Setting	**Description**
Show button help	Sets the number of seconds to show the pop-up help. If set to 1, the help is permanently visible. If set to 0, the help is never shown automatically (but can still be shown via button 3).
Button 4 function	Chooses the menu to which button 4 is a shortcut when in the single algorithm view.
Button ordering	Chooses whether buttons 1-4 are numbered from left to right or right to left.
Primary encoder	Chooses which encoder is used to navigate the menus.
Pot threshold	Sets a threshold for movement of the pots.
Add algorithms bypassed	If set, new algorithms are added to the preset with their Bypass parameter enabled.
Knob record min playback	Sets the minimum amount of time that a knob recorder will be playback after recording before it drops out of playback when you move the knob. Provided since most people continue to rotate the knob slightly as they release it. Specified in multiples of 10ms (i.e. the default value of 20 means 200ms).

Engine

Setting	Description
Sample rate	Allows you to choose the module's sample rate (32, 44.1, 48, 88.2, or 96kHz). Does not take effect until you click on '> Apply <'.
Block size	Allows you to choose the module's audio buffer size. Does not take effect until you click on '> Apply <'.
Allow USB rate change	If on, the USB host (e.g. your DAW) can change the module sample rate. If off, USB audio is fixed at the sample rate you choose. This setting takes effect after a reboot (since it changes the USB descriptors).
NTX-8CV	This submenu allows you to configure the disting NT to use any attached NTX-8CV expansion modules. NTX-8CV channels are enabled in groups of 8, and correspond to 8 consecutive busses. "Channels 1-8" correspond to Aux busses 1-8, "Channels 9-16" to Aux busses 9-16, and so on. Note that each NTX-8CV should have the corresponding channel group chosen in its settings. There is no harm in enabling NTX-8CV channels for which you don't actually have an NTX-8CV attached, except that doing so consumes CPU resources. "Also use ES-5" allows the NTX-8CVs to share the connection with an ES-5 module and up to two connected ESX expanders (using slots GT1/2/3 on the ES-5). Activating this halves the bandwidth available to the NTX-8CVs, so it is preferable to leave this off unless you actually need it. NTX-8CV modules which share their connection with an ES-5 should have their own "Also use ES-5" setting enabled as well.

Globals

Setting	Description
Tuning A=	Sets a global tuning standard, the default being A=440Hz. The value set here corresponds to MIDI note 69 (A4).

Startup

Options for things that happen when the module powers up.

Setting	Description
Load last preset	Sets whether the module loads the most recently used (that is, saved or loaded) preset at startup. The "Reset preset" menu also has the effect of having the module forget the most recently used preset so it won't be loaded at startup. Please note that the module does not "auto-save". You always have to save manually, if you want to save any changes to the loaded preset.

Select Bus

Setting	Description
Speed multiplier	An experimental feature that allows you to run the Select Bus at higher than usual speeds. Leave this at ‘1’ (normal speed) unless you know you’re working with other Select Bus devices that also allow you to change the speed.

MIDI

Setting	Description
CC changes view	If set to ‘Parameter’, mapped MIDI CCs that change parameter values will also change the currently visible parameter (if the algorithm involved is the currently visible one). If set to ‘Parameter and algorithm’, the current algorithm will also be changed.
SysEx changes view	If enabled, MIDI SysEx messages that change parameter values will also change the currently visible parameter (if the algorithm involved is the currently visible one).
Soft takeover	Chooses the style of soft takeover applied to CCs. See below.
Send CCs	When to send MIDI CCs when parameters change. The options are 'Off', 'On preset load', 'On parameter change', or 'Both'.
Send CCs if changed by MIDI map	Sets whether CCs are sent when a parameter changes, if that parameter was changed by a MIDI mapping. This can help avoid loops, if for example the MIDI controller sending the CCs also sends received CCs back out.
Learn is exclusive	If enabled, when a MIDI CC mapping is set via 'Learn', any conflicting mapping of the same CC is unset.
SysEx Device ID	The SysEx device ID to respond to.
Program Change channel	The MIDI channel on which to respond to Program Change messages (see below).
Program Change uses Bank	Sets whether Bank Select (CC #0) is also used when responding to Program Change messages.
Program Change shows message	Sets whether a ‘preset loaded’ message is displayed when a preset is loaded by a MIDI Program Change message.
Forwarding	Sets whether incoming MIDI on the various ports (USB, Breakout, Select Bus) is passed on to the other ports. The option for “Breakout to Breakout” means that MIDI arriving on the breakout MIDI in will be passed to the breakout MIDI out.

MIDI CC soft takeover

If enabled, the disting NT supports the familiar 'soft takeover' paradigm for MIDI CCs, to avoid parameter jumps if the controller's physical position doesn't match that of the preset.

Three variants of soft takeover are available:

- **Sudden** – after loading a preset the controlling MIDI CC has to pass through the value that would set the current parameter value before assuming control. Before that happens, the CC has no effect.
- **Gradual** – similar to the way the module knobs work, the CC gradually takes control of the parameter value until the positions match, at which point soft takeover deactivates and the CC controls the parameter directly.
- **Always on** – like ‘Gradual’ except soft takeover does not deactivate; it is always applied for CC changes, just like it is for front panel knob changes.

Note that the disting NT also supports ‘relative’ MIDI CCs, which do away with the need for soft takeover. If a mapping is set to use relative CCs (see above), the soft takeover setting has no effect on it.

MIDI Program Change

If enabled, the module responds to MIDI Program Change messages by loading a preset. The following rules are followed to determine which preset is loaded:

- If a preset filename contains the program number preceded by a hash (‘#’), that preset is loaded. For example, if a Program Change is sent with a program number of 42, that will load a preset with “#42” anywhere in its name.
- Otherwise, the filenames are simply sorted alphabetically and the program number is used to index into the list.

I2C

Setting	Description
Address (as follower)	The module's I2C address.
CC changes view	If set to ‘Parameter’, mapped controllers that change parameter values will also change the currently visible parameter (if the algorithm involved is the currently visible one). If set to ‘Parameter and algorithm’, the current algorithm will also be changed.

Real-time clock

This submenu allows you to set the real-time clock, which is used for timestamping files when writing to the MicroSD card. Set the year, month, day, hour, and minute as desired and then select “> Set <” to actually set the time.

The time will be accurately updated while the module is powered, but will be lost when the module is powered off, since it lacks a battery. However, the MicroSD card is scanned at startup and the clock set to a time after the latest file found, so the times on any newly saved files will at least be later than

those that came before.

Calibration

This menu contains various functions related to the calibration of the module. The module requires accurate calibration so that it can receive and generate accurate voltages and translate them to/from software.

The module is calibrated at the factory, so you shouldn't need to worry about doing it yourself. If you do decide to calibrate the module yourself, you will need an accurate source of a +3V reference voltage. Other modules which are designed to deliver accurate pitch CVs are usually a good choice for this, for example, MIDI/CV converters.

Any changes you make to the calibration are not saved to flash unless you specifically do so with the "Save to calibration to flash" menu.

Zero input offsets

This function is not really calibration at all, but is a quick way to remove any DC offsets on the module inputs. Disconnect any patch cables from the module first.

Calibrate an input

This menu allows you to calibrate a single input. Choose an input, select "> Calibrate <", and follow the displayed instructions.

Calibrate an output

This menu allows you to calibrate a single output. Note that it relies on Input 1 already being well calibrated. Choose an output, select "> Calibrate <", and follow the displayed instructions.

Save to calibration to flash

Saves any changes you've made to the calibration to flash memory.

Reset calibration

Resets the calibration to a vaguely sensible, but uncalibrated, state. Note that this function does not save anything to flash.

Read factory calibration

Loads the calibration as performed by the factory. Note that this function does not save anything to flash. Use "Save to calibration to flash" if you want to save the factory calibration as your new calibration.

View calibration

Allows you to see the various values that have been measured during calibration.

View input voltages

Allows you to see the voltages that the module thinks are on its various inputs. Ideally these would all be very close to zero when no patch cables are connected. The readings here will only be accurate if the module is well calibrated.

Audio recordings folder

This setting allows you to choose the folder for new recordings made by the Audio recorder algorithm.

Factory reset

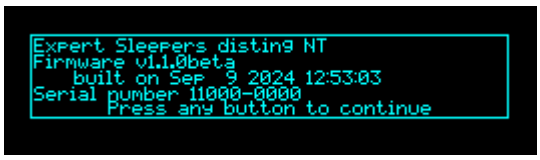
This menu restores all settings to their default values. Note that this does not affect the calibration.

Misc(ellaneous) menu

The ‘Misc’ menu contains a variety of functions that don’t fit anywhere else.

About

Displays the current firmware version information and the module serial number.



MicroSD card

This submenu contains some utilities related to the MicroSD card.

Remount

Lets the module know that you’ve removed the MicroSD card and reinserted it. Unlike a full-size SD card, a MicroSD card has no physical mechanism for letting the module know that the card was removed – as far as the module is concerned, the card simply stopped responding properly. Selecting this menu allows the module to reinitialise communication with the card.

Test speed

Performs a speed test on the card, and displays some statistics about the results.

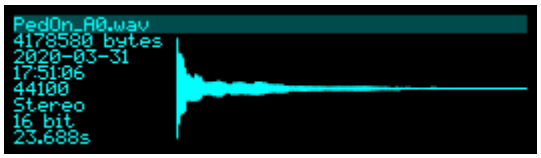
Browse

Allows you to browse the contents of the card, display information about files, and in some cases view the contents of the files.

Files with the extensions “.txt”, “.json”, and “.lua” can be viewed as text. Use the left pot or left

encoder to scroll through the file.

Files with the extension ".wav" can be viewed as a waveform, with the file's sample rate, channel count, bit depth, and length (in seconds) displayed.



Enter USB disk mode

Video[40]

This reboots the module into a mode where the MicroSD card appears as a USB disk. All other functionality is suspended while in this mode.

If you connect the disting NT to a computer while in this mode it will appear as a removable drive, just as if you'd plugged in a USB thumb drive or portable harddrive. You can use this to manage files on the card without having to remove the card from the module and plug it into a MicroSD adaptor on your computer.

Remember to eject the drive on your computer before powering off the module or rebooting into normal mode.

Press both encoders together to reboot the module and resume normal operation.

Reboot

Reboots the module, as if you'd turned the power off and on again.

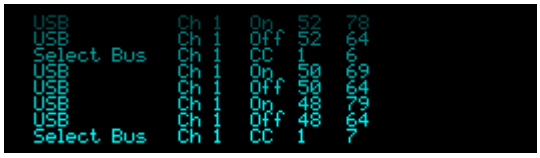
Enter bootloader mode

Puts the module into a mode in which it can communicate with the firmware update tool. Note that there is no way back from this mode other than to physically power cycle it.

MIDI monitor

Runs a MIDI monitor, showing the MIDI messages that the module is receiving as text. Select the "> GO <" item to start the monitor.

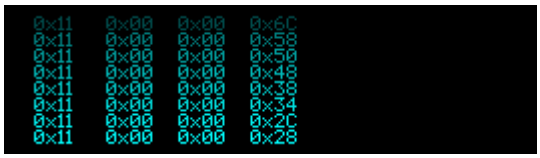
40 https://www.youtube.com/watch?v=nHvTzokMLYA



The ‘Show Clocks’ option selects whether to include MIDI clock messages (F8h) in the display, which tend to flood out any useful information if shown (unless, of course, you’re actually checking to see if you’re receiving clocks or not).

I2C monitor

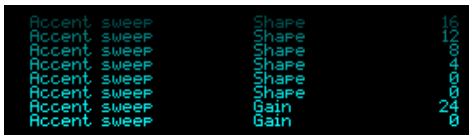
Runs an I2C monitor, showing the I2C messages that the module is receiving as text. Select the “> GO <” item to start the monitor.



The ‘Show Getters’ option selects whether to include I2C messages that are querying values from the NT, as opposed to sending commands.

Mappings monitor

Runs a mappings monitor, showing the parameters that have been changed as a result of MIDI or I2C mappings.



MIDI

This menu contains a couple of MIDI utilities.

- **All notes off** sends a MIDI “all notes off” message (CC 123, value 0) on all 16 MIDI channels, to all destinations (breakout, Select Bus, USB, and all algorithms).
- **All sound off** is similar but sends a MIDI “all sound off” message (CC 120, value 0).

Reset state

This instantly clears all audio delay buffers (echoes, reverbs etc.) in the preset. It could be useful when restarting a track (e.g. in rehearsal, or when doing overdubs), when you want to start from silence, without all the delay tails that you might have ringing on if you stop a track in the middle.

UI Scripts menu

This menu allows you to run a UI script. See below.

Algorithm-specific menu

The last menu, if present, offers functions specific to the current algorithm (the one highlighted in the algorithm overview). For example, the Granulator menu allows you to load and save samples. See each algorithm for details of its menu, if any.

Common algorithm features

A number of the disting NT algorithms share some features, which are described below.

Bypass

All algorithms have a ‘Bypass’ parameter. When enabled, the algorithm’s processing is completely skipped, and it consumes no CPU.

When bypassed, an algorithm appears in the overview ‘grayed out’.

A bypassed algorithm still responds to parameter automation via CV or MIDI/I2C (so you can enable/disable bypass remotely).

Note that this is not the same as what might be termed a “through” bypass. For example if you have a reverb taking input from input 1 and outputting to output 1, if you bypass it, you will have nothing on output 1 – not the dry signal from input 1.

Microtuning (Scala/MTS)

Microtuning is supported by both Scala files and MTS (MIDI Tuning Standard). Please see the section above for how files defining these tunings are arranged on the MicroSD card.

There are a number of parameters which together define how microtuning will be applied in the algorithm. The most important is ‘Microtuning’, which can be ‘None’, ‘Scala’, or ‘MTS’.

Scala

If ‘Microtuning’ is set to ‘Scala’, the tuning is defined by a Scala .scl file and optionally a .kbm file as well.

It is possible to send the files over MIDI SysEx, but note that sending Scala over MIDI is not a standard operation – to our knowledge it is currently only supported by our own tool, which you'll find in our GitHub[41]. Choose ‘From SysEx’ for the ‘Scala .scl’ or ‘Scala .kbm’ parameter to use the file sent in this manner.

It is much more common to use files directly, which in our case means from the MicroSD card. Simply choose the files you want to use via the ‘Scala .scl’ and ‘Scala .kbm’ parameters.

41 https://github.com/expertsleepersltd/distingNT/tree/main/tools

The ‘Scala .kbm’ parameter has another option, which is ‘Automatic’. In this case, a .kbm file is not required, and the algorithm constructs the equivalent information from two further parameters, ‘Auto kbm root’ and ‘Auto kbm Hz’. Refer to the official .kbm format documentation here[42]. The automatic keyboard map generated is equivalent to the following .kbm file:

```
! Template for a keyboard mapping
!
! Size of map. The pattern repeats every so many keys:
<number of pitches in the .scl file>
! First MIDI note number to retune:
0
! Last MIDI note number to retune:
127
! Middle note where the first entry of the mapping is mapped to:
‘Auto kbm root’ value
! Reference note for which frequency is given:
‘Auto kbm root’ value
! Frequency to tune the above note to (floating point e.g. 440.0):
‘Auto kbm Hz’ value
! Scale degree to consider as formal octave (determines difference in pitch
! between adjacent mapping patterns):
<number of pitches in the .scl file>
! Mapping.
! The numbers represent scale degrees mapped to keys. The first entry is for
! the given middle note, the next for subsequent higher keys.
! For an unmapped key, put in an "x". At the end, unmapped keys may be left out.
0
1
2
<etc. up to the number of pitches in the .scl file minus 1>
```

MTS

If ‘Microtuning’ is set to ‘MTS’, the tuning can come either from MIDI SysEx messages, or loaded from the MicroSD card. This is selected by the ‘MTS .syx’ parameters. If this is set to ‘From SysEx’, the algorithm will use the tuning sent to it over MIDI; otherwise, it will load a SysEx dump from the card and use that.

The supported format is the bulk tuning dump as described here[43].

Chord/quantizer scales

The available scales are as follows.

Name	**Notes**	**Example (on C)**
Chromatic	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12	
Major	1, 3, 5, 6, 8, 10, 12	C D E F G A B
Natural Minor	1, 3, 4, 6, 8, 9, 11	C D E ♭ F G A ♭ B ♭
Harmonic Minor	1, 3, 4, 6, 8, 9, 12	C D E ♭ F G A ♭ B

42 https://www.huygens-fokker.org/scala/help.htm#mappings

43 https://midi.org/midi-tuning-updated-specification

Dominant	1, 3, 5, 6, 8, 10, 11	C D E F G A B ♭
Fully Diminished	1, 3, 4, 6, 7, 9, 10, 12	C D E ♭ F F♯ A ♭ A B
Dominant Dim	1, 2, 4, 5, 7, 8, 10, 11	C D ♭ E ♭ E F♯ G A B ♭
Augmented	1, 4, 5, 8, 9, 12	C E ♭ E G A ♭ B
Whole Tone	1, 3, 5, 7, 9, 11	C D E F♯ G♯ A♯

Common polysynth features

A number of the disting NT algorithms are “polysynths”, and share some features, which are described below.

Min/max note

All polysynths have two parameters which define the range of MIDI note numbers that will be recognised - ‘Min note’ and ‘Max note’. These default to 0 and 127 respectively (i.e. the full MIDI range). Notes outside of this range will be ignored. Using these, you can set up keyboard splits.

These parameters also apply when using I2C and CV/gate, not just MIDI.

Velocity/pressure curves

All polysynths have parameters which set a scaling curve for MIDI velocity and pressure. These only apply to MIDI; not to I2C or CV/gate.

Velocity curve	-100	100	0		The scaling curve for MIDI velocity. ‘0’ is linear; either side is exponential or logarithmic.
Pressure curve	-100	100	0		The scaling curve for MIDI pressure (aftertouch). ‘0’ is linear; either side is exponential or logarithmic.

CV/gate setup

Polysynths can use up to 6 input busses as gates. Each gate can use up to 11 input busses as CVs. The CV busses follow immediately after the gate bus e.g. if you select input 1 as the gate, and choose to use two CVs for that gate, the CVs will be inputs 2 and 3.

While it might be convenient to have free assignment of each CV bus, this rapidly becomes unwieldy when you have a lot of busses to assign, so the present system is offered as a compromise between usability and flexibility.

Q: I thought CV/gates came in pairs? What does it mean to have one gate and multiple CVs?

A: It means that a gate will trigger a chord – one note per CV.

Gate velocity

Polysynth gates are “velocity sensitive” - the voltage of the gate is used as a velocity value, just like the velocity that you get from a MIDI keyboard. Typically this is used to scale note volume, but various synths might map it to filter cutoff, wavetable position, etc.

A gate voltage of 5V is “maximum velocity”.

Microtuning (Scala/MTS)

All polysynths support microtuning, as described above.

MPE

Polysynths support MPE (MIDI Polyphonic Expression). Per-note pitch bend, aftertouch, and the “Y dimension” (CC #74) are supported. Precisely what each polysynth does with this information is dependent on the synth.

MIDI channels for MPE are set using the ‘MIDI channel’ and ‘MPE channels’ parameters. The ‘MIDI channel’ sets the MPE common channel (typically channel 1). The ‘MPE channels’ parameter sets the range of channels that will be used for notes. If the common channel is 1, then the note channels will be from 2 up to a maximum. A special case is common channel as 16, in which case the note channels will go from 15 down to a minimum (this is an official MPE usage called the “upper zone”).

Chord generator/arpeggiator

The polysynths contain chord generators, which can add harmony notes to the notes you play, and an arpeggiator, to play those harmony notes as patterns instead of all at the same time.

The arpeggiator pattern is advanced by the gate, if playing via CV/gate, or by the MIDI or I2C note on message.

Harmony modes

There are two different ways in which the chord notes can be chosen.

The first, 'Shape’, builds up a chord using the original note as the root note, according to the shape and inversion parameters.

The second, 'SATB’, builds a chord according to common practice[44] four part harmony. (SATB stands for soprano/alto/tenor/bass.) The original note is assigned to one of the four voices according to the 'cantus firmus' parameter, and the remaining voices fill out the chord according to the root degree and position parameters. The root can also optionally be set from a CV input.

Note that the root is not the same as the key. In the key of C major for example, a root degree of II gives the chord D-F-A. In the key of D major, a root degree of I gives D-F♯-A.

44 https://en.wikipedia.org/wiki/Common_practice_period

Chords, arpeggios, and multiple CV inputs per gate

How these features interact warrants some clarification.

If there are multiple pitch CVs per gate, the chord generation is applied to each pitch CV. For example, if you supply the notes C and D, and set the chord generation to 'triad' in C major, you'll get the notes C, E, G, D, F, A.

If you enable arpeggiation, but not chords, then the arpeggiation is over the input pitch CVs. So for example if you have three CVs and one gate, then the arpeggiator will run over the three notes supplied to the CV inputs. Note that this is useful even if you have a single pitch CV per gate, since you can still use the arpeggio range setting to get octaves.

If chords and arpeggiation are both enabled, then the arpeggio is over all the notes generated by the chords. In most cases, the notes are sorted and then the Up, Down, etc. direction imposed. The exception is 'As Played' mode – in this mode the notes are ordered according to the CV input that generated them. In the above example, 'As Played' mode would give you C, E, G, D, F, A, in that order, whereas 'Up' would give you C, D, E, F, G, A.

Chord/arp parameters

Name	Min	Max	Default	Unit	Description
Enable	0	2	0		Enables the chord generator and chooses which variant to use. The options are ‘Off’, ‘Shape’, and ‘SATB’.
Key	-12	12	0		Sets the key of the chord (C, D ♭ , D, etc.).
Mode	1	12	1		Sets the mode, that is, the rotation of the notes within the scale. If the selected scale is ‘Major’, the mode is displayed by one of the familiar names Ionian, Dorian, Phrygian, Lydian, Mixolydian, Aeolian, or Locrian.
Scale	0	8	1		Sets the scale to use to build the chord. See above.
Shape	0	13	0		In ‘Shape’ mode: sets the chord shape. See below.
Inversion	0	3	0		In ‘Shape’ mode: sets the chord inversion. For example the first inversion takes the first note of the chord and moves it an octave up, so the lowest note in the chord is now the second (e.g. C E G becomes E G C). See e.g. here[45] for a fuller explanation of inversions.
Root degree	0	7	1		In ‘SATB’ mode: the root degree. Specified manually (I-VII) or as ‘From CV’.

45 https://en.wikipedia.org/wiki/Inversion_(music)#Inversions

Root CV	0	64	0		The input bus to use as the root degree pitch if ‘Root degree’ is ‘From CV’.
Cantus firmus	0	3	3		In ‘SATB’ mode: which voice (soprano/alto/tenor/bass) corresponds to the original note.
Position	0	5	0		In SATB mode: chooses Close or Open position, or one of the T-voice options. See below.
Arpeggio	0	14	0		Enables the arpeggiator and chooses the arpeggiation mode. See below.
Range	1	3	1		When set to 1, the arpeggio is simply the notes formed by the chord. When set to 2 or 3, a copy of the chord is appended to the pattern, one or two octaves higher, creating a longer pattern that spans multiple octaves.
Arp reset	0	64	0		The input bus to use as the arpeggiator reset. A trigger pulse into this input will reset the arpeggiator back to step 1.
Break time	0	1000	0	ms	The 'break' time for chords. See below.
Break dir	0	2	0		The 'break' direction. See below.

Chord shapes

The available shapes are as follows.

Name	**Notes (within scale)**	**Example (in C major)**
None	1	C
Octave	1-1(8ve)	C C(8ve)
Two Octaves	1-1(8ve)-1(15ma)	C C(8ve) C(15ma)
Root/Fifth	1-5	C G
Root/Fifth + 8ve	1-5-1(8ve)	C G C(8ve)
Triad	1-3-5	C E G
Triad + 8ve	1-3-5-1(8ve)	C E G C(8ve)
Sus4	1-4-5	C F G
Sus4 + 8ve	1-4-5-1(8ve)	C F G C(8ve)
Sixth	1-3-5-6	C E G A

Sixth + 8ve	1-3-5-6-1(8ve)	C E G A C(8ve)
Seventh	1-3-5-7	C E G B
Seventh + 8ve	1-3-5-7-1(8ve)	C E G B C(8ve)
Ninth	1-3-5-7-2(8ve)	C E G B D

SATB mode – positions

In SATB mode, the 'Position' parameter controls the spacing between the various voices.

When set to 'Close', the voices adopt the closest triad notes possible to the cantus firmus. For example, if the cantus firmus is the soprano, the key is 'C', and the soprano voice is C4, then the alto, tenor, and bass notes will be G3, E3, and C3 respectively.

When set to 'Open', as compared to 'Close', the alto takes the tenor note, and the tenor takes the alto note dropped by an octave. The bass also drops an octave if required to keep it below the tenor. So in our example above, the alto, tenor, and bass notes will be E3, G2, and C2 respectively.

The remaining options for position are based on the 'Tintinnabuli'[46] style developed by the composer Arvo Pärt. They make most sense when the cantus firmus is the alto part, which we will assume here – if not, the voices are calculated as explained and then simply permuted so what we call the alto here is the desired voice.

The bass voice is simply chosen as one octave below the alto. The tenor and soprano are chosen as follows:

T-vce +1 -1	Soprano is first position superior (the closest triad note above the alto). Tenor is first position inferior (the closest triad note below the alto).
T-vce +2 -1	Soprano is second position superior (the triad note one up from first position). Tenor is first position inferior.
T-vce +1 -2	Soprano is first position superior. Tenor is second position inferior (the triad note one down from first position).
T-vce +2 -2	Soprano is second position superior. Tenor is second position inferior.

Arpeggio modes

The options are as follows:

Name	Behaviour	Example (on C major triad)
Up	Notes are played from lowest to highest.	C E G C E G ...

46 See e.g. https://en.wikipedia.org/wiki/Tintinnabuli

Down	Notes are played from highest to lowest.	G E C G E C ...
Alt	Notes are played alternately up and down.	C E G E C E G ...
Alt2	Notes are played alternately up and down, repeating the top & bottom notes.	C E G G E C C E G ...
Up -8ve	See below.	
Down -8ve	See below.	
Alt -8ve	See below.	
Alt2 -8ve	See below.	
Step Up	The lowest note alternates with the other notes in the chord, in rising order.	C E C G C E C G ...
Step Down	The highest note alternates with the other notes in the chord, in descending order.	G E G C G E G C ...
Random	Notes are played in a random order.	
Random2	Notes are played in a random order, except that the same note cannot be played twice in a row.	
Random3	The notes in the chord are played in a random permutation, then another random permutation, and so on.	
As Played	Notes are played according to the order of their CV inputs.	

The “-8ve” modes differ from the basic modes in how they treat the Range parameter (above), for shapes which end in “+8ve”. As an example, consider the Triad+8ve shape in C major, which contains the notes:

C E G C(8ve)

If Range is set to 2, this pattern is repeated an octave higher, so modes Up/Down/Alt/Alt2 will arpeggiate the notes:

C E G C(8ve) C(8ve) E(8ve) G(8ve) C(15ma)

Note how C(8ve) is repeated. The “-8ve” modes skip this repeated note, so for example the Up-8ve mode will play:

C E G C(8ve) E(8ve) G(8ve) C(15ma) C E G …

Broken chords

The 'Break time' and 'Break direction' parameters allow you to automatically 'break' chords i.e. the notes are played one at a time, instead of simultaneously. With a small time, this gives a 'strumming'

effect, though you can specify much longer times if you wish so each note is quite distinctly separated.

Note that this applies to multiple notes triggered when using a single gate input with multiple CV inputs, as well as when chord mode is activated.

The 'Break direction' allows you to specify whether the chord is played from bottom to top (‘Up’), from top to bottom (‘Down’), or alternately up and down (‘Alternating’).

The arpeggio reset input, if used, also serves to reset the alternating break direction to up.

Sustain mode

The polysynth ‘Sustain mode’ parameter sets the behaviour of the sustain function, when triggered either by MIDI or by the ‘Sustain’ parameter. The options are “Synth” (sustained notes cannot be retriggered) and “Piano” (sustained notes can be retriggered).

Plug-ins

As well as the various options for scripting (e.g. Lua Script algorithm, below), the disting NT supports *bona fide* plug-ins via a C++ API.

Here we'll focus on installing and load plug-ins from a user perspective, not on developing them.

Developers interested in the API are referred to our GitHub[47] and to our Discord server.

Currently plug-ins can define new algorithms. It is possible that in the future plug-ins might offer other new functionality. We're open to suggestions.

Plug-in gallery

There is a third party website called NT Gallery[48] which offers a browsable and searchable database of plug-ins the for the disting NT.

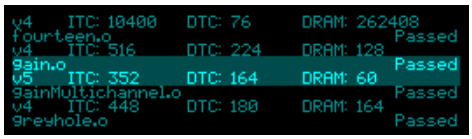
Installing plug-ins

Plug-ins are placed on the MicroSD card in the folder `/programs/plug-ins`

The expected file extension for plug-ins is `.o`

You can install plug-ins manually by simply copying the file to the MicroSD card. You can also install them via the NT Helper[49] app for macOS/Windows/Linux/iOS/Android.

The module scans the folder for plug-ins at startup (and if the card is ever remounted). You can view a list of what it found via the Misc / Plug-ins / View info… menu, which shows a screen like this:



The entry for each plug-in shows the filename (top left), whether it passed or failed scan (top right), and statistics for how much of various types of memory will be consumed if the plug-in is loaded. The API version that the plug-in was built with is shown below the filename.

Loading plug-in algorithms

Algorithms loaded from plug-ins are handled by the system just like any other algorithm. The only difference is that they need to be loaded before use. This can be done manually from the 'Add algorithm' menu; the module also attempts to load plug-ins automatically when loading a preset file that uses them.

47 https://github.com/expertsleepersltd/distingNT_API

48 https://nt-gallery.nosuch.dev

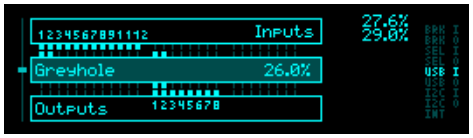
49 https://nosuch.dev/nt-helper/

Plug-in algorithms appear after the built-in algorithms in the list:

If the plug-in is not loaded, where you would normally see “> Add <” you’ll see “> Load <”:

If you click the encoder as usual, the plug-in will be loaded. If this succeeds, “> Load <” will change to the usual “> Add <”:

You can then add the algorithm as normal.



There is currently no way to unload plug-ins via the user interface, other than rebooting the module. There is a mechanism for resetting and rescanning the MicroSD card for plug-ins, which is intended as part of the development workflow.

Faust

One of the motivations for the disting NT’s C++ API was to be able to support Faust[50].

From their website:

> Faust (Functional Audio Stream) is a functional programming language for sound synthesis and audio processing with a strong focus on the design of synthesizers, musical instruments, audio effects, etc. created at the GRAME-CNCM Research Department.
>
> Faust targets high-performance signal processing applications and audio plug-ins for a variety of platforms and standards.
>
> The core component of Faust is its compiler. It allows to "translate" any Faust digital signal processing (DSP) specification to a wide range of non-domain specific languages such as C++, C, LLVM bit code, WebAssembly, Rust, etc. In this regard, Faust can be seen as an alternative to C++ but is much simpler and intuitive to learn.

50 https://faust.grame.fr

> Thanks to a wrapping system called "architectures," codes generated by Faust can be easily compiled into a wide variety of objects ranging from audio plug-ins to standalone applications or smartphone and web apps, etc.

We have created such an “architecture” to allow Faust programs to be compiled for the disting NT. This is currently in our GitHub here[51], though the intention is to contribute this to the actual Faust project so it will ultimately be part of the Faust installation itself.

Whether Faust is in fact more “intuitive” than C++ is open for debate, but there are undeniably some very attractive features:

- There’s a large library[52] of existing work written in Faust which can simply be picked up and compiled for any platform, now including the disting NT.
- There’s an online IDE[53], where you can interactively develop your code and run it, processing audio files that you supply or live audio from your computer.
- There’s even a GUI app[54] where you can construct your DSP code by dragging in blocks of processing and connecting them up with virtual wires.

It’s early days for Faust on the disting NT – if you have any experience to share, or thoughts on future development, please let us know. Currently it works quite well for audio effects (reverbs, filters etc.) and you’ll find working examples of these in our GitHub.

51 https://github.com/expertsleepersltd/distingNT_API/tree/main/faust

52 https://faustlibraries.grame.fr/libs/

53 https://faustide.grame.fr/

54 https://faustplayground.grame.fr

Algorithm reference

The pages that follow detail the various algorithms available.

Accent sweep

“AD envelope with variable peak”

File format guid: 'acsw'

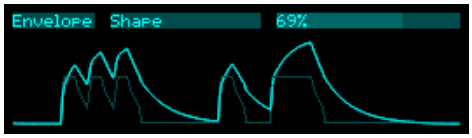
Specifications: None

Description

This algorithm recreates a very specific aspect of the Roland TB-303’s circuitry. There is a lengthy write-up on it on the Devil Fish page here[55], but essentially it creates an AD (attack/decay) envelope which will reach a higher maximum level if repeatedly triggered.

GUI

The display shows the input (dark) and output (light) signals.



Envelope parameters

Name	Min	Max	Default	Unit	Description
Input mode	0	1	0		Sets how the input is treated: “Direct” feeds the input signal directly into the AD shaper. “Trigger” uses the input to trigger an internal pulse, which is fed into the shaper.
Trigger length	1	200	100	ms	If the input mode is “Trigger”, the length of the pulse.
Trigger voltage	0.1	10.0	10.0	V	If the input mode is “Trigger”, the voltage of the pulse.
Shape	0	100	100	%	This is the position of the pot (potentiometer), if you studied the Devil Fish write-up. At 0% the output is quite close to the input; at 100% it is heavily affected by the AD shaper.
Gain	0	24	0	dB	Applies a gain to the output shape.
Charge	1	470	47	kΩ	The value of the charge resistor.

55 https://www.firstpr.com.au/rwi/dfish/303-unique.html

Discharge	1	470	100	kΩ	The value of the discharge resistor.
Pot	1	470	100	kΩ	The value of the potentiometer.
Capacitance	0.1	10.0	1.0	µF	The value of the capacitor.

Routing parameters

Name	**Min**	**Max**	**Default**	**Unit**	**Description**
Input	1	64	1		The bus to use as input.
Output	1	64	15		The bus to use as output.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Arpeggiator

“Makes sequences from held chords”

File format guid: 'arpg'

Specifications: None

Description

This algorithm is a very flexible arpeggiator, in some respects an interactive, performance-oriented sequencer. It is based on the arpeggiator in our FH-2[56] module. You may like to watch these videos, which highlight certain features in the FH-2 version: link[57], link[58], link[59].

It takes MIDI or CV/gate input and generates MIDI and/or CV/gate output.

The key to using this as an interactive sequencer is to map the ‘Commands’ parameters, so you can use them alongside playing notes to manipulate your patterns.

When using it as a MIDI-to-MIDI arpeggiator, and sending the MIDI internally to other algorithms, remember that algorithms in the disting NT do not ‘consume’ their input MIDI – the MIDI entering the arpeggiator will still arrive at the other algorithms as well. So, set the output channel of the arpeggiator to a different channel than the input, and have the other algorithms listen to that channel.

For the purposes of MIDI and CV/gate input this algorithm is a polysynth. Please refer to the section “Common polysynth features” above.

Latch

If 'Latch' is enabled, notes are held until all keys are released and a new note played. This makes it particularly easy to play a chord, have it repeat and arpeggiate, play another chord, and so on.

Note that this is different to using the sustain pedal. If sustain is held, notes continue to accumulate until the sustain is released. When latch is used, the selection of held notes starts from scratch each time a new note is played after all previous notes have been released.

56 https://expert-sleepers.co.uk/fh2.html

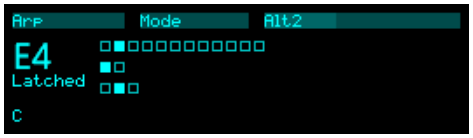
57 https://www.youtube.com/watch?v=norj2n1cofg

58 https://www.youtube.com/watch?v=9ucrlQAqeqU

59 https://www.youtube.com/watch?v=Hi94yKj2njg

GUI

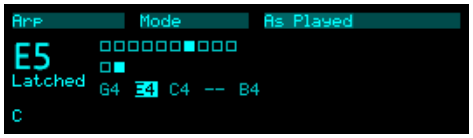
The display shows various information relating to the current arpeggiation pattern.



On the left you see the current output note, whether the pattern is currently latched, and the current root note (see the discussion of transposition below).

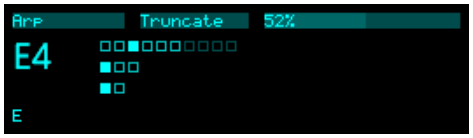
The three lines in the main section of the display show, from top to bottom, a representation of the entire arpeggiation pattern, the octaves of the pattern (when ‘Range’ is used), and the note within a single octave of the pattern.

For the ‘as played’ modes, the lowest line instead shows actual note numbers:



A ‘--’ represents a rest added with the ‘Rest’ command.

When ‘Truncate’ is used, the omitted steps are greyed out:



Arp parameters

Name	Min	Max	Default	Unit	Description
Mode	0	11	0		The arpeggiation mode – see below.
Range	1	3	1		The number of octaves to repeat the arpeggiation pattern over.
Latch	0	1	0		Enables latching. See above.
Truncate	0	100	100	%	Shortens the arpeggiation pattern by only playing the first portion, according to the percentage set.
Gate type	0	1	0		Sets the gate type: “Trigger” or “% of clock”.
Length (ms)	1	100	10	ms	Sets the length of the gate output if the type is “Trigger”.

Length (%)	1	100	50	%	Sets the length of the gate output if the type is “% of clock”.

Commands parameters

Name	Min	Max	Default	Unit	Description
Add	0	1	0		While enabled, incoming notes will be added to the arpeggiation pattern. This is most useful when the Mode is ‘As Played’ and Latch is on. It allows you to have repeated notes in the pattern.
Remove	0	1	0		While enabled, incoming notes will be removed from the pattern (if the note in question is actually in the pattern already). If the same note appears multiple times, the last occurrence will be removed.
Rest	0	1	0		When enabled, a rest is added to or removed from the arpeggiation pattern, according to whether Add or Remove is currently active.
Transpose	0	1	0		While enabled, incoming notes will transpose the current pattern, by the amount of the difference between the incoming note and the root note of the pattern. The root note is by default the lowest note of the pattern, but see ‘Root note’ below.
Transpose in key	0	1	0		Like Transpose, but rather than a straight chromatic transposition, the pattern is transposed within the key defined by the root note. For example, if the pattern is C-E-G (root note C) and D is played, 'Transpose' will produce D-F♯-A, whereas 'Transpose in key' will produce D-F-A.
Root note	0	1	0		While enabled, an incoming note overrides the root note, so that subsequent transposition is based on the overridden note. For example, if the pattern is E-G-C, the default root note will be E, but you could use the ’Root note’ command to let the arpeggiator know the key is C.

MIDI/I2C parameters

Name	Min	Max	Default	Unit	Description
MIDI channel	0	16	1		The MIDI channel to listen on.

MPE channels	1	16	1		Controls how the algorithm will respond to MPE. See above.
I2C channel	0	255	1		Sets the I2C channel.
Sustain	0	1	0		Directly controls the sustain (like a MIDI sustain pedal).
Sustain mode	0	1	0		The standard polysynth sustain mode parameter. See above.

Sync parameters

Name	Min	Max	Default	Unit	Description
Clock input	0	64	0		The bus to use for the clock input.
Reset input	0	64	0		The bus to use for the reset input.
Reset mode	0	1	0		The reset mode, one of ‘Trigger’ or ‘Run/stop’.
Follow MIDI clock	0	1	0		Sets whether the arpeggiator follows MIDI clock.
Divisor	0	19	4		Sets the clock divisor when following MIDI clock.
Auto reset	0	128	0		Sets a number of quarter notes after which the arpeggiator will automatically reset to step 1 (only available when following MIDI clock).

CV/gate in parameters

The algorithm uses the standard polysynth CV/gate parameters, as described above.

MIDI out parameters

Name	Min	Max	Default	Unit	Description
MIDI channel	0	16	0		The output MIDI channel.
Output to breakout	0	1	0		Enables MIDI output to the breakout.
Output to Select Bus	0	1	0		Enables MIDI output to the Select Bus.
Output to USB	0	1	0		Enables MIDI output to USB.

Output to internal	0	1	0		Enables internal MIDI output – that is, MIDI is sent to the other algorithms.

Microtuning parameters

The algorithm uses the standard polysynth microtuning parameters, as described above.

CV/gate out parameters

Name	Min	Max	Default	Unit	Description
Gate output	0	64	0		The bus to use for the gate output.
Gate mode	0	1	0		The standard Add/Replace mode selector as described above, for the gate output.
Pitch output	0	64	0		The bus to use for the pitch output.
Pitch mode	0	1	0		The standard Add/Replace mode selector as described above, for the pitch output.
Velocity output	0	64	0		The bus to use for the velocity output.
Velocity mode	0	1	0		The standard Add/Replace mode selector as described above, for the velocity output.

Arpeggiator modes

Off	The arpeggiator is disabled.
Up	Notes are used in order from lowest to highest.
Down	Notes are used in order from highest to lowest.
Alt	Alternately up & down (e.g. 4 notes held gives you a 6 note pattern).
Alt2	Alternately up & down, repeating the highest and lowest notes (e.g. 4 notes held gives you an 8 note pattern).
As Played	The notes are repeated in the order in which they were played.
Step Up	The lowest note alternates with the other notes in the chord, in rising order.
Step Down	The highest note alternates with the other notes in the chord, in descending order.
Step As Played	The first note played alternates with the other notes in the chord, in the order played.

Random	The notes held play in a random order.
Random2	Notes are played in a random order, except that the same note cannot be played twice in a row.
Random3	The notes in the chord are played in a random permutation, then another random permutation, and so on.

Attenuverter

“Attenuates and offsets signals”

File format guid: 'attn'

Specifications:

- Channels, 1-12: The number of bus channels to process.

Description

This simple algorithm scales and offsets signals. It is probably most useful for CVs, but can be used for audio as well.

The signal is scaled and then offset i.e.

$$output = offset + (input \times scale)$$

Various offset parameters are provided, which might be useful in different scenarios – they are all simply added together.

Note that it is valid to use this algorithm with no input bus selected, in which case the output is simply the offset voltage.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Input	0	64	1		The input bus to process.
Enable	0	1	0		Enables the channel. Disabled channels have no effect on the bus.
Output	0	64	0		The output bus. If set to ‘None’, the input bus is used as output, and the mode is always ‘Replace’.
Output mode	0	1	1		The standard Add/Replace mode selector as described above.
Scale	-200.0	200.0	100.0	%	Sets the channel scale.
Offset	-10.0	10.0	0.0	V	Sets the channel offset (Volts).
Fine	-1000	1000	0	mV	Sets the channel offset (millivolts).
Octaves	-10	10	0	V	Sets the channel offset (whole Volts).
Semitones	-60	60	0	ST	Sets the channel offset (semitones i.e. 1/12th of a Volt).

Audio recorder

“Records audio to the MicroSD card”

File format guid: 'wavr'

Specifications:

- Max files, 1-10: The maximum number of simultaneous files to record.

Description

This algorithm records WAV files to the MicroSD card. A typical use for this would be to record the module’s inputs, but any bus can be the source for a recording, so you could also record the output of other algorithms – for example, you could run a mixer and record the stereo mix-down.

The algorithm can also play back its recordings. It does so to the same bus channels as are chosen for recording.

Since the module is fully DC-coupled, it can also record and play CVs.

Up to 10 files can be recorded simultaneously, each mono or stereo. It goes without saying that the more files you record, and the larger the chosen bit depth, the more demands are placed on the MicroSD card. You would be advised to check your card performance before relying on the algorithm for any crucial recordings.

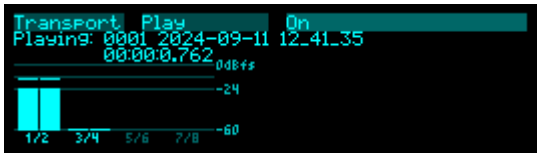
GUI

The display shows a level meter for each file. This will be grayed out if the file is disabled. The meters show the input levels when idle or recording, and the playback levels when playing.

While recording, the display shows the name of the current recording, and the elapsed time. It also shows a gauge of the pressure on the MicroSD card.

When idle, the display shows the name of the last recording completed or played.

During playback, the display shows the name of the recording being played, and the elapsed time.



Parameters

Name	Min	Max	Default	Unit	Description
Record	0	1	0		Starts/stops recording.
Play	0	1	0		Starts/stops playback.
Record lock	0	1	0		If on, changes to the Record parameter are ignored.
Play lock	0	1	0		If on, changes to the Play parameter are ignored.
Bit depth	0	2	0		Chooses the bit depth for recording. The options are 16, 24, or 32 bit.
Normalisation	0	1	0		Sets the voltage that corresponds to full-scale in the recorded files. The options are 10V or 12V. Applies to both recording and playback.
Which recording					Selects the recording to play.
Gain	-40	24	0	dB	Sets the playback level.

Per-file parameters

Name	Min	Max	Default	Unit	Description
Enable	0	1	0		Enables the file for recording and playback.
Left/mono input	1	64	1		Sets the left or mono channel to record.
Right input	0	64	2		Sets the right channel to record.
Playback	0	1	1		Enables the file for playback. (Both ‘Enable’ and ‘Playback’ must be on for the file to be included in playback.)

Augustus Loop

“Tape-like delay”

File format guid: 'augu'

Specifications:

- Max delay time, 1-44 seconds: The maximum delay time.

Description

Augustus Loop is a disting NT implementation of one of Expert Sleepers' oldest products, the VST plug-in of the same name (here[60]). Essentially, it's a tape-inspired stereo delay.

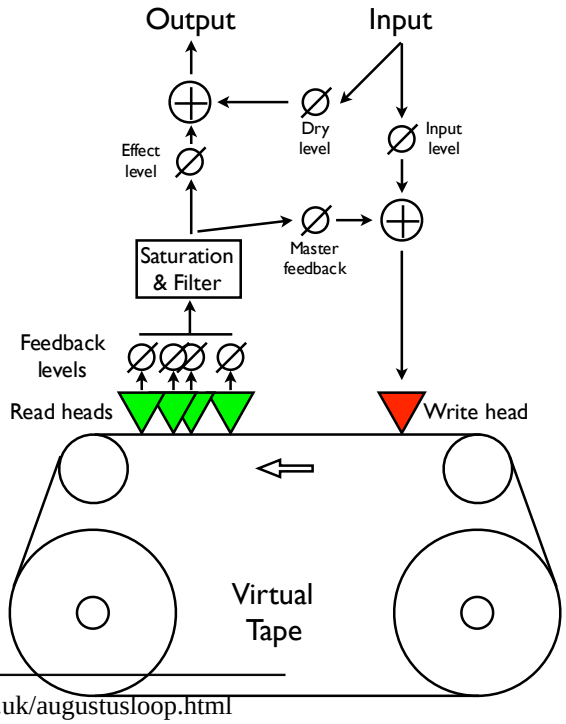
The delay time can be dialled in manually, or set by tap tempo or a clock input. The longest delay possible is around 44 seconds.

Being a tape delay, you can change the tape speed via CV. Patching an LFO into here is your route to all manner of subtle detuning or extreme mangling effects. You can also stop and reverse the tape.

There is an option to run the delay signal through an 'effects loop', allowing you to insert other effects or processing into the guts of the delay.

Note that the 'Pitch CV' input changes the tape speed. It is labelled pitch rather than speed to emphasise the fact that it is scaled 1V/octave.

This diagram is reproduced from the VST plug-in user manual, and explains the signal flow graphically:



60 https://expert-sleepers.co.uk/augustusloop.html

There are four 'tape read heads' with independent delay times and stereo positions, allowing for straight stereo delays, ping pong delays, or hybrid multi-tap style effects.

GUI

The display shows the feedback amount bottom left and the delay time bottom right. The centre area shows an animation indicating the tape direction and speed.

If using a clock input, the word ‘Clock’ flashes up every time the algorithm receives a clock pulse.

The word ‘Tap’ is displayed if the tap tempo function will set the delay time on the next tap i.e. it shows whether tap tempo is ‘live’.

Delay parameters

Name	Min	Max	Default	Unit	Description
Time (coarse)	0.0	43.7	1.0	s	Sets the delay time. The coarse and fine delay times are added to produce the actual delay time.
Time (fine)	-100	100	0	ms	Sets an adjustment to the delay time, in milliseconds. The coarse and fine delay times are added to produce the actual delay time.
Delay multiplier	0	23	15		A multiplier to apply to the delay time set by the parameters, the tap tempo, or the clock. See below for the available values.
Feedback	0	100	50	%	The overall delay feedback amount.
L-L Time	0	100	100	%	Scales the delay time of the left-to-left feedback path, as a percentage of the overall delay time.
L-R Time	0	100	50	%	Scales the delay time of the left-to-right feedback path, as a percentage of the overall delay time.
R-L Time	0	100	50	%	Scales the delay time of the right-to-left feedback path, as a percentage of the overall delay time.
R-R Time	0	100	100	%	Scales the delay time of the right-to-right feedback path, as a percentage of the overall delay time.
L-L Level	0	100	100	%	Scales the amount of the delayed left signal mixed into the left feedback path.
L-R Level	0	100	0	%	Scales the amount of the delayed left signal mixed into the right feedback path.

R-L Level	0	100	0	%	Scales the amount of the delayed right signal mixed into the left feedback path.
R-R Level	0	100	100	%	Scales the amount of the delayed right signal mixed into the right feedback path.
Mono-ize	0	100	0	%	Reduces the stereo width of the incoming signal. At zero the signal is reduced to mono, at a pan position set by the 'Initial pan' parameter.
Initial pan	-100	100	-100	%	Sets the pan position of the mono-ized signal. -100 is fully left; 100 is fully right.

Mix parameters

Name	Min	Max	Default	Unit	Description
Dry gain	-40	6	-40	dB	The amount of the dry signal to mix into the outputs. At “-40” there is no dry signal at all i.e. it's actually –∞dB.
Effect gain	-40	6	-3	dB	The amount of the effect (delay) signal to mix into the outputs. At “-40” there is no effect signal at all i.e. it's actually –∞dB.
Input level	0	100	100	%	Attenuates the input signal fed to the tape write head.

Tape parameters

Name	Min	Max	Default	Unit	Description
Pitch inertia	0	125	64		Sets the amount of 'inertia' or slew on the pitch CV input. At zero, the tape speed follows the pitch input closely; at the maximum value, pitch changes are quite gradual.
Stop tape	0	1	0		When on, the tape speed is set to zero. Note that the Pitch inertia affects how quickly the tape stops and starts.
Reverse tape	0	1	0		When on, the tape is reversed. Note that the Pitch inertia affects how quickly the tape reverses.
Bit depth	0	1	1		Controls the bit depth used in the delay memory (note, not the bit depth used in any other processing). Setting this to '0' (16 bit) doubles the maximum delay time.
Inertia free	0	1	0		Enables 'Inertia free' mode. See below.

Inertia fade time	1	1000	100	ms	The fade time to use when in Inertia free mode.
Pitch CV input	0	64	0		The CV input to use for pitch.
Pitch LFO speed	0	127	96		Sets the speed of the pitch modulation LFO.
Pitch LFO depth	0	100	0	%	Sets the depth of pitch modulation by the LFO.
Clear loop	0	1	0		When on, instigates a rapid clear of the delay buffer (while maintaining passthrough of the dry signal).

Filter/Sat parameters

Name	Min	Max	Default	Unit	Description
Filter type	0	400	0		Sets the filter type. See below.
Filter freq	0	127	64		Sets the filter frequency.
Filter Q	0	100	20		Sets the filter resonance.
Saturation enable	0	1	1		Enables the saturation processing (on the tape output, before the filter).
Saturation	0	110	0		Sets the depth of the saturation effect, by applying gain before the saturation waveshaper.
Saturation shape	0	100	100		Controls the shape of the saturation. At '100' the effect is that of soft saturation and clipping. At '0' the effect is of hard digital clipping. Note that at shape settings other than '0', some alteration is applied to the signal even if the Saturation level is zero.

Tempo parameters

Name	Min	Max	Default	Unit	Description
Clock input	0	64	0		The CV input to use as the clock. The delay time is set as the time between two rising clock edges.
Tap tempo	0	1	0		When this parameter transitions from 0 to 1, the algorithm acts on a tap tempo event. See tap tempo, below.
Clocks required	1	10	1		Sets the number of consistent clocks required to change the delay time. See below.

Routing parameters

Name	Min	Max	Default	Unit	Description
Left input	1	64	1		Sets the bus for the left input.
Right input	1	64	1		Sets the bus for the right input.
Left output	1	64	13		Sets the bus for the left output.
Right output	1	64	14		Sets the bus for the right output.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.
FX Loop position	0	2	0		Enables the effects loop and sets in position the signal flow. See below.
FX Loop output L	0	64	0		Sets the left output to the effect loop.
FX Loop output R	0	64	0		Sets the right output to the effect loop.
FX Loop input L	0	64	0		Sets the left input from the effects loop.
FX Loop input R	0	64	0		Sets the right input from the effects loop.

Delay time multipliers

The available delay time multipliers are as follows:

1/64	1/48	1/32	1/24	1/16	1/12	1/8	1/6
3/16	1/4	5/16	1/3	3/8	1/2	3/4	x1
x1.5	x2	x3	x4	x5	x6	x8	x16

Tap tempo

The 'Tap tempo' parameter allows for a tap tempo function. You might like to control this from a button push in a UI script, or from an external MIDI controller.

Two taps are required to set the delay time. Taps more than 11 seconds apart are ignored.

Clocks required

When using the clock input, the algorithm's default behaviour is to follow every clock pulse and immediately change the delay time. This is appropriate if you're using a clock with variable timing (perhaps the gate output from a sequencer rather than a clock per se).

However, sometimes you're actually wanting a steady clock, but occasionally the time between clocks changes anyway – for example, if the clock is coming from your DAW or sequencer, the clock will stop when the transport stops, and then the first clock when the transport starts will be interpreted as a really long clock (the time between stopping and starting the transport).

The 'Clocks required' parameter is a solution to this problem. By raising the value above '1', you're telling the algorithm to only change the delay time when it receives a number of clocks of the same time in succession ('same' here means within 10%) - so it will ignore the rogue clock you get from stopping and starting the transport.

Inertia free mode

“Inertia free” mode relates to the algorithm's behaviour when the delay time is changed, either by changing the master delay time, the multiplier, or the four L-L, L-R, etc. times.[61]

When inertia free mode is not activated, the effect is as if the physical tape heads on a tape machine were slid along the tape to adjust the write/read head gap. This results in a characteristic and fairly drastic pitch slew sound.

When inertia free mode is activated, the algorithm crossfades between the old and new delay times, which is a much more subtle effect. The length of the crossfade can be set with the 'Inertia fade time' parameter.

Filter

A second order state variable filter is available within the delay feedback path. The 'Filter type' parameter lets you smoothly fade between the various responses – 'thru' (i.e. no filtering), low pass, band pass, high pass, and back to thru.

Effects Loop

Enabling the 'FX Loop position' (i.e. setting it to something other than Off) breaks the internal delay feedback path and sends it out of and back into the module, allowing you to insert other effects or processing into the delay. Note that this is different to simply inserting another effect after the delay output – using the effects loop, each delay repeat is progressively more processed by the external effects.

Simple ideas include putting external VCAs in the loop to control the amount of feedback. Or you could put other delays, reverbs, or pitch effects (e.g. chorus) into the loop. Putting a pitch shifter into the loop gives you the classic “pitch spiralling off to infinity” sound.

61 Granted, the name of this parameter isn't particularly well chosen, but this is what it's called in the VST version, and we're sticking with it for the sake of consistency.

The two options for the effects loop position are 'Pre-Filter' and 'Post-Filter', which as you might expect places the external loop either before or after the filter, giving you the option of filtering before you send the audio to the external effects, or filtering the result that comes back in.

Auto-calibrator

“Calibrates pitch CV outputs”

File format guid: 'cali'

Specifications:

- Channels, 1-8: The number of bus channels to process.

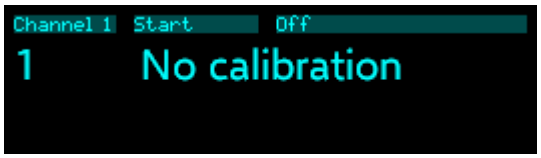
Description

This algorithm offers oscillator calibration, similar to that in our Silent Way Voice Controller VST plug-in, in the FH-2 module, and in various algorithms on the disting EX.

Pass the pitch CV from another algorithm (for example, the Quantizer) or another module through this algorithm, and connect the algorithm output to your VCO’s CV input.

To perform the calibration, you will also need to connect an output from the VCO to an input on this algorithm. This is only required during calibration, and can be disconnected afterwards (unless you’re using continuous mode, as described below).

To start the calibration, set the ‘Start’ parameter to ‘On’. The module will output a series of voltages, ranging from -4 Volts up to +6 Volts, and analyse the pitch of the resulting signal from the VCO. As it does so, it will draw a graph of the results, which will ideally look like a nice straight line.



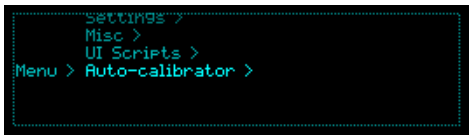
Once the calibration is complete, the algorithm will continuously modify the pitch CV it receives to achieve the expected frequency from the VCO.

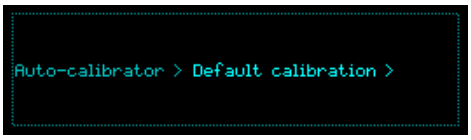
To successfully calibrate, the algorithm needs a fairly clean waveform from the VCO – for example a nice sine or triangle wave. If your VCO is unable to provide this, consider using the default calibration and continuous mode, as described below.

When discussing this process we may say “calibrate an output” or “calibrate a VCO” but what is really being calibrated is the *combination* of the disting NT output *and* the VCO. Both may in fact be perfectly well calibrated, in terms of tracking, but the absolute pitch of a VCO is usually determined by a physical tuning knob (not to mention temperature and other factors) and so is hard to know precisely. This process saves you having to tune by hand.

Default calibration

You can use the algorithm's menu to set a 'default' calibration – the perfect straight line you'd get if everything was perfectly calibrated. This is in fact functionally equivalent to disabling the channel altogether, except that it allows you to use the calibrator in 'continuous' mode, as described next.



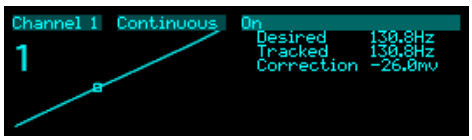


Continuous mode

Activating a channel's 'Continuous' parameter causes it to continually listen to the VCO output and tweak its pitch CV to achieve the correct pitch. (In this case, you have to leave the VCO's output connected, rather than removing it after calibration.)

During continuous calibration the algorithm is much more forgiving of the VCO waveform (because it already has an idea of what it's looking for). For some VCOs which might prove tricky to successfully calibrate using the standard, non continuous procedure, you might get better results using the default calibration, tuning the VCO by hand to the approximately correct pitch, and then letting continuous mode fine tune it.

When continuous mode is enabled, the GUI shows the desired VCO pitch, the actual pitch that the algorithm has determined that the VCO is outputting, and the correction voltage that the algorithm is applying in order to keep the VCO in tune.



Per-channel parameters

Name	Min	Max	Default	Unit	Description
Enable	0	1	1		Enables the algorithm. If disabled, the algorithm passes through CVs unmodified.
Start	0	1	0		Used to start the calibration process.
CV input	1	64	16		The pitch CV input bus.
CV output	1	64	16		The pitch CV output bus. Always uses "Replace" output mode.

Audio input	1	64	1		The audio input to use during calibration.
Continuous	0	1	0		Activates continuous calibration mode.
Max error	0	1000	50	mV	The maximum voltage error that continuous calibration mode will attempt to correct – beyond this, it assumes it has just failed to track the pitch correctly.
Time	0.1	10.0	1.0	s	The time constant for continuous mode correction. Use a high value if you just want to correct for long-term oscillator drift; use a low value for rapid, autotune-like pitch correction.

Auto-sampler

“Creates multisamples”

File format guid: 'auto'

Specifications: None

Description

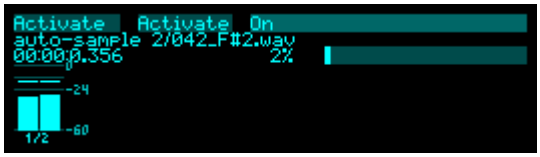
This algorithm allows you to automatically create multi-sampled instruments in a format that can be used by the Poly Multisample algorithm, by triggering an external synth (by MIDI or CV/gate) and recording the resulting audio.

You can set the range of notes to be sampled, and the step size (e.g. every note, every fourth note etc.). You can also choose to sample multiple velocity levels per note, and multiple round-robins of each note.

The samples are stored on the MicroSD card, in a folder within the root ‘samples’ folder.

GUI

While sampling, the display shows a level meter, the filename of the sample currently being recorded, and the elapsed time. It also shows a gauge of the pressure on the MicroSD card.



Recording parameters

Name	Min	Max	Default	Unit	Description
Folder name					The name of the folder to create when starting a recording. If the folder already exists, a number will be appended to make the new name unique.
Bit depth	0	2	0		Chooses the bit depth for recording. The options are 16, 24, or 32 bit.
Normalisation	0	1	0		Sets the input voltage that corresponds to full-scale in the recorded files. The options are 10V or 12V.
Left/mono input	1	64	1		The left channel input to record.
Right input	0	64	2		The right channel input, if recording in stereo.

Outputs parameters

Name	Min	Max	Default	Unit	Description
CV output	0	64	14		The pitch CV output bus.
Gate output	0	64	15		The gate output bus.
MIDI output	0	4	0		The MIDI output port (None, Breakout, Select Bus, USB, Internal).
MIDI channel	1	16	1		The output MIDI channel.

Setup parameters

Name	Min	Max	Default	Unit	Description
Start Note	0	127	21		The first note to sample.
End Note	0	127	108		The last note to sample.
Note Step	1	127	1		The number of notes to increment by after each sample (e.g. ‘1’ samples every note, ‘12’ samples every octave etc.).
Vel switches	1	9	1		The number of velocity switch layers to sample.
Min velocity	1	127	25		The minimum velocity value to use when sampling with velocity switch layers.
Max velocity	1	127	127		The maximum velocity value to use when sampling with velocity switch layers, or the fixed value to use if not using velocity switches.
Round robins	1	9	1		The number of round-robins to sample.
Length	0.1	60.0	1.0	s	The note length (i.e. how long the gate is held high, or between MIDI note on and note off).
Gap	0.1	60.0	0.1	s	The time to keep recording after the note is released.
Latency	0	2048	0		Adjusts for latency. See below.
Preview note	0	127	48		Sets the MIDI note number to use when previewing.

Activate parameters

Name	Min	Max	Default	Unit	Description
Activate	0	1	0		Activates auto-sampling when on.
Test	0	1	0		When on, triggers a note so you can test the timing parameters (Length and Gap) and the latency.

Latency

When you activate the Test note, the module listens for incoming audio and measures the delay (latency) between generating the note (sending MIDI or outputting a gate) and receiving audio. This latency figure is shown in the display.

Set the Latency parameter to something close to the shown value to avoid a silence at the start of every recorded sample. Some experimentation may be required to find the optimum value.

Bit Crusher

“Sample rate and bit depth reduction”

File format guid: 'btcr'

Specifications: None

Description

This algorithm is an implementation of the disting mk4 algorithm of the same name. You may like to review the video about that algorithm, which is here[62].

The algorithm is a 'bit crusher' – it applies sample rate and sample depth reduction to deliberately introduce quantisation and aliasing artefacts. It also optionally applies bitwise modification of the samples for non-linear distortion effects.

The ‘Bit reduce’ parameter sets the bit depth reduction. There are two types of bit reduction available:

- **Type I** – the signal is quantised to a 16 bit word, and the low bits thrown away. The resulting signal uses a power of 2 bits. Changing between bit depths is therefore discontinuous.
- **Type II** – quantisation is achieved via the limited precision of integer maths when dividing the signal by a factor. Since the factor can be continuously varied, this offers a smooth variation between 'bit depths'.

Furthermore, you may choose symmetric or asymmetric bit reduction via the ‘Symmetric’ parameter.

- Asymmetric bit reduction treats the whole signal range as one number to be quantised.
- Symmetric bit reduction treats positive and negative sections of the input signal differently. Negative sections are flipped positive, quantized, and flipped back.

The ‘Reduction’ parameter selects the type of bit reduction. The positive and negative sections of the input signal can have different types of reduction applied.

‘Reduction’ value	Positive signal	Negative signal
0	Type I	Type I
1	Type II	Type II
2	Type I	Type II
3	Type II	Type I
4	Type I	None
5	Type II	None
6	None	Type I
7	None	Type II

The ‘Mangling’ parameter selects the bit mangling mode.

‘Mangling’ value	Bit mangling

62 https://www.youtube.com/watch?v=Tv5c8gPYwFs

0	None
1	Bit swap variant 1
2	Bit swap variant 2
3	Bit swap variant 3
4	Bit rotation
5	Previous sample XOR variant 1
6	Previous sample XOR variant 2
7	Previous sample XOR variant 3

Crush parameters

Name	Min	Max	Default	Unit	Description
Bit reduce	0.0	100.0	0.0	%	Sets the amount of bit reduction.
Symmetric	0	1	0		Chooses symmetric or asymmetric bit reduction.
Reduction	0	7	0		Sets the bit reduction type.
Mangling	0	7	0		Sets the mangling mode.
Downsample	0.00	10.00	0.00	V	Controls the downsampling (sample rate reduction), scaled at 0.8V/octave.

Routing parameters

Name	Min	Max	Default	Unit	Description
Input	1	64	1		The first input bus to process.
Width	1	8	1		The number of busses to process, starting from the Input. For example, for a stereo signal, set this to 2.
Output	1	64	13		The first output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.
Downsample input	0	64	0		The bus to use for the downsample CV, which is added to the Downsample parameter value. (This corresponds to the Y input on the disting mk4 algorithm.)

Chaos

“CVs from the Lorenz equations”

File format guid: 'xaoc'

Specifications: None

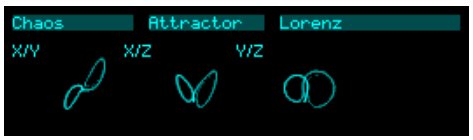
Description

This algorithm generates chaotic CVs according to a variety of so-called ‘attractors’, including the well-known Lorenz equations[63]. Each generates three CVs, named X, Y, and Z, which are available on separate outputs.

You can modify the parameters of the Lorenz equation – ‘Rho mod’ affects 'r' (aka 'ρ'), and ‘Beta mod’ affects 'b' (aka 'β'). With the values at 0V, the parameters are the classic values as studied by Lorenz (64 and 8/3 respectively).

GUI

The display shows a graphical representation of the outputs, projected onto the X/Y, X/Z, and Y/Z planes.



Chaos parameters

Name	Min	Max	Default	Unit	Description
Attractor	0	4	0		Chooses which equations to use: Lorenz, Rössler, Aizawa, Arneodo, or Thomas.
Speed range	-64	32	0		Sets the range of the Speed control, in quarter octaves (i.e. a change of 4 doubles or halves the speed).
Speed	-500	500	0		Controls the speed of the simulation.
Rho mod	-10.00	10.00	0.00	V	Modulates the Lorenz ‘r’ parameter.
Beta mod	-10.00	10.00	0.00	V	Modulates the Lorenz ‘b’ parameter.

63 https://en.wikipedia.org/wiki/Lorenz_system

Reset parameters

Name	Min	Max	Default	Unit	Description
Reset	0	1	0		When this parameter is set to 1, the system is reset to the values given by the Reset X/Y/Z parameters.
Reset X	-20.00	20.00	0.10		The X value to reset to.
Reset Y	-20.00	20.00	0.00		The Y value to reset to.
Reset Z	-20.00	20.00	0.00		The Z value to reset to.

Scale/offset parameters

Name	Min	Max	Default	Unit	Description
X scale	-20.0	20.0	10.0	V	The scale of the X output.
X offset	-20.0	20.0	0.0	V	An offset added to the X output.
Y scale	-20.0	20.0	10.0	V	The scale of the Y output.
Y offset	-20.0	20.0	0.0	V	An offset added to the Y output.
Z scale	-20.0	20.0	10.0	V	The scale of the Z output.
Z offset	-20.0	20.0	-8.0	V	An offset added to the Z output.

Routing parameters

Name	Min	Max	Default	Unit	Description
X output	0	64	15		The bus to use for the X output.
X output mode	0	1	0		The standard Add/Replace mode selector as described above.
Y output	0	64	16		The bus to use for the Y output.
Y output mode	0	1	0		The standard Add/Replace mode selector as described above.
Z output	0	64	17		The bus to use for the Z output.
Z output mode	0	1	0		The standard Add/Replace mode selector as described above.
Reset input	0	64	0		The bus to use for the reset input. A positive-going edge here resets the system to the values given by the Reset X/Y/Z parameters.

Chorus/Flange

“Chorus or Flange effect”

File format guid: 'chof'

Specifications: None

Description

This algorithm is a stereo chorus or flanger effect.

For a chorus effect, select a longish Delay with a low Depth and low Feedback. For a flanger, you want a short Delay and a high Depth, and add feedback to taste.

The Spread control sets the phase difference in the LFOs for the left and right channel. When at the default setting of 180 you get the 'stereo chorus' effect provided on certain classic synths to generate a stereo output from an essentially mono source.

Routing parameters

Name	Min	Max	Default	Unit	Description
Left/mono input	1	64	1		The left or mono input bus.
Right input	0	64	0		The right input bus, if stereo.
Left/mono output	1	64	13		The left output bus.
Right output	0	64	14		The right output bus, if stereo.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Effect parameters

Name	Min	Max	Default	Unit	Description
Delay	0.1	150.0	20.0	ms	The size of the delay lines.
Speed	0	1000	500		The LFO speed; exponential scaling from 0.01Hz to 20Hz.
Feedback	-100.0	100.0	0.0	%	The internal feedback.
Depth	0	100	15	%	The LFO depth.

Spread	0	360	180	degrees	The LFO phase difference between channels.
Mix	0	100	50	%	The wet/dry mix.
Level	-40	0	0	dB	The gain applied to the wet signal.

Chorus (Vintage)

“Chorus effect based on an 80s polysynth”

File format guid: 'junc'

Specifications: None

Description

This algorithm is a stereo chorus effect, modelled on that of the classic Juno-6 polysynth, based on measurements of the author's own unit.

Those with keen ears may recognise it as the chorus that is built into the Poly Wavetable algorithm on the disting EX.

Routing parameters

Name	Min	Max	Default	Unit	Description
Left/mono input	1	64	1		The left or mono input bus.
Right input	0	64	0		The right input bus, if stereo.
Left output	1	64	13		The left output bus.
Right output	1	64	14		The right output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Chorus parameters

Name	Min	Max	Default	Unit	Description
Mode	0	2	2		The chorus effect to apply. The options are “None”, “Mode 1”, and “Mode 2”.

Clock

“Generates clocks”

File format guid: 'clck'

Specifications:

- Outputs, 1-8: The number of clock outputs to generate.

Description

This algorithm produces and/or receives analogue clock pulses and/or MIDI clock. It is designed to synchronise the module with others, or with other devices that it communicates with via MIDI, or simply to be a source of clock pulses for other algorithms within the module.

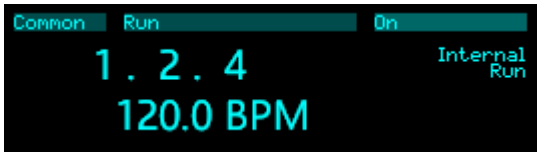
It can generate its own BPM-based clock, or sync to external clock pulses, or sync to MIDI.

It can output clock pulses, or send MIDI clock, or both.

If syncing to external clock, the algorithm expects a 24ppqn DINsync style clock. If you only have a slower clock available, use the Clock multiplier algorithm to get the clock up to the right speed before this algorithm sees it.

GUI

The display shows the current tempo, and the transport location in bars, beats, and sixteenths. It also shows the clock source, and whether the transport is running or stopped.



The GUI can also show information about the swing settings – see below.

Common parameters

Name	Min	Max	Default	Unit	Description
Source	0	2	0		The clock source. The options are “Internal”, “External”, and “MIDI”.
Tempo	30.0	240.0	120.0	BPM	The internal clock tempo.
Run	0	1	0		Starts and stops the internal clock.
Time sig numerator	1	99	4		The time signature numerator.

Time sig denominator	0	4	2		The time signature denominator: one of 1, 2, 4, 8, or 16.
Clock input	1	64	1		The external clock input.
Run/stop input	0	64	2		The run/stop input for the external clock.
Output to breakout	0	1	0		Enables MIDI output to the breakout.
Output to Select Bus	0	1	0		Enables MIDI output to the Select Bus.
Output to USB	0	1	0		Enables MIDI output to USB.
Output to internal	0	1	0		Enables internal MIDI output – that is, MIDI is sent to the other algorithms.

Swing parameters

Name	Min	Max	Default	Unit	Description
Swing type	0	7	0		The type of swing to apply. See below for more details.
Swing	-100.0	100.0	0.0	%	The amount of swing to apply.
16th note 1	2	9	2		The position of the 1st 16th note, when using tuplet swing.
16th note 2	2	9	4		The position of the 2nd 16th note, when using tuplet swing.
16th note 3	2	9	5		The position of the 3rd 16th note, when using tuplet swing.

Per-output parameters

Name	Min	Max	Default	Unit	Description
Enable	0	1	0		Enables the output.
Output	0	64	0		The output bus.
Output mode	0	1	1		The standard Add/Replace mode selector as described above.
Type	0	3	0		The output type, one of “Clock”, “Run/stop”, “Reset”, or “Trigger”.

Divisor	0	19	9		The clock division to generate. See below.
Low voltage	-10.0	10.0	0.0	V	The output voltage when the clock is low/inactive.
High voltage	-10.0	10.0	5.0	V	The output voltage when the clock is high/active.
Ratchet mode	0	2	1		The ratchet mode, one of “Off”, “Twos”, and “Twos and threes”.
Ratchet	0	7	0		The ratchet division for the clock. If mode is “Twos”, selects from 1, 2, 4, 8, 16. If mode is “Twos and threes”, selects from 1, 2, 3, 4, 6, 8, 12, 16.
Trigger length	1	100	10	ms	The length of the trigger pulse, if the type is “Reset” or “Trigger”.
ES-5 Expander	0	6	0		If set, the ES-5 expander header to use for output instead of the bus specified by the Output parameter. “1” means the ES-5 itself.
ES-5 Output	1	8	1		If using an ES-5 expander output, sets which of the 8 outputs on the expander to use.

Clock divisors

The following options are available for the ‘Divisor’ parameter:

1/64T, 1/32T, 1/32, 1/16T, 1/16, 1/8T, 1/8, 1/4T, 3/16, 1/4, 1/2T, 3/8, 1/2, 1/1T, 3/4, 1/1, 3/2, 2/1, 3/1, 4/1

Swing

Various kinds of swing are available.

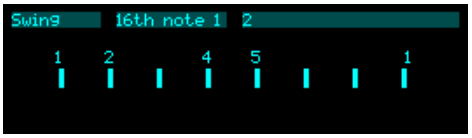
‘16ths’ – sixteenth note swing. Eighth notes are on the grid; the sixteenth notes between them can be moved backwards and forwards in time using the Swing parameter.

‘8ths’ – eighth note swing. Quarter notes are on the grid; the eighth notes between them can be moved backwards and forwards in time using the Swing parameter.

‘Pentuplet’ etc. – tuplet swing. Quarter notes are on the grid. Each quarter note is divided into a tuplet, and then the four sixteenth notes within the quarter note can be placed on the tuplet divisions. The Swing parameter interpolates between the sixteenth note grid (at 0%) and the tuplet positions (at 100%).

The algorithm’s GUI shows the tuplet grid settings if active:


Swing
Swing type
Septuplet
1
2
4
5
1


Swing
16th note 1
2
1
2
4
5
1

Clock Divider

“Divides clocks”

File format guid: 'clkd'

Specifications:

- Channels, 1-8: The number of clock channels to process.

Description

This algorithm is a simple clock divider, outputting slower clocks from a faster one. The channels can use completely independent clocks, or share clocks.

There is a shared reset input and a per-channel reset input. If either is active, the channel is reset.

Common parameters

Name	Min	Max	Default	Unit	Description
Reset input	0	64	0		The shared reset input bus.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Type	0	2	0		The divisor type. The options are “Free”, “Metrical (2)”, and “Metrical (2,3)”.
Divisor	1	32	2		The divisor, if the type is “Free”. The divisor value is simply the parameter value.
Divisor	0	5	1		The divisor, if the type is “Metrical (2)”. Chooses between 1, 2, 4, 8, 16, & 32.
Divisor	0	9	1		The divisor, if the type is “Metrical (2,3)”. Chooses between 1, 2, 3, 4, 6, 8, 12, 16, 24, & 32.
Enable	0	1	0		Enables the channel. A disabled channel outputs nothing.
Input	1	64	1		The clock input bus to divide.
Reset input	0	64	0		The reset input bus for this channel.
Output	1	64	15		The clock output bus.
Output mode	0	1	1		The standard Add/Replace mode selector as described above.

ES-5 Expander	0	6	0		If set, the ES-5 expander header to use for output instead of the bus specified by the Output parameter. “1” means the ES-5 itself.
ES-5 Output	1	8	1		If using an ES-5 expander output, sets which of the 8 outputs on the expander to use.

Clock Multiplier

“Multiplies clocks”

File format guid: 'clkm'

Specifications: None

Description

This algorithm is a simple clock multiplier, generating a faster clock from a slower one.

The output clock rate is updated on every clock received, except that the clock duration is limited to at most double each time. This is mainly to prevent the output clock rate changing when the input clock is paused and restarted, which would otherwise be interpreted as a really long clock pulse.

Parameters

Name	Min	Max	Default	Unit	Description
Clock input	1	64	1		The clock input bus.
Clock output	1	64	1		The clock output bus.
Output mode	0	1	1		The standard Add/Replace mode selector as described above.
Multiplier	1	24	2		The clock multiplier.
Low voltage	-10.0	10.0	0.0	V	The output voltage when the clock is low/inactive.
High voltage	-10.0	10.0	5.0	V	The output voltage when the clock is high/active.
ES-5 Expander	0	6	0		If set, the ES-5 expander header to use for output instead of the bus specified by the Clock output parameter. “1” means the ES-5 itself.
ES-5 Output	1	8	1		If using an ES-5 expander output, sets which of the 8 outputs on the expander to use.

Compressor

“A simple compressor”

File format guid: 'comp'

Specifications:

- Channels, 1-12: The number of bus channels to process.

Description

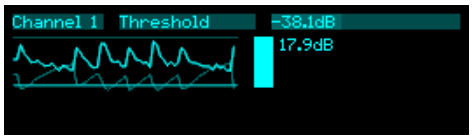
This algorithm is a multi-channel compressor. Each channel is fully independent – this being a multi-channel algorithm is simply a convenience so that if you, say, want a compressor on all 12 module inputs, you don’t need to add 12 copies of the algorithm to do so.

A sidechain input is available, to compress one signal according to the envelope of another.

The algorithm always works as an insert effect i.e. the output replaces the input.

GUI

The display shows a graph of the signal level (travelling from right to left). Superimposed on this is the gain reduction amount. To the right of the graph, the bar represents the gain reduction currently being applied.



Per-channel parameters

Name	Min	Max	Default	Unit	Description
Enable	0	1	0		Enables the channel.
Threshold	-60.0	0.0	-12.0	dB	The threshold at which gain reduction starts.
Ratio	0	1023	100		The compression ratio. Exponential scaling from 1:1 up to 1000:1.
Attack	0	1023	100		The attack time of the compressor. Exponential scaling from 0.1ms to 1000ms.
Release	0	1023	100		The release time of the compressor. Exponential scaling from 1ms to 3000ms.

Auto make-up	0	1	0		Enables auto make-up gain, raising the gain automatically according to the threshold and ratio.
Make-up gain					The make-up gain, if “Auto make-up” is off.
Lookahead	0.0	10.0	1.0	ms	The lookahead time. Increase this to prevent the compressor from missing sharp attack transients. Note that the audio is delayed by the amount of the lookahead time i.e. this adds latency.
Sidechain mix	-40.0	12.0	-40.0	dB	Sets the amount of the sidechain signal mixed into the output.
Sidechain solo	0	1	0		If enabled, the sidechain signal is passed through to the output and the main signal muted.

Per-channel routing parameters

Name	**Min**	**Max**	**Default**	**Unit**	**Description**
Left/mono input	1	64			The left or mono input bus.
Right input	0	64	0		The right input bus, if stereo.
Sidechain input	0	64	0		The sidechain input bus.
Reduction output	0	64	0		The bus to use for the gain reduction output, scaled as 1V per 6dB of gain reduction.
Reduction mode	0	1	0		The standard Add/Replace mode selector as described above, for the gain reduction output.

Convolver

“Computes audio convolutions”

File format guid: 'conv'

Specifications:

- Max impulse, 1-10 seconds: The maximum length of the impulse sample.

Description

This algorithm performs real-time convolution[64] of the input signal and another signal (often called an 'impulse response'), which is loaded from the MicroSD card.

This is often used for realistic reverb effects, since the required impulse responses can be sampled from a physical location, but the technique is much more general and can be a very creative way to mangle and combine sounds.

Convolution is notoriously CPU-intensive, and the disting NT is a resource-constrained embedded system – you should not expect to achieve the same results as you might in the latest new fangled convolution reverb plug-in in your DAW, with orders of magnitude more CPU power and RAM. However, this algorithm remains a useful tool within the context of a modular synth.

The algorithm is based on the disting EX algorithm of the same name. There is an in-depth video for that algorithm here[65].

Impulse length and the factors that affect it

When choosing the parameters for this algorithm there are various trade-offs to be made, all of which affect the maximum length of impulse response that you can use. (A longer impulse response is usually desirable, especially if you're using this for reverb.) These factors are:

Allocated memory: perhaps most obviously, the more memory you allocate to the algorithm via its specifications, the longer the impulse it can handle.

Mono vs stereo: if the output is stereo, the algorithm has to do almost twice the work, so the impulse time is more or less halved. Using the routing parameters, you can select mono, mono-to-stereo, or stereo operation. The last two count as stereo in terms of processing time. Note that the stereo option passes the dry signal in stereo but sums the two input channels to mono before convolution; it does not do true stereo convolution, which would double the CPU cost again.

Latency: the algorithm works on blocks of audio at a time. It is more efficient to work on larger blocks, but conversely using larger blocks means a longer wait until you get the results i.e. higher latency. The Latency parameter offers you the choice between lower latency with smaller impulse time, or higher latency with longer impulse time. Note that in either case, the dry signal is still synced

64 https://en.wikipedia.org/wiki/Convolution

65 https://www.youtube.com/watch?v=tDFZA9b2A0Y

up with the convolution result. So for example you could use this as a master bus insert effect to put a nice reverb on your whole mix and probably get away with a long latency.

Sample rate: the disting NT's algorithms all run at 48kHz. However, by downsampling to a lower sample rate internally, a given time's worth of audio is cheaper to process and a longer impulse time can be achieved, at the expense of the bandwidth of the processed audio. The 'Sample rate' parameter offers several choices, from a high-quality 48kHz to a decidedly murky 6kHz. Note that the dry signal is always 48kHz.

Impulse responses on the MicroSD card

Any sample file on the card can be loaded as an impulse response. For convenience, when the algorithm is loaded it defaults to a folder called 'impulses', if such a folder exists. Any kind of WAV file is supported, but stereo 32 bit float is preferred. For fastest loading, choose a sample rate for the file equal to the sample rate at which you want to run the algorithm, but other sample rates will be converted.

For the avoidance of doubt, the 'impulses' folder is considered a folder of samples, and as such should be within the top level ‘samples’ folder.

There are plenty of free examples online, for example these[66] by EchoThief, or these[67] rather lovely Bricasti M7 impulses. You may find that you have a plug-in in your DAW which contains a library of impulse responses (e.g. Live's Hybrid Reverb) that you could copy to the MicroSD card.

GUI

The display shows a graphical representation of the impulse sample on the left of the screen. On the right side, it shows the mono/stereo processing mode, the maximum impulse time (as a result of the current parameter values), the actual length of the sample file, and the current overall CPU usage.

Convolver parameters

Name	Min	Max	Default	Unit	Description
Folder	1				Chooses the folder from which to load the impulse response sample.
Sample	1				Chooses the impulse response sample within the folder.

66 https://www.echothief.com/downloads/

67 https://web.archive.org/web/20190201211631/http://www.samplicity.com/bricasti-m7-impulse-responses/

Latency	0	3	2		Sets the processing latency (5ms, 11ms, 22ms, or 43ms).
Sample rate	0	3	0		Sets the processing sample rate (48kHz, 24kHz, 12kHz, or 6kHz).
Partitions	1				Allows you to fine tune the maximum impulse length, and consequently the CPU usage.

Mix parameters

Name	Min	Max	Default	Unit	Description
Dry gain	-40	12	0	dB	The output level of the dry signal.
Convolution gain	-40	12	0	dB	The output level of the convolution result.

Routing parameters

Name	Min	Max	Default	Unit	Description
Left/mono input	1	64	1		The left or mono audio input bus.
Right input	0	64	0		The right audio input bus.
Left output	1	64	13		The left audio output bus.
Right output	0	64	14		The right audio output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Crossfader

“Crossfades signals”

File format guid: 'xfad'

Specifications: None

Description

This algorithm crossfades between two or more sets of signals (from mono up to sets of eight channels).

When crossfading between more than two sets of signals, it behaves like a ‘scanning’ or ‘morphing’ mixer[68].

Three different crossfade curves are available:

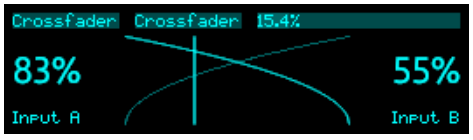
- **Equal gain** – Appropriate for crossfading phase-coherent material.
- **Equal power** – Appropriate for crossfading non-phase-coherent material.
- **Transition** – DJ-style crossfade where both sources are at full gain at the 50% position.

Inputs A to L, and the output, use a contiguous range of busses, starting with the one set via the parameters, and with the channel count set by the ‘Width’ parameter. For example, to crossfade stereo signals, set the Width to 2.

The crossfade can be controlled by both a parameter and a CV, which are simply summed if both are in use.

GUI

The display shows a graphical representation of the crossfade curves, and the amount of the two signals that are combined in the output mix.



Crossfader parameters

Name	Min	Max	Default	Unit	Description
Crossfader	0.0	100.0	50.0	%	The crossfade position, from 0% (input A only) to 100% (input B only).
Curve	0	2	1		The crossfade curve, as above.

68 Compare with, for example, the Doepfer module pair A-144/A-135-1.

Routing parameters

Name	Min	Max	Default	Unit	Description
Input A	1	64	1		The first bus for input A.
Input B	1	64	2		The first bus for input B.
Output	1	64	13		The first bus for the output.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.
Width	1	8	1		The number of busses to process. For example for stereo signals, set this to 2.
Crossfade input	0	64	0		The CV input to drive the crossfade, scaled so that 5V covers the range 0-100%.
Number of inputs	2	12	2		The number of inputs to use.
Input C	1	64	3		The first bus for input C.
Input D	1	64	4		The first bus for input D.
Input E	1	64	5		The first bus for input E.
Input F	1	64	6		The first bus for input F.
Input G	1	64	7		The first bus for input G.
Input H	1	64	8		The first bus for input H.
Input I	1	64	9		The first bus for input I.
Input J	1	64	10		The first bus for input J.
Input K	1	64	11		The first bus for input K.
Input L	1	64	12		The first bus for input L.

Debouncer

“A switch debouncer”

File format guid: 'debo'

Specifications:

- Channels, 1-8: The number of bus channels to process.

Description

This algorithm implements a simple switch debouncer (see e.g. this Wikipedia page[69]).

It was added primarily so that simple passive guitar footswitches can be attached to the module’s inputs (with a suitable voltage source) and used with, for example, the Looper algorithm.

Globals parameters

Name	Min	Max	Default	Unit	Description
Time	1	100	10	ms	The hold-off time. Once the output has changed, it is not allowed to change again for the this duration.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Input	0	64	13		The input (and output) bus.
Enable	0	1	0		Enables debouncing on this channel.
Output	0	64	0		The output bus. If set to ‘None’, the input bus is used as output, and the mode is always ‘Replace’.
Output mode	0	1	1		The standard Add/Replace mode selector as described above.
Output Volts	-10	10	5	V	The output voltage when the input is ‘high’, in Volts.
Output semitones	-60	60	0	ST	The output voltage when the input is ‘high’, in 12-TET semitones (one twelfth of a Volt).

69 https://en.wikipedia.org/wiki/Switch#Contact_bounce

Delay (Mono)

“A simple mono delay effect”

File format guid: 'delm'

Specifications:

- Max delay time, 1-30 seconds: The maximum delay time.

Description

This algorithm is a simple delay effect.

The delay time can be set manually, via a clock pulse, or via MIDI clock.

It can also be modulated via its ‘V/oct input’, which halves the delay time for every 1V rise in CV (conversely, doubles it for every 1V fall in CV).

The ‘Fractional’ parameter allows for delay times that are not integer multiples of the sample rate. This slightly increases CPU load but will sound significantly better if the delay time is being modulated, for example for chorus or flange type sounds. Conversely, turning it off makes for an extremely clean digital delay, with no colouration in the repeats.

Delay parameters

Name	Min	Max	Default	Unit	Description
Mix	0	100	100	%	The wet/dry mix.
Level	-40	0	0	dB	The gain applied to the delay/wet signal.
Time	1	32767	250	ms	The delay time.
Feedback	0	100	50	%	The delay feedback.
Delay multiplier	0	23	15		Sets a multiplier to apply to the delay time. Affects both the ‘Time’ parameter and the delay time set by the clock.
Time change	0	1	0		Sets how changes of delay time will be handled. The options are ‘Slew’ (the delay time is smoothly interpolated, sounding a bit like a tape being played slower or faster) and ‘Crossfade’ (the effect rapidly crossfades from one time setting to the other).
Fractional	0	1	0		Enables fractional delay times. See above.

Block DC	0	1	1		Enables a DC blocker at the delay input. Recommended for audio use; disable when processing CVs.

Sync parameters

Name	Min	Max	Default	Unit	Description
Clock source	0	2	1		Chooses the clock source: ‘None’, ‘Clock input’, or ‘MIDI clock’.
Clock input	0	64	0		The input bus to use for the clock, if the source is ‘Clock input’.
MIDI divisor	0	19	9		The MIDI divisor, if the clock source is ‘MIDI clock’.
MIDI numerator	1	16	1		The MIDI numerator, if the clock source is ‘MIDI clock’.

Routing parameters

Name	Min	Max	Default	Unit	Description
Input	1	64	1		The audio input bus.
Output	1	64	13		The audio output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.
V/oct input	0	64	0		The input bus to use for 1V/octave modulation of the delay time.

Delay (Stereo)

“A simple stereo delay effect”

File format guid: 'dels'

Specifications:

- Max delay time, 1-30 seconds: The maximum delay time.

Description

This algorithm is a simple stereo in/stereo out delay effect.

The delay time can be set manually, via a clock pulse, or via MIDI clock.

It can also be modulated via its ‘V/oct input’, which halves the delay time for every 1V rise in CV (conversely, doubles it for every 1V fall in CV).

The ‘Fractional’ parameter allows for delay times that are not integer multiples of the sample rate. This slightly increases CPU load but will sound significantly better if the delay time is being modulated, for example for chorus or flange type sounds. Conversely, turning it off makes for an extremely clean digital delay, with no colouration in the repeats.

Delay parameters

Name	Min	Max	Default	Unit	Description
Mode	0	2	0		The delay mode. The options are “Stereo”, “Stereo ping-pong” (each echo swaps left and right), and “Ping-pong” (the input to the delay is summed to mono, panned, and then swaps left and right on each echo).
Mix	0	100	100	%	The wet/dry mix.
Level	-40	0	0	dB	The gain applied to the delay/wet signal.
Time	1	32767	250	ms	The delay time.
Feedback	0	100	50	%	The delay feedback.
Initial pan	-100	100	-100	%	If the mode is “Ping-pong”, sets the pan position of the first delay.
Delay multiplier	0	23	15		Sets a multiplier to apply to the delay time. Affects both the ‘Time’ parameter and the delay time set by the clock.

Time change	0	1	0		Sets how changes of delay time will be handled. The options are ‘Slew’ (the delay time is smoothly interpolated, sounding a bit like a tape being played slower or faster) and ‘Crossfade’ (the effect rapidly crossfades from one time setting to the other).
Fractional	0	1	0		Enables fractional delay times. See above.
Block DC	0	1	1		Enables a DC blocker at the delay input. Recommended for audio use; disable when processing CVs.

Sync parameters

Name	Min	Max	Default	Unit	Description
Clock source	0	2	1		Chooses the clock source: ‘None’, ‘Clock input’, or ‘MIDI clock’.
Clock input	0	64	0		The input bus to use for the clock, if the source is ‘Clock input’.
MIDI divisor	0	19	9		The MIDI divisor, if the clock source is ‘MIDI clock’.
MIDI numerator	1	16	1		The MIDI numerator, if the clock source is ‘MIDI clock’.

Routing parameters

Name	Min	Max	Default	Unit	Description
Left input	1	64	1		The left input bus.
Right input	1	64	2		The right input bus.
Left output	1	64	13		The left output bus.
Right output	1	64	14		The right output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.
V/oct input	0	64	0		The input bus to use for 1V/octave modulation of the delay time.

Delay (Tape)

“A tape delay effect”

File format guid: 'delt'

Specifications:

- Max tape length, 1-30 seconds: The maximum length of the ‘tape’.
- Stereo: Whether the algorithm is mono or stereo.

Description

This algorithm is based on the disting mk4 algorithm “D-2 Tape Delay”, which is itself a simplified version of the “Augustus Loop” effect. It is a delay/echo effect which simulates a variable speed tape loop echo device.

The ‘Tape length’ parameter sets the range of delay times available. Note though that changing this may introduce clicks and pops. The primary means of changing the delay time is the “Tape speed” parameter and/or the Speed CV input.

The tape speed can be set in the range from half speed (0.5x) to double speed (2x). If using the CV input, the scaling is 8V/octave i.e. +8V gives double speed and -4V gives half speed.

Three options are available for how the wet/dry mix is controlled:

- **With feedback** - the amount of delay in the mix rises in direct proportion to the amount of feedback.
- **Crossfade** - the ‘Mix’ parameter is a crossfader.
- **Add delay** - the dry level is constant, and the ‘Mix’ parameter adds in the delay signal.

Delay parameters

Name	Min	Max	Default	Unit	Description
Mix mode	0	2	0		Sets how the delay and dry signals are mixed. See above.
Mix	0	100	100	%	Controls the wet/dry mix, unless the ‘Mix mode’ is ‘With feedback’.
Feedback	0	110	50	%	The delay feedback.
Tape length	1		250	ms	Sets the length of the tape i.e. the delay time at 1x speed.
Tape speed	-500	1000	0		Sets the tape speed (in conjunction with the CV input).

Routing parameters

Name	Min	Max	Default	Unit	Description
Left/mono input	1	64	1		The left input bus.
Right input	1	64	2		The right input bus.
Left/mono output	1	64	13		The left output bus.
Right output	1	64	14		The right output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.
Speed input	0	64	0		The tape speed CV input bus.

Delayed Function

“Generates functions after a delay”

File format guid: 'delf'

Specifications:

- Channels, 1-8: The number of functions to generate.

Description

This algorithm generates a variety of one-shot functions after a delay, when triggered. The logic of each channel is as follows:

- When triggered, it immediately jumps to its idle voltage, if not already there.
- It stays at that voltage for the duration set by the Delay parameter.
- It then executes the function, for the duration set by the Time parameter, if relevant (for example, the Step function has no duration).
- Depending on the function, it then remains at the active or idle voltage until triggered again.

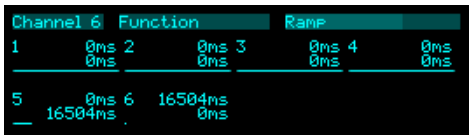
The available functions are:

Pulse	A simple rectangular pulse of the specified duration.
Step	A step – the output goes from idle to active after the delay, and remains at the active level until retriggered.
Ramp	A rising ramp of the specified duration. Stays at the active level after the ramp is complete.
Sawtooth	A rising ramp of the specified duration, which jumps back to the idle level when complete.

GUI

The display shows, for each channel, from top to bottom:

- The remaining time of the delay, once triggered.
- The remaining time of the function.
- The output function as a bar graph.



Common parameters

Name	Min	Max	Default	Unit	Description
Disable all	0	1	0		If set, disables all channels.
Detection	0	1	0		Sets the method of trigger detection, either ‘Efficient’ or ‘Accurate’.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Enable	0	1	0		Enables the channel.
Function	0	3	0		Chooses the function from the table above.
Delay	0	1000	0		Sets the delay time.
Delay multiplier	0	6	0		Sets a multiplier for the delay time, one of x1, x2, x5, x10, x20, x50, or x100.
Time	1	1000	100		Sets the function duration.
Time multiplier	0	6	0		Sets a multiplier for the function duration, one of x1, x2, x5, x10, x20, x50, or x100.
Active voltage	-10.00	10.00	5.00	V	Sets the channel’s active voltage.
Idle voltage	-10.00	10.00	0.00	V	Sets the channel’s idle voltage.

Per-channel routing parameters

Name	Min	Max	Default	Unit	Description
Trigger input	0	64	1		The bus to use for the trigger input.
Output	0	64	15		The bus to use for the channel output.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.
MIDI channel	0	16	0		The MIDI channel on which notes can trigger the function generator.
I2C channel	0	255	0		The I2C channel on which notes can trigger the function generator.

Note number	-1	127	-1		The MIDI or I2C note number that will trigger the function generator, or ‘-1’ for ‘Any’.

Display Awakener

“Wakes the display from sleep”

File format guid: 'wake'

Specifications: None

Description

This simple algorithm generates no sound – it simply exists to give you a way to wake the display from sleep without physically touching the module.

It has 12 parameters, which can be mapped to whatever you’d like to wake the display. For example, you might map one to a MIDI CC so you can wake the display from a MIDI controller. Or, you might map one to an input CV so that the module wakes whenever it gets a clock pulse from your sequencer.

Note that you can treat audio as CV for this purpose; if you map an audio input to one of the wake parameters, you can have a setup where the module display wakes whenever you play audio into the module – great for pedalboard use.

Parameters

Name	Min	Max	Default	Unit	Description
Wake 1-12	0	1	0		Wakes the display if the parameter value is ‘1’.

DJ Filter

“Low/High pass filter”

File format guid: 'djfi'

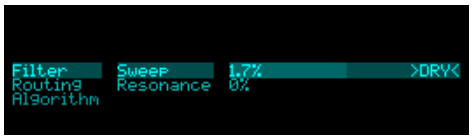
Specifications: None

Description

This algorithm is an implementation of the disting mk4 algorithm of the same name. You may like to review the video about that algorithm, which is here[70].

The algorithm is a filter which sweeps from lowpass to highpass on a single control. The Sweep parameter sets the filter cutoff. At the central, zero position no filter effect is applied. If the sweep goes negative, a lowpass filter is applied, with a cutoff frequency that sweeps down the more you go. If the sweep goes positive, a highpass filter is applied, with a cutoff frequency that sweeps up the more you go.

There is a deadzone around zero where the filter has no effect, indicated by “>DRY<” next to the parameter value.



Filter parameters

Name	Min	Max	Default	Unit	Description
Sweep	-100.0	100.0	0.0	%	Controls the filter sweep.
Resonance	0	100	0	%	Sets the filter resonance/Q.

Routing parameters

Name	Min	Max	Default	Unit	Description
Input	1	64	1		The first input bus to process.
Width	1	8	1		The number of busses to process, starting from the Input. For example, for a stereo signal, set this to 2.
Output	1	64	13		The first output bus.

70 https://www.youtube.com/watch?v=fII_lscKbBg

Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Dream Machine

“Just intonation drone synth”

File format guid: 'drea'

Specifications: None

Description

This algorithm is an implementation of the original Dream Machine algorithm on the disting EX.

It is designed to generate drones, allowing the user to explore non-traditional harmonies based on prime ratios. It was inspired by the theories of composer La Monte Young[71]. An interesting read is the pdf “Notes on The Theatre of Eternal Music” available in a number of places online e.g. here[72].

The output is a combination of five sounds – the fundamental and four harmonies. The prime ratios that define the frequency relationships are controlled by parameters.

The algorithm uses wavetable synthesis to generate the tones. Additionally the fundamental may instead be a pure sine, triangle, or square wave.

Each tone has a simple attack/release envelope, controlled by its own gate parameter.

By default, little is mapped to the CV inputs, and it is perfectly possible to drive the algorithm entirely by hand. You may however like to map inputs as FM inputs, or to control the gates.

Setting the fundamental

The fundamental is the fixed tone (usually the bass note) that everything else revolves around. You may like to set it to a concert pitch (it defaults to concert B ♭) or some other frequency (La Monte Young sometimes chose the mains frequency – 60Hz in the USA – or you could tune it to the resonant frequency of whatever environment you find yourself in).

Note that the Hz value shown in the display also takes into account the octave parameter.

Setting the primes

Four parameters let you choose the set of prime numbers that will make up the allowable values for the frequency ratios. La Monte Young famously chose the primes 2, 3, 7, & 31, and moreover specifically avoided the prime 5, thereby excluding major thirds from his tunings.

If you want less than four primes, set the unwanted parameters to '1'.

71 https://en.wikipedia.org/wiki/La_Monte_Young

72 https://www.melafoundation.org/theatre.pdf

Setting the frequency ratios

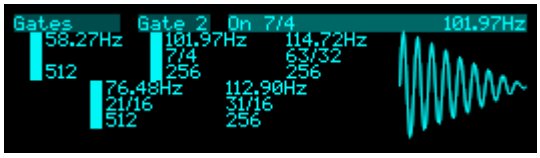
To set the ratios of tones 1-4 relative to the fundamental, set the parameters for the denominator and the four numerators.

When setting the numerators, the pitch of the tone is shown, as well as the ratio reduced to its lowest form. For example 48/32 reduces to 3/2, the familiar form of the perfect fifth in just intonation.

GUI

The display shows the current waveform within the wavetable on the right of the screen.

The rest of the screen shows the frequency, ratio, and gate level of the five tones (from left to right).



Wavetable parameters

Name	Min	Max	Default	Unit	Description
Wavetable					Chooses the wavetable for the oscillators.
Wave offset	-100.0	100.0	0.0	%	Sets the position within the wavetable.
Wave input	0	64	0		Sets an input bus to modulate the wave offset.
Waveform 0	0	3	0		Chooses the waveform for the fundamental. The options are “Wavetable”, “Sine”, “Triangle”, and “Square”.

Pitches parameters

Name	Min	Max	Default	Unit	Description
Prime 1	2	32749	2		Sets one of the four primes that may be multiplied to create the denominator and numerators of the frequency ratios.
Prime 2	1	32749	3		Sets the second prime.
Prime 3	1	32749	7		Sets the third prime.
Prime 4	1	32749	11		Sets the fourth prime.
Fundamental	0.001	32.767	29.135	Hz	Sets the fundamental frequency.

Octave	-8	8	1		Sets an octave shift for the fundamental.
Transpose	-60	60	0	ST	Adjusts the tuning of all frequencies in (12-TET) semitone steps.
Denominator	1	256	32		Sets the denominator of the frequency ratios.
Numerator 1	1	1024	42		Sets the numerator of the frequency ratio of tone 1.
Numerator 2	1	1024	56		Sets the numerator of the frequency ratio of tone 2.
Numerator 3	1	1024	62		Sets the numerator of the frequency ratio of tone 3.
Numerator 4	1	1024	63		Sets the numerator of the frequency ratio of tone 4.

Gates parameters

Name	Min	Max	Default	Unit	Description
Gate 0	0	1	0		Gate for tone 0.
Gate 1	0	1	0		Gate for tone 1.
Gate 2	0	1	0		Gate for tone 2.
Gate 3	0	1	0		Gate for tone 3.
Gate 4	0	1	0		Gate for tone 4.

Mix parameters

Name	Min	Max	Default	Unit	Description
Gain 0	-40	6	0	dB	Gain for the fundamental. “-40” is treated as –∞dB.
Gain 1	-40	6	0	dB	Gain for tone 1.
Gain 2	-40	6	0	dB	Gain for tone 2.
Gain 3	-40	6	0	dB	Gain for tone 3.
Gain 4	-40	6	0	dB	Gain for tone 4.
Pan 1	-100	100	0	%	Stereo pan position for tone 1.
Pan 2	-100	100	0	%	Stereo pan position for tone 2.

Pan 3	-100	100	0	%	Stereo pan position for tone 3.
Pan 4	-100	100	0	%	Stereo pan position for tone 4.

Other parameters

Name	Min	Max	Default	Unit	Description
Attack time	0	127	0		Attack time for the envelopes.
Decay time	0	127	0		Decay time for the envelopes.
FM input 1	0	64	0		Which input bus to use to frequency modulate (FM) tone 1. The input is scaled according to the FM Range parameter.
FM input 2	0	64	0		As above for tone 2.
FM input 3	0	64	0		As above for tone 3.
FM input 4	0	64	0		As above for tone 4.
FM Range	0	3	0		Sets the scaling for the FM inputs. The options are 1Hz/V, 10Hz/V, 100Hz/V or 1kHz/V.
Frequency slew	0	127	0		Sets a slew rate for any change in voice frequency.
Crossfade time	0	127	0		Sets a crossfade time for any change in voice frequency.

Routing parameters

Name	Min	Max	Default	Unit	Description
Left output	1	64	13		The left output bus.
Right output	1	64	14		The right output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

EQ Parametric

“Multi-band parametric EQ”

File format guid: 'eqpa'

Specifications:

- Channels, 1-12: The number of bus channels to process.
- Bands: 1-4: The number of EQ bands per channel.

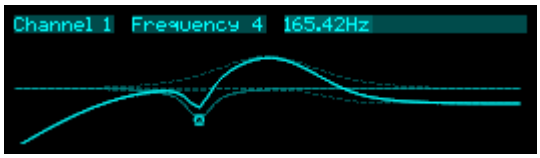
Description

This algorithm is a multi-channel, multi-band, parametric EQ.

Each channel can process a number of busses - you might typically have a channel processing one or two busses for a mono or stereo signal, but you could, say, process all 12 of the inputs with the same EQ settings by using a single channel with a width of 12.

GUI

The display shows a graphical representation of the EQ curve for the current channel. The frequency and gain of the current band within the channel is indicated with a small box.



Per-channel parameters

Name	Min	Max	Default	Unit	Description
Input	0	64	1		The first input bus to process.
Width	1	12	1		The number of busses to process, starting from the Input. For example, for a stereo signal, set this to 2.
Output	0	64	0		The first output bus. If set to ‘None’, the input bus is used as output, and the mode is always ‘Replace’.
Output mode	0	1	1		The standard Add/Replace mode selector as described above.

Per-channel per-band parameters

Name	Min	Max	Default	Unit	Description

Enable	0	1	1		Enables the EQ band.
Type	0	6	6		Chooses the type of the EQ band. The options are “Low pass (1st)”, “High pass (1st)”, “Low pass (2nd)”, “High pass (2nd)”, “Low shelf”, “High shelf”, and “Peak”.
Frequency	0	16383	8192		Sets the EQ band centre frequency.
Q	0	127	48		Sets the EQ band ‘Q’ or resonance. Note that some EQ types do not use this parameter.
Gain	-12.0	12.0	0.0	dB	Sets the EQ band gain. Note that some EQ types do not use this parameter.

Envelope (AR/AD)

“A simple attack/release envelope”

File format guid: 'env2'

Specifications:

- Channels, 1-8: The number of envelopes to generate.

Description

This algorithm generates simple attack/release or attack/decay envelopes. It is based on the disting mk4 algorithm “E-1 AR Envelope”.

In the disting mk4 version the envelope times were always set together from a single knob; this version adds the option of setting them independently.

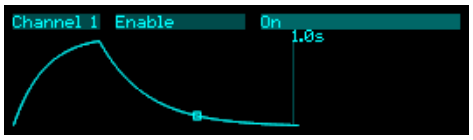
The envelope times and shapes are set globally; each output channel can however be independently scaled and offset.

In “Attack/release” mode, the envelope will rise to full level and stay there as long as the input is high. In “Attack/decay” mode, the envelope will execute one full attack/decay cycle in response to a trigger input. In “Looped AD” mode, the envelope will continually execute attack/decay cycles as long as the trigger input is high.

In “Macro (asymmetric)” mode, the “Joint time” parameter sweeps from short A & D, through short A & long D, through long A & D, through long A & short D, and finally back to short A & D. In “Macro (symmetric)” mode, the “Joint time” parameter sets the A & D times to the same value, from very short times (about 10ms) to very long times (about 8s). Note that the actual times are heavily dependent on the shape parameters.

GUI

The display shows the envelope shape. If the visible parameter page is one of the Channel pages, it also shows the current envelope level.



Parameters

Name	Min	Max	Default	Unit	Description
Trigger mode	0	2	0		The trigger mode. The options are: ● Attack/release ● Attack/decay ● Looped AD
Time mode	0	2	0		The envelope time mode. The options are: ● Macro (asymmetric) ● Macro (symmetric) ● Independent
Joint time	0	1023	0		Sets the attack and release times if the ‘Time mode’ is one of the two ‘Macro’ options.
Attack time	0	1023	64		Sets the attack time if the ‘Time mode’ is ‘Independent’.
Release time	0	1023	64		Sets the release time if the ‘Time mode’ is ‘Independent’.
Attack shape	0	128	104		Sets the attack shape, from an exaggerated exponential curve at 0 to an almost linear shape at 128.
Release shape	0	128	40		Sets the release shape, from an exaggerated exponential curve at 0 to an almost linear shape at 128.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Enable	0	1	0		Enables the channel.
Trigger input	0	64	1		Sets the trigger input bus.
Output	0	64	15		Sets the envelope output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.
Amplitude	-10.00	10.00	8.00	V	Sets the amplitude of the envelope. Note that negative values give an inverted envelope.
Offset	-10.00	10.00	0.00	V	Sets a constant voltage offset for the envelope.
MIDI channel	0	16	0		The MIDI channel on which notes can trigger the envelope.

I2C channel	0	255	1		The I2C channel on which notes can trigger the envelope.
Note number	-1	127	-1		The MIDI or I2C note number that will trigger the envelope, or ‘-1’ for ‘Any’.
Velocity depth	0	100	0	%	The amount by which the trigger velocity affects the envelope scale (applies only to MIDI and I2C).

Envelope (DAHDSR)

“A versatile envelope generator”

File format guid: 'envq'

Specifications:

- Channels, 1-8: The number of envelopes to generate.
- Shapes, 1-8: The number of envelope shapes shared between the envelopes.

Description

This algorithm is based on the disting EX “Quad Envelope” algorithm. You may like to review the video on that algorithm, which is here[73].

It is a complex (DAHDSR) envelope generator. Various trigger modes include gated, triggered (one-shot) and looping. The envelopes can be synced to clocks (analogue pulses, or MIDI).

'Shapes' and 'Envelopes'

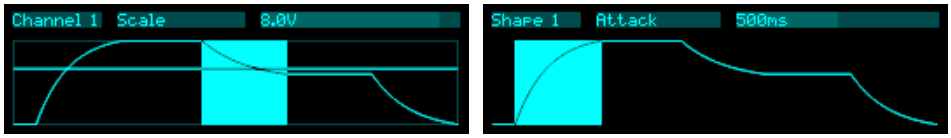
This algorithm separates the concepts of an envelope shape (its attack etc. times, sustain level) and the envelope generator itself.

You can share shapes between the generators. For example, in the case of a four voice polyphonic synth, you might want four envelopes, all using the same shape. By sharing the shape between the generators, you only need to edit e.g. one attack time to change the attack time for all four voices.

At the other extreme, you might want to have four different shapes, but share a common trigger – for example, to modulate four different parts of a monophonic patch.

GUI

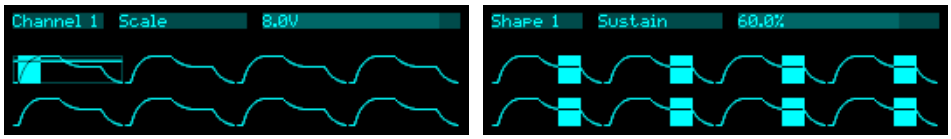
The display shows a graphical representation of the envelope(s).



If the current parameter is one of the shape parameters, the relevant part of the envelope is highlighted – for example, the attack section in the image above. Otherwise, the current envelope is shown, with a horizontal line overlaid to indicate the current output level, and the currently active section is highlighted – the decay section, in the image above.

73 https://www.youtube.com/watch?v=PIRCsGBi8Es

When there are many envelopes, the display is subdivided into regions for each one. When showing shape parameters, the relevant section will be highlighted in all envelopes that share that shape.



Per-channel parameters

Name	Min	Max	Default	Unit	Description
Enable	0	1	0		Enables the envelope.
Shape	1	8	1		Chooses the shape used by the envelope generator.
Trigger mode	0	5	0		See ‘Trigger modes’, below.
Clock mode	0	3	0		See ‘Clock modes’, below.
Clock source	0	1	0		The clock source, one of ‘Input’ or ‘MIDI’.
MIDI divisor	0	19	4		The MIDI clock divisor, if the source is MIDI.
Scale	-10.0	10.0	8.0	V	The magnitude of the output envelope.
Offset	-10.0	10.0	0.0	V	The offset (starting/ending value) of the envelope.
Vel -> scale	0	100	100	%	The amount by which the trigger velocity affects the envelope scale.
Vel -> attack	-100	100	0	%	The amount by which the trigger velocity affects the envelope attack time.

Per-channel routing parameters

Name	Min	Max	Default	Unit	Description
Trigger input	0	64			The bus to use for the envelope’s trigger input.
Clock input	0	64	0		The bus to use for the envelope’s clock input.
Output	0	64	15		The bus to use for the envelope’s output.
Output mode	0	1	0		The standard Add/Replace mode selector as described above, for the envelope’s output.

End of cycle output	0	64	0		The bus to use for the envelope’s end of cycle trigger output. Outputs a 5V trigger when the envelope becomes idle.
End of cycle mode	0	1	0		The standard Add/Replace mode selector as described above, for the envelope’s end of cycle output.
Active output	0	64	0		The bus to use for the envelope’s active output. Outputs 5V while the envelope is not idle.
Active mode	0	1	0		The standard Add/Replace mode selector as described above, for the envelope’s active output.
MIDI channel	0	16	0		The MIDI channel on which notes can trigger the envelope.
I2C channel	0	255	1		The I2C channel on which notes can trigger the envelope.
Note number	-1	127	-1		The MIDI or I2C note number that will trigger the envelope, or ‘-1’ for ‘Any’.

Per-shape parameters

Name	**Min**	**Max**	**Default**	**Unit**	**Description**
Delay	0	1000	0	ms	Sets the delay time. In milliseconds, multiplied by the Range parameter.
Attack	0	1000	500	ms	Sets the attack time.
Hold	0	1000	0	ms	Sets the hold time.
Decay	0	1000	500	ms	Sets the decay time.
Sustain	-100.0	100.0	0.0	%	Sets the sustain level.
Release	0	1000	500	ms	Sets the release time.
Attack shape	0	127	64		Sets the attack curve shape. Low values are more exponential; high values more linear.
Decay shape	0	127	64		Sets the decay and release curve shapes. Low values are more exponential; high values more linear.
Range	0	4	2		The time range (multiplier), from ‘x 0.01’ to ‘x 100’.

Trigger modes

Gate	The envelope is triggered by a rising edge (or MIDI note on). If it reaches the sustain stage, it stays there while the gate is high (or the MIDI note is held). When the gate goes low (or the MIDI note is released) the envelope moves immediately to the release stage, whichever stage it is in at the time.
Trigger	The envelope is triggered by a rising edge (or MIDI note on). It moves through its delay, attack, hold, decay and release stages once, and then is idle.
Gated loop	The envelope loops round its delay, attack, hold, decay and release stages as long as the gate is high (or the MIDI note held).
Looping	The envelope loops forever, once triggered.
Trigger by clock	As 'Trigger', but triggered by whatever is specified for the envelope's clock input. It may still be triggered by its trigger input, or MIDI.
Trigger no retrigger	As 'Trigger', but the envelope cannot be retriggered while it's active. It always completes a full cycle before it can start again.

Clock modes

Off	No clocked behaviour.
No stretch	If the clock time is less than the total envelope time (the sum of the delay, attack, hold, decay and release times), the envelope's overall time is shrunk to match the clock time. If the clock time is longer than the total envelope time, the envelope's time is unchanged.
Stretch all	The overall envelope time is matched to the clock time, by stretching or reducing each stage time proportionally.
Stretch sustain	If the clock time is less than the total envelope time, the envelope's overall time is shrunk to match the clock time. If the clock time is longer than the total envelope time, a sustain stage is introduced to make the overall time match the clock time.

Envelope Follower

“Tracks the volume envelope of a signal”

File format guid: 'enfo'

Specifications:

- Channels, 1-12: The number of bus channels to process.

Description

This algorithm is a simple envelope follower. It outputs a CV according to the volume envelope of the input audio.

When operating on stereo (or multichannel) signals, it tracks the maximum of the various signals and outputs a single CV. If you want to separately track the left and right sides of a stereo signal, use this algorithm configured as two mono channels.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Enable	0	1	0		Enables the channel.
Input	1	64	1		The input bus to process.
Width	1	8	1		The number of busses to process, starting from the Input. For example, for a stereo signal, set this to 2.
Output	1	64	15		The output bus. If set to ‘None’, the input bus is used as output, and the mode is always ‘Replace’.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.
Attack	0	1023	100		The follower attack time. Exponential scale from 0.1ms to 1000ms.
Release	0	1023	100		The follower release time. Exponential scale from 1.0ms to 3000ms.

Envelope Sequencer

“A really, really big curve sequencer”

File format guid: 'ensq'

Specifications: None

Description

This algorithm is a 16-step envelope or curve sequencer. It can output a CV and/or MIDI. It shares a lot of its design with the Step Sequencer (below), and some of the documentation for this algorithm cross-references with that for the other.

Each step has a curve shape, a scale, and an offset. It also has a “Division”, which can either be a repeat count (the step is repeated a number of times on subsequent clocks) or a ratchet count (the step repeats a number of times within the duration of one clock).

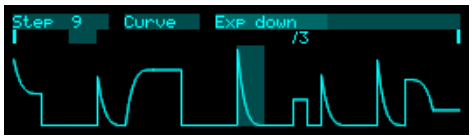
The algorithm has a function for randomising the curves in various ways.

Each Envelope Sequencer has internal storage for 32 sequences, which you can use like ‘snapshots’ or ‘patterns’ to prepare a number of sequences and then switch between them (manually, or under CV control) to build up larger structures.

Remember that every curve shape, level, etc. is a parameter and therefore mappable.

GUI

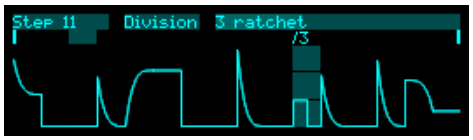
The display shows an overview of the 16 steps.



The current step being played is indicated by a highlight above the step. The start and end steps are indicated by solid bars.

The step whose parameter is being edited is shown highlighted.

Divisions (repeats or ratchets) are shown as breaks in the highlight, with a number above to indicate the division.



Menu functions

Some extra functions are available via the ‘Envelope Sequencer’ menu.

Copy sequence

This menu allows you to copy a sequence from one slot to another.

Reset sequence

This menu allows you to reset the current sequence to initial conditions.

Copy step

This menu allows you to copy a step within a sequence to another step.

Sequencer parameters

Name	Min	Max	Default	Unit	Description
Sequence	1	32	1		Selects the current sequence.
Start	1	16	1		The first step to play. A reset will jump to this step.
End	1	16	16		The last step to play.
Direction	0	6	0		The sequencer direction. See below.
Permutation	0	3	0		The sequencer permutation. See below.
Reset offset	-16	16	0		Offsets the step that will jumped to when the sequencer is reset.
Length type	0	1	0		Sets how the length of a step is determined: “% of clock” or “Fixed”.
% length	1	100	100	%	Sets the length of the step if the type is “% of clock”.
Fixed length	1	1000	100	ms	Sets the length of the step if the type is “Fixed”.

Step parameters

Name	Min	Max	Default	Unit	Description
Curve	0	11	0		Chooses the curve shape for the step.
Scale	-10.00	10.00	0.00	V	Sets the amplitude of the curve over the step.
Offset	-10.00	10.00	0.00	V	Sets the offset added to the curve during the step.
Division	0	14	7		Sets the step repeat or ratchet count.

Mute	0	100	0	%	Sets the probability that the step will be muted.
Skip	0	100	0	%	Sets the probability that the step will be skipped.
Reset	0	100	0	%	Sets the probability that the step causes a reset.
Repeat	0	100	0	%	Sets the probability that the step will be repeated.

Randomise parameters

Name	Min	Max	Default	Unit	Description
Randomise	0	1	0		When this parameter changes from 0 to 1, the sequence steps between the current start and end steps will be randomised according to the parameters that follow.
Levels	0	1	1		If set, the curve levels (scales) will be randomised.
Curves	0	1	0		If set, the curve shapes will be randomised.
Repeats	0	1	0		If set, the repeats/ratchets will be randomised.
Min repeat	2	8	2		The minimum repeat count, if a step is a repeat.
Max repeat	2	8	8		The maximum repeat count, if a step is a repeat.
Min ratchet	2	8	2		The minimum ratchet count, if a step is a ratchet.
Max ratchet	2	8	8		The maximum ratchet count, if a step is a ratchet.
Repeat probability	0	100	0	%	The probability that a step will be a repeat.
Ratchet probability	0	100	0	%	The probability that a step will be a ratchet.

Routing parameters

Name	Min	Max	Default	Unit	Description
Clock input	0	64	0		The bus to use as the clock input.
Reset input	0	64	0		The bus to use as the reset input.
Reset mode	0	2	0		Sets the mode for the reset input. The options are: ● the input is a reset trigger. ● the input is a run/stop signal. ● the input is a ‘one shot’ trigger.
Output	0	64	15		The bus to use for the CV output.

Output mode	0	1	0		The standard Add/Replace mode selector as described above.
Sequence CV input	0	64	0		The bus to use to select the sequence. See below.
Sequence trigger input	0	64	0		The bus to use for the sequence select trigger. See below.
Sequence change mode	0	1	0		How the sequence CV will be used to change sequences: ‘Switch’ or ‘Load’. See below.

MIDI parameters

Name	**Min**	**Max**	**Default**	**Unit**	**Description**
Follow MIDI clock	0	1	0		Sets whether the sequencer follows MIDI clock.
Divisor	0	19	4		Sets the clock divisor when following MIDI clock.
MIDI channel	0	16	0		Sets the output MIDI channel.
MIDI CC	0	127	0		Sets the MIDI CC number to send.
Output to breakout	0	1	0		Enables MIDI output to the breakout.
Output to Select Bus	0	1	0		Enables MIDI output to the Select Bus.
Output to USB	0	1	0		Enables MIDI output to USB.
Output to internal	0	1	0		Enables internal MIDI output – that is, MIDI is sent to the other algorithms.

ES-5 Encoder

“Converts gates into ES-5 outputs”

File format guid: 'es5e'

Specifications:

- Channels, 1-8: The number of gates to process.

Description

This algorithm allows you to drive the outputs of an attached ES-5 expansion module, or of an ESX-8GT expander attached to the ES-5.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Enable	0	1			Enables the channel.
Input	1	64			Chooses the input bus to convert.
Expander	1	6	1		Chooses the expander module – 1 for the ES-5 itself, or 2-6 for the attached ESX-8GTs.
Output	1	8	1		Chooses the output on the selected expander.

ESX-8CV Combiner

“Drives ESX-8CV outputs”

File format guid: 'esx8'

Specifications: None

Description

This algorithm allows you to drive the outputs of an ESX-8CV module attached to an ES-5 expander which is in turn attached to the disting NT.

Adaptive Mode

The ‘Adaptive’ parameter controls whether the algorithm will update all eight of the hardware outputs continuously (non-adaptive mode), or only those outputs that are changing (adaptive mode). In general the best performance will be in adaptive mode, unless you want the precise timing of the output updates to be predictable, in which case you may prefer non-adaptive mode.

As an example, assume that the system is running at a sample rate of 48kHz. In non-adaptive mode, each output is continuously updated in turn, and each update takes three samples. So, 24 samples are required to update all eight outputs, and so each output effectively runs at 2kHz.

In adaptive mode, the update rate depends on the number of channels that are changing. Each output update again takes 3 cycles, so if only one output of the ESX-8CV is changing, its update rate will be 16kHz. If two outputs are changing, they will be updated at 8kHz, and so on, down to 2kHz (as in non-adaptive mode) for eight outputs all changing all the time.

Parameters

Name	Min	Max	Default	Unit	Description
Expander	1	6	1		Chooses the expansion header on the ES-5 to which the ESX-8CV is connected.
Adaptive	0	1	0		See above.
1	0	64	1		The bus from which to drive output 1.
2	0	64	2		The bus from which to drive output 2.
3	0	64	3		The bus from which to drive output 3.
4	0	64	4		The bus from which to drive output 4.
5	0	64	5		The bus from which to drive output 5.

6	0	64	6		The bus from which to drive output 6.
7	0	64	7		The bus from which to drive output 7.
8	0	64	8		The bus from which to drive output 8.

Euclidean Patterns

“Generates Euclidean rhythm patterns”

File format guid: 'eucp'

Specifications:

- Channels, 1-8: The number of simultaneous patterns to generate.

Description

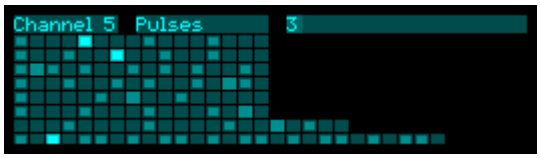
This algorithm generates rhythmic patterns of output pulses known as Euclidean patterns. For a detailed description of these patterns and how they are commonly found in music around the world see e.g. here[74] or here[75].

A pattern is described by the total number of steps and the number of pulses (i.e. the number steps on which a pulse is output).

The patterns can also be rotated. At zero rotation, the first step in the pattern will always be a pulse, and the remaining pulses distributed according to the algorithm. The rotation setting moves the first pulse later by a number of steps.

GUI

The display shows a graphical representation of the patterns (channel 1 at the top).



Globals parameters

Name	Min	Max	Default	Unit	Description
Clock input	1	64	1		The clock input bus. A rising edge advances the patterns by one step.
Reset input	0	64	0		The reset input bus. A high level resets the patterns to step 1 (and holds them there).
Follow MIDI clock	0	1	0		Sets whether the pattern generator follows MIDI clock.
Divisor	0	19	4		Sets the clock divisor when following MIDI clock.

74 https://en.wikipedia.org/wiki/Euclidean_rhythm

75 https://www.hisschemoller.com/blog/2011/euclidean-rhythms/

Auto reset	0	128	0		Sets a number of quarter notes after which the pattern will automatically reset to step 1 (only available when following MIDI clock).
Output to breakout	0	1	0		Enables MIDI output to the breakout.
Output to Select Bus	0	1	0		Enables MIDI output to the Select Bus.
Output to USB	0	1	0		Enables MIDI output to USB.
Output to internal	0	1	0		Enables internal MIDI output – that is, MIDI is sent to the other algorithms.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Enable	0	1	0		Enables the channel.
Steps	1	32	16		The number of steps in the pattern.
Pulses	1	32	4		The number of pulses in the pattern.
Rotation	0	32	0		The rotation of the pattern.
Repeat	0	128	0		The overall repeat count i.e. the number of clocks until the pattern repeats. If this is zero, the number of steps is used as the repeat. If the repeat count is greater than the number of steps, the pattern is extended with silence.
Output type	0	1	0		Sets whether the output is a fixed length trigger, or a clock pulse related to the period of the input clocks.
Length	1	100	10	ms	Sets the trigger length, if the ‘Output type’ is ‘Trigger’.
Length	1	100	50	%	Sets the output clock pulse width, if the ‘Output type’ is ‘% of clock’.

Per-channel routing parameters

Name	Min	Max	Default	Unit	Description
Output	0	64	15		The output bus for the channel.
Output	0	1	0		The standard Add/Replace mode selector as

mode					described above.
ES-5 Expander	0	6	0		If set, the ES-5 expander header to use for output instead of the bus specified by the Output parameter. “1” means the ES-5 itself.
ES-5 Output	1	8	1		If using an ES-5 expander output, sets which of the 8 outputs on the expander to use.
MIDI channel	0	16	0		The MIDI channel on which to send a note when a pulse fires.
MIDI note	0	127	0		The MIDI note number to send.

Feedback Receive/Send

“Receive/send for internal feedback loop”

File format guid: 'fbrx'/'fbtx'

Specifications:

- Channels, 1-8: The number of busses to receive/send.

Description

These two algorithms are taken together, as they are two parts of a mechanism to introduce feedback into presets. Though perhaps they are misnamed – they can just as easily feed forward. Think of them as two ends of a teleport/hyperspace tunnel – what goes in one end (the send) comes out of the other (the receive). If the receive is placed above the send in the preset you have feedback; if the receive is placed after the send, you have a way of teleporting busses past a section of the preset.

Both algorithms have an ‘Identifier’ parameter. This allows the system to match up send/receive pairs if there is more than one.

The send and receive can process a number of channels, according to the specifications. If these don’t match, the smaller number of channels will be processed.

Be careful when setting up feedback loops. There’s a reason that the receive’s gain parameter defaults to fully silent. Raise this with caution.

Receive: Common parameters

Name	Min	Max	Default	Unit	Description
Identifier	1	32	1		An identifier to match up the receive with a send.

Receive: Channel parameters

Name	Min	Max	Default	Unit	Description
Output	1	64			The bus on which to output the received signal.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.
Gain	-40.0	24.0	-40.0	dB	The gain applied to the received signal.

Send: Common parameters

Name	Min	Max	Default	Unit	Description
Identifier	1	32	1		An identifier to match up the send with a receive.

Send: Channel parameters

Name	Min	Max	Default	Unit	Description
Enable	0	1			Enables or disables the channel.
Input	1	64			The bus to send.

Filter bank

“A fixed filter/resonator bank”

File format guid: 'fbnk'

Specifications:

- Filters, 1-12: The number of filters in the bank.

Description

This algorithm is based on the disting EX algorithm of the same name. You may like to review the in-depth video for that algorithm here[76].

The disting EX manual states “this algorithm provides a bank of eight parallel stereo bandpass filters or resonators. The filters' levels can be controlled manually, via CV, or with envelopes driven from gates, and their pitch can also be set manually or via CV or MIDI. The resonators in particular are good for being played as chords over MIDI, in the manner of the Alesis Quadraverb Plus (see e.g. here[77]).”

This implementation goes further in that the number of filters can be increased to 12, and the filters can be mono, stereo, or up to 8 channels wide.

Please refer to the section “Common polysynth features” above.

Filter modes

The filters can be set to one of three modes: Resonator, Bandpass or Multiband.

Resonators are peaking all-pass filters that greatly emphasise the narrow band of frequencies around their centre frequencies.

Bandpass filters are a basic filter type that attenuate frequencies away from their centre frequencies.

In both of the modes above, all of the filters are completely independent. In Multiband mode however, the filter frequencies set a series of crossover points, such as you might find in a multiband compressor, for example. Therefore the frequencies passed by each band are bounded by its own frequency on the one side, and the frequency of the next band on the other.

76 https://www.youtube.com/watch?v=rn04rBEhIPM

77 https://www.youtube.com/watch?v=ZObDS3Hvfuo&t=90

GUI

The display shows the notes to which each filter is tuned, overlaid on a bar to represent the envelope level of the voice.

Globals parameters

Name	Min	Max	Default	Unit	Description
Audio input	1	64	1		The first audio input bus.
Width	1	8	1		The number of audio input busses (e.g. set to 2 for stereo).
Mode	0	2	0		The filter mode – see above.
Resonance/ Q	1	100	50	%	Sets the filter gain (for resonators) or resonance (for bandpass/multiband).
Attack time	0	1023	64		The envelope attack time. The scale is exponential, from 1ms up to 4s.
Release time	0	1023	64		The envelope release time. The scale is exponential, from 1ms up to 4s.
Gain	-40	24	0	dB	An overall gain control.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Pitch	0	127	0		The filter frequency/pitch, as a MIDI note number.
Gate	0	1	0		The filter gate.
Output	1	64	13		The filter’s (first) output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.
Gain	-40.0	24.0	0.0	dB	The filter’s gain.

Microtuning parameters

The algorithm uses the standard polysynth microtuning parameters, as described above.

MIDI/I2C parameters

Name	Min	Max	Default	Unit	Description
MIDI channel	0	16	1		The MIDI channel to listen on.
MPE channels	1	16	1		Controls how the algorithm will respond to MPE. See above.
I2C channel	0	255	0		Sets the I2C channel.
Controlled voices	1	12	12		Sets the number of filters that will be controlled by MIDI or CV/gate. By reducing this, you can elect to control some filters by MIDI while the rest remain under manual control.
Transpose	-60	60	0	ST	Coarse tuning control.
Fine tune	-100	100	0	cents	Fine tuning control.
Sustain	0	1	0		Directly controls the sustain (like a MIDI sustain pedal).
Bend range	0	48	2		The MIDI pitch bend range.
Unison	1	8	1		The number of voices to play simultaneously for each note triggered.
Unison detune	0	100	10	cents	The detune amount when Unison is active.
Sustain mode	0	1	0		The standard polysynth sustain mode parameter. See above.
Press -> volume	-100	100	0	%	The amount by which pressure (per note for MPE) affects the note volume.
MPE Y -> volume	-100	100	0	%	The amount by which the MPE Y dimension affects the note volume.

CV/gate parameters

The algorithm uses the standard polysynth CV/gate parameters, as described above.

Chord/arp parameters

The algorithm uses the standard polysynth chord and arpeggiator parameters, as described above.

Frequency Shifter

“Shifts frequency (not pitch)”

File format guid: 'fqsh'

Specifications: None

Description

This algorithm is based on the Frequency Shifter algorithm on the disting EX. You may like to view the video on that algorithm, which is here[78].

The algorithm is a frequency shifter. Note that this is very different from a pitch shifter. A pitch shifter multiplies the frequencies of the constituent harmonics of a sound by the same amount, and so preserves their harmonic relationship. A frequency shifter adds a shift to the frequencies of the constituent harmonics of a sound, and so tends to break harmonic relationships, making sounds more ‘clangorous’ or ‘bell-like’.

Shift parameters

Name	Min	Max	Default	Unit	Description
Shift	-1.000	1.000	0.000	Hz	Sets the frequency shift.
Multiplier	0	3	2		Set a multiplier for the shift amount: x1, x10, x100, or x1000.
Hz/V	0	3	1		Sets the scaling of the shift CV input, in Hz/Volt. One of: 1Hz/V, 10Hz/V, 100Hz/V, or 1kHz/V.
Mix	0	100	100	%	The output wet/dry mix.
Level	-40	0	0	dB	The output gain applied to the effected signal.

Routing parameters

Name	Min	Max	Default	Unit	Description
Shift input	0	64	0		The bus to use for the frequency shift CV.
Input	1	64	1		The first audio input bus.
Width	1	8	1		The number of busses to process, starting from the Input. For example, for a stereo signal, set this to 2.

78 https://www.youtube.com/watch?v=FaH9u9E2K9k

+ve shift output	0	64	13		The bus to use for the positive shifted output.
+ve shift mode	0	1	0		The standard Add/Replace mode selector as described above for the positive shifted output.
-ve shift output	0	64	0		The bus to use for the negative shifted output.
-ve shift mode	0	1	0		The standard Add/Replace mode selector as described above for the negative shifted output.

Granulator

“A granulator”

File format guid: 'gran'

Specifications:

- Max buffer size, 1-32 seconds: The maximum size of the recording buffer.

Description

This algorithm is an implementation of the original Granulator algorithm on the disting EX.

It implements a granular synthesis[79] engine, taking as its source material either live audio input or audio loaded from the MicroSD card. Live audio can be recorded into a buffer or streamed continuously.

Granular synthesis works by playing many short snippets of sound, or 'grains', typically of the order of 100ms in length. Often various properties of the grains (e.g. their timing, length, pitch, stereo panning etc.) are randomised to some extent.

In this algorithm, the creation ('spawning') of grains is controlled by 'notes'. Notes control when grain clouds begin and end, and affect other features e.g. the grain pitch.

Notes can be played via CV/gate pairs, or MIDI, or I2C. The algorithm also offers three 'drone' voices, which can simply be enabled via the usual parameter interface. When using the algorithm as an audio processing effect you're likely to just enable one or more of these drones and leave them on while manipulating other grain parameters.

The algorithm does nothing until you've recorded some audio into it or loaded some audio from the SD card.

Two suggested ways of getting started:

1) Connect an audio input, enable Record and enable Drone 1.

2) Connect CV/gate or MIDI, load a sample from the card, and play.

You may like to review the in-depth videos about the disting EX algorithm - the YouTube playlist is here[80].

Effects and routing

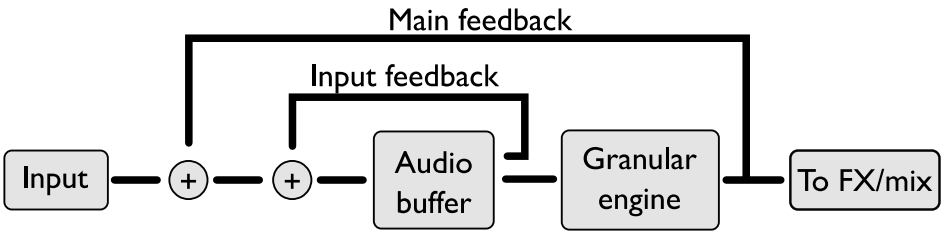
Those familiar with the disting EX version of this algorithm may wonder where the delay and reverb effects have gone. You can of course simply add them as extra algorithms on the disting NT, and do

79 Essential reading on granular synthesis includes Microsound by Curtis Roads, https://mitpress.mit.edu/9780262681544/microsound/

80 https://www.youtube.com/playlist?list=PLlY5j4QDwxWtIlNfDq_dCdryxR2hXTqvg

so with more routing flexibility.

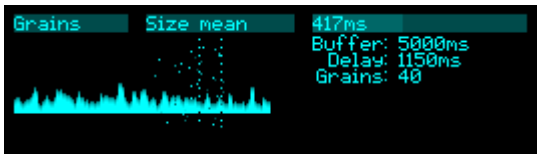
The “Main feedback” (which echoes the granulator output to the buffer input) and “Input feedback” (which effectively applies an echo on the input signal) paths are still available.



GUI

The display shows a waveform representation of the audio buffer, on which are superimposed dots indicating the currently playing grains. The two dotted lines represent the “Size mean” (by their separation) and the “Delay mean” (by their position within the buffer).

On the right of the screen you can see the current buffer size, the current delay, and the number of grains currently playing.



Loading/saving audio

Functions for loading and saving audio from/to the MicroSD card are available through the ‘Granulator’ menu.

Once a sample has been loaded via the menu, the algorithm remembers it, and displays its name in the GUI:

If the preset is then saved and reloaded, the sample will also be reloaded into the buffer.

Load sample

This submenu offers four versions of loading a WAV file from the MicroSD card into the Granulator’s audio buffer.

- **Into buffer/As buffer** - The ‘into buffer’ options keep the buffer size as-is, and load the sample into it. If the sample is shorter than the buffer, the remaining space is cleared. The ‘as buffer’ options change the buffer length to match the sample length (up to the maximum

buffer size as per the specifications).

- **& Normalize** - The options with ‘& normalize’ post-process the sample to normalize its level so the peak signal level is the nominal maximum. Can be useful when loading quiet samples.

Forget sample

This tells the algorithm to forget the sample location, so it will not be reloaded when the preset is loaded. The audio remains in the buffer, however.

Save recording

This saves the current buffer contents to the MicroSD card as a WAV file.

The file is saved into a folder called ‘granulator’, which will be created if it doesn’t already exist.

The filename is ‘saved <date and time>.wav’ where ‘date and time’ come from the real-time clock.

Mix parameters

Name	Min	Max	Default	Unit	Description
Input gain	-34	12	0	dB	Gain applied to the audio being recorded (does not affect the dry signal).
Dry gain	-40	6	0	dB	Level of the input signal in the output mix.
Granulator gain	-40	6	0	dB	Level of the granulator signal in the output mix.
Input feedback	0	100	0	%	The amount of feedback to apply around the audio buffer itself when recording (resulting in an echo effect on the input material, with a delay time equal to the buffer size).
Main feedback	0	100	0	%	The amount of the granulator output to feed back into the audio buffer when recording.
Normalize	0	1	1		If enabled, the overall volume of the grain cloud is lowered according to how many grains are active.

Record parameters

Name	Min	Max	Default	Unit	Description
Record	0	1	0		Enables recording into the buffer.
Buffer size	100			ms	The audio buffer size in milliseconds.
Record fade	0	1000	5	ms	The duration of the fade applied when starting and stopping recording, to avoid clicks.

Grains parameters

Name	Min	Max	Default	Unit	Description
Shape	0	5	0		The grain envelope/window shape. See below.
Spawn mode	0	4	0		How grains are spawned. See below.
Rate mean	1	1000	5	ms	The average time between new grains being spawned.
Rate spread	0	200	10	%	The amount of variation in the spawn rate, expressed as a percentage of 'Rate mean'.
Size mean	0	1000	100	ms	The average grain size. '0' has a special meaning – see below.
Size spread	0	200	10	%	The amount of variation in grain size, expressed as a percentage of 'Size mean'.
Pitch mean	-24	24	0	ST	The average grain pitch shift (in semitones). This is added to the pitch shift determined by the note's pitch CV.
Pitch spread	0	1200	0	cents	The amount of variation in grain pitch shift.
Pan mean	-100	100	0	%	The average grain pan position.
Pan spread	0	100	10	%	The amount of variation in grain pan.
Delay mean	0	100	50	%	The average grain delay (equivalently, the position in the audio buffer), expressed as a percentage of the buffer size.
Delay spread	0	100	5	%	The amount of variation in grain delay, expressed as a percentage of the buffer size.
Reverse	0	100	0	%	Sets the probability that a grain will be played backwards.
Grain limit	1	40	40		Imposes an arbitrary limit on the number of simultaneous grains.
Natural pitch	0	127	48	ST	Sets the natural pitch of the audio i.e. the MIDI note number that will play back the audio at the same pitch at which it was recorded.
Pitch quantize	0	5	0		Quantizes the random pitch deviation (the sum of the 'Pitch mean' and 'Pitch spread') to musical intervals. See below.
Opacity	0	100	100	%	The 'opacity' of a note, which is the percentage of

					grains that would normally make up the note that actually sound.
Grain position	0	1	0		Sets how the grain position is computed relative to the position set by the Delay mean/spread parameters. The options are ‘Centred on delay’ and ‘Start at delay’.

Modulation parameters

Name	Min	Max	Default	Unit	Description
LFO depth	-100	100	0	%	The depth of the LFO that affects the grain delay, expressed as a percentage of the buffer size.
LFO speed	0	255	196		The speed of the grain delay LFO. This is scaled relative to the buffer size – at the default value of 196 the LFO will cause the 'play head' (to use a tape metaphor) to advance at 1x speed.
LFO shape	0	2	0		Sets the LFO shape. The options are “Triangle”, “Ramp up”, and “Ramp down”.
Attack time	0	127	64		The envelope attack time, from 100ms to 30s with an exponential scaling.
Release time	0	127	64		The envelope release time, from 100ms to 30s with an exponential scaling.
Env -> opacity	0	100	0	%	The amount by which the note envelope affects the note opacity.
Env -> level	0	100	100	%	The amount by which the note envelope affects the note level (volume).
Veloc -> level	0	100	100	%	The amount by which the note velocity affects the note level (volume).
Veloc -> delay	0	100	0	%	The amount by which the note velocity affects the grain delay.
Pitch -> pitch	0	100	100	%	The amount by which the note pitch affects the grain pitch. Commonly this will either be 100% (normal pitch tracking) or 0% (the incoming pitch doesn't affect the grain pitch at all, but may still affect e.g. the grain delay).
Pitch -> delay	0	100	0	%	The amount by which the note pitch affects the grain delay.

Drone 1-3 parameters

Name	Min	Max	Default	Unit	Description
Drone 1-3 pitch	0	127	48/36/60	ST	The MIDI note number for the drone. Remember that ‘Natural pitch’ sets how these are interpreted. For the default Natural pitch of 48, done pitch 48 is the same pitch, drone pitch 36 is an octave down, and drone pitch 60 is an octave up.
Drone 1-3 enable	0	1	0		Enables (gates) the drone.
Drone 1-3 opacity	0	100	100	%	The opacity of the drone.
Drone 1-3 level	-40	6	0	dB	The level (volume) of the drone.

Setup parameters

Name	Min	Max	Default	Unit	Description
MIDI channel	0	16	0		The MIDI channel to listen on.
MPE channels	1	16	1		Controls how the algorithm will respond to MPE. See above.
I2C channel	0	255	1		Sets the I2C channel.
Bend range	0	48	2	ST	The MIDI pitch bend range.
Delay unit	0	1	0		Determines whether the Delay mean and spread parameters work in terms of % or ms.

Routing parameters

Name	Min	Max	Default	Unit	Description
Left input	1	64	1		The left audio input bus.
Right input	1	64	1		The right audio input bus.
Left output	1	64	13		The left audio output bus.
Right output	1	64	14		The right audio output bus.

Output mode	0	1	0		The standard Add/Replace mode selector as described above.
Delay mean input	0	64	0		The input bus to use to control the grain delay. A CV of 5V corresponds to 100% of the buffer size.
Delay mean sampled	0	1	0		If enabled, the Delay mean input is sampled once at the start of a note. Otherwise, it is sampled afresh as every grain spawns.

CV/gate parameters

The algorithm uses the standard polysynth CV/gate parameters, as described above.

Spawn mode

The spawn mode parameter controls the algorithm by which new grains are spawned. The options are:

- **Stochastic** (the default): grains are spawned randomly according to the 'Rate mean' and 'Rate spread' parameters.
- **Mid-grain**: a new grain is spawned at the middle point of the current grain. This is intended to be used with 'Size spread' at zero and with 'Shape' as 'Equal power', resulting in smooth crossfade looping, but you are of course free to use it creatively as you wish.
- **Single**: exactly one grain is spawned when a note is triggered. Essentially this gives you manual control over when grains are spawned.

The Stochastic and Mid-grain options also have 'fixed' variants ('Stochastic (fixed)' and 'Mid-grain (fixed)'). These behave identically unless recording is active. The difference relates to how the grain positions are calculated, using the delay mean and spread parameters.

In the non-fixed modes, the grain positions are relative to the current audio buffer write position – effectively, relative to 'now'.

In the fixed modes, the grains' positions are relative to the buffer write position at the time the note started. Conceptually, the grains are fixed relative to the audio, and indeed if you watch the grains spawn in the visual display you'll see them travelling across the screen as the audio scrolls.

In practice this means that in the 'fixed' modes, notes tend to have a fixed timbre for their duration, whereas in the non-'fixed' modes the timbre will vary as the audio scrolls past under the note position.

Size mean '0'

If the 'Size mean' parameter is set to zero, the grain size is calculated from the note pitch, and set to be the duration of one cycle of an oscillation that corresponds to the given pitch. This is particularly effective in conjunction with the 'Mid-grain (fixed)' spawn mode.

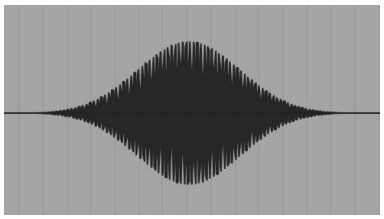
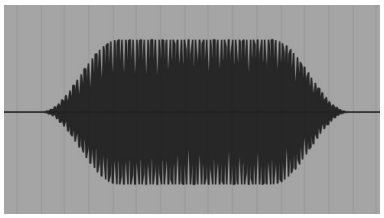
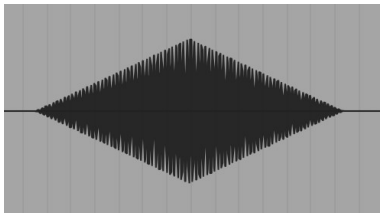
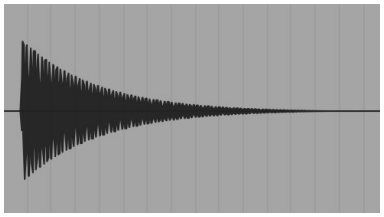
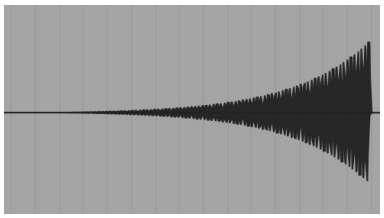
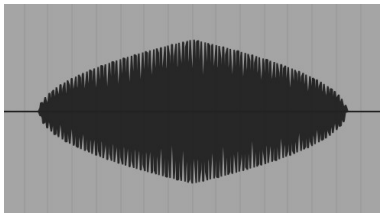
Pitch quantize

The available values for the Pitch quantize parameter are as follows.

0	Off
1	Octaves
2	Fourths
3	Fifths
4	Major Triad
5	Minor Triad

Shape

The 'Shape' parameter sets the volume envelope (also called 'window' in some of the literature) of the grains. The options are as follows.

Value	Name	Description	Image
0	Gaussian	A gaussian bell curve.	
1	Tukey	A rectangle convolved with a raised cosine.	
2	Triangle	A simple triangle shape.	
3	Expodec	A decaying exponential curve.	
4	Rexpodec	A rising exponential curve.	
5	Equal power	Back-to-back square root curves.	

Kirbinator

“Stochastic audio processor”

File format guid: 'krby'

Specifications:

- Buffer size, 1-44 seconds: The maximum size of the audio buffer.

Description

This algorithm continuously records audio into a buffer and plays slices of the buffer under the influence of various probabilities. The slices can be pitched up and down, played forwards and backwards, and panned in stereo.

Key to the operation of the algorithm are two trigger inputs, the Mark input and the Trigger input. These can be the same signal, but more interesting results often arise when they’re not. Typically both signals would be regular clocks, but this is by no means a requirement.

The Mark input defines the slices in the audio buffer. Say you have a rhythmic pattern going into the audio input, and the Mark signal is a (synchronised) clock. If that clock is a quarter note clock, the audio slices will be a quarter note long; if the clock is faster, say an eighth note, then the slices will be shorter. If your audio pattern is playing eighth notes and the clock is an eighth note, then one slice will be one note.

The Trigger input tells the Kirbinator when to choose a new slice to play and to change its playback parameters. In our example before, if the Trigger were a quarter note clock, the playback would jump and potentially change pitch and direction on every beat.

Note that playback continues with the new parameters until another Trigger is received; it doesn’t, say, play one slice and then stop. However, if the Play probability is less than 100%, a new Trigger may cause the playback to stop. A trigger on the Stop input also causes playback to stop.

The audio buffer is stereo, but the algorithm can be used as mono, stereo, or mono-in stereo-out, according to the routing parameters.

Random seeds and ‘mutate’

By default the operation of the Kirbinator is completely random. However, via the ‘Random seed’ parameter, the behaviour can be made repeatable. Every time the ‘Apply seed’ parameter is set to 1, the internal random number generator is set to the ‘random seed’, and so its sequence of random actions will be repeated. Choose a different seed, and a different set of actions will be repeated. You might typically use this with a clock mapped to apply the seed every so often (say, every bar) to achieve a pattern which recognisably repeats in time with your track.

Additionally, the ‘Mutate’ parameter allows you to smoothly blend from one set of probabilities to another, so you can set up a repeating pattern, gradually blend to a different one, and then return to the

first.

Simon Kirby gave a nice explanation of this (with video) in the Discord server, here[81].

Transient detection

If you enable the ‘Detect’ parameter, the algorithm will detect transients in the incoming audio and automatically apply marks in the audio buffer, allowing you to use the algorithm without an explicit Mark input.

Note that you can use both together; automatic marks from transient detection supplemented by marks applied via the Mark input.

Unclocked operations

While we expect the most common use of the Kirbinator to involve sending regular clocks to its Mark and Trigger inputs, it also makes an interesting effect with more *ad hoc* triggers.

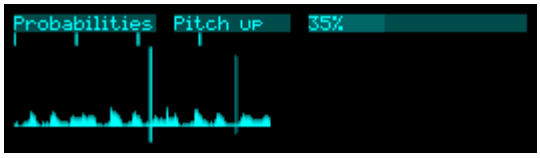
In this case you may want to set the play probability to 100%, so that you get playback every time you trigger it. In this case, simply sending a trigger to the trigger input starts playback. If you leave the mark input unassigned, and there are no existing marks, a mark will be made at the buffer position at which you triggered playback.

The ‘Stop’ input can be used to stop playback, which otherwise continues indefinitely (or until a trigger with a non-100% play probability stops it). There is also a parameter that allows for playback while you hold down the play trigger, and for stopping when you release it – if you set ‘Trigger hold time’ to something other than ‘Off’, that sets the time that the trigger has to be held after which releasing it stops playback. This works particularly nicely with footswitches – playback occurs while you step on the switch, and stops when you lift your foot.

You may also like to combine the above with the ‘Feedback’ parameter, which essentially turns the effect into a delay/echo (try it with reverse playback).

GUI

The display shows a waveform representation of the audio buffer. The shorter, darker line represents the current recording position. The longer line or lines show the current playback position (if any). The small checkmarks along the top of the waveform indicate the ‘marks’ defined by the Mark input.



81 https://discord.com/channels/1268195809502036049/1292493724462485647/1330595944374468682

Probabilities parameters

Name	Min	Max	Default	Unit	Description
Pitch up	0	100	0	%	The probability that playback will be pitched up.
Pitch down	0	100	0	%	The probability that playback will be pitched down.
Fifths	0	100	0	%	The probability that, if playback is pitched up or down, it will be by a perfect fifth; otherwise, it will be by an octave.
Reverse	0	100	0	%	The probability that playback will be reversed.
Glide	0	100	0	%	The probability that a glide will be applied to any pitch change.
Stutter	0	100	0	%	The probability that playback of the new slice will be stuttered, that is, that the initial part of the playback will be repeated.
Triplet	0	100	0	%	If ‘Metrical’ stutter has been selected, the probability that a stutter will be a triplet division, rather than a division by a power of 2.
Play	0	100	100	%	The probability that playback will actually happen; otherwise, playback stops.

Jumps parameters

Name	Min	Max	Default	Unit	Description
Glide control	0	1	0		Sets whether a single glide time is used, or independent times for up and down glides.
Glide	0	1000	100	ms	The time of the glide time to apply, if a glide is activated by the probability.
Glide up	0	1000	100	ms	The glide up time, if the control is independent.
Glide down	0	1000	100	ms	The glide down time, if the control is independent.
Through zero	0	1	1		Whether glides are allowed when the playback direction reverses. Such glides can produce a very strong “tape stop” effect.
Min stutter	1	16	2		If a stutter is activated, the minimum number of restarts.
Max stutter	1	16	3		If a stutter is activated, the maximum number of restarts.

Min jump	0	16	0		The minimum number of slices away from the current record position to jump.
Max jump	0	16	0		The maximum number of slices away from the current record position to jump.
Stutter	0	1	1		Sets whether stutter is ‘Free’ (unconstrained apart from by the minimum and maximum parameters), or ‘Metrical’ (restricted to powers of 2 and 3, according to the Triplet probability).

Tweaks parameters

Name	**Min**	**Max**	**Default**	**Unit**	**Description**
Pitch offset	-24	24	0	ST	Applies a constant pitch change to playback, added to any probabilistic pitch change.
Glide offset	-1000	1000	0	ms	Applies a constant glide time, added to any probabilistic glide.
Fade	0	1000	10	ms	Sets the fade in/out time for slice playback.
Random seed	0	32767	0		If non-zero, a random seed to apply.
Apply seed	0	1	0		Applies the random seed when the parameter goes from 0 to 1.
Mutate	0	100	0	%	The degree of mutation from the given random seed to the next.
Trigger hold time	0.0	5.0	0.0	s	Sets the time after which releasing the trigger input will be treated as a stop trigger.

Buffer parameters

Name	**Min**	**Max**	**Default**	**Unit**	**Description**
Buffer size	1.00			s	The audio buffer size (the maximum is set by the algorithm specifications).
Feedback	-24.0	0.0	-24.0	dB	The amount by which existing buffer content is attenuated on each recording pass. -24dB is treated as –∞dB i.e. the previous content is completely replaced.

Transients parameters

Name	Min	Max	Default	Unit	Description
Detect	0	1	0		Enables transient detection. See above.
Sensitivity	0	200	100	%	The transient detection sensitivity.

Mix parameters

Name	Min	Max	Default	Unit	Description
Dry gain	-40	6	-40	dB	The level of the input signal passed through to the output.
Effect gain	-40	6	0	dB	The output level of the Kirbinator effect.
No-play dry gain	-40	6	-40	dB	The level of the dry signal passed through when (and only when) a slice is not playing.
Pan mean	-100	100	0	%	The average pan position of slice playback.
Pan deviation	0	100	0	%	The amount of random deviation away from the average pan position, calculated for each trigger.

Routing parameters

Name	Min	Max	Default	Unit	Description
Left/mono input	1	64	1		The left or mono audio input.
Right input	0	64	0		The right audio input.
Left/mono output	1	64	13		The left or mono audio output.
Right output	0	64	0		The right audio output.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.
Mark input	0	64	3		The bus to use as the Mark signal.
Trigger input	0	64	3		The bus to use as the Trigger signal.
Stop input	0	64	0		A trigger on this bus causes the currently playing slice to stop.

Pitch offset input	0	64	0		A 1V/octave pitch CV on this bus is added to the ‘Pitch offset’ parameter.

Knob Recorders

“Configures the Knob Recorders”

File format guid: 'knob'

Specifications:

- Time, 1-7200 seconds: The total amount of knob recorder time.

Description

This algorithm is an intrinsic part of the Performance Page (above). It is implemented as an algorithm so that clocks can be routed into it via busses, and so that the user can choose how much memory to allocate to the knob recorders from the pool available to the preset.

Globals parameters

Name	Min	Max	Default	Unit	Description
Reset input	0	64	0		The bus to use for a reset pulse, to reset the clock dividers to step 1.
Page select enable	0	1	0		Enables the ‘Page select’ parameter.
Page select	1	10	1		When this parameter changes, it changes the current page in the Performance Page custom UI. This will be most useful when mapped to a MIDI CC, for example, so you can remotely control the visible page.

Per-item parameters

Name	Min	Max	Default	Unit	Description
Enable	0	1	1		Enables the channel.
Time	1	1000	100	%	Scales the amount of time allocated to this item from the available total (as set in the specification).
Clock source	0	2	0		Sets the clock source for this item: one of ‘None’, ‘Clock input’, or ‘MIDI’.
Clock input	1	64	1		The bus to use for the clock, if the ‘Clock source’ is ‘Clock input’.

Divisor	4	19	9		The MIDI clock divisor, if the ‘Clock source’ is ‘MIDI’.
Clock multiplier	1	96	1		A multiplier applied to the clock source.

LFO

“A low frequency oscillator”

File format guid: 'lfo '

Specifications:

- Channels, 1-4: The number of LFO outputs.

Description

This algorithm offers a number of flexible low frequency oscillators.

The global ‘Quality’ parameter offers two options:

- ‘Stepped’ quality is extremely low CPU usage, and is appropriate for automating parameters of other algorithms (via mapping).
- ‘Linear’ quality should be selected if you plan to use the LFOs to output CVs to other modules.

The LFOs can run freely, or be synced to external clocks or to MIDI clock.

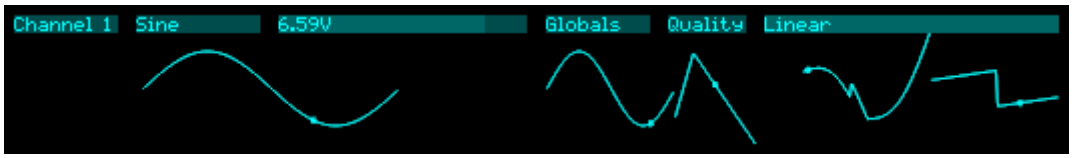
Phase-locking

When there is more than one LFO channel, channels 2-4 can be locked to the phase of LFO 1. This allows you to, for example, create a quadrature LFO.

Use the ‘Phase’ parameter of LFOs 2-4 to set the phase relationship in degrees to LFO 1. Note that in this case, the ‘Speed’ and ‘Multiplier’ parameters of the phase-locked LFO have no effect.

GUI

The display shows the shape of the current channel’s output, or if the ‘Globals’ page is active, the shape of all the channels.



Globals parameters

Name	Min	Max	Default	Unit	Description
Quality	0	1	1		Sets the LFO output quality. The options are “Stepped” and “Linear”.

Clock input	0	64	0		The bus to use as the clock input.
Reset input	0	64	0		The bus to use as the reset input.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Enable	0	1	0		Enables the channel.
Speed	0	16383	8605		Sets the LFO speed, from 0.05Hz to 15Hz with an exponential scaling.
Multiplier	0	3	1		Sets a multiplier for the LFO speed. The options are “x0.1”, “x1”, “x10”, & “x100”.
Sine	-10.00	10.00	0.00	V	Sets the amplitude of a sine wave to add into the LFO output.
Triangle	-10.00	10.00	0.00	V	Sets the amplitude of a triangle wave to add into the LFO output.
Saw	-10.00	10.00	0.00	V	Sets the amplitude of a saw wave to add into the LFO output.
Square	-10.00	10.00	0.00	V	Sets the amplitude of a square/pulse wave to add into the LFO output.
Pulse width	0.0	100.0	50.0	%	Sets the pulse width of the square/pulse wave.
Offset	-10.00	10.00	0.00	V	Adds a constant voltage to the LFO output.
Asymmetry	-100.0	100.0	0.0	%	Sets the asymmetry. This is somewhat like applying a pulse width variation to a square wave, but applies equally to all the waveforms.
Random	-10.00	10.00	0.00	V	Adds a stepwise random voltage to the LFO output.
Phase	0	360	0		Phase-locks LFOs. See above.

Per-channel routing parameters

Name	Min	Max	Default	Unit	Description
Output	0	64	15		The LFO output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Per-channel sync parameters

Name	Min	Max	Default	Unit	Description
Sync	0	4	0		Selects how the LFO will sync to external clocks or MIDI. See below.
Clock multiplier	0	23	15		Sets the clock multiplier, if syncing to external clocks.
MIDI divisor	0	19	9		Sets the MIDI clock divisor, if syncing to MIDI clock.

LFO sync modes

The Sync parameter allows LFOs to synchronise themselves to external clocks or to MIDI clock. The options are:

Clock	The LFO syncs its speed to a clock on the bus specified by the ‘Clock input’ parameter.
Clock is also reset	As ‘Clock’, but the clock also causes the LFO to reset its phase.
MIDI tempo	The LFO syncs its speed to incoming MIDI clock.
MIDI transport	The LFO syncs its speed and phase to incoming MIDI clock.
Clock and reset	As ‘Clock’, and a trigger on the reset input causes the LFO to reset its phase.

Linear/Exponential

“Converts e.g. V/Oct <-> Hz/V”

File format guid: 'lexp'

Specifications: None

Description

This algorithm implements a linear-to-exponential converter and an exponential-to-linear converter, typically used when interfacing between Volt/octave and Hz/Volt CV standards.

The 0V point for the V/octave signals is taken as C3 (130.8Hz approximately).

GUI

The display shows the input and output voltages, and the internal calculated frequency, for each converter.

Linear/Expo parameters

Name	Min	Max	Default	Unit	Description
Hz/V	1	2000	1000	Hz	The Hz/Volt scaling.
Offset	-1000	1000	0	mV	A voltage offset subtracted from the input.
Input	0	64	0		The input (linear, Hz/V) bus.
Output	0	64	0		The output (exponential, V/octave) bus.
Output mode	0	1	1		The standard Add/Replace mode selector as described above.

Expo/Linear parameters

Name	Min	Max	Default	Unit	Description
Hz/V	1	2000	1000	Hz	The Hz/Volt scaling.
Offset	-1000	1000	0	mV	A voltage offset added to the output.

Input	0	64	0		The input (exponential, V/octave) bus.
Output	0	64	0		The output (linear, Hz/V) bus.
Output mode	0	1	1		The standard Add/Replace mode selector as described above.

Little Spacey

“A bucket-brigade delay effect”

File format guid: 'lisp'

Specifications:

- Stereo: Whether the algorithm is mono or stereo.

Description

This algorithm is a version of the Expert Sleepers VST plug-in of the same name, which you can find here[82]. From that page:

> Little Spacey is delay effect, inspired by classic analogue ‘bucket brigade’ delays but with refinements only possible in the digital arena – not to mention a maximum delay time that would cost a small fortune to realise in hardware.
>
> Great care has been taken to give Little Spacey the smooth, creamy sound which makes analogue delays still popular today.
>
> In common with the best analogue delays Little Spacey allows you to modulate the delay time for chorus or vibrato effects. Unlike most analogue delays, Little Spacey is not restricted to mono operation, and works well in stereo settings, offering variation of the effect between the channels for particularly rich and involving sounds.

Unlike the VST version, this implementation also allow the delay time to be synchronised to clocks (analogue or MIDI).

Feedback path

In common with all ‘bucket brigade’ designs, Little Spacey uses filters either side of the delay to remove aliasing and clock noise.

The Path setting lets you control whether the feedback goes around the full signal path (including the filters) or just around the delay itself (not the filters).

In practice this means that the ‘inner’ path gives a relatively bright delay while the ‘outer’ path tends to lose the top end off the delays much more quickly. Settings between ‘inner’ and ‘outer’ simply use a mixture of both feedback paths.

LFOs

LFO 1 modulates the delay time. LFO 2 modulates the depth of LFO 1.

82 https://expert-sleepers.co.uk/littlespacey.html

Regarding ‘Spread’ – at 0%, the LFOs are in phase i.e. the same modulation is applied to each channel. At 50% the LFOs are exactly out of phase i.e the LFO for the left channel is at its maximum when the LFO for the right channel is at its minimum. This is useful for creating ‘wide’-sounding effects from mono sources.

GUI

The display simply shows the current delay time, taking into account any clock syncing, and the limits on the delay time imposed by the currently selected BBD size.

Delay parameters

Name	Min	Max	Default	Unit	Description
BBD Size	0	5	0		Sets the number of bucket-brigade stages.
Delay time	0.0	100.0	50.0	%	Sets the delay time, between the minimum and maximum times determined from the BBD size and the current sample rate.
Spread	0	100	0	%	Sets the difference in delay times between the left and right channels. At 0%, the two channels have the same delay time.
Colour	0	100	0	%	Allows you to lower the cut-off frequencies of the low pass filters, resulting in delays that lose more of their high frequency components.
Feedback	0.0	110.0	0.0	%	Sets the amount of the delayed signal that gets routed back to the input of the delay, causing repeated echoes. Note that the control goes up to 110%. Settings above 100% cause ‘positive feedback’ i.e. the delays get louder over time.
Feedback path	0	100	0	%	Sets the feedback path, as described above.

Mix parameters

Name	Min	Max	Default	Unit	Description
Mix	0	100	100	%	The wet/dry mix.
Level	-40	0	0	dB	The gain applied to the delay/wet signal.

Dry mix	0	100	0	%	Controls whether the ‘dry’ output is indeed the true dry sound or a filtered version. The filtered version can be useful when using fairly ‘dark’-sounding delays to match the dry sound more closely to the delayed sound.

LFOs parameters

Name	Min	Max	Default	Unit	Description
LFO 1 Speed	0	1000	700		The LFO speed. Exponential scale from 0.05Hz to 20Hz.
LFO 1 Depth	0	100	0	%	The LFO depth.
LFO 1 Spread	0	100	0	%	Varies the phase of the LFO applied to each (stereo) channel.
LFO 2 Speed	0	1000	700		The LFO speed. Exponential scale from 0.05Hz to 20Hz.
LFO 2 Depth	0	100	0	%	The LFO depth.
LFO 2 Spread	0	100	0	%	Varies the phase of the LFO applied to each (stereo) channel.

Env Mod parameters

Name	Min	Max	Default	Unit	Description
Ducking	0	40	0	dB	Reduces the level of the delays when the signal level is high. This is typically used to let the initial note ‘cut through’ the delays, with the delays then fading up during the note’s decay tails.
Env->LFO 1 Speed	-100	100	0	%	Sets the amount by which the envelope modifies the modulation speed. Positive settings cause the modulation to speed up when the sound is louder; negative settings cause the modulation to slow down when the sound is louder.
Env->LFO 1 Depth	-100	100	0	%	Sets the amount by which the envelope modifies the modulation depth. Positive settings cause the modulation depth to increase when the sound is louder; negative settings cause the modulation depth to decrease when the sound is louder.

Env->LFO 1 Spread	-100	100	0	%	Sets the amount by which the envelope modifies the modulation LFO’s spread.

Sync parameters

Name	Min	Max	Default	Unit	Description
Clock source	0	2	1		Chooses the clock source: ‘None’, ‘Clock input’, or ‘MIDI clock’.
Clock input	0	64	0		The input bus to use for the clock, if the source is ‘Clock input’.
MIDI divisor	0	19	9		The MIDI divisor, if the clock source is ‘MIDI clock’.
MIDI numerator	1	16	1		The MIDI numerator, if the clock source is ‘MIDI clock’.
Delay multiplier	0	23	15		A delay multiplier, if the clock source is not ‘None’.

Routing parameters

Name	Min	Max	Default	Unit	Description
Left/mono input	1	64	1		The left or mono input bus.
Right input	1	64	2		The right input bus.
Left/mono output	1	64	13		The left or mono output bus.
Right output	1	64	14		The right output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Logic

“Performs various logical operations”

File format guid: 'logi'

Specifications:

- Channels, 1-8: The number of channels to process.

Description

This algorithm is based on the disting mk4 algorithm of the same name. You may like to review the video on that algorithm, which is here. It also includes comparator functions, thus fulfilling the role of the disting mk4 Comparator algorithm, the video for which is here[83].

Each channel of the algorithm is independent and performs logical operations on its X & Y inputs.

The following operations are available:

‘Operation’ value	Operation
0	AND
1	OR
2	XOR
3	NAND
4	NOR
5	XNOR
6	SR flip-flop
7	D flip-flop
8	> (greater than)
9	< (less than)

The SR flip-flop is set high by a rising edge on input X, and cleared low by a rising edge on input Y.

The D flip-flop takes the level of input X on a rising edge on input Y.

The global parameters set the threshold and hysteresis of the input comparators which generate logical signals from arbitrary CVs (for an explanation of hysteresis see here[84]).

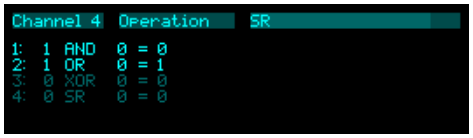
Complex logical operations can be constructed by combining operations. The inputs of each channel can be set to the outputs of other channels, allowing you to chain them internally to the algorithm.

83 https://www.youtube.com/watch?v=Ttp7RCtofg0

84 http://en.wikipedia.org/wiki/Hysteresis#Electronic_circuits

GUI

The display shows the various logical operations being performed. Disabled channels are grayed out.



Inverted inputs and outputs are indicated using the standard logic notation of a bar (horizontal line) above the inverted quantity.

Globals parameters

Name	Min	Max	Default	Unit	Description
Threshold	0.00	10.00	1.00	V	Sets the input threshold. Signals above this voltage are treated as a logic ‘1’/true.
Hysteresis	0.00	10.00	0.50	V	Sets the input hysteresis. Once over the threshold, signals must fall by this amount before being treated as a logic ‘0’/false.
Output true	-10.00	10.00	5.00	V	The output voltage for logic ‘1’/true. Logic ‘0’/false is always output as 0V.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Enable	0	1	0		Enables the channel.
Operation	0	9	0		Selects the logical operation from the table above.
Invert X	0	1	0		Inverts the X input (before it is used by the logical operation).
Invert Y	0	1	0		Inverts the Y input (before it is used by the logical operation).
Invert output	0	1	0		Inverts the output.
Input X	1		1		Chooses the X input bus or channel for the operation.
Input Y	1		2		Chooses the Y input bus or channel for the operation.

Output	0	64	15		Chooses the output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Looper

“A looper”

File format guid: 'loop'

Specifications:

- Loops, 1-4: The number of simultaneous loops.
- Max time, 1-90: The maximum total loop time.

Description

This algorithm is a looper, based on the disting EX algorithm of the same name, but significantly extended. You may like to review the video about the EX version, which is here[85].

Loops can have crossfades, they can fade in and out, and overdubbing can fade in and out, all of which is designed to make it easy to achieve smooth, ambient looping.

All looping operations can also be synchronised to a clock input, if tight rhythmic looping is more your thing. The looper can also generate clocks (analogue and/or MIDI), if you prefer the looper to be the source of tempo in your setup.

Loops may also be reversed, and played at half speed (an octave down).

A ‘layers’ system is also implemented, which is like the ‘undo/redo’ found on some loopers but more powerful.

The core control setup is a two button record/play/overdub paradigm in the manner of many stomp-box loopers, though many more controls are available. These can be mapped to CV, MIDI, or I2C as usual; a dedicated MIDI-note based interface is also provided for ease of use with MIDI pedalboards which send notes when footswitches are pressed.

You may like to review the Debouncer algorithm (above) which is designed to make it easy to hook up guitar footswitches to the looper controls.

A UI script (see below for information about UI scripts) is available which puts the looper’s main record/play buttons on the module’s two encoders. This is available from our GitHub[86] if it’s not already on your MicroSD card.

Available loop time

The total time set in the specifications is shared equally between the loops, so if you choose to have one loop with a ‘max time’ of 90 seconds, that loop can be up to 90 seconds, but if you choose to have four loops with the same ‘max time’, each loop can only be 22.5 seconds long.

The actual maximum time for each loop also depends on its bit depth and whether it’s stereo or mono.

85 https://www.youtube.com/watch?v=ZkL7v5CCzgc

86 https://github.com/expertsleepersltd/distingNT

The quoted time in the specification is for mono 32 bit. Making a loop stereo halves its maximum length; making it 16 bit doubles the maximum length.

Be aware that clipping can occur when using 16 or 24 bit modes, so watch your input levels. No clipping can occur when using 32 bit float.

Changing parameters that affect a loop’s maximum memory cause an instant clear of the loop, reverting it to the ‘empty’ state. These parameters are ‘Channels’, ‘Bit depth’, and ‘Minimum layers’.

Loop targets and commands

Fundamental to the operation of the Looper is the concept of the 'target loop'. This is how we expose the control of up to four independent loopers through a two button interface. The record/overdub/play/etc. operations are considered as 'commands' that apply to whichever loop or loops is the 'target'.

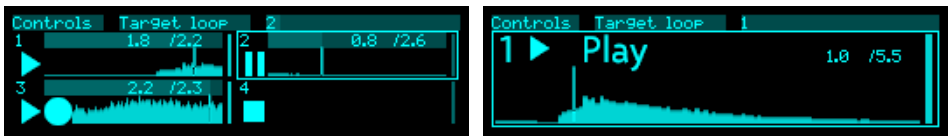
The target loop is set by the ‘Target loop’ parameter. The commands are given by the remaining parameters on the ‘Controls’ page. The command is given when the parameter changes from 'Released' to 'Held', with the exception of ‘Target next’, which is given when the parameter changes from ‘Held’ to ‘Released’. This is because ‘Target next’ also has a ‘long hold’ function, controlled via the ‘Next hold time’ parameter.

Each command parameter has an equivalent MIDI note parameter on the ‘MIDI’ page.

All commands are simple toggles, with the exception of the ‘Target 1/2/3/4’ commands. While at least one of these is held, holding another adds to the targetted loops. When all are released, the next target held replaces the target selection. Thought of in terms of MIDI notes, this means you can hold ‘chords’ to select multiple target loops at once.

GUI

The display shows a considerable amount of information about the loop(s). A special large version of the GUI is shown if there is only a single loop.



The target loops are indicated with a box drawn around their section of the display.

Each loop shows an icon to indicate its state, as follows:

- Square: Empty
- Circle: Record (or Record x-fade if the circle is pulsing)
- Triangle: Play (reversed if triangle points left; octave down if a small hole in the triangle)
- Triangle and circle: Overdub
- Two vertical lines: Stop
- Two vertical lines with downward triangle: Fade to stop, or Fade to clear

- ‘M’ instead of two vertical lines indicates ‘mute’ rather than ‘pause’

The length of the loop, and the position within the loop, are shown top right (both in seconds).

The loop’s envelope level is indicated by the vertical bar at the right side.

Basic looping

The procedure is as follows:

- Connect an input signal.
- Choose the target loop.

  The display will initially show a square, which is the 'stopped' symbol. The target loops are shown with an outline.
- Press 'Record' to begin recording.

  The symbol for the target loop will change to a circle, for 'recording'. The recording time will start counting up in the upper right of the display.

  We say ‘Press Record’ etc. here – obviously there’s not a dedicated record button. By this we mean activate record however you’ve chosen to do it – with a MIDI note, a CC or CV mapped to the parameter, via the UI script, or simply by manually changing the parameter.
- Press 'Record' again to end recording and immediately enter playback.

  The symbol for the loop will change to a triangle, for 'play'. The upper right area shows the current time in the loop, and the loop length. The moving line on the waveform display also shows the loop playback position.
- If the recording approaches the maximum loop time, the upper left of the display will show 'remaining' and the remaining loop time.

  Recording will automatically stop if it reaches the maximum time.
- Recording can also be ended by pressing 'Play'.
- If ‘Record’ is held down for longer than the time set by the ‘Record hold time’ parameter, releasing ‘Record’ is then treated as if you’d pressed it again. To reiterate – recording can be done by either tapping ‘Record’ twice (once to start and once to stop), or by holding it once.

Recording a blank loop

Sometimes it is useful to set the loop length without actually recording anything into it. To do this, press 'Play' instead of 'Record' to begin recording.

The display will show the record icon (circle) with an 'M' for mute.

Pausing/muting/retriggering the loop

Once a loop is playing, pressing 'Play' pauses or mutes the loop, depending on the 'Pause/mute' parameter.

If the loop is paused, the symbol changes to the 'double vertical line' icon.

If the loop is muted, the symbol changes to an 'M'.

Pressing 'Play' again returns the loop to play mode.

A third option, 'Retrigger', is like Mute except that when the loop starts playing again it does so from the beginning of the loop.

Loops may also be retriggered with the ‘Retrigger’ control.

Overdubbing

While a loop is playing, pressing 'Record' enters overdub mode. Incoming material is recorded on top of the previous loop. The loop's symbol shows both the play (triangle) and record (circle) icons.

Press 'Record' again to stop overdubbing.

The ‘Overdub fade’ parameter controls how previous material fades out during overdubbing, if at all. If you’re building up a rhythmic part piece by piece, you would probably leave this at 0dB i.e. no fade. If you’re layering up some swooshy ambience, you probably want this down at maybe -3 to -6dB.

The ‘Record hold time’ parameter also applies to overdubbing, that is, you can enter and leave overdub by holding down ‘Record’ instead of tapping it twice.

Clearing the loop

When a loop is paused or muted, you can clear the loop, which means erasing it completely and returning to 'stopped'.

To do so, hold down 'Record'.

While you hold the button, the word 'CLEAR' will appear with a graphical countdown. When the countdown reaches zero, the loop will be cleared.

Crossfades

Each loop can have a crossfade time set, which helps to make smooth loops. Note that raising the crossfade time effectively reduces the loop time, since two passes of the loop have to be overlaid and mixed.

If the crossfade time is set before recording the loop, the recording will run on after the end-of-record command, in order to record extra material for the crossfade. The overall loop time will still be correct i.e. the time between the start- and end-of-record commands.

Envelopes

Two attack-decay envelopes are provided; one for when a loop is started and stopped, and one for when overdubbing starts and ends.

While the loop is in the decay stage, fading out towards pause/mute, a downwards-pointing triangle is shown next to the icon in its display.

While the end of overdub decay stage is active, the record indicator (the circle) pulses.

Clocked operation

The algorithm has a ‘Clock source’ parameter, which can be ‘None’, ‘Clock input’, ‘MIDI clock’, or ‘Loop 1’. If a clock source is chosen, any looping command is deferred until the next clock that arrives. This makes it possible to synchronise the loop length, and the starting and stopping of playback etc., with a clock.

When in clocked mode, the 'Lock range' parameter activates automatic re-triggering of the loops to keep them locked to the clock. The parameter specifies a time window around the clock – if a loop would naturally loop during this window, it is retriggered so that it begins exactly on the clock pulse.

The ‘Bypass clock’ parameter is provided to allow the clocks to be temporarily ignored (for example if you want to implement an “instant stop” even if clocked operation is active).

The ‘MIDI transport’ parameter allows you to have the Looper start and stop playback along with whatever is sending it MIDI clock.

If the Clock source is Loop 1, then the first loop operates unsynchronised, and the other loops are synchronised to the start of the first. Note that this allows for the other loops to be multiples of the first loop in length, not just to be the same length.

Generating clocks

Each loop can output a clock, synced to its length. By default this is a single trigger at each loop start, but by using the ‘PPQN (out)’ and ‘1/4 notes (out)’ parameters you can choose how many trigger/clock pulses each loop generates. Typically you would set this value before recording your loop.

The first loop can also generate MIDI clock. MIDI expects 24ppqn, so set ‘PPQN (out)’ to 24 if you want the ‘1/4 notes (out)’ parameter to actually mean quarter notes.

Stop all

When ‘Stop all’ is pressed, all currently playing loops are selected as targets, and stopped. When released, all stopped loops (including loops that were already stopped before the note was sent) are selected as targets (which can then all be conveniently cleared if desired).

Layers and Undo/redo

If layers are enabled, each time an overdub is started the contents of the loop are saved as a new layer, so you can go back to it at a later time. This is like the undo/redo system offered by some stomp-box loopers, but rather than there being a single undo, you can go up and down a whole stack of layers.

You can move between layers with the ‘Undo’ and ‘Redo’ commands, or by directly manipulating the ‘Current layer’ parameter. If you ‘Undo’ on the first layer this is treated as a ‘Redo’ – this is a convenience for situations where you only have a single pedal/button set up.

The ‘Layer fade’ parameter sets the duration of a crossfade that is applied when moving between layers.

A new layer is started only when overdub is activated, not every time around the loop while you remain in overdub.

The number of available layers depends on the available memory. If, say, you have 60 seconds of loop memory, and record a 10 second loop, then you can use up to 6 layers – but if you record a loop longer than 30 seconds, only one layer will be possible.

If you’re already on the last available layer when you activate overdub, your overdub will be recorded into the existing layer. You can still undo to the previous layer, but redoing to the last layer will bring back the recording after the second overdub.

The ‘Minimum layers’ parameter tells the looper how many layers you want to reserve space for. For example, if you have 60 seconds of loop memory and request a minimum of 4 layers, then the longest loop you can record will be 15 seconds.

If you set ‘Minimum layers’ to ‘Layers disabled’, layers won’t be used at all. Starting an overdub will not create a new layer, even if there’s enough memory to do so.

Only record targets

The ‘Only record targets’ parameter offers a slightly different way of approaching overdubs.

When this parameter is off (the default), the ‘Target loop’ parameter only affects which loops receive looping commands. All loops receive the input audio. Therefore, you can put a loop into overdub, select a different target loop to control, and the first loop will continue to overdub as normal.

When ‘Only record targets’ is on, the ‘Target loop’ parameter also affects which loops are able to receive audio. There is effectively a mute control in front of every loop, and only loops chosen as targets are unmuted. Taking the same example as above, if you put a loop into overdub, and then select a different target loop, the input to the first loop will be silenced.

This feature is intended to be useful when you want to leave a loop in overdub with the overdub fade parameter set to have the loop fade out, like a long delay/echo effect. With ‘Only record targets’ on, you can leave this loop gradually decaying while you record or overdub into a different loop.

Controls parameters

Name	Min	Max	Default	Unit	Description
Target loop	0		1		Sets the target loop(s).
Record	0	1	0		Activates record or overdub.
Play	0	1	0		Activates play or stop.
Reverse	0	1	0		Activates ‘reverse’ (toggles the target loops between playing forwards and backwards).
Octave down	0	1	0		Activates ‘octave down’ (toggles the target loops between playing at normal speed and half speed).
Retrigger	0	1	0		Activates retrigger.
Stop all	0	1	0		Activates ‘stop all’. See above.
Bypass clock	0	1	0		While this parameter is held, commands such as Record, Play, etc. behave as if the ‘Clock source’ parameters were ‘None’.
Undo	0	1	0		Activates ‘Undo’. See Layers, above.
Redo	0	1	0		Activates ‘Redo’. See Layers, above.
Target 1	0	1	0		Activates loop 1 as a target loop.
Target 2	0	1	0		Activates loop 2 as a target loop.
Target 3	0	1	0		Activates loop 3 as a target loop.
Target 4	0	1	0		Activates loop 4 as a target loop.
Target next	0	1	0		Activates the loop after the current target as the new target.
Fade to clear	0	1	0		Activates ‘fade to clear’ – the same as pressing Play to stop the loops, except that the loops will be automatically cleared when the fade out has completed.

Globals parameters

Name	Min	Max	Default	Unit	Description
Pause/mute	0	2	0		Controls what happens when a loop is stopped. See above.

Attack time	0	1000	0		The loop envelope attack time. The scale is exponential, from 1ms to 30s.
Decay time	0	1000	0		The decay envelope attack time. The scale is exponential, from 1ms to 30s.
Attack shape	0	100	0	%	The loop envelope attack stage shape.
Decay shape	0	100	0	%	The loop envelope decay stage shape.
Overdub attack time	0	1000	0		The overdub envelope attack time. The scale is exponential, from 1ms to 30s.
Overdub decay time	0	1000	0		The overdub envelope attack time. The scale is exponential, from 1ms to 30s.
Overdub fade	-24.0	0.0	0.0	dB	Sets how much the previous loop content will fade out on each pass while overdubbing.
Retrigger fade	0.5	100.0	2.0	ms	Sets the duration of the crossfade applied when retriggering.
Layer fade	1	1000	10	ms	Sets the duration of the crossfade when changing layers.
Clear hold time	0.5	5.0	1.0	s	Sets how long Record needs to be held before a Clear happens.
Record hold time	0.5	5.0	1.0	s	Sets how long Record needs to be held before releasing it is treated as pressing it again.
Next hold time	0.1	5.0	0.3	s	Sets how long ‘Target next’ needs to be held before the target loop is set to None (or all loops, if it is already None).
Next hold order	0	1	0		If set, reverses the order of targetting none or all loops by holding ‘Target next’.
Mixed play/stop	0	2	0		Sets what happens when you hit ‘Play’ and some targetted loops are playing while others are stopped. The default ‘Toggle’ applies the command to each loop individually – playing loops stop, while stopped loops play. ‘All play’ means that in such a situation, all the targetted loops will play; ‘All stop’ means all the targetted loops will stop.
Controls wake UI	0	1	0		If on, activating any looper control wakes the UI and dismisses the screensaver, as if you’d moved an encoder/pot/button on the module itself.

Only record targets	0	1	0		If enabled, only loops set as targets are able to record. Other loops' inputs are muted.

Sync parameters

Name	Min	Max	Default	Unit	Description
Clock source	0	3	0		The clock source for synchronisation – one of 'None', 'Clock input', 'MIDI clock', or 'Loop 1'.
Clock input	1	64	3		The input bus to use as clock if the 'Clock source' is 'Clock input'.
MIDI divisor	0	19	9		The MIDI clock divisor to sync to, if the 'Clock source' is 'MIDI clock'.
MIDI numerator	1	16	1		A multiplier for the MIDI clock. For example, if your song is in 5/4 time, and you want to sync the loops to whole bars, you could set the divisor to 1/4 and the numerator to 5.
MIDI transport	0	1	0		If enabled, MIDI clock start will start (or retrigger) loop playback, and MIDI clock stop will stop playback.
Lock range	0	1000	0	ms	See 'Clocked operation', above.
2nd press cancels	0	1	0		If enabled, issuing a command a second time while the first command is still waiting for a clock will cancel the command.

Sync out parameters

Name	Min	Max	Default	Unit	Description
PPQN (out)	1	48	1		The number of pulses per quarter note for output clocks. Set this to '24' when generating MIDI clock.
Output to breakout	0	1	0		Enables MIDI output to the breakout.
Output to Select Bus	0	1	0		Enables MIDI output to the Select Bus.
Output to USB	0	1	0		Enables MIDI output to USB.
Output to internal	0	1	0		Enables internal MIDI output – that is, MIDI is sent to the other algorithms.

Per-loop parameters

Name	Min	Max	Default	Unit	Description
Channels	0	1	0		Selects stereo or mono operation.
Bit depth	0	2	0		Selects ‘32 bit (float)’, ‘24 bit’, or ‘16 bit’.
Crossfade time	0	1000	70		Sets the loop crossfade time. The scale is exponential, from 0.5ms to 5s.
Minimum layers	0	8	0		The minimum number of layers the loop should have, or ‘0’ for ‘Layers disabled’.
Current layer	1	999	1		The current layer for the loop.
1/4 notes (out)	1	128	1		Length of the loop in quarter notes. Used to determine the rate at which to output clocks.

Per-loop routing parameters

Name	Min	Max	Default	Unit	Description
Input left/mono	1	64	1		The input bus for the left (if stereo) or mono signal.
Input right	1	64	2		The input bus for the right signal, if stereo.
Output left/mono	1	64	13		The output bus for the left (if stereo) or mono signal.
Output right	1	64	14		The output bus for the right signal, if stereo.
Output mode	0	1	0		The standard Add/Replace mode selector as described above, for the loop outputs.
Trigger output	0	64	0		The output bus for an end-of-loop trigger pulse or clock.
Trigger mode	0	1	0		The standard Add/Replace mode selector as described above, for the trigger output.

MIDI parameters

Name	Min	Max	Default	Unit	Description
MIDI channel	0	16	0		The MIDI channel on which to receive notes.

Target 1	0	127	48		The MIDI note to automate ‘Target 1’.
Target 2	0	127	50		The MIDI note to automate ‘Target 2’.
Target 3	0	127	52		The MIDI note to automate ‘Target 3’.
Target 4	0	127	53		The MIDI note to automate ‘Target 4’.
Target next	0	127	54		The MIDI note to automate ‘Target next’.
Record	0	127	55		The MIDI note to automate ‘Record’.
Play	0	127	57		The MIDI note to automate ‘Play’.
Reverse	0	127	59		The MIDI note to automate ‘Reverse’.
Octave down	0	127	60		The MIDI note to automate ‘Octave down’.
Retrigger	0	127	58		The MIDI note to automate ‘Retrigger’.
Stop all	0	127	62		The MIDI note to automate ‘Stop all’.
Bypass clock	0	127	71		The MIDI note to automate ‘Bypass clock’.
Undo	0	127	72		The MIDI note to automate ‘Undo’.
Redo	0	127	74		The MIDI note to automate ‘Redo’.
Fade to clear	0	127	76		The MIDI note to automate ‘Fade to clear’.

Saving/loading loops

Functions for saving and loading audio to/from the MicroSD card are available through the ‘Looper’ menu.

Save loops

This menu saves the contents of any non-empty loops to the MicroSD card as WAV files.

A new folder with the name ‘saved <date> <time>’ is created within the top level ‘looper’ folder (which is created if it doesn’t already exist).

The WAV files are named ‘loop1.wav’, ‘loop2.wav’ etc. if the loop has no layers, or ‘loop1_layer01.wav’ etc. if there are layers.

Load loops

This menu loads WAV files from the MicroSD card. It expects to find them using a naming convention as used by the ‘Save loops’ menu.

The mono/stereo channel count and bit depth of the loops will be changed to match the loaded WAV files, but the algorithm will not be respecified if the number of loops doesn't match the number of saved loop files on the card – it will simply load as many as it can.

Looper (MicroSD)

“A MicroSD-based looper”

File format guid: 'lpsd'

Specifications:

- Loops, 1-4: The number of simultaneous loops.
- Layers, 1-8: The number of layers for each loop.

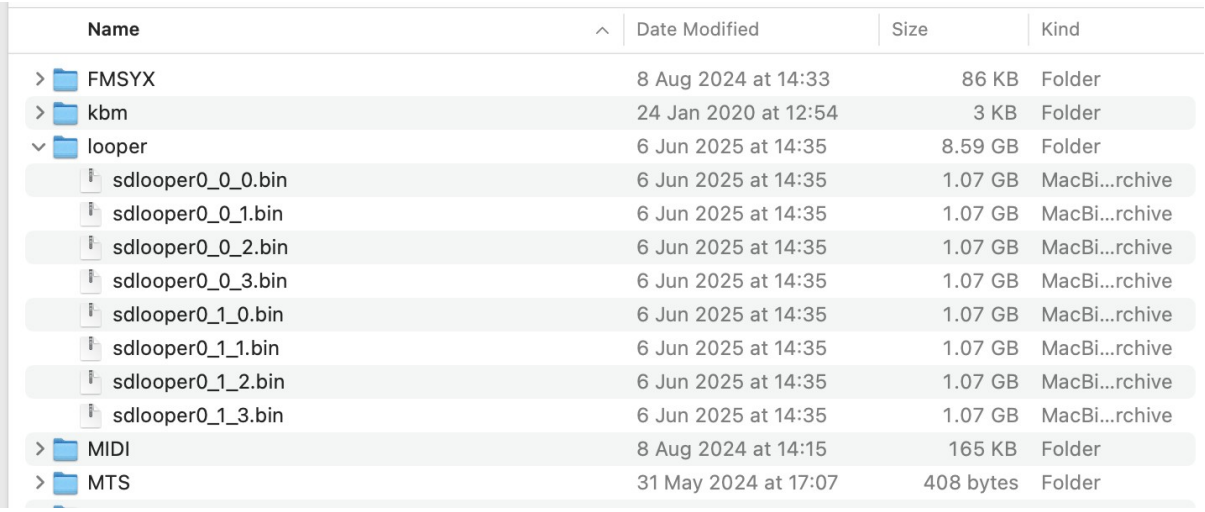
Description

This algorithm is a version of the Looper algorithm (above) which uses the MicroSD card for storage instead of the module’s memory. Consequently, the maximum loop times are much, much longer.

Operation of the algorithm is identical to that of the Looper. Please refer to the documentation above; this section will cover only the differences between the two algorithms.

MicroSD card usage

When the algorithm is instantiated, it creates files on the MicroSD card, one for each layer of each loop. These are located in the ‘looper’ folder on card.



Each file is (rather arbitrarily) 1GB in size, giving a maximum loop time of about 1.5 hours (using mono 32 bit float, 48kHz; other bit depths affect the time as for the regular Looper algorithm).

You will therefore want to have a suitable amount of free space on the card to use this algorithm.

To improve read/write performance, each file is also allocated as contiguous space on the card. This may not be possible if the card is badly fragmented.

It is recommended to use ‘exFAT’ formatting for cards to be used with this algorithm.

‘Busy’

Certain operations that are instant when using the regular Looper take effect after a small delay when using this algorithm, because data has to be read from the card before the operation can proceed. These operations are:

- Starting overdub into a new layer.
- Changing layer (including undo/redo).
- Switching into/out of reverse.

The first of these in particular (overdub into new layer) can take a while, and is dependent on the length of the loop.

During these delays, “Busy” is displayed in the UI, and for the overdub case, a percentage indicator is also shown.

Setting the number of layers

In the Looper algorithm, the maximum number of layers is calculated when a loop finishes recording, by dividing the available memory by the length of the loop. The ‘Minimum layers’ parameter sets the minimum number of layers required, or disables layers entirely.

In this algorithm, the number of layers is a specification, set when instantiating the algorithm (or when changed via the Respecify menu). Instead of ‘Minimum layers’, each loop has a ‘Number of layers’ parameter, which sets how many of the available layers it will actually use.

Setting this to ‘1’ effectively disables layers for this loop, which might be preferred if it’s important to avoid the delay associated with starting overdub into a new layer.

Lua Expression

“A single line of Lua”

File format guid: 'luae'

Specifications:

- Expressions, 1-8: The number of independent expressions to evaluate.

Description

This algorithm allows you to define an arbitrary CV or audio process by simply entering a line of text on the module, for example “a + b” to add two signals together.

It is designed to catch those little corner cases where stringing together, say, a bunch of attenuverters would be tedious, and writing a full blown Lua script seems like overkill (or, if you’re out and about without a computer on which to write the script).

For more extensive discussion of Lua on the disting NT, see the Lua Script algorithm (below) and the UI Scripts (below).

Expressions

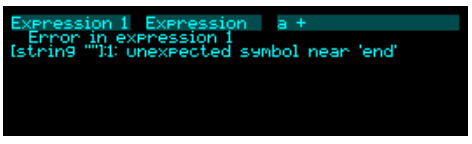
The expressions are interpreted as Lua. All built-in language features (e.g. simple maths +, -, *, /) and the math library (e.g. math.floor()) are available.

The inputs to the algorithm are represented by the variables a-h.

You may like to review the full set of button & encoder operations available when editing text parameters, above.

GUI

The display shows any errors encountered when parsing or running the expression, or “No errors” if everything is OK.



Globals parameters

Name	Min	Max	Default	Unit	Description
Input A-H	1	64			The bus to use for the correspondingly named expression variable.

Input A-H mode	0	1	0		Whether the input is interpreted as a voltage (a floating point number in the expression) or a logic input (a boolean value in the expression).

Per-expression parameters

Name	**Min**	**Max**	**Default**	**Unit**	**Description**
Output	1	64	15		The bus to use for the expression output.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.
Enable	0	1			Enables evaluation of the expression.
Expression					The expression text.
Rate	0	1	0		Whether the expression is evaluated at control rate or audio rate.

Lua Script

“Algorithm defined by a Lua script”

File format guid: 'lua '

Specifications: None

Description

This algorithm allows you to run what are effectively scripted “plug-ins”, loaded from the MicroSD card.

Such scripts are ideal for CV processing, especially for quirky little sequencers or generative things that are never going to be implemented as official algorithms because they’re so specific to your particular use case.

Scripts are not intended for audio processing. If you want to implement custom DSP code, look at the Three Pot algorithm (below) or the C++ API (above).

Installing scripts

Scripts are text files with the extension ‘.lua’. They should be placed on the MicroSD card in the folder `/programs/lua`

One level of subdirectories is allowed.

Where to get scripts

Some example scripts are included in the GitHub repository[87]. You can download the .lua files and copy them to your MicroSD card. Some of these may already be on the card that came with the module.

We have an active community on our Discord server, which has a dedicated channel for Lua scripting here[88].

Writing scripts

Further documentation of Lua scripting on the disting NT can be found in the separate Lua scripting document, which can be found on the firmware download page (here[89]), right next to the download link for this manual.

87 https://github.com/expertsleepersltd/distingNT/tree/main/lua

88 https://discord.com/channels/1268195809502036049/1331216813194739804

89 https://expert-sleepers.co.uk/distingNTfirmwareupdates.html

Program parameters

Most of the parameters here will be defined by the script.

Name	Min	Max	Default	Unit	Description
Program	0	999	0		Selects the program to run.

Routing parameters

These are entirely defined by the script, and relate the script’s inputs and outputs to the system busses.

Macro Oscillator

“It's Braids!”

File format guid: 'maco'

Specifications: None

Description

This algorithm is an implementation of the open source Braids[90] module by Émilie Gillet.

This implementation should behave and sound identical to the original, with the caveat that the original module runs at 96kHz – for a totally accurate recreation, you would need to run the disting NT at 96kHz. Any demos or tutorials you may find online should apply just as well to this version. The user manual for the original module is here[91]. The documentation below assumes some familiarity with this.

META mode

The original Braids has a separate mode in which the FM CV controls the choice of synthesis model. The disting NT doesn't require a special mode for this since you can use Mappings to control the model parameter however you like.

Braids parameters

Name	Min	Max	Default	Unit	Description
Model	0	46	0		Chooses the synthesis model.
Timbre	0	1024	0		Sets the timbre control.
Colour	0	1024	0		Sets the colour control.
AD Attack	0	15	0		Sets the envelope attack time.
AD Decay	0	15	5		Sets the envelope decay time.
AD FM	0	15	0		Sets the amount by which the envelope affects the FM.
AD Timbre	0	15	0		Sets the amount by which the envelope affects the timbre.
AD Colour	0	15	0		Sets the amount by which the envelope affects the colour1.

90 https://github.com/pichenettes/eurorack/tree/master/braids

91 https://pichenettes.github.io/mutable-instruments-documentation/modules/braids/manual/

AD VCA	0	1	0		Sets whether the envelope is applied to the volume of the output.
Coarse	-48	48	0	ST	Coarse tuning control.
Fine	-100	100	0	cents	Fine tuning control.
Scale	0	48	0		Sets the quantizer scale.
Root	0	11	0		Sets the quantizer root note.

Tweaks parameters

Some of these are the parameters hidden behind the original Braids' menu system. Descriptions in quotation marks are from the Braids manual.

Name	Min	Max	Default	Unit	Description
Resolution	0	6	6		Sets the output bit depth.
Sample rate	0	6	6		Sets the sample rate decimation.
Trigger source	0	1	0		Enables automatic triggering on pitch changes.
Trigger delay	0	6	1		Sets the trigger delay.
Flatten	0	1	0		"applies a detuning in the lower and higher frequencies, to recreate some of the tuning imperfections of VCOs."
Drift	0	4	0		"recreates the drifting of a disastrously designed VCO."
Signature	0	4	0		"applies grungy glitches/waveform imperfections to the output signal. The exact behavior of this option is unique to each module built."
Gain	-40	12	0	dB	Sets an output gain.

Inputs parameters

Name	Min	Max	Default	Unit	Description
Trigger input	0	64	0		Selects which bus is used as the trigger input.
Pitch input	0	64	0		Selects which bus is used as the pitch input.

FM input	0	64	0		Selects which bus is used as the FM input.
Timbre input	0	64	0		Selects which bus is used as the timbre input.
Colour input	0	64	0		Selects which bus is used as the colour input.

Outputs parameters

Name	Min	Max	Default	Unit	Description
Output	1	64	13		The bus for the output signal.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Control parameters

Name	Min	Max	Default	Unit	Description
MIDI mode	0	3	0		Controls the algorithm's response to MIDI. See below.
I2C mode	0	3	0		Controls the algorithm's response to I2C. See below.
MIDI channel	0	16	1		The MIDI channel to receive on.
I2C channel	0	255	1		The I2C channel to receive on.

MIDI and I2C modes

The MIDI mode and I2C mode parameters allow you to choose how MIDI and I2C will be used. Both have the same options:

Off	Notes will not be used.
Pitch	Note on events will set the pitch.
Trigger	Note on events will cause a trigger.
Pitch & trigger	Note on events will set the pitch and cause a trigger.

Macro Oscillator 2

“It's Plaits!”

File format guid: 'mac2'

Specifications: None

Description

This algorithm is an implementation of the open source Plaits[92] module by Émilie Gillet.

This implementation should behave and sound identical to the original. Any demos or tutorials you may find online should apply just as well to this version. The user manual for the original module is here[93]. The documentation below assumes some familiarity with this.

This version of the disting NT firmware is based on v1.2 of the Plaits code.

Input normalization

The original Plaits module made extensive use of detecting which inputs had jacks plugged in and adjusting its behaviour accordingly. For example, if nothing is plugged into the Trigger input, the envelope is not used and the module constantly emits sound.

The disting NT has no way of knowing whether its sockets are connected or not, so this algorithm relies on the various 'input' settings being enabled or not. To take the same example, if the 'Trigger input' setting is set to ‘None’, the algorithm will generate constant sound; if the setting is something else, the algorithm assumes you're going to supply triggers on the input you choose.

Outputs

As on the original Plaits, there are two outputs, ‘Main’ and ‘Aux’. If you select the same bus for both outputs they are mixed.

Plaits parameters

Name	Min	Max	Default	Unit	Description
Model	0	23	0		Chooses the synthesis model.
Coarse tune	-60	60	0	ST	Coarse tuning control.
Fine tune	-100	100	0	cents	Fine tuning control.
Harmonics	0	127	64		Sets the harmonics control.

92 https://github.com/pichenettes/eurorack/tree/master/plaits

93 https://pichenettes.github.io/mutable-instruments-documentation/modules/plaits/manual/

Timbre	0	127	64		Sets the timbre control.
Morph	0	127	64		Sets the morph control.
FM	-100	100	0	%	Sets the FM depth.
Timbre mod	-100	100	0	%	Sets the timbre modulation depth.
Morph mod	-100	100	0	%	Sets the morph modulation depth.
Low-pass gate	0	127	127		Sets the LPG colour (VCFA-VCA blend).
Time/decay	0	127	64		Sets the envelope decay time.

Inputs parameters

Name	Min	Max	Default	Unit	Description
Trigger input	0	64	0		Selects which bus is used as the trigger input.
Level input	0	64	0		Selects which bus is used as the level input.
CV input	0	64	0		Selects which bus is used as the pitch CV input.
FM input	0	64	0		Selects which bus is used as the FM input.
Harmonics input	0	64	0		Selects which bus is used as the harmonics input.
Timbre input	0	64	0		Selects which bus is used as the timbre input.
Morph input	0	64	0		Selects which bus is used as the morph input.

Outputs parameters

Name	Min	Max	Default	Unit	Description
Main output	0	64	13		The bus for the ‘Main’ output.
Aux output	0	64	0		The bus for the ‘Aux’ output.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Main gain	-40	6	0	dB	The level of the ‘Main’ output signal.
Aux gain	-40	6	0	dB	The level of the ‘Aux’ output signal.

Control parameters

Name	**Min**	**Max**	**Default**	**Unit**	**Description**
MIDI mode	0	3	0		Controls the algorithm's response to MIDI. See below.
I2C mode	0	3	0		Controls the algorithm's response to I2C. See below.
MIDI channel	0	16	1		The MIDI channel to receive on.
I2C channel	0	255	1		The I2C channel to receive on.

MIDI and I2C modes

The MIDI mode and I2C mode parameters allow you to choose how MIDI and I2C will be used. Both have the same options:

Off	Notes will not be used.
Pitch	Note on events will set the pitch.
Trigger	Note on events will cause a trigger.
Pitch & trigger	Note on events will set the pitch and cause a trigger.

Note that in terms of the input normalization logic described above, activating MIDI or I2C for pitch and/or trigger is taken to be equivalent to connecting a CV input for that parameter.

MIDI CC Generator

“Generates MIDI CCs”

File format guid: 'micc'

Specifications:

- CCs, 1-8: The number of CCs to generate.

Description

This simple utility generates MIDI CCs. These can be sent to the various external MIDI destinations, and/or routed to other algorithms within the module. Some example uses include:

- CV to MIDI conversion: by mapping the Value parameter to a CV, you can generate a MIDI CC from a control voltage.
- Macro control: you can send the CCs from this algorithm to other algorithms, and map parameters in those algorithms to be controlled by the CC. Then, by changing the Value parameter in this algorithm, you can change many parameters in other algorithms at once. This is particularly useful in combination with the Performance Page.

Globals parameters

Name	Min	Max	Default	Unit	Description
Output to breakout	0	1	0		Enables MIDI output to the breakout.
Output to Select Bus	0	1	0		Enables MIDI output to the Select Bus.
Output to USB	0	1	0		Enables MIDI output to USB.
Output to internal	0	1	0		Enables internal MIDI output – that is, MIDI is sent to the other algorithms.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Enable	0	1	0		Enables the CC generator.
MIDI channel	1	16	1		The MIDI channel for the CC.
MIDI CC	0	127	1		The CC number to generate.

CC	0	127	0		The CC value.
Name					A text name for the channel for identification.
Mode	0	2	0		The type of CC to send – one of ‘7 bit’, ‘14 bit - low first’, or ‘14 bit - high first’. Note that for 14 bit CCs, the CC number should be in the range 0-31.

MIDI Filter

“Filters MIDI messages”

File format guid: 'mifi'

Specifications: None

Description

This algorithm is unusual inasmuch as it neither processes nor generates audio or CVs. Instead it filters and/or modifies MIDI on its way to the algorithms in the preset.

Much like the way in which audio flows through the algorithms (via the busses) from top to bottom, any MIDI that the module receives is processed by the algorithms in order. This algorithm sits in the preset and affects the MIDI seen by the algorithms that follow it. (It has no affect on the MIDI seen by algorithms that precede it in the preset.)

Processing can be specified independently for each of the 16 MIDI channels. The various types of MIDI message (notes, controllers, etc.) can be passed through unaltered, suppressed (filtered out), or they may have their MIDI channel modified.

Min/max notes

For MIDI messages which contain a note number (that is, note on, note off, and poly pressure), the Min/Max note parameters are used.

If the min note is less than the max note, only notes in the range from the min note (inclusive) up to the max note (exclusive) are affected by the filter. For example if min is 48 and max is 60, notes in the range 48-59 are affected and other notes are not.

If the max note is less than the min note, notes in the range from the max note (inclusive) up to the min note (exclusive) are unaffected by the filter, while all other notes are. For example, if max is 48 and min is 60, notes in the range 48-59 are unaffected; notes from 0 to 47 and from 60 to 127 are affected.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Enable	0	1	0		Enables processing on this MIDI channel.
Note off	0	18	0		Sets what should be done with note off messages.
Note on	0	18	0		Sets what should be done with note on messages.

Poly pressure	0	18	0		Sets what should be done with poly pressure (aka polyphonic aftertouch) messages.
CC	0	18	0		Sets what should be done with CC (continuous controller) messages.
Program change	0	18	0		Sets what should be done with program change messages.
Channel pressure	0	18	0		Sets what should be done with channel pressure (aka aftertouch) messages.
Pitch bend	0	18	0		Sets what should be done with pitch bend messages.
Min note	0	128	0		Sets the minimum MIDI note for which the filter applies.
Max note	0	128	128		Sets the maximum MIDI note for which the filter applies.

MIDI Player

“Plays MIDI files from the MicroSD card”

File format guid: 'midp'

Specifications: None

Description

This algorithm plays standard MIDI files, outputting the MIDI messages, and converting the MIDI to CVs and gates. It will use an internal clock (using the file's tempo, if specified), or sync to analogue clocks or incoming MIDI clock.

See the MicroSD card section above for information on supported MIDI file formats, and how to arrange the files on the card.

Optionally, this algorithm will also process live MIDI.

While the MIDI/CV converters built into this algorithm are convenient for simple monophonic use, you may like to pair this algorithm with Poly CV (below) for more flexibility.

MIDI file limitations

This algorithm will use the first 24 tracks of each MIDI file. Tracks beyond that will be ignored.

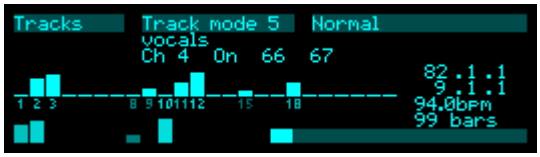
Looping playback

By default all MIDI files loop.

To prevent a file from looping, include “_ONESHOT” in its filename.

GUI

On the right side, the display shows the current transport position (in bars, beats, and sixteenths) and the current position within the file (which would be different if, say, the file had looped, or you select a new file to play without stopping the transport). Below that is the file’s tempo (if it defines one) and the file’s length in bars. At the bottom right is a simple progress bar, indicating the playback position within the file.



There are two lines of activity indicators. The upper one represents the tracks in the file; the lower one represents MIDI channels.

If the current parameter is one of the ‘Track mode’ parameters, further information about that track is shown: the track name (if defined in the file) and the most recent MIDI message played on that track.

Transport parameters

Name	Min	Max	Default	Unit	Description
Folder					Chooses the folder of MIDI files on the card.
File					Chooses the MIDI file within the folder.
Play	0	1	0		Starts/stops playback using the internal timebase.
Start from bar	1	256	1		Sets the bar number within the file from which to start playback.
File change	0	2	0		Sets the behaviour when a new file is chosen. The options are: ● Wait until the end of the bar and then change to the new file. ● Wait until the end of the file and then change to the new file. ● Change immediately.
File restart	0	1	0		Sets how the file position changes when switching to a new file. The options are: ● The new file starts from the beginning. ● The new file starts from the current position in the old file (modulo the length of the new file).

Tracks parameters

Name	Min	Max	Default	Unit	Description
Track mode 1-24	0	1	0		Set the playback mode for the tracks. The options are: ● Normal playback. ● Track is muted.

MIDI/CV parameters

Name	Min	Max	Default	Unit	Description
1-4 MIDI channel	1	16	1-4		MIDI channel for the MIDI/CV converter.
1-4 Pitch output	0	64	0		Bus for the MIDI/CV converter pitch output.

1-4 Gate output	0	64	0		Bus for the MIDI/CV converter gate output.
Drum first note	0	127	36		The first (lowest) note for the drum converter.
Drum channel	0	16	0		MIDI channel for the drum converter.
Trigger length	1	100	10	ms	Sets the length of output trigger pulses.
Convert live MIDI	0	1	0		If enabled, MIDI sent to the algorithm is converted by its MIDI/CV converters as if it originated in the MIDI file.

Clock parameters

Name	Min	Max	Default	Unit	Description
Clock multiplier	0	4	0		Sets the multiplier for incoming clock pulses. The options are: ● clock is interpreted as 24ppqn. ● clock is interpreted as 1/32 notes. ● clock is interpreted as 1/16 notes. ● clock is interpreted as 1/8 notes. ● clock is interpreted as 1/4 notes.
Clock input	0	64	0		The bus to use as the clock input.
Reset input	0	64	0		The bus to use as the reset input.
Reset mode	0	1	0		Sets the mode for the reset input. The options are: ● the input is a reset trigger. ● the input is a run/stop signal.
Follow MIDI clock	0	1	0		Sets whether the algorithm follows incoming MIDI clock.
Output MIDI clock	0	1	0		Set whether the algorithm outputs MIDI clock.

Outputs parameters

Name	Min	Max	Default	Unit	Description

Output to breakout	0	1	0		Enables MIDI output to the breakout.
Output to Select Bus	0	1	0		Enables MIDI output to the Select Bus.
Output to USB	0	1	0		Enables MIDI output to USB.
Output to internal	0	1	0		Enables internal MIDI output – that is, MIDI is sent to the other algorithms.

MIDI Triggers

“Generates triggers from MIDI/I2C”

File format guid: 'midt'

Specifications:

- Channels, 1-12: The number of triggers to generate.

Description

This algorithm generates gates and triggers from MIDI or I2C note messages.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Enable	0	1	0		Enables the trigger.
Mode	0	1	0		Sets whether to generate a trigger from note on messages, or a gate from note on/off messages.
Length	1	100	10	ms	The length of the trigger pulse.
Level	-10.0	10.0	5.0	V	The magnitude of the trigger/gate.
Velocity	0	1	0		Sets whether to apply velocity scaling to the trigger/gate magnitude.
Min level	-10.0	10.0	1.0	V	If applying velocity scaling, the voltage corresponding to minimum velocity.
MIDI channel	0	16	0		The MIDI channel, or ‘Off’.
I2C channel	0	255	0		The I2C channel, or ‘Off’.
Note number	-1	127	-1		The note number to respond to, or ‘Any’.
Output	1	64	15		The output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.
ES-5 Expander	0	6	0		If set, the ES-5 expander header to use for output instead of the bus specified by the Output parameter. “1” means the ES-5 itself.

ES-5 Output	1	8	1		If using an ES-5 expander output, sets which of the 8 outputs on the expander to use.

Minimum/maximum

“Min/max, or half-wave rectifier”

File format guid: 'mima'

Specifications:

- Channels, 1-12: The number of bus channels to process.

Description

This simple algorithm computes the minimum and maximum of two input voltages.

Note that with one input disabled, or at 0V, this effectively performs half-wave rectification.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Input A	0	64	1		One of the two input busses to process.
Input B	0	64	2		One of the two input busses to process.
Min output	0	64	0		The output bus for the minimum voltage.
Min output mode	0	1	0		The standard Add/Replace mode selector as described above for the minimum voltage.
Max output	0	64	0		The output bus for the maximum voltage.
Max output mode	0	1	0		The standard Add/Replace mode selector as described above for the maximum voltage.

Minky Starshine

“An additive polysynth”

File format guid: 'mist'

Specifications:

- Voices: 1-16: The number of simultaneous voices.

Description

This algorithm is a version of the very early (2007) Expert Sleepers VST plug-in of the same name, which you can find here[94]. From that page:

> Minky Starshine is at heart an additive synthesiser, augmented with possibilities for subtractive synthesis (i.e. filtering) and pulse width modulation.
>
> A waveform is created by summing 16 independently controllable 'partials'. By setting the relative levels of the partials appropriately you can create familiar waveforms such as sawtooth and squarewave, but you can also create an infinite range of timbres in between.
>
> As well as simply setting the levels of the partials, you can flexibly assign them to three 'groups'. Each group has a very flexible envelope generator and an LFO, so you can change the balance of the partials (and therefore the timbre of the sound) over time.
>
> The LFOs have a wide frequency range and can also be used as ring modulators.
>
> The waveform as a whole can have its pulsewidth controlled. An envelope and LFO is provided, as well as control by note velocity.
>
> A resonant filter is available, capable of lowpass, bandpass and highpass and a continuous range in between. Again, control by envelope, LFO and note velocity are possible.

Please refer to the section “Common polysynth features” above.

The VST version includes a modulation post-effect – this is available on the disting NT as a separate algorithm, More Like Space (below).

Partials and groups

The basis of every sound in Minky Starshine is a set of 16 sine wave oscillators ('partials') which all sound at multiples of the root frequency.

94 https://www.expert-sleepers.co.uk/minkystarshine.html

The level parameters let you set the level of each partial. The centre value of the slider is zero – values below zero effectively reverse the phase of the oscillator.

Each partial can be assigned to one of four groups – 1, 2, 3, or 4. If a partial is in group 1, 2, or 3 its amplitude is affected by the envelope with the same numbering. Partials in group 4 are only affected by the main amplitude envelope. There is also an LFO per group.

LFOs

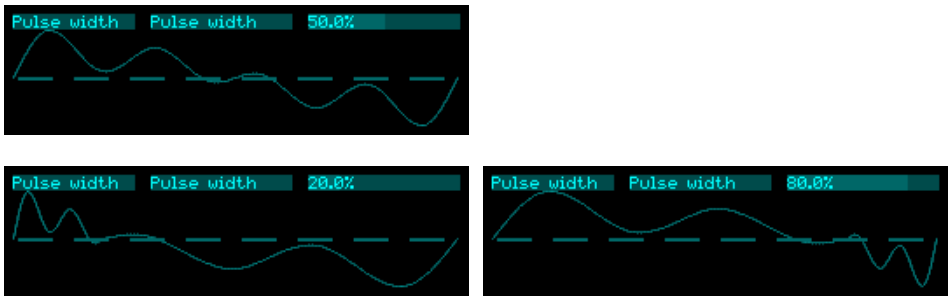
The four LFOs each have controls for speed, offset and amount. The resultant LFO waveform is multiplied with the group envelope and partial level to give the overall level of a partial.

The LFO offset sets the level about which the LFO oscillates. The LFO amount controls how far around this value it oscillates.

For example, if the offset is set to 100%, then the LFO modulates the partial level around its nominal (unmodulated) value. On the other hand, if the offset is set to 0%, then the LFO signal itself is being multiplied by the partial, which is effectively ring modulation. This is why the LFOs have such a wide frequency range – because the higher frequencies are useful for ring modulation effects.

Pulse width

The pulse width controls allow you to adjust the shape of the whole waveform, which can result in extreme timbral modulation.



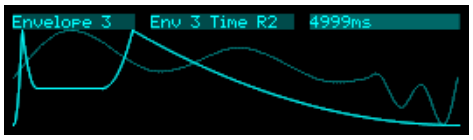
There is an LFO dedicated to modulating the pulse width. There is also an ADSR envelope, the depth of which is controlled with the ‘PW Env Amount’ parameter.

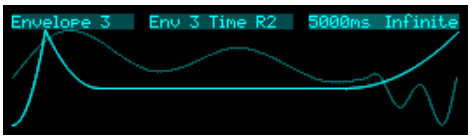
Envelopes

The overall amplitude envelope, and the filter and pulse width envelopes, are simple ADSR shapes.

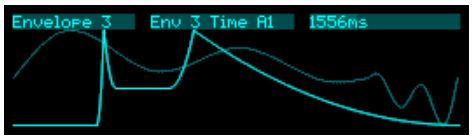
The three group envelopes are more complex, multi-stage affairs.

The ‘Time R2’ parameters have a special case when set to their maximum value – the release time becomes ‘Infinite’. When this setting is made, the envelope stops at the point of level R1, and does not decay to zero. This is useful when you want the group envelope to remain fixed during the release, and let the main overall ADSR envelope control the release sound.





When level A1 is set to zero, time A1 effectively gives you control over a time delay before the envelope kicks in.



Filter

The filter section is applied after the envelopes and LFOs. The filter 'type' control lets you smoothly choose between no filtering, lowpass filtering, bandpass filtering and highpass filtering.

The filter cutoff and resonance ('Q') controls function exactly as you would expect.

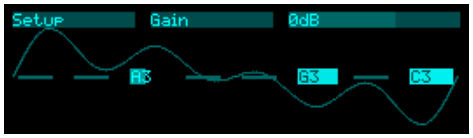
The filter keyboard tracking control sets how much the filter frequency is affected by the note being played. With this control at 'zero', the filter frequency does not depend on the note. With this control at 'one', the filter frequency rises or falls by an octave per octave that the note varies from middle 'C' (MIDI note number 60).

Like the pulsewidth section, the filter has an LFO and an envelope, and the controls are much the same. 'Filter LFO Amount' sets the amount by which the LFO modulates the filter cutoff; 'Filter LFO Type Amount' sets the amount by which the LFO modulates the filter type.

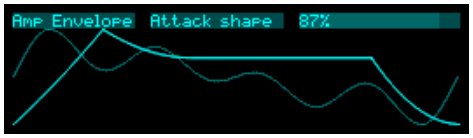
GUI

The display varies depending on the current parameter being edited. In all cases, a waveform view is drawn in the background, to aid in visualisation of the timbre being created.

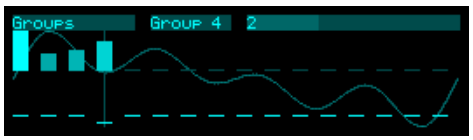
By default the display shows the notes being played by the various voices, each superimposed on a bar to indicate the note envelope.



When an envelope is being edited, the display shows the envelope curve.



When the levels and groups are being edited, the display shows the levels as vertical bars, and the groups as small lines below. The current partial is indicated by a vertical line.



Setup parameters

Name	Min	Max	Default	Unit	Description
Transpose	-60	60	0	ST	Adjusts the tuning in semitones.
Fine tune	-100	100	0	cents	Adjusts the tuning in cents.
Bend range	0	48	2		The MIDI pitch bend range.
Sustain mode	0	1	0		The standard polysynth sustain mode parameter. See above.
Sustain	0	1	0		Directly controls the sustain (like a MIDI sustain pedal).
Quality	0	1	1		Sets the quality of the sine wave approximation used for the partials. Low quality uses less CPU.
Gain	-40	24	0	dB	Applies an overall output gain.

Routing parameters

Name	Min	Max	Default	Unit	Description
Left/mono output	1	64	13		The left or mono output bus.
Right output	0	64	0		The right output bus.
MIDI channel	0	16	1		The MIDI channel to listen on.
MPE channels	1	16	1		Controls how the algorithm will respond to MPE. See above.

I2C channel	0	255	1		Sets the I2C channel.

Levels parameters

Name	Min	Max	Default	Unit	Description
Level 1	-100.0	100.0	100.0	%	The amplitude of the first partial (the fundamental).
Level 2-16	-100.0	100.0	0.0	%	The amplitude of partials 2-16.

Groups parameters

Name	Min	Max	Default	Unit	Description
Group 1-16	1	4	4		The group for the respective partial. Groups 1-3 are affected by envelopes 1-3 and LFOs 1-3 respectively. Group 4 is affected by LFO 4.

Amp Envelope parameters

Name	Min	Max	Default	Unit	Description
Attack	0	5000	500	ms	The amplitude envelope attack time.
Decay	0	5000	500	ms	The amplitude envelope decay time.
Release	0	5000	500	ms	The amplitude envelope release time.
Attack shape	0	100	0	%	The shape of the attack stage of the amplitude envelope.
Decay shape	0	100	0	%	The shape of the decay stage of the amplitude envelope.
Release shape	0	100	0	%	The shape of the release stage of the amplitude envelope.
Sustain	0.0	100.0	70.0	%	The amplitude envelope sustain level.

Envelope 1-3 parameters

The three multi-stage group envelopes have the same parameters, as follows.

Name	Min	Max	Default	Unit	Description
Time A1	0	5000	0	ms	The time for the first attack stage.
Time A2	0	5000	0	ms	The time for the second attack stage.

Time A3	0	5000	0	ms	The time for the third attack stage.
Time R1	0	5000	0	ms	The time for the first release stage.
Time R2	0	5000	5000	ms	The time for the second release stage.
Shape A1	0	100	0	%	The shape of the A1 stage.
Shape A2	0	100	0	%	The shape of the A2 stage.
Shape A3	0	100	0	%	The shape of the A3 stage.
Shape R1	0	100	0	%	The shape of the R1 stage.
Shape R2	0	100	0	%	The shape of the R2 stage.
Level A1	0.0	100.0	0.0	%	The level at the end of the A1 stage.
Level A2	0.0	100.0	100.0	%	The level at the end of the A2 stage.
Sustain	0.0	100.0	100.0	%	The sustain level.
Level R1	0.0	100.0	100.0	%	The level at the end of the R1 stage.

LFO 1-4 parameters

The four LFOs have the same parameters, as follows.

Name	**Min**	**Max**	**Default**	**Unit**	**Description**
Speed	0	1000	500		The LFO speed. Exponential scale from 0.05Hz to 20Hz.
Offset	0.0	100.0	100.0	%	The LFO offset – the value around which the LFO oscillates. At 100% this is a normal ‘tremolo’ effect; at 0% the LFO becomes a ring modulator.
Amount	0.0	100.0	0.0	%	The depth of the LFO around the offset.
KB Track	0.0	100.0	0.0	%	Keyboard tracking – the amount by which the note pitch affects the LFO speed.
Multiplier	0	2	0		A multiplier for the LFO speed: 1x, 10x, or 100x.

Pulse width parameters

Name	**Min**	**Max**	**Default**	**Unit**	**Description**
Pulse width	0.0	100.0	50.0	%	Sets the pulse width.

PW LFO Speed	0	1000	500		Sets the pulse width LFO speed. Exponential scale from 0.05Hz to 20Hz.
PW LFO Amount	0.0	100.0	0.0	%	Sets the amount of pulse width LFO modulation.
PW Env Attack	0	5000	1000	ms	Sets the pulse width envelope attack time.
PW Env Decay	0	5000	1000	ms	Sets the pulse width envelope decay time.
PW Env Release	0	5000	1000	ms	Sets the pulse width envelope release time.
PW Env Sustain	0.0	100.0	70.0	%	Sets the pulse width envelope sustain level.
PW Env Amount	-100.0	100.0	0.0	%	Sets the pulse width envelope modulation amount.

Filter parameters

Name	**Min**	**Max**	**Default**	**Unit**	**Description**
Cutoff	0	1000	500		Sets the filter cutoff frequency. Exponential scale from 50Hz to 12kHz.
Q	1.0	100.0	10.0	%	Sets the filter resonance.
Type	0	400	0		Sets the filter type. Smoothly transitions from no filter, to lowpass, to bandpass, to highpass, and back to no filter.
LFO Speed	0	1000	500		Sets the filter LFO speed. Exponential scale from 0.05Hz to 20Hz.
LFO Amount	0.0	300.0	0.0	%	Sets the amount of filter LFO modulation.
LFO Type Amount	0	400	0	%	Sets the amount of filter type LFO modulation.
Env Attack	0	5000	1000	ms	Sets the filter envelope attack time.
Env Decay	0	5000	1000	ms	Sets the filter envelope decay time.
Env Release	0	5000	1000	ms	Sets the filter envelope release time.
Env Sustain	0.0	100.0	70.0	%	Sets the filter envelope sustain level.

Env Amount	-100.0	100.0	0.0	%	Sets the filter envelope modulation amount.
KB Track	0.0	100.0	0.0	%	Sets the amount of filter keyboard tracking.

Modulation parameters

Name	Min	Max	Default	Unit	Description
Vel Env Amount	-100.0	100.0	0.0	%	Sets how velocity affects the overall amplitude of the note. When set to zero, all notes play at full volume regardless of velocity.
Vel Filter Amount	-800.0	800.0	0.0	%	Sets how velocity affects the filter cutoff.
Vel Filter Env Amount	-100.0	100.0	0.0		Sets how velocity affects the depth of the filter envelope.
Vel PW Amount	-100.0	100.0	0.0		Sets how velocity affects the pulse width.
Vel PW Env Amount	-100.0	100.0	0.0		Sets how velocity affects the depth of the pulse width envelope.

Microtuning parameters

The algorithm uses the standard polysynth microtuning parameters, as described above.

Chord/arp parameters

The algorithm uses the standard polysynth chord and arpeggiator parameters, as described above.

CV/gate parameters

The algorithm uses the standard polysynth CV/gate parameters, as described above.

Mixer Mono

“A mono mixer”

File format guid: 'mix1'

Specifications:

- Channels, 1-12: The number of mixer channels.
- Sends, 0-4: The number of aux sends per channel.

Description

This algorithm is a mono mixer - it mixes mono inputs to a mono output. Up to four aux send busses are available, which can be switched to pre- or post-fade.

GUI

The display shows a level meter for each channel. The thin line at the top indicates 0dBFS; if a channel goes over this, a clip indicator will show (a small rectangle over the 0dBFS line).

The chevrons to the right of the level meters indicate the mixer gains.

Mute and Solo are indicated with a small ‘M’ and ‘S’ respectively. Muted level meters are grayed out.

Common parameters

Name	Min	Max	Default	Unit	Description
Output	0	64	13		The mixer output bus.
Duplicate output	0	64	0		An optional duplicate output. Convenient if you want to drive a stereo bus from a mono mix.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.
Output gain	-70.0	6.0	0.0	dB	The mixer output gain.

Per-send parameters

Name	Min	Max	Default	Unit	Description
Destination	1	64	13		The output bus for the send.
Pre/post	0	1	1		Whether the send is Pre-fade or Post-fade.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Input	0	64	1		The channel input bus.
Gain	-70.0	6.0	-70.0	dB	The channel gain.
Mute	0	1	0		Mutes the channel.
Solo	0	1	0		Solos the channel.
Name					A text name for the channel for identification.

Per-channel per-send parameters

Name	Min	Max	Default	Unit	Description
Send gain	-70.0	6.0	-70.0	dB	The send gain.

Mixer Stereo

“A stereo mixer”

File format guid: 'mix2'

Specifications:

- Channels, 1-12: The number of mixer channels.
- Sends, 0-4: The number of aux sends per channel.

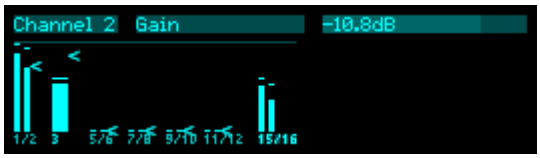
Description

This algorithm is a stereo mixer – it mixes mono or stereo inputs to a stereo output. Up to four aux send busses are available, which can be switched to pre- or post-fade. Sends may also be set to be mono or stereo.

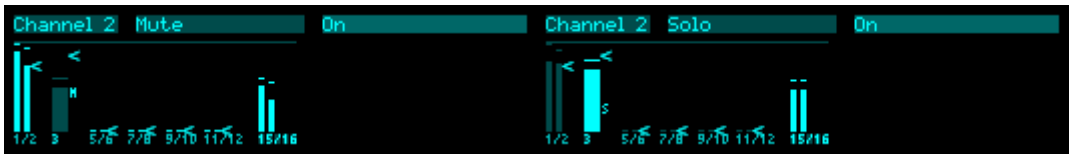
GUI

The display shows a level meter for each channel. The thin line at the top indicates 0dBFS; if a channel goes over this, a clip indicator will show (a small rectangle over the 0dBFS line).

The chevrons to the right of the level meters indicate the mixer gains.



Mute and Solo are indicated with a small ‘M’ and ‘S’ respectively. Muted level meters are grayed out.



Common parameters

Name	Min	Max	Default	Unit	Description
Left output	0	64	13		The left output bus.
Right output	0	64	14		The right output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.
Output gain	-70.0	6.0	0.0	dB	The mixer output gain.

Per-send parameters

Name	Min	Max	Default	Unit	Description
Destination	1	64	13		The output bus for the send.
Pre/post	0	1	1		Whether the send is Pre-fade or Post-fade.
Width	0	1	0		Whether the send is mono or stereo. Stereo sends use the destination bus and the next one up.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Input left/mono	0	64	1		The channel left input bus.
Input right	0	64	1		The channel right input bus (if stereo).
Gain	-70.0	6.0	-70.0	dB	The channel gain.
Pan	-100	100	0	%	The channel stereo pan.
Mute	0	1	0		Mutes the channel.
Solo	0	1	0		Solos the channel.
Name					A text name for the channel for identification.

Per-channel per-send parameters

Name	Min	Max	Default	Unit	Description
Send gain	-70.0	6.0	-70.0	dB	The send gain.

More Like Space

“Modulation Effect”

File format guid: 'mols'

Specifications: None

Description

This algorithm is a modulation effect of the chorus/flange/stereo-widening sort. It originates in the VST version of the Minky Starshine instrument, which is also present as a disting NT algorithm (above).

The signal is fed into a number of delays in parallel. The 'Voices' parameter sets the number of delays – a higher number of voices tends to give a lusher sound. The delay time of the delays is each modulated by an LFO.

Effect parameters

Name	Min	Max	Default	Unit	Description
Mix	0	100	100	%	The wet/dry mix.
Level	-40	0	0	dB	The gain applied to the effect/wet signal.
Delay	0.0	500.0	10.0	ms	Sets the minimum delay time.
Sweep	0.0	50.0	1.0	ms	Sets the amount of modulation of the delay time by the LFO (added to the minimum).
Rate	0	1000	500		Sets the LFO speed. Exponential scale from 0.05Hz to 20Hz.
Feedback	-100.0	100.0	0.0	%	Introduces feedback into the delay lines, for sustained echos or deeper flanging effects.
Spread	0.0	100.0	50.0	%	Controls the amount of phase variance between the LFOs of the voices. E.g. two voices with a spread of 50% gives the classic quadrature phase stereo chorus effect.
Voices	1	4	2		The number of voices (delay lines).

Shape	0.0	100.0	0.0	%	Applies waveshaping to the LFO waveform, which has a corresponding effect on the nature of the pitch alterations induced by the modulation. It's hard to describe exactly how it affects the sound – experiment with it! For classic chorus and flange effects, leave this setting at zero.

Routing parameters

Name	Min	Max	Default	Unit	Description
Input	1	64	1		The input bus.
Left output	1	64	13		The left output bus.
Right output	1	64	14		The right output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Multi-Switch

“Sequential or CV-controlled switch”

File format guid: 'musw'

Specifications:

- Switches, 1-6: The number of switches.

Description

This algorithm is based on the disting EX algorithm of the same name. You may like to review the video on that algorithm here[95].

It offers up to six highly configurable sequential or voltage controlled switches. Being DC-coupled, it can switch audio or CVs.

Each switch consists of two sub-switches: an input sub-switch, and an output sub-switch. The selected input is routed to the selected output. When multiple switches share the same output, their signals are summed.

Each sub-switch can crossfade when transitioning, for click-free switches (when the fade is very short) or noticeable blends between sources/targets (when the fade is long).

The switches can be controlled by CV inputs, or by one of the 'Macro' parameters, which can in turn be controlled by one of the mapping sources (CV, MIDI, or I2C).

GUI

The display shows which input and output busses are currently joined by the various switches.

Common parameters

Name	Min	Max	Default	Unit	Description
Macro 1-6	0	127	0		A macro parameter that can be chosen as a sub-switch control source.
Output mode	0	1	0		The standard Add/Replace mode selector as described above, shared by all switch outputs.

95 https://www.youtube.com/watch?v=bbwHnLNqOXo

Per-switch parameters

Name	Min	Max	Default	Unit	Description
In control type	0	9	0		The type of control source for the input switch. See 'control types' below.
In control source	0	34	0		The control source for the input switch – one of the 64 busses, or one of the 6 macro parameters.
In link switch	0	6	0		The linked switch, if the ‘In control type’ is ‘Link’.
In link offset	0	11	0		The linked switch offset, if the ‘In control type’ is ‘Link’.
Out control type	0	9	0		The type of control source for the output switch. See 'control types' below.
Out control source	0	34	0		The control source for the output switch – one of the 64 busses, or one of the 6 macro parameters.
Out link switch	0	6	0		The linked switch, if the ‘Out control type’ is ‘Link’.
Out link offset	0	11	0		The linked switch offset, if the ‘Out control type’ is ‘Link’.
Reset source	0	34	0		The reset source for the switch – one of the 64 busses, or one of the 6 macro parameters.
Fade	0	1000	1	ms	The switch's crossfade time, in milliseconds.

Per-switch I/O parameters

Name	Min	Max	Default	Unit	Description
Input A-L	0	64	0		The bus to use for the 12 possible inputs of the input sub-switch.
Output A-H	0	64	0		The bus to use for the 8 possible outputs of the output sub-switch.

Control types

None	
Trig fwds	The chosen source is used as a trigger to advance the switch one step forwards.

Trig rev	The chosen source is used as a trigger to advance the switch one step backwards.
Trig pong	The chosen source is used as a trigger to advance the switch, the direction alternating on each pass. For example, if the chosen inputs/outputs are 1-2-3, the switch will advance like so: 1-2-3-2-1-2-3-2-1-etc.
Trig rand	The chosen source is used as a trigger – the switch adopts a randomly chosen position on each trigger.
Unipolar	The chosen source is used directly to select a switch position. Values from 0V to 5V map to switch positions from first to last.
Bipolar	The chosen source is used directly to select a switch position. Values from -5V to 5V map to switch positions from first to last.
Uni rev	The chosen source is used directly to select a switch position. Values from 0V to 5V map to switch positions from last to first.
Bi rev	The chosen source is used directly to select a switch position. Values from -5V to 5V map to switch positions from last to first.
Link	The switch position is linked to another switch's position, plus an offset.

Noise gate

“A simple noise gate”

File format guid: 'nsgt'

Specifications:

- Channels, 1-12: The number of bus channels to process.

Description

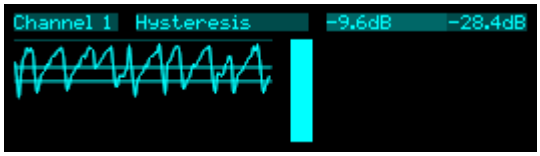
This algorithm is a multi-channel noise gate. Each channel is fully independent – this being a multi-channel algorithm is simply a convenience so that if you, say, want a noise gate on all 12 module inputs, you don’t need to add 12 copies of the algorithm to do so.

A sidechain input is available, to gate one signal with another.

The algorithm always works as an insert effect i.e. the output replaces the input.

GUI

The display shows a graph of the signal level (travelling from right to left). Superimposed on this are the threshold and hysteresis levels. To the right of the graph, the bar represents the gain reduction currently being applied.



Per-channel parameters

Name	Min	Max	Default	Unit	Description
Enable	0	1	0		Enables the channel.
Left/mono input	1	64	1		The left or mono input bus.
Right input	0	64	0		The right input bus, if stereo.
Sidechain input	0	64	0		The sidechain input bus.
Threshold	-70.0	0.0	-24.0	dB	The threshold level required to open the gate.
Hysteresis	-24.0	0.0	-3.0	dB	The level, relative to the threshold, that the input must drop below before the gate closes again.

Attack	0	1023	100		The attack time for the gate opening, from 0.2ms to 200ms with an exponential scale.
Hold	0	1023	100		The minimum time that the gate remains open, from 1ms to 1000ms with an exponential scale.
Release	0	1023	100		The release time for the gate closing, from 2ms to 2000ms with an exponential scale.
Lookahead	0.0	10.0	1.0	ms	The lookahead time. Increase this to prevent the gate from missing sharp attack transients. Note that the audio is delayed by the amount of the lookahead time i.e. this adds latency.
Gain reduction	-80	0	-80	dB	Sets the amount by which the signal is attenuated when the gate is closed. ‘-80’ is actually treated as ‘-∞dB’ i.e. fully attenuated.

Noise generator

“Generates various colours of noise”

File format guid: 'nois'

Specifications:

- Channels, 1-8: The number of output channels.

Description

This algorithm is a simple noise generator. Various standard “colours” of noise can be generated.

Globals parameters

Name	Min	Max	Default	Unit	Description
Gain	-40	6	0	dB	An overall gain to apply, in addition to the per-channel gain.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Output	0	64	13		The output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.
Amplitude	0.00	10.00	10.00	V	The amplitude of the noise signal (before the gain is applied).
Gain	-40	6	0	dB	The output level.
Colour	0	4	0		The noise colour. The options are “Blended”, “Violet”, “White”, “Pink”, and “Red”.
Blend	0	300	0		If “Blended” is selected, sets the blend between noise colours.

Notes

“A place to store some text”

File format guid: 'note'

Specifications: None

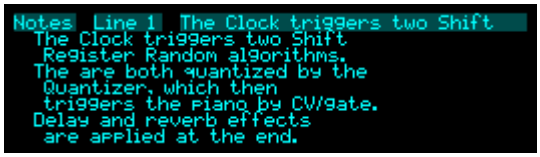
Description

This algorithm is simply a place to enter some text, perhaps an explanation of how the preset works, or your set list, or a reminder to buy milk.

It has absolutely no effect on the busses and consumes no CPU.

GUI

The display simply shows the lines of text all on one screen.



Oscilloscope

“Oscilloscope for viewing waveforms”

File format guid: 'oscs'

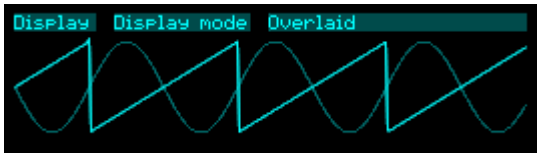
Specifications: None

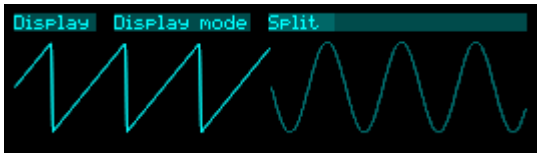
Description

This algorithm implements a simple but useful 2-channel oscilloscope.

GUI

The display shows one or both waveforms according to the display mode.





Inputs parameters

Name	Min	Max	Default	Unit	Description
Input 1	1	64	1		The bus for input 1.
Input 2	1	64	2		The bus for input 2.

Trigger parameters

Name	Min	Max	Default	Unit	Description
Trigger type	0	4	0		The type of trigger, to synchronise the display to the waveform. The options are: ● Free running. ● “1 rising” - trigger on the rising edge of input 1. ● “1 falling” - trigger on the falling edge of input 1. ● “2 rising” - trigger on the rising edge of input 2. ● “2 falling” - trigger on the falling edge of input 2.
Trigger voltage	-11.0	11.0	0.0	V	Set the trigger voltage – the voltage the input signal has to cross to trigger the oscilloscope.

Time range	0	16	7		Sets the time range corresponding to the full width of the display. The special value “Auto” sets the time range to the time between consecutive triggers.

Display parameters

Name	Min	Max	Default	Unit	Description
Display mode	0	4	2		Sets which waveforms are displayed and how. The options are “Overlaid”, “Split”, “Channel 1”, “Channel 2”, and “XY”. In “XY” mode, inputs 1 & 2 are used as the coordinates to draw, instead of using time as the abscissa.
Draw mode	0	2	0		Sets how the waveform signals are drawn. The options are “Lines (antialiased)”, “Lines”, and “Points”.
Vertical scale 1	0	3	0		Sets the vertical scale for channel 1. The options are 10V, 5V, 2V, and 1V.
Vertical scale 2	0	3	0		Sets the vertical scale for channel 2.
Vertical offset 1	-11.0	11.0	0.0	V	Sets an amount by which to move the display of channel 1 vertically.
Vertical offset 2	-11.0	11.0	0.0	V	Sets an amount by which to move the display of channel 2 vertically.
Disable screensaver	0	1	0		If set, and the oscilloscope custom UI is displayed, the screensaver is prevented from starting. The 0V line is also disabled, to prevent screen burn-in.

Phase Shifter

“Shifts the phase of signals”

File format guid: 'phsh'

Specifications: None

Description

This algorithm is based on the Phase Shifter algorithm on the disting mk4. You may like to view the video on that algorithm, which is here[96].

The algorithm is a phase shifter – it generates two outputs, one of which is in a fixed phase relationship to the other. Unlike the Phaser algorithm (see below), which is simply designed to create a swooshy sound, this algorithm is designed for mathematical accuracy, and can produce some interesting psycho-acoustic effects. It can also produce swooshy sounds.

Shift parameters

Name	Min	Max	Default	Unit	Description
Shift	-360	360	0	deg	Sets the phase shift.
Mix	0	100	100	%	The output wet/dry mix.
Level	-40	0	0	dB	The output gain applied to the effected signal.

Routing parameters

Name	Min	Max	Default	Unit	Description
Shift input	0	64	0		The bus to use for the phase shift CV, scaled as 5V per 360°.
Input	1	64	1		The first audio input bus.
Width	1	8	1		The number of busses to process, starting from the Input. For example, for a stereo signal, set this to 2.
Unshifted output	0	64	0		The bus to use for the unshifted output signal. Note that this is not the same as the dry signal unless the input is a pure sine wave.
Unshifted mode	0	1	0		The standard Add/Replace mode selector as described above for the unshifted output.

96 https://www.youtube.com/watch?v=MKjDCUPGxvU

+ve shift output	0	64	13		The bus to use for the positive shifted output.
+ve shift mode	0	1	0		The standard Add/Replace mode selector as described above for the positive shifted output.
-ve shift output	0	64	0		The bus to use for the negative shifted output.
-ve shift mode	0	1	0		The standard Add/Replace mode selector as described above for the negative shifted output.

Phaser

“Classic phaser effect”

File format guid: 'phas'

Specifications: None

Description

This algorithm is based on the Phaser algorithm on the disting mk4. You may like to view the video on that algorithm, which is here[97]. It’s a classic phaser effect, using a number of all-pass filters to introduce notches into the audio’s frequency spectrum. You may like to apply an LFO to the sweep CV input.

Phaser parameters

Name	Min	Max	Default	Unit	Description
Sweep	-1.000	1.000	0.000		Sweeps the all-pass filters over the audio spectrum.
Stages	1	32	8		Sets the number of all-pass filter stages.
Feedback	-100.0	100.0	0.0	%	Sets the amount of feedback around the effect. Negative values invert the signal.
Warp	0	1	1		If enabled, warps the sweep parameter/CV for a more perceptually linear sweep.
Input scale	-100	100	20	%	Sets the scaling of the sweep CV input.
Mix	0	100	50	%	The output wet/dry mix.
Level	-40	0	0	dB	The output gain applied to the effected signal.

Routing parameters

Name	Min	Max	Default	Unit	Description
Sweep input	0	64	0		The bus to use for the sweep CV.
Input	1	64	1		The first audio input bus.
Width	1	8	1		The number of busses to process, starting from the Input. For example, for a stereo signal, set this to 2.
Output	1	64	13		The first output bus.

97 https://www.youtube.com/watch?v=XLxi45ONXVA

Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Pitch reference

“Generates a pitch reference tone”

File format guid: 'ptch'

Specifications: None

Description

This algorithm simply generates a sine wave tone at a particular pitch or frequency.

Note that if using a note to specify the pitch, it respects the global tuning setting.

GUI

The display shows the chosen pitch as a MIDI note name plus cents, as a fractional MIDI note number, and as a frequency in Hz.

Pitch parameters

Name	Min	Max	Default	Unit	Description
Pitch mode	0	1	0		Sets the mode, either “Pitch”, or “Frequency”.
Note	0	127	69		Sets the note pitch.
Frequency	27.0	880.0	440.0	Hz	Sets the frequency.

Output parameters

Name	Min	Max	Default	Unit	Description
Output	1	64	13		The output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.
Amplitude	0.00	10.00	10.00	V	The amplitude of the signal (before the gain is applied).
Gain	-40	6	-40	dB	The level of the output signal.

Pitch Shifter

“A time-domain pitch shifter”

File format guid: 'ptsh'

Specifications: None

Description

This algorithm performs pitch shifting, using a time-domain algorithm[98].

The pitch shift amount is the sum of the various shift amount parameters and the CV input.

The algorithm was extracted from the disting EX ‘Tracker’ algorithm, as was the disting NT’s ‘Tracker’, below. As such these two algorithms make a natural pairing.

GUI

The display shows the current shift amount in cents.

Shifter parameters

Name	Min	Max	Default	Unit	Description
Shift algorithm	0	1	1		Chooses the pitch shifting algorithm, ‘2-phase’ or ‘3-phase’. The latter usually sounds better, at the expense of more CPU usage.
Grain delay	1	682	300	ms	The maximum length of the delay line used by the pitch shifter. The mean latency of the algorithm is half this value.
Octaves	-10	10	0		The pitch shift amount, in octaves.
Semitones	-48	48	0	ST	The pitch shift amount, in semitones.
Cents	-1200	1200	0	cents	The pitch shift amount, in cents.

98 By which we mainly mean that it’s not using a Phase Vocoder or similar technique.

Routing parameters

Name	Min	Max	Default	Unit	Description
Audio input	1	64	1		The bus to use as the audio input.
Shift input	0	64	0		The bus to use as a V/octave pitch shift amount CV.
Output	1	64	13		The output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Poly CV

“A polyphonic MIDI/CV converter”

File format guid: 'pycv'

Specifications:

- Voices: 1-14: The number of simultaneous voices.

Description

This algorithm takes the polysynth control logic from the other polysynths and uses it not to produce sound, but to output CVs and gates.

This makes it most obviously useful as a MIDI/CV converter, but it can also be used as a chord/arpeggio engine controlled by CV/gate inputs.

In fact, the algorithm can also generate MIDI as well as or instead of CV/gates, and so works as a MIDI-to-MIDI chord engine, for example[99].

Please refer to the section “Common polysynth features” above.

Voice output allocation

The ‘First output’ parameter sets the lowest-numbered bus to use for outputs. For each voice, outputs are assigned in the order gate, pitch CV, velocity.

For example, if ‘First output’ is ‘Output 1’, ‘Voices’ is set to 2, gate and pitch outputs are enabled, and velocity outputs are disabled, then the two voices will use busses as follows:

- Voice 1: gate on Output 1, pitch CV on Output 2.
- Voice 2: gate on Output 3, pitch CV on Output 4.

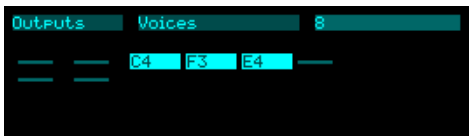
MIDI output

When using the algorithm to output MIDI, remember that the Voices parameter (and specification) still applies. For example, if you want to generate SATB harmony, you’ll need to select at least 4 for the number of voices.

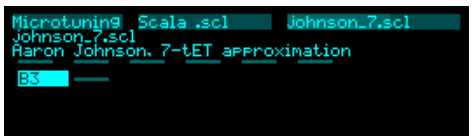
99 Perhaps it’s time to give this algorithm a new name.

GUI

The display shows the notes being played by the various voices.



When adjusting the microtuning parameters, more information about the chosen tuning is displayed.



MIDI/I2C parameters

Name	Min	Max	Default	Unit	Description
MIDI channel	0	16	1		The MIDI channel to listen on.
MPE channels	1	16	1		Controls how the algorithm will respond to MPE. See above.
I2C channel	0	255	1		Sets the I2C channel.
Sustain mode	0	1	0		The standard polysynth sustain mode parameter. See above.
Sustain	0	1	0		Directly controls the sustain (like a MIDI sustain pedal).
Transpose	-60	60	0	ST	Coarse tuning control.
Fine tune	-100	100	0	cents	Fine tuning control.
Bend range	0	48	2		The MIDI pitch bend range.
Min vel gate	0.0	5.0	1.0	V	Sets the minimum gate voltage, if the gates are set to be scaled by velocity. Velocity gates range from this voltage (for MIDI velocity 0) up to 5V (for MIDI velocity 127).
Voice allocation	0	1	0		Sets the voice allocation scheme for polyphonic converters – one of ‘Round robin’ or ‘Lowest voice’.

Outputs parameters

Name	Min	Max	Default	Unit	Description

Voices	1	14	1		The number of voices to use (up to the maximum set by the specification).
First output	1	64	15		The first output bus to use.
Gate outputs	0	2	1		Whether to include gate outputs. If set to ‘Velocity’, the gate outputs are scaled by the note velocity.
Pitch outputs	0	1	1		Whether to include pitch CV outputs.
Velocity outputs	0	1	0		Whether to include velocity outputs.
Gate mode	0	1	1		The standard Add/Replace mode selector as described above for gate outputs.
Pitch mode	0	1	1		The standard Add/Replace mode selector as described above for pitch CV outputs.
Velocity mode	0	1	1		The standard Add/Replace mode selector as described above for velocity outputs.
ES-5 Expander	0	6	0		If set, the ES-5 expander header to use for the gate output instead of allocating a gate from the regular busses. “1” means the ES-5 itself.
ES-5 Output	1	8	1		If using an ES-5 expander output, sets which of the 8 outputs on the expander to use.
Paraphonic gate	0	64	0		The output bus for a paraphonic gate (which is high while any voice gate output is high).
Para gate mode	0	1	1		The standard Add/Replace mode selector as described above for the paraphonic gate output.
Apply global tuning	0	1	1		If enabled, apply the global tuning setting to the output pitch CVs. Disable this when using the pitch CVs internally to the NT to control another algorithm, since the controlled algorithm will be using the global tuning setting to adjust its pitch.

MIDI output parameters

Name	**Min**	**Max**	**Default**	**Unit**	**Description**
MIDI channel	0	16	0		The output MIDI channel.

Output to breakout	0	1	0		Enables MIDI output to the breakout.
Output to Select Bus	0	1	0		Enables MIDI output to the Select Bus.
Output to USB	0	1	0		Enables MIDI output to USB.
Output to internal	0	1	0		Enables internal MIDI output – that is, MIDI is sent to the other algorithms.

Microtuning parameters

The algorithm uses the standard polysynth microtuning parameters, as described above.

Chord/arp parameters

The algorithm uses the standard polysynth chord and arpeggiator parameters, as described above.

CV/gate parameters

The algorithm uses the standard polysynth CV/gate parameters, as described above.

Poly FM

“A polyphonic FM synthesizer”

File format guid: 'pyfm'

Specifications:

- Timbres, 1-4: The number of timbres.
- Voices: 1-24: The number of simultaneous voices.

Description

This algorithm is inspired by the original Poly FM algorithm on the disting EX.

It is a polyphonic, multitimbral, FM synthesizer.

It uses the six-operator FM engine from the open source Plaits[100] module by Émilie Gillet (exactly as used in the Macro Oscillator 2 algorithm, above). It loads Yamaha DX7 voice bank SysEx files from the MicroSD card.

Please refer to the section “Common polysynth features” above.

Timbres

The algorithm supports up to four different 'timbres', which here mainly means a different bank/voice, plus some other parameters that make up a particular sound.

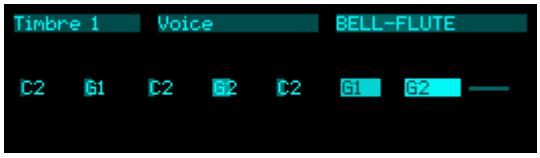
The various timbres share the pool of (up to 24) simultaneous voices.

Voice vs Voice

Yamaha's original documentation made the unfortunate decision to name the choice of sound within a bank a 'voice', which conflicts with the more common term for one of the sound-generating elements of the hardware (as in the phrase “eight voice polyphonic” meaning eight notes at once can be sounded). We've tried to be consistent with Yamaha's usage here (as in “a bank contains 32 voices”).

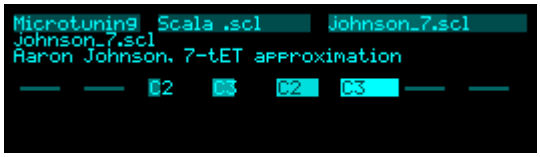
GUI

The display shows the notes being played by the various voices, each superimposed on a bar to indicate the note envelope.



100 https://github.com/pichenettes/eurorack/tree/master/plaits

When adjusting the microtuning parameters, more information about the chosen tuning is displayed.



Globals parameters

Name	Min	Max	Default	Unit	Description
Global gain	-40	6	0	dB	A global gain adjustment applied to all timbres (in addition to their individual gains).
Sustain mode	0	1	0		The standard polysynth sustain mode parameter. See above.

Microtuning parameters

The algorithm uses the standard polysynth microtuning parameters, as described above.

Per-timbre parameters

Name	Min	Max	Default	Unit	Description
Bank					Chooses the voice bank.
Voice	1	32	1		Chooses the voice within the bank.
Transpose	-60	60	0	ST	Coarse tuning control.
Fine tune	-100	100	0	cents	Fine tuning control.
Brightness	-100	100	0	%	An overall scaling on the modulator depths. Corresponds to the Plaits 'Timbre'.
Envelope scale	-100	100	0	%	An overall scaling on the envelope times. Corresponds to the Plaits 'Morph'.
Gain	-40	6	0	dB	Sets the output level.
Sustain	0	1	0		Directly controls the sustain (like a MIDI sustain pedal).
Press -> volume	-100	100	0	%	The amount by which pressure (per note for MPE) affects the note volume.
Press -> bright	-200	200	0	%	The amount by which pressure (per note for MPE) affects the brightness.

MPE Y -> volume	-100	100	0	%	The amount by which the MPE Y dimension affects the note volume.
MPE Y -> bright	-200	200	0	%	The amount by which the MPE Y dimension affects the brightness.

Chord/arp parameters

The algorithm uses the standard polysynth chord and arpeggiator parameters, as described above.

Per-timbre setup parameters

Name	Min	Max	Default	Unit	Description
Left/mono output	1	64	13		The left or mono output bus.
Right output	0	64	0		The right output bus.
MIDI channel	0	16	1		The MIDI channel to listen on.
MPE channels	1	16	1		Controls how the algorithm will respond to MPE. See above.
I2C channel	0	255	1		Sets the I2C channel.
Bend range	0	48	2	ST	The MIDI pitch bend range.

CV/gate parameters

The algorithm uses the standard polysynth CV/gate parameters, as described above.

Poly FM (Editable)

“A polyphonic FM synthesizer”

File format guid: 'pyfe'

Specifications:

- Voices: 1-24: The number of simultaneous voices.

Description

This algorithm is a version of the Poly FM algorithm (above) which forgoes multi-timbrality in favour of being able to edit the actual FM voice parameters, just as if you were editing a real DX7.

Please refer to the documentation of the Poly FM algorithm for everything not covered here.

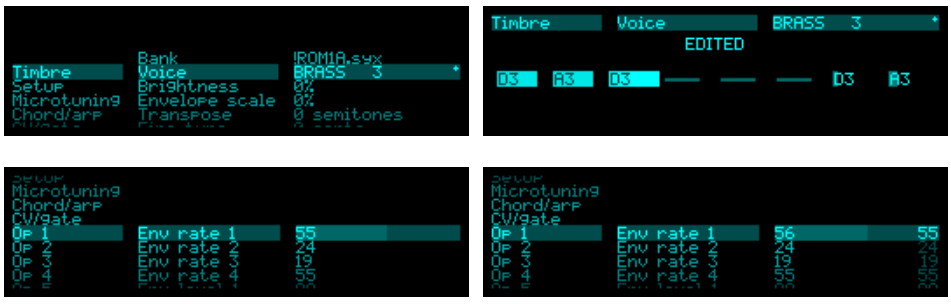
Loading banks/voices

As for the Poly FM algorithm, banks are loaded from the MicroSD card. Unlike that algorithm however, this version has a ‘Confirm’ step before changing to a new voice. This is because you’re actually ‘loading’ it, potentially replacing your edits, rather than simply choosing a (read only) voice to play.

Viewing edits

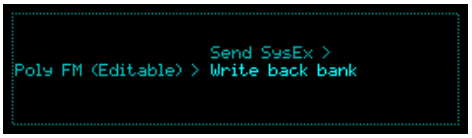
If you edit one of the FM voice parameters, three changes occur in the UI to let you know you’ve done it:

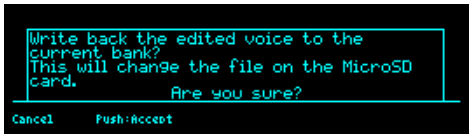
- A star (*) will appear next to the voice name.
- ‘EDITED’ will appear in the algorithm’s UI.
- The original (unedited) parameter values will appear down the right side of the display.



Saving edits

Any changes you make can be saved back to the bank on the MicroSD card. This is done via the algorithm's menu:





Using MIDI SysEx

This algorithm supports a subset of Yamaha's System Exclusive protocol for editing the DX7. This enables you to use the algorithm with editor/librarian apps on your computer (for example, Dexed[101]).

Specifically, the supported SysEx messages are as follows.

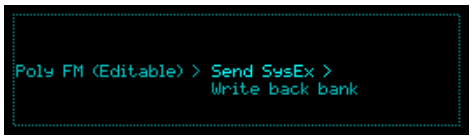
Received:

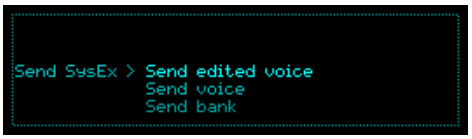
- Single voice bulk data (sub-status 0, format 0).
- Parameter change (sub-status 1).
- Request for single voice bulk data.
- Request for bank (32 voices) bulk data.

Transmitted:

- Single voice bulk data (sub-status 0, format 0).
- Bank (32 voices) bulk data (sub-status 0, format 9).

Voice and bank SysEx dumps can be initiated from the algorithm's menu:





101 https://asb2m10.github.io/dexed/

Poly Macro Oscillator 2

“It's lots of Plaits!”

File format guid: 'pym2'

Specifications:

- Voices: 1-8: The number of simultaneous voices.

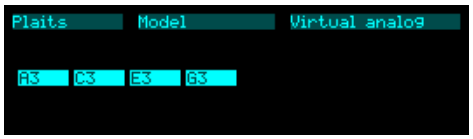
Description

This algorithm is a polysynth where each voice is essentially a Plaits (the open source Plaits[102] module by Émilie Gillet).

Please refer to the section “Common polysynth features” above.

GUI

The display shows the notes being played by the various voices. Note that Plaits voices never really end – in theory they decay forever.



Outputs

As on the original Plaits, there are two outputs, ‘Main’ and ‘Aux’. If you select the same bus for both outputs they are mixed.

Plaits parameters

Name	Min	Max	Default	Unit	Description
Model	0	23	0		Chooses the synthesis model.
Coarse tune	-60	60	0	ST	Coarse tuning control.
Fine tune	-100	100	0	cents	Fine tuning control.
Harmonics	0	127	64		Sets the harmonics control.
Timbre	0	127	64		Sets the timbre control.
Morph	0	127	64		Sets the morph control.

102 https://github.com/pichenettes/eurorack/tree/master/plaits

FM	-100	100	0	%	Sets the FM depth.
Timbre mod	-100	100	0	%	Sets the timbre modulation depth.
Morph mod	-100	100	0	%	Sets the morph modulation depth.
Low-pass gate	0	127	127		Sets the LPG colour (VCFA-VCA blend).
Time/decay	0	127	64		Sets the envelope decay time.

Inputs parameters

Name	Min	Max	Default	Unit	Description
Level input	0	64	0		Selects which bus is used as the level input.
FM input	0	64	0		Selects which bus is used as the FM input.
Harmonics input	0	64	0		Selects which bus is used as the harmonics input.
Timbre input	0	64	0		Selects which bus is used as the timbre input.
Morph input	0	64	0		Selects which bus is used as the morph input.

Outputs parameters

Name	Min	Max	Default	Unit	Description
Main output	0	64	13		The bus for the ‘Main’ output.
Aux output	0	64	0		The bus for the ‘Aux’ output.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.
Main gain	-40	6	0	dB	The level of the ‘Main’ output signal.
Aux gain	-40	6	0	dB	The level of the ‘Aux’ output signal.

Setup parameters

Name	Min	Max	Default	Unit	Description

MIDI channel	0	16	1		The MIDI channel to listen on.
MPE channels	1	16	1		Controls how the algorithm will respond to MPE. See above.
I2C channel	0	255	1		Sets the I2C channel.
Bend range	0	48	2		The MIDI pitch bend range.
Sustain mode	0	1	0		The standard polysynth sustain mode parameter. See above.
Sustain	0	1	0		Directly controls the sustain (like a MIDI sustain pedal).

Modulation parameters

Name	Min	Max	Default	Unit	Description
Press -> volume	-100	100	0	%	The amount by which pressure (per note for MPE) affects the note volume.
Press -> harmonics	-127	127	0		The amount by which pressure (per note for MPE) affects the harmonics.
Press -> timbre	-127	127	0		The amount by which pressure (per note for MPE) affects the timbre.
Press -> morph	-127	127	0		The amount by which pressure (per note for MPE) affects the morph.
MPE Y -> volume	-100	100	0	%	The amount by which the MPE Y dimension affects the note volume.
MPE Y -> harmonics	-127	127	0		The amount by which the MPE Y dimension affects the harmonics.
MPE Y -> timbre	-127	127	0		The amount by which the MPE Y dimension affects the timbre.
MPE Y -> morph	-127	127	0		The amount by which the MPE Y dimension affects the morph.

Microtuning parameters

The algorithm uses the standard polysynth microtuning parameters, as described above.

Chord/arp parameters

The algorithm uses the standard polysynth chord and arpeggiator parameters, as described above.

CV/gate parameters

The algorithm uses the standard polysynth CV/gate parameters, as described above.

Poly Multisample

“A polyphonic sample player”

File format guid: 'pymu'

Specifications:

- Voices: 1-16: The number of simultaneous voices.
- Record time: 0-60 seconds: The maximum length of audio that can be recorded (sampled).

Description

This algorithm is inspired by the original “SD Multisample” algorithm on the disting EX, with hints of the Crossfade Loop Synth[103] VST plug-in, and the “Audio Playback with Crossfade” algorithm on the disting mk4 (video here[104]).

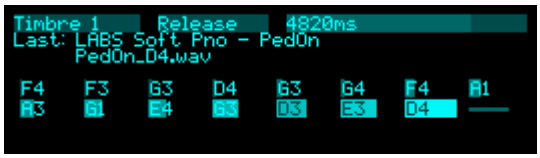
It is a polyphonic sample playback instrument, playing WAV files from the MicroSD card. It can also record audio (i.e. sample in the traditional sense) and play that. It can apply crossfade looping to make smooth looping pads out of any sample material.

It can be played via CV/gate, MIDI, or I2C. It supports both velocity switches and round robins per sample.

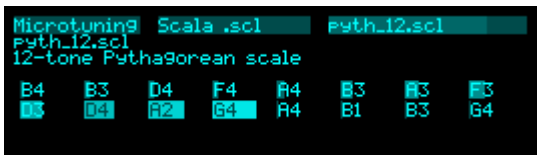
Please refer to the section “Common polysynth features” above, and to the section on samples on the MicroSD card, also above. The section on “Recording (sampling)” below also applies.

GUI

The display shows the notes being played by the various voices, each superimposed on a bar to indicate the note envelope. The folder and filename of the last sample played are also shown.



When adjusting the microtuning parameters, more information about the chosen tuning is displayed.



When adjusting trim and loop points, this algorithm uses the UI described for the sample players (below).

103 https://www.expert-sleepers.co.uk/xfadelooper.html

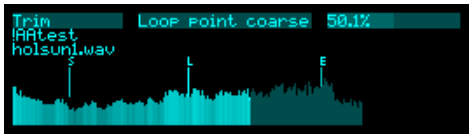
104 https://www.youtube.com/watch?v=x3bmglh36mE

Crossfade looping

If enabled, the sample will be looped using a crossfade technique, which avoids the necessity for carefully setting up click-free loop points. Because during the crossfades two samples are being played at once, the polyphony is halved when crossfade looping is enabled.

In the view that shows the currently playing notes, a horizontal bar at the bottom of the block indicates the crossfade position for each voice (and clearly shows the two voices causing the reduced polyphony).

Playback begins at the start trim point, and proceeds to the end trim point, then loops between the end trim point and the loop point. The looped section can be always played forwards, or played alternately forwards and backwards. These three points are displayed when editing any of them in the GUI view.



As with non-crossfade looping, playback continues forever once triggered unless you've enabled the envelope, so you will usually want to do that.

If you change the sample while a note is held, it will smoothly crossfade to the new sample and start looping that one instead. (This is the magic sauce behind the disting mk4 Audio Playback with Crossfade algorithm already mentioned.)

Sample parameters

Name	Min	Max	Default	Unit	Description
Folder	0		0		Sets the sample folder to use.
Sample	-1		-1		Allows you to choose a single sample within the folder to play, or "<MULTISAMPLE>" for the usual case where you want the algorithm to choose the sample automatically based on the note pitch.
Gain	-40	24	0	dB	Sets the output level.
Pan	-100	100	0	%	Sets the stereo pan position.
Transpose	-60	60	0	ST	Adjusts the tuning in semitones.
Fine tune	-100	100	0	cents	Adjusts the tuning in cents.

Crossfade	0	2	0		Sets the crossfade looping mode: Off, Forwards, or Alternating.
Crossfade	0.1	100.0	50.0	%	Sets the length of the crossfade, as a percentage of the loop time (the difference between the loop point and end trim point).

Trim parameters

The algorithm uses some of the standard Sample Player trim parameters, as described below, plus these additional parameters.

Name	Min	Max	Default	Unit	Description
Loop point coarse	0.0	100.0	0.0	%	Sets the loop point, as a percentage of the sample length.
Loop point fine	-500	500	0		Adjusts the loop point, in sample frames.

Setup parameters

Name	Min	Max	Default	Unit	Description
Sustain mode	0	1	0		The standard polysynth sustain mode parameter. See above.
Sustain	0	1	0		Directly controls the sustain (like a MIDI sustain pedal).
Gate offset	0.0	10.0	0.2	ms	Offsets (delays) the gate inputs relative to the pitch inputs. This is useful to allow pitch CVs to settle before they are sampled on the rising gate, and also to cope with modules which output both a pitch and gate but change their gate first.
Round robin mode	0	3	0		The round-robin mode. See above.
Loop	0	2	0		Sets whether the sample will loop. ‘From WAV file’ will loop if the sample has loop points defined in the file – see above – and not otherwise. ‘Off’ and ‘On’ force the sample to loop or not.
Bend range	0	48	2	ST	The MIDI pitch bend range.
Confirm change	0	3	0		Sets whether the folder and/or sample changes immediately or requires confirmation.

Envelope parameters

Name	Min	Max	Default	Unit	Description
Envelope	0	1	0		Enables an ADSR volume envelope. If the envelope is disabled, the sample simply plays once until the end.
Attack	0	127	0		Sets the envelope attack time (from 1ms to 15s, exponential response).
Decay	0	127	60		Sets the envelope decay time (from 20ms to 15s, exponential response).
Sustain	0	100	100	%	Sets the envelope sustain level.
Release	0	127	77		Sets the envelope release time (from 10ms to 30s, exponential response).
Velocity	0	100	100	%	Sets the amount by which the note velocity affects the note volume.
Press -> volume	-100	100	0	%	The amount by which pressure (per note for MPE) affects the note volume.
MPE Y -> volume	-100	100	0	%	The amount by which the MPE Y dimension affects the note volume.

Microtuning parameters

The algorithm uses the standard polysynth microtuning parameters, as described above.

Chord/arp parameters

The algorithm uses the standard polysynth chord and arpeggiator parameters, as described above.

Routing parameters

Name	Min	Max	Default	Unit	Description
Left/mono output	1	64	13		The left or mono output bus.
Right output	0	64	0		The right output bus.
MIDI channel	0	16	1		The MIDI channel to listen on.
MPE channels	1	16	1		Controls how the algorithm will respond to MPE. See above.

I2C channel	0	255	1		Sets the I2C channel.

CV/gate parameters

The algorithm uses the standard polysynth CV/gate parameters, as described above.

Record parameters

The algorithm uses the standard Sample Player record parameters, as described below.

Poly Multisample (legacy)

“A polyphonic sample player”

File format guid: 'pyms'

Specifications:

- Timbres, 1-4: The number of timbres.
- Voices: 1-16: The number of simultaneous voices.

Description

This algorithm is inspired by the original “SD Multisample” algorithm on the disting EX.

It is a polyphonic, multitimbral, sample playback instrument, playing WAV files from the MicroSD card. It can be played via CV/gate, MIDI, or I2C. It supports both velocity switches and round robins per sample.

Please refer to the section “Common polysynth features” above, and to the section on samples on the MicroSD card, also above.

This algorithm was the “Poly Multisample” up until firmware version 1.10, in which it was replaced by the current “Poly Multisample”, which allows audio recording.

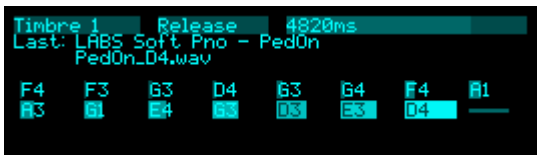
Timbres

The algorithm supports up to four different 'timbres', which here mainly means a different sample folder, plus some other parameters that make up a particular sound.

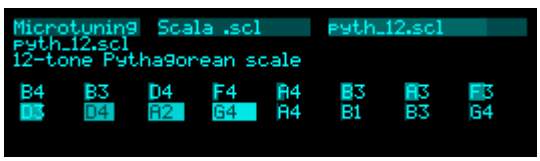
The various timbres share the pool of (up to 16) simultaneous voices.

GUI

The display shows the notes being played by the various voices, each superimposed on a bar to indicate the note envelope. The folder and filename of the last sample played are also shown.



When adjusting the microtuning parameters, more information about the chosen tuning is displayed.



Globals parameters

Name	Min	Max	Default	Unit	Description
Global gain	-40	6	0	dB	A global gain adjustment applied to all timbres (in addition to their individual gains).
Sustain mode	0	1	0		The standard polysynth sustain mode parameter. See above.
Gate offset	0.0	10.0	0.2	ms	Offsets (delays) the gate inputs relative to the pitch inputs. This is useful to allow pitch CVs to settle before they are sampled on the rising gate, and also to cope with modules which output both a pitch and gate but change their gate first.
Round robin mode	0	3	0		The round-robin mode. See above.

Microtuning parameters

The algorithm uses the standard polysynth microtuning parameters, as described above.

Per-timbre parameters

Name	Min	Max	Default	Unit	Description
Folder	1		1		Sets the sample folder to use.
Gain	-40	24	0	dB	Sets the output level.
Pan	-100	100	0	%	Sets the stereo pan position.
Transpose	-60	60	0	ST	Adjusts the tuning in semitones.
Fine tune	-100	100	0	cents	Adjusts the tuning in cents.
Envelope	0	1	0		Enables an ADSR volume envelope. If the envelope is disabled, the sample simply plays once until the end.
Attack	0	127	0		Sets the envelope attack time (from 1ms to 15s, exponential response).
Decay	0	127	60		Sets the envelope decay time (from 20ms to 15s, exponential response).
Sustain	0	100	100	%	Sets the envelope sustain level.
Release	0	127	77		Sets the envelope release time (from 10ms to 30s, exponential response).

Velocity	0	100	100	%	Sets the amount by which the note velocity affects the note volume.
Sustain	0	1	0		Directly controls the sustain (like a MIDI sustain pedal).

Chord/arp parameters

The algorithm uses the standard polysynth chord and arpeggiator parameters, as described above.

Per-timbre setup parameters

Name	Min	Max	Default	Unit	Description
Left/mono output	1	64	13		The left or mono output bus.
Right output	0	64	0		The right output bus.
MIDI channel	0	16	1		The MIDI channel to listen on.
MPE channels	1	16	1		Controls how the algorithm will respond to MPE. See above.
I2C channel	0	255	1		Sets the I2C channel.
Bend range	0	48	2	ST	The MIDI pitch bend range.

CV/gate parameters

The algorithm uses the standard polysynth CV/gate parameters, as described above.

Poly Resonator

“It's lots of Rings!”

File format guid: 'pyri'

Specifications:

- Voices: 1-8: The number of simultaneous voices.

Description

This algorithm is a polysynth where each voice is essentially a Rings (the open source Rings[105] module by Émilie Gillet).

Rings itself does support a kind of polyphony, but it’s limited by having a single pitch and gate input – it’s very much designed so that only one note is triggered at a time. This algorithm gives you a full Rings implementation for each voice, with the convenience of shared parameter control.

Please refer to the section “Common polysynth features” above.

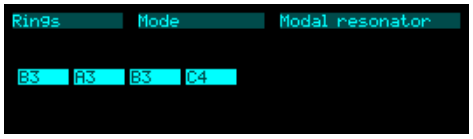
Resolution

Rings has a hidden internal parameter called ‘resolution’. Essentially, it’s the number of resonator building blocks that contribute to the overall sound. A Rings with no polyphony uses a resolution of 64; when you turn on polyphony, that 64 is divided by the number of voices e.g. if the polyphony is 4, each voice has a resolution of 16. This keeps the CPU load roughly constant.

This algorithm allows you to control the resolution manually, so you can choose both the sound that you want and its CPU cost. It defaults to 16, so it sounds the same as a Rings set to 4 voice polyphony.

GUI

The display shows the notes being played by the various voices. Note that Rings voices never really end – in theory they decay forever.



105 https://github.com/pichenettes/eurorack/tree/master/rings

Rings parameters

Name	Min	Max	Default	Unit	Description
Mode	0	4	0		Selects the resonator mode.
Synth effect	0	1	0		If the mode is 'Synth', selects the audio effect applied to the synth output.
Coarse tune	-36	24	0	ST	Provides a coarse tuning control.
Fine tune	-100	100	0	cents	Provides a fine tuning control.
Resolution	8	64	16		Sets the resolution of the resonators. See above.
Structure	0	127	64		Controls the resonator 'structure'.
Brightness	0	127	64		Controls the resonator 'brightness'.
Damping	0	127	64		Controls the resonator 'damping'.
Position	0	127	64		Controls the resonator 'position'.
Chord	0	10	0		Chooses the chord to use for resonator modes that use one. In the original, set from the Structure knob.
Noise gate	0	1	1		Enables a noise gate on the audio input. Always enabled in the original Rings.
Input gain	-40	12	0	dB	A gain applied to the input audio, before it hits the resonator.

Routing parameters

Name	Min	Max	Default	Unit	Description
Audio input	0	64	0		Chooses which input bus to use for the audio input.
Odd output	0	64	13		The bus to use for the Odd output.
Even output	0	64	13		The bus to use for the Even output.
Odd output mode	0	1	0		The Add/Replace mode for the Odd output.
Even output mode	0	1	0		The Add/Replace mode for the Even output.

Output gain	-40	12	0	dB	The output level (of both the Odd and Even outputs).
Dry gain	-40	12	-40	dB	The level of the input audio mixed into the output(s).

Setup parameters

Name	Min	Max	Default	Unit	Description
MIDI channel	0	16	1		The MIDI channel to listen on.
MPE channels	1	16	1		Controls how the algorithm will respond to MPE. See above.
I2C channel	0	255	1		Sets the I2C channel.
Bend range	0	48	2		The MIDI pitch bend range.
Sustain mode	0	1	0		The standard polysynth sustain mode parameter. See above.
Sustain	0	1	0		Directly controls the sustain (like a MIDI sustain pedal).

Modulation parameters

Name	Min	Max	Default	Unit	Description
Press -> volume	-100	100	0	%	The amount by which pressure (per note for MPE) affects the note volume.
Press -> structure	-127	127	0		The amount by which pressure (per note for MPE) affects the structure.
Press -> brightness	-127	127	0		The amount by which pressure (per note for MPE) affects the brightness.
Press -> damping	-127	127	0		The amount by which pressure (per note for MPE) affects the damping.
Press -> position	-127	127	0		The amount by which pressure (per note for MPE) affects the position.
MPE Y -> volume	-100	100	0	%	The amount by which the MPE Y dimension affects the note volume.
MPE Y -> structure	-127	127	0		The amount by which the MPE Y dimension affects the structure.

MPE Y -> brightness	-127	127	0		The amount by which the MPE Y dimension affects the brightness.
MPE Y -> damping	-127	127	0		The amount by which the MPE Y dimension affects the damping.
MPE Y -> position	-127	127	0		The amount by which the MPE Y dimension affects the position.

Microtuning parameters

The algorithm uses the standard polysynth microtuning parameters, as described above.

Chord/arp parameters

The algorithm uses the standard polysynth chord and arpeggiator parameters, as described above.

CV/gate parameters

The algorithm uses the standard polysynth CV/gate parameters, as described above.

Poly Wavetable

“A polyphonic wavetable synthesizer”

File format guid: 'pywt'

Specifications:

- Voices: 1-24: The number of simultaneous voices.

Description

This algorithm is inspired by the original Poly Wavetable algorithm on the disting EX.

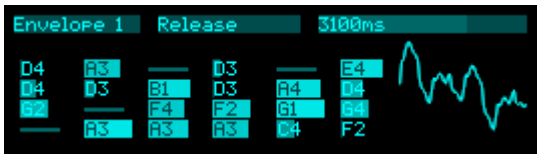
It is a complete polyphonic synthesizer, using wavetable oscillators. Each voice has two envelopes, a filter and an LFO.

Please refer to the section “Common polysynth features” above, and to the section on wavetables on the MicroSD card, also above.

There are a number of in-depth videos on the disting EX version of this algorithm, which mostly apply equally well to this version. The YouTube playlist is here[106].

GUI

The display shows the notes being played by the various voices, each superimposed on a bar to indicate the note envelope. The waveform within the wavetable of the most recently played note is also shown.



Wavetable parameters

Name	Min	Max	Default	Unit	Description
Wavetable	0		0		Chooses the wavetable from those installed on the MicroSD card.
Wave offset	-100	100	0		An offset for the wavetable position, added to that set from the wave input.
Wave spread	-100	100	0		An amount by which to spread out the per-voice wavetable positions.

106 https://www.youtube.com/watch?v=6ytLfM9xyRY&list=PLlY5j4QDwxWunH7I7wvOGYGNsRvCMZ2Hx

Wave input	0	64	0		Which input bus to use to control the position in the wavetable.

Envelope 1-2 parameters

Name	Min	Max	Default	Unit	Description
Attack	0	127	20		Envelope attack time. Range 1ms-15s.
Decay	0	127	60		Envelope decay time. Range 20ms-15s.
Sustain	0	127	80		Envelope sustain level.
Release	0	127	60		Envelope release time. Range 10ms-30s.
Attack shape	0	127	64		Envelope attack shape. '0' is highly exponential; '127' is almost linear.
Decay shape	0	127	64		Envelope decay & release shape.

Filter parameters

Name	Min	Max	Default	Unit	Description
Filter type	0	3	0		The filter type; 'Off', 'Lowpass', 'Bandpass' or 'Highpass'.
Filter freq	0	127	64		The filter frequency, specified as a MIDI note number.
Filter Q	0	100	50		The filter resonance.

LFO parameters

Name	Min	Max	Default	Unit	Description
LFO speed	-100	100	90		The LFO speed. Range 0.01Hz-10Hz.
LFO retrigger	0	2	0		Sets whether the LFOs are retriggered at note on. The options are 'Poly' (each voice's LFO triggers independently), 'Mono' (all LFOs are retriggered when the first note is played), or 'Off' (LFOs are free-running).

LFO spread	0	90	0	Degrees	Sets the phase to which LFOs are retriggered. The value, in degrees (360° per LFO cycle), is multiplied by the voice number to give the initial LFO phase. When retrigger is off, this sets the phase relationship between the free-running LFOs.

Modulation parameters

Name	Min	Max	Default	Unit	Description
Veloc -> volume	0	100	100	%	The amount by which the note velocity affects the note volume.
Veloc -> wave	-100	100	0	%	The amount by which the note velocity affects the wavetable position.
Veloc -> filter	-127	127	0		The amount by which the note velocity affects the filter frequency.
Veloc -> peak	0	100	0	%	The amount by which the note velocity affects the envelope attack peak.
Veloc -> sustain	0	100	0	%	The amount by which the note velocity affects the envelope sustain level.
Pitch -> wave	-100	100	0	%	The amount by which the note pitch affects the wavetable position.
Pitch -> filter	-100	100	0	%	The amount by which the note pitch affects the filter frequency.
Env-1 -> wave	-100	100	0	%	The amount by which envelope 1 affects the wavetable position.
Env-1 -> filter	-127	127	0		The amount by which envelope 1 affects the filter frequency.
Env-2 -> wave	-100	100	0	%	The amount by which envelope 2 affects the wavetable position.
Env-2 -> filter	-127	127	0		The amount by which envelope 2 affects the filter frequency.
Env-2 -> pitch	-12.0	12.0	0.0	ST	The amount by which envelope 2 affects the note pitch.
LFO -> wave	-100	100	0	%	The amount by which the LFO affects the wavetable position.
LFO -> filter	-127	127	0		The amount by which the LFO affects the filter frequency.

LFO -> pitch	-12.0	12.0	0.0	ST	The amount by which the LFO affects the note pitch.
Press -> volume	-100	100	0	%	The amount by which pressure (per note for MPE) affects the note volume.
Press -> wave	-100	100	0	%	The amount by which pressure (per note for MPE) affects the wavetable position.
Press -> filter	-127	127	0		The amount by which pressure (per note for MPE) affects the filter frequency.
Press -> sustain	-100	100	0	%	The amount by which pressure (per note for MPE) affects the envelope sustain level.
MPE Y -> volume	-100	100	0	%	The amount by which the MPE Y dimension affects the note volume.
MPE Y -> wave	-100	100	0	%	The amount by which the MPE Y dimension affects the wavetable position.
MPE Y -> filter	-127	127	0		The amount by which the MPE Y dimension affects the filter frequency.

Girth parameters

Name	**Min**	**Max**	**Default**	**Unit**	**Description**
Unison	1	8	1		The number of voices to play simultaneously for each note triggered.
Unison detune	0	100	10	cents	The detune amount when Unison is active.
Output spread	-100	100	0	%	The amount of output spread.
Spread mode	0	2	0		The output spread mode. See below.

Microtuning parameters

The algorithm uses the standard polysynth microtuning parameters, as described above.

Chord/arp parameters

The algorithm uses the standard polysynth chord and arpeggiator parameters, as described above.

Setup parameters

Name	Min	Max	Default	Unit	Description
Gain	-40	24	0	dB	Applies an overall output gain.
Left output	1	64	13		The left output bus.
Right output	0	64	14		The right output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.
MIDI channel	0	16	1		The MIDI channel to listen on.
MPE channels	1	16	1		Controls how the algorithm will respond to MPE. See above.
I2C channel	0	255	1		Sets the I2C channel.
Pitchbend input	0	64	0		Sets the input bus to use for pitch bend.
Max voices	1	24	24		Sets the maximum number of simultaneous voices.
Sustain mode	0	1	0		The standard polysynth sustain mode parameter. See above.
Sustain	0	1	0		Directly controls the sustain (like a MIDI sustain pedal).
Bend range	0	48	2		The MIDI pitch bend range.
Trigger mode	0	1	0		Sets the note triggering mode. See below.

CV/gate parameters

The algorithm uses the standard polysynth CV/gate parameters, as described above.

Spread modes

The available values for the ‘Spread mode' parameter are as follows:

Spread by voice	Voices are spread across the stereo field from left to right.
Spread by voice 2	Voices are spread across the stereo field in an alternating left/right manner, by a small amount for low numbered voices, increasing for the remaining voices.
Spread by pitch	Voices are spread across the stereo field according to their pitch, with note 48 at

	the centre.

Trigger modes

The 'Trigger mode' parameter controls how notes are triggered. The options are as follows:

Gated	Normal operation. Notes start at gate on and are released at gate off.
Triggered	Notes start at gate on; envelopes proceed through one full cycle of attack, decay and release. Gate off is ignored.

Quadraphonic Mixer

“A quadraphonic mixer”

File format guid: 'quad'

Specifications:

- Channels, 1-12: The number of mixer chnannels.

Description

This algorithm is based on the disting EX algorithm of the same name. You may like to review the video about that algorithm, here[107]. It is a quadraphonic mixer – that is, it has four outputs for four speakers, rather than the usual two for a stereo mixer. If you need a primer on quadraphonic audio, Wikipedia has a page here[108].

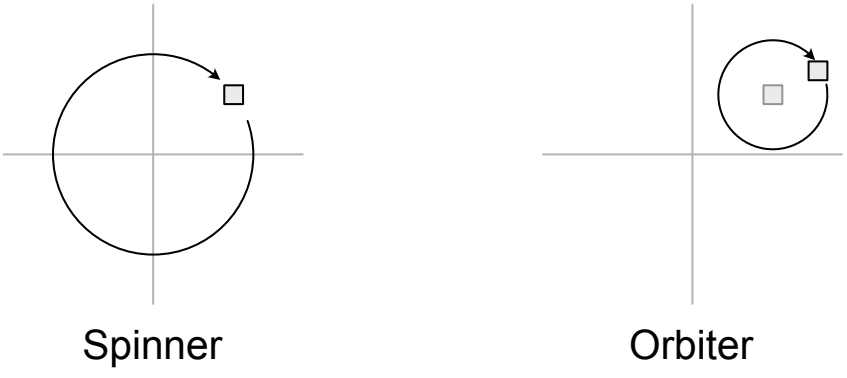
Spinners and orbiters

The algorithm provides two means to perform that classic quadraphonic trick, spinning the audio around the audience.

Each channel has a Spinner and an Orbiter. By default, each channel uses its own, but you can set it up to share spinners and orbiters between channels.

The spinner internally modulates the Angle parameter; it only works if the channel is using polar coordinates. Therefore it spins the position around the centre.

The orbiter is what was called the spinner in the disting EX version. It spins the audio position around the nominal position set by the coordinates; it therefore works in both polar and rectangular coordinate modes.

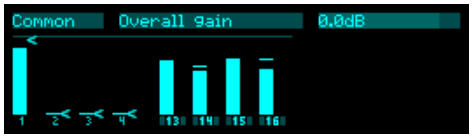


107 https://www.youtube.com/watch?v=Y5KzBcD-Y5o

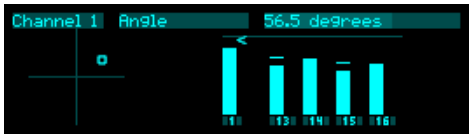
108 https://en.wikipedia.org/wiki/Quadraphonic_sound

GUI

When on the common parameters page, the GUI shows a mixer view similar to that of the Mixer Stereo algorithm, though with four outputs:



When viewing the parameters of a channel, a graphical view is shown representing the channel position in the quadraphonic field:



Common parameters

Name	Min	Max	Default	Unit	Description
Front left output	1	64	13		The bus to use for the front left speaker output.
Front right output	1	64	14		The bus to use for the front left right output.
Rear left output	1	64	15		The bus to use for the rear left speaker output.
Rear right output	1	64	16		The bus to use for the rear right speaker output.
Output mode	0	1	0		The standard Add/Replace mode selector as described above, for all outputs.
Overall gain	-70.0	-12.0	0.0	dB	An overall gain control; combines with the front and rear gain controls.
Front gain	-70.0	-12.0	0.0	dB	A gain control for the front speakers.
Rear gain	-70.0	-12.0	0.0	dB	A gain control for the rear speakers.
Front bass	-24.0	24.0	0.0	dB	A shelving cut/boost bass tone control for the front speakers.
Front treble	-24.0	24.0	0.0	dB	A shelving cut/boost treble tone control for the front speakers.
Rear bass	-24.0	24.0	0.0	dB	A shelving cut/boost bass tone control for the rear speakers.

Rear treble	-24.0	24.0	0.0	dB	A shelving cut/boost treble tone control for the rear speakers.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Input	0	64			The channel’s input bus.
Gain	-70.0	6.0	-70.0	dB	The channel’s gain.
Coordinates	0	1	0		Whether the channel uses Rectangular or Polar coordinates.
X	-100	100	0	%	If using Rectangular coordinates, the channel’s X position, from left (-100%) to right (100%).
Y	-100	100	0	%	If using Rectangular coordinates, the channel’s Y position, from rear (-100%) to front (100%).
Angle	-180.0	180.0	0.0	deg	If using Polar coordinates, the channel’s angle, where 0 is front centre and ±180 is rear centre.
Radius	0	100	0	%	If using Polar coordinates, the channel’s radius, where 0% is centre and 100% is the maximum distance from the centre.
Spinner	1				Which channel’s spinner to use.
Spin rate	-100	100	0		The rate of spin to apply.
Orbiter	1				Which channel’s orbiter to use.
Orbit rate	-100	100	0		The rate of the orbiter.
Orbit radius	0	100	0	%	The radius of the orbiter.
Orbit phase	-180	180	0	deg	Sets the phase angle to add to the orbiter. Different inputs with the same orbiter but different phase follow each other around the circle at a fixed offset.
Mute	0	1	0		Mutes the channel.
Solo	0	1	0		Solos the channel.
Name					A text name for the channel for identification.

Distance gain	-24.0	24.0	0.0	dB	Sets the amount by which the gain will change when moving away from the centre. For example, if the listener is notionally in the centre position, you may like the audio to get quieter as the source moves away from the centre, in which case set this to a negative value.

Quantizer

“A CV quantizer”

File format guid: 'quan'

Specifications:

- Channels, 1-12: The number of bus channels to process.

Description

This algorithm is a multi-channel CV quantizer, loosely based on the Quad Quantizer algorithm on the disting EX. It supports microtuning (see above), but this is not a requirement.

It quantizes CV inputs and generates CV outputs and/or MIDI output.

The quantization parameters are common to all channels; the individual channels’ parameters are solely related to routing.

Quantization signal flow

There are a number of steps between the input CV and the output.

1. The input CV is sampled. If the quantizer has a gate input selected, the CV is sampled when the gate goes high. Otherwise, it is sampled when it has changed sufficiently to select a new note.
2. The input transpose is applied. This simply adjusts the CV in multiples of 12-TET semitones i.e. one twelfth of a Volt.
3. The CV is quantized into the ‘lookup’ scale, yielding a note number.
4. The shift parameter is applied.
5. The note number selects a CV in the ‘output’ scale.
6. The output transpose is applied.

Steps 3 & 5 warrant further explanation, and depend on the selected quantization mode.

In “Nearest” and “Warped” modes, the lookup and output scales are the same. The CV is simply quantized to the nearest note in the scale.

In “Mapped” mode, the lookup scale is 12-TET. The input CV is effectively selecting a standard MIDI note number, with 12 equally spaced semitones per octave. This note number is then used to choose an output CV in the chosen scale, according to its keyboard map.

By way of an example, consider the case where you load a Scala file defining 31-EDO i.e. there are 31 notes per octave. Say you apply a CV from an LFO, ramping up over a couple of Volts.

- In Nearest and Warped modes, you’ll get all 31 notes per octave, and a 1V change in the input will give you a one octave rise in pitch.
- In Mapped mode, the output will depend on the chosen keyboard map (kbm) file. You might

have a keyboard map which chooses one of the 31-EDO pitches per semitone, but still only defines 12 pitches per octave. In this case you'll get your 12 microtonally tuned pitches per octave, and a 1V CV change will give you a one octave pitch change. Or, you might have a keyboard map which lays out all 31 pitches over 31 'semitones', as you might if you wanted to play in 31-EDO from a standard MIDI keyboard and be able to access all 31 pitches. In this case, the LFO in our example would have to rise over 2.5V to cover an octave, passing though all 31 pitches on the way.

Warped mode differs from Nearest mode in that the voltages in the lookup scale are redistributed to be evenly spaced. This is often useful in scenarios where you want to create note patterns from LFOs or similar. Consider the case where you're quantizing to a major scale, and the input is a simple ramp LFO. In Nearest mode (which is how most quantizers work), the differing gaps between the notes (e.g. a tone from D to E but a semitone from E to F) will result in an uneven rhythm as the notes of the scale are passed by the LFO voltage. Warped mode neatly solves this, and gives you a regular rhythm even though the notes are unevenly spaced. (Try it – it's much easier to hear this than it is to imagine it.)

Key, scale, and mode

The Key, Scale, and Mode parameters can be used to constrain the quantized notes to a particular scale. Furthermore, you can select which notes in the chosen scale are used by the quantizer by applying a 'mask' – see below.

The scale-based logic is applied when

- no microtuning is applied, or
- the selected tuning has 12 scale degrees.

So if for example you're using a 19 tone tuning, the key, scale, and mode parameters are ignored.

The available scales are the common selection (see above), plus two 'scales' that use live MIDI input to define the notes to quantize to. When these scales are used, playing MIDI notes adds notes to the selection, so long as any key is held. When all keys are released, the next note played clears the selection and starts a new one.

- 'MIDI (any octave)' allows the quantizer to choose notes as held, but in any octave. For example, if you hold any C and any G on your keyboard, the quantizer is free to choose any C or G.
- 'MIDI (exact)' allows the quantizer to choose only the actual notes played. For example, if you hold C4 and G5, the quantizer may only choose C4 or G5, not any other C or G.

When one of the MIDI scales is used, the Key parameter becomes a simple transpose. For example, if you hold C and G, and set the Key to 'D' (parameter value 2), the quantizer may choose D or A.

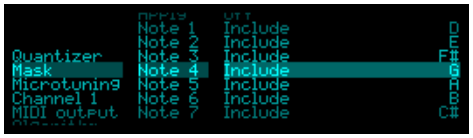
Note that constraint to scales works internally by removing the unwanted notes from the keyboard map, so if you're already using a .kbm file with unmapped notes you may get unexpected results.

Mask

The ‘mask’ parameters can be used instead of, or as well as, the key, scale, and mode parameters just described. If you only want to use the mask parameters, set the scale to chromatic.

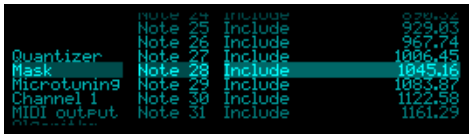
Exactly how these parameters are interpreted depends on whether a 12 degree tuning is being used (i.e. the same criteria as for whether the scale-based logic is applied).

For a 12 degree tuning, the note names are displayed next to the mask parameters, for example in D major, Ionian mode:



To stop the quantizer choosing certain notes, set the mask for the note to ‘Off’ and set ‘Apply’ to ‘On’. For example, to quantize to a pentatonic scale:

If a tuning is chosen with more or less than 12 degrees, the mask parameters refer directly to the degrees of the tuning. For example, 31 EDO:



GUI

The display for the Quantizer has a couple of different screens, depending on what is selected as the current parameter, and on the selected tuning.

If the current parameter is the key, scale, or mode, or one of the mask parameters, and if the 12 degree logic is being applied (see the previous two sections), then a graphical view of one octave is shown:

The blocks represent one octave of notes in a piano key arrangement. The bright coloured notes are the ones chosen for quantization; the darker ones are deselected. The current output note is shown as an unfilled rectangle.

Otherwise, the display shows detailed information about the current quantizer channel.

The top line shows the current input, as a voltage, and as a standard 12-TET MIDI note – a fractional note number, and again as a note name and cents. The second line shows the sampled voltage. The third and fourth lines show the quantized note numbers, first the lookup and then the output. The fifth line shows the output, again as a voltage and as a note number/name/cents. If a Scala file is in use, the tuning of the current note is shown, either in cents or as a ratio. In the examples above, a scale is used which starts on C and which uses A=440 as a reference. The one on the left uses an equal temperament scale, so the quantized note A is 900 cents from the root. The one on the right uses a just intonation, where A, the sixth, uses a ratio of 5/3. Note that in both cases, the output pitch is exactly A5+0, since the scale uses A as the tuning reference. Contrast these with the below:

Now the quantized note is G. The equal temperament scale on the left shows G as 700 cents, and gives G5+0. The just scale on the right shows G as 3/2, and a pitch of G5+17.

Quantizer parameters

Name	Min	Max	Default	Unit	Description
Quantize mode	0	2	0		Sets the quantization mode: ‘Nearest’, ‘Mapped’, or ‘Warped’.
Input transpose	-48	48	0	ST	Sets the input transposition, in 12-TET semitones.
Shift	-48	48	0		Sets the in-scale shift.
Key	-12	12	0		Sets the key (C, D ♭, E, etc.).
Scale	0	10	2		The scale to quantize into. Defaults to Chromatic. See above.
Mode	1	12	1		Sets the mode, that is, the rotation of the notes within the scale. If the selected scale is ‘Major’, the mode is displayed by one of the familiar names Ionian, Dorian, Phrygian, Lydian, Mixolydian, Aeolian, or Locrian.
Output transpose	-48	48	0	ST	Sets the output transposition, in 12-TET semitones.

Output gate mode	0	2	0		Sets the behaviour of the gate/trigger outputs. The options are: ‘Triggers', 'Inv Triggers' (inverted triggers, always high except for a low pulse when the trigger is fired, useful for driving an envelope generator which is generally in sustain but which needs to be retriggered when the chord changes), and 'Gates' (in which case the output is gated by the corresponding gate input). Note that this also applies to any MIDI output.
Gate offset	0.0	10.0	2.0	ms	Offsets (delays) the gate inputs relative to the pitch inputs. This is useful to allow pitch CVs to settle before they are sampled on the rising gate, and also to cope with modules which output both a pitch and gate but change their gate first.
MIDI channel (in)	0	16	1		The MIDI channel to receive on.

Microtuning parameters

The algorithm uses the standard microtuning parameters, as described above.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
CV input	0	64	1		The pitch CV input bus.
Gate input	0	64	0		The (optional) gate input bus.
CV output	0	64	15		The pitch CV output bus. Always uses ‘Replace’ output mode.
Gate output	0	64	0		The gate output bus, according to the ‘Output gate mode’ parameter. Always uses ‘Replace’ output mode.
Change output	0	64	0		The ‘change trigger’ output bus. Fires a trigger pulse when the quantized note changes. Always uses ‘Replace’ output mode.
MIDI channel (out)	0	16	0		The MIDI channel to output notes on.

MIDI output parameters

Name	Min	Max	Default	Unit	Description

Output to breakout	0	1	0		Enables MIDI output to the breakout.
Output to Select Bus	0	1	0		Enables MIDI output to the Select Bus.
Output to USB	0	1	0		Enables MIDI output to USB.
Output to internal	0	1	0		Enables internal MIDI output – that is, MIDI is sent to the other algorithms.
Send pitch bend	0	1	0		If enabled, a pitch bend message is sent with each note on message to adjust the note tuning according to the selected microtuning.
Pitch bend range	1	48	2	ST	Sets the pitch bend range, in semitones. For the “Send pitch bend” function to work properly, this should match the pitch bend range of the receiving synth.

Rectifier

“Full-wave rectifier/absolute value”

File format guid: 'absv'

Specifications:

- Channels, 1-12: The number of bus channels to process.

Description

This simple algorithm computes the absolute value of the input voltage.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Input	0	64	1		The input bus to process.
Enable	0	1	0		Enables the channel. Disabled channels have no effect on the bus.
Output	0	64	0		The output bus. If set to ‘None’, the input bus is used as output, and the mode is always ‘Replace’.
Output mode	0	1	1		The standard Add/Replace mode selector as described above.

Resonator

“It's Rings!”

File format guid: 'resn'

Specifications: None

Description

This algorithm is an implementation of the open source Rings[109] module by Émilie Gillet.

It can be played via MIDI or I2C, or simply via CV/gate as the original.

Any demos or tutorials you may find online should apply just as well to this implementation. The user manual for the original module is here[110]. The documentation below assumes some familiarity with this.

Input normalization

The original Rings module made extensive use of detecting which inputs had jacks plugged in and adjusting its behaviour accordingly. For example, if nothing is plugged into the Strum input, the module generates strums internally from pitch changes or audio transients.

The disting NT has no way of knowing whether its sockets are connected or not, so this algorithm relies on the various 'input' parameters being enabled or not. To take the same example, if the 'Strum input' parameter is set to ‘None’, the algorithm will generate its own strums; if the parameter is something else, the algorithm assumes you're going to supply strums on the input you choose.

Rings parameters

Name	Min	Max	Default	Unit	Description
Mode	0	6	0		Selects the resonator mode.
Synth effect	0	5	0		If the mode is 'Synth', selects the audio effect applied to the synth output.
Polyphony	1	4	1		Sets the number of simultaneous voices.
Coarse tune	-36	24	0	ST	Provides a coarse tuning control.
Fine tune	-100	100	0	cents	Provides a fine tuning control.
Structure	0	127	64		Controls the resonator 'structure'.

109 https://github.com/pichenettes/eurorack/tree/master/rings

110 https://pichenettes.github.io/mutable-instruments-documentation/modules/rings/manual/

Brightness	0	127	64		Controls the resonator 'brightness'.
Damping	0	127	64		Controls the resonator 'damping'.
Position	0	127	64		Controls the resonator 'position'.
Chord	0	10	0		Chooses the chord to use for resonator modes that use one. In the original, set from the Structure knob.
Noise gate	0	1	1		Enables a noise gate on the audio input. Always enabled in the original Rings.
Input gain	-40	12	0	dB	A gain applied to the input audio, before it hits the resonator.

Inputs parameters

Name	Min	Max	Default	Unit	Description
Strum input	0	64	0		Chooses which input bus to use for 'strum'.
V/Oct input	0	64	0		Chooses which input bus to use for the V/octave pitch CV.
Audio input	0	64	0		Chooses which input bus to use for the audio input.
Brightness input	0	64	0		Chooses which input bus to use for 'brightness'.
Frequency input	0	64	0		Chooses which input bus to use for 'frequency'.
Damping input	0	64	0		Chooses which input bus to use for 'damping'.
Structure input	0	64	0		Chooses which input bus to use for 'structure'.
Position input	0	64	0		Chooses which input bus to use for 'position'.

Outputs parameters

Name	Min	Max	Default	Unit	Description
Odd output	0	64	13		The bus to use for the Odd output.
Odd output mode	0	1	0		The Add/Replace mode for the Odd output.

Even output	0	64	13		The bus to use for the Even output.
Even output mode	0	1	0		The Add/Replace mode for the Even output.
Output gain	-40	12	0	dB	The output level (of both the Odd and Even outputs).
Dry gain	-40	12	-40	dB	The level of the input audio mixed into the output(s).

Control parameters

Name	Min	Max	Default	Unit	Description
MIDI mode	0	3	0		Controls the algorithm's response to MIDI. See below.
I2C mode	0	3	0		Controls the algorithm's response to I2C. See below.
MIDI channel	0	16	1		The MIDI channel to receive on.
I2C channel	0	255	1		The I2C channel to receive on.

MIDI and I2C modes

The MIDI mode and I2C mode parameters allow you to choose how MIDI and I2C will be used. Both have the same options:

Off	Notes will not be used.
Pitch	Note on events will set the pitch.
Strum	Note on events will cause a strum.
Pitch & strum	Note on events will set the pitch and cause a strum.

Note that in terms of the input normalization logic described above, activating MIDI or I2C for pitch and/or strum is taken to be equivalent to connecting a CV input for that parameter. E.g. setting the MIDI mode to 'Strum' will stop the algorithm generating its own strums internally.

Reverb

“A general purpose reverb effect”

File format guid: 'revb'

Specifications: None

Description

This algorithm offers a classic algorithmic reverb effect. It does not seek to emulate any particular hardware, or for that matter, any particular physical reverberant space.

Reverb parameters

Name	Min	Max	Default	Unit	Description
Mix	0	100	100	%	Sets the wet/dry mix.
Time	400	30000	1700	ms	Sets the reverb time.
Model	0	3	1		Chooses the reverb model.
Size	1	100	100	%	Sets the size of the reverb space – mainly affects the times of the early reflections.
High damp	0	100	60		Sets the amount of high frequency damping in the reverb tail.
Modulation speed	0.01	5.00	2.50	Hz	Sets the speed of reverb modulation.
Modulation depth	0	100	25		Sets the depth of reverb modulation.
Early gain	-40	6	-12	dB	Sets the output level of the early reflections.
Diffuse gain	-40	6	-6	dB	Sets the output level of the diffuse reflections.

Routing parameters

Name	Min	Max	Default	Unit	Description
Left input	1	64	1		The left audio input bus.
Right input	1	64	2		The right audio input bus.
Left output	1	64	13		The left audio output bus.

Right output	1	64	14		The right audio output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Reverb (Clouds)

“The reverb from Clouds”

File format guid: 'revc'

Specifications: None

Description

This algorithm implements the reverb and diffuser parts of the Clouds[111] module by Émilie Gillet.

The diffuser is a simple network of all-pass filters which ‘smears’ the sound slightly. It is effective at turning a mono source into something that sounds stereo.

Effects parameters

Name	Min	Max	Default	Unit	Description
Diffuser mix	0	100	0	%	The diffuser wet/dry mix.
Reverb mix	0	100	100	%	The reverb wet/dry mix.
Reverb time	0	100	0	%	The reverb time.
Reverb diffusion	0	100	70	%	The reverb diffusion. (In Clouds this is fixed at 70%.)
Reverb low pass	0	100	50	%	The cut-off frequency of a low-pass filter within the reverb.
Reverb input gain	-40	0	-14	dB	The input gain to the reveberator.
Sample rate	0	1	0		Allows you to run the algorithm at the disting NT’s own sample rate (‘Native’) or the sample rate of a real Clouds module (‘Authentic (32kHz)’).

Routing parameters

Name	Min	Max	Default	Unit	Description
Left input	1	64	1		The left audio input bus.
Right input	1	64	2		The right audio input bus.

111 https://github.com/pichenettes/eurorack/tree/master/clouds

Left output	1	64	13		The left audio output bus.
Right output	1	64	14		The right audio output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Rotary

“Rotary speaker emulation”

File format guid: 'roty'

Specifications: None

Description

This algorithm is based on the algorithm of the same name on the disting mk4. You may like to watch the video on that algorithm here[112]. It provides an emulation of a rotary speaker effect.

Rotary parameters

Name	Min	Max	Default	Unit	Description
Mix	0	100	100	%	The wet/dry mix.
Level	-40	0	0	dB	The gain applied to the effect/wet signal.
Speed	0	1023	512		The rotation speed. The second half of the range corresponds to speeds from 0.8Hz to 6.7Hz. There is a dead zone from 414 to 512; below that, the speed gradually slows to a full stop at zero.
Mod depth	0	100	38	%	The depth of the pitch modulation.
Speed slew	0	100	10	%	The slew rate for changes to the speed.
Crossover	0	1024	480		Sets the frequency of a (first order) crossover filter. The part of the signal above the filter frequency is fed into the rotary pitch modulation effect. The bass signal (below the crossover frequency) is not pitch modulated, but is amplitude modulated at a slightly slower rate than the treble modulation. If this is set to ‘Off’, the crossover is disabled and the separate bass modulation is not applied.
Bass mod	0	100	23	%	The depth of the bass modulation. If the crossover is used, but this parameter is zero, this essentially removes the effect entirely from the bass portion of the input, keeping it centre stereo and at unmodulated pitch.

112 https://www.youtube.com/watch?v=fT-1Bceazbs

Routing parameters

Name	Min	Max	Default	Unit	Description
Input	1	64	1		The input bus.
Left output	1	64	13		The left output bus.
Right output	1	64	14		The right output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Sample and Hold

“Simple sample (or track) and hold”

File format guid: 'saho'

Specifications:

- Channels, 1-8: The number of bus channels to process.

Description

This algorithm is a simple sample/track and hold utility. A common gate/trigger input controls the sampling/tracking of a number of signal channels.

There are two modes of operation:

- ‘Sample and hold’ - the output is a sample of the input, taken when the gate/trigger input goes high.
- ‘Track and hold’ - the output follows the input while the gate is high, and freezes when the gate goes low.

Common parameters

Name	Min	Max	Default	Unit	Description
Gate input	1	64	1		The input bus to use for the gate/trigger.
Gate offset	0.0	10.0	2.0	ms	Offsets (delays) the gate input relative to the signal inputs. This is useful to allow signal CVs to settle before they are sampled on the rising gate, and also to cope with modules which output both a pitch and gate but change their gate first.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Input	1	64			The signal input bus to sample/track.
Output	0	64	0		The output bus.
Output mode	0	1	1		The standard Add/Replace mode selector as described above.
Mode	0	1	0		Chooses between ‘Sample and hold’ and ‘Track and hold’.

Sample Player

“A simple sample player”

File format guid: 'samp'

Specifications:

- Triggers, 1-8: The number of individual sample triggers.

Description

This algorithm plays samples from the MicroSD card. It is loosely based on the “SD 6 Triggers” algorithm on the disting EX.

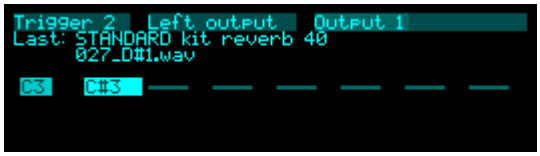
Whereas the “Poly Multisample” algorithm is aimed more at playing pitched instruments, this algorithm is mainly designed for playing simple one-shot samples e.g. drums.

It supports both velocity switches and round robins per sample.

The trigger inputs are velocity sensitive – the voltage of the gate signal is used like the velocity of a MIDI note. 5V corresponds to maximum velocity.

GUI

The display shows the MIDI note names of the samples being played by the various triggers, superimposed on a bar indicating the trigger’s envelope. The folder and filename of the most recently triggered sample is also shown.



Globals parameters

Name	Min	Max	Default	Unit	Description
Gain	-40	24	0	dB	An overall gain to apply in addition to the per-trigger gain.
Round robin mode	0	3	0		The round-robin mode. See above.

Per-trigger parameters

Name	Min	Max	Default	Unit	Description

Folder	1				Sets the folder from which to choose a sample.
Sample	1				Sets the sample within the folder.
Transpose	-60	60	0	ST	Sets the sample tuning in semitones.
Fine tune	-100	100	0	cents	Sets the sample fine tuning.
Gain	-40	24	0	dB	The output level.
Pan	-100	100	0	%	The stereo pan position.
Vel(ocity) depth	0	100	100	%	Sets the amount by which the velocity affects the playback level.
Choke group	0	8	0		The voice's “choke group”. When a voice in a choke group is triggered, it ends the playback of any other voices in the same choke group.
Play	0	1	0		Manually triggers sample playback.
Velocity	1	127	127		The velocity to use when ‘Play’ triggers playback.
Loop	0	2	0		Sets whether the sample will loop. ‘From WAV file’ will loop if the sample has loop points defined in the file – see above – and not otherwise. ‘Off’ and ‘On’ force the sample to loop or not.

Per-trigger setup parameters

Name	Min	Max	Default	Unit	Description
Left output	1	64	13		The left audio output bus.
Right output	0	64	14		The right audio output bus.
Trigger input	0	64	0		The input bus to use as the gate/trigger.
MIDI channel	0	16	1		The MIDI channel to respond to.
MIDI note	-1	127	48		The MIDI note to respond to, or “Any”.
I2C channel	0	255	1		Sets the I2C channel.

Per-trigger envelope parameters

Name	Min	Max	Default	Unit	Description

Envelope	0	1	0		Enables an ADSR volume envelope. If the envelope is disabled, the sample simply plays once until the end.
Attack	0	127	0		Sets the envelope attack time.
Decay	0	127	60		Sets the envelope decay time.
Sustain	0	100	100	%	Sets the envelope sustain level.
Release	0	127	77		Sets the envelope release time.

Sample Players

The algorithms that follow – Sample Player (Clocked), Sample Player (Scrub), Sample Player (Sliced), and Sample Player (Speed) – offer variations on the theme of sample playback. Whereas the Poly Multisample algorithm is primarily designed for playing multisampled instruments (pianos and the like), and the preceding Sample Player algorithm is great for setting up, say, a full drum kit, these algorithms offer more detailed control over manipulating a single sample.

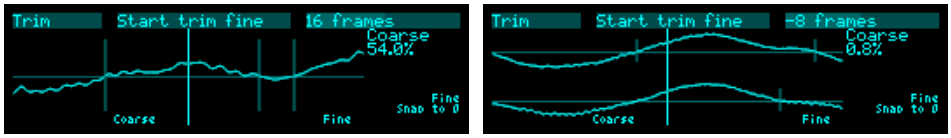
GUI

Each algorithm has unique details to its GUI, but all are based around a graphical view of the sample. Note that this updates as the sample plays – it does not immediately reflect the new sample as soon as it's triggered.

The lines above the waveform view show the folder and filename of the last triggered sample.

GUI – fine sample positions

A number of settings that define positions in the sample (e.g. trim points, or slices in the Sample Player (Sliced) algorithm) are split into a pair of coarse/fine parameters. A custom UI is displayed when editing the fine parameter, which changes the usual behaviour of some of the encoders and buttons.



The waveform is displayed as a line (mono) or two lines (stereo), at a scale of one pixel per sample frame. Zero crossings in the waveform(s) are shown as vertical lines.

The fine position, as the current parameter, is adjusted with the right pot and the right encoder as usual. However, in this view, the coarse position can also be adjusted, with the left encoder. This allows you to edit both parameters without constantly switching between them.

Pressing button 4 also has a special function in this view – it advances the fine position to the next zero crossing. This allows you to rapidly find click-free start/end points in the audio.

Triggering playback

Sample playback can be triggered by

- a trigger on the trigger input bus.
- a MIDI note.
- the Play parameter (which can be mapped, if desired).

Sample/folder selection

Using the ‘Confirm change’ parameter, you can choose whether folder or sample changes take effect immediately, or require you to click the right encoder to confirm.

Retrigger/restart on change

The default behaviour is that when you change the sample selection, nothing happens until you trigger playback. The ‘Retrigger on change’ parameter allows sample selection to instantly retrigger the sample if it’s already playing. This is great if you have, say, a selection of fairly long or looping samples, and you want to scrub through them, like turning the tuning dial on a radio. The ‘Restart on change’ parameter lets you choose whether samples retriggered in this way start from the beginning, or from an equivalent position within the sample. The default is the latter – if you’re one minute into a four minute sample, and switch to another sample, the playback will pick up one minute into the new sample.

Start position

When a sample is triggered, it starts playback at a position determined by the parameters grouped under ‘Start parameters’. There are two offsets, which are added – the ‘Trim’ and the ‘Offset’.

‘Trim’ is simple – it just throws away a certain amount of the start of the sample.

‘Offset’ applies to the remainder of the sample, after the trim. The reason for needing both is that the offset can be determined as a percentage or fraction of the remaining sample time. Say you have a nice one bar loop but there’s a short silence before the recording begins – you can use the trim to get rid of the initial silence, and then have convenient metrical control over the offset.

Note that the ‘Offset div’ parameter is a division of the sample length, not a tempo-based quantity. So for example an ‘Offset div’ of 1/4 means a quarter of the sample, however long the sample is, not a quarter note at the current tempo. (The exception is the Sample Player (Clocked) algorithm where the ‘Offset div’ is indeed tempo-based.)

Time stretching

The ‘Warp’ parameter sets how samples are re-pitched and/or stretched when the speed and/or pitch changes.

In the default ‘Re-Pitch’ mode, the pitch and speed of samples is linked: to play a sample faster or slower, it is pitched up or down respectively; to change the pitch of a sample, it is simply played faster

or slower.

The ‘Stretch’ mode allows you to play samples faster or slower without changing the pitch, and to pitch them up and down without changing the speed. It does this by applying a time-domain pitch shifter to account for the pitch change when changing the playback speed.

The ‘Phase vocoder’ mode is similar to ‘Stretch’ except that it uses a spectral-domain method. It is only able to play samples at their original speed or slower, not faster. It will generally give a better result than ‘Stretch’ when playing samples much slower than normal speed.

Recording (sampling)

If the algorithm has been allocated some memory into which to do so (via its specifications), it can record audio and play that from memory instead of playing files from the MicroSD card – that is to say, you can sample, in the traditional Akai S1000 sense.

The steps to do this are:

- Allocate recording time in the specifications, when adding the algorithm or via the Respecify menu.
- Assign the input busses. You can assign a single bus for mono recording or two busses for stereo recording.
- Use the ‘Record’ parameter to actually do the recording.
- Set ‘Use recording’ to On to use the recording instead of the currently selected sample on the MicroSD card.

You can save your samples to the MicroSD card for later use. This is done via the algorithm’s menu, which has a ‘Save recording’ submenu. The ‘Apply trim’ option allows you to save the sample with the start & end trims removed; otherwise the entire recording is saved.

The file is saved to a location defined by the ‘Save to folder’ parameter, which is used as a folder name within the top level ‘/samples’ folder. The filename is based on the ‘Save as’ parameter, made unique with the addition of a numeric suffix. The actual filename saved is displayed when the save is complete.

Audio threshold-triggered recording

Once you’re set up to record, setting the ‘Triggered’ parameter to on enables automatic record triggering based on an audio threshold level (set by the ‘Threshold’ parameter).

With Triggered set to on, when you set Record to on the algorithm will listen for audio on the selected input(s), and display “Triggered record pending”. When the audio level reaches the threshold, recording will begin.

Parameters

The following parameters are common to the various Sample Player algorithms.

Sample parameters

Name	Min	Max	Default	Unit	Description
Folder	1				Sets the folder from which to choose a sample.
Sample	1				Sets the sample within the folder.
Gain	-40	24	0	dB	The output level.
Pan	-100	100	0	%	The stereo pan position.
Vel(ocity) depth	0	100	100	%	Sets the amount by which the velocity affects the playback level.
Play	0	1	0		Manually triggers sample playback.
Velocity	1	127	127		The velocity to use when ‘Play’ triggers playback.
Warp	0	2	0		The timestretch mode: ‘Re-Pitch’, ‘Stretch’, or ‘Phase vocoder’.
Shift	0	1	0		The pitch shift method, if Warp is Stretch: ‘2-phase’ or ‘3-phase’.
Grains	1	341	50	ms	The maximum length of the delay line used by the pitch shifter.

Setup parameters

Name	Min	Max	Default	Unit	Description
Round robin mode	0	3	0		The round-robin mode. See above.
Loop	0	2	0		Sets whether the sample will loop. ‘From WAV file’ will loop if the sample has loop points defined in the file – see above – and not otherwise. ‘Off’ and ‘On’ force the sample to loop or not.
Vel(ocity) depth	0	100	100	%	Sets the amount by which the velocity affects the playback level.
Confirm change	0	3	0		Sets whether the folder and/or sample changes immediately or requires confirmation.
Retrigger on change	0	1	0		If enabled, changing the sample selection while the sample is playing retriggers the sample.

Restart on change	0	1	0		Chooses where to start playback from if the sample is retriggered as a result of ‘Retrigger on change’. If set, the sample restarts from the beginning; otherwise, the playback position within the sample is preserved.

Trim parameters

Name	Min	Max	Default	Unit	Description
Start trim coarse	0.0	100.0	0.0	%	Sets an amount to trim off the start, as a percentage of the sample length.
Start trim fine	-500	500	0		Adjusts the amount to trim off the start, in sample frames.
End trim coarse	0.0	100.0	0.0	%	Sets an amount to trim off the end, as a percentage of the sample length.
End trim fine	-500	500	0		Adjusts the amount to trim off the end, in sample frames.
Offset type	0	2	0		Sets how the start offset will be determined.
Offset ms	0	32767	0	ms	The start offset, if the type is ‘Milliseconds’.
Offset %	0.0	99.9	0.0	%	The start offset, if the type is ‘Percent’.
Offset div	0	19	4		The start offset divisor, if the type is ‘Division’.
Offset mult	0	128	0		The start offset multiplier, if the type is ‘Division’.

Envelope parameters

Name	Min	Max	Default	Unit	Description
Envelope	0	1	0		Enables an ADSR volume envelope. If the envelope is disabled, the sample simply plays once until the end.
Attack	0	127	0		Sets the envelope attack time.
Decay	0	127	60		Sets the envelope decay time.
Sustain	0	100	100	%	Sets the envelope sustain level.
Release	0	127	77		Sets the envelope release time.

MIDI parameters

Name	Min	Max	Default	Unit	Description
MIDI channel	0	16	1		The MIDI channel to respond to.
MIDI note	-1	127	48		The MIDI note to respond to, or “Any”.
I2C channel	0	255	1		Sets the I2C channel.

Record parameters

Name	Min	Max	Default	Unit	Description
Record	0	1	0		Set to ‘On’ to activate recording and ‘Off’ to stop recording.
Record input L	0	64	0		The bus to use for the left, or mono, audio input.
Record input R	0	64	0		The bus to use for the right audio input.
Record gain	-40	24	0	dB	A gain to apply when recording.
Use recording	0	1	0		Set to ‘On’ to use the recording instead of a sample on the MicroSD card. Ignored if there is no recording.
Save to folder					Sets the folder name in which to save the sample to the MicroSD card.
Save as					Sets the base file name to use when saving the sample to the MicroSD card.

Routing parameters

Name	Min	Max	Default	Unit	Description
Trigger input	0	64	0		The input bus to use as the gate/trigger.
Left output	1	64	13		The left audio output bus.
Right output	0	64	14		The right audio output bus.

Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Sample Player (Clocked)

“Plays samples synced to a clock”

File format guid: 'samc'

Specifications:

- Record time: 0-60 seconds: The maximum length of audio that can be recorded (sampled).

Description

This algorithm plays samples, stretching or shrinking them to fit the tempo provided by a clock input, or MIDI. It is the spiritual successor to the disting mk4 Clocked Audio Playback algorithm, but works somewhat differently. Still, it might be fun to review the video (actually for the disting mk3) here[113].

Please review the section on the features common to the various Sample Players, above.

To use the algorithm, first give it a clock, either from an input bus, or by telling it to follow MIDI clock. You also need to trigger the sample – you can do this via a bus, or via MIDI, or manually, but it’s also possible to have the algorithm continuously trigger the sample in a loop via the ‘Auto trigger’ parameter.

The algorithm looks at the length of the chosen sample, and uses the current tempo and time signature to figure out a number of bars that best fits. It then adjusts the playback speed so that the sample will fit that number of bars. You can use the ‘Speed tune’ parameter to adjust the speed – this is particularly useful if the sample is not actually a full number of bars long.

GUI

As with the other Sample Players, the display shows the waveform of the sample, which is updated as the sample plays. Unlike the other Sample Players, which always show the full length of the sample, this algorithm’s display shows the full length of the time of the calculated loop. How the sample fits into this time depends on, among other things, the ‘Speed tune’ parameter.

Above the waveform are tick marks for the bars and beats.

To the right, the algorithm shows the length of the sample (in seconds), the calculated number of bars that the sample would play for at its natural speed, and the number of bars that the algorithm has chosen to fit the sample to.

113 https://www.youtube.com/watch?v=9wZHqDfXvGg

Choice of input clock divider

The algorithm recalculates the playback speed every time it receives a clock. To track varying tempos, it's therefore better to have fairly frequent clocks. However, longer clocks tend to follow the tempo more accurately (since the longer measurement interval reduces jitter), so if your tempo is more or less constant it might be better to choose a slower input clock.

Sample parameters

In addition to the common parameters, this algorithm adds:

Name	Min	Max	Default	Unit	Description
Speed tune	50.00	200.00	100.00	%	Adjusts the playback speed, relative to the one calculated by the algorithm.

Sync parameters

Name	Min	Max	Default	Unit	Description
Input clock div	0	19	4		Sets the interpretation of the input clock pulse.
Time sig num	1	99	4		The time signature numerator.
Time sig denom	0	4	2		The time signature denominator: one of 1, 2, 4, 8, or 16.
Auto trigger	0	2	0		Sets whether the sample will automatically be triggered by the clock. This can be at a fixed interval, or set automatically by the length of the loop.
Auto divisor	0	19	15		Sets how often the sample will be triggered, if 'Auto trigger' is 'Fixed'.

MIDI parameters

In addition to the common parameters, this algorithm adds:

Name	Min	Max	Default	Unit	Description
Clock	0	1	0		Whether to follow MIDI clock.
Clock divisor	0	19	4		Sets how often the MIDI clock will cause an internal clock event. This should normally match the 'Input clock div' parameter.

Routing parameters

In addition to the common parameters, this algorithm adds:

Name	Min	Max	Default	Unit	Description
Clock input	0	64	0		The bus to use as the clock input.
Reset input	0	64	0		The bus to use as the reset input.
Reset mode	0	1	0		Sets the type of reset signal: ‘Trigger’ or ‘Run/stop’.

Sample Player (Scrub)

“Plays samples with scrub control”

File format guid: 'samb'

Specifications:

- Record time: 0-60 seconds: The maximum length of audio that can be recorded (sampled).

Description

This algorithm is based on the Audio Playback with Scrub algorithm on the disting mk4. You might like to watch the video on that algorithm, here[114]. From the disting mk4 manual:

“In this algorithm the sample playback position is directly driven from the [input]. Imagine a piece of audio tape passing over a tape head – in other algorithms, that tape is constantly moving at some speed or other as if driven by a motor, but in this algorithm you're basically dragging the tape back and forwards over the playhead yourself.”

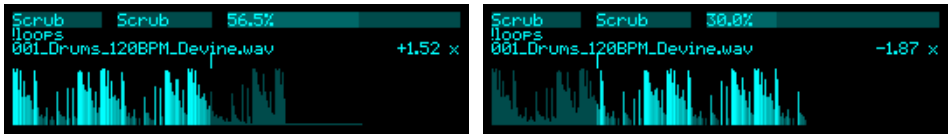
You can scrub the audio using a CV input, or the Scrub parameter (which can of course be mapped).

Remember that you need to trigger the sample before any playback will occur, and again to change the sample (unless you have ‘Retrigger on change’ on).

Please review the section on the features common to the various Sample Players, above.

GUI

The display shows the waveform and the playhead position as a check mark above it:



The current playback speed is also shown. Positive values indicate forwards playback; negative values indicates reversed playback.

Scrub parameters

Name	Min	Max	Default	Unit	Description
Scrub	0.0	100.0	50.0	%	Sets the scrub position.
Max speed	1	100	16		Sets the maximum playback speed.

114 https://www.youtube.com/watch?v=-Z3v_W-s2rI

Slew	0	1000	400		Sets the amount of slew applied to the scrub position.
Min CV	-10	10	-5	V	Sets the CV voltage corresponding to scrubbing to the start of the sample.
Max CV	-10	10	5	V	Sets the CV voltage corresponding to scrubbing to the end of the sample.

Routing parameters

In addition to the common parameters, this algorithm adds:

Name	**Min**	**Max**	**Default**	**Unit**	**Description**
Scrub input	0	64	0		The bus to use as the scrub CV input.

Sample Player (Sliced)

“Plays sliced samples”

File format guid: 'saml'

Specifications:

- Record time: 0-60 seconds: The maximum length of audio that can be recorded (sampled).

Description

This algorithm allows you to place a number of marker points on a sample, and conveniently trigger playback from the various marker points. It’s your go-to algorithm for breakbeat slicing.

Please review the section on the features common to the various Sample Players, above.

Triggering slices

Once the markers have been placed, the slices can be triggered either by MIDI notes or CV/gate.

If triggering by MIDI, the ‘MIDI base note’ parameter sets the MIDI note number for slice 1. Subsequent slices are triggered by the following MIDI notes. So if say slice 1 is on C4, slice 2 will be C♯4, slice 3 on D4, etc. The MIDI note below the base note (B3 in our example) stops playback.

If triggering by CV/gate, the situation is very similar. A pitch CV is provided to the ‘Slice input’ bus, and a trigger or gate to the ‘Trigger input’. The slice CV is scaled such that a standard 1V/octave pitch CV selects a new slice on each semitone. A CV outside of the range of active slices stops playback.

Stepping through the slices

A trigger signal on the ‘Advance input’ will advance the ‘Slice’ parameter by one, looping back to 1 when it hits the value set by the ‘Slices’ parameter. A trigger signal on the ‘Reset input’ will return the ‘Slice’ parameter to 1.

You can assign the Advance and Trigger inputs to the same bus, so that an input trigger will always play the next slice; or you can keep them separate, allowing you to trigger slices multiple times (or not at all) before advancing.

Automatic loop playback

If you give the algorithm a clock source (a clock input or MIDI clock) and enable ‘Auto trigger’, it will automatically trigger the sample in a loop, triggering each slice at the appropriate time to match the sample length to the tempo.

GUI

The display shows the sample waveform with the slices indicated as check marks above:

The last triggered slice is shown as the large number bottom right. Above this is the sample length (in seconds), the sample length in bars at the current tempo, and the calculated number of bars that the sample will loop over, if a clock is supplied and ‘Auto trigger’ is enabled.

Menu functions

A number of functions for manipulating slices are available via the ‘Sample Player (Sliced)’ menu.

Reset slices

This menu resets the slices to a chosen number of equally spaced positions.

Detect transients

This menu scans the sample for transients in the audio and automatically places slice markers accordingly.

Load from WAV

This menu loads any markers stored in the sample file, if any, and uses them to place the slices.

If ‘Always include start’ is enabled, a slice marker will always be added for the start of the file, even if the markers in the sample file do not include a marker there.

Save to WAV

This menu replaces the sample file with a new one that includes the currently defined slice positions as markers.

Before replacing the file, the existing one is moved to the backups folder on the MicroSD card.

Sample parameters

In addition to the common parameters, this algorithm adds:

Name	Min	Max	Default	Unit	Description
Slice	1	24	1		Offsets the choice of which slice to play. Provided as a convenience when triggering playback with the ‘Play’ parameter.
Slices	1	24	24		Sets the number of slices in use.

Speed	-200.0	200.0	100.0	%	Sets the playback speed. Multiplied by the per-slice Speed parameter.
Transpose	-60	60	0	ST	Transposes the playback speed in semitones (across all slices).
Fine tune	-100	100	0	cents	Adjusts the playback speed in cents (across all slices).

Setup parameters

In addition to the common parameters, this algorithm adds:

Name	Min	Max	Default	Unit	Description
Stop at slice end	0	1	0		If enabled, playback will stop when it reaches the next slice. Otherwise, playback continues to the end of the sample.
Detect transients	0	1	0		When this parameter changes from 0 to 1, it invokes the ‘Detect transients’ function exactly as if the user had selected it from the menu.

MIDI/Sync parameters

In addition to the common MIDI parameters, this algorithm adds:

Name	Min	Max	Default	Unit	Description
Clock source	0	2	0		Sets the clock source: ‘Off’, ‘External’, or ‘MIDI’.
Auto trigger	0	2	0		Sets whether the sample will automatically be triggered by the clock. This can be at a fixed interval, or set automatically by the length of the loop.
Auto divisor	0	19	15		Sets how often the sample will be triggered, if ‘Auto trigger’ is ‘Fixed’.
Time sig num	1	99	4		The time signature numerator.
Time sig denom	0	4	2		The time signature denominator: one of 1, 2, 4, 8, or 16.

Routing parameters

In addition to the common parameters, this algorithm adds:

Name	Min	Max	Default	Unit	Description
Slice input	0	64	0		The bus to use for the slice CV input (V/oct, scaled at one semitone per slice).
Clock input	0	64	0		The bus to use for the clock input (24 ppqn).
Run/stop input	0	64	0		The bus to use for the run/stop input.
Advance input	0	64	0		The bus to use for the advance input (to step through the slices).
Reset input	0	64	0		The bus to use for the reset input (to reset to slice 1).

Slice parameters

Name	Min	Max	Default	Unit	Description
Coarse	0.0	100.0	0.0	%	The coarse slice position, as a percentage of the sample length.
Fine	-500	500	0		The fine slice position, in sample frames.
Speed	-200.0	200.0	100.0	%	The playback speed for the slice.

Sample Player (Speed)

“Plays samples with speed/pitch control”

File format guid: 'sams'

Specifications:

- Record time: 0-60 seconds: The maximum length of audio that can be recorded (sampled).

Description

This algorithm plays samples with flexible control over the playback speed, including the possibility of playing the sample backwards. It provides the features of a number of disting mk4 algorithms, including Audio Playback, Audio Playback with V/Oct, Audio Playback with Z Speed, and Audio Playback with Reverse.

Please review the section on the features common to the various Sample Players, above.

Sample parameters

In addition to the common parameters, this algorithm adds:

Name	Min	Max	Default	Unit	Description
Speed	-200.0	200.0	100.0	%	Sets the playback speed. Negative values play backwards.
Transpose	-60	60	0	ST	Transposes the playback speed in semitones.
Fine tune	-100	100	0	cents	Adjusts the playback speed in cents.

Routing parameters

In addition to the common parameters, this algorithm adds:

Name	Min	Max	Default	Unit	Description
Speed input	0	64	0		The bus to use for speed control, scaled as 5V for 100%.
Pitch input	0	64	0		The bus to use for 1V/octave pitch control.

Saturation

“Soft-clipping saturation”

File format guid: 'satu'

Specifications:

- Channels, 1-8: The number of bus channels to process.

Description

This algorithm implements a simple soft-saturation effect. It can be used simply for creative tone-shaping, but it is particularly useful for avoiding harsh digital clipping when signals get loud. (It is the same processing that is built into many of the disting EX algorithms as a final “output bus” limiter.)

Globals parameters

Name	Min	Max	Default	Unit	Description
Pre gain	-40	24	0	dB	A gain to apply before the saturation.
Post gain	-40	6	0	dB	A gain to apply after the saturation.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Input	0	64	13		The input (and output) bus.
Enable	0	1	0		Enables saturation on this channel.
Width	1	12	1		The number of busses to process, starting from the Input. For example, for a stereo signal, set this to 2.

Seaside Jawari

“Tanpura synth by Seaside Modular”

File format guid: 'ssjw'

Specifications: None

Description

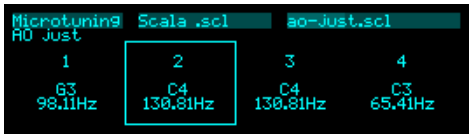
This algorithm is based on the open source Jawari[115] VCV Rack module by Seaside Modular. It is a simulation of the tanpura, the Indian drone instrument.

The tanpura is a four-stringed instrument, without frets. The four strings are typically played in sequence, in a continuous loop, and this is what this algorithm does. Each time it gets a strum (from the strum input bus, or via the strum parameter) it advances to and plays the next string.

The second, third, and fourth strings of the tanpura are generally tuned in octaves, with the fourth string an octave below the other two. The first string is tuned to another note of the scale, very often the fifth. For a completely authentic sound, you might consider using the microtuning support to tune the strings to just intervals.

GUI

The display shows the tuning of the four strings, with the last string played highlighted.



Jawari parameters

Name	Min	Max	Default	Unit	Description
Bridge shape	0.000	1.000	0.500		Sets the blend between the raw string pluck and a comb filtered version.
Tuning (1st string)	0	11	7		Sets the tuning of the first string, in semitones above the fourth string.
Transpose	-36	36	0	ST	Transposes the whole instrument.
Fine tune	-100	100	0	cents	Fine tuning control for the whole instrument.
Strum	0	1	0		Causes a strum when changed from 0 to 1.

115 https://github.com/abluenautilus/SeasideModularVCV

Reset	0	1	0		Resets the sequence so that the next strum plays the first string.
Velocity	1	127	127		Sets the strum velocity.

Tweaks parameters

Name	Min	Max	Default	Unit	Description
Damping	0.000	1.000	0.995		The damping factor for the string model.
Length	0.000	2.000	1.000	s	The decay time for the string model.
Bounce count	1	10000	1200		The number of bridge bounces after the initial strum.
Strum level	0.0	100.0	25.0	%	The relative strength of the strum vs the bounces.
Bounce level	0.0	1000.0	150.0	%	The relative strength of the bounces vs the strum.
Start harmonic	1	20	3		The emphasised harmonic at the start of the bounces.
End harmonic	1	20	10		The emphasised harmonic at the end of the bounces.
Strum type	0	1	0		The shape of the initial strum applied to the string model: Flat or Ramped.

Microtuning parameters

The algorithm uses the standard microtuning parameters, as described above.

Routing parameters

Name	Min	Max	Default	Unit	Description
Strum	0	64	0		The bus to use for a strum trigger.
Bridge shape	0	64	0		The bus to use for CV control over the ‘bridge shape’.
Reset	0	64	0		The bus to use for a reset trigger.
V/oct main	0	64	0		The bus to use for pitch control over the instrument as a whole.
V/oct 1st string	0	64	0		The bus to use for pitch control over the tuning of the first string.

Output	1	64	13		The output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Shift Register Random

“Generates random CVs”

File format guid: 'srra'

Specifications: None

Description

This algorithm generates random CVs via the popular rotating shift register method, often known simply as a “Turing Machine” after this module[116].

The joy of this method is that it generates a loop of CVs, with a controllable likelihood of change, including the possibility to lock the loop so it does not change.

On each clock, the shift register rotates and a new CV is output. On each rotation, there is the possibility that one bit of the shift register will be flipped, changing the pattern.

This algorithm can output either CVs, or triggers, or both. You may like to feed the CVs into the Quantizer algorithm to generate random melodies.

GUI

The display shows a graphical representation of the shift register. If the unrandomness is ±100%, it also shows “<LOCKED>”.

Register parameters

Name	Min	Max	Default	Unit	Description
Unrandomness	-100	100	0	%	Sets the likelihood of the pattern not changing on each clock. At 0%, there is a 50:50 chance of a bit flip, which is the most random setting. At 100%, there will never be a fit flip, so the pattern is locked. At -100% there will always be a bit flip, which has the effect of locking the pattern with a repeat count of twice the length.
Direction	0	1	0		Sets whether the pattern moves forwards or in reverse when clocked.

116 https://www.musicthing.co.uk/Turing-Machine/

Length	1	32	8		Sets the length of the shift register, and so sets the number of steps in the sequence.

CV parameters

Name	Min	Max	Default	Unit	Description
Scale	-20.0	20.0	10.0	V	Scales the output CV.
Offset	-10.0	10.0	0.0	V	Offsets the output CV.

Trigger parameters

Name	Min	Max	Default	Unit	Description
Cause	0	4	0		Chooses what will cause a trigger to be generated: ● The new bit value is high. ● The new bit value is low. ● The bit changes. ● The bit goes from low to high. ● The bit goes from high to low.
Type	0	1	0		Chooses whether the output trigger will be of a fixed length, or calculated as a percentage of the clock length.
Length (ms)	1	100	10	ms	The trigger length, if fixed length is chosen.
Length (%)	1	100	50	%	The trigger length, if percentage of clock is chosen.

Routing parameters

Name	Min	Max	Default	Unit	Description
Clock input	0	64	1		Chooses the clock input bus.
Modify input	0	64	0		Chooses the “modify” input bus. If this gate input is high, the pattern will always be modified on a clock, even if the pattern is locked.
CV output	0	64	15		The output bus for the CV.
Trigger output	0	64	0		The output bus for the trigger.

CV output mode	0	1	0		The Add/Replace mode for the CV bus.
Trigger output mode	0	1	0		The Add/Replace mode for the trigger bus.

MIDI parameters

Name	Min	Max	Default	Unit	Description
Follow MIDI clock	0	1	0		Sets whether the pattern generator follows MIDI clock.
Divisor	0	19	4		Sets the clock divisor when following MIDI clock.
Output to breakout	0	1	0		Enables MIDI output to the breakout.
Output to Select Bus	0	1	0		Enables MIDI output to the Select Bus.
Output to USB	0	1	0		Enables MIDI output to USB.
Output to internal	0	1	0		Enables internal MIDI output – that is, MIDI is sent to the other algorithms.
MIDI channel	0	16	0		The MIDI channel on which to send a note when a trigger fires.
MIDI note	0	127	0		The MIDI note number to send.

Slew rate limiter

“Smooths CVs and creates glissandos”

File format guid: 'slew'

Specifications:

- Channels, 1-8: The number of bus channels to process.

Description

This algorithm is a simple slew rate limiter, offering both logarithmic and linear slew.

You can choose to use a single slew rate for both rising and falling signals, or specify them separately. You can also choose to use the same rates for all the busses, or specify them individually.

Common parameters

Name	Min	Max	Default	Unit	Description
Through	0	1	0		If enabled, the algorithm behaves as if all slew times were set to zero.
Up/shared slew	0	1000	0		The slew time for both, or only rising, signals.
Down slew	0	1000	0		The slew time for falling signals.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Type	0	1	0		Chooses Logarithmic or Linear slew rate limiting.
Control	0	3	0		Chooses which controls will set the slew times. ● ‘Single’ - use ‘Up/shared slew’ for both slew times. ● ‘Dual’ - use separate slew times for rising and falling signals. ● ‘Common single’ and ‘Common dual’ - as above but using the shared times from the Common parameters.
Up/shared slew	0	1000	0		The slew time for both, or only rising, signals.
Down slew	0	1000	0		The slew time for falling signals.

Input	1	64	1		The input bus.
Output	0	64	0		The output bus. If set to ‘None’, the input bus is used as output, and the mode is always ‘Replace’.
Output mode	0	1	1		The standard Add/Replace mode selector as described above.

Slope Detector

“Detects slopes”

File format guid: 'sldt'

Specifications:

- Channels, 1-12: The number of detector channels.

Description

This algorithm provides slope detection, a classic modular synthesizer function that has been part of the synthesist’s toolkit since the very early days of modular.

It takes a CV input, and provides three outputs – simple gates that tell you whether the signal is rising, falling, or steady.

If you would prefer to have trigger outputs (e.g. a trigger that tells you when a signal has started to rise), you can feed the outputs from this algorithm into a Delayed Function algorithm (above) to convert the gates into triggers.

If you need simple inverse outputs (e.g. say you want ‘changing’ instead of ‘steady’), use a Logic algorithm (above).

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Enable	0	1	0		Enables the channel.
Input	1	64	1		The input bus to process.
Rising output	0	64	15		The output bus for the ‘rising’ signal.
Rising output mode	0	1	1		The standard Add/Replace mode selector as described above, for the ‘rising’ signal.
Steady output	0	64	15		The output bus for the ‘steady’ signal.
Steady output mode	0	1	1		The standard Add/Replace mode selector as described above, for the ‘steady’ signal.
Falling output	0	64	15		The output bus for the ‘falling’ signal.

Falling output mode	0	1	1		The standard Add/Replace mode selector as described above, for the ‘falling’ signal.
Sensitivity	1	1000	100	ms	The detector sensitivity.

Spectral Conquest

“Spectral (FFT) based audio manipulation”

File format guid: 'spcn'

Specifications:

- Max block size, 6-11: The maximum processing block size.
- Polysynth voices, 1-24: The number of voices for the ‘polysynth’ section.
- Stereo: Whether the algorithm is mono or stereo.

Description

This algorithm is based on our VST plug-in of the same name, which you can read more about here[117]. Originally released in 2010, Spectral Conquest experienced a surge in popularity in 2022 when it was prominently mentioned by producer Fred Again in a podcast[118].

From the website:

“Spectral Conquest is an effect plug-in that lets you directly manipulate the frequency spectrum of audio signals.

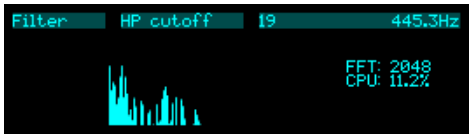
“The incoming signal is analysed via a FFT (Fast Fourier Transform), resulting in the familiar spectrum plot where the full audio spectrum is split into a number of narrow frequency bands. The outgoing audio is then regenerated by an Inverse FFT. The interesting bit is that the plug-in lets you modify the frequency spectrum in between.”

The plug-in lets you interactively sculpt the frequency spectrum with a mouse. This is obviously quite hard to achieve on a Eurorack module, so instead the algorithm provides a number of ways of manipulating the spectrum from a relatively small number of parameters, including playing the spectrum from a keyboard (a good way to set up the chordal resonance which Fred Again describes).

The various processes are applied in the order implied by the parameter page ordering (shift, then gate etc.), and may be used at the same time.

GUI

The display shows the frequency spectrum of the output signal. It also shows the current FFT size, and the module’s total CPU usage (NB not the algorithm’s CPU usage, the total usage).



117 https://www.expert-sleepers.co.uk/spectralconquest.html

118 https://tapenotes.co.uk/project/tn105-fred-again

Analysis parameters

Name	Min	Max	Default	Unit	Description
Latency	0	5	4		Sets the processing latency (which indirectly sets the FFT size).
Sample rate	0	3	0		Sets the sample rate reduction (which indirectly sets the FFT size, and affects the bandwidth).

Mix parameters

Name	Min	Max	Default	Unit	Description
Dry gain	-40	24	-40	dB	The output level of the dry signal.
Wet gain	-40	24	0	dB	The output level of the wet (processed) signal.

Shift parameters

Name	Min	Max	Default	Unit	Description
Shift enable	0	1	0		Enables a spectral shift – the spectral energy is shifted by a number of bins up or down.
Shift			0		The number of bins to shift by.
Reverse	0	1	0		Reverses the spectrum (high frequencies become low frequencies and *vice versa*).

Gate parameters

Name	Min	Max	Default	Unit	Description
Gate enable	0	1	0		Enables a spectral gate – each bin is zeroed unless it falls within the gate range.
Upper threshold	-140.0	6.0	0.0	dB	Frequency bins with a level above this threshold will be zeroed.
Lower threshold	-140.0	6.0	-140.0	dB	Frequency bins with a level below this threshold will be zeroed.

Comb parameters

Name	Min	Max	Default	Unit	Description

Comb enable	0	1	0		Enables a spectral comb filter. ‘Width’ out of every ‘step’ bins are zeroed.
Comb step	2		2		Sets the number of bins at which the pattern repeats.
Comb width	1		1		Sets the number of bins out of every repeat which are zeroed. The effect will be more pronounced when this is close to ‘step’.
Comb offset	0		0		Offsets the start point of the repeating pattern i.e. it sweeps the comb frequencies up & down.

Sieve parameters

Name	**Min**	**Max**	**Default**	**Unit**	**Description**
Harmonic enable	0	1	0		Enables a harmonic sieve. Only frequencies which are harmonics of the chosen base frequency will be retained.
Harmonic base	0.0	2000.0	100.0	Hz	The base frequency for the harmonic sieve.
Harmonic count	1	40	20		The number of harmonics to retain.
Chord enable	0	1	0		Enables a chord sieve. Only frequencies which are in the chosen chord will be retained.
Chord root	0	127	48		The chord root note, as a MIDI note number.
Chord tonic	-40	0	0	dB	The level of the chord tonic.
Chord min 2nd	-40	0	-40	dB	The level of the chord minor 2nd.
Chord maj 2nd	-40	0	-40	dB	The level of the chord major 2nd.
Chord min 3rd	-40	0	-40	dB	The level of the chord minor 3rd.
Chord maj 3rd	-40	0	-40	dB	The level of the chord major 3rd.
Chord 4th	-40	0	-40	dB	The level of the chord 4th.
Chord tritone	-40	0	-40	dB	The level of the chord tritone.
Chord 5th	-40	0	-40	dB	The level of the chord 5th.

Chord min 6th	-40	0	-40	dB	The level of the chord minor 6th.
Chord maj 6th	-40	0	-40	dB	The level of the chord major 6th.
Chord min 7th	-40	0	-40	dB	The level of the chord minor 7th.
Chord maj 7th	-40	0	-40	dB	The level of the chord major 7th.
Chord octaves	1	10	1	dB	The number of octaves over which to repeat the chord pattern.

Polysynth parameters

Name	Min	Max	Default	Unit	Description
Polysynth enable	0	1	0		Enables a sieve based on notes played by CV/gate, MIDI, or I2C. Please refer to the section “Common polysynth features” above.

Filter parameters

Name	Min	Max	Default	Unit	Description
LP enable	0	1	0		Enables a low-pass filter.
LP cutoff	0		0		The low-pass filter cutoff frequency, as a bin number.
HP enable	0	1	0		Enables a high-pass filter.
HP cutoff	0		0		The high-pass filter cutoff frequency, as a bin number.
BP enable	0	1	0		Enables a band-pass filter.
BP low	0		0		The low pass band frequency of the band-pass filter, as a bin number.
BP high	0		0		The high pass band frequency of the band-pass filter, as a bin number.
Notch enable	0	1	0		Enables a notch (band-reject) filter.
Notch low	0		0		The low stop band frequency of the notch filter, as a bin number.

Notch high	0		0		The high stop band frequency of the notch filter, as a bin number.

Routing parameters

Name	Min	Max	Default	Unit	Description
Left/mono input	1	64	1		The left or mono input audio bus.
Right input	1	64	2		The right input audio bus (if the algorithm is set to stereo in the specifications).
Left/mono output	1	64	13		The left or mono output audio bus.
Right output	1	64	14		The right output audio bus (if the algorithm is set to stereo in the specifications).
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Spectral Freeze

“An audio ‘freeze’ effect”

File format guid: 'spfz'

Specifications:

- Voices, 1-8: The number of simultaneous voices.
- History, 4-2048: The length of the history stored for each voice.
- FFT size, 8-12: The size of the FFT used internally.

Description

This algorithm is an audio ‘freeze’ effect – it captures sound at a moment in time and sustains it indefinitely. The ‘spectral’ part of the name indicates that it works in the frequency domain – unlike simple loopers or granular effects, which work in the time domain, this algorithm captures the spectrum of a sound (using an FFT) and uses that to resynthesize audio[119].

As well as completely freezing sound, the algorithm can actually play it back at any desired speed (i.e. it does time stretching) and pitch.

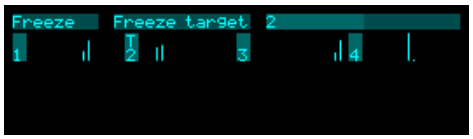
There are a number (depending on the specification) of different ‘voices’ i.e. you can freeze different moments of audio and play them simultaneously.

The algorithm is based on the disting EX algorithm of the same name. There is an in-depth video for that algorithm here[120].

Note that there is also a Temporal Freeze algorithm – see below.

GUI

The display shows information about each of the voices.



The vertical bar on the left shows the envelope level. The small vertical lines visualise the segment of the audio that is being used. The ‘T’ above to the voice number indicates the ‘Freeze target’; if there is no ‘T’, the target is set to ‘Auto’.

Movement

The simplest spectral freeze would take just one FFT ‘frame’ and use that to continuously synthesize audio. This generates a completely static sound, which, while sometimes a useful effect, doesn’t give

119 For the technically minded, it uses the phase vocoder technique.

120 https://www.youtube.com/watch?v=8HFm0zzLHRM

you the sort of textural sustain that is often wanted. For this reason, this algorithm offers means of moving the synthesis source around in a small history window previous to the moment of freezing.

The ‘Movement’, ‘Rate’, and ‘Depth’ parameters allow you to apply an amount of motion automatically.

Depth sets the amount of time over which motion can occur. It is specified in terms of FFT frames, though an equivalent time in milliseconds is also shown. If this is set to 1, motion is effectively disabled.

Rate sets the speed of motion. If set to 0, motion is stopped. If set to 1000, the audio is played at its natural speed. In between those two extremes, playback is slowed down by a factor which is shown in the display.

Movement sets the direction of motion. ‘Forwards’ plays the audio forwards, in a loop. Note though that since this is a resynthesis algorithm, not a looper, the jump from the end of the loop to the start is spectrally interpolated, not simply crossfaded. ‘Backwards’ is similar, except that it plays the audio in reverse. ‘Alternating’ plays the audio alternately forwards and backwards. ‘Random walk’ moves randomly forwards and backwards – the playback position is continuous, but randomly changes direction. ‘Random skip’ randomly jumps around within the audio.

Finally, the ‘Offset’ parameter gives you direct control over the playback position within the buffer. Note that the motion set by Movement, Rate, and Depth is added to the Offset position – if you only want to have Offset control the playback, set Depth to 1 and/or Rate to 0.

Pitch shifting

Each voice can have a pitch shift specified. It should be noted that pitch shifting is not the primary focus of this algorithm, and the results are not the last word in quality pitch shifting. Nevertheless, it’s provided as a potentially useful option.

There are coarse and fine shift controls which apply globally to all of the voices, and individual controls for each voice. The global and per-voice shifts are simply added.

Etherization

The ‘Etherization’ parameter progressively drops out components of the sound which are judged to be ‘transient’. The result tends towards a small number of pure tones which represent the strongest harmonics of the material captured.

Freeze parameters

Name	Min	Max	Default	Unit	Description
Freeze toggle	0	1	0		Activates a new freeze when the parameter changes from ‘0’ to ‘1’.

Freeze gate	0	1	0		Activates a new freeze when the parameter changes from ‘0’ to ‘1’, and releases the freeze when the parameter changes from ‘1’ to ‘0’.
Freeze target	0		2		Chooses the voice for the next freeze, or ‘0’ for ‘Auto’.
Unfreeze all	0	1	0		When this parameter changes to ‘1’, all active freezes are released.

Movement parameters

Name	Min	Max	Default	Unit	Description
Movement	0	4	2		Chooses the type of movement. The options are ‘Forwards’, ‘Backwards’, ‘Alternating’, ‘Random walk’, and ‘Random skip’.
Rate	0	1000	340		Sets the rate of movement. ‘0’ is stopped; ‘1000’ is ‘normal speed’ (as recorded). In between, the scale is highly non-linear to give more control at slower speeds.
Depth	1		2		Sets the maximum amount of movement.
Offset	0		0		Offsets the freeze position back in time.

Changes parameters

Name	Min	Max	Default	Unit	Description
Etherization	0.0	100.0	0.0	%	Drops out transient elements of the sound.
Live process	0	1	0		When this parameter changes to ‘1’, a new voice is started which uses live audio input instead of actually freezing. This is provided mainly as a means to use the etherization effect on live audio.
Attack	0	127	10		Sets the envelope attack time.
Release	0	127	10		Sets the envelope release time.
Coarse shift	-48	24	0	ST	Sets the coarse pitch shift amount for all voices. Added to the individual voice shifts.
Fine shift	-100	100	0	cents	Sets the fine pitch shift amount for all voices. Added to the individual voice shifts.

Per-voice parameters

Name	Min	Max	Default	Unit	Description
Freeze	0	1	0		Activates a new freeze on the corresponding voice when the parameter changes from ‘0’ to ‘1’, and releases the freeze when the parameter changes from ‘1’ to ‘0’.
Coarse shift	-48	24	0	ST	Sets the coarse pitch shift amount.
Fine shift	-100	100	0	cents	Sets the fine pitch shift amount.
Output	1	64	13		Sets the output bus for the voice.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Routing parameters

Name	Min	Max	Default	Unit	Description
Input	1	64	1		Sets the input audio bus.

Spectral Vocoder

“Spectral (FFT) based vocoder”

File format guid: 'spvc'

Specifications:

- Max block size, 6-11: The maximum processing block size.
- Stereo: Whether the algorithm is mono or stereo.

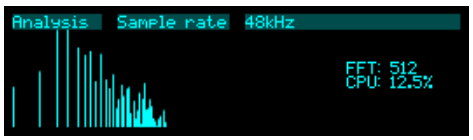
Description

This algorithm implements a vocoder. The spectral characteristics of the modulator input are applied to the carrier input. In classic usage, the modulator might be a human voice, and the carrier might be a synth sound, or simply noise.

This version uses a digital, FFT-based, method. There is also the Vocoder (below) which uses a more traditional bandpass filter approach.

GUI

The display shows the frequency spectrum of the output signal. It also shows the current FFT size, and the module’s total CPU usage (NB not the algorithm’s CPU usage, the total usage).



Analysis parameters

Name	Min	Max	Default	Unit	Description
Latency	0	5	2		Sets the processing latency (which indirectly sets the FFT size).
Sample rate	0	3	0		Sets the sample rate reduction (which indirectly sets the FFT size, and affects the bandwidth).

Env Tracker parameters

Name	Min	Max	Default	Unit	Description
Enable	0	1	0		Enables per-band envelope trackers; effectively slew rate limiters for the band amplitudes.

Attack	0	1000	0		The attack time; exponential scaling from 1ms to 1s.
Release	0	1000	0		The release time; exponential scaling from 10ms to 30s.

Mix parameters

Name	Min	Max	Default	Unit	Description
Dry gain	-40	24	-40	dB	The output level of the dry signal.
Wet gain	-40	24	0	dB	The output level of the wet (vocoder) signal.

Routing parameters

Name	Min	Max	Default	Unit	Description
Modulator input	1	64	3		The bus to use for the modulator signal.
Carrier left/mono input	1	64	1		The left or mono input audio bus for the carrier signal.
Carrier right input	1	64	2		The right input audio bus for the carrier (if the algorithm is set to stereo in the specifications).
Left/mono output	1	64	13		The left or mono output audio bus.
Right output	1	64	14		The right output audio bus (if the algorithm is set to stereo in the specifications).
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Spectrum Analyzer

“Displays the frequency spectrum of a signal”

File format guid: 'spal'

Specifications:

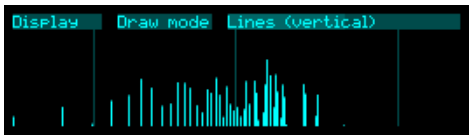
- FFT size, 8-12: The size of the analysis FFT.

Description

This algorithm implements a simple but useful spectrum analyser.

GUI

The display shows the frequency spectrum of the input signal, according to the display mode.



Inputs parameters

Name	Min	Max	Default	Unit	Description
Left/mono input	1	64	1		The left or mono input audio bus.
Right input	0	64	0		The right input audio bus (simply summed with the left bus, if provided).

Display parameters

Name	Min	Max	Default	Unit	Description
Draw mode	0	2	0		Sets the draw mode – Lines (continuous), Lines (vertical), or Points.
X axis	0	1	1		Sets whether the X axis is linear or logarithmic.
Min dB	-140	0	-140	dB	Sets the lower bound of the Y axis.
Max dB	-140	0	-12	dB	Sets the upper bound of the Y axis.

SSB Modulator

“Single Side Band modulator”

File format guid: 'ssbm'

Specifications: None

Description

This algorithm is based on one mode of the Frequency Shifter algorithm on the disting EX. It modulates two signals with each other, and produces the upper and lower sidebands on separate outputs.

For a (rather technical) description of single side band modulation, see the Wikipedia article here[121].

If you use a pure sine wave for one of the inputs, you essentially have a frequency shifter, but one where you control the shift by controlling the frequency of your input oscillator.

Mix parameters

Name	Min	Max	Default	Unit	Description
Mix	0	100	100	%	The output wet/dry mix.
Level	-40	0	0	dB	The output gain applied to the modulated signal.

Routing parameters

Name	Min	Max	Default	Unit	Description
Input 1	1	64	1		The first input bus to process for the first signal.
Input 2	1	64	2		The first input bus to process for the second signal.
Width	1	8	1		The number of busses to process, starting from the Input. For example, for a stereo signal, set this to 2.
Upper SB output	0	64	13		The first output bus for the upper side band.
Upper SB mode	0	1	0		The standard Add/Replace mode selector as described above for the upper side band.
Lower SB output	0	64	0		The first output bus for the lower side band.

121 https://en.wikipedia.org/wiki/Single-sideband_modulation

Lower SB mode	0	1	0		The standard Add/Replace mode selector as described above for the lower side band.

Step Sequencer

“A simple note sequencer”

File format guid: 'spsq'

Specifications: None

Description

This algorithm is a 16-step note sequencer, based on the sequencers in the Expert Sleepers FH-2[122] module. It can output CVs and/or MIDI.

Each step has pitch, velocity, and a general-purpose “modulation” value. It also has a “Division”, which can either be a repeat count (the step is repeated a number of times on subsequent clocks) or a ratchet count (the step fires a number of gates within the duration of one clock). Each substep (either repeat or ratchet) can be on or off – this is set by the “Pattern” parameter. Furthermore each step or substep can be a tie – this is set by the “Ties” parameter.

A tie means a note which continues until just after the beginning of the next (sub)step. When outputting MIDI, this means the note off will follow the next note on i.e. the notes are played legato. When outputting CV, a tie means that the gate will not go low between the notes, and that the pitch CV will glide (if the glide time is non-zero).

The algorithm has a function for randomising the pitches, rhythm, or both.

Each Step Sequencer has internal storage for 32 sequences, which you can use like ‘snapshots’ or ‘patterns’ to prepare a number of sequences and then switch between them (manually, or under CV control) to build up songs.

Remember that every pitch etc. is a parameter and therefore mappable. Something as simple as an LFO modulating a sequence step’s pitch can produce fascinating evolving patterns. If you need to constrain the sequencer to a scale/key, follow it with a Quantizer.

GUI

The display shows an overview of the 16 steps.

The step pitches are displayed across the top of the screen. The current step being played is indicated by a highlight on the pitch value. The start and end steps are indicated by solid bars between the pitches.

122 https://expert-sleepers.co.uk/fh2.html

The step whose parameter is being edited is shown highlighted. The steps' divisions, patterns, and ties are shown graphically. If the current parameter is a pitch, velocity, or modulation, these values are shown via the thin lines to the left of the steps.

Divisions (repeats or ratchets) are shown as stacks of blocks:

Ties are indicated by longer bars:

Setting pitches via MIDI

You can choose a MIDI channel via the 'MIDI channel (in)' parameter which will be used to program the sequencer pitches by sending in MIDI notes e.g. from your MIDI controller keyboard.

If the current parameter page is one of the 'Step' pages, sending a MIDI note will set the pitch of the current step, and advance to the next step's page.

Selecting sequences via CV

The 'Sequence CV input' parameter can be used to set up a CV input to control the current sequence. This is scaled such that a semitone (1/12th of a Volt) corresponds to one sequence increment, the idea being that this allows you to use a Step Sequencer (or MIDI keyboard etc.) to sequence the sequences.

If a 'Sequence trigger input' is also provided, the sequence CV takes effect when the trigger fires. Otherwise, the sequence changes whenever the CV changes.

The 'Sequence change mode' gives you two options for what is actually mean by 'changing the sequence'. 'Switch' is perhaps the more obvious one – the CV causes the 'Sequence' parameter to change, so the desired sequence becomes the current one. The other option is 'Load', which causes the selected sequence to be copied into Sequence 1. The intention here is that you would prepare a number of sequences in slots 2-32, and then during playback leave the Sequence parameter on 1. You can then 'recall' your various prepared sequences, and perhaps mess about ('perform') with them, without corrupting your pre-prepared work.

Chaining sequences

Using the 'Next sequence' parameter, you can have sequences automatically follow one another, to build up longer structures. If you do this, you may also like to set the 'Reset sequence' parameter, so that you always start on the same sequence when your transport is reset.

Menu functions

Some extra functions are available via the ‘Step Sequencer’ menu.

Copy sequence

This menu allows you to copy a sequence from one slot to another.

Reset sequence

This menu allows you to reset the current sequence to initial conditions.

Sequencer parameters

Name	Min	Max	Default	Unit	Description
Sequence	1	32	1		Selects the current sequence.
Start	1	16	1		The first step to play. A reset will jump to this step.
End	1	16	16		The last step to play.
Direction	0	6	0		The sequencer direction. See below.
Permutation	0	3	0		The sequencer permutation. See below.
Gate type	0	1	0		Sets the gate type: “% of clock” or “Trigger”.
Gate length	1	99	50	%	Sets the length of the gate output if the type is “% of clock”.
Trigger length	1	100	10	ms	Sets the length of the gate output if the type is “Trigger”.
Glide	0	1000	100	ms	Sets the glide time for tied notes.
Next sequence	0	32	0		Selects a sequence that will automatically play after this one, or ‘Off’ to stay on the current sequence.
Reset sequence	0	32	0		Selects the sequence that will play when the sequencer receives a reset signal, or ‘Off’ to stay on the current sequence.

Step parameters

Name	Min	Max	Default	Unit	Description
Pitch	0	127	48		Sets the step pitch, as a MIDI note number.
Division	0	14	7		Sets the step repeat or ratchet count.
Pattern	0	255	0		Sets the step pattern – which substeps are on or off.

Ties	0	255	0		Sets the step tie pattern – which substeps are ties.
Velocity	1	127	64		Sets the step velocity.
Mod	-10.0	10.0	0.0	V	Sets the step modulation value.
Mute	0	100	0	%	Sets the probability that the step will be muted.
Skip	0	100	0	%	Sets the probability that the step will be skipped.
Reset	0	100	0	%	Sets the probability that the step causes a reset.
Repeat	0	100	0	%	Sets the probability that the step will be repeated.

Randomise parameters

Name	**Min**	**Max**	**Default**	**Unit**	**Description**
Randomise	0	1	0		When this parameter changes from 0 to 1, the sequence steps between the current start and end steps will be randomised according to the parameters that follow.
Randomise what	0	3	3		What will be randomised – “Nothing”, “Pitches”, “Rhythm”, or “Both”.
Note distribution	0	1	0		Chooses the probability distribution for the note pitches – “Uniform” or “Normal”.
Min note	0	127	36		The minimum note pitch, if the distribution is “Uniform”.
Max note	0	127	60		The maximum note pitch, if the distribution is “Uniform”.
Mean note	0	127	48		The mean note pitch, if the distribution is “Normal”.
Note deviation	0	127	12		The note pitch deviation, if the distribution is “Normal”.
Min repeat	2	8	2		The minimum repeat count, if a step is a repeat.
Max repeat	2	8	8		The maximum repeat count, if a step is a repeat.
Min ratchet	2	8	2		The minimum ratchet count, if a step is a ratchet.
Max ratchet	2	8	8		The maximum ratchet count, if a step is a ratchet.
Note probability	0	100	50	%	The probability that a note will be on.
Tie probability	0	100	0	%	The probability that a step will be a tie.

Accent probability	0	100	0	%	The probability that a step will be accented.
Repeat probability	0	100	0	%	The probability that a step will be a repeat.
Ratchet probability	0	100	0	%	The probability that a step will be a ratchet.
Unaccented velocity	1	127	1		The velocity value to use for unaccented steps. Accented steps use velocity 127.

Routing parameters

Name	Min	Max	Default	Unit	Description
Clock input	0	64	0		The bus to use as the clock input.
Reset input	0	64	0		The bus to use as the reset input.
Reset mode	0	2	0		Sets the mode for the reset input. The options are: ● the input is a reset trigger. ● the input is a run/stop signal. ● the input is a ‘one shot’ trigger.
Pitch output	0	64	15		The bus to use for the pitch CV output.
Pitch output mode	0	1	1		The standard Add/Replace mode selector as described above, for the pitch CV output.
Gate output	0	64	16		The bus to use for the gate output.
Gate output mode	0	1	1		The standard Add/Replace mode selector as described above, for the gate output.
Velocity output	0	64	0		The bus to use for the velocity output.
Velocity output mode	0	1	1		The standard Add/Replace mode selector as described above, for the velocity output.
Mod output	0	64	0		The bus to use for the modulation CV output.
Mod output mode	0	1	1		The standard Add/Replace mode selector as described above, for the modulation CV output.
Step output	0	64	0		The bus to use for the step CV output. This represents the current sequencer step number scaled as 1/12th of a Volt per step.
Step output mode	0	1	1		The standard Add/Replace mode selector as described above, for the step CV output.

Sequence CV input	0	64	0		The bus to use to select the sequence. See above.
Sequence trigger input	0	64	0		The bus to use for the sequence select trigger. See above.
Sequence change mode	0	1	0		How the sequence CV will be used to change sequences: ‘Switch’ or ‘Load’. See above.

MIDI parameters

Name	**Min**	**Max**	**Default**	**Unit**	**Description**
Follow MIDI clock	0	1	0		Sets whether the sequencer follows MIDI clock.
Divisor	0	19	4		Sets the clock divisor when following MIDI clock.
MIDI channel (out)	1	16	1		Sets the output MIDI channel.
Output to breakout	0	1	0		Enables MIDI output to the breakout.
Output to Select Bus	0	1	0		Enables MIDI output to the Select Bus.
Output to USB	0	1	0		Enables MIDI output to USB.
Output to internal	0	1	0		Enables internal MIDI output – that is, MIDI is sent to the other algorithms.
Mod CC	0	127	1		The MIDI CC to generate from the modulation lane.
MIDI channel (in)	0	16	0		Sets the input MIDI channel for assigning step pitches from MIDI notes.
Record velocity	0	1	0		If on, record velocity from input MIDI notes as well as the note number.

Sequencer directions

Forwards	The steps play in ascending order e.g. 1, 2, 3, 4, 1, 2, 3, 4 …
Reverse	The steps play in descending order e.g. 4, 3, 2, 1, 4, 3, 2, 1 …

Alt	The steps play alternately ascending and descending e.g. 1, 2, 3, 4, 3, 2, 1, 2 …
Alt2	The steps play alternately ascending and descending, repeating the first and last steps e.g. 1, 2, 3, 4, 4, 3, 2, 1, 1, 2 …
Random	The steps play in a random order.
Random2	Steps are played in a random order, except that the same step cannot be played twice in a row.
Random3	All the steps are played once in a random order, then again in another random order, and so on.

Sequencer permutations

Note that if the direction is set to one of the random options, the permutation has no effect.

Straight	The steps are played in their numerical order e.g. 1, 2, 3, 4, 5, 6, 7, 8 …
Odd/Even	Odd numbered steps play, followed by the even numbered steps e.g. 1, 3, 5, 7, 2, 4, 6, 8, 1 …
Halves	The second half of the steps are interleaved with the first half e.g. 1, 5, 2, 6, 3, 7, 4, 8, 1 …
Inwards	The first and last steps play, then the second and penultimate, and so on e.g. 1, 8, 2, 7, 3, 6, 4, 5, 1 …

Step Sequencer Head

“An extra playback head”

File format guid: 'spsh'

Specifications: None

Description

This algorithm shares the sequence of another Step Sequencer algorithm (above), and replicates the parameters that define how that sequence is played. This allows you to, for example, have the same sequence played in different directions simultaneously, or at different speeds, or with different transpositions.

The Step Sequencer Head refers to the nearest Step Sequencer above it in the algorithm list. You can have as many Step Sequencer Heads as you like for each Step Sequencer.

Sequencer parameters

Name	Min	Max	Default	Unit	Description
Sequence	1	32	1		Selects the current sequence.
Start	1	16	1		The first step to play. A reset will jump to this step.
End	1	16	16		The last step to play.
Direction	0	6	0		The sequencer direction. See above.
Permutation	0	3	0		The sequencer permutation. See above.
Gate type	0	1	0		Sets the gate type: “% of clock” or “Trigger”.
Gate length	1	99	50	%	Sets the length of the gate output if the type is “% of clock”.
Trigger length	1	100	10	ms	Sets the length of the gate output if the type is “Trigger”.
Glide	0	1000	100	ms	Sets the glide time for tied notes.
Octave	-10	10	0		Transposes the sequence in octaves.
Transpose	-60	60	0	ST	Transposes the sequence in semitones.

Routing parameters

Name	Min	Max	Default	Unit	Description
Clock input	0	64	0		The bus to use as the clock input.
Reset input	0	64	0		The bus to use as the reset input.
Reset mode	0	1	0		Sets the mode for the reset input. The options are: • the input is a reset trigger. • the input is a run/stop signal.
Pitch output	0	64	15		The bus to use for the pitch CV output.
Pitch output mode	0	1	1		The standard Add/Replace mode selector as described above, for the pitch CV output.
Gate output	0	64	16		The bus to use for the gate output.
Gate output mode	0	1	1		The standard Add/Replace mode selector as described above, for the gate output.
Velocity output	0	64	0		The bus to use for the velocity output.
Velocity output mode	0	1	1		The standard Add/Replace mode selector as described above, for the velocity output.
Mod output	0	64	0		The bus to use for the modulation CV output.
Mod output mode	0	1	1		The standard Add/Replace mode selector as described above, for the modulation CV output.
Step output	0	64	0		The bus to use for the step CV output. This represents the current sequencer step number scaled as 1/12th of a Volt per step.
Step output mode	0	1	1		The standard Add/Replace mode selector as described above, for the step CV output.
Sequence CV input	0	64	0		The bus to use to select the sequence. See above.
Sequence trigger input	0	64	0		The bus to use for the sequence select trigger. See above.

MIDI parameters

Name	Min	Max	Default	Unit	Description
Follow MIDI clock	0	1	0		Sets whether the sequencer follows MIDI clock.

Divisor	0	19	4		Sets the clock divisor when following MIDI clock.
MIDI channel (out)	1	16	1		Sets the output MIDI channel.
Output to breakout	0	1	0		Enables MIDI output to the breakout.
Output to Select Bus	0	1	0		Enables MIDI output to the Select Bus.
Output to USB	0	1	0		Enables MIDI output to USB.
Output to internal	0	1	0		Enables internal MIDI output - that is, MIDI is sent to the other algorithms.
Mod CC	0	127	1		The MIDI CC to generate from the modulation lane.

Stopwatch

“Real-time clock for timing things”

File format guid: 'stpw'

Specifications: None

Description

This algorithm is simply a clock, for timing how long things (for example, your performance) have been going on, or for showing a countdown (for example, until you should stop your performance and get off the stage).

It has no effect on any input/output bus and consumes almost no CPU.

GUI

The display shows the stopwatch time.

While the stopwatch is visible and running the screensaver is prevented from activating.

Setup parameters

Name	Min	Max	Default	Unit	Description
Mode	0	1	0		Chooses whether to display a timer or a countdown.
Start/stop mode	0	1	0		Sets whether the start/stop control is a gate (start when high, stop when low) or a trigger (trigger to start, another trigger to stop).

Countdown parameters

Name	Min	Max	Default	Unit	Description
Hours	0	24	0		Sets the number of hours for the countdown.
Minutes	0	59	0		Sets the number of minutes for the countdown.
Seconds	0	59	0		Sets the number of seconds for the countdown.

Controls parameters

Name	Min	Max	Default	Unit	Description
Start/stop	0	1	0		Starts and stops the timer/countdown.
Reset	0	1	0		Sets the timer to zero, or resets the countdown to that specified by the countdown parameters.

Temporal Freeze

“An audio ‘freeze’ effect”

File format guid: 'tpfz'

Specifications:

- Voices, 1-16: The number of simultaneous voices.
- History, 0.1-2.0s: The length of the history stored for each voice.

Description

This algorithm is an audio ‘freeze’ effect – it captures sound at a moment in time and sustains it indefinitely. The ‘temporal’ part of the name indicates that it works in the time domain, in contrast to the Spectral Freeze algorithm (above).

There are a number (depending on the specification) of different ‘voices’ i.e. you can freeze different moments of audio and play them simultaneously.

You can also layer up new audio on top of existing freezes.

You can smoothly apply a detuning to the frozen audio, offering effects from a simple octave down, to complex microtonal chords.

There is a playlist of videos demonstrating this algorithm on YouTube here[123].

Usage

There are two workflows supported by the algorithm:

- You control each voice/freeze manually, or
- You simply trigger the algorithm to perform a freeze and the algorithm manages the voices.

In the first case, you can manipulate the Freeze parameter of each voice directly, or you can use the ‘Freeze target’ parameter along with the ‘Freeze toggle’ or ‘Freeze gate’.

In the second case, set the ‘Freeze target’ to Auto, and use ‘Freeze toggle’ or ‘Freeze gate’.

It is expected that you will probably want to map ‘Freeze toggle’ or ‘Freeze gate’ to a CV, or a MIDI or I2C controller, but you can of course simply manipulate it manually from the UI.

Activating the ‘Layer’ parameter on a voice adds a layer – new audio is captured and faded into the voice’s freeze buffer. The old audio is attenuated according to the ‘Layer fade’ parameter. The time over which the fade between old and new audio occurs is controlled by the ‘Layer time’ parameter.

123 https://www.youtube.com/playlist?list=PLlY5j4QDwxWv5QW7Toc1gjvbTeQiYZUwQ

‘Pedal’ control

Each voice has a parameter named ‘Pedal’, which is intended to let the voice operate as a single footswitch operated ‘freeze’ device, in the manner of certain popular guitar pedals.

With a voice in an inactive, unfrozen state, the first pedal press activates a freeze. Subsequent presses activate a new layer. To unfreeze a voice, you either hold down the pedal for a longer time, or double-tap it, depending on the ‘Pedal style’ parameter.

Detuning

You can play the frozen audio at a different pitch, or even as chords. This is done with the ‘Detuned heads’/‘Detuned’ parameters, and the parameters in the Detuning, Gains, and Microtuning pages.

The freeze playback consists of a number of ‘play heads’ (to use a tape metaphor). This number is set by the ‘Play heads’ parameter and defaults to 4.

Each of these play heads can be assigned a detuned pitch, and then you can choose how many of the heads use their normal pitch or the detuned pitch. The ‘Detuned heads’ parameter lets you set the number of detuned heads directly, one at a time; the ‘Detuned’ parameter allows you to smoothly increase the number of detuned heads, crossfading each transition.

For example if the number of heads is 4, and the Detuned parameter is 25%, only the first head will use its detuned pitch; if Detuned is 50%, two heads will use the detuned pitch. Between 25% and 50% the second head will crossfade between the two pitches.

By default, the detunings are specified as semitones. If you enable microtuning and specify a Scala file to use, the detunings are specified as scale degrees (so e.g. if your Scala file is for a 7 tone scale, a detuning of -7 means down one octave).

Gains are provided for each detuned head. The motivation for this is that heads detuned up in pitch have a tendency to stick out a bit, and need to be turned down, but you can of course use the gains however you like.

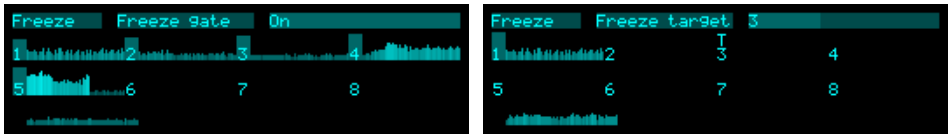
Stereo operation

If the ‘Right input’ parameter is not None, the algorithm captures audio in stereo. This halves the maximum history length for each voice. It also raises the CPU usage.

The voice output bus assignments can be mono or stereo – if the input is mono and the output is stereo, the same mono signal is used for both outputs.

GUI

The display shows information about each of the voices.



The vertical bar on the left shows the envelope level. The ‘T’ above to the voice number indicates the ‘Freeze target’; if there is no ‘T’, the target is set to ‘Auto’.

During a layer transition, the vertical bar splits into two, to show the progress of the layer fade as well as the envelope level.

The vertical bar changes to a bright colour when the pedal parameter has been held down long enough to trigger the release of the voice (if ‘Pedal style’ is ‘Long hold’).

Each voice also shows a very condensed waveform view showing the section of audio that was captured. The similar waveform at the bottom left of the screen shows the incoming audio.

Freeze parameters

Name	Min	Max	Default	Unit	Description
Freeze toggle	0	1	0		Activates a new freeze when the parameter changes from ‘0’ to ‘1’.
Freeze gate	0	1	0		Activates a new freeze when the parameter changes from ‘0’ to ‘1’, and releases the freeze when the parameter changes from ‘1’ to ‘0’.
Freeze target	0		0		Chooses the voice for the next freeze, or ‘0’ for ‘Auto’.
Unfreeze all	0	1	0		When this parameter changes to ‘1’, all active freezes are released.
Auto voices	2				When the freeze target is Auto, sets how many voices will be used by the automatic voice selection. The remainder can be frozen manually.

Tweaks parameters

Name	Min	Max	Default	Unit	Description
Length	10		200	ms	Sets the amount of audio just preceding the moment of the freeze to use.
Play heads	1	7	4		Sets the number of ‘playback heads’.

Detuned heads	0	7	0		Sets how many of the playback heads use their detuning setting (as a simple count).
Detuned	0.0	100.0	0.0	%	Sets how many of the playback heads use their detuning setting (as a percentage, with smooth blending between values).
Gain	-40	6	0	dB	Applies a gain to the algorithm outputs.

Detuning parameters

Name	Min	Max	Default	Unit	Description
Detuning 1-7	-24	24	-12	ST	The detuning amount for play heads 1-7 respectively.

Gains parameters

Name	Min	Max	Default	Unit	Description
Detuned 1-7	-40	12	0	dB	The gain to apply to play heads 1-7 respectively when detuned.

Envelope parameters

Name	Min	Max	Default	Unit	Description
Attack	0	1000	100		Sets the envelope attack time. Exponential scale from 0.01s to 30s.
Release	0	1000	100		Sets the envelope release time. Exponential scale from 0.01s to 30s.
Mode	0	1	0		Sets the envelope mode: ‘Normal’ or ‘Complete attack’. In the latter mode, the envelope attack segment always completes, even if the gate is released during the attack, which is useful to trigger long swells from a short trigger pulse.

Layer parameters

Name	Min	Max	Default	Unit	Description
Layer fade	-40.0	0.0	0.0	dB	The amount by which old material fades out when a new layer is added.
Layer time	0	1000	100		The crossfade time when adding a new layer. Exponential scale from 0.1s to 10s.

Pedal style	0	1	0		The style of pedal gesture that will release a freeze - ‘Long hold’ or ‘Double tap’.
Pedal hold time	0.1	5.0	0.3	s	The time required to hold a pedal to release a freeze, if the pedal style is ‘Long hold’.
Double tap time	0.1	5.0	0.3	s	The time during which a pedal must be double-tapped to release a freeze, if the pedal style is ‘Double tap’.

Per-voice parameters

Name	Min	Max	Default	Unit	Description
Freeze	0	1	0		Activates a new freeze on the corresponding voice when the parameter changes from ‘0’ to ‘1’, and releases the freeze when the parameter changes from ‘1’ to ‘0’.
Layer	0	1	0		Triggers the addition of a layer when the parameter changes from ‘0’ to ‘1’.
Pedal	0	1	0		Controls the ‘pedal’ interface to the voice.
Left/mono output	1	64	13		Sets the left output bus for the voice.
Right output	0	64	14		Sets the right output bus for the voice.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Microtuning parameters

Name	Min	Max	Default	Unit	Description
Microtuning	0	1	0		Enables the use of microtuning for the detuned play heads.
Scala .scl	0		0		Selects the Scala .scl file to use for microtuning.

Routing parameters

Name	Min	Max	Default	Unit	Description
Left/mono input	1	64	1		Sets the left or mono input audio bus.

Right input	0	64	2		Sets the right input audio bus.

Temporal MIDI Quantizer

“Quantizes MIDI notes in time”

File format guid: 'tmqn'

Specifications:

- Channels, 1-16: The number of MIDI channels to process.

Description

This algorithm is a MIDI quantizer, but not a pitch quantizer – it quantizes the position of notes in time (much like applying quantization to the timing of a MIDI clip in your DAW). Timing can be derived from MIDI clock or from clock pulses on the algorithm’s input bus.

It only affects MIDI notes – other MIDI messages are not processed.

Because there is no concept of ‘replacing’ or ‘consuming’ MIDI events in the disting NT, you will usually want to use different input and output MIDI channels e.g. receive MIDI notes from your keyboard on channel 1 and output quantized notes on channel 2, and then have your synth algorithm(s) respond on channel 2.

Common parameters

Name	Min	Max	Default	Unit	Description
Sync	0	2	0		The timing source to sync to: None, MIDI, or Clock.
Clock input	1	64	1		The input bus to use for clock if Sync is set to Clock.
Output to breakout	0	1	0		Enables MIDI output to the breakout.
Output to Select Bus	0	1	0		Enables MIDI output to the Select Bus.
Output to USB	0	1	0		Enables MIDI output to USB.
Output to internal	0	1	1		Enables internal MIDI output – that is, MIDI is sent to the other algorithms.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Enable	0	1	0		Enables the channel.

Input channel	1	16	1		The MIDI channel on which to receive notes.
Output channel	1	16	2		The MIDI channel on which to send notes.
Divisor	0	19	9		The MIDI clock divisor, if Sync is set to MIDI.
Prune short notes	0	1	0		If enabled, notes that are already over by the time the clock pulse comes around are suppressed.

Texture Synthesizer

“It's Clouds!”

File format guid: 'clou'

Specifications: None

Description

This algorithm is an implementation of the open source Clouds[124] module by Émilie Gillet.

This implementation should behave and sound identical to the original. Any demos or tutorials you may find online should apply just as well to this version. The user manual for the original module is here[125]. The documentation below assumes some familiarity with this.

Clouds parameters

Name	Min	Max	Default	Unit	Description
Sample rate	0	1	0		Allows you to run the algorithm at the disting NT’s own sample rate (‘Native’) or the sample rate of a real Clouds module (‘Authentic (32kHz)’).
Mode	0	3	0		The effect mode: Granular, Stretch, Looping delay, or Spectral.
Buffer channels	1	2	2		Whether the internal audio buffer is mono or stereo.
Input channels	1	2	2		Whether the audio input is mono or stereo.
Bit depth	0	1	0		Enables “lo-fi” mode (8 bit and reduced sample rate).
Freeze	0	1	0		“This latching [parameter] stops the recording of incoming audio. Granularization is now performed on the last few seconds of audio kept in memory in the module.” (This and other quotes are from the Clouds manual linked above.)
Position	0	100	50	%	“Selects from which part of the recording buffer the audio grains are played. Turn the knob clockwise to travel back in time.”
Size	0	100	50	%	Sets the grain size.

124 https://github.com/pichenettes/eurorack/tree/master/clouds

125 https://pichenettes.github.io/mutable-instruments-documentation/modules/clouds/manual/

Pitch	-100	100	0	%	Sets the grain transposition.
Coarse tune	-48	48	0	ST	Sets the grain pitch in semitones.
Fine tune	-100	100	0	cents	Sets the grain pitch in cents.
Density	-100	100	50	%	“At 12 o'clock, no grains are generated. Turn clockwise and grains will be sown randomly, counter-clockwise and they will be played at a constant rate. The further you turn, the higher the overlap between grains.”
Texture	0	100	50	%	“Morphs through various shapes of grain envelopes: square (boxcar), triangle, and then Hann window. Past 2 o'clock, activates a diffuser which smears transients.”
Wet/dry	0	100	50	%	Wet/dry mix balance.
Spread	0	100	50	%	“Stereo spread (amount of random panning/balance applied to the grains).”
Feedback	0	100	50	%	Sets the amount of feedback.
Reverb	0	100	50	%	Sets the amount of reverb applied to the output.

Routing parameters

Name	**Min**	**Max**	**Default**	**Unit**	**Description**
Left input	1	64	1		The left audio input bus.
Right input	1	64	2		The right audio input bus.
Left output	1	64	13		The left audio output bus.
Right output	1	64	14		The right audio output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.
V/Oct input	0	64	0		The bus to use for a V/octave pitch CV.
Trigger input	0	64	0		The bus to use for a trigger input.

Three Pot

“Runs DSP code from the SD card”

File format guid: 'spin'

Specifications:

- Code memory, 1-24: The amount of code memory to assign to the algorithm (in multiples of 512 bytes).

Description

This algorithm allows you to run what are effectively DSP “plug-ins”, loaded from the MicroSD card.

It was designed particularly to support programs written for the Spin Semiconductor[126] FV-1 chip. This popular IC has ended up in numerous products including some Eurorack modules (e.g. the FX Aid[127] and the Timiszoara[128]). Its popularity and the relative ease of developing for it has led to a large number of (generally open source) DSP programs being available, and it is this that the Three Pot algorithm was designed to leverage.

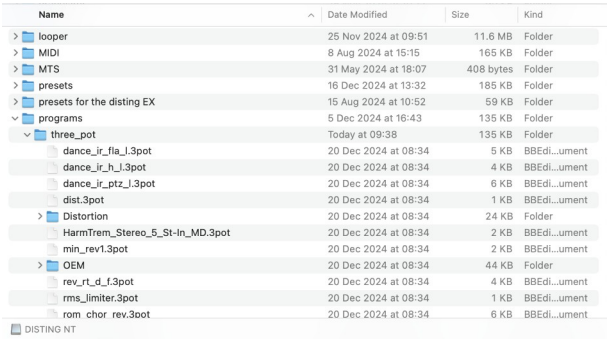
The name “Three Pot” comes from the FV-1 hardware design – all it has are stereo inputs and outputs, and connections for three pots for user controls.

Although the focus is on converted FV-1 programs, the conversion works by translating the programs to C++ and then compiling them for the disting NT – so you can if you like actually write C++ directly and use this algorithm to run it on the module.

Installing programs

Programs are files in JSON format, with the extension ‘.3pot’. They should be placed on the MicroSD card in the folder `/programs/three_pot`

One level of subdirectories is allowed. For example, your card might look like this:



126 http://www.spinsemi.com

127 https://happynerding.com/category/fx-aid/

128 http://xaocdevices.com/main/timiszoara/

Where to get programs

Some example programs are included in the GitHub repository[129]. You can just download the 3pot files and copy them to your MicroSD card.

The community on our Discord server (see above) were very active during the development of this algorithm, and amassed a large library of programs converted to 3pot format. Please visit the server for the latest information on this.

Otherwise, a quick internet search will turn up plenty of (unconverted) programs e.g. here[130].

You can of course also write your own, either in raw SpinAsm code, or using a GUI such as SpinCAD Designer[131].

How to convert programs

A makefile-based system for converting programs is in the GitHub repository mentioned above. Documentation for this system can be found there.

Pre/post gain

The FV-1 works in a normalised numeric format, where all signals are between ±1.0. The disting NT on the other hand works internally in voltages, usually in the range ±10.0V. For linear processes (e.g. reverb, delay) this doesn't matter, but for anything where the actual signal level is important (e.g. compressors, distortions) it's necessary to adjust the gain to maintain a sensible amount of headroom.

The 'Pre/post gain' parameter does this. The signal is attenuated by the gain specified before the program is run, and amplified by the inverse gain afterwards.

GUI

The UI for this algorithm is currently unique on the disting NT in that it not only provides a custom display, but it also changes the way the knobs work. Having three physical pots on the module, matched up to the three controls on each program, was just too good an opportunity to ignore.

The display shows the program name, pot labels, and description (all from the 3pot file if provided).

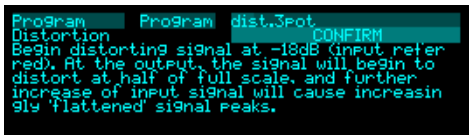
The three pots on the module control the three pot parameters directly. Additionally, the right encoder changes the program.

129 https://github.com/expertsleepersltd/distingNT/tree/main/tools/three_pot

130 https://mstratman.github.io/fv1-programs/

131 https://github.com/HolyCityAudio/SpinCAD-Designer

While selecting a new program, the whole display is given over to showing the program description, which can be quite long:



Program parameters

Name	Min	Max	Default	Unit	Description
Program	0	999	0		Selects the program to run.
Pot 1	0.0	100.0	50.0	%	The first program ‘pot’ parameter.
Pot 2	0.0	100.0	50.0	%	The second program ‘pot’ parameter.
Pot 3	0.0	100.0	50.0	%	The third program ‘pot’ parameter.

Mix parameters

Name	Min	Max	Default	Unit	Description
Pre/post gain	-36	0	-20	dB	Sets the gain adjustment applied before the program’s DSP code. See above.
Mix style	0	2	0		Chooses the dry/wet mix style: ‘None’, ‘Wet/dry’, or ‘Independent’.
Mix	0	100	100	%	Sets the mix if the style is ‘Wet/dry’.
Dry gain	-40	24	-40	dB	Sets the dry gain if the mix style is ‘Independent’.
Wet gain	-40	24	0	dB	Sets the wet gain if the mix style is ‘Independent’.

Sample rate parameters

Name	Min	Max	Default	Unit	Description
Rate conversion	0	1	0		Enables sample rate conversion.
Sample rate	8.0	40.0	32.0		The sample rate (in kHz) at which to run the program.

Routing parameters

Name	Min	Max	Default	Unit	Description

Left input	1	64	1		The left audio input bus.
Right input	1	64	2		The right audio input bus.
Left output	1	64	13		The left audio output bus.
Right output	1	64	14		The right audio output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Tracker

“A pitch and envelope tracker”

File format guid: 'trak'

Specifications: None

Description

This algorithm is based on the disting EX algorithm of the same name. You may like to review the video on that algorithm, here[132]. The pitch shifter part of the EX algorithm was split out into a separate algorithm on the NT (above), which you can combine with this algorithm if it suits your needs.

The algorithm tracks the pitch and envelope of an audio signal, which are output as CVs. It can also generate chords to harmonise with the tracked pitch – either as pitch CVs (to drive VCOs etc.) or as pitch shifter CVs (to drive the Pitch Shifter, to make harmony copies of the tracked audio).

The chords can be tuned microtonally using Scala or MTS.

GUI

The display shows the tracked note at the bottom of the screen, as a MIDI note number plus cents, and as a frequency in Hz. To the right of this is a bar that shows the envelope, and the tuning as a deviation from the true pitch of the tracked note.

Above this is the closest note to the tracked note that lies within the current scale (on the left), and the four harmony notes (on the right).

Tracker parameters

Name	Min	Max	Default	Unit	Description
Range	1	20	9		Sets the range of pitches which can be tracked. Set this appropriately for the lowest note that you need to track.
Track bias	0	100	10	%	An internal parameter of the pitch tracking algorithm, which can help it avoid errors of octave in some circumstances. The default value is recommended in most cases.

132 https://www.youtube.com/watch?v=wSjuOA9yMBk

Env threshold	-80	0	-40	dB	Sets a signal threshold below which no attempt is made to track pitch.

Harmony parameters

The Harmony parameters are mostly the standard polysynth chord parameters, as described above. There are three additional parameters:

Name	Min	Max	Default	Unit	Description
Harmony mode	0	2	0		Sets the harmony mode – one of ‘Shape’, ‘SATB’, or ‘MIDI’. The first two are the standard modes described in the polysynth documentation. The last uses MIDI notes provided to the algorithm to directly set the chord notes.
MIDI channel	0	16	0		Sets the MIDI channel on which to receive notes if the harmony mode is ‘MIDI’.
Force in tune	0	1	0		Sets whether the chord pitches should be absolutely in tune (allowing expressive pitch bending in a solo part over a straight backing), or in tune with the tracked audio (allowing the generated chords to bend with the tracked audio).

Microtuning parameters

The algorithm uses the standard microtuning parameters, as described above.

Routing parameters

Name	Min	Max	Default	Unit	Description
Input	1	64	1		The audio input bus.
Pitch output	0	64	15		The pitch CV output bus.
Pitch output mode	0	1	0		The standard Add/Replace mode selector as described above, for the pitch CV output.
Env output	0	64	16		The envelope CV output bus.
Env output mode	0	1	0		The standard Add/Replace mode selector as described above, for the envelope CV output.
Change trigger out	0	64	0		The change trigger output bus – emits a trigger pulse when the current tracked note changes.

Change trigger mode	0	1	0		The standard Add/Replace mode selector as described above, for the change trigger output.

Routing (chord) parameters

Name	Min	Max	Default	Unit	Description
Chord 1 output	0	64	0		The pitch CV output bus for the first chord note.
Chord 1 output mode	0	1	0		The standard Add/Replace mode selector as described above, for the first chord note CV output.
Chord 2 output	0	64	0		The pitch CV output bus for the second chord note.
Chord 2 output mode	0	1	0		The standard Add/Replace mode selector as described above, for the second chord note CV output.
Chord 3 output	0	64	0		The pitch CV output bus for the third chord note.
Chord 3 output mode	0	1	0		The standard Add/Replace mode selector as described above, for the third chord note CV output.
Chord 4 output	0	64	0		The pitch CV output bus for the fourth chord note.
Chord 4 output mode	0	1	0		The standard Add/Replace mode selector as described above, for the fourth chord note CV output.

Routing (shift) parameters

Name	Min	Max	Default	Unit	Description
Shift 1 output	0	64	0		The pitch shift CV output bus for the first pitch shifter.
Shift 1 output mode	0	1	0		The standard Add/Replace mode selector as described above, for the first pitch shifter CV output.
Shift 2 output	0	64	0		The pitch shift CV output bus for the second pitch shifter.

Shift 2 output mode	0	1	0		The standard Add/Replace mode selector as described above, for the second pitch shifter CV output.
Shift 3 output	0	64	0		The pitch shift CV output bus for the third pitch shifter.
Shift 3 output mode	0	1	0		The standard Add/Replace mode selector as described above, for the third pitch shifter CV output.

Transient Detector

“Detects transients”

File format guid: 'trdt'

Specifications:

- Channels, 1-12: The number of detector channels.

Description

This algorithm generates triggers from transients in audio signals. It is essentially the same code that detects transients in the Kirbinator and in the Sample Player (Sliced) algorithms, broken out into a separate algorithm.

While designed for use on audio, we have found it also gives good results when applied to the signals generated by piezo-electric pickups, for example, the simple electronic pickups that can be clipped to drums (e.g. this[133]).

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Enable	0	1	0		Enables the channel.
Input	1	64	1		The input bus to process.
Trigger output	0	64	15		The output bus for the trigger signal.
Trigger output mode	0	1	1		The standard Add/Replace mode selector as described above, for the trigger.
Level output	0	64	16		The output bus for the level (velocity) signal.
Level output mode	0	1	1		The standard Add/Replace mode selector as described above, for the level.
Gain	-40	24	0	dB	The gain applied to the input. Has no effect on the sensitivity, but does affect the level output.
Sensitivity	0	200	100	%	The detector sensitivity.

133 https://www.roland.com/us/products/rt-30h/

Tuner (fancy)

“A sophisticated tuner”

File format guid: 'tunf'

Specifications: None

Description

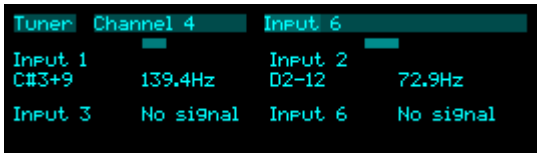
This algorithm provides four simultaneous tuners.

It uses an autocorrelation pitch detection method which is fairly CPU heavy, but which provides reliable pitch detection for most signals. For simpler tones, you may prefer the “Tuner (simple)” algorithm.

Microtuning via Scala or MTS is supported.

GUI

The display shows the detected pitch of the active channels, as a MIDI note number and cents, and as a value in Hz. It also shows a bar which grows to the left or right depending on how flat or sharp the note is relative to the ‘true’ pitch of the detected note.



Parameters

Name	Min	Max	Default	Unit	Description
Channel 1	0	64	1		The input bus for tuner number 1.
Channel 2	0	64	0		The input bus for tuner number 2.
Channel 3	0	64	0		The input bus for tuner number 3.
Channel 4	0	64	0		The input bus for tuner number 4.
Range	1	20	9		Sets the range of pitches which can be tracked. Set this appropriately for the lowest note that you need to tune.
Track bias	0	100	10	%	An internal parameter of the pitch tracking algorithm, which can help it avoid errors of octave in some circumstances. The default value is recommended in most cases.

Env threshold	-80	0	-40	dB	Sets a signal threshold below which no attempt is made to track pitch.

Microtuning parameters

The algorithm uses the standard microtuning parameters, as described above.

Tuner (simple)

“A basic tuner for simple tones”

File format guid: 'tuns'

Specifications: None

Description

This algorithm provides four simultaneous tuners.

It uses a simple pitch detection method which will only work reliably for simple tones, but happily “simple tones” covers most of those that you will get from an analogue VCO, which in a Eurorack environment you might find yourself tuning quite often.

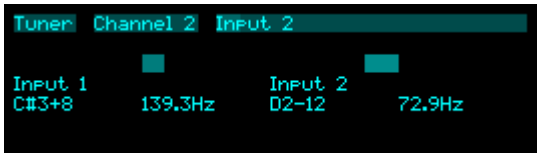
Being simple, it consumes little CPU.

Microtuning via Scala or MTS is supported.

For a more sophisticated tuner, please see the “Tuner (fancy)” algorithm.

GUI

The display shows the detected pitch of the active channels, as a MIDI note number and cents, and as a value in Hz. It also shows a bar which grows to the left or right depending on how flat or sharp the note is relative to the ‘true’ pitch of the detected note.



Parameters

Name	Min	Max	Default	Unit	Description
Channel 1	0	64	1		The input bus for tuner number 1.
Channel 2	0	64	0		The input bus for tuner number 2.
Channel 3	0	64	0		The input bus for tuner number 3.
Channel 4	0	64	0		The input bus for tuner number 4.

Microtuning parameters

The algorithm uses the standard microtuning parameters, as described above.

USB audio (from host)

“Receives audio from USB host”

File format guid: 'usbf'

Specifications: None

Description

This algorithm exposes USB audio from the host, if one is connected.

The host sees 8 output channels. Each one of these can be output to any of the disting NT’s busses, or to the ES-5 output channels (which you would use in conjunction with the Silent Way plug-ins that drive the ES-5 and attached expanders).

Note that the signals from the USB host don’t just have to emerge directly from the module’s outputs; they could just as well be the inputs to other algorithms, allowing you to process audio from the host through the disting.

Note also that “USB audio” doesn’t mean that the signals have to be audio; they could also be CVs, if you’d like to exchange signals with, say, VCV Rack or similar.

Parameters

Name	Min	Max	Default	Unit	Description
to	0	66			The bus on which to output USB channel 1.
mode	0	1	1		The standard Add/Replace mode selector as described above, for channel 1.
Etc. up to channel 8					

USB audio (to host)

“Sends audio to USB host”

File format guid: 'usbt'

Specifications: None

Description

This algorithm routes signals to the USB audio host, if one is connected.

The host sees 12 input channels. Each one of these can be set to receive any of the disting NT’s busses.

Note that the signals seen by the algorithm, and so by the host, depend where in the list of algorithms this one appears. If it’s the first algorithm, it can send to the host the module’s own inputs, unmodified; if it’s the last algorithm, it can send to the host the outputs of all the other algorithms.

Note also that “USB audio” doesn’t mean that the signals have to be audio; they could also be CVs, if you’d like to exchange signals with, say, VCV Rack or similar.

Common parameters

Name	Min	Max	Default	Unit	Description
USB channel 1 from	1	64	1		The bus to send to USB channel 1.
USB channel 2 from	1	64	2		The same for USB channel 2.
Etc. up to channel 12					

VCA/Multiplier

“Four quadrant multiplier”

File format guid: 'vcam'

Specifications:

- Channels, 1-8: The number of bus channels to process.

Description

This algorithm is a voltage multiplier, which can be used as a VCA. The CV on the common channel is used to multiply the voltages on the other channels.

The algorithm supports zero-cross detection, in the manner of our Persephone[134] analogue VCA module.

Common parameters

Name	Min	Max	Default	Unit	Description
Input	1	64	1		The common input bus.
Clamp to 0V	0	1	1		Whether to clamp the common input to 0V (typical VCA usage) or not (four quadrant multiplier usage).
Divider	1	12	8	V	Sets the scaling of the input CV that corresponds to “unity gain”.

Per-channel parameters

Name	Min	Max	Default	Unit	Description
Input	1	64	1		The channel input bus.
Output	0	64	0		The output bus. If set to ‘None’, the input bus is used as output, and the mode is always ‘Replace’.
Output mode	0	1	1		The standard Add/Replace mode selector as described above.
Zero cross	0	1	0		Activates zero-cross detection. Changes in the VCA CV are only applied when the channel signal crosses 0V.

134 https://expert-sleepers.co.uk/persephone.html

VCF (State Variable)

“Second order LP/BP/HP filter”

File format guid: 'fsvf'

Specifications: None

Description

This algorithm is a voltage controlled filter using the common ‘State Variable’ topology, which yields simultaneous low-, band-, and highpass filter responses.

All three filter outputs are available individually, plus an output which can be blended from lowpass, through bandpass, to highpass.

The input and the various outputs use a contiguous range of busses, starting with the one set via the parameters, and with the channel count set by the ‘Width’ parameter. For example, to filter stereo signals, set the Width to 2.

GUI

The display simply shows the filter centre frequency, in Hz and as a MIDI note number.

Filter parameters

Name	Min	Max	Default	Unit	Description
Blend	0	200	0		Sets the blend for the blended output.
Sweep	-36.00	84.00	0.00	ST	A manual frequency sweep control; offsets the frequency CV.
Resonance	0	100	20		Sets the filter resonance.
Saturate	0	1	1		Enables an internal saturation stage, which can keep high resonances from running out of control.

Gain parameters

Name	Min	Max	Default	Unit	Description

Blended gain	-40	6	0	dB	Level control for the blended output.
Lowpass gain	-40	6	0	dB	Level control for the lowpass output.
Bandpass gain	-40	6	0	dB	Level control for the bandpass output.
Highpass gain	-40	6	0	dB	Level control for the highpass output.

Routing parameters

Name	Min	Max	Default	Unit	Description
Audio input	1	64	1		The audio input bus.
Width	1	8	1		The number of busses to process.
Frequency input	0	64	0		The frequency CV input bus (1V/octave).
Resonance input	0	64	0		The resonance CV input bus. Scaled such that 5V corresponds to the full range.
Blended output	0	64	13		The output bus for the blended signal.
Blended mode	0	1	0		Add/replace mode for the blended signal.
Lowpass output	0	64	0		The output bus for the lowpass signal.
Lowpass mode	0	1	0		Add/replace mode for the lowpass signal.
Bandpass output	0	64	0		The output bus for the bandpass signal.
Bandpass mode	0	1	0		Add/replace mode for the bandpass signal.
Highpass output	0	64	0		The output bus for the highpass signal.
Highpass mode	0	1	0		Add/replace mode for the highpass signal.

MIDI parameters

Name	Min	Max	Default	Unit	Description
MIDI channel	0	16	0		Sets the MIDI channel on which to respond to note on events.
Keyboard track	0	100	100	%	Sets the amount by which the pitch of received MIDI notes affects the filter cutoff.

VCF (24dB/oct LP)

“Fourth order LP filter”

File format guid: 'fmoo'

Specifications: None

Description

This algorithm is a 24dB/octave lowpass filter, implemented using circuit emulation of a four pole transistor ladder filter[135]. It is a port of the same algorithm on the disting EX, for which there is a video here[136].

GUI

The display simply shows the filter centre frequency, in Hz and as a MIDI note number.

Filter parameters

Name	Min	Max	Default	Unit	Description
Sweep	-36.00	84.00	0.00	ST	A manual frequency sweep control; offsets the frequency CV.
Resonance	0	100	20		Sets the filter resonance. The filter will self-oscillate at high resonances.
Drive	-12.0	48.0	0.0	dB	A gain control for the input to the filter. Boosting the input gives a subjectively pleasant non-linear distortion.
Output gain	-40	12	0	dB	A simple output gain control.
Normalize	0	1	1		If set, boosts the gain to compensate for the natural tendency of the circuit to lose low-frequency level as the resonance is increased.

135 Developed using hard sums by Eddie Edwards at Reanimator Ltd – see Acknowledgments, below.

136 https://www.youtube.com/watch?v=YfG_Bg_n7fs

Routing parameters

Name	Min	Max	Default	Unit	Description
Audio input	1	64	1		The audio input bus.
Width	1	8	1		The number of busses to process.
Frequency input	0	64	0		The frequency CV input bus (1V/octave).
Resonance input	0	64	0		The resonance CV input bus. Scaled such that 5V corresponds to the full range.
Output	1	64	13		The output bus.
Output mode	0	1	1		The standard Add/Replace mode selector as described above.

MIDI parameters

Name	Min	Max	Default	Unit	Description
MIDI channel	0	16	0		Sets the MIDI channel on which to respond to note on events.
Keyboard track	0	100	100	%	Sets the amount by which the pitch of received MIDI notes affects the filter cutoff.

VCO - Pulsar

“A pulsar VCO”

File format guid: 'vcop'

Specifications: None

Description

This algorithm based on the Pulsar VCO algorithm on the disting mk4. You might like to watch the video on that algorithm here[137]. It is an implementation of pulsar synthesis, inspired by the description in Curtis Roads's book Microsound (MIT Press) pp137-157.

In pulsar synthesis, a small grain of sound (known as a *pulsaret*, typically a pulse or single cycle waveform) is repeated periodically, followed by a small section of silence. The length of the sound within the repeating period (equivalently, the speed at which it is played) offers a means of control the timbre without affecting the perceived pitch of the sound. By the same token, changing the fundamental pitch without changing the speed at which the grain of sound is played offers pitch change without simply shifting the entire spectrum up and down, an effect reminiscent of 'vocal' or 'formant' synthesis.

There are therefore two pitch inputs:

- the fundamental frequency – the rate at which the pulsaret train is generated.
- the formant frequency – the rate at which the pulsaret is played.

If the ‘Pitch mode’ parameter is ‘Independent’, these two pitches are completely independent. If the mode is ‘Tracking’, the fundamental pitch is added to the formant pitch, so so changing the fundamental frequency also changes the formant frequency.

This algorithm uses wavetables from the SD card as the source of the pulsarets. Please see the section above on how wavetables are formatted and arranged on the MicroSD card.

Masking

Masking removes pulsarets from the train. There are two modes:

- Stochastic – pulsarets are randomly masked. The likelihood of a pulsaret being masked is set by the Masking parameter.
- Burst – masking works in groups of pulsarets, the length of which is set by the Burst length parameter. The Masking parameter sets how many pulsarets within each group will be masked.

There are two outputs from the algorithm – the inverse output will emit a pulsaret when the main

137 https://www.youtube.com/watch?v=Ekz4Eq8fJ9Q

output does not, and *vice versa*.

GUI

The display shows a single cycle of the current waveform, and the VCO's pitch as a MIDI note name plus cents and as a frequency in Hz. The wavetable waveform will be stretched or expanded according to the relative pitch of the VCO and the formant.

VCO parameters

Name	Min	Max	Default	Unit	Description
Wavetable	0				Chooses the wavetable.
Wave offset	-100.0	100.0	0.0	%	Sets the position within the wavetable.
Window	0	3	0		Sets the window applied to the pulsarets: one of Rectangular, Linear attack, Linear decay, or Gaussian.
Pitch mode	0	1	0		Sets whether the formant frequency is independent of the fundamental frequency or linked.
Masking mode	0	1	0		Sets the masking mode: Stochastic or Burst.
Masking	0	100	0	%	Sets the amount of masking.
Burst length	2	100	2		Sets the burst length, if the masking mode is Burst.
Octave	-16	8	0		Adjusts the VCO tuning in octaves.
Transpose	-60	60	0	ST	Adjusts the VCO tuning in semitones.
Fine tune	-100	100	0	cents	Adjusts the VCO tuning in cents.

Gain parameters

Name	Min	Max	Default	Unit	Description

Amplitude	0.00	10.00	10.00	V	Sets the amplitude of the output (before it’s affected by the gain).
Gain	-40	6	0	dB	Sets the level of the output.

MIDI parameters

Name	Min	Max	Default	Unit	Description
MIDI channel	0	16	0		If enabled, MIDI notes on the specified channel will set the pitch of the VCO.

Routing parameters

Name	Min	Max	Default	Unit	Description
Pitch input	0	64	1		The pitch CV input (1V/octave).
Formant input	0	64	0		The formant CV input (1V/octave).
Wave input	0	64	0		Sets an input bus to modulate the wave offset.
Output	0	64	13		The output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above for the main output.
Inverse output	0	64	0		The output bus with inverse masking.
Inverse output mode	0	1	0		The standard Add/Replace mode selector as described above for the inverse output.

VCO - Waveshaping

“Simple VCO with adjustable outputs”

File format guid: 'vcow'

Specifications: None

Description

This algorithm is based on the OG disting algorithms “B-8 VCO with waveshaping” and “B-7 VCO with linear FM”.

There are four oscillator outputs:

- Triangle/saw
- Square/pulse
- Sub-octave square
- Sine

A shared waveshape parameter/CV controls both the shape of the triangle/saw and sine waves and the pulse width of the square/pulse wave.

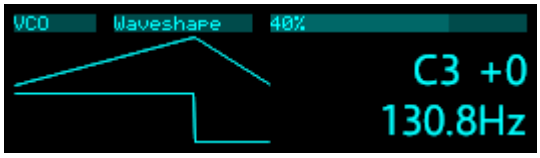
Linear through-zero FM is available via the ‘FM input’ parameter.

If the ‘Sync’ input is used, a rising edge on the input sets the VCO phase to zero, which can produce 'oscillator sync' sounds[138].

Note that it is perfectly valid to set the output bus for the four signals to the same bus, in which case they can be blended to taste with their gain parameters.

GUI

The display shows a single cycle of the triangle/saw and square/pulse waveshapes, and the VCO’s pitch as a MIDI note name plus cents and as a frequency in Hz.



VCO parameters

Name	Min	Max	Default	Unit	Description
Waveshape	-100	100	0	%	Sets the wave shape/pulsewidth.

138 https://en.wikipedia.org/wiki/Oscillator_sync

Octave	-16	8	0		Adjusts the VCO tuning in octaves.
Transpose	-60	60	0	ST	Adjusts the VCO tuning in semitones.
Fine tune	-100	100	0	cents	Adjusts the VCO tuning in cents.
Oversampling	0	2	0		Enables oversampling, to reduce aliasing noise at higher frequencies. The options are “None”, “2x”, and “4x”.
FM scale	1	1000	100	Hz	Sets the Hz/V sensitivity of the linear FM input.

Gain parameters

Name	Min	Max	Default	Unit	Description
Triangle/saw amplitude	0.00	10.00	10.00	V	Sets the amplitude of the triangle/saw output (before it’s affected by the gain).
Square/pulse amplitude	0.00	10.00	10.00	V	Sets the amplitude of the square/pulse output (before it’s affected by the gain).
Sub amplitude	0.00	10.00	10.00	V	Sets the amplitude of the sub-octave output (before it’s affected by the gain).
Sine amplitude	0.00	10.00	10.00	V	Sets the amplitude of the sine output (before it’s affected by the gain).
Triangle/saw gain	-40	6	0	dB	Sets the level of the triangle/saw output.
Square/pulse gain	-40	6	0	dB	Sets the level of the square/pulse output.
Sub gain	-40	6	0	dB	Sets the level of the sub-octave output.
Sine gain	-40	6	0	dB	Sets the level of the sine output.

MIDI parameters

Name	Min	Max	Default	Unit	Description
MIDI channel	0	16	0		If enabled, MIDI notes on the specified channel will set the pitch of the VCO.

Routing parameters

Name	Min	Max	Default	Unit	Description
Pitch input	0	64	1		The pitch CV input (1V/octave).
Shape input	0	64	0		The waveshape CV input (scaled so that ±5V corresponds to the ±100% parameter range).
Sync input	0	64	0		The oscillator sync input.
FM input	0	64	0		The linear FM input.
Triangle/ saw output	0	64	13		The output bus for the triangle/saw signal.
Triangle/ saw mode	0	1	0		The add/replace mode for the triangle/saw signal.
Square/ pulse output	0	64	0		The output bus for the square/pulse signal.
Square/ pulse mode	0	1	0		The add/replace mode for the square/pulse signal.
Sub output	0	64	0		The output bus for the sub-octave signal.
Sub output mode	0	1	0		The add/replace mode for the sub-octave signal.
Sine output	0	64	0		The output bus for the sine signal.
Sine output mode	0	1	0		The add/replace mode for the sine signal.

VCO - Wavetable

“A wavetable VCO”

File format guid: 'vcot'

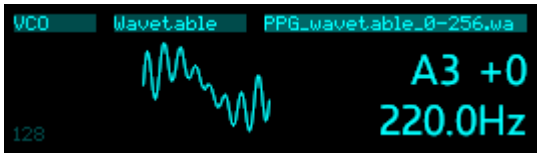
Specifications: None

Description

This algorithm is a simple VCO which uses wavetables for its wave shapes. Please see the section above on how wavetables are formatted and arranged on the MicroSD card.

GUI

The display shows a single cycle of the current wavetable waveform, and the VCO’s pitch as a MIDI note name plus cents and as a frequency in Hz.



VCO parameters

Name	Min	Max	Default	Unit	Description
Wavetable	0				Chooses the wavetable.
Wave offset	-100.0	100.0	0.0	%	Sets the position within the wavetable.
Octave	-16	8	0		Adjusts the VCO tuning in octaves.
Transpose	-60	60	0	ST	Adjusts the VCO tuning in semitones.
Fine tune	-100	100	0	cents	Adjusts the VCO tuning in cents.

Gain parameters

Name	Min	Max	Default	Unit	Description
Amplitude	0.00	10.00	10.00	V	Sets the amplitude of the output (before it’s affected by the gain).
Gain	-40	6	0	dB	Sets the level of the output.

MIDI parameters

Name	Min	Max	Default	Unit	Description
MIDI channel	0	16	0		If enabled, MIDI notes on the specified channel will set the pitch of the VCO.

Routing parameters

Name	Min	Max	Default	Unit	Description
Pitch input	0	64	1		The pitch CV input (1V/octave).
Wave input	0	64	0		Sets an input bus to modulate the wave offset.
Sync input	0	64	0		The oscillator sync input. A rising edge on this input resets the VCO phase to zero.
Output	0	64	13		The output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Vocoder

“A classic vocoder”

File format guid: 'voco'

Specifications: None

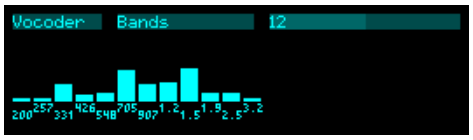
Description

This algorithm implements a vocoder. The spectral characteristics of the modulator input are applied to the carrier input. In classic usage, the modulator might be a human voice, and the carrier might be a synth sound, or simply noise.

This version emulates an analogue vocoder inasmuch as it uses a number of bandpass filters to do the analysis and synthesis. There is also the Spectral Vocoder (above) which uses an FFT method.

GUI

The display shows the current level of the modulator bands.



Vocoder parameters

Name	Min	Max	Default	Unit	Description
Bands	4	20	12		The number of bands into which the signals are split.
Min frequency	50	1000	200	Hz	The centre frequency of the lowest band.
Max frequency	1000	10000	3200	Hz	The centre frequency of the highest band.
Bandwidth	25	400	100	%	Applies a scaling to the width of the bandpass filters.
Attack	0	1000	303		Sets the attack time of the band envelope trackers. Exponential scaling from 0.5ms to 1000ms.
Decay	0	1000	394		Sets the decay time of the band envelope trackers. Exponential scaling from 0.5ms to 1000ms.
High pass	10	1000	50	Hz	Sets the frequency of a high-pass filter applied to both the modulator and carrier.

Gain	-40	24	0	dB	Applies an overall gain.

Noise parameters

Name	Min	Max	Default	Unit	Description
Noise	-40	24	-40	dB	Sets the level of noise added to the carrier input, which can improve speech intelligibility. The noise is high-pass filtered at the frequency of the highest vocoder band.

Routing parameters

Name	Min	Max	Default	Unit	Description
Modulator input	1	64	1		The bus to use for the modulator signal.
Carrier input	1	64	2		The first carrier bus to process.
Carrier width	1	8	1		The number of carrier busses to process. For example, for a stereo signal, set this to 2.
Output	1	64	13		The first output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Vowel Filter

“Simulates vowel sounds”

File format guid: 'vowl'

Specifications: None

Description

This algorithm is based on the Dual Vowel Filter algorithm on the disting mk4. You may like to watch the video on that algorithm here[139]. From the disting mk4 manual:

“A set of three bandpass filters (BPFs) is used to mimic the response of the human vocal tract, resulting in vowel-like sounds when provided suitable source material.”

The available vowels are as follows:

Vowel	Example word
/ow/	bought
/oo/	boot
/a/	hot[140]
/uh/	but
/er/	bird
/ae/	bat
/e/	bet
/i/	bit
/iy/	beet

Vowel parameters

Name	Min	Max	Default	Unit	Description
Mode	0	1	0		One of ‘Switched’ or ‘Blended’.
Vowel	0	8	0		The chosen vowel, if the mode is Switched.
Vowel	0	800	0		The chosen vowel, if the mode is Blended.
BPF gain 2	-20	0	0		Adjusts the gain of the second BPF.
BPF gain 3	-20	0	-6		Adjusts the gain of the third BPF.

139 https://www.youtube.com/watch?v=-ogn0JonlPY

140 American pronunciation!

Routing parameters

Name	Min	Max	Default	Unit	Description
Audio input	1	64	1		The first input bus to process.
Width	1	8	1		The number of busses to process, starting from the Input. For example, for a stereo signal, set this to 2.
Vowel input	0	64	0		The bus to use to control the vowel. Scaled such that 0-5V covers the full range of vowels.
Output	1	64	13		The first output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Waveform Animator

“Embiggens simple waveforms”

File format guid: 'wfan'

Specifications: None

Description

This algorithm is version of the disting mk4 algorithm of the same name. You may like to watch the video on that algorithm here[141]. From the disting mk4 manual:

“This algorithm recreates a popular analogue circuit variously known as a waveform animator or wave multiplier. Such a circuit, when given (typically) a sawtooth waveform as input, uses a comparator to generate a square wave of suitable phase and pulsewidth such that when the square and saw waves are added together the result is a phase shifted saw waveform. This is usually done a number of times and the results added with the overall effect of turning the original, rather plain, waveform into a much richer one which to all intents and purposes is the same as if you'd started with a number of VCOs rather than just one, and so sounds "fatter". In combination with LFOs varying the comparator thresholds, very rich textures can be generated. Here, four comparators and four LFOs are used.”

The comparator thresholds are set by a combination of the Threshold parameter and the Threshold CV input, and the Separation parameter which defines a 'spread' of the individual thresholds around the centre.

The amplitude of the square waves can be determined automatically (by an envelope tracker) or manually, according to the ‘Track’ parameter. The recreation of phase shifted sawtooths works best when the square wave amplitude is matched to that of the saw. However, fixing the amplitude of the square waves and varying the input signal's amplitude is a useful effect in its own right.

Animate parameters

Name	Min	Max	Default	Unit	Description
Threshold	-10.0	10.0	0.0	V	Sets the comparator threshold.
Track	0	1	1		Whether to track the input signal amplitude.
Amplitude	0.0	10.0	5.0	V	The square wave amplitude, if tracking is not being used.
Squares	1	4	4		The number of square waves to generate.

141 https://www.youtube.com/watch?v=htkdsLm0oa8

Separation	0	100	50	%	The spread of the actual comparator thresholds around the value set by ‘Threshold’.
Mix	-100	100	0	%	The output mix, from -100% (only the input signal) to 100% (only the squares). At 0% both the input and squares are at full amplitude.
LFO rate	0	1000	699		The LFO rate, scaled exponentially from 0.2Hz to 20Hz.
LFO depth	0	100	10	%	The LFO depth.

Routing parameters

Name	**Min**	**Max**	**Default**	**Unit**	**Description**
Input	1	64	1		The first input bus to process.
Width	1	8	1		The number of busses to process, starting from the Input. For example, for a stereo signal, set this to 2.
Output	1	64	13		The first output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.
Threshold input	0	64	0		The CV input bus for the comparator threshold.

Waveshaper

“Waveshaper/wavefolder”

File format guid: 'wavs'

Specifications: None

Description

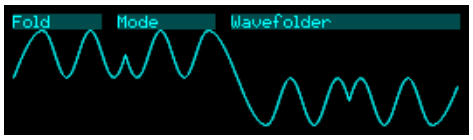
This algorithm is based on the Dual Waveshaper algorithm on the disting mk4. You may like to watch the video on that algorithm here[142]. From the disting mk4 manual:

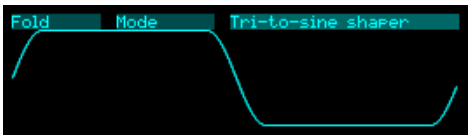
“Type 0 is what is usually termed a wavefolder. This increases the harmonic content of the sound in interesting ways, especially as the gain changes.

“Type 1 is a triangle-to-sine waveshaper. Used on most audio this is a relatively gentle form of overdrive/saturation. However, when fed with the right level of triangle wave, the output is exactly a sine wave, which is useful when you have a triangle wave VCO handy but really want a pure sine wave instead.”

GUI

The display shows how an input sine wave will be affected by the waveshaper. It is not an oscilloscope – it’s not showing what your actual audio looks like (unless you happen to be feeding it a sine wave). It’s just a graphical aid to get an idea of how much waveshaping is being applied.





Fold parameters

Name	Min	Max	Default	Unit	Description
Input gain	-32.00	32.00	1.00		The input gain. Negative values invert the signal.
Mode	0	1	0		The waveshaping mode: ‘Wavefolder’ or ‘Tri-to-sine shaper’.
Mix	0	100	100	%	The output wet/dry mix.
Level	-40	0	0	dB	The output gain applied to the shaped signal.

142 https://www.youtube.com/watch?v=-B1VzxQ2qOo

Routing parameters

Name	Min	Max	Default	Unit	Description
Gain input	0	64	0		The bus to use to control the input gain. Scaled such that a 5V CV gives 32x gain.
Input	1	64	1		The first input bus to process.
Width	1	8	1		The number of busses to process, starting from the Input. For example, for a stereo signal, set this to 2.
Output	1	64	13		The first output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

Wavetable Waveshaper

“Waveshaper that uses a wavetable”

File format guid: 'wtws'

Specifications: None

Description

This algorithm is based on the Wavetable Waveshaper algorithm on the disting mk4. You may like to watch the video on that algorithm here[143]. From the disting mk4 manual:

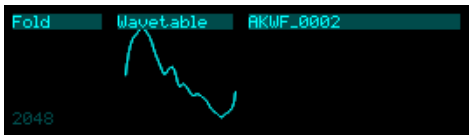
“Applied to audio, this algorithm is a wavetable-based waveshaper effect. More generally, considering the input as a CV lookup, this algorithm provides a wavetable-based transfer function.”

As such it offers a great way to roll your own phase distortion synthesis.

Please see the section above on how wavetables are formatted and arranged on the MicroSD card.

GUI

The display shows the current wavetable.



Fold parameters

Name	Min	Max	Default	Unit	Description
Wavetable	0				Chooses the wavetable.
Wave offset	-100.0	100.0	0.0	%	Sets the position within the wavetable.
Input gain	-32.00	32.00	1.00		The input gain. Negative values invert the signal.
Mix	0	100	100	%	The output wet/dry mix.
Level	-40	0	0	dB	The output gain applied to the shaped signal.

Routing parameters

Name	Min	Max	Default	Unit	Description

143 https://www.youtube.com/watch?v=8FHf27D2byA

Gain input	0	64	0		The bus to use to control the input gain. Scaled such that a 5V CV gives 32x gain.
Wave input	0	64	0		Sets an input bus to modulate the wave offset.
Input	1	64	1		The first input bus to process.
Width	1	8	1		The number of busses to process, starting from the Input. For example, for a stereo signal, set this to 2.
Output	1	64	13		The first output bus.
Output mode	0	1	0		The standard Add/Replace mode selector as described above.

UI Scripts

The disting NT offers the ability to completely redefine the module's UI using the popular scripting language Lua[144].

It is anticipated that this will be particularly useful in a live performance scenario, where you might want to reduce the available controls to a few key items, and not want to see (or risk changing) the bulk of the preset.

Scripts can be loaded from anywhere on the MicroSD card, but the examples provided are in the 'ui_scripts' folder.

> This feature can probably best be considered a "technology preview" - it works, but there's so much more that could be added. We're very interested in your feedback on whether this is a useful feature to develop further.

Running a script

From the 'UI Scripts' menu, choose 'Run UI script', which will let you browse the MicroSD card for your chosen script.

The script completely takes over the module UI until it is exited. The combination of pressing all four pushbuttons at once is reserved as a means of exiting a UI script. You are also free to define your own choice of means of leaving the scripted UI via the script itself.

If the script fails for any reason the module will do its best to return any error information generated by the Lua interpreter.

Writing scripts

Further documentation of UI scripts and of Lua scripting on the disting NT in general can be found in the separate Lua scripting document, which can be found on the firmware download page (here[145]), right next to the download link for this manual.

144 https://www.lua.org

145 https://expert-sleepers.co.uk/distingNTfirmwareupdates.html

MIDI SysEx reference

The disting NT supports a variety of functions via MIDI System Exclusive (SysEx) messages.

The browser-based tools here[146] can be considered as informal documentation of these messages, as in most cases the messages only exist to support these tools.

SysEx Header

All SysEx messages are prefixed with a manufacturer's ID, which is a unique series of hex bytes assigned by the MIDI Manufacturers Association. The Expert Sleepers ID is 00H 21H 27H, so all SysEx messages relating to Expert Sleepers hardware will begin

F0 00 21 27

Messages for the disting NT follow this with 6DH:

F0 00 21 27 6D

followed by the module's SysEx ID (see the Settings, above)

F0 00 21 27 6D <SysEx ID>

and then with a byte to identify the specific type of message e.g.

F0 00 21 27 6D <SysEx ID> 01

16 bit values

Where '16 bit value' is indicated below, this is a sequence of 3 bytes:

<most significant 2 bits> <middle 7 bits> <least significant 7 bits>

Received SysEx messages

01H – Take screenshot

F0 00 21 27 6D <SysEx ID> 01 F7

This causes the disting NT to respond with a SysEx message containing a screenshot of what is currently on the module's display, using the '33H – Screenshot' format, below.

04H – Set real-time clock

F0 00 21 27 6D <SysEx ID> 04 <time MSB> <time> <time> <time> <time LSB> F7

This sets the module’s real-time clock to the time represented as a 32 bit number of seconds since the start of the epoch (January 1st, 1970).

146 https://github.com/expertsleepersltd/distingNT/tree/main/tools

07H – Wake from screensaver

F0 00 21 27 6D <SysEx ID> 07 F7

Resets the timer that controls the screensaver, and if the screensaver is active, dismisses it.

08H – Execute Lua

F0 00 21 27 6D <SysEx ID> 08 <text> F7

Executes the enclosed text as a Lua chunk. If there is any output, it is returned using the ‘09H – Lua output’ message.

09H – Install Lua

F0 00 21 27 6D <SysEx ID> 08 <algorithm index> <text> F7

Executes the enclosed text as a Lua chunk and installs the result as the program of the indexed Lua Script algorithm. If there is any output, it is returned using the ‘09H – Lua output’ message.

This message is intended to facilitate rapid iteration of Lua scripts during development. The final script still needs to be installed to the MicroSD card as usual.

11H – .scl file

F0 00 21 27 6D <SysEx ID> 11 <ignored byte> <file contents> F7

Sends a Scala .scl file for the module to use.

12H – .kbm file

F0 00 21 27 6D <SysEx ID> 11 <ignored byte> <file contents> F7

Sends a Scala .kbm file for the module to use.

20H – Set display mode

F0 00 21 27 6D <SysEx ID> 20 <mode> F7

Sets the module display mode.

22H – Request version string

F0 00 21 27 6D <SysEx ID> 22 F7

This causes the disting NT to respond with a SysEx message containing the module's version string as text, using the '32H – Message' format, below.

30H – Request number of algorithms

F0 00 21 27 6D <SysEx ID> 30 F7

Queries the number of algorithms – that is, the number of different algorithms that the module can

run, not the number of algorithms instantiated in the current preset. The response uses the '30H – Number of algorithms' format, below.

31H – Request algorithm info

F0 00 21 27 6D <SysEx ID> 31 <16 bit index> F7

Queries for details of the indexed algorithm (in the list returned by the 30H message). The response uses the '31H – Algorithm info' format, below.

32H – Add algorithm

F0 00 21 27 6D <SysEx ID> 32 <guid> <16 bit specification value> <16 bit specification value> <16 bit specification value> F7

Adds an algorithm to the current preset. If the requested algorithm is a plug-in, load the plug-in first.

33H – Remove algorithm

F0 00 21 27 6D <SysEx ID> 33 <algorithm index> F7

Removes an algorithm from the current preset.

34H – Load preset

F0 00 21 27 6D <SysEx ID> 34 <append> <NULL terminated ASCII string> F7

Loads or appends the preset specified by the filename string.

35H – New preset

F0 00 21 27 6D <SysEx ID> 35 F7

Starts a new preset (exactly as the "New Preset" menu).

36H – Save preset

F0 00 21 27 6D <SysEx ID> 36 <option> F7

Saves the preset. The option value can be 0 (prompt for file overwrite), 1 (never overwrite), or 2 (always overwrite).

37H – Move algorithm

F0 00 21 27 6D <SysEx ID> 37 <algorithm index> <new index> F7

Requests that an algorithm be moved to a new position in the list.

38H – Load plug-in

F0 00 21 27 6D <SysEx ID> 38 <guid> F7

Loads the plug-in that defines an algorithm with the given guid.

40H – Request algorithm guid

F0 00 21 27 6D <SysEx ID> 40 <algorithm index> F7

Queries the indexed algorithm in the current preset. The response uses the '40H – Algorithm guid' format, below.

41H – Request preset name

F0 00 21 27 6D <SysEx ID> 41 F7

Requests the current preset name. Responds with '41H – Preset name', as below.

42H – Request number of parameters

F0 00 21 27 6D <SysEx ID> 42 <algorithm index> F7

Requests the number of parameters in the indexed algorithm. Responds with '42H – Number of parameters', as below.

43H – Request parameter info

F0 00 21 27 6D <SysEx ID> 43 <algorithm index> <16 bit parameter number> F7

Requests information for the given parameter in the indexed algorithm. Responds with '43H – Parameter info', as below.

44H – Request all parameter values

F0 00 21 27 6D <SysEx ID> 44 <algorithm index> F7

Requests the current values of all parameters in the indexed algorithm. Responds with '44H – All parameter values', as below.

45H – Request parameter value

F0 00 21 27 6D <SysEx ID> 45 <algorithm index> <16 bit parameter number> F7

Requests the value of the given parameter in the indexed algorithm. Responds with '45H – Parameter value', as below.

46H – Set parameter value

F0 00 21 27 6D <SysEx ID> 46 <algorithm index> <16 bit parameter number> <16 bit value> F7

Sets the value of the given parameter in the indexed algorithm.

47H – Set preset name

F0 00 21 27 6D <SysEx ID> 47 <NULL terminated ASCII string> F7

Sets the preset name.

48H – Request unit strings

F0 00 21 27 6D <SysEx ID> 48 F7

Requests string descriptions of the possible parameter units (Hz, ms etc.). Responds with '48H – Unit strings', as below.

49H – Request enum strings

F0 00 21 27 6D <SysEx ID> 49 <algorithm index> <16 bit parameter number> F7

Requests the value strings for an 'enum' type parameter. Responds with '49H – Enum strings', as below.

4AH – Set focus

F0 00 21 27 6D <SysEx ID> 4A <algorithm index> <16 bit parameter number> F7

Sets the given parameter to be the currently active one in the module's display.

4BH – Request mappings

F0 00 21 27 6D <SysEx ID> 4B <algorithm index> <16 bit parameter number> F7

Requests the mappings for a parameter. Responds with ‘4BH – Mapping’, as below.

4DH – Set mapping

F0 00 21 27 6D <SysEx ID> 4D <algorithm index> <16 bit parameter number> <packed mapping data> F7

Sets the CV mapping for the given parameter.

4EH – Set MIDI mapping

F0 00 21 27 6D <SysEx ID> 4E <algorithm index> <16 bit parameter number> <packed mapping data> F7

Sets the MIDI mapping for the given parameter.

4FH – Set I2C mapping

F0 00 21 27 6D <SysEx ID> 4F <algorithm index> <16 bit parameter number> <packed mapping data> F7

Sets the I2C mapping for the given parameter.

50H – Request parameter value string

F0 00 21 27 6D <SysEx ID> 50 <algorithm index> <16 bit parameter number> F7

Requests the value string for a parameter. Responds with '50H – Parameter value string', as below.

51H – Set algorithm name

F0 00 21 27 6D <SysEx ID> 51 <algorithm index> <string> F7

Sets the algorithm’s custom name (as shown on the overview screen).

52H – Request parameter pages

F0 00 21 27 6D <SysEx ID> 52 <algorithm index> F7

Requests the parameter pages for an algorithm. Responds with '52H – Parameter pages', as below.

53H – Set string parameter value

F0 00 21 27 6D <SysEx ID> 53 <algorithm index> <16 bit parameter number> <string> F7

Sets the value of a string-valued parameter.

56H – Request paths

F0 00 21 27 6D <SysEx ID> 56 F7

Requests the current preset paths. Responds with ‘56H – Preset paths’, as below.

60H – Request number of algorithms

F0 00 21 27 6D <SysEx ID> 60 F7

Requests the number of algorithms in the current preset. Responds with ‘60H – Number of algorithms’, as below.

61H – Request routing

F0 00 21 27 6D <SysEx ID> 61 <algorithm index> F7

Requests routing information for the given algorithm. Responds with ‘61H – Routing information’, as below.

62H – Request CPU usage

F0 00 21 27 6D <SysEx ID> 62 F7

Requests CPU usage information. Responds with ‘62H – CPU usage’, as below.

7FH – Reboot

F0 00 21 27 6D <SysEx ID> 7F F7

F0 00 21 27 6D <SysEx ID> 7F 7F F7

There are two forms of this message. The first simply reboots the module. The second (with the double 7FH) reboots the module into bootloader mode.

Sent SysEx messages

09H – Lua output

F0 00 21 27 6D <SysEx ID> 09 <text> F7

This message may be transmitted in response to ‘08H – Execute Lua’ or ‘09H – Install Lua’.

30H – Number of algorithms

F0 00 21 27 6D <SysEx ID> 30 <16 bit value> F7

This message is transmitted in response to ‘30H – Request number of algorithms’.

31H – Algorithm info

F0 00 21 27 6D <SysEx ID> 31 <16 bit index> <4 byte guid> <number of specifications> [<specification min> <max> <default> <type> …] <NULL terminated algorithm name> [<NULL terminated specification name> …] <plugin?> <loaded?> <NULL terminated filename> F7

This message is transmitted in response to ‘31H – Request algorithm info’.

32H – Message

F0 00 21 27 6D <SysEx ID> 32 <NULL terminated ASCII string> F7

This message is transmitted in response to any request for a string e.g the version string.

33H – Screenshot

F0 00 21 27 6D <SysEx ID> 33 00 <screenshot data> F7

This message is transmitted in response to a '01H – Take screenshot' message.

40H – Algorithm guid

F0 00 21 27 6D <SysEx ID> 40 <index> <4 byte guid> <string> F7

This message is transmitted in response to a ‘40H – Request algorithm guid’ message. The string is the algorithm’s custom name, as shown on the overview screen.

41H – Preset name

F0 00 21 27 6D <SysEx ID> 41 <NULL terminated ASCII string> F7

Contains the current preset name.

42H – Number of parameters

F0 00 21 27 6D <SysEx ID> 42 <algorithm index> <16 bit number of parameters> F7

Contains the number of parameters in the indexed algorithm.

43H – Parameter info

F0 00 21 27 6D <SysEx ID> 43 <algorithm index> <16 bit parameter number> <16 bit minimum> <16 bit maximum> <16 bit default> <unit> <ASCII string> <scaling | flags> F7

Contains information for the given parameter in the indexed algorithm. Scaling is a 2 bit value and uses the low 2 bits.

44H – All parameter values

F0 00 21 27 6D <SysEx ID> 44 <algorithm index> [<16 bit value>] F7

Contains the current values of all parameters in the indexed algorithm. The high byte of each 16 bit value uses the spare space to include flags in bits 2 and up.

45H – Parameter value

F0 00 21 27 6D <SysEx ID> 45 <algorithm index> <16 bit parameter number> <16 bit value> F7

Contains the value of the given parameter in the indexed algorithm. The high byte of the 16 bit value uses the spare space to include flags in bits 2 and up.

48H – Unit strings

F0 00 21 27 6D <SysEx ID> 48 <number of strings> [<ASCII string>] F7

Contains an array of string descriptions of the possible parameter units (Hz, ms etc.).

49H – Enum strings

F0 00 21 27 6D <SysEx ID> 49 <algorithm index> <16 bit parameter number> <number of strings> [<ASCII string>] F7

Contains the value strings for an 'enum' type parameter.

4BH – Mapping

F0 00 21 27 6D <SysEx ID> 4B <algorithm index> <16 bit parameter number> <version number> <mapping data> F7

Contains the mapping information for one parameter.

50H – Parameter value string

F0 00 21 27 6D <SysEx ID> 50 <algorithm index> <16 bit parameter number> <ASCII string> F7

Contains a value string for a parameter.

52H – Parameter pages

F0 00 21 27 6D <SysEx ID> 52 <algorithm index> <number of pages> [<title> <number of parameters> [<2 byte parameter index>]] F7

Contains the parameter pages for an algorithm.

56H – Preset paths

F0 00 21 27 6D <SysEx ID> 56 <preset path> <End-of-chain preset path> <Preset save folder> F7

Sent in response to ‘56H – Request paths’, as above. Each path is null-terminated, and may simply be a single null character if empty.

60H – Number of algorithms

F0 00 21 27 6D <SysEx ID> 60 <count> F7

Sent in response to ‘60H – Request number of algorithms’, as above.

61H – Routing information

F0 00 21 27 6D <SysEx ID> 61 <algorithm index> <routing data> F7

Sent in response to ‘61H – Request routing’, as above.

62H – CPU usage

F0 00 21 27 6D <SysEx ID> 62 <percentage bytes> F7

Sent in response to ‘62H – Request CPU usage’, as above. Each byte is a percentage (0-100). The first is the audio thread total usage; the second is the overall usage. Following these is one byte for each algorithm in the preset.

7AH – SD card access

F0 00 21 27 6D <SysEx ID> 7A <variable> F7

Message 7AH defines a whole sub-protocol for accessing the module’s MicroSD card. See below for details.

MicroSD card access

The module’s MicroSD card can be accessed via the 7AH message. The format of the message is

F0 00 21 27 6D <SysEx ID> 7A <command> [<command specific data>] <checksum> F7

The checksum is chosen so that the low 7 bits of the sum of the message bytes from the command up to and including the checksum are zero.

If a command fails, the module responds with

F0 00 21 27 6D <SysEx ID> 7A 01 <NULL terminated error message> F7

If a command succeeds, the module responds with

F0 00 21 27 6D <SysEx ID> 7A 00 <command> [<command specific data>] F7

The available commands are as follows.

01H – Directory listing

F0 00 21 27 6D <SysEx ID> 7A 01 <folder path string> <checksum> F7

The response data is a NULL terminated array of entries, one for each item in the directory, formatted as

40H | <FAT32 attrib byte>
<FAT32 date, short as 3 bytes>
<FAT32 time, short as 3 bytes>
<file size, 64 bit value as 10 bytes>
<NULL terminated filename>

02H – Download file

F0 00 21 27 6D <SysEx ID> 7A 02 <file path string> <checksum> F7

The response data is the entire file, with each byte in the file transmitted as two bytes, high nybble first.

03H – Delete file

F0 00 21 27 6D <SysEx ID> 7A 03 <file path string> <checksum> F7

Deletes the specified file. The response is simply success or failure.

04H – Write file

F0 00 21 27 6D <SysEx ID> 7A 04 <file path string> <create always?> <offset> <count> <file data> <checksum> F7

Writes data to the specified file. “Create always?” is a flag – if true, the file will always be created anew (and any existing file of the same name deleted), otherwise the data will be appended to any existing file. “Offset” and “Count” are 64 bit values represented as 10 bytes, high bits first. The file data is transmitted as two bytes per byte in the file, high nybble first.

The response is simply success or failure.

05H – Rename

F0 00 21 27 6D <SysEx ID> 7A 05 <file path string> <new path string> <checksum> F7

Renames or moves the specified file. The response is simply success or failure.

06H – Remount

F0 00 21 27 6D <SysEx ID> 7A 06 <checksum> F7

Remounts the MicroSD file system. The response is always success.

07H – New folder

F0 00 21 27 6D <SysEx ID> 7A 07 <folder path string> <checksum> F7

Creates a new folder. The response is simply success or failure.

08H – Rescan plug-ins

F0 00 21 27 6D <SysEx ID> 7A 08 <checksum> F7

Attempts to reset the plug-in list and rescan the MicroSD card for plug-ins. The response is simply success or failure. (Failure would indicate that one or more plug-ins is still in use.)

I2C reference

In general the disting NT acts as a follower on the I2C bus, not as a leader. It receives messages in the following format:

<address> <command> <optional bytes according to command>

A table of supported commands is below.

Some commands are “get” commands. The disting NT expects the get command to be followed immediately by a read of the requested data.

Devices on an I2C bus have an address, which a sending device uses to identify the intended recipient. The disting NT's address is set in the Settings (see above).

Value ranges

Preset, algorithm & parameter numbers are 1-based.

Voltages (and related quantities e.g. pitch) are signed and scaled as 16384 ↔ 10V.

Velocities are 0-16384.

I2C Channels

The note on/off messages have the concept of a ‘channel’, which is entirely analogous to a MIDI channel. Algorithms which receive notes will have a parameter to set which I2C channel to receive on, and will only respond to notes on that channel. Messages which don’t explicitly contain a channel use the last channel set via the 0x6B message.

The “current algorithm”

Much of this I2C protocol is based on that developed for the disting EX, which ran a single algorithm, so commands to set e.g. algorithm parameters didn’t need to specify which algorithm was the target.

Rather than develop a whole new protocol, the current solution (or workaround, depending on your point of view) is to define a “current algorithm”, which all algorithm-relative commands refer to. This is set using the 0x46 or 0x47 messages, using a parameter number of 255. The value set to parameter 255 (which is an invalid parameter number on all existing algorithms) will be used as the index (1-based) in the preset of the “current algorithm”.

Messages

Controllers/parameters

Set i2c controller X to value Y

<address> 0x11 <controller number> <value MSB> <value LSB>

These messages are used via the Mappings (see above) to control algorithm parameters.

Set parameter X to value Y (using the actual parameter value)

<address> 0x46 <parameter number> <value MSB> <value LSB>

As a temporary workaround, parameter number 255 is used to specify the target algorithm for subsequent parameter change messages.

Set parameter X to value Y (using 0..16384 range)

<address> 0x47 <parameter number> <value MSB> <value LSB>

The 0-16384 value range will be scaled to the actual parameter value range.

Get parameter value

<address> 0x48 <parameter number>

Returns 2 bytes.

Get parameter min

<address> 0x49 <parameter number>

Returns 2 bytes.

Get parameter max

<address> 0x4A <parameter number>

Returns 2 bytes.

Notes

Set pitch for note id.

<address> 0x54 <note id> <pitch MSB> <pitch LSB>

Note on for specified note id.

<address> 0x55 <note id> <velocity MSB> <velocity LSB>

Note off for specified note id.

<address> 0x56 <note id>

All notes off.

<address> 0x57

Set pitch for note id, with channel.

<address> 0x68 <channel> <note id> <pitch MSB> <pitch LSB>

Note on for specified note id, with channel.

<address> 0x69 <channel> <note id> <velocity MSB> <velocity LSB>

Note off for specified note id, with channel.

<address> 0x6A <channel> <note id>

Set channel for subsequent note based commands.

<address> 0x6B <channel>

MIDI

Send MIDI message from the breakout.

<address> 0x4F <status> <optional data byte 0> <optional data byte 1>

Send MIDI message from the Select Bus.

<address> 0x50 <status> <optional data byte 0> <optional data byte 1>

Send MIDI to the disting NT.

<address> 0x81 <byte 0>

<address> 0x82 <byte 0> <byte 1>

<address> 0x83 <byte 0> <byte 1> <byte 2>

Messages 0x81-83 allow you to send arbitrary MIDI to the disting NT, as if the I2C connection were just another MIDI cable into the module.

The various message forms are a convenience (e.g. the 3 byte form is convenient for note on/off or CC messages), but in terms of processing, the disting NT treats them all as a byte stream.

Updating the firmware

The disting NT's firmware is updated over its USB connection. You will need a computer and the appropriate USB cable to connect the module to it.

You can jump directly to any firmware version – you don't need to apply them incrementally. You can upgrade or downgrade at will, but note that older firmware versions may not be able to read presets saved with newer versions; similarly, if you downgrade you may find that your settings reset to defaults.

There are a number of ways of updating the firmware. The simplest for most users will be to use the NT Helper[147] app for macOS/Windows/Linux. Skip down to "Method 1: NT Helper" if you're most users.

We also support two somewhat lower-level methods of flashing the firmware – one using a GUI tool written by the chip vendor NXP, and one using a script. The latter will probably be quicker and simpler if you're comfortable with using the terminal/command prompt on your computer.

In both cases, you first need to:

- Download the firmware from our website here[148].
- Put the disting NT into bootloader mode, which you do via the 'Misc' menu. Select "Enter bootloader mode", click past the "Are you sure?", and the module will show you the message "Entering serial downloader" and appear to hang.

If you decide you don't actually want to flash new firmware at this point, simply turn the module off and on again.

Method 1: NT Helper

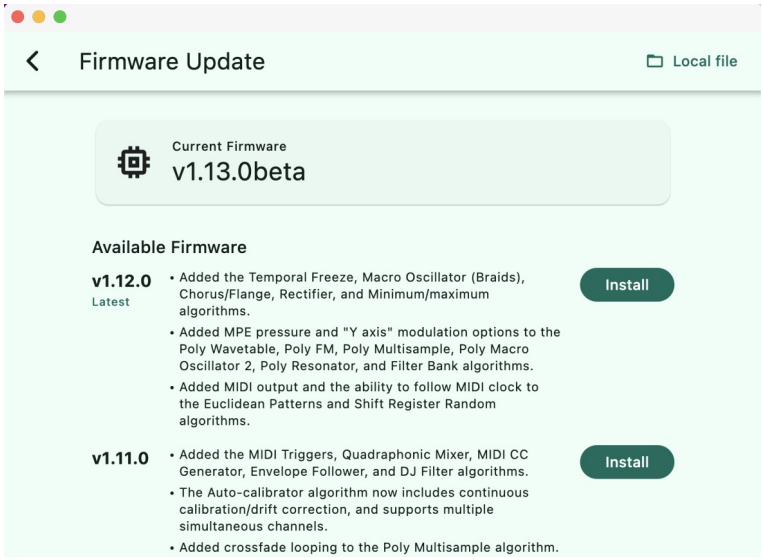
Instructions are on the author's page here[149]. You can access the firmware update page by clicking on the version number (bottom right) or via the menu (top right).

The app allows you to simply choose an official release directly from our website, or to install a file that you've downloaded locally.

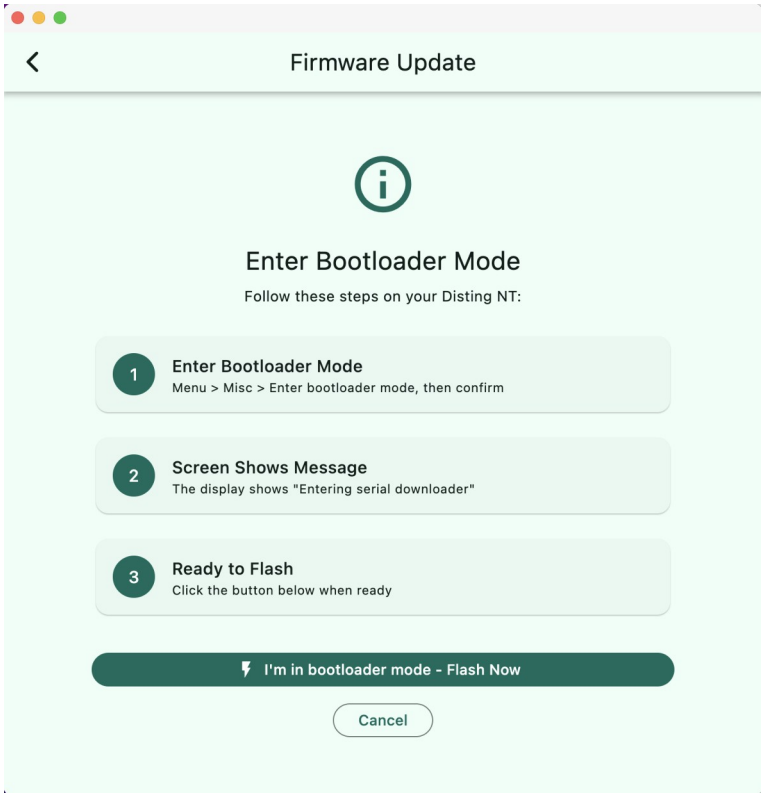
147 https://nosuch.dev/nt-helper/

148 https://expert-sleepers.co.uk/distingNTfirmwareupdates.html

149 https://nosuch.dev/nt-helper/#firmware-update



Once you've chosen the version to install, the app walks you through the update process.



Method 2: NXP GUI tool

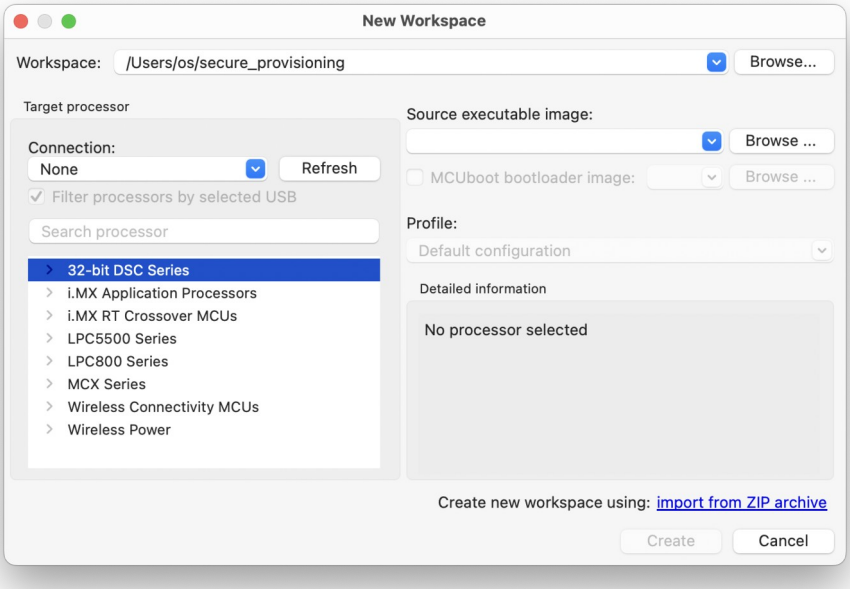
When downloading the firmware, do not let the browser automatically unzip the file (in Safari the option is "Open safe files after downloading" - that needs to be off). It is most unlikely to work if you attempt to make a zip from the unarchived files.

Download and install the correct version of the "MCUXpresso Secure Provisioning Tool". It is available for macOS, Linux, and Windows. Note that various versions of the tool may be available.

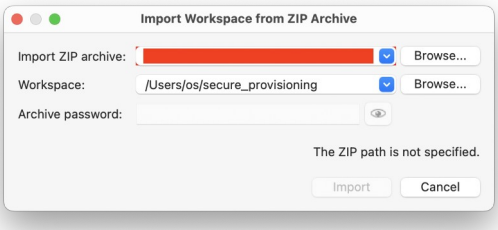
You need to use the correct version that matches the disting NT firmware package that you want to install. The tool version for each firmware version is noted on the firmware download page.

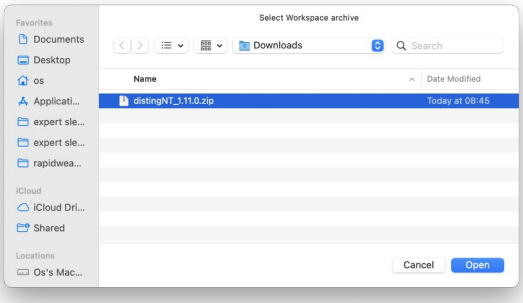
Download the tool for your platform of choice from this page[150]. The NXP website requires you to sign up for an account to do so and asks for a surprising amount of information. You can of course just make this up if you prefer to remain anonymous.

When you first run the tool, you should be presented with the “New Workspace” dialog:

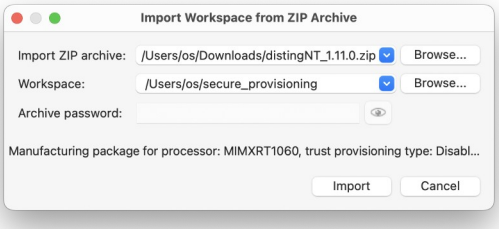


Click the text at the bottom that says “import from ZIP archive”. Browse for the firmware zip file that you downloaded.



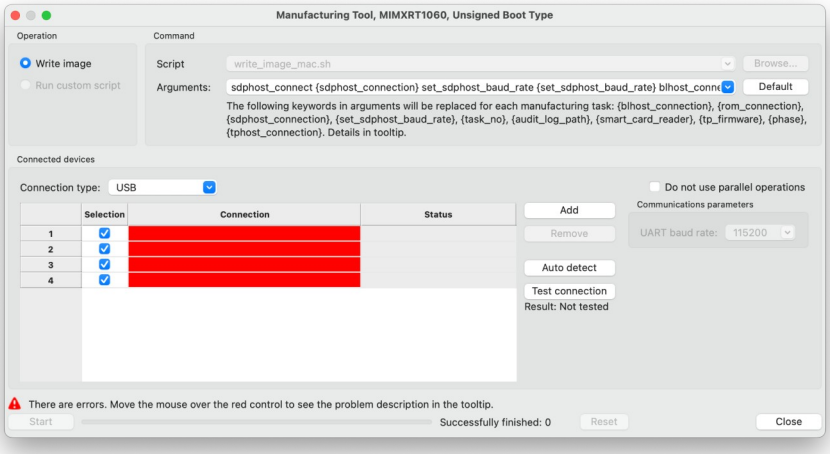


150 https://www.nxp.com/design/design-center/software/development-software/mcuxpresso-software-and-tools-/mcuxpresso-secure-provisioning-tool:MCUXPRESSO-SECURE-PROVISIONING

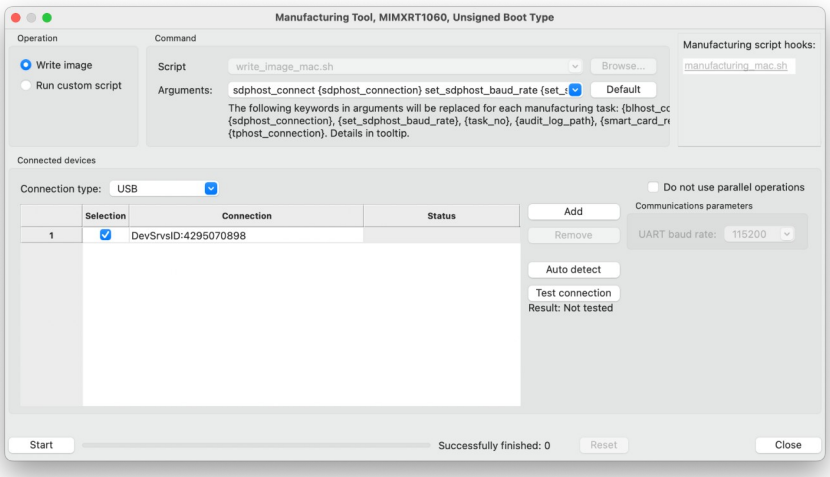


You can also choose where the tool creates its “workspace” - a folder containing a variety of temporary files that you can delete later, or just leave for next time.

Click the “Import” button.

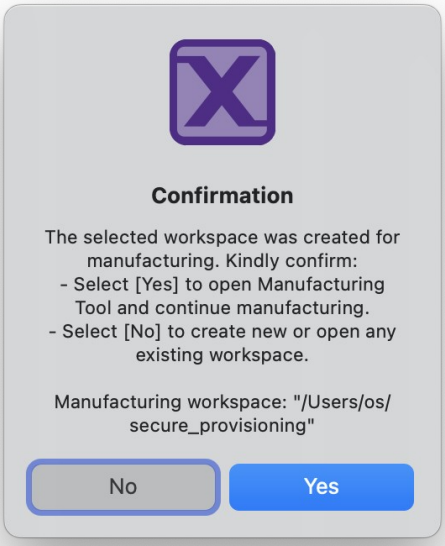


You should now see a window titled “Manufacturing Tool”. The connection section may be all red; if so, click “Auto detect”. If a disting NT in bootloader mode is connected via USB, the tool will now find it.



You can then click on the “Start” button (bottom left) to actually perform the firmware update. This should take maybe 15 seconds, and show “Success” when done. Turn the module off and back on to resume normal operation. You may like to check that the “About” screen on the disting NT shows the expected firmware version.

Next time you run the tool, you may see this dialog:



Click “Yes” if you want to reflash the same firmware as before (perhaps you’re the lucky owner of two disting NTs). Otherwise, click “No”, which will take you back to the “New Workspace” dialog as above.

Method 3: Script

As mentioned above, this method is provided for users who prefer a command line approach over using a GUI application. In preparation for this:

1. Install SPSDK. Instructions are here[151]. Note that you may also need to install development tools; for example on macOS you may need to install Xcode.
2. Download the scripts from our GitHub. You might choose to clone the project using git or GitHub Desktop, or simply download a release from the releases page[152]. The scripts are in the ‘flash’ folder.

To flash the firmware:

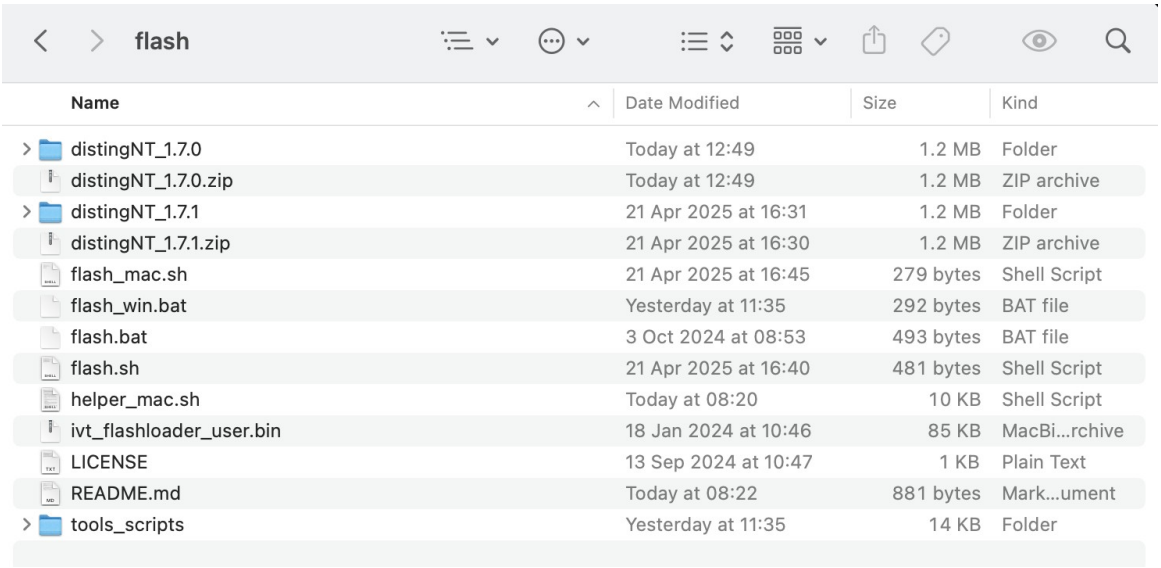
1. Put the downloaded firmware package into the ‘flash’ folder, and unzip it.
2. Put the disting NT into bootloader mode as described above.
3. Run the script ‘flash_mac.sh’ (macOS/Linux) or ‘flash_win.bat’ (Windows), supplying it the

151 https://spsdk.readthedocs.io/en/latest/examples/_knowledge_base/installation_guide.html

152 https://github.com/expertsleepersltd/distingNT/releases

folder name of the firmware you want to install.

For example, your ‘flash’ folder might look like this (here we’ve downloaded two firmware packages and unzipped them):



You would then run e.g.:

```
(base) oss-mac-studio:flash os$ ./flash_mac.sh distingNT_1.7.0
### Parse input arguments ###
### Check presence of FlashLoader ###
### FlashLoader is not running yet, download and run it ###
### Check communication with target bootloader ###
```

Acknowledgments

The Kirbinator algorithm was designed in collaboration with Simon Kirby[153].

The 24dB/oct VCF algorithm was written by Eddie Edwards at Reanimator Ltd[154].

Many thanks to all the beta testers.

Fonts

The small pixel font used throughout the UI is ‘PixelMix’ by Andrew Tyler, for which Expert Sleepers Ltd has a commercial license.

The large font is Microsoft Selawik (Copyright 2015, Microsoft Corporation), licensed under the SIL Open Font License version 1.1.

The tiny font used in parts of the UI is an adapted version of 'Tom Thumb' by Robey Pointer (MIT license).

Freetype

Lua

153 https://www.simonkirby.net

154 http://www.reanimator.ltd.uk