

REAKTOR 6

GETTING STARTED

The information in this document is subject to change without notice and does not represent a commitment on the part of Native Instruments GmbH. The software described by this document is subject to a License Agreement and may not be copied to other media. No part of this publication may be copied, reproduced or otherwise transmitted or recorded, for any purpose, without prior written permission by Native Instruments GmbH, hereinafter referred to as Native Instruments.

“Native Instruments”, “NI” and associated logos are (registered) trademarks of Native Instruments GmbH.

Mac, Mac OS, GarageBand, Logic, iTunes and iPod are registered trademarks of Apple Inc., registered in the U.S. and other countries.
Windows, Windows Vista and DirectSound are registered trademarks of Microsoft Corporation in the United States and/or other countries.
All other trade marks are the property of their respective owners and use of them does not imply any affiliation with or endorsement by them.

Document authored by: Adam Hanley, Jan Ola Korte

Software version: 6.3 (04/2019)

NATIVE INSTRUMENTS GmbH
Schlesische Str. 29-30
D-10997 Berlin
Germany
www.native-instruments.de

NATIVE INSTRUMENTS North America, Inc.
6725 Sunset Boulevard
5th Floor
Los Angeles, CA 90028
USA
www.native-instruments.com

NATIVE INSTRUMENTS K.K.
YO Building 3F
Jingumae 6-7-15, Shibuya-ku,
Tokyo 150-0001
Japan
www.native-instruments.co.jp

NATIVE INSTRUMENTS UK Limited
18 Phipp Street
London EC2A 4NU
UK
www.native-instruments.co.uk

NATIVE INSTRUMENTS FRANCE SARL
113 Rue Saint-Maur
75011 Paris
France

www.native-instruments.com

SHENZHEN NATIVE INSTRUMENTS COMPANY Limited
5F, Shenzhen Zimao Center

111 Taizi Road, Nanshan District, Shenzhen, Guangdong

China

www.native-instruments.com

NATIVE INSTRUMENTS

© NATIVE INSTRUMENTS GmbH, 2019. All rights reserved.

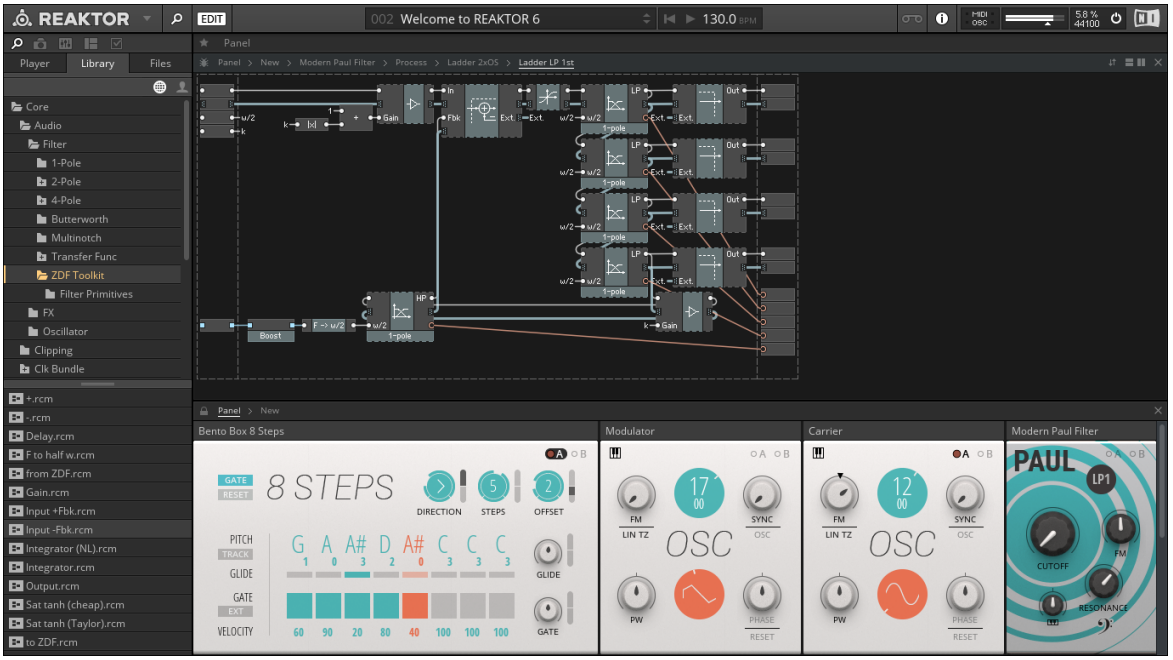
Table of Contents

1 Welcome to REAKTOR 6
1.1 System Requirements 7
1.2 REAKTOR Modes: Full, Player and Demo 7
1.3 The REAKTOR 6 Documentation 8
1.4 REAKTOR Content Documentation 9
1.5 Manual Conventions 12
2 Getting to Know REAKTOR 13
2.1 The Interface at a Glance 13
2.2 The Toolbar 14
2.2.1 Transport 16
2.2.2 Info Hints 17
2.2.3 Audio Input & Output Level Meters 18
2.2.4 System Info 19
2.2.4.1 Changing the Sample Rate 19
2.3 Play Mode vs Edit Mode 20
2.4 File Types and Hierarchy 21
2.5 Third-Party and Protected Content 26
3 Setting Up REAKTOR 28
3.1 Audio Settings 28
3.2 MIDI Settings 33
4 REAKTOR as a Plug-in 37
4.1 Editing Ensembles and Automatic Saving 38
4.2 Resolving a Broken Ensemble Link 42

5 Browsing and Loading Ensembles 46
6 Using Ensembles and Instruments 51
6.1 Instrument Header 51
6.2 A/B View 52
6.3 Snapshots and Presets 53
6.4 Recalling Snapshots 54
6.5 Storing Snapshots 55
6.6 Presets 57
7 Using Racks 61
7.1 Overview 61
7.2 Loading Racks 63
7.3 Saving Racks 65
7.4 Creating a New Rack 66
7.5 Additional Settings 66
8 Snapshots and Presets 67
9 Controlling REAKTOR 68
9.1 MIDI Learn 68
9.2 Connection Manager 69
9.3 MIDI Clock 76
9.4 Synchronization using Ableton Link 77
9.4.1 Connecting to a Network 78
9.4.2 Joining and Leaving a Link Session 78
10 Getting Deeper into REAKTOR 80
Index 82

1 Welcome to REAKTOR

REAKTOR is a visual programming environment based on modules and wires, allowing you to create any musical tool imaginable. Its many uses include music creation, digital art, instrument and effects building, as well as scientific applications. REAKTOR 6 builds on a long history of legendary software including NI's first product, GENERATOR.



Welcome to REAKTOR 6

Known for its unrivalled sound quality, REAKTOR is the complete toolkit for creating instruments and effects with high-quality DSP components. The included content allows you to look behind the curtain and explore the state-of-the-art technology at the core of many of our products. REAKTOR Racks facilitate analog-style modular patching with Blocks—the perfect introduction to using modules and wires to create your own instruments.

From playing and combining pre-built instruments and effects to building your own tools with its extensive Library or creating custom DSP algorithms, REAKTOR suits musicians, builders, and programmers alike. Based on a hierarchical structure with a coherent interface, REAKTOR is optimized to facilitate all of these use cases. This means that you can approach REAKTOR level by level, making yourself familiar with its deep functionality one step at a time.

The REAKTOR 6 documentation adapts this approach, guiding you through the different levels of functionality in a clear and comprehensible manner. The **Getting Started** and **Diving Deeper** documents cover all basic functions and settings in REAKTOR, while the tutorials and instructions provided in **Building in Primary** and **Building in Core** make creating your own musical tools more accessible than ever before.

REAKTOR is fortunate to be supported by a large community of experienced users who actively share their Ensembles in the REAKTOR User Library on our website. There are thousands of REAKTOR Ensembles available, ready to be played, combined and modified—a great source of inspiration, both musically and technologically. We invite you to join this vibrant community and are looking forward to seeing and hearing your new creations!

1.1 System Requirements

Your computer system needs to meet certain requirements in order to run REAKTOR.

For the latest REAKTOR system requirements and compatibility information visit:

www.native-instruments.com/reaktor6specifications

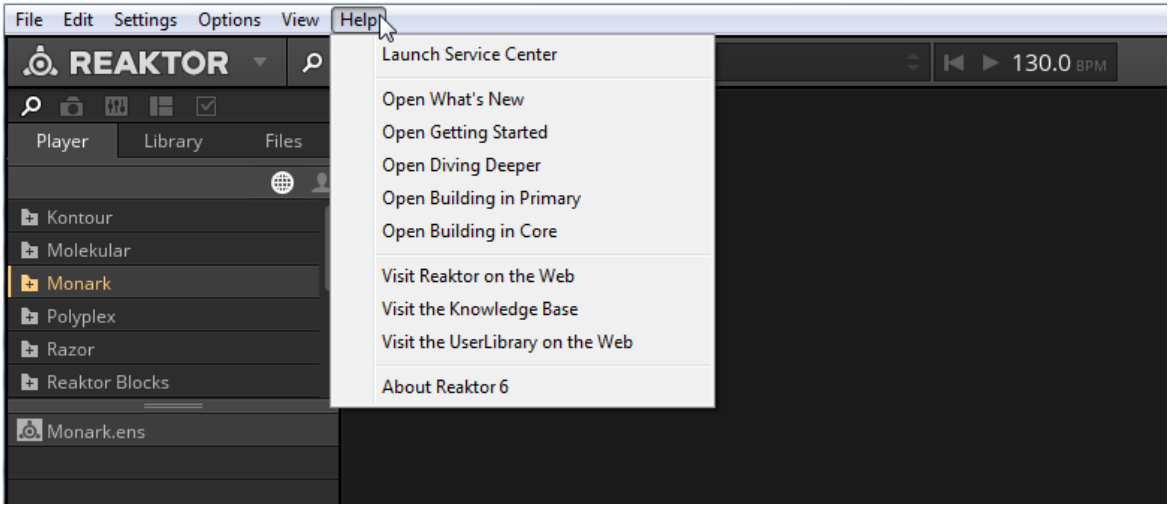
1.2 REAKTOR Modes: Full, Player and Demo

What you can do with the REAKTOR version installed on your computer depends on which mode REAKTOR is running in. This mode depends on the license you are using:

- **Full mode:** If you have already installed and activated REAKTOR, as a single product or as part of a bundle like KOMPLETE, you are using it in Full mode. In Full mode, all functionality is available without restrictions.

- **Player mode:** You have installed and activated any of the KOMPLETE instruments and effects for REAKTOR and the free REAKTOR Player. In Player mode you can play Instruments and change their settings, but cannot edit their internal structure. Play mode also allows you to patch and play with Blocks in Racks. All Instrument settings are saved with your host project when using REAKTOR as a plug-in.
- **Demo mode:** You installed REAKTOR and a number of KOMPLETE Instruments as demo versions, but did not activate any of the products yet. In this case you are using the demo license for REAKTOR, and REAKTOR is running in Demo mode. In Demo mode, you have all functionality but the application will stop generating sound after 30 minutes. Furthermore, storing is disabled.

1.3 The REAKTOR 6 Documentation



The REAKTOR documentation is accessible from the Help menu

The documentation for REAKTOR 6 is divided into five separate documents, guiding you from loading and playing pre-built Ensembles to building your own Instruments.

- **REAKTOR 6 What Is New** is written for users who are already familiar with previous versions of REAKTOR and only describes the latest features in brief.

- **REAKTOR 6 Getting Started** is for new users. It is the only document needed for users who intend to use REAKTOR for loading and playing pre-built REAKTOR instruments and effects.
- **REAKTOR Blocks Quick Start Guide** is an online resource showing you how to use REAKTOR Blocks in the Rack format. Find it here: www.native-instruments.com/reaktor-blocks-quickstart
- **REAKTOR 6 Diving Deeper** expands on the concepts introduced in the Getting Started document. It provides more detail on subjects like Snapshots (REAKTOR's preset system), and introduces advanced topics like OSC control and combining multiple Instruments in one Ensemble.
- **REAKTOR 6 Building in Primary** shows you how to build your own Instruments in REAKTOR's Primary level. It focuses on a series of tutorials that guide you through building your first synthesizers, effects, and sequencers.
- **REAKTOR 6 Building in Core** describes the Core level of REAKTOR with its low-level building features, which can be used for implementing custom DSP algorithms. It includes reference information about the Core Macro Library, an comprehensive collection of DSP building blocks.

With the exception of the What Is New document, each of the documents listed above builds on the knowledge of the previous documents. While it is not necessary to read all of every document, some of the more advanced documents, like Building in Primary, assume knowledge from the previous documents.



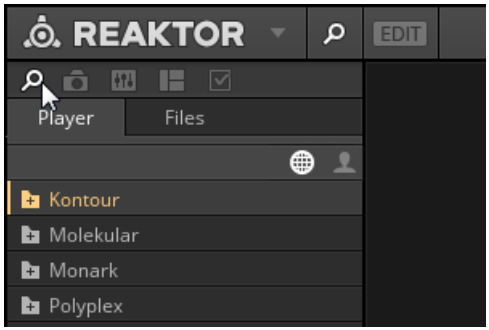
In-depth information about using REAKTOR Blocks in Ensembles can be found in the REAKTOR Blocks Manual. To learn about opening manuals for REAKTOR content, refer to ↑1.4, REAKTOR Content Documentation.

1.4 REAKTOR Content Documentation

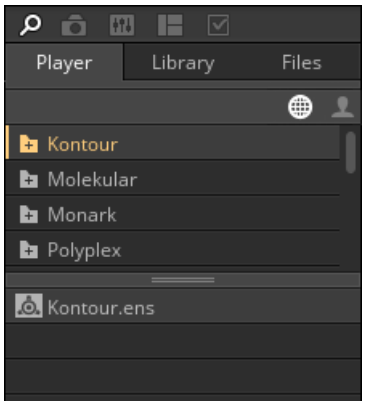
REAKTOR comes with a comprehensive Factory Library, including many pre-built instruments and effects, as well as Blocks, a new modular synthesizer framework that brings the experience of patching an analog modular synthesizer to REAKTOR. The manuals for this content, as well as any REAKTOR-based KOMPLETE instruments or effects you have installed, can be accessed from the Browser in REAKTOR's Side Pane.

To open the manuals for REAKTOR content:

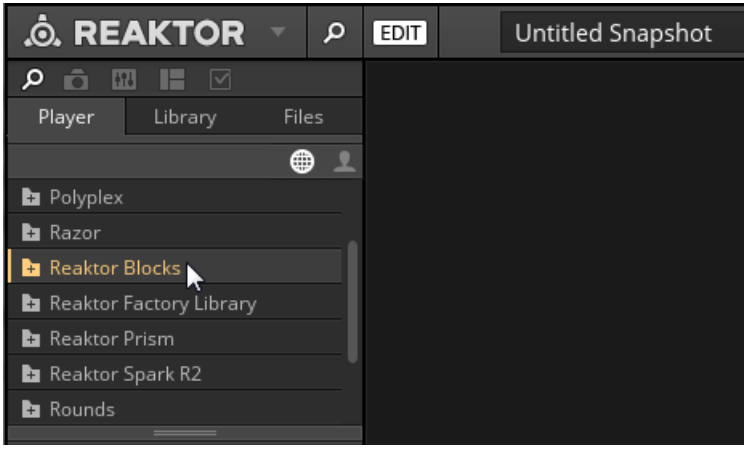
1. Click on the Browser icon in the Side Pane to open the Browser.



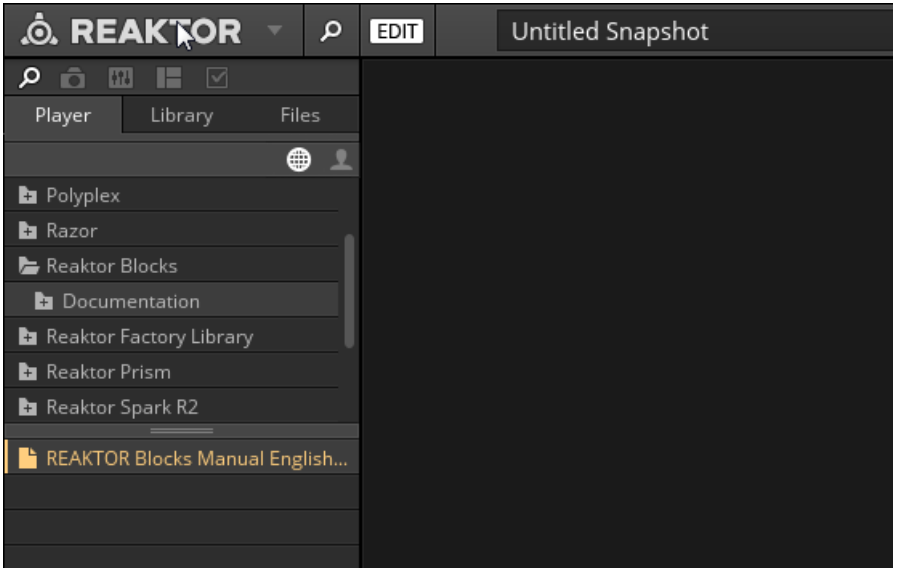
2. Select the Player tab.



3. Double-click on the folder of the content you wish to view the manual for.



4. Select the Documentation subfolder. The manual shows up as a PDF file in the lower section of the Browser.



5. Double-click on the PDF file.

⇨ The manual will open in your standard PDF reader.

1.5 Manual Conventions

This section introduces you to the signage and text highlighting used in this manual.

- Text appearing in (drop-down) menus (such as Open..., Save as... etc.) and paths to locations on your hard disk or other storage devices is printed in italics.
- Text appearing elsewhere (labels of buttons, controls, text next to checkboxes etc.) is printed in blue. Whenever you see this formatting applied, you will find the same text appearing somewhere on the screen.
- Important names and concepts are printed in **bold**.
- References to keys on your computer's keyboard you'll find put in square brackets (e.g., "Press [Shift] + [Enter]").

1. Single instructions are introduced by this play button type arrow.

⇨ Results of actions are introduced by this smaller arrow.

An indented, gray paragraph contains additional, contextual information.

Furthermore, this manual uses particular formatting to point out special facts and to warn you of potential issues. The icons introducing these notes let you see what kind of information is to be expected:



The speech bubble icon indicates a useful tip that may often help you to solve a task more efficiently.



The exclamation mark icon highlights important information that is essential for the given context.



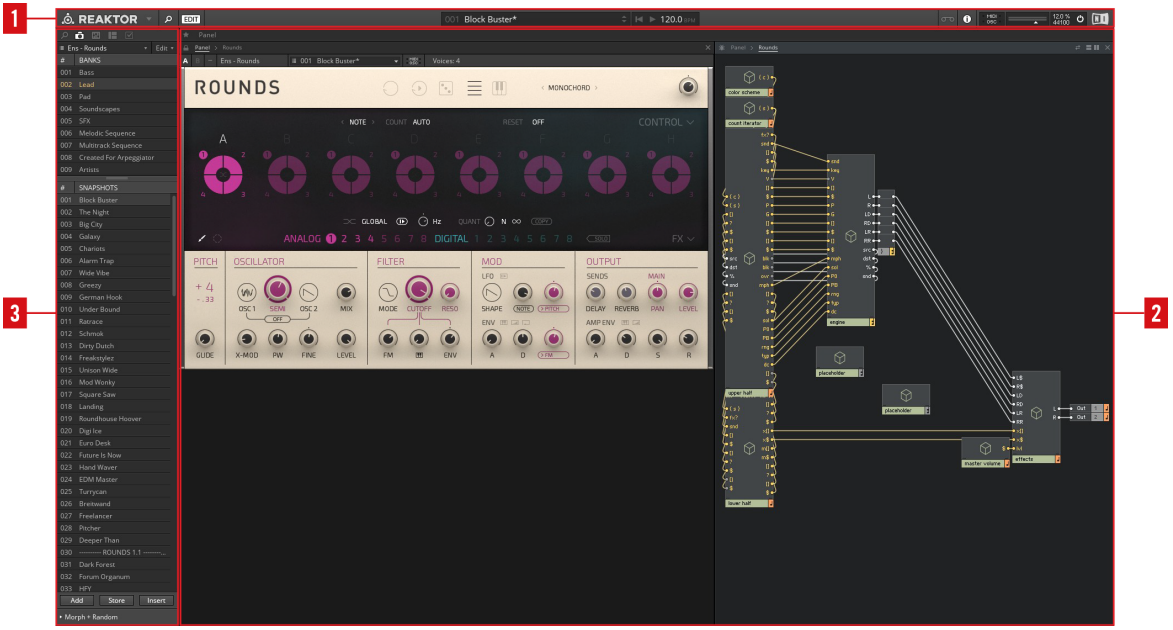
The red cross icon warns you of serious issues and potential risks that require your full attention.

2 Getting to Know REAKTOR

REAKTOR can be used in many different ways, from loading and playing pre-built Ensembles, to building new Instruments using the Modules and Macros Library, to programming your own DSP building blocks in Core. REAKTOR's interface and file hierarchy are optimized to facilitate all these use cases, offering clear workflows for the task at hand. This section gives an overview over the interface, the Play Mode and the Edit Mode, as well as the file types used in REAKTOR.

2.1 The Interface at a Glance

REAKTOR's interface can be divided into three main areas:



The REAKTOR Interface

(**1**) **Toolbar**: The area along the top is called the Toolbar and contains a lot of REAKTOR's global options and settings (see section ↑2.2, The Toolbar).

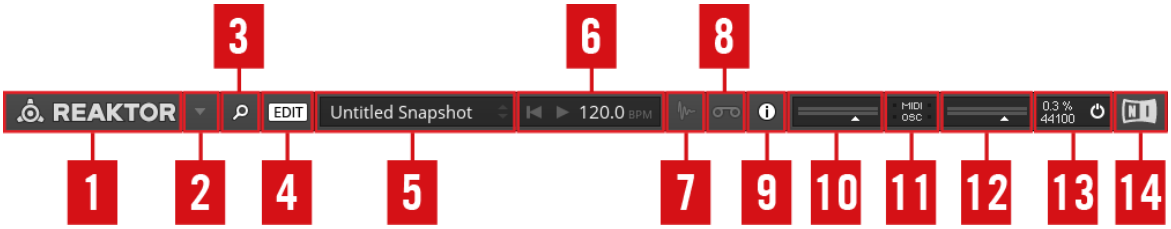
(**2**) **Main area**: The main area in the center displays either the loaded Instrument Panel, the Structure of the Ensemble, or both (the area can be split with the options in the upper-right corner).

(**3**) **Side Pane**: The area to the left is called the **Side Pane** and contains a number of different functions categorized by tabs:

- Browser (see section The Browser)
- Snapshots (see section ↑6.3, Snapshots and Presets)
- Connection Manager (see section ↑9.2, Connection Manager)
- Panelsets (find more information in the Diving Deeper document)
- Properties (find more information in the Diving Deeper document)

2.2 The Toolbar

The toolbar, located along the top of REAKTOR's interface, gives you access to many of the main options and parameters in REAKTOR.



The toolbar in REAKTOR

(**1**) **REAKTOR icon**: Clicking on the REAKTOR icon opens a window displaying information about the software. This window can be used to find the version number of your current REAKTOR installation, which is located to the top left of the window.

(**2**) **Open main menu**: The arrow to the right of the REAKTOR icon opens the main menu. This contains many different options and functions which are referenced throughout the documentation.

(**3**) **Show Side Pane**: The magnifying glass button shows or hides the Side Pane. The Side Pane hosts a variety of functions and settings, organized in tabs. See sections The Browser, ↑6.3, Snapshots and Presets, and ↑9.2, Connection Manager for more information. The Panelsets and Properties tabs are described in the Diving Deeper document.

(**4**) **Toggle Edit Mode**: The EDIT button toggles Edit Mode on or off. See section ↑2.3, Play Mode vs Edit Mode for more information.

(**5**) **Snapshot menu**: The Snapshot menu is where you can browse and load Snapshots from the loaded Ensemble. Snapshots are REAKTOR's main sound preset type and are explored in more detail in section ↑6.3, Snapshots and Presets. Clicking on the Snapshot name opens the snapshot menu. The up and down arrows to the right of the menu can be used to cycle through the sSnapshots.

(**6**) **Transport**: The transport controls are used to control the global playback. See section ↑2.2.1, Transport for more information.

(**7**) **Show Sample Map Editor**: The button with the waveform icon shows or hides the Sample Map Editor. This button is only available if the loaded Ensemble contains a Sampler Module. The Sample Map Editor is described in the Diving Deeper document.

(**8**) **Show Player and Recorder**: The button with the tape reel icon shows or hides the Player and Recorder. The Player and Recorder utilities allow you to play back and record audio files. The Player and Recorder are described in the Diving Deeper document.

(**9**) **Show Info Hints**: This button shows or hides the tool tip information when placing the mouse over an element in the user interface. See section ↑2.2.2, Info Hints for more information.

(**10**) **Audio input level meter**: The audio input level meter lets you monitor and adjust the input level for signals arriving at REAKTOR's inputs. The audio input meter is only visible if the loaded Ensemble includes audio inputs. See section ↑2.2.3, Audio Input & Output Level Meters for more information.

(**11**) **MIDI/OSC input and output**: The MIDI/OSC input and output indicators light up when REAKTOR sends or receives any MIDI and OSC messages. MIDI and OSC are the main ways of controlling REAKTOR with hardware. The indicators are useful for checking if your hardware is communicating with REAKTOR. See section ↑9, Controlling REAKTOR for more information about MIDI and OSC.

(**12**) **Audio output level meter**: The audio output level meter lets you monitor and adjust the output level for signals sent from REAKTOR's outputs. See section ↑2.2.3, Audio Input & Output Level Meters for more information.

(**13**) **System info**: The system info area contains displays and controls for the system and audio status. See section ↑2.2.4, System Info for more information.

(**14**) **NI icon**: Clicking on the NI icon opens a window displaying information about the software. This window can be used to find the version number of your current REAKTOR installation, which is located to the top left of the window.

2.2.1 Transport



The Transport controls in REAKTOR

Since REAKTOR can be used to create sequencers, REAKTOR has its own transport for controlling playback.

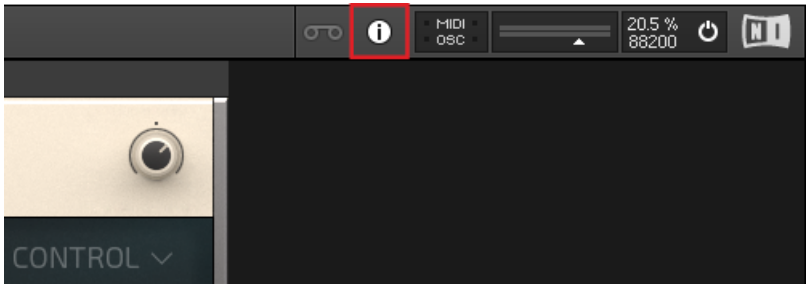
The transport controls are not available in the plug-in version of REAKTOR since it uses the host's transport settings.

- The leftmost button resets the play position (meaning it rewinds the play head to position zero). You can also reset the play position by pressing [ctrl] + [space] (Windows) or [cmd] + [space] (OS X) on your computer keyboard.
- The center button is the play button. It is used to start or pause playback. You can also start or pause playback by pressing [space] on your computer keyboard.
- The value display to the right is where you can define the playback tempo in BPM (beats per minute). You can do this either by clicking and dragging the value up and down, or by double-clicking and typing in a new tempo.

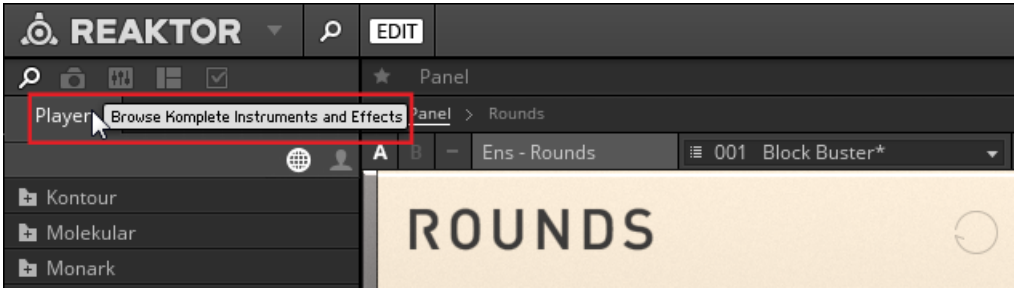
2.2.2 Info Hints

Many of the elements in REAKTOR's user interface show tool tip information when you place the mouse over them. This information is called an Info Hint in REAKTOR. The Info Hints can be enabled from the toolbar.

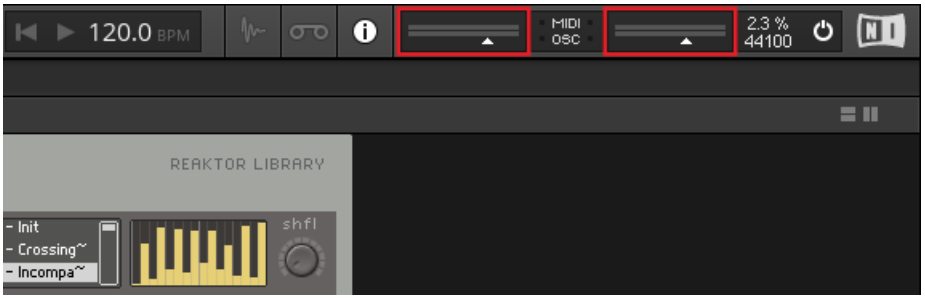
1. Click on the Show Info Hints button in the toolbar to enable the Info Hints. Alternatively, press the keyboard shortcut [ctrl] + [i] (Windows) or [cmd] + [i] (OS X).



⇨ The Info Hints are now shown as soon as you place the mouse over an element in REAKTOR'S user interface.



2.2.3 Audio Input & Output Level Meters



The audio input and output level meters in REAKTOR

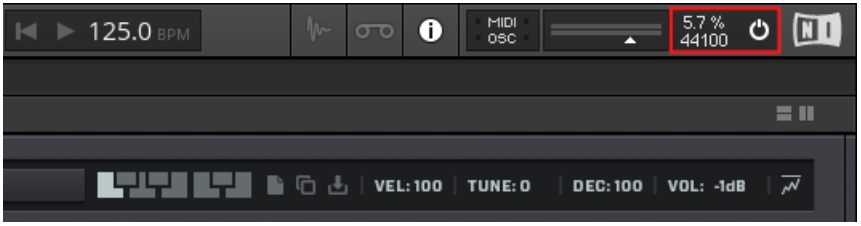
The audio input and output level meters let you monitor and adjust the level of audio signals arriving at REAKTOR's audio inputs as well as audio signals sent from REAKTOR's audio outputs.

The audio input level meter is only visible if the loaded Ensemble includes audio inputs.

If the signal level is too high and the input or output is overloading, red lights will appear at the right edge of the meters.

Below each of the meters is an arrow. Clicking and dragging on the arrow sets the input or output signal level. For example, if the output of an instrument is too loud, you can lower the signal level by moving the arrow below the output meter to the left.

2.2.4 System Info



The system info in REAKTOR

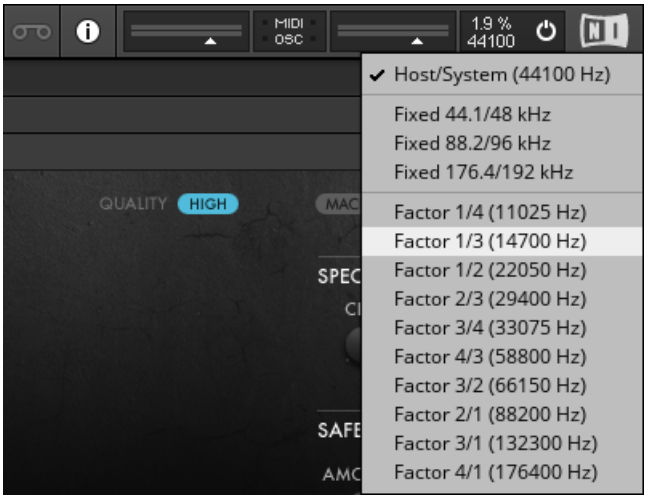
The rightmost area in the toolbar contains the system info and controls.

- The top left desplay shows a measure of the CPU load in percent.
- The bottom left display shows the currently set sample rate. Clicking on it opens a drop-down menu for selecting the sample rate (see section ↑2.2.4.1, Changing the Sample Rate).
- The power button to the right stops and starts REAKTOR's audio engine. If the audio coming from an Ensemble cuts out unexpectedly, sometimes switching the power button off and on can fix the problem.

2.2.4.1 Changing the Sample Rate

By default REAKTOR uses the sample rate of your audio interface for processing. This is also the default setting for most Ensembles. However, some Ensembles (e.g. MONARK, ROUNDS) use a higher sample rate to improve audio quality.

1. Clicking on the sample rate display opens the sample rate drop-down menu.



The sample rate setting in the toolbar

The sample rate menu contains a number of entries, organized in three categories:

- Host/System: This entry sets REAKTOR to operate at the same sample rate as your audio interface. This is the default setting for most Ensembles.
- Fixed:: The fixed entries are multiples of your audio interface's sample rate. They can be saved with the Ensemble and are used to improve the audio quality. Keep in mind that higher sampling rates require more CPU.
- Factor:: These entries allow you to reduce or increase the sample rate in the current REAKTOR session.

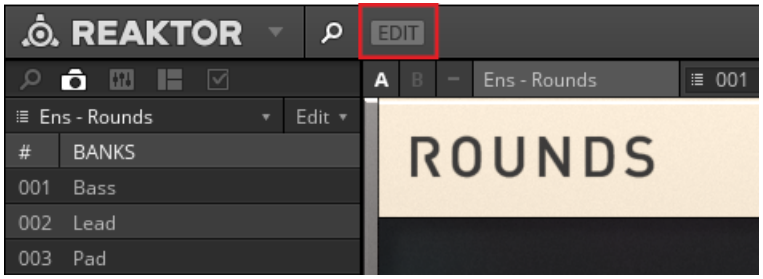
2.3 Play Mode vs Edit Mode

REAKTOR has two modes of operation:

- **Play Mode**: This mode optimizes REAKTOR for loading and playing Ensembles and Racks. It decreases loading times and streamlines the interface, but removes the options for deep editing of Ensembles and Instruments. This mode is especially useful when using REAKTOR in a DAW or host.

- **Edit Mode**: This mode gives you access to all of the controls and options REAKTOR has to offer, allowing you to create or edit Ensembles and Instruments.

1. To toggle Edit Mode on or off, click on the EDIT button in REAKTOR's toolbar.

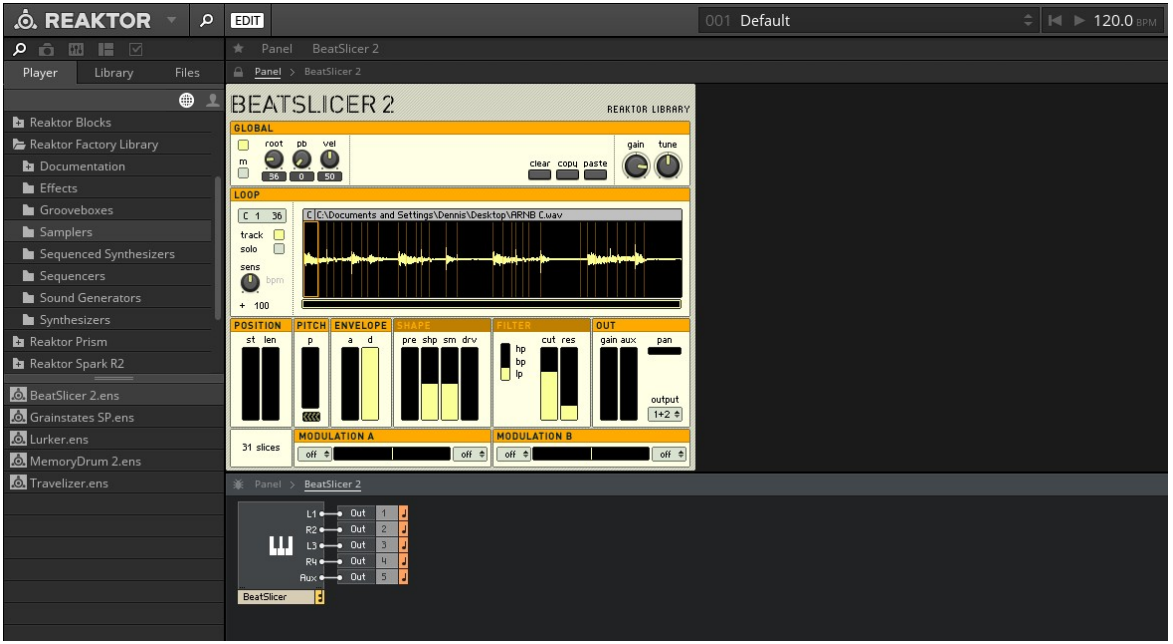


2.4 File Types and Hierarchy

REAKTOR uses several distinct file types, each representing a different structural level in the application. These levels help to organize your REAKTOR projects in a meaningful way, and the corresponding files can be used to share aspects of your work between projects or with other users.

Ensembles

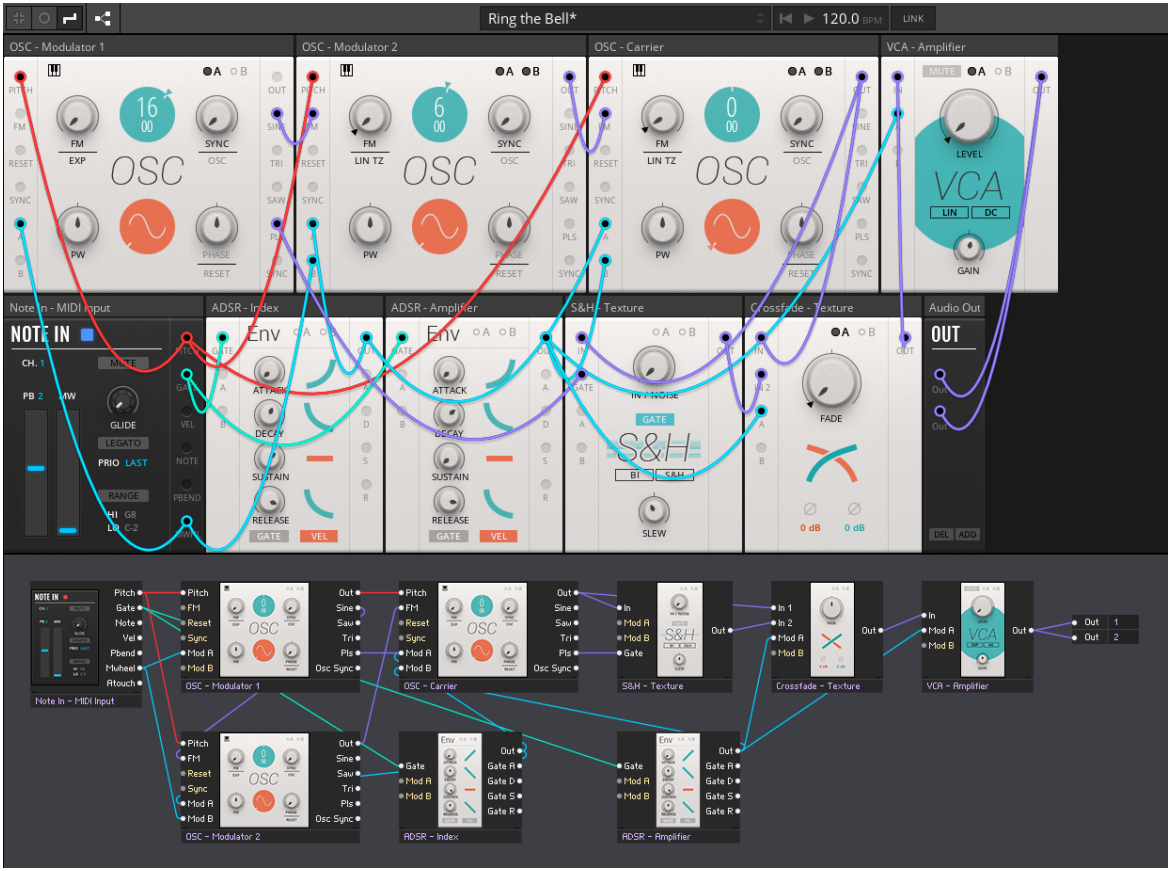
At the top level is the **Ensemble**, which holds all relevant information of a REAKTOR session. It represents the basic project file you are working with in REAKTOR. An Ensemble can contain a single instrument or a full suite of sequencers, synthesizers, and effects. Pre-built instruments and effects also come as Ensembles, ready to be used in your music or sound design projects. Ensembles are saved as *.ens files.



A REAKTOR Ensemble containing one Instrument

Racks

Racks facilitate analog-style modular synthesizer patching using pre-built modules in the Blocks format. In a Rack, you can arrange, connect, and control Blocks directly in the Panel. The patch is also mirrored in the Structure, allowing you to choose your preferred patching workflow. All settings including the arrangement of Blocks and the connections made between them are saved and recalled with your host sequencer project (host chunk saving). Racks are saved as *.nksr files.



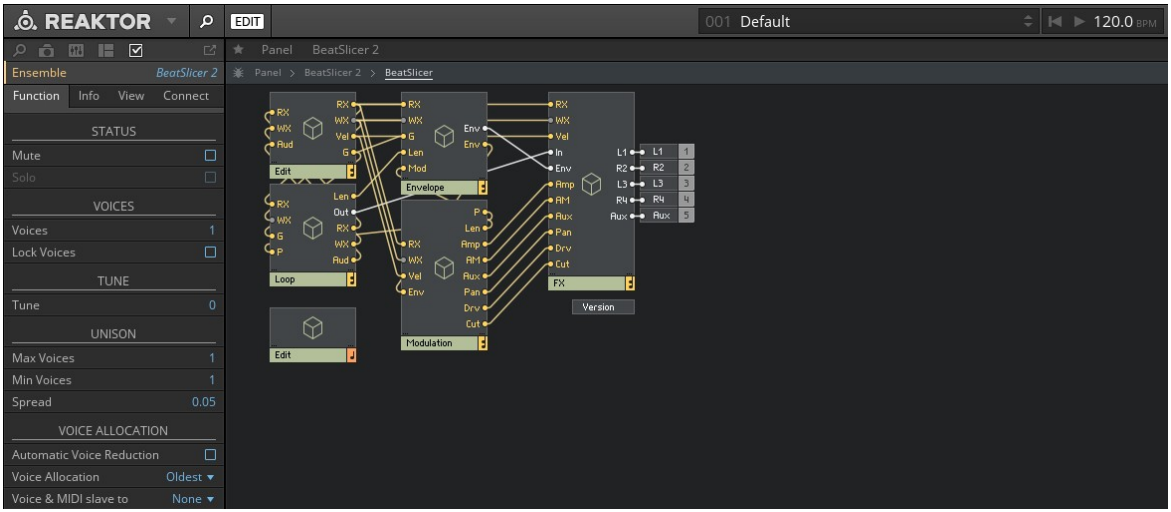
A REAKTOR Rack containing Blocks

Information about Racks can be found in the Getting Started and REAKTOR Blocks Manual documents.

Instruments

An **Instrument** is a container for a synthesizer, sequencer, an effect, or any other musical tool you create, representing the so-called Primary level of REAKTOR. It includes the internal Structure, MIDI processing, Panel (user interface), and Snapshots (presets) associated with your musical tool. In order to connect an Instrument with REAKTOR's audio inputs and out-

puts, it has to be loaded in an Ensemble. You can combine multiple Instruments in one Ensemble. In REAKTOR Blocks, each module is a separate Instrument, allowing you to patch between multiple modules within an Ensemble or a Rack. Instruments are saved as *.ism files.



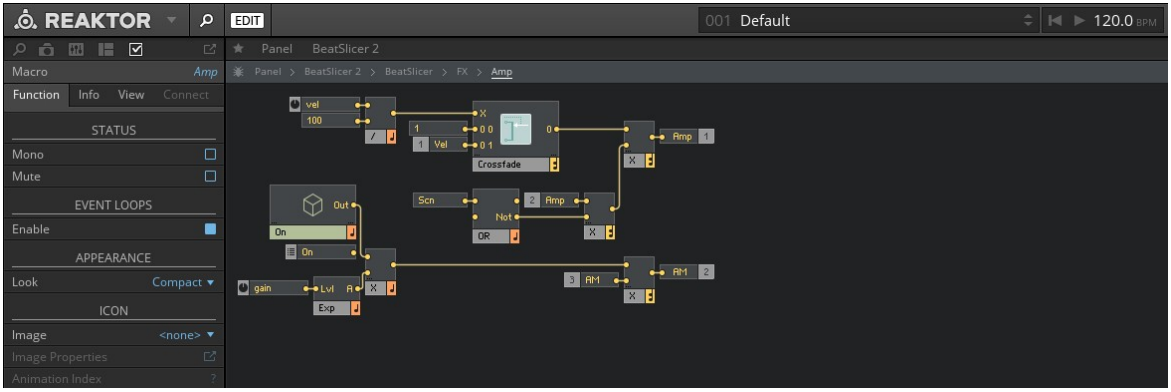
A REAKTOR Instrument containing several Macros.



Information about Instruments can be found in the Getting Started and Diving Deeper documents.

Modules and Macros

Instruments are created using **Modules**. These are REAKTOR's basic building blocks. Modules can be organized into containers called **Macros**. This is especially useful if your Instrument consists of a complex structure with a large number of Modules. Macros can be saved as *.mdl files.

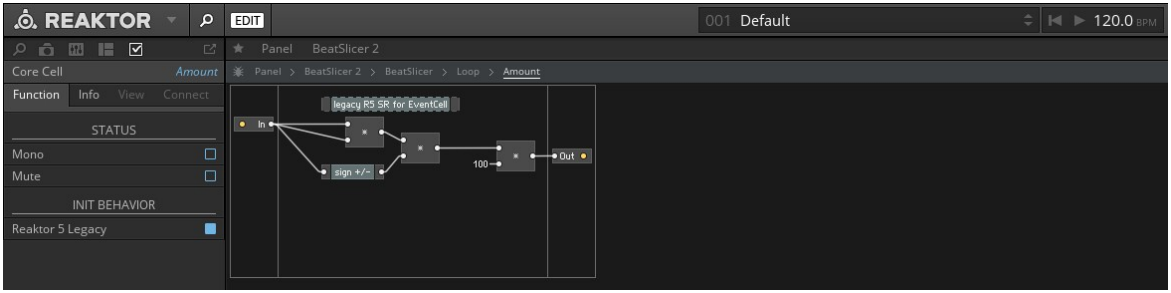


A REAKTOR Macro containing several Modules and another Macro.

Information about Modules and Macros can be found in the Building in Primary document.

Core Cells

Core Cells are a way of creating your own DSP modules using REAKTOR's visual programming language. Core is the most advanced level of programming in REAKTOR. Core Cells can be saved as *.rcc files.



A REAKTOR Core Cell.

Information about Core Cells can be found in the Building in Core document.

2.5 Third-Party and Protected Content

Third-party developers are welcome to develop official content products for REAKTOR. These products are activated using Native Access.

Find information about installing and activating third-party content products for REAKTOR using Native Access in the following article on our website: https://support.native-instruments.com/hc/articles/115004430405

After installing and activating your third-party content product, you will find it in the Player tab of the Browser, sorted by vendor:



The Player tab containing content by Native Instruments and third-party developer Tim Exile

Blocks Content for Use in Racks

Blocks need to be officially licensed for use in REAKTOR Racks. After obtaining the license for an official Blocks content product you can install it via Native Access. After the installation, the included Racks and Blocks are shown in the Browser's Player tab, as well as the Library tab and the search box when a Rack is loaded.

Content Protection

Any official REAKTOR content product from third-party developers and Native Instruments can be protected. This means that the Structure of the Ensemble – or Instrument in the case of a protected module for REAKTOR Blocks – cannot be accessed by the user. As a user, you will notice the following restrictions:

- If you load a protected Ensemble into REAKTOR's Full mode, you will not be able to enable Edit Mode. The EDIT button will be greyed out, as if you were loading an Ensemble into REAKTOR's Player mode.
- If you add a protected Instrument (module for REAKTOR Blocks) to an Ensemble, you will not be able to access the Structure of the added Instrument.
- If you import a protected Ensemble into another Ensemble, you can still enable Edit Mode, however you will not be able to access the Structure of the imported Ensemble.



If you add a protected Instrument or import a protected Ensemble, the Properties for the Instrument or Ensemble are still available. For example, this allows you to setup MIDI and OSC control.

Samples that are embedded into a content product can also be protected. In this case, you will notice the following restrictions:

- Embedded samples are marked as [Copy Protected Content] in the Sample Map Editor, showing a lock icon in the Embed column of the Sample List.
- Certain functions are disabled in the context menu and the corresponding buttons are greyed out.

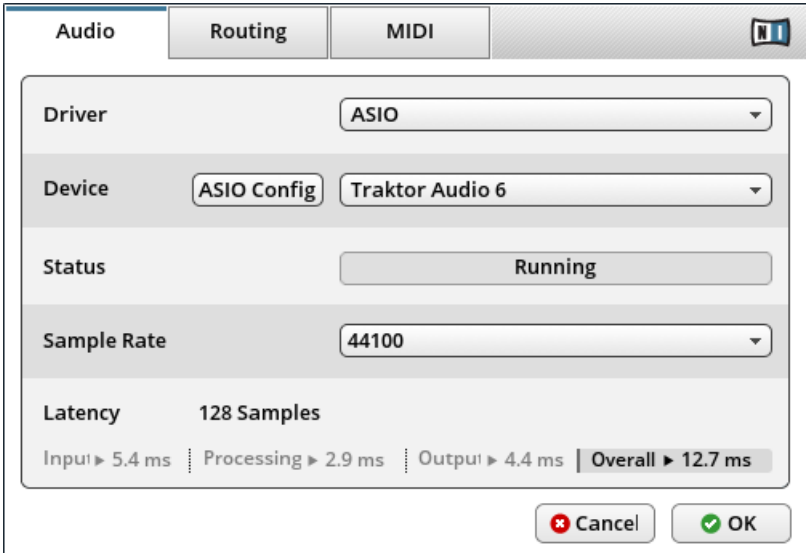


If you are a developer and want to contact Native Instruments about your content products, please use the email address provided on the following website: https://www.native-instruments.com/en/specials/komplete/developer/

3 Setting Up REAKTOR

In order to be able to send and receive audio as well as interact with the REAKTOR stand-alone application, you need to configure your audio and MIDI hardware devices (i.e. audio interface, MIDI keyboard) in REAKTOR's Audio and MIDI Settings. When using the REAKTOR plug-in, these settings are taken over by the host software. In this case, REAKTOR automatically connects with the host software in order to send and receive audio and MIDI.

When you first open the REAKTOR stand-alone application, the Audio and MIDI Settings window appears, prompting you to make the required settings:



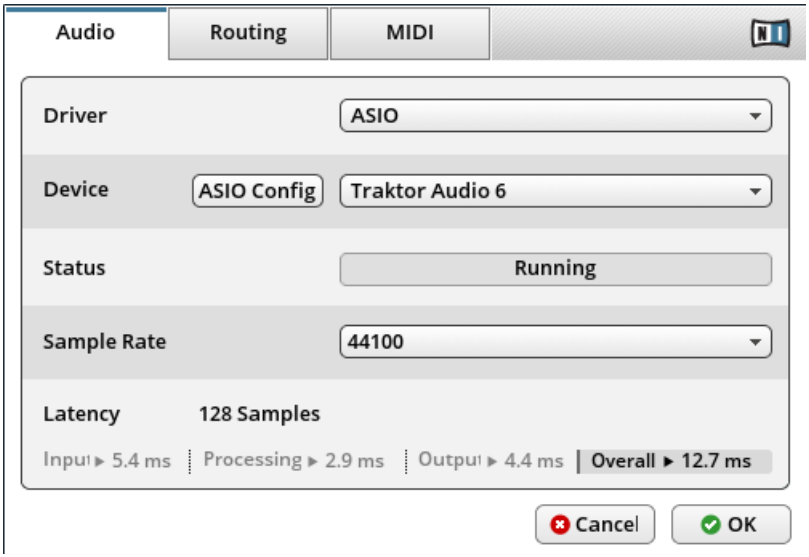
The Audio and MIDI Settings window

3.1 Audio Settings

The audio settings are located in the Audio and MIDI Settings window.

1. To access the audio settings, open the File menu and select Audio and MIDI Settings...

The first tab in the Audio and MIDI Settings window is the Audio tab, which contains the main audio preferences. This is where you select your audio interface and adjust its settings for use with REAKTOR.



The Audio tab in the Audio and MIDI Settings window

The options presented in this tab are ordered in such a way as to lead you through the audio set up process. Follow the steps below to set up your audio interface:

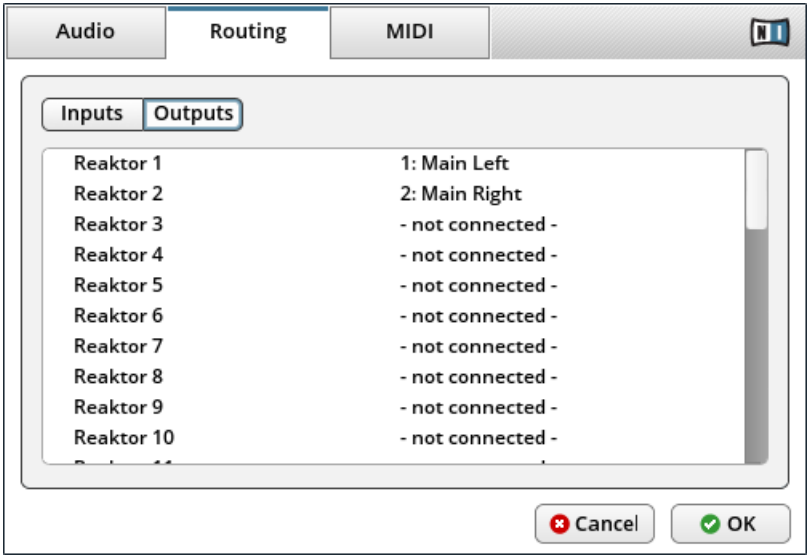
1. Select the type of driver you want to use from the Driver drop-down menu. On computers running Windows, using ASIO is highly recommended. On computers running OS X, Core Audio is pre-selected.
2. Select your audio interface from the Device drop-down menu. The devices available in this menu depend on your Driver type selection.
3. Check the Status of your audio interface. If the hardware is connected and working, this area shows the message Running.
4. Set the Sample Rate for your audio interface. Sample rates of 44100 and 48000 Hz are commonly used and recommended. If you are using an ASIO driver on a Windows system, click on the ASIO Config button to access the ASIO settings for your audio interface and set the sample rate.

5. Set the Latency for your audio interface. Latency refers to the period of time the computer needs to process an audio signal. It is determined by the size of the audio buffer (in samples). Buffer sizes of 128 and 256 samples are commonly used and recommend.

If the computer is not able to provide the amount of processing power needed to complete all audio calculations in time, crackles and drop-outs in the audio playback might occur. In this case, gradually increase the size of the audio buffer until the playback is clean.

Setting up the Audio Routing

After you have completed setting up your audio interface in the Audio tab, you can then choose which audio inputs and outputs REAKTOR should use in the Routing tab. This is where you connect REAKTOR's inputs and outputs to the inputs and outputs of your audio interface.



The Routing tab in the Audio and MIDI Settings window

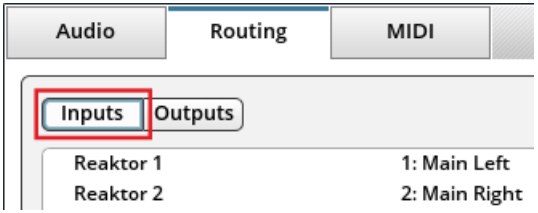
The Routing tab has two pages:

- Inputs: On this page you can manage the connection options for REAKTOR's inputs.
- Outputs: On this page you can manage the connection options for REAKTOR's outputs.

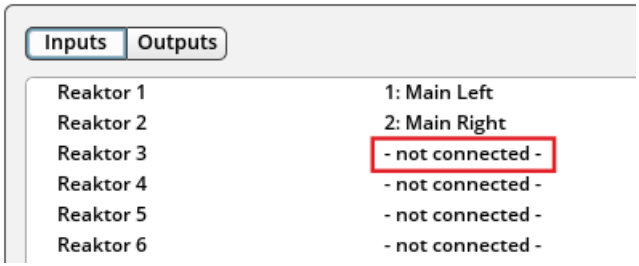
REAKTOR can have multiple inputs and outputs. Most of the time you will only need to use two of each (for the left and right stereo channels), but in some cases more outputs are required. For example, an Ensemble hosting multiple Instruments can benefit from individual outputs for each of them.

To set the audio inputs:

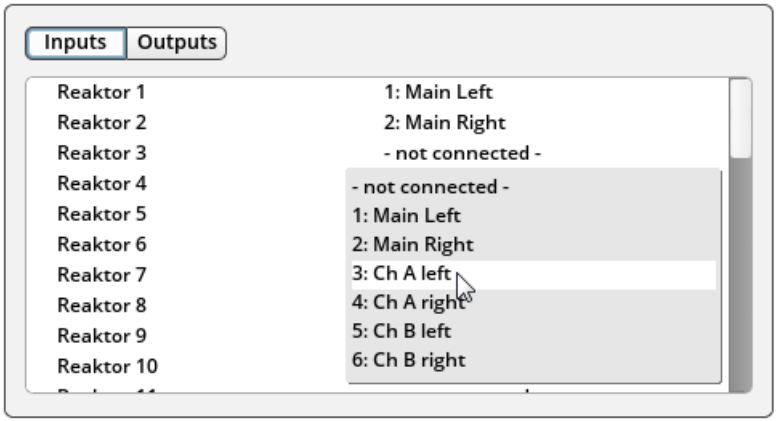
1. Click on the Inputs tab.



2. Click on the area labeled - not connected - beside the REAKTOR input you wish to define.

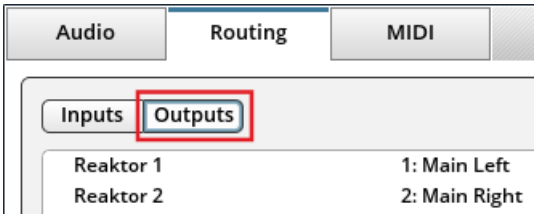


3. Select the input you wish to use from the menu.

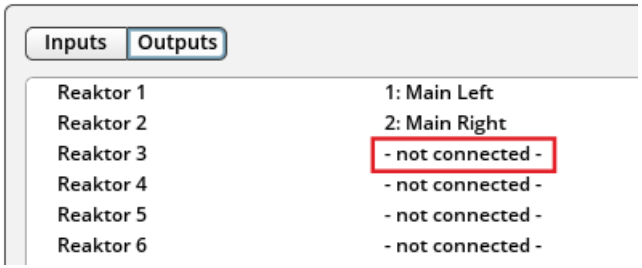


To set the audio outputs:

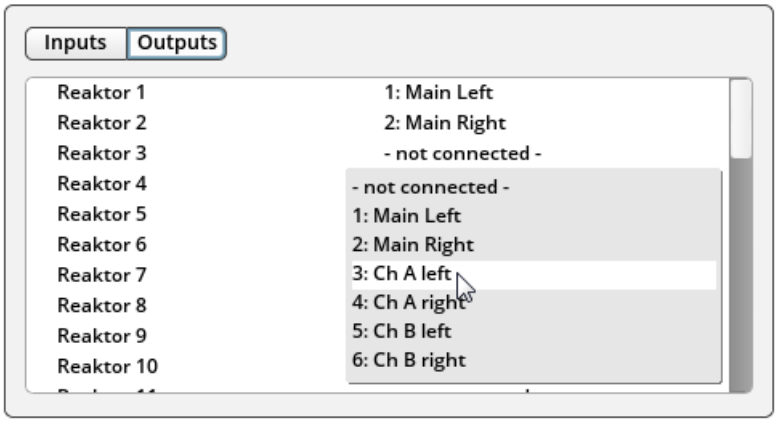
1. Click on the Outputs tab.



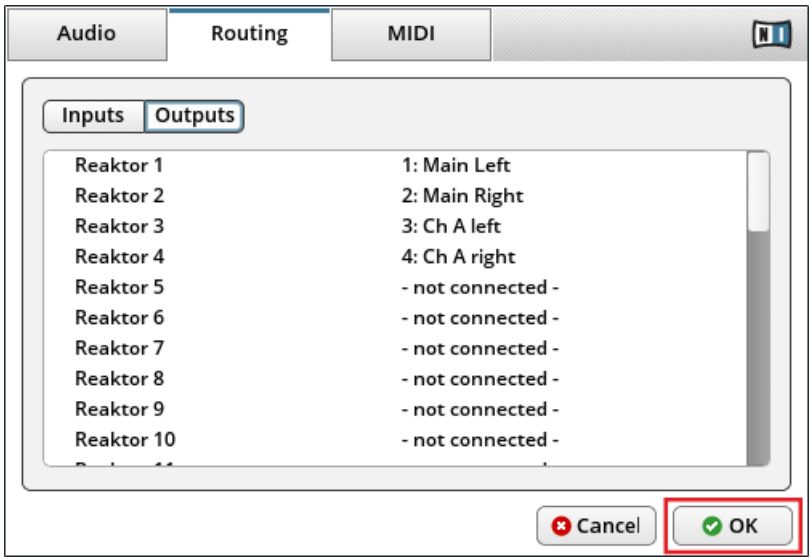
2. Click on the area labeled - not connected - beside the REAKTOR output you wish to define.



3. Select the output you wish to use from the menu.



1. Once you have completed the Audio settings as well as the Inputs and Outputs selection process, click the OK button to confirm your settings.

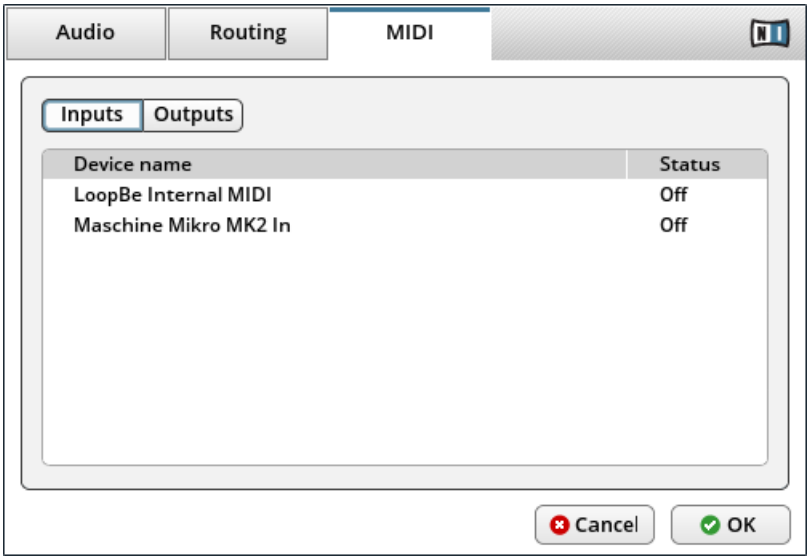


3.2 MIDI Settings

The MIDI settings are located in the Audio and MIDI Settings window. To access the MIDI settings:

1. Open the File menu and select Audio and MIDI Settings...
2. Click on the MIDI tab.

The MIDI tab contains the main MIDI preferences. This is where you enable your MIDI devices for use with REAKTOR, for example MIDI keyboards, USB MIDI controllers, or MIDI interfaces.



The MIDI tab in the Audio and MIDI Settings window

The MIDI tab has two pages:

- Inputs: On this page you can select the devices which REAKTOR receives MIDI data from.
- Outputs On this page you can select the devices REAKTOR sends MIDI data to.

Both of these pages show similar looking tables with the same columns:

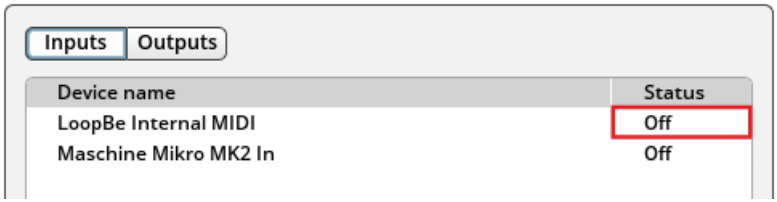
- Device name: This column shows a list of the MIDI devices connected to your computer.
- Status: This column shows the current On/Off state of the device.

To enable MIDI data being received from a MIDI device (for example a MIDI keyboard):

1. Click on the Inputs tab.



2. Click on the Status of the device you wish to use.



3. Select On from the menu.



⇨ REAKTOR now receives MIDI data from the MIDI device.

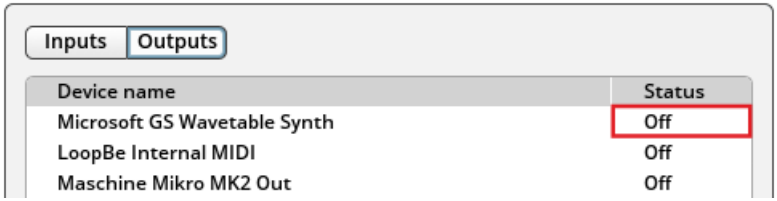
You can test if the connection is working by sending MIDI data to REAKTOR (e.g. by pressing keys on your keyboard) and checking if the MIDI indicator in the Toolbar is blinking (see section ↑2.2, The Toolbar).

To enable MIDI data being sent to a MIDI device (for example a MIDI-controlled synthesizer):

1. Click on the Outputs tab.



2. Click on the Status of the device you wish to use.



3. Select On from the menu.



⇨ REAKTOR now sends MIDI data to the MIDI device.

4 REAKTOR as a Plug-in



REAKTOR as a plug-in

A common use case for REAKTOR is loading it as a plug-in in a host sequencer. When using the REAKTOR plug-in in a host sequencer project, you typically make changes to the Ensemble by tweaking its knobs and other Panel elements. These changes are saved as part of your host sequencer project.

The same applies to using REAKTOR Racks for patching with Blocks. All settings including the arrangement of Blocks and the connections made between them are saved and recalled with your host sequencer project.

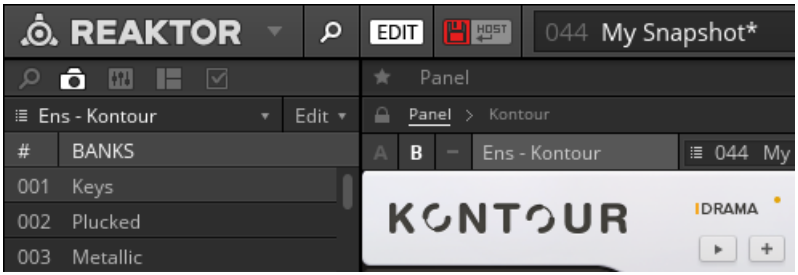
When working with an Ensemble, the REAKTOR plug-in also stores a link to the loaded Ensemble file. When you re-load a host project, REAKTOR looks for the Ensemble file, loads it, and then applies any control changes you may have made.

By default, the REAKTOR plug-in opens in Play Mode, meaning that the Ensemble Structure and the Ensemble's advanced settings are locked. This protects any loaded Ensemble, allowing you to use it as an instrument or effect without needing to worry about accidentally altering the Ensemble and losing the changes you may have made.

You can still enter Edit Mode in the REAKTOR plug-in and make changes to the Structure or the Ensemble's advanced settings, but this means you will need to save and manage your Ensemble file in addition to your host sequencer project (see section ↑4.1, Editing Ensembles and Automatic Saving).

4.1 Editing Ensembles and Automatic Saving

When you are using the REAKTOR plug-in in Edit Mode and the Ensemble is modified in such way that the changes are not saved in the host sequencer's project, the Save Ensemble button in the Toolbar turns red, indicating that these changes will be lost when closing the project:



The red Save Ensemble Button

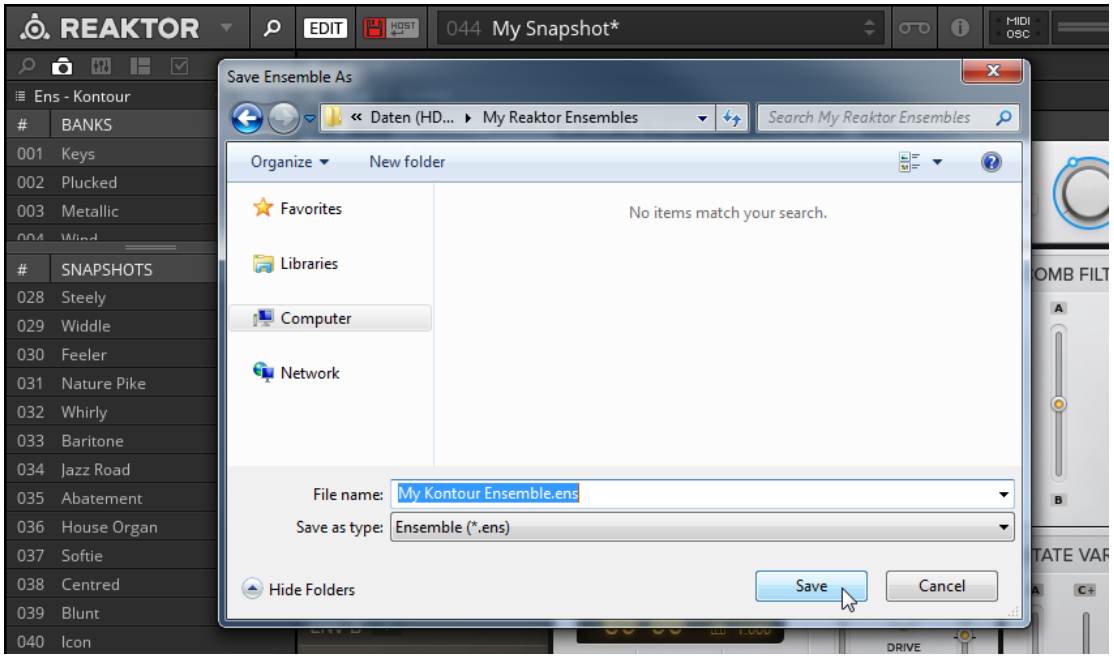
To preserve the changes you have made, you need to create a local copy of the Ensemble. A link to this local copy is saved in the host sequencer's project and the Ensemble will be automatically loaded the next time you open the project.

To save a local copy of the Ensemble:

1. Click on the Save Ensemble button in the Toolbar.



2. In the file dialog, choose the desired location and name for the local copy of your Ensemble and click Save.



⇨ A local copy of your Ensemble is saved and the Save Ensemble button turns from red to grey, indicating that all changes have been saved.



The next time the Save Ensemble button turns red to indicate that changes will be lost when closing the project, you can simply click on it to save the changes to your previously created local copy.

Enabling Automatic Saving

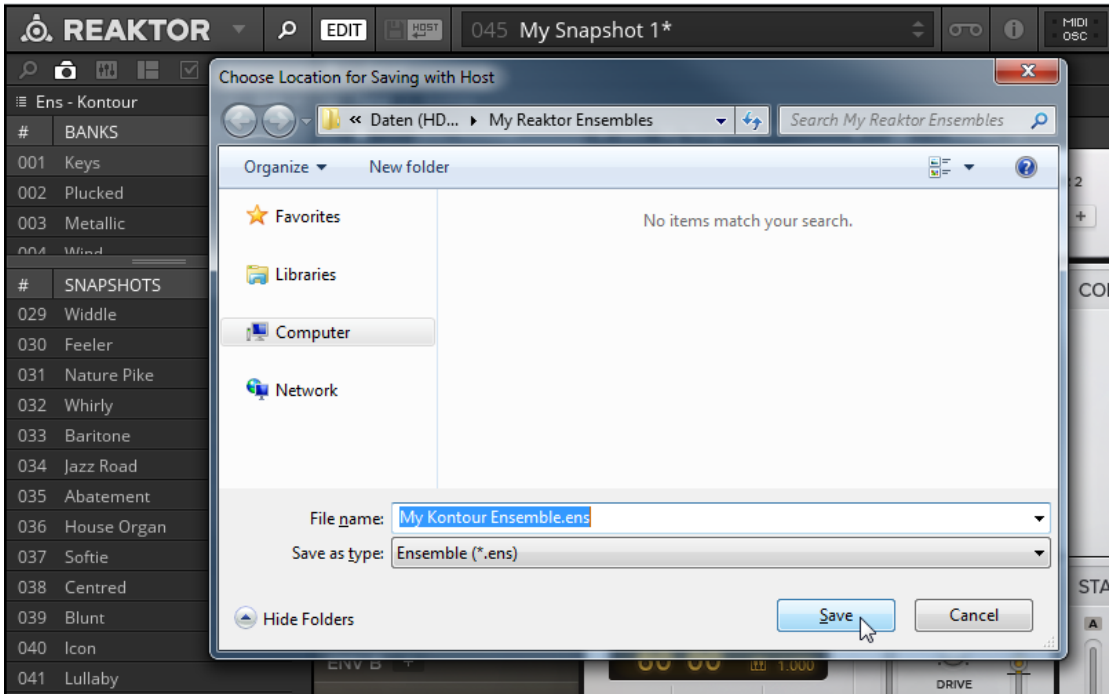
You can also enable automatic saving of the Ensemble. Again, a local copy of the Ensemble is created, however it is saved automatically every time you save the host sequencer's project. This way, you do not have to keep track of the save status of the Ensemble as long as your host sequencer's project is saved correctly.

To enable automatic saving of the Ensemble:

1. Click on the Enable Automatic Saving with Host button in the Toolbar.



2. In the file dialog, choose the desired location and name for the local copy of your Ensemble and click Save.



⇨ The Enable Automatic Saving with Host button lights up, indicating that automatic saving of your Ensemble is now enabled. The local copy of the Ensemble is saved every time you save your host sequencer's project, and will be automatically loaded the next time you open the project.

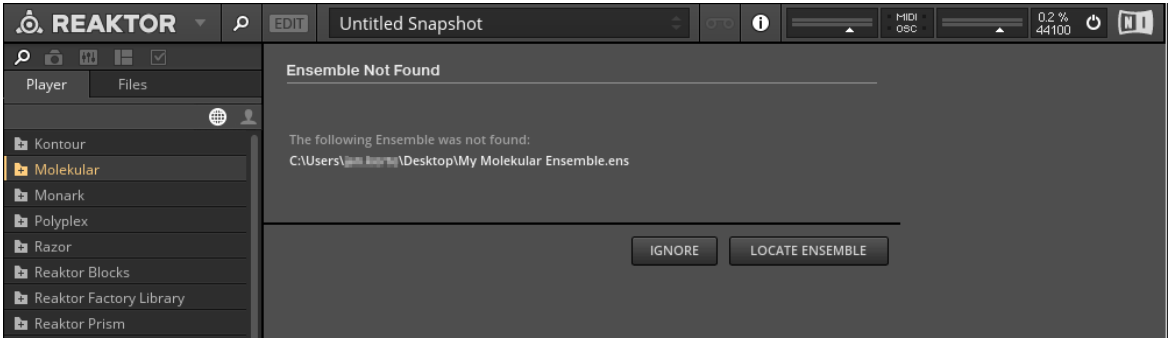


You can disable automatic saving by clicking on the lit Enable Automatic Saving with Host button. This resets the automatic saving, and you can click on the button again to enable automatic saving with a new local copy of your Ensemble.

4.2 Resolving a Broken Ensemble Link

Since the REAKTOR plug-in loads the Ensemble that linked in your host sequencer project from an external file, it is possible for the link to the Ensemble to become broken, for example if you rename the folder where the Ensemble is located.

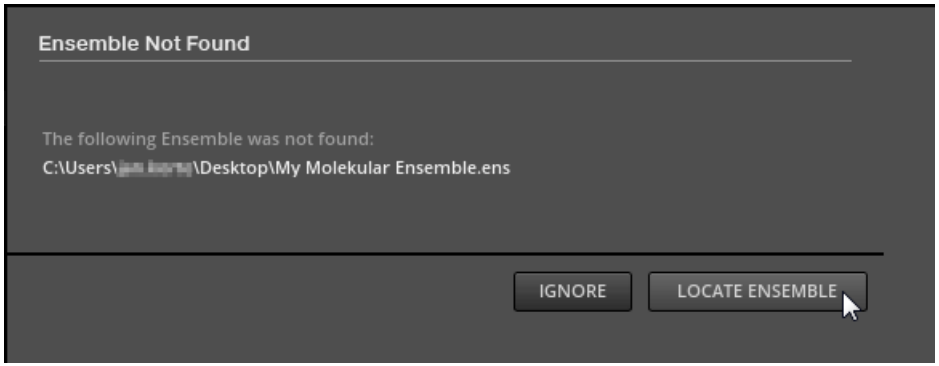
If this happens, REAKTOR reports that it cannot find the linked Ensemble and asks you locate it:



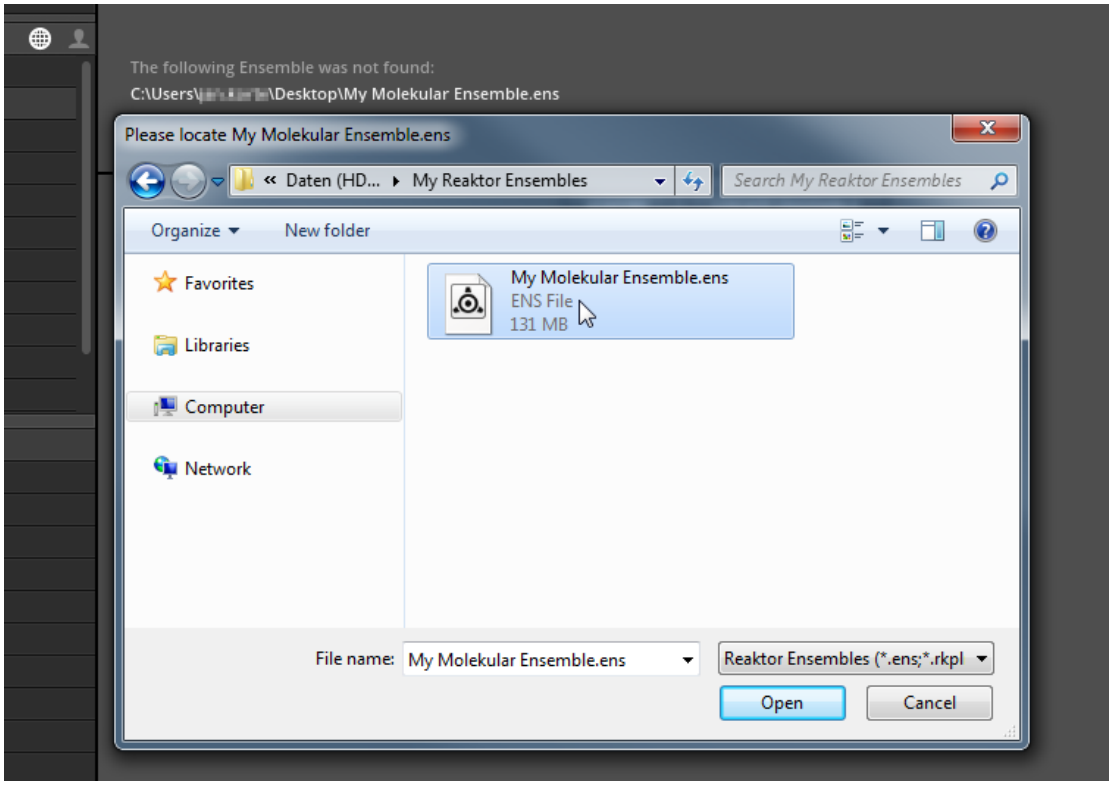
REAKTOR Ensemble not found

To locate the Ensemble file and re-establish the link, follow the instructions below:

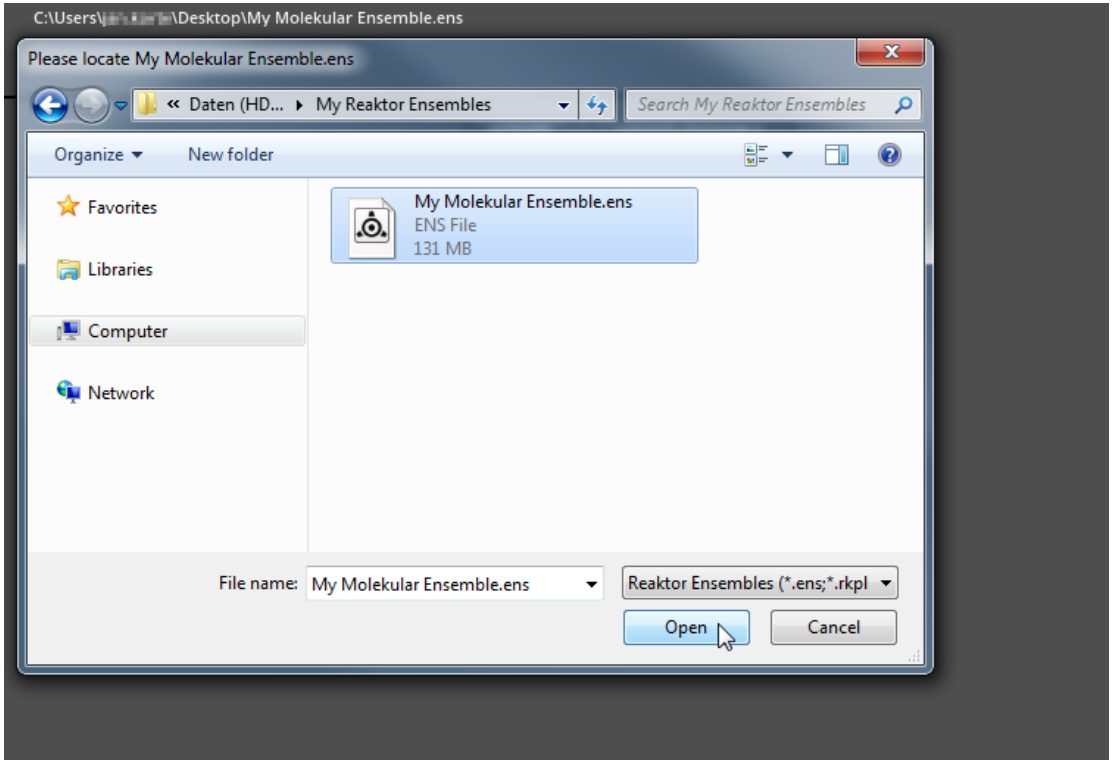
1. Click the LOCATE ENSEMBLE button.



2. In the file dialog, navigate to your Ensemble file and select it.



3. Click open to confirm your selection.



⇨ The Ensemble is loaded, and the re-established link can be saved in your host sequencer project.



Alternatively, you can click the IGNORE button to permanently break the link to the Ensemble file and start using REAKTOR from scratch.

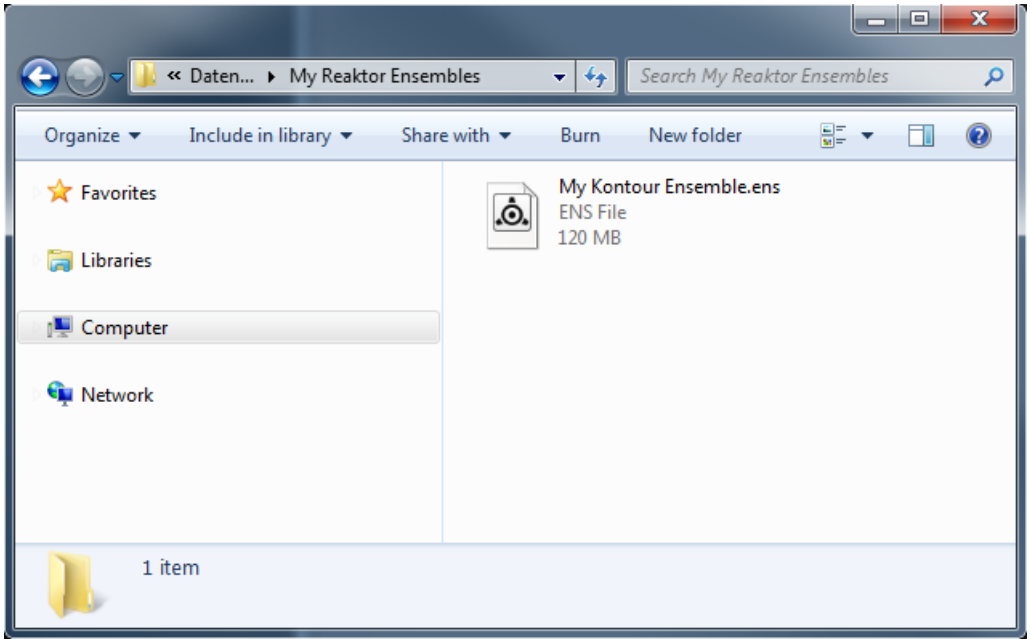
5 Browsing and Loading Ensembles

Ensembles are the main file type used for final REAKTOR instruments and effects. There are a number of ways to browse for and load Ensemble (*.ens) files.

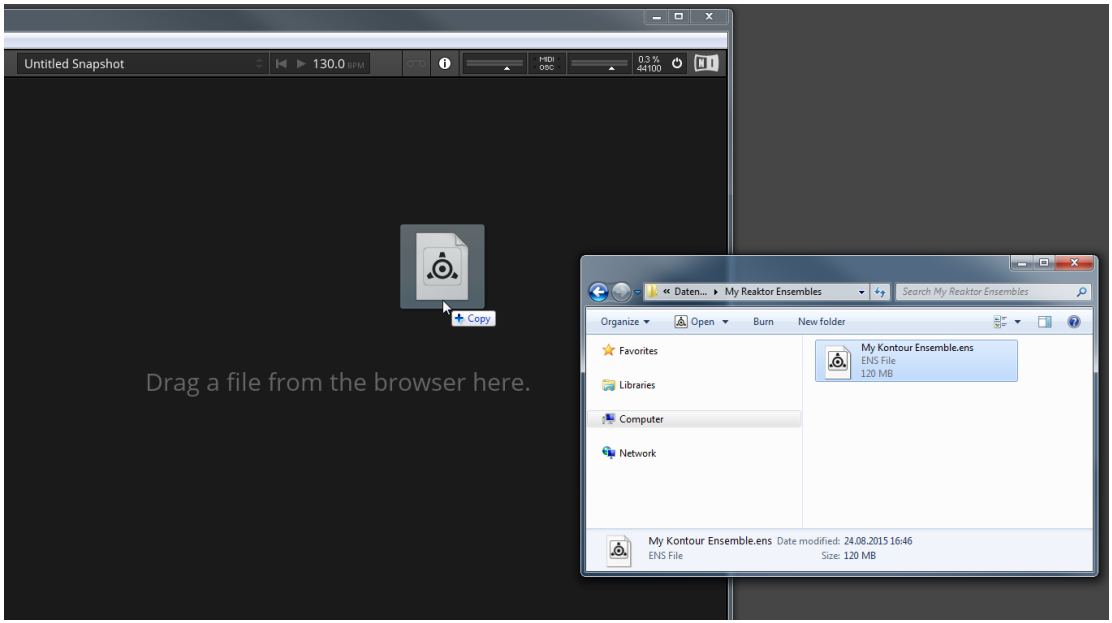
Drag and Drop

The most basic way of loading Ensemble files is via drag and drop:

1. Using your operating system's file browser, locate the Ensemble file you wish to load.



2. Click on the Ensemble file and drag it into the empty main area in REAKTOR's interface. If you have already loaded an Ensemble, you have to drag the Ensemble file into the Panel of the existing Ensemble.



3. Release the mouse button.

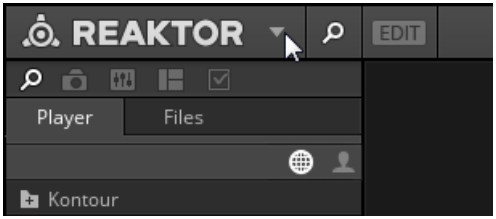
⇨ REAKTOR will load the Ensemble file.

If you drag an Ensemble file into the Structure of an existing Ensemble, it will be added to the existing Ensemble as an Instrument.

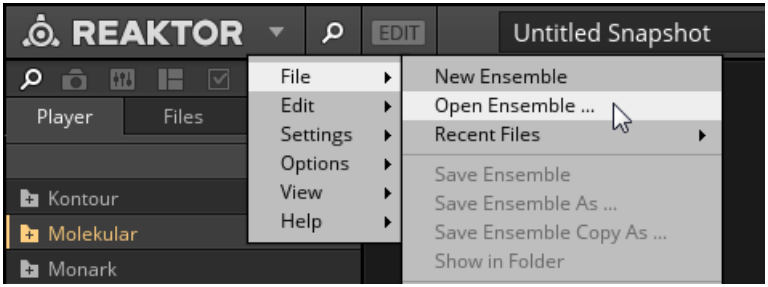
File Menu

You can also open Ensembles via the main menu:

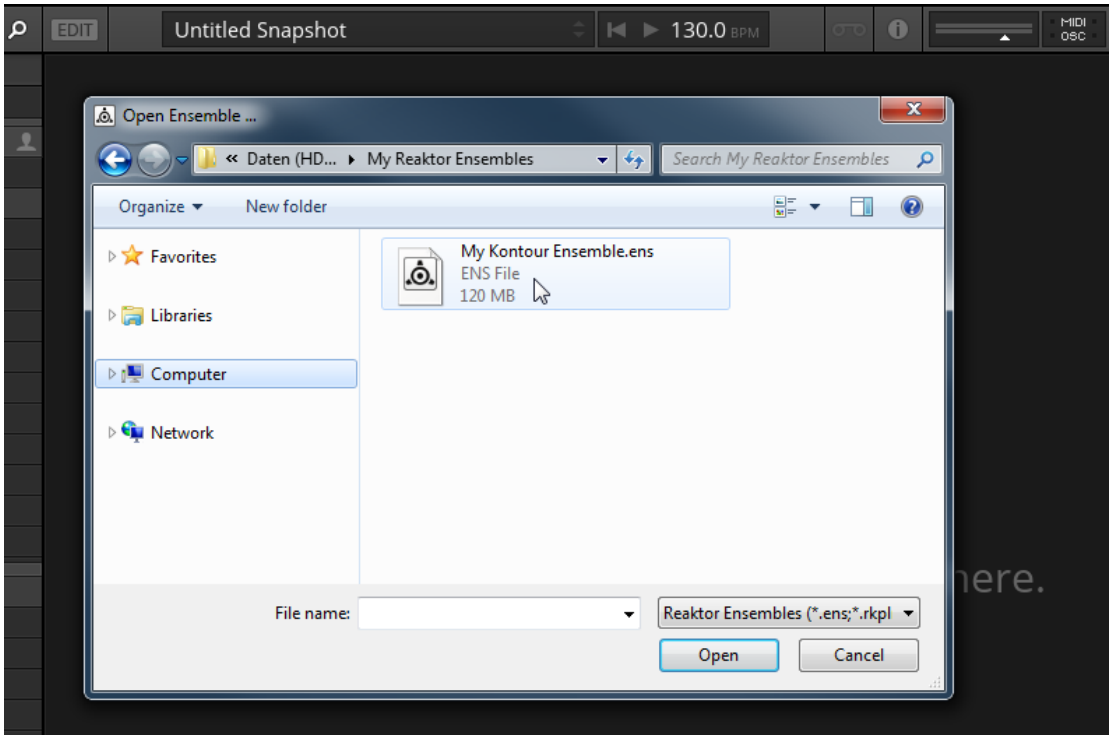
1. Click on the arrow next to the REAKTOR to open the main menu.



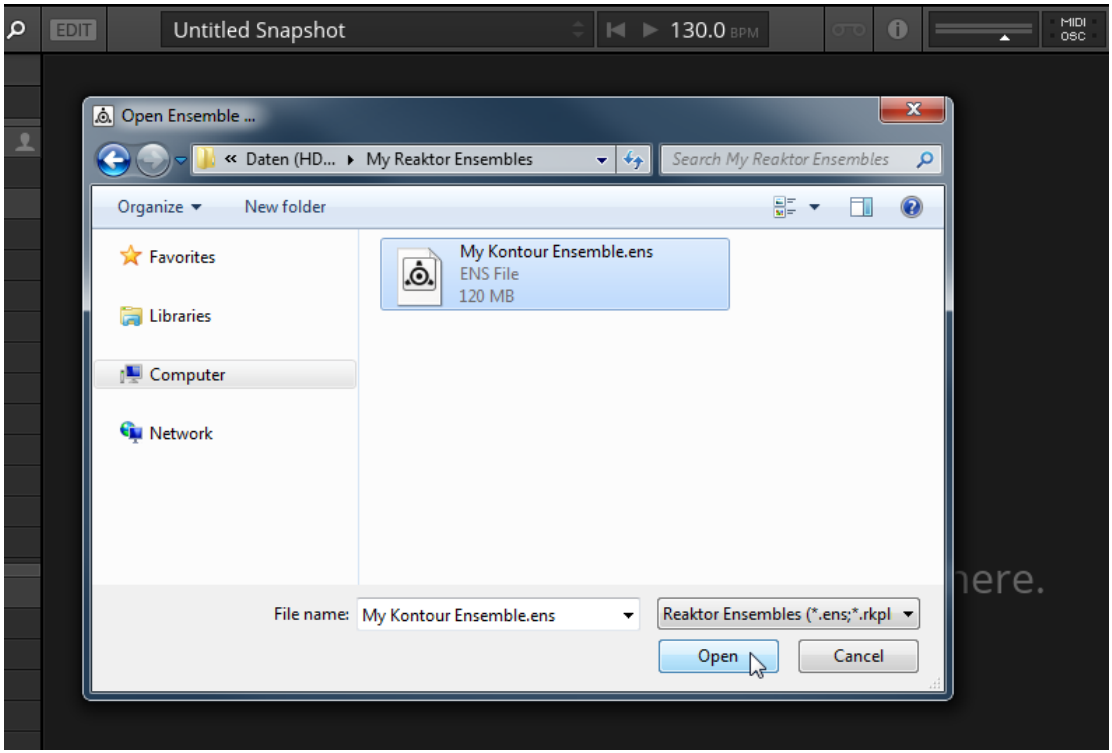
2. Go to the File menu and select Open Ensemble...



3. In the file dialog, navigate to your Ensemble file and select it.



4. Click open to confirm your selection.



⇨ REAKTOR will load the selected Ensemble.

6 Using Ensembles and Instruments

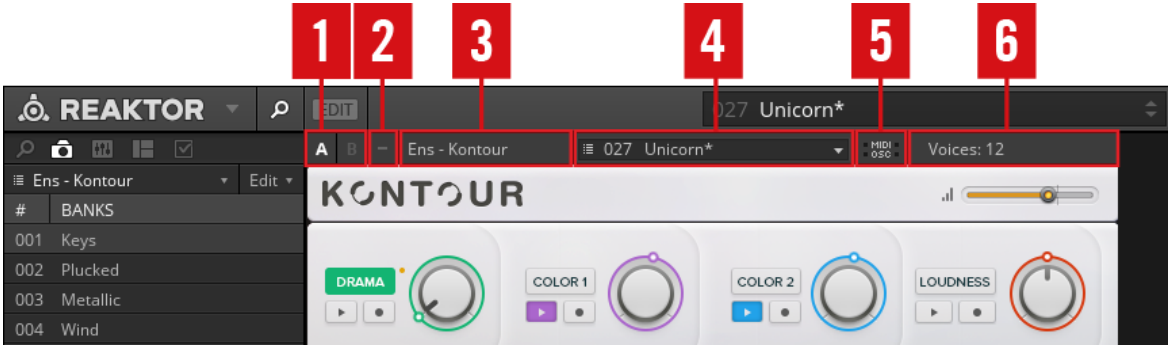
An Instrument is a container for a synthesizer, sequencer, or an effect in REAKTOR. In order to connect an Instrument with REAKTOR's audio inputs and outputs, it has to be loaded in an Ensemble. An Ensemble holds all relevant information of a REAKTOR session. It represents the basic project file you are working with in REAKTOR. An Ensemble can contain a single instrument or a full suite of sequencers, synthesizers, and effects. If you open an Instrument from the Browser's Player tab, it automatically loads in an Ensemble and can be used immediately.

See also

- A/B View [→ 52]
- Snapshots and Presets [→ 53]

6.1 Instrument Header

The Instrument Header is located at the top of the panel of any Instrument you load into REAKTOR.



The Instrument Header

(**1**) A/B **View**: Selects A or B view for the Instrument. See section ↑6.2, A/B View ↑6.2, A/B View for more information on these controls.

(**2**) **Minimize**: Hides the Instrument Panel to show only the Instrument Header. Clicking on this button again will reveal the Instrument Panel.

(**3**) **Instrument Name**: Displays the name of the Instrument.

(**4**) **Snapshot Menu**: A menu for loading Instrument Snapshots. See section ↑6.3, Snapshots and Presets ↑6.3, Snapshots and Presets for more information on Snapshots.

(**5**) MIDI/OSC **Indicators**: A set of 4 activity indicators, which light up whenever the Instrument sends or receives MIDI and OSC messages.

(**6**) **Polyphony Info**: Shows the number of voices used by the Instrument. Generally this equates to the number of notes that the Instrument can produce at once, but this may not always be the case.



The Instrument Header can contain any combination of the described controls and displays, or even not be shown at all for specific Instruments.

6.2 A/B View

REAKTOR Instruments can have two different Panel layouts, which are called the A view and the B view. Not all Instruments use this feature.

Generally the A view is the main view, featuring the main controls needed to use the instrument or effect.

The B view can be any number of things. MONARK for example uses the B view for the settings, MOLEKULAR uses the B view for a simplified interface, and KONTOUR uses the B view to display the full set of controls.

1. You can select to view the A or B views by clicking on the A and B buttons in the Instrument Header.



6.3 Snapshots and Presets

Snapshots and Presets, which are REAKTOR's sound preset formats used with Ensembles and Instruments, enable you to store and recall the state of Panel controls. When you recall a Snapshot, all the Panel controls are restored to the state they were in when the Snapshot was originally created.

Most pre-built Ensembles include a number of Snapshots for the included Instruments, ready to be used in your music. Each REAKTOR Instrument can store 16 Snapshot Banks containing 128 Snapshots each, for a total of 2048 Snapshots.

While both Snapshots and Presets accomplish the same thing on the surface, the details of their use and function differ:

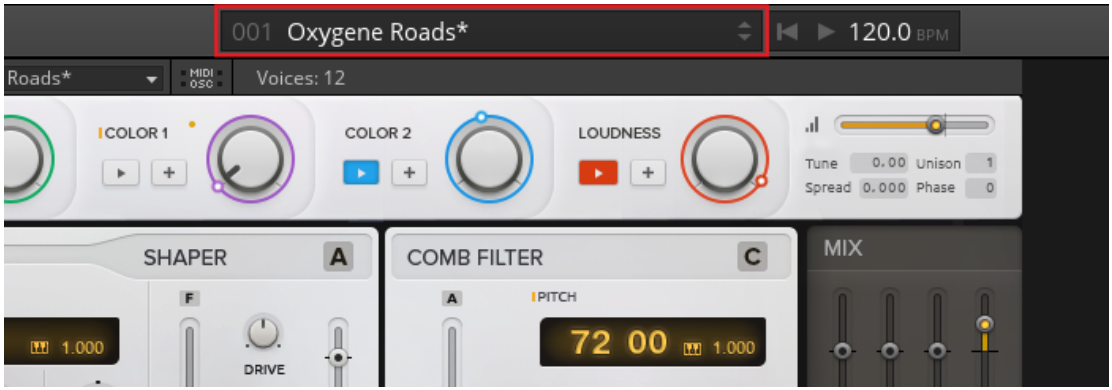
- Snapshots are embedded in the Ensemble file, whereas Presets are external files.
- Snapshots can be recalled using MIDI Program Change messages.
- Presets can be saved when using REAKTOR is Play mode; Snapshots are written into the Ensemble file, and so REAKTOR must be on Edit mode in order to create new Snapshots.
- Presets save the state of the Ensemble, whereas Snapshots can be per Instrument.
- An Ensemble can contain only a limited number of Snapshots (2048).

For more information about Snapshots in REAKTOR, refer to the Diving Deeper document.

6.4 Recalling Snapshots

You can recall Snapshots from several locations:

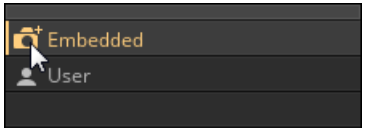
- **The Snapshot menu in REAKTOR's Toolbar**: This method recalls Snapshot settings for all Instruments in an Ensemble.



- **The Instrument Header**: This method recalls Snapshot settings for this particular Instrument only.



- **The Presets tab in the Side Pane**: Clicking on the camera icon labelled Embedded will display the Snapshot Banks contained in an Ensemble/Instrument. This area gives you a better overview over the available Snapshots for the selected Instrument. In Edit Mode, the Snapshot tab allows you to store and edit your own Snapshots (see section ↑6.5, Storing Snapshots).

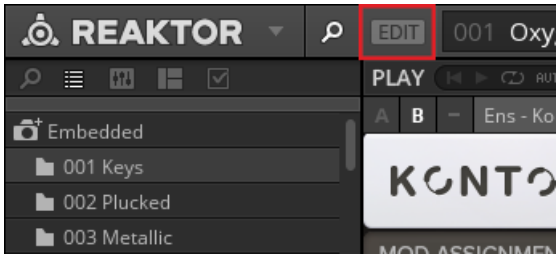


- **Via MIDI**: REAKTOR can accept Program Change messages and use them to select Snapshots. The MIDI Program Change message selects a Snapshot by its number (an integer in the range [0 … 127]). A Program Change value of "0" selects the first Snapshot; a Program Change value of "1" selects the second Snapshot, and so on.

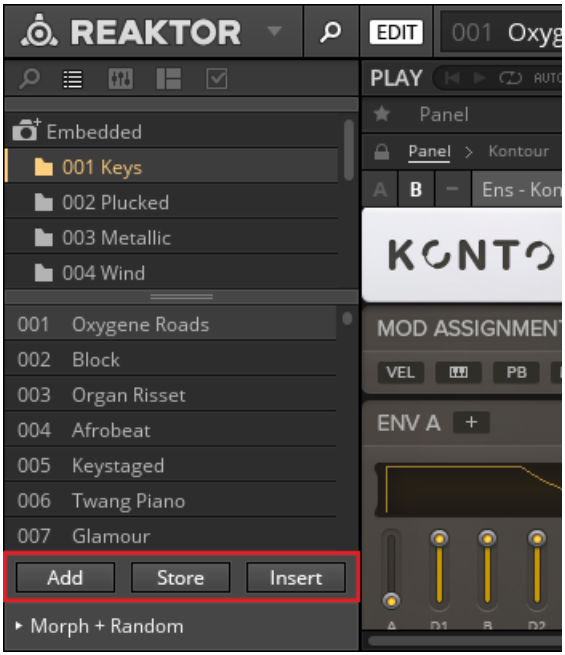
6.5 Storing Snapshots

Storing and editing Snapshots can be done from the Snapshots tab in the Side Pane.

1. In order to be able to store and edit Snapshots, enable Edit Mode by clicking on the EDIT button in the Toolbar.



When viewing Snapshots in the Presets pane, there are three buttons at the bottom for storing Snapshots:



The buttons to Add, Store, and Insert Snapshots

- Add: Creates a new Snapshot with the current settings of your Instrument at the end of the Snapshot list.
- Store: Overwrites the currently highlighted Snapshot with the current settings of your Instrument.
- Insert: Creates a new Snapshot with the current settings of your Instrument directly after the currently highlighted Snapshot, moving any other Snapshots down one slot in order to make space.

Snapshots are stored in the Ensemble, so you need to save the Ensemble or Instrument in order to make changes to the Snapshots permanent.

6.6 Presets

REAKTOR Preset files are a single Snapshot, saved as an external file. Presets contain a link to their original Ensemble file, which is automatically loaded as soon as the Preset file is opened. If the Ensemble file is already open, then the Preset is loaded like a Snapshot.

Presets can also be saved and loaded when using REAKTOR PLAYER, or the full version of REAKTOR when in Play Mode.

Preset files can be saved and loaded via the Files menu:

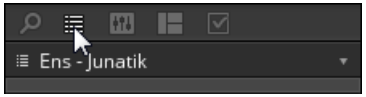
1. To save a preset, go to the main menu and select File > Save Preset As ....This opens a dialog window asking you choose a location and a name for the new Preset file.
2. To load a preset, go to the main menu and select File > Open Preset .... This opens a dialog window asking you choose a Preset file from your hard drive to load.

Presets can also be saved and loaded via the Presets tab of the Side Pane.

Creating Presets and Folders in the Presets Tab

As an example, this section will walk through the process of creating a Preset folder and saving Presets into that folder.

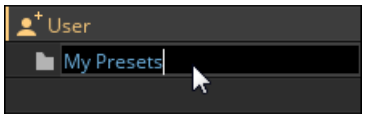
1. Click on the list icon to open the Presets tab in the Side Pane.



2. Right-click on the icon labelled User to open the context menu and select New Folder.



3. When the folder appears, you are able to name it. Type in the name **My Presets** and press Enter.



⇨ The My Presets folder will appear in the Presets folder list.

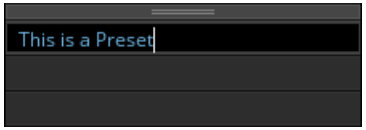
Now that you have a Presets folder created, it is time to populate it with Presets.

To create a Preset:

1. Select the folder into which you wish to store the Preset.
2. Click on the Add button at the bottom of the Presets tab.



3. Type a name for the Preset.



4. Press Enter to save the Preset.

⇨ The Preset will be saved and added to the selected Preset folder.

To overwrite a saved Preset:

1. Select a Preset from the Presets list.
2. Click on the Store button at the bottom of the Presets tab.

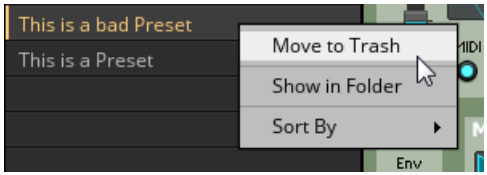
⇨ The selected Preset will be overwritten with the current settings.

Deleting Presets and Folders

To remove a Preset or Folder:

1. Right-click on the item you wish to delete.

2. Select Move to Trash from the menu.



3. A window will pop up asking you to confirm the action; click OK to confirm.

⇨ The Preset or Folder will be deleted.

If you delete a Folder, you will also delete all Presets contained within it.

Also note that this deletion is not permanent, as the Preset files and folders are move to your operating system's recycle bin, and so can be restored or permanently deleted from there.

Managing Presets outside REAKTOR

Since Presets and Preset folders exist as files and folders on your hard disk, they can be managed from there as well. You can rename, sort, and delete Presets without even having REAKTOR open.

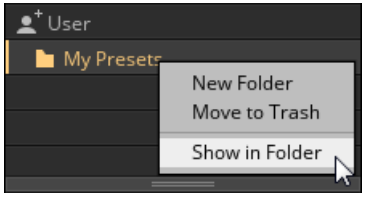
User presets are located in the following directories:

- On Windows: C:\Users\<User>\Documents\Native Instruments\User Content\Reaktor\<Ensemble Name> \
- On OS X: Macintosh HD:\Users\<User>\Documents\Native Instruments\User Content\Reaktor\<Ensemble Name> \

Powered by REAKTOR products (like THE MOUTH or ROUNDS) will have their own folders beside the Reaktor folder, as they are managed as their own products.

You can jump to these locations from within REAKTOR:

1. To view the Preset or folder in your operating system's file browser, right-click on it and select Show in Folder.



⇨ A window will open displaying the location of the folder or Preset on your hard disk.

If you are managing Presets outside REAKTOR, it is advised that you have REAKTOR or the target Ensemble closed while you do so.

7 Using Racks

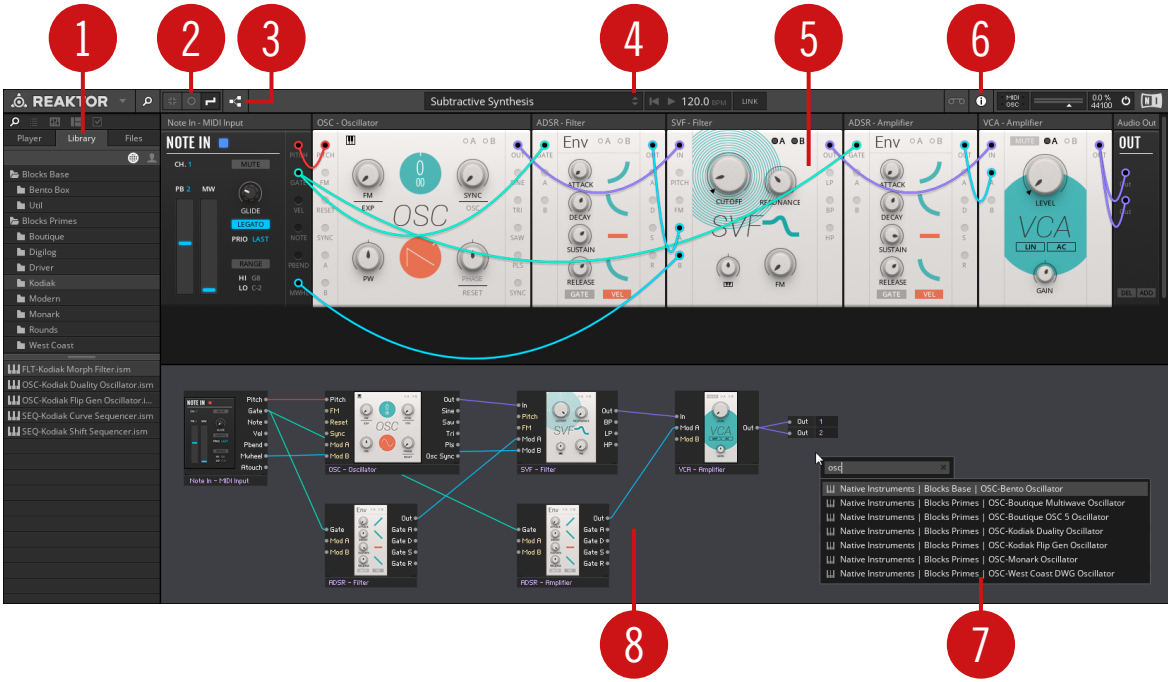
Racks facilitate analog-style modular synthesizer patching using pre-built modules in the Blocks format. In a Rack, you can arrange, connect, and control Blocks directly in the Panel. The patch is also mirrored in the Structure, allowing you to choose your preferred patching workflow.

This chapter provides an overview of the Racks and explains how to manage them. If you want to learn more about Racks and all the available Blocks, refer to the following resources:

- Learn how to patch using REAKTOR Blocks in a Rack by following the Quick Start Guide online: www.native-instruments.com/reaktor-blocks-quickstart
- Learn all about the Blocks and their controls in the application by using REAKTOR's Info Hints. For more information, refer to ↑7.1, Overview.

7.1 Overview

The following overview shows you how a Rack is displayed in REAKTOR and where you can find the most important controls.



Overview of a Rack in REAKTOR

(**1**) **Library**: This tab in the Browser shows all installed Blocks. They are organized in folders. The contents of the selected folder are shown in the lower section of the Browser. You can drag and drop the Blocks from the lower section into the Rack to build your patch.

(**2**) **View modes**: These three buttons control the visibility of ports and wires in the Rack.

- **Compact View**: All ports and wires are hidden. You can focus on tweaking the controls of the Blocks.
- **Ports View**: Only the wires are hidden. You have a clear view of all controls while the ports are still visible, indicating connections in the patch by color coding.
- **Ports and Wires View**: The full patch including ports and wires is visible and can be changed by making new connections or removing existing ones.

(**3**) **Show Structure**: This button shows or hides the additional Structure view (**8**). This is useful for creating and managing large patches. The Structure shows all connections even when Compact View or Ports View are selected.

(**4**) **Rack switching**: The arrow buttons in the Rack menu switch between Racks in the selected category folder of the factory content. For more information, refer to ↑7.2, Loading Racks.

(**5**) **Panel view**: Shows all Blocks in the Rack. You can change the controls on each Block to adjust its parameters. Inputs are located on the left side of a Block, while outputs are located on the right side. You can connect Blocks by clicking on an output and dragging a wire to an input. You can remove a wire by double-clicking on the connected input.

(**6**) **Info Hints**: Enables or disables tooltips for all Blocks and their controls. When enabled, you can hover over the header of a Block to view general information. Hovering over a control shows detail information.

(**7**) **Search box**: Press the Enter key on your computer keyboard to call up the search box. You can find Blocks by entering their function, family, or name. The search box can be used both in Panel and Structure views.

(**8**) **Structure view**: Shows all Blocks in the Rack. Inputs are located on the left side of a Block, while outputs are located on the right side. You can connect Blocks by clicking on an output and dragging a wire to an input. You can click on a wire to select it and press Delete or Backspace on your computer keyboard to delete a wire.

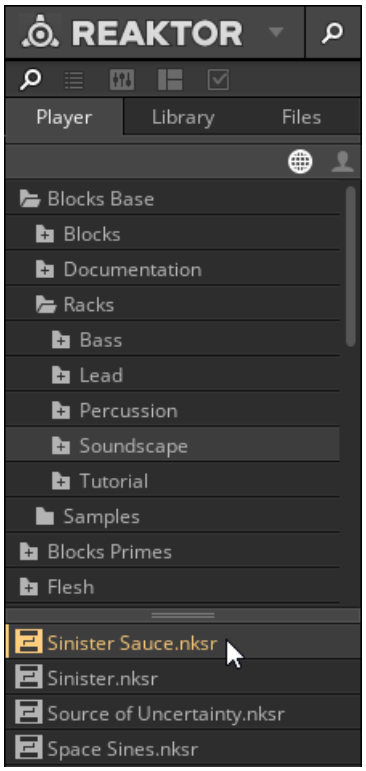
7.2 Loading Racks

Each Blocks product includes premade Racks as factory content. You can load these Racks from the Player tab in the Browser. Alternatively, you can open Racks saved on your hard drive.

Loading Racks from a Blocks Product

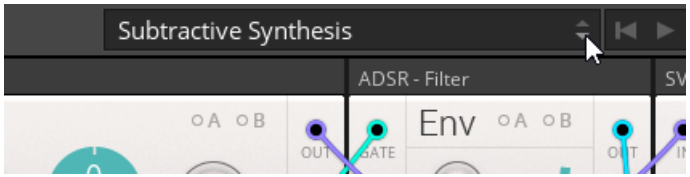
To load a Rack included in a Blocks product:

1. Open the folder of the Blocks product in the Browser's **Player** tab, select a category folder and load a Rack by double-clicking on it in the lower section.



Once you have selected a category folder, you can also switch between Racks using the Rack menu in the toolbar.

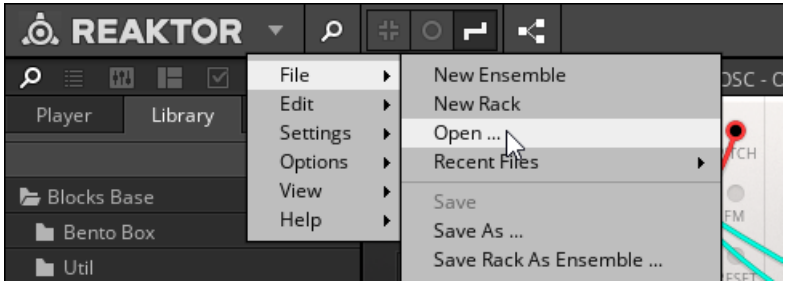
1. Click on the arrows in the Rack menu to switch between Racks in the selected category folder.



Opening a Rack saved on the Hard Drive

To open a Rack saved on the hard drive:

1. Click on the drop-down menu next to the REAKTOR logo, go to the **File** menu and select **Open**.



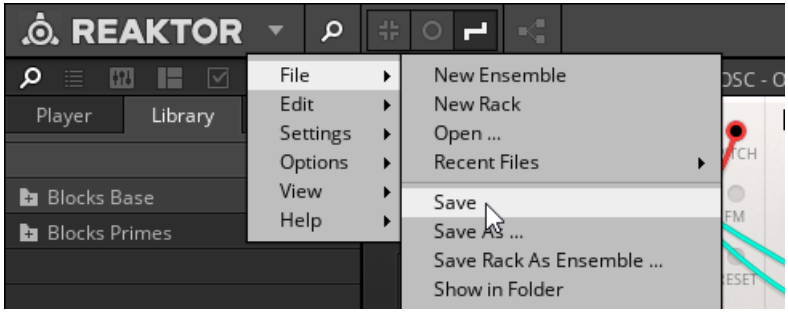
2. In the file dialog, browse to the Rack file you saved, select it, and click **Open**.

⇨ The Rack opens in REAKTOR and you can start using it.

7.3 Saving Racks

When you have created a sound you like, you can save your Rack in a file. This way you can recall it later and share it with others.

1. Click on the drop-down menu next to the REAKTOR logo, go to the **File** menu and select **Save**.



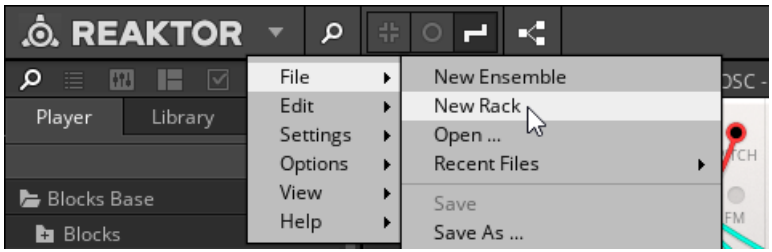
2. In the file dialog, browse to a location of your choice, type in a name for your Rack, and click **Save**.

When using REAKTOR as a plug-in, all settings including the arrangement of Blocks and the connections made between them are also saved and recalled with your host sequencer project (host chunk saving).

7.4 Creating a New Rack

When you want to create a Rack from scratch, you need to open a new Rack.

1. Click on the drop-down menu next to the REAKTOR logo, go to the **File** menu and select **New Rack**.



7.5 Additional Settings

Additional settings for Racks are available in REAKTOR's side pane.

- Panel Sets allow you to save different configurations of the Panel view in the same Rack file. You can show or hide Blocks (while they remain active) or switch between arrangements of Blocks for specific use cases. For more information, refer to the REAKTOR 6 Diving Deeper document, chapter **Panelsets**.
- You can assign any of the controls in the Rack to your MIDI controller. For more information, refer to ↑9, Controlling REAKTOR.
- You can also use OSC to control your Rack, or route MIDI and OSC data from and to individual Blocks in the Rack. For more information, refer to the REAKTOR 6 Diving Deeper document, chapter **Controlling REAKTOR with MIDI and OSC**.

8 Snapshots and Presets

Snapshots and Presets, which are REAKTOR's sound preset formats used with Ensembles and Instruments, enable you to store and recall the state of Panel controls. When you recall a Snapshot, all the Panel controls are restored to the state they were in when the Snapshot was originally created.

Most pre-built Ensembles include a number of Snapshots for the included Instruments, ready to be used in your music. Each REAKTOR Instrument can store 16 Snapshot Banks containing 128 Snapshots each, for a total of 2048 Snapshots.

While both Snapshots and Presets accomplish the same thing on the surface, the details of their use and function differ:

- Snapshots are embedded in the Ensemble file, whereas Presets are external files.
- Snapshots can be recalled using MIDI Program Change messages.
- Presets can be saved when using REAKTOR is Play mode; Snapshots are written into the Ensemble file, and so REAKTOR must be on Edit mode in order to create new Snapshots.
- Presets save the state of the Ensemble, whereas Snapshots can be per Instrument.
- An Ensemble can contain only a limited number of Snapshots (2048).



For more information about Snapshots in REAKTOR, refer to the Diving Deeper document.

9 Controlling REAKTOR

REAKTOR Instruments can be remotely controlled from a wide range of hardware controllers. This enables you to play REAKTOR with a physical interface, be it a MIDI keyboard, a special MIDI controller with knobs and buttons, or a device supporting Open Sound Control.

REAKTOR uses two protocols for sending and receiving control messages:

- **MIDI**: Musical Instrument Digital Interface - a standard communication protocol for musical instruments created in the early 80s that is still commonly used today. Virtually all keyboard controllers use MIDI to communicate with your computer. It is also used by hosts to communicate with the REAKTOR plug-in.
- **OSC**: Open Sound Control - a more modern protocol that offers faster data rates, higher resolutions, and a more flexible architecture compared to MIDI. However, OSC is less common and requires a more advanced setup.

This Getting Started document only describes basic MIDI options. Since the MIDI and OSC functionality is tied together, the following description refers to combined functions in some cases.



For more information on using the OSC protocol with REAKTOR, refer to the Diving Deeper document.

9.1 MIDI Learn

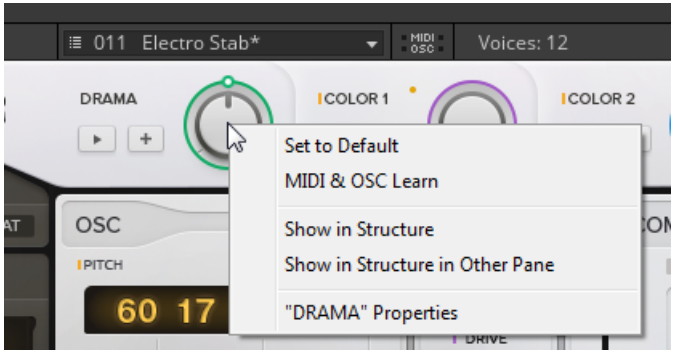
In order to be able to remotely control REAKTOR from your hardware controller, its MIDI controls need to be connected to the Panel controls in REAKTOR.



Your hardware controller has to be enabled in REAKTOR's MIDI settings first. See section ↑3.2, MIDI Settings for more information.

The easiest way to connect a MIDI control to a Panel control is the MIDI learn function. It establishes the connection between the MIDI control and the Panel control in a semi-automatic fashion.

1. Right-click on the Panel control you want to connect the MIDI control to.



2. Select MIDI & OSC Learn to enable the MIDI learn function.



3. Move the MIDI control you want to use.

⇨ The MIDI control is connected to the Panel control, which can now be remotely controlled from your MIDI controller. The MIDI learn function is disabled automatically as soon as the connection is established.

9.2 Connection Manager

The Connection Manager is located in the Side Pane. It allows you to manage the MIDI connections in the currently loaded Ensemble.

1. To view the Connection Manager, click on the Side Pane tab with the controller icon.



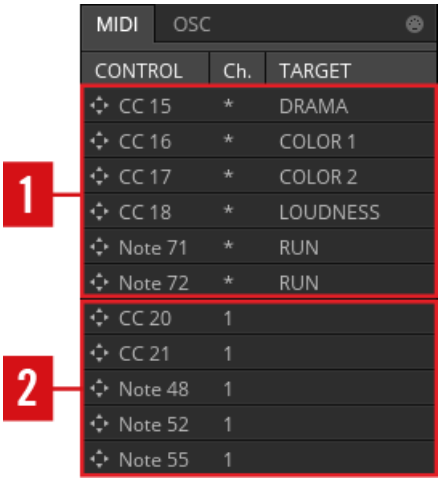
⇨ The Connection Manager appears in the Side Pane.



The Connection Manager has two tabs, one for MIDI and one for OSC.

1. When working with MIDI, select the MIDI tab.

All established connections are shown in the list at the top of the MIDI tab. They appear in the following order:



MIDI messages in the Connection Manager

(**1**) **Current**: The current connections appear at the top of the list. They are identifiable by the information present in the Target column.

(**2**) **Recent**: Recently received messages appear below the current connections (by default, this list is only refreshed each time you open the tab).

Note messages appear below controller messages, both for current connections as well as for recently received messages.

1. To view incoming messages in real-time, activate the Auto button below the list. Not only will this re-populate the list in real-time, but it will also highlight currently used messages.
2. To clear any unconnected controllers from the table, click on the Clean button below the list.

Creating Connections

You can connect MIDI controls to Panel controls from the Connection Manager by drag and drop:

1. Click on the arrow icon next to the MIDI control you wish to use and keep the mouse button pressed.



2. Drag the control over the Panel control you wish to connect it to and release the mouse button.

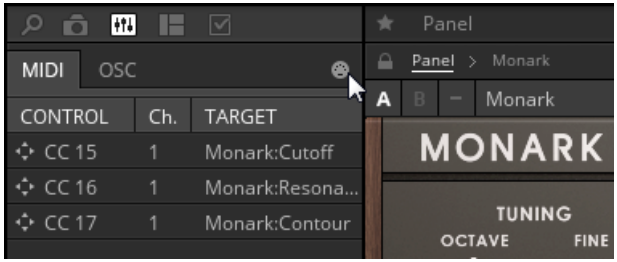


⇨ The MIDI control is connected to the Panel control and appears among the other current connections at the top of the list.



Alternatively, you can activate the MIDI Learn function from the Connection Manager:

1. Click on the Learn button at the top right of the Connection Manager to activate MIDI Learn.

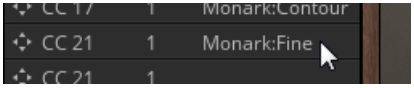


2. Click on the Panel control you want to connect the MIDI control to.



3. Move the MIDI control you want to use.

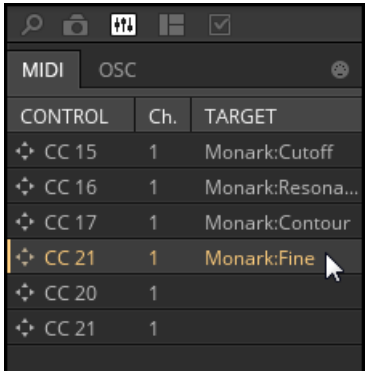
⇨ The MIDI control is connected to the Panel control and appears among the other current connections at the top of the list.



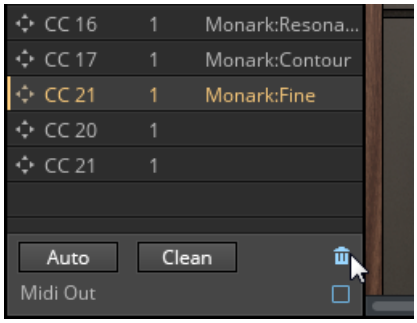
Deleting Connections

You can also use the Connection Manager to remove MIDI connections:

1. Select the connection you wish to remove by clicking on it in the list.



2. Click on the Delete icon below the list or press the [delete] key on your keyboard.



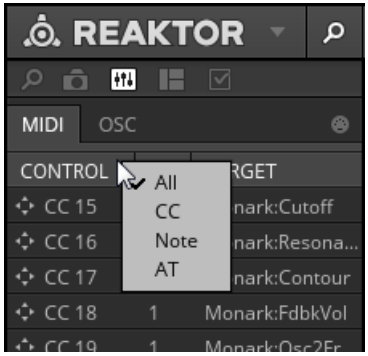
⇨ The selected connection is removed.

Filtering the Connections List by Controller Data Type

You can filter the MIDI connections list by the data type. The available filters are:

- All: All MIDI message types are shown.
- CC: Only MIDI CC (Continuous Controller) data is shown.
- Note: Only MIDI Note data is shown.
- AT: Only MIDI Aftertouch data is shown.

1. To filter the list of controller messages by data type, right-click on the Control column's header and select one of the available data type filters from the context menu.



Filtering the MIDI List by MIDI Channel

You can also filter the MIDI connections list by MIDI channel.

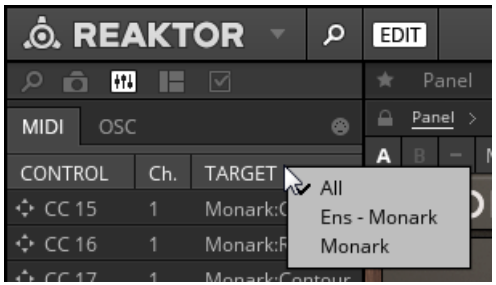
1. To filter the list of controller messages by MIDI channel, right-click on the Ch (Channel) column's header and select the MIDI channel from the context menu.



Filtering the Lists by Ensemble or Instrument

Finally, you can filter the MIDI connections list so that only controller messages assigned to a specific Ensemble or Instrument are shown:

1. To filter the list of controller messages by Ensemble or Instrument, right-click on the Target column's header and select the Ensemble or Instrument from the context menu.

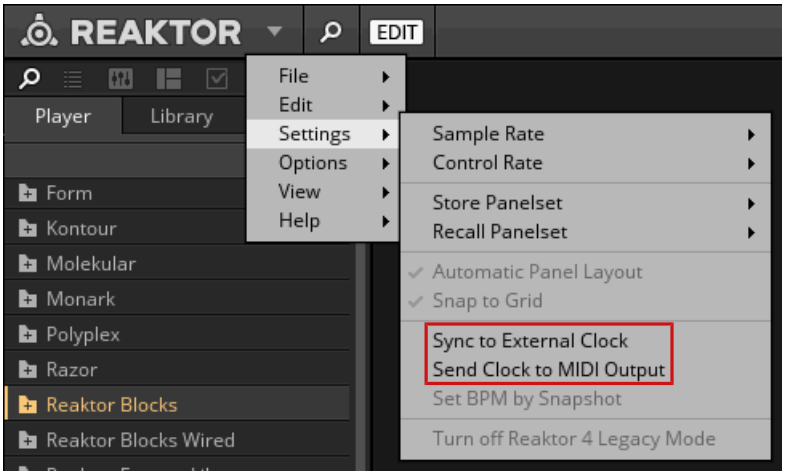


9.3 MIDI Clock

If REAKTOR is running as a stand-alone application, it can synchronize its internal clock to other devices via MIDI clock. This could be hardware such as a drum machines, hardware sequencers, synthesizers, and effects processors, or even software running on another computer.

REAKTOR can also synchronize its internal clock to Ableton Link. For more information, see section ↑9.4, Synchronization using Ableton Link.

1. To access the MIDI clock settings, go the Settings sub-menu in the REAKTOR main menu.



The MIDI clock settings

(**1**) Sync to External Clock: When enabled, all *Sync Clock* and *1/96 Clock* modules synchronize to MIDI clock received on any of the active MIDI inputs (see section ↑3.2, MIDI Settings for information about changing the status of MIDI inputs). In this case, the BPM value of REAKTOR's transport in the toolbar cannot be adjusted manually; instead, the internal clock is automatically adjusted according to the external clock. Sync to External Clock also allows for control of *Start/Stop* modules by external MIDI-Start/Stop messages.

(**2**) MIDI Clock Out: When enabled, REAKTOR's transport sends out MIDI clock ticks on all active MIDI outputs inputs (see section ↑3.2, MIDI Settings for information about changing the status of MIDI outputs). Note that playback needs to be running in REAKTOR, otherwise no MIDI clock will be sent.

9.4 Synchronization using Ableton Link

Ableton Link is a protocol that synchronizes beat, phase and tempo of Link-enabled applications on the same computer or over a shared network. This means you can conveniently keep applications synchronized across different devices or join a group jam with others with minimal setup. Connecting applications via Link provides you with the ability to synchronize them to a shared timeline and change the tempo globally from each of the connected applications.

9.4.1 Connecting to a Network

Applications that support Ableton Link can join a Link session when connected to the same network. To enable Link within REAKTOR, simply make sure the computer running REAKTOR is connected to the same local network as the other applications you want to link to. This can either be a local network or an ad-hoc (computer-to-computer) connection.

To ensure maximum security and reliability while using Link, there are several things you may want to consider:

- For reliability, connect to the local network using a router via an Ethernet cable.
- If you do use a Wi-Fi network, be sure to use a password to prevent excluded users from joining your session.
- For the most secure and reliable connection, create a Link session via a direct connection from one computer to another using a Thunderbolt or Ethernet cable.

For more general information on setting up and using Link, please refer to the Link FAQs article located on the Ableton website:

https://help.ableton.com/hc/en-us/articles/209776125-Link-FAQs.

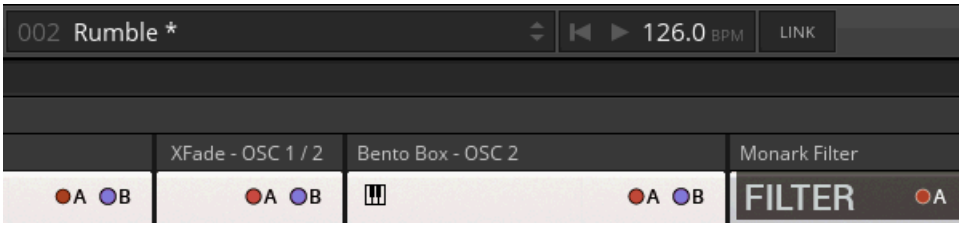
9.4.2 Joining and Leaving a Link Session

It is possible to join a Link session at any time by clicking the LINK button. When another Link-enabled application is connected, a moving bar within the LINK button is displayed even though REAKTOR's transport is not running. This bar represents the global phase of Link that all participating applications lock into.

After clicking the Play button, playback will resume on the downbeat once the moving bar within the LINK button is filled. The first participant to join the session sets the initial tempo, from then on any participant in the session can change the tempo in their respective application. If multiple participants change the tempo at the same time, the last tempo change will take effect.

To start or join a Link session, proceed as follows:

1. Click the LINK button located in the header.



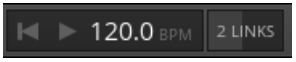
⇨ REAKTOR will either join an existing Link session or begin a new session.

2. Start REAKTOR's playback by clicking the play button.

⇨ REAKTOR's playback is now synchronized with the Link timeline.

3. To stop using Link, and leave the session, click LINK again.

While connected, the LINK button displays how many other Link-enabled applications are connected via Link. The example below indicates that there are two other applications connected.



Active Link session with two other instances connected.

When Link is active, it is not possible to synchronize REAKTOR to incoming MIDI Clock or OSC Sync as a slave. Link will take priority. Sending MIDI Clock and OSC Sync from REAKTOR as a master is still possible.

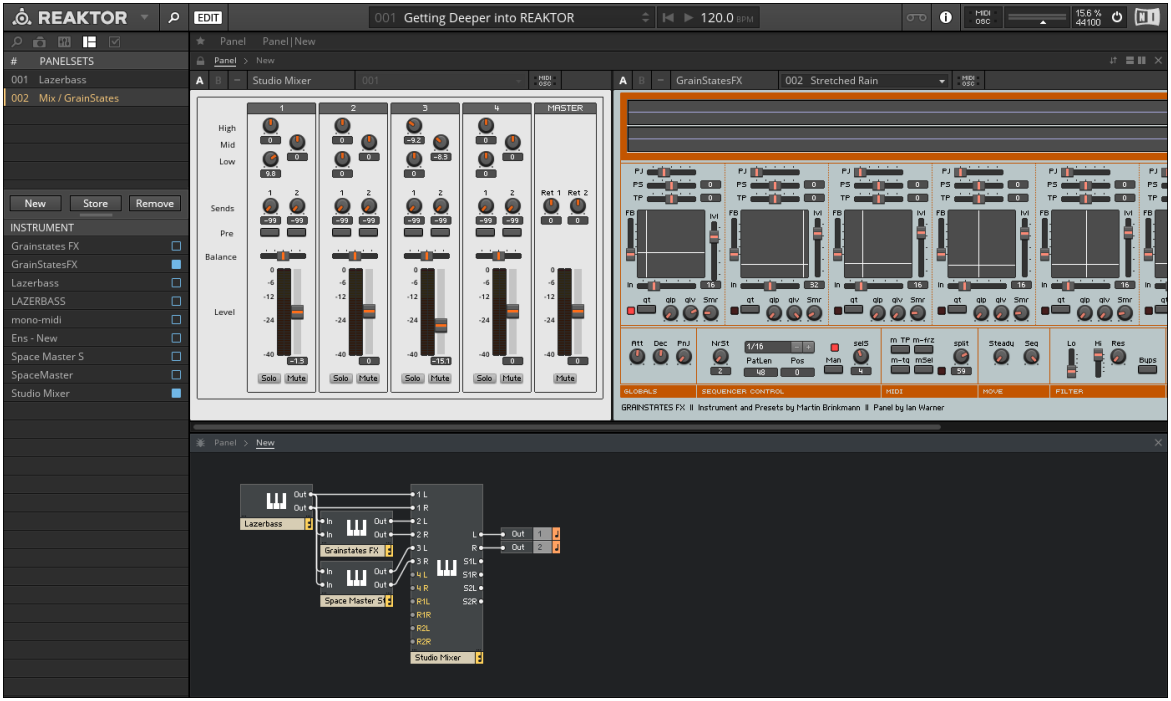
10 Getting Deeper into REAKTOR

The present document has given you an overview over REAKTOR's interface and functionality, and you have learned how to load and play REAKTOR instruments and effects.

To expand your knowledge about topics like Snapshots and controller integration in REAKTOR, as well as make yourself familiar with new topics like Panelsets and the Sample Map Editor, it is suggested that you read the REAKTOR 6 Diving Deeper document next.

If you want to use REAKTOR Racks for patching with Blocks the REAKTOR Blocks Manual provides you with all required information. It is available as part of the REAKTOR Blocks content product. For more information about opening the manual of a content product, refer to ↑1.4, REAKTOR Content Documentation.

If your goal is to build your own REAKTOR Instruments, it is still recommended that you read the Diving Deeper document before moving on to the Building in Primary document.



Getting deeper into REAKTOR

Index

A

A/B View [51] [52]

Ableton Link

- connecting to a network [78]
- joining a Link Session [78]

About screen [14] [16]

Audio and MIDI Settings [28]

- Audio [28]
- MIDI [33]

Audio and MIDI Setup

- Routing [30]

Audio input [18]

Audio output [18]

Automatic saving

- Host sequencer [38]

C

Connection Manager [69]

- Create connection [71]
- Delete connection [73]
- Filter connection [74]
- Filter Ensemble [75]
- Filter Instrument [75]
- Filter MIDI Channel [75]

Core Cell [25]

D

Demo [7]

E

Edit Mode [15] [21]

Ensemble [21]

- Browse [46]
- Edit [38]
- Load [46]
- Locate Ensemble [42]
- Not Found [42]
- Save [38]

F

File type [21]

- Core Cell [25]
- Ensemble [21]
- Instrument [23]
- Macro [24]
- Module [24]
- Rack [22]

H

Host sequencer [37]

- Automatic saving [38]

I

Info Hints [15] [17]

Input level [15]

Instrument [23]

- A/B View [51] [52]
- Header [51]
- Name [52]

L

Latency [30]

Level meter

- Input [15] [18]
- Output [16] [18]

M

Macro [24]

Main area [13]

Main Menu [15]

MIDI [68]

- Connection Manager [69]
- Controller [68]
- Input [16]
- Learn [68]
- Output [16]

Module [24]

O

OSC [68]

Output level [16]

P

Play Mode [20]

Player [15]

Plug-in [37]

Preset [53] [57] [67]

- Load [57]
- Save [57]

R

Rack [22]

REAKTOR Player [7]

Recorder [15]

S

Sample Map Editor [15]

Sample rate [19] [29]

- Factor [20]
- Fixed [20]
- Host/System [20]

Setup

- Audio [28]
- MIDI [28] [33] [68]

Side Pane [13] [15]

Snapshot [53] [67]

- Menu [52]
- Recall [54]
- Snapshot menu [15]
- Store [55]

System info [16] [19]

System requirements [7]

T

Toolbar [13] [14]

Transport [15] [16]

U

User Interface

- Main area [13]
- Overview [13]
- Side Pane [13]
- Toolbar [13]

V

Voices [52]


REAKTOR 6
DIVING DEEPER

The information in this document is subject to change without notice and does not represent a commitment on the part of Native Instruments GmbH. The software described by this document is subject to a License Agreement and may not be copied to other media. No part of this publication may be copied, reproduced or otherwise transmitted or recorded, for any purpose, without prior written permission by Native Instruments GmbH, hereinafter referred to as Native Instruments.

“Native Instruments”, “NI” and associated logos are (registered) trademarks of Native Instruments GmbH.

Mac, Mac OS, GarageBand, Logic, iTunes and iPod are registered trademarks of Apple Inc., registered in the U.S. and other countries.
Windows, Windows Vista and DirectSound are registered trademarks of Microsoft Corporation in the United States and/or other countries.
All other trademarks are the property of their respective owners and use of them does not imply any affiliation with or endorsement by them.

Document authored by: Adam Hanley, Jan Ola Korte

Software version: 6.2 (08/2017)

NATIVE INSTRUMENTS GmbH
Schlesische Str. 29-30
D-10997 Berlin
Germany
www.native-instruments.de

NATIVE INSTRUMENTS North America, Inc.
6725 Sunset Boulevard
5th Floor
Los Angeles, CA 90028
USA
www.native-instruments.com

NATIVE INSTRUMENTS K.K.
YO Building 3F
Jingumae 6-7-15, Shibuya-ku,
Tokyo 150-0001
Japan
www.native-instruments.co.jp

NATIVE INSTRUMENTS UK Limited
18 Phipp Street
London EC2A 4NU
UK
www.native-instruments.co.uk



© NATIVE INSTRUMENTS GmbH, 2017. All rights reserved.

Table of Contents

1 Welcome to Diving Deeper 7
1.1 The REAKTOR 6 Documentation 8
1.2 Manual Conventions 12
2 Play Mode vs. Edit Mode 13
2.1 Play Mode 13
2.2 Edit Mode 14
3 Controlling REAKTOR with MIDI and OSC 15
3.1 Configuring OSC Settings 16
3.2 Making OSC Connections 22
3.2.1 MIDI and OSC Learn 22
3.2.2 Connection Manager 23
3.3 Connect Properties 26
3.3.1 MIDI Properties 28
3.3.2 OSC Properties 28
3.3.3 Ensemble and Instrument Properties 32
4 Automation 35
4.1 Automation Properties 35
4.2 Common Issues 37
5 Snapshots and Presets 39
5.1 Snapshots for Multiple Instruments 39
5.2 Snapshot Master 40
5.2.1 Changing the Snapshot Master 41
5.3 Bank Name 42
5.4 Storing and Editing Snapshots 43
5.5 Re-ordering Snapshots 44

5.6 Morph and Random 45
5.7 Snapshot Banks 48
5.7.1 Creating and Editing Snapshot Banks 49
5.8 Default Values 51
6 The Player and Recorder 52
6.1 The Player 52
6.2 The Recorder 53
6.2.1 Recording Audio 54
6.2.2 Recorder Settings 55
7 The Preferences Window 57
7.1 General Preferences 57
7.2 CPU Usage Preferences 60
7.3 Directories Preferences 61
7.4 Usage Data 66
8 The Sample Map Editor 69
8.1 Sample Maps 70
8.1.1 Importing and Exporting Sample Maps 71
8.2 The Sample List 72
8.2.1 Editing the Sample List 73
8.3 The Map View 78
8.4 Waveform View 84
8.5 Note Actions 86
8.6 Sample Map Editor Settings 88
8.7 External Editor 89

9 Panel Layouts 90
10 Panelsets 94
10.1 Recalling Panelsets 94
10.2 Creating and Editing Panelsets 98
11 An Introduction to Building 101
11.1 Navigating the Structure View 102
11.2 Navigation Tools 105
11.2.1 Split Screen 105
11.2.2 Breadcrumbs 109
11.2.3 Bookmarks 109
11.3 Loading and Connecting Instruments 111
11.4 Learning More 115
12 Keyboard Shortcuts 117
Index 121

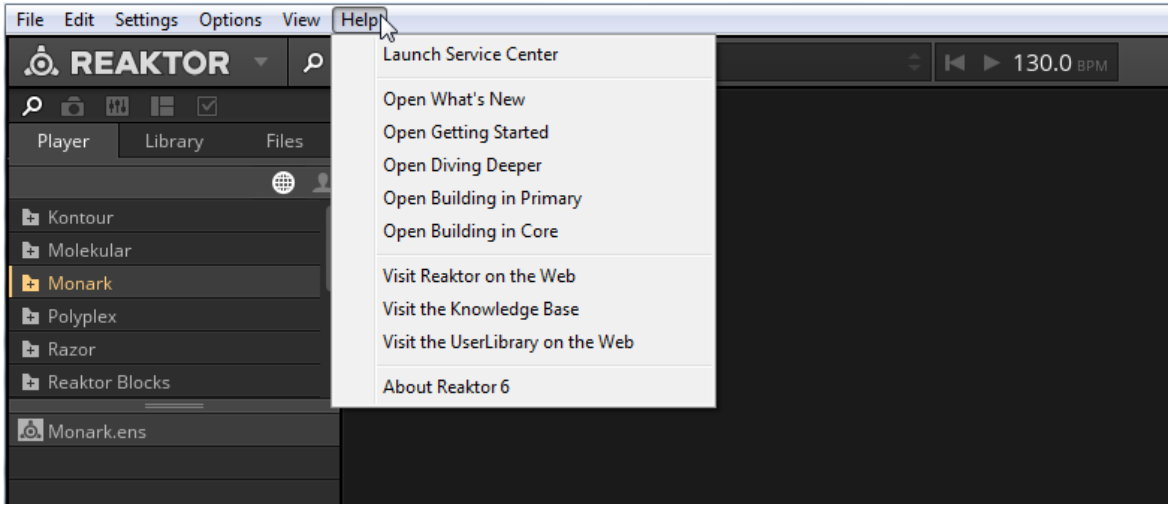
1 Welcome to Diving Deeper

Welcome to Diving Deeper into REAKTOR. This document picks up where the Getting Started left off, covering some of the more advanced techniques and options available when using REAKTOR. As such, it assumes you are familiar with the main areas of the user interface, the general audio and midi settings, and how to load and play Ensembles, including Snapshot handling and basic MIDI control. If this sounds unfamiliar to you, we recommend that you read the Getting Started document first.

This document is designed to show you how to get the most out of your REAKTOR instruments and effects. It covers subjects that you have already been introduced to, like Snapshots and MIDI connections, in more detail. It also introduces more complex concepts like controlling REAKTOR with OSC.

The last section in this document introduces you to the Structure View as a precursor to the REAKTOR 6 Building in Primary document.

1.1 The REAKTOR 6 Documentation



The REAKTOR documentation is accessible from the *Help* menu

The documentation for REAKTOR 6 is divided into five separate documents, guiding you from loading and playing pre-built Ensembles to building your own Instruments.

- **REAKTOR 6 What Is New** is written for users who are already familiar with previous versions of REAKTOR and only describes the latest features in brief.
- **REAKTOR 6 Getting Started** is for new users. It is the only document needed for users who intend to use REAKTOR for loading and playing pre-built REAKTOR instruments and effects.
- **REAKTOR 6 Diving Deeper** expands on the concepts introduced in the Getting Started document. It provides more detail on subjects like Snapshots (REAKTOR's preset system), and introduces advanced topics like OSC control and combining multiple Instruments in one Ensemble.

- **REAKTOR 6 Building in Primary** shows you how to build your own Instruments in REAKTOR's Primary level. It focuses on a series of tutorials that guide you through building your first synthesizers, effects, and sequencers.
- **REAKTOR 6 Building in Core** describes the Core level of REAKTOR with its low-level building features, which can be used for implementing custom DSP algorithms. It includes reference information about the Core Macro Library, an comprehensive collection of DSP building blocks.

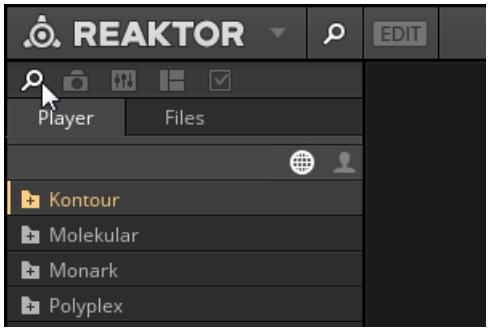
With the exception of the What Is New document, each of the documents listed above builds on the knowledge of the previous documents. While it is not necessary to read all of every document, some of the more advanced documents, like Building in Primary, assume knowledge from the previous documents.

REAKTOR Content Documentation

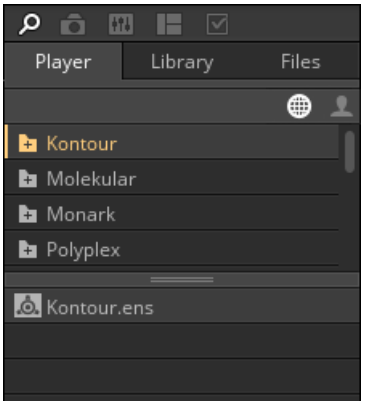
REAKTOR comes with a comprehensive Factory Library, including many pre-built instruments and effects, as well as Blocks, a new modular synthesizer framework that brings the experience of patching an analog modular synthesizer to REAKTOR. The manuals for this content, as well as any REAKTOR-based KOMPLETE instruments or effects you have installed, can be accessed from the Browser in REAKTOR's Side Pane.

To open the manuals for REAKTOR content:

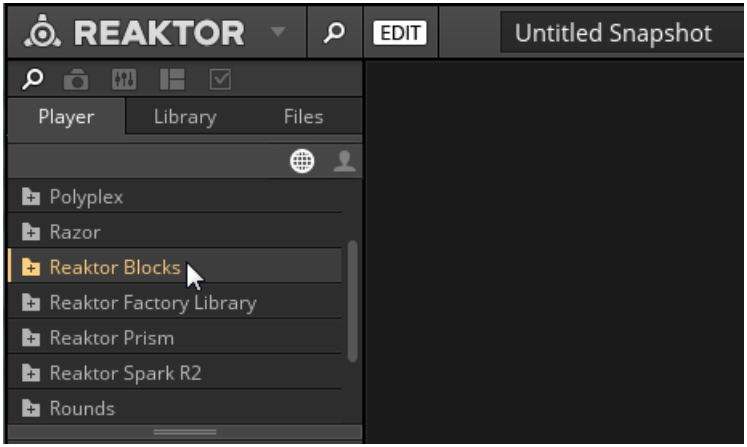
1. Click on the Browser icon in the Side Pane to open the Browser.



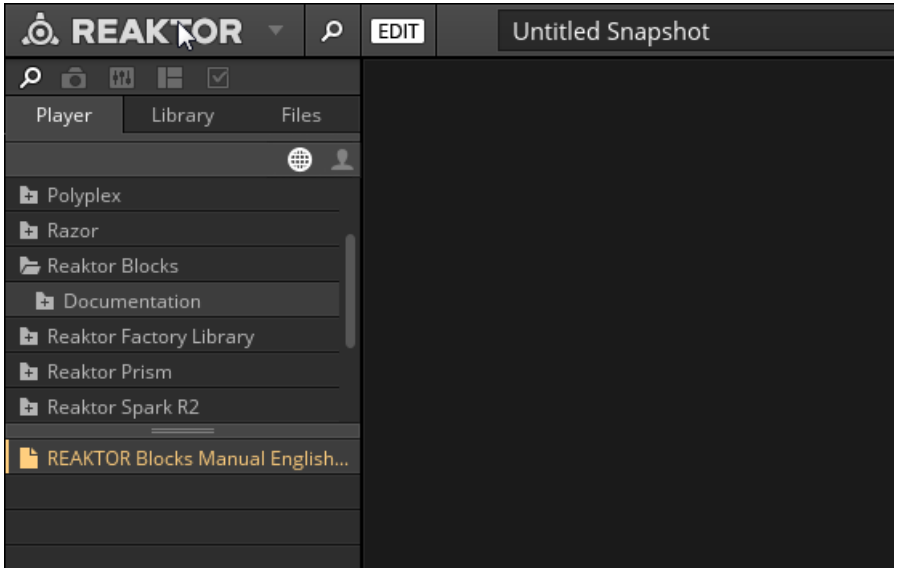
2. Select the Player tab.



3. Double-click on the folder of the content you wish to view the manual for.



4. Select the Documentation subfolder. The manual shows up as a PDF file in the lower section of the Browser.



5. Double-click on the PDF file.

→ The manual will open in your standard PDF reader.

1.2 Manual Conventions

This section introduces you to the signage and text highlighting used in this manual.

- Text appearing in (drop-down) menus (such as *Open...*, *Save as...* etc.) and paths to locations on your hard disk or other storage devices is printed in *italics*.
- Text appearing elsewhere (labels of buttons, controls, text next to checkboxes etc.) is printed in blue. Whenever you see this formatting applied, you will find the same text appearing somewhere on the screen.
- Important names and concepts are printed in **bold**.
- References to keys on your computer's keyboard you'll find put in square brackets (e.g., "Press [Shift] + [Enter]").

▶ Single instructions are introduced by this play button type arrow.

→ Results of actions are introduced by this smaller arrow.

An indented, gray paragraph contains additional, contextual information.

Furthermore, this manual uses particular formatting to point out special facts and to warn you of potential issues. The icons introducing these notes let you see what kind of information is to be expected:



The speech bubble icon indicates a useful tip that may often help you to solve a task more efficiently.



The exclamation mark icon highlights important information that is essential for the given context.



The red cross icon warns you of serious issues and potential risks that require your full attention.

2 Play Mode vs. Edit Mode

REAKTOR has two modes of operation:

- **Play Mode**: Optimizes REAKTOR for using instruments and effects. Editing features are limited in favor of a more streamlined interface and faster loading times.
- **Edit Mode**: Displays all the controls available for building and editing Ensembles.

When you first launch REAKTOR in stand-alone, and every time you load it as a plug-in, it will start in Play Mode.

2.1 Play Mode

The main benefits of Play Mode are:

- **Streamlined interface to maximize the Ensemble Panel space**: In Play mode, all of the Structure navigation controls are hidden, presenting a more streamlined workspace for those who want to use instruments and effects built in REAKTOR, but do not intend to edit the Ensemble Structures.
- **Optimized loading times for Ensembles**: Play Mode is required to be active in order for REAKTOR to use the Compiled Core Cell Code Cache, and so can decrease loading times of larger Ensembles significantly. For more information on the Compiled Core Cell Code Cache please read REAKTOR 6 Building in Primary document.
- **Increased ease of use as a plug-in**: Play Mode also makes working with REAKTOR as a plug-in easier by disabling any control options that are not saved in a host chunk — that is to say, any options that require you to save the Ensemble file when you are using REAKTOR as a plug-in. This means you no longer need to worry about the auto-save functions and host recall for your projects.
- **Double-click knobs to set them to their default value**: In Edit Mode, double-clicking anything on the Ensemble Panel generally opens the Properties for that Module. Since editing a Module's Properties is not possible in Play Mode, double-clicking is redefined to reset knobs and faders to their default value. This makes it easy, for example, to reset an EQ to a clean setting.

2.2 Edit Mode

► To activate Edit Mode, click on the EDIT button in the Toolbar.



Edit Mode makes it possible to alter the Structure of an Ensemble, and edit any other settings that are saved in the Ensemble file.

For example, recalling Panelsets is possible in both Edit Mode and Play Mode, but creating Panelsets is only possible in Edit Mode. The same is also true for Snapshots.

Some of the features described in this manual will require activating Edit Mode, so if you notice a feature described that you cannot see in your instance of REAKTOR, try activating Edit Mode.

3 Controlling REAKTOR with MIDI and OSC

REAKTOR can use two standards for sending and receiving control messages:

- **MIDI**: Musical Instrument Digital Interface - a standard communication protocol for musical instruments created in the early 80s that is still commonly used today. Virtually all keyboard controllers will use MIDI to communicate with your computer.
- **OSC**: Open Sound Control - a more modern protocol that offers faster data rates, higher resolutions, and a more flexible architecture when compared to MIDI. However, OSC is less common and requires a little more technical knowledge to set up.

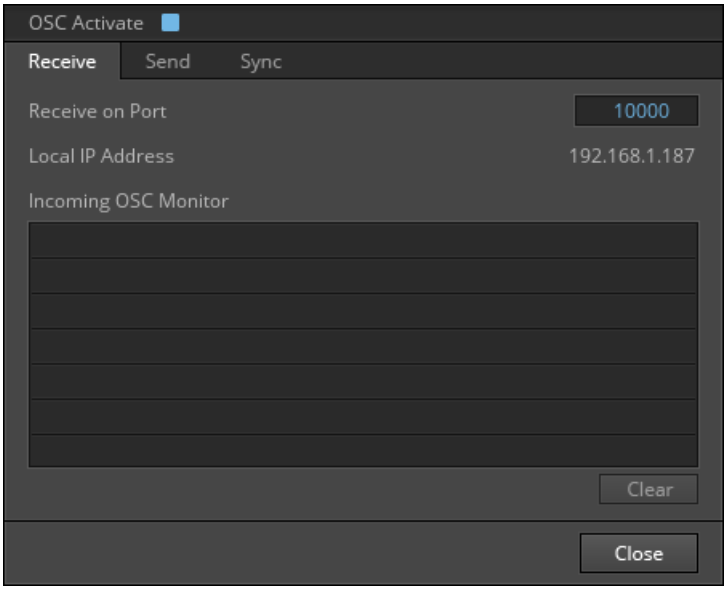


Setting up MIDI and the assigning MIDI controllers is covered in the REAKTOR 6 Getting Started Document. Therefore the following sections will mainly focus on the OSC protocol.

3.1 Configuring OSC Settings

► To open the OSC Settings window, enter the File menu and select *OSC Settings...*



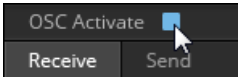


The OSC Settings Window

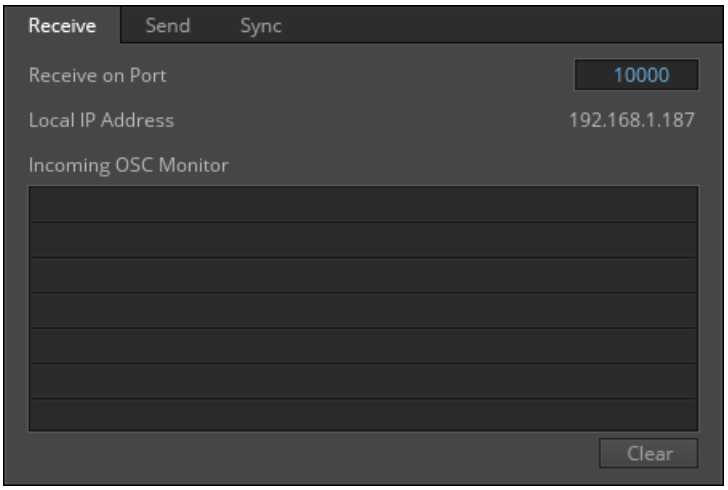
The OSC Settings uses tabs to navigate between the three different sections:

- Receive: Here you can set up communication from an OSC Device to REAKTOR.
- Send: Here you can set up communication from REAKTOR to other OSC Devices.
- Sync: This tab contains the OSC sync options.

► In order to use OSC in REAKTOR, check the OSC Activate check box in the top left corner of the OSC Settings window.



OSC Receive



The OSC Receive Tab

The Receive tab contains three main areas:

At the top is the Receive on Port value display. This is where you can view or edit REAKTOR's OSC input Port number. The Port number is important for use in setting up communication from your OSC device. REAKTOR only receives OSC messages via this Port, so the number has to match the Port number set on your OSC device.

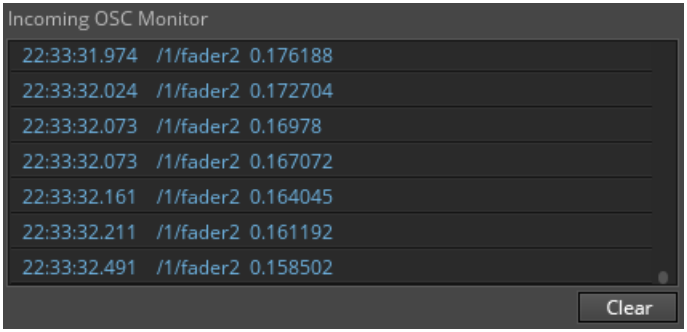
► To change the Port number, double-click on the value displayed, and enter a new value.



Below this you will see a display of the Local IP Address currently accessed by REAKTOR.

These addresses are not editable from REAKTOR, but are displayed for use when setting up your OSC device.

The large area at the bottom of the window is the Incoming OSC Monitor.



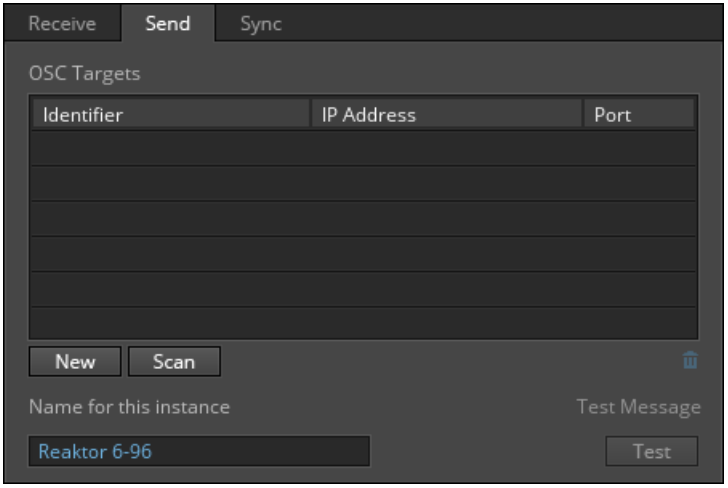
The Incoming OSC Monitor

This displays all of the incoming OSC messages received by REAKTOR, and is primarily used for checking your connection is working correctly.

► To clear the monitor contents, click on the Clear button at the bottom right of the monitor display.



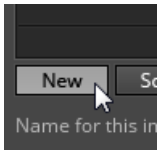
OSC Send



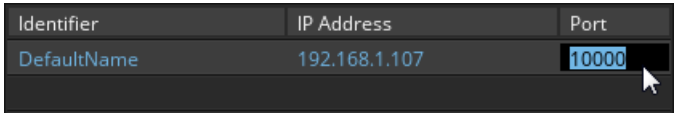
This area is used for entering the details of your OSC device(s) to enable communication with REAKTOR.

To enter the OSC target's information:

1. Click on the New button below the OSC Targets window. A new target will then appear in the window with default information.



2. Enter the relevant information into the fields by clicking on the field and typing the settings from your OSC device.



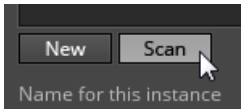
3. When all information is entered correctly, you can send a test message by clicking the Test button.



4. The test message will be sent out to all of the OSC targets in the list. If your OSC controller has an OSC input indicator or monitor, it should show activity when the test message is sent.

You can also automatically search for other OSC enabled instances of REAKTOR:

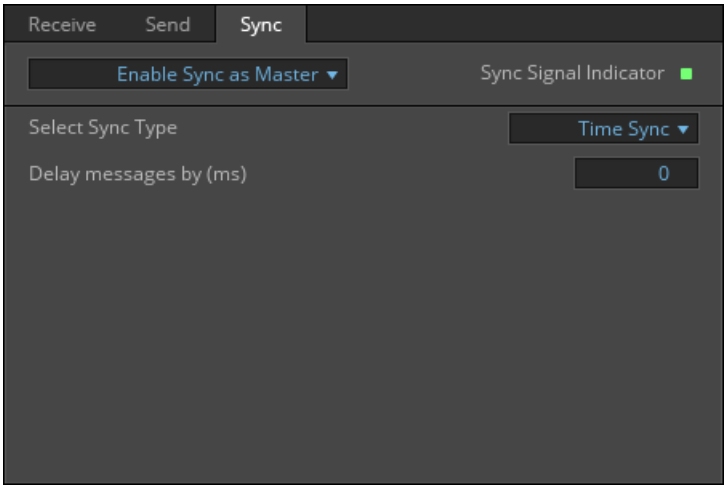
1. Click the Scan button. REAKTOR will send a 'ping' message and listen for a 'pong'.



   A list of devices that sent a 'pong' reply to REAKTOR will appear in the OSC Targets list.
2. Remove any targets that you do not wish to use by clicking on the target name in the list to select it, and then clicking the trashcan icon to remove them.

Note that Scan only scans between port ranges 10000 and 10015.

OSC Sync

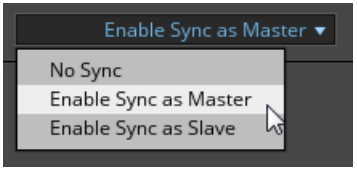


The OSC Sync Tab

The OSC Sync tab contains options for how REAKTOR should use OSC Sync messages.

To change the OSC Sync setting:

1. Click on the dropdown menu.



2. Select the Sync setting you wish to use from the list:
   - *No Sync*: REAKTOR will not send, or respond to OSC Sync messages.
   - *Enable Sync as Master*: REAKTOR will send OSC Sync messages to any device connected in the OSC Send tab.

- *Enable Sync as Slave*: REAKTOR will slave to OSC Sync messages sent from any device connected in the OSC Receive tab.

If you enable OSC Sync, additional options will become available.

The first additional option is always the Select Sync Type menu.



The Select Sync Type Menu

It contains two options: ##CORRECT? DOUBLE CHECK!##

- *Time Sync*: This option sends/receives transport messages (start/stop/continue) as well as tempo information. This can be thought of as song position sync.
- *Clock Sync*: This option sends/receives transport messages only. So in this case, the OSC messages can start and stop playback, but they do not ensure that the tempo of the OSC devices are in sync.

If you selected the Enable Sync as Master option, then another control becomes available: Delay messages by (ms)



Delay messages by (ms) Parameter

Changing the value of this parameter will delay the OSC Sync messages so that they are sent a select number of milliseconds after REAKTOR has generated them.

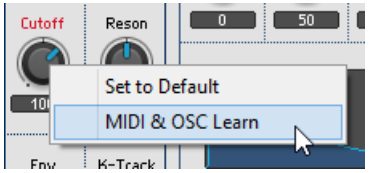
3.2 Making OSC Connections

3.2.1 MIDI and OSC Learn

The easiest way to make an OSC connection is via the MIDI & OSC Learn function.

1. Right-click the control on the interface to which you want to connect.

2. Select *MIDI & OSC Learn*.



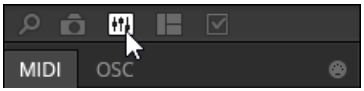
3. Move the OSC controller you want to use.

→ The OSC controller will now be connected to the REAKTOR Ensemble control.

3.2.2 Connection Manager

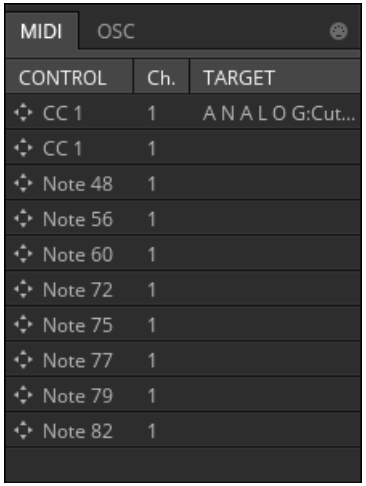
The Connection Manager is located in the Side Pane. It allows you to manage the MIDI and OSC connections in the loaded Ensemble.

► To view the Connection Manager, click on the Side Pane tab with the controller icon.



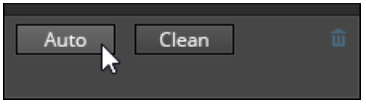
The Connection Manager has two tabs, one for MIDI and one for OSC. Even though they use the different protocols, both tabs work in a very similar way, although the MIDI tab has a few more options than the OSC tab.

- The current connections appear at the top of the list (they will be identifiable by the information present in the Target column).

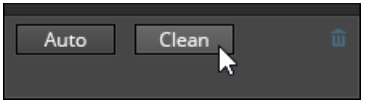


- Recently received messages appear below the current connections (by default, this list will only be refreshed each time you open the tab).
- For MIDI, controller messages will appear above note messages.

▶ To view incoming messages in real-time, activate the Auto button below the list. Not only will this re-populate the list in real-time, but it will also highlight currently used messages.



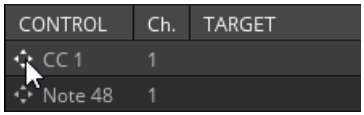
▶ To clear the unconnected controllers in the table, click on the Clean button below the list.



Creating and Deleting Connections

You can make MIDI and OSC connections from the Connection Manager by drag and drop:

1. Click on the arrow icon next to the OSC control you want to use.



2. While holding the mouse, drag the control over the interface parameter you want to control.

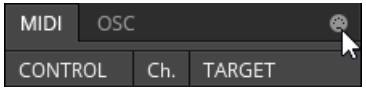


3. Once you are over the parameter, release the mouse.

→ A connection will be made between the OSC control and the REAKTOR Ensemble control.

You can also activate MIDI and OSC Learn from the Connection Manager:

1. Click on the MIDI port icon to activate MIDI and OSC Learn.



2. Click on the REAKTOR control to which you want to connect.
3. Move the MIDI or OSC controller you want to use.

→ A connection will be made between the MIDI/OSC control and the REAKTOR Ensemble control.

You can also use the Connection Manager to remove OSC connections:

1. Select the connection you want to remove by clicking on it in the list.



2. Click on the trashcan icon below the list or press the [delete] key on your keyboard.



→ The selected connection will be removed.

3.3 Connect Properties

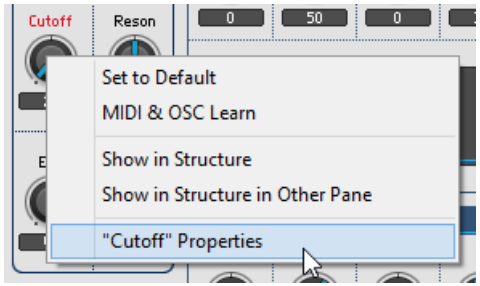
Every item in REAKTOR that can be connected to MIDI or OSC will have a Connect tab in their Properties.

To access the Connect Properties:

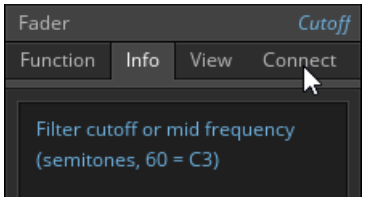
1. Click on the EDIT button in the Toolbar to activate Edit Mode.



2. Right- click on the control you want to edit.
3. Select *"<control name>" Properties*.



4. The Properties for this control will open in the Side Pane.
5. Click on the Connect tab to view the Connect Properties.



The Connect Properties for a control contains a number of sections, but this document will only cover the MIDI and OSC related sections. The other sections will be covered in the REAKTOR 6 Building in Primary document available from the *Help* menu



The MIDI and OSC sections of the Connect Properties

It is possible to set up MIDI and OSC connections both into and out of REAKTOR from the Connect Properties.

3.3.1 MIDI Properties

The MIDI sections feature the following controls:

- MIDI
  - Type: Select the MIDI message type that the control will listen for.
  - Note/CC Number: Select the MIDI Number (if MIDI Type is set to Controller) or MIDI Note (if the MIDI Type is set to Note or Poly Aftertouch)
- MIDI IN
  - On: Activates MIDI to control the selected object.
  - Soft Takeover: Activates smoothing between the MIDI input and the Panel control, so that values do not suddenly jump when a new MIDI message arrives.
  - Incremental: Some dials and rotary knobs can be turned continuously — they do not have fixed lowest and highest positions. Such MIDI controllers send increment messages rather than absolute MIDI controller values. If you are operating your Panel control with such a MIDI controller, activate this option.
- MIDI OUT
  - On: When active, the Panel control will send MIDI messages when you interact with it from the interface.
  - Remote Thru: Forwards the incoming MIDI messages directly to the output.

3.3.2 OSC Properties

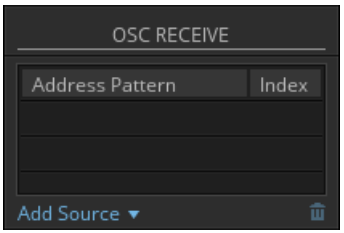
OSC Messages use Address Patterns to identify different controls. These Address Patterns are written similarly to website locations or file locations: strings of text separated by /

For example, an Address Pattern could be `/Page1/Cutoff`

The Address Pattern is then followed by the control value.

OSC can also send arrays of values (i.e. several values in one packet) in a single Address Pattern.

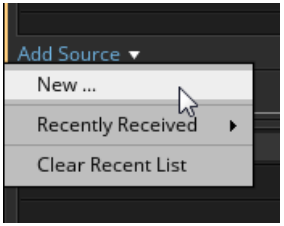
OSC Receive



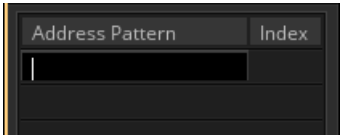
The OSC Receive Properties

You can connect an OSC control to a REAKTOR control via the OSC RECEIVE section of the Connect Properties.

1. Click on the Add Source menu.
2. Select *New...*

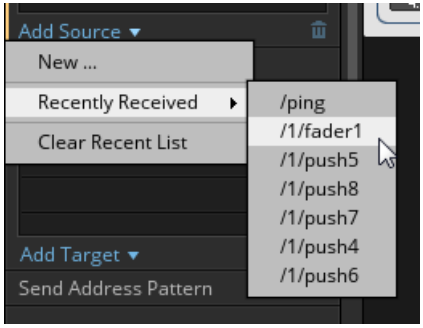


3. A new OSC source will be added to the table above.
4. Enter the Address Pattern of the OSC control.



5. If the OSC control sends data as an array of values, enter an index value in the Index column, otherwise leave it at 0.

6. Alternatively, select from the *Recently Received* sub-menu to automatically enter the Address Pattern of a recently received OSC message.

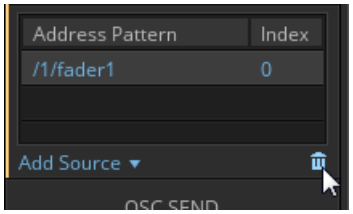


→ The OSC control will be connected to the REAKTOR control.

► If the *Recently Received* sub-menu list is too cluttered, click on the *Clear Recent List* to clear the OSC control addresses shown in the sub-menu.

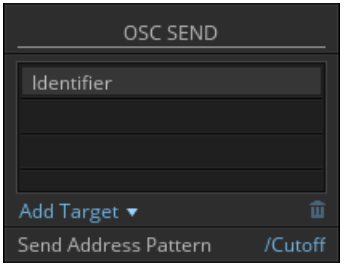
To delete an OSC connection:

1. Select the OSC connection you want to delete from the table.
2. Click on the trashcan icon below the table.



→ The OSC connection will be removed.

OSC Send

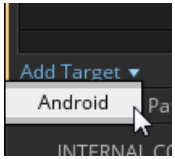


The OSC SEND Properties

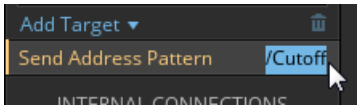
The OSC SEND section works in a different manner when compared to the OSC RECEIVE section.

To make an OSC Send connection:

1. Select an OSC Device from the Add Target menu.



2. Enter a Send Address Pattern. By default this will be the Panel control's name, but you can change this by selecting the text and entering new text. Send Address Patterns should always begin with a /



→ The OSC send connection has been created.



It is a good idea to create both an OSC Receive and an OSC Send connection for a control. This enables bidirectional communication, so that moving the REAKTOR control will update the OSC control and vice versa, keeping both controls at the same value.

3.3.3 Ensemble and Instrument Properties

The Connect Properties for Instruments and Ensembles look different to those of Panel controls.



These Properties deal with more general settings for controlling the Ensemble or Instrument.

MIDI In Settings

There are a number of MIDI In settings, which control how the Ensemble/Instrument will react to incoming MIDI messages.

- External: If you have multiple MIDI devices connected to REAKTOR, you can select which device will control the Ensemble/Instrument.
  - Select *All* if you want the Ensemble to accept MIDI messages from all the MIDI devices.
  - Select *None* if you do not want the Ensemble to accept MIDI message from any device.
- Channel: Select which MIDI channel to connect to the Ensemble.
- Internal: MIDI messages can be passed from Instruments within REAKTOR. From this menu you can select an Instrument from which the current Instrument will receive MIDI messages. This is a useful option for connecting a sequencer Instrument (like *Spiral*) to a synthesizer or sampler.
- Upper Limit: Sets the upper keyboard limit. MIDI notes above this limit will be ignored.
- Lower Limit: Sets the lower keyboard limit. MIDI notes below this limit will be ignored.
- Note Shift: Incoming MIDI notes will be transposed by this amount.
- Sustain Ctrl: Selects which MIDI CC messages will be used to control sustain pedal behavior. The General MIDI standard for this is MIDI CC 64.
- Sustain Ctrl Active: Toggles the sustain pedal behavior on or off.
- Hold Ctrl: Selects which MIDI CC messages will be used to control hold pedal behavior. The General MIDI standard for this is MIDI CC 66.
- Hold Ctrl Active: Toggles the hold pedal behavior on or off.
- Morph Ctrl: Selects a MIDI CC to connect to the Snapshot Morph Slider (see section ↑5.6, Morph and Randomfor more information on this feature).
- Morph Ctrl Active: Toggles the MIDI control of the Snapshot Morph Slider on or off.

MIDI Out Settings

There are three options for an Ensemble's MIDI output:

- External: If you have multiple MIDI devices connected to REAKTOR, you can select to which device the Ensemble will send MIDI messages.
  - Select *All* if you want the Ensemble to send MIDI messages to all of the MIDI devices.
  - Select *None* if you do not want the Ensemble to send any MIDI messages.
- Channel: Select on which MIDI channel the Ensemble will send its MIDI messages.
- Internal: MIDI messages can be passed from Instruments within REAKTOR. From this menu you can select an Instrument to which the current Instrument will send MIDI messages. This is a useful option for connecting a sequencer Instrument (like *Spiral*) to a synthesizer or sampler.

4 Automation

Like other plug-ins, controls in REAKTOR Ensembles can be automated by a host sequencer. Instruments and effects from Native Instruments (like MONARK or THE FINGER) already have their automation settings set up to function correctly.

However, some Ensembles downloaded from other sources (including the REAKTOR User Library) might not be correctly set up, in which case you will need to fix the Automation IDs of the Ensemble.

If you are combining multiple Instruments in an Ensemble, you will also need to manage the automation settings.

Automation IDs

Every control in REAKTOR is given an Automation ID, which it compiles into a list to send to the host sequencer, telling it how many automatable parameters the REAKTOR plug-in contains.

Automation IDs are automatically generated, but are not always automatically updated.

- Removing controls will not update the automation list. If you add two controls to your Ensemble, and then delete the first one you added, a hole will be left in the automation list.
- Removing Instruments will update the Ensemble automation list. If you create an Ensemble with two Instruments, and delete the first one you added, the Automation IDs of the second Instrument will be moved up to fill the gap. Note that this behavior will not fill any gaps in the automation list of the Instrument.

In some cases, this should not be a problem, especially if you are using REAKTOR in standalone. However there are cases in which you will need to manage the Automation IDs, and this can be done from the Automation Properties.

4.1 Automation Properties

The settings related to automation can be found in the Properties for an Ensemble or Instrument, under the Connect tab in a section labelled AUTOMATION.



The Automation Properties

The following settings are available:

- Show Name: This checkbox selects whether or not the Instrument name will be included in the Automation Name of the control. Note that this option is not present in the Ensemble Automation Properties.

- Base ID: Displays the lowest Automation ID in use by the Instrument/Ensemble. This display can be used to show where an Instrument is positioned in the Ensemble's main Automation ID list.
- Max ID: This option selects the number of Automation IDs needed by the Instrument/Ensemble. Most of the time, this can be set to the same value as the Max ID in use display.
- Max ID in use: Displays the highest Automation ID of all the Modules inside the Instrument/Ensemble that can be automated.
- ID Actions: From this menu you can select several functions to perform on the Automation IDs in the Instrument/Ensemble:
  - *Instrument Up*: Moves the Instrument higher in the Ensemble's main Automation ID list, reducing its Base ID value.
  - *Instrument Down*: Moves the Instrument lower in the Ensemble's main Automation ID list, increasing its Base ID value.
  - *Compress IDs*: Select this option to compress the Automation ID list, effectively removing any gaps in the Automation ID list by changing the Automation IDs of the controls.
  - *Sort and Compress IDs*: Similar to the function above, but the controls are sorted in alphabetical order and given new Automation IDs accordingly.

4.2 Common Issues

If you are using REAKTOR Ensembles in a host sequencer and find that automation is not working as expected, there are often common causes that can be fixed quite easily.

The Max ID Setting Is Not High Enough.

If you are trying to automate a control and nothing is happening, try the following for each Instrument in the Ensemble and for the Ensemble itself:

1. Use the *Compress IDs* function to remove any gaps in the Automation ID list.
2. Increase the Max ID setting so that it is equal to the Max ID in use value.



If you find that automation for the control still does not work, then it is possible that automation has been blocked for that control by the builder of the Ensemble, which usually means that automating this control might cause problems.

The List of Automatable Controls in the Host Sequencer is Difficult to Read.

If you are trying to locate a control in the Automation ID list of your host sequencer, but find the Automation ID list difficult to read, try to following:

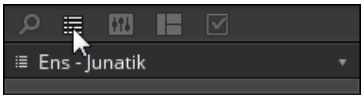
1. Use the *Sort and Compress IDs* function to arrange the Automation IDs in alphabetical order, and to remove any gaps in the Automation ID list.
2. Uncheck the Show Name option to remove excess text from the Automation Names.

5 Snapshots and Presets

Snapshots and Presets enable you to store and recall the settings of an Instrument's parameters.

The basics of storing and recalling Snapshots and Presets are covered in the REAKTOR 6 Getting Started document. This following section introduces you to more advanced Snapshot techniques and tools, available from the Snapshot tab in the Side Pane.

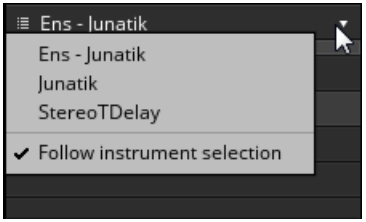
► Click on the list icon to view the Presets tab in the Side Pane.



5.1 Snapshots for Multiple Instruments

Since an Ensemble can contain multiple Instruments, and Instruments can have their own Snapshots, there needs to be a way of selecting which list of Snapshots is displayed in the Snapshot tab.

► Click on the menu at the top of the Snapshot tab to select an Instrument.



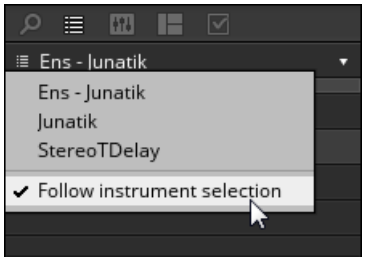
→ The Snapshots from the selected Instrument will be displayed in the Snapshot list.

The Ensemble can also have Snapshots, and so it also has an entry in this menu. The Ensemble entry is always located at the top and is marked with the prefix Ens - .

The Snapshot tab can also follow your Instrument selection automatically:

1. Open the menu at the top of the Snapshot list.

2. Select the *Follow instrument selection* option.



With this option active, if you edit some parameters in an Instrument, that Instrument's Snapshots will be displayed in the Snapshot list, and the Snapshot list will always display the Snapshots of the last Instrument you interacted with via the mouse.

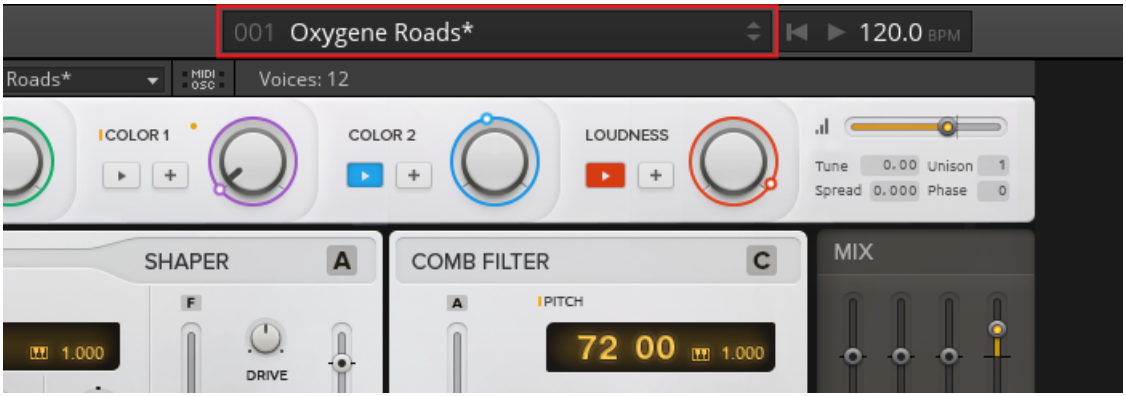
5.2 Snapshot Master

If an Ensemble contains multiple Instruments, Snapshots for all of the included Instruments can be recalled at once from the Snapshot Master.

Recalling a Snapshot from the Snapshot Master will recall the control values for all Instruments in the Ensemble and the Ensemble itself, including the selected Snapshots for each of the Instruments.

By default the Snapshot Master is the Ensemble.

The Snapshots of the Snapshot Master are the Snapshots that are used to populate the Snapshot menu in the Toolbar.



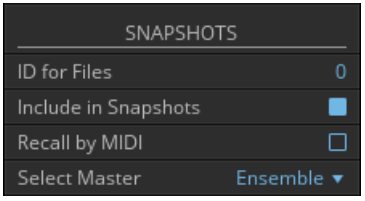
The Snapshot menu in the Toolbar

5.2.1 Changing the Snapshot Master

The Snapshot Master can be changed from the Ensemble to any of the Instruments inside it.

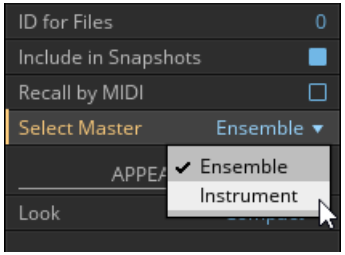
To change the Snapshot Master:

1. Select the Ensemble.
2. Open the Ensemble Properties.
3. Select the FUNCTION tab.
4. Find the section called SNAPSHOTS at the bottom of the Properties.



5. In this section, click on the Select Master menu.

6. Select the Instrument you wish to use as the Snapshot Master from this menu and click on it.



→ The selected Instrument is now the Snapshot Master for the Ensemble.

5.3 Bank Name

User Presets for Ensembles are located in the following directories:

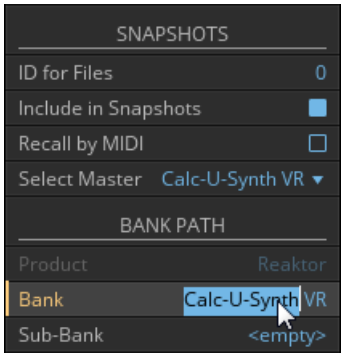
- On Windows: C:\Users\<User>\Documents\Native Instruments\User Content\Reaktor \<Bank Name> \
- On OS X: Macintosh HD:\Users\<User>\Documents\Native Instruments\User Content\Reaktor\<Bank Name> \

The Bank Name defines the folder into which Presets will be saved. By default this will be the Ensemble name, but you can change that if you wish.

To change the Bank name:

1. Select the Ensemble and open the Properties tab.
2. Navigate to the Function sub-tab.
3. Scroll to the area called BANK PATH.

4. Enter the name you wish to use in the Bank property.



→ The Bank name will be used as the folder name used by Presets for the Ensemble.

5.4 Storing and Editing Snapshots

Storing and editing snapshots can be done from the Snapshots tab in the Side Pane.

In order to store and edit Snapshots, you need to have REAKTOR in Edit Mode. See section ↑2.2, Edit Mode ↑2.2, Edit Mode for more information.

At the bottom of the Snapshot pane are three buttons for storing Snapshots.



The Snapshot Storage Buttons

- Add: Creates a new Snapshot at the end of the Snapshot list.
- Store: Overwrites the currently highlighted Snapshot.
- Insert: Creates a new Snapshot directly after the currently highlighted Snapshot, moving any other Snapshots down one slot in order to make space.

Snapshots are stored in the Ensemble, so you will need to save the Ensemble or Instrument in order to make changes to the Snapshots permanent.

If you want to delete a Snapshot, you can do this in two ways:

► Select the Snapshot you wish to delete and press the [Delete] key on your keyboard.

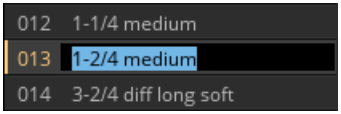
Alternatively…

1. Right-click on the Snapshot you wish to delete to open the context menu.
2. Select *Delete*.



You can select multiple Snapshots by holding the [Shift] key while clicking on the Snapshots.

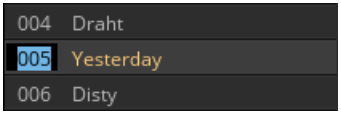
► You can rename a Snapshot by double-clicking on its current name and typing in a new name.



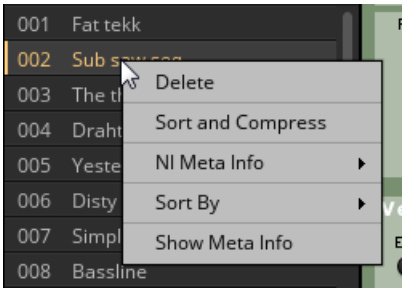
5.5 Re-ordering Snapshots

You can re-arrange Snapshots in a Bank in a number of ways:

► Double-clicking on the Snapshot number and entering a new number will move the Snapshot to that location, and displace any Snapshots that may be in the way of that action.



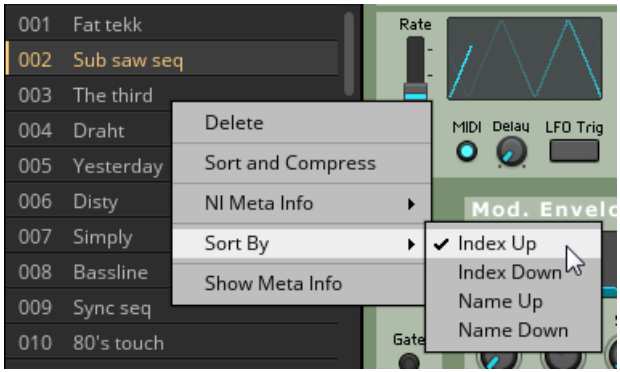
► You can also right-click on any Snapshot in the Snapshot list and select the *Sort and Compress* option to automatically re-order the Snapshots so that no empty slots are left between them.



Sorting Snapshots

You can also re-organize the Snapshots without permanently re-ordering them:

1. Right-click on any Snapshot to open the Snapshot context menu.
2. Navigate to the *Sort By* entry to open the sorting sub-menu.



3. Click on an option to select how you wish the Snapshots be organized.

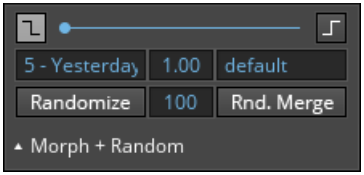
5.6 Morph and Random

REAKTOR gives you the ability to morph Snapshots, as well as randomize them.

The Morph and Random controls are hidden by default.

► To display the Snapshot Morph and Random controls, click on the Morph + Random arrow at the very bottom of the Snapshot pane.





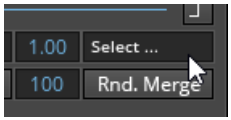
The Snapshot Morph and Random Controls

Morphing Snapshots

The Snapshot morph feature allows you to blend the parameter values of two different Snapshots.

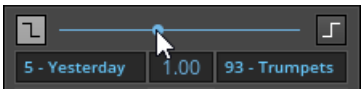
To use this feature, you must first select the two snapshots you want to blend.

1. The selected Snapshot will automatically be selected as the first Snapshot, you will see it in the box on the left.
2. To change this, simply select a new Snapshot as normal.
3. To select the second Snapshot, click on the box to the right of the Morph area (above the Rnd. Merge button). This will set the text in the box to Select ...



4. While this text is displayed, select a Snapshot as normal.

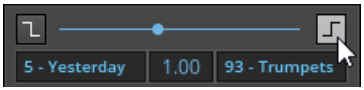
► With the two Snapshots selected, blend between them by clicking and dragging on the blue slider.



Note that when you move the Morph slider past the 50% mark, selecting a new Snapshot will place that Snapshot in the right box.

Binary controls, like buttons and switches, will not blend with the Morph slider, because they can only ever be on or off with no states in between. Instead you select one of the Snapshots whose settings will be used for any binary controls in the instrument. By default this is the first Snapshot.

► Clicking on the buttons to the left and right of the morph slider will select which Snapshot will be used for button and switch settings.



The Morph slider has a blend time, which smoothes the movement of the slider.

► To set the Morph time, click and drag vertically on the numeric value between the two Snapshot boxes. This will set the smoothing time in seconds.



You can also set the Morph time by double clicking on this value and entering a specific number (from 0 to 60)

Randomizing Snapshots

The bottom row in the Morph + Random area contains the Snapshot Randomization controls.



The Snapshot Randomization Controls

There are two buttons that randomize the parameter settings of the Instrument in different ways:

- Randomize: triggers the randomization of the values based on their current settings.
- Rnd Merge: randomizes each parameter by blending between the two Morphing Snapshots. Each parameter is blended by its own random amount, so this technique produces results not possible when doing a straight morph using the Morph slider.

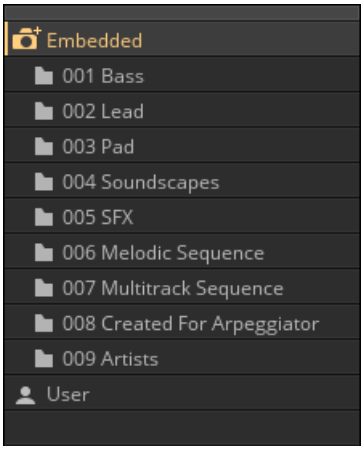
► You can set the amount of randomization by editing the value between these buttons, either by clicking and dragging vertically, or by double-clicking and entering a specific value (from 0 to 100).



5.7 Snapshot Banks

The upper half of the Snapshot tab contains the list of Snapshot Banks.

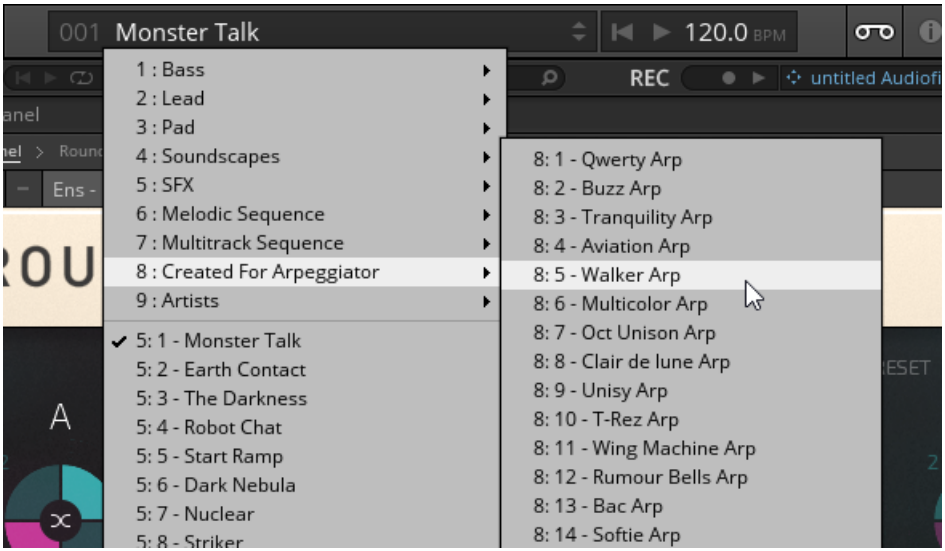
► To display the Snapshot Banks, click on the Snapshot icon labelled Embedded at the top of the Snapshot tab.



Snapshot Banks from ROUNDS

A Snapshot Bank can contain up to 128 Snapshots, and are a useful way to categorize your Snapshots.

Snapshot Banks appear as sub-menu categories in the Instrument Snapshot Menu, and in the Snapshot Menu in REAKTOR's Toolbar.



Snapshot Banks as Snapshot Sub-menus

► You can select a Bank by clicking on its name in the Snapshot tab.

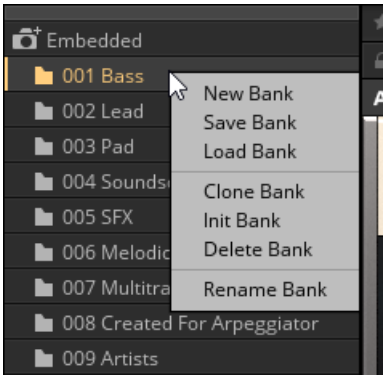
→ The Snapshot table will display the names of the Snapshots in the selected tab.



This action will not load a Snapshot, so you can browse the contents of the Banks before loading a new Snapshot.

5.7.1 Creating and Editing Snapshot Banks

The options for creating and editing Snapshot Banks are all found by right-clicking on the Snapshot Banks.



Editing Options for Banks

The options available are:

- *New Bank*: creates a new empty Snapshot Bank and places it in the first available Snapshot Bank slot.
- *Save Bank*: opens a dialog window, which allows you to save the Snapshot Bank as an external file. You can use this function to export a Snapshot Bank when you want to share it.
- *Load Bank*: opens a dialog window, from where you can browse your file system for Snapshot Banks.

Loading a Snapshot Bank will overwrite the contents of the currently selected Snapshot Bank, so it is advised to create a new empty bank before loading a Snapshot Bank.

- *Clone Bank*: creates a duplicate of the selected Snapshot Bank, including duplicates of all the Snapshots contained within.
- *Init Bank*: initializes the Snapshot Bank, deleting all of its Snapshots.
- *Delete Bank*: removes the Snapshot Bank and deletes its contents.
- *Rename Bank*: enables renaming of the Snapshot Bank. After selecting this option, type in the new name for the bank, then press [enter] / [return] to finish editing.

► You can also rename a Snapshot Bank by selecting it, pressing the [enter] / [return] key on your keyboard, and then typing a new name.



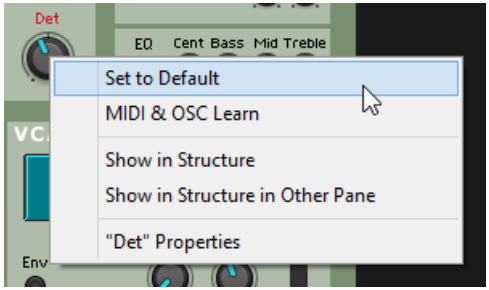
5.8 Default Values

All parameters that can be stored in a Snapshot have a default value. Generally this is a neutral or middle setting. For example, an EQ gain slider would have a default setting of 0dB (no gain change).

► When in Play Mode, you can set a control to its default value by double-clicking on it.

When in Edit Mode, double-clicking opens the control's Properties, so there is another way to set it to its default value:

1. Right-click ([ctrl] + click on OSX) on the control to open a menu.
2. Select *Set to Default* from the menu.

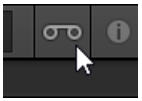


→ The control will now be set to its default value.

6 The Player and Recorder

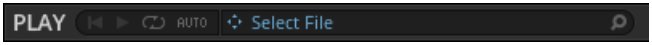
The Player and Recorder utilities allow you to play back and record audio files.

► Click on the tape icon in the Toolbar to open the Player and Recorder.



→ The Player and Recorder bar will open below the Toolbar, with the Player on the left and the Recorder on the right.

6.1 The Player



The File Player

The File Player enables you to route the Audio signal from an audio file (*.wav, *.aif, or *.aiff) into your REAKTOR Structure. The output of the File Player is sent directly to the first two Audio Inputs of the top-level Ensemble. This is particularly useful for testing when building an effect.

When the File Player is playing, all other signals that would otherwise arrive at the first two Audio Inputs are overridden in favor of the File Player's output signal.

If the Ensemble contains no audio inputs, then no sound will be heard when you use the File Player.

There are three ways to load an audio file into the Player:

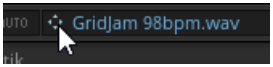
- The first way is to double-click on any audio file in the Files tab of the Browser in the Side Pane. This action opens the Player/Recorder Bar (if not already open), loads the file into the Player and automatically starts the file playback.

- Alternatively, you can drag and drop audio files into the text field of the File Player (when first opened, this will say Select File). In addition to dragging and dropping from the Browser, you can fetch files from other places like from the Desktop of your operating system. Again, after you have performed this action the File Player automatically starts the file playback.
- The third way is to press the AUTO button of the File Player. This activates the Player's Auto Mode. A single click on any file audio file in the Browser results in the file being loaded and played back in the File Player.

The Player has three playback controls. They are (from left to right):

- Reset play position to zero. This control is only visible after a file has been played.
- Play/pause audio playback.
- Loop mode on/off.

▶ If you have a file loaded in the Player, you can use the anchor icon to drag and drop this file to another location. For example, you could drag the file into REAKTOR's Sample Map Editor.

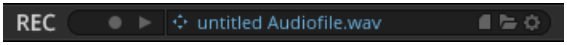


▶ Click on the magnifying glass icon to the far right of the Player to open the Files tab of the Browser in the Side Pane.



The Magnifying glass icon is only visible if the Files Browser is not visible.

6.2 The Recorder



The File Recorder

The File Recorder enables you to record an Ensemble's audio output from its first two Audio Outputs. The recorded audio signal is written to an audio file on your disk.

Windows systems record *.wav files and OS X systems record *.aif files.

6.2.1 Recording Audio

To record a new audio file with the File Recorder:

1. Click on the Folder icon in the File Recorder.



2. Select a location, name, and format (wav or aiff) for the audio file in the dialog window.
3. Click Save to create the audio file.
4. Click on the Record Arm circle to activate record mode.



5. When you are ready to record, click on the Play button.



6. Click on the Stop button to stop recording.

→ The recorded audio will be saved in the audio file you created in steps 2 and 3.

If you do not specify a file name and location as directed in steps 1 to 3 above, then the audio file you record into will be located in the Audio Files sub-folder of the User Library location.

The recorded audio file will be a 16bit uncompressed wav or aiff file. The sample rate of the audio file will be the same as the sample rate REAKTOR was running at when the recording took place.

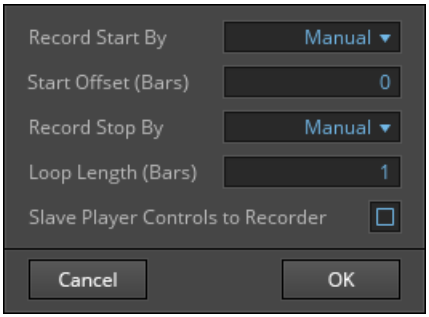
It is also possible to trigger the recording process via other means. These options are located in the Recorder Settings, which are described in the following section.

6.2.2 Recorder Settings

► You can access the Recorder's settings by pressing the Recorder Settings button (the cog/gear icon).



→ The Recorder Settings dialog window will be displayed.



The Recorder Settings

The Recorder Settings has 5 options available:

- Record Start By: This menu allows you to select a recording method (when the Recorder is armed). The options available are:
  - *Manual*: Recording can only be started by pressing the Recorder Start button.
  - *Clock Start*: Recording starts when the REAKTOR Clock is started with the Start Clock button in the Main Toolbar.
  - *Note On*: You can also start recording by a MIDI Note On event, i.e. when a key is pressed on your MIDI or computer keyboard.
- Start Offset: In this edit field you can specify a time delay in bars before the recording starts after a "record" event (specified in the Record Start By menu) has arrived.
- Record Stop By: This menu allows you to choose under which conditions the recording process is stopped:

  - *Manual Only*: You can only stop the recording process using the Recorder Stop button.
  - *Clock Stop*: Recording stops when the REAKTOR Clock is stopped with the Stop Clock button in the Main Toolbar.
  - *Note Off*: You can also stop recording by a MIDI Note Off event, i.e. when a key is released on your MIDI or computer keyboard.
  - *Loop Length*: The File Recorder stops recording when the loop length duration, as specified by the Loop Length edit field in bars, has been reached.
- Loop Length: This edit field lets you specify the duration of the recording in bars. This value is only used when the *Loop Length* option in the Record Stop By menu has been selected.
- Slave Player Controls to Recorder: With this checkbox you can cause the Player transport controls to follow those of the Recorder. This can be useful when you want to use the Player to send an audio file through a REAKTOR effect Instrument and simultaneously record the result at the output using the Recorder.

7 The Preferences Window

The Preferences Window contains options for some of REAKTOR's general functions and behavior. These are options that affect REAKTOR overall, but not the loaded Ensembles. Things like Library folder directories and color schemes are located here.

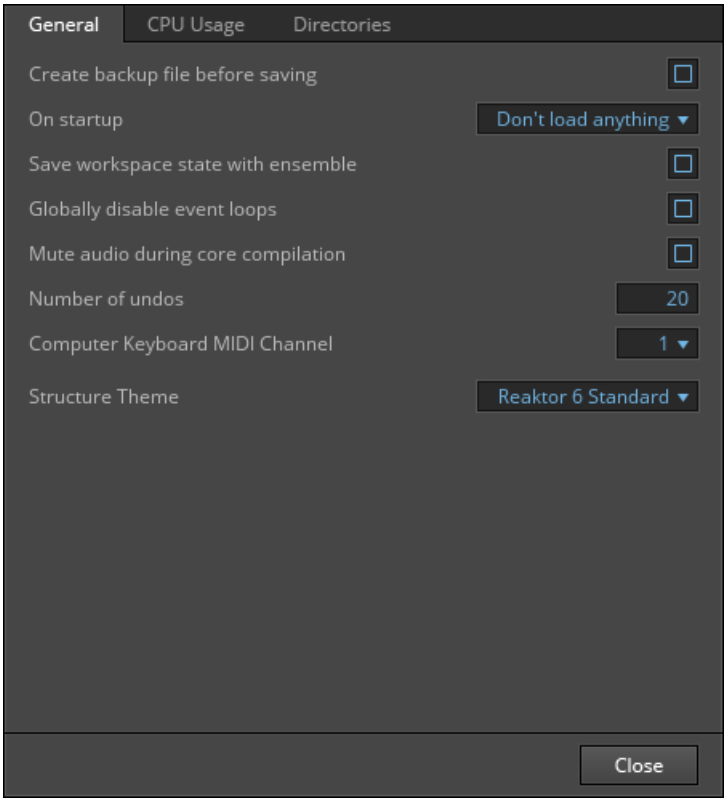
► The Preferences Window can be accessed from the File Menu, by clicking on *Preferences …*, or by using the keyboard shortcut [Ctrl]/[Cmd] + [,]



The Preferences Window has four tabs, which will be described in the following sections…

7.1 General Preferences

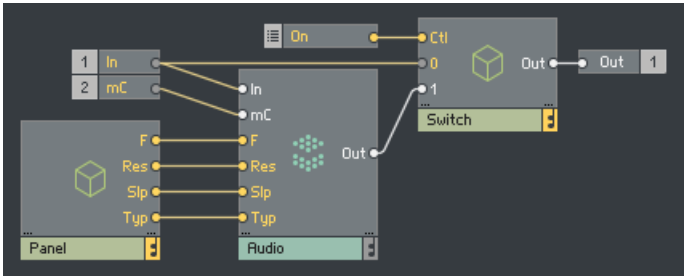
► Click on the General tab to display the General Preferences.



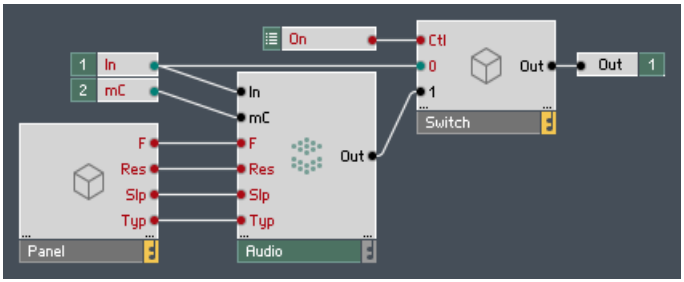
The General Preferences

- Create backup file before saving: When this checkbox is engaged, REAKTOR will create a backup file of your Ensemble titled “Backup of <Ensemble name>” when you execute the Save Ensemble command. This backup is the version of the Ensemble as it was from the last save or load command.
- On startup: This menu determines which Ensemble, if any, REAKTOR will load when it starts up. There are three options:
  - *Load new ensemble*
  - *Load last ensemble*
  - *Don’t load anything*

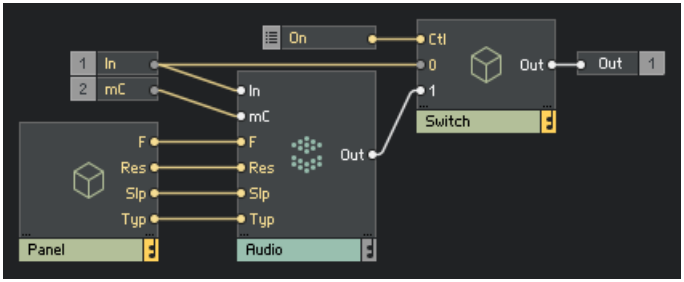
- Save workspace state with ensemble: This option restores the Ensemble's GUI (graphical user interface) to the state it was when the Ensemble was last saved.
- Globally disable event loops: This prevents Event signal feedback loops from occurring in Ensembles.
- Mute audio during core compilation: When enabled, this option turns off all audio processing during the period of time that it takes to compile a Core Object. This makes compilation faster.
- Number of undos: This edit field allows you to set the maximum number of undos that can be performed in a row. Enter "0" to disable the undo function completely. More undos require more memory, but also provides more opportunities to correct mistakes.
- Computer Keyboard MIDI Channel: This drop-down menu specifies the MIDI channel for MIDI Notes played from the computer keyboard. This is set to 1 by default because REAKTOR Ensembles are set to receive MIDI messages on channel 1 by default.
- Structure Theme: This menu contains color scheme options for the Structure View. There are four options to choose from:
  - *Reaktor 6 Standard*: the default color scheme.



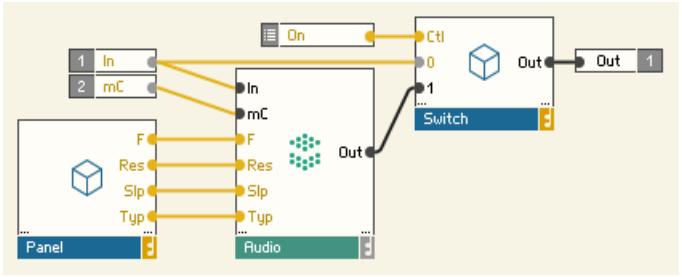
- *Reaktor 5*: the color scheme of REAKTOR 5.



- *Dark Grey*: a darker variant of the Standard color scheme.

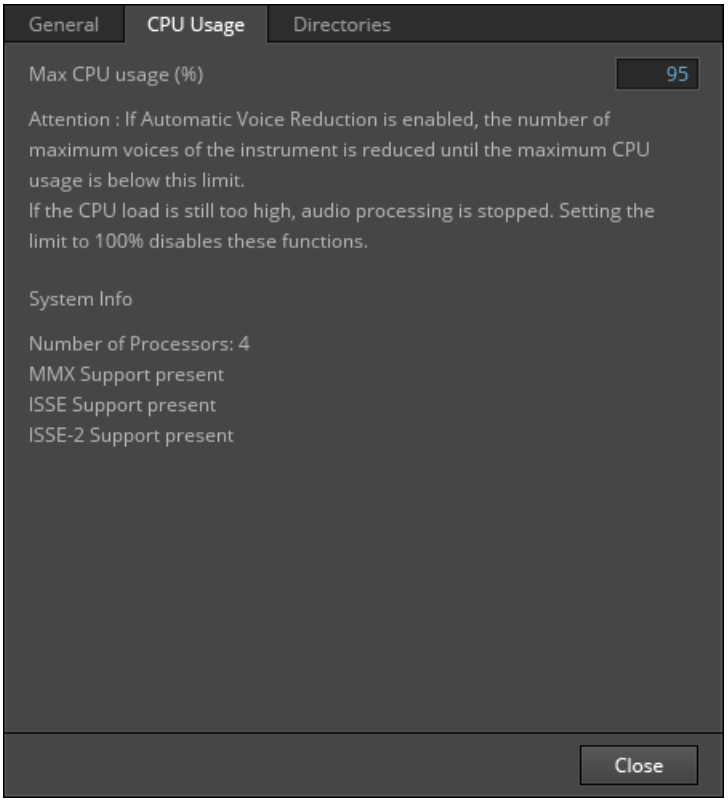


- *White*: a bright color scheme with white Modules and Macros.



7.2 CPU Usage Preferences

► Click on the CPU Usage tab to view the CPU Usage Preferences.



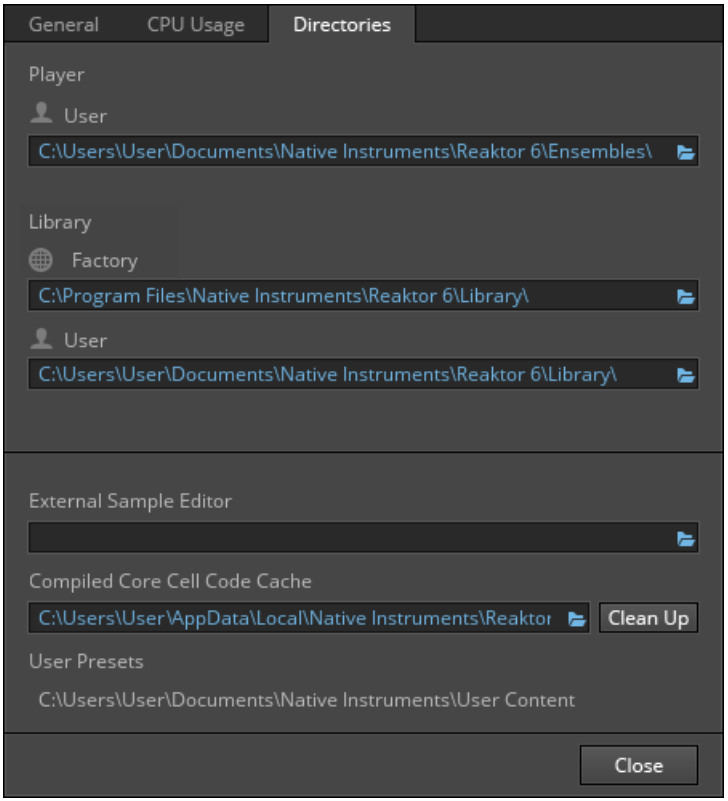
The CPU Usage Preferences

The Max CPU usage (%) setting in this window is directly linked to the Automatic Voice Reduction feature, which is described in the REAKTOR 6 Building in Primary document.

Below the description of this feature is a section called System Info that displays technical information about your computer's CPU. This information may be useful to include if you ever need to contact technical support.

7.3 Directories Preferences

► Click on the Directories tab to view the Directories Preferences.



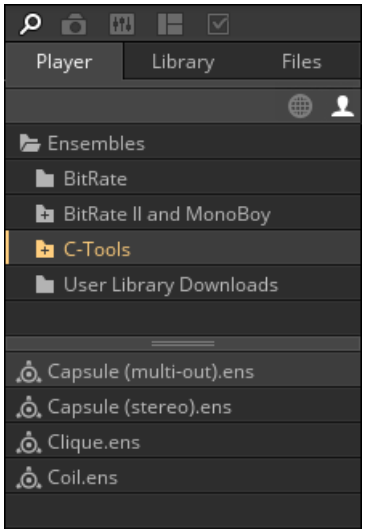
The Directories Preferences

From the Directories Preferences you can set where on your computer's hard drive REAKTOR will search for files and folders in order to populate the Browser.

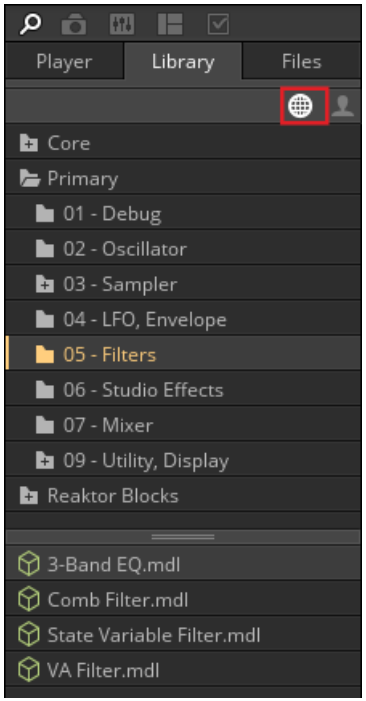
Content Directories

In the top half of this window there are three locations you can set for REAKTOR to search for REAKTOR content like Ensembles, Instruments, and Macros:

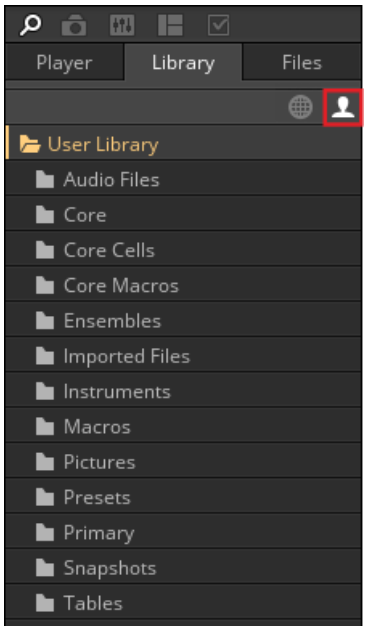
- Player - User: Defines the path for content to populate the User sub-section of the Player tab in the Browser. This is a good location for any Ensembles you may have purchased or downloaded from third-party developers that are not official content products activated using Native Access.



- Library - Factory: Defines the location of official content products from Native Instruments as well as third-party developers. Find more information about official content products from third-party developers in section Third-Party and Protected Content in the REAKTOR 6 Getting Started document.



- Library - User: Defines the location of content for the User sub-section of the Library tab in the Browser.



► Click on the folder icon to change the location of any of these directories.



External Sample Editor

It is possible to launch an external sample editing program from the Sample Map Editor (see section ↑8, The Sample Map Editor ↑8, The Sample Map Editor for more information on launching an external sample editor).

You can define which program REAKTOR will launch as the external sample editor from this area.

► Click on the folder icon to open a dialog window, and then select an application that can load wav and aiff files (for example, Audacity)



→ The selected program will launch whenever you open the external sample editor from the Sample Map Editor.

Compiled Core Cell Code Cache

When opening an Ensemble with large Core Cell structures in REAKTOR, a cache file of the compiled Core Cell code is saved on your hard drive. The next time you load the Ensemble, REAKTOR will access this cache file and skip the compiling stage, which significantly speeds up the loading process.

- The Compiled Core Cell Code is only used when REAKTOR is in Play Mode.
- Switching to Edit Mode when using Ensembles with large Core Cell structures causes a re-load of the Core Cells, which can cause REAKTOR to pause momentarily.

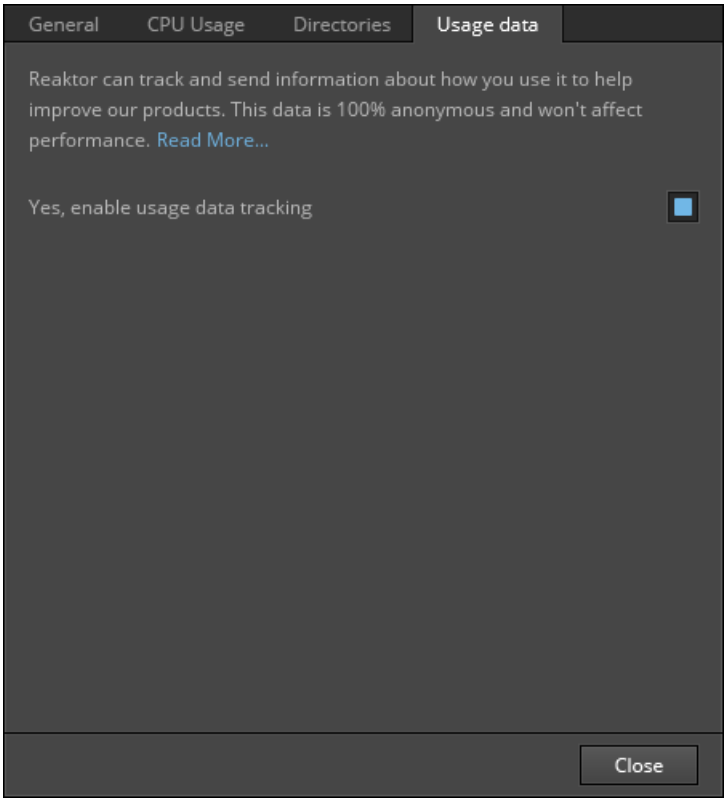
More information on the Compiled Core Cell Code Cache can be found in the REAKTOR 6 Building in Primary document available from the Help menu.

User Presets

This folder is the default directory for KOMPLETE KONTROL and MASCHINE user content. The path is not editable, but gives you the information required to add user content to the MASCHINE and KOMPLETE KONTROL Browser. After the folder is scanned by MASCHINE or KOMPLETE KONTROL, you can access the content from the user content area in the MASCHINE and KOMPLETE KONTROL Library Pane. More information about preparing Ensembles for KOMPLETE KONTROL and MASCHINE can be found in the Building in Primary document.

7.4 Usage Data

► Click on the Usage Data tab to view the Usage Data Preferences.



The Usage Data Tab

REAKTOR tracks anonymous usage data and sends that data to Native Instruments so that they can improve future updates by seeing which features are most often used and in which environments REAKTOR is most commonly used.

- In the Usage Data tab, click on the Read More... link to be directed to the Native Instruments webpage with more information on Usage Data tracking.
- You can opt in or out of Usage Data tracking by clicking on the checkbox labelled Yes, enable usage data tracking.

Data Being Tracked

The following data will be transmitted by Usage Data Tracking:

- System profile (your Operating System, Audio Drivers, etc)
- Current location (country and city)
- Data about your interaction with the software (e.g. actions you perform or audio interfaces you connect)

Data Not Being Tracked

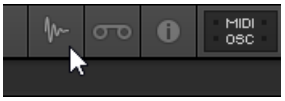
The following data will not be gathered by Usage Data Tracking:

- Personal data (Name, email address, payment details, serial numbers, etc.)
- User content (file names being referenced, samples, songs, tags, comments, mappings etc.)
- Any user activity outside of the NI application

8 The Sample Map Editor

If the Ensemble you are working with includes any of the Sampler Modules, you will be able to access the Sample Map Editor.

When this is the case, a button with a waveform icon will appear on the right side of the Toolbar, beside the Player/Recorder button and the Info Hints button.



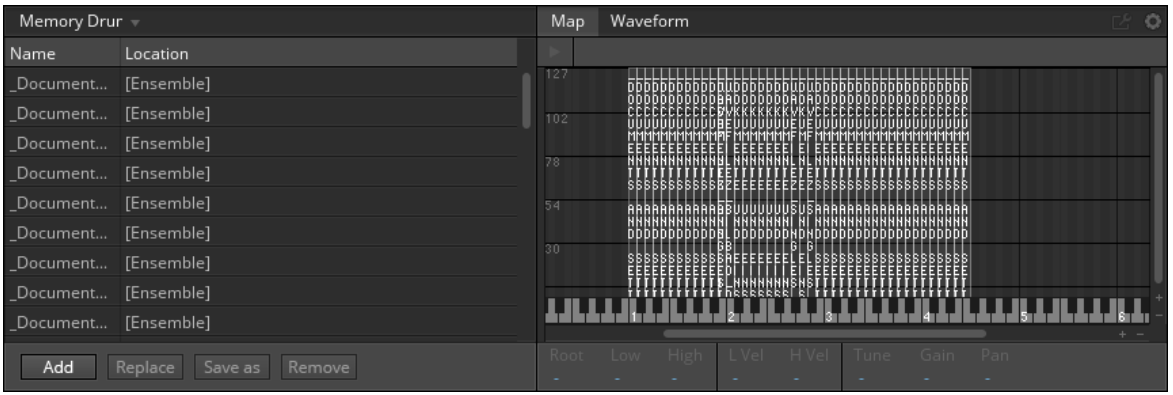
The Sample Map Editor Button

► Clicking on this icon will open the Sample Map Editor.

The Sample Map Editor is where you manage and edit the samples used by the Ensemble.



For the purposes of illustration, this chapter will look at the Memory Drum 2 Ensemble, which is available from the REAKTOR Factory Library, in the Samplers folder. Memory Drum 2 is a good Ensemble to use if you want to learn how the Sample Map Editor works.



The Sample Map Editor

The Sample Map Editor is divided into two sections:

- On the left is the **Sample List**.
- The right side can switch between the Map (key mapping view) and the Waveform (sample waveform view.

If certain functions are disabled in the context menu and the corresponding buttons are greyed out, the selected sample is protected. Find more information about this in section Third-Party and Protected Content in the REAKTOR 6 Getting Started document.

8.1 Sample Maps

Sample Maps are a format REAKTOR uses to contain a list of audio samples along with their key mapping Properties.

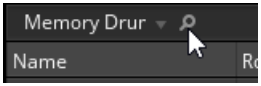
Sample Maps can be exported and imported from the Sample Map Editor, allowing you save and share different collections of samples and their mappings. Samples can even be embedded in the Sample Map file.

Each Sampler Module in an Ensemble can hold one Sample Map, but an Ensemble can contain several Sampler Modules and thus an Ensemble can contain more than one Sample Map.

► You can select between the different Sample Maps in the Ensemble from the dropdown menu above the Sample List.



► When in Edit Mode, the Magnifying Glass button to the right of the Sample Map selector menu is used to locate the Sampler Module that uses the currently selected Sample Map in the Ensemble Structure.



8.1.1 Importing and Exporting Sample Maps

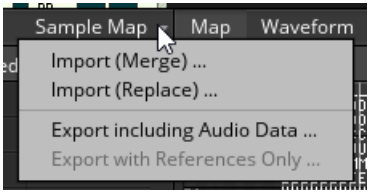
In order to import and export Sample Maps, you must activate Edit Mode.

► To activate Edit Mode, click on the EDIT button in the Toolbar.



The Sample Map import and export options are accessed from the Sample Map menu above the Sample List.

► Click on the Sample Map menu to display the Sample Map import and export options.



The Sample Map menu contains four options:

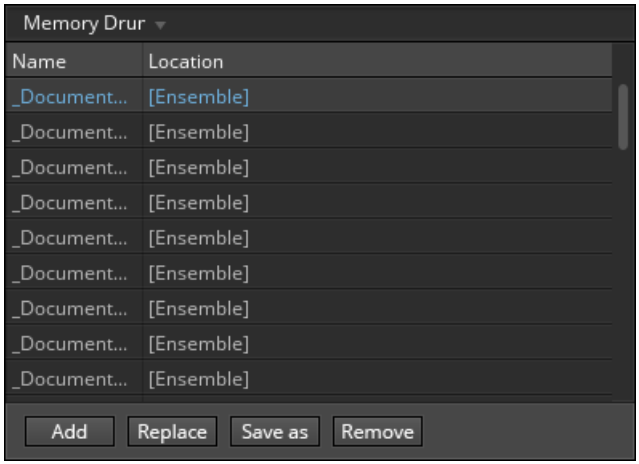
- *Import (Merge) ...*: Opens a dialog window from which you can locate and import a REAKTOR Sample Map file (a .map file). The imported Sample Map will be merged with the current Sample Map.
- *Import (Replace) ...*: The imported Sample Map will replace the previous Sample Map, and all audio data from the previous Sample Map will be deleted.
- *Export including Audio Data ...*: Opens a dialog window from which you can save the Sample Map as a .map file. The exported Sample Map will have all the necessary audio data embedded in the .map file.
- *Export with References Only ...*: This option is only available if all of the samples in the Sample Map are not embedded (i.e. their Embed option is unchecked). The exported Sample Map will act as a pointer to the external audio files.

8.2 The Sample List

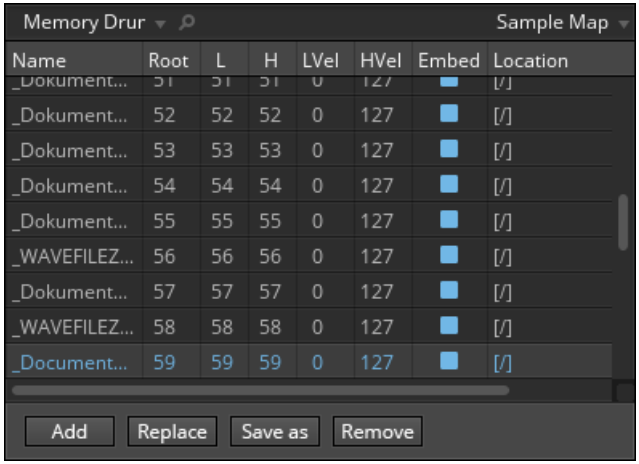
The Sample List is a table view of the samples used in the selected Sample Map.

The Sample List will have a different appearance depending whether you are using Edit Mode or Play Mode:

- When using the Sample List with REAKTOR in Play Mode, only the Sample Name and the Sample Location will be visible in the list.



- Activating Edit Mode expands the number of columns in the list to include the key mapping parameters for each sample, and the embed state.

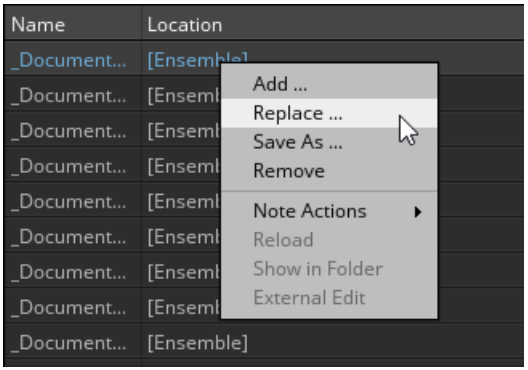


8.2.1 Editing the Sample List

Below the Sample List table are four buttons that can be used to perform quick edits to the Sample List:

- Add: Opens a dialog window for you to browse and load an audio sample file into the Sample List.
- Replace: Functions in a similar manner to the Add button, but the loaded audio sample will replace a selected sample in the Sample List. The newly loaded audio sample takes the key mapping settings of the sample it has replaced.
- Save as: Opens a dialog window for you to save a selected sample as an audio file.
- Remove: Deletes the selected sample from the Sample List.

These four options are also available from a menu that opens when you right-click on the Sample List.



The Sample List Menu

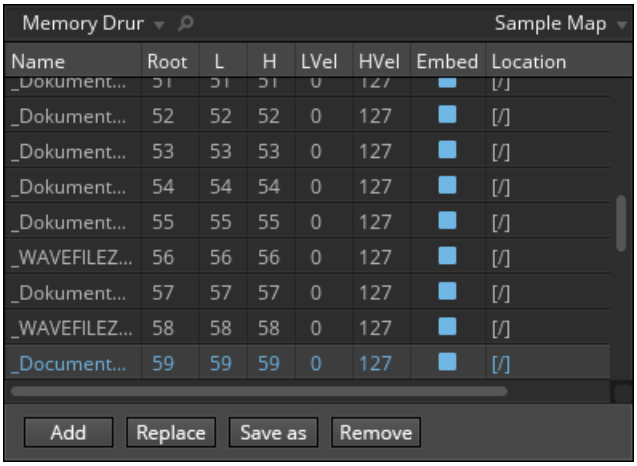
Editing the Mapping

In order to edit the mapping of the samples from the Sample List, you must activate Edit Mode.

► To activate Edit Mode, click on the EDIT button in the Toolbar.



When looking at the Sample List with Edit Mode active, it is possible to view and edit the key mapping of the samples.



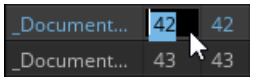
The Sample List when viewed in Edit Mode

There are five columns in the Sample List that relate to the key mapping:

- Root: The root MIDI note of the sample, i.e. the note at which the sample will play back at its original pitch.
- L (Low): The lowest MIDI note that will trigger the sample.
- H (High): The highest MIDI note that will trigger the sample.
- LVel (Low Velocity): The lowest velocity that will trigger the sample.
- HVel (High Velocity): The highest velocity that will trigger the sample.

Each parameter listed above can be edited from the table:

1. Double-click on the value you want to edit.



2. Type in a new value (0-127).
3. Press [Enter] or click away to set a new value.



When editing the key mapping from the Sample List, you can view your changes in the Key mapping View for a graphical representation of the Sample Map, which may be easier to read.

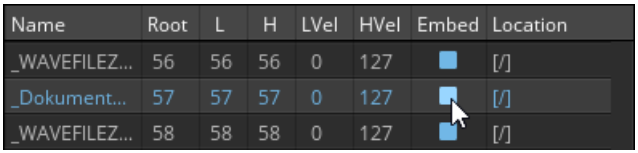
Referencing and Embedding Samples

Samples can be embedded in the Sample Map (meaning that the audio file itself is saved in the Sample Map and thus also the Ensemble) or they can be referenced from an external location.

When you add a sample to the Sample List, it will be embedded by default. Embedding samples ensures that the audio files will always be available to the Ensemble, since they are saved inside it. However, this might not always be desirable as this will also increase the file size of the Ensemble. Keeping audio files as external files also means that they can be shared by other programs, so you could have a KONTAKT Instrument referencing the same audio files as a REAKTOR Ensemble.

To set a sample to not be embedded in the Sample Map:

1. Activate Edit Mode by clicking on the EDIT button in the Toolbar.
2. In the Sample List, locate the sample you wish to edit.
3. Click on the button in the Embed column of the Sample List.



4. A dialog window will open, asking you to save a copy of the audio file.
5. Browse to the folder where you want to save the file and click Save.

→ The Ensemble will now reference the sample from the location specified when you saved the audio file.

Referencing samples as external files can be risky. Moving or deleting the audio file may cause REAKTOR to be unable to load the sample next time you open the Ensemble.

► If you wish to embed the sample again later, you can do this by clicking on the Embed button again.

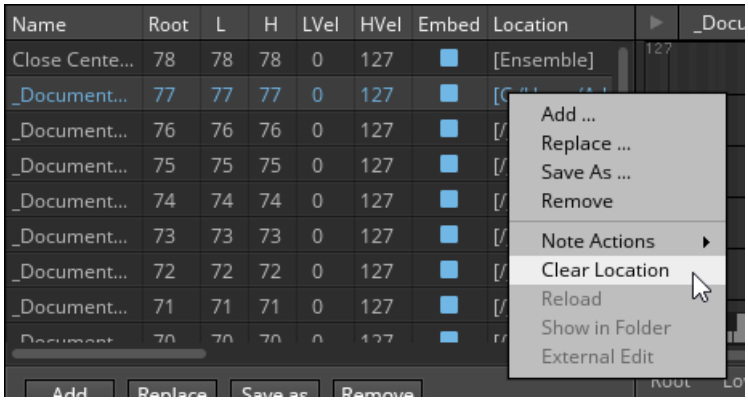
The Sample Location

When you add a sample to the Sample List, the location of the file will be displayed in the Location column.

If the Ensemble you are working on is intended for public release, you may want to hide this information.

To hide the audio file location:

1. Select the sample you want to edit, or hold [Shift] to select multiple samples to edit at once.
2. Right-click on the selected sample(s) in the Sample List.
3. Select the *Clear Location* option from the menu.

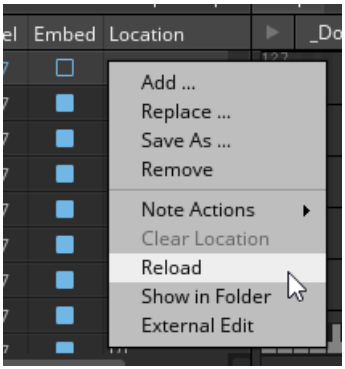


→ The information in the Location column of sample will change from the original audio file location to [Ensemble].

Clearing the sample location is only possible if the sample is embedded in the Sample Map.

Additional Options

The additional options in the right-click menu are only available if the selected sample is not embedded.



The right-click menu of a non-embedded sample

These options are:

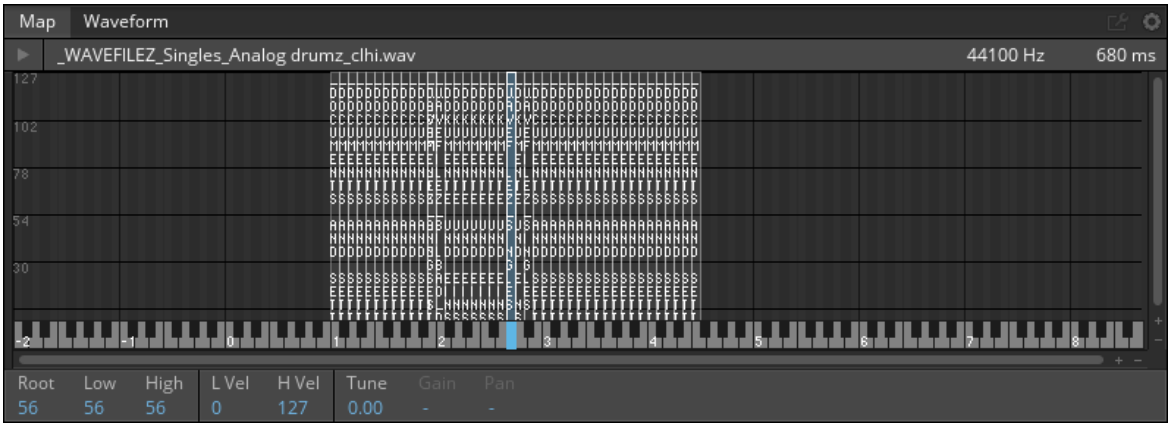
- *Reload*: Reloads the audio file from the hard disk to refresh the sample held in REAKTOR's memory. You may want to use this function if you have edited the audio file in an external audio editor.
- *Show in Folder*: Opens a file explorer window in your operating system at the location of the selected sample.
- *External Edit*: Opens the selected audio file in an external audio editing program. The external editor is set in the Directories tab of the Preferences Window, which is described in section ↑7.3, Directories Preferences ↑7.3, Directories Preferences.

8.3 The Map View

The Map View (key mapping) is located on the right side of the Sample Map Editor.

► If it is not currently displayed, click on the Map tab to display the Map View.





The Map View

The Map View displays a map of the samples according to their key and velocity ranges (the Map). The key range is mapped along the x axis, and the velocity range is mapped along the y axis. A sample can be set as a 2-dimensional zone in this map.

Above the Map is a display of information about the selected sample including (from left to right):

- The sample name.
- The sample rate.
- The length of the sample.

To the left of the sample name display is a play button.

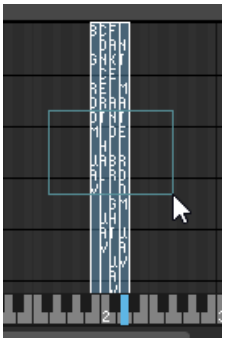
► Click on the play button to audition the sample.

Editing the Map (key mapping)

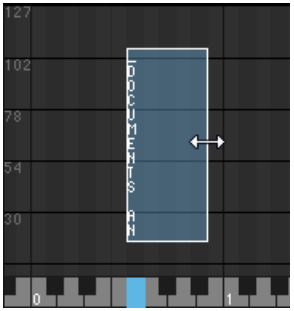
It is possible to edit the mapping parameters for a sample in the Sample Map by altering the size of the sample's zone in the Map.

► Select a sample for editing by clicking on its zone in the Map. A selected sample zone will be highlighted in blue.

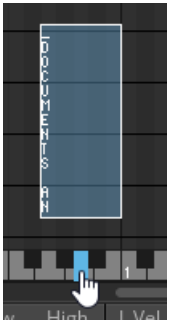
- Click and drag a selection box over the zones to select a group of samplesfor editing.



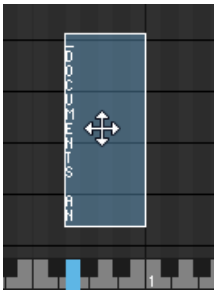
- Click and drag from the edge of the zone to alter the zone's height (velocity range) and width (key range).



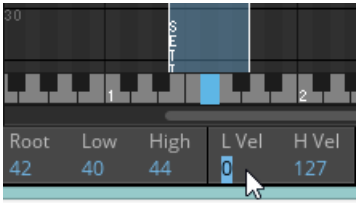
- Click and Drag on the blue note on the virtual keyboard to change a sample's root note.



- Move the whole zone to change all of these parameters at once.



- The mapping parameters can also be edited via the values below the Map, either by clicking and dragging on them, or by double-clicking on them and entering a new value.

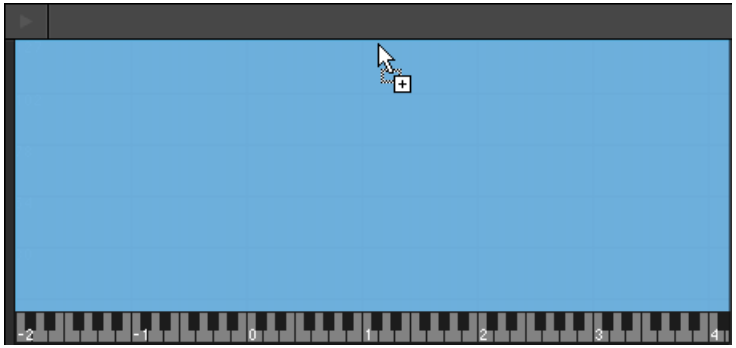


Adding and Removing Samples

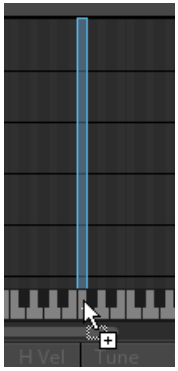
To add samples to the Map, you can drag and drop them from a file browser into the main Map area.

Where you drag the samples will specify their mapping parameters:

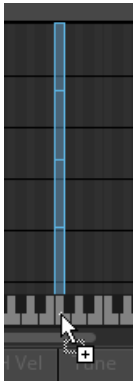
- Drag to the top of the Map to map the samples to all keys and velocities, with a root note of middle C (MIDI note 60).



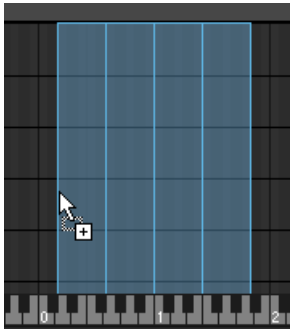
- Drag to a single key to set all the mapping parameters to that key.



- Drag several samples to a single key to map them to that key. This will distribute them evenly between different velocity ranges.



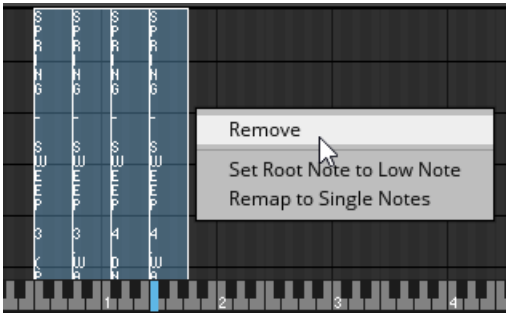
- Drag to any location in between to map the sample(s) with varying key-range distances. The key-range will depend on the location of the mouse in the Map.



To delete samples from the Map view:

► Select the sample(s) you want to remove and press the [Delete] key on your keyboard.

► Select the sample(s) you want to remove, right-click in the Map area, and select *Remove* from the menu.



8.4 Waveform View

The Waveform view is located on the right side of the Sample Map Editor.

► If it is not currently displayed, click on the Waveform tab to display the Sample Waveform View.





The Waveform view

The Waveform view shows the waveform of the selected sample and contains a handful of controls related to the playback of the sample.

Above the main waveform is a display of information about the selected sample including (from left to right):

- The sample name.
- The sample rate of the sample.
- The length of the sample.

To the left of the sample name display is a play button.

► Click on the play button to audition the sample.

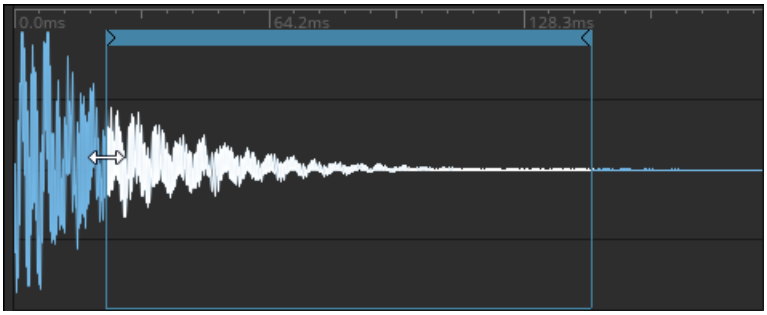
Below the waveform are a series of controls:

- Reverse: Toggles reverse playback of the sample.
- Loop: Toggle looping of the sample playback on or off.
- Start: Sets the loop start point.
- End: Sets the loop end point.
- Alternate: Activates alternating loop mode, in which the playback will alternate between forwards and reverse modes while looping.
- Release: Checking this option makes the loop continue even after releasing the note.
- Phase Align: This option is only available if the Sample Map is used in one of the Grain Sampler Modules. It applies signal analysis to align the phases of the grains. This can improve the sound quality, but requires more CPU resources.

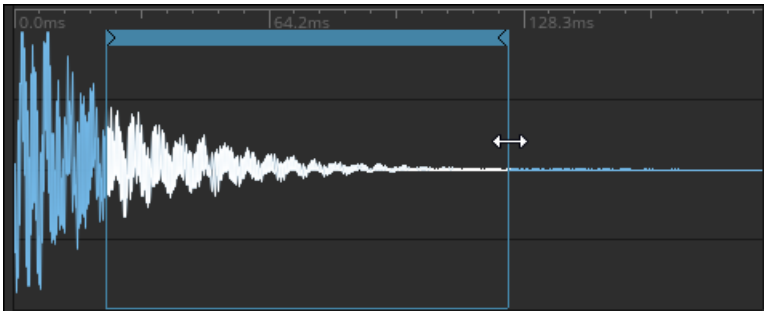
Editing Loop Points

You can edit the Start and End positions of a loop from the controls below the waveform. However, it is also possible to set the loop points on the waveform itself.

► Click and drag on the left side of the blue box to set the loop start point.



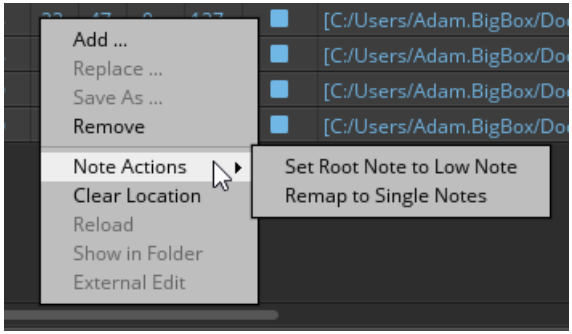
► Click and drag on the right side of the box to set the loop end point.



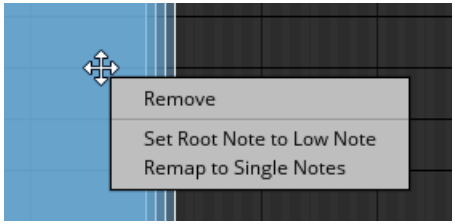
8.5 Note Actions

Note actions are a way of automatically mapping one or more samples. They can be used to speed up the sample mapping process.

- In the Sample List, the Note Actions are located in the right-click-menu, under the Note Actions sub-menu.



- In the Map view, the Note Actions are located in the main right-click menu.

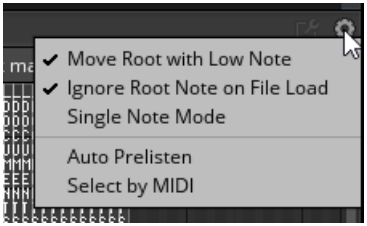


There are two note actions:

- *Set Root Note to Low Note*: Click on this option to set the root note value to the low note value for all selected samples.
- *Remap to Single Notes*: Selecting this option remaps the samples so that each sample is on a single key. The keys are selected in such a way that the samples are mapped chromatically (on consecutive keys). Clicking on this option remaps all samples, regardless of how many you currently have selected. With a single sample selected, the root note of that sample will be used as the note from which the mapping will begin (in other words, it will be set as the lowest mapped key).

8.6 Sample Map Editor Settings

► Click on the cog icon in the top right corner of the Sample Map Editor to open The Sample Map Editor Settings menu.



The Sample Map Editor Settings menu contains five options that can be activated or deactivated by clicking on their entry.

The first three options all relate to the mapping behavior:

- *Move Root with Low Note*: With this option selected, changing the low note of the sample key range via the numeric editors will automatically match the root note to the same value. This option does not affect the behavior of the Map area.
- *Ignore Root Note on File Load*: Certain audio files can have root note meta-data. REAKTOR can read this meta-data to map the root note of the sample automatically, but with this option selected, REAKTOR will ignore this meta-data.
- *Single Note Mode*: With this option selected, changing the low note (L) in the Sample List will also automatically update the Root note and the high note (H) to the same value.

The next two options control aspects of how the Sample Map Editor behaves when selecting Samples:

- *Auto Prelisten*: With this option selected, clicking on a sample in the Sample List or the Map will trigger playback of the sample.
- *Select by MIDI*: When active, playing a MIDI note will automatically select the sample mapped to that note. If multiple samples are mapped to a single note, the sample that was most recently added to the Sample Map will be selected.

Having both of these options selected at once can cause a double-triggering of the sample when playing the MIDI keyboard.

8.7 External Editor

If you are mapping samples in REAKTOR and realize that one or more of the audio files need edited in another program, you can open the samples in that program from inside REAKTOR.

The application that REAKTOR opens can be defined in the Directories tab of the Preferences (see section ↑7.3, Directories Preferences ↑7.3, Directories Preferences for more information).

To the left of the Sample Map Editor Settings menu is a button that opens an external sample editor.

► Click on the wrench icon to open the external sample editor.



The external editor can only be opened if the selected sample is not embedded in the Sample Map.

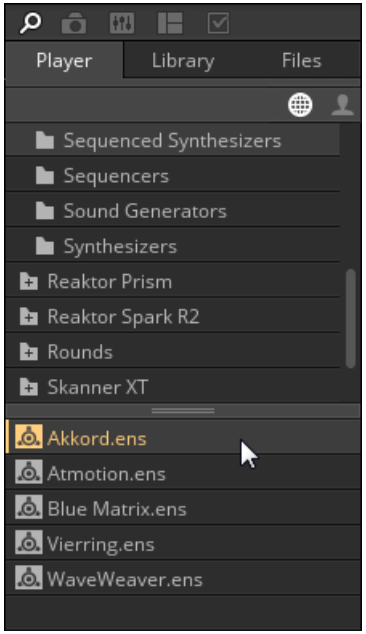
9 Panel Layouts

If you are using an Ensemble that contains more than one Instrument, you can move these Instruments around into a setup that suits you.

The REAKTOR 6 Getting Started document covers the A/B Views of an Instrument, but this section will look into altering the Panel View further.

For the purposes of illustration, this document will use the *Akkord* Ensemble from the Factory Library, as it uses two Instruments (one for the sequencer, and one for the Synthesizer).

1. To edit Panel Layouts, you must use REAKTOR in Edit Mode. To activate Edit More, click on the EDIT button in the Toolbar.
2. Load the *Akkord* Ensemble from the Factory Library; it is located in the Sequenced Synthesizers section.



3. Click on the Header of the Lower Instrument (the synthesizer).



4. Drag it over the sequencer Instrument.



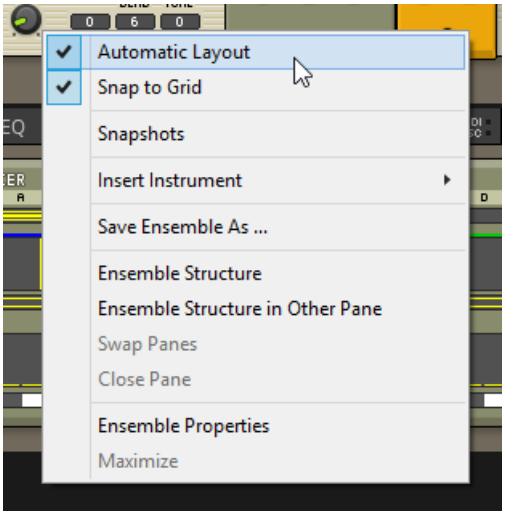
5. Release the synthesizer and it will be placed above the sequencer.



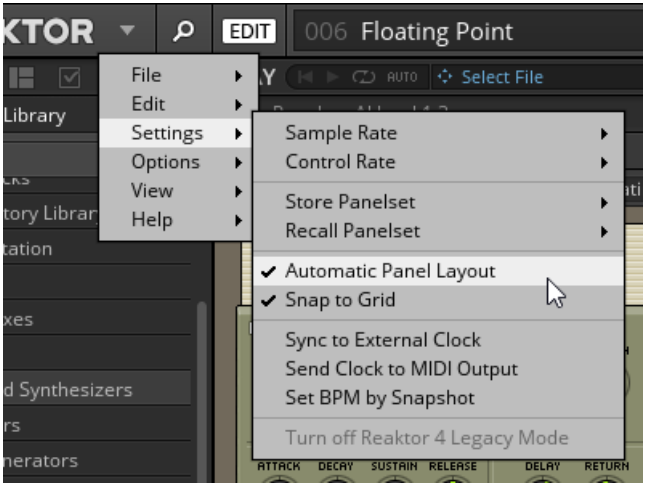
Notice that as you drag and drop the Instruments, they lock into place. This behavior is called **Automatic Layout**, and can be deactivated.

1. To deactivate Automatic Layout, right-click on an empty space in the Panel View.

2. Uncheck the *Automatic Layout* option from the menu by clicking on it.



3. This option is also found in the Settings menu as *Automatic Panel Layout*.



→ With this option unchecked, when you move an Instrument, it will remain wherever you drop it and will no longer lock into place.

10 Panelsets

Panelsets are a way of saving and recalling REAKTOR Panel Layouts. They are especially useful when an Ensemble includes more than one Instrument, as they can be used to quickly cycle between different panel configurations of the Instruments.

Panelsets can be viewed and edited from the Side Pane.

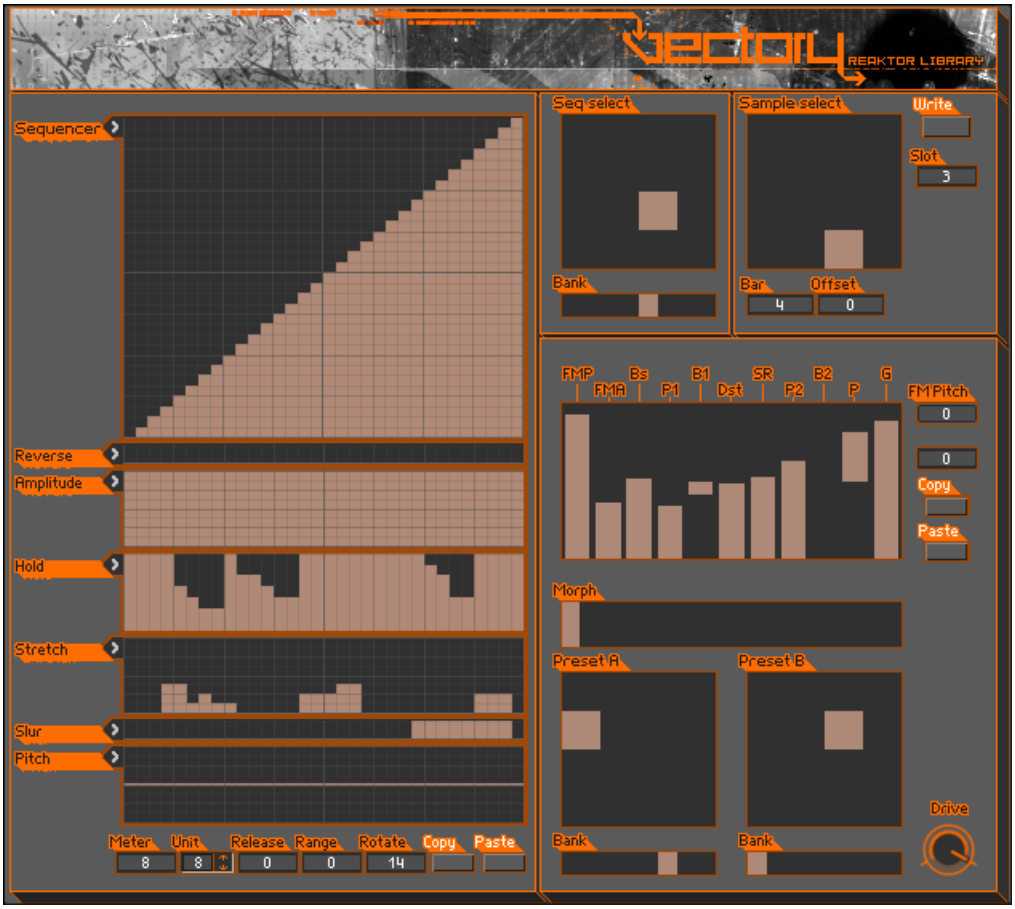
► To view the Panelsets, click on the Panelset tab in the Side Pane.



10.1 Recalling Panelsets

An Ensemble in the REAKTOR Factory Library that makes good use of the Panelsets is *Vectory*, so it will be used to illustrate how to recall Panelsets.

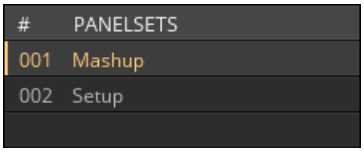
► Load *Vectory* from the Grooveboxes section of the Reaktor Factory Library.



The Vectory Ensemble

Vectory manipulates samples to produce its sound, but when you first load the Ensemble, it appears as if there is no way to load samples into the Ensemble from the interface.

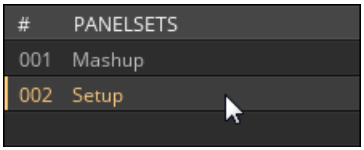
However, looking at the Panelsets, you will see that Vectory has two Panelsets: Mashup and Setup.



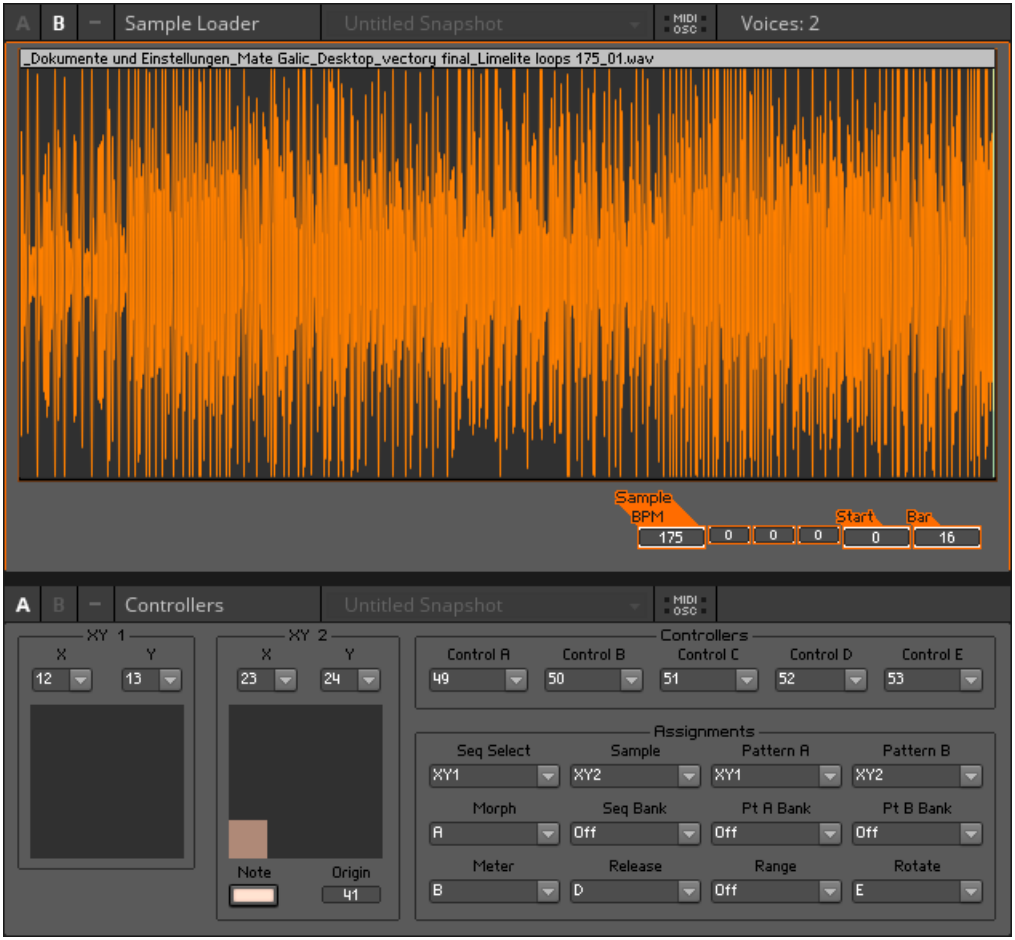
Vectory's Panelsets

The initial view that loads with the Ensemble is the Mashup Panelset.

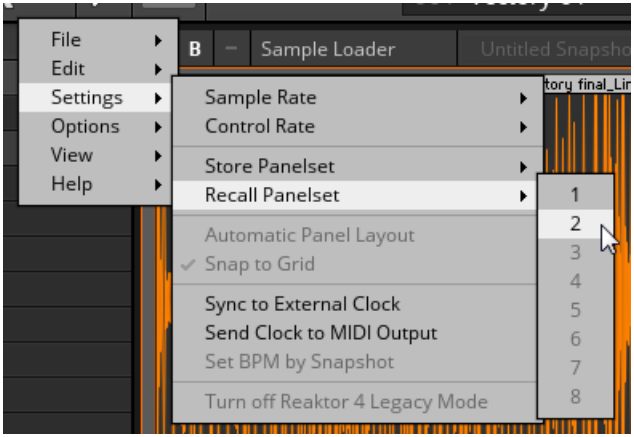
► Click on the Setup entry in the Side Pane to load the Setup Panelset.



→ The Panel changes to display different Instruments: a Sample Loader and a MIDI Controller Setup Instrument.



► You can also recall Panelsets from the Settings menu, by clicking on the relative entry in the *Recall Panelset* sub-menu.



10.2 Creating and Editing Panelsets

In order to create or edit Panelsets, you need to have REAKTOR in Edit Mode.

► To activate Edit Mode, click on the EDIT button in the Toolbar.



In Edit Mode, there are three additional buttons below the Panelset list:

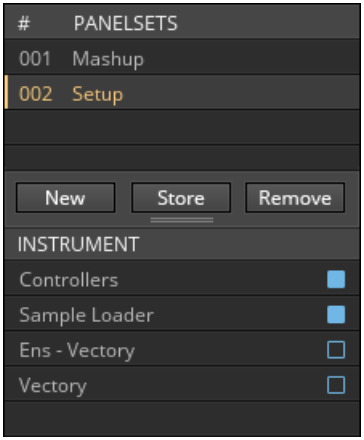


The Panelset Buttons

These buttons allow you to create, save or delete Panelsets.

- New: Creates a new Panelset that stores the current Panel configuration.
- Store: Overwrites the selected Panelset with the current Panel configuration.
- Remove: Deletes the selected Panelset.

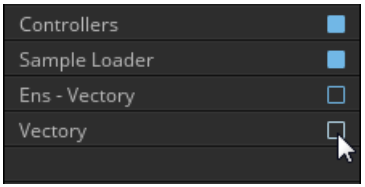
Below the Panelset buttons you will see a list of the Instruments in the loaded Ensemble.



The Instruments in Vectory

To edit and store a Panelset:

1. Show and hide Instruments by clicking on the buttons beside their name.



2. You can then arrange the Instruments on the Panel.
3. When you are finished, save the Panelset clicking on the New button.
4. Rename the new Panelset by double-clicking on its name and typing a new name.



5. Press [Enter] or click away to set the new name.

→ The newly created Panelset will be available for recall.



Remember to save the Ensemble file to make these changes permanent.

What is Stored in a Panelset

Panelsets store the following information:

- The visibility of each Instrument in the Ensemble.
- The Header visibility of each Instrument and the Ensemble.
- The A/B view selection and minimized/maximized state of the Instruments.
- The positions of the Instruments on the Panel.
- The Snap to Grid and Automatic Layout Panel settings.

11 An Introduction to Building

The following sections will introduce you to the Structure View, giving you grounding in the skills needed to start building in REAKTOR.

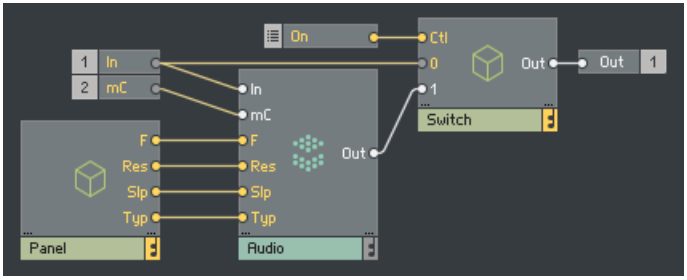
If you are interested in learning to build instruments and effects in REAKTOR, it is suggested that you read these sections and then move on to the REAKTOR 6 Building in Primary document available from the Help menu.



All of the following sections require REAKTOR to be in Edit Mode.

Building in REAKTOR is done in the **Structure View**.

In the Structure View you can view the building blocks of a REAKTOR Ensemble (called **Macros**, **Modules**, and **Core Cells**). These building blocks are connected by wires.



A REAKTOR Structure

REAKTOR's programming environment is analogous to a signal flow diagram, in which the blocks represent functions and the wires represent signals going into or coming out of these functions.

In simple terms, REAKTOR works a little like a modular system, or a rack. You can create an oscillator, which you then connect to a filter via a wire. A keyboard can be connected to the oscillator via another wire, and so on.

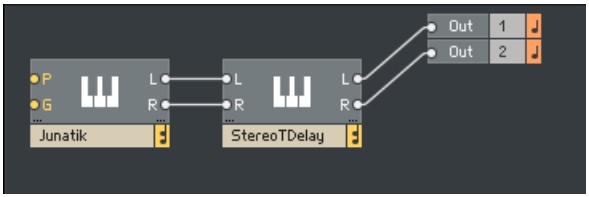
11.1 Navigating the Structure View

Navigating a structure in REAKTOR is almost always accomplished by double-clicking.

1. Load an Ensemble (for the purposes of illustration, this document will show the *Junatik* Ensemble from the Factory Library)
2. [Ctrl] + [Shift] + double-click ([Cmd] + [Shift] + double-click on OS X) on a blank space in the Panel View.



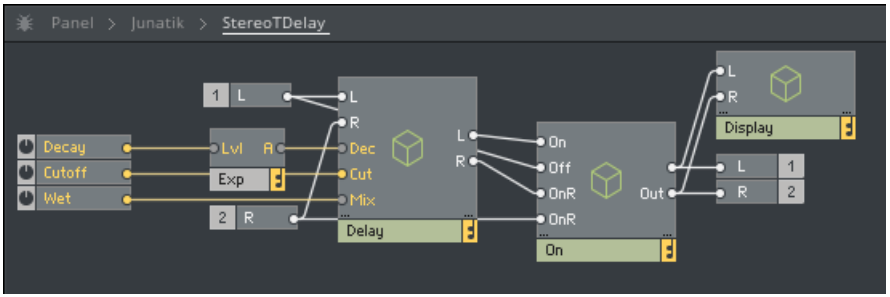
→ The first level of the Ensemble Structure will be displayed in the Structure View.



In the first level of the Ensemble Structure you will see Instruments as blocks. These Instruments contain deeper levels of the Structure.

Instruments can usually be identified by the keyboard icon.

► Double-click on an Instrument to view its Structure.



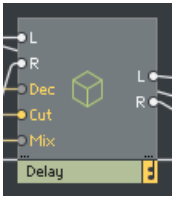
The StereoTDelay Instrument Structure

Holding [Ctrl] + [Shift] or [Cmd] + [Shift] when double-clicking is only necessary to enter the first level of the structure, after that all navigation between levels only requires a double-click.

In the next level down you will see more blocks, with wires running between them. These are **Macros** and **Modules**.

- **Modules** are the lowest level in REAKTOR Primary Structures, they cannot be edited and thus you cannot enter them.
- **Macros** are containers that can hold any number of Modules and other Macros. The contents of a Macro can be edited.

Macros are usually identified by their cube icon.



A Macro

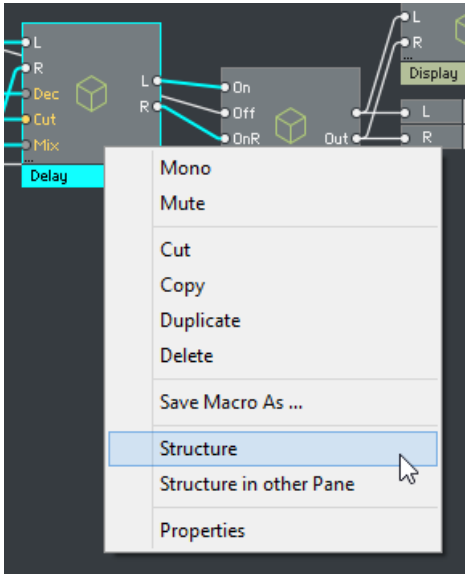


In some cases this cube can be replaced by another icon, though that doesn't occur in the Factory Library.

► Double-click on the Macro to enter its Structure.

Alternatively, you can enter Macro structures using the right-click menu:

1. Right-click on a Macro to open the menu.
2. Click on the *Structure* option.



→ The Structure of the Macro will be displayed in the Structure View.

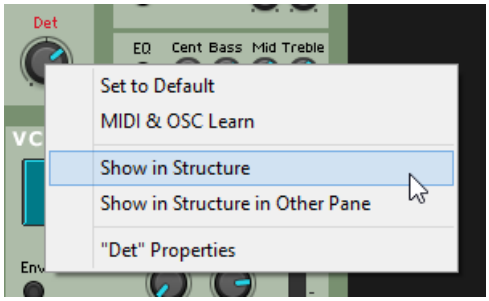
You can continue to navigate deeper and deeper in the Structure View using these methods until you reach a Macro that contains only Modules.

► To return one level higher, double-click on an empty space in the structure.

► There are also some shortcuts to enter specific places in the Ensemble Structure.

1. Return to the Panel View of the Ensemble.

2. Right-click on a knob to open the menu.



The menu contains the following entries that relate to navigation:

- *Show in Structure*: Opens the Structure View at the level in which the Knob Module can be found. The Knob Module will be highlighted.
- *Show in Structure in Other Pane*: Functions like *Show in Structure*; only this option activates the Screen Split and opens the Structure View in the Other Pane, allowing you to view the Panel and the Structure at the same time.

11.2 Navigation Tools

11.2.1 Split Screen

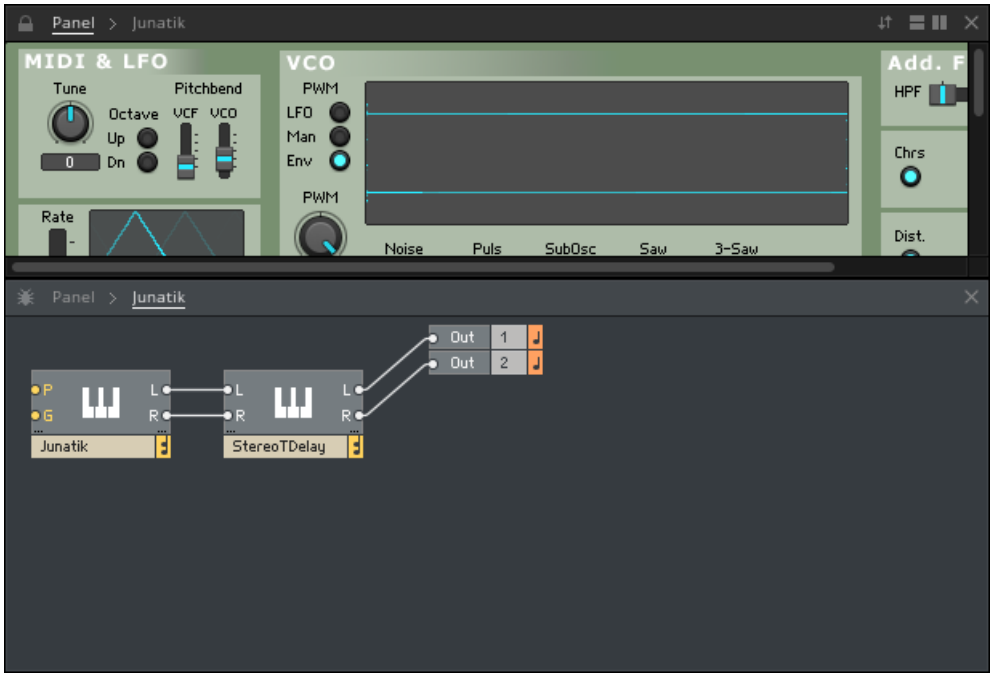
Having read the previous section, you have already been introduced to the Split Screen.

The Split Screen allows you to see two parts of the Ensemble Structure at the same time by dividing the screen into two panes.

► Activate Split Screen by clicking on either of the rectangle icons to the top right of the Structure View.



- The left icon splits the screen horizontally.



- The right icon splits the screen vertically.

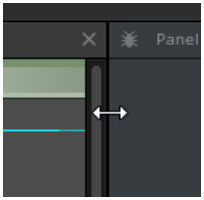


When Split Screen is active, you can swap the contents of the Panes, alter the split point, and close the Split Screen:

- Click on the arrow icon to swap the Panes.



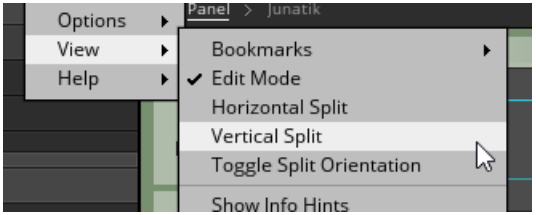
- Click and drag the central column to alter the split point.



- Click on the X to the top right of a Pane to close that Pane and thus close the Split Screen.



These Split Screen options can also be found in the View menu.

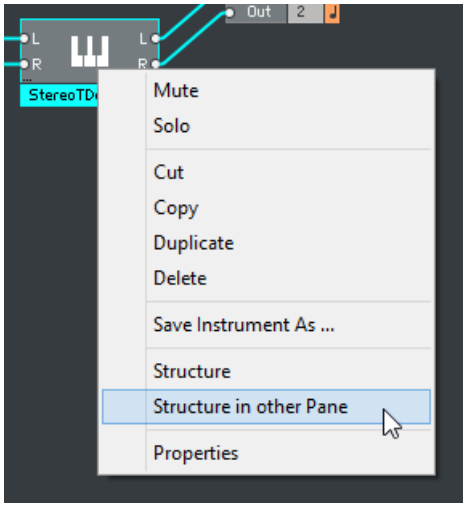


Split Screen options in the View menu.

You can also open the Split Screen, or navigate between the Panes, by using the right-click menus:

1. Right-click on a Macro or Instrument.

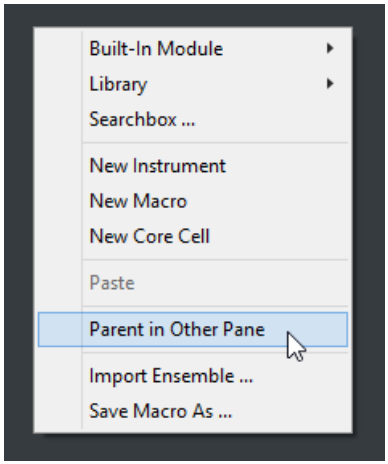
2. Select *Structure in other Pane*.



→ The Structure of the Macro or Instrument will be displayed in the other Pane.

Inside a Macro, you can also navigate upwards in this way:

1. In a Macro Structure, right-click on an empty space.
2. Select *Parent in Other Pane*.

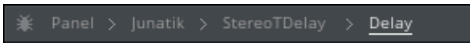


→ The Structure of the level above the current Macro Structure will be displayed in the other Pane.

11.2.2 Breadcrumbs

Breadcrumbs give you a map to your current location in the Structure. They show you the names of all of the Instruments and Macros you have entered to get to your current location.

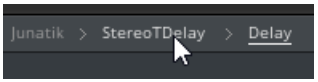
Breadcrumbs are located in the bar above the structure view.



Breadcrumbs

Your current location will be underlined.

► Click on a Breadcrumb to jump to that location in the Structure.



11.2.3 Bookmarks

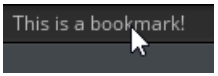
Bookmarks allow you to save your current location to recall later. If you find yourself navigating back and forth between the same locations in the Structure, Bookmarks can make this process a lot easier.



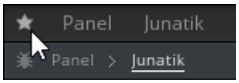
Bookmarks also save the Split Screen settings, so they can be used to save and recall workspace setups.

Bookmarks are located above the Breadcrumbs.

► Click on a Bookmark to recall its location settings.

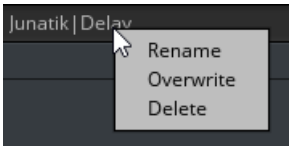


► Click on the star icon to create a new Bookmark of your current location settings.



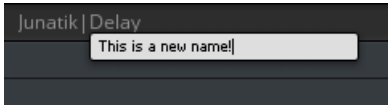
→ A new Bookmark will appear in the Bookmark bar.

► Right-click on a Bookmark to edit it.



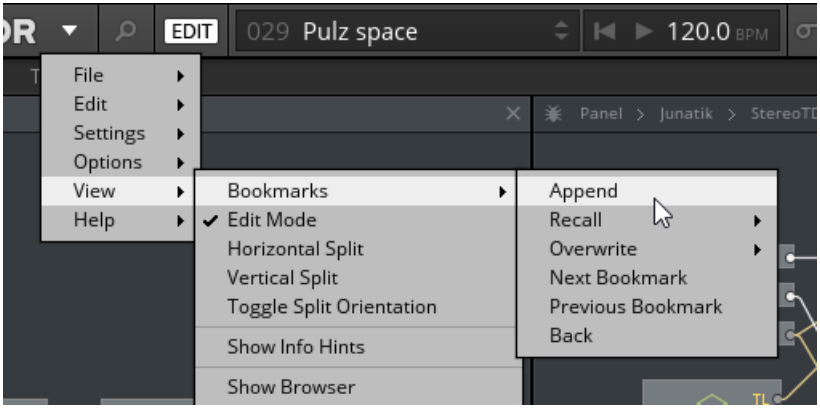
The Bookmark menu contains three options:

- *Rename*: Selecting this option will open a text box, into which you can enter a new name for the Bookmark.



- *Overwrite*: Saves the current location settings in the selected Bookmark.
- *Delete*: Removes the selected Bookmark.

Bookmark options can also be found in the View menu in the Bookmarks sub-menu.



The Bookmarks sub-menu

The options available mostly mirror those available in the Bookmark bar:

- *Append*: Creates a new Bookmark, like clicking on the star icon.
- *Recall*: Opens a sub-menu from which you can select a Bookmark to recall.
- *Overwrite*: Opens a sub-menu from which you can select a Bookmark to overwrite with the current location.
- *Next Bookmark*: Recalls the Bookmark after the last Bookmark you recalled.
- *Previous Bookmark*: Recalls the Bookmark in the list before the last Bookmark you recalled.
- *Back*: Returns to your previous location settings.

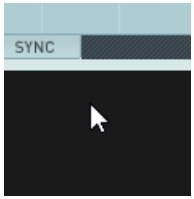
11.3 Loading and Connecting Instruments

An Ensemble can be made up of multiple Instruments, which can be connected together or just run side by side in a single instance of REAKTOR.

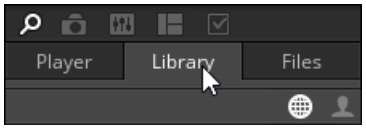
In the following section you will learn how to load and connect Instruments in an Ensemble.

1. Load a Synthesizer Ensemble. For the purposes of illustration, this document will use *Carbon 2* from the Factory Library.

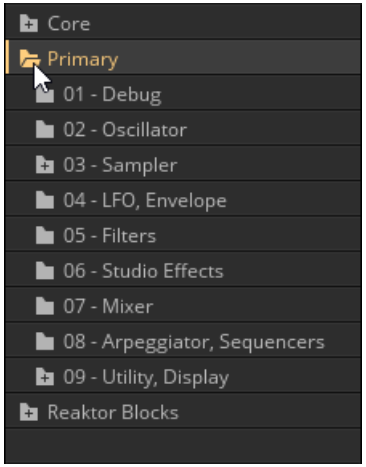
2. Enter the Ensemble Structure by [Ctrl] + [Shift] + double-clicking ([Cmd] + [Shift] + double-click on OS X) on an empty space outside the Instrument Panel.



3. Open the Library Browser in the Side Pane by clicking on the Library tab.



4. Open the Primary folder to view its contents.



5. Select the 06 - Studio Effects folder.
6. Drag and Drop one of the ism (Instrument) files from the lower half of the Browser into the Structure.



→ The selected effect Instrument will appear as an object in the structure.

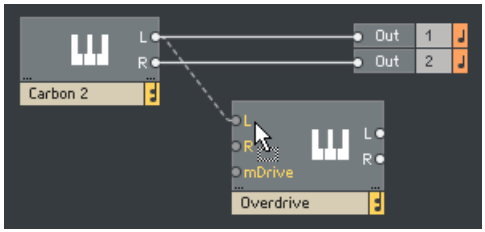


The Ensemble now contains two Instruments, one synthesizer and one effect. The next step will be to connect them.

1. Move the Instrument you just loaded by clicking on it and dragging it to where you want it.

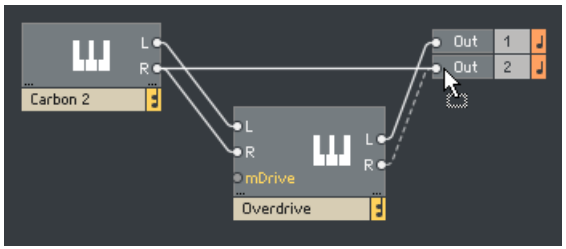


2. To connect two ports, click on one port and drag a wire to the port you want to connect it to.

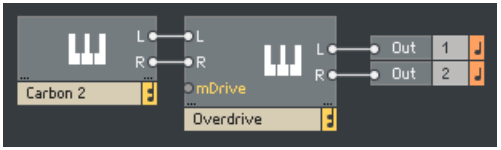


3. Inputs are located on the left side of any object in the structure, and outputs are on the right side. Connect the L and R output ports of the synthesizer to the L and R input ports of the effect.

4. Connect the output ports of the effect to the main output Modules of the Ensemble (the two objects labelled Out on the right side of the Structure).



→ The structure should now look something like this.



When you return to the Panel View, you will see the effect you loaded below the synthesizer.



Carbon 2 with the Overdrive effect

You can also import Ensembles as Instruments by dragging them from the Browser into the Structure.

11.4 Learning More

If you are interested in learning more about building in REAKTOR read the REAKTOR 6 Building in Primary document available from the Help menu. It contains tutorials for creating instruments and effects that teach the fundamentals of building, as well as guides to the advanced features of REAKTOR.

After working through that document, and if you are interested in diving deeper into building, you can move on to the REAKTOR 6 Building in Core document. The Core level of REAKTOR is a programming environment for building your own algorithms.

12 Keyboard Shortcuts

Views

Function	Shortcut (Mac OSX)	Shortcut (Windows)
Browser	[F1]	[F1]
Snapshots	[F2]	[F2]
Connection Manager	[F3]	[F3]
Panelsets	[F4]	[F4]
Properties	[F5]	[F5]
Sample Map Editor	[F6]	[F6]
Player and Recorder	[F7]	[F7]
Preferences	[Cmd] + [,]	[Ctrl] + [,]
Audio & MIDI Settings	[Cmd] + [.]	[Ctrl] + [.]
OSC Settings	[Cmd] + [Alt] + [.]	[Ctrl] + [Alt] + [.]
Hide/show Horizontal Split	[F10]	[F10]
Hide/show Vertical Split	[F11]	[F11]
Full Screen	[F12]	[F12]

General

Function	Shortcut (Mac OSX)	Shortcut (Windows)
Run/Stop Audio Engine	[Cmd] + [R]	[Ctrl] + [R]
MIDI/OSC Learn	[Cmd] + [M]	[Ctrl] + [M]
Start/Stop Clock	[Space]	[Space]
Reset Clock	[Alt] + [Space]	[Ctrl] + [Space]
New Ensemble	[Cmd] + [N]	[Ctrl] + [N]
Open Ensemble	[Cmd] + [O]	[Ctrl] + [O]

Function	Shortcut (Mac OSX)	Shortcut (Windows)
Save Ensemble	[Cmd] + [S]	[Ctrl] + [S]
Save Ensemble As	[Cmd] + [Shift] + [S]	[Ctrl] + [Shift] + [S]
Quit	[Cmd] + [Q]	[Alt] + [F4]

Snapshots

Function	Shortcut (Mac OSX)	Shortcut (Windows)
Next	[Down Arrow]	[Down Arrow]
Previous	[Up Arrow]	[Up Arrow]
Jump Forward 10 Snapshots	[Page Down]	[Page Down]
Jump Back 10 Snapshots	[Page Up]	[Page Up]

Bookmarks

Function	Shortcut (Mac OSX)	Shortcut (Windows)
Create New	[Alt] + [`]	[Alt] + [`]
Recall 1…8	[Alt] + [1]…[8]	[Alt] + [1]…[8]
Overwrite 1…8	[Alt] + [1]…[8]	[Alt] + [1]…[8]



The [`] key is system specific for English keyboards. The Create New Bookmark shortcut uses [Alt] + the key to the left of 1.

Help/Debug

Function	Shortcut (Mac OSX)	Shortcut (Windows)
Info Hints	[Alt] + [I]	[Alt] + [I]
Measure CPU Usage	[Alt] + [C]	[Alt] + [C]
Show Module Sorting	[Alt] + [S]	[Alt] + [S]
Show Event Init Order	[Alt] + [R]	[Alt] + [R]
Wire Debugging	[Alt] + [D]	[Alt] + [D]

Structure/Panel

Function	Shortcut (Mac OSX)	Shortcut (Windows)
Enter Structure from Panel	[Cmd] + [Shift] + Double Click	[Ctrl] + [Shift] + Double Click
Open Searchbox	[Enter]	[Enter]
Cut	[Cmd] + [X]	[Ctrl] + [X]
Copy	[Cmd] + [C]	[Ctrl] + [C]
Paste	[Cmd] + [V]	[Ctrl] + [V]
Duplicate	[Cmd] + [D]	[Ctrl] + [D]
Delete	[Backspace]	[Backspace]
Select All	[Cmd] + [A]	[Ctrl] + [A]
Undo	[Cmd] + [Z]	[Ctrl] + [Z]
Redo	[Cmd] + [Y]	[Ctrl] + [Y]
Lock/Unlock Panel	[Cmd] + [E]	[Ctrl] + [E]
Move Panel Module by 1 Pixel	[Cmd] + [Shift] + [Arrow Key]	[Ctrl] + [Shift] + [Arrow Key]
Move Panel Module by 40 Pixels	[Shift] + [Arrow Key]	[Shift] + [Arrow Key]
Control Finetuning	[Shift] + Drag with Mouse	[Shift] + Drag with Mouse



You can also duplicate Modules and Macros by holding [Alt] as you drag with the mouse.

Advanced View Functions

Function	Shortcut (Mac OSX)	Shortcut (Windows)
Move Focus to Next Pane	[Tab]	[Tab]
Move Focus to Previous Pane	[Shift] + [Tab]	[Shift] + [Tab]
Pan in Structure View	[Alt] + Drag Background with Mouse	[Alt] + Drag Background with Mouse
Open Macro Structure in Other Pane	[Alt] + Double-click	[Alt] + Double-click

Function	Shortcut (Mac OSX)	Shortcut (Windows)
Navigate Through Breadcrumbs	[Alt] + [Left Arrow] and [Alt] + [Right Arrow]	[Alt] + [Left Arrow] and [Alt] + [Right Arrow]
Toggle Between Horizontal and Vertical Screen Split	[Ctrl] + [H]	[Ctrl] + [H]

Index

A

Akkord [90]

audio

- playback [52]
- record [53]

automation [35]

B

banks [48]

- creating and editing [49]

bookmark [109]

breadcrumb [109]

browser [62]

- library [64]
- player [63]

building [101]

- documentation [115]
- navigation [102]

C

Carbon 2 [111]

Connection Manager [23]

connections [113]

core [116]

- CCCCC (Compiled Core Cell Code Cache) [66]

CPU (Central Processing Unit) [61]

D

default value [51]

E

edit mode [13]

I

instruments

- connecting [113]
- loading [111]

J

Junatik [102]

K

Keyboard Shortcuts [117]

KOMPLETE KONTROL [66]

M

macro [103]

MASCHINE [66]

Memory Drum 2 [69]

MIDI & OSC Learn [22]

MIDI (Musical Instrument Digital Interface) [15]

- MIDI in [33]
- MIDI out [33]
- Properties [28]

module [103]

Monark [35]

O

OSC (Open Sound Control) [15]

- learn [22]
- Properties [28]
- setting up [16]

OSC Settings [16]

- Receive [18]
- Send [20]
- Sync [21]

P

panel

- automatic layout [92]
- layout [90]

panelsets [94]

- creating and editing [98]
- recall [94]

play mode [13]

player and recorder [52]

- file player [52]
- file recorder [53]
- recorder settings [55]

Preferences [57]

- CPU Usage [60]
- Directories [61] [66]
- General [57]

Properties

- Connect [26] [32]

R

Recorder Settings [55]

S

sample
- external sample editor [65] [89]
- importing [73] [81]
- location [77]
- loop points [85]
- mapping [74] [79]
- sample map [69]

sample map [70]
- embedding [76]
- importing and exporting [71]
- sample map editor [69]

sample map editor
- note actions [86]
- sample list [72]
- settings [88]
- waveform [84]

Shortcuts [117]

snapshot [39]
- bank [48]
- default values [51]
- deleting [44]
- master [40]
- Morph + Random [45]
- renaming [44]
- sorting [44]
- storing [43]

split screen [105]

structure view [102]

T

The Finger [35]

V

Vectory [94]



Manual

The information in this document is subject to change without notice and does not represent a commitment on the part of Native Instruments GmbH. The software described by this document is subject to a License Agreement and may not be copied to other media. No part of this publication may be copied, reproduced or otherwise transmitted or recorded, for any purpose, without prior written permission by Native Instruments GmbH, hereinafter referred to as Native Instruments.

“Native Instruments”, “NI” and associated logos are (registered) trademarks of Native Instruments GmbH.

Mac, Mac OS, GarageBand, Logic, iTunes and iPod are registered trademarks of Apple Inc., registered in the U.S. and other countries.
Windows, Windows Vista and DirectSound are registered trademarks of Microsoft Corporation in the United States and/or other countries.
All other trade marks are the property of their respective owners and use of them does not imply any affiliation with or endorsement by them.

Document authored by: Native Instruments GmbH

NATIVE INSTRUMENTS GmbH
Schlesische Str. 29-30
D-10997 Berlin
Germany
www.native-instruments.de

NATIVE INSTRUMENTS North America, Inc.
6725 Sunset Boulevard
5th Floor
Los Angeles, CA 90028
USA
www.native-instruments.com

NATIVE INSTRUMENTS K.K.
YO Building 3F
Jingumae 6-7-15, Shibuya-ku,
Tokyo 150-0001
Japan
www.native-instruments.co.jp

NATIVE INSTRUMENTS UK Limited
18 Phipp Street
London EC2A 4NU
UK
www.native-instruments.com



© NATIVE INSTRUMENTS GmbH, 2015. All rights reserved.

Table of Contents

1 Welcome to the REAKTOR Factory Library 17

2 Effects 18

2.1 Analogic Filter Box 18

2.1.1 Introduction 18

2.1.2 Structure and Signal Flow 18

2.1.3 Modulation 19

2.2 Anima 20

2.3 Banaan Electrique 23

2.3.1 Introduction 23

2.3.2 Getting Started 24

2.3.3 Structure and Signal Flow 24

2.4 Classic Vocoder 25

2.4.1 Introduction 25

2.4.2 Quick Start 25

2.4.3 Structure and Signal Flow 25

2.4.4 The Vocoding Engine 26

2.5 Cyan 26

2.6 Echomania 29

2.6.1 Introduction 29

2.6.2 Quick Start 29

2.6.3 Structure and Signal Flow 29

2.7 EnFX 31

2.7.1 EnFX Delay 1.3 32

2.7.2 EnFX Distortion 1.3 34

2.7.3 EnFX Filter 1.3 35

2.8 Fast FX 37
2.9 FlatBlaster 2 45
2.9.1 Introduction 45
2.9.2 What's New in 2.0.2? 46
2.9.3 Multi-band Compressor 46
2.9.4 Full-band Peak Limiter 47
2.10 Flatblaster 48
2.10.1 Introduction 48
2.10.2 Quick Start 49
2.10.3 Structure and Signal Flow 49
2.10.4 Frequency-Specific Compressor 49
2.10.5 Full-Band Peak Limiter 50
2.11 Fusion Reflections 51
2.11.1 Introduction 51
2.11.2 Quick Start 52
2.11.3 Structure and Signal Flow 52
2.11.4 Diffusion Delays 52
2.12 Grainstates FX 53
2.12.1 Introduction 53
2.12.2 Quick Start 54
2.12.3 Structure and Signal Flow 54
2.12.4 Additional Controls 55
2.12.5 MIDI Control 55
2.13 Longflow 56
2.14 Resochord 59
2.15 Space Master 2 65

2.15.1 Introduction 65
2.15.2 Input and Output Stage 66
2.15.3 Reflections 66
2.15.4 Frequency Response 67
2.16 SpaceMaster 68
2.16.1 Introduction 68
2.16.2 SpaceMaster 5.1 Surround 70
2.16.3 SpaceMaster Quad 71
2.16.4 SpaceMaster Stereo 72
2.17 Spring Tank 73
2.17.1 Introduction 74
2.17.2 Quick Start 74
2.17.3 Structure and Signal Flow 74
2.18 Two Knees Compressor 75
2.18.1 Introduction 75
2.18.2 Quick Start 76
2.18.3 Structure and Signal Flow 76
3 Grooveboxes 77
3.1 Aerobic 77
3.1.1 Sound Engine 78
3.1.2 Sequencer 80
3.1.3 Master / Mixer 81
3.2 Aerobic 84
3.2.1 Introduction 85
3.2.2 Sound Engine 85
3.2.3 Sequencer 87

3.2.4 Master/Mixer 88
3.3 GoBox 91
3.3.1 Introduction 91
3.3.2 Quick Start 92
3.3.3 Structure and Signal Flow 92
3.3.4 The Sequencer 93
3.3.5 Morph 94
3.4 Krypt 95
3.5 L3 103
3.5.1 Introduction 103
3.5.2 Pattern Sequencer 104
3.5.3 Step Sequencer 104
3.5.4 Sampler 106
3.6 Limelite 108
3.7 Massive 1.1 120
3.7.1 Introduction 121
3.7.2 Control Section 121
3.7.3 Modulation Section 125
3.7.4 Sequencer 126
3.7.5 Sound Engine 129
3.7.6 Snapper 134
3.7.7 Output 137
3.8 Newscool 139
3.8.1 Introduction 140
3.8.2 Life Sequencer 140
3.8.3 Newscool 142

3.9 Random Step Shifter 144
3.9.1 Introduction 145
3.9.2 SQ2 145
3.9.3 Sampler 146
3.10 Rhythmaker 149
3.11 Scenario 157
3.11.1 Introduction 157
3.11.2 Quick Start 158
3.11.3 Structure and Signal Flow 159
3.11.4 Loading Samples 161
3.11.5 Tips and Tricks 161
3.12 Sinebeats 2 163
3.12.1 Introduction 163
3.12.2 Sequencer 164
3.12.3 Noise Synthesizer 165
3.12.4 Sine Synthesizers 166
3.12.5 FX 1 & 2 168
3.12.6 Mixer 169
3.12.7 EQ and Compressor 169
3.12.8 Master 170
3.12.9 Snapshot System 170
3.13 Splitter 172
3.13.1 Introduction 173
3.13.2 Sequencer 173
3.13.3 Splitter 175
3.14 Vectory 178

3.14.1 Introduction 179
3.14.2 Sample 179
3.14.3 Sequencer 180
3.14.4 Grain Effect 181
3.14.5 Sample Loader 183
3.14.6 MIDI Controller 183

4 Samplers 185

4.1 BeatSlicer 2 185
4.1.1 Introduction 186
4.1.2 Global Section 186
4.1.3 Loop Section 187
4.1.4 Slice Parameters 188
4.1.5 Modulation 189
4.2 Memory Drum 2 191
4.2.1 Introduction 192
4.2.2 Global Parameters 193
4.2.3 Sample & Edit 193
4.2.4 Sample Parameters 195
4.2.5 Modulation 196
4.3 Grainstates SP 198
4.4 Travelizer 199
4.4.1 Introduction 199
4.4.2 Quick Start 200
4.4.3 Structure and Signal Flow 200
4.5 Lurker 202
4.5.1 Introduction 203

4.5.2 Global 203
4.5.3 Sequencer 204
4.5.4 Delay Units 205
4.5.5 Filter 207
4.5.6 Master and Envelope 207
4.5.7 Additional Delay 208
5 Sequenced Synthesizers 210
5.1 Akkord 210
5.2 Atmotion 218
5.3 BlueMatrix 224
5.3.1 Introduction 224
5.3.2 Quick Start 225
5.3.3 Structure and Signal Flow 225
5.3.4 The Sequencer 227
5.3.5 Modulation 229
5.4 Vierring 231
5.4.1 Introduction 231
5.4.2 Structure and Signal Flow 232
5.5 WaveWeaver 233
5.5.1 Introduction 233
5.5.2 Quick Start 234
5.5.3 The Sequencer 235
6 Sequencers 237
6.1 Spiral 237
6.1.1 Introduction 238
6.1.2 Using Spiral 238

6.2 SQ16 241
6.2.1 Introduction 242
6.2.2 Details 242
6.3 SQ8 243
6.3.1 Introduction 243
6.3.2 Details 244
6.4 SQ 8x8 245
6.4.1 Introduction 246
6.4.2 Details 246
6.5 SQP 247
6.5.1 Introduction 247
6.5.2 Details 248
6.6 SQX 249
6.6.1 Introduction 250
6.6.2 SnapSeq 250
6.6.3 TrackSeq 251
6.6.4 ToneGen 253
7 Sound Generators 256
7.1 Metaphysical Function 256
7.1.1 Introduction 257
7.1.2 Quick Start 257
7.1.3 Details 258
7.2 Skrewell 262
7.2.1 Introduction 263
7.2.2 Operation Modes 263
7.2.3 Sound Engine 263

7.2.4 Master Controls 264
7.3 Space Drone 265
7.3.1 Introduction 265
7.3.2 Sound Engine 266
7.3.3 Reverb 267
8 Synthesizer 269
8.1 2-Osc 269
8.1.1 Introduction 270
8.1.2 List of Controls 270
8.2 Carbon 2 276
8.2.1 Introduction 276
8.2.2 Oscillators 277
8.2.3 Filter 279
8.2.4 Effects 280
8.2.5 Modulation Sources 282
8.2.6 Global Controls 285
8.3 Carbon 287
8.3.1 Introduction 288
8.3.2 Quick Start 288
8.3.3 Signal Flow and Structure 288
8.3.4 Modulation 289
8.3.5 Global Parameters 290
8.4 Carbon 290
8.4.1 Introduction 291
8.4.2 Quick Start 291
8.4.3 Signal Flow and Structure 291

8.4.4 Modulation 292
8.4.5 Global Parameters 293
8.5 Equinoxe Deluxe 293
8.5.1 Introduction 293
8.5.2 String Section 294
8.5.3 Ensemble, KleinStein, and Delay Sections 294
8.6 FM4 296
8.6.1 Introduction 296
8.6.2 Operator Sections 1-4 297
8.6.3 FM Sources Section 299
8.6.4 LFO Section 299
8.6.5 Voice Mode Section 300
8.6.6 Effects Sections 301
8.7 Gaugear 303
8.7.1 Introduction 303
8.7.2 Overview 304
8.8 Green Matrix 309
8.8.1 Introduction 309
8.8.2 Quick Start 310
8.8.3 Structure and Signal Flow 310
8.8.4 Modulation 311
8.9 Grobian 312
8.10 Junatik 317
8.11 Kaleidon 319
8.11.1 Introduction 320
8.11.2 Quick Start 320

8.11.3 Signal Flow and Structure 320
8.11.4 Global Parameters 322
8.12 Lazerbass 323
8.12.1 Introduction 324
8.12.2 Oscillator 325
8.12.3 Group Attenuation 325
8.12.4 Oscillator Phase 326
8.12.5 Ratio Multiply 326
8.12.6 Ratio Add 327
8.12.7 Dispersion 327
8.12.8 Partial Beating 327
8.12.9 Visualization Area 328
8.12.10 Brightness 328
8.12.11 Periodic Filter 329
8.12.12 Using the Modulation Matrix 331
8.12.13 Modulation Envelope 332
8.12.14 Modulation LFO 332
8.12.15 Modulation Macro Controls 333
8.12.16 Panning/Master 333
8.12.17 Master Envelope / Release Mode 333
8.12.18 Global Pitch 334
8.12.19 Gate Mode 335
8.12.20 Glide 335
8.12.21 Pitch Mod. 336
8.13 Nanowave 337
8.14 Oki Computer 2 339

8.14.1 Introduction 340
8.14.2 MIDI In 340
8.14.3 Oscillator 341
8.14.4 Filter / Out 343
8.14.5 Envelope, CC1, Sequencer and LFO 343
8.14.6 Modulation Matrix 345
8.15 Photone 347
8.16 SoundSchool Analog 369
8.16.1 Introduction 369
8.16.2 Interface Areas and Control Elements 369
8.16.3 Page B - Instrument Architecture 377
8.17 Steam Pipe 2 378
8.17.1 Introduction 378
8.17.2 Steam 379
8.17.3 Pipe 380
8.17.4 Global Controls 383
8.17.5 Space Master Deluxe 384
8.17.6 Input and Output Stage 384
8.17.7 Reflections 385
8.17.8 Frequency Response 385
8.18 Steam Pipe 387
8.18.1 Introduction 387
8.18.2 Quick Start 388
8.18.3 Structure and Signal Flow 388
8.19 SubHarmonic 390
8.19.1 Introduction 390

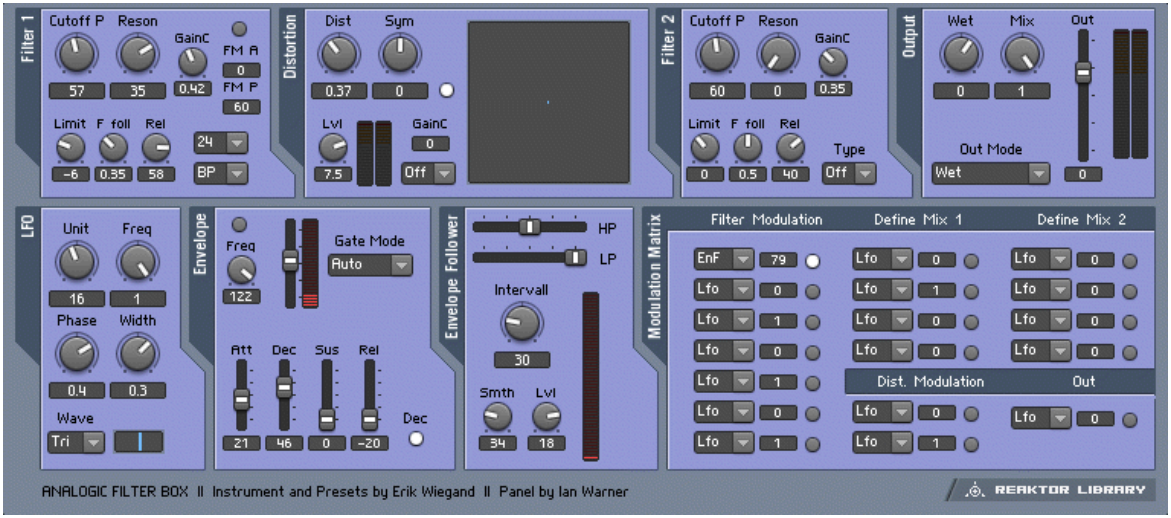
8.19.2 Voice 391
8.19.3 Vibrato 391
8.19.4 Amplitude and Modulation Envelope 392
8.19.5 Sub Oscillator 393
8.19.6 Formant Oscillator 393
8.19.7 Mix and Output 394
8.19.8 Reverb 394
8.20 Sum Synth 396
8.20.1 Introduction 396
8.20.2 Quick Start 397
8.20.3 Structure and Signal Flow 397
8.21 Titan 399

1 Welcome to the REAKTOR Factory Library

The REAKTOR Factory Library comes with more than 70 pre-built Ensembles, ranging from synthesizers and effects to grooveboxes and sequencers. This classic and renowned selection of musical tools unfolds more than ten years of legacy within electronic music. All Ensembles are ready to use in your music and sound design projects, but their open Structures also allow you to look behind the curtain and discover how they work.

2 Effects

2.1 Analogic Filter Box



1.1 Analogic FIlter Box user interface

2.1.1 Introduction

The Analogic Filter Box sandwiches a rich and meaty distortion unit between two hearty analog-style filters to create a sound-shaping tool for every appetite. Juicy modulation is also provided on the side: An envelope, LFO, and envelope follower can be freely routed to the most important filter and distortion parameters. Analogic Filter Box can handle everything from fat disco-loop tweaking to full-on mangling of any sound source imaginable.

2.1.2 Structure and Signal Flow

The signal is routed from input to Filter 1, to the Distortion, to Filter 2, and then out.

Both filters offer multiple operation modes. For the first filter, you can choose between Lowpass, Highpass, Bandpass, Peak EQ, and Notch filters, with a choice of a 12 or 24 dB/octave slope per filter type. The second filter is designed for shaping the sound after the distortion, so it offers four different lowpass filters, three bandpass filters, and a bandpass/lowpass combo. Both filters were very carefully designed to produce warm, analog sounds even at extreme resonance and cutoff settings.

Beside the normal cutoff frequency control available in both filters, Filter 1 also provides fast modulation of its frequency by an additional oscillator (which can itself be modulated by the LFO, envelope, or envelope follower!).

The Distortion section between the two multimode filters also features multiple modes of operation. While the clipper mode provides a relatively harsh distortion sound and the saturator mode results in warm overdrive, the several wrapping modes (marked by the name of the waveform used for wrapping) produce unique sounds from subtle to extreme.

An additional quantize mode converts the incoming signal into a step waveform, for familiar bit-reduction effects to mimic the character of vintage samplers, for instance. A visual display of the distortion function helps to see what's going on inside.

2.1.3 Modulation

Analogic Filter offers six modulation sources (A built-in LFO, envelope follower, enveloper, MIDI note pitch, modulation wheel, and pitch bend wheel). The modulation sources and the flexible matrix signal routing system at the bottom of this effect transforms it into an incredibly powerful machine.

The modulation signal routing system provides a source selector for each parameter to be modified; among those modifiable parameters are the cutoff frequencies and resonance settings of both filters and the distortion amount and symmetry control of the distortion section. In addition to gate information from MIDI note-on events that can even be used to trigger the filter, you can also use the pitch and mod wheels as modulation sources.

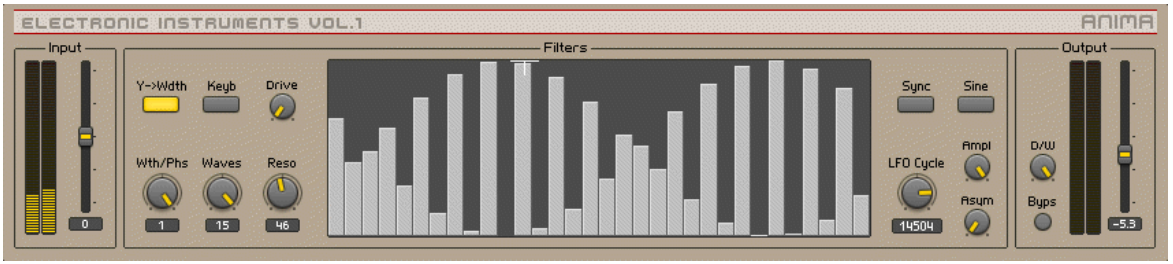
An internal LFO, envelope follower, and auto-trigger envelope can add movement to the sound without the need any external MIDI controllers. The LFO offers different waveforms and can also be synchronized to the global tempo or MIDI clock (the small Unit knob syncs the LFO to MIDI clock and sets the musical note-units that are shown under Freq). The Envelope Follower calculates its modulation amount from the incoming signal: At high levels there is a high modulation level, and at low levels it's low. The Interval knob controls the response time to fast

level changes. Use the cutoff controls of the internal highpass and lowpass filters to select a specific frequency band of the incoming signal to trigger the envelope follower. The Envelope is a standard attack-decay-sustainrelease envelope generator, triggered by MIDI note on events. However, an additional auto-trigger feature allows it to be triggered by the incoming audio, settable with the Tresh slider.

It's even possible to combine any two MIDI controllers to make one dependent on the other - for instance, to have the amount of LFO modulation dependent on MIDI pitch. You can define custom mix modulation combinations in the Define Mix 1 and 2 areas as the bottom of the instrument.

* You can play a sound through the built-in loop-player, through the realtime audio inputs, or you can process audio in realtime by using Reaktor as an effect plugin. Please check your Reaktor or Reaktor Session user's guide for helpful information.

2.2 Anima



2.1 Anima user interface

Introduction

Anima is a polyphony-dependent filter bank, animated by an LFO that can produce complex wave effects.

Input

<Level> Sets the input level, in dB. The meters show the post-fader input level.

Filters

A single bandpass filter corresponds to every voice of polyphony. All of the filters are controlled together, in the main XY display.

- <Y->Wdth> If ON, the Y coordinate of the Xy controller in the center of this panel modulates the width of the frequency band that is covered by the bandpass filters. If off, the Y coordinate modulates the phase offset or starting point of the wave that determines the amplitude of the bands. The "missing" parameter can be set with the "Wdth/Phse" knob.
- <Keyb> If ON, incoming midi notes shift the cutoff frequencies of the bandpass filters. Note that the actual cutoff frequency as set by MIDI notes is not displayed in the XY panel.
- <Wdth/Phse> Width / Phase control. If the "Y-Wdth" button is off this knob controls the width of the frequency band that is covered by the bandpass filters in semitones. If the "Y-Wdth" button is on it controls the phase offset or starting point of the wave that determines the amplitude of the bandpass filters. Values between 0 and 1 set the phase between 0 and 360 degrees.
- <Waves> Sets the number of cycles of the wave that determines the amplitude of the bandpass filters.
- <Resonance> Sets the resonance of the bandpass filters in dB. The amplification is compensated internally so the resonance controls only the narrowness of the bandpass filters.
- <Drive> Sets the amplification before the distortion unit. The amplification is compensated internally so it controls the amount of distortion.
- <XY Panel> The XY panel shows a graphical representation of the amplitude and frequency of all bandpass filters.
    - Display: X->frequency of the filters
    - Display: Y->amplitude of the filters
    - Controller: X->center of the frequency range that is covered by the bandpass filters.
    - Controller: Y->either the width of the frequency range or the phase offset of the wave that determines the amplitude of the bandpass filters (depends on the "Y->width" button on the left).

LFO

- <Sync> If ON, the LFO is synced to the global clock. Its length is then set in 16th notes. The global clock can be started and stopped by the corresponding buttons in Reaktor's toolbox. If Reaktor is used as a plug in the clock of the host is used.
- <LFO Cycle> Sets the cycle length of the low frequency oscillator (LFO). The LFO modulates the phase offset of the wave that determines the amplitude of the bandpass filters. If the "Snyc" button is on the length are set in 16th notes. If off the length is set in milliseconds.
- <Sine> Toggles the waveform of the LFO from triangle to sine wave.
- <Ampl> Sets the amplitude of the LFO.
- <Asym> Sets the amount of asymmetry of the LFO. E.g. a triangle waveform turns in a sawtooth with a rising slope if the knob is turned fully clockwise. Counter clockwise it turns into a sawtooth with a falling slope. In the center it remains a triangle waveform.

Output

- <Level> Sets the output level, in dB. The meters show the post-fader output level.
- <D/W>Dry / Wet control. It crossfades between the incoming signal (fully left) and the processed signal (fully right).
- <Byps> Bypass switch which turns the effect off. When off, the CPU is no longer used.

2.3 Banaan Electrique



1.2 Banaan Electrique user interface

2.3.1 Introduction

Banaan Electrique is a sophisticated guitar and bass amp simulator with built in effects. It's like having a pedal board full of vintage pedals and a vintage amp, with the added advantages of not needing to cable anything together or risk electrocuting yourself by running too much

current through the amp. You can still get shocked by its lush sound, however. Naturally, Banaan is happy to eat whatever types of sounds you feed it, whether they're vocals, drum loops, synths, or scallops.

2.3.2 Getting Started

Banaan Electrique was designed to be so easy that even guitar players can use it! Just play some audio through it and let a rip.* Check out the presets to see the variety of tones that Banaan can achieve.

2.3.3 Structure and Signal Flow

The audio runs through Banaan Electrique linearly:

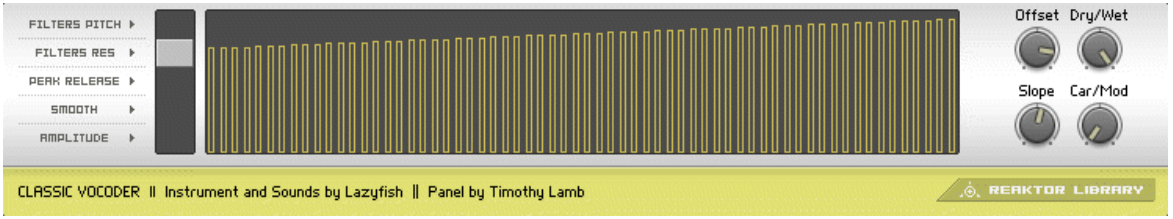
Input > mono gate > 3-band EQ > Compressor 1 > Amp Simulator > Phaser > Rotor > Dual Delay > Reverb > Compressor 2

The incoming signal is converted to mono, amplified, gated, equalized and compressed to control the input's amplitude level and to enhance those sounds wanted to be in the recording, getting rid of any unwanted noise. Then, a guitar amplifier is simulated, including distortion, overdrive and filtering. Its output signal is sent to a rotor module, placing the mono sound in the stereo field dynamically. This signal is then routed to a phaser effect and, afterwards, to a stereo echo unit whose delay times can be synced to master tempo or MIDI clock. A high-quality reverb enhances the spatial sound once more, feeding its output into the final compressor.

Not all modules have to be active at the same time - in fact, it's a good idea to turn off any modules that you're not using to save CPU. You can turn modules on and off with their respective Power buttons.

* You can play a sound through the built-in loop-player, through the realtime audio inputs, or you can process audio in realtime by using Reaktor as an effect plugin. Please check your Reaktor or Reaktor Session user's guide for helpful information.

2.4 Classic Vocoder



1.3 Classic Vocoder user interface

2.4.1 Introduction

The Classic Vocoder was designed to faithfully emulate the well-known tones of singing robots made popular in the seventies. The instrument combines a vocoding engine, a vintage-type synthesizer, and a four-band dynamics processor for a warm, smooth sound.

2.4.2 Quick Start

The audio input is vocoded with the built-in synthesizer. Play some audio into the vocoder.* You don't have to sing into it - the Classic Vocoder also gives great results with drum loops or other sounds.

2.4.3 Structure and Signal Flow

Audio input 1 is the modulator, and is vocoded with the Classic 2-VCO synth, which is the carrier. The entire signal flow is mono. If you are using a stereo signal, only the left channel will be used. The output of the vocoder is fed into the four-band normalizer to smooth-out the sound and remove any uncomfortable signal peaks that could come with vocal sibilants or drum transients.

2.4.4 The Vocoding Engine

Sonically, Vocoding uses the characteristics of one sound to control another. To achieve the popular robotic-singing effect, a voice (technically called the modulator) is vocoded with a constant sound, such as a synth or string sound (the carrier). The frequency content of the voice is split up into many different bands - the number of bands has an obvious impact on the sound, with fewer bands leading to more synthetic voices, and higher bands make the voice easier to understand. You can adjust the number of voices of the vocoder instrument to change how many bands are used. Up to 128 voices (bands) are possible. All changes in number of bands are immediately shown in the graphical display.

The amplitude of each frequency band of the voice is linked to the frequency bands in the string or synth sound. The re-shaped bands of the carrier signal are mixed together, providing the output signal of the vocoder.

If you're interested in vocoding, you may also want to do check out another NI product - the VOKATOR. The VOKATOR features vocoding up to 1024 bands, a built in synthesizer and granular sampler, and many high-end vocoding features. www.ni-vokator.com.

* You can play a sound through the built-in loop-player, through the realtime audio inputs, or you can process audio in realtime by using Reaktor as an effect plugin. Please check your Reaktor or Reaktor Session user's guide for helpful information.

2.5 Cyan



3.2 Cyan user interface

Introduction

Like a traditional Chorus effect, Cyan uses a bank of modulated-delay units to add width, depth and richness to the input signal. And due to painstaking attention to detail in the design process, Cyan can reproduce classic chorus-type sounds with the finest sound quality. However, Cyan is far more than just a high-quality chorus unit. Indeed, due to its advanced features such as tempo-synchronisation and extended delay modulation control, Cyan offers many new and exciting sound-shaping possibilities.

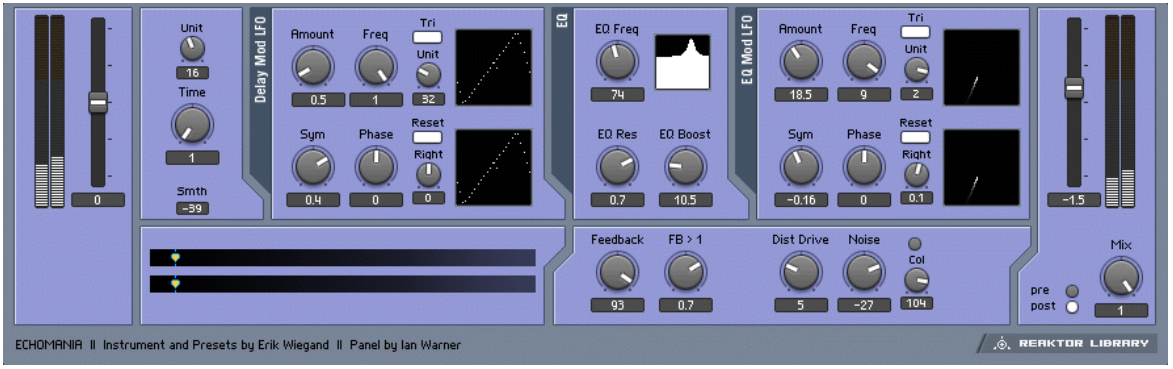
Details

The effect's architecture consists of up to six independent stereo delay lines (see sections Distance, Intensity and LFO) that are fed with the incoming audio signal (see Input). Optionally, the delay lines' output can be routed again to their input to establish an additional feedback circuit (Feedback). After summing the delay lines' signal the sound is routed to a saturation unit and an equalizer to dynamically and harmonically re-shape the chorused signal (Saturation and Equalizer). Finally, the signal is mixed with the unprocessed one and sent to the output (Mix Automation, Additional Mix and Output).

Section	Function
Power and Input	These two sections control whether the chorus is switched on or off and which signal is used as input: The complete stereo signal, the left channel alone, or a mono mix of both.
Distance, Intensity, and Feedback	The **Delay** knob on the other hand adjusts the delay times offset, thus controlling the phase variation applied to the chorused signal in respect to the unprocessed signal. The **Spread** control additionally changes this offset slightly for each LFO independently to enhance this effect. The **Intensity** knob mainly controls the sound of the chorus: It determines the amount by which the delay times vary. After passing through the delay units, the signal is routed to the Feedback section where it can be filtered (using the **Colour** knob to control the filter's frequency), and is then mixed with dry signal. The **FB** amount knob is bi-polar: At the right there is normal feedback, at the left the feedback signal's phase is inverted; at mid position the feedback is turned off.

Section	Function
LFO	**Waveform** and **LFO** determine the shape and number of low frequency oscillators. **Rate** controls how fast the delay times are modulated by the low frequency oscillators. This speed is displayed either in Hertz or, if **MIDI Sync** is enabled, in rhythmical values. MIDI Sync is only available if **Freq. Mode** is set to 'Phaselocked', meaning that the phase of the LFOs is linked to each other. If it is set to 'Freerun', the LFOs' phase can be spread (like the delay time offset) by the **Ph.Spread** knob. **Ph.Shift** controls the phase to which the oscillators are reset at synchronization events that can be specified with the **LFO Reset** switch; the button below this menu can be used to manually synchronize the oscillators.
Saturator and Equalizer	After the chorus section itself the audio signal can be saturated and equalized.
Mix automation and output	Finally, the chorused signal is mixed with the unprocessed sound, defined by the **Dry**/**Wet** balance knob position. Mix balance can also be automated by a ramp generator, the speed of which is adjustable by the **Speed** knob. The ramp generator trigger source can be selected at the left of the Att knob. The small switch below the main **Dry**/**Wet** knob mutes the unprocessed audio, effectively converting the Dry/Wet knob into an amplifier for the processed signal. This is especially useful when using Cyan as a send effect. Tremolo enables amplitude modulation, synchronized to the delay time modulation. The Stereo knob controls the spread of the audio within the stereo panorama.

2.6 Echomania



1.4 Echomania user interface

2.6.1 Introduction

Echomania is an advanced and spectacular-sounding delay box that excels at tight, tempo-synced rhythms. It includes two LFOs to to modulate the delay time and built-in EQ. The delay time is handily synced to the global MIDI clock, for creating stretchy rhythmic effects. A drive and noise circuit combined with a feedback offset control recreate vintage sounds. Get dubbing!

2.6.2 Quick Start

Play some audio through Echomania.* Flip through the snapshots and discover the gritty (or crystalline) depths of Tapedelay.

2.6.3 Structure and Signal Flow

The delay time is set in 16th note increments, synced to the global MIDI clock. The Unit and Divisor windows, to the right of the Time control, allow you to fine tune the beat division and create different rhythmic feels—to be precise, the enumerator divided by the denominator scales the delay time fraction.

There is also an LFO, with blendable sine/triangle/square/slow random waveshapes that is hardwired to modify delay time. You can set the amount of time modulation with the Amount knob. The Freq knob controls the rate of the LFO in 16th notes. The Unit knob divides the speed of the global clock by the Unit amount. For example, if the Unit knob is set to "6", then the LFO Freq amount will be in 16th notes. The Width control morphs between sine and pulse-like, or, if the Tri button is engaged, triangle and saw tooth waveshapes. The starting phase of the LFO can be adjusted positively or negatively using the Phse knob. If you engage the Snc button, the right-side LFO can be phase-offset from the left side by the amount set with the Right knob. This can create a variety of woozy stereo spinning and phasing effects.

The EQ module processes the delayed signal. It is essentially a parametric EQ that contains an LFO identical to the one in the Delay module. You can create synchronized filter sweeps, fizzing hi-frequency delay tails, and all manner of dubby effects by boosting and modulating select frequency bands. The Eq Res control lets you dial in the peak width of the frequency, while EQ Boost lets you crank it up.

Tapedelay's Feedback module provides an offset control, labeled FB > 1, which boosts and shapes the feedbacl signal, making it seem to get louder and louder (but without degenerating into uncontrollable noise).

The Mixer lets you add a tape saturation-like Dist Drive effect and Noise to give everything that just-pulled-out-of-the-closet feeling.

The Dry Wet section allows you to balance the amount of dry and delayed signal. You can also use the Tap buttons to select whether the delay tap comes before or after the saturation/noise circuit.

* You can play a sound through the built-in loop-player, through the realtime audio inputs, or you can process audio in realtime by using Reaktor as an effect plugin. Please check your Reaktor or Reaktor Session user's guide for helpful information.

2.7 EnFX



2.3 EnFX user interface

Introduction

EnFX consists of three separate effects: an envelope-follower filter, distortion, and delay. Even though the effects are together in one ensemble, they can also be used independently. The ensemble is routed in series: Filter -> Distortion -> Delay. Each effect features a identical Input, Envelope Follower, and Output sections. These sections will be explained first:

Input

<Level> Sets the input level, in dB. The meters show the post-fader input level.

Envelope Follower

An envelope follower isn t a dog that chases the postman. Instead, an envelope follower listens to the incoming audio, and when the level of the audio exceeds a certain threshold, then a simple HR (hold/release) envelope is triggered. This envelope can control different parameters, depending on the effect. The result is an effect that wraps itself around the incoming audio for an organic, natural sound.

- <On> Switches the envelope follower on or off to save cpu power.
- <Hold>, <Release> Set the hold and release values of a HR-envelope that is triggered by a rising of the input level. It shapes the output of the peak detector. Lower values, i. e. shorter hold and release times, result in a sharper response to level decays.
- <Mono> Switches between stereo mode (off) and mono mode (on). Use mono mode for a more stable envelope. In stereo mode, the two incoming channels are processed seperately -a separate envelope of the left and the right channel is generated. In mono mode, both channels are mixed together before extracting the envelope. ATTENTION: If a mono input source is connected to both effect input channels, be careful using this switch: The envelope is already mono, so there's no additional benefit; instead, by mixing, i. e. adding both channels the signal is amplified and exceeds the normal range.
- <PkRel> Sets the release time of the amplitude peak detector. Lower values at the left result in more sensitivity to smaller peaks (e. g. HiHats), higher values instead only detect really high peaks like bass drums etc.
- <Smth> Sets the amount of smoothing of the shaping envelope. Higher values result in a more smoothed signal following the envelope of the incoming signal.
- <Envelope Meter L & R> Show the output of the envelope follower section.

2.7.1 EnFX Delay 1.3

At the center the EnFX delay is a delay whose time is modulated by an envelope follower. It s useful for very long delays due to internal feedback enhancements as well as for unique pitch shifting. See the envelope follower info above for information about the envelope follower section.

- <Left> Sets the delay time of the left audio channel. The display below shows the delay time in 16th notes according to the midi tempo. Internally, the linear output of the knob is processed by a function mapping it onto a grid to allow very short decays as well as very long decays. The grid also performs triplet division of the time (0.3333, 0.1667, 0.0833 etc).
- <-E+> Sets the amount of modulation and its polarity applied to the delay time of the left channel. Turn to the right for a longer delay times at high envelope follower signals, turn to the left for shorter delay times at high envlope follower signals. Use mid position to switch the modulation off.
- <Right> Sest the delay time of the right audio channel.
- <-E+> Sets the amount of modulation and its polarity applied to the delay time of the left channel. Turn to the right for a longer delay times at high envelope follower signals, turn to the left for shorter delay times at high envlope follower signals. Use mid position to switch the modulation off.
- <LP> Sets the cutoff freqeuncy of a lowpass filtering performed to the feedback signal.
- <HP> Sets the cutoff freqeuncy of a highpass filtering performed to the feedback signal.
- <Lift> Sets the maximum level lifting. Turn right to amplify very silent signals in the feedback tail, turn left to amplify only the louder signals. Use it to make very long and damped feedbacks audible.
- <Smth> Sets the amount of smoothing in the level lifting algotithm. Turn to the left for smooth transistions, turn to the right for sharper and faster response to level changes.
- <Swap> Sets the amount and mode of channel swapping of the feedback signal. Turn to the left for no swapping, i. e. the left channel remains the left one, the right channel the right one. Turn to mid position for mono processing, i. e. both signals occur in both channels. Turn to the right for crossfading, i. e. the left channel signal becomes the right channel signal and vice versa.
- <Fb> Sets the amount of feedback in the delay. More feedback leads to a decay with more repeats.
- <Dry/Wet> Crossfades between the dry (unprocessed) signal at the left and the wet (processed) delayed audio at the right.

- <-E+> Sets the amount of modulation and its polarity applied to the dry/wet balance. Turn to the right for a more wet signal at high envelope follower signals, turn to the left for a more dry signal at high envlope follower signals. Use mid position to switch the modulation off. The yellow meter displays the actual value of the parameter modulated by —E +.

Output

- <Level> Sets the output level, in dB. The meters show the post-fader output level.
- <D/W>Dry / Wet control. It crossfades between the incoming signal (fully left) and the processed signal (fully right).

<Byps> Bypass switch which turns the effect off. When off, the CPU is no longer used.

2.7.2 EnFX Distortion 1.3

EnFX Distortion is a overdrive effect with a pre- and a post-distortion equalization filter whose center freqeuncies are controlled by an envelope follower. Usefull for all kinds of overdive effects, but can also be used for guitar wahwah etc. due to the responce to level changes of the incoming signal. In the distorion section, the left knobs belong to the "push" filter that modifies the signal before the distortion; the right knobs belong to the "pop" post-distortion equalizer. See the envelope follower info above for information about the envelope follower section.

- <rEs L& R> Set the amount of resonance of the peak eq filters.
- <Push> Sets the freqeuncy of the pre-distortion peak equalizer
- <Res> Sets the resonance of the pre-distortion peak equalizer.
- <-E+> Sets the amount of modulation and its polarity applied to the frequency of the pre-distortion peak equalizer. Turn to the right for a higher frequency at high envelope follower signals, turn to the left for lower freqeuncies at high envlope follower signals. Use mid position to switch the modulation off. The yellow meter displays the actual value of the parameter modulated by —E+.
- <Dist>Sets the amount of distortion. On high distortion levels the signal might tend to fade. Increase the drive value to fade it back in.
- <Boost> Sets the amount of amplification performed to the freqeuncies set by the Push and Pop knobs.

- <Drive> Sets the amount of overdrive amplification.
- <Pop>Sets the freqeuncy of the post-distortion peak equalizer.
- <-E+> Sets the amount of modulation and its polarity applied to the frequency of the post-distortion peak equalizer Turn to the right for a higher frequency at high envelope follower signals, turn to the left for lower freqeuncies at high envlope follower signals. Use mid position to switch the modulation off. The yellow meter displays the actual value of the parameter modulated by —E+.
- <Res> Sets the resonance of the post-distortion peak equalizer.
- <Dry/Wet> Crossfades between the dry (unprocessed) signal at the left and the wet (processed) distorted audio at the right.
- <-E+> Sets the amount of modulation and its polarity applied to the dry/wet balance. Turn to the right for a more wet signal at high envelope follower signals, turn to the left for a more dry signal at high envlope follower signals. Use mid position to switch the modulation off. The yellow meter displays the actual value of the parameter modulated by —E +.

2.7.3 EnFX Filter 1.3

An envelope follower filter is traditionally used to create ªwah wah sounds, but the flexiblity of EnFX s design makes much more experimental sounds possible. See the envelope follower info above for information about the envelope follower section.

- <rate> Sets the rate of a slow random generator. The value displayed below shows the time in milliseconds that goes by before another random event is generated.
- <Depth> Sets the depth of modulation, i. e. the amount by which the envelope follower signal is modulated by the slow random generator. Turn right for more modulation.
- <Res> Set the amount of resonance, the left knob at the lower cutoff freqeuncy, the right one at the higher freqeuncy. (See also <cut/pass>.)
- <Filter cutoff freq meter> Show the current cutoff freqeuncy. Here you can see also the effect of modulation by the envelope follower and the slow random generator. Left represents a low freqeuncy, right a high freqeuncy. The meter on the left below the <LOW> knob displays the frequency used in the lower filter, while the other one displays the freqeuncy of the higher filter. (See also <cut/pass>.)

- <-E+> Set the amount of modulation of the filter cutoff freqeuncies by the envelope follower) whose signal before was modulated by the slow random generator). Turn left to make the filter freqeuncy be reduced on high incoming envelopes, turn right for the opposite, i. e. high envelope signals increase the cutoff freqeuncy. The left knob modifies the lower cutoff freqeuncy, the right one the higher freqeuncy. (See also <Cut/Pass>.)
- <Low> Sets the lower cutoff freqeuncy. (See also <Cut/Pass>.)
- <High> Sets the higher cutoff freqeuncy. (See also <Cut/Pass>.)
- <Cut/Pass> Crossfades between cut mode (left position) and pass mode (right position). In cut mode, all freqeuncies below the lower freqeuncy set by <Low> and above the higher freqeuncy set by <High> are cut off, resulting in a bandpass filter whose boundaries are specified by the lower and higher freqeuncy. In pass mode, all freqeuncies below the lower freqeuncy set by <Low> and above the higher freqeuncy set by <High> are passed through, resulting in a noth filter whose boundaries are specified by the lower and higher freqeuncy.

2.8 Fast FX



3.3 FAST FX user interface

Introduction

Fast FX is a set of 6 different effects: Freeze, zZZzZ, Slicemanipulator, Gate, Delay and Filter. These effects can be chained in a variety of ways offering a large range of sounddesign possibilities. However, Fast FX's speciality is live performance. Not only are all controls designed to be tweaked in real-time, but Fast FX features a special 'live' snapshot system – enabling any of the 256 states to be instantly recalled via simple keyboard strokes. This ability to instantly switch between sets is what really brings Fast FX to life.

Quick Start

The most important thing to note is that Fast FX does not use the standard REAKTOR snapshot system. Instead it uses a special 256-slot system operated by the panel and computer / MIDI keystrokes. The 256 snapshots are made up of 32 x 8 slot banks. Select bank by the using the 32-step **Program Selector** slider at the bottom of the panel, and select the slot by using the **QWERTUI Selector** above.

When **Flow** is enabled, selecting a new bank or slot automatically loads the preset. Press **Write** to save the current ensemble state to the selected bank/slot.

Fast FX is supplied with a Loop-player for a useful sound source in REAKTOR standalone mode, but it can also process live input (e.g. when used as a plug-in in a host sequencer.) The Loopplayer input is selected by default, but to enable the Live input press the **In 1/2** button in the Input section. Fast FX will now process the audio input from REAKTOR audio-in channels 1 and 2.

Global Controls

Section	Function
Input	Fast FX has two pairs of stereo inputs. The first pair of inputs are connected to the Loopplayer instrument, whereas the second pair are connected to REAKTOR's audio-in ports. Enable the **1/2** button to select the second pair (i.e. for use with live audio input, or when using Fast FX as a plug-in inside a host application). **Level** and **Mute** control the input level.
Junction	This section allows configuration of the signal flow order between the 6 effects. Eight possibilities are available in the drop-down menu.

Section	Function
Output	**Level** and **Mute** control the output level. All effects can be bypassed simultaneously with the Bypass button.
Mix	The mix section controls the overall mix between the dry input signal and the wet, post FX signal. The mix is defined separately for low and high frequencies with the Low and High knobs. Link synchronises the two knobs.

The Effects

Section	Function
Freeze	The Freeze effect makes looping and scratching sounds. **Loop** sets the loop length in musical time, from 1/16th to 4/1 (1 bar = 16/16th's). **Spin** determines how the loop length changes when the effect is enabled. A setting of 1/32 will result in a loop length that is 32 times shorter, a setting of 32/1 will result in a loop length 32 times longer. **Spin Time** determines the speed of loop length changing, and **Jtr** sets the delay time jitter – the amount of slow, random variation.
zZzzZZ	This effect slices incoming sound into grains, and then slides the pitch, so that the pitch of each successive grain is either decreasing or falling compared to the previous. **Slice A** and **Slice B** set the length of first and second slices respectively, in 16th steps. **Length** sets the grain length in milliseconds. Below, the knob labelled **Mod** sets the amount of grain length modulation. **Slide+** controls pitch slide, and the **Mod** knob below determines the amount of pitch modulation. For both Length and Pitch modulation, a collection of pseudorandom, MIDI clock synchronised patterns are used as the modulation source. The **Rnd** XY field is used to select the current pseudo-random pattern (each pixel represents one of the 1920 available patterns). **Asym** sets the grain size asymmetry between the left and right channels. **Decay** controls the dry signal decay time, the envelope is retriggered with each new slice (i.e. at intervals according to the Slice A and Slice B controls).

Section	Function
Slice Manipulator	This effect also slices the incoming sound, and then shifts the position and playback direction of the slices. **Slice A** and **Slice B** set the length of first and second slices respectively, in 16th steps. The two slicers work on a rotating basis, so therefore the slice lengths alternate between the two settings. The **SncVar** enables a mathematical formula to automatically rearrange and vary the slices in time. Set to zero for no variation. **Rev A** and **Rev B** enable reverse playback of the first and second slices.
Gate	This section features a 16-step sequencer that modulates the amplitude of the incoming signal. The horizontal slider above the sequencer display selects one of the 20 available patterns for the sequencer, and the lower horizontal slider sets the sequencer clock speed. The preset sequence can be varied by reducing the loop length with the **Length** knob, or by varying the playback position with the **PosVar** knob. This control applies one of 16 variation patterns to the sequence playback order. **Gate** determines the length of the open gate, as a proportion of a sequencer step. **Release** sets the time the gate takes to close.

Section	Function
Delay	The section features a stereo delay line with a modulated filter in the feedback path. **Delay** sets the delay time in 16th steps, **Fbck** sets the feedback amount and **Asym** sets the delay time asymmetry between the left and right channels. **Filter** sets the centre frequency of the feedback filter. The cutoff is also modulated a beat-synced LFO, controlled by the central XY field and **Speed** knob. The XY X axis sets the LFO phase; the Y axis sets the modulation depth; and the Speed knob sets the speed (cycles per bar). **Smth** controls the speed that delay time changes (from the Delay / Asym knobs) are applied, and also the crossfade speed of enabling/disabling the effect.
Filter	This section features a sequenced filter. The horizontal slider above the sequencer display selects one of the 20 available patterns for the sequencer, and the lower horizontal slider sets the sequencer clock speed. **Length** sets the loop length of the sequencer. **Mod** defines how much the extent to which the sequencer modulates the filter cutoff. **Center** and **Reso** set the filter cutoff and resonance respectively, and **Width** sets the filter width. **Smth** controls the speed that of changes to the cutoff frequency (from the sequencer or the Center knob).

Snapshot control

Section	Function
Q W E R TYU I	In Fast FX, the REAKTOR snapshot menu is not used for handling snapshots. Instead, with live performance in mind, the snapshot handling is dealt with on the panel.
	Fast FX has a total of 256 snapshot slots, represented by 32 x 8 slot programs. The 32 programs are selectable by the **Program Selector** at the bottom of the panel. Either click on the selector itself, use the **Prev** and **Next** buttons, or press computer keys 'L' or ';'.
	Within each program, the 8 sub-slots are selectable by the **QWERTUI Selector**. To select a slot, click on the selector, or press the associated computer keyboard key (i.e. Q,W,E,R,T,Y,U,I) or MIDI note (60,62,64,65,67,69,71,72).
	The total 256 slot data is stored in a large event table inside the structure. If you wish to export snapshot files, simply save the table data by right clicking on the table display either beneath the QWERTYUI selector or the Program Selector.
Bookmark	The bookmark feature enables you to store your favourite presets numbers and quickly access them by using the dropdown menu.
	To set a bookmark to the current state, take a look at the **POSITION** display, then open the Bookmark list properties, and create an entry there with a name of your choice, and the value displayed in the position meter.
Flow and Write	When **Flow** is enabled, selecting a snapshot position (via either the QWERTUI or Program selector) automatically recalls the data. Disable this if you want to copy the current state to another slot (i.e. disable Flow, select the desired Slot and then press **Write**).
	Write writes the ensemble state to the current bank and QWERTUI slot.

Loopplayer

Section	Function
Panel controls	This is an additional instrument for playing loops, it uses the REAKTOR Beatloop module, and thus automatically stretches loops to the current tempo. You can drop samples on the sampler window, or double click on it to open the REAKTOR map editor. Transpose should be set to 0 for all samples Select the desired sample from the map using the Select knob. **Loop Lng** and **Offset** and define the loop length, and position offset in 16th steps respectively. **Pitch** transposes the sample in semitones. The Loopplayer has 16 different automatic variation patterns, these can be selected with the **Pos Variator** knob. **Center** and **Reso** set the filter cutoff and resonance respectively, and **Width** sets the filter width. The filter can be bypassed by pressing the **Bps** switch. **Rnd Mrph** determines the amount to which a slow randomiser is applied to the filter cutoff. **Fit** links sample pitch with time stretching, like a conventional sampler (i.e. the slower the tempo, the lower the pitch). **Smooth** controls the sampler grain smoothness, and **Gain** determines the output level.
Snapshot handling	The Loopplayer state is not saved in the Fast FX snapshot table, but pressing the Write creates a snapshot in the Loopplayer instrument. Programs 1 to 16 are stored in bank 1 and programs 16 to 32 are stored in bank2. These are recalled when recalling snapshots on the Fast FX panel.

2.9 FlatBlaster 2



4.1 The Flat Blaster 2 user interface

2.9.1 Introduction

The great final-stage mastering tool FlatBlaster 2 has been rebuilt using the new REAKTOR Core features. This patch combines four frequency-specific compressors with a full-spectrum peak-limiter to produce a high-end package for your multiband dynamics shaping needs. As it does not introduce any delay it is not limited to mastering use but can also be applied on a per channel basis. The controls might appear intimidating at first glance, but are actually straightforward when you examine the signal chain. The separately compressed bands get mixed together and then processed by a full-band peak limiter. Please note that the master bypass for the complete patch is situated to the left above the X Over section.

2.9.2 What's New in 2.0.2?

Version 2.0.2 of Flatblaster 2 sports the following improvements:

- Lookaheads in compressors and limiter
- Smoother behaviour of envelope the followers
- Knee instead of saturation in the limiter
- Overall CPU consumption reduced

2.9.3 Multi-band Compressor

After the input stage the signal gets split into four independent frequency bands as defined by the X Over section. Each frequency band gets processed by independent, identical compressors and can be independently muted, soloed and bypassed. Separate saturators for each band make it possible, for example, to add punch and bite to the mids without affecting clarity in the lower registers.

Section	Control	Function
Input	Input	Trims the input gain to prevent overload.
	Bypass	Bypasses the complete effect. This is the master bypass switching off all compressors and the Limiter.
X-Over	High	Sets the crossover frequency between the High and Mid High compressor bands.
	Mid	Sets the crossover frequency between the Mid High and Mid Low compressor bands.
	Low	Sets the crossover frequency between the Mid Low and Low compressor bands.
High, Mid High, Mid Low and Low Compressors	Stereo	Sets stereo width of the frequency band. 0 is mono, 1 is original stereo, 2 is extra stereo.
	Tresh	Sets the point at which the compressor will begin to work (in db). Levels below this threshold remain unprocessed.
	Ratio	Adjusts the ratio of the input level to the output level after compression.

Section	Control	Function
	Knee	This parameter adjusts how gradually the full amount of compression is introduced. Think of it as a slope control for the attack time.
	Sat	Drives the band into saturation.
	Link	Activates stereo linking of the two input channels. When active, the compressor takes the max of the left and right peak levels and uses it for both channels. This preserves a clean stereo image and is lighter on CPU cycles.
	Att	This dial adjusts the attack time. It is the time the compressor takes to react to an above-threshold signal.
	Rel	With this control you set the release time. This is the time the compressor takes to return the signal to normal when it falls below the compression threshold.
	Out Gain	Sets the amount of amplification applied to the compressed signal of the specific band before it gets mixed with the other bands.
	Bypass	Bypasses the compressor for the respective band.
	Mute	Turns the sound of the respective band off.
	Solo	Turns all other bands off, leaving only the signal of the soloed band. Use it to fine tune single compressor bands.

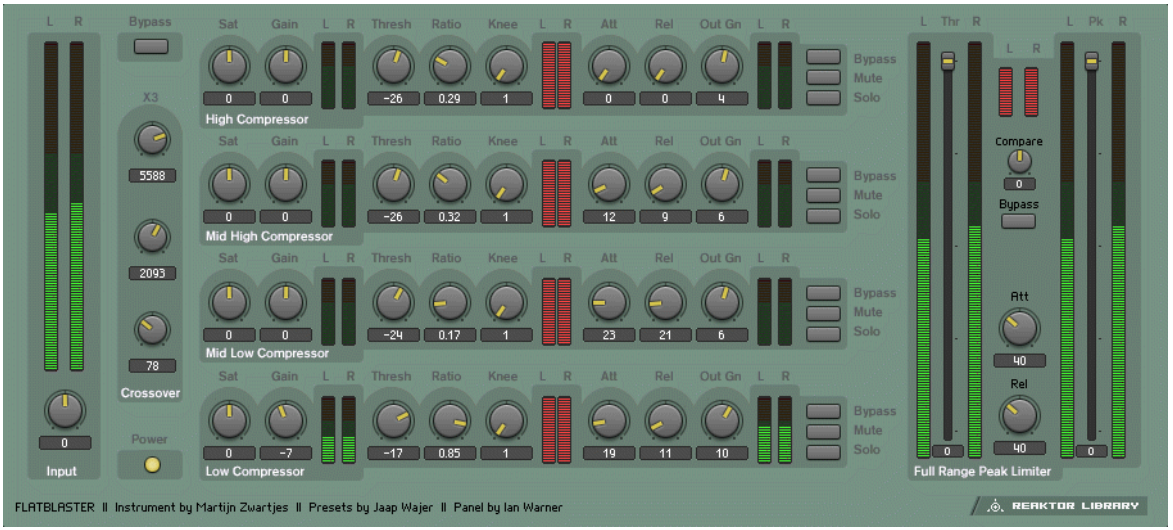
2.9.4 Full-band Peak Limiter

The peak limiter affects the full band signal. For clean mastering purposes we recommend a limiter threshold setting of about -3 to -4 db and a peak setting at 0db. Should pumping effects be desired, adjust the threshold to more extreme values.

Control	Function
Thr	Adjusts the threshold of the limiter. Levels above this value get processed.
Peak	Adjusts the hard limit of the signal. No signal will exceed this limit.
Rel	This adjusts the release time. It is the time the limiter takes to return the signal to normal when it falls below the limiting threshold.
Soft / Hard	Balances between soft saturation and hard clipping of the above peak signal.

Control	Function
Compare	Controls the amplification of the uncompressed signal if Bypass is active. If you want compression without amplification set it to 0 and make sure there is no change in level when toggling the Bypass button.
Link	Activates stereo linking of the two input channels.
Bypass	Bypasses the Full Band Peak Limiter only, leaving the 4 compressors active.

2.10 Flatblaster



1.5 Flatblaster user interface

2.10.1 Introduction

Flatblaster is a high-end finalizing and multiband dynamic shaping tool. Flatblaster combines four frequency-specific compressors with a full-spectrum peak-limiter. It is an excellent final-step mastering plug-in, but it can also be used while mixing since it doesn't introduce any de-

lay to the signal. Each of the compressors has a saturator, so you could saturate just the upper-mid frequencies, for instance, without muddying the bass. It also makes an excellent de-esser and sibilant reducer.

2.10.2 Quick Start

Even though Flatblaster gives control over many sound-shaping parameters, there's no need to be intimidated by its complexity. A full range of presets shows off its capabilities and gives good starting points for tweaking the effect for your sound. Simply play some audio through Flatblaster* and step throught the presets. Try experimenting with muting, soloing, and bypassing each individual band so you can carefully hear what the 'blaster's doing. Be careful when adjusting the saturation of each band! The sound can potentially get very loud if you don't first reduce the make-up gain (the knob labeled Gain to the right of the Sat knob).

2.10.3 Structure and Signal Flow

The input signal is divided into four bands - the three crossover frequencies are adjustable independently. Each frequency band is processed by an independent, identical compressor. Each band can be muted, soloed, and bypassed (no compression). The signal is summed before going through a full-band peak limiter, which can also be independently bypassed. The master bypass for the effect is located to the right of the input meters, above the crossover settings.

2.10.4 Frequency-Specific Compressor

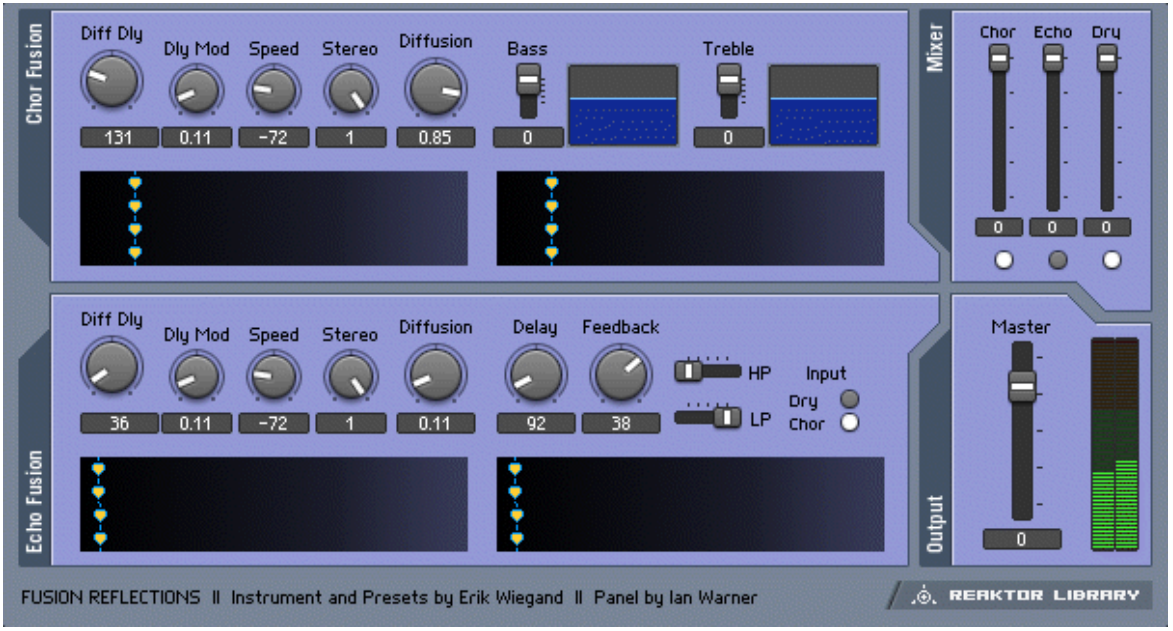
Each of the four compressors are absolutely identical. In fact, they have to be! If they weren't, then unwanted phase shifts could creep in. Each compressor gives control over saturation, saturation makeup gain, threshold, compression ratio, adjustable knee, attack, release, and output makeup gain. Note that the Ratio has to be higher than 0 for the compressor to have any effect - at a Ration of 1 (maximum), the compressor acts as a limiter. The red meters show the amount of peak reduction. The Attack and Release knobs control how the compressor responds to transient signals.

2.10.5 Full-Band Peak Limiter

The peak limiter affects the full frequency range of the audio, after each of the four frequency bands has been compressed separately. The Threshold slider controls when the peak limiter will start working. With Threshold at 0, the peak limiter will have no effect. For mastering, it's recommended to have the Threshold set to around -3 or -4 dB. Severe Threshold settings will lead to pumping, which may or may not be desirable. The Attack and Release knobs control how the peak limiter responds to transient signals. The final Peak slider sets the output level of the signal. When Peak is set to its maximum 0 dB, then the audio will be as loud as possible. There's really no reason to ever set Peak lower than this, unless you needed to ensure a certain amount of headroom.

* You can play a sound through the built-in loop-player, through the realtime audio inputs, or you can process audio in realtime by using Reaktor as an effect plugin. Please check your Reaktor or Reaktor Session user's guide for helpful information.

2.11 Fusion Reflections



1.6 Fusion Reflections user interface

2.11.1 Introduction

Fusion Reflections is a delay-based effect that can create early reflections, shimmering choruses, fluttering delays, and even ambient reverbs. Two distinct diffusion engines are chained together to create an extremely wide range of effects. Each finely-tuned diffusion engine consists of four stereo modulation delays and an innovative graphical display that shows the actual delay time for each delay. Just five controls control the core parameters of each diffusion engine.

2.11.2 Quick Start

Play some audio through the effect* and select Preset 1 Long Decay Echo. This preset only uses the Chor Fusion diffusion engine. Play with the Diff Dly knob - note how the graphic display constantly updates to show you the current delay times. Stop the incoming audio to listen to the sound's decay. Play with Dly Mod and Speed to see what effect they have on the sound and the graphic. Be sure to check out all the other presets to see how versatile this effect can be!

2.11.3 Structure and Signal Flow

The sound passes through two diffusion engines serially. It's important to note that the two engines are similar but not identical - they each offer unique sound-shaping capabilities. The first diffuser, Chor Fusion, is the simpler of the two and is designed for early reflections, choruses, and atmosphere. If offers a high and low shelf EQ, with a graphical display of the EQ curve. The second diffusion engine, Echo Fusion, adds a feedback delay before the diffusion delays. The diffusion delays in the Chor and Echo engines are identical. Echo Fusion is perfectly tailored to late reflections, long delays, and long reverbs. A highpass filter (HP) cuts the low frequencies after the delay and before the diffusors, while a lowpass filter (LP) can reduce the brightness at the last step. The input to the Echo Fusion engine can be switched between the dry signal, and the signal coming out of Chor Fusion.

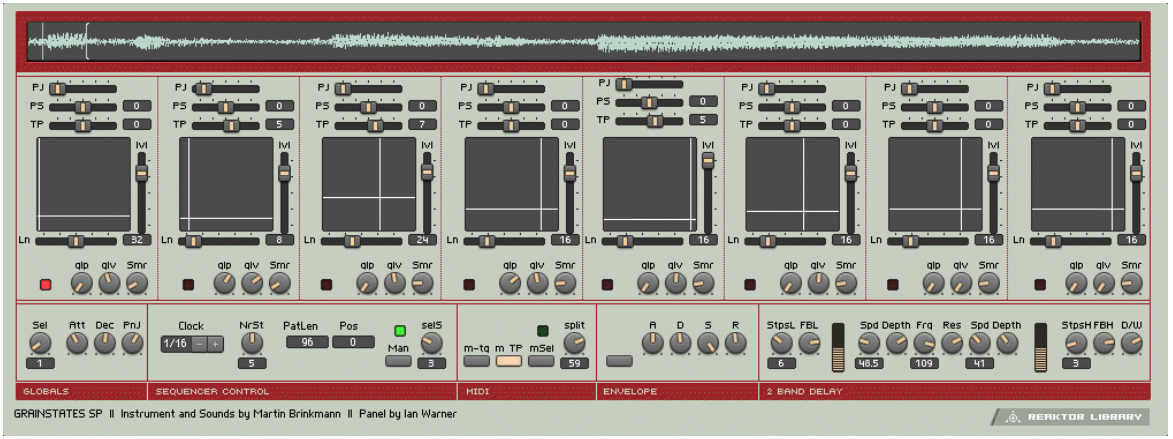
In the Mixer section the signals of Chor Fusion> and Echo Fusion are combined and mixed with the dry signal. Each section can also be switched on or off to save CPU.

2.11.4 Diffusion Delays

Both the Chor and Echo Fusion engines contain identical diffusion delays. As mentioned above, Echo fusion adds a single feeback delay before the diffusion delays, but the operation of the diffusion delay section is identical per effect. Each diffusion delay consists of four stereo modulation delays and an innovative graphical display that shows the actual delay time of each delay. The main delay time is controlled by the Diff Dly knob. The Dly Mod knob controls the amount of delay time modulation by the internal LFOs. Speed controls the speed of the LFO. Stereo sets the stereo spread of the delays, and Diffusion sets the inaccuracy of the delay times. The sum effect of these five knobs is graphically shown in the display underneath, where each "pendulum" represents the delay time of a single delay.

* You can play a sound through the built-in loop-player, through the realtime audio inputs, or you can process audio in realtime by using Reaktor as an effect plugin. Please check your Reaktor or Reaktor Session user's guide for helpful information.

2.12 Grainstates FX



1.7 Grainstates FX user interface

2.12.1 Introduction

Grainstates is a granular texture maker that works wonders for creating dense, breathing atmospheres. Taking advantage of Reaktor 4's grain cloud delay module, GrainStates lets you create granular soundscapes in realtime. You can even freeze the live audio - imagine playing a guitar into Grainstates, freezing the audio, then playing a counterpoint to the granular texture. Eight scenes - each scene storing information about grain size, density, pitch, pitch spread, and more - are sequentially recalled in sync with the master tempo. A dual-frequency delay adds depth to the sound by letting you specify independent delay and feedback times for the high and low frequencies.

Grainstates consists of two ensembles: GrainStatesFX and GrainStatesSP. GrainStatesFX is an effect using the grain cloud delay that works on live input, while GrainStatesSP is centered around the grain cloud sampler module. The FX ensemble has the advantages that you can

easily process audio without loading anything into a sampler and you can freeze the incoming audio stream - great for live performance! Since the sound passes through the ensemble, however, there's no way to save the sound data with the preset. GrainStatesSP stores the sample with the preset so it can easily be recalled at a later time, but you must first load your sound into the grain cloud sampler.

2.12.2 Quick Start

GrainStates FX: Start the system clock and run some audio through the ensemble.* You'll notice a graphical representation of the sound "marching through" the granular buffer. To freeze the buffer, press the Freeze button to the left of this graphical display. If you stop the system clock, the sound will continue, but the scenes won't advance.

GrainStatesSP: Start the system clock and step through the presets. If you stop the system clock, the sound will continue, but the scenes won't advance.

2.12.3 Structure and Signal Flow

At the heart of GrainStates is the granular grain cloud module. GrainStatesFX is based on two (for true stereo operation) grain cloud delay modules, while GrainStatesSP is based on a single grain cloud sampler module. Both the grain cloud delay and grain cloud sampler have identical controls, with the grain cloud delay adding the ability to freeze the sound. All of GrainState's controls (with the exception of the 2Band Delay) are used to control the grain cloud.

A master sequencer runs through eight scenes are run through sequentially, with each scene providing control over various granular parameters. Every scene can have its own length, settable by the Ln slider, whose units are set in the Seq Control macro. You can also set the total number of scenes (NrSt), and if you want to disable the scene sequencer simply click on "man" and you can select a scene manually with the SelS knob.

Each scene provide control over pitch jitter (PJ: amount of pitch randomization, in semitones), pitch shift (PS: in semitones), transposition (TP: in semitones), volume (Lvl), and an XY panel lets you set two parameters graphically at once: The horizontal axis sets the start position of the grain (relative to the graphical display on top of the internal buffer), while the vertical axis sets the length of the grain. Three additional knobs provide control over the smoothness of scene transitions, and the grain density smear (Smr).

GrainStatesFX only: The sound is filtered by the Filtor macro with independently adjustably highpass (HP) and lowpass (LP) frequencies. Res adjusts the resonance of both filters, while Byps disables the filter.

The output of the filters is fed back into the grain cloud delay, with feedback independently adjustable per scene with the FB slider to the right of the XY control. The feedback is only active then the grain cloud delay is not frozen. When the grain cloud delay is frozen, it ignores any signal to its inputs.

The sound then passes through a 2Band Delay, which gives independent control over delay time in sixteenth notes (StepsL and StepsH), feedback (FBL and FBH), and a filter-modulating LFO. The cutoff and resonance of the filter that splits the two bands is determined by the Frq and Res knobs. Finally, a D/W knob sets the mix level. To bypass the filter, simply set D/W to zero.

2.12.4 Additional Controls

Global Params. In the global parameters, you can set the global attack and decay of the grains, and the amount of pan jitter (stereo randomization). In GrainStatesSP, this is also where you select the active sample, with Sel.

GrainStatesFX only: the Move macro controls a built-in ramp oscillator that controls the delay time. The Steady knob is the amount of delay modulation - when at zero, then the ramp oscillator does not change the delay time.

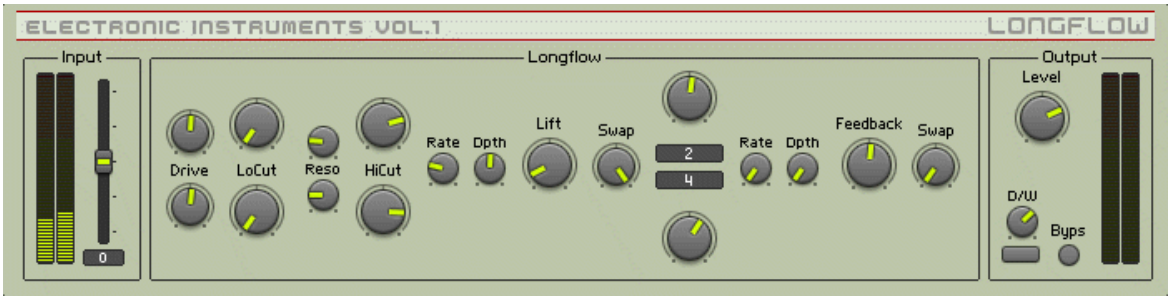
2.12.5 MIDI Control

In addition to automated sequencer control, GrainStates also lets perform with a MIDI keyboard. In the MIDI macro, you have control over the MIDI functionality. If "m TP" is activated, then MIDI notes will pitch the sound, like in a conventional sampler. In GrainStates, however, all notes are the same length, regardless of pitch. When "mSel" is active, each scene is mapped onto a note pitch between 48 and 59 (only the white keys of a keyboard); by pressing one of the notes the respective scene is selected. With the Split knob you can specify another keyrange that recalls scenes. "m-tg" toggles MIDI triggering of sound on and off. GrainStatesFX only: "m-frz" lets you toggle the freeze effect on/off via MIDI.

GrainStatesSP only: If you play GrainStatesSP over MIDI, you can activate the envelope (env macro) to also control the amplitude of the sound with MIDI, according to the settings of the envelope.

* You can play a sound through the built-in loop-player, through the realtime audio inputs, or you can process audio in realtime by using Reaktor as an effect plugin. Please check your Reaktor or Reaktor Session user's guide for helpful information.

2.13 Longflow



2.5 Longflow user interface

Introduction

Longflow is a delay effect with a finely-tuned feedback loop to create very slowly decaying echoes, or rhythmic grooves.

Input

<Level> Sets the input level, in dB. The meters show the post-fader input level.

Longflow

In Longflow, all controls that are vertically mirrored are separate controls for left and right. Left is the top channel, and R is underneath. The following controls are mirrored — there is one control each for the L and R channels:

- <Drive> Sets the amount of amplification performed to the left incoming signal. Use mid position for no level change.
- <LoCut> Adjusts cutoff frequency of a 4-pole highpass filter. Together with the lowpass filter that is controlled by the higher bounding frequency, they perform a bandpass filtering.
- <Reso> Sets amount of resonance for the filters.
- <HiCut> Adjusts cutoff frequency of a 4-pole lowpass filter. Together with the highpass filter that is controlled by the lower bounding frequency, they perform a bandpass filtering.
- <Delay L, R> Sets the delay time. The display shows the delay time in 16th notes according to the MIDI tempo. Internally, the linear output of the knob is processed by a function mapping it onto a grid to allow very short decays as well as very long decays. The grid also performs triplet division of the time (0.3333, 0.1667, 0.0833 etc.).

The following controls effect both channels:

- <Rate> Sets the rate of a low freqeuncy oscillator whose output is modulating the higher boundary freqeuncy of the filter. Turn right for higher rates, i. e. faster modulation.
- <Dpth> Sets the amplitude of a low freqeuncy oscillator whose output is modulating the higher boundary freqeuncy of the filter. Turn right for higher amplitudes, i. e. more modulation of the freqeuncy.
- <Lift> Sets the maximum level lifting. Turn right to enhance very quiet signals, esp. in a long feedback tail with decaying audio. Be careful with this knob as also quiet noise and clicks etc. are amplified and become very audible. Its effect increases if the drive knobs are turn to the left.
- <Swap> Sets the amount and mode of channel swapping of the incoming signal. Turn to the left for no swapping, i. e. the left channel remains the left one, the right channel the right one. Turn to mid position for mono processing, i. e. both signals occur in both channels. Turn to the right for crossfading, i. e. the left channel signal becomes the right channel signal and vice versa.
- <Rate> Sets the rate of a low freqeuncy oscillator whose output is modulating the delay time. Turn right for higher rates, i. e. faster modulation.

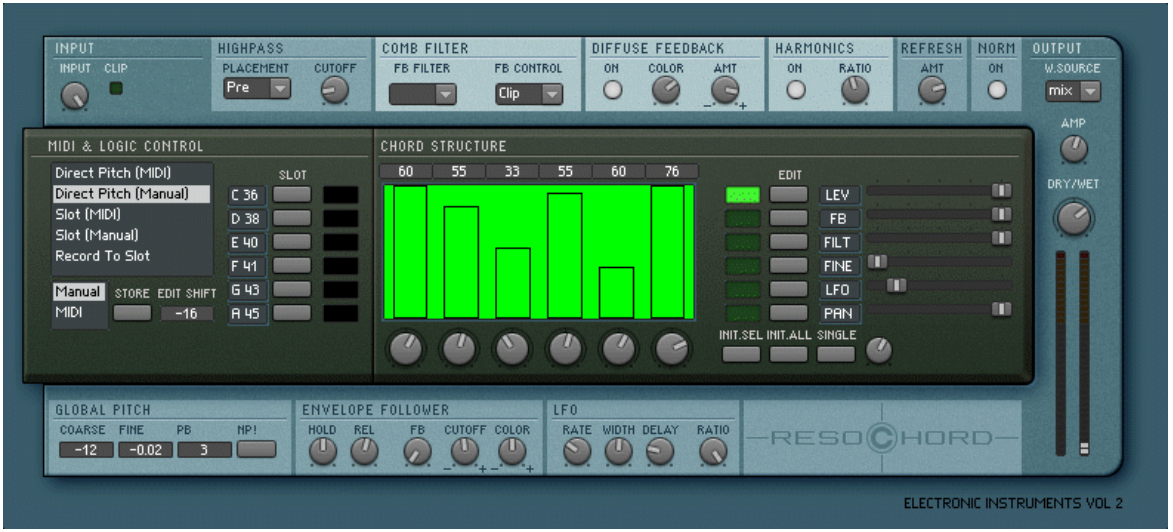
- <Dpth> Sets the amplitude of a low freqeuncy oscillator whose output is modulating the delay time. Turn right for higher amplitudes, i. e. more modulation of the delay time.
- <Feedback> Sets the feedback level, i. e. the loudness of the delayed signal that is mixed with the incoming signal and delayed once more.
- <Swap> Sets the amount and mode of channel swapping of the output signal. Turn to the left for no swapping, i. e. the left channel remains the left one, the right channel the right one. Turn to mid position for mono processing, i. e. both signals occur in both channels. Turn to the right for crossfading, i. e. the left channel signal becomes the right channel signal and vice versa.

Output

- <Level> Sets the output level, in dB. The meters show the post-fader output level.
- <D/W> Dry / Wet control. It crossfades between the incoming signal (fully left) and the processed signal (fully right).

<Byps> Bypass switch which turns the effect off. When off, the CPU is no longer used.

2.14 Resochord



3.8 Resochord user interface

Introduction

Resochord imprints harmonic structures onto any input signal – including percussive sounds with no existing harmonics. It essentially consists of 6 feedback units, which can be considered analogous to the strings of a guitar, which are effectively 'plucked' by the shape and rhythm of the input signal. As each 'string' resonates at a different frequency, rich and luscious chords are generated – and thus remarkably, any input signal can be transformed into a unique, MIDI playable instrument.

Quick Start

As an effect unit, Resochord requires an input signal. In REAKTOR standalone mode, the REAKTOR tape-player is an ideal source. To open the tape-player, click on the REAKTOR 'View' menu, and select, 'Show Playerbox', then load an audio file as desired.

There are various ways with which to control the pitch of the 6 resonators, as detailed later in this manual. However, to play Resochord in real time from your MIDI keyboard, select **Direct Pitch (MIDI)** from the MIDI & Logic control menu on the ensemble panel.

Signal-flow Overview

First the input signal passes through an optional high-pass filter and is then routed to the main resonators. The output from the resonators is passed on to a high shelf equalizer (Refresh) and a dynamic range controller (Norm) before it is mixed again with the dry, unprocessed input signal and sent to the output.

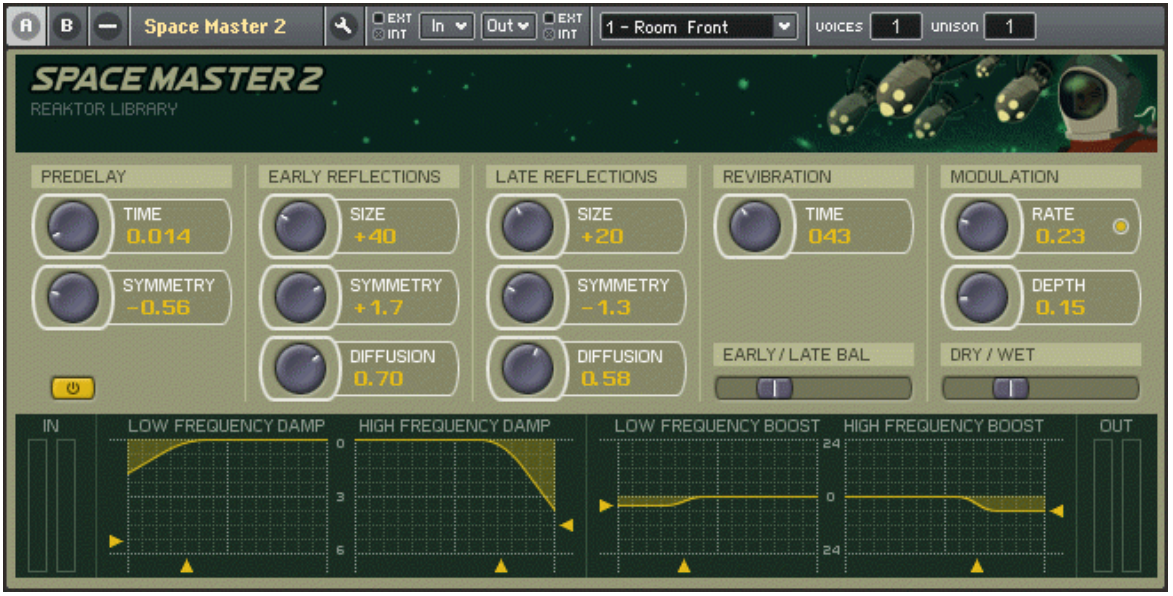
Section	Function
MIDI & Logic Control	The main menu in this section selects from various modes of controlling the pitch of the 6 resonators. Regardless of the control method, the pitch of the resonators is always displayed in the 6 values at the top of the Chord Structure menu. When **Direct Pitch (MIDI)** is selected, the pitch of resonators is determined by MIDI note input. This mode enables live play as with any keyboard instrument. **Direct Pitch (Manual)** enables the 6 large pitch knobs at the bottom of the Chord Structure section. The Slot modes allow the user to store and recall the pitch of the resonators in 6 memory slots. This offers an alternative method of live performance (especially for those with lesser keyboard skills!), allowing the user to recall from the 6 predefined pitch configurations as and when desired. When **Slot (Midi)** is enabled, the 6 memory slots are recalled by pressing MIDI notes C36, D38, E40, F41, G43, and A45. Alternatively, in **Slot (Manual)** mode, the 6 slots can be recalled by the 6 slot buttons on the panel. To store a pitch configuration to a slot, select **Record to Slot**, then select the desired destination slot by clicking on the appropriate slot button. In **Manual** mode, firstly configure the desired 6 pitches using the knobs in the Chord Structure section, and then press **Store**. In **MIDI** mode, firstly select the desired slot, and then simply press the desired notes on your MIDI keyboard.
Chord Structure	As mentioned above, the 6 values at the top of this section display the current pitches of the resonators. The 6 knobs at the bottom are used for defining the 6 pitches in manual pitch modes. The main window allows configuration of various parameters for the 6 resonators. Adjustable parameters are **Lev** (volume level), **FB** (resonator feedback amount), Cut (filter cutoff), **Fine** (fine pitch adjustment), **LFO** (amplitude modulation depth by the LFO), and **Pan** (position in the left-right stereo field).

Section	Function
	To select which parameters to edit, click on the button next to the parameter label. Then click and draw in the main window to edit the parameter for the 6 resonators. **InitSel** or **InitAll** reset the values for the 6 resonators for either the selected parameter, or all parameters respectively.
Input	**Input** defines the amplitude of the input signal (0 - 100%). The **Clip** lamp will light if the input signal is too loud.
Highpass	This section applies a 12db highpass filter to each of the 6 resonators. The cutoff frequency is linked to the pitch of each resonator, **Cutoff** sets the offset between each filter cutoff and the actual resonator pitch (-40 to +40 semitones). **Placement** defines the filter's position within the signal flow. It can be placed before the main comb filter section (**Pre**) or after it (**Post**), or disabled altogether (**Off**).
Resonator	A resonator is essentially a feedback loop with a short delay time, the length of which determines the resonator pitch. While the specific configuration of the 6 resonators is defined in the Chord Structure section, the Resonator section allows configuration of 2 global parameters – the **Feedback Filter** mode (lowpass or notch) and the method of **Feedback Control**, required to prevent overflow from the feedback. **Sat** and **Clip** offer soft-curve, or hard-clipping of feedback above 0db, and both work on individual samples, whereas **Limit** operates more akin to an analog compressor, reducing the volume of the feedback when the signal exceeds 0db.
Diffuse Feedback	The Diffuse Feedback unit applies an allpass filter to the feedback chain that adjusts the signal's phase for interference effects. **Color** sets the center frequency (26..8400 Hz) of the feedback's allpass filter. As the filtered signal is added to the unfiltered one, some special interference effects become audible. **Amount** sets the amount of diffuse feedback. This is a bipolar control – when centered there is no feedback, at the right there is normal feedback, and at the left the feedback signal is inverted for special interference with the allpass filter.

Section	Function
Harmonics	This section adds additional harmonics to output of each resonator. In essence it is an additional resonator stage (without feedback) for the output of each main resonators. The **Ratio** knob adjusts the ratio between the original pitch and the additional harmonic pitch. It ranges from infinitely higher (at hard-left) to 1 octave higher (i.e. twice as high) (at hard-right) than the main resonator pitch.
Refresh	Sets the amount of amplification (0db to 10db) applied to frequencies above 1 kHz. Use this knob to 'refresh' dull sounding signals.
Norm	Enabling this section maximizes the volume of the output signal.
Output	**W. Srce** (Wet Source) determines whether the output signal is taken from the beginning of the comb filter's feedback loop (**Pre**), or from the end (**Post**). If the former, the dry, unprocessed signal is more prominent in the wet signal. **Amp** sets the amplitude of the wet signal before it is mixed with the dry signal. **D / W** (dry / wet) defines the balance between the dry, unprocessed signal (left) and the wet signal (right).
Global Pitch	**Coarse** and **Fine** shift the overall pitch of the resonators in semitones. **PB** sets the pitch bend range in semitones, and **NP!** enables global pitch shifting by MIDI note keys C60 to C96.

Section	Function
Envelope Follower	The envelope follower tracks the amplitude of the input signal, and then uses the derived signal to modulate resonator feedback, cutoff and colour. **Hold** adjusts the length of the hold period of the envelope follower after an envelope peak in the input signal (1 to 10000 msec), and **Release** adjusts the length of the release period (1..10000 msec). **FB**, **Cut** and **Colour** define the extent to which the envelope follower modulates resonator feedback, cutoff and diffuse feedback colour respectively. Note that as that FB is unipolar, setting to hard-left results in no modulation, whereas Cut and Colour are bipolar, and thus setting to centre results in no modulation.
LFO	**Rate** specifies the oscillation rate of the LFO (0Hz to 3.6 Hz). **Width** sets the symmetry of the LFO shape. **Delay** specifies the amount to which the LFO modulates the resonators delay time (i.e. amount of vibrato). **Ratio** sets the amount of influence of the LFO on the Harmonics' Ratio.

2.15 Space Master 2



4.3 The Space Master 2 user interface

2.15.1 Introduction

The well-known Space Master series of reverb modellers has been updated for REAKTOR 5. Based on several diffusion delays, Space Master 2 can produce a wide array of high-quality natural or experimental ambiences. The patch's efficient set of reverb parameters include an early reflections section, a late reflections module and a post EQ. Dials for main reverb time, control of balance between the two reflection stages, and between dry and wet signal round off the controls.

2.15.2 Input and Output Stage

You can introduce an initial delay into the reverb signal with the predelay [Time] dial and control the predelay's stereo position with the [Symmetry} knob. The [Early / Late Balance] slider can be used to move the source in space – more early reflections bring the signal to the front and more late reflections make it appear further back in space. At the end of the signal chain, the [Dry / Wet] slider crossfades between the dry original signal and the processed sound.

Section	Control	Function
Predelay	Time	Sets an initial delay for the wet signal.
	Symmetry	Introduces a difference into the delay times for the right and left predelay channels. Use this to shift the signal around in the stereo image.
Mixing	Early/Late Balance	With this parameter you can set how much of the early and late reflections, respectively, can be heard in the output.
	Dry / Wet	This controls the balance between dry and wet signal.

2.15.3 Reflections

Use the two [Size] and [Diffusion] parameters to dial in the early and late stages of variable density diffused reflections. The early stage commonly represents the direct response of the virtual space, whereas the late reflections define the sound when the early reflections have died away.

For dynamic reverb effects you can use the Modulation section. It offers an LFO routed to the delay times with [Rate] and [Depth] control. The LFO can enhance your reverb signal by adding liveliness.

Section	Control	Functions
Early/Late Reflections	Size	Determines the range of space generated by the early or late reflections modules by adjusting delay time of the underlying diffusion delays. Higher values give the impression of larger spaces.
	Symmetry	Introduces a stereo shift into the generated reflections.
	Diffusion	Adjusts the perceived density of the generated reflections. Dial for a sparser or fuller reverb sound.

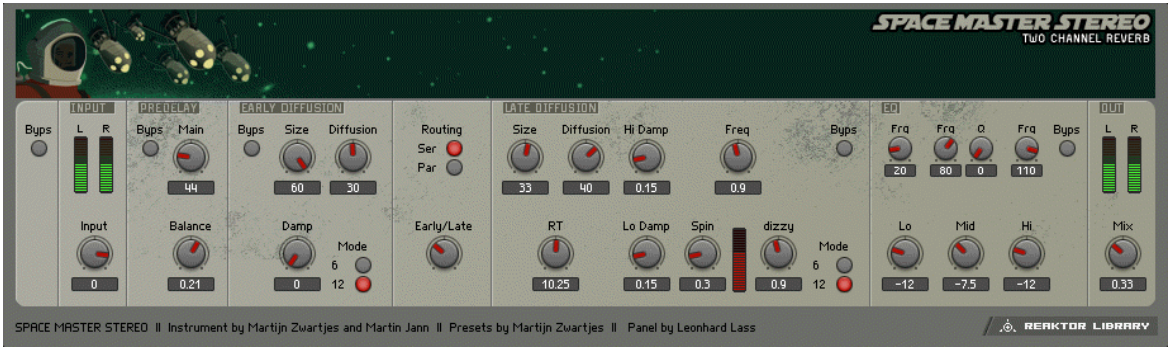
Section	Control	Functions
	Reverberation Time	This control alters the decay time of the reverb response.
Modulation	Rate	Control of LFO frequency modulating the delay times.
	Depth	This adjusts the LFO's modulation depth. Higher values give you higher amplitude of the modulation.

2.15.4 Frequency Response

The two EQ sections serve slightly different needs. The Damping EQs are integrated into the reflection stages and influence their frequency responses. The Post EQ acts on the main output of the patch should be used to color the overall sound.

Section	Control	Function
Frequency Damping	Low Frequency Damp	Low shelving filter that cuts into diffusion delay frequency response of both early and late reflections. Use the horizontal slider to adjust cutoff frequency and the vertical slider to adjust cut or boost.
	High Frequency Damp	High shelving filter that cuts into diffusion delay frequency response of both early and late reflections. Use the horizontal slider to adjust cutoff frequency and the vertical slider to adjust cut or boost.
Post EQ	Low Frequency Boost	A low shelving EQ that acts on the main output of the reverb. Use the horizontal slider to adjust cutoff frequency. The vertical slider adjusts cut or boost.
	High Frequency Boost	A high shelving EQ that acts on the main output of the reverb. Use the horizontal slider to adjust cutoff frequency. The vertical slider adjusts cut or boost.

2.16 SpaceMaster



1.8 Space Master 1 user interface

2.16.1 Introduction

The SpaceMaster series of reverbs breaks new ground in reverb modelers for Reaktor. Spacemaster uses two different Diffusion modules to achieve stunningly convincing room sounds. And to fully exploit SpaceMaster's lushness, there are three versions - SpaceMaster stereo, SpaceMaster Quad, and SpaceMaster 5.1 Surround.

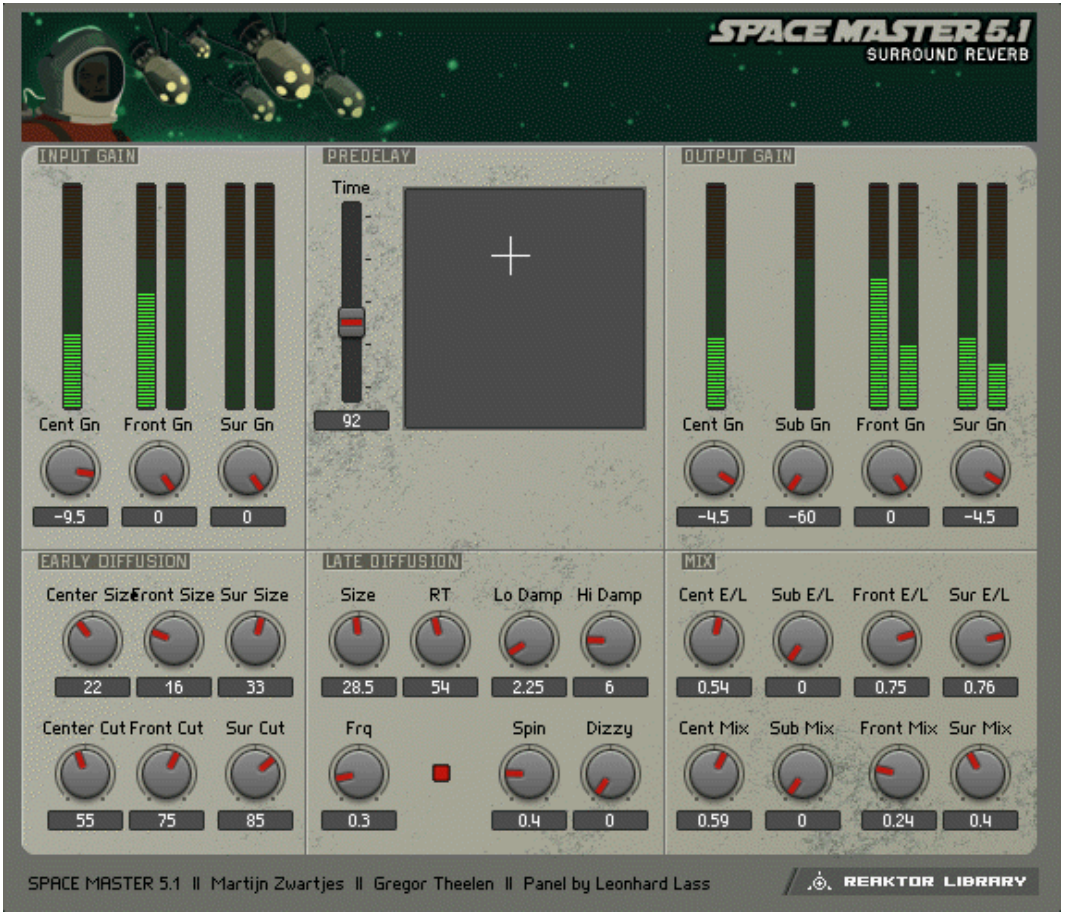
Quick Start

To really get a feel for the kinds of lush atmospherics SpaceMaster can provide, it's a good idea to either hook a beat looper or external sound source up to it, or, to utilize it in plugin mode in your favorite audio sequencer. Stepping through the presets should give you a good impression of the kinds of real and imaginary spaces SpaceMaster can emulate. Adjusting the controls of the Early and Late Diffusion modules will have the most effect on the sound, and will give you an idea of how the two main components interact in creating ambiences - especially since they can be arranged in serial or parallel signal paths.

Structure and Signal Flow

This guide will use the Stereo SpaceMaster ensemble to outline the various controls for shaping the reverb signal, since most of them are the same among the three types of SpaceMasters. Read on below to learn about the specifics of the Quad and Surround flavors of SpaceMaster reverb.

2.16.2 SpaceMaster 5.1 Surround



1.9 Space Master 5.1 Surround

The Surround version of SpaceMaster is effectively the same as the Quad version (↑2.16.3, SpaceMaster Quad), except that it adds a Center channel. The PreDelay XY pad adds control for the center channel. SpaceMaster Surround also adds an Output Gain section that allows you to use precise metering to control the relative volumes of the Center, Sub, Front, and Sur-

round reverb signals. The balance of Early/Late signal can also be adjusted for each channel using the Mix module, which has familiar controls for wet/dry levels. Finally, SpaceMaster Surround does not include the EQ module found in the Stereo and Quad versions.

2.16.3 SpaceMaster Quad



1.10 Space Master Quad user interface

The SpaceMaster Quad ensemble works in the same way as the stereo version (↑2.16.4, SpaceMaster Stereo), but it splits the reverb signal into four discrete outputs for using in a four-speaker system. The signal for the forward speakers is referred to as the Front signal, while the signal for the rear speakers is referred to as the Surround, or Sur, signal. In the Input section, you can adjust the gain for Front and Sur signals to determine how much signal sources reaches each signal chain.

The PreDelay section is simplified from its stereo brother by use of an XY pad to finely adjust Front and Surround L/R offset. The overall PreDelay time can be controlled by the Time fader.

The Early Diffusion section is also simplified, offering only Size and high frequency Cut controls for both Front and Surround diffusion signals.

The Late Diffusion module is effectively the same as the one found in SpaceMaster Stereo. Refer to the description above the learn how it works.

The Mix section allows you to balance the Front and Surround signal amount and to adjust the wet/dry mix for each.

2.16.4 SpaceMaster Stereo

SpaceMaster Stereo is composed of a PreDelay module, an Early Diffusion module, a Late Diffusion module, and a post EQ. The Diffusion modules can be combined together to create a complex impression of space. By adjusting the balance of the Early and Late diffusion modules, you can precisely move the origin of the reflections from near to far (or front to back), making SpaceMaster perfect for Surround mixing situations in which a truly room-filling reverb can be created.

The Input section, at the fat left, allows you to trim the input gain to avoid overloading the audio signal. The input is next processed by the PreDelay. Use this to add an initial delay to the wet signal. You can also use the L/R offset knob to add some perceived stereo width by altering the delay time of the left-side single delay module. You can bypass the PreDelay with the Byps button.

Next in the signal path, the Early Diffusion module, which is actually a series of up to 12 diffusion delays, provides the near reverb processing. The Size control allows you to determine the range of space that the close reflections will be generated by. It changes delay time in milliseconds. The Diffusion knob lets you adjust the density of the reverb signal. To further adjust the reverb depth of the Early Diffusion module, the Mode switch allows you to select 6 or 12 diffusion modules. Watch your CPU load carefully to make sure your computer can handle the strain of processing with 12 (or 24) diffusion modules. The Damp knob controls the frequency of a 1-pole low pass filter for attenuating high frequencies. As in the PreDelay section, clicking the Byps button will take the Early Diffusion module out of the signal path.

The Routing switch, located between the Early and Late Diffusion modules, lets you determine how the input signal will be routed through the two modules. The Ser button engages Serial mode, where the Early Diffusion module is simply routed directly into the Late module. The Par switch engages Parallel mode, allowing the Early and Late modules to maintain separate signal paths, until they are mixed at the EQ module. You can also use the Early/Late knob to balance the amount of early and late reverb signal. This is a great way to change the perception of location within a space, by shifting between the early and late reverb sounds (to make it seem like a sound is moving around inside of the "room").

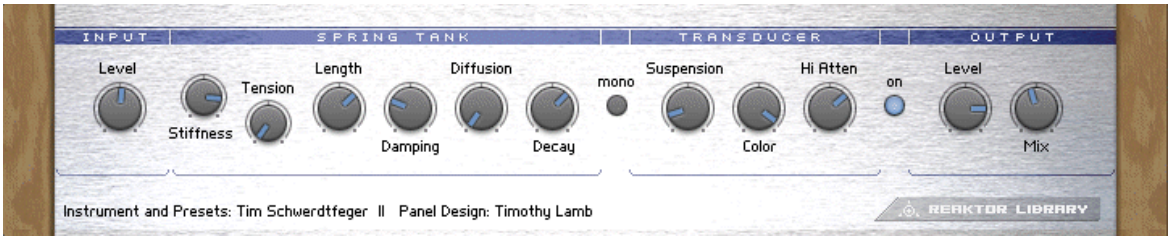
The Late Diffusion module provides the capabilities for creating a larger and more richly defined space. The Size and Diffusion knobs work the same way as those in the Early section, but there a few new options. The RT knob controls a feedback loop, allowing you to stretch out the apparent reverb time, or the time it takes for the echo to return to the point of origin. There are

also controls for high and low shelving EQs, called Hi Damp and Lo Damp, respectively, that allow you to shape the frequency range of the reverb signal. You can modulate the delay time and apparent position of the reverb signal by using the modulation controls. The Spin knob adjusts the amplitude of a sine wave LFO, while the Dizzy knob controls amplitude for a Slow Random LFO. You can create complex modulations by adjusting the balance of sine wave and random LFOs. The Freq knob controls the rate of both LFOs together. As in the Early section, 6 or 12 diffusion modules can be selected with the Mode switch. This section can also be bypassed with the Byps button.

Both the Early and Late Diffusion sections feed into the EQ. The EQ consists of stereo low shelf, parametric, and high shelf filters. Starting at the left, the Lo knob lets you control attenuation or boost of the low frequency set with the Frq immediately above. The Mid knob controls the parametric EQ band. The Frq and Q knobs above it let you adjust the frequency and bandwidth of the parametric. The Hi knob controls attenuation of the high shelf filter. Use the Frq immediately above the Hi knob to adjust the frequency. As in the other modules, the EQ can be taken offline with the Byps button.

Finally, the mixture of wet and dry signal can be adjusted to taste with the Mix knob in the output section.

2.17 Spring Tank



1.11 Spring Tank user interface

2.17.1 Introduction

Spring Tank pays homage to the trashy, unrealistic spring reverbs of the past. While Spring Tank isn't exactly a physical model of a spring, it goes a long way toward recreating the spring reverb characteristics: dull, transducer-saturated, and boingy, with the familiar nonlinear resonating decay.

2.17.2 Quick Start

Spring Tank is meant to be an experimental effect, rather than a realistic one, play some audio through the effect* and experiment! It allows control over the elements of the "spring's" morphology, so you can design your own spring.

2.17.3 Structure and Signal Flow

The input source is amplified or attenuated with the Level knob at the far left of the instrument. High levels here can introduce transducer saturation (which is not such a bad thing). The signal is then fed into the spring tank. Here, you can adjust the spring's physical characteristics. The Damp knob sets high frequency damping for the spring. Turning the knob clockwise increases the damping. Stiffness, which mimics the flexibility of the spring, gives brighter, more resonant sounds at higher settings. The Shape knob allows you to crossfade between a round spring shape to the left, and a rectangular one to the right. The round shape emphasized a ringing sound, while the rectangular shape creates less ringing but a more diffuse sound.

The Thickness of the spring determines the virtual diameter of spring winding, and therefore the overall length of the spring. Longer spring settings result in reduced brightness. The Length setting reflects global length. Turn this up to increase the shattering decay sounds. The Decay knob changes the length of the decays. It alters the amount of feedback in the system.

Switching the Mono button off engages an additional short delay in one channel, resulting in the simulation of stereo.

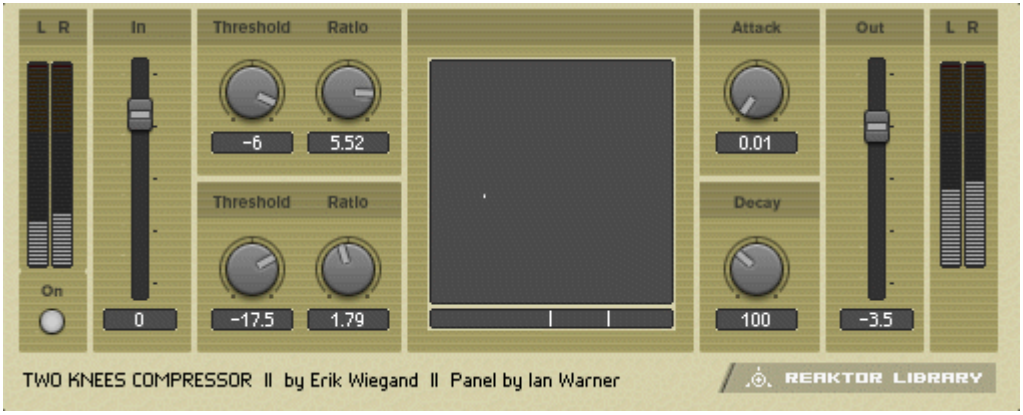
The Suspension section allows you to mimic the type of spring suspension in the system. When you turn the knob to the left, you increase the "softness" of the suspension material. This will lengthen the decay rate of lower frequencies. What this is actually doing is adjusting the cutoff

of a high pass filter in the feedback loop. Suspension will therefore also have an effect on the sound of the Transducer saturation. The Color knob lets you adjust a global tone control. Turning this clockwise results in a brighter sound, and vice versa.

The Level knob controls the volume of the wet signal. You can adjust the balance of wet signal to dry signal with the Mix knob. Finally, you can bypass the effect all together by clicking the On button.

* You can play a sound through the built-in loop-player, through the realtime audio inputs, or you can process audio in realtime by using Reaktor as an effect plugin. Please check your Reaktor or Reaktor Session user's guide for helpful information.

2.18 Two Knees Compressor



1.12 Two Knees Compressor user interface

2.18.1 Introduction

Two Knees Compressor is a simple compressor with an important quirk it has two separate adjustable thresholds and ratio controls. 2-Knees Compressor can perform as a precisely accurate compressor/limiter, a distorting sound shaper, a transient modifier, or anything in between. It includes a compression curve display to show the relationship between the amplitude of the input signal and the amplitude of the processed signal.

2.18.2 Quick Start

Play some audio through Two Knees*. Try to input a beat and run through the snapshots to hear a clear example of how it can alter transients.

2.18.3 Structure and Signal Flow

Set the input level with the In fader to achieve 0 dB on the input meters. The top set of Threshold and Ratio knobs refer to the upper threshold of the compressor. The upper compression ration is applied if the amplitude of the input signal exceeds the upper threshold. The ratio determines how many dB the signal must exceed the threshold to result in a 1dB increase of the output signal. The lower threshold and ration controls are directly beneath the upper controls. The lower compression ratio is applied if the amplitude signal is within the range between the upper and lower thresholds.

The Thresh setting shows up on the display just below the compression curve display. The upper threshold setting will be on the right side of the display and the lower will appear on the left. This will help you to dial in your threshold range to get particular hard-knee or soft-knee sounds. Try adjusting the degree of separation between the upper and lower thresholds and watching the display to see where the compression crosses from sharp-angled hard-knee style to rounded soft-knee.

You can adjust the attack time of the compression in milliseconds using the Attack knob. The Attack setting determines how fast the compressor responds to (turns down, basically) a signal that exceeds the threshold limits. Slower attack times tend to let more transients through. The Release knob lets you set how fast the compressor returns to unity gain (or zero gain) after falling below the threshold limits. You can manipulate the apparent sustain time of some sounds with the Threshold adjustment. Long release times tend to sound more "natural".

Finally, you can make up for gain reduction after compression by adjusting the Out level. You can also bypass the compressor by turning off the On switch.

* You can play a sound through the built-in loop-player, through the realtime audio inputs, or you can process audio in realtime by using Reaktor as an effect plugin. Please check your Reaktor or Reaktor Session user's guide for helpful information.

3 Grooveboxes

3.1 Aerobic



2.1 Aerobic's user interface

Aerobic is a step sequencer that controls a virtual analogue drum synthesizer. The instrument produces tight, innovative sounds far beyond the range of traditional drum computers. This, combined with the sequencer's capacities and the mixer's flexible routing options, makes Aerobic a versatile beat production environment that can be used in live performances.

The drum synthesizer contains six similar, independent units (selectable by the tabs at the top of the panel). Each unit combines an oscillator and a noise section into one signal that can be equalized before it is sent to the master mixer. The sequencer (in the middle of the panel) contains two tracks for each sound unit, selectable via the same tabs as the units themselves. The filled rectangles within the sequencer's display represent the unit's trigger signals and their velocity; the unfilled rectangles form a modulation track whose signal can be used to change nearly every parameter of the sound engine and mixer over time: Use the [Modulation] switch in the unit's master section to select the destination of the modulation. The master mixer provides classical mixing parameters for each unit (solo/mute, pan, and, of course, level), along with controllers to adjust the complete ensemble's reaction on MIDI messages. Each unit can be triggered by a selectable MIDI note; on a more complex level, note messages can recall complete ensemble snapshots.

3.1.1 Sound Engine

The drum synthesizer is built by two sound generators, an equalizer, and a master section that also controls modulation routing. While the oscillator part (on the left side) is based on sine waveforms with frequency modulation capacities, the noise part (on the right side) contains a white noise generator with a multi-mode filter. The mixed signal is sent through an EQ and (within the master section) a final saturator unit before it is passed on to the mixer.

Section	Control	Function
Oscillation	Envelope	Selects the operation mode of the envelope shaping the unit's amplitude. [Lin] activates a standard AD envelope whose transition times are controlled by the [Attack] and [Decay] knobs. When in [Roll] mode, this envelope is re-triggered fast until the next beat; the [Attack] knob in this case also controls the re-triggering frequency. [Roll+Lin] adds both signals of the modes described above. [Noise Env] uses the envelope of the Noise section (see below).
	Attack	Sets the time that passes until the amplitude envelope reaches its peak. In [Roll] mode (see [Envelope]) the knob also controls the rate at which the envelope is re-triggered.

Section	Control	Function
	Decay	Sets the time that passes after the amplitude envelope has reached its peak before it decays to silence.
	Oscillator	Selects the operation mode of the oscillator. While [Sin] represents a standard sine wave, [Sin2] activates a squared sine wave with a different frequency spectrum. Similarly, [FM2] selects the squared signal of [FM] which is generated by a sine oscillator modulating the frequency of another one. (This frequency modulation does not interfere with the modulation controlled by [F-Mod], [F] and [Fmod].) [Phase] uses the output of a phase oscillator.
	F-Mod	Selects the source signal used to modulate the main oscillator's frequency. While [Osc Env] and [Noise Env] select the respective amplitude envelopes, the [Sine], [Tri] and [Random] entries use independent oscillators whose frequency can be adjusted with [Rate].
	F	Sets the base frequency of the main oscillator.
	FMod	Sets the amount of frequency modulation applied to the main frequency by the selected source signal.
	Rate	Sets the frequency of the independent oscillator modulating the main oscillator's frequency.
Mix	Mix	Sets the ratio of the oscillator section's output and the noise section's sound in the signal that is passed on to the equalizer.
Noise	Envelope	Similar to [Oscillator][Envelope], applied to the noise generator's filter.
	Attack	Similar to [Oscillator][Attack], applied to the noise generator section.
	Decay	Similar to [Oscillator][Decay], applied to the noise generator section.
	Noise	Selects the operation mode of the noise section. [White] uses unfiltered noise, [White Mod] modulates the noise generator's algorithm by the noise section's envelope signal.
	Filter	Selects the type of 2-pole filter applied to the noise. Highpass, bandpass, and lowpass filters are available, providing 24 dB damping per octave.
	Freq	Sets the center frequency of the filter.
	Peak	Sets the amount of modulation applied to the filter's center frequency by the envelope.
	Res	Sets the amount of filter resonance.
EQ	Hz	Sets the frequency of the equalizer.

Section	Control	Function
	dB	Sets the amount of volume boost (or cut) applied to the adjusted frequency.
Master	Modulation	Selects the target of the sequencer's modulation track. The modulation shows no effect until the [Track] button is pushed.
	Track	Activates the modulation of the target selected by the sequencer's modulations track.
	Amp	Sets the amplitude of the signal before it is routed to the final shaper unit (see [Shape]).
	Shape	Selects the operation mode of the shaper unit. [Polysat], [Sinesat], and [Hypersat] saturate the signal with tube-like effects; the effect increases the more the signal is amplified before (see [Amp]). [Clean] doesn't perform any compression; [Amp] simply controls the amount of amplification before the signal is routed to the master mixer.

3.1.2 Sequencer

The sequencer provides two tracks for each of the six drum synthesizer units: a gate pattern and a modulation track. The gate pattern determines the trigger signals and their velocity. The modulation track signal can be routed to any parameter of the sound engine (see [Sound Engine][Master][Modulation]). A roll mode bar provides three different roll modes for fast re-triggering of a drum sound.

Control	Function
Tempo	Selects the tempo of the track: Each step of the sequence can be interpreted as sixteenth note, etc. Thus, the sequencer is always synchronized to the master MIDI clock; use the host sequencer or REAKTOR's internal MIDI clock to start the sequencer. (See also [Global Tempo].)
Global Tempo	Sets the [Tempo] value of all six tracks.
Swing	Sets the amount of swing, i.e. the amount by which every second step of the sequence is delayed to shuffle the strict MIDI rhythm.
Roll Factors	Sets the number of times the trigger signal is repeated if the [Roll Mode] is set to the respective colors.
Init All	Deletes all sequence patterns and modulation tracks and sets [Swing] to its default values.

Control	Function
Track	Selects the track that can be edited within the [Edit Display]
Step Count	Displays the number of the current step (1 to 16). If the gate is off, the number is dark; if the gate is on, the number is light. This can be helpful when editing the modulation track.
Edit Display	Displays the trigger pattern (filled rectangles) as well as the modulation track (unfilled rectangles), depending on the [Track] setting. Clicking within the display allows the patterns to be edited. High values of the trigger pattern represent high velocity; in the modulation track they cause the modulated knob to turn to the right, and low values turn the knob to the left.
Roll Mode	Selects how often the trigger signal is sent. Normally, it is only sent once per beat; by clicking with the mouse one can step through three differently colored modes where the trigger signal is sent more often (see [Roll Factors]).
Loop	Controls the length and position of the played sequence: Only those steps within the rectangle are used. Drag the ends of the rectangle to adjust the loop's start and end points. A second, smaller bar represents the current read out's position.

3.1.3 Master / Mixer

This section has two functions. First, it mixes the six drum synthesizers down to a single signal – or to four signals if [Single Outs] is activated. Second, it controls the snapshots of the complete ensemble as the sound engine and the sequencer are slaved to this part of the instrument. An advanced recall system allows for fast changes of sound/pattern settings via a single MIDI note, making the complex drums computer-controllable from a keyboard in live stage use.

Section	Controls	Function
Mixer	Level	Sets the volume of the sound unit.
	Solo	Switches the sound unit to solo playing, i.e. mutes all other units.
	Mute	Mutes the sound unit.
	Pan	Positions the sound unit's mono signal within the stereo field.
	Wide	Enhances the spatial appearance of the sound unit.
	Ext. Learn	Activates the learn feature. When pressed, the next MIDI note will be assigned to this track and can be used as external trigger signal, in addition to the internal gate signals of the sequencer. (See also [External].)
	Output	Selects to which of the four stereo outputs the sound unit is routed. This shows no effect until [Single Outs] is activated.

Section	Controls	Function
Snapshot	Power	Switches the snapshot handling on or off.
	Key	Turns on or off snapshot recall by external MIDI note messages. See [Root Note] and [Root Snap] for details.
	Quantize	Turns quantization of external MIDI notes on or off. When on, incoming MIDI messages will be synchronized to a pattern that is selected by [Quantization Select].
	Quantization Select	Selects the quantization pattern to which external MIDI messages can be synchronized.
	Snapshot	Recalls a snapshot of the master mixer. Since all other components are slaved to this one, storing or recalling a snapshot here affects all other instruments, i.e. the sound units and sequencer.
	Root Snap	Sets the snapshot number recalled when the MIDI note adjusted by [Root Note] is received; [Snap Via Key] has to be activated. The note above [Root Note] recalls the snapshot that follows on the [Root Snap] etc.
	Root Snap Learn	The first snapshot recalled after pressing this button will be used as new [Root Snap].
	Root Note	Sets the external MIDI note that, if [Snap Via Key] is on, recalls the snapshot adjusted by [Root Snap].
	Root Note Learn	The first MIDI note received after pressing this button will be used as new [Root Note].
	Store	Stores the current settings of the complete ensemble to the current snapshot number (see [Snap]). If [Store To Next Snap] is activated, the subsequent snapshot number will be used to store the data. Any data previously stored there will be overwritten. Therefore, one should start a completely new bank of snapshots when working on a project.
	Store +1	If activated, upon pressing [Store] the settings of the complete ensemble will not be saved to the current snapshot as displayed by [Snap] but to the next one.
Level	Master	Sets the master volume.
	Velocity	Slaves the [Master] control to the velocity of incoming MIDI notes used to recall snapshots.

Section	Controls	Function
	Single Outs	Switches on or off the sound units' routing to different outputs. When off, all sound units are mixed to one stereo signal; when on, there are four stereo outputs to which the six sound units can be individually routed (see [Output]).
	External	Switches on or off the triggering of the sound units by external MIDI notes. When off, the sound units are only triggered by the internal sequencer. (See also [Ext. Learn].)

3.2 Aerobic



4.4 Aerobic's user interface

3.2.1 Introduction

Aerobic is a step sequencer that controls a virtual analogue drum synthesizer. The instrument produces tight, innovative sounds far beyond the range of traditional drum computers. This, combined with the sequencer's capacities and the mixer's flexible routing options, makes Aerobic a versatile beat production environment that can be used in live performances.

The drum synthesizer contains six similar, independent units (selectable by the tabs at the top of the panel). Each unit combines an oscillator and a noise section into one signal that can be equalized before it is sent to the master mixer. The sequencer (in the middle of the panel) contains two tracks for each sound unit, selectable via the same tabs as the units themselves. The filled rectangles within the sequencer's display represent the unit's trigger signals and their velocity; the unfilled rectangles form a modulation track whose signal can be used to change nearly every parameter of the sound engine and mixer over time: Use the [Modulation] switch in the unit's master section to select the destination of the modulation. The master mixer provides classical mixing parameters for each unit (solo/mute, pan, and, of course, level), along with controllers to adjust the complete ensemble's reaction on MIDI messages. Each unit can be triggered by a selectable MIDI note; on a more complex level, note messages can recall complete ensemble snapshots.

3.2.2 Sound Engine

The drum synthesizer is built by two sound generators, an equalizer, and a master section that also controls modulation routing. While the oscillator part (on the left side) is based on sine waveforms with frequency modulation capacities, the noise part (on the right side) contains a white noise generator with a multi-mode filter. The mixed signal is sent through an EQ and (within the master section) a final saturator unit before it is passed on to the mixer.

Section	Control	Function
Oscillation	Envelope	Selects the operation mode of the envelope shaping the unit's amplitude. [Lin] activates a standard AD envelope whose transition times are controlled by the [Attack] and [Decay] knobs. When in [Roll] mode, this envelope is re-triggered fast until the next beat; the [Attack] knob in this case also controls the re-triggering frequency. [Roll+Lin] adds both signals of the modes described above. [Noise Env] uses the envelope of the Noise section (see below).
	Attack	Sets the time that passes until the amplitude envelope reaches its peak. In [Roll] mode (see [Envelope]) the knob also controls the rate at which the envelope is re-triggered.
	Decay	Sets the time that passes after the amplitude envelope has reached its peak before it decays to silence.
	Oscillator	Selects the operation mode of the oscillator. While [Sin] represents a standard sine wave, [Sin2] activates a squared sine wave with a different frequency spectrum. Similarly, [FM2] selects the squared signal of [FM] which is generated by a sine oscillator modulating the frequency of another one. (This frequency modulation does not interfere with the modulation controlled by [F-Mod], [F] and [Fmod].) [Phase] uses the output of a phase oscillator.
	F-Mod	Selects the source signal used to modulate the main oscillator's frequency. While [Osc Env] and [Noise Env] select the respective amplitude envelopes, the [Sine], [Tri] and [Random] entries use independent oscillators whose frequency can be adjusted with [Rate].
	F	Sets the base frequency of the main oscillator.
	FMod	Sets the amount of frequency modulation applied to the main frequency by the selected source signal.
	Rate	Sets the frequency of the independent oscillator modulating the main oscillator's frequency.
Mix	Mix	Sets the ratio of the oscillator section's output and the noise section's sound in the signal that is passed on to the equalizer.
Noise	Envelope	Similar to [Oscillator][Envelope], applied to the noise generator's filter.
	Attack	Similar to [Oscillator][Attack], applied to the noise generator section.
	Decay	Similar to [Oscillator][Decay], applied to the noise generator section.

Section	Control	Function
	Noise	Selects the operation mode of the noise section. [White] uses unfiltered noise, [White Mod] modulates the noise generator's algorithm by the noise section's envelope signal.
	Filter	Selects the type of 2-pole filter applied to the noise. Highpass, bandpass, and lowpass filters are available, providing 24 dB damping per octave.
	Freq	Sets the center frequency of the filter.
	Peak	Sets the amount of modulation applied to the filter's center frequency by the envelope.
	Res	Sets the amount of filter resonance.
EQ	Hz	Sets the frequency of the equalizer.
	dB	Sets the amount of volume boost (or cut) applied to the adjusted frequency.
Master	Modulation	Selects the target of the sequencer's modulation track. The modulation shows no effect until the [Track] button is pushed.
	Track	Activates the modulation of the target selected by the sequencer's modulations track.
	Amp	Sets the amplitude of the signal before it is routed to the final shaper unit (see [Shape]).
	Shape	Selects the operation mode of the shaper unit. [Polysat], [Sinesat], and [Hypersat] saturate the signal with tube-like effects; the effect increases the more the signal is amplified before (see [Amp]). [Clean] doesn't perform any compression; [Amp] simply controls the amount of amplification before the signal is routed to the master mixer.

3.2.3 Sequencer

The sequencer provides two tracks for each of the six drum synthesizer units: a gate pattern and a modulation track. The gate pattern determines the trigger signals and their velocity. The modulation track signal can be routed to any parameter of the sound engine (see [Sound Engine][Master][Modulation]). A roll mode bar provides three different roll modes for fast re-triggering of a drum sound.

Control	Function
Tempo	Selects the tempo of the track: Each step of the sequence can be interpreted as sixteenth note, etc. Thus, the sequencer is always synchronized to the master MIDI clock; use the host sequencer or REAKTOR's internal MIDI clock to start the sequencer. (See also [Global Tempo].)
Global Tempo	Sets the [Tempo] value of all six tracks.
Swing	Sets the amount of swing, i.e. the amount by which every second step of the sequence is delayed to shuffle the strict MIDI rhythm.
Roll Factors	Sets the number of times the trigger signal is repeated if the [Roll Mode] is set to the respective colors.
Init All	Deletes all sequence patterns and modulation tracks and sets [Swing] to its default values.
Track	Selects the track that can be edited within the [Edit Display]
Step Count	Displays the number of the current step (1 to 16). If the gate is off, the number is dark; if the gate is on, the number is light. This can be helpful when editing the modulation track.
Edit Display	Displays the trigger pattern (filled rectangles) as well as the modulation track (unfilled rectangles), depending on the [Track] setting. Clicking within the display allows the patterns to be edited. High values of the trigger pattern represent high velocity; in the modulation track they cause the modulated knob to turn to the right, and low values turn the knob to the left.
Roll Mode	Selects how often the trigger signal is sent. Normally, it is only sent once per beat; by clicking with the mouse one can step through three differently colored modes where the trigger signal is sent more often (see [Roll Factors]).
Loop	Controls the length and position of the played sequence: Only those steps within the rectangle are used. Drag the ends of the rectangle to adjust the loop's start and end points. A second, smaller bar represents the current read out's position.

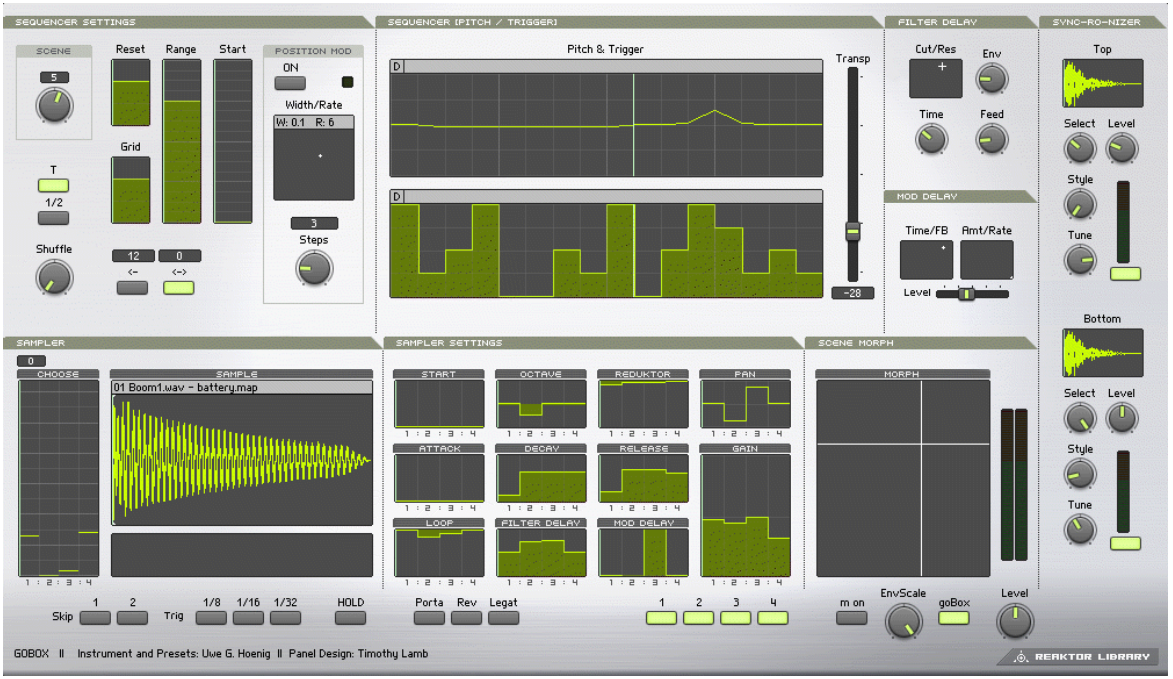
3.2.4 Master/Mixer

This section has two functions. First, it mixes the six drum synthesizers down to a single signal – or to four signals if [Single Outs] is activated. Second, it controls the snapshots of the complete ensemble as the sound engine and the sequencer are slaved to this part of the instrument. An advanced recall system allows for fast changes of sound/pattern settings via a single MIDI note, making the complex drums computer-controllable from a keyboard in live stage use.

Section	Controls	Function
Mixer	Level	Sets the volume of the sound unit.
	Solo	Switches the sound unit to solo playing, i.e. mutes all other units.
	Mute	Mutes the sound unit.
	Pan	Positions the sound unit's mono signal within the stereo field.
	Wide	Enhances the spatial appearance of the sound unit.
	Ext. Learn	Activates the learn feature. When pressed, the next MIDI note will be assigned to this track and can be used as external trigger signal, in addition to the internal gate signals of the sequencer. (See also [External].)
	Output	Selects to which of the four stereo outputs the sound unit is routed. This shows no effect until [Single Outs] is activated.
Snapshot	Power	Switches the snapshot handling on or off.
	Key	Turns on or off snapshot recall by external MIDI note messages. See [Root Note] and [Root Snap] for details.
	Quantize	Turns quantization of external MIDI notes on or off. When on, incoming MIDI messages will be synchronized to a pattern that is selected by [Quantization Select].
	Quantization Select	Selects the quantization pattern to which external MIDI messages can be synchronized.
	Snapshot	Recalls a snapshot of the master mixer. Since all other components are slaved to this one, storing or recalling a snapshot here affects all other instruments, i.e. the sound units and sequencer.
	Root Snap	Sets the snapshot number recalled when the MIDI note adjusted by [Root Note] is received; [Snap Via Key] has to be activated. The note above [Root Note] recalls the snapshot that follows on the [Root Snap] etc.
	Root Snap Learn	The first snapshot recalled after pressing this button will be used as new [Root Snap].
	Root Note	Sets the external MIDI note that, if [Snap Via Key] is on, recalls the snapshot adjusted by [Root Snap].
	Root Note Learn	The first MIDI note received after pressing this button will be used as new [Root Note].

Section	Controls	Function
	Store	Stores the current settings of the complete ensemble to the current snapshot number (see [Snap]). If [Store To Next Snap] is activated, the subsequent snapshot number will be used to store the data. Any data previously stored there will be overwritten. Therefore, one should start a completely new bank of snapshots when working on a project.
	Store +1	If activated, upon pressing [Store] the settings of the complete ensemble will not be saved to the current snapshot as displayed by [Snap] but to the next one.
Level	Master	Sets the master volume.
	Velocity	Slaves the [Master] control to the velocity of incoming MIDI notes used to recall snapshots.
	Single Outs	Switches on or off the sound units' routing to different outputs. When off, all sound units are mixed to one stereo signal; when on, there are four stereo outputs to which the six sound units can be individually routed (see [Output]).
	External	Switches on or off the triggering of the sound units by external MIDI notes. When off, the sound units are only triggered by the internal sequencer. (See also [Ext. Learn].)

3.3 GoBox



1.13 GoBox user interface

3.3.1 Introduction

GoBox is a monophonic sampler specially designed for live use. It features some hands controls for altering patterns, sounds, and modulations during a performance. A series of event tables form an easy to use interface that make it easy to see what's going on. Apart from sample modulation capabilities, there are a tempo-synced Filter Delay, a Mod Delay, and the Sync-ronizer module, which can play patterns of short samples metronomically.

3.3.2 Quick Start

Start the system clock and check out some of the presets. As the sequence plays, try adjusting some of the sample modulation parameters, like Attack, Loop, or Octave. Try out the tap 'n' drag slider located under the waveform display to manually modulate a sample as it plays.

3.3.3 Structure and Signal Flow

GoBox is a sequenced sampler of an unusual sort. Instead of playing loops it triggers one short sample at a time, monophonically. This can lead to bizarre junky/funky beats, uncertain patterings, and minimalist dance floor workouts. Two event tables sequence the pitch and trigger state of each of four sampler modules.

The Start point, Octave, and Pan position can be set using event tables for each of the samples. Start determines where in the sample the loop will start, while Octave tunes each sample in octave steps. Pan lets you position the sample in the stereo field.

The samples have their amplitudes contoured by four ADR envelopes. The tables for these are located in the second row in the sample modulation area. Overall volume for each of the four sample channels can be set using the Gain control. The Reduktor performs sample rate-and bit depth-reduction for each sample. Use this to add some dirt to your samples or to reduce them to staticy noise. All of the envelopes can be scaled, that is, generally shortened or lengthened in time, with the Scale button below the Morph XY pad. Use the Scale control to tighten up beats or to make beats seem louder by increasing the overall decay and release times.

The Loop table controls how much of the sample loops as it is triggered beginning at the point set by the Start control. Odd loop lengths can create stuttering and staccato rhythm effects.

After being contoured, the samples can be fed into the effects modules in differing amounts with the Filter Delay and the Mod Delay tables. The Filter Delay's delay time is automatically synchronized to the system clock in 16th notes. Filter frequency and resonance can be adjusted with the XY pad. Use the Envelope knob to modulate filter cutoff with the sample envelope.

A Mod Delay for chorus and reverb-like effects accompanies the Filter Delay. Two xy pads let you adjust the Time/Feedback balance and the Amount/Rate of delay time modulation.

The Sync-ro-nizer module sits on the end of the signal chain after the effects. It offers two synced sample players for generating simple, metronomic beats to help keep everything in line and lock down your rhythms. Select a sample using the Select knob. The sample pitch can be

adjusted using the Tune control. The Style knob selects a timing value on a preset sample trigger grid (i.e. 1/8th notes, 1/16th notes, 1/32nd notes, etc). You can turn each sample on and off using the mute buttons below the level meters.

3.3.4 The Sequencer

Event tables are used to sequence the pitch and trigger values for the sample loop module. The Pitch table sets the pitch for each step of a 16th note grid. The Trigger table selects which of the four possible samples plays on each step of the grid. You can place a rest into a sequence by selecting no sample for that point on the grid. As the clock runs, the sample selections play on the steps they are set for. A Transpose fader to the right of the Pitch and Trigger tables shifts the pitch sequence up or down.

You can select a sample for each of the four samplers using the Choose table. Load new sample maps into the sampler module by double clicking on the waveform display. You can label your sample categories by double clicking on the text field to the left of the Choose table and typing in the corresponding areas.

The clock controls for running and modulating the sequence all appear in the upper left hand corner. You can use the Scene knob to store sequencer and sample settings independently of the snapshot menu. You can store up to eight scenes. The Reset selector allows you to determine when the sequencer will reset its start point to the step chosen with the Start selector. From top to bottom, the sequence can be set to: reset every bar, reset every 2nd bar, reset every fourth bar, or no reset at all. The Grid selector, below Reset, is used to set the beat resolution of the sequence grid, i.e. 1/32, 1/16, 1/8, 1/4.

Set the length of the sequencer pattern using the Range selector. A range of "0" will mute the sequencer. The T button will instantly switch the sequencer resolution to triplets of the chosen grid value, introducing interesting rhythmic twists. The 1/2 button will set the sequence playback to half time. Since this is a "nondestructive" event, not changing the timing, goBox will always stay on the beat. The Reverse switch, labeled by an arrow pointing to the left, has the effect of reversing the sequence playback. This is also a "nondestructive" effect, instantly switching on the beat back to forward play. The Bi-Directional sequence playback button, to the right of Reverse, causes the sequence to alternate forward and reverse playback. Finally, the Shuffle knob allows you to add varying degrees of swing to the sample triggering.

goBox can shift the sequencer patterns automatically to keep your beats fresh by using the Position Mod LFO. Position Mod is a low frequency square wave with adjustable width and rate that modulate the current sequencer step position. You can specify the range of modulation in sequencer steps by using the Steps control.

Along goBox's bottom edge are a variety of hands-on real-time sequence modulation controls. The Skip buttons causes the sequencer to skip 1, 2, or 3 steps - adding dramatic rhythmic variations. The Trig buttons cause the currently playing sample to retrigger, or roll, at 1/8, 1/16, or 1/ 32 notes. Try these buttons while the sequencer is running to drop rolls and trills into the sequence as fills or turn-arounds.

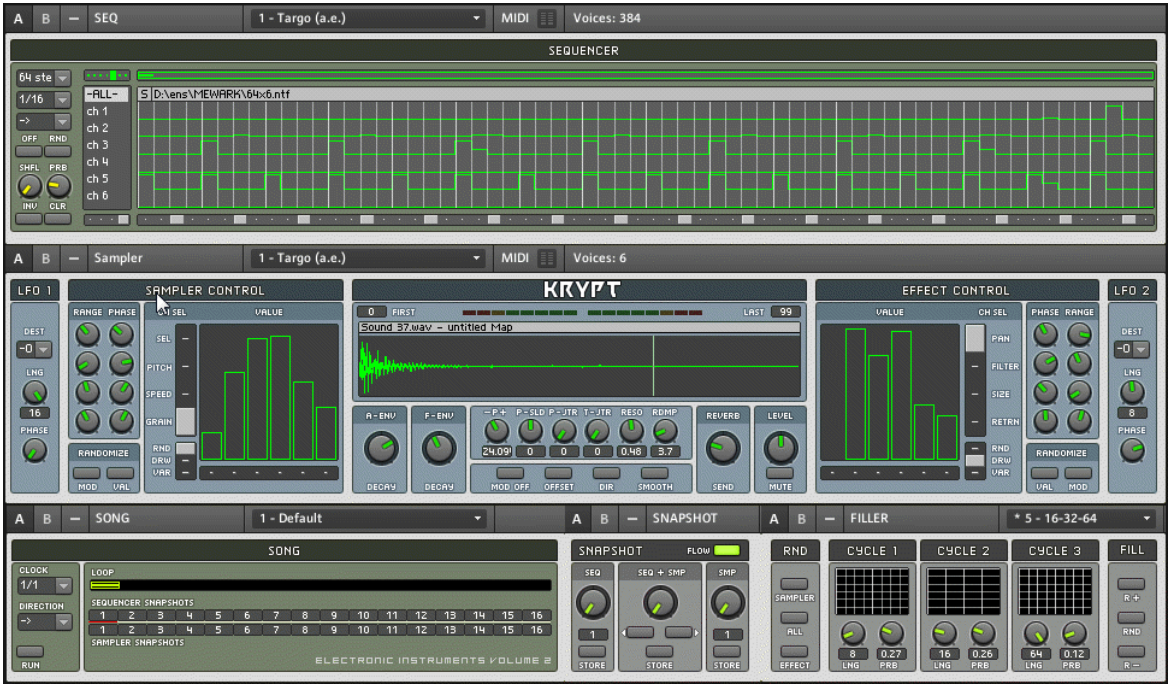
The Hold button repeats the current sequence step when depressed. This is perfect for creating breaks to heighten drama in a performance. Porta introduces a small amount of pitch glide to the samples. The Rev button creates a sort of reverse playback. The Legat button forces consecutive sequencer steps with the same gate value NOT to retrigger the envelope. This causes some samples not to sound on their sequencer steps, stripping back the beat.

3.3.5 Morph

The Morph pad allows you to make overall changes to goBox's settings with one easy to use XY control. It can be used to momentarily transform your sequence and/or sample settings, or to add small amounts of almost random change to your performance.

The Y-axis morphs the sequence parameters, while the X-axis handles the sampler parameters. Clicking on the pad introduces the Morph effect at that position for as long as the mouse button is held down. Settings return to their previous positions when the mouse button is disengaged. Clicking the M On button holds the Morph setting as long as the button is on.

3.4 Krypt



3.4 Krypt user interface

Introduction

Krypt is a drum machine quite unlike any other. In essence, it consists of a granular sampler and a six-channel sequencer featuring an intelligent pattern generation algorithm.

Quick start

Open the Krypt ensemble, and press play on the REAKTOR toolbar to start the clock. The sequencer will run and trigger the sampler unit. The sequencer features 6 channels, each of which corresponds to the one of the 6 voices of the sampler. While the controls located in the centre of the samplers' panel are global and are applied to all 6 voices, 8 additional parame-

ters can be edited individually for each of the 6 voices. These 8 parameters are visualised in the two value displays at the left and right of the central sampler unit. By adjusting the knobs labelled 'Mod', these 8 special parameters can be dynamically modulated by velocity and the two LFO's. It is these special, dynamic parameters that are key to Krypt's unique sound, and when combined with the powerful pattern generation capabilities on offer, the ensemble offers a completely unique automatic rhythm generation tool.

Overview

The ensemble consists of five instruments: **Sequencer**, a sequencer with 6 channels, which drives **Sampler**, the main sound engine based on a 6-voice granular sampler; **Snapshot**, which handles the sound sets; **Playlist**, used to automatically recall and sequence snapshots; and, finally, **Randomise** which generates random sequencer patterns and sampler configurations.

Section	Function
Sampler overview	Sampler is the sound engine of the Krypt ensemble, and in essence is a 6-voice granular sampler with a wide range of parameters. Certain parameters are static and global, in that they are applied to all 6 voices simultaneously, and cannot be modulated. The remaining parameters are dynamic – they are defined individually for each voice, and can be modulated by velocity and/or the LFO's.
Sampler static parameters	Although the Krypt ensemble is preloaded with a large sample map, you may wish to load your own samples. In this case, click on the main central window (which displays the currently played sample) to open the REAKTOR map editor. As with the default map, samples should be configured so that each MIDI key is assigned to a unique sample, with a transposition of 0. Once the map is configured, adjust the **First** and **Last** knobs so that they correspond to the highest and lowest MIDI note used in the map. This is essential for sample selection modulation to operate correctly. The sampler has two basic modes of operation, toggled by the **Offset** button. Enabling Offset permits the modulation of the sample playback position, which is particularly useful when using drum loop samples. In this mode, all samples are automatically stretched to tempo by slicing the loop into a number of sections, as defined by the **Quan** knob (Quantise). Note that pitch modulation is not available in offset mode. When Offset is disabled, the sampler works much like a 'standard' sampler, ideal for maps consisting of single drum hits. In this mode, **-P+** sets the pitch modulation range in semitones. (This will be discussed in more detail in the dynamic parameters section, as it relates to the dynamic pitch value.) P (pitch) defines the base pitch of the sampler thus transposes all samples. **P-jtr** (pitch jitter) defines the amount of random fluxuations in pitch in semitones, and **T-jtr** (Time jitter) sets the amount of random fluxuation in sample playback position. **A-decay** sets the decay time of the samplers' amplitude envelope, and the sampler output is then passed through a lowpass filter. The filter cutoff is determined by two factors: the base cutoff value (a dynamic parameter -see the next section) and a second decay envelope, defined by the **F-decay** control. **Reso** sets the resonance of a lowpass filter.

Section	Function
	The filter output is then routed through a reverb unit. The **Reverb** control mixes between dry (filter output) and wet (reverb output). **Rdamp** defines the extent of damping of high and low frequencies inside the reverb unit. Rev reverses all samples, and Smth enhances the smoothness of the grains.

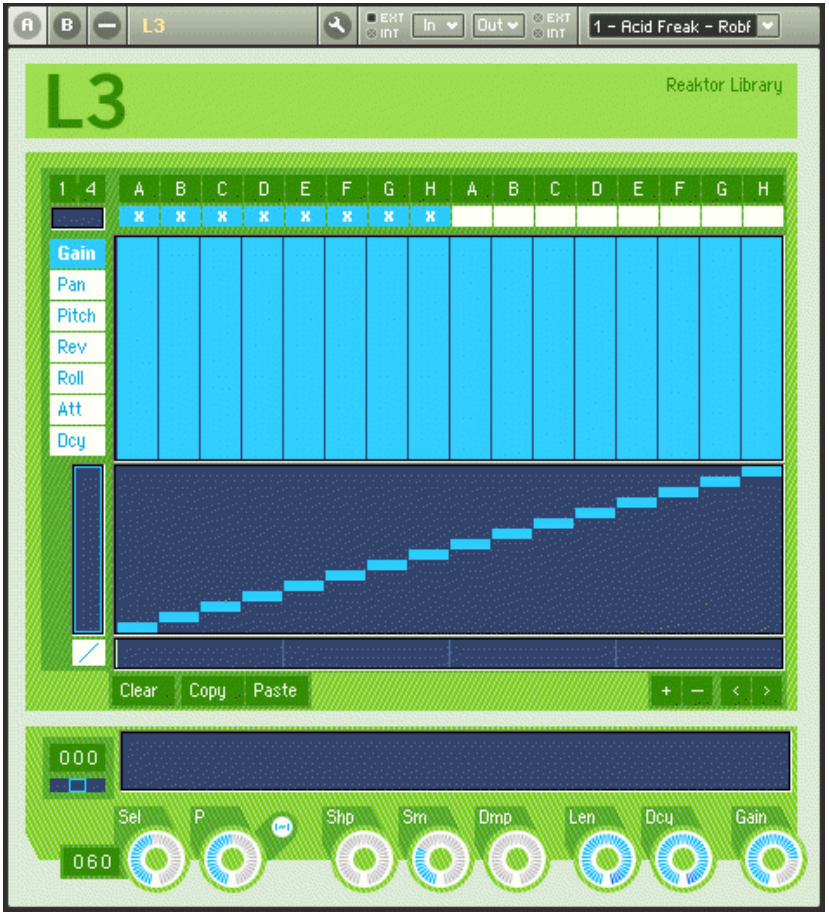
Section	Function
Sampler dynamic	In addition to the static and global parameters described parameters above, the sampler features an additional 8 dynamic parameters that can be defined individually for each of the 6 voices, and modulated by both velocity and the LFO's. The dynamic parameters are split into two groups – those concerning the sampler itself (sample select, pitch, speed and grain size) and those concerning the additional effects applied to the sampler's output (pan, filter cutoff, reverb size and reverb level). The base value of the dynamic parameters are edited using the two 6-voice **Value** displays. Click on the block at the side of the display to select which parameter to edit. Then click and drag on the value display itself to edit parameter for the 6-voices. The effect of mouse movements on the parameter values depends on the drawing mode. **Rnd** results in randomly generated values, **Drw** allows each voice to be set individually and **Var** shifts the values of all 6-voices while preserving the relative offset of each voice. The 8 **Rnge** knobs define the modulation range for the 8 dynamic parameters. So the actual value for a given parameter is determined by the base value, and depending on the modulation range, also the trigger velocity (see the sequencer section) and the LFO output. The Phase (phase) knobs determines the relationship between velocity and the base value, effectively rotating the effect of high and low velocities when the knobs are set to hard right. Clicking on **Val** instantly randomises all 6 voices for all 4 parameters. **Mod** randomises the range and phase controls. To enable either LFO, select a parameter to modulate from the **Dest** menu. **Len** sets the length of 1 LFO cycle in bars, and **Phs** sets the phase offset of the LFO cycle. Mute and Solo are available for all 6-voices. To mute a voice, click on the bar under the left value display. To solo a voice, click on the bar under the right value display.

Section	Function
Sequencer	Press play on the REAKTOR toolbar to run the sequencer, it will now drive the sampler. Channels 1 to 6 of the sampler correspond to voices 1 to 6 of the sampler. Different velocity values affect the sampler instrument as described in the above section. The maximum loop length of the sequence is determined by the menu in the top left corner (16, 32 or 64 steps), and the loop range can be restricted further by clicking and dragging on the loop bar above the main sequencer window. The second menu in the top left sets the speed of the sequencer (in musical time), and the third menu determines the playback direction and mode (-> forwards; <- backwards, <-> both directions; <<->> both directions with edge steps twice; <?> random direction; <??> random previous or next step; ???? random step). **Rnd** generates random values in the main sequencer window, each step being subject to a randomise algorithm with a probability defined by the **Prb** knob. Only steps on the selected channel, with edit mask enabled (the bar beneath the main window) are affected. Four edit masks are available (select on the bar next to let of the edit mask), and in addition to drawing the edit mask directly with the mouse, the overall edit mask can be inverted (**Inv**) or reset (**Clr**). **Shfl** determines the extent to which a humanizing, shuffling algorithm is imposed onto the beat. The algorithm effectively swings time forwards and back again on alternating steps. **Off** disables the sequencer.

Section	Function
Randomise	While some randomisation controls are available within the Sequencer and Sampler instruments themselves, the main randomisation controls, allowing simultaneous randomisation of all parameters, and sophisticated sequence generation, are found in the Randomise instrument. Random, yet rhythmic, patterns can be generated with the Filler by virtue of three random 'cycles'. While **Lng** adjusts the length of each cycle in steps, the **Prb** controls the probability of steps being generated within the cycle length. The cycles are combined and repeatedly across the entire length of the sequencer when the **Rnd** button is depressed. By pressing **R+**, additional events will be added to the current sequence, whereas, **R-** subtracts some events. Pressing **All** (or shortcut key 'R') also results in writing the random sequence to the sequencer, but also simultaneously triggers both sets of Mod and Val buttons in the Sampler instrument (as described earlier). **Smp** (or shortcut key 'S') triggers the sample (left-side) Mod and Val buttons only, whereas **Eff** (or shortcut key 'E') triggers the effect (right-side) Mod and Val buttons.
Snapshot and Playlist	While individual snapshots of the sampler and sequencer can be stored and recalled via REAKTOR's internal snapshot handling, they can additionally be managed with the Snapshot instrument. Snapshots can be stored by pressing the **Store** buttons on the left (for the sequencer), on the right (for the sampler) or in the middle (for both simultaneously). The knobs labelled **Seq** and **Smp** select the destination snapshot number for the sequencer and sampler respectively; **Seq+Smp** sets both to the same value simultaneously. In addition to using the knobs, the **Prev** (or shortcut key 'L') and **Next** (or shortcut key ';') buttons can also be used to move between snaps. When **Flow** is enabled, changing the value of those knobs also results in recalling the respective snapshot.

Section	Function
	These features allow copy & paste of complete ensemble settings. To do this, enable Flow and select the preset number you wish to copy with the central knob. Then, disable Flow and the select a destination snapshot to paste to. Finally, press the central Store button to write to the slot. The Playlist instrument can be used to program a sequence of particular snapshots for both the sequencer and the sampler instruments. The loop bar, speed and direction controls are similar to those found in the sequencer. The sequencer snapshot sequence is programmed in the upper row, and the sampler sequence in the lower row. Press Run to enable the module.

3.5 L3



4.16 The L3 user interface

3.5.1 Introduction

L3 is a sequenced drum loop recycler: Load a loop, click some stuff, mangle the loop.

The instrument's panel can be broken down into three sections. The top section contains a pattern sequencer and global controls for pattern length and tempo swing. The middle section has the main step sequencer where patterns can be edited (up to eight patterns can be programmed and arranged per snapshot). The lower section contains the sample playback engine controls (including the sampler window where loops are loaded).

L3 is driven by the MIDI clock. This means that when used in the stand-alone version of REAKTOR, the play button on the REAKTOR toolbar must be pressed. When used as a plug-in, L3 will only run when the host sequencer song is playing.

3.5.2 Pattern Sequencer

Each L3 snapshot consists of up to eight individual patterns labeled A through H. As you will see on the panel, there are 16 [Pattern Selector Boxes] along the top each displaying a letter (A to H). The pattern playback order depends on the arrangement of the letters displayed in these 16 boxes. The leftmost box selects the pattern to be played for the first bar, the second box selects the pattern for the second bar, and so on.

Control	Function
Pattern Selector Boxes	Controls the sequence of the patterns to be played one after another. Click and drag vertically on any of the 16 boxes to select a pattern. Use the [Loop Area Bar] to determine the length and position of the loop.
Loop Area Bar	Defines the area of the [Pattern Selector Boxes] played in a loop. If only one box is selected, only this pattern will be played; this can be useful to edit and audition a pattern.
Bars per Pattern	Adjusts the number of bars in each pattern.
Beats per Bar	Controls the number of beats in each bar. A beat is interpreted as a quarter note; each step represents a sixteenth note.
Swing	Sets the amount of shuffle, i.e. the amount of slight delay on off-beats.

3.5.3 Step Sequencer

Things get more interesting here. L3 features eight parameters which can be sequenced by programming their value at each 16th. This is what turns old loops into new loops...

The most obvious of these eight parameters is slice order. This is displayed in the lower of the two large windows (the [Slice Position Sequencer]). By clicking the mouse here you can rearrange slices of the original loop. The window is 16 steps high which means you can select the first 16 slices of your loop. If the selected loop has more than 16 steps, use the [Scroll Bar] at the left to see more slices. The right-mouse button has a function here too: it restores any step to its default value (i.e. to the original slice order).

The upper window (the [Parameter Sequencer]) is for editing the remaining seven parameters: gain, pan, pitch, reverse, roll, attack, and decay. Clicking the right-mouse button resets steps to their default value.

Control	Function
Parameter Sequencer	Controls the values of the various parameters for each sequencer step. Use the right mouse button to reset a parameter to its default value.
Gain	Adjusts the gain of each slice.
Pan	Adjusts the position of each slice within the stereo field.
Pitch	Adjusts the pitch of each slice, i.e. its transposition in respect to the original pitch of the sample file.
Reverse	Determines the playback direction. At minimum (the default setting), slices will play forward as normal. At any other value they will play in reverse. At lower values, playback will start from near the end of the slice, whereas with higher values, playback will start from nearer the beginning of the slice.
Roll	Causes the slice to retrigger repeatedly within each step. With higher values the slice will retrigger more quickly.
Attack	Causes the loop volume to suddenly cut out and then fade back in. At maximum, the fade in time is exactly 1 beat (i.e. four steps).
Decay	Modulates the envelope decay time. At center (default) decay time is unaffected. With higher values the decay time is extended, and with lower values the decay time is reduced. (Therefore the effect depends on the envelope decay time setting controlled by the [Decay] control of the [Sampler] section.)
Slice Position Sequencer	Controls the order of the slices. Low values represent slices at the beginning of the sample file, high values slices at its end. Thus, a line from the bottom-left to the top-right results in normal playback order as defined by the sample file without any rearrangement.

Control	Function
Scroll Bar	Scrolls the [Slice Position Sequencer] vertically. This can be useful if a long loop with many slices is loaded: As the [Slice Position Sequencer] can only display sixteen vertical values, slices after the sixteenth cannot be controlled. Use this bar to scroll to those higher values.
Edit Range Bar	Controls the area of steps within the [Slice Position Sequencer] onto which the edit functions are applied. The edit functions are:
Reset Slices	Sets each step within the edit range to its default position.
Shift Up / Down	Shifts each step within the edit range up or down by one position.
Shift Left / Right	Rotates each step within the edit range to the left or right by one position.
Clear	Resets all steps within the edit range in both the [Slice Position Sequencer] and the [Parameter Sequencer].
Copy	Copies all steps within the edit range of both the [Slice Position Sequencer] and the [Parameter Sequencer] into an internal buffer.
Paste	Copies all steps from the internal buffer into the edit range of both the [Slice Position Sequencer] and the [Parameter Sequencer].

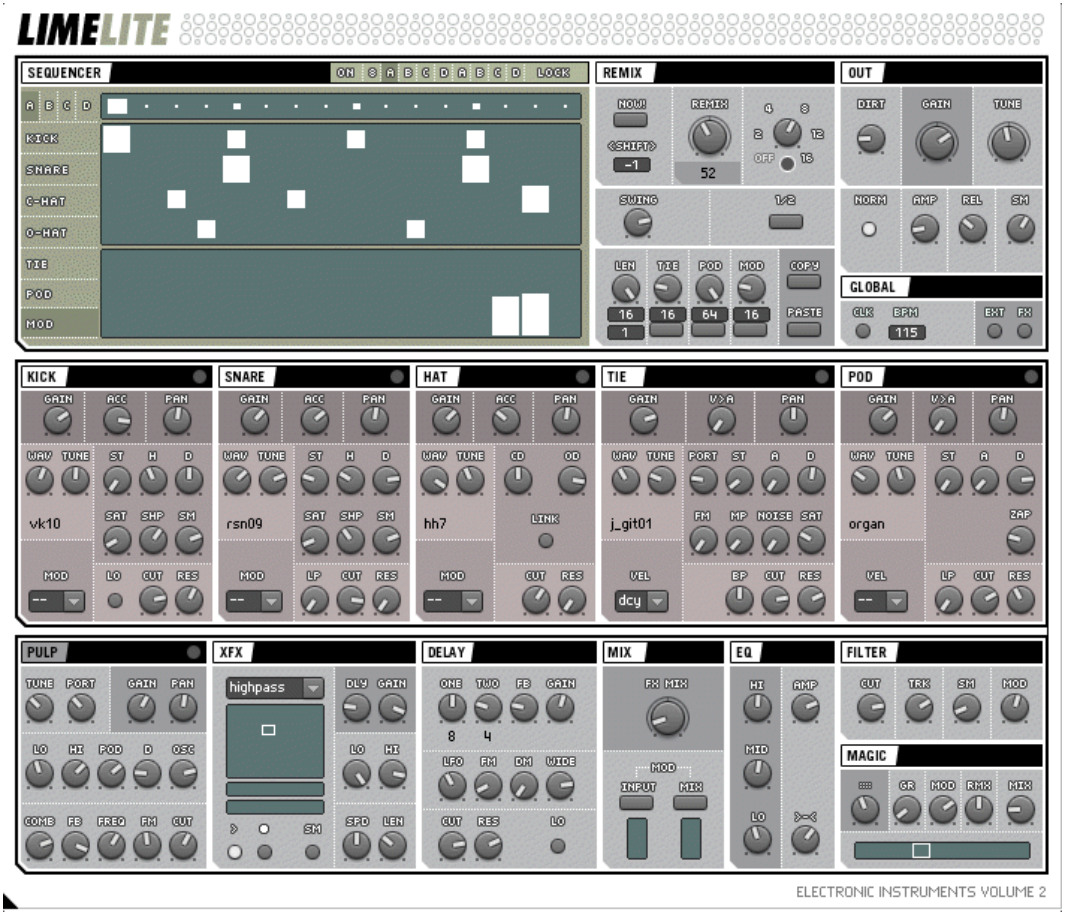
3.5.4 Sampler

The main window is for loading loops and displays the currently selected waveform. After loading a loop, make sure it is selected using the [Sample Select] knob, and then check that the detected tempo is correct (it's displayed in the box to the left of the sample window). If incorrect, the tempo can be adjusted using the slider bar beneath. If the correct tempo cannot be selected, then the loop is not an integer number of bars in length, in which case you cannot use it.

All of the sampler controls in this section are stored per-pattern. Clicking a knob with the left mouse-button writes to the current pattern only, whereas clicking with the right-button writes to all eight patterns simultaneously (A to H). Also, double-clicking on a knob resets it to its default position.

Control	Function
Sample Display	Displays the sample currently selected by [Sample Select]. Double-click to open REAKTOR's Sample Map Editor and to load a sample file.
Tempo Control	Displays the automatically extracted tempo of the sample loop in beats per minute. Use the slider to select a different value.
Sample Select	Selects a sample from the map within REAKTOR's Sample Map Editor of the [Sample Display].
Pitch	Transposes the overall pitch of the loop in semitones.
Stretch	Calculates the pitch at which one bar of the audio file will be the same length as one bar of the actual current song tempo, and then transposes the loop accordingly. In other words when 'stretched', there will be no gaps between slices (caused by the original loop tempo being slower than the current tempo), nor will slices be prematurely truncated (caused by the original loop tempo being faster than the current tempo). It is still possible to transpose the loop when the stretch button is active, but obviously the loop will no longer be perfectly stretched to tempo. In other words, to be correctly stretched the pitch knob must be set to zero.
Shape	Determines the compressor gain curve (see also [Smooth] and [Damp].)
Smooth	Reduces the amount of distortion by smoothing gain changes; it controls the attack and release of the compressor. (See also [Shape] and [Damp].)
Damp	Attenuates high frequencies, reducing 'grainy' sounding compression artifacts. (See also [Shape] and [Damp].)
Length	Sets the hold period (i.e. the length) of the envelope shaping each slice's amplitude.
Decay	Sets the decay time of the envelope shaping each slice's amplitude. This is the master control that can be varied for each step independently.
Gain	Sets the output level for the current pattern.

3.6 Limelite



3.5 Limelite user interface

Introduction

Combining powerful sequencing options, a huge bank of samples, and an awesome array of effect processors, Limelite takes the beat-box concept to a whole new level.

Quick start

Limelite's sound shaping possibilities are vast and complex, as reflected by the number of controls on the panel. However, those wishing to start programming beats straight away will need to know a few essentials:

- All samplers and FX units can be quickly enabled or bypassed by clicking on the panel label at the top of each section. When bypassed, the label will appear faded. Click again to enable the section.
- The sequencer section contains 3 windows, the uppermost of which simply shows the current beat position. The middle window is for programming the main rhythm, (composed of a kick, snare and hi-hat) -simply click the mouse in this window to toggle hits on or off, and to toggle hit accent, click and hold the mouse button until the hit changes size.

The lower window is for editing 3 additional sequences - the Pod and Tie samplers, and a multi-purpose, freely-routable modulation sequence (Mod). To select which of the 3 sequences to edit, click the mouse on the appropriate label next to the window.

Overview

Limelight's panel can essentially be separated into 4 areas:

1. The Sequencer and Remix sections are where the basic drum pattern is programmed.
2. The Out section in the upper right, which allows final adjustment of the overall sound.
3. The middle row of the panel, which houses the 5 samplers. Kick, Snare and Hat carry the basic groove, which is then complimented by the two additional samplers – the Tie and Pod.
4. The bottom row of the panel, which contains a variety of FX processors.

Section/Control	Function
Sequencer section - main window	The main window in the sequencer section is where the basic beat main window is programmed (kick, snare and hat). Click the mouse inside this area to toggle hits on or off. To toggle between normal and accented (i.e. louder) hits, click and hold the mouse button.
Pattern sequencer	The main sequencer window has 4 patterns – A, B, C and D. When **Track** is enabled, the main window will always follow and display the current pattern (A, B, C or D) being played. To select which to edit, click on the desired pattern in the top-left of the sequencer area. Patterns A, B, C and D can be arranged in any order by using the pattern sequencer, (at the top-right of the sequencer section). Enable the pattern sequencer by clicking on the on/off button, and click and drag on the adjacent number to set the sequence length, (2 to 8 bars). Finally, click and drag on the 8 pattern letters to define the pattern arrangement. The **Copy** and **Paste** buttons allow any of the 4 patterns to be copied and pasted, either within or between snapshots. To clear the current pattern, click **Paste** while simultaneously holding a MIDI note.

Section/Control	Function
Tie, Pod and Mod	The Tie, Pod and Mod sequences can be edited in the lower window. To select which of the 3 sequences to edit, click on the label to the left of the window. To reset a sequence, click on the label while simultaneously pressing any MIDI note. Note that unlike the Tie and Mod sequences, the Pod sequence is bi-polar. Both negative and positive values can be entered, for +/- pitch transposition for example. When editing the pod sequence, double-clicking the mouse results in a centre (zero) value. This will trigger the sample but yield no velocity modulation. Note that the Tie, Pod and Mod sequences are independent of the main sequence. While the main sequence has 4 patterns, which can be freely arranged, the Tie/Pod/Mod sequences loop continuously and independently of the A/B/C/D pattern sequencer.
Remix section - length controls	This remix section contains a variety of patterns for defining and rearranging the main and Tie/Pod/Mod sequencers. The length of the main pattern, and the Tie/Pod/Mod patterns can be set using the controls at the bottom of the section (labelled **Len**, **Tie**, **Pod** and **Mod**). The start position can be edited with the value beneath the Len knob. The 3 buttons beneath the Tie, Pod and Mod length knobs stretch the Tie/Pod/Mod sequences to fit the length of the main sequence. So for example, if the main sequence is 16 steps long, and the Pod sequence is 32 steps with stretch is enabled, the Pod sequence will run twice as fast as the main sequence. Similarly, if Pod length is 8, it would run at half the speed of the main sequence.

Section/Control	Function
Pattern remixing	Typically drummers repeat a basic sequence with a variation on the th or 8th bar, for example. The remix sections allow this style of play to be emulated without the need for manually programming the variation pattern. The **Remix** knob selects a pattern from the 127 pre-programmed variations. The rate of the variation pattern (i.e. every other bar, every 4th bar, every 8th bar etc) can then be set with the remix frequency knob (top-right). When the **Now!** button is enabled, the remix frequency knob is overridden, and the remixed pattern will play continuously. This is especially useful for when browsing through the 127 patterns with the Remix knob. **<Shift>** varies the remix pattern in one of two ways. Positive values shift the output position of the remix patterns, whereas negative values shift the start position of the remix patterns. Thus, bearing in mind that all remix patterns are 16 steps in length, consider the example of pattern one (1,2,3,4,5,6,7,8,9,10,1,2,3,4,5,6). A shift value of 1 will increment all steps (resulting in 2,3,4,5,6,7,8,9,10,11,2,3,4,5,6,7). A shift value of -1 will rotate all steps, so that step 16 is rotated back to the first step, (resulting in 6,1,2,3,4,5,6,7,8,9,10,1,2,3,4,5).
Remix - Swing, ½ tempo and accent controls	**Swing** determines the extent to which a humanizing, shuffling algorithm is imposed onto the beat. The algorithm effectively swings time forwards and back again on alternating steps. The **1 / 2** button halves the tempo (e.g. 120 becomes 60).

Section/Control	Function
Out	The controls in this section influence the overall master output of Limelite. **Gain** controls master volume, and Pitch defines global tuning (+/-12 semitones). **Dirt** controls a distortion algorithm, while **Norm** enables a volume maximising algorithm. For the volume maximiser, **Amp** defines the maximum amount of volume maximisation, **Rel** determines the release time of a peak detector and Sm determines the amount of slew (the speed) of shifts in gain. Loud, but harsh sounds will be obtained with high gain (full-right) and fast Rel and **Sm** settings (hard-left). Better results will usually be obtained with more modest settings.
Ext and FX buttons	**Ext** allows external input to be fed into the Pulp unit and FX processors. Note that external input is always routed to Pulp if Pulp input is set to Ext, regardless of the global Ext button. The **FX** switch disables the samplers, so that Limelite can be used solely as an FX unit, with lower CPU demand. Note that the FX switch overrides the Ext switch setting – in that external input will always be enabled.

The Samplers

Section	Function
Shared controls	Several controls are common to all samplers: **Gain** and **Pan** determine mix volume and panning respectively. **Acc** determines the range between lo and hi accented hits. Higher values result in greater differentiation. While the **Solo** buttons deactivates other samplers for solo auditioning, each sampler can also be muted individually by clicking on the section label. **Wav** selects the sample (each sampler has 128 samples) and Tune defines pitch transposition. **Cut** and **Res** define filter cutoff and resonance respectively. Select the filter type with the **LP** / **BP** / **HP** knobs (for low-pass, band-pass and high-pass respectively).
Kick and snare	The Kick and Snare share identical controls, with the exception of the filters. The kick has a lowpass filter and an optional low cut button to remove low frequencies. The Snare features a filter which can operate in either lowpass, bandpass or highpass mode (the knob next to cut selects which mode). **St** defines sample start position; **H** and **D** determine the hold and release time of the envelope respectively. **Sat** determines the amount of saturation distortion, and the **Shp** knob controls a sound shaper that can either 'thin' (left) or 'fatten' (right) samples. **Sm** controls the amount of smoothing (effectively the attack and release time of gain changes). High settings produce less undesirable artefacts at the expense of decreased sound shaping. **Mod** routes the mod sequence to one of several parameters in the section – **Wav** (wav selection), **P** (pitch), **D** (decay time), or **Cut** (filter cutoff). The Snare mod menu also includes an entry for **Flam** time.

Section	Function
Hat	The Hat unit contains 2 samplers, one for open and one for closed hat samples. Thus, there are 2 decay times -**CD** (closed decay) and **OD** (open decay). When **Link** is enabled, the 2 samplers operate monophonically, in that triggering one will terminate the other – thus emulating a real hi-hat on an acoustic drum kit. The Mod menu is the same as with Kick and Snare.
Tie	**Port** determines portamento speed, and **A** defines envelope attack time. **FM** sets frequency modulation depth, and **MP** defines the pitch of the FM modulator oscillator. The sample can be mixed with white noise using the **Noise** knob. The **Vel** (velocity) menu selects the modulation destination for velocity. Velocity can be routed to amplitude by adjusting the **V>A** knob (hard-left modulation, hard-right for full modulation).
Pod	**Zap** controls FM depth, but unlike the Tie, the FM source is not an oscillator but a decay envelope.

Pulp and FX

Section	Function
Pulp	Pulp is a signal generator that transforms the input signal (selectable by the central knob) into a percussive-type output. First the pitch and amplitude of the incoming signal are detected. The pitch range is clipped with the **Lo** and **Hi** knobs, transposed by **Tune** and slewed by **Port**. Amplitude is detected by a peak detector, controllable by the **D** (decay) knob. The detected pitch is then routed to a sine oscillator and the filter cutoff of a noise generator. The **Osc** knob fades between the sine oscillator (hard-left), the input signal (centre) and the noise generator (hard-right). Finally, the output is fed into a comb filter. **FB** determines the amount of feedback, **Freq** determines the delay time, **FM** the delay time modulation amount (signal amplitude being the modulation source), and **Cut** defines the lowpass cutoff of the comb filter. **Comb** cross-fades between oscillator output (hard-left) and the comb filter output (hard-right).
XFX overview	The XFX section contains a bank of 22 effects, and a recordable XY pad with which to automate the 2 parameters of each effect. Some of the 22 effects also have a non-automatable 3rd parameter, which can be set using the slider bar beneath the XY panel. Details of parameters for each effect can be found in Appendix 1. Press **Play** to enable automation playback, and while **Rec** is enabled, hold the mouse button and move the mouse to write to the automation sequence. The XY automation is stored in 2 x 64 data points, and **Sm** toggles between stepped or smoothed interpolation between each point. **Spd** determines automation speed and **Len** automation length. When **Spd** is centred and Len is set to hard-right, the automation length will last for exactly 4 times the kick/snare/hat sequence length. **Dly** defines the input signal into the XFX section. At hard-left, the input signal comes direct from the Samplers + Pulp unit. At hard-right, the input signal is the Delay unit output. When Dly is centred, the XFX input is a 50/50 mix of the two. Correspondingly, the Dly knob also determines the amount to which the delay section output is sent to the Mix (fx mixer) section. At hard-left, both xfx and delay output are sent to the fx

Section	Function
	mixer, but at hard-right, only the xfx output is sent to the mix. Thus, in other words, the Dly knob also cross-fades between xfx+delay parallel mixing and dly>xfx serial mixing. **Lo** and **hi** can be used to attenuate low and hi frequency elements of the xfx output. Set both to hard-right for no attenuation. **Gain** determines the overall output level.
Delay	The delay section is a stereo tempo-synced delay unit, with independent delay times for the left and right channels. **One** and **Two** set the time (in 16th ticks) for the two channels, **FB** determines the amount of feedback, and **Gain** the overall output level. LFO controls the speed of a beat-synced LFO, set to hard-left for a cycle time of 4 bars, and hard-right for a cycle time of a 16th tick. FM determines the amount of cutoff modulation via the LFO, whereas DM the amount of delay time modulation. **Width** controls the amount of stereo separation. **Cut** and **Res** define the cutoff frequency and resonance of a post-delay LP filter. **Lo** enables damping of pre-delay low frequencies.
Mix	**FX Mix** cross-fades between the dry signal (pre XFX and delay) and the wet signal (post XFX and delay). When the **Input** button is enabled, the mod sequence determines the input level to the XFX and delay. Clicking on the meter below the input button toggles between proportional or binary (i.e. on/off) modulation. When **Mix** is enabled, mod is routed to the fx mix cross-fade position (as a proportion of the current fx mix knob position, i.e. if mod = 1 then fx mix will equal the actual fx mix knob position). Clicking on the meter below the input button toggles between proportional or binary (i.e. on/off) modulation.
EQ	The EQ unit allows for boosting/attenuation of **Hi**, **Mid** and **Lo** frequencies. Set each knob to centre for no change. The **>-<** knob shifts the frequency of the 3 bands. **Amp** determines the amount of drive into a saturator distortion unit.

Section	Function
Filter	**Cut** defines the cutoff frequency of a 12db lowpass filter. **Trk** determines the amount to which an envelope follower modulates the cutoff frequency, and **Sm** determines the smoothness of the envelope follower. The mod sequence can be routed to cutoff via the **Mod** knob.
Magic	Magic features a notch filter and a granular-pitch shifter. ::: defines the notch filter cutoff, and the large slider at the bottom of the section controls the amount of pitch shifting (centre for no shifting). **Gr** sets the pitch-shifter grain size. **Mod** and **Rmx** define the amount to which the mod sequence and the pattern remix position modulate the pitch shifter respectively. **Mix** cross-fades between the dry and wet signals.

Appendix 1 – XFX Parameter Reference

	X	Y	Additional
Highpass	Cutoff	Resonance	-
Bandpass	Cutoff	Resonance	LFO depth
Lowpass	Cutoff	Resonance	-
Phaser	LFO speed	Centre	LFO depth
Vowel	Cutoff 1	Cutoff 2	-
Flanger	LFO speed	LFO depth	Stereo
FM	Frequency	Depth	-
Reson	Envelopd mod	Delay	Stereo
Feedback	Pitch	Envelope length	Envelope shape
Reflex	Cutoff	Resonance	Speed
Reverb	Size	Decay	Mod
UFO	?!	?!	?!
Rainman	?!	?!	?!
Distort	Cutoff	Drive	-
Ringmod	Ringmod Pitch	Pitch 2	Mix 1 / 2
Lofi	Hz	Bits	-
Pitchshift	Grain	Pitch	-

	X	Y	Additional
Gate	Decay	Threshold	Envelope shape
Chop	Decay	Envelope shape	-
Slice	Reverse rate	Pattern (1 to 16)	Pattern bank (A or B)
Freeze	Loop length	Pitch mod	Pitch mod depth / dir
Spin	Stereo	Depth	Length

Appendix 2 – MIDI / English QWERTY Keyboard Shortcuts

	MIDI note	Qwerty
Remix now!	61	2
Half tempo	56	H
Kick on / off	60	Q
Snare on / off	62	W
Hat on / off	64	E
Tie on / off	65	R
Pod on / off	67	T
Pulp on / off	69	Y
Kick solo	48	Z
Snare solo	50	X
Hat solo	52	C
Tie solo	53	V
Pod solo	55	B
Pulp solo	57	N
XFX on / off	63	3
Delay on / off	66	5
Mix on / off	68	6
EQ on / off	70	7

	MIDI note	Qwerty
Filter on / off	73	9
Magic on / off	75	0

3.7 Massive 1.1



4.5 Massive user interface

3.7.1 Introduction

This drum computer is “massive” in at least two ways. First, it contains a vast range of signal-shaping capacities: samples in the six drum tracks don’t determine the instrument’s sound (like in a standard drum machine), they only provide the material from which the beats can be sculpted. Envelopes, filters, and a potent grain re-synthesis algorithm mangle the fundamental sound until it is completely different, but still musical. Second, these versatile sound editing features are combined with an advanced step sequencer offering copy and paste functionality, three different roll types, a triplet mode, independent loop length for each of the six drum tracks, three modulation tracks whose signal can be routed to nearly every parameter of the sound engine – the list of features could be continued. Yet, those capacities are not hidden behind an endless array of knobs and faders that prevent productive working. The panel is optimized for usability and fast access to all controllers, making Massive to a powerful sound design workstation. At the same time – thanks to a complex and glitch-free snapshot recall system – Massive can be used in live performances, or slaved to a master song sequencer that changes the snapshots automatically (see [Snapper]). Three panel sets give access to Massive and two related utilities. The default view shows Massive’s main panel alone, ready to program a snapshot’s sound. The alternative view contains a small display of Massive, mainly focused on the list of already existing snapshots, Snapper (the song sequencer), and a soundcard utility to route the instrument’s signals to several output slots. A third view shows nothing: the instrument still works but does not consume CPU power to process the graphical user interface.

3.7.2 Control Section



4.6 The Control Section at the top of the instrument

On the left, an edit mode section defines how the various step sequencer displays react to mouse actions. The copy and paste controls are here, too. Next to it a snapshot management system allows for fast storage and recall, and also provides the link to an external song se-

quencer (see [Snapper]). This section is followed by four edit pages. These control the tracks' quantization and micro-timing, two effects, and an equalizer. Finally, there are three knobs to control the output levels for the dry signal and both effects.

The effects section contains a delay unit and a gater. The delay consists of a sequenced input level and a pre-delay high-pass filter, a post-delay low-pass filter with modulated cut-off frequency, and controls for feedback and pan. The gater is triggered each sixteenth or eight triplet note and remains open for an adjustable amount of time. This amount can be modulated by one of the sequencer tracks, and is useful in subtle sound sculpting, e.g. enhancing the sound's transients in a drum'n'bass context.

Section	Control	Function
Edit	Edit Mode	Selects the way the various step sequencer displays react to mouse actions. When [Draw] is selected, the mouse can set each step value (see also [Lock] and [Sequencer][Value Display]). When [Copy] is activated, an area of steps can be selected with the mouse that is automatically copied to the [Edit Buffer]. In [Paste] mode the buffer's data is copied back to any area selected with the mouse; if the paste area is longer than the buffer's content, the material to be pasted is looped. [Remote] enables the separate [Copy!] and [Paste!] buttons.
	Copy	If [Remote] is selected as [Edit Mode], pressing this button activates the same behavior of the step sequencer displays as the separate [Copy] mode of the [Edit Mode]. This button can easily be activated by pressing the {C} key on the computer keyboard (i.e. MIDI note 52). Thus, one can quickly edit the sequencers' data with one hand on the keyboard and one on the mouse. (See also [Paste!] and [Lock].)
	Paste	If [Remote] is selected as [Edit Mode], pressing this button activates the same behavior of the sequencer displays as the separate [Paste] mode of the [Edit Mode]. This button can easily be activated by pressing the {V} key on the computer keyboard (i.e. MIDI note 53). One can quickly edit the sequencers' data with one hand on the keyboard and the other on the mouse. (See also [Copy!] and [Lock].)
	Lock	Keeps the mouse locked on the selected sequencer step in [Draw] mode (see [Edit Mode]). This can also be activated by pressing the {Z} key on the computer keyboard (i.e. MIDI note 48).
	Edit Buffer	Displays the content of the buffer into which data is copied in [Copy] mode and that is used in [Paste] mode (see [Edit Mode]).

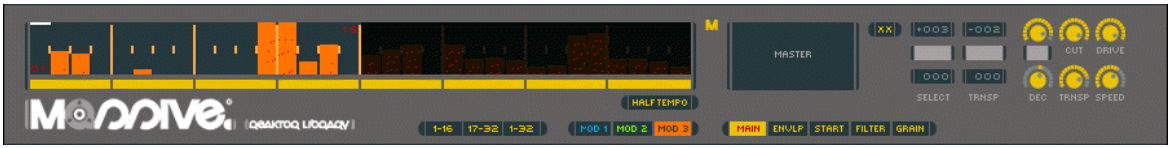
Section	Control	Function
Snapshot	Snapshot Mode	Selects whether the snapshots are only recalled via internal signals or if external control signals received at the instrument's [Snap] port are recognized, too. (See also [Snapper].)
	Snapshot Number	Adjusts the snapshot slot whose data is loaded by pressing the [Snapshot Load] button and to which the data is saved upon pressing the [Snapshot Save] button.
	Snapshot Bank	Selects the bank of snapshots that is used by the complete snapshot management system.
	Snapshot Save	Stores the current controllers' settings to the slot specified by [Snapshot Number] and [Snapshot Bank]. By the smaller buttons (labeled – and +) at the left and right of the button the current instrument's data can be saved to the previous or next snapshot. There is no undo for this function.
	Snapshot Load	Loads the preset selected by [Snapshot Number] and [Snapshot Bank]. The previous or next snapshot can be loaded using the smaller buttons (labeled – and +) at the left and right of the load button. This is particularly useful in live situations.
	Snapshot List	Provides a list of all adjustable snapshots of the currently selected snapshot bank. Clicking on an entry loads the entry's data.
	Randomization	Sets all parameters of the sequencer tracks and effects to random values. The sequencers' data and the tracks' level controls are not randomized.
Quantization and Timing	Quantization Select	Selects one of twelve quantization presets. Horizontally, each preset ranges over sixteen steps; the higher the displayed vertical value, the more delay is applied to this step. The first preset, for example, alternates between low and high values, so each second step will be delayed, resulting in a standard off-beat shuffle. The presets only define relative times; the effective delay time at maximum values is set by the [Master Shuffle] and [Track Shuffle] controls.
	Master Shuffle	Adjusts the maximum delay time. This delay time is scaled for each track individually by the [Track Shuffle] control.
	Grid	Controls the grid of the step sequencer displays. It has no effect on the instrument's sound.
	Track Shuffle	Scales the delay time of the [Master Shuffle] control for each track individually. At maximum value the respective track uses the main delay time; at the minimum value there is no delay.

Section	Control	Function
	Track Timing	Adjusts the static micro-delay (of up to 35 milliseconds) for each track.
Delay	Input Modulation	Selects the modulation track that controls the level of the effect's input. High steps in the modulation track represent full amplitude, at minimum step values the input is muted.
	Highpass Frequency	Adjusts the cut-off frequency of the high-pass filter that is applied to the input signal before it is sent to the delay unit itself.
	Delay Time	Sets the delay time. There are three vertical control boxes: the topmost adjusts the delay time for the left channel and the middle one adjusts the right channel. The bottom control switches between sixteenth notes and eighth triplets as fundamental unit of the delay times.
	Feedback Amount	Controls the amount of feedback.
	Lowpass Frequency	Sets the frequency above which the signal will be dampened.
	Modulation Amount	Sets the amount of modulation applied to the [Lowpass Frequency] (see also [Modulation Rate]).
	Modulation Rate	Controls the speed of modulation applied to the [Lowpass Frequency]. The modulations source is a triangular LFO.
	Lowpass Resonance	Adjusts the resonance of the lowpass filter.
	Pan	Places the delay's signal within the stereo field.
Gater	Gate Display	Shows the current values of the gate length (horizontal axis) and the output level (vertical axis).
	Quantization	Selects the trigger event quantization, switching between sixteenth notes and eighth triplets.
	Length	Sets the gate length; the higher this value, the longer the gate remains open after a triggering event (see also [Quantization]).
	Length Modulation	Selects the modulation track that modulates the value adjusted by [Length].
	Release	Sets the release time of the gate.

Section	Control	Function
	Output Modulation	Selects the modulation track that controls the level of the effect's output. High steps in the modulation track represent full amplitude, at minimum step values the output it muted.
	Shuffle	Scales the amount of shuffle applied to the effect unit. This is independent of the main quantization pattern and is bound to a standard off-beat shuffle.
EQ	Low Frequency	Sets the frequency below which all frequencies will be amplified/dampened by a low shelf equalizer (see also [Low Boost]).
	Low Boost	Controls the amount of amplification or dampening applied to the low frequencies.
	High Frequency	Sets the frequency above which all frequencies will be amplified/dampened by a high shelf equalizer (see also [High Boost]).
	High Boost	Controls the amount of amplification or dampening applied to the high frequencies.
	Boost Reset	Sets both boost controls to their default values.
Output	Delay Level	Adjusts the output level of the delay effect. This is independent of the [Main Level].
	Gater Level	Adjusts the output level of the gater effect. This is independent of the [Main Level].
	Main Level	Adjusts the output level of the main signal that is not processed by an effect unit.

3.7.3 Modulation Section

The three step sequencers of this section don't trigger samples - they are modulation sources which change sound engine parameters on the six sample sequencers.

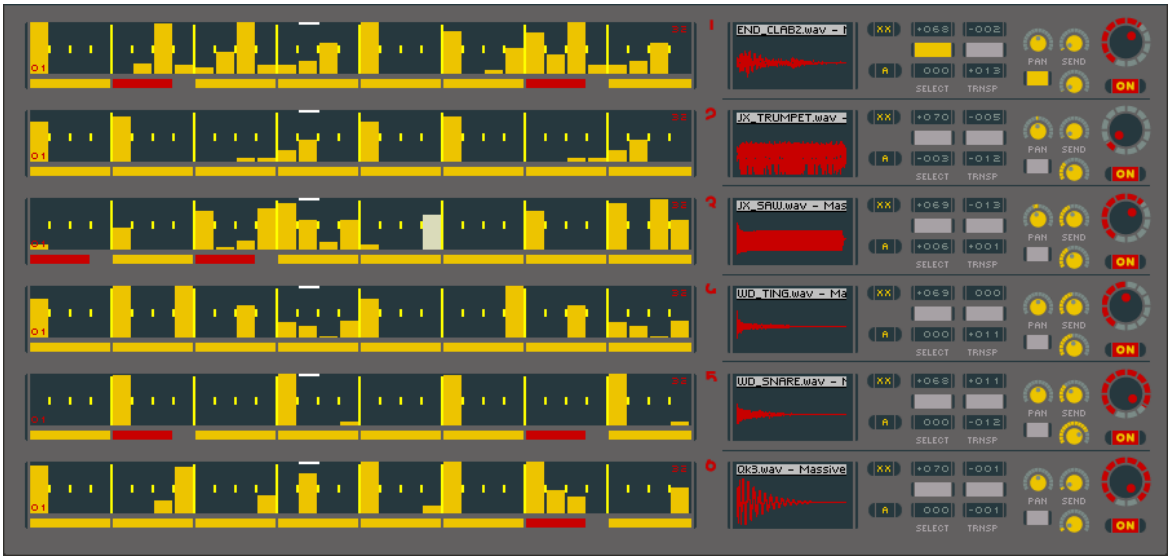


4.7 The Modulation section

Each of the modulation tracks is identified with a color that can be selected within the various modulation source selection controls (e. g. below the [Transpose] control of the Master section). Normally, you adjust the modulation amount below the source selection control of the modulated parameter.

Control	Function
Track Select	Switches between modulation track 1 (blue), 2 (green) and 3 (orange).
Sequencer Display	The modulation tracks' step sequencers act like the one described in the [Sequencer] section; the only difference is the absence of a roll mode.
Half Tempo	Switches between normal and half speed of the respective track read out: when pressed, each step is interpreted as an eight note; otherwise each step is interpreted as a sixteenth note.

3.7.4 Sequencer



4.8 The Sequencer section

Each of the six sequencer tracks provides two main areas: the main display with the individual steps and the smaller triplet control bar below. The triplet control bar groups every four sixteenth steps into a unit. If the triplet control bar is yellow, the steps above will be interpreted as sixteenth notes. If it is red, the steps are played as eighth triplets (and the last of the four steps becomes inactive).

Track Control Area	Function
Main Display	Displays the main gate sequence. The height of each step represents its velocity and can also be used as modulation source. With the left mouse button, the values can be drawn, copied or pasted depending on [Edit][Edit Mode]. With the right mouse button, the loop start can be set by clicking in the low part of the display. By clicking in the high part, the loop length can be adjusted. In the middle of the display, the right mouse button adjusts the roll mode of each step.
Triplet Control Bar	Controls whether a group of four steps (1-4, 5-8, etc.) is interpreted as sixteenth notes or as eighth triplet. In the latter case, each fourth step is not played.

The display is handled with the mouse:

- With the left mouse button, the steps and their velocity can be drawn, copied or pasted, depending on [Edit][Edit Mode]; a gate signal triggering the track's sampler unit is only sent at non-zero steps (see also [Always] in the sound engine).
- With the right mouse button, the loop start can be set in the lower area of the main display, and the loop's length in the upper area. In the middle, on the steps, the right mouse button adjusts the roll mode of each step through right-clicking and pulling the cursor up or down on a step. There are three roll modes that change their meaning depending on the triplet control.

Triplet Control Bar	Step Roll Mode	Resulting Length of a Step
yellow	No roll	Sixteenth notes
	Roll 1	Thirty-second notes; the step is triggered twice
	Roll 2	Thirty-second triplets; the step is triggered three times.
	Roll 3	Thirty-second triplets with pauses; at odd steps, the first and third note of the triplet is played, at even steps the second note. Thus, if two subsequent steps are set to this roll mode, a sixteenth triplet is played.
red	No roll	Eighth triplets.

Triplet Control Bar	Step Roll Mode	Resulting Length of a Step
	Roll 1	Sixteenth triplets; the step is triggered twice.
	Roll 2	Thirty-second triplets; the step is triggered four times.
	Roll 3	Similar to Roll 2.

At the top end of each track there is a control for adjusting the loop length, indicated by a small red number. Right-click and drag the mouse to select the loop area. A small marker indicates the current read-out position within the loop. A small white bar marks the current read-out position of the sequencer track.

3.7.5 Sound Engine



4.9 The sound engine

The sound engine contains a master section and six independent sampler units. It is located to the right of the modulation section; it sets global sample select offset and scales other parameters for all samplers simultaneously. If, for example, [Master][Transpose] is set to 12, all transpose controls of the six drum tracks are scaled to a range of 12 semitones; if the master control is set to 0, transposition is switched off for all tracks.

The parameters that control the samplers are grouped across five pages. The main page contains the sampler module itself where you load sample files; on the panel it is represented by the sample's waveform. Further controls select the sample from the map and adjust the pitch shift. The envelope section controls the sample's amplitude. The parameters of this page can be used to fine-tune the sample, particularly the influence of the gate velocity on attack and decay times. The start section adjusts the sample playback starting point. The filter section contains a low-pass and a high-pass filter whose sound can be smoothly cross-faded. The grain section, finally, controls the grain re-synthesis of the sampler; it is only available in the upper three sampler tracks. Here the frequency at which the sample is re-triggered can be controlled as [Speed]. At high values the sample is re-triggered so fast that only a very short part of it is heard, creating a familiar grain re-synthesis sound. Use the [Sampler][Start] control to modulate the grain sound.

Section	Control	Function
Master	Sample Select	Adjusts an offset for all [Sampler][Sample Select] controls of the six independent sampler tracks.
	Sample Select Modulation Source	Selects the modulation track that modulates the [Sampler][Select] parameter. (See also [Sample Select Modulation Amount].)
	Sample Select Modulation Amount	Controls the amount of modulation applied to the [Sampler][Select] parameter by the track selected in [Sample Select Modulation Source].
	Transpose	Adjusts an offset for all [Sampler][Transpose] controls of the six independent sampler tracks. (See also [Transpose Scale].)
	Transpose Modulation Source	Selects the modulation track that modulates the [Transpose] parameter. (See also [Transpose Modulation Amount].)
	Transpose Modulation Amount	Controls the amount of modulation applied to the [Transpose] parameter by the track selected in [Transpose Modulation Source].
	Decay Scale	Scales all decay times for each sampler track with the [Sampler][Decay] control.
	Decay Scale Modulation Source	Selects the modulation track that modulates the [Decay Scale] parameter. (See also [Decay Scale Modulation Amount].)
	Decay Scale Modulation Amount	Controls the amount of modulation applied to the [Decay Scale] parameter by the track selected in [Decay Scale Modulation Source].
	Cutoff Scale	Scales all filters cut-off frequencies for each sampler track with the [Sampler][Cutoff] control.

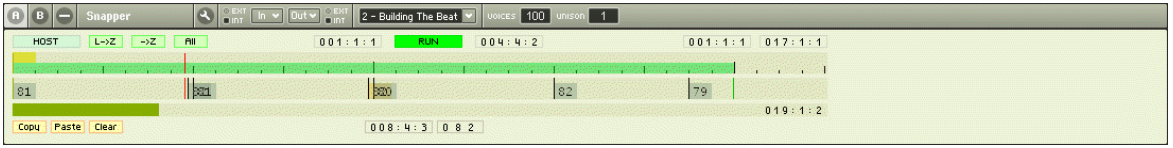
Section	Control	Function
	Drive Scale	Scales all pre-filter saturation drive amounts for each sampler track with the [Sampler][Drive] control.
	Transpose Scale	Scales all sample transpositions for each sampler track with the [Sampler][Transpose] control. (See also [Transpose].)
	Speed Scale	Scales all speed controls for each of the three upper sampler tracks with the [Sampler][Speed] control.
	Reset	Restores the values of all sampler tracks and the master section to default values.
Page Select		Selects the page displayed on the panel, controlling the [Sound Engine][Sample] section. The [Main] page contains the following parameters: [Select], [Transpose] and [Reset]. The [Envelope] page contains the following parameters: [Mute by Track], [Velocity], [Decay], [Dynamic Attack] and [Dynamic Decay]. The [Start] Page contains the [Start] parameter. The [Filter] page contains the following parameters: [Drive], [Filter Power], [Cutoff], [Resonance], [Low-pass / High-pass Crossfade] and [Reset]. The [Grain] page is only available for two of the six sampler tracks; it contains the following parameters: [Speed], [Grain] and [Reset]. The controls for [Pan], [Send Level] and [Track Level] are available in every page. Additional modulation controls are available for some of the controls mentioned above.
Sample	Sample Map Editor	Displays the currently selected sample. Double-click with the left mouse button to open Reaktor's sample map editor. One of the map's samples can be selected with the [Sample Select] control.
	Sample Select	Selects the track's sample played upon a trigger signal of the step sequencer. (See also [Sound Engine][Master][Sample Select].)
	Sample Select Modulation Source	Selects the source that modulates the [Sample Select] parameter. (See also [Sample Select Modulation Amount].)
	Sample Select Modulation Amount	Adjusts the amount of modulation applied to the [Sample Select] parameter by the source selected in [Sample Select Modulation Source].
	Transpose	Sets the amount of transposition applied to the selected sample. For the upper three sampler tracks this transposition does not affect the sample's playback speed (due to the underlying grain resynthesis algorithm); for the three lower sampler tracks it also changes the playback speed. (See also [Sound Engine][Master][Transpose] and [Sound Engine][Master][Transpose Scale].)

Section	Control	Function
	Transpose Modulation Source	Selects the source that modulates the [Transpose] parameter. (See also [Transpose Modulation Amount].)
	Transpose Modulation Amount	Adjust the amount of modulation applied to the [Transpose] parameter by the source selected in [Transpose Modulation Source].
	Always	If switched on, the pitch modulation (see [Transpose Modulation Amount]) is always active; if switched off, new modulation events are only recognized when the triggering gate is open.
	Velocity	Adjusts the amount of influence of the sequencer's gate velocity on the sample's amplitude. Turn to the left for no influence, i.e. constant maximum amplitude at every gate value; turn to the right for a complete mapping of the gate value onto the sample's amplitude.
	Decay	Sets the decay time of the amplitude envelope triggered by a gate event. (See also [Dynamic Decay] and [Sound Engine][Master][Decay Scale].)
	Dynamic Attack	Sets the amount of modulation by the source selected by [Dynamic Source] applied to the attack time of the amplitude envelope. Turn to the left for no modulation; turn to the right for long attack times at low modulation signals.
	Dynamic Decay	Sets the amount of modulation by the source selected by [Dynamic Source] applied to the decay time of the amplitude envelope. Turn to the left for no modulation; turn to the right for long decay times at high modulation signals. (See also [Decay].)
	Dynamic Source	Selects the source track that modulates the amplitude envelope's attack time and the [Decay] parameter.
	Mute By Track	Selects the mute track. If the specified track receives a gate signal, this track's gate is closed. The feature is particularly useful for programming hi-hats, e.g. the first track plays a closed hi-hat and the second one plays an open hi-hat -- since both tracks mute each other, an open hi-hat sound will be muted when the closed hi-hat sample is triggered. There is also a bypass option to exclude the track from muting.
	Start	Controls the sample playback start position upon a trigger signal.

Section	Control	Function
	Start Modulation Amount	Adjusts the amount and polarity of modulation applied to the read-out starting point. The value adjusted by [Start] is used as minimum offset: at normal modulation (values at the right), the modulation source's value is simply added to the [Start] value, scaled by the modulation amount of this control. At inverse modulation (values at the left), however, the modulation source's value is not subtracted from the offset, but the amount is inverted and added to the offset: At high modulation signals there is less modulation than at low modulation signals.
	Start Modulation Source	Selects the sequencer track whose signal modulates the [Start] parameter.
	Drive	Sets the amount of pre-filter saturation drive. (See also [Sound Engine][Master][Drive Scale].)
	Cutoff	Controls the cut-off frequency of the track's filter. (See also [Sound Engine][Master][Cutoff Scale].)
	Cutoff Modulation Source	Selects the sequencer track whose signal modulates the [Cutoff] parameter.
	Cutoff Modulation Amount	Sets the amount of modulation applied to the [Cutoff] parameter by the source selected by [Cutoff Modulation Source].
	Resonance	Sets the resonance of the track's filter.
	Mode Crossfade	Fades between the unfiltered signal (at the left), the signal of a high-pass filter (at mid position) and a lowpass filter signal (at the right). Both filters use the parameters adjusted by [Cutoff] and [Resonance].
	Speed	Sets the speed at which the sample is re-triggered and the playback is looped back to the position defined by the [Start] parameter. As the sampler's playback speed remains constant, the played length of the sample is reduced at high re-triggering frequencies; finally only grains of the original sample are heard. The starting sample grain can be selected by the [Start] parameter.
	Speed Modulation Source	Selects the sequencer track whose signal modulates the [Speed] parameter.
	Speed Modulation Amount	Sets the amount of modulation applied to the [Speed] parameter by the source selected by [Speed Modulation Source].

Section	Control	Function
	Reset	Sets all controls of the respective controller page to their default values. Clicking with the right mouse button resets the parameters on all controller pages.
	Pan	Positions the track's signal within the stereo field.
	Pan Modulation Source	Selects the sequencer track whose signal modulates the track's [Pan] parameter.
	Send Level	Sets the volume of the track's signal that is sent to the delay effect (upper control) and the gater effect (lower control).
	Track Level	Sets the volume of the track's signal that is routed to the main output.
	Solo/Mute	The left mouse button switches the track's signal on or off. Double-clicking with the left mouse button switches all tracks on. Clicking the right mouse button solos the track (i.e. it plays alone).

3.7.6 Snapper



4.10 Snapper user interface

Snapper is an additional instrument that works as a song sequencer. It synchronizes to the global MIDI clock and automatically recalls snapshots of the slaved instrument, i.e. Massive. As Massive's snapshot management system is optimized for fast snapshot loading, the patterns are loaded immediately, with no interruptions of the audio output. (Be sure to set the [Snapshot Mode] control to external!) Complete songs can be built using Snapper's snapshot timeline.

The instrument has several horizontal rows of controls. The most important one contains the snapshot numbers to be recalled (third from the bottom). A new recall event can be created by double-clicking with the left mouse button into an empty area of the row; double-clicking an

existing marker deletes it. Each marker can be dragged horizontally with the left mouse button along the time axis; by dragging with the right mouse button the snapshot number to be recalled can be altered.

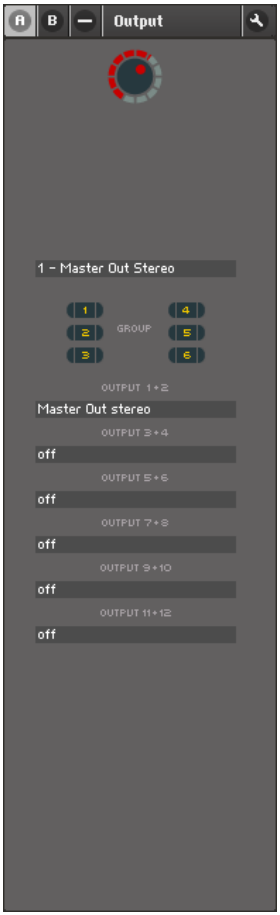
Below this main row there is the display row; here, dragging with the left mouse button moves the area to be displayed above while the right mouse button changes the length of the displayed area. Above the main snapshot row there is the loop bar that defines the read-out loop as well as the edit area of the edit functions (see [Copy] etc.). Additionally, there is a small marker to position the read-out pointer manually; its position indicates the place where the readout starts after a MIDI clock reset (i.e. not after a MIDI continue event).

In the top and bottom rows, finally, several values are displayed numerically, always in the format of bar : quarter : sixteenth note. There are also buttons to copy and paste sequencer data, to zoom the currently displayed loop, to disable recall (during sequence editing) and to switch the sequencer on or off. The sequencer is bound to the MIDI clock, so Reaktor‘s global clock or an external sequencer‘s clock has to be running for Snapper to work.

Control	Function
Host	Turns Snapper into slave mode. A host program (like Logic or Cubase) can send MIDI program change events to select Massive snapshots. In this case, Snapper’s song sequence is bypassed. The host’s track containing the MIDI program change events should be played a few milliseconds (depending on the host) before all other tracks to ensure correct timing within Massive.
Zoom Loop	Zooms the sequence display (see [Display Marker]) to the current loop area (see [Loop Marker]).
Previous Zoom	After zooming the display to the loop area with [Zoom Loop], this button returns the display to the previous zoom factor.
Zoom All	Displays the complete song sequence.
Start Position	Displays the position at which the sequencer starts running upon a MIDI start event. This position can be edited by using the [Start Marker].
Current Position	Displays the current position of the sequence read-out.
Run Switch	Switches the sequencer on or off. When on, the instrument reacts to MIDI clock events (e.g. from Reaktor’s global clock) like start, stop and continue.
Loop Start	Displays the start of the playback loop area. This position can be edited by the [Loop Marker].

Control	Function
Loop Length	Displays the length of the playback loop area. This position can bed edited by the [Loop Marker].
Start Marker	Displays the position at which the sequencer starts running upon a MIDI start event. This can be altered by dragging the marker with the left mouse button.
Loop Marker	Displays the playback loop. When the read-out pointer reaches the end of the loop area, it is set back to the loop's beginning. The loop's start can be changed by dragging the marker with the left mouse button, dragging with the right mouse button changes its length.
Sequence Edit	Defines the sequence of snapshot recall events. A new event can be created by double-clicking an empty area of the sequence with the left mouse button; double-clicking on an existing marker deletes it. The markers can be selected and moved along the horizontal time axis with the left mouse button. The snapshot number can be adjusted With the right mouse button. The position and snapshot number of the currently selected marker are displayed as [Recall Position] and [Recall Snapshot] at the bottom of the instrument; they can also be edited there.
Display Marker	Shows the area of the complete song that is shown within the [Sequence Edit] display. With the left mouse button the start position (also displayed at the left of the row) can be moved, with the right mouse button the display length can be adjusted (its end is displayed numerically at the right of the row). The [Zoom Loop] button sets the [Display Marker] to the position and length of the current loop (see [Loop Marker]).
Copy	Copies the snapshot sequence of the current loop area into the internal clipboard.
Paste	Pastes the snapshot sequence stored in the internal clipboard into the current loop area.
Clear	Deletes all snapshot recall markers within the current loop area.
Recall Position	Displays the position of the currently selected snapshot recall event. A snapshot recall event can be selected within the [Sequence Edit] display with the left mouse button.
Recall Snapshot	Displays the snapshot number that is recalled by the currently selected snapshot event.

3.7.7 Output

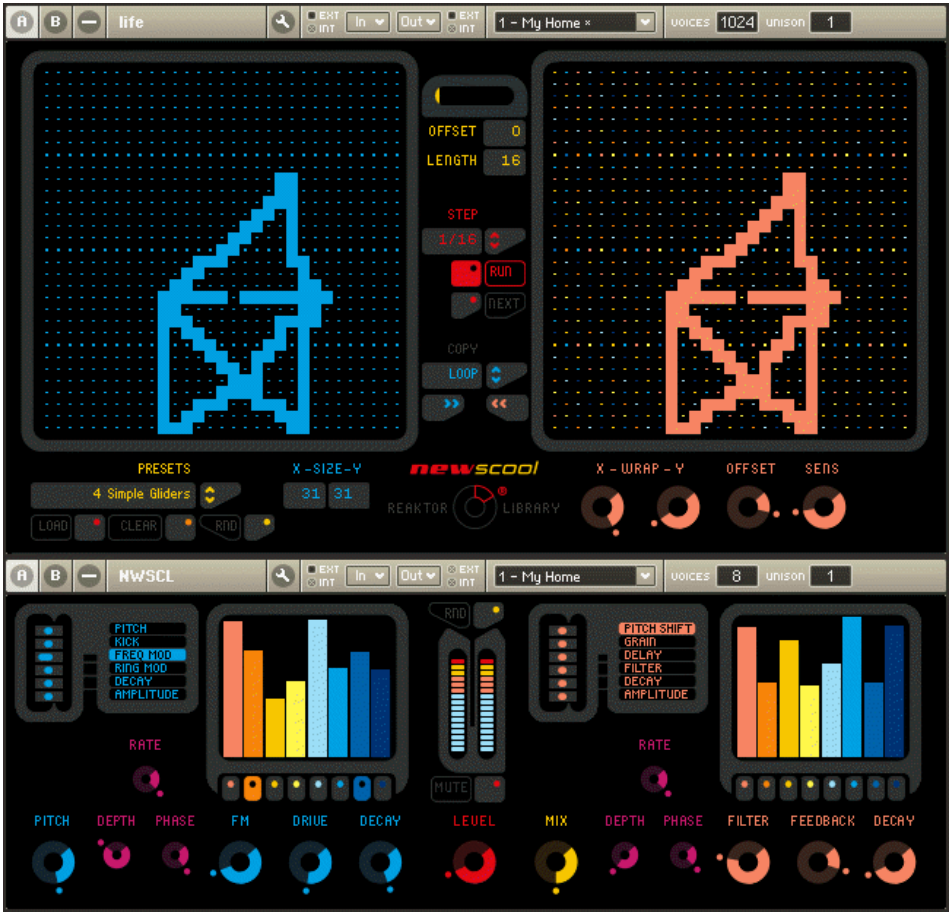


4.11 Massive's output utility

This utility controls the routing of Massive's output ports to the soundcard. There are six stereo pairs of output channels. Each one can output an individual track, the mixed master stereo signal or a group submix. The group submix can be any combination of the six tracks (e.g. the hihat tracks).

Control	Function
Master Level	Adjusts the master level of the complete ensemble. The ensemble is automatically muted when the MIDI clock is stopped.
Preset Select	Selects one of the instrument's presets. Those presets are normal instrument snapshots, so the list of presets can be extended individually.
Group	Defines the tracks whose signal is mixed to a group signal. This group signal can then be routed to one of the outputs. Each track can be selected or deselected with the mouse.
Outputs	The six stereo output slots are connected to Reaktor's soundcard output. Each slot can route an individual track, the group submix, or the complete mix to the soundcard for external mixing.

3.8 Newscool



4.12 The Newscool user interface

3.8.1 Introduction

Newscool is a REAKTOR classic– now it's completely rebuilt, with an innovative sequencer (at the top) and the characteristic sound engine (at the bottom). The engine consists of a tone generator on the left and a multi-effect unit on the right. The signal is produced by eight parallel oscillator units whose parameters are modulated extensively. The effect unit parameters – providing pitch shifting, delay and filter - are similarly modulated.

The sequencer is based on the Life model developed by John Conway in the 1970s. A two-dimensional pattern is processed in steps: An element of the pattern becomes alive (dark in this implementation) in the following step if three of its eight neighbors are alive in this step; it remains alive in the subsequent one if two or three neighbors are alive in the current one – else it dies (and becomes a light square again). Several patterns emerge over time by this set of rules: Gliders move over the grid, crosses oscillate in several phases, some objects remain stable and don't change from step to step while others remain unstable forever. These patterns trigger the sound engine, generating "lively" sequences.

3.8.2 Life Sequencer

As explained above the sequencer proceeds from one step to the next one by a set of Life rules that translate the current pattern into the following one. The two-dimensional Life pattern is mapped onto the eight channels of the tone generator by the grid of the [Performer Display]: By using the [Wrap X/Y] controllers this mapping can be modified smoothly. The [Sensitivity] knob also interacts with the trigger signals.

Within the [Board Display] Life patterns can be loaded from a bank of factory presets. These patterns can be altered, or you can build completely new ones. The [Board Display]'s content can be copied to the [Performer Display] manually, at the beginning of the Life evolution or at the beginning of each loop.

Control	Function
Loop Display	Shows the process of the loop steps. (See also [Run] and [Length].)
Offset	Sets an offset in steps to the sequencer read-out.
Length	Adjusts the length of the loop in steps. Since the pattern of the [Board Display] can be copied automatically to the [Performer Display] at the beginning of each loop cycle, the loop length controls how often the performer resets to the initial pattern.

Control	Function
Step	Selects the step length of the life sequencer in MIDI units, e.g. selecting a sixteenth calculates a new pattern life phase each sixteenth of the MIDI clock.
Run	Switches the life process on or off. When on, each MIDI clock step (see [Step]) generates a new phase of the pattern according to the life rules (see the instrument description); the result is displayed in the [Performer Display]. The MIDI clock has to be running, or else this button shows no effect.
Next	Calculates the next life sequencer phase independently of the MIDI clock.
Copy	Selects at which point the pattern of the [Board Display] is copied to the [Performer Display]: manually (by pressing the [To Performer] button), at the start of the sequencer when the [Run] button is pressed, or at the beginning of each loop cycle (see [Length]).
To Performer	Copies the pattern of the [Board Display] to the [Performer Display].
To Board	Copies the pattern of the [Performer Display] to the [Board Display].
Board Display	This is a buffer where life patterns can be loaded from the preset list (see [Presets]), edited, or randomly generated. You can draw patterns directly into the display with the mouse.
Presets	Selects a pattern from a list of factory presets, which can then be loaded into the [Board Display] by pushing the [Load] button.
Load	Copies a pattern from the list of factory presets into the [Board Display].
Clear	Deletes the current pattern of the [Board Display].
Random	Randomly generates a pattern within the [Board Display].
Size X/Y	Sets the size of the [Board Display]. When the pattern is copied to the [Performer Display], the size parameters are also adapted to the performer.
Performer Display	Shows the current life phase; its pattern is also used to calculate the next phase. It cannot be edited, patterns can only be copied to it from the [Board Display] (see also [Copy] and [Length]). The grid behind the pattern is used to map the two-dimensional pattern onto a one-dimensional rhythmic sequence (see [Wrap X/Y]).
Wrap X/Y	Controls the projection of the pattern onto the audible sequence; the ratio between horizontal and vertical wrap parameters is visible as a grid within the [Performer Display].

Control	Function
Offset	Adds an offset to the [Wrap X/Y] parameters, thus altering the sequence by shifting it in time.
Sensibility	Determines how many trigger signals are generated from the pattern of the [Performer Board]. Turn to the right for dense trigger sequences, turn to the left for the opposite effect.

3.8.3 Newscool

The sound engine consists of a tone generator (in the parameter list below referred to as TG) and a multi- effect unit. Both achieve their characteristic sounds via vast modulation of their parameters by two simple LFOs. Those parameters control eight independent synthesizer tracks that are triggered by the [Life Sequencer]; each of the tracks can be muted. The [Random] button sets all those parameters to random values; within the [TG / Effect] Poly Control] areas they can be controlled manually. The parameter shown within these displays is selected using [TG / Effect Parameter Select] controls.

Control	Function
TG Poly Control	Sets the parameters for the tone generator. There are eight bars, one for each track; the value can directly be drawn into the display. The parameter displayed is selectable by [TG Parameter Select].
TG Mute Track	Switches the tracks' tone generators individually on or off.
TG Parameter Select	Selects which parameter of the tone generator is displayed and edited within [TG Poly Control]. There are six parameters available: Pitch, Kick Amount, Frequency Modulation Amount, Ring Modulation Amount, Decay Time and Amplitude.
TG Parameter Modulation	Displays the modulation value for each parameter; by clicking into the display the modulation of the respective parameter can be switched on or off. For modulation, a sine LFO is used (see [TG Modulation Rate/Depth/Phase]).
TG Modulation Rate	Sets the speed of modulation in sequencer steps.
TG Modulation Depth	Sets the amount of modulation.
TG Modulation Phase	Sets the phase of the sine LFO.

Control	Function
Pitch	Sets the absolute range of the pitch modulation. This is a bipolar control: turn the knob to the left for inverse modulation and to the right for normal modulation. There are individual (relative) values for each track adjustable in the [TG Poly Control].
FM	Sets the absolute amount of frequency modulation. There are individual (relative) values for each track adjustable in the [TG Poly Control].
Decay	Sets the absolute decay time. There are individual (relative) values for each track adjustable in the [TG Poly Control].
Drive	Sets the amount of saturation drive applied to the tone generator's signal.
Effect Poly Control	Sets the parameters for the tone generator. There are eight bars, one for each track; the value can directly be drawn into the display. The parameter displayed is selectable by [Effect Parameter Select].
Effect Mute Track	Switches the tracks' effect units individually on or off.
Effect Parameter Select	Selects which parameter of the effect unit is displayed and edited within [Effect Poly Control]. There are six parameters available: pitch shift amount, pitch shift grain size, pitch shift delay time, filter frequency, decay time, and amplitude.
Effect Parameter Modulation	Displays the modulation value for each parameter; by clicking on the display the modulation of the respective parameter can be switched on or off. A sine LFO is used for modulation (see [Effect Modulation Rate/Depth/Phase]).
Effect Modulation Rate	Sets the speed of modulation in sequencer steps.
Effect Modulation Depth	Sets the amount of modulation.
Effect Modulation Phase	Sets the phase of the sine LFO.
Filter	Sets an absolute offset to the effect's filter frequency, shifting the individual values of each track that can be edited in the [Effect Poly Display].
Feedback	Sets the level of the signal that is routed from the effect's output back to its input.
Decay	Sets an absolute offset to the effect's decay time, shifting the individual values of each track that can be edited in the [Effect Poly Display].
Mix	Controls the ratio between the unprocessed, dry sound (at the left) and the effect's wet signal (at the right).
Level	Sets the instrument's master level.

Control	Function
Mute	Mutes the complete instrument.
Random	Randomly sets all parameters of each track within [TG Parameter Display] and [Effect Parameter Display].

3.9 Random Step Shifter



4.17 The Random Step Shifter's user interface

3.9.1 Introduction

Random Step Shifter uses intelligent pseudo-random principles to cut-up and rearrange sample loops, on-beat, in real-time. There's an intuitive three-part sequencer that triggers sample playback. It also modulates sample selection, positional offset, and playback pitch. In addition, these modulations can be mangled by various pseudo-random sequences. This instrument will create new sample loops for you very easily! You can load in any loop, just keep in mind that you need to cut the loops accurately so that they play correctly when they are looped over their entire length.

3.9.2 SQ2

The Randomstepshifter contains a simple step-sequencer. It consists of three parts, the [Select], [Offset], and [Pitch] sequencers. Each of the parts has two tracks - the trigger track at the bottom of the sequencer, and the modulation track above. The trigger tracks can be used independently from the modulation tracks but you can't modulate without a trigger. In other words, you can trigger the envelope without sending any modulations, but not vice-versa. In the [Envelope] section you can choose which of the three trigger tracks is used for starting the envelope. The trigger events of the [Offset] track can also be used to reset sample offset. The modulation tracks can be used to modulate the sample player's main parameters. These are the [Select] parameter for sample selection, the [Offset] parameter for controlling the start position in the currently selected sample, and the [Pitch] parameter controlling pitch of sample playback.

Control	Function
Loop bar	The bar above the sequencer grid represents the sequencer's loop region. Right-click (ctrl-click for Mac users) to set length. Left-click and drag to move.
Modulation tracks	Click on the grid to create modulation events. Drag the mouse up or down to set the level. These events are wired to the [Sample select], [Sample offset], and [Pitch] modules, respectively. They can be used to modulate the parameters of the respective modules in a controlled or randomly varied way. These parameters are sample selection, sample offset, and pitch of playback. Right-click (ctrl-click if you're on a Mac) to delete the event, along with its trigger event. You will automatically create [Trigger] steps associated with the modulation events. See [Trigger tracks] for more information. Drag the modulation bar completely down to have a trigger event without modulation output. The three tracks can be selected with three buttons below the sequencer ([Select], [Offset], and [Pitch]).

Control	Function
Trigger track	Click to create events that trigger the [Envelope]. Drag down the corresponding modulation event to zero if you want a sole trigger without modulation. All three trigger tracks can be used to start the envelope. Use the respective buttons in the [Envelope] section to choose which track will do it. Furthermore, the [Offset] trigger track can reset the sample offset if the [Seq] button in the [Sample offset] section is active.
Select / Offset / Pitch	These buttons switch the view to show the three tracks of the sequencer. The modulation part of the [Select] track is wired to the [Sample Select] module, the modulation part of the [Offset] track is wired to the [Sample Offset] module and the modulation parts of the [Pitch] track is wired to the [Pitch] module.
Copy	Copies the current loop region into the clipboard.
Paste	Pastes the pattern clipboard into the current pattern.
Rand	Randomizes the current loop region.
Clear	Clears the current loop region.
Zoom Level (16 st, 32 st, 64 st)	Click and drag mouse up or down to zoom in and out of the currently displayed pattern.
Clock divider (1/6, 1/8, 1/12, 1/16, 1/24, 1/32)	Choose between different clock divisions. This speeds up or slows down the pattern but retains a metric relationship to the original speed. You get original speed with the 1/16 setting.
Run	Starts and stops the sequencer.

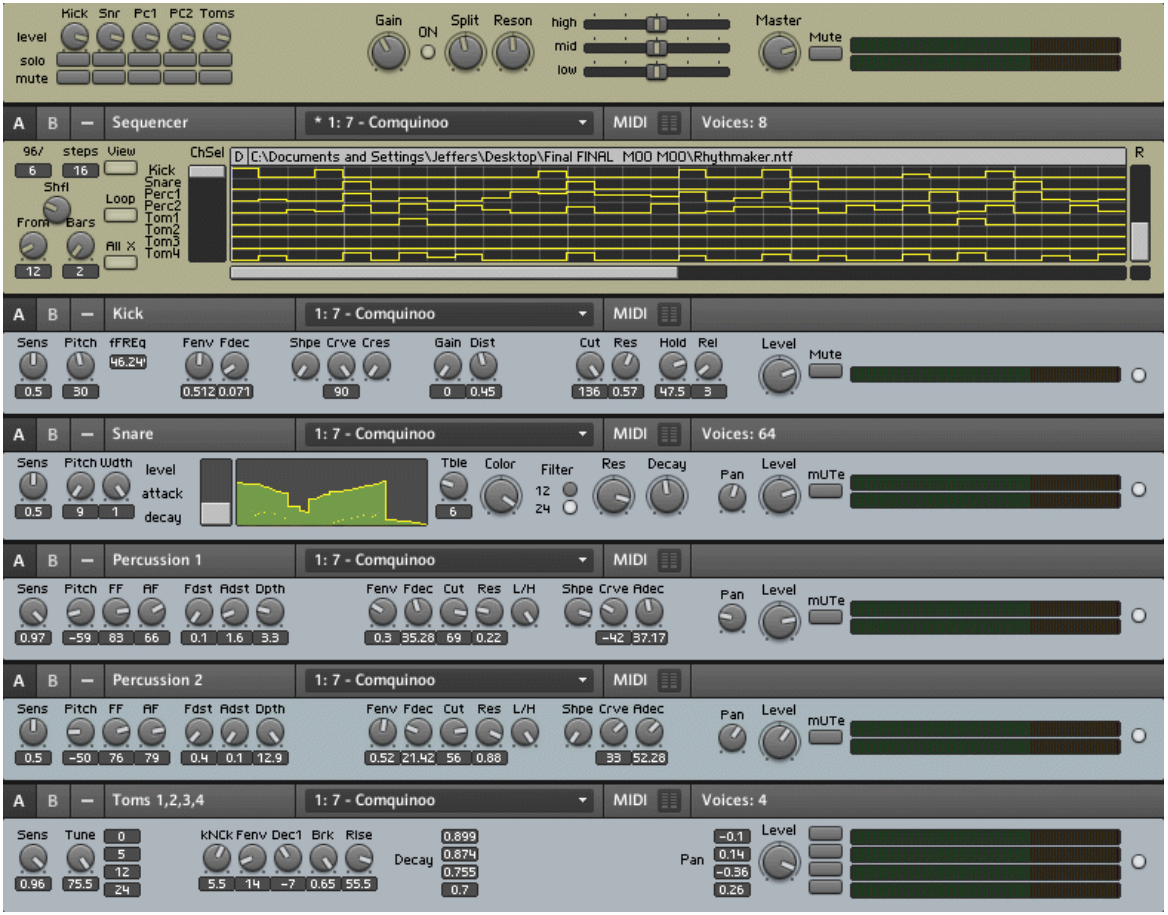
3.9.3 Sampler

At the heart of the Randomstepshifter lies the sample player. Just load your pre-cut loops into the sample map and the patch will rearrange them. You are free to use the modulation events from the three [Sequencer] tracks to control the parameters for [Sample select], [Sample offset] and [Pitch], or to let these three parameters be randomly varied. Activate random mode (the [Rnd] buttons) and adjust the three [RAND] knobs to get different pseudo-random results. Try moving the [QNTZ] knob in the [Sample offset] module while the sequencer runs for interesting dynamic sample cut-up in real-time. Please make sure that all samples in the sample map have a transposition value of 0.

Section	Control	Function
Sample select	Rand	This knob selects one of the pseudo-random sequences. Each incoming value from the modulation track is transformed in a pseudo-random way.
	Seq / Rnd / Off	These three buttons switch the modulation modes for the [Select] parameter. Choose between direct modulation by the modulation track, random variation based on the modulation track, and no modulation.
	Select	Sets the base for sample selection via the modulation track. This is the sample that gets played when [MOD] is inactive.
	FIRST	Defines the start point of the range for sample selection in the sample map.
	LAST	Sets the end point of the range for sample selection in the sample map.
Sample offset	RAND	This knob selects one of the pseudo-random sequences. Each incoming value from the modulation track is transformed in pseudo-random way.
	Seq / Rnd / Off	These three buttons switch the modulation modes for the [Offset]] parameter. Choose between direct modulation by the modulation track, random variation based on the modulation track, and no modulation.
	MOD	Switches the modulation input from the sequencer on for the [Sample offset] module.
	Offset	Sets the base offset in the sample. This is the offset that is applied when [MOD] is inactive.
	QNTZ	Controls sample offset quantization. 1 = 1/16th, 2= 1/8, 4 = 1/4, etc.
	Smth	Controls for re-synthesis smoothness of sample playback. This alters the sound when introducing extreme pitch settings.
Pitch Env	RAND	This knob selects one of the pseudo-random sequences. Each incoming value from the modulation track is transformed in a pseudo-random way.
	Seq / Rnd / Off	These three buttons switch the modulation modes for the [Pitch] parameter. Choose between direct modulation by the modulation track, random variation based on the modulation track, and no modulation.

Section	Control	Function
	MOD	Switches on the sequencer's modulation for the [Pitch] module.
	Pitch	Sets the transposition of the sample. This is the transposition applied when [MOD] is inactive. This interacts with the [RANGE] control and is independent from tempo as long as [Fit] is inactive.
	RANGE	Sets the range of bipolar sample transposition in semitones. A value of 12 gives you a transposition range from -12 to +12 semitones.
	Fit	When active, the pitch follows the tempo changes of sample playback, like in a conventional sample player.
	Attack	Sets the attack time of an ADSR envelope triggered by the sequencer events.
	Decay	Sets the decay time of an ADSR envelope triggered by the sequencer events.
	Sustain	Sets the maximum level the envelope will reach.
	Release	Sets the time that passes until the envelope is completely faded out after it has reached the sustain level.
	Sel / Offs / P	Selects the trigger input for the envelope. It can be the trigger track from the select-, offset-, or pitch track, respectively.
	on	Switches the envelope on or off.
Output	Mute	Mutes sound output from the sample player.
	Gain	Controls the main volume of the sample player.

3.10 Rhythmaker



2.6 Rhythmaker user interface

Introduction

Rhythmaker is a sequenced drum machine that can also be played over MIDI. It is composed of a single 8 channel sequencer and eight sound generators: kick, snare, two percussion modules, and a four-voice tom generator. The system clock needs to be running for the sequencer to run. To use Rhythmaker as a MIDI controlled percussion module, simply play MIDI notes C1 to G1, which correspond to the following map:

- C1: Kick
- C#1: Snare
- D: Percussion 1
- D#: Percussion 2
- E: Toms 1
- F: Toms 2
- F#: Toms 3
- G: Toms 4

Since this ensembles contains many similar controls, repetitive structures are explained only once. For instance, the <Lev>, <Power> and <Mute> controls in the drum panels are explained only once as they occur in each drum module.

Sequencer

The eight channel event table sequencer controls each of the eight percussion generators. The tempo of the sequencer is controlled by the system clock. The sequencer supports odd time signatures, randomization, and shuffle.

- <96/> Sets the division factor for the clock frequency. The internal clock produces 96 events per beat - each event makes the sequencer proceed to the next step. Using every event would result in a seqeuncer with a 96th grid. However, if only every sixth event is used, the seqeuncer has a traditional 96 / 6 = 16th grid. The speed and rhythm of the seqeuncer can easily be changed by modifying the this control. A value of 6 translates to a 16th notes, 12 is 8th notes, 8 gives you 8th triplets, etc.
- <steps> Sets the number of steps in one bar
- <from> Sets the offset in the event table in bars, i. e. the first bar that is used.

- <bars> Sets the number of bars to be displayed and looped
- <View> When off, only the selected channel (see <chsel>) is displayed and can easily be edited; when on, all nine data rows of the seqeuncer are shown.
- <loop> Turns the loop mode on or off. When on, the loop length is set by <bars> and starts at bar <from>.
- <all X> When off, only the currently playing bar is displayed; when on the complete loop is shown.
- <chsel> Selects the channel to be edited (see <View>).
- <Event Table> Contains the seqeuncing events. Deselect <View> to leave 2D mode for editing of data.
- <R> This small vertical XY control to the right of the sequencer randomizes values in the event table. <R> either randomizes the events in the current channel <ChSel> or, when <all> is selected, all channels.
- <all> When on, adjusting <R> results in filling the visible area of the Event Table with randomized values. When off, clicking into <R> only alters the current channel <ChSel>.
- <shfl> Sets the amount of shuffle, i. e. the amount of delaying quantization.

Kick

- <Sens> Sets the sensibility of the module to incoming gate events. At low values, varying velocity of the gate signal doesn t effect the sound much, while at higher values, velocity modulates amplitude and pitch.
- <Pitch> Sets the fundamental pitch.
- <fFreq> Shows the fundamental pitch of the kick drum in Hz
- <Fenv> This value is added to the fundamental pitch to calculate the initial pitch when the kick drum is triggered. The pitch will glide down to the fundamental pitch, with the glide speed set by <Fdec>.
- < Fdec > Sets the time time of the pitch slide.Technically speaking, this sets the decay time of a DR-envelope whose output is added to the fundamntal pitch; by increasing this value, it will take longer for the envelope to get to the fundamental pitch set by <Pitch>.

- <Shpe> Crossfades between the a modified frequency envelope and the unmodified one (see <Crve> and <Cres>). Turn left for the unmodified envelope.
- <Cres> and <Crve> modify the envelope used to interpolate between the initial und fundamental pitch (see <fENv> and <fDEc>) by a 2-pole filter that acts as a smoother.
- <Gain> Sets the amount of overdrive distortion.
- <Dist> Sets the amount of quantizing distortion.
- <Cut> Sets the cutoff frequency of a 2-pole filter, used after the distortion modules.
- <Res> Sets the resonance of the filter.
- <Hold> Sets the hold value of a HR-envelope to control the amplitue of the kick drum
- <Rel> Sets the release value of the HR-envelope.
- <Level> Sets the output level.
- <On> Turn the whole instrument on / off. When off, CPU is saved, but there is a slight delay in turning the instrument on again. For quick muting and unmuting, it s better to use <Mute>.
- <Mute> Instantly mutes the whole instrument, but the CPU is still used.

Snare

The Snare module uses spectral synthesis to allow hand-drawable frequency response curves. By using Reaktor s polyphony, the Snare module has 64 partials. You can change the sound of the snare by changing the number of voices. Naturally, fewer voices means that less CPU will be needed, but the sound of the snare will be less complex.

- <Sens> Sets the sensibility of the module to incoming gate events. At low values, varying velocity of the gate signal doesn t effect the sound much, while at higher values, velocity modulates amplitude and pitch.
- <Pitch> Sets the fundamental pitch.
- <Wdth> Sets the snare s frequency range.
- <Level, Attack, Delay> Selects which spectral envelope table to edit.

- <Event Table> Shows the spectral envelope for either the Level, Attack, or Delay. The horizontal access corresponds to frequency (lowest = <Pitch>, highest = <Pitch> +<Wdth>).Vertial axis is the amplitude at that frequency. (-40 ... 0 db -for "level" channel, 0 ... 60 db for "attack" and "decay" channels).
- <Color> Filters the snare. Minimum values are more thin, while maximum values are more rich.
- <Filter> Selects the filtertype for <Color> — 12 pole or 24 pole.
- <Res> Filter resonance for <Color>.
- <Pan> Stereo panning.
- <Level> Sets the output level.
- <On> Turn the whole instrument on / off. When off, CPU is saved, but there is a slight delay in turning the instrument on again. For quick muting and unmuting, it s better to use <Mute>.
- <Mute> Instantly mutes the whole instrument, but the CPU is still used.

Percuss 1 and 2

- <Sens> Sets the sensibility of the module to incoming gate events. At low values, varying velocity of the gate signal doesn t effect the sound much, while at higher values, velocity modulates amplitude and pitch.
- <Pitch> Sets the basic pitch. It sets the basic pitch of the multi-sine oscillator at the heart of the synth whose parameters are modulated by eight triangular oscillators. These eight oscillators are controlled by the knobs <FF>, <AF>, <Fdst>, <Adst> and <Dpth>.
- <FF> Sets the main pitch of the four triangular oscillators used to modify the freqeuncies of the basic multi-sine oscillator. This pitch is modified by adding different multiples of <Fdst> to get the effective pitch
- <AF> Sets the pitch of the four triangular oscillators used to modify the amplitudes of the basic multi-sine oscillator. This pitch is modified by adding different multiples of <Adst> to get the effective pitch
- <Fdst> Sets the distance (the step) between the pitches of the four freqeuncy modulating triangular oscillators (see also <FF>).

- <Adst> Sets the distance (the step) between the pitches of the four amplitude modulating triangular oscillators (see also <AF>).
- <Dpth> Sets the amplitude of the frequency modulating triangular oscillators.
- <Fenv> This value is added to the fundamental pitch to calculate the initial pitch when the drum is triggered. The pitch will glide down to the fundamental pitch, with the glide speed set by <Fdec>.
- < Fdec > Sets the time time of the pitch slide.Technically speaking, this sets the decay time of a DR-envelope whose output is added to the fundamntal pitch; by increasing this value, it will take longer for the envelope to get to the fundamental pitch set by <Pitch>.
- <Cut> Sets the cutoff frequency of the filter.
- <Res> Sets filter resonance.
- <L/H> Sets the balance betwen lowpass (left) and highpass (right) filter.
- <Shpe> Crossfades between the <Crve> modified amplitude envelope and the unmodified one (see <cRVe>). Turn left for the unmodified envelope.
- <Crve> Sets the cutoff frequency of a 1-pole filter used to smooth the amplitude envelope.
- <Adec> Sets the decay of a DR-envelope used to control the amplitude of the synth.
- <sPRd> Sets the position of the synth in the stereo field. The position is modified by the velocity of the gate signal: the more quiet the note, the more centered the sound is.
- <Level> Sets the output level.
- <On> Turn the whole instrument on / off. When off, CPU is saved, but there is a slight delay in turning the instrument on again. For quick muting and unmuting, it s better to use <Mute>.
- • <Mute> Instantly mutes the whole instrument, but the CPU is still used.

Toms 1,2,3,4

- Unlike the other three synth modules (Kick, Snare, and Percuss), the Toms module is four-voice polyphonic. Four channels in the sequencer are dedicated to sequencing the toms.

- <Sens> Sets the sensibility of the module to incoming gate events. At low values, varying velocity of the gate signal doesn t effect the sound much, while at higher values, velocity modulates amplitude and pitch.
- <Tune> Sets the main pitch which is modified by <offset 1 - 4> to get the effective pitches of the four toms. Technically speaking, this pitch is used to control the cutoff freqeuncy of a 2-pole filter whose self-oscillation is used as sound source of the synth (see also <kNCk> and <Decay 1 - 4>).
- <Tune offset 1 - 4> Sets the offset values of the four independent toms that are added to the main pitch controlled by <tune>.
- <kNCk> Sets the attack, decay and release value of an ADR-envelope. This envelope is used as impulse input for the self-oscillating filter.
- <Fenv> This value is added to the fundamental pitch to calculate the initial pitch when the drum is triggered. The pitch will glide down to the fundamental pitch, with the glide speed set by <Dec1>, <Brk>, and <Rlse>. Unlike the frequency envelopes in the other synths, Toms uses a DBR (decay break release) envelope instead of a simple DR (decay release) one.
- <Dec1> Sets the first decay value of the DBR-envelope.
- <Brk> Sets the break level of the DBR-envelope.
- <Rlse> Sets the second decay and release value of the DBR-envelope.
- <Decay 1 - 4> Sets the decay time of the four toms. These values control the resonance of the self-oscillating filter, Lower resonance results in less self-oscillation which results in a shorter decay time.
- <Pan 1 - 4> Set the position of the four toms in the stereo field.
- <Level> Sets the master output level.
- <Mute 1 - 4> Instantly mutes toms 1-4, but the CPU is still used.
- <On> Turn the whole instrument on / off. When off, CPU is saved, but there is a slight delay in turning the instrument on again. For quick muting and unmuting, it s better to use <Mute>.
- <XY-Panel> Displays a visualization of the amplitude of the four toms.

Mixer

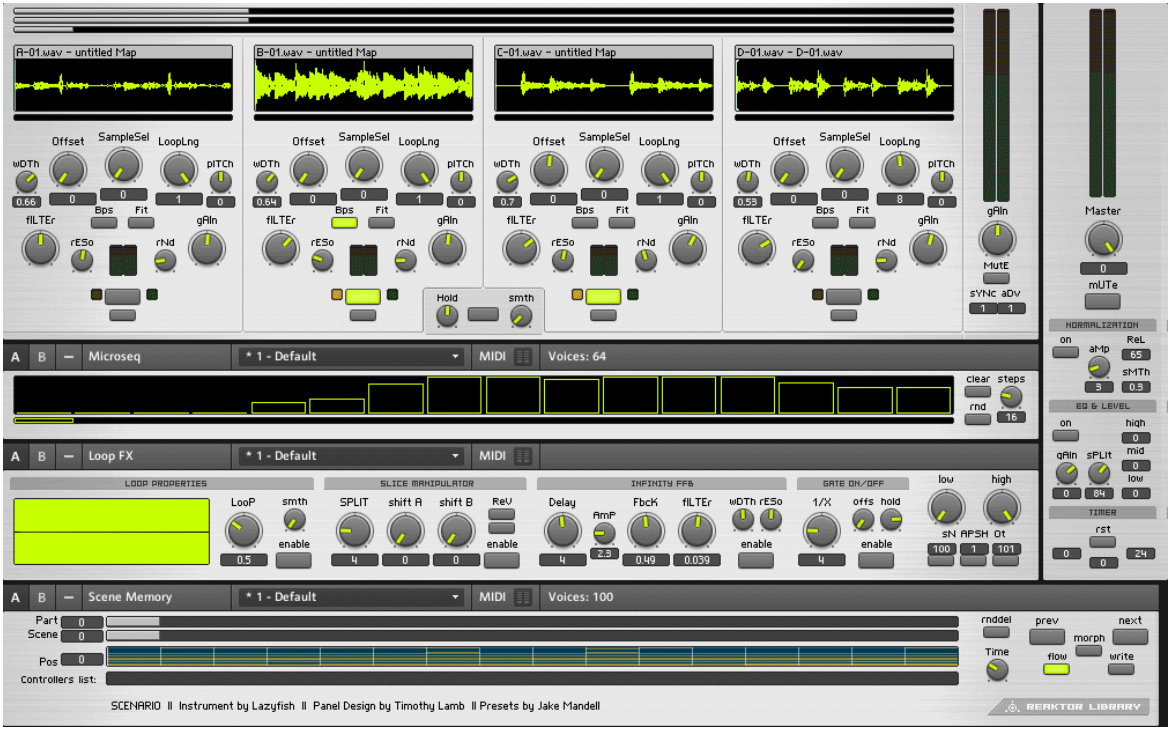
Each drum module has its own channel in the mixer. <solo> washes your dishes, <mute> paints your walls, and the little knob under the channel name does your laundry.

Saturator

A three band saturator adds volume and depth to the mix before the output.

- <Gain> Sets the amount of overdrive distortion.
- <Split> Sets the cutoff freqeuncy of the main 2-pole filters used to get 3 bands of the input sound: one processed with a lowpass filter, the second with a bandpass and the third with a highpass. After filtering, each band is saturated separately.
- <Reso> Sets the resonance of the 2-pole filter (see <Split>).
- <low> Sets the amplification value of the low frequencies.
- <mid> Sets the amplification value of the middle frequencies.
- <high> Sets the amplification value of the high frequencies.
- <ON> Turns the effect on or off.

3.11 Scenario



1.14 Scenario Ensemble interface

3.11.1 Introduction

Scenario is a complete live-performance environment with realtime timestretching, performance-oriented effects, and memory and instant recall of thousands of scenes. Have you ever played live with a computer and been frustrated that you don't have enough control over the audio? Or - at the oppositve end of the spectrum - have you ever been performing and been overwhelmed with the possibilities? Scenario solves both of these seemingly opposing problems in an elegant and ingenious way.

First, let's take a look at what Scenario is, and then we'll see why it's an ideal live-performance tool. Scenario consists of a Loop Engine that contains four identical time-stretching loop players that fits each loop to the system tempo. Each loop player lets you perform in realtime with the loop length, loop start, pitch, and animated filter. A powerful performance-oriented effects block is also included, which lets you do things never possible before since the Scene Effects and Loop Engine are controlled by the same sample-accurate system clock. For instance, you can reshuffle and even reverse the music, rhythmically gate the sound exactly on the beat, create super-tight loops, and more. The Miniseq instrument can shuffle and reorder each loop. Just draw a pattern into Miniseq, then activate the shuffle function by clicking the small button in the Loop Engine at the bottom of each loop player.

And now, the kicker: All of Scenario's settings can be stored into a "scene". You can store more than sixteen thousand scenes in the built-in Scene Memory. Scenes can be recalled sequentially for "one-touch" performance, but you still have full control over all parameters within each scene. That's what makes Scenario so revolutionary - it gives you the power to lay out an entire live set in advance so you're assured that things go smoothly. Not only you determine when to advance to the next scene (this would be quite boring!), but you also have complete control over every scene. You can adjust the pitch of the loops, the loop length, the filtering, and the effects - every single parameter on the screen. When it's time, switch to the next scene, and - Bam! - it comes in perfectly on the beat, every knob perfectly recalled, including the effects.

3.11.2 Quick Start

For an easy tour of what Scenario can do, take a look at the demo live-set that is included in it. Start the system clock from the toolbar - you should hear some music now - and then after a few bars, press the Next button on the Scene Memory instrument on the bottom. You just advanced to the next scene. Keep on pressing Next every few bars - notice that every single parameter of Scenario is updated, including all effect settings. Feel free to play with any of the knobs to see how they influence the sound. Notice that each effect must be first enabled in order to hear it simply click on each effect's Enable button. You can only hear a cell when it's active - to activate or mute a cell, simply click on the button below that cell's level meters.

3.11.3 Structure and Signal Flow

The Scene Loop Engine makes all the noise. It consists of four identical loop players and a handy bar/beat counter so you can always keep track of where you are without having to count: The three marching bars on top denote the number of 4-bars, bars, and beats. Every sixteen bars it's reset to zero. The summed output of all four loopers goes through the Scene Effects, and then to the output.

Let's take a look at an individual looper in the Scene Loop Engine. Each looper gives precise control over the loop length, pitch, and filter of the loop. The tempo of the loopers are all synchronized to the master system clock, whose tempo is settable in the toolbar. The audio will stretch or compress in realtime to fit at the given tempo. The time stretching algorithm is very high tuned for rhythmical material. SampleSel selects which loop to play. Each looper can hold a Map of 128 different loops, and SampleSel simply selects the desired loops. See the section below, "Loading Samples," for more information on how to easily load your sounds into "Scenario." Offset controls the sample start time, in sixteenth notes - where the sample will start from at the downbeat. LoopLng sets the loop length, also in sixteenth notes. Pitch lets you modify the pitch of the sample without changing its time. Wdth, Filter, Reso, rnd, and Bps all control the looper's internally-animated filter. The filter is a bandpass consisting of two two-pole filters (a high-pass filter that cuts the lows, and a low-pass filter that cuts the highs). The width between the two filters is controlled by Wdth, and the combined resonance of both filters is set by Reso. Filter sets the center frequency of the two filters, and Bps bypasses the filter completely. Rnd controls how animated the filter is. If set to zero, then the filter will stay at the frequency set by Filter. But if Rnd is high, then the filter will slowly move around to create movement. Fit changes the pitch of the loop to "fit" it to the current tempo, just like a turntable changes the pitch when it slows or speeds a record up. Finally, two more controls: Gain controls the volume of the looper, and the unlabelled button underneath the level meters activates (on) or mutes (off) the track. The small button underneath this activation button activates the retrigger sequencer (Miniseq).

The output of the Scene Loop Engine gets fed into the Scene Effects block. Scene Effects consists of four effects (Loop, Slicemanipulator, InfinityFFB, and Gate), routed serially, and a master two-channel crossfader. The first effect is a looper - it simply loops the incoming audio, but it does it very precisely. The Loop knob controls how long the loop will be (in bars), and the Loop button activates and deactivates the loop. Slicemanipulator is an extremely unusual, yet performance-friendly effect. It divides the audio into "slices" then lets you rearrange and even reverse the audio in time. Split controls how many slices there will be per bar. Shift A

moves even slices forward in time (in sixteenth notes), while Shift B moves odd slices forward. Rev A reverses the even slices, while Rev B reverses the odd slices. Enable turns the effect on or off.

InfinityFFB is a delay effect with a finely-tuned feedback path. This means that you can freely perform with the Delay, Feedback, and Filter knobs without risk of overloading or running into distortion. Delay sets the delay time in sixteenth notes, while Smth controls how smooth changes in delay time will be. Fback sets the amount of feedback within the delay, while Filter controls the filter frequency of a filter built into the delay feedback path. Wdth and Reso also control that filter - the structure of this filter is also a dual-two pole filter like the filter for each looper channel. Enable turns the effect on or off. The last effect, Gate, rhythmically gates the audio with the frequency set by Freq (in sixteenth notes). Offs sets the latency of the gate - a higher offset value lets more audio through before the gate kicks in. Hold controls how long the gate will be open for - at small values hardly any audio will pass through. The usual Enable button turns the effect on or off. Finally, we have a two band crossfader as a master dry/wet control. We have independent control over the high and low frequency ranges. Low controls the mix of the low frequencies, and High controls the highs. When the crossfader is fully up, the effect is active for that frequency range.

The third instrument is the Miniseq, where you can draw in a retrigger sequencer for the Loop players. All loop players share the same sequence, but the retrigger sequencer can be turned on and off for each Loop player independently. The small bottom-most button underneath each Loop player activates the retrigger sequencer. The bottom-central Hold and Smooth knob adjust the hold time and smoothness (envelope time) of the retrigger sequencer for all loops.

The fourth instrument in Scenario is the Scene Memory, the brain that doesn't make any sound. This brain stores all scenes into a giant table. Don't be frightened by the complex-looking table - it's there just to show if a scene has information in it or not. The current scene is shown in two ways: by the read-only PartN and SceneN numerical displays to the bottom left, and by the two draggable rectangles above the table. You can change the scene in two ways: By pressing Prev/Next to go the previous or next scene, respectively, or by dragging on one of the two rectangles above the table to go directly to a specific scene. There are two settings to adjust the size of the table: Parts and Scenes. The total number of scenes is Part * Scenes - each part can contain from 16 to 128 scenes. Since there can also be from 16 to 128 parts, you can adjust the size of the scene memory from 128 (16 * 16) to 16,384 (128 * 128). Generally, depending on the style of music, 128 scenes should be sufficient for a 30 to 60 minute live set. Working with high numbers of scenes has the disadvantage that the draggable part and scene-selector rectangles (above the scene memory table) become very small. The Write

button (all the way on the right) does exactly what you'd expect - the current positions of every parameter is magically written into the table at the current scene position. The Flow button is an important feature in organizing your scenes - when Flow is on, anytime you go to a new scene (either with the Prev/Next buttons or with the draggable scene and part selectors above the table), then the new scene will be immediately active. This is probably what you want for live performance, but Flow can be turned off for when you're composing your set. With flow off, when you go to a new scene, it does not load. This is very useful for copying and pasting scenes from one location to another.

3.11.4 Loading Samples

The most convenient way to load samples into Scenario is with the Browser. Simply double click on the visual display of the sample to enter the sample map editor. If you want to clear the map to start fresh, simply double twice on the delete button. Using drag and drop, just load your samples into the sample map editor. You probably don't want to use the graphical keyboard sample map editor unless you're very careful to align one sample per key. The text-based sample map editor will take care of this automatically. Please see your user's manual for more helpful information about loading samples and using the sample-map editor.

Because of Scenario's advanced time-stretching algorithm, all samples must first be analyzed before they can be used. When you drag the samples into the sample-map editor, it will first analyze them for you. You will then be asked if you want to save the analysis data into the samples

- it's recommended to do so, so you don't have to analyze them again the next time the live set is loaded.

3.11.5 Tips and Tricks

There are many tricks you can use Scenario for. No one every said you had to use it live, either. You can also use it in the studio to create complete tracks with.

Filter-splitting - create one track out of four. You can take advantage of Scenario's filters to combine several tracks into one. Like the bass drum from one track and the synth from another and the high hat from a third? Use Scenario to create track mutations. Simply load a different loop into each of Scenario's loop players in the Loop Engine, and tweak the filter settings to focus on a different frequency range in each loop.

Make good quality loops. Loops have to be cut exactly to 1, 2, 4, 8, 16, etc bars in order for the beat looper to play them correctly. The loops can be at any tempo, however, and Scenario will automatically fit all loops to the same tempo. All common sound editing programs let you bounce audio with exact bar sizes, so check your manual for detailed information.

Make good scenes, then freely improvise. For an effective live performance, it often pays to spend a lot of time making good scenes, and having a good flow between the scenes. You can copy and paste scenes in the Scene Memory simply by activating a scene by going to that scene with Flow enabled, then by turning off Flow and going to the position you want to write the scene to. Press Write, and you just copied and pasted a scene from one part to another. When performing, you can be assured that if any live improvisation gets out of hand, you can just go to the next scene and everything will be on track again.

Use the two-band crossfader in the Scene Effects. The two-band crossfader is a well-kept secret for music played on a powerful PA system. By only effecting the high frequencies, the bass stays clear and doesn't get muddy.

MIDI control. Any of the Scenario's parameters can be MIDI controlled. Due to its complex internal structure (using internal OSC communication between the three instruments), you can't use MIDI learn on the parameters, or they will not be remembered by the Scene Memory any more. If your MIDI controller is programmable, such as a Peavey PC1600 (www.peavey.com) or Bitsream (www.wave-idea.com), then it's better to program your MIDI controller instead of using MIDI learn.

Saving live sets. Each live set should be saved as its own ensemble so it can be recalled with all samples and scene memories intact.

3.12 Sinebeats 2



4.13 The Sinebeats 2 user interface

3.12.1 Introduction

The REAKTOR library classic Sinebeats has undergone an overhaul for REAKTOR 5. Sinebeats is a beatbox based on three sine oscillators and a noise generator. Its synthetic nature in combination with the flexible effects section has made Sinebeats a classic for electronic sequence production. Each of the four instruments features a sequencer and individual sound parameters including distortion and filter. Two flexible filters and two delays which are fed via a send/return feature in the mixer add even more motion to the generated beats.

In its new incarnation, Sinebeats got an enhanced mixer with the possibility of routing the individual sound units to single outputs, a two-band equalizer, and a simple compressor for the sum. The sequencers got updated with individual looping, individual clock settings, and the

possibility to introduce rolls for every step. You can also record the pitch information via MIDI input. Modulation of sound parameters has undergone a major overhaul, the sine instruments now have multimode filters, and all the instruments now sport an individual overdrive section and an equalizer. A valuable addition to the effects section are the two modulation sequencers that provide dynamic effects sequencing. Also, there is a new snapshot system that enables you to trigger complete snapshots including the sequencer tracks via MIDI note triggers and in sync with the global tempo.

3.12.2 Sequencer

Each of the four instruments is equipped with its own 16-step sequencer with 2 tracks. The first contains the triggers for the sound units. The second track sends modulation data which can modulate different sound parameters in the instrument. A great addition to your sequencing options is the roll/slide track of the sequencer. You can define 2 different rolls per sound unit that can be assigned to individual steps and you also can introduce pitch slides between steps. The sequencer functionality also allows various direction modes, individual tempo settings and individual loop control. Pitch recording via MIDI input has also been added. If you want to hear Sinebeats without triggering the snapshots via MIDI notes (see [Snapshot system]), you have to switch off [Velocity] in the [Master] section.

Control	Function
Sound units	Switch the view between the [Noise] percussion and the four [Sine] synthesizers and their corresponding sequencers.
Rec	Activates velocity/modulation recording Hit [Rec] and let the sequencer run. Then play notes on your keyboard to write triggers and velocity values for the instruments. For the [Sine] units the note pitch information will be written into the [Pitch dials]. Recording does not delete existing events. They remain untouched as long as no new data is coming in via MIDI.
Run	Starts or stops the sequencer.
Pitch dials	Dial in the desired note pitch per step for the [Sine] units. Ypu can also record the values via midi note input (see [Rec]). These are not available for the [Noise] unit.
Init	Completely initializes the displayed unit's sequencer. This includes deletion of the modulation and velocity tracks, and, in the case of the [Sine] sequencers, resets the Pitch dials.

Control	Function
Direction	Choose between four different direction modes: forwards (->), backwards (<-), and two ping-pong modes (<->, inverted: >-<).
Tempo	Choose an individual tempo for the currently displayed unit sequencer. These are clock division settings that always keep the sequencer in sync with the global tempo.
Loop bar	The bar above the sequencer grid is for setting up a loop region for the currently displayed sequencer. Drag the start or the end to change length and drag the bar to change position.
Random	Randomizes the modulation- or velocity track of the displayed unit sequencer, respectively.
Track selector	Switches the view between the modulation- and the velocity track.
Event grid	Click into the grid and drag up or down to create modulation events or velocity triggers. Right-click (ctrl-click if you're on a Mac) to delete events.
Roll settings	Assign the roll speeds for three freely assignable roll mode colours (yellow, blue and red). You can assign roll overdrive values between 2-times and 16-times. In case of the [Sine] units, the red roll is used for pitch slides only.
Roll and slide modes	The bar below the sequencer grid is for defining roll modes for the individual steps. Left-click to create a roll and repeat the click to change the mode. Right-click (ctrl-click if you're on a Mac) to delete the roll. You can define three roll modes under [Roll settings]. For the [Sine] unit sthe red mode always stands for a pitch slide, with the [Nois] instrument you can freely define it as a third roll mode.

3.12.3 Noise Synthesizer

The noise unit features a simple envelope for controlling the volume. You can stack the different outputs of the multi-mode filter and you can also adjust the parameters cutoff, resonance and envelope modulation intensity. Release, cutoff and resonance can be modulated by the second sequencer track. An overdrive / bit reduction effect and a small equalizer are also at your disposal.

Control	Function
Amp-release	Release time of the amplitude envelope.
Release mod	Bipolar (-/+) amount for the modulation track output of the sequencer, altering release time. This can be initialized with the [init mod] button.
Init mod	Initializes all [MOD] controls of the [Noise] unit.

Control	Function
Filter stack	Switches for the output of six different filters. The outputs can be stacked. Since the addition of six signals can produce clipping, the overall level of the instrument will be divided by the number of activated filter signals.
Drive switch	The [Drive] section reduces bit depth and sample rate of the signal, and includes a saturator. Switch on or off with the power button.
Drive	Controls overdrive intensity.
Bit	Controls bit depth reduction.
EQ switch	Switches the 1-band equalizer on or off.
Freq	1-band EQ frequency.
Amt	Bipolar EQ cut / boost. -/+ 24 db.
Cutoff	Filter frequency of the noise filter. Shown in pitch values.
Cutoff mod	Bipolar (-/+) amount for the modulation track output of the sequencer, altering the cutoff frequency. Can be initialized with the [init mod] button.
Reso	Resonance of the noise filter.
Reso mod	Bipolar (-/+) amount for the modulation track output of the sequencer, altering the resonance. Can be initialized with the [init mod] button.

3.12.4 Sine Synthesizers

The three sine instruments are structured identically. Each of them features a release parameter for the decay time, a pitch parameter and a simple pitch envelope with an intensity- and a release parameter. Here you can modulate the intensity of the pitch envelope and the decay time. Pitch can also be slurred with variable glide. As with the noise instrument, you can use an overdrive / bit reduction effect and a small equalizer. You also get a multi-mode filter with variable cutoff and resonance. Both parameters can be targets of the modulation track of the sequencer.

Control	Function
Amp-release	Release time of the amplitude envelope.
Release mod	Bipolar (-/+) amount for the modulation track output of the sequencer, altering release time. Can be initialized with the [init mod] button.

Control	Function
Init mod	Initializes all mode controls of the [Sine] unit.
Glide	Controls glide time. This only works when the sequencer reaches a red [Roll] step. Refer to the description of the sequencer.
Octave	Master octave of the Sine unit.
Tune	Pitch for the sine oscillator of the Sine unit.
Fine / Integer	Toggles pitch control of the sine oscillator between fine and integer mode. In fine mode the range of the dial is +/- 100 cents. These get added to or substracted from the integer value chosen in integer mode.
Penv	Controls the amount of a percussive pitch envelope applied to the sine oscillator.
Penv mod	Bipolar (-/+) amount for the modulation track output of the sequencer, altering pitch envelope amount. Can be initialized with the [init mod] button.
Prel	Controls the release time of the pitch envelope.
Prel mod	Bipolar (-/+) amount for the modulation track output of the sequencer, altering pitch envelope release time. Can be initialized with the [init mod] button.
Drive switch	The [Drive] section reduces bit depth and sample rate of the signal. Switch on or off with the power button.
Drive	Controls overdrive intensity.
Bit	Controls bit depth of the sound. Reduce to introduce harsh aliasing.
Eq switch	Switches the 1-band equalizer on or off.
Freq	Control of EQ frequency.
Amt	Bipolar EQ cut / boost. -/+ 24 db.
FilterMode	Click multiple times to switch through the available filter modes. You can choose between low-pass high-pass and band-pass.
Cutoff	Filter frequency of the noise filter. Depicted in pitch values.
Cutoff mod	Bipolar (-/+) amount for the modulation track output of the sequencer altering the cutoff frequency. Can be initialized with the [init mod] button.
Reso	Resonance of the multi-mode filter.
Reso mod	Bipolar (-/+) amount for the modulation track output of the sequencer, altering the resonance. Can be initialized with the [init mod] button.

3.12.5 FX 1 & 2

You get 2 effects units that receive their input from the sends of the [Mixer]. The effects units are identical and offer a stereo delay with an integrated resonant multimode filter, feedback and a return level dial. The filter cutoff can be modulated by an integrated three-waveform, tempo-synced LFO. Both effects units have a small step-sequencer including loop- and tempo control that allow for bipolar modulation of the filter parameters, of [Return level], and [Feedback].

Control	Function
DLY / R	Delay Time for the right channel. Units are beats per echo.
DLY / L	Delay Time for the left channel. Units are beats per echo.
CUT	Cutoff frequency of the multimode filter.
CUT MOD	Switches on modulation of the cutoff frequency by the effects unit's modulation sequencer.
RES	Resonance control of the multimode filter.
RES MOD	Switches on modulation of the resonance by the effects unit's modulation sequencer.
Filter mode	Click multiple times to switch through the available filter modes. You can choose between low-pass high-pass and band-pass. Right-click (ctrl-click if you're on a Mac) to choose the low-pass mode directly.
TEMPO	Dial in the tempo of the LFO. Units are fractions of one bar in quarter notes.
TEMPO MOD	Switches on modulation of the LFO tempo by the effects unit's modulation sequencer.
AMT	Control of LFO modulation depth.
AMT MOD	Switches on modulation of the LFO depth by the effects unit's modulation sequencer.
LFO waveform	Choose between sine, pulse and triangle for the LFO waveform.
Modulation sequencer	Click into the sequencer and drag the mouse up and down to change the value of the bipolar sequencer steps. The sequencer output can be routed to the filter [Cutoff] frequency, filter [Resonance], LFO [Tempo], LFO [Amount], and [Feedback].
RANDOM	Click to randomize the steps of the modulation sequencer.
Loop bar	With the loop bar you can define a region in the sequencer that gets repeated. Drag the start or the end to change length and drag the bar to change position.

Control	Function
DIR	Choose between four different direction modes: forwards (->), backwards (<-), and two ping-pong modes (<->, inverted: >-<).
TEMPO	Via this menu you can choose a tempo for the effects unit's modulation sequencer. These are clock division settings that keep the sequencer always in sync with the global tempo.

3.12.6 Mixer

The four-channel mixer provides control over [Pan], [Volume], and two effects [sends]. It has a routing system, allowing you to send the different channels either to the master stereo bus or into up to 4 individual stereo busses. This output routing system has to be activated in the [Master] section. The four stereo channels are identical in function and carry the signals of the [Noise] synthesizer and the three [Sine] synthesizers.

Control	Function
Power	Switches the respective channel on or off. Use to mute single or multiple sound units.
PAN	Dial in the position of the respective sound unit in the stereo field.
VOL	Volume of the respective sound unit.
FX 1	Send level to effects unit 1.
FX 2	Send level to effects unit 2.
Output busses	Click multiple times to choose the 4 available stereo busses. Works only if [use single outs] is activated in the [Master] section. Also, this is sensible only if your sound hardware has multiple outputs, or if you route the individual outs of Sinebeats into further REAKTOR instruments. Right-click (ctrl-click if you're on a Mac) to reset to outputs 1 / 2.

3.12.7 EQ and Compressor

With Sinebeats 2 you also get a little effects section that works on the sum. In terms of signal flow it sits between the [Mixer] and the [Master] section. A two-band shelving equalizer and a simple compressor help to spice up the sum of your Sinebeats tracks.

Section	Control	Function
EQ	EQ Power	Switches the sum-EQ on or off.
	F-LOW	Frequency of the low-band shelving EQ. Units are Hz.

Section	Control	Function
	LOW AMT	Cut or boost for the low-band shelving EQ (+/-20 db).
	F-HIGH	Frequency of the high-band shelving EQ. Units are Hz.
	HIGH-AMT	Cut or boost for the low-band shelving EQ (+/-20 db).
Compressor	Compressor power	Switches the sum compressor on or off.
	comp	Dial in the threshold and ratio of compression. These two parameters are combined into one.
	speed	Control for the release time of the compressor.
	soft	If on, the compressor works in soft-knee mode, meaning that the ratio increases gradually to the selected [comp] level. If off, the compression is applied to only to signals above the threshold.

3.12.8 Master

The Master section provides control over master volume, a switch for velocity sensitivity of triggering the sequences via MIDI input, and a toggle for activation of the multiple output routing system.

Control	Function
Master	Controls the master volume of the patch.
Velocity	Toggles velocity sensitivity for triggering of snapshots via MIDI. If you want to hear Sinebeats' output without MIDI triggering switch this control and the [Snap via key] toggle off.
Use single outs	Activates the three additional stereo outputs. You can route sound into them with the [Mixer].

3.12.9 Snapshot System

The snapshot system is a new feature of Sinebeats 2, enabling you to store and recall snapshots from within the patch. The most intriguing feature of this module is the [snap via key] function. When this is active you can trigger complete stored sequences including all sound units via incoming MIDI note data. This happens glitch-free and in real-time. Use it to trigger Sinebeats sequences in a live situation via a MIDI controller. You can also trigger sequences from another sequencer.

Control	Function
On	This is the bypass switch for the snapshot calling/storing system.
Snap via key	Enables snapshot calling via pitch input.
Start note	Dial in the note that mapping of snapshots across the keyboard starts with.
Learn start note	The first MIDI note value received after pressing this button will be used as new [Start note].
Start snap	Sets the snapshot number that is recalled when the MIDI note adjusted by [Root Note] is received; [Snap Via Key] has to be actiavted.
Learn start snap	The first snapshot recalled after pressing this button will be used as new [Start snap].
Key-sync	Incoming MIDI notes are quantized at the given resolution relative to the global tempo.
Key sync on / off	Enables / disables key-sync of snapshot calling. This quantizes the start of the next triggered snapshot to a metric value between 1/16th and whole notes.
SnapShot	Choose a snapshot number to store to.
Store	Stores the current settings of the complete ensemble to the current snapshot number (see [Snap]). If [Store+1] is activated, the subsequent snapshot number will be used to store the data. Any data stored there before will be overwritten. Therefore, one should start a complete new bank of snapshots when working on a project.
Store+1	If activated, upon pressing [Store] the settings of the complete ensemble will not be saved to the current snapshot as displayed by [Snap] but to the subsequent one.

3.13 Splitter

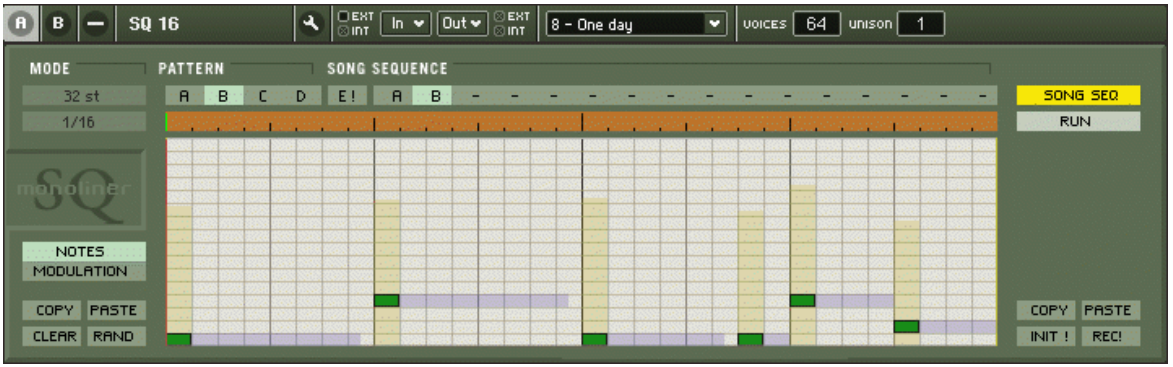


4.18 The Splitter user interface

3.13.1 Introduction

The Splitter is a sonically flexible sequenced sample-player. Geared towards granular beat production, it can also be used for melodies or padwork. The main idea behind this sequencer / sample-player combo are the 16 sample slots. You can assign different fragments of the selected sample with individual settings for all parameters to the different slots above the waveform display. You can also assign individual MIDI notes.

3.13.2 Sequencer



4.19 Sequencer user interface

The monophonic sequencer delivers classic step-sequencing in a very useable package. It offers 16 parallel tracks where each track represents one of Splitter's sample slots (see also [Splitter] below). The leftmost sample slot corresponds to the bottom track, the rightmost sample slot corresponds to the top track. A note placed on the track triggers the respective slot; the vertical bar crossing the note represents its velocity; the horizontal bar, its length.

By clicking with the left mouse button on an existing note one can alter its position on the horizontal time axis or shift it to a different track vertically. Clicking an empty area with the left mouse button creates a new note, double-clicking deletes an existing one. A note's velocity (vertical movements) and length (horizontal movements) can be adjusted by clicking with the right mouse button.

Beside the notes sequencer there is an additional modulation sequencer. Here, a value can be adjusted for each step of the timeline; this value can be used as modulation source within the [Splitter] instrument.

Finally, a complete song sequencer is also implemented: at the top of the instrument's panel the currently displayed and edited pattern can be selected (A – D). To the right of the pattern select a pattern sequence can be programmed to automatically switch the patterns during playback.

Control	Function
Zoom Level	Selects the number of steps shown within the main [Pattern Edit] display.
Quantization	Selects the rhythmical unit at which the steps of the sequencer are interpreted. The units are always bound to the global MIDI tempo.
Pattern Select	Selects one of the four patterns A/B/C/D to be shown within the main [Pattern Edit] display. If [Song Seq] is off, this also selects the pattern to be played (see also [Loop Bar]).
Song Edit	Switches the edit mode of the song sequence on or off (see also [Sequence Edit]).
Sequence Edit	If [Song Edit] is on, one can program a sequence of patterns by dragging the pattern symbols with the left mouse up and down. The complete song sequence is repeated continuously. Empty steps within the sequence are omitted.
Song Seq	Switches the song sequencer on or off.
Run	Switches the complete sequencer on or off. If on, the sequencer synchronizes itself to the global MIDI clock. Thus, the MIDI clock has to be running to make this sequencer work.
Pattern Copy	Copies the currently selected pattern (see [Pattern Select]) into the internal clipboard.
Pattern Paste	Pastes the contents of the internal clipboard into the currently selected pattern (see [Pattern Select]).
Pattern Init	Deletes all notes of the currently selected pattern (see [Pattern Select]) and resets the modulation sequence. As this cannot be undone, the button must be double-clicked to delete the content.
Pattern Rec	Switches recording of MIDI note input to the sequencer on or off.
Pattern Edit	Displays the currently selected monophonic pattern (see [Pattern Select]). Depending on the [Notes/Modulation Switch] sixteen note tracks or a modulation curve is displayed. The pattern can be edited with the mouse or recorded via MIDI (see [Pattern Rec]).

Control	Function
Notes/Modulation Switch	Switches between displaying the note tracks or the modulation curve of the currently selected pattern.
Loop Bar	Selects the range of the current pattern that is played back. This range is stored for each pattern independently, so each can have a different length. Use the left mouse button to position the loop. The length can be adjusted with the right mouse button (Mac: ctrl-click).
Loop Copy	Copies the currently active loop into the internal clipboard.
Loop Paste	Pastes the contents of the internal clipboard into the currently active loop.
Loop Clear	Initializes the note tracks and the modulation curve within the range of the currently active loop.
Loop Rand	Sets the note tracks and the modulation curve within the range of the currently active loop to random values.

3.13.3 Splitter

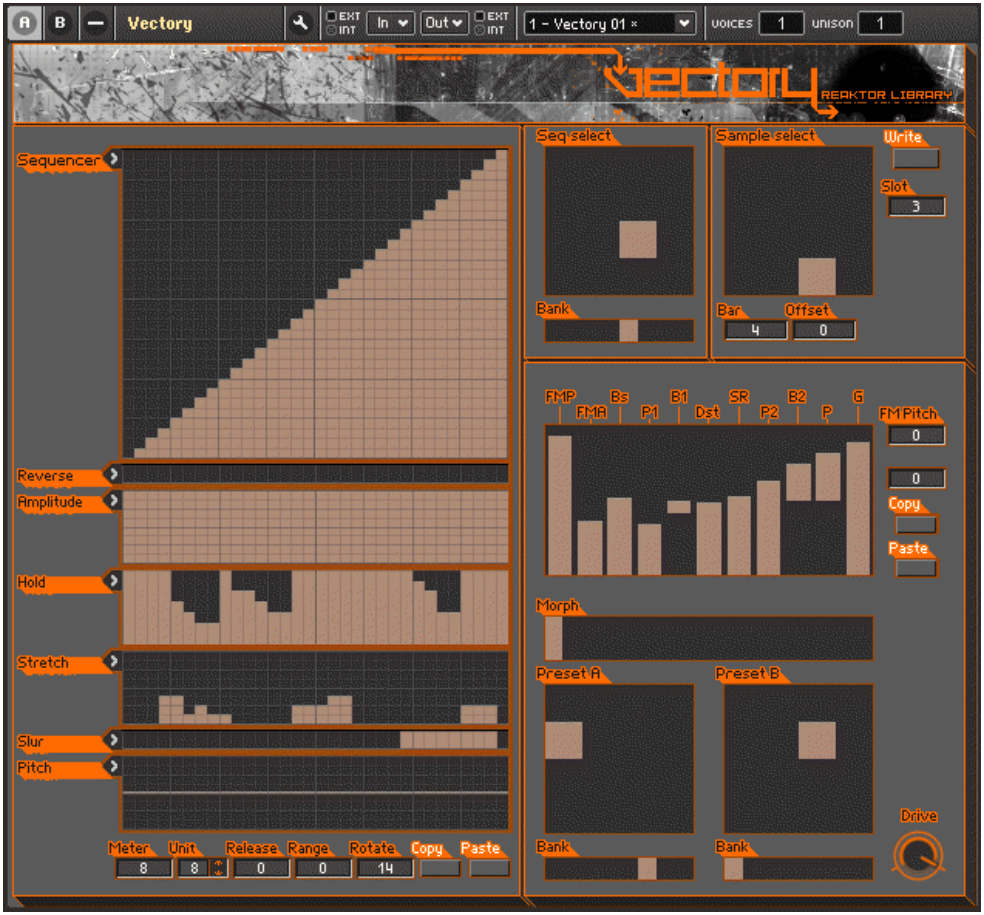
The granular sample-player enables you to load samples and trigger defined parts of them with individually stored settings for envelope, pitch, speed and grain length via the built-in sequencer or MIDI input. You also have a tempo-syncable LFO and some control over routing modulations and quantizing sample playback parameters.

Section	Control	Function
Slots	Edit	If active, you can change the parameters of the selected slot (see [LFO], [Modulation], [Shape], [Envelope] and [Output]). Also, an incoming MIDI note value is assigned to the selected slot. You can also assign notes with the mouse (see [Slots]).
	Slots	If [Edit] is active, you can change the slot's parameters and change the assigned MIDI note by clicking into the slot and dragging up or down. You can also assign MIDI notes via MIDI in (see [Edit])
	Copy	Copies all parameters of the selected slot.
	Paste	Pastes the slot clipboard into the selected slot, overwriting all the slot's parameters.
Samples	Waveform Display	Right-click on the title bar of the waveform display to access the sample map menu (ctrl-click for Mac users). When in Edit mode, you can set the red sample-start line by clicking on the waveform.

Section	Control	Function
Control	Speed Mode	These settings affect the [Speed] parameter of all sample slots. In free mode the speed can be dialed in freely; in grid mode it is quantized to 16th note values.
	Grain Mode	These settings affect the [Grain] parameter of all sample slots. In free mode the grain length can be dialed in freely; in grid mode it is quantized to 16th note values; in note mode the grain length is quantized to steps corresponding to the 127 MIDI notes.
	Length	Shows the length (in 16ths) of the sample currently selected from the sample map.
	Pitch	Shows the sample's pitch deviation from original pitch when played at the current tempo.
	Speed	Shows the sample's pitch deviation when played at the current tempo. 1 represents original speed.
LFO	Speed	Adjusts the frequency of the Low Frequency Oscillator.
	snc	Activates synchronization of the LFO to the song tempo.
	Waveform	With this menu you can choose between six different LFO waveforms (sine, sawtooth, reverse sawtooth, pulse, and two random modes).
Modulations	LFO	Routes the LFO to different parameters. Choose which parameter of the sample-player gets modulated by the LFO's oscillations. Target parameters are [Offset], [Pitch], [Speed], and [Grain].
	LFO Dpth	This control determines how much the LFO affects the chosen parameter.
	Seq	Routes the modulation track of the sequencer to different parameters. Choose which parameter of the sample-player is modulated by the Modulation track. Target parameters are [Offset], [Pitch], [Speed], and [Grain].
	Seq Dpth	This parameter determines how much the modulation track of the sequencer affects the target parameter.
Shape	Sample start point (waveform display)	You set the start point of the sample fragment by clicking into the waveform display. It is indicated by a red line.
	Speed Switch	Binds the sample's playback speed to the song tempo. The effect is similar to speeding up or slowing down a record player.
	Pitch Switch	Binds the sample's playback pitch to the song tempo. The effect is similar to speeding up or slowing down a record player.

Section	Control	Function
	Rev	When active, the sample-fragment is played in reverse, starting from the set start point.
	OfsQntz	When active, the [Offset] parameter works quantized to 16ths.
	Select	Selects a sample from the sample map.
	-offs+	This dial offsets the sample start of the currently edited sample-fragment.
	Pitch	Transposes the pitch of sample playback independently from speed. 0 is original pitch.
	-fine+	Fine-tunes the sample-fragment transposition.
	Speed Knob	Sets the speed of sample playback independently from pitch. 1 is original tempo, 2 is double speed. This interacts with the [Speed Factor] control.
	Speed Factor	With this dial you can multiply the [Speed] control value by a factor. Use it to change the range of the [Speed] knob.
	Grain	Control of grain length. This interacts with the [Grain Factor] control.
	Grain Factor	With this dial you can multiply the [Grain] control value by a factor. Use it to change the range of the [Grain] knob.
Envelope	Attack	Sets the time that passes until the amplitude envelope reaches its peak after triggering.
	Hold	Sets the time the envelope stays at its peak value before it moves to the release phase.
	Release	Sets the time that passes until the amplitude envelope is completely faded out after the hold period has ended.
	Amp	Sets the maximum level the envelope will reach. This gets modulated by velocity if [VelSns] is on.
Out	Gain	Sets Splitter's master volume.
	Mute	Mutes Splitter's audio output.

3.14 Vectory



4.20 The Vectory user interface

3.14.1 Introduction

Vectory is an aggressive sample destruction unit. It consists of a sampler (on the left side) with vast re-arranging capacities whose signal is fed into a grain multi-effect (on the right side) that re-synthesizes the sound.

This structure is optimized for live use, with low-level REAKTOR Core DSP. Complete settings for sample loops, re-arrangements, and grain effects can be recalled by moving the markers within the large square selection displays – changes occur instantly and with no audio drop-out. The effect unit even offers morphing between two settings.

The sample is loaded in a sub-instrument of Vectory called **Sample Loader**. Press Ctrl+2 to open its panel; to return to the main Vectory display press Ctrl+1. Only one sample can be loaded at a time. However, this sample can be quite long and contain several discrete loops.

The second panel set also contains a further sub-instrument labeled **Controllers**. This is designed to automate Vectory's parameters via MIDI / VST.

3.14.2 Sample

This section at the top-right of the panel selects the sample material from the Sample Loader. By using the large square markers one of sixteen slots can be selected; each slot contains the data of the sample loop's beginning, measured from the start of the sample file in bars and sixteenth.

Control	Function
Sample Selection Display	Selects the active sample loop slot; each slot stores independent values for [Bar] and [Offset]. Those two parameters control the starting point within the loaded sample; thus, they define the sample material being played, which is then subject to re-arrangement by the [Sequencer] section. The length of the loop within the sample is controlled by [Sequencer][Meter] and [Sequencer][Unit].
Write	Stores the current values of [Bar] and [Offset] into the current slot.
Slot	Displays the number of the active slot in the [Selection Display].

Control	Function
Bar	Sets the starting point of the sample readout. This control adjusts the number of bars to be skipped in the sample file. (See also [Sample Loader][Bar] and [Sequencer][Position].)
Offset	Sets the starting point of the sample readout. This control adjusts the number of sixteenth steps that are added to the number of bars set by the [Bar] control. (See also [Sample Loader][Tempo] and [Sequencer][Position].)

3.14.3 Sequencer

The sequencer contains two sections: The [Seq Select] part chooses one of several sequencer settings; each setting is defined within the main [Sequencer] part that fills the left side of the instrument's panel. The sequencer pattern is mapped onto the sample material selected in the [Sample] section.

Control	Function
Sequence Selection Display	Selects the active sequencer pattern. There are sixteen slots in each bank. (See also [Bank].)
Bank	Selects the bank from which the [Selection Display] loads the sequencer pattern. There are eight banks available.
Position	Defines the rearrangement pattern. The sequence is read in sixteenth steps from left to right; the vertical axis sets the offset from the sample readout starting point for each step in sixteenths; e.g. a scale from the bottom-left to the top-right represents the normal sample readout while a scale from the bottom-right to the top-left results in inverse readout: first the last sixteenth of the sample, then the one before the last etc. The starting point of the sample readout is controlled within the [Sample] section.
Reverse	Defines whether the sixteenth note selected by the [Position] pattern is played from the end to its beginning or in its normal playback direction.
Amplitude	Adjusts the amplitude for each sequencer step.
Hold	Adjusts the hold time for each sequencer step. (See also [Release].)

Control	Function
Stretch	Lengthens the sample at this sequencer step. The higher the value, the more it is stretched – the first square represents a ratio of 2:1, the next one is 3:1, etc. The sample is stretched by a grain re-synthesis algorithm; therefore, the grain pitch and grain frequency parameters of the [Grain Effect][Parameter Display] will greatly affect the sound of stretched steps. As the sequence moves on without being influenced by the stretch, parts of the stretched sample that do not fit in the step are cut (at a ratio of 2:1 the second half will be cut, etc.) See also [Slur].
Slur	Ties stretches over consecutive sequencer steps. If slur is off, the stretch will be re-triggered at each sequencer step; if it is switched on, the stretch will be continued. This also affects the [Reverse] function.
Pitch	Adjusts the pitch shift for each sequencer step. The values set here are relative ones; the absolute range of shifting is controlled by [Range].
Meter	Controls the loop length in steps; the step length is adjusted by [Unit].
Unit	Sets the rhythmic unit (fourth, eighth or sixteenth, according to the current MIDI tempo) used as step for the [Meter] control.
Release	Adjusts the release time after each sequencer step's hold period. (See also [Hold].)
Range	Sets the absolute pitch range available for the sequencer steps. To actually adjust the pitch in each step, use the [Pitch] pattern, which exerts relative control over pitch. With higher [Range] values, identical [Pitch] patterns produce more drastic pitch shifts.
Rotate	Sets an offset to the sequencer readout.
Copy	Copies the current sequencer pattern into a buffer that can be read out by pushing the [Paste] button. A complete pattern can easily be duplicated or moved by selecting another slot with [Sequencer Select][Selection Display] and [Sequencer Select][Bank] before pasting the buffer.
Paste	Pastes the buffer's data into the current sequencer pattern, overwriting the old values. (See also [Copy].)

3.14.4 Grain Effect

This section controls the multi-effect placed after the sampler and the re-arrangement sequencer. The separation between sound generator and effect unit is only true for the panel; internally, those sections are closely interrelated. For example, both the frequency modulation

and the grain resynthesis parameters show no results within the effect unit, but inside the sampler itself; they are placed here because they impact the instrument's sound as much as the other effect parameters.

There are two slots named A and B containing two different sets of effect unit settings. The [Morph] control interpolates between both settings for smooth transitions in live use.

Control	Function
Parameter Display	Shows the currently active effect parameters. If [Morph] is set completely to the left or right – thus selecting either the A settings or the B settings and not an interpolation – the parameters can also be edited. There are eleven parameters: FM Pitch, FM Amount, Bias, Pre-Quantize EQ Frequency, Pre-Quantize EQ Amount, Distortion (overdrive saturation), Sample Rate Reduction (frequency quantization), Post-Quantize EQ Frequency, Post-Quantize EQ Amount, Grain Pitch, Grain Frequency. Their technical meaning cannot be explained here in detail; their influence on the sound, however, can easily be heard when changing the values.
Grain Random	Adjusts the amount of randomness applied to the grain synthesis. The lower the value, the more constant is the frequency at which new grains are generated.
Copy	Copies the current effect parameter setting into a buffer that can be read out with the [Paste] button. The data can easily be moved to another storage position by selecting another parameter slot with [Morph], [A/B Selection Display] and [A/B Bank] before pasting.
Paste	Copies the buffer's data into the current parameter setting, overwriting old values. (See also [Copy].)
Morph	Interpolates between the parameter settings selected with [A Selection Display] and [B Selection Display]. Move the marker completely to the left to activate and edit preset A; move it completely to the right to activate and edit preset B. Moving the marker in the space between gradually morphs from one preset to the other; during morphing no editing of the preset settings is possible.
Preset A	Selects the slot whose parameter settings are active (and editable if [Morph] is moved completely to the left). Each bank has sixteen slots. (See [A Bank].)
Bank A	Selects the bank from which the [A Selection Display] loads its data. Eight banks are available.
Preset B	Selects the slot whose parameter settings are active (and editable if [Morph] is moved completely to the right). Each bank has sixteen slots. (See [B Bank].)

Control	Function
Bank B	Selects the bank from which the [B Selection Display] loads its data. Eight banks are available.
Drive	Adjusts the amount of compression applied to the final output signal. High values represent high compressor thresholds; all audio data below this threshold will be amplified.

3.14.5 Sample Loader

The [Sample Loader] imports audio material. Only one sample can be loaded, but you can assign playback for different loops and parts.

Control	Function
BPM	Sets the tempo of the loaded sample in beats per minute. This should be done accurately – three small boxes to the right of the main BPM box allow you to set the tempo with three decimals – as this value is used to calculate the positions within the sample file (see [Sample][Bar] and [Sample][Offset]).
Start	Adjusts an offset in milliseconds at the beginning of the sample file that is skipped by all calculations concerning positions within the sample file.
Bar	Sets the number of sixteenth notes (according to the tempo adjusted in [BPM]) within one bar (see [Sample][Bar]).

3.14.6 MIDI Controller

This sub-instrument of Vectory provides various automation possibilities to control the parameters via MIDI or VST. Five MIDI continuous controllers can be selected as modulation sources [Control A] to [Control E]. Additionally, two two-dimensional sources are available as [XY1] and [XY2]; they are controlled by two MIDI Ccs, one for the horizontal movements, the other one for vertical ones. [XY2] can also be controller via the MIDI pitch. Those modulation sources can be assigned to various parameters of Vectory within the [Assignment] section of the sub-instrument.

Controls		Function
Control A .. E		Selects the MIDI CC number that is referred to as [Control A] to [Control E] respectively.
XY1	X	Selects the MIDI CC number that controls the horizontal position of the marker.

Controls		Function
	Y	Selects the MIDI CC number that controls the vertical position of the marker.
XY2	X	Selects the MIDI CC number that controls the horizontal position of the marker.
	Y	Selects the MIDI CC number that controls the vertical position of the marker.
	Note	Switches between MIDI CC mode (off) and MIDI note mode (on). In MIDI note mode, the position of the marker is controlled by the pitch of incoming MIDI events. The pitch adjusted by [Origin] selects the first position, the next pitch selects the second position etc.
	Origin	Sets the MIDI pitch that selects the first position of the marker if [Note] is on.
Assignments		The four square displays [Sample / Sequence / A / B Selection Display] of the main instrument can be controlled by all seven modulation sources; all other parameters only provide the five one-dimensional sources [Control A] to [Control B].

4 Samplers

4.1 BeatSlicer 2



4.14 Beatslicer 2 user interface

4.1.1 Introduction

BeatSlicer 2 will separate any waveform into smaller component 'slices', which can then be individually tweaked by adjusting pitch, envelope and FX settings. BeatSlicer 2 is designed primarily for drum-loop manipulation, but the extensive range of parameters offers creative possibilities with any material. For a quick start, right-click (PC) / {Ctrl}-click (Mac) on the large central window, select 'File', 'Load data into table ...' and then choose an audio loop from your sound library. The loop will be scanned and MIDI notes from C-2 (by default) will be assigned to the detected slices.

BeatSlicer 2 is designed to be programmed with MIDI controllers. To assign a MIDI controller to a parameter, use the MIDI learn function on the XY modules on the panel, e.g. the [Pitch] control. You can also assign MIDI controllers to certain other controls that use Multi Picture modules, such as the [Shape] on/off control. To do this, right click on the module and select 'show in structure'. Then assign a MIDI controller to the hidden button module beneath the Multi Picture module. Be aware that whenever BeatSlicer 2 receives an event from an assigned MIDI controller, it will write to the memory of whatever slice is currently selected. This can be a problem with certain hosts, which send extra MIDI controller data to plug-ins on initialization or when stopping/starting playback. Or it can be a problem if you accidentally move a MIDI controller yourself... The safest thing to do is take regular snapshots of your loop settings.

With the exception of the audio loop itself, all global and per-slice parameters are saved in the host plug-in edit buffer. This means that if you do not change the loaded loop, you do not need to save a new copy of the ensemble. However, as you'll most likely work with different loops in different songs, you should always use the REAKTOR autosave feature. This will create a new copy of the BeatSlicer 2 ensemble and save it with your song.

4.1.2 Global Section

The master controls are located at the top of the instrument panel. They control global settings that are applied to the complete sample loop and not to individual slices. Note that both pitch-bend and velocity can be assigned to slices individually using the modulation matrix, in which case it is probably best to set the global knobs to zero.

BeatSlicer 2 has four output channels. By default, these are used as two stereo output channels, but can be used as 4 discrete mono channels by enabling the [Mono] button.

Control	Function
Power	Switches the complete instrument on or off. Technically, this mutes the output; the instrument is still on and consumes CPU power.
Mono	Switches the mono mode on or off. If on, there are four single outs; if off, there are two stereo outputs. (See also [Out][Pan] and [Out][Out Port].)
Root	Selects the root key for the loop – i.e. the first slice will be assigned to this key, (60=Middle C, 48 C-1 etc.).
Pitchbend	Determines the amount by which the pitchbend wheel affects the pitch of the entire loop.
Velocity	Specifies the amount by which note-on velocity affects amplitude for all slices.
Gain	Controls the overall output level in decibels.
Tune	Transpose the pitch of the entire loop in semitones
Clear	Resets all settings of the current slice to their default values.
Copy	Copies all settings of the current slice to an edit buffer (see also [Paste]).
Paste	Copies all settings from the edit buffer into the current slice (see also [Copy]).

4.1.3 Loop Section

The large window displays the waveform of the current loop; it is also the place to load loop files. BeatSlicer 2 can slice loops by employing a transient detection algorithm, or by dividing the loop into equal-length sections. With either method, adjusting the Sens (sensitivity) knob will result in more or less slices. To slice the loop into equal length sections, ensure that the BPM button is active, and that the detected tempo (displayed at the bottom of the loop section) is correct. If the tempo isn't correct, even when adjusted by clicking on the + and - buttons, then the loop will not be an integer number of bars. In this case, you should use the transient detection method (deactivate the BPM button).

Click anywhere on the loop window to select (and audition) selected slices. You can either edit slices individually, or you can choose to edit all slices simultaneously by right clicking on the window. The indicator in the top-left of the Loop section displays the MIDI note assigned to the current slice.

Control	Function
Waveform Display	Shows the waveform of the current loop. To load a different loop, right-click (PC) / ctrl-click (Mac) on the upper part of the window (where the filename is displayed) and select 'File', 'Load data into table …'.
Zoom Bar	Scrolls the viewable area across the entire loop. Click the right mouse button on this scrollbar to zoom in or out.
MIDI Note	Displays the MIDI note number that is triggering the currently selected slice.
Track	If activated, the received MIDI notes not only trigger the playback of the slices, but also select them for editing.
Solo	If activated, only the currently selected slice can be triggered by MIDI notes.
Sensitivity	Adjusts the amount of slices. If [BPM Switch] is on, this determines whether the loop is sliced into eighth notes, sixteenth notes or thirty-second notes; if the switch is off, the knob controls the sensitivity of the transient detection algorithm.
BPM Switch	Toggles between automated transient detection (off) and a slicing of the loop into parts of equal length (on).
BPM Control	Adjusts the tempo of the loop. A tempo is extracted from the length of the sample file; by using the [+] and [-] controls this value can be modified.

4.1.4 Slice Parameters

This section below the waveform display adjusts the parameters of the currently selected slice. These parameters control the slice's start position and length, its transposition, an amplitude envelope, a hybrid compression / distortion unit entitled [Shape], and a filter. By clicking their headlines, the [Envelope], [Shape] and [Filter] part of the section can be switched on or off. Note that the envelope section, if switched off for each slice, can still be used as a modulation source.

Section	Control	Function
Position	Start	Adjusts the start position of the current slice.
	Length	Adjusts the length of the current slice.
Pitch	Transpose	Determines the amount of transposition and its direction – up or down.
	Reverse	Switches between forward and reverse playback of the current slice.

Section	Control	Function
Envelope	Attack	Specifies the time taken to reach full amplitude, as a proportion of slice length. Thus, if set to 50%, the envelope will reach peak value halfway through the slice.
	Decay	Determines decay time and shape as a proportion of the time remaining after the attack phase. At maximum, the envelope will sustain at full amplitude for the entire slice (or the remainder of the slice after the attack phase). When between 50% and maximum, the envelope consists of a sustain phase followed by a decay phase. At less than 50%, there is no sustain period, just a decay stage.
Shape	Pre	Increases the compressor input level.
	Shape	Determines the compressor gain curve.
	Smooth	Reduces the amount of distortion by smoothing gain changes as it controls the attack and release of the compressor.
	Drive	Saturates the output signal.
Filter	Mode	Selects the operation mode of the filter unit. Low-pass, band-pass and high-pass modes are available.
	Cut-off	Sets the center frequency of the filter.
	Resonance	Sets the resonance of the filter at the cut-off frequency.
Out	Gain	Adjusts the output level in decibels
	Aux Send	Sets the level of the auxiliary output port of the instrument.
	Pan	Controls the position of the sound within the stereo field.
	Out Port	Selects the output of the instrument to which the slice's sound will be routed. Depending on the global [Mono] switch either two stereo ports or four mono ports are available.

4.1.5 Modulation

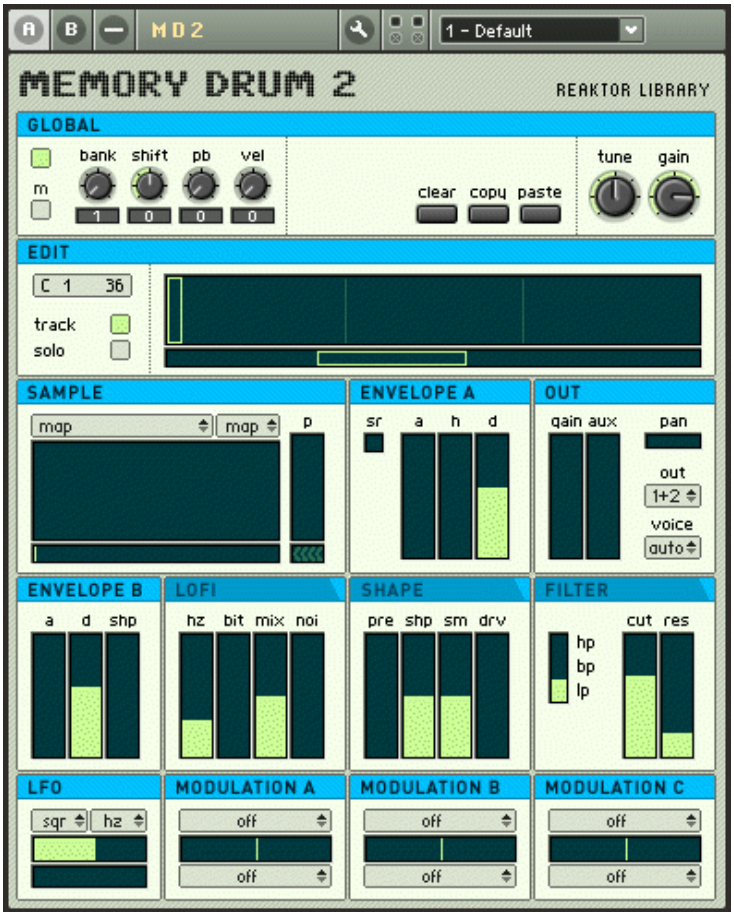
BeatSlicer 2's advanced modulation routing allows various parameters to be modulated by a variety of sources (both MIDI and internal). In both modulation sections (A and B) the left-hand box displays the current source, and the right-hand box displays the current destination. Click and drag vertically to change the source or destination. The slider bar in between speci-

fies the amount (and direction) by which the source modulates the destination. For example, to assign velocity to amplitude, select 'Vel' as the source, 'Amp' as the destination, and set the slider bar to the full right-hand position.

Some modulation sources have a variation denoted by '/H'. This option samples the value of the source when triggered. Try assigning the Pitchbend wheel to Pan and repeatedly trigger the sample while modulating the Pitchbend wheel. Change the source to 'PB/H' and listen for the difference.

Section	Control	Function	Polarity/Range
Mix Sources	Vel	MIDI note on velocity.	Unipolar
	PB	MIDI pitchbend wheel.	Bipolar
	PB/H	MIDI pitchbend wheel, sampled at note-on.	Bipolar
	CC1	MIDI controller 1 (the modulation wheel).	Unipolar
	CC1/H	MIDI controller 1, sampled at note-on.	Unipolar
	CC7	MIDI controller 7 (the volume slider).	Unipolar
	CC7/H	MIDI controller 7, sampled at note-on.	Unipolar
	Env	Envelope generator.	Unipolar
	Rnd	Random value generator.	Bipolar
Destinations	Amp	Slice amplitude	(-100% to +100%)
	Pan	Stereo pan	(-100% to +100%)
	P	Slice pitch	(-12 to +12 semitones)
	Len	Slice length	(-100% to +100%)
	Drv	Overdrive amount	(-60 to +60 decibels)
	Cut	Filter cutoff	(-120 to +120 semitones)
	Aux	Aux send level	(-100% to 100%)

4.2 Memory Drum 2



4.15 Memory Drum 2 user interface

4.2.1 Introduction

Memory Drum 2 is an advanced sampler that enables the independent configuration of up to 128 samples in a compact, easy-to-use interface. Specifically designed for drum sampling, it features an attack-hold-decay envelope, a range of effects, multiple output channels, and complex modulation options. The intuitive interface allows drum kits to be constructed quickly and easily, yet the extensive range of sound-design options offer vast creative possibilities for generating new sounds from your existing samples.

For a quick start, double-click on the sampler window, open the REAKTOR sample map editor, and load Memory Drum 2 with a few drum samples. As you trigger the samples from your keyboard, notice that the green box in the Edit section moves to indicate the current MIDI note. Any parameter that you adjust will be stored for that MIDI key. For example, press a MIDI note and then adjust the Envelope attack and decay time. Now press another key, and adjust some parameters for that sample, and so on …

Memory Drum 2 is designed to be programmed with MIDI controllers. To assign a MIDI controller to a parameter, use the MIDI learn function on the XY modules on the panel. You can also assign MIDI controllers to some parameters that use Multi Picture modules, such as the 'Shape' on/off button. To do this, right click on the module and select 'show in structure'. Then assign a MIDI controller to the hidden button module beneath the Multi Picture module. (You can also assign controllers to the Bank and Sample controls in the Sample section in this way, allowing you to browse through the sample map using MIDI controllers.)

Be aware that whenever Memory Drum 2 receives an event from an assigned MIDI controller, it will write to the memory of whatever sample (or samples) are currently selected. This can be a problem with certain hosts, which send extra MIDI controller data to plug-ins on initialization or when stopping/starting playback. Or it can be a problem if you accidentally move a MIDI controller… The safest thing to do is take frequent snapshots of your drum kit configuration.

With the exception of the sampler map configuration, all parameters are saved in the host plug-in edit buffer. This means that if you do not change the sample map, you do not need to save a new copy of the ensemble. But if you make any changes to the sample map, you should use the REAKTOR autosave feature. This will create a new copy of the Memory Drum 2 ensemble and save it with your song. When in doubt, use the autosave feature to avoid data loss.

4.2.2 Global Parameters

The master controls are located at the top of the instrument panel. They adjust the instrument's global settings, which affect all loaded samples. Note that both pitchbend and velocity can be assigned to samples individually using the modulation matrix, in which case it is probably best to set the global knobs to zero.

Control	Function
Power Switch	Mutes the entire instrument. This does not switch the instrument off to save CPU power.
Mono	When activated, the instrument provides four independent mono channels as output ports; otherwise, it offers two stereo ports.
Bank Number	Sets the number of sample banks. See the [Edit and Sample] section for details.
Shift	Transposes MIDI note input up or down as required.
Pitchbend	Determines the amount that the pitchbend wheel affects the pitch of the entire kit.
Velocity	Specifies the extent to which note-on velocity affects amplitude for all samples.
Tune	Transpose the pitch of the entire drum kit in semitones.
Gain	Controls the overall output level in decibels.
Clear	Resets all parameters of the current note to their default values.
Copy	Copies all parameters of the current note to an internal buffer.
Paste	Copies all parameters of the internal buffer to the current note.

4.2.3 Sample & Edit

The [Edit] section displays the sample map: Each slot represents a MIDI note; if this MIDI note is received, the sample selected within the [Sample] section is triggered. As there is a maximum of 128 different MIDI notes, normally only 128 samples can be loaded into a REAKTOR sample map.

However, the two selection controls above the waveform display of the [Sample] section – entitled [Bank Select] and [Sample Select] – override this limitation. The best way to explain this is by example. Imagine you have a total of 512 drum sounds on your hard disk (this is just a hypothetical example!), and you wanted to load them all into Memory Drum so that they are all readily available for selection. Start by setting the number of banks to four (using the bank

knob in the global section). Next, load the first 128 samples (using the sample map editor), assigning them to MIDI notes 0 to 127, and from velocity 1 to velocity 31. Then load the next 128 samples to MIDI notes 0 to 127, from velocity 32 to 63. Repeat this process for the remaining two 'banks' of 128 samples. You can now select any sample in the map by using the [Bank Select] and [Sample Select] lists on the panel. Although initial map creation is time-consuming, it can be extremely useful once set-up. Imagine having 128 kick drums loaded into the first bank, 128 snares loaded into the second bank, 128 hi hats into the third bank and so on... This can enable quick and easy kit creation, and convenient auditioning of samples on the fly.

Section	Control	Function
Edit	Sample Map Display	Selects the current MIDI note slot for editing. You can select a range of notes to edit simultaneously by clicking the right mouse button and dragging the mouse. Double-clicking on this bar automatically selects all notes for simultaneous editing (double-click again to return to the previous selection)
	Zoom Bar	Scrolls the viewable area across the entire MIDI note range. Clicking the right mouse button on this scrollbar cycles between 3 different zoom states.
	MIDI Note	Displays the current MIDI note selected for editing within the [Sample Map Display].
	Track	If activated, the received MIDI notes not only trigger the playback of the sample, but also select them for editing.
	Solo	If activated, only the currently selected sample can be triggered by MIDI notes.
Sample	Bank Select	Selects the bank from which the [Sample Select] controller picks a sample.
	Sample Select	Selects the sample that is played when the currently active MIDI note is received (see also [MIDI Note] and [Sample Map Display]).
	Sampler	Displays the wave file selected by [Sample Select]. You can also load new files into REAKTOR's internal sample map editor here.
	Start Position	Adjusts the start position within the sample file.
	Reverse	Switches between forward and reverse playback of the sample.
	Pitch	Transposes the sample up or down in semitones.

4.2.4 Sample Parameters

In this section you can adjust the parameters and effect settings of the currently selected sample (see [Edit][Sample map Display]). There is an envelope controlling the sample's amplitude, a lo-fi distortion effect, a compression / saturation unit labeled [Shape], a multi-mode filter and a final output part. [Lofi], [Shape] and [Filter] can be toggled on and off for the selected sample by clicking the respective section's title.

Section	Control	Function
Envelope A	Sustain/ Release Mode	If activated, the envelope remains at full amplitude after the attack time until the MIDI gate signal is closed; then the decay time is interpreted as release time.
	Attack	Specifies the time taken to reach full amplitude.
	Hold	Specifies the time held at full amplitude.
	Decay	Specifies the time taken for amplitude to fall back to zero.
Lofi	Hertz	Adjusts the re-sampling frequency in Hertz.
	Bit	Adjusts the bit depth of the re-sampling algorithm.
	Mix	Crossfades between the unprocessed, dry signal and the processed, wet sound.
	Noise	Crossfades between the re-sampled signal and a noise generator to be mixed with the unprocessed sound.
Shape	Pre	Increases the compressor input level.
	Shape	Determines the compressor gain curve.
	Smooth	Reduces the amount of distortion by smoothing gain changes as it controls the attack and release of the compressor.
	Drive	Saturates the output signal.
Filter	Mode	Selects the operation mode of the filter unit. You can choose between low-pass, band-pass and high-pass filter modes.
	Cut-off	Sets the center frequency of the filter.
	Resonance	Sets the resonance of the filter at the cut-off frequency.
Out	Gain	Adjusts the output level in decibels.

Section	Control	Function
	Aux Send	Sets the level of the auxiliary output port of the instrument.
	Pan	Positions the sound within the stereo field.
	Out Port	Selects the output of the instrument to which the slice's sound will be routed. Depending on the global [Mono] switch either two stereo port or four mono ports are available.
	Voice Group	By default, REAKTOR rotates voices to minimize voice-stealing. However in the context of drums, voice-stealing can often be desirable. Consider the example of a pair of open and closed high-hat samples – you may want these two samples to share the same voice, so that the triggering the open-hat sample truncates the closed-hat sample and vice versa. Please note that voice groups will only work effectively if: (1) the highest group number in use does not exceed the number of voices in the instrument properties, (which is four by default); and (2) all samples are manually assigned to a voice group (rather than a mixture of auto and manual voice assignments).

4.2.5 Modulation

Memory Drum 2's advanced modulation routing allows various parameters to be modulated by a variety of sources. Beside the MIDI sources – like the modulation wheel and the pitchbend control – there are two envelopes and an LFO. ([Envelope A] is hard-wired to the amplitude of the sample playback but can also be used as freely assignable modulation source.) Some modulation sources have a variation denoted by '(hold)'. This option samples the value of the source when triggered. Try assigning the Pitchbend wheel to Pan and repeatedly trigger the sample while moving the Pitchbend wheel. Change the source to 'Pitchbend/H' and listen to the difference.

Section	Control	Function
Envelope B	Shape	Morphs the envelope shape from concave (at low values) to linear (at center) to convex (at high values).
	Attack	Specifies the time taken to reach full amplitude.
	Decay	Specifies the time taken for amplitude to fall back to zero.
LFO	Waveform	Selects the waveform of the low frequency oscillator.

Section	Control	Function
	Operation Mode	In [Hz] and [Sync] modes, the LFO phase is reset every time the note is triggered, the only difference being that the frequency is quantized to tempo in [Sync] mode. In [Lock] mode, the LFO frequency snaps to the MIDI tempo and the LFO phase gets locked to the MIDI song position.
	Speed	Sets the rate at which the LFO oscillates.
	Phase	Sets the phase to which the LFO is reset when triggered by note events.

In each modulation section (A, B and C) the top box displays the current source, and the bottom box displays the current destination. Click and drag vertically to change the source or destination. The slider bar specifies the amount (and direction) to which the source modulates the destination. For example, to assign velocity to amplitude, select 'Velocity' as the source, "Amp" as the destination, and set the slider bar to the full right hand position.

Section	Control	Function	Polarity/Range
Sources	Velocity	MIDI note on velocity.	Unipolar
	Pitchbend	MIDI pitchbend wheel.	Bipolar
	Pitchbend (hold)	MIDI pitchbend wheel, sampled at note-on.	Bipolar
	CC1	MIDI controller 1 (the modulation wheel).	Unipolar
	CC1 (hold)	MIDI controller 1, sampled at note-on.	Unipolar
	CC7	MIDI controller 7 (the volume slider).	Unipolar
	CC7 (hold)	MIDI controller 7, sampled at note-on.	Unipolar
	Env A	Envelope generator A.	Unipolar
	Env B	Envelope generator B.	Unipolar
	LFO	The LFO.	Bipolar (except Sqr+ & Ramp)
	LFO (hold)	The LFO, sampled at note-on.	Bipolar
	Rnd	Random value generator.	Bipolar
Destination	Amp	Slice amplitude	(-100% to +100%)
	Pan	Stereo pan	(-100% to +100%)

Section	Control	Function	Polarity/Range
	Pitch	Slice pitch	(-12 to +12 semitones)
	Start	Sample start position	(-1 to +1)
	Env A Attack	Envelope A attack time	(approximately -10 to +10 seconds)
	Env A Decay	Envelope A decay and hold time	(-100% to +100%)
	Hz	Lofi resampling frequency	(-100 to +100 semitones)
	Drive	Overdrive amount	(-60 to +60 decibels)
	Cut	Filter cutoff	(-120 to +120 semitones)
	Aux	Aux send level	(-100% to 100%)
	Env B Amp	Envelope B amplitude	(-100% to 100%)
	LFO Amp	LFO amplitude	(-100% to 100%)

4.3 Grainstates SP

See 1.1.7.

4.4 Travelizer



1.15 Travelizer user interface

4.4.1 Introduction

Travelizer is the latest version of the classic granular texture maker. Travelizer lets you scrub through any sample using the grain cloud module. It can be played over MIDI, allowing you to create ethereal pads and leads. Travelizer differs from its earlier versions because grain length can be quantized to 16th notes, allowing textures and granular rhythms to be synced to MIDI clock.

4.4.2 Quick Start

Step through the presets with the system clock on and feel free to experiment! Try using the travel xy pad to control sample playback position and grain length while a sample is playing. If you click on the pad, the volume Attack-Release envelope will trigger and start the sound. Click the Gate button to continuously trigger the volume envelope so you can make adjustments while the sample is looping.

4.4.3 Structure and Signal Flow

Travelizer granulates any sample loaded into it, giving you a variety of controls over the quality of the granulation. The sample grains pass through a 3-voice Resonator with independent controls for tuning and the capability to track midi notes. The Resonator is followed by a stereo delay with paired with a high-pass filter. An attack-release envelope allows you to contour the loudness of the sound.

Travelizer's grain cloud module allows you to break the sample down into grains, specifying certain qualities of the sound, like grain size and smoothness.

The Pitch module has controls to tune the sample and set whether it will track incoming MIDI notes. The MIDI switch turns note-tracking of sample pitch on. The Jitter control introduces small amounts of pitch modulation. The Slide switch next to this makes the sample pitch glide from note to note. Slide intensity can be positive or negative. An LFO allows you to modulate Travelizer's pitch - turn it on and off with the small switch above the Shp fader. The LFO can be crossfaded between triangle wave and sine wave shapes with the Shp fader. Clicking and dragging on the Amount and Rate x-y pad lets you adjust modulation depth and speed. The LFO can be set to retrigger on MIDI gate events with the Gate button.

The Position module allows you to modulate the playback position of the sample.

You can smoothly, or roughly, sweep through the sample using a crossfadeable sine/triangle LFO like the one in the Pitch module. The Inertia control allows you to smooth out the LFO movement, in effect limiting its swing. The Jitter control adds small amounts of random position modulation to achieve a glitchy rhythmic edge. Like the Pitch LFO, the Position LFO can be retriggered by MIDI gate events by clicking the Gate button.

The Smoothness control lets you adjust volume crossfading between sample grains. A high Smoothness amount will result in a more defined sound with less obvious granulation. The lower the Smoothness control is set, the more granulated the output will sound.

The sample is now fed into the Resonator. The Resonator uses three voices to create tuned atmospheric and spatial effects. Each of the three voices can be tuned in semitones relative to each other. Clicking the MIDI switch lets you control the root pitch using MIDI notes. You can control damping and decay of the Resonator with the xy pad. A rhythmic loop can be turned into a harmonic soundscape by manipulating the damping and decay controls and creating chords with the voice tunings.

From the Resonator, the sample passes through the Delay module, which includes a high-pass filter. You can use this to manipulate the sonic character of the sample before it hits the delay by removing low frequencies with the filter. The Delay xy pad lets you set 16th note-quantized delay times for the left and right channels.

Finally, an attack-release envelope allows you to contour the loudness of your sound. Use the sample control to select a sample or double-click on the grain cloud module to load your own.

4.5 Lurker



4.2 Lurker's user interface

4.5.1 Introduction

Lurker is a hybrid effect capable of classic phaser sounds, spring reverbs and feedback echoes – but most of all it transforms any incoming signal into stunning rhythmic sequences, mangling pitches and re-arranging the sound. This is technically possible because all those effects are based on a delay unit (and this instrument is an extremely versatile one).

Four internal sequencer tracks are the most prominent feature. They allow for fast, visual creation of musical patterns that you can use to modulate parameters such as delay times of the two independent delay units. Those times can be set in sixteenth note multiples (for tempo-based effects) or in milliseconds (for comb filter-like effects that map a new pitch onto the signal). A filter, a gating envelope generator, and a final delay further enrich the sound.

4.5.2 Global

This top section of the instrument panel contains three parts: the input control (at the left), the snapshot management (in the middle), and the shuffle control (at the right).

The input control provides a simple sampler to load files and re-trigger their playback synchronized to the sequencers. The level of external signals can be controlled here. The snapshot handling and the shuffle system are identical to those of Massive; see that instrument's manual for further details.

Section	Control	Function
Input	Loop Switch	Controls the events that re-trigger the sampler. If on, the sampler starts playback at the file's beginning when the loop controlled by [Length Control] and [Unit Select] is returning to its origin; if off, the sampler is re-triggered only when the global MIDI clock starts playing.
	Length Control	Sets the length of the loop that controls the sampler's re-triggering if the [Loop Switch] is on. (See also [Unit Select].)
	Unit Select	Selects the rhythmical unit on which [Length Control] is based. This unit refers to the global MIDI clock.
	Sampler	Displays the currently active sample (see [Sample Select]). Double-click to open the Sample Map Editor where sample files can be loaded and organized.

Section	Control	Function
	Sample Select	Selects one of the samples loaded into the [Sampler].
	Sample Pitch	Transposes the selected sample. This also affects the samples playback speed. (An octave up or down sets the playback to double or half speed respectively.)
	Internal Level	Controls the amplitude of the sampler.
	External Level	Controls the amplitude of the external signal.
	External Mute	Disables the external input.
	External Display	Shows the level of the external signal.
Snapshot	Snapshot Store	With the left mouse button, a snapshot slot number can be selected; by pressing the right mouse button, the current instrument settings (including all sequencer data) are stored into this snapshot slot.
	Snapshot Recall	Displays a list of the available snapshots; selecting a snapshot with the mouse recalls all its data, including sequences.
	Snapshot Mode	Selects whether the snapshots are only recalled internally or if external control signals received at the instrument's [Snap] port are recognized, too. This allows connection to a master song sequencer.
Shuffle	Quantization Select	Selects one of twelve quantization presets. Each preset ranges over sixteen steps; the higher the value within the display, the more delay is applied to this step. The first preset, for example, alternates between low and high values, so every second step will be delayed, resulting in a standard off-beat shuffle. The presets only define relative times; the effective delay time at maximum values is set by the [Shuffle] control.
	Shuffle	Scales the preset of the [Quantization Select] control. Turn to the left for no quantization – independent of the selected preset –, to the right for full delay times.

4.5.3 Sequencer

There are two step sequencers (tracks [A] and [B]) and two tracks that glide from step to step ([C] and [D]). Each sequencer provides individual control over length and speed.

Control	Function
Length Control	Sets the length of the loop that can be edited within the sequencer display in steps. (See also [Unit Select].)
Unit Select	Selects the rhythmic unit with which each step of the sequencer track is interpreted. This refers to the global MIDI clock.
Sequencer	Defines and displays the track's rhythmic pattern.

4.5.4 Delay Units

Two identical delay units form Lurker's core. They can be used either in parallel or serially. Each offers independent delay times for the left and the right audio channels, both defined as multiples of sixteenth notes or in milliseconds. At the left of the controls that adjust the delay times, their modulation can be controlled; even the depth of modulation is subject to modulation, resulting in complex interactions of several modulation patterns. The knobs at the delay time controls' right define the channel swap, the amount of feedback and the feedback's filtering.

Control	Function
Depth	Sets the amount of modulation applied to the delay time. This is independent of the static delay time and ranges from no modulation (at the left) to a modulation of about 260 milliseconds (at the right). The modulation signal is selected by the [Modulations Source] control below. (See also [Depth Modulation Amount].)
Modulation Source	Selects the sequencer track that modulates the delay time. The amount of modulation at maximum modulation signals is controlled by [Depth].
Depth Modulation Amount	Adjusts the amount of modulation of the [Depth] control. Turn to the left for inverted modulation (i.e. there is much modulation at low modulation signals and vice versa), to a mid position for no modulation and to the right for normal modulation. High values at the right result in very much modulation of the [Depth] control, increasing its maximal modulation amount to approx. 2400 milliseconds. The signal that actually modulates the [Depth] control is selected below.
Depth Modulation Source	Selects the sequencer track that modulates the modulation depth. The amount of modulation is controlled by [Depth Modulation Amount].

Control	Function
Modulation Slur	Sets the amount of interpolation applied to subsequent steps of the modulation track. Turn to the left for no interpolation and fast delay time changes, to the right for soft and slow ramps between different states.
Modulation Invert	Inverts the modulation signal i.e. if on, the modulation signal is not added to the static delay times controlled by [Quantized Delay Time Left / Right] and [Millisecond Delay Time Left / Right], but subtracted.
Quantized Delay Time Left / Right	Sets the static delay time of the left channel (upper control) and right channel (lower control) respectively in multiples of sixteenth notes of the global MIDI clock. The actual delay time is calculated from the sum of this value, the delay time adjusted by [Millisecond Delay Time Left / Right] and the modulation signal (see [Depth]),
MillisecondDelay Time Left / Right	Sets the static delay time of the left channel (upper control) and right channel (lower control) respectively in milliseconds. The actual delay time is calculated from the sum of this value, the delay time adjusted by [Quantized Delay Time Left / Right] and the modulation signal (see [Depth]),
Channel Swap Amount	Controls the modulation applied to the interaction of the left and right feedback signal. At low modulation signals the left channel's feedback signal is routed again to the left channel; at mid modulation values both channels are mixed to a mono sound that is fed back into both channels identically; at high modulation signals the channels are swapped and the left channel's signal is routed to the right channel (and vice versa). This control scales the modulation signal, i.e. at mid position high modulation signals are mapped to mid modulation signals; at the complete left there is no modulation and no channel swapping. The modulation signal is selected below.
Channel Swap Modulation Source	Selects the sequencer track that modulates the channel swap. The amount of modulation is controlled by [Channel Swap Amount].
Cutoff	Sets the cut-off frequency of the low-pass filter within the feedback loop.
Reset	Sets all controllers of the delay unit to their default values.
Feedback Amount	Controls the amount of feedback.
Bypass Switch	Toggles between the dry, unprocessed signal (when on) and the wet, delayed signal.

Control	Function
Mode Select	Switches between parallel and serial modes. In parallel mode, both delay units receive the same input signal and the [Crossfade] control can crossfade between their output signals. In serial mode, the signal enters the upper delay unit and is then routed to the lower unit.
Crossfade	Mixes between the sound of the upper and lower delay unit when [Mode Select] is set to parallel.

4.5.5 Filter

The filter is placed after the two delay units. The low-pass filter's cut-off frequency and resonance can be edited (you can adjust left and right channels independently); the cut-off can also be modulated by one of the four modulation tracks.

Control	Function
Cutoff	Adjusts the cut-off frequency of the filter. The horizontal axis controls the left channel, the vertical axis the right one.
Cutoff Modulation Amount	Sets the amount and polarity of the modulation applied to the low-pass filter's cut-off frequency.
Cutoff Modulation Source	Selects the modulation track that is used to modulate the filter's cut-off frequency.
Resonance	Adjusts the resonance of the filter. The horizontal axis controls the left channel, the vertical axis the right one.
Reset	Sets all controllers of the filter to their default values.

4.5.6 Master and Envelope

The master section simply controls the instrument's output level before its signal passes to the additional delay. The [Env] control enables an envelope generator that is triggered by one of the two step sequencer tracks. This can be used to gate the instrument's signal.

Section	Control	Function
Master	Output	Controls the instrument's main output level.
	Bypass	Mutes the effect and directly routes the input signal to the output.

Section	Control	Function
	Envelope Amount	Adjusts the influence of the envelope generator on the output amplitude. Turn to the left for complete independence. Turn to the right for full shaping of the amplitude by the envelope.
Envelope	Source Select	Selects one of the two step sequencers to be used as trigger signal. (See also [Gate Threshold].)
	Gate Threshold	Controls which steps of the selected modulation track are used as trigger signals. All steps with values below the one adjusted by this control are ignored.
	Velocity Amount	Controls the influence of the trigger gate's height (i.e. velocity) on the envelope's amplitude. Turn to the left for full amplitude with every trigger signal, turn to the right to map the step's value onto the envelope's amplitude.
	Velocity Attack	Controls the amount of modulation applied to the envelope's attack time by the triggering step's velocity. At low velocities (i.e. low step values) the attack time is increased if the knob is turned to the right. At left positions the velocity doesn't affect the attack time. This is independent of the [Velocity] control.
	Velocity Decay	Controls the amount of modulation applied to the envelope's decay time by the triggering step's velocity. At low velocities (i.e. low step values) the decay time is decreased if the knob is turned to the right. At left positions the velocity doesn't affect the decay time. This is independent of the [Velocity] control. (See also [Decay].)
	Decay	Sets the static decay time of the envelope that can be modulated by the triggering step's velocity (see [Velocity Decay]).

4.5.7 Additional Delay

The delay unit after the output section allows for further manipulation of the signal. It is similar to the main delay units, but the delay times can't be modulated and the channels can't be swapped; instead, there is a high-pass filter within the feedback loop. As a special feature, the ratio between dry, unprocessed signal and wet, delayed sound can be modulated by one of the modulation tracks.

Control	Function
Quantized Delay Time Left / Right	Sets the static delay time of the left channel (upper control) and right channel (lower control) respectively in sixteenth note multiples of the global MIDI clock. The actual delay time is calculated as the sum of this value and the delay time adjusted by [Millisecond Delay Time Left / Right].
Millisecond Delay Time Left / Right	Sets the static delay time of the left channel (upper control) and right channel (lower control) respectively in milliseconds. The actual delay time is calculated as the sum of this value and the delay time adjusted by [Quantized Delay Time Left / Right].
Feedback Amount	Controls the amount of feedback.
Highpass	Sets the cut-off frequency of the high-pass filter within the feedback loop.
Lowpass	Sets the cut-off frequency of the low-pass filter within the feedback loop.
Mix Modulation Source	Selects the sequencer track modulating the ratio between dry, unprocessed signal and wet, delayed sound (see [Mix]).
Mix	Controls whether the ratio between dry, unprocessed signal (heard at mid position) and wet, delayed sound (at the right). Turn to the left to use the sequencer track selected by [Mix Modulation Source] as control: At high modulation levels the wet sound is passed on; at low values, the dry signal.

5 Sequenced Synthesizers

5.1 Akkord



3.1 Akkord user interface

Introduction

Akkord is a hybrid additive / FM synthesizer, featuring an original tone generation engine combining 3 oscillators, a harmonics generator and flexible wave shaping options. The rich and distinctive sound is enhanced by a range of internal effects, and animated by 2 feedback envelopes and beat-syncable LFOs. Further, the ensemble features a unique chord sequencer with which to drive this powerful synthesiser.

Global controls

Section	Function
Synth settings	Akkord is a polyphonic synthesiser, but it can optionally operate monophonically by enabling the **Mono** button. In this mode, when multiple MIDI notes are pressed simultaneously, the lowest note is used. **VelAtt** sets the extent to which the envelopes attack time is influenced by velocity (softer velocity results in longer attack times). Set to hard-left for no influence. **Porta** switches between one of two portamento modes. When the button is off, playing notes legato-style triggers pitch glide, but when the button is on, pitch glides every time a note is played. **Time** sets the portamento time (from no glide at hard-left to very slow at hard-right). **Key** and **Fine** transposes the note input in semitones, and **PBnd** sets the pitchbend range in semitones.
Envelopes 1 and 2	Envelope 1 is hard-wired to amplitude, but like Envelope 2, it can also be freely assigned to other parameters using the drop-down menus in the other sections. The 2 envelope controls are identical -**Attack**, **Decay** and **Release** set the times of the respective stages and the **Sustain** knob sets the level of the sustain stage. When the envelope Delay sections are enabled, the envelope signal is fed through a tempo-synced feedback-delay unit, creating more complex, rhythmic envelope shapes. **Delay** sets the delay time in 16th steps, and **Return** controls the amount of feedback.
LFO 1 and 2	The two (identical) low frequency oscillators can be used as modulation sources for the oscillators, the Harmonics / Shaper section and the filter. Five waveforms are available – Sin (sine); Tri (triangle); Pls (pulse); Rnd1 (slow random waveform) and Rnd2 (random waveform).

Section	Function
	When **Sync** is enabled, the LFO cycle is retriggered on each MIDI note (phase start position is set with the **Phase** knob). The **BPM** switch enables the LFO speed to be set in cycles- per-bar, and the **Fade** switch enables LFO fading in and out (fade time is set using the **Fade** knob). **Freq** specifies the LFO speed (in either cycles-per-bar, or cycles-per-second) and **Width** adjusts the waveform shape.

Tone Generation

Section	Function
Main and modulation oscillators	**Main Osc** selects the main oscillator waveform, and **Mod Osc** selects the modulator oscillator waveform. Five identical shapes are available for both oscillators -Par (parabolic waveform), Tri (triangle), Saw (Sawtooth), Pls (Pulse) or Imp LP (lowpass filtered impulse).
Sub oscillator	The Sub Osc section provides an additional oscillator to be mixed with the Main Oscillator. The upper menu selects the waveform, and the Gain knob determines the sub oscillator output level. The lower menu selects a pitch modulation source, and **Mod** specifies the modulation amount. **-Oct+** enables the oscillator pitch to be transposed, in octaves (+/- 3).

Section	Function
Noise oscillator	The noise oscillator is a second additional oscillator, offering 5 different types of noise to be mixed with the main and sub oscillator output. The upper drop-down menu selects the noise type from 1 of 5 choices – Noise (white noise); Rnd Mono (monophonic random sample-and-hold noise, more sharp); Rnd Poly (polyphonic random sample-and-hold noise, more smooth); Geiger (produces clicks at random intervals); Geiger Poly (produces polyphonic clicks at random intervals). **Gain** defines the noise output level. The **Colour** knob allows fine adjustment of the noise generator. The effect of the colour knob depends on the noise type selected - in Noise mode, the colour knob controls the balance between low and high frequencies; for the Rnd modes it determines the frequency of the random waveform; and in the Geiger modes, it defines the impulse density. The Colour parameter can be modulated – select a modulation source from the lower drop-down menu; and set the amount of modulation with the **Mod** knob.
Pitch and Amplitude	The section labelled PM (pitch modulation) above the modulation main/mod oscillator menus provides a low frequency sine oscillator dedicated to oscillator pitch modulation (for main, mod and sub oscillators). When the **PM** power switch is enabled, **Depth** defines the amount of pitch modulation and **Rate** sets the LFO speed. The LFO is polyphonic and each new MIDI note resets the phase. Similarly, the section labelled AM provides a dedicated amplitude modulation LFO. (The amplitude modulation is applied after the Harmonics & Shaper section.)

Section	Function
Harmonics/Shapers	This section is where the mod oscillator modulates the main oscillator to create additional harmonics. This section also offers several wave shaping distortion algorithms. **Harm** determines the amount of harmonic distortion of the oscillator signal (main, sub and noise) via the modulation oscillator. The harmonic distortion is created by applying a small delay to the signal and modulating the delay time via the modulation oscillator. **Shaper** determines the amount of wave-shaping distortion. Five distortion modes are available from the drop-down menu in the top-left corner of this section. In If Pulse then Width mode, the shaper knob sets the pulse width of the main, mod or sub oscillators (if any of them are set to pulse). Falling slope mirror mode inverts falling signals; Slope to Sign converts rising signals to a static +1 signal and fallings signals to a static –1 signal (similar to a pulse wave); Parabolica runs the signal through a parabolic function; and Sign converts positive signals to a static +1 signal and negative signals to a static -1 signal. Note that selecting shaper bypass from the shaper mode menu bypasses the shaper only – harmonics distortion cannot be bypassed. **Env** and **LFO** control the amount of modulation of both the **Harm** and **Shaper** parameters via the envelopes or LFOs. The particular modulation source can be selected via the drop down menus at the left. **HR/SR** mixes between harmonic distortion only and harmonics plus shaped signal. If shaper is set to bypass then this control mixes between harmonic distortion and the dry input signal.
Filter	Six filter types are available, selectable by the drop-down menu in the top-left. Allpass 1 is a 1-pole allpass phase shifter; Allpass 2 is a 1-pole allpass with feedback and delay, and a more pronounced phase shifting effect; LP2 is a 2-pole lowpass filter; LP4 is a 4- pole lowpass filter; HP2 is a 2-pole highpass filter and HP4 is a 4-pole highpass filter.

Section	Function
	Cutoff determines the filter cutoff frequency, and **Reso** determines the filter resonance. **Ktr** (key track) sets the amount of influence of MIDI note on cutoff frequency. The middle drop-down menu selects an envelope modulation source, which corresponds to the **Env** depth knob. Similarly, the lower menu selects an LFO modulation source, which corresponds to the **LFO** depth knob.

Effects section

Section	Function
Phaser	**Center** determines the phaser centre frequency; **Depth** determines the amount of frequency modulation; and **Rate** defines the frequency modulation rate. The frequency modulation is conducted by 4 internal LFOs, and the **Phase** knob determines the phase offset between them. **FB** defines the amount of feedback, and **Flange** enables flanger mode (a small delay is applied to the feedback signal). **Mix** crossfades between the input and the wet signals.
Chorus / Tremolo	This section features a stereo chorus/tremolo effect, which splits the input signal into high and low frequency bands and then modulates them independently. **Centre** sets the pitch at which low and high bands are divided, and **D Lo** and **D Hi** set the delay time for the low and high bands. **Rate** sets the modulation rate; **Phase** determines the amount of phase offset between the internal LFOs; **D Mod** controls the amount of delay time modulation; and **A Mod** controls the amount of amplitude modulation. **Mix** crossfades between the input and the wet signals.

Section	Function
Delay	A stereo, tempo-synced delay unit with feedback-filter. **Del L** and **Del R** set the delay time for the left and right channels in 16th steps. **Fb** sets the amount of feedback. The feedback path features both a high-pass and a low-pass filter. The **HP** and **LP** knobs define the respective cutoff frequencies. **Mix** crossfades between the input signal and the wet, delayed signal.
EQ, O-Drive	The **Low** and **High** knobs set the frequency of a pair of low and high shelf EQ units, for which the Boost knobs determine the amount of frequency attenuation/amplification The **Drive** knob determines the amount of signal amplification before a saturator unit.

The Sequencer

The Akkord ensemble also features a powerful 8-pattern, 32-step, polyphonic chord sequencer. As the sequencer is driven by the MIDI clock, please ensure that the play button is pressed on the REAKTOR toolbar.

The 8-patterns (labelled A to H) can be selected by clicking on the pattern selector at the top of the main Sequencer section. Click once on the pattern selector to select a single pattern to play/edit, or click and drag to select a range of patterns to loop.

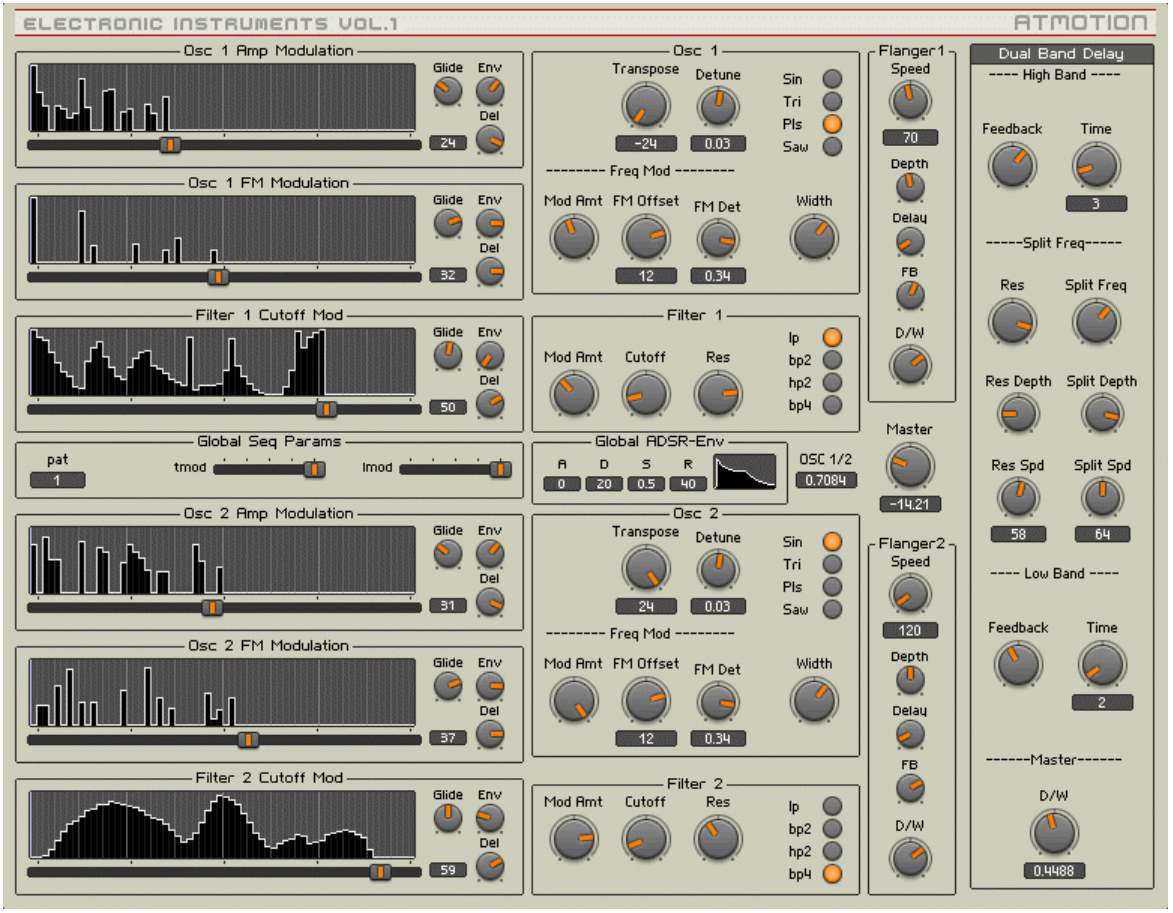
Beneath the pattern selector lies the 32-step sequencer (the central display). Each of the 32 steps can store up to 32 notes. Click on any of the 32 steps to be able to edit with the note editor (bottom display). In the note editor, you can either create chords by drawing individual notes by hand, or by selecting a chord from the **Chords** list, enabling the chord generator (press **On**), and then clicking on the root note in the editor.

The order of the 32 steps can also be rearranged. To move steps, click and drag in the 32-step sequencer display (the middle display).

On the right hand side are **Copy** and **Paste** controls for both the 32-step sequencer (to allow copying and pasting between patterns) and the note editor (to allow copying and pasting between steps). Pressing the arrow buttons shift the notes up and down the keyboard on the selected step.

The Octave transposes MIDI output, and the **Gate** knob determines the length of sustain for notes (in 16th steps).

5.2 Atmotion



2.2 Atmotion user interface

Introduction

Atmotion was designed to create moving sounds like pads, atmospheres, and special effects. Osc amplitude, FM modulation amount, and filter cutoff are controlled by LFOs and sequencers to be modified rhythmically. Many different types of sounds are possible in addition to atmospheric sounds, such as percussive sounds, pads, and leads with slight and subtle LFO / sequencer modulation.

Signal Flow and Overview

Atmotion consists of two identical oscillator->filter->flanger sections. Each section has three step sequencers, which are hard wired to oscillator level, frequency modulation (FM) amount, and filter cutoff. After the flangers, the two sections are mixed, put through a traditional ADSR amplitude envelope, then sent through a chorus and dual-band delay effect:

[osc1]-[filter1]-[flanger1]

and

[osc2]-[filter2]-[flanger2]

sent through

[envelope]-[chorus]-[dual-band delay]-[out]

There is also a lfo modulating the fm depth and the pulse width; its controls are to be found in instrument panel. All modulation sources are monophonic, so all voices are modulated the same. Each of the 6 event sequencers are followed by an envelope and an event feedback delay that modify the modulating signals, using a structure as follows. It s important to remember that the envelope and delay only operate on the EVENTS that the sequencers output.

[sequencer]-[envelope]-[delay]-[oscillator level]

[sequencer]-[envelope]-[delay]-[FM depth]

[sequencer]-[envelope]-[delay]-[filter cutoff]

The envelope is triggered if the difference between to consequent sequencer steps is greater than the threshold set by the "thrsh" knob. Before, the sequencer output can be smoothed using the "Glide" knob. Both signals -the one modified by the envelope and the smoothed one - can be mixed using the "envd" knob. The pattern length of each sequencer can be adjusted with the slider below its display. These controls can be found on the Instrument panel.

The "tmod" and "lmod" sliders fade the modulation of the seqeuncesr (tmode) and the lfos (lmod) in and out to softly control their effect on the sound. They are also controlled by midi.

Programming new sounds, some things sould be kept in mind:

- Use the "pat" knob to select a new and empty row in the seqeuncer tables. If you modify the existing table values, also the sound of the old snapshots (that use that data) will be modified.
- The sequencers are controlled by the global clock that has to be running to start the seqeuncers.

Sequencers

Each of Atmotion s six sequencers operates identically.

- <Table> Displays the modulation data. Data can be graphically edited.
- <Slider> Sets the length of the modulation data sequencer, in 16th notes.
- <Glide> Smooths the modulation data.
- <Env> Controls the mix between the triggered envelopes and raw modulation data.
- <Del> Controls the mix between the we/dry event delay. Left is the dry signal, and right is the wet (delayed) signal.

Further sequencer parameters can be found in the Instrument panel (double click on Atmotion's title bar to go to the instrument panel): The <Env> and <Del> controls appear in both the Ensemble (main) view and Instrument panel.

- <Thrs> Sets the envelope trigger threshold (the difference between two sequence steps)
- <A> Sets the triggered envelope attack time.
- <D> Sets the triggered envelope decay time.
- <Ste> Sets the delaytime of the event delay, in 16th notes.

- <FB> Sets the feedback of the event delay.
- <C/Dv> 1/96 clock divisor. Reaktor s internal clock sends out values every 1/96th note. A <c/Dv> value of 6 would therefore correspond to 1/16th notes. A value of 12 corresponds to 1/8th notes, etc. This control is useful for odd-time sequences.

Oscillators

Atmotion's two oscillators are identical. They feature a choice of four waveforms, transposition, detuning, and have an FM operator as well.

- <Transpose> Transposes the oscillator against the MIDI pitch.
- <Detune> Sets the detuning of the oscillator, in cents.
- <Sin> Selects a sine wave as the oscillator waveform.
- <Tri> Selects a triangle wave as the oscillator waveform.
- <Pls> Selects a pulse wave as the oscillator waveform, with pulse width modulateable by the LFO.
- <Saw> Selects a sawtooth wave as the oscillator waveform.
- <Mod Amt> Sets the amount of FM modulation by the event table sequencer.
- <FM Offset> Sets the transposition of the FM operator in semitones.
- <FM Det> Sets the detuning of the FM operator, in cents.
- <Width> Sets the pulsewidth of the pulse wave.

LFOs

Further oscillator parameters can be found in the Instrument panel (double click on Atmotion's title bar to go to the instrument panel). These three parameters control an LFO, which is hard-wired to pulse width modulation and/or vibrato

- <Freq> Sets the speed of the LFO in 16th notes.
- <WM> Sets the LFO pulsewidth modulation depth
- <Vbr> Sets the LFO vibrato depth.

Filters

The filters comprise band-pass, high-pass, and low-pass modes, with five types of low-pass filter to round things out. The dbl filter engages two serially-connected low-pass filters; Cut2 and Res2 adjust cutoff and resonance offset for the second filter. The fb mode allows you to control filter feedback with the FB knob. From the main ensemble panel, the following parameters can be adjusted:

- <Mod Amt> Sets the amount of filter cutoff modulation by the event table sequencer.
- <Cutoff> Sets filter cutoff.
- <Res> Sets filter resonance.
- <lp> Selects lowpass filter mode. Actual filter type is set from the Instrument panel.
- <bp2> Selects a 2-pole band pass filter.
- <hp2> Selects a 2-pole high pass filter.
- <bp4> Selects a 4-pole band pass filter.

From the Instrument panel, the following additional controls are available:

- <LPType> Selects which lowpass filter to use in <lp> filter mode:
  - <cs> Lowpass filter with a characteristic that resembles the Yamaha cs-15 filter
  - <dbl> Distorting 24dB/oct filter.
  - <fb> Lowpass Filter with feedback
  - <p52> Pro-52 filter.
  - <2pol> 12dB/octave standard lowpass filter.
- <Cut2> Sets second filter s cutoff. Only active in <dbl> lowpass filter mode.
- <Res2> Sets second filter s resonance. Only active in <dbl> lowpass filter mode.
- <FB> Sets feeback of <fb> filter.
- <KTrk> Sets filter keytracking. Value = 0: cutoff frequency not affected by note pitch; Value = 1: cutoff frequency tracks note pitch exactly

Dual Band Delay

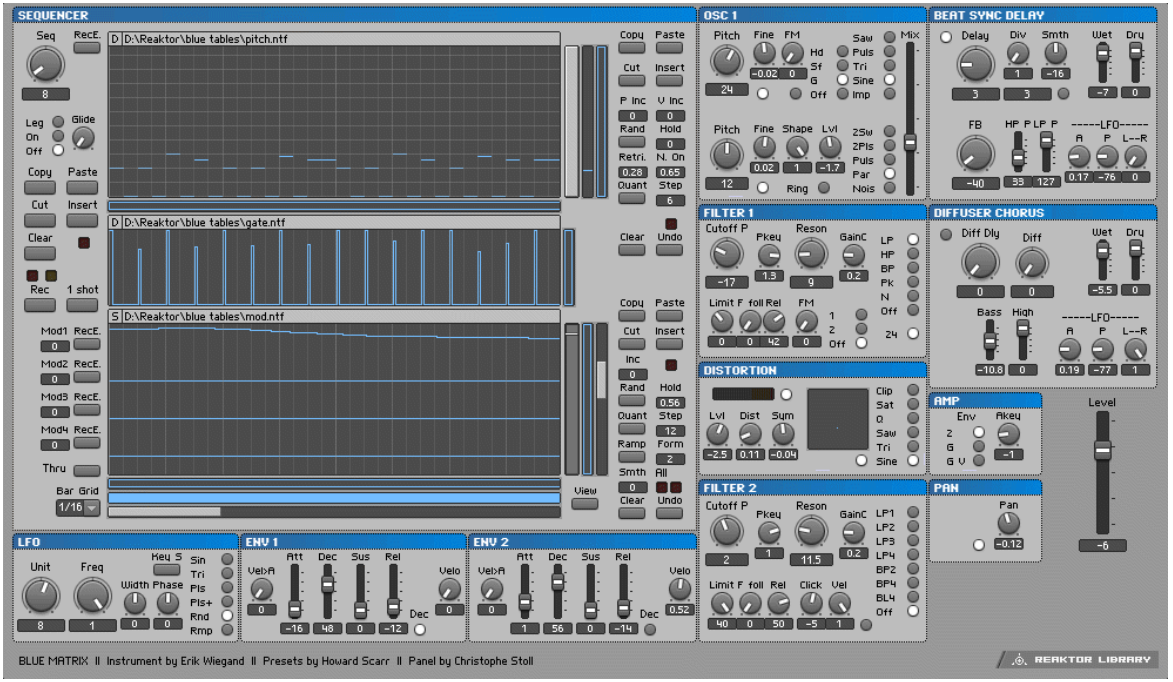
The two oscillater/filter/flanger chains are merged and then sent through a dual-band delay. The dual band delay effect first divides the sound into two frequency bands: Low and High. Each band has separate delaytime and feedback contols. Two LFOs modify the resonance and cutoff of the splitting filter.

- <Res> Sets the resonance of the filter that divides the sound into two bands.
- <Res Depth> Sets the depth of the LFO that modulates the resonance of the splitting filter.
- <Res Spd> Sets the speed of the LFO that modulates the resonance of the splitting filter.
- <Split Freq> Sets the middle frequency that divides the sound into two bands.
- <Split Depth> Sets the depth of the LFO that modulates the cutoff of the splitting filter.

<Split Spd> Sets the speed of the LFO that that modulates the cutoff of the splitting filter. Each band of the delay operates identicaly:

- <Feedback> Sets amount of feedback for the frequency band.
- <Time> Sets delay time of the frequency band, in 16th notes.

5.3 BlueMatrix



1.16 BlueMatrix user inetrface

5.3.1 Introduction

BlueMatrix is an incredible-sounding sequenced synthesizer with a classic analog-style sound engine and an integrated pitch, gate, and modulation sequencer whose capabilities rival those of many standalone programs. Its sound engine features two multiple-waveform oscillators, a multi-mode filter, multi-mode distortion, and finally a second multi-mode filter. A beat-synced delay and a diffusion delay round out the effects. The full-featured sequencer offers independent control over gate and pitch for smooth pitch glide, and also offers four channels of graphi-

cal modulation sequencing. A complete modulation matrix lets you flexibly route the two envelopes, LFO, four channels of sequenced modulation, and the usual MIDI controllers to every important synth engine parameter.

5.3.2 Quick Start

Start the system clock and step through the presets! Feel free to experiment with the synth controls - try changing oscillator pitch or cutoff frequency, for instance. If you draw in the tables, you will permanently change that preset if you save the ensemble, so it's probably better to read the Sequencer section of this guide first.

5.3.3 Structure and Signal Flow

BlueMatrix's synth engine is based on traditional analog-style synthesis. It has two unique oscillators, and a sound-shaping section consisting of Filter 1 > Distortion > Filter 2. Both Blue Matrix and its polyphonic, MIDI-playable brother Green Matrix are created from Reaktor 4's Classic Modular macro library which puts the sound quality and capabilities of the classic modular synths into your computer. For more information on building your own instruments with the Classic Modular macro library, please check your Reaktor 4 user's guide. The Classic Modular library is included on the Reaktor 4 CD.

Osc1 and Osc2 each offer a different waveform selection. Osc 1 offers the traditional sawtooth, pulse, triangle, sine, and impulse waveforms, while Osc 2 lets you choose between bipolar ramp, bipolar pulse, pulse, parabolic, and noise. The bipolar waveforms' shape can be modified in realtime with the small Shape knob. Osc 1 can be frequency modulated by Osc 2 for FM sounds (with the small FM knob and switch above), while Osc 2 can be ring modulation by Osc 1 for metallic and "gong" sounds (with the small Ring switch). Osc 1 can also be synced to Osc 2, with the choice between hard, soft, and MIDI gate-activated modes. Since Osc 1 and 2 each offer unique features, they can both be switch on or off to save CPU with the small switch on the right. The base pitch and fine tuning of each oscillator is done with their respective Pitch and Fine knobs.

The two oscillators feed into Filter 1 which offers a choice of 12 or 24 dB/octave lowpass, highpass, bandpass, peak EQ, and notch modes. Cutoff and Reson adjust the cutoff frequency and resonance, respectively, and the FM knobs controls how much Osc 1 or Osc 2 will frequency-modulate the cutoff frequency (settable with the small switch to the right of the FM knob). Pkey determines how the filter tracks the pitch of the notes from the sequencer. When

Pkey is set to 1, then the filter cutoff will exactly follow the pitch of the sequencer notes, meaning that higher notes will be brighter, and lower notes will be duller. If set to zero, then the cutoff frequency won't change. The other knobs, GainC, Limit, F foll, and Rel all control the fine-tuned shaping of the filter across the frequency and resonance spectrum. GainC controls how much loudness reduction will be applied at high resonance values. Limt, F foll, and Rel control a built-in resonance limiter to avoid unwanted harmonic thumps with high resonance values.

The Distortion section between the two multimode filters also features multiple modes of operation. While the clipper mode provides a relatively harsh distortion sound and the saturator mode results in warm overdrive, the several wrapping modes (marked by the name of the waveform used for wrapping) produce unique sounds from subtle to extreme.

A visual display of the distortion function helps to see what's going on inside. Note that the same Filter 1 > Distortion > Filter 2 section of Blue Matrix is available as an effect in the Analogic Filter Box.

Filter 2 is similar in layout to Filter 1, but its primary function to to further shape the sound after the Distortion unit. There are a choice of four lowpass filters, two bandpass filters, and a combined bandpass/ lowpass filter (BL4). In addition to the controls you're now familiar with from Filter 1, Filter 2 also offers the additional possibility to trigger self-resonance with a built in click. The Click knob controls how loud the click is, and the Vel knob determines how the click responds to velocity.

After Filter 2, the sound passes serially through two delay-based effects: The Beat Sync delay and the Diffuser Chorus. Note that extended versions of these excellent effects are available in the Fusion Reflections and Echomania ensembles. The Beat Sync delay's large Delay knob sets the delay time in sixteenth notes. The Div knob to its right divides the delay time to create dotted and triplet times, while the display above the Div knob shows the actual delay time. For instance, a Delay of seven sixteenths and a Div of two will produce a delay time of

3.5 sixteenths. The small Q switch turns quantization on and off in respect to modulation coming from the sequencer - yes, it is possible to sequence the delay time! You can find more information on the sequencer below. Smth sets an internal smoother on the delay time. Naturally, feedback sets the feedback level, while the HP P and LP P set the frequency of a highpass and lowpass filter in the feedback path. The delay time can be modulated with an internal LFO, with A (amplitude) controlling the amount of modulation, P (pitch) controlling the frequency, and L-R controlling the stereo spread.

The Diffuser Chorus consists of four stereo modulation delay. The main delay time is controlled by the Diff Dly knob. Diff sets the inaccuracy of the delay times to "spread" the sound. Like the Beat Sync delay, an LFO with similar controls modulates the delay time.

5.3.4 The Sequencer

Blue Matrix's sequencer consists of three areas where sequencer data can be drawn in - the uppermost handles the pitch of the notes in the sequence, the field below displays the trigger and velocity information of those notes, and the last one contains four independent tracks where additional modulation data can be stored and edited. There are a number of horizontal and vertical control bars for zooming, selecting the length of the sequence, and selecting the "edit" range for commands such as randomize, quantize, and copy/paste/insert.

Important Note!! As the complete sequencer is based on Reaktor's event tables, please note that the table data and snapshot data are independent of each other. The snapshots correctly store the position of all knobs relating to the sequencer, but they don't store any of the table data. Naturally, it is possible for each snapshot to have its own sequence, but you must take tare that the global Seq knob is set to the same number as the current snapshot. Otherwise, if different presets share the same table Seq, you could end up changing a table sequence and inadvertently changing the sound of all other snapshots that use the same table. If each snapshot has its unique Seq number, you will avoid this problem. Please note, however, that table data is excluded from the snapshot functions such as morph, randomize, and compare. To be really safe, if you spent a lot of time working on a sequence that you love, save the entire ensemble using the Save As... command.

Pitch-table: To the right of pitch edit-table are three vertical bars. The first is the vertical zoom/scroll bar. Clicking and dragging near the top or bottom of this bar will act as a zoom, while dragging in the middle of the bar scrolls the display. The next vertical bar sets the trigger threshold. Any notes below this threshold won't be triggered. The third bar sets the edit range. The edit range can be extended by clicking and dragging near the top or bottom, or the edit range can be dragged up or down by clicking and dragging in the middle of it. An additional, horizontal editor bar is provided to select notes in time. The copy/paste/cut/ insert functions to the right of the vertical bars do the usual functions (for the area defined by the horizontal and vertical edit bars), with a couple neat tricks. P Inc sets how much the pitch will be incremented when data is inserted or pasted, in semitones. If you wanted to transpose a figure by five

semitones, for instance, simply arrange the horizontal and vertical edit bars to cover the length of that figure, hit copy, then insert. Likewise, V inc sets the velocity increment of pasted or inserted data.

Randomizing and Quantizing data: In addition to copying and pasting the range denoted by the edit bars, you can also randomize data within this range. Simply click on Rand, and the edit areas will be filled with random pitches whose properties are determined by the Hold, Retrig, and N. On functions. Data in the edit range can also be quantized with the Quant button, with resolution set by Step. An undo function is available for copy/paste, randomizing, quantization, clear, and recording data into the table.

Gate table and Glide: Underneath the Pitch table is a table where you can draw in gate (note-on) events. The height of the event corresponds to velocity. Note that in Blue Matrix's sequencer, pitch and gate are separate. The sequencer still sends out pitch information even if there is no gate signal. The means that you can create custom glides. The Glide time is set with the small Glide knob to the left of the pitch table. Small values of glide mean the oscillator pitch reflects what's shown in the pitch table. Large glide values create a ramp, or smoothing, from one pitch to the next. In Leg (Legato) mode, then glide is only active for overlapping notes.

Modulation Table: The third table in the sequencer is a four channel modulation sequencer, where you can independently draw and edit four different modulations. You can route the modulations to synth parameters in the matrix modulation section. Similar to the pitch sequencer, to the right of the mod sequencer there are three vertical bars, each with a different function. The first one is a zoom/scroll bar. The second selects the vertical edit range, while the third bar selects which channel of modulation is shown in the display. You can either view and edit a single channel, or you can view (but not edit) all four channels simulta-neously. To switch between these two modes, simply click on the View button underneath the three vertical control bars. When there is a vertical grid shown, then you can draw in your modulation data. After drawing in a shape, click on View again and you will see the two-dimensional representation, where blue represents minimum values and red represents high values. In two-dimensional mode, you can view all four modulation channels simultaneously.

Global loop length and zoom controls: Underneath the modulation table are three horizontal global control bars that affect all tables. The first one is the edit-range bar, that is also linked to the edit range bar for the pitch table. The middle horizontal bar is the sequence loop length bar. Similar to the other controls you're now familiar with, this bar can be resized by dragging near the beginning or end, and can be moved by dragging in the middle. You can set the quan-

tize value for the loop bar with the Bar control to its left. The bottom-most bar is the horizontal zoom/range bar, which operates the same as all the other zoom/ range bars: dragging the beginning or end acts as zoom, and dragging the middle scrolls.

Recording data into the sequencer: It's possible to save MIDI data sent to the instrument, e. g. by a keyboard. The Rec> buttons at the left of the data fields select and activate the different sequencer parts for recording. Note that both upper data fields can only be recorded together (as MIDI events always contain both pitch and trigger information) while each modulation track can be recorded separately. The actual recording starts when the Rec button is pushed. If at this moment another region of the sequence is playing than the region selected by the horizontal edit bar, the W lamp will light up to mark the sequencer's waiting status; when the global read-out pointer enters the edit region, the R lamp is will light to show that the instrument is recording.

Two hints for using the tables:

1. If you accidentally draw into a table, you can always use the table's Undo command.
2. Use the global copy and paste commands to work on a copy of a sequence, or to build a new sequence up from an existing one. Set the horizontal edit bar (either the one under the pitch table or the modulation table, they're always linked) to the length of the entire sequence.

5.3.5 Modulation

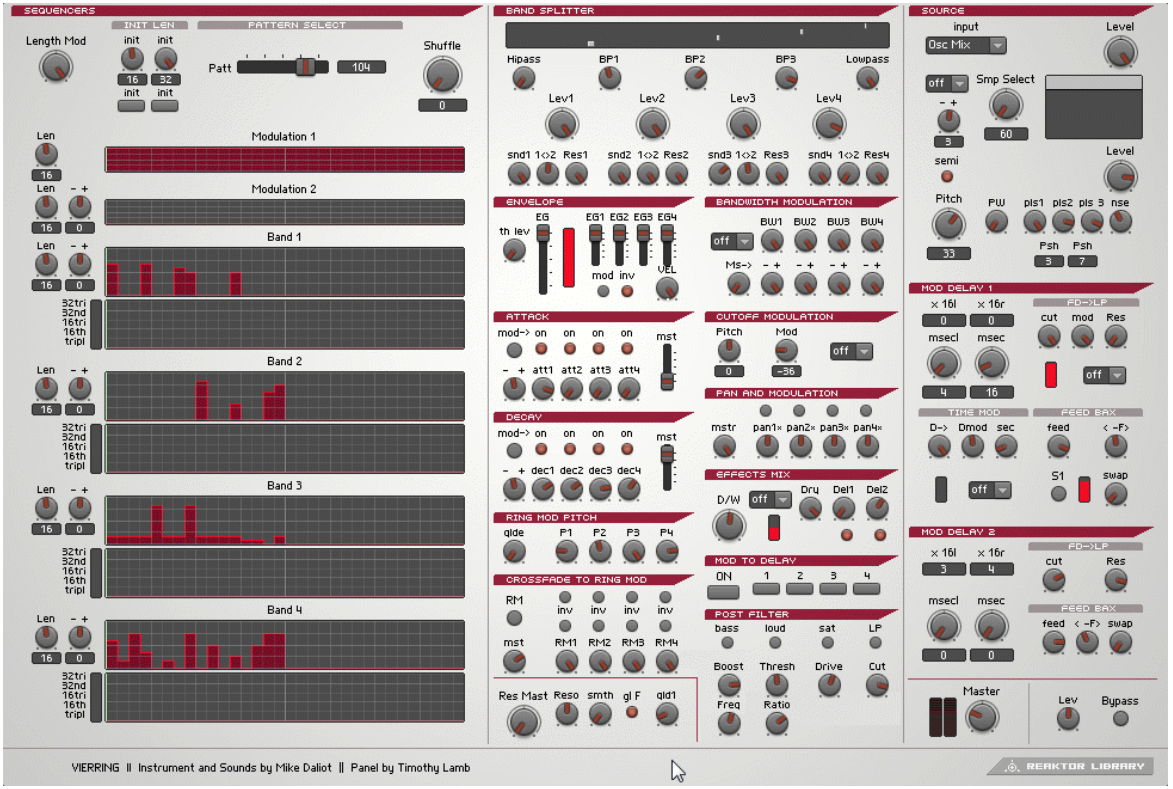
Blue Matrix features two identical standard ADSR envelopes and a single LFO that syncs to tempo. Both envelopes are velocity sensitive: Vel>A sets how much the velocity will affect attack time, while Velo sets how much velocity will control the amplitude of the entire envelope. The speed of the LFO can be set in musical note values, or freely. When Unit is set to 0, then the Frequency of the LFO is set in Hz. When Unit is set to any other value, the frequency of the LFO takes on the musical units set by Unit. For example, if Unit is 96 and frequency is 1, then the LFO will be set to the very fast speed of 1/96th note per cycle. The display underneath the Freq knob depends on how Unit is set: if Unit is set to 0, then the display automatically shows values in Hz, and if Unit is set to a musical value, then the Freq display will automatically update. Width and Phase control the shape of the LFO, which can be selected by the switch to the right. Key S determines if the LFO will synchronize to note-on events, as delivered by the sequencer.

All of Blue Matrix's modulation sources are routed in a comprehensive modulation matrix: The LFO, two envelopes, four modulation channels from the sequencer, and typical MIDI controls like mod/pitchbend/velocity can be routed to synthesizer destinations using the comprehensive modulation matrix. Switch to the "B" view of the synth to see the modulation matrix. You can easily switch to the B view by clicking the B button in the title bar of the synth. You can route the modulation to the following synth parameters: Oscillator pitch (osc 1, osc 2, or osc 1 + 2), osc 1 FM, osc 2 shape, osc ½ mix, filter cutoff (osc 1, osc 2, or osc 1

+ 2), filter 1 FM, filter resonance (1 or 2), envelope parameters, and even distortion, delay time, and delay feedback. In short, nearly everything!

The modulation sources (LFO, envelope, sequencer modulation, etc) are arranged vertically. The modulation destinations (synth parameters such as filter cutoff, oscillator pitch, etc) are arranged vertically. Simply activate the switches for the combination you'd like to modulate. Underneath each modulation column is a switch to turn the modulation on or off, and a knob to set the modulation amount.

5.4 Vierring



1.17 Vierring user interface

5.4.1 Introduction

Vierring is a creative rhythm generator where you can run any audio through and get percolating percussive patterns. Vierring divides the incoming audio into four frequency bands, and each band can be graphically sequenced for amplitude and ring modulation. Two delay effects enhance the output.

5.4.2 Structure and Signal Flow

Before sound enters the Vierring ring modulation engine, the filters in the Freq Shift section split up the sound into four bands. Hicut and locut will remove high and low frequencies respectively, while the three band pass (BP) knobs set the crossover points to create four frequency ranges.

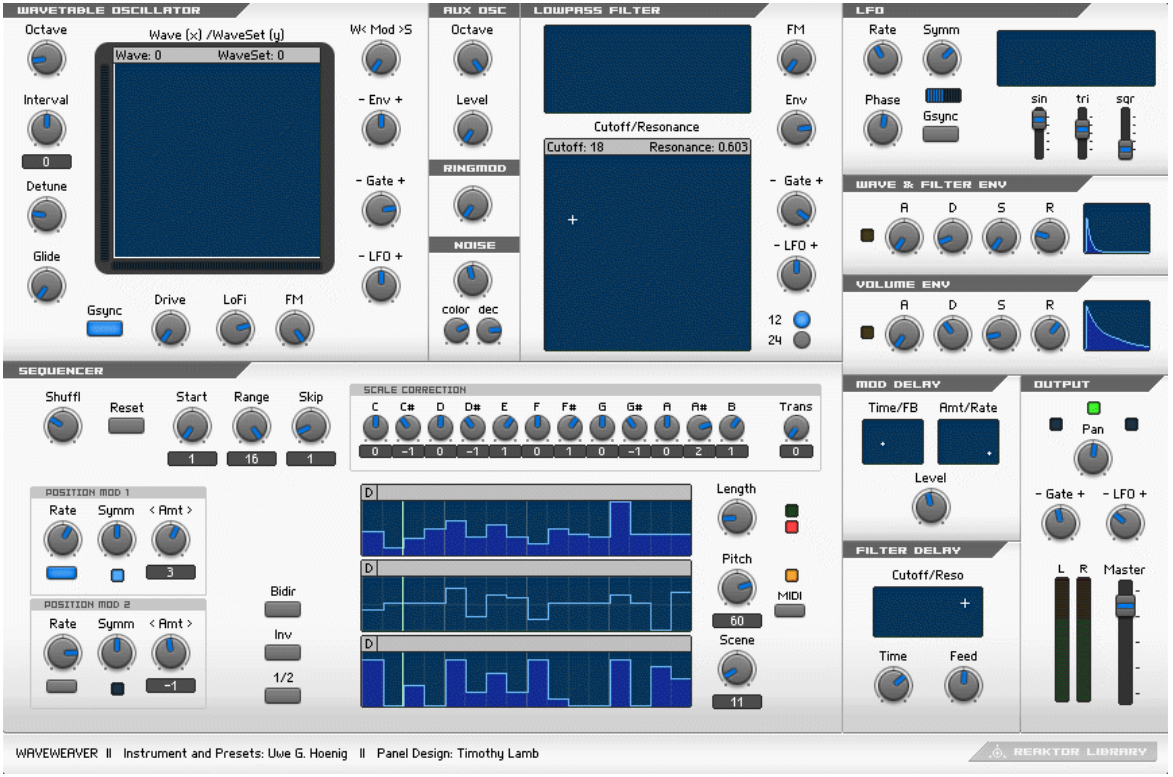
The controls for each of the four band are identical. Lev sets the volume of the band, and Pan controls the position in the stereo field. The BW and Res controls set the parameters for the band pass filter used to feed audio into the band. The RM button turns ring modulation on, and the RM knobs adjusts the amount of ring modulation. You can adjust the attack and decay characteristics of the trigger envelope with the att and dec knobs, respectively. The Length-64 parameter sets the pattern length of that band. Note that it's possible to have an 8-step lowband, and a 7-step midband, for instance!

With the event table to the right of the band controls, you can sequence he amplitude and ring modulation amount (top sequence) and the effect send (bottom sequencer). The 1<>2 and wetL knobs determine to which delay the signal to, and the volume of the wet signal. The snd and viaT2 knobs adjust the amount of delay send, and the amount of delay send as specified by the second table. Underneath the two tables is a small horizontal bar that shows the frequency range that that band represents.

In the Global section, you can set the active set of tables with the Pattern knob. The attack, dec, res, and BW knobs perform the same function as the similarly-labeled knobs in each band, but they act globally. Note that the filter bandwidth and the global filter frequency can be modulated by the graphical mod sequencer.

* You can play a sound through the built-in loop-player, through the realtime audio inputs, or you can process audio in realtime by using Reaktor as an effect plugin. Please check your Reaktor or Reaktor Session user's guide for helpful information.

5.5 WaveWeaver



1.18 WaveWeaver user interface

5.5.1 Introduction

The WaveWeaver ensemble uses wavetable synthesis to produce a range of subtle or harsh tones and rhythmic pads. The oscillator in WaveWeaver is actually several sets of short, looping soundfiles, each used as an oscillator waveform. The soundfiles in each wavetable can be can be blended into each other, and you can use an XY pad to simultaneously select among them

and modify the loop start point. A built-in sequencer with scale correction allows you to create some unique, shifting grooves. MoDelay and FilterDelay let you add some effects to your sounds.

5.5.2 Quick Start

WaveWeaver's wavetable, which can be extensively modulated and morphed in tone, feeds through a distortion circuit and a bit reducer before joining an auxiliary sine wave oscillator and a noise source at the 4-pole low pass filter. After the filter, the signal travels through the ModDelay and the FilterDelay. Along the way, an LFO, two ADSR envelopes, and the sequencer modulate the wavetable and filter.

The Wavetable Oscillator section, in the upper left corner, is where the controls for sound generation are located. There are actually two wavetables, which can be detuned from one another for a wider sound. You can crossfade between the various soundfiles of the waveset and set loop start points by moving the cursor in the XY pad. Soundfiles are located on the vertical, or Y, axis and the start is on the horizontal, or X, axis. The Octave knob to the left of the Wavetable pad lets you set the octave shift for both the main oscillator pair and the auxiliary oscillator. The Interval knob, immediately below this, controls the pitch shift, in semitones, of the second wavetable. The Detune knob allows you to fine tune the interval of the two oscillators. You can also set the amount of pitch glide for the oscillators using the Glide knob.

The four knobs to the right of the XY pad control the intensity and polarity of different modulation sources that can have an effect on the wavetable. At the top, the W< Mod >S knob sets the amount of position modulation within the waveset when turned to the left, and modulates the selection of the soundfile when turned to the right. Below this knob, the - Envelope +, - Gate +, - and LFO + knobs control the mixing of the various modulation sources. You could, for example, mix the negative output of the filter envelope with the positive output of the LFO to modulate soundfile selection. The signal from the wavetable passes through a distortion circuit and a bit reducer. The Drive and LoFi knobs control control the amount of processing by these effects. In addition, the wavetables can be frequency modulated by the auxiliary sine wave oscillator, for extreme FM sounds. You can set the amount with the FM knob. Pressing the Gsync switch causes the two wavetable oscillators to trigger in sync with one another on gate signals. Otherwise the oscillators start at their current free-running position. Playing with the Drive, LoFi, and FM controls can yield a variety of sounds ranging from spiky to metallic to liquid.

Before reaching the filter, the wavetables can be mixed with the aux sine wave, which features tuning relative to the main oscillators in octaves, and a noise source with adjustable filter color and envelope decay. The sine oscillator can ring modulate the wavetables, and the resulting signal can also be mixed into the filter input with the Ringmod level knob.

All of these sources are fed into a 4-pole low pass filter, whose cutoff frequency and resonance can be set with the Filter XY pad. Much like in the Wavetable section, the knobs along the right edge of the Filter XY pad control the amount of positive or negative filter modulation from the envelope, LFO, and gate signals.

These modulation sources reside on the right side of the ensemble. The LFO allows you to mix sine, triangle, and square wave shapes to achieve complex modulations. LFO frequency and waveform symmetry can be controlled with the Rate and Symm knobs, respectively. You can also set the start point of the LFO wave when it is retriggered using the Phase knob. When the Gysinc button is switched on, the LFO is retriggered with each gate signal; it restarts at the point in the waveform set by the Phase knob.

An ADSR envelope modulates the wavetable and/or the filter frequency. Another ADSR controls the amplitude contour.

The final elements in the signal chain are the MoDelay and the FilterDelay. The MoDelay produces dimensional chorus and flanging sounds with its delay time LFO. The Time/FB XY pad allows you to smoothly adjust delay time and feedback. The Amt/Rate XY pad lets you set the amount and frequency of the LFO modulating delay time.

The FilterDelay is a tempo-synced delay with a resonant low pass filter to alter the sound output. You can change the filter cutoff and resonance with the Cut/Res XY pad. Since the delay time is automatically synced to the system clock, you can use the Time knob to change delay time by beat divisions.

5.5.3 The Sequencer

The three-part sequencer is started and tempo-controlled by the global MIDI clock. Swing can be added with the Shuffle control. You can adjust the sequence start point and loop length with the Start and range knobs. Pressing the Reset button in the Clock section forces the sequence to reset itself to the position determined by the Start knob. The Skip knob sets the number of steps in the sequencer's event table left out after each time the sequence advances.

The Bidir and Inv buttons change the order in which the sequence steps play back. Bidir causes the sequence to run forward, then backward, while Inv inverts the playback order. These controls are extremely useful for nondestructively altering the character of a sequence in performance, for example. You can also modulate the current playback position of the sequencer over time with the two Position Mod LFOs. Each can have the modulation amount inverted and have control over LFO waveform symmetry with the Symm knobs.

The top sequencer row controls the length of each note. The middle row lets you program a pitch sequence, while the bottom row sets gate value, which is wired to effect velocity, or volume, for each step. The pitch can be offset to transpose the sequence using the Pitch knob. Engaging the MIDI switch next to Pitch allows you to transpose the base pitch of the sequence and play chords with a MIDI keyboard. You can use these controls to create complex, modulating arpeggiations. The Scale Correction section gives you the option of mapping pitch data onto a specific scale, for example a c major or pentatonic scale, or a Dorian mode. This can create subtle and moving effects when you transpose the sequence pitch using a keyboard.

6 Sequencers

6.1 Spiral



4.28 Spiral user interface

6.1.1 Introduction

Spiral was initialy developed for REAKTOR ANIMATED CIRCUITS. Purely a MIDI instrument, it does not create any sound by itself: it is used as drone-like sequencer for other instruments, and its generative algorithms create fractal melodies that merge mathematics with music. To understand the way Spiral works have a look at the picture above. It shows the Ensemble's panel within REAKTOR, focused on a central circular display. Several objects are located around the display's center, each with a specific distance. Each object has a specific speed that determines how fast it rotates around the center.

The rotating objects turn into music by triggering MIDI notes. Each time an object passes a virtual pick-up line at the top of the display it creates a note according to the object's specific pitch, and velocity settings. By attaching any sound generator that listens to MIDI events to Spiral, you can create any type of sound with this experimental sequencer. Nearly all other controls you see within the panel are used to control how pitch, velocity and rotation speed are assigned to the various objects. There is a pitch correction mechanism (for use if you prefer a minor scales for example) Also there is a more general section that determines the rotation's main speed as well as quantization to a MIDI clock (so that the melody remains synchronized to other MIDI events within REAKTOR).

6.1.2 Using Spiral

There are four main areas within the panel: Pitch Correction on the left side, Spiral, Pitch and Velocity on the right, with the central display and Clock positioned in the center.

Clock

The controls of the clock section determine in which way Spiral reacts to an incoming MIDI clock signal. This signal is created by REAKTOR. As the same clock signal is sent to all instruments, it automatically synchronizes their rhythmical activities.

- **Speed**: Controls how fast the objects rotate around the center, correlating to the MIDI Clock. It influences the rotation speed of all objects simultaneously and can therefore be used to increase or decrease the general density of notes: The faster an object rotates, the earlier it triggers a new note by passing the virtual pick-up line.
- **Quant**: Determines to which rhythmical grid all notes are quantized, e.g. a sixteenth note. Whether quantization is active or not depends on the Depth control.

- **Depth**: Specifies to which extent trigger events are delayed in order to match the rhythmical quantization grid. At a minimum value, a note is triggered immediately when an object passes the virtual pick-up line. At a maximum value, the trigger signal is delayed until the MIDI clock reaches the next full quantization value, e.g. the next full sixteenth note. This control can become very effective when using values just below the maximum: The rhythmical grid will become audible, but it will be interpreted with slight inaccuracies due to human interpretation and perception.

Pitch Correction

With the controls of this section, the generated notes can be re-organized so that they fit a musical scale. As explained above, each rotating object has a specific pitch value which is used when triggering a note. These pitch values can be adjusted within the Pitch tab (explained below), yet these values come from the chromatic scale: They contain all twelve pitch classes (c, c#, d, d# and so on). Often, Spiral's melodies become more musically usable when restricting the pitch classes to a traditional scale like the major scale, where only seven distinct classes are used (c, d, e, f and so on).

- **Scale**: Determines the scale to which all pitch values are mapped. Major and Minor are commonly known, but all others can also be used when the actual scale is undefined. They can be treated as colors that are blended onto the melodies.
- **Root**: Specifies the root pitch of the scale. For example if Scale is set to major, a value of "C" will result in a C major scale (C, D, E, F and so on). Consequently, C# will use a C# major scale (C#, D#, E#, F# and so on).

Spiral, Pitch, Velocity

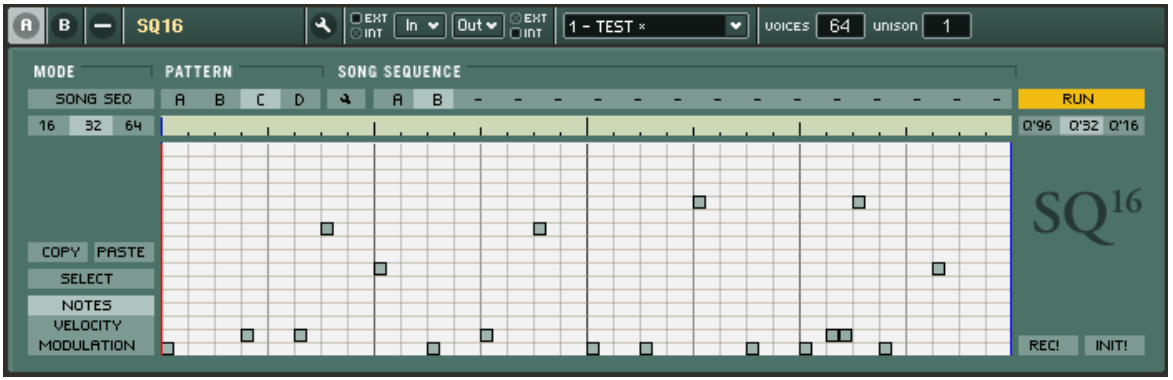
These three tabs control the values for rotation speed (Spiral tab), pitch and velocity are assigned to each object. Within the Spiral tab the number of rotating objects can be controlled as well.These values are the main controls to manipulate the fractal melodies of the Spiral ensemble. Note that their usage is simpler than their detailed explanation. If the following descriptions don't help, just try experimenting!

- **Spiral Objects**: Controls the number of objects rotating around the display's center. The higher the number the higher the density of notes triggered by the Spiral sequencer.

- **Spiral Spiral**: Controls how much the individual objects differ in speed. The higher the value, the faster fast objects will rotate while slow objects will remain slow. This can be used to increase the "chaos" of a melody. However as it strongly interacts with the Step control it needs to be used experimentally, together with the other parameters.
- **Spiral Step**: Controls the difference in speed between two adjacent objects. Higher values result in more complex melodies. Technically speaking, all objects are aligned along a line. The Step parameter defines how much faster the next object within this line is, in relation to its predecessor. However, at a maximum level (which is defined by the Spiral parameter) the Step value is inverted, so that the next object is slower than its predecessor. At the minimum speed level, the Step value is inverted again, so that objects become faster again - and so on, until all objects have a specific speed. High Step levels increase the frequency of those inversion points, and thus the melody's variation.
- **Spiral Shift**: Controls the initial object's speed from which all other speed values are derived, dependent on the Step control. It can be used to create a variation of an existing melody.
- **Pitch Pitch**: Controls the pitch range of the melody. High values result in a very widespread melody, while low values concentrate all objects' pitch values around the central pitch.
- **Pitch Spread**: Controls the pitch range of the melody. High values result in a very widespread melody, while low values concentrate all objects' pitch values around the central pitch.
- **Pitch Step**: Controls the difference in pitch between two adjacent objects. Higher values result in more complex melodies. All objects are aligned along a line and The Step parameter defines by how many semitones the next object's pitch is above its predecessor's. At a maximum level (which is defined by the Spread parameter) the Step value is inverted so that the next object has a lower pitch than its predecessor. At the minimum pitch level (also defined by the Spread parameter), the Step value is inverted again, so that objects become higher. This process continues until all objects have a specific pitch. High Step levels increase the frequency of those inversion points and therefore the melody's variation also.
- **Pitch Shift**: Controls the initial object's pitch (from which all other pitch values are derived dependent on the Step control) in relation to the central pitch. It can be used to create a variation of an existing melody.

- **Velocity Velocity**: Controls the central velocity of the melody. The velocity values of all objects are grouped around this center. Increase or decrease this value to change the whole melody's volume.
- **Velocity Spread**: Controls the velocity range of the melody. High values result in a very widespread velocity range, while low values concentrate all objects' velocity values around the central velocity value.
- **Velocity Step**: Controls the difference in velocity between two adjacent objects. Higher values result in more complex melodies. All objects are aligned along a line and The Step parameter defines how much louder the next object is in relation to its predecessor. At a maximum level (which is defined by the Spread parameter) the Step value is inverted, therefore the next object is quieter than its predecessor. At the minimum velocity level (again defined by the Spread parameter), the Step value is inverted again so that objects become louder. This process continues until all objects have a specific velocity. High Step levels increase the frequency of those inversion points and therefore the melody's variation also.
- **Velocity Shift**: Controls the initial object's velocity (from which all other velocity values are derived dependent of the Step control) in relation to the central velocity. It can be used to create a variation of an existing melody.

6.2 SQ16



4.29 SQ 16 user interface

6.2.1 Introduction

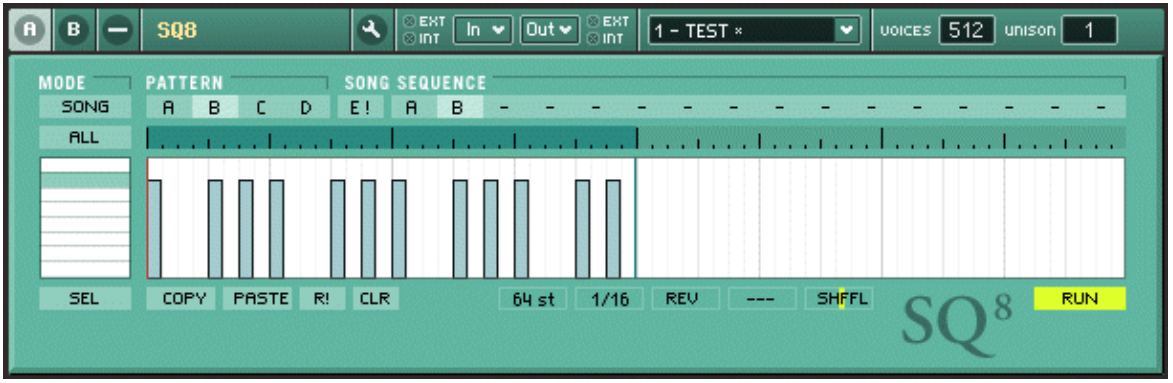
The SQ16 sequencer delivers classic step-sequencing in a very useable package. It features 16 notes tracks with velocity control plus additional modulation tracks, a song mode, and the ability to record incoming MIDI notes.

6.2.2 Details

Section	Control	Function
Control	Song Seq	Toggles song mode on and off. When on, the pattern defined under [Song Sequence] is played. When off, the currently selected pattern is played and looped.
	Zoom Level	Choose whether 16, 32, or 64 steps are displayed. This has no influence on the notes played.
	Notes	Displays the notes track of the sequencer. Click the notes grid to create notes, right-click to delete notes (ctrl-click for Mac users). Note length depends on the quantization settings on the top right side of the sequencer.
	Velocity	Displays the sequencer's velocity track. Each note in the notes grid has a velocity bar in the velocity track. Drag with the mouse to change the levels.
	Modulation	Displays the sequencer's modulation track. Enter the desired modulation steps by dragging with the mouse. Quantized in 16ths.
Pattern	A/B/C/D	When not in song mode (see [Song Seq]), the selected pattern is played and looped.
Song Sequence	Song Edit	The edit button enables you to assign patterns to the [Pattern Slots].
	Pattern Slots	When [Song Edit] is active, click into a pattern slot and drag the mouse up / down to select the desired pattern.
Global controls	Loop Bar	The brown bar above the sequencer grid represents the loop region. Right-click (ctrl-click for Mac users) to set length and left-click and drag to move.
	Run	Switches sequence playback on or off.
	Q'96 / Q'32 / Q'16	Quantization setting for the note-length resolution. Q'96 means 96th resolution, Q'32 is 32th resolution, and Q'16 is 16th resolution.

Section	Control	Function
	Copy	Copies the currently selected notes or modulation events into the clipboard.
	Paste	Pastes the pattern clipboard into the current pattern.
	Select	Toggles select mode on or off. When on, you can select multiple notes in the notes track by clicking or by dragging a square around them. You can also select a range of the modulation track.
	Rec !	Switches on note recording via MIDI in.
	Init !	Deletes all pattern notes and resets the modulation track events to zero. (You need to double click on it!)

6.3 SQ8



4.30 The SQ 8 user interface

6.3.1 Introduction

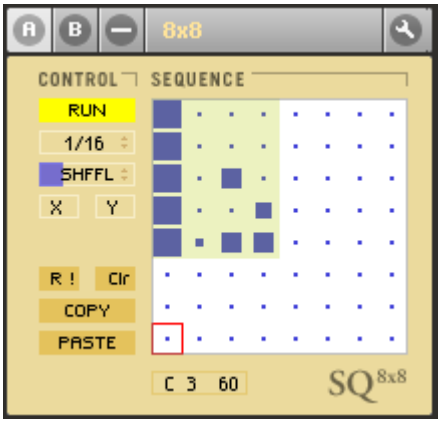
The SQ8 is your standard building block for rhythmic step-sequencing. It sports a clean interface: 4 patterns with 8 tracks (consisting of 64 steps each). You also get variable looping, shuffle, reverse play, and multiple viewing options. On top of that, you can chain 16 patterns together into a song.

6.3.2 Details

Section	Control	Function
Mode	Song Seq	Toggles the song mode on and off. If on, the pattern sequence defined under [Song Sequence] is played. If off, the currently selected pattern is played and looped.
Pattern	A/B/C/D	When not in song mode (see [Song Seq]), the selected pattern is played and looped.
Song Sequence	Song Edit	The edit button enables you to assign patterns to the [Pattern Slots].
	Pattern Slots	When [Song Edit] is active, click into a pattern slot and drag the mouse up / down to select the desired pattern.
Global controls	Pattern view / Track view	Click on [All] to see the complete pattern with all tracks. Click on any button to the left of a track to view the track exclusively. In track view you can also change the velocity of the individual notes.
	Loop Bar	The darkish green bar above the sequencer grid represents the sequencer's loop region. Right-click (ctrl-click for Mac users) to set length and left-click and drag to move.
	Notegrid	Click into the grid to add or delete note events.
	Sel	Toggles note select mode on or off. When on, you can select an area of the note grid to be cleared, copied from, or pasted to. This works in all viewing modes.
	Copy	Copies the content of the current pattern.
	Paste	Pastes the pattern clipboard into the current pattern, overwriting all events.
	Rec !	Switches on note recording via MIDI in.
	Clr	Deletes all selected notes of the pattern.
	Zoom Level (16 st, 32 st, 64 st)	Click and drag mouse up or down to zoom in and out of the currently displayed pattern.

Section	Control	Function
	Clock divider (1/6, 1/8, 1/12, 1/16, 1/24, 1/32)	Chooses between different clock divisions. This speeds up or slows down the pattern but retains a metric relationship to the original speed. You get original speed with the 1/16 setting.
	Rev	Toggles reverse play on and off. The direction is reversed by pattern mirroring.
	Stepshifter	This menu determines the playback mode. --- is normal, 1324 and 1432 let the steps swap their positions, <?> plays in random direction, <??> randomly jumps to the previous or next step, ???? jumps to a completely random step.
	Shffl	Shuffle function. Click and drag mouse up or down to select the amount of shuffle.
	Run	Starts and stops the sequencer.

6.4 SQ 8x8



4.31 The SQ 8x8 user interface

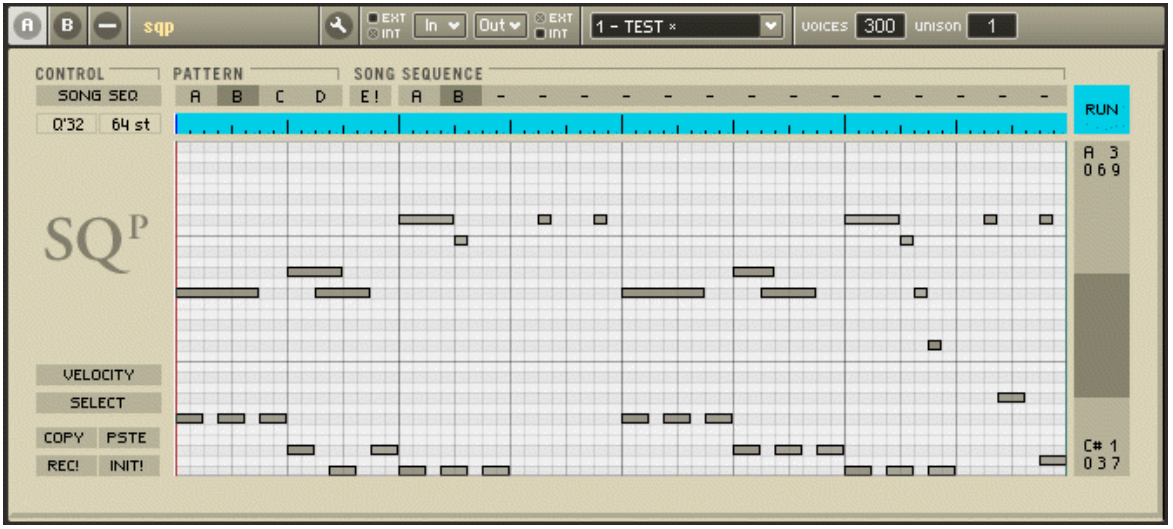
6.4.1 Introduction

The SQ 8x8 is a small step-sequencer with a twist. You put events in a grid and drag a rectangle around a group of them by right-clicking and dragging (ctrl-click when you're on a Mac). This rectangle defines the sequencer's loop area, controlling what gets played line-wise. You can change this area in realtime. Think of it as a two-dimensional loop-bar. Some nice realtime step-shifting and shuffle features are also part of the package.

6.4.2 Details

Control	Function
Mute	Mutes the sequencer output.
Notegrid	Click the grid to add or delete note events. Click and drag the mouse up or down to change velocity. Right-click (ctrl-click for Mac users) and drag to define the loop-area.
Clock divider (1/6, 1/8, 1/12, 1/16, 1/24, 1/32)	Choose between different clock divisions. This speeds up or slows down the pattern but retains a metric relationship to the original speed. You get original speed with the 1/16 setting.
Shffl	Shuffle function. Click and drag mouse up or down to adjust the amount of shuffle.
X - playback modes	-- - walk normally X - +/- random step in X direction XX - +/- random step in whole X row
Y - playback modes	-- - walk normally Y - +/- random step in Y direction YY - random in whole column
R!	Randomizes the current loop area.
Clr	Clears the current loop area.
Copy	Copies the content of the current loop area.
Paste	Pastes the pattern clipboard into the current loop area, overwriting all events.

6.5 SQP



4.32 The SQP user interface

6.5.1 Introduction

The SQP is a piano roll-style sequencer covering a very wide midi note-range. You can enter notes via the mouse or record incoming MIDI notes. If you want to input longer events with the mouse, just click and drag the start or end of an existing note. Move events by clicking and dragging them around. When [Select] is on you can move selected events as a group.

6.5.2 Details

Section	Control	Function
Control	Song Seq	Toggles song mode on and off. If on, the pattern sequence defined under [Song Sequence] is played. If off, the currently selected pattern is played and looped.
	Quantization	Controls the quantization of note events. Choose between quantization in 16th, 24th and 32th. You can also switch off quantization.
	Zoom Level	Choose whether 16, 32, or 64 steps are displayed. This has no influence on the notes played.
	Velocity	Displays the sequencer's velocity track. Each note in the notes grid has a velocity bar in the velocity track. Drag with the mouse to change the levels.
	Select	Toggles select mode on and off. If on, you can select note events by clicking them or by dragging a square around them.
	Copy	Copies the currently selected events.
	Pste	Pastes the pattern clipboard into the current pattern.
	Rec !	Switches on note recording via MIDI in.
	Init !	Deletes all note events of the pattern. (Needs to be double clicked!)
Pattern	A/B/C/D	When not in song mode (see [Song Seq]), the selected pattern is played and looped.
Song Sequence	Song Edit	The edit button enables you to assign patterns to the [Pattern Slots].
	Pattern Slots	When [Song Edit] is active, click into a pattern slot and drag the mouse up or down to select the desired pattern.
Global controls	Loop Bar	The blue bar above the sequencer grid represents the sequencer's loop region. Right- click (ctrl-click for Mac users) to set length and left-click and drag to move.
	Run	Starts and stops the sequencer.

6.6 SQX



4.33 SQX user interface

6.6.1 Introduction

This ensemble enlarges the collection of sequencer instruments supplied with Reaktor 5. It provides four tracks designed to trigger drum sounds. Each track is bound to a maximum length of eight quarters; however – this is what makes the instrument special - each quarter can be subdivided into an individual number of smaller steps, giving access to complex, polyrhythmic beat programming.

The ensemble also contains an additional song sequencer that can be used to automatically recall snaps of the sequencer to build large-scale patterns. Finally, a small but versatile percussive tone generator demonstrates the sequencing capacities.

6.6.2 SnapSeq



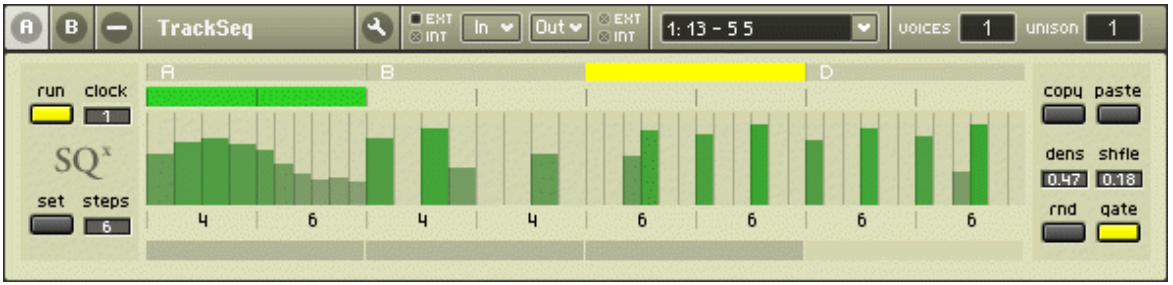
4.34 SnapSeq user interface

The snap sequencer recalls a maximum of 16 different snapshots of the dependent instruments TrackSeq and ToneGen (see below). They are recalled according to the line defined by the steps 1 to 16. Upon a MIDI start signal the playback begins at the step defined by the Start control and loops back to this point after a number of steps to be set with the Length controller. The duration of each step is regulated by the Unit control. The Run button switches the snap sequencer on or off. The sequencer is synchronized to the MIDI songposition; therefore, there has to be a MIDI clock signal (e.g. the global Reaktor clock) to make this instrument work.

The SnapSeq recalls its own snapshots. By doing so, however, both other instruments are also controlled: although they are complete instruments with their own snapshot management (that can both be used without SnapSeq), in this ensemble they are placed within the SnapSeq instrument‘s structure; thus they are slaved to the master snapshot management of SnapSeq. This also enables the feature to create a set of sequences and a set of drum sounds independently as snapshots of the subordinated instruments. A variety of combinations can be saved as snapshots of the SnapSeq and can then be recalled by the snap sequencer.

Control	Function
Run	Switches the snap sequencer on or off. If on, MIDI clock signals are recognized.
Unit	Defines the length of each step as a multiple of bars. Each bar is made up of 96 MIDI clock ticks, interpreted as 96th notes: a bar represents four quarters. This length is divided by the lower value of the control, the result is multiplied by the upper value. For instance, a setting of 3 (in the upper field) and 2 (in the lower one) results in dividing the bar by two – each step will last for two quarters – and then multiplying it with five, so that, finally, each step of the snap sequencer will last for six quarters.
Step 1 to 16	Controls the sequence of recalled snapshots.
Start	Controls the step at which the playback begins upon a MIDI clock start event.
Length	Defines the length of the playback loop. It adjusts the number of steps after which the playback jumps back to the snap controlled by Start and begins again.

6.6.3 TrackSeq



4.35 TrackSeq user interface

This is the main part of the ensemble. Each of the four parallel tracks (named A to D) triggers one of the four channels of the ToneGen instrument (also called A to D). A track is made up by eight parts normally interpreted as quarters (this can be changed by the Clock control); the Loop control above the main edit area defines which parts are played back when the MIDI clock is running. Each part can contain three, four, six or eight steps, adjustable for each part individually by the Subdivision controls. A step's bar height represents the velocity of this step that can be used inside the ToneGen instrument to modulate the ensemble's sound.

There are several additional edit features. The Copy and Paste buttons can be used to duplicate the current loop area at a different position or within a different track. The Randomize button sets all steps of the current loop area to random velocities; if the Gate button is switched on, only steps that already have a velocity above zero are randomized to preserve the rhythmical structure. The Set button sets all parts of the current track to the selected division factor. Except the Run, Copy and Paste buttons all controls are bound to one of the four tracks and can be set to individual values.

Control	Function
Run	Switches the track sequencer on or off. When on, MIDI clock signals are recognized. This is independent of the snap sequencer's Run control.
Clock	Controls the speed at which the track is played back. A value of one represents normal speed: each part is interpreted as a quarter note according to the global MIDI tempo. Lower values slacken the tempo (at a value of 0.5 the sequence is played at half speed, i.e. each part lasts for two quarters), higher values accelerate the read-out (at a value of 2 each part represents a eighth note).
Set /Steps	Pushing the Set button resets all parts of the track to the subdivision adjusted by the Steps control.
Track Select	Selects one of the four tracks named A to D. Each track is hard-wired to the respective channel of the ToneGen instrument.
Loop Control	Controls the length (right mouse button) and position of the loop (left mouse button). This area is played back when the sequencer is working; additionally, it defines the area that is modified by the Copy, Paste and Randomize buttons.
Sequence Edit	Edits the main sequence. Each step triggers a sound of the ToneGen instrument if its velocity is higher than zero. The steps' velocity can be used to shape the respective channel's sound. A yellow bar represents the current read-out position during playback.
Subdivison	Sets the number of steps available per part. As the length of each part is fixed (depending on the Clock control), a higher number of steps also accelerates the read-out, e.g. at a Clock value of one a Subdivison value of three interprets each step as eighth triplet, a value of eight results in 32th notes.
Trigger Display	Shows the trigger signals that are sent to the four channels of the ToneGen instrument.
Copy	Copies the content of the current loop area to an internal buffer. This buffer is not bound to a specific track and can also be used to copy between them.
Paste	Pastes the content of the internal buffer to the current loop area. This buffer is not bound to a specific track and can also be used to copy between them.

Control	Function
Shuffle	Sets the amount of delay applied to each second step. At Subdivision values of four this results in standard swing shuffle; at other Subdivisions this produces more experimental rhythmic shifts.
Randomize	Sets the steps within the loop area to randomize velocities. This function uses Density and Gate as parameters.
Density	Controls the density of the rhythmic pattern generated when pressing the Randomize button.
Gate	If switched on, only steps with a velocity above zero are randomized; if off, every step within the loop area is set to new velocity.

6.6.4 ToneGen



4.36 ToneGen user interface

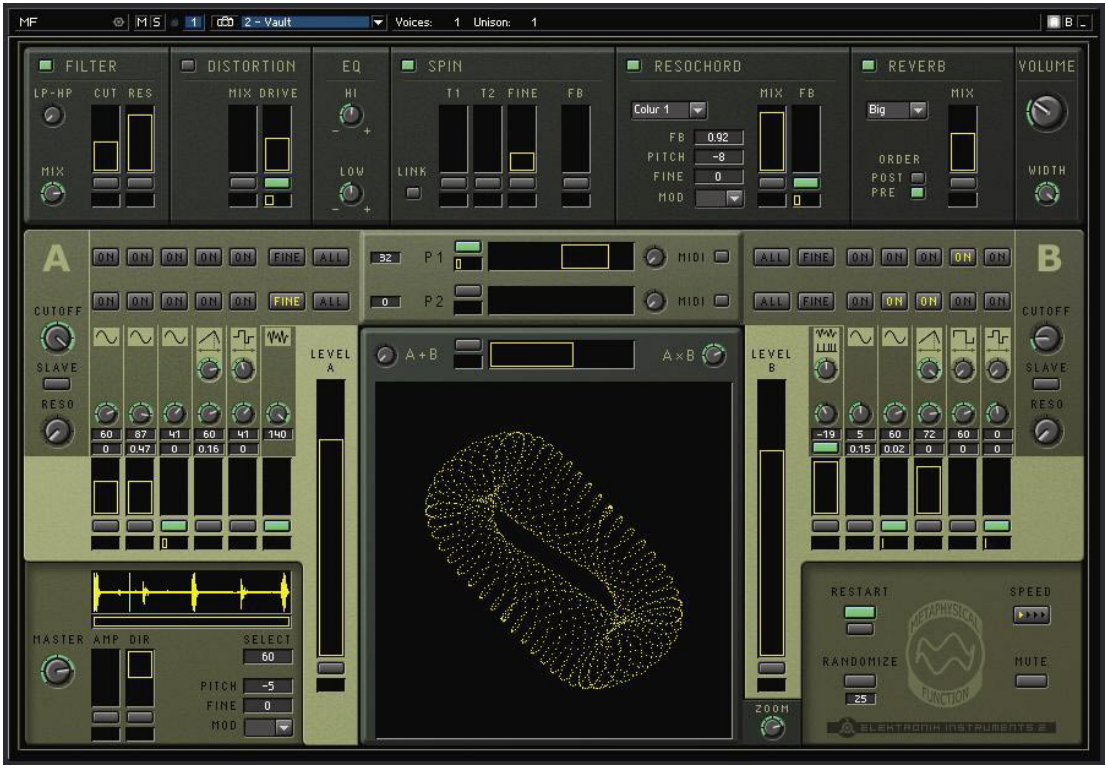
This instrument audibly illustrates the programming capacities of the track sequencer; however, its sound quality exceeds that of a mere demonstration instrument. It consists of four identical channels, connected to the four tracks of the TrackSeq instrument. Each channel contains (on the instrument's panel from top to bottom) a trigger envelope; a noise generator whose signal is shaped by the trigger envelope; a high quality resonant filter (based on Reaktor's Core Technology) attracted by the envelope/noise signal; and, finally, an output section. Each channel's sound can additionally be routed to a second stereo output where further effect instruments can be chained.

Section	Control	Function
Envelope	Decay	Sets the decay length of the envelope triggered by the TrackSeq instrument. After this period, the Release time starts; thus, the envelope is divided into two parts with individual duration and slope.
	Release	Sets the length of the release that starts after the decay period of the envelope has ceased. Thus, the envelope is divided into two parts with individual duration and slope.
	Velocity	Switches on or off the influence of the trigger signal's velocity on the envelope's amplitude.
Noise	Noise	Sets the amount of noise that is mixed to the envelope signal. Turn to the left for no noise and the pure envelope signal; turn to the right for a noise signal shaped by the envelope. This signal is used to attract the resonant filter, so different amounts of noise result in very different resonance sounds.
	Color	Shapes the filtering of the noise; in each Filter Mode this control cross-fades between two characteristic settings of the filter. This control can be modulated by the trigger signal's velocity (see Velocity).
	Filter Mode	Selects one of three filter types that can be used to shape the noise generator's output. A fourth entry turns the noise signal off; the Noise control then only dampens the envelope signal instead of mixing it with a noise sound.
	Velocity	Controls the amount of influence of the trigger signal's velocity on the Color control.
	Invert	Inverts the polarity of modulation applied to the Color control by the trigger signal's velocity.
Resonator	Resonance	Sets the amount of self-oscillation (i.e. resonance) of the filter.
	Frequency	Sets the frequency of the filter; at high Resonance settings, this frequency is audible as a distorted sine sound.
	Filter Mode	Cross-fades between a lowpass (at the left), a bandpass (at mid position) and a highpass filter's output.
	Envelope	Sets the amount of modulation applied to the Frequency control by the trigger envelope.
	Invert	Inverts the polarity of modulation applied to the Frequency control by the trigger envelope.
Output	Saturate	Thickens the channel's signal, i.e. quiet sounds are amplified and loud ones are slightly distorted.

Section	Control	Function
	Gain	Sets the amplification of the channel within the output signal's mix.
	Pan	Controls the channel's position within the stereo panorama.
	FX	Controls the routing of the signal. At the left, the signal is sent to the normal output ports of the instrument; at the right the signal is sent to the additional effect output.
	Mute	Mutes the channel's signal in the output mix.
FX	Level	Sets the level of the effect signal.
	Mute	Mutes the effect signal.
Out	Level	Adjusts the main output level.
	Mute	Mutes the main output signal.

7 Sound Generators

7.1 Metaphysical Function



9.3 Metaphysical Function user interface

7.1.1 Introduction

More than 'just another' synthesiser, Metaphysical Function offers a truly engaging audio/visual experience. By taking full advantage of REAKTOR's complete freedom to innovate, experiment and express, it offers a sound design tool quite unlike any other, with a vast and unique sonic palette ranging from beautiful soothing tones to dark, disturbing soundscapes.

7.1.2 Quick Start

Metaphysical Function constantly generates sound from the moment the ensemble is opened, until it is closed. Pressing notes on your MIDI keyboard does not stop or start notes, or control pitch in the way you might expect.

Instead, the key to controlling Metaphysical Functions unique sound lies with the 32 automatable faders, which control a variety of parameters, including the amplitude and pitch of the oscillators, and the settings of the FX units.

Each fader has two modes of operation – static or automated. While static simply operates like a traditional fader, automated requires some explanation. To enable automation mode, press the button directly beneath the slider. The sliders will now begin playback of the last recorded automation sequence. In automation mode, the playback position is displayed underneath automation button. When it reaches the end of the sequence, it will loop back to the beginning.

The playback position display also functions as a recording arm button. Click it once to enter recording mode – the display will flash. Now place the mouse cursor over the fader, and while depressing the mouse button, move the mouse cursor. Notice that the position indicator will stop flashing and will instead display the recording position as a proportion of the available recording time. When you have finished recording, release the mouse button. The position indicator will return to recording arm state and resume flashing. Click it once to return to playback mode – the newly recorded sequence will now commence playback.

Two of the automated faders, labelled P1 and P2, are dedicated to pitch modulation. They can be assigned to any or all of the oscillators using the pitch modulation matrix, which is positioned above section A and B, either side of the P1 and P2 faders. Click the **All** button to route the fader to all oscillators, or assign them individually by clicking on the buttons above the oscillator icons. Automation data is saved and recalled with snapshots as with other knob and fader settings.

7.1.3 Details

Signal-flow overview

Metaphysical Function features two independent signal generation sections named A and B. Section A features three sine waves, a triangular wave, a bipolar-pulse wave, and a lowpass-filtered noise generator. Section B is composed of two sine waves, a triangular wave, a pulse wave, a bipolar pulse wave and a two-mode noise generator.

The oscillators in each section are summed and then amplified, and then passed through a 12db filter. The output of the section A and B filters are then combined, both additively and by ring-modulation. The mix section is used to cross-fade between the summed signal and the ring-modulated signal. Metaphysical Function also features a sampler module, the output of which is mixed with the combined output of sections A and B.

The signal is then passed through the Master Filter, the EQ unit, the Spin module, and then the Resochord and Reverb effects. The signal is then amplified according to the main volume control. The main XY display visualises the post-Spin output.

Sections A, B, and Sampler Module

Control	Function
Oscillator Level, Pitch and Shape	The amplitude of each oscillator can be controlled by the row of automatable faders in each bank. Above the amplitude faders are knobs to define the base pitch of each oscillator – the upper knob sets the pitch in semitones and the lower knob is for fine adjustment. For the noise oscillators, the knob determines lowpass-filter cutoff. Additionally, the shape of certain oscillators (triangle and pulse) can be defined.
Filter	The oscillators are summed and passed through a 12db lowpass filter (controlled by the **Cutoff** and **Reso** knobs). When the **Slave** button is depressed, the cutoff knob is overridden by the master filter cutoff. (The main filter is located in the FX section at the top left of the panel).

Control	Function
Section Level	The overall **Level** of each section can be controlled and automated using the large faders either side of the waveform display.
Sampler module	Samples can be selected using the **Select** knob, and playback pitch can be defined by the **Pitch** and **Fine** knobs. Either P1 or P2 can be routed to the sample pitch via **Mod**. Loop range can be defined using the bar below the sampler

Pitch Routing Matrix

Section/Control	Function
P1 and P2	These 2 faders can be used to control the pitch of the oscillators, the sampler and the Resochord effect. Movements can be recorded by mouse movements as with the other sliders, but also via **MIDI** input by enabling the MIDI button. Glide time can be set with the adjacent knobs.
Individual and All buttons	To the left and right of the central P1 / P2 faders are controls for routing them to the oscillators in section A and B respectively. Click the **All** button to route to all oscillators in the section simultaneously, or click on the **Individual** buttons to assign oscillators one-by-one.

Mix section

Section/Control	Function
Mix controls	The central fader crossfades between the two mix types: left for A+B (summed) and right for AxB (ring-modulated). The **A+B** knob offsets the B section level relative to the A section in the summed mixing. The **AxB** knob controls the overall volume of the ring-modulated mix (ring-modulation tends to be quieter than summed mixing).
Waveform display	The phase offset of the X and Y axis can be adjusted by clicking on the screen and dragging the mouse – in effect rotating the image. **Zoom** amplifies or attenuates the input signal.

FX section

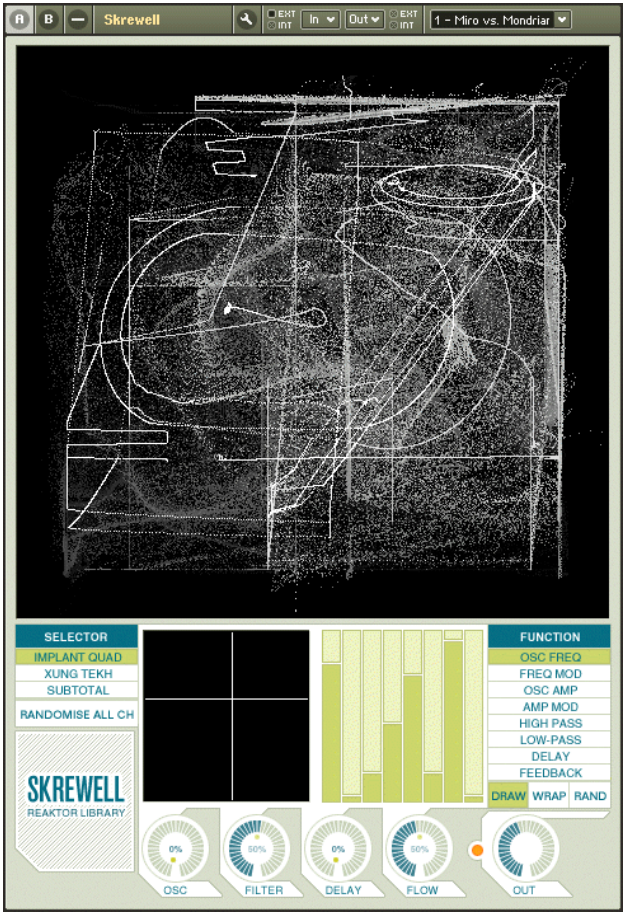
Section/Control	Function
Filter	The filter cutoff and resonance can both be controlled and automated by the **Cut** and **Res** faders. **LP-HP** crossfades between the lowpass filter [at the left] and the highpass filter [at the right]. **Mix** crossfades between the unprocessed signal [at the left] and the filter output [at the right].
Distortion	**Mix** crossfades between the unprocessed signal [at the bottom] and the distorted signal [at the top]. Drive determines the extent of the distortion.
EQ	**Hi** determines high frequency shelving cut/boost. Set to the centre for no change. Similarly, **Low** determines low frequency shelving cut/boost.
Spin	A delay effect, containing controls for left (**T1**) and right channel (**T2**) delay time and feedback (**FB**). The effect can be used for traditional delay sounds, for spreading the sound within the stereo panorama field, and for delay-modulation sounds (similar to pitch shifting).
Resochord	The Resochord uses 6 parallel delay lines, each at differing delay times representing the notes of a chord. This in effect imposes an additional harmonic structure onto the input signal. The preset list selects from 20 chord types. Delay feedback is the sum result of the **FB** knob and FB fader. The overall delay time is offset in semitones via **Pitch**, **Fine** and either P1 or P2 via **Mod**. Mix crossfades between the dry signal [at the bottom] and the Resochord output [at the top].

Section/Control	Function
Reverb	The preset list selects from 6 room types. **Order** determines the serial routing order of signal through the Resochord and Reverb units, (after the Spin module). Select **Pre** to route the post-spin signal through the Reverb first, then Resochord. Select **Post** to route through the Resochord first, then Reverb. **Mix** crossfades between the dry signal [at the bottom] and the Reverb output [at the top].
Volume	Volume determines the overall level, and **Width** determines the amount of stereo spread.

Additonal Controls

Control	Function
Restart	When this button is activated, starting or ending any automation recording will restart all sliders. Pressing the button below will restart all sliders immediately (which simulates recalling a snapshot).
Speed	The overall automation speed can be set to one of four levels by clicking on this button.
Randomize	Randomizes all controls.
Mute	Mutes the output.

7.2 Skrewell



4.37 Skrewell's user interface

7.2.1 Introduction

Skrewell is an intuitive and visual sound design workstation whose soundscapes can range from meditative atmospheres to crackling harshness. Its sound engine uses eight parallel oscillator sections (channels) that blend into a single, complex signal. This unique construction means that its interface is unlike that of a classic additive/subtractive synthesizer. The [Draw] edit mode allows for standard value adjustment for each channel's parameters. The parameters are represented by eight vertical bars (one bar for each oscillator section), which control the channels' oscillators, integrated filters and feedback delays. The [Wrap] and [Rand] edit modes provide special ways of altering your chosen parameter across all eight channels simultaneously. The Skrewell unit structure – and therefore the available parameters for each channel – is different in each of the three operation modes. Four main knobs manipulate the sound globally, mainly by mapping the individual channels' parameters. Additionally, a large display visualizes the audio output as Lissajous figure.

7.2.2 Operation Modes

There are three operation modes, each one based on a unique tone generator system. In Implant Quad mode, each channel consists of a pulse oscillator with subsequent feedback delay; within the delay line a normalizer and a filter alter the signal. Xung Tekh is similar, except the filter is placed before the feedback delay. Subtotal uses a parabolic waveform instead of the pulse waveform and omits the filter completely. The parameters of the tone generators are adjusted in the [Sound Engine] section.

Control	Function
Operation Mode	Selects the main way of operation.
Randomize All Ch.	Sets all parameters of all channels to random values. The [Output Volume] should be lowered to prevent unexpected bursts of noise.

7.2.3 Sound Engine

This section adjusts the parameters of the tone generators. Depending on the [Operation Mode] setting, a list of the currently available parameters is in the [Parameter Select] display. The selected parameter can then be edited within the [Edit Area], where each bar represents one of the eight parallel oscillator units that form the tone generators.

Control	Function
Function	Switches between the various parameters that control the channels. Depending on the [Operation Mode], there are different sets of available parameters. The values of the selected parameter are displayed for each oscillator section in the [Edit Area].
Edit Mode	Selects in which way the instrument interprets mouse movements within the [Edit Area]. [Draw] allows direct adjustment of each bar. [Wrap] lowers / raises all bars simultaneously, keeping their ratio constant. If a value exceeds the value range it is mirrored. [Rand] performs random changes on all eight bars.
Edit Area	Displays the selected parameter, with one bar representing the parameter's value for each of the eight channels. Mouse movements within this area alter those values, controlled by the [Edit Mode].
Display Control	Scales the Lissajous display.

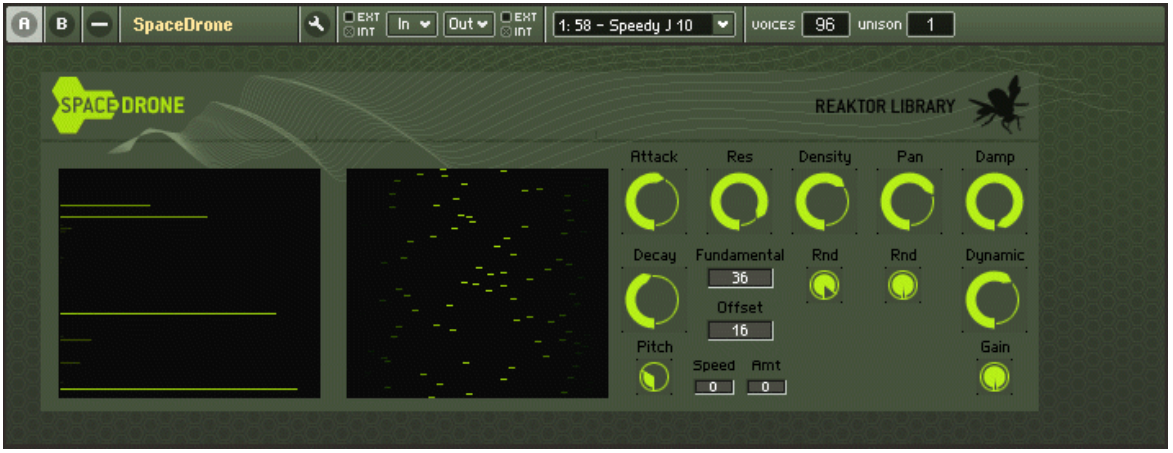
7.2.4 Master Controls

These master controls either scale the settings of the [Sound Engine] section (e. g. [Delay Time]) or adjust additional tone generator parameters (e. g. [Flow Amount]). As they affect all eight channels of the tone generators simultaneously they can be used to alter the overall sound.

Control	Function
Oscillator Pitch	Modifies the pitch of all channels. Technically, it controls a mapping function that modulates the values adjusted within the [Edit Area] for each channel. Only very high individual settings will result in high pitches if turned to the left, while less high values will be mapped onto low pitches; move to the right for the opposite effect.
Filter Cutoff	Modifies the filter cutoff frequency of all channels; see [Oscillator Pitch] for technical details.
Delay Time	Modifies the delay time of all channels; see [Oscillator Pitch] for technical details. By turning the knob to the left the delay times can be dramatically shortened, resulting in comb filter-like effects.

Control	Function
Flow Amount	Adjusts various amounts of modulation, depending on the selected [Operation Mode], e.g. frequency modulation amount, amplitude modulation amount etc. Like the [Oscillator Pitch], this knob maps the individual channel's value. Turn to the left for less modulation and more inertia, turn to the right for the opposite effect.
Output Volume	Sets the master output volume. As slight variations of Skrewell's parameters might result in extreme volume changes, this control should be handled carefully. There is an additional [Mute] button at the knob's left.

7.3 Space Drone



4.38 Space Drone user interface

7.3.1 Introduction

SpaceDrone generates atmospheric pads which range from light rain or howling wind noises to deep and uncanny space sounds. Technically, the instrument is based on 96 parallel voices spread across the frequency spectrum. Each voice consists of a noise generator; the signal's amplitude is shaped by an envelope, its frequency content gets modified by a bandpass filter, and finally it gets positioned in the stereo field.

7.3.2 Sound Engine

The parameters of the sound engine are in the A panel of the instrument. They control the noise generators, their subsequent bandpass filters, the amplitude shaping envelope and corresponding triggering algorithm, and the pan, gain and damping of the signals.

Control	Function
Attack	Sets the time that passes until the amplitude envelope reaches its peak after triggering. The [Density] knob controls speed at which the envelope is re-triggered.
Decay	Sets the time that passes until the amplitude envelope completely fades out after it has reached its peak. The [Density] knob controls speed at which the envelope is re-triggered.
Pitch	Sets the amount by which the amplitude envelope modulates the voice's pitch, i. e. the bandpass filter's center frequency. Turn to the left for inverse modulation – the higher the envelope signal, the lower the pitch. Turn to the right for the opposite effect.
Resonance	Sets the bandpass filter's resonance.
Fundamental	Adjusts the fundamental frequency, i. e. the pitch of the lowest voice.
Offset	Sets the offset of the filter harmonics: All voices are harmonics of the fundamental frequency (see [Fundamental]); all harmonics below the one adjusted here are skipped.
Speed	Controls the rate at which a LFO modulates each voice's frequency randomly.
Amount	Sets the amount by which the voice's frequency is changed by the random LFO.
Density	Sets the speed at which each voice's amplitude envelope is re-triggered.
Random	Sets the randomness of the re-triggering events. Turn to the left for completely regular re-triggering; turn to the right to give each voice a slightly varied re-triggering speed.
Dynamic	Sets the dynamic range of the amplitude envelope. Turn to the left to bind every voice to a constant maximum level; turn to the right to allow some (randomly picked) voices to be quieter.
Pan	Sets the rate at which each voice is rotated within the stereo field.
Random	Sets the randomness of the panning speed. At high values each voice has a slightly different pan rate.
Damp	Sets the amount of damping applied to high frequencies.
Gain	Sets the amount of amplification applied to each voice independently.

7.3.3 Reverb



4.39 Space Drone's reverb unit

The reverb unit is contained within the panel's B view. It can further enhance the spatial character of the atmospheric pads. When not in use it should be turned off by the [Power] control to save CPU power. Although it is built completely within the new and efficient REAKTOR core layer, it is designed to produce high-quality reverberation sounds.

Control	Function
Size	Sets the size of the virtual reverberation room.
Symmetry	Places the signal in the virtual reverberation room. Turn to the left or right to move the signal away from the center.
Diffusion	Sets the amount of diffusion of the reverb signal. Turn to the right for a less echoic sound.
Release	Adjusts the time that passes before the reverberation sound has decayed.
Spin	Sets the amount of modulation applied to the reverb. Technically, the modulation affects the delay time of the delay modules on which the reverb is build.
Frequency	Sets the rate of the LFO used as modulation source (see [Spin]).
High Cutoff	Sets the cutoff frequency of the lowpass filter that is damping the high frequencies.

Control	Function
High Damp	Sets the amount of damping applied to the frequencies above the [High Cutoff] frequency.
Low Cutoff	Sets the cutoff frequency of the highpass filter that is damping the low frequencies.
Low Damp	Sets the amount of damping applied to the frequencies below the [Low Cutoff] frequency.
Mix	Crossfades between the unprocessed, dry signal (at the left) and the reverberated, wet sound (at the right).
Power	Switches the reverb unit on or off. Turn off to save CPU power if the reverb is not used.

8 Synthesizer

8.1 2-Osc



4.40 2-OSC user interface

8.1.1 Introduction

2-Osc, finally, is a synthesizer based on the classical subtractive model: oscillator – filter – effect and vast parameter modulation capacities. This particular implementation of the concept features two parallel oscillators, two serial filters and a final stereo delay effect unit as well as three envelope generators, two low frequency oscillators and, additionally, an arpeggiator.

Like in the FM4 instrument the technical implementation is more interesting than the underlying synthesis concept. All Oscillator section modes are built completely within Core, providing anti-aliased sine, triangular, pulse and sawtooth waveforms as well as two special modes (Buzz and 2Saw). The second oscillator block also features soft synchronization, i.e. the oscillator is not always re-triggered when the first oscillator's signal rises above zero but only at certain phase states of the second oscillator. The Filter sections contain an additional saturation level that helps to tightly control the filter's resonance; the internal filter processing uses over-sampling for more accurate sounds. (The second filter is a Ladder filter: a lowpass filter that provides resonance also when using its 1-pole mode with 6 dB damping per octave.) The envelope generators and both LFOs provide similar high-end features; see below for details.

8.1.2 List of Controls

The instrument's panel is rather simple and similar to any analogue-like subtractive synthesizer. Particular features of the sections are explained with the relevant controls.

Section	Control	Function	Range
Osc 1 / 2	Pitch	Controls the frequency of the oscillator as offset to the pitch triggered by a MIDI Note On event in semitones. (See also [Key].)	-60...+60
	Fine	Controls the oscillator's frequency in cent. This value is added to the [Pitch] control.	-.5... +0.5
	PW (Pulse Width)	Controls the pulse width of the [Pulse] waveform (50% at a value of 0) and the phase offset of the second sawtooth wave in [2Saw] mode (180 degrees at a value of 0).	-1...+1

Section	Control	Function	Range
	Waveform	Selects one of the waveforms. Beside the well known forms there is the [2Saw] mode within the first Oscillator section that consists of two sawtooth oscillators of the same frequency whose phase offset can be controlled by the [Pulse Width] knob. The second Oscillator section provides a [Buzz] mode that uses a random waveform which is re-triggered at the pitch's frequency.	-
	Key	Available only within the second Oscillator section this disables the MIDI pitch following. The oscillator is set to a fixed frequency adjusted by the [Pitch] control, interpreted as offset to MIDI note 60 (C3).	on / off
	Synchronize	Available only within the second Oscillator section this control adjusts the amount of soft synchronization. At a value of 0 there is no synchronization, at a value of 1 there occurs normal hard sync, i.e. the second oscillator is reset to its initial phase each time the first oscillator's signal rises above zero. In between soft synchronization is used, e.g. at a value of 0.33 the second oscillator is reset when the first oscillator's signal rises above zero but only if its own phase is in the last third of its cycle.	0...1
	Mix	Sets the volume of each Oscillator section when mixed with the Ring Modulator section's signal.	0...1
Ring Mod	Mix	Sets the volume of the Ring Modulator's signal when mixed with the Oscillator section's signals.	0...1
	Cutoff	Sets the filter's center frequency as MIDI pitch. Depending of the filter's operation mode (see [Lp-Bp-Hp] and [Pole Select] resp.) the frequencies below, above or around this frequency are passed, all others are damped.	
	Res(onance)	Controls the resonance of the filter. Within the first Filter section the resonance is also influenced by the [2p-4p] control, i.e. the amount of damping. The second Filter section's Ladder filter features constant resonance independent of the amount of damping, even when only using one pole. Values above 1 produce self-oscillation of the filter. This is controlled with the [Sat Lvl] knob.	0...1.1
	Lp–Bp–Hp	Crossfades between three operation modes of the first Filter section. At the left a lowpass is activated, rejecting all frequencies above the [Cutoff] frequency. At the right a highpass is used, rejecting all frequencies below the [Cutoff] frequency. At a mid position only the frequencies around the [Cutoff] value are passed.	0...1

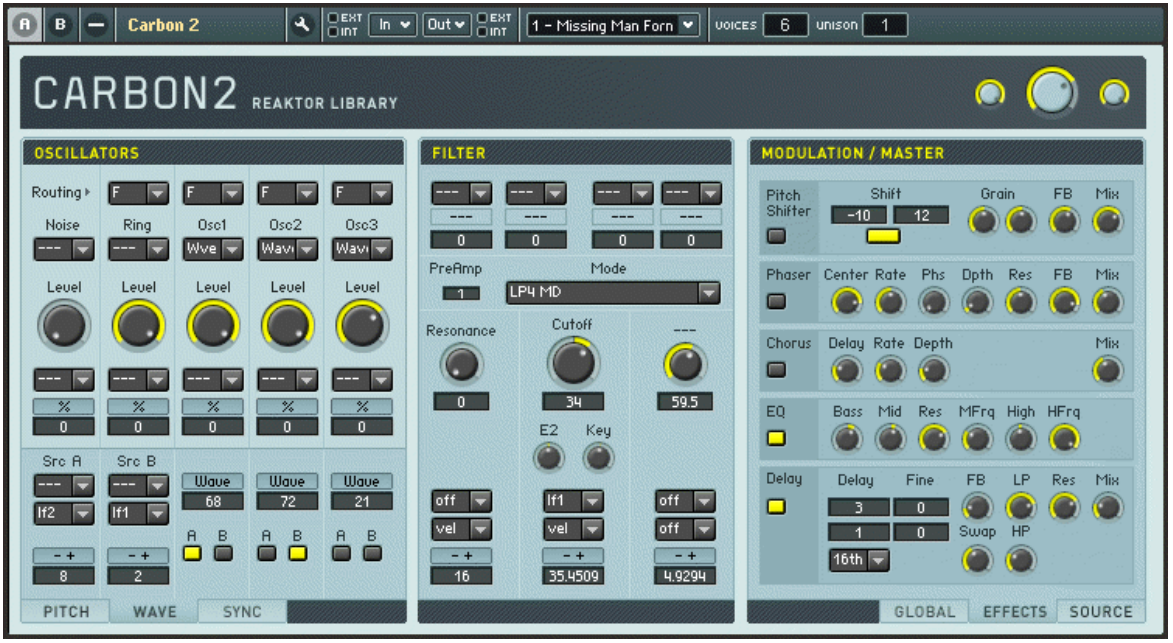
Section	Control	Function	Range
	2p–4p	Crossfades between the 2-pole and 4-pole outputs of the first filter. At the left the filter offers a damping of 12 dB per octave, at the right the damping is 24 dB per octave.	0...1
	Lp 1 ... Lp 4	Selects the amount of poles and zeros of the second Filter. Each additional pole results in 6 dB damping per octave, i.e. there is maximum damping of 24 dB. Using the Ladder filter model, the signal passes four one-pole stages and is then fed back to the input of the first stage, so that there is a resonance effect at each of the four outputs.	Selector
	Sat(uration) Level	Sets the level where the signal is internally limited by a slight distortion, preventing problems at high resonance values.	-12...+12
	Key(board to Cutoff)	Controls the amount of keyboard mapping of the filters' cutoff frequency. At the left the cutoff frequency is constant – at the right it follows the played MIDI notes, i.e. the [Cutoff] control specifies the offset of the filter's cutoff to the triggered MIDI note's pitch.	0...1
	Env(elope to Cutoff)	Adjusts the amount of cutoff modulation by the Filt Enve section in semitones. At negative values the envelopes signal is inverted.	-100..+100
	Osc2 FM	Sets the amount of modulation applied to the first Filter's cutoff frequency by the signal of the second Oscillator. Negative amounts can give a different sound color.	-100...+100
	On	Activates the filter. In the Off position the filter is bypassed and has no load on the CPU.	on/ off
Osc Env, Filter Env, Amp Env	A (Attack)	Sets the attack time of the envelope generators respectively.	-20...+100
	D (Decay)	Sets the decay time of the envelope generators respectively.	-20...+100
	S (Sustain)	Sets the sustain level at which the envelope generators' signal is hold after the attack-decay phase as long as the MIDI Note gate signal is present.	0...1
	R (Release)	Sets the release time of the envelope generators respectively.	-20...+100

Section	Control	Function	Range
	Vel(ocity to Amplitude)	Sets the amount of velocity mapped onto the envelope's amplitude. At a value of 0 the amplitude is independent of the MIDI note's velocity; at a value of 1 it is completely determined by the velocity.	0...1
	Key(board to Times)	Controls the amount of key following of the envelope generator's transition times (i.e. attack, decay and release). At positive values the times become shorter at high MIDI note pitches (simulating the faster response of mechanical instruments at high pitches); at negative values the times become longer at high pitch settings.	+1...-1
	Legato	If monophonic mode is used (see [Mode] in the Voice Mode section) and this is switched on, tied notes don't re-trigger the envelope generator.	on / off
	P(itch) Range	Controls the range by which the Osc Env section modulates the pitch of the two oscillators. This adjusts the maximum level available to the [Pitch 1/2] controls.	-
	Pitch 1/2	Controls the amount of modulation applied to the oscillators' pitch by this envelope generator. At negative values the envelope's signal is inverted. At maximum settings the value adjusted with [P Range] is reached.	-1...+1
	FM 1->2	Sets the amount of frequency modulation applied to Osc 2 by using the signal of Osc 1. This is the maximum value applied when the envelope reaches its peak amplitude.	0...120
LFO 1 / 2	(Clock) Sync	Switches synchronization of the LFO's rate to the current MIDI clock on or off. Depending on this setting the rate can be controlled in Hz (by the [Speed] control) or as a time specified as multiples of a certain division of a 4/4-bar.	on / off
	Speed	If the LFO is not in Sync mode this knob controls its rate in Hz.	0.05...30
	N (Beats per Cycle)	Controls the duration of one LFO cycle as number of beats. The length of a beat is specified by the [Beats per Bar] control.	1...32
	----- (Beats per Bar)	Sets the number of beats into which one 4/4 bar is divided, e.g. a value of 8 sets a beat's length to a eighth note's duration. The bar length is derived from the current MIDI tempo.	-
	Waveform	Selects the waveform of the low frequency oscillator. [S&H] samples and holds a white noise signal at the given rate, [Brown] is similar but uses brown noise (i.e. with less high frequency components).	-

Section	Control	Function	Range
	PW (Pulse Width)	Controls the pulse width if [Pulse] or [Triangular] are selected as waveforms. At a value of 0 the waveform is symmetrical.	0...1
	Phase	Sets the phase of the LFO in synchronized mode.	-0.5...+0.5
	Wheel	If on, the LFO amplitude is controlled by the MIDI modulation wheel.	on / off
	P(itch) Range	Controls the range by which the LFO section modulates the pitch of the two oscillators. This adjusts the maximum level available to the [Pitch 1/2] controls.	-
	Pitch 1 / 2	Controls the amount of modulation applied to the oscillators' pitch by the LFO (i.e. vibrato). At negative values the LFO's signal is inverted. At maximum settings the value adjusted with [P Range] is reached.	-1...+1
	PWM	Controls the amount of pulse width modulation applied to the two Oscillator sections by the LFO.	0...1
	Filter	Sets the amount of cutoff modulation applied to the Filter sections by the LFO in semitones.	0...120
	Amp	Adjusts the amount of modulation applied to the main amplitude by the LFO (tremolo).	0...1
Voice Mode	Mode	Selects one of the voice allocation modes; the arpeggiator is activated here, too. There is a polyphonic mode as well as a 1-voice monophonic mode; in [Unisono] mode only one note is played at a time (like in mono mode), but all voices are used to play the same pitch, slightly detuned (see [Detune]). This results in an effect similar to a chorus.	-
	Arp Mode	Selects the way the arpeggiator works: [Up] plays all triggered MIDI notes in an upward sequence, i.e. the lowest pitch first; [Down] results in the opposite effect. [Up+Down] alternates between the upward and the downward sequence and [Random] plays the pressed keys in a random sequence.	-
	Arp Speed	Selects the rate at which the arpeggio notes are played, synchronized to the MIDI clock.	-
	Glide	Switches portamento (pitch gliding) on or off.	on / off
	Glide Time	Sets the time used to reach a new note's pitch if [Glide] is active in seconds.	0...60

Section	Control	Function	Range
	Pitchbend Range	Controls the range of the MIDI Pitchbend wheel in semitones.	0...24
	Detune	Controls the amount of detuning applied to the voices in cent. In [Unisono] mode it controls the beatings in the monophonic sound. In other modes it simulates the detuned oscillators of a vintage analog synth.	0...0.5
Output	Spread	Controls the amount of stereo panning applied to each voice independently. At low values all voices are placed in the middle of the stereo panorama; at high values they are spread to the left and right.	0...1
	Volume	Sets the output level of the instrument in dB before it is routed to the final Stereo Delay effect.	-300...0
Stereo Delay	(Clock) Sync	Switches between manual delay time adjustment and times synchronized to the current MIDI clock tempo.	on / off
	Left / Right Delay	Controls the time by which the left and right channel's signal is delayed. Depending on [Sync] the times can be set as multiples of MIDI clock increments (like in the LFOs) or can be adjusted in milliseconds.	-
	Feedback	Adjusts the amount of the delayed signal routed back to the inputs of the two delay lines.	0...1
	X Amount	Controls the amount of cross feedback. between the two channels as part of the [Feedback].	0...1
	Hi(gh) Cutoff	Controls the cutoff frequency (in Hz) of a lowpass filter that damps the delayed signal's high frequencies.	100..20000
	Mix	Controls the balance between the dry, unprocessed signal (at 0) and the wet, delayed signal (at 1).	0...1
	On	Switches the effect unit on or off (saving CPU power).	on / off

8.2 Carbon 2



4.41 Carbon 2 user interface

8.2.1 Introduction

Carbon 2 is based on REAKTOR 4's well-known workhorse synthesizer, but it has been completely rebuilt. In particular the oscillators and filters are now based on REAKTOR Core components developed particularly for this instrument. The panel has been optimized for usability, with a clear structure providing fast access to all parameters while hiding the technical complexity.

Basically, Carbon 2 is a classical subtractive synthesizer. The signal of the three oscillator section (left column of the panel) passes through a multi-mode filter (middle column) and is then routed to the effect units (right column). Several modulation sources such as envelope genera-

tors and LFOs (placed in a second page in the right panel column) and the global parameters (a third page in the right column) control the sound, adding additional liveliness and movement.

8.2.2 Oscillators

The oscillator section produces the instrument's basic signal. Three oscillator slots provide several different waveforms; along with traditional analogue types like sine and sawtooth there is a digital wavetable oscillator containing a wide array of waveforms that can be crossfaded smoothly. A noise generator and a ring modulator based on the signal of the three main oscillators are added for a total of five basic sound sources.

Each oscillator slot offers control over volume, pitch, and waveform synchronization. The pitch and sync controls are placed in two pages at the bottom of the panel, grouped with a third page controlling the waveform. This third page is only active if the digital wavetable or the doubled sawtooth is selected.

Section	Control	Function
Main	Routing	Sets the destination of the respective oscillator's signal. On [F], the sound is sent to the [Filter] section; switching to [D] bypasses the filter and routes the signal directly to the effect units.
	Noise	Switches the white noise generator on or off.
	Ring	Selects which oscillator signals are fed into the ring modulator. Switch off to save CPU power if the ring modulator is not used.
	Osc1/2/3	Selects the waveform of each oscillator slot. Along with the standard waveforms (sawtooth, pulse, triangular, sine and noise), you will find a doubled sawtooth, a quantized sine, a buzz oscillator based on a noise generator, and a digital wavetable. (See the [Wave] page for additional information on the doubled sawtooth and the digital wavetable.)
	Level	Sets the slot's volume level.
	Level Modulation Source	Selects the slot's volume level modulation source.
	Level Modulation Amount	Sets the amount and polarity of modulation applied to the slot's volume level. Clicking on the control's title bar restores the value to its default.
Pitch	A/B Modulation Source	Selects sources to modulate the oscillators' pitch. The two individual slots ([A] and [B]) can mix up to two sources.

Section	Control	Function
	A/B Modulation Amount	Adjusts the amount and polarity of modulation applied to the oscillators' pitch. The left side of the control adjusts coarse values, the right side is used for fine-tuning; clicking the control's title bar restores the value to its default.
	Osc1/2/3 Pitch Shift	Transposes the oscillators' sound respectively. The left side of the control adjusts coarse values, the right side is used for fine-tuning; by clicking on the controls title bar with the mouse the value is reset to its default.
	Osc1/2/3 Modulation Switch A/B	Turns modulation of the oscillator's pitch by modulation slot [A] or [B] on or off.
Wave	A/B Modulation Source	Selects sources to modulate the waveform. The two individual slots ([A] and [B]) can mix up to two sources. This will show no effect until the doubled sawtooth or the wavetable is selected in [Osc1/2/3].
	A/B Modulation Amount	Adjusts the amount and polarity of modulation applied to the waveform. Clicking on the control's title bar resets the value to its default. This will show no effect until the doubled sawtooth or the wavetable is selected in [Osc1/2/3].
	Osc1/2/3 Waveform Control	This either selects a digital waveform from the wavetable, or – if the doubled sawtooth is activated in [Osc1/2/3] – this controls the ratio between the phases of both sawtooth waves.
	Osc1/2/3 Modulation Switch A/B	Turns modulation of the waveform selection by modulation slot [A] or [B] on or off.
Sync	Gate Sync Switch	Turns synchronization of the oscillators' waveforms to the MIDI gate on or off. If on, all three oscillators are reset to the phase adjusted in [Gate Sync Phase] when a note is pressed.
	Gate Sync Phase	Controls the phase to which all oscillators are set on MIDI gate events. Clicking on the control's title bar restores the default value.
	Osc2/3 Sync Switch	Switches on or off the synchronization of the oscillators 2 and 3 respectively to the signal of oscillator 1. If on, the oscillator is reset to the phase adjusted in [Osc2/3 Sync Phase] when the signal of oscillator 1 rises above zero. (See also [Osc2/3 Mode Fade].)

Section	Control	Function
	Osc2/3 Sync Phase	Controls the phase to which the oscillators 2 and 3 are reset when the signal of oscillator 1 rises above zero. Clicking on the control's title bar restores the default value. (See also [Osc2/3 Mode Fade].)
	Osc2/3 Mode Fade	Interpolates between hard synchronization (at low values) and soft synchronization (at high values). In hard synchronization mode the oscillator is always reset if the signal of oscillator 1 rises above zero; with soft synchronization this is not always the case, producing a mix between the synced waveform and the non-synced one. Clicking on the control's title bar restores the default value.

8.2.3 Filter

The filter section is placed between the oscillators and the effects; it sculpts the oscillators' basic sounds. Before the signal is routed to the filter it passes two effects that provide saturation and quantization, as well as additional low- and high-shelf equalizers. The filter itself contains several modes, optimized for a warm yet crisp sound. You'll find standard low-pass and high-pass, band-pass, and band-reject filters, a special feedback filter (called [Zwnl]), and a peak EQ and comb filter. After the main filter comes another effect section, similar to the previous one.

Section	Control	Function
Pre-Filter Effects	Effect A/B Mode Select	Selects the effect units applied to the signal before it passes to the filter. There are low and high shelf EQs in the left [A] menu and saturation and quantization in the subsequent right [B] menu.
	Effect A/B Amount	Sets the parameter of the effect unit selected by [Effect A/B Mode Select]. For the equalizers, this is the amount of damp or boost applied to the signal. For the saturator it's the amount of drive, and for the quantizer it's the amount of distortion.
Main	PreAmp	Controls the level correction of the signal after it has passed the [Pre-Filter Effects] section and before it enters the main filter.
	Mode	Selects the filter mode. There are high-pass, bandpass and band-reject filters, several low-pass modes, a feedback lowpass, a peak equalizer, and a comb filter.
	Cutoff	Sets the frequency of the filter.
	Resonance	Sets the resonance of the filter.

Section	Control	Function
	Bandwidth	Sets the width of the band for the bandpass and bandreject filters. If the peak equalizer is selected, this parameter sets the amount of boost applied.
	E2	Controls the amount and polarity of modulation applied to the cutoff control by the second envelope generator. Turn to the left for negative modulation, i.e. low cutoff values at high envelope signals. Turn to the right for normal positive modulation.
	Key	Controls the amount and polarity of modulation applied to the cutoff control by the current pitch. Turn to the left for negative modulation, i.e. low cutoff values at high pitches. Turn to the right for normal positive modulation. This modulation is independent of the Key Scaler of the [Modulation] section.
	Cutoff/Resonance/ Bandwidth Modulation Source	Selects the sources used to modulate the filter's cutoff, resonance and bandwidth. Up to two sources can be selected, and their signals are summed together. In case of the cutoff modulation, these signals are added to the hard-wired modulation by the second envelope generator and the MIDI pitch.
	Cutoff/Resonance/ Bandwidth Modulation Amount	Adjusts the amount and polarity of modulation applied to the filter's cutoff, resonance and bandwidth. Clicking on the control's title bar restores the default value. In case of the cut-off modulation, this amount doesn't affect the hard-wired modulation by the second envelope generator and the MIDI pitch.
Post-Filter Effects	Effect A/B Mode Select	Selects the effect units applied to the signal after the filter, before it gets routed to the main effect units. You'll find saturation and quantization in the left [A] menu, and lowpass and highpass filters in the right [B] menu.
	Effect A/B Amount	Sets the parameter of the effect unit selected by [Effect A/B Mode Select]. For the saturator this is the amount of drive; for the quantizer the amount of distortion; and for both filters the cut-off frequency.

8.2.4 Effects

The effects additionally enhance the instrument's sound. There are five units: a pitch shifter, a phaser, a chorus, an equalizer and a delay. These standard effects are engineered for the finest of results.

Section	Control	Function
	Power & Mix	Each effect unit provides a power switch and a mix button. The mix button crossfades between the dry, unprocessed signal (at the left) and the wet effect sound (at the right). To save CPU power, turn the power switch off when the specific effect is not in use.
Pitch Shifter	Shift L / R	Determine the pitch shift of the left and right channel respectively in semitones.
	Grain Size L/ R	Adjust the grain size of the pitch shifting algorithm for the left and right channel respectively. Turn to the left for large chunks and echoic sounds, to the right for tiny grains and an accurate pitch shift.
	Feedback	Controls the amount of feedback.
	Reverse	Switches between forward and reverse grain playback.
Phaser	Center Frequency	Sets the center frequency of the filters that produce the phaser signal.
	Modulation Rate	Sets the speed at which the [Center Frequency] is modulated.
	Phase	Sets the phase of the LFO modulating the [Center Frequency]. (See also [Modulation Rate].)
	Depth	Sets the amount of modulation.
	Resonance	Sets the resonance of the internal filters.
	Feedback	Sets the amount of feedback.
Chorus	Delay	Sets the main delay of the chorus.
	Depth	Sets the amount of modulation applied to the [Delay] time.
	Rate	Sets the speed at which the [Delay] time is modulated.
Equalizer	Bass Boost	Controls the boost (or damping) applied to the bass frequencies below 300 Hz.
	Mid Frequency	Adjusts the frequency of the peak equalizer applied to the middle frequency spectrum.
	Mid Boost	Controls the boost (or damping) applied to the middle frequencies around the [Mid Frequency].
	Mid Resonance	Sets the resonance of the mid equalizer.
	High Frequency	Adjusts the frequency of the high shelf equalizer.

Section	Control	Function
	High Boost	Controls the boost (or damping) applied to the frequencies above the [High Frequency].
Delay	Delay L / R	Sets the delay times of the left and right channel respectively. The time is controlled in increments selected by the [Quantize] control.
	Fine L / R	Adds an offset to the values controlled by [Delay L / R] in milliseconds.
	Quantize	Selects the unit by which the delay times are quantized. Sixteenth notes and eighth triplets are available.
	Feedback	Sets the amount of feedback.
	Wrap	Controls the amount of cross-feedback. Turn to the left to route the every channel's feedback to itself; turn to the right to route it to the other channel.
	Resonance	Sets the amount of resonance applied to the low-pass and high-pass filters within the feedback circuit.
	Lowpass	Controls the frequency of the low-pass filter within the feedback circuit.
	Highpass	Controls the frequency of the high-pass filter within the feedback circuit.

8.2.5 Modulation Sources

Several modulation sources are available: two ADSR envelope generators, a recordable envelope, and two LFOs combined with a key-scaler that provides four independent control points and four freely assignable MIDI controllers. The envelope generators and LFOs offer several types of MIDI clock interaction for rhythm-based modulation effects.

Section	Control	Function
Envelope Generators 1/2	Trigger	Selects the events that re-trigger the envelope generator. [Gate] only activates the MIDI gate signal. [Clock Gate] re-triggers the envelope at each unit selected by [Quantization] as long as the MIDI gate is open. [SP Clock Gate] is similar, but synchronizes the quantization to the global MIDI song position; therefore, the MIDI clock has to be running. (See also [Globals][EG Mode].)
	Quantization	Selects the metrical unit used to re-trigger the envelope if [Trigger] is set to [Clock Gate] or [SP Clock Gate].

Section	Control	Function
	Key	Controls the amount and polarity of modulation applied to the envelope's transition times by the current pitch. Turn to the left for negative modulation, i.e. shorter attack, decay and release times at low pitches. Turn to the right for normal positive modulation, i.e. longer times at low pitches.
	Velocity	Controls the current velocity's influence on the envelope amplitude. At low values the envelope triggers with the same amplitude; at high values the MIDI velocity determines its peak value.
	Transition Time Modulation Select	Selects the additional modulation applied to the envelope generator's transition times. The attack phase can be modulated by the MIDI velocity while the decay time can be modulated by the velocity and the four MIDI controllers (see [MIDI Controllers]). The amount and polarity of modulation is controlled by [Transition Time Modulation Amount].
	Transition Time Modulation Amount	Controls the amount and polarity of modulation applied to the destination selected by [Transition Time Modulations Select]. Turn to the left for negative modulation, i.e. shorter attack or decay times at low modulation source values. Turn to the right for normal positive modulation, i.e. longer times at low values.
	Attack	Sets the attack time of the envelope generator.
	Decay	Sets the decay time of the envelope generator.
	Sustain	Sets the sustain level of the envelope generator.
	Release	Sets the release time of the envelope generator.
	Hold	Sets the duration of an additional hold phase between attack and decay.
	Delay	Adds an initial delay period before the trigger signal restarts the envelope
	R=D	Links the release time to the decay time. If on, the value adjusted by [Decay] is also used to control the release phase.
Envelope Generator 3	Record	Arms the recordable envelope. The recording is started when a MIDI gate is received and ends when the gate closes. All movements of the [Value] knob are stored and can be played back as envelope (see [Play]).

Section	Control	Function
	Play	Enables playback of the recorded movements, triggered like an envelope by MIDI gate signals.
	Loop	Loops the recorded movement on playback.
	Value	When recording (see [Record]), every movement of this knob is stored to the memory. During playback (see [Play]), the knob displays the recorded movements.
LFO 1/2	Waveform	Selects the waveform of the Low Frequency Oscillator. There are the standard waveforms [Sine], [Triangular], [Pulse], and [Random Steps], and several derivations: [Pulse+] is a pulse waveform with all negative values clipped to 0; [Saw Up+] and [Saw Down+] are triangular forms with only rising resp.falling ramp and only positive values; [Hsin+] is a multiplication of [Pulse+] and [Sine] etc.
	Amplitude Modulation Source	Selects the source used to modulate the LFO's amplitude. Clicking the control's title bar restores the value to its default.
	Amplitude Modulation Amount	Adjusts the amount and polarity of modulation applied to the LFO's amplitude.
	Trigger Mode	Selects the events that re-trigger the LFO. In [Freerun] mode no reset occurs; in [Gate] mode the LFO is set to the phase adjusted by [Reset Phase] on a MIDI gate event. [Clock Gate] is similar to [Gate] mode but also activates a grid for the LFO's frequency (see [Rate]). [SP Clock Gate] additionally synchronizes the reset to the global MIDI song position.
	Reset Phase	Adjust the phase to which the LFO is set on re-triggering events.
	Rate Modulation Source	Selects the source used to modulate the LFO's frequency. If [Trigger Mode] is set to [Clock Gate] or [SP Clock Gate], frequency modulation is not available.
	Rate Modulation Amount	Adjusts the amount and polarity of modulation applied to the LFO's frequency. Clicking the control's title bar restores the value to its default. If [Trigger Mode] is set to [Clock Gate] or [SP Clock Gate], frequency modulation is not available.
	Rate	Sets the frequency of the LFO. If [Trigger Mode] is set to [Clock Gate] or [SP Clock Gate], a grid is applied to this control, quantizing the LFO's rate to the metrical units selected in [Rate Quantization].

Section	Control	Function
	Rate Quantization	Selects the metrical unit used as quantization grid for [Rate] when [Trigger Mode] is set to [Clock Gate] or [SP Clock Gate].
KeyScaler	Sliders	Provides a signal derived from the current pitch that can be used as modulation source. The four sliders define the function used to map the MIDI pitch onto the modulation signal: At low pitches, the value of the leftmost slider is used as modulation signal; at high pitches the value of the rightmost slider is selected. In between, interpolation occurs, using the two middle sliders as control points. In addition to the normal signal, there is a modulation source that multiplies the key-scalers value by the current MIDI velocity.
MIDI Controllers	Faders	The leftmost fader is hard-wired to the MIDI modulation wheel. All others can easily be assigned to any MIDI Continues Controller via MIDI Learn. They are available as modulations sources, named C1, x1, x2 and x3.

8.2.6 Global Controls

The global controls access several different functions. First – and most important – the voice allocation of the synthesizer can be controlled, providing polyphonic and monophonic modes; by selecting the unison mode all available voices are set to the same pitch (as in a monophonic synth), but each one is slightly detuned. This results in waveform interference and a thick, chorus-like sound. Monophonic modes also provide portamento. Parameters determine the master pitch shift and MIDI pitchbend range, and adjust global tremolo and vibrato. Voices' position within the stereo field can also be adjusted.

Control	Function
Gate Mode	Selects the global operation mode. [Poly] selects the only polyphonic mode; portamento does not work in this mode (see [Glide Speed]). [Mono] results in a monophonic gate signal that is triggered on every MIDI note. [Legato] is similar but generates a new gate trigger signal only when the gate has been closed before, i.e. no note is already pressed. [Uni Mono] and [Uni Legato] activate the unison modes: A monophonic gate signal is used for all voices, but all available voices are used and detuned by the [Unisono] and [Unisemi] controls.
Envelope Mode	Selects the envelopes' behavior during the release period if a new attack is triggered. [Re-trigger] starts the attack phase beginning with the current envelope amplitude; [Reset] starts the attack with a value of zero. Thus, [Reset] might lead to unwanted clicks if used without care.

Control	Function
Unisono	Sets the amount of detuning applied to each voice when [Uni Mono] or [Uni Legato] is selected as [Gate Mode]. Slight detuning results in thick, chorus-like sounds.
Unisemi	Sets the amount of pitch shifting applied to each voice when [Uni Mono] or [Uni Legato] is selected as [Gate Mode]. This acts like the [Unisono] control but detunes the voices in semitones, e. g. a value of 12 will set all voices one octave apart.
Drift	Enables a drift mode that slightly detunes higher pitches. This results in a more analogue like sound.
Key	Activates key-scaling for the unisono control. If pressed, the [Unisono] value is lowered automatically at high pitches for a more constant sound over the complete pitch range of the instrument.
Velocity	Selects the mapping applied to the MIDI velocity. While [Linear] doesn't change the velocity, [Log] results in a compressor like effect while [Expo] produces the opposite effect.
Coarse	Sets the global tuning of the instrument in semitones, ranging from -63 to +64.
Fine	Sets the global tuning of the instrument in semitones, ranging from -0.5 to +0.5
Glide Speed	Adjusts the speed at which new pitches are reached if they are slurred, i.e. if the previous note was still held when the new one was pressed. This portamento effect only works in monophonic modes (see [gate Mode]).
Pitchbend Range	Sets the range in semitones by which the MIDI pitchbend wheel transposes the global pitch.
Vibrato Mode	Selects whether vibrato is off, on, or faded in by the MIDI modulation wheel.
Vibrato Amount	Sets the amount of vibrato. Clicking the control's title bar restores the value to its default.
Vibrato Style	Selects between three different vibrato modes.
Key	Adjusts the amount of key scaling applied to the vibrato. Turn to the left for no scaling, to the right for less vibrato at low pitches, producing a more musical effect.
Tremolo Mode	Selects whether tremolo is off, on, or faded in by the MIDI modulation wheel.
Tremolo Amount	Sets the amount of tremolo. Clicking the control's title bar restores the value to its default.
Vibrato & Tremolo Frequency	Sets the speed of both vibrato and tremolo.

Control	Function
Voice Panning Switch	Selects whether the instrument's voices are placed at different positions within the stereo field. Especially in combination with the [Unisono] control this can produce impressing spatial effects.
Voice Panning Amount	Sets the amount of voice panning. Clicking the control's title bar restores the value to its default.
Master 1/2	Defines the instrument's output level. Use the large middle knob to adjust the preset's maximum level; the smaller knob to the right controls the instrument's output amplitude in all patches.
Key Amp	Adjusts the amount of automated amplitude correction in respect to the synthesizer's pitch. Turn to the left for no influence of the pitch onto the output level, to the right to damp high pitches. This can be used to simulate the sound of analogue synthesizers.

8.3 Carbon



1.19 Carbon user interface

8.3.1 Introduction

The Carbon synthesizer is a fantastic-sounding machine with over a hundred sounds from a different planet. At its heart, Carbon is a powerful subtractive synthesizer with four oscillators, copious modulation, and a filter. We even wouldn't hesitate to say that Carbon has one of the best-sounding digital filters you've heard. In fact, Carbon has a choice of eleven different filter types and designs, each painstakingly shaped for silky smoothness across the entire frequency range. Carbon's LFOs include innovative features such as the possibility to vary their frequency slightly in each voice for an organic, living sound. Each one of Carbon's breathtaking pads, cutting leads, ripping basses, and quivering atmospheres will convince you that all life is based on Carbon.

8.3.2 Quick Start

Even though Carbon's panel may show many controls, its clear layout and function grouping make operation easy. Simply step through some presets at random to get a feel for the type of sounds that Carbon can make. In the filter section, try modifying Cutoff and Resonance. You can even try choosing different filter types in the menu panel to the left of Cutoff to see how the different filters change the sound.

8.3.3 Signal Flow and Structure

Carbon's sound is routed as follows:

Oscillators > Filter > Saturator > Four-band EQ > Chorus > Delay

Carbon features four oscillators plus a noise generator. The level of each oscillator is set with the large knob on top of each oscillator channel, while LFO modulation of the amplitude can be switched on or off directly underneath this knob. Each of Carbon's four oscillators offer two different modes, with each mode making a unique sound. Oscillator 1 is focused on pulse-width modulation, switchable between square or sinusoidal waveforms. Oscillator 2 concentrates on saw waves, switchable between a massive 7-saw (seven parallel saw oscillators, each slightly detuned), or a single sawtooth waveform. Oscillator 3 is based on interrupting the waveform - with a quantized sine wave being "interrupted" in amplitude to create low-fi effects, while a sawtooth oscillator is "interrupted" in time for the familiar sync effect. Oscillator 4 is mainly (but not always) used as a sub oscillator, offering standard pulse and sine waveforms with pulse width modulation. Below each oscillator mode selector, the tuning of each oscillator

can be set in semitones and cents in the PitchShift section. The pitch of each oscillator can be modulated by either the second envelope (Env 2), or LFO 1. Oscillators 1 and 4 can have the width of their waveforms also modulated by Env 2 or LFO 1. The additional parameters in the oscillator section, such as "odet" (oscillator detune), "LCo," (loudness correction) etc., are used only for very fine shaping of the amplitude and range of modulation of the oscillators across the frequency spectrum. We warned you that Carbon was from another planet!

All four oscillators plus noise are summed and sent to Carbon's outstanding filter .The filter provides eight different filter modes where nearly every mode uses straight forward technology, mainly to emulate the warm and powerful sound of analogue filters. The modes Va, CLa and J/S offer an additional alternative mode which can be switched on by the Alt control below the main mode selector, increasing the effective number of modes to 11. Many of the filter types are very similar and offer a quick and easy way to modify the sound of each preset - simply select a preset at random, then choose a new filter for it. The main parameters of each filter mode - cutoff frequency, frequency bandwidth, resonance and feedback - can be set at the right of the mode selector. Below each parameter knob there are additional controls to modulate the respective value, including the possibility to map the MIDI velocity and the modulation wheel onto the filter's sound.

The sound comes out of the filter and goes right into a saturator and four-band EQ for even more fine-control over the sound's frequency spectrum. Besides a low shelf EQ, a high shelf EQ, and peak EQ with adjustable center frequency, there is also an additional highpass filter used to increase the sound's brightness.

8.3.4 Modulation

Carbon has two envelopes and two LFOs. Envelope 1 always controls the amplitude, while Envelope 2 can control different parameters, such as oscillator volume, oscillator detune, filter cutoff, filter resonance, filter bandwidth, and filter feedback delay. There is no modulation matrix in Carbon - all modulations are set in their respective sections. For instance, to modulate filter cutoff by Envelope 2, simply look underneath the Cutoff knob in the filter section and you'll see a small knob for E2 and a button to activate the modulation amount. Carbon's LFOs offer valuable sound-shaping functions. Using the FSpr knob in either LFO section adds a frequency spread to the LFO frequency across multiple voices. This means that if you play a chord, each note would have slightly different LFO speeds for a shimmering, organic sound. Also, the modulation of the LFO can be delayed, for instance to program a vibrate which takes a few moments before coming in.

8.3.5 Global Parameters

Parameters effecting the entire synth are set in the Global section, such as Mono/Poly (set with the Mono button), tuning, unison spread, and glide. Hold lets you specify how long each note will last for, no matter how long the actual MIDI note was held down for (value set in mSec underneath the Hold button). Old controls how much pitch randomness is introduced to emulate vintage hardware.

8.4 Carbon



1.19 Carbon user interface

8.4.1 Introduction

The Carbon synthesizer is a fantastic-sounding machine with over a hundred sounds from a different planet. At its heart, Carbon is a powerful subtractive synthesizer with four oscillators, copious modulation, and a filter. We even wouldn't hesitate to say that Carbon has one of the best-sounding digital filters you've heard. In fact, Carbon has a choice of eleven different filter types and designs, each painstakingly shaped for silky smoothness across the entire frequency range. Carbon's LFOs include innovative features such as the possibility to vary their frequency slightly in each voice for an organic, living sound. Each one of Carbon's breathtaking pads, cutting leads, ripping basses, and quivering atmospheres will convince you that all life is based on Carbon.

8.4.2 Quick Start

Even though Carbon's panel may show many controls, its clear layout and function grouping make operation easy. Simply step through some presets at random to get a feel for the type of sounds that Carbon can make. In the filter section, try modifying Cutoff and Resonance. You can even try choosing different filter types in the menu panel to the left of Cutoff to see how the different filters change the sound.

8.4.3 Signal Flow and Structure

Carbon's sound is routed as follows:

Oscillators > Filter > Saturator > Four-band EQ > Chorus > Delay

Carbon features four oscillators plus a noise generator. The level of each oscillator is set with the large knob on top of each oscillator channel, while LFO modulation of the amplitude can be switched on or off directly underneath this knob. Each of Carbon's four oscillators offer two different modes, with each mode making a unique sound. Oscillator 1 is focused on pulse-width modulation, switchable between square or sinusoidal waveforms. Oscillator 2 concentrates on saw waves, switchable between a massive 7-saw (seven parallel saw oscillators, each slightly detuned), or a single sawtooth waveform. Oscillator 3 is based on interrupting the waveform - with a quantized sine wave being "interrupted" in amplitude to create low-fi effects, while a sawtooth oscillator is "interrupted" in time for the familiar sync effect. Oscillator 4 is mainly (but not always) used as a sub oscillator, offering standard pulse and sine waveforms with pulse width modulation. Below each oscillator mode selector, the tuning of each oscillator

can be set in semitones and cents in the PitchShift section. The pitch of each oscillator can be modulated by either the second envelope (Env 2), or LFO 1. Oscillators 1 and 4 can have the width of their waveforms also modulated by Env 2 or LFO 1. The additional parameters in the oscillator section, such as "odet" (oscillator detune), "LCo," (loudness correction) etc., are used only for very fine shaping of the amplitude and range of modulation of the oscillators across the frequency spectrum. We warned you that Carbon was from another planet!

All four oscillators plus noise are summed and sent to Carbon's outstanding filter .The filter provides eight different filter modes where nearly every mode uses straight forward technology, mainly to emulate the warm and powerful sound of analogue filters. The modes Va, CLa and J/S offer an additional alternative mode which can be switched on by the Alt control below the main mode selector, increasing the effective number of modes to 11. Many of the filter types are very similar and offer a quick and easy way to modify the sound of each preset - simply select a preset at random, then choose a new filter for it. The main parameters of each filter mode - cutoff frequency, frequency bandwidth, resonance and feedback - can be set at the right of the mode selector. Below each parameter knob there are additional controls to modulate the respective value, including the possibility to map the MIDI velocity and the modulation wheel onto the filter's sound.

The sound comes out of the filter and goes right into a saturator and four-band EQ for even more fine-control over the sound's frequency spectrum. Besides a low shelf EQ, a high shelf EQ, and peak EQ with adjustable center frequency, there is also an additional highpass filter used to increase the sound's brightness.

8.4.4 Modulation

Carbon has two envelopes and two LFOs. Envelope 1 always controls the amplitude, while Envelope 2 can control different parameters, such as oscillator volume, oscillator detune, filter cutoff, filter resonance, filter bandwidth, and filter feedback delay. There is no modulation matrix in Carbon - all modulations are set in their respective sections. For instance, to modulate filter cutoff by Envelope 2, simply look underneath the Cutoff knob in the filter section and you'll see a small knob for E2 and a button to activate the modulation amount. Carbon's LFOs offer valuable sound-shaping functions. Using the FSpr knob in either LFO section adds a frequency spread to the LFO frequency across multiple voices. This means that if you play a chord, each note would have slightly different LFO speeds for a shimmering, organic sound. Also, the modulation of the LFO can be delayed, for instance to program a vibrate which takes a few moments before coming in.

8.4.5 Global Parameters

Parameters effecting the entire synth are set in the Global section, such as Mono/Poly (set with the Mono button), tuning, unison spread, and glide. Hold lets you specify how long each note will last for, no matter how long the actual MIDI note was held down for (value set in mSec underneath the Hold button). Old controls how much pitch randomness is introduced to emulate vintage hardware.

8.5 Equinoxe Deluxe



4.42 Equinoxe Deluxe user interface

8.5.1 Introduction

Equinoxe Deluxe simulates special features of old analogue synthesizers and effects with Reaktor's new Core technology. First, there is an Octave Lock mode emulating analogue synthesis techniques (see below for details). Second, the Ensemble and KleinStein effects provide high quality sounds that aim on reproducing the warmth of their old predecessors; particularly the KleinStein phaser is of high technical complexity within the Core cell.

8.5.2 String Section

The Strings section is the central sound generator of this instrument. Essentially, this is a sawtooth oscillator shaped by an amplitude envelope and a 4-pole ladder lowpass filter. It features an additional Octave Lock mode simulating an old analogue technique that used twelve oscillators, each one dedicated to one pitch class: By this, octaves are exactly of the doubled frequency, and there occurs no beating while the harmonics are shaped in a special way.

Control	Function	Range
Octave Lock	Switches the Octave Lock feature on or off.	on/off
Cutoff	Controls the cutoff frequency of the lowpass filter shaping the sawtooth signal in semitones.	20..120
Resonance	Sets the resonance of the ladder lowpass filter.	0..1
Envelope Mode	Switches between three envelope modes. They differ regarding their behaviour if a MIDI note is triggered while another one is still playing. In [Poly] mode the old note isn‘t changed while the new one starts playing. In [Mono] mode the old note is stopped and the new pitch is played. This is also true for the [Reset] mode, but here the amplitude envelope (see [Attack] and [Release]) is also re-triggered by the new note event.	-
Attack	Adjusts the attack time of the envelope generator controlling the sawtooth oscillator‘s amplitude.	0...80
Release	Adjusts the release time of the envelope generator controlling the sawtooth oscillator‘s amplitude.	0...80

8.5.3 Ensemble, KleinStein, and Delay Sections

The Ensemble, KleinStein and Delay sections crucially enrich the basic sawtooth sound and make Equinoxe Deluxe to an analogue-sounding string instrument. While the Ensemble section applies a high quality chorus effect to the sound the KleinStein section is based on a modulated phaser. The Delay, finally, provides two (stereo) lines with independent controls.

Section	Controller	Function	Range
Ensem-ble	Rate 1 / 2	Controls the rate of the LFO that modulates the delay time controlled by [Time].	-40...10
	Amount 1 / 2	Sets the amount of modulation applied to the main delay time.	0...1
	Time	Adjusts the main delay time in milliseconds.	1...20
	Mix	Controls the balance between dry, unprocessed signal (at the left) and wet effect signal (at the right).	0...1
Klein-Stein	Rate	Sets the rate of the LFO modulating the phaser's center frequency.	0.01...3.16
	Mix	Controls the balance between dry, unprocessed signal (at the left) and wet effect signal (at the right).	0...1
	Colour	Switches between two phaser modes. If on, the effect is stronger.	on / off
Delay L/R	Time	Sets the delay time in milliseconds	100...1000
	Feedback	Sets the amount of feedback.	0...1
	LFO Rate	Sets the rate of the LFO that modulates the main delay time in Hertz.	0.1...1
	LFO Amount	Controls the amount of delay time modulation in milliseconds.	0...20
	Mix	Controls the balance between dry, unprocessed signal (at the left) and wet effect signal (at the right).	0...1

8.6 FM4



4.43 FM4 user interface

8.6.1 Introduction

FM4, as tells the name, is a synthesizer based on the familiar frequency modulation technique. It contains four operator sections; their outputs are not only mixed and sent to the soundcard but they also modulate each others‘ frequency, controlled by a flexible routing matrix. While this is a well known way to generate sounds since the 1980‘s, the FM4 excels in its implementation of the basic idea by using Reaktor Core.

The instrument‘s control logic and logical structure is implemented in classical Reaktor, the audio processing in Reaktor Core. By doing so a huge Core Cell, implementing the complete functionality of the instrument, is avoided, instead this results in several re-usable Cells. But

why is the audio processing realized in Core Cells? This makes it possible to enrich Reaktor‘s default capacities with specialized high quality audio processing. Frequency modulation, for instance, is implemented as phase modulation within this instrument‘s Core Cells. In contrast to the algorithm used within the standard Reaktor modules phase modulation results in a slightly softer sound that additionally remains constant in its modulation amount all over the MIDI pitch range. As another example the LFO is built as a Core Cell allowing for special features like the One Shot capacity while maintaining the instrument‘s full polyphony. The chorus, as a last example, uses Core Technology to provide an effect of high quality without consuming loads of CPU power. See below a detailed description of the instrument’s controls and their functions.

8.6.2 Operator Sections 1-4

The four Operator sections of the instrument‘s panel control the sine oscillators capable of frequency modulation (FM). Each oscillator is defined by its frequency and its amplitude, the latter one specified by an envelope generator. When an oscillator modulates the frequency of another oscillator (controlled by the FM Sources section) the amount of modulation is – obviously – scaled by the amplitude, so the envelope generators also modulate the FM amount.

Control	Function	Range
A (Attack)	Sets the attack time of the envelope controlling the operator’s amplitude.	-20...+80
D1 (Decay 1)	Sets the first decay time of the envelope controlling the operator’s amplitude. It starts after the envelope generator has passed its attack phase.	0...100
B (Break-point)	Sets the level of the breakpoint. This level – relative to the maximum level after the attack phase – is reached after the first decay time and is used as starting point for the second decay.	0...1
D1 (Decay 2)	Sets the first decay time of the envelope controlling the operator’s amplitude. It starts after the envelope generator has decayed to the breakpoint. At high settings the envelope becomes similar to a ADSR envelope with B controlling the sustain level.	0...100
R (Release)	Sets the release time of the envelope controlling the operator’s amplitude. This is used after the triggering MIDI signal has ceased.	0...100

Control	Function	Range
Vel (Velocity)	Adjusts the amount of MIDI velocity applied to the envelope's amplitude. At a value of 1 the envelope's maximum level is defined completely by the triggering MIDI signal's velocity. At a value of 0 the level is always at its maximum, independent of the MIDI velocity.	0...1
KA (Keyboard to Amplitude)	Controls the influence of the MIDI note's pitch on the operator's amplitude. At a value of +1 the amplitude rises by 6 dB per octave; at a value of -1 it falls by the same amount. At a MIDI pitch of 60 (C3) there is no influence on the amplitude.	-1...+1
KT (Keyboard to Times)	Controls the influence of the MIDI note's pitch on the envelope's transition times. At positive values the times become longer ar high keys; at negative values they become shorter. At a MIDI pitch of 60 (C3) there is no influence on the envelope times.	-1...+1
Ratio	Defines the ratio between the instrument's main pitch and the operator's frequency. Musically speaking, this adjusts the harmonic of the fundamental frequency to be played, i.e. at a value of 1 the first harmonic is played etc. Integer values can be controlled at the left side of the control, fractional values at its right. For a fixed oscillator frequency set the Ratio to zero.	0...64
Fix(ed) Freq(uency)	Adjusts a frequency in Hertz that, as an offset, is added or subtracted to/from all other frequencies of this operator. At the right side a value can be adjusted, at the left side the decade is controlled.	0...9955
Fix Freq Sign	Specifies whether the fixed frequency is added or subtracted from the variable frequency.	- / +
Phase	Sets the phase in degrees at which the operator's oscillator starts when synchronization to the keyboard signal is activated.	0...360
Key Sync	Switches synchronization of the operator's oscillator to MIDI Note On signals on or off. If on, the oscillator starts at the position adjusted by the [Phase] control.	on / off
Out (Output Volume)	Sets the output volume of this operator. This volume is used when mixing the four operators. The FM signal is not affected by this control; to adjust the amount of frequency modulation, the FM Sources Section has to be used.	0...1

8.6.3 FM Sources Section

The FM Sources section controls the modulation matrix: It defines a signal as a mix of all four Operator sections to be used as modulation source, independently for each oscillator. (this also includes self-modulation.) Each operator‘s signal can additionally be scaled by a MIDI modulation source (like the pitchbend wheel) or the instrument‘s polyphonic LFO section by using the controls of the FM Mod section.

Control	Function	Range
1, 2, 3, 4	Sets the amount of phase modulation applied to one of the operators (specified by the row of the Control) by one of the other operators (specified by the number). The amount rises quadratically.	0...10
self	Sets the amount of self-modulation for each of the four oscillators.	0...10
Mod 1, 2, 3, 4	Scales the modulation signal from the FM Mod Source applied to the level of operators 1...4 as sources of the FM matrix (for all destinations). At 0 the full operator level is sent to the matrix. At higher values the level is dynamically attenuated by the FM Mod Source.	0...1
Source	Selects a source, e.g. the LFO or the LFO scaled by the MIDI modwheel, that is routed to control the sources of the FM matrix.	-
CC Nr	Adjusts the MIDI CC number if CC is selected as [Source]. The MIDI modwheel can be addressed by CC 1.	0...127

8.6.4 LFO Section

The LFO section is polyphonic, i.e. each note triggered via MIDI can have its own, independent LFO signal with specific rate and phase. The LFO has a special One Shot feature that stops the modulator after half a period, reducing it to a AD envelope generator (to use this the [Key Sync] has to be activated.) The modulation signal can be used to scale the FM amounts using the FM Mod section, but it can also be routed directly to the oscillators‘ pitch and amplitude, resulting in vibrato and tremolo.

Control	Function	Range
Waveform	Selects the waveform of the LFO. The negative waveform is similar to the positive one but starts at a phase of 180 degrees; this can be particularly useful when using the [One Shot] feature.	-
Asymetry	Controls the shape of the waveform; higher values shift the point of highest amplitude to the end of the waveform.	0...1
Rate	Sets the LFO's oscillation frequency as a MIDI pitch. This is the fundamental rate modulated by the [Keyboard To Rate] value.	0.05...20
KR (Keyboard To Rate)	Controls the amount of rate modulation by the triggered MIDI note's pitch. At a value of 1 the rate is doubled per octave, using the original [Rate] setting at C3 (MIDI pitch 60). This shows no effect if [Mono] is activated.	-1...+1
Mono	Deactivates the polyphonic mode. If on, all voices use the same LFO signal; if off, each voice is assigned to an independent LFO.	on / off
Key Sync	Switches re-triggering of the LFO by an incoming MIDI Note-On.	on / off
One Shot	If on, the LFO stops oscillating after half a period. This can be used to transform the LFO into an AD envelope generator. It doesn't work if [Key Sync] is off.	on / off
Pitch Amount	Adjusts the amount of pitch modulation as the vibrato's range in semitones. The MIDI Modwheel's vibrato amount is added to this value (see [MW Pitch Amount] in the Voice Mode section).	0...100
Amplitude Amount	Adjusts the amount of amplitude modulation as the tremolo's intensity.	0...1

8.6.5 Voice Mode Section

The Voice Mode section specifies the way the instrument handles polyphony, i.e. several notes that are triggered simultaneously. Beside the polyphonic mode there are several monophonic modes that feature additional functions: legato (no re-triggering of the envelope generators if a note is held when another one is pressed) and portamento (pitch glide). The Output section spreads those voices within the stereo panorama and controls the main panning and volume.

Control	Function	Range
Pitchbend Range	Adjusts the range by which the instrument's pitch can be changed via the MIDI Pitchbend Wheel in semitones.	0...24
MW Pitch Amount	Sets the range of pitch modulation via the LFO signal in semitones. This vibrato can be scaled by the MIDI Modwheel.	0...5
Voice Mode	Switches between different voice assignment modes. There is one polyphonic mode (i.e. several notes can be played simultaneously) and four monophonic modes (where only one note can be played at a time; a new note automatically stops the previous one). In Legato mode the envelope generators are not re-triggered upon a new MIDI Note On message if another note is already playing, only pitch and velocity are changed. While the [1] modes only use one voice the [All] modes combine the available voices to one unisono chorus that can be controlled with the [Detune] knob.	-
Transpose	Shifts the instruments main pitch in steps of a semitone.	-48...+48
Tune	Shifts the instruments main pitch in steps of a cent.	-0.5...+0.5
Glide	Sets the time used to reach a new note's pitch in seconds.	0...4
Detune	Sets the amount of detuning. In all monophonic modes this specifies the max. interval in semitones between the different voices ment to play unisono. If [Voice Mode] is set to polyphonic, this specifies the spread of the tempered MIDI tuning in semitones per octave.	0...12

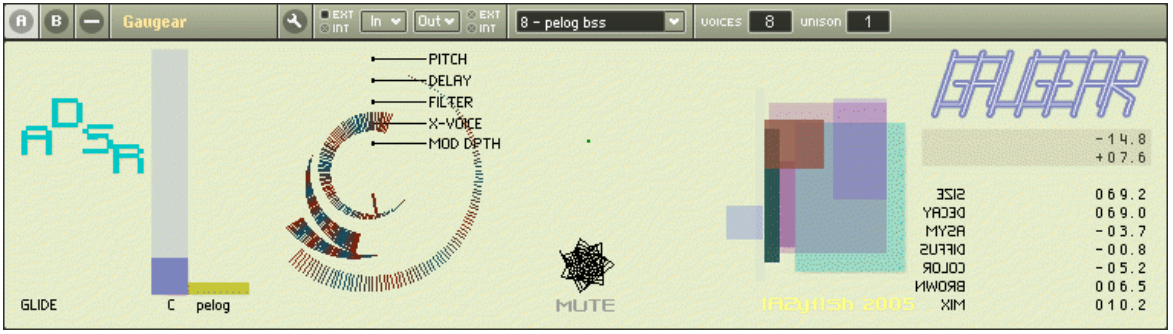
8.6.6 Effects Sections

The three effects, finally, provide additional sound shaping. The Equalizer, Chorus/Flanger and Delay sections on the panel are straightforward, yet they aim for high quality sound.

Section	Control	Function	Range
Equalizer	Low	Controls the amount of boost/cut applied to the frequencies below [Low Frequency] in dB.	-24...+24
	(Low) Freq(uency)	Specifies the frequency below which the signal is altered in Hz.	30...1000
	Mid	Controls the amount of boost/cut applied to the frequencies around the [Mid Frequency] in dB.	-24...+24

Section	Control	Function	Range
	(Mid) Freq(uency)	Specifies the center frequency for the middle band in Hz; the bandwidth can be controlled with [Mid Width]	100...5000
	(Mid) Width	Adjusts the width of the middle band.	0.01...1
	High	Controls the amount of boost/cut applied to the frequencies above [High Frequency] in dB.	-24...+24
	(High) Freq(uency)	Specifies the frequency above which the signal is altered in Hz.	400...15000
Chorus / Flanger	Left / Right Delay	Controls the average delay times of the left and right channel effect units in milliseconds. The delay times are modulated by an LFO around these values.	0.1...20
	Depth	Adjusts the amount of modulation applied to the delay times as a part of the average delay time.	0...1
	Rate	Sets the modulation speed as rate of an internal LFO modulating the delay time in Hertz.	0...10
	Feedback	Controls the amplitude of the delayed signal mixed with the unprocessed incoming signal. Negative values shift the wet signal's phase by 180 degrees.	-1...+1
	Mix	Adjusts the balance between dry, unprocessed sound (at 0) and wet, delayed sound (at 1).	0...1
Stereo Delay	Left / Right Delay	Controls the delay time for both channels. If [Sync] is activated this time can be adjusted according to the current MIDI tempo; else it can be set in milliseconds.	0.1...1000
	(Clock) Sync	Switches between free delay time control (off) or synchronized to the MIDI tempo (on).	on / off
	Feedback	Controls the amplitude of the delayed signal mixed with the unprocessed incoming signal.	0...1
	X Amount	Controls the amount of cross feedback as percentage of the main feedback amount (see [Feedback]).	0...1
	High Cutoff	Controls the cutoff frequency of a lowpass filter in Hz applied to the feedback signal.	100...20000
	Mix	Adjusts the balance between dry, unprocessed sound (at 0) and wet, delayed sound (at 1).	0...1

8.7 Gaugear



4.44 Gaugear user interface

8.7.1 Introduction

Gaugear is an experimental monophonic synthesizer: its sound explores sonic worlds that cannot be classified by MIDI pitches or a quantized clock, and its unique user interface invites new ways of sound design by omitting conventional knobs or faders. The basic structure of the instrument is simple to understand: The tone generator's signal is shaped by an amplitude envelope and passed through reverb and equalizer.

8.7.2 Overview



4.45 Envelope controls

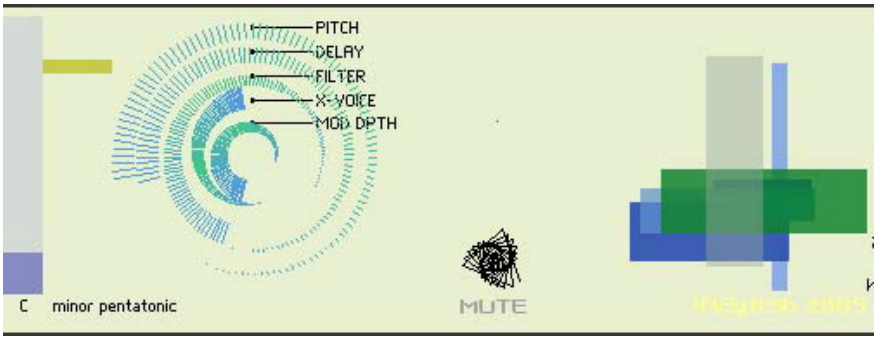
The controls of the additional units are localized at the panel's left (four letters for the envelope's parameters: A, D, S, R).



4.46 Damping and reverberation

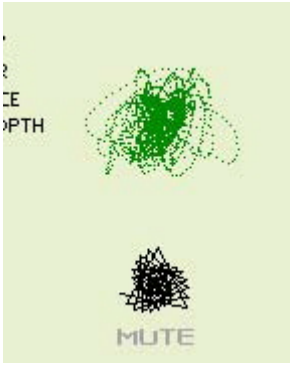
To the right you'll find high and low damping as well as various reverberation settings.

In between there are the experimental controls of the tone generator. Gaugear's signal is generated by eight parallel FM/AM pulse oscillators with subsequent delays and lowpass filters.



4.47 Oscillator controls

The circular controls at the panel's left adjust the main parameters of these oscillators (like pitch, delay and filter cut-off): vertical movements with the left mouse button pressed alter the position of each control ring, horizontal movements change its spread and polarity. Thus, a range of values is adjusted for each parameter out of which one discrete value is picked by the rectangular control at the right. Each of its eight boxes represents one of the eight parallel oscillators; by moving the mouse within this area the boxes can be moved and their colors can be changed, visualizing a different set of values taken from the range of values provided by the circular controls. This sounds complicated; but a short tutorial will help using the instrument. Before the tutorial, five more controls should be explained.



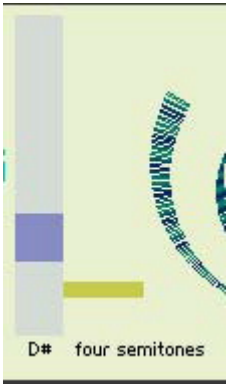
4.48 Volume control

A small display between the circular and the rectangular controls shows Lissajous figures of the signal; below, the instrument's volume can be adjusted by dragging the geometrical figure up and down.



4.49 Glide control

The glide label functions as a fader to set the amount of portamento between tied notes.



4.50 Pitch control

Beside the envelope sliders two controls select the pitch set to which the pitch control is mapped; e.g. by selecting "b" and "major pentatonic" (topmost entries of both controls) only pitches of the pentatonic major scale with b as fundamental pitch are used for the eight oscillators.

Obviously, this interface does not allow you to assign a particular set of parameters to one of the eight oscillators: you have to trust your ears. For a first exploration of this instrument's capacities set the reverb's mix control to zero to listen to the isolated dry signal, then place all five circular controls at the "bottom" and set them to minimum spread. Now the instrument will generate a relatively dull sound upon MIDI note events; as the range of each parameter is

small, movements within the rectangular control will only show small effects on the sound. Drag the pitch ring to the right – it will spread around the left side of the control, and the oscillators' pitches will also spread. If the control ring reaches the upper center, it will wrap around, producing the lowest pitches after the highest ones. Movements within the box area now change the sound audibly as the controller picks values from a larger pitch range. Drag the pitch ring a bit to the bottom: the complete spectrum will be lowered. Finally, drag the delay ring to the left to add some long delay times and add some reverb – an evolving atmosphere has been created.

By stepping through the instrument's presets you'll hear a great variety of possible sounds. There are deep, percussive bass patches as well as light, ethereal pads. Use them as starting points for your own explorations into the instrument's sonic world (always remember that you can easily alter each preset by moving the boxes of the rectangular display while preserving the main character) or use the random functions: Double-click the center of either the circular or the rectangular display to randomize its settings. (By clicking once within the center of the circular control you can set its colors to random values, but this does not influence the sound.)

Below you'll find is a reference of all available instrument controls; but in order to design new sounds, simply ignore those details and listen.

Section	Control	Function
Envelope	Attack	Sets the attack time of the amplitude envelope generator.
	Decay	Sets the decay time of the amplitude envelope generator.
	Sustain	Sets the sustain level of the amplitude envelope generator.
	Release	Sets the release time of the amplitude envelope generator.
Glide		Controls the amount of pitch glide (portamento) that occurs when notes are tied.
Pitch Map	Fundamental Pitch	Selects the fundamental pitch of the scale selected by [Scale Select].
	Scale Select	Selects a scale out of a vast range of pitch sets. The pitches of all oscillators are mapped to this scale.
Tone Generator	Pitch	Controls the position and range of pitches available for the oscillator units (see also [Scale Select]). The left side provides high pitches; the right side, low pitches.
	Delay	Controls the position and range of delay times available for the oscillator units. The left side provides short delay times; the right side, long ones.

Section	Control	Function
	Filter	Controls the position and range of cutoff frequencies available for the oscillator units' lowpass filters. The left side provides high frequencies; the right side, low ones.
	X-Voice	Controls the routing of the modulation signals. Each oscillator's pitch and amplitude is modulated by another oscillator's output; this control defines the cross-modulation matrix.
	Modulation Depth	Controls the amount of modulation applied to the oscillators' frequency and amplitude. The ring's position adjusts the amount of FM (low FM values at the right side); its length adjusts the amount of AM (low AM values at short length settings).
	Setting Morph	Morphs through the oscillator settings. Each box's position and color represents a set of parameters controlling one of the eight oscillators. The range of available settings is controlled by the circular controls of the Tone Generator like [Pitch] and [Filter].
Volume	Level	Adjusts the main output level.
	Mute	Mutes the entire instrument.
EQ	High	Controls the amount of amplification or attenuation applied to the signal's high frequencies.
	Low	Controls the amount of amplification or attenuation applied to the signal's low frequencies.
Reverb	Size	Sets the size of the room simulated by the reverberation unit.
	Decay	Sets the time that passes until all reverberation signals have decayed.
	Asymmetry	Places the signal to the left (negative values) or right (positive values) of the simulated room's center.
	Diffusion	Sets the amount of diffusion applied to the room simulation.
	Color	Controls the internal damping.
	Brown	Controls the internal room size modulation.
	Mix	Crossfades between the unprocessed signal and the reverb signal.

8.8 Green Matrix



1.20 Green Matrix user interface

8.8.1 Introduction

Green Matrix is the MIDI-playable brother of Blue Matrix. Both use the same analog-style synthesis engine and modulation matrix to generate some amazing and challenging sounds. Its sound engine features two multiple-waveforms oscillators, a multi-mode filter, multi-mode distortion, and finally a second multi-mode filter. A beat-synced delay and a diffusion delay round out the effects. A complete modulation matrix lets you flexibly route the two envelopes, LFO, four channels of sequenced modulation, and the usual MIDI controllers to every important synthesis parameter.

8.8.2 Quick Start

Plug in a MIDI keyboard and flip through the presets! Go ahead and experiment with the synthesizer controls - try changing LFO speed or cutoff frequency, for example.

8.8.3 Structure and Signal Flow

Green Matrix's synth engine is based on traditional analog synthesis. It has three oscillators, two filters, distortion, and effects. Both Green Matrix and its sequencer-driven brother Blue Matrix are created from Reaktor 4's Classic Modular macro library which puts the sound quality and capabilities of the classic modular synths into your computer. For more information on building your own instruments with the Classic Modular macro library, please check your Reaktor 4 user's guide. The Classic Modular library is included on the Reaktor 4 CD.

Oscillators 1, 2, and 3 each offer a different waveform selection. Osc 1 offers the traditional sawtooth, pulse, triangle, sine, and impulse waveforms, while Osc 2 lets you choose between bipolar ramp, bipolar pulse, pulse, parabolic, and noise. The bipolar waveforms' shape can be modified in realtime with the small Shape knob. Osc 3.

Osc 1 can be frequency modulated by Osc 2 or 3 for FM sounds (with the small FM knob and switch below), while Osc 2 and 3 can be ring modulated by Osc 1 for metallic and "gong" sounds (with the small Ring switch). Osc 1 can also be synced to Osc 2, with the choice between hard, soft, and MIDI gate-activated modes. Since Osc 1, 2, and 3 each offer unique features, they can be switched on or off to save CPU with the small on switch on their left sides. The base pitch and fine tuning of each oscillator is done with their respective Pitch and Fine controls.

The two oscillators feed into Filter 1 which offers a choice of 12 or 24 dB/octave lowpass, highpass, bandpass, peak EQ, and notch modes. Cutoff and Reson adjust the cutoff frequency and resonance, respectively, and the FM knobs controls how much Osc 1 or Osc 2 will frequency-modulate the cutoff frequency (settable with the small switch to the right of the FM knob). Pkey determines how the filter tracks the pitch of the notes from the sequencer. When Pkey is set to 1, then the filter cutoff will exactly follow the pitch of the sequencer notes, meaning that higher notes will be brighter, and lower notes will be duller. If set to zero, then the cutoff frequency won't change. The other knobs, GainC, Limit, F foll, and Rel all control the fine-tuned shaping of the filter across the frequency and resonance spectrum. GainC con-

trols how much loudness reduction will be applied at high resonance values. Limt, F foll, and Rel control a built-in resonance limiter to avoid unwanted harmonic thumps with high resonance values.

Filter 2 is similar in layout to Filter 1. There are a choice of four lowpass filters, two bandpass filters, and a combined bandpass/lowpass filter (BL4). In addition to the controls you're now familiar with from Filter 1, Filter 2 also offers the additional possibility to trigger self-resonance with a built in click. The Click knob controls how loud the click is, and the Vel knob determines how the click responds to velocity.

After Filter 2, the sound passes through a distortion unit then a chorus/ delay effect. Note that an extended version of the chorus/delay effect is available in the Fusion Reflections ensemble. The Delay/Chorus consists of four stereo modulation delays. The main delay time is controlled by the Delay knob. Diffusion sets the inaccuracy of the delay times to "spread" the sound.

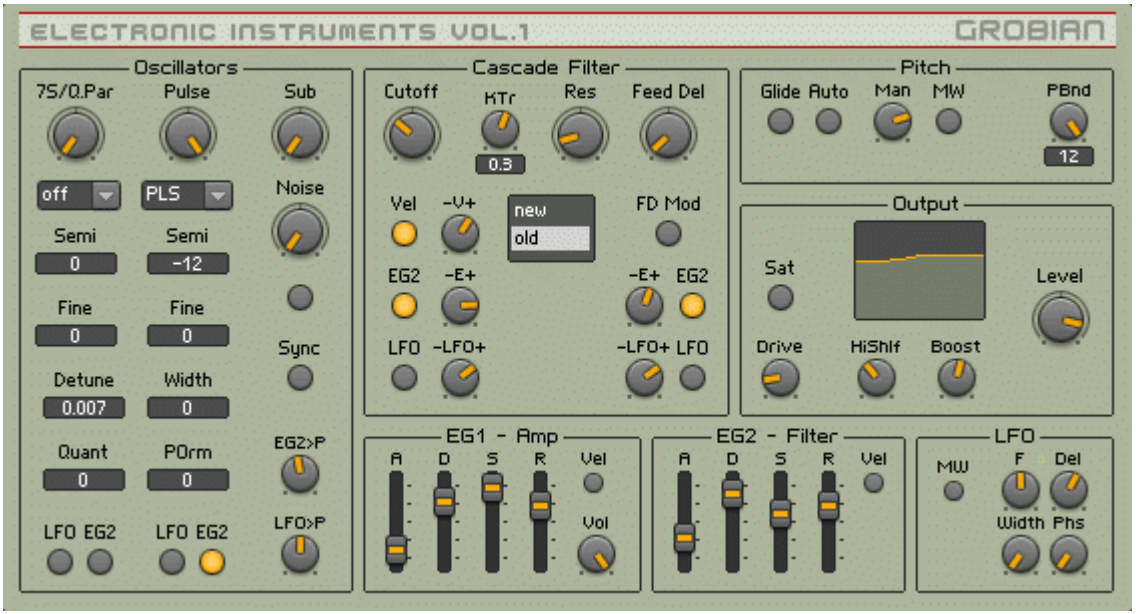
8.8.4 Modulation

Green Matrix features three standard ADSR envelopes and a two LFOs that syncs to tempo. All three envelopes are velocity sensitive: Vel>A sets how much the velocity will affect attack time. The speed of the LFO can be set in musical note values, or freely. When Unit is set to 0, then the Frequency of the LFO is set in Hz. When Unit is set to any other value, the frequency of the LFO takes on the musical units set by Unit. For example, if Unit is 96 and frequency is 1, then the LFO will be set to the very fast speed of 1/96th note per cycle. The display underneath the Freq knob depends on how Unit is set: if Unit is set to 0, then the display automatically shows values in Hz, and if Unit is set to a musical value, then the Freq display will automatically update. Width and Phase control the shape of the LFO, which can be selected by the switch to the right. Key S determines if the LFO will synchronize to note-on events, as delivered by the sequencer.

All of Green Matrix's modulation sources are routed in a comprehensive modulation matrix: The LFO, two envelopes, four modulation channels from the sequencer, and typical MIDI controls like mod/pitchbend/ velocity can be routed to synthesizer destinations using the comprehensive modulation matrix. Switch to the "B" view of the synth to see the modulation matrix. You can easily switch to the B view by clicking the B button in the title bar of the synth. You can route the modulation to the following synth parameters: Oscillator pitch (osc 1, osc 2, or osc 1 + 2), osc 1 FM, osc 2 shape, osc ½ mix, filter cutoff (osc 1, osc 2, or osc 1 + 2), filter 1 FM, filter resonance (1 or 2), envelope parameters, and even distortion, delay time, and delay feedback. In short, nearly everything!

The modulation sources (LFOs, envelopes, etc) are arranged vertically. The modulation destinations (synth parameters such as filter cutoff, oscillator pitch, etc) are arranged vertically. Simply activate the switches for the combination you'd like to modulate. Underneath each modulation column is a switch to turn the modulation on or off, and a knob to set the modulation amount.

8.9 Grobian



2.4 Grobian user interface

Introduction

Grobian is a bass synthesizer that can produce very powerful and distorted sounds. It s excellent for drum and bass, industrial, and techno. Within the framework of a standard two-oscillator/one-filter design, it features an unconvential seven-sawtooth oscillator, and an analog-style filter.

Oscillators

Grobian features two unique oscillators, plus a sub oscillator and a noise generator. The first oscillator can be switched between a stack of seven sawtooth oscillators or a single sine oscillator with bit quantization. The second oscillator is a Pulse (square wave) generator that can be ring modulated. The sub oscillator is always a sine wave.

- <7Saw> Sets the amplitude of the saw oscillators, i. e. their level when mixed with the other sound sources. As those are used the seven saw oscillators (seven are used parallely to achieve a more fat sound), a pulse oscillator, a noise generator and a sine sub oscillator.
- <Pulse> Sets the amplitude of the pulse oscillator.
- <Mode Saw> Sets the mode of the saw oscillator: It can be turned off, used normally with seven parallel saw oscillators or be replaced by a sine oscillator with a quantizer.
- <Mode Pls> Sets th mode of the pulse oscillator. It can be turned off, used normally with a pulse oscillator or fed through a Ring Modulator whose other input is connected to a sine oscillator. (See <POrm>).
- <PitchShift> Sets the shift of the pitch in semitones of the saw oscillators resp. the pulse oscillator. The value of this control is added to the pitch of the midi input.
- <Fine> Adjusts the pitch of the saw oscillators resp. the pulse oscillator in fine steps for detuning. Detune one oscillator slightly for sweeping effects.
- <Quant> Sets the stepsize of the quantizer used by the sine oscialltors (See also <Mode Saw>). Greater steps result in more distortion.
- <Width> Sets the pulse width of the pulse oscillator.
- <Step> Sets the amount of detuning of the seven saw oscillators. Those oscillators do not use the same pitch, instead their pitches, grouped around the center pitch, vary by steps whose size is adjusted by this control.
- <POrm> Sets the shift of the pitch of the sine oscillator whose signal is used for ring modulation with the pulse oscillator.
- <lfo7>, <eg7> Control wether the lfo or the second envelope EG2 modify the pitch of the saw oscillators

- <lfo2>, <eg2> Control wether the lfo or the second envelope EG2 modify the pitch of the pulse oscillator.
- <Sub> Sets the amplitude of the sine sub oscillator.
- <Noise> Switches the noise generator on or off.
- <Noise Lvl> Sets the amplitude of the noise generator.
- <EG2>P> Sets the amount of pitch modulation by the second envelope EG2 (See also <eg7> and <eg2>).
- <LFO>P> Sets the amount of pitch modulation by the low frequency oscillator (See also <lfo7> and <lfo2>).

Cascade Filter

To get the warm and rich sound associated with analog filters, Grobian s filter uses a neat trick where the saturated output is sent through the filter again after a very short delay. In addition to the usual Cutoff and Resonance controls, you ll notice another large knob to control the delay in the feedback path of the filter. Not only can Grobian s filter self-resonate (a very unusual feature in a digital filter), but the sound remains rich and warm even with very high resonance values (an even more unusual feature in a digital filter!). You can switch between two filter modes: New and Old. The technical differences between the two are slight, but New sounds warmer and Old sounds more resonant.

- <Cutoff> Sets the basic cutoff freqeuncy of the filter.
- <KTr> Sets the amount of key-tracking. Positve values result in a cutoff freqeuncy that follows the pitch of the oscillators, at mid poisition there is now key tracking, i. e. the cutoff freqeuncy is not affected by the midi pitch, and negative values produce an inverted key tracking, i. e. the higher the pitch of the oscillator, the lower the cutoff frequency.
- <Vel>, <EG2>, <LFO> Switch the modulation of the cutoff frequency (resp. the feed delay, see <Feed Del>) by the correspondent source (midi velocity, envelope EG2, LFO) on or off.
- <-E+>, <-LFO+> Set the amount of modulation by the correspondent source. Turn right for much modulation, turn left for inverted modulation, i. e. when the input signal is positive, the final modulating signal is negative and vice versa.
- <Res> Sets the amount of resonance of the filter.

- <FiltMode> Switches between two different filter modes. Their technical differences are difficult to explain: The sound of the new one is more fat, the old one is more sensitive to cutoff frequency, resonance and feed delay changes.
- <Feed Del> Sets the amount of delay applied to the feedback. The feedback is used to generate the resonance, try different settings at different cutoff and resonance values.
- <FD Mod> Switches the modulation of the feedback delay time on or off.

Envelopes: EG1 and EG2

The two envelope generators in Grobian are both standard ADSR envelopes. EG1 always controls volume, while EG2 can control a variety of parameters, such as filter cutoff or oscillator pitch. The EG2 control assignments are made in the OSCILLATORS or CASCADE FILTER sections, with the —E+ controls.

- <A>, <D>, <S>, <R> Set the attack, decay and release times as wqell as the sustain level of an ADSR-envelope. There are two envelope generators: EG1 is used for main amplification, EG2 for modulation of the oscillator's pitch and the filter.
- <Vel> Switches the sensitivity to the midi velocity on or off.
- <Vol> Sets the amount of amplification applied to the envelope signal. As it is used for the main amplification of the sound, this effects directly the main volume.

LFO

Grobian s single LFO can control the oscillators or the filter. Like the envelopes, LFO assignments are made in the OSCILLATORS or CASCADE FILTER sections, with the +LFOcontrols. When the MW switch is on, then the LFO is activated by the mod wheel. The LFO only has one shape — it s always a square wave, but its pulse width and phase can be varied.

- <MW> Switches the controlling of the LFO output by the modulation wheel on or off.
- <F> Sets LFO frequency (speed).
- <Del> Sets LFO delay — the time it takes before the LFO is active.
- <Width> Sets LFO pulse width.
- <Phs> Sets LFO starting phase.

Pitch Options

This section lets you switch glide (portamento) on, and controls the pitch bend range. When Auto glide mode is activated, the pitch will only glide when notes are played legato. In Manual glide mode, the glide time is controlled by the Man knob.

- <Glide> Switches the pitch glide mode (portamento) on or off.
- <Auto> Switches the auto pitch glide mode on or off. When on, the pitch only glides when the notes are played legato.
- <Source> Sets the controller that adjusts the glide speed, i. e. the time that is used to reach the final pitch. MW is the modulation wheel, Man selects the <Man> knob for controlling.
- <Man> Sets the time used for gliding. See also <Source>.
- <PBnd> Sets the range of the pitch bend wheel in semitones.

Output

The output section gives you control over the saturation, a one-band high-shelf EQ, and master level.

- <Sat> Turns the saturation of the output signal on or off.
- <Drive> Sets the amount of amplification before the siognal is saturated.
- <HiShlf> Sets the cutoff freqeuncy of the high shelf equalizer.
- <Boost> Sets the amount of amplification of the high shelf.
- <Level> Sets the final output amplification level.

8.10 Junatik

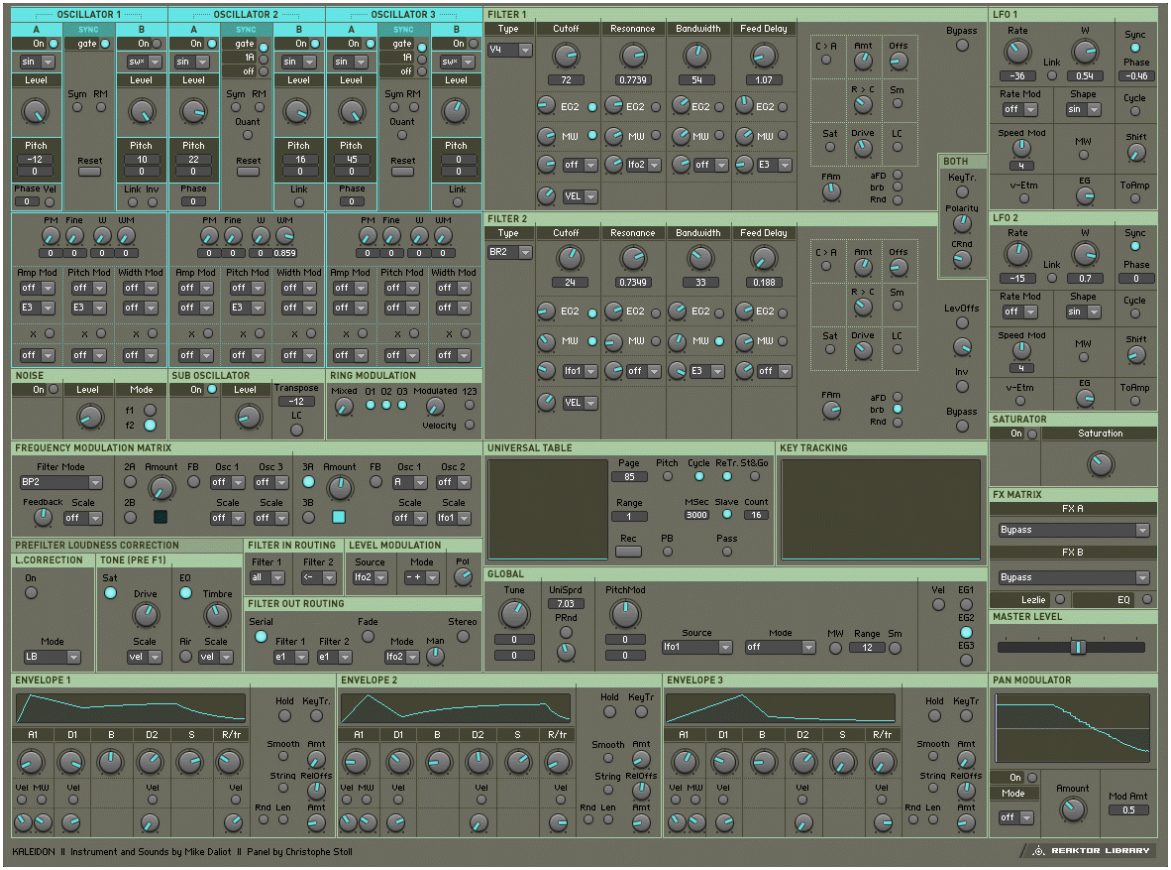


1.21 Junatik user interface

1982, about one year before the era of FM Synthesis began, polyphonic analogous synthesizers finally became (more) affordable. This was mainly due to a japanese company bringing an attractively priced synthesizer to market which offered six voices and was named after a very important roman goddess. Although the first version of that synth had no memory locations, it convinced with its charismatic and full sound. In fact, the synthesizer was not complicated at all and sounds could be set up very easily and quickly, therefore the absence of memory locations (which were rather expensive at that time) wasn't very tragical. The fat sound of the synth did not come from its oscillator/sub oscillator combination, but rather from a good sounding lowpass filter and especially an integrated chorus effect. Unfortunately the chorus was very noisy, but it broadened the somehow sterile sound of the oscillator/sub oscillator significantly and made a big contribution to the popularity of the synth.

Junatik offers a surprisingly authentic recreation of the still very popular synth. The symphatically straight sound architecture of Junatik was modelled on the original and carefully supplemented with some important functions. Besides an authentic sounding filter, Junatik therefore offers an improved oscillator section. With its threefold and detunable saw wave it is now even capable of producing detuned and very fat sounds. An optional velocity sensitivity, which was not available in the original, has also been implemented. Additionally, the sound generation has been supplemented by a 3-band EQ with a semiparametric middle band, a very good sounding distortion unit and a tempo based stereo delay with filter – effects which perfectly complement the strong sound of Junatik. It goes without saying that the unrenounceable chorus effect of the original has also been implemented – yet without the noise.

8.11 Kaleidon



1.22 Kaleidon user interface

8.11.1 Introduction

Kaleidon is the biggest synth in the entire Reaktor Library. Its sounds cover an exceptionally wide range, from realistic mallets, organs, reeds, and saxophones to house chords, techno basses, and otherwordly atmospheres. Its specs are formidable: six oscillators plus noise and sub-osc with sync, FM, and ring modulation; two multimode filters; three feature-rich envelopes; three LFOs (one which lets you draw you own waveforms); graphical key-tracking; and a full effects section featuring many unique effects, from rich reverb and chorus to a graphically sequenced delay line and quad ring-modulator.

8.11.2 Quick Start

Kaleidon is Carbon's big brother, so you may want to take a few moments to get familiar with the Carbon synth as well. Instead of Carbon's four oscillators plus noise, Kaleidon offers six plus sub and noise. Instead of Carbon's single filter-design, Kaleidon features two filters with flexible routing. Instead of Carbon's integrated effects, Kaleidon offers full-featured effects in rack format.

Due to its complexity, Kaleidon takes advantage of both view "A" and view "B" - view "B" is designed to be the day-to-day interface of the synth, with control over the most important synth parameters. View "A" is a tweak-head's delight, as it shows every single parameter directly. A monitor with 1280x1024 resolution is recommended to use the full view "A." The effects are laid out in rack-style underneath the synth. You can maximize or minimize each effect, as needed.



Note that all controls in the reduced "B" view are also duplicated in the full "A" view.

8.11.3 Signal Flow and Structure

Kaleidon's sound begins with three oscillator sections, with each section consisting of two independent oscillators. Each oscillator has a choice of pulse, sawtooth, triangle, sine, bi-saw, parabolic with PWM, and noise. All three oscillator sections are nearly identical, with oscillator sections 2 and 3 giving enhanced sync and waveform options. When the Qnt button is activated in Osc section 2 or 3, additional harmonics are added to the waveform by reducing the bit

depth. Osc sections 2 and 3 also offer a three-saw waveform where three sawtooth waveforms are slightly detuned to create a rich sound. The amplitude, pitch, and width of each oscillator section (acting on both of the oscillators in the section) can be modulated with the pull-down menus at the bottom of each oscillator. An additional noise generator and sub oscillator are also available underneath the oscillators. The A and B oscillators of each oscillator section can be ring modulated with the /RM button.

The oscillators can be flexibly routed to the two filters with the Filter In Routing and Filter Out Routing controls. Here you can set which oscillators go to which filters, and you can set serial or parallel filter routing. The Prefilter loudness correction fine-tunes the amplitude of the oscillators before they hit the filters, to avoid unwanted resonant peaks.

The two filters are nearly identical, with each offering multiple modes, including various flavors of lowpass, bandpass, and highpass. Each filter has a slightly different selection of filter types. Four large knobs control the important parameters of each filter: Cutoff, resonance, bandwidth, and feedback delay. Feedback delay is only applicable to the Grobian and V4 filter types. Underneath each of the large knobs are the modulation possibilities for that knob. The small knob adjusts the amount of modulation, and the switch turns the modulation on or off for its labeled parameter.

After the filters, the sound passes through Kaleidon's innovative Pan modulator. Here, the sound is spread in stereo across the frequency range. For FX Matrix A, you can choose between chorus, the Vierring sequenced ring modulator, (also available separately in the library), and diffusion delays (also available as the Fusion Reflections effect). After FX Matrix A, the sound passes through a saturator, and then through FX Matrix B. For the B effect, you have a choice of a tempo delay, an innovative sequenced "unit" delay, and a reverb. The reverb core is taken from the SpaceMaster reverb, available in the Library.

The Sequenced Unit Delay effect is unique to Kaleidon. It consists of four delays whose delay time can be sequenced graphically. An additional filter rounds out the effect to create dynamically shifting atmospheres. The "Lurker" patch is an excellent demonstration of what the Unit Delay can do.

Three envelopes can control many of Kaleidon's parameters, and their shape can also be modulated by velocity or the mod wheel. The modulation is controlled in the same manner as the filters.

8.11.4 Global Parameters

Kaleidon's global parameters are nearly identical to Carbon's. Parameters effecting the entire synth are set in the Global section, such as Mono/Poly (set with the Mono button), tuning, unison spread, and glide. Hold lets you specify how long each note will last for, no matter how long the actual MIDI note was held down for (value set in mSec underneath the Hold button). Old controls how much pitch randomness is introduced to emulate vintage hardware.

8.12 Lazerbass



4.51 Lazerbass user interface

8.12.1 Introduction

Lazerbass is the newest addition to the Reaktor Library. Lazerbass is a monophonic synthesizer delivering a pure and direct sound without any additional effect units. Its sonic capabilities range from metallic effect sounds to pervasive sub-bass sounds as well as your bread and butter leads, all with a new standard in signal quality. It serves well for science-fiction and old-school effects as well as for a wide field of electronic and digital music styles such as dubstep, where it can deliver the typical deep “wobbling” bass sounds.

Lazerbass uses additive synthesis to create its sounds, which allows creating a wide range of musical timbres. These timbres are composed of multiple harmonic or non-harmonic partials (individual sine waves) of different frequencies and amplitudes. Additive synthesis allows the emulation of sounds by giving control over the frequency and amplitude of each partial and how these change over time. But don't worry too much about the science, you don't need to know the details to make great sounds.

The basic layout of Lazerbass is similar to a subtractive synthesizer. The **first two columns** on the left allow creating the **overall timbre** of the sound. This is done by first choosing a basic oscillator wave form and subsequently sculpting the sound by changing the amplitudes and frequencies of the individual partials through different mathematical algorithms. The signal is then affected by a **filter section** (**middle column**). The **different filter types** can be modulated in real-time by the modulation **section at the bottom** of the panel. The **upper middle section** allows **visual control of the signal** through different analyzers and an oscilloscope, which are crucial to understanding how the different controls affect the synthesizer signal. The **column on the right** contains different sections for **global parameter settings**, such as master volume or the master volume envelope.

Note that most of the individual sections of Lazerbass can be switched on or off using the round buttons on the sections' headers. If switched off, these sections won’t affect the signal, following the simple rule of “if you don’t see it, you won’t hear it.” Each Section also has a default button on the top right. Clicking this button will set the values of the section to its default. Try experimenting with the default settings of the individual sections and turning them off and on to understand how they affect the sound. Also, the included tutorial bank will walk you through learning this powerful synthesizer.

8.12.2 Oscillator

The Oscillator section provides the basic signal of the synthesizer, offering a choice of different basic waveforms. Eventhough these shapes look like your familiar analogue oscillator waveforms like sawtooth and square, they are created digitally and alias free using additive synthesis.

Control	Function
Select Type	Selects the waveform of the digital oscillator. You can choose between saw (full, dual, multi), square (full, dual, multi, pulsewidth) and impulse.
Partials	Sets the number of individual partials used for the creation of the sound. A higher number will result in a higher resolution and a higher sound quality. Lowering it can act like an artificial low pass filter on the shape and also frees up some CPU.
Transpose	Sets the amount of transposition. This is not an overall transposition, only a certain pattern of partials will be transposed. The left side of the value field adjusts integer values, the right side is used for fine-tuning. Double-clicking the left side of the value field restores the value to its default, double clicking the right side will set the value to the closest integer value.
Number	Sets the number of individual oscillators (for multi saw and multi square only).
Fundamental	Adjusts the level of the fundamental frequency. If multisaw or multisquare is used, only the fundamental of one of the parallel oscillators is affected.
Beating	Adjusts the amount of beating which results in a detuned sound. This form beating is not affected by the pitch of the note played.
x < LFO	Turns synchronization of the beating frequency to the LFO on or off.
Width	Adjusts the pulse width of the PW square waveform (PW square only).

8.12.3 Group Attenuation

The group attenuation section allows for the attenuation of defined group of partials, which are divided by a recurring pattern mask. This allows altering the basic timbre of the sound.

Control	Function
Balance	Adjusts the balance between the different groups. At a value of 0.5 the groups will be evenly balanced and the signal will not be affected. At the values 0 or 1 only one group of partials will be audible.
Pattern	Defines the size of the recurring pattern of individual partials.
msk sym	Adjusts the symmetry of the pattern mask, by defining the number of partials in each group.

8.12.4 Oscillator Phase

This section affects the phasing of the individual partials. If this section is switched on, the phase of each individual partial will be automatically reset when a new phrase is played. The phasing of the individual partials can be shifted by a random value, which will result in a more lively sound, which is more suitable for tonal material. Note that under certain circumstances this can add pops to the signal. In this case this section should be switched off.

Control	Function
Random	Adjusts the intensity of the random phasing.
Shape	Changes the random phasing patterns.

8.12.5 Ratio Multiply

This section changes the frequencies of individual partials by multiplication with a set value. The affected partials are defined by a pattern mask. The multiplication results in a transposition of the affected partials, which allows significant changes of the basic timbre. Please refer to the “ratio” section of the visualization area.

Control	Function
Multiply	Sets the multiplication factor for the frequencies of the affected partials.
Pattern	Defines the size of the recurring pattern of affected partials.

8.12.6 Ratio Add

This section changes the frequencies of the individual partials by adding a set value. The affected partials are defined by a pattern mask. The addition results in a frequency shifting of the affected partials, which allows changing the basic timbre. Please refer to the "ratio" section of the visualization area.

Control	Function
Add	Sets the addition value for the frequencies of the affected partials.
Pattern	Defines the size of the recurring pattern of affected partials.

8.12.7 Dispersion

The dispersion section simulates the natural rigidity of materials (e.g. a string) by 'bending up' the higher partials of the signal. The dispersion section has a strong influence on the sound. Please refer to the "ratio" section of the visualization area.

Control	Function
Dispersion	Sets the dispersion value.
Key Track	Sets the amount of key tracking for the dispersion. Higher pitched notes will be less affected by the dispersion if switched on.
Shape	Adjusts the shape of the dispersion.

8.12.8 Partial Beating

The Partial Beating section allows for a different and less obvious form of beating as in the oscillator section and is defined by a pattern mask. This form of beating is not affected by the pitch of the note played. Please refer to the "scope" section of the visualization area.

Control	Function
Beating	Adjusts the beating frequency.
x LFO	Turns synchronization of the beating frequency to the LFO on or off.
Pattern	Defines the size of the recurring pattern of affected partials.

8.12.9 Visualization Area

The visualization area offers different tools for visually controlling and analyzing the synthesizer signal and should be used extensively to understand how the individual sections affect the signal.

Control	Function
Off	Switches the visualization off.
Brightness	Shows the effect of the Brightness section. The horizontal axis displays the frequency.
Periodic Filter	Shows the effect of the Periodic Filter section. The horizontal axis displays the frequency.
Ratio	Shows the effect of the Ratio Multiply, Ratio Add and the Dispersion sections on the individual partials. The horizontal axis displays the individual partials, the vertical axis the applied ratio.
Scope	Shows the overall wave shape of the synthesizer signal.

8.12.10 Brightness

This section acts as a low-pass filter and offers a basic modulation matrix. Please refer to the “Brightness” section of the visualization area.

Control	Function
Brightness	Adjusts the filter frequency. The left side of the value field adjusts integer values, the right side is used for fine-tuning. Double-clicking the left side of the value field restores the value to its default, double clicking the right side will set the value to the closest integer value.
Modulation Matrix	The upper part of the modulation matrix allows switching between the modes off, env and lfo, which allow modulating the filter frequency with the Envelope or LFO section at the bottom of the Lazerbass panel. If switched on, the left value field will adjust the minimum value while the right value field will adjust the maximum value for the filter frequency. The modes on the lower part of the matrix (off, c1, c2) allow manually fading between two different individual filter settings by using the modulation macro controls c1 or c2 (bottom left section of the Lazerbass Panel).

Control	Function
Key Track	Adjusts the intensity of key tracking for the filter. Higher values will result in increased damping of higher notes.
Floor	Sets the floor value for the filter frequency.

8.12.11 Periodic Filter

The periodic filter section applies a periodic filter effect to the synthesizer signal, which allows creating a variety of more advanced effects (such as phaser-, flanger- and resonance effects). All parameters can be modulated with the help of a modulation matrix. Please refer to the "Periodic Filter" section of the visualization area.

Control	Function
Cycles	Adjusts the number of cycles for the periodic filter. The left side of the value field adjusts integer values, the right side is used for fine-tuning. Double-clicking the left side of the value field restores the value to its default, double clicking the right side will set the value to the closest integer value.
Cycles Modulation Matrix	The upper part of the modulation matrix allows switching between the modes off, env and lfo, which allow modulating the number of cycles with the Envelope or LFO section at the bottom of the Lazerbass panel. If switched on, the left value field will adjust the minimum number while the right value field will adjust the maximum number of filter cycles. The modes on the lower part of the matrix (off, c1, c2) allow manually fading between two different individual settings by using the modulation macro controls c1 or c2 (bottom left section of the Lazerbass Panel).

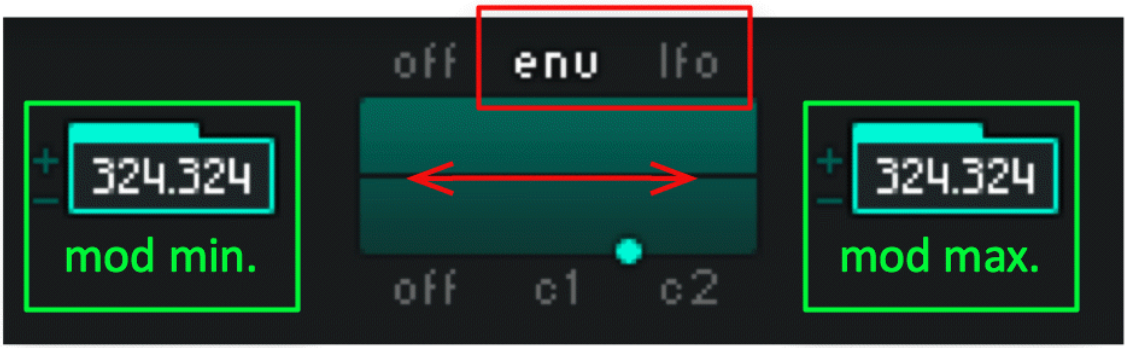
Control	Function
Phase shift	Adjusts the amount of phase shifting for the periodic filter. The left side of the value field adjusts integer values, the right side is used for fine-tuning. Double-clicking the left side of the value field restores the value to its default, double clicking the right side will set the value to the closest integer value.
Phase Shift Modulation Matrix	The upper part of the modulation matrix allows switching between the modes off, env and lfo, which allow modulating the phase shifting with the Envelope or LFO section at the bottom of the Lazerbass panel. If switched on, the left value field will adjust the minimum value while the right value field will adjust the maximum value for the phase shifting. The modes on the lower part of the matrix (off, c1, c2) allow manually fading between two different individual phase shift settings by using the modulation macro controls c1 or c2 (bottom left section of the Lazerbass Panel).

Control	Function
Pinch	Adjusts the amount of pinch (stretching) of the periodic filter. The left side of the value field adjusts coarse values, the right side is used for fine-tuning. Double-clicking the left side of the value field restores the value to its default, double clicking the right side will set the value to the closest integer value.
Pinch Modulation Matrix	The upper part of the modulation matrix allows switching between the modes off, env and lfo, which allow modulating the amount of pinch with the Envelope or LFO section at the bottom of the Lazerbass panel. If switched on, the left value field will adjust the minimum amount while the right value field will adjust the maximum amount of pinch. The modes on the lower part of the matrix (off, c1, c2) allow manually fading between two different individual pinch settings by using the modulation macro controls c1 or c2 (bottom left section of the Lazerbass Panel).

Control	Function
Apply	Sets the overall amount of the periodic filter effect.
Stretch	Switches between two different filter modes. If switched off, the periodic filter works in a linear way (flanger-like sounds), if switched on, the filter works in a logarithmic way (phaser-like sounds).

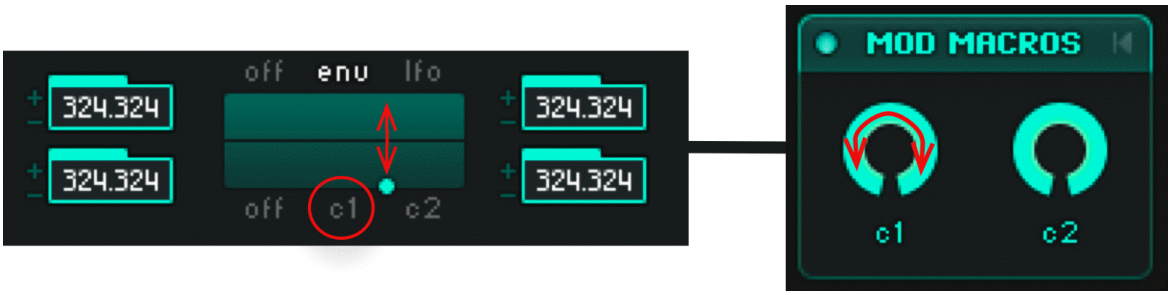
Control	Function
Block	Switches the filter type of the periodic filter to a square wave shape. The result is a more artificial and digital sound.
Peak	Sets the peak value for the individual cycles. High values will result in a more artificial, resonant and bell-like sound.

8.12.12 Using the Modulation Matrix



4.52 Modulation Matrix area

If the upper mode is switched to env or lfo, the sound parameter will be modulated by the Modulation Envelope or Modulation LFO section (bottom of the Lazerbass panel). You can set the minimum and maximum values for the modulation in the value fields.



4.53 Fading between modulation settings

The modes on the lower part of the matrix (c1, c2) allow manually fading between two individual modulation settings (upper and lower value fields) by turning the c1 or c2 knob in the Modulation Macros section. This makes it easy to do manual filter rides with a midi controller.

8.12.13 Modulation Envelope

This section controls the modulation envelope settings for the brightness and periodic filter parameters. The results can be seen in each modulation matrix (if set to env)

Control	Function
Att	Sets the attack time for the modulation envelope.
Rel	Sets the release time for the modulation envelope.
Vel ?Lev	Adjusts the velocity sensitivity for the level of the modulation envelope.
Time	Adjusts the envelope time.
Invert	Inverts the modulation envelope.

8.12.14 Modulation LFO

This section controls the modulation LFO settings for the brightness and periodic filter parameters. The results can be seen in each modulation matrix (if set to LFO).

Control	Function
BPM	Changes the LFO frequency mode. If switched on, it allows for adjusting the LFO frequency relative to the global tempo.
Snap	Locks the LFO frequency to defined beat values.
LFO Frequency	Adjusts the LFO frequency with absolute (Hertz) or relative values (beats).
Times	Selects between three different settings for the LFO frequency values (if BPM is switched on)
dot-trip	Selects between dotted (left), normal (middle) or triplet (right) values.
Restart	Resets the LFO to its initial position every time a note is played.
Shape	Adjusts the shape of the LFO waveform.

8.12.15 Modulation Macro Controls

The macro controls can be used for manually controlling the modulation settings for the brightness and periodic filter sections (by blending between the two different settings in each modulation matrix). These controls should ideally be mapped to a midi controller knob for fast access.

Control	Function
C1	Blends between the different filter settings (the upper and lower part of the modulation matrix) if c1 is chosen.
C2	Blends between the different filter settings (the upper and lower part of each modulation matrix) if c2 is chosen.

8.12.16 Panning/Master

These sections allows adjusting important master parameters for the instrument such as stereo spread, sub-bass and master volume.

Control	Function
Pattern	Defines the scattering pattern of the stereo spread function.
Stereo	Adjusts the amount of stereo spread for the overall signal. This will scatter the individual partials in the stereo field and creates a spatial effect.

Control	Function
Sub	Adjust the amount of the sub-bass.
Master	Adjusts the overall volume.

8.12.17 Master Envelope / Release Mode

This section can be used to individually shape the overall sound and the volume envelope for the “Note On” and “Note Off” phase of the note played. This allows interesting release effects (e.g naturally occurring string release noises). This section is subdivided into the “Note On” settings (left side) and the “Note Off” settings (right side).

Section	Control	Function
Note On	Dec	Sets the decay time for the ‘Note On’ phase.
	Vel ?Lev	Adjusts the velocity sensitivity for the level of the ‘Note On’ phase.
	Spec.B	Switches on a different timbre for the ‘Note On’ phase as selected in the Spectrum B field.
Note Off	Rel	Sets the release time for the ‘Note Off’ phase.
	Dec=Rel	Sets the release time to the same amount as the decay time. (If Rel.Env is deactivated)
	Rel.Env	Activates a individual volume envelope for the ‘Note Off’ phase.
	Att	Sets the attack time for the ‘Note Off’ phase.
	Lev	Adjusts the level of the ‘Note Off’ envelope.
	Brightness	Adjusts the Brightness filter for the ‘Note Off’ phase. The left side of the value field adjusts integer values, the right side is used for fine-tuning. Double-clicking the left side of the value field restores the value to its default, double clicking the right side will set the value to the closest integer value.
	Spec.B	Switches on a different timbre for the ‘Note Off’ phase as selected in the Spectrum B field.
	Select Spectrum B	Selects a different overall timbre that can be individually applied to the ‘Note On’ or ‘Note Off’ phases. The different settings globally affect the amplitude and ratio parameters of the synthesizer timbre (but are invisible to the user).

8.12.18 Global Pitch

This section allows adjusting the global transposition and pitch bend range for the instrument.

Control	Function
Transpose	Sets the global transposition value. The left side of the value field adjusts integer values, the right side is used for fine-tuning. Double-clicking the left side of the value field restores the value to its default, double clicking the right side will set the value to the closest integer value.
Pitch Bend	Sets the global pitch bend range. The left side of the value field adjusts integer values, the right side is used for fine-tuning. Double-clicking the left side of the value field restores the value to its default, double clicking the right side will set the value to the closest integer value.

8.12.19 Gate Mode

This section is used for choosing the global gate mode of the instrument when several individual keys are played on the keyboard at the same time. You can choose between:

Control	Function
Normal	Oscillator phase and modulation (envelopes and LFO) will be reset with every keystroke. The 'Note Off' Phase starts after releasing the last key played.
Hold	Oscillator phase and modulation (envelopes and LFO) will be reset with every keystroke. The 'Note Off' phase starts after all keys are released.
Legato	Oscillator Phase and modulation (envelopes and LFO) will not be reset. The 'Note Off' phase starts after all keys are released.

8.12.20 Glide

This section allows for adjusting the global pitch glide parameters of the instrument.

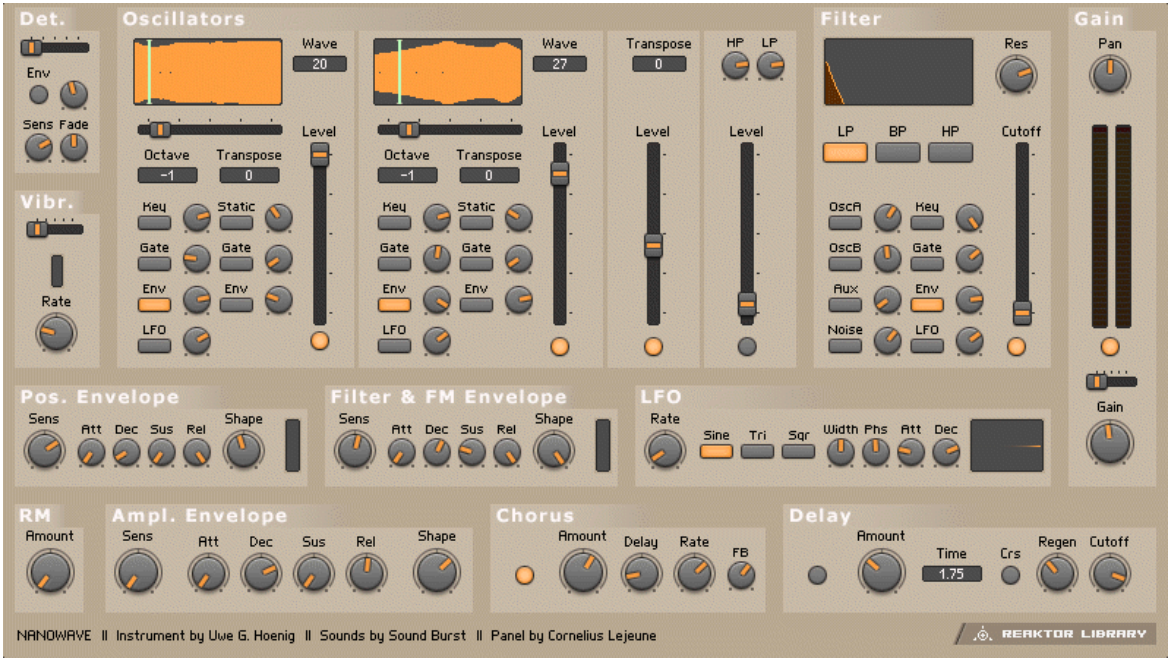
Control	Function
Legato	Activates the legato mode. If switched on the glide effect will only be used if more than one key is played.
Glide Time	Adjusts the glide time between two notes.
abs/rel	Chooses between an absolut glide time or a relative glide time value (glide time is relative to the pitch distance).

8.12.21 Pitch Mod.

This section allows for adjusting the global pitch modulation parameters of the instrument.

Control	Function
Mode	Selects the modulation source. You can choose between EG (envelope) and LFO. The modulation settings can be adjusted in the modulation area at the bottom of the Laserbass instrument panel.
Range	Sets the pitch modulation range. The left side of the value field adjusts integer values, the right side is used for fine-tuning. Double-clicking the left side of the value field restores the value to its default, double clicking the right side will set the value to the closest integer value.
PM amnt	Adjusts the amount of pitch modulation.

8.13 Nanowave



1.23 Nanowave user interface

Nanowave is an homage to the legendary WaveTable synthesizers made by PPG and Waldorf. The sound generation – called "WaveSet" synthesis here – is structured similarly to the classic subtractive synthesis used in analog synthesizers, but it has some special features on the oscillator level which result in a significantly wider variety of sounds.

The WaveSet oscillators of Nanowave are not based on the common waveforms like sawtooth or square. They use so-called WaveSets which contain many different waveforms. 43 of these WaveSets are included with Nanowave. Within a WaveSet, a particular waveform can be chosen manually but the really crucial feature in WaveSet Synthesis is the option to "travel" in the WaveSet, i.e. switching dynamically between the waveforms of a WaveSet.

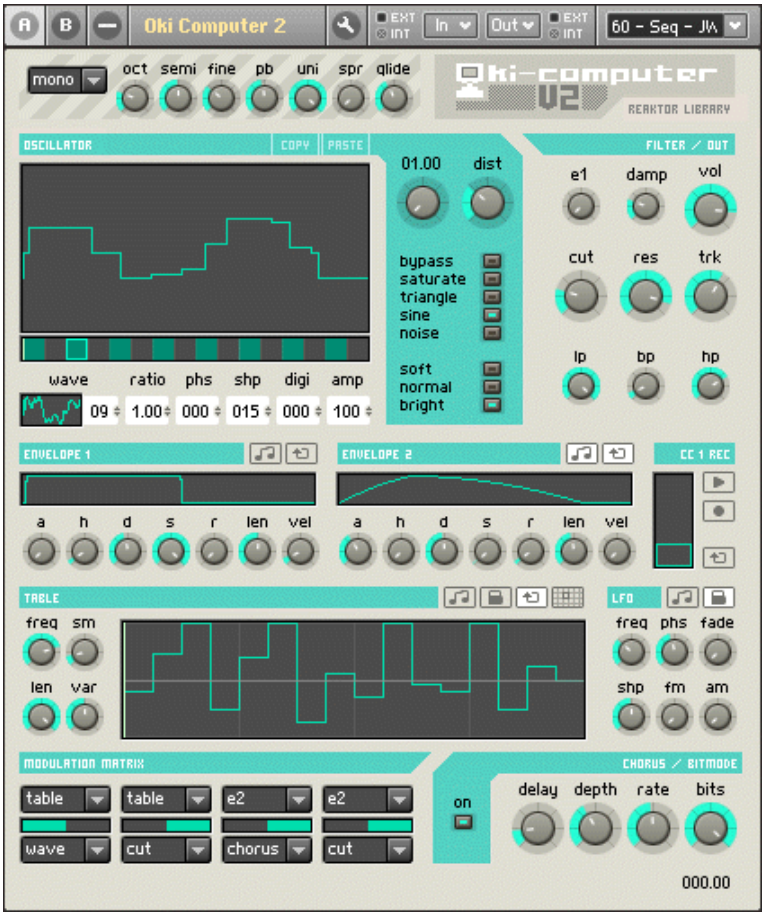
Both of the Nanowave oscillators have an identical structure: Each gives you the option to select the WaveSet, the octave and the pitch tuning in semitones. With the slide switch you can manually move through the WaveSet, or alternatively the waveform position can be modulated by a separate wave envelope, by attack, keytracking or by the LFO. Additionally frequency modulation is possible. A couple of other sound sources are also available: a simple sine oscillator, a noise generator with selectable tone colour and the ring-modulation signal produced from waves 1 and 2.

Nanowave's multi-mode filter works with a slope of 12 dB/oct and can be modulated in various ways.

The LFO starts with note onset, and consequently allows envelope-like modulations. It also has a separate envelope for the LFO amplitude.

The three envelopes in Nanowave can be modulated by the keyboard velocity and have a special parameter which enhances the exponential characteristic of this process – very good for extremely percussive sounds.

8.14 Oki Computer 2



4.54 Oki-Computer 2 user interface

8.14.1 Introduction

If you get excited by words such as analog and vintage, look away now. Oki Computer 2 is a compact wavetable synthesizer, a specialist in digital lo-fi sounds that hails back to the era of 8-bit beeps and bleeps... It is also rather capable at creating buzzing leads, rhythmic sequences, and odd tasty bass tones.

Oki Computer 2's panel is compact but packed with features. Thankfully, most sections should be fairly straightforward to the average synth user. However, the [Oscillator] section is somewhat unique and users are strongly encouraged to read this part of the manual. Oki Computer 2 features a bank of 50 waves. For every patch you can load any 16 of these waves into the oscillator in any order. This flexibility represents a major improvement over the original ensemble (where the oscillator was permanently 'hardwired' to the same 16 waves). What's more, each wave loaded into the oscillator can be processed in a variety of ways.

What's New in 2.1?

This update version features an improved oscillator section. Internally, the anti-aliasing of the algorithm has been improved; on the panel this becomes visible in the new [Oscillator Tone] control below the [Distortion Mode] selector. It provides three settings: [bright] represents the old mode while [normal] and [soft] remove the sound's harshness as far as it is caused by aliasing artefacts. Due to this improvement the [Copy] and [Paste] buttons were moved and can now be found above the waveform display. The exiting snapshots have been updated to use this new feature; some have even been added. Snapshots saved with Oki Computer 2.0 are compatible to this version, however they will use random settings for the new control unless they are saved with Oki Computer 2.1.

8.14.2 MIDI In

The drop-down list at the panel's top-left switches between polyphonic and monophonic operation modes. In polyphonic mode, Oki Computer operates as a standard poly-synth. Monophonic mode does not restrict the number of voices to 1; it enables some very musical features – legato, glide, and unison.

Control	Function
Gate Mode	Selects whether the instrument is used as polyphonic or monophonic synthesizer.
Unison	Determines the number of simultaneous voices. This is only active if [Gate Mode] is set to [mono].
Spread	Defines the amount of voice detuning in semitones. This is only active if [Gate Mode] is set to [mono].
Glide	Sets the amount of portamento, i.e. the time used to reach a new MIDI pitch. This is only active if [Gate Mode] is set to [mono].
Octave	Transposes the pitch of the entire oscillator in octave steps.
Semitone	Transposes the pitch of the entire oscillator in semitone steps.
Fine	Fine-tunes the entire oscillator's pitch.
Pitchbend	Defines the MIDI pitch bend wheel range in semitones.

8.14.3 Oscillator

The [Wavetable Position Bar], located beneath the main oscillator window, is perhaps the most difficult part of the synthesizer to understand. This bar has two purposes. Firstly, the square box indicates the current wave slot selected for editing (there are 16 slots). Secondly, the bright green line indicates the current wavetable position. The current wavetable position is set by the [Wavetable Position Knob] (to the left of the drive knob), plus any modulation routed to the wave table position (see [Modulation Matrix]).

The best way to explain how the [Wavetable Position Bar] works is by example: Click on the snapshot menu and recall preset number 1 - 'Default'. In this preset, the oscillator is loaded with 16 sine waves (needless to say, this doesn't sound particularly interesting). Click the left-most box on the Wavetable Position Bar - this will select the first slot for editing. Note that in the box labeled [Wave] (beneath the [Wavetable Position Bar]) a picture of a sine wave is displayed, with a number zero next to it. This indicates that the sine wave (wave number zero from the master bank) is loaded into the current slot. To load a different wave, click and drag vertically on the Wave Selector. Now click on the second wave slot (i.e. the adjacent dark gray box). The Wave Selector will display a sine wave (remember this patch had 16 identical sine waves loaded). Now try loading a different wave to slot 2, again by clicking and dragging on the Wave Selector.

In the 'Default' snapshot, the [Wavetable Position Knob] is set to 1.00. This means that when you play a note, you will hear (and see) the wave loaded into slot 1. Press a note on your keyboard, and slowly move the knob from 1.00 to 2.00. You will hear and see the wave loaded into slot 1 morph into the wave loaded into slot 2. Notice how the wave position indicator moves correspondingly. This is the key to how Oki Computer 2 produces dynamic sounds: by morphing between adjacent waves in the wavetable. While this can be done manually with the wave position knob, things get much more interesting when the various modulators (e.g. envelope, sequencer, LFO) are used to mix between waves.

Apart from the Wave Selector, all the controls underneath the wavetable position bar are used to modify wave shape. When using these controls, it is important to remember that they only affect the wave in the selected slot (i.e. the green box), which is not necessarily the wave shape currently being played (i.e. the green line).

Control	Function
Ratio	Sets the number of times the wave shape will repeat in a single oscillator cycle. Note that the integer and decimal values can be set independently, also note that adjusting the ratio will cause pitch shifting.
Phase	Rotates the wave start position within the oscillator cycle.
Shape	Skews the wave shape to either the left or right (on the pulse wave this is identical to a pulse width control).
Digitize	Reduces the wave's bit depth.
Amp	Attenuates the wave volume.
Copy	Stores the current settings into an edit buffer that can be read out again by the [Paste] button.
Paste	Recalls the data from the edit buffer (see [Copy]).
Distortion Amount	Controls the amount of distortion. (See also [Distortion Mode].)
Distortion Mode	Selects the way the signal is distorted. [Saturate] applies a 'standard' saturation curve to the signal. [Triangle] and [Sine] both involve wrapping their respective shapes around the input signal. When used on a sine wave, these two functions can sound somewhat reminiscent of FM. [Noise] enables a noise generator.

8.14.4 Filter / Out

This section controls the shaping applied to the sound's frequency spectrum (filter) and amplitude.

Control	Function
Amplitude Envelope Mode	Selects the main output envelope. In most cases, [E1] will be the preferred choice. Sometimes however, you may want to use Envelope 1 for modulation purposes only. In this case, select either [G] (MIDI gate, ignoring velocity) or [Vel] (MIDI gate, including velocity).
Damp	Controls the amount of high-frequency damping.
Volume	Sets the main output volume in decibels.
Cut-off	Sets the filter's cut-off frequency.
Resonance	Adjusts the filter's resonance amount.
Track	Defines the amount of cut-off pitch tracking. At 100%, cut-off is increased by one semitone for each increment in MIDI pitch. At -100%, cut-off is decreased by one semitone for each increment in MIDI pitch. At +/- 200%, cut-off changes by two semitones for every one semitone change in pitch.
Low-pass, Band-pass, High-pass	Determines the mix ratio of the high-pass, band-pass and low-pass components of the filter output signal.

8.14.5 Envelope, CC1, Sequencer and LFO

Oki Computer 2 features two envelope generators. Both can be used for general modulation via the Modulation Matrix, but envelope 1 can also be routed directly to output volume in the [Filter / Out] section. Otherwise, the envelope generators are identical.

The CC1 section allows you to record modulation wheel movements. To record a sequence, click the [Record] button. The button will flash, indicating that it is armed and waiting. Recording will begin when a MIDI note is pressed, and finish when the note is released (or when all recording memory is used up). You can record movements with the mouse, or the CC1 knob on a MIDI controller. As long as the [Play] button is depressed, recordings will play back each

time a note is triggered. The recording is sent to MIDI CC1. Therefore, to use the recording as a modulation source, select CC1 in the Modulation Matrix. Note that even though sequences are recorded monophonically, playback operates in full polyphony.

The sequencer is a highly flexible modulation source. It can operate as an arpegiator, a custom shape LFO or an additional envelope. You can draw its steps with the mouse. You'll find a standard LFO next to the sequencer.

Section	Control	Function
Envelope 1/2	Attack	Controls the attack time of the envelope generator.
	Hold	Controls the hold time of the envelope generator.
	Decay	Controls the decay time of the envelope generator.
	Sustain	Controls the sustain level of the envelope generator.
	Release	Controls the release time of the envelope generator.
	Speed	Multiplies the overall envelope time.
	Velocity	Determines the extent to which envelope amplitude is linked to velocity.
	Clock Sync	Synchronizes the envelope times to the global MIDI tempo.
	Loop	Activating this button results in the attack, hold and decay stages looping when MIDI notes are depressed.
CC1	Record	Arms the CC1 recorder.
	Play	Enables playback of the recorded movements, triggered like an envelope by MIDI gate signals.
	Loop	Loops the recorded movement on playback.
Sequencer	Clock Sync	Synchronize the sequencer to the MIDI clock. Note that when both [Clock Sync] and [Phase Lock] are activated, the sequencer becomes locked to the MIDI song position.
	Phase Lock	Locks the phase of the sequencer. When this is active, MIDI note events will not re-trigger the sequencer. Note that when both [Clock Sync] and [Phase Lock] are activated, the sequencer becomes locked to the MIDI song position.
	Loop	When enabled, the sequencer will loop indefinitely, otherwise it will play back only once when triggered.

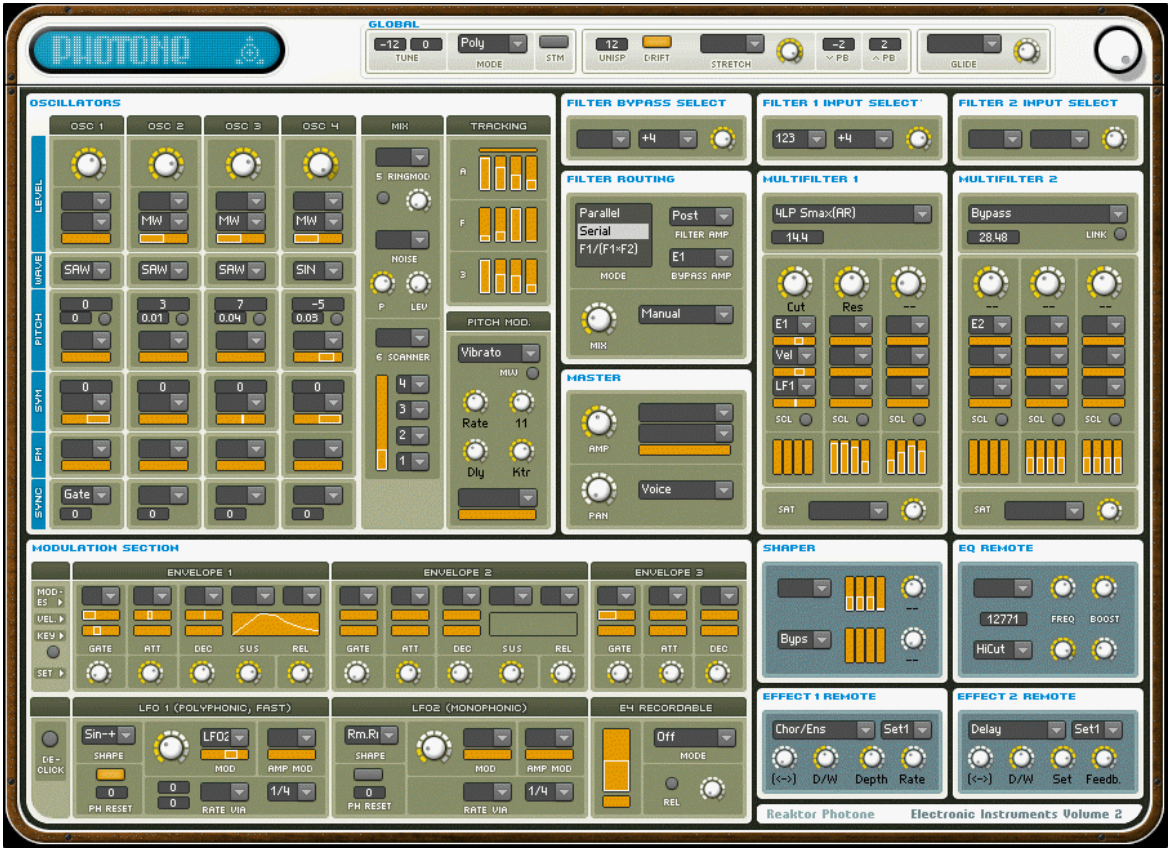
Section	Control	Function
	Snap	Activates a vertical grid with a step size of 1/12 of the complete height.
	Frequency	Determines the sequencer speed.
	Length	Sets the sequencer length in steps.
	Smooth	Determines the amount of interpolation between adjacent steps (at hard-right, the sequencer produces smooth envelope-type output).
	Variation	Imparts a kind of swing onto the sequencer movement, so steps play alternately faster and slower. Center this control for equal length steps.
LFO	Clock Sync	Synchronize the LFO to the MIDI clock. Note that when both [Clock Sync] and [Phase Lock] are activated, the LFO becomes locked to the MIDI song position.
	Phase Lock	Locks the phase of the LFO. When this is active, MIDI note events will not re-trigger the LFO. Note that when both [Clock Sync] and [Phase Lock] are activated, the LFO becomes locked to the MIDI song position.
	Frequency	Sets the LFO speed.
	Phase	Determines the point in the LFO wave where oscillation begins when a note is triggered. This only function when [Phase Lock] is off.
	Fade	Sets the fade-in time of the LFO (i.e. the time taken to reach full amplitude).
	Shape	Skews the LFO shape to either the left or right.
	FM / AM	Determine the amount which the modulation wheel (including recorded movements) modulatea frequency and amplitude respectively.

8.14.6 Modulation Matrix

The modulation matrix enables any four modulation sources to be routed to any four destinations. You can select modulation sources using the upper drop-down menus. Destinations are selected with the lower menus. The sliders in-between these menu set the modulation amount. The full list of modulation sources and destinations is summarized in the following table:

Section	Control	Function	Range
Sources	Vel	MIDI note on velocity	(0 to 1)
	PB	MIDI pitchbend wheel	(-1 to 1)
	CC1	MIDI CC1 - the modulation wheel. Note that recorded CC1 movements (in the recordable envelope section) are routed to this parameter.	(0 to 1)
	E2	Envelope generator 2	(0 to 1)
	Seq	The sequencer	(-1 to 1)
	LFO	The LFO	(-1 to 1)
Destinations	Amp	Output volume	(-100% to +100%)
	Pitch	Oscillator pitch	(-12 to +12 semitones)
	Wave	Oscillator wave position	(-16 to +16)
	Cutoff	Filter cutoff	(-60 to +60 semitones)
	Chorus	Chorus frequency	(-100% to 100%)

8.15 Photone



3.7 Photone user interface

Introduction

In essence, Photone employs a relatively straightforward ‘subtractive’ architecture, featuring a bank of oscillators, two filters, and an array of effect processors. However, several features set Photone apart. In addition to the unprecedented level of sound quality, Photone boasts an ar-

ray of special features, including custom oscillator types, flexible control over filter routing, vintage analog drift and detuning emulation, and advanced modulation options. In short, Photone redefines the standard for REAKTOR synthesis.

Quick Start

Photone is a complex and comprehensive synthesis workstation. While it is strongly recommended that you read this manual in its entirety, those experienced enough to begin programming Photone straight away should at least read this Quick Start section. You are also advised to read all control hints, available on the panel.

Section	Function
Filter Input Selection	To summarise, Photone's signal flow is *oscillators > filter > effects > output*. However, unlike many synths with a similar architecture, the oscillators are not hardwired directly into the filter inputs. Instead, using the three Input Select sections, the oscillators can be individually routed into the filters (or the filter bypass pathway). Thus, only oscillators that are selected in these 3 sections will be audible. Note also that the Filter Bypass Select section is only relevant if an amp source for the bypass signal is selected in the Filter Routing box. Further, the Filter 2 Input Select configuration is only relevant in Parallel routing mode. Envelope 1 is always applied to the amplitude of the filtered signal, and thus determines the main contour of the sound.
Modulation Menus	It is possible to assign a range of modulation sources to many of Photone's parameters. In each section, Photone has a uniform system for assigning modulation sources – the source is selected from a drop down menu, and the amount of modulation is set by clicking and dragging on the yellow bar beneath. In most cases this bar is bipolar, so that centre position yields no modulation. Available Modulation sources include external MIDI ones (e.g. velocity, pitchbend), and internal ones (4 envelopes and 2 LFOs). Internal sources are configured in the Modulation Section.
Tracking Devices	It is also possible to specify certain parameters differentially across the notes on your keyboard. The tracking devices are used for this purpose. The four bars pertain to MIDI notes 36, 56; 76 and 96. Interpolation occurs in-between these 4 breakpoints. While some tracking devices are permanently hardwired to certain parameters (in the shaper section for example), the tracking devices in the Tracking section are freely assignable to various parameters using the drop down menus.

Overview

Photone consists of an oscillator section (in the upper left), a filter bank (upper right), and an array of effect processors (lower right). Many of the parameters can be animated over time by MIDI sources, as well as the internal sources available in the Modulation section (lower left).

Global Section

Section	Control	Function
Tune and Pitchbend	Tune	**Tune** offers coarse and fine transposition. The **PB** controls define the effect of the pitchbend wheel on note pitch - the range for upward and downward movements are specified independently.
Gate modes		**Mode** determines whether Photone operates polyphonically or monophonically, and whether envelope attack reset (AR) is enabled. In the attack-reset modes, the envelopes are always reset to zero on each gate on event to yield the full attack range (i.e. from zero to peak). This is in contrast to standard behaviour where the attack-range is from the envelopes current output level to peak. The difference between attack-reset and standard modes is especially noticeable with long envelope release times. Five monophonic modes are available: Mono (standard 1 voice monophonic); Mono AR (as above plus attack reset); Mono AR 2V (as above but with 2 unison voices); Mono AR 3V (3 unison voices) and Mono AR 4V (4 unison voices). In these 3 unison modes, **Unisp** determines the amount of voice detuning (or spread). In mono modes, **STM** selects single-trigger, legato gate mode. **Drift** simulates analog equipment by applying a small amount of random drift to the note pitch signal, and is particularly effective in the unison monophonic modes. **Stretch** mode simulates analog equipment in a second way by introducing slight inaccuracies to the note scaling. This results in octaves which are not exactly twice the frequency of each other. The amount knob determines the amount and direction of the scaling error – to the left results in slight under-scaling (octaves are less than double-frequency), whereas to the right results in slight over-tuning (octaves are more than double-frequency). In mode 1, the inaccuracies are only introduced above middle C / note 60, whereas in mode 2, error will be present both above and below middle C (resulting in middle C being the only correctly tuned note on the scale).
Glide		When a monophonic gate is selected, two **Glide** modes are available: Standard, and Auto. In auto mode, note pitch will only glide when notes are played legato style. The knob next to the menu sets the glide time.

Oscillators

Photone's sound generation engine features 3 'standard' oscillators (Osc1 – Osc3), and an additional oscillator with several custom waveforms (Osc 4). Oscillators 1 to 4 also all share a digital wavetable mode, with 37 high quality wavesets. The oscillator section also features two slots that provide particular combinations of the four generators, via ring-modulation (slot 5) and a scanner (slot 6).

Control	Function
Osc 1 – Osc 3	Oscilators 1 to 3 have identical controls. From top to bottom, each oscillator section contains controls for Level, Wave (waveform selection), Pitch, Sym (waveform symmetry), FM (frequency modulation) and Sync (hard synchronization). As denoted in brackets in the Wave menu, Sym, FM and Sync are only relevant to certain wave types.
Osc 4	Oscillator 4 is distinguished from Osc's 1-3 by virtue of its 'special' wave types, each of which features a special modulatable parameter. Both the Sym and FM sections are used to route modulation sources to the special parameter. Note that FM is not possible for any wave types in Osc 4. See Appendix 4 for more details.
Osc 1–4 Level	The Level section offers a main amplitude knob, two menus offering a choice of amplitude modulation sources, and a slider bar to determine the amount of amplitude modulation. The upper menu offers two modulation sources - Tracker A or Tracker A inverted. Tracker A is a key position-tracking device located in the Tracking section (see below). The lower modulation source menu allows the selection of other various internal modulators, such as envelopes (see the section entitled 'Modulation'), and external MIDI sources (Vel, CC1, Pitchbend and aftertouch).

Control	Function
Osc 1–4 Wave	A wide range of wave types are available, SAW (Sawtooth); PLS (Pulse); TRI (Triangle); PAR (Parabolic); SIN (Sine); IMP (Impulse); NOI (White noise); DIGI (Digital wavetable, see below). Sym, FM and/or Sync denote that those sections are available for the given waveshape. Oscillator 4 has an extended selection of custom wave types, as detailed in Appendix 1.
DIGI oscillator mode	Oscillators 1 to 4 all have a DIGI oscillator type available in the Wave menu. The DIGI oscillators consist of 37 carefully sampled wavesets. (Each of the 37 waves is multisampled across the keyboard range to avoid anti-aliasing artefacts). In DIGI mode, the Sym section acts as waveset selector, (note however that modulation of wave selection is not possible). Note also that in DIGI mode, FM is not available, and Gate sync is always on regardless of the Sync menu setting.
Osc 1-4 Pitch	The main value control defines the oscillator pitch in semitones. The smaller control beneath is for fine-tuning the oscillator pitch (+/- 1 semitone). However, it also provides a second, 'hidden' feature - if fine tune is set to 1, MIDI note tracking will be disabled for the oscillator. The small button next to fine tune toggles between exponential, semitone modulation (off) or linear, Hz modulation (on). Exponential modulation results in positive input having a greater effect than negative input. In linear mode, positive and negative variation are the same.
Osc 1-4 Sym	The shape of certain oscillator types (those labelled Sym) can be adjusted using this control. This control selects tge waveset in DIGI wavetable mode. For the special oscillators in oscillator 4, the effects of this section vary – see Appendix 1.

Control	Function
Osc 1-3 FM	The frequency of oscillators 1-3 can be modulated by the output of the other oscillators, or the output from filter 1 or 2. Select a source from the menu and set the amount of modulation using the slider. Note that FM is not available for oscillator 4.
Osc 1-4 Sync	Here you can select a source to restart the oscillator phase. Available sources are MIDI gate (i.e. note on events) or the other oscillators. Every time the source passes above zero the oscillator phase will be restarted.
5 - Ringmod / Noise	Section 5 is not actually an oscillator; instead, it combines the output of a choice of oscillators 1 – 4 by ring modulation. (That is, the outputs of the selected oscillators are multiplied by each other). Section 5 also features a noise generator, the output of which is added to the ring-modulated signal. As the Narrow, Colored and Sizzle noise types are all sample based, pitch transposition is available with the **P** knob. Pitch transposition is not available for the pure White noise mode.
6 - Scanner	Section 6 combines the output of the oscillators 1 to 4 by cross-fading between them. The cross-fade position is determined by a combination of one of the several modulation sources and the slider bar.

Control	Function
Tracking	The tracking section provides 3 key tracker modulation sources. For every MIDI note key press, the tracker computes a value depending on the MIDI note number and the positions of the 4 sliders. The leftmost slider corresponds to MIDI note 36; the second slider corresponds to MIDI note 56; the third slider to note 76; and the fourth to note 96. Other MIDI notes falling between these 4 points result in an interpolated output value. Notes falling outside the 36 to 96 MIDI note range are clipped so that they produce the value of the first, or fourth, slider respectively. The first of the 3 key trackers (labelled A) in this section can be routed to oscillator amplitude (in Osc 1 – 4 level sections). This can be used, for example, to dampen the amplitude at high pitches, or to fade between two oscillators depending on the notes played (select TrkA for one oscillator and TrkA Inv for the other). The second tracker (labelled FM) modulates the amount of FM for Osc 1 to Osc 4. The third tracker (labelled 3) is for general purpose key scaling, and can be routed to a variety of parameters.
Pitch Mod	Below the tracking controls lies the global (i.e. all oscillators) pitch modulation section. Selecting vibrato mode menu enables an LFO to modulate pitch. Ramp enables a ramp type envelope to either increase or decrease the oscillators pitch over time. At the bottom of this section there is an additional menu to select another source for global pitch modulation.

Filter Section

The outputs of the 6 oscillators are routed to the filter bank, which features 3 internal pathways – Filter 1, Filter 2 and Filter bypass (the signal direct from the oscillators). The output of Filter 1 and 2 are then combined in according to the filter mix configuration, and then added to the filter bypass pathway.

Section / Control	Function
Input Select	Photone provides complete control over which oscillators are routed through which pathway (Filter 1, 2 or bypass). This is configured using the menus in the 3 Input Select boxes at the top of the filter section. For convenience, filter 2 has an additional option in its second input menu – As F1. When this is selected, Filter 2 receives the same input as Filter 1. (Note that the Filter 2 input menu is only relevant in Parallel mode - see below).
Filter Routing	This section configures the relationship between filter 1 and 2, and control over the amplitude of the bypass signal. If Parallel is selected, then the 2 filters operate with independent inputs (configured by the input select boxes). If Serial is selected, then the Filter 2 Input Select configuration is ignored – and instead the output of filter 1 is fed into the input of filter 2. In both Parallel or Serial routing modes, the output of the 2 filters is crossfaded according to the **Mix** control. If 'Man' is selected, then the adjacent knob defines the crossfade position. Selecting any other entry overrides the manual knob and instead uses the selected source to define the crossfade position. **Filter Amp** determines at which point the amplitude envelope (Envelope 1) is applied to the signal – pre or post the filter section. **Bypass Amp** selects a source for the bypass signal amplitude envelope.
Multifilter 1 and 2	The two filter sections are identical, with the exception that Multifilter 2 features a **Link** button. When this is enabled, the cutoff of filter 2 is relative to (i.e. it is added to) filter 1. The effect of the 3 main knobs in each unit is dependent on the filter mode (selected from the main menu), as denoted in brackets after each menu entry. (In most cases, the left knob controls cutoff and the central knob controls resonance).

Section / Control	Function
	Beneath each knob are slots for selecting up to 3 modulation sources. Further, key scaling can be applied to the each control – enable the Scl button and adjust the 4 key-scaling sliders as required (as in the oscillator tracking section). Both filters also have an optional saturator distortion unit, which can either be placed pre or post filter.

Effects

Photone is equipped with an impressive range of effect processors. The Shaper section differs from the Effect 1/2 and EQ sections in that its placement in the signal chain is flexible. In contrast, Effect 1 and Effect 2 are always applied post-filter, and the EQ section is always applied after that, at the end of the signal chain.

Section / Control	Function
Shaper	The Shaper can be distinguished from the other effect units by virtue of its flexible placement feature, which allows it be activated at one of several points inside the internal signal flow. The placement position can be selected using the upper menu. The lower menu then selects the Shaper mode. The Sine shaper distorts the sound by passing it through a sine function; the upper key tracking block controls the scaling of the input signal (i.e. the input drive – the amount of distortion applied to the signal), the lower one shifts the phase of the sine function used for mapping. The adjacent knobs scale all entries simultaneously. Drive enables a saturator distortion algorithm (similar to the filter Sat sections), for which the upper tracker/knob controls the amount of drive. (The lower block has no effect). **SR** and **BR** offer Sample Rate reduction and Bit Rate reduction respectively, adjustable by the upper tracker/knob. **LP1**, **HP1** and **BP1** offer 6db low-pass, high-pass and band-pass filters. The upper tracker and knob control cutoff. Finally, **AE1** and **AE2** are two audio enhancer algorithms. The amount of enhancement is adjustable by the upper tracker/knob.
Effect 1 and 2	Effect 1 and 2 Effect 1 offers four different effects. Chorus uses a bank of modulated-delay units to add width, depth and richness to the input signal; Rotary simulates a Leslie cabinet (for electronic organ sounds); Drive overdrives the signal resulting in a louder/harsher sound; and Resochord imprints harmonic structures onto the input signal by using 6 comb filters. Each effect has 4 internal modes, or configurations, as selected by the preset menu (Set 1 to Set 4). The user can then adjust 3 further parameters (the function of which depends on the effect type). The output of Effect 1 is then passed to the output of Effect 2, which offers a further 3 effect types – Delay (tempo-sync feedback delay unit); Reverb (room simulator); and Limiter (a volume-maximising peak-clipper).

Section / Control	Function
	See Appendix 2 for a detailed effects reference.
EQ	The EQ section provides the final stage in the Photone signal chain, and offers 2 multi-mode equalisers. The **Freq** knobs determine the centre frequency (in Hz), whereas **Boost** sets the amount of amplification or attenuation (in decibels). The first of the two equalisers offers two modes - Low Shelf (which boosts or attenuates frequencies around and below the centre frequency) and Peak (which boosts/attenuates frequencies around the centre frequency). The second equaliser offers 3 modes – High Shelf (which boosts or attenuates frequencies around and above the centre frequency), Peak (as above), and HiCut (which attenuates frequencies above the centre frequency by 6db per octave). (The boost knob has no effect in HiCut mode.)

Master Section

Control	Function
Level	The **Level** knob adjusts the overall post FX volume, and the upper menu selects a key-tracking source for the overall level. Amplitude can be modulated by selecting an additional source from the lower menu. The corresponding slider adjusts the modulation depth.
Pan	**Pan** defines the position of the output signal in the stereo field. The position can also be modulated by selecting a source from the adjacent menu.

Modulation

The modulation section offers 4 envelope generators and 2 low frequency oscillators. These sources provide time-varying signals that can modulate the parameters in the other sections.

A complete summary of available modulators (including MIDI ones) is detailed in Appendix 3.

Section/Control	Function
Envelope 1, 2, and 3	Envelope 1 and 2 are identical internally, and can be routed flexibly to a variety of parameters (using the modulation menus in the other sections). However, note that Envelope 1 is also permanently hard-wired for global amplitude modulation i.e. it determines the contour of the final output signal (see the Filter Routing section above). **Att**, **Dec**, and **Rel** define the attack, decay and release times respectively. **Sus** sets the sustain level of the envelope. **Gate** determines the hold time (the period of time that the envelope remains at the peak level, between the attack and decay stages). Above the main knobs lies a matrix with which the envelopes relationship with note velocity and key position can be defined. In the Gate column, setting the **Vel** slider to hard right will result in velocity determining the peak value of the envelope. Setting the **Key** slider to the right will result in higher keys producing higher peak values, (whereas setting it to the left will result in lower keys producing higher peak values). Similarly, the attack and decay times can be influenced by velocity and key position either positively (to the right) or negatively (to the left). The Modes menus offer further configuration of the envelopes. This includes adjusting the envelope shape (Parabolic and Sinoid shapes in the Gate menu), and modulation of attack / decay times and sustain level via MIDI sources (e.g. Modulation wheel and Pitchbend wheel). Another interesting feature is the ability to loop the attack-decay part of the envelope if the sustain level is 0 (in the Att menu). Envelope 3 is similar to envelopes 1 and 2, but without hold, sustain and release phases.
Envelope 4	Envelope 4 is a special 'recordable' envelope. Mouse movements of the slider can be recorded and then played back every time a MIDI note is pressed. To enable automation, select either One Shot or Loop mode. To record a new sequence, arm recording by clicking on the progress display below the slider (it will flash), then click and drag on the slider. Release the mouse button to finish the recording and then exit recording arm by clicking a second time on the progress bar. Incoming gate signals will re-trigger the recorded movement. It will playback once in One Shot mode, or loop indefinitely In Loop mode. If **Rel** is enabled, the envelope amplitude will fade-out on note release, according to the adjacent release time knob.

Section/Control	Function
LFO 1 and 2	Photone features 2 LFOs. Their controls are similar, but the most important difference is that LFO 1 is polyphonic while LFO 2 is monophonic. **Shape** selects the oscillator type. Normally the frequency of the LFOs is specified (in Hz) using the large knob, however, certain other options are also available, as selected by the Rate Via menu. If **Rate Via** is set to MIDI, the frequency is set in musical time using the bottom-right menu. By selecting NP, LFO 1 (by virtue of its polyphony) can also be set to oscillate at note pitch. In this mode, the LFO frequency can be transposed in semitones with the 2 controls beneath the rate knob. With the **Ph Reset** button enabled, the LFO will restart (at the phase position set beneath) each time a gate signal is received. Finally, there are also two modulation source selector menus that allow modulation of rate (**Mod**) and amplitude (**Amp Mod**).
De-click	If activated, triggering Envelope 1 is always delayed until the next zero crossing in the audio signal. This can prevent unwanted clicks at the beginning of notes.

Appendix 1 – Oscillator 4 Special Wave Types

Photone features a collection of special wavetypes, each with a special modulatable parameter. For these oscillators, the Sym and FM sections are used to route modulation sources to the special parameter.



Note that the sin oscillator is unique in that it features 2 modulatable parameters Drive (from the Sym section), and Phase Offset (from the FM section).

Wave	Description	Mod 1 (Sym)	Mod 2 (FM)	Sync
SinQ	This is a Sine wave sent through a quantizer -effectively reducing the bit rate. The SinQ is useful for electric bass imitation when used with a filter, with the quantisation producing a certain "metallic" quality. An another interesting sound can be obtained by tuning the SinQ very low and then heavily overdriving it with the filter saturator unit. The SinQ can also produce nice sounds when used in unison modes with a small amount of detuning.	Quantisation step size	Quantisation step size	Available
Saw > Square	The oscillator features a Sawtooth and a Square (a pulse at 50% width), routed into a crossfader. The crossfade position is controlled via the Sym and FM sections. Note that only the crossfade position is modulated, the width of the square is always static) Nice for some growling basses when used with the filters.	Crossfade position	Crossfade position	Available
Bass	Two pulse oscillators, that when used with LP Filters this can sound like a "plecktryfied" bass. Sync phase shifts one of the oscillators only, resulting in offset between the two oscillators for different timbres. Try adding a sine wave for warmer sounds.	Pulse width of one of the two oscillators	Pulse width of one of the two oscillators	Always synced via Gate

Wave	Description	Mod 1 (Sym)	Mod 2 (FM)	Sync
Additive	Additive Consists of 6 sine waves, offset to each other in terms of both pitch and amplitude. This is mainly designed for organ sounds and does not necessarily need to be complimented by other oscillators (indeed the 6 sine waves use substantial CPU by themselves). Try using this with the Rotary effect.	Shifts the relative offset of pitch and amplitude between the six sine waves	Shifts the relative offset of pitch and amplitude between the six sine waves	Available
Venom	This is a dirty one.... based on two pulse oscillators and a distorted high-pass filter. The Sync phase shifts one of the oscillators only, resulting in offset between the two oscillators for different timbres (a very subtle effect). Try using with the Peak, Notch, BandPass and BandReject filters.	Colour	Colour	Available

Wave	Description	Mod 1 (Sym)	Mod 2 (FM)	Sync
Sinus to Sinus	This is a sine wave sent through a sine function. The sound is similar, but distinguishable from FM. Phase offset has a big influence on the sound, and this is controlled exclusively by the FM section. For this reason, a '1 constant' is included in the modulation source so that the FM modulation depth bar can then be used as a normal slider. When carefully configured, this oscillator can produce UK Garage and 2-Step bass-type sounds, and also electric piano sounds. Using the trackers (Sc3) can be particularly useful with this oscillator. Try combining with the saturators and/or the Shaper in Sine mode.	Drive	Phase offset	Available
Wurly	Wurly This oscillator imitates the 'octave over-rolling' of a Wurlitzer/Epiano in the lower ranges. Good results can be obtained by setting the amount of over-rolling with trackers and velocity.	Over-rolling amount	Over-rolling amount	N/A

Wave	Description	Mod 1 (Sym)	Mod 2 (FM)	Sync
Geiger	The Geiger oscillator produces impulses at a frequency somewhere between totally random and the note pitch. Try using this as a synchronisation of FM source for other oscillators, or to add some 'vintage' crackle.	Randomness	Randomness	N/A
Click	Click Not actually an oscillator, instead it Sample mix N/A N/A produces a single impulse click on each MIDI note on event. This is useful for triggering the comb filter (for physical modelling) or to colour the attack of a sound. The Click is sample based. You can mix between two samples by using Sym.	Sample mix	N/A	N/A

Appendix 2 – Effect Section Reference

Effect	Description
Chorus	Uses a bank of modulated-delay units to add width, depth and richness to the input signal. D/W defines the dry / wet mix balance; Depth determines the amount of delay modulation; and Rate sets the delay modulation speed. Note that when depth is set to zero (hard left) no delay modulation occurs (i.e. the delay times are static). This can be used for 'early reflections' type reverb. Set 1: Quad chorus, for all purposes especially chords and wide stereo imaging. The input is passed through a 6db low-pass filter. Set 2: Double chorus, featuring one LFO inverted for the second line. This represents the classic stereo chorus configuration. This configuration uses less CPU than the other settings. Set 3: Quad chorus, with a large delay offset. Useful for pads. The input is passed through a low-pass filter followed by a high-pass filter. Set 4. Two delay lines, with feedback for a Flange effect. The input is passed through a low-pass filter followed by a high-pass filter.
Rotary	Emulates a Leslie cabinet, especially useful for organ-type sounds. The 4 sets offer different configurations of rotor speed and acceleration. D/W defines the dry / wet mix balance, and Drive sets the amount of distortion/overdrive. Mode selects between 3 rotary speed control modes. At hard left, the rotary is permanently in fast mode. However, when cantered, the MIDI Modulation wheel is used to dynamically switch between slow and fast modes, whereas at hard right, the Pitch-bend wheel is used for the same purpose.

Effect	Description
Distortion	The distortion effect offers 3 different distortion algorithms. D/W defines the dry / wet mix balance; Drive determines the amount of distortion and Colour controls the filtering/tone of the wet sound (dependant on the set). Set 1 and Set 2 use two different overdrive-saturation algorithms. Set 3 uses a feedback-saturator for a more extreme effect. Set 4 uses the same distortion algorithm, but is velocity sensitive.
Resochord	The Resochord Imprints harmonic structures onto the input signal by using 6 comb filters. Chord selects from 16 preset chords; Shift transposes the chord in semitones; and Damp dampens the feedback signal. Set 1: Dry signal mixed with wet signal. Set 2: Wet signal only. Set 3: Dry signal mixed with wet signal, chord is transposed by MIDI note pitch (monophonic). Set 4: Wet signal only, chord is transposed by MIDI note pitch (monophonic).
Delay	A tempo-synced feedback delay unit. D/W defines the dry / wet mix balance; Set controls the delay time; and Feedb. Determines the amount of feedback. Note that at short delay times, the left and right channel times are spread for stereo imaging. Set 1: No L/R channel swapping. Set 2: L/R channel swapping at 50%. Set 3: L/R channel swapping at 100%, and a post delay high-pass filter. Set 4: No channel swapping, delay time is set in multiples of 12th notes (as opposed to 16ths for Set 1 to 3).

Effect	Description
Reverb	For the reverb, sets 1 to 4 offer 4 different room types. D/W controls the dry / wet mix balance; Size determines the room size; and Damp determines the extent of high frequency damping.
Dimension Expander	Subtle room simulator. **Exp** and **Tone** define the room type, **D/W** controls the dry / wet mix balance. Sets 1 to 4 have no effect.

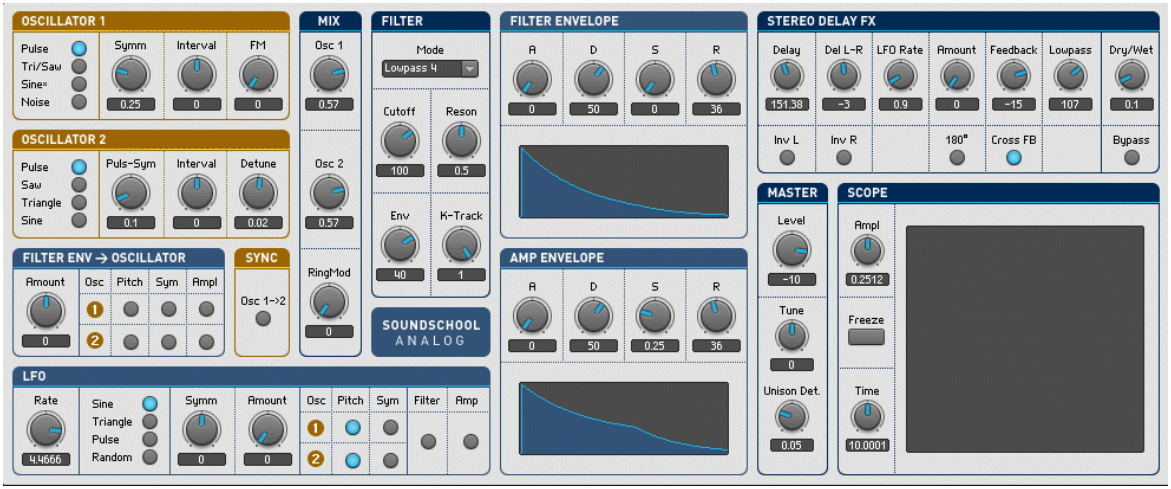
Appendix 3 – Modulation Sources

The following table contains a complete list of modulation sources available in the drop down menus. (Not all sources are available to all parameters).

Name	Description	Location/Source
TrkA	Key Tracker A Key Tracker, routable to individual oscillator amplitude and also master amplitude	Tracking section
TrkA Inv	As above, inversed.	Tracking section
TrkF	Key Tracker F Key scaler hardwired to FM depth, but also routable to other destinations.	Tracking section
Trk3	Key Tracker 3 (General-purpose key scaler)	Tracking section
E1	Envelope 1 (AHDSR envelope generator)	Modulation section
E2	Envelope 2 (AHDSR envelope generator)	Modulation section
E3	Envelope 3 (AHD envelope generator)	Modulation section
E4	Envelope 4 (Recordable envelope)	Modulation section
LFO1 (or LF1)	LFO 2 (Polyphonic low frequency oscillator)	Modulation section
LFO2 (or LF2)	LFO 2 (Monophonic low frequency oscillator)	Modulation section

Name	Description	Location/Source
Key	Note position Generates a value of 0 to 1 according to the note position (MIDI note 36 to MIDI note 96)	MIDI
Vel	Velocity Note on velocity	MIDI
MW	Modulation wheel (CC1)	MIDI
PB	Pitchbend wheel (Bipolar)	MIDI
PB+	Pitchbend wheel (Upward movements only)	MIDI
PB-	Pitchbend wheel (Downward movements only)	MIDI
AT	Aftertouch	MIDI
Vibr	Vibrato low frequency oscillator (Routable to filter cutoff only)	Pitch Mod section
Voice	Voice 1/voice number (Routable to pan only)	(Internal)

8.16 SoundSchool Analog



1.24 SoundSchool Analog user interface

8.16.1 Introduction

SoundSchool Analog is a well equipped but easy to use software-synthesizer. It is very suitable as an introductory tool to the subtractive synthesis realm of both analog and virtual analog synthesis.

8.16.2 Interface Areas and Control Elements

Oscillator 1

The first oscillator (Osc 1) is capable of generating the following waveforms:

- **Pulse**: A rectangular signal changing between two levels. Another interpretation is to distinguish between pulse and pause. You can adjust the pulse width with Symm, the LFO, and with the Filter Envelope.

- **Tri/Saw**: Tri stands for triangle and Saw for sawtooth. The triangle's curve has a linear shape. The symmetry is a product of the relation between rise time and decay time. It‘s adjustable by Symm and can also be controlled with the LFO, and with the Filter Envelope.
- **Sine***: This signal has less overtones (at Symm = 0). It is parabolically shaped. The relation between rise time and decay time (symmetry) of the signal form is variable. It is adjustable by Symm and can also be controlled with LFO, and with the Filter Envelope.
- **Noise**: produces (white) noise. It is based on a fast-running random generator and has no pitch and no adjustable signal form.

Symm lets you set the symmetry (or respectively the asymmetry) of the Pulse, the Rectangle, and the Sine waveform: At a value of 0, it is totally symmetric (50:50). At higher values, the negative part of the Pulse and the increasing parts of the Rectangle and Sine waveform are getting longer, while the positive (and resp. falling) part is shortened. This also creates a brighter sound, because short pulses as well as steep flanks manifest in increased overtones. At a value of 1, a triangle becomes a sawtooth with a vertically falling flank. Sine behaves quite similarly. Pulse becomes infinitively narrow and disappears.

With the **Interval** controller, you can detune the pitch of the triggered note by up to 60 half tones (5 octaves) up or down.

FM determines the Osc 2's modulation frequency amplitude using the signal of Osc 1. The value equals the maximum change of frequency in Hz. The frequency modulation produces a complex mixture of additional frequencies in Osc 2, which can be very useful for metallic/noisy sounds.

Oscillator 2

The second oscillator (Osc 2) can be frequency modulated by Osc 1 and hard synchronized to Osc 1. It is capable of generating the following waveforms:

- **Pulse**: A rectangular signal changing between two levels. Another interpretation is to distinguish between pulse and pause. You can adjust the pulse width with Symm, the LFO, and with the Filter Envelope.
- **Saw**: A sawtooth waveform, not adjustable.
- **Triangle**: A triangle waveform, not adjustable.
- **Sine**: A pure sine without any overtones, not adjustable.

Puls-Symm lets you set the symmetry (or respectively the asymmetry) of the Pulse: At a value of 0, it is totally symmetric (50:50). At higher values, the negative part of the Pulse is getting longer, while the positive part is shortened. This also creates a brighter sound, because short pulses as well as steep flanks manifest in increased overtones. At a value of 1, the Pulse becomes infinitively narrow and disappears.

With the **Interval** controller, you can detune the pitch of the triggered note by up to 60 half tones (5 octaves) up or down.

Detune allows for fine detuning of the second oscillator. This lets you adjust floating and well-tuned intervals between both oscillators. The controller's range is a half tone up and a half tone down, with a resolution of 1/100 halftone (= 1 cent).

Sync

Osc 1>2 activates hard synchronization of Osc 2 to Osc 1. When active, the phase of Osc 2 is set to zero for every time Osc 1's curve passes zero upwards. Independently of the current position, the waveform starts again from the beginning. The pitch of the signal created is determined by Osc 1, while the sound is strongly dependent on the waveform and the frequency of Osc 2.

Mix

The output of both oscillators is mixed in the mixer module. Additionally, a ring modulator creates an amplitude-modulated product of both signals, which consists of mostly disharmonic frequencies. The signals are mixed using mixer controls Osc 1, Osc 2, and RingMod.

Filter

The filter's transmission characteristics are frequency dependent and have a strong influence on the tone color. Overtones can be partially emphasized (by resonance) or softened.

The **Mode** menu offers ten different types of filters and bypass:

- **Bypass**: Deactivates the filter. Signals are sent directly to the amp.
- **Notch 4**: A 4-pole notch filter, which strongly damps a narrow frequency range nearby the cutoff frequency, causing a "hole" in the frequency response. With increasing resonance, the notch becomes more narrow and you will hear resonance effects on its margins.

- **LP Notch 4**: The combination of a 4-pole notch filter and a 4-pole lowpass (LP= lowpass). The effect is quite similar to a lowpass, but by resonance peak and notch both affecting the wavefiorm at the same time, the resonance effect becomes more complex and softened.
- **Highpass 1**: 1-pole highpass filter with a frequency response rising by 6dB/octave below the cutoff frequency and running even after. No resonance peaking.
- **Lowpass 1**: 1-pole lowpass filter with an even frequency response until the cutoff frequency and falling by 6db/octave after. No resonance peaking.
- **Highpass 2**: 2-pole highpass filter with a frequency response rising by 12dB/octave until the cutoff frequency and running even after. At the cutoff frequency, a peaking or, resp., a self-oscillation appears, depending on the resonance adjusted.
- **Bandpass 2**: 2-pole bandpass filter with a frequency response rising by 6dB/octave before the cutoff frequency and falling by 6dB/octave after. At the cutoff frequency, a peaking or, resp., a self-oscillation appears, depending on the resonance adjusted.
- **Lowpass 2**: 1-pole lowpass filter whith a frequency response running even until its cutoff frequency and falling by 12db/octave after. At the cutoff frequency, a peaking or, resp., a self-oscillation appears, depending on the resonance adjusted.
- **Bandpass 4**: 4-pole bandpass filter with a frequency response rising by 12dB/octave until the cutoff frequency and falling by 12db/octave after. At the cutoff frequency, a peaking or, resp., a self-oscillation appears, depending on the adjusted resonance.
- **Bandpass1/3**: 4-pole bandpass filter with a frequency response rising by 6dB/octave until the cutoff frequency and falling by 18db/octave after. At the cutoff frequency, a peaking or, resp., a self-oscillation appears, depending on the resonance adjusted.
- **Lowpass 4**: 4-pole lowpass filter with a frequency response running even until the cutoff frequency and falling by 24db/octave after. At the cutoff frequency, a peaking or, resp., a self-oscillation appears, depending on the resonance adjusted.
- **Cutoff**: The most important parameter of every filter is the cutoff frequency. This parameter shifts passing range, cutting range, and the resonance spot. The sound can be affected substantially. The values shown are half tones.

- **Reson**: Another important parameter of most filter types, adjusting the intensity of resonance. At higher values, a peaking of the frequency response appears. From a value of 0.97 on, a self-oscillation will be created. The filter can now be used as an additional oscillator.
- **Env**: This controller determines by how far the cutoff frequency is modulated by the filter envelope. It has a bi-polar (negative and positive) range. In the positive range, the cutoff frequency of the envelope moves upwards, in the negative range the envelope works in reverse and moves the cutoff frequency downwards. The effect's range is scaled in halftones.
- **K-Track** (keyboard tracking): With this function, you attach the cutoff frequency to the pitch of the current note. With a value of 0, the pitch doesn‘t affect the filter at all, while at 1, the cutoff filter is lead by the note's pitch.

Filter Envelope

The filter envelope controls the „Attack,“ „Decay,“ „Sustain,“ and „Release“ parameters of a filter, which is also presented graphically. Starting with hitting a key, the filter reaches a peak value, which depends on how hard the key is hit (velocity), and falls back to zero on release of the key. The envelope affects the cutoff frequency via the **Env**-controller, and it can be set to affect other modulation targets via **Filter Envelope > Oscillator**.

- **A** (Attack): This control determines how long it takes the envelope to reach its initial peak.
- **D** (Decay): This control determines how long it takes the envelope to run down from its peak to the sustain level.
- **S** (Sustain): Here you determine the envelope's level during the main sequence of the note's duration.
- **R** (Release): This control determines how long it takes the envelope to decay from sustain level to zero after releasing the key.

Amplitude Envelope

The amplitude envelope controls the „Attack,“ „Decay,“ „Sustain,“ and „Release“ parameters of the note amplification, which is also presented graphically. Starting with hitting a key, the amplification reaches a peak value, which depends on how hard a key is hit (velocity), and falls back to zero on release of the key. This envelope affects the volume of the signal at the output.

- **A** (Attack): This control determines how long it takes the envelope to reach its initial peak.
- **D** (Decay): This control determines how long it takes the envelope to run down from its peak to the sustain level.
- **S** (Sustain): Here you determine the envelope's level during the main sequence of a note's duration.
- **R** (Release): This control determines how long it takes the envelope to decay from sustain level to zero after releasing the key.

Filter Envelope to Oscillator

This function allows for connecting the filter envelope's output to the three control inputs of both oscillators.

- **Amount**: Here you determine the envelope's effect on pitch and symmetry. The controller allows for positive and negative modulations and affects all targets at the same time. The amplitude-modulation will not be affected.
- **Pitch (Osc 1, 2)**: The Pitch switches activate the filter envelope as a control source for the oscillators' pitch. The modulation depth depends on the Amount controller setting.
- **Sym (Osc 1, 2)**: The Sym switches activate the filter envelope as a control source for the oscillators' symmetry. The modulation depth depends on the Amount controller setting.
- **Ampl (Osc 1, 2)**: The Ampl switches activate the filter envelope as a control source for the oscillators' amplitude. The amplitude is determined by the envelope.

LFO

The LFO is a slowly oscillating oscillator, which can be used for controlling purposes.

- **Rate**: Adjust the speed (frequency in Hz) here.
- **Waveform switch**: Choose between sine, triangle, rectangle, and a random signal.
- **Sine**: Smooth shape with rounded peaks and adjustable symmetry.
- **Triangle**: Ramp-like shape, symmetry adjustable.
- **Pulse**: Jerkily changing between two levels, symmetry adjustable.
- **Random**: Rhythmical jumps to randomly generated values.

- **Symm**: Influences the first three waveforms' symmetry. At a value of 0, the signal is perfectly symmetric (50:50), underneath 0, the increasing part (at pulse the upper part) becomes shorter and the decreasing part becomes longer. Above zero, the opposite is the case.
- **Amount**: Here you determine how strong LFO affects the modulation targets. The control affects all targets at the same time.
- **Pitch (Osc 1, 2)**: The Pitch switches activate the LFO as a control source for the oscillators' pitch. The modulation depth depends on the Amount controller setting (max. +/- 20 halftones).
- **Sym (Osc 1, 2)**: The Sym switches activate the LFO envelope as a control source for the oscillators' symmetry. The modulation depth depends on the Amount controller setting.
- **Filter**: The Filter switches activate the LFO envelope as a control source for the oscillators' cutoff frequency. The modulation depth depends on the Amount controller setting (max. +/- 60 halftones).
- **Amp**: The Amp switches activate the LFO envelope as a control source for the oscillators' amplitude. The modulation depth depends on the Amount controller setting. A value of 0.5 would be an equivalent to an amplitude range between 0.5 (= 1 - 0.5) and 1.5 (= 1 + 0.5).

Master

- **Level**: Controls the overall output in dB. (+/- 6 dB are equivalent to a doubling/halving of the amplitude).
- **Tune**: Controls the overall tuning (A3=440 Hz).
- **Unison Det.**: If you use more than one voice at a time while unisono is activated, this control slightly detunes these voices against each other, which creates a very lively and floating sound.

Stereo Delay FX

The Stereo Delay FX consists of two delay units, which delay the audio signal by up to 800ms through digital storage. The delays are wired to the left and the right output channel. The delay time is adjustable and can be modulated by an internal LFO. Moreover, the flanger, chorus and the echo effects depend on the internal mixture and the feedback of the signals.

- **Delay**: Delay influences the (mid-) running time of both delays. In addition, the half time difference (Del L-R) will be subtracted from the running time at the left delay, while it will be added at the right.
- **Del L-R**: Set the difference of the running time of both delays here. This produces more natural sounding reflections.
- **LFO Rate**: The speed of the LFO modulating the delay time.
- **Amount**: The amount of LFO modulation. Use moderately, as it affects pitch modulation.
- **Feedback**: With longer delays, this control sets the number of audible echoes. With shorter delays, the comb filter effect will be amplified by the feedback, which produces intense tone colorings.
- **Lowpass**: The delay signals are filtered by lowpass filters. The cutoff frequencies of those filters are controlled here. The lower you set this control, the more dull the output signal will sound.
- **Dry/Wet**: Control the mix between un-affected, original signal, and effect signal here. With the control all the way left (dry), only the original signal is audible. Turning it to the right (wet), you'll hear more and more of the effect signal.
- **Inv L**: Inverts the left delay's signal before it gets mixed with the other signals. With short delay times, this changes the sound of the comb filter effect.
- **Inv R**: Inverts the right delay's signal before it gets mixed with the other signals. With shorter delay times, this changes the sound of the comb filter effect.
- **180°**: Activate this controller for a reversed LFO modulation of both the left and the right channel.
- **Cross FB**: With this switch turned on, the signals will be fed back cross wise, that is, the left delay's output is fed to the right one's and vice versa.
- **Bypass**: Deactivates the whole effect and the signals are bypassed.

Scope

The oscilloscope provides a graphical representation of the output signal.

- **Ampl**: Here you can set the sensitivity (or the amplitude).

- **Freeze**: Pressing this button freezes the oscilloscope's picture. The amplitude can be manually adjusted afterwards.
- **Time**: Set the length of the oscilloscope's time frame here.

Additional Functions

The **Compare**-control switches between the original and the effect sound. **Default** sets all operating elements to a neutral position.

8.16.3 Page B - Instrument Architecture

Interface page B holds a graphical representation of SoundSchool Analog's internal structure. The oscillators 1 and 2 are depicted in orange, while the signal processing units are shown in dark blue. The sources for controlling and modulation of the signals are represented by a brighter blue.

The audio signals of the oscillators are sent to the mixer and intermingled. The mixed signal gets filtered, amped and then sent to the outputs L and R via Stereo Delay FX (effect).

The Scope (oscilloscope) represents the L-signal as a time curve.

The oscillators are connected in two ways: Oscillator 1 controls the frequency of the second oscillator via **FM** (frequency modulation). **Sync** again allows the second oscillator to be hard synchronized to the first one.

The **Filter Envelope** controls the filter‘s cutoff frequency for every triggered note, while the **Amp Envelope** controls its volume. With the **FILT-ENV>OSC** module, the Filter Envelope can be set to also effect the pitch (**P**), the symmetry (**S**) signal form, and the amplitude (**A**) of the oscillators.

A further source for modulation is the **LFO** (low frequency oscillator), which produces slow oscillations. It has influence on the pitch (**P**) and the symmetry (**S**) of both oscillators, the **Filter**'s cutoff frequency, and the **Amp** (amplifier).

Except for the **Stereo Delay FX** and the **Scope**, all parts of the sound processing are polyphonic. All signals are intermingled after having passed the amp.

The **Stereo Delay FX** consists of two delay units, which are wired to the left and the right output signal.

8.17 Steam Pipe 2



4.55 The Steam Pipe 2 user interface

8.17.1 Introduction

Steam Pipe 2 is a physical-modeling synthesizer that effectively models air being blown through a tunable pipe. It uses a tuned resonator to create bowed, blown, and plucked sounds, as well as strange new hybrid sounds. In addition to a tuned all-pass filter and many controls for the “shape” of the pipe, there is a mod wheel-controlled filter to achieve damping and breath noise effects. The excellent-sounding SpaceMaster Deluxe reverb unit adds dimension to the overall signal. You can find it on panel B.

SteamPipe 2 simulates air passing through a pipe of variable size and resonance. It’s physical-modeling techniques use contoured noise signals passing through tuned and filtered feedback delays. The ensemble is basically split into two parts: Steam and Pipe. The Steam module generates shaped and filtered noise. Think of the Steam module as SteamPipe 2’s oscillator.

Steam provides the sound energy that will be pitch-formed by the Pipe. The Pipe module gives the “wind” pitch and resonance. The patch also has an ADSR volume envelope and a low pass filter. Both can be modulated by key- and velocity-tracking.

Steam Pipe 2 can be a very expressive synthesizer, so make sure that you plug in your MIDI keyboard and check out the presets with the mod wheel in action.

8.17.2 Steam

The timbral shaping of the DC/Noise source occurs in the Steam section. The low pass filter works in 1-pole or 2-pole mode, though the resonance control only applies to the 2-pole filter. After the noise is filtered, the signal is fed into the Pipe module.

Section	Control	Function
Envelope	Attack	Sets the attack time of an ADSR envelope triggered by MIDI gate events and used to generate a short initial steam signal; logarithmic control.
	Decay	Sets the decay time of an ADSR envelope triggered by MIDI gate events and used to generate a short initial steam signal; logarithmic control.
	Sustain	Sets the maximum level the envelope will reach. This gets modulated by velocity if [VelSns] is on.
	Release	Sets the time that passes until the envelope is completely faded out after the note-off signal.
	Velocity	Controls the velocity sensitivity of the envelope. The higher this value is, the higher the peak value of the envelope will be.
	Scaling	This scales the envelope times depending on the pitch of the incoming MIDI notes. Turn to the left for no keyboard scaling, to the right for shorter envelope times on higher notes.
	Legato	Toggles legato mode on or off. If on, the envelope restarts only when the gate changes from zero to positive.
Generator	DC / Noise	Crossfades between the DC component at the left and filtered noise at the right. The mixed signal is used as steam input of the resonating pipe.
	Cutoff	The cutoff frequency of the low-pass filter.
	Reso	Sets the level of resonance of the filter. Only works when the filter is in 2-pole mode.
	Poles	Toggles between 1-pole and 2-pole low-pass.

Section	Control	Function
	Key-track	Controls the key-tracking of the filter. This will scale the cutoff frequency according to keyboard position. The lower the note pitch, the lower the cutoff frequency will be.
	Vel-Track	Controls the velocity scaling of the filter. Turn to the right for higher cutoff frequencies at higher MIDI velocities.
	Env-Amt	Sets the amount of envelope to the cutoff frequency.

8.17.3 Pipe

The Pipe module is made up of a number of sub-modules for creating pitch and resonance. The noise signal is fed from a single tuned delay providing pitch, into the [Allpass] module for generating resonance. Next, a [Saturator] receives the signal and applies edge and break-up. The [MW Filter] completes the signal chain with an overall tone shaping stage. The [Feedback] and [Push-Pull] sections act on signals diverted from the main signal chain and passed back into it via feedback loops. Unlike the [Feedback] section, which simulates the pipe itself, the Push-Pull section controls the air and its oscillations within the pipe.

The [Delay Tune] module contains the tuned delay that provides pitch to the Steam. The [Tune] and [Fine tune] knobs allow you to set the signal's fundamental pitch. The A440 oscillator at the bottom of the ensemble provides a reference pitch for tuning purposes. The Delay pitch can be swept negatively or positively by the mod wheel, with the amount of modulation set by the [MW] knob.

The Allpass filter receives the tuned signal from the resonant delay. It can be turned on and off with the [Power] button in the [Allpass] section. This allpass can be tuned to create resonance effects. You can produce glassy, metallic and bell-like sounds by detuning the allpass filter against the delay. By adjusting the [Diffusion] knob, you can also create a variety of reverb sounds - the simulation of air echoing along a pipe's hard surface. The Saturation module morphs between saturation and clipping, overdriving or breaking up the signal before it hits the MW Filter.

The [MW Filter], controlled by the mod wheel, features a 1-pole high pass followed by a 1-pole low pass filter. Each filter allows you to set a wheel-down and a wheel-up setting, making it possible to set up complex timbre changes and damping effects. Each filter can have its own [key track] setting.

The [Polarity] switch inverts the pipe polarity, changing the timbre of the sound. This often transforms high frequency tones to deep ones and vice versa.

The [Feedback] module processes the feedback in the signal chain. The [Rev-Time] knob extends or shortens the reverb generated by this feedback signal. The reverb signal can be muffled with the [Damp] control. Damping can be modified by [Key-Track] amount. High [Key-Track] values result in more damping on higher pitches. This allows SteamPipe 2 to emulate struck or plucked instruments like pianos, harps, and acoustic guitars.

Section	Control	Function
Delay Tune	Tune	Sets the fundamental pitch of the signal. For standard musical tunings set it to the [A440] oscillator at the bottom of the patch.
	Fine Tune	Fine control of signal pitch.
	SREC	Sample rate error correction. Adjusts the tuning correction of the pipe. When the signal modifications in this patch and REAKTOR's sampling rate interfere with the physical model of SteamPipe, this extra-fine pitch tuning becomes necessary. Tune against the A440 section.
	Mod-Whl	Sets the amount of pitch modification by the MIDI modulation wheel. This simulates pitch changes of pipes getting blown softer or harder.
Feedback	Rev-Time	Adjusts the time of the pipe's reverberation, i.e. the amount of damping applied to the feedback's signal before it is mixed again with the new incoming signal. The longer the reverb time, the more the incoming noise steam signal becomes a tone with recognizable pitch.
	Damp	Sets the amount of high frequency damping of the pipe at key-up.
	Key-track	Controls the feedback's key-tracking. Turn to the right for longer reverb times at high MIDI pitches.
Allpass Tune	Tune	Controls the pitch of the allpass resonance. If the allpass filter is switched off, this control shows no effect.
	Fine Tune	Fine tunes the allpass resonance pitch. If the allpass filter is switched off, this control shows no effect.
	SREC	Sample rate error correction. Adjusts the tuning correction of the pipe. When the signal modifications in this patch and REAKTOR's sampling rate interfere with the physical model of SteamPipe, this extra fine-tuning of pitch becomes necessary. Tune against the A440 section. If the allpass filter is switched off, this control shows no effect.

Section	Control	Function
	Mod-Whl	Sets the amount of pitch modification by the MIDI modulation wheel. This simulates pitch changes of pipes getting blown softer or harder.
Allpass	On / Off	Turns the allpass module on or off. Switch on for additional attack effects of the pipe's sound.
	Diffusion	Sets the diffusion of the resonances generated by the allpass module. Turn to the left for additional attack effects of the pipe's sound. It also enhances the sound of harmonic frequencies which are not multiples of the main pitch, like e.g. in bells.
Push-Pull	Offset	Sets the offset amount added to the reverberating steam signal. This parameter influences the incoming steam and its reverberation in the pipe. It interacts tonally with the Polarity button.
	Push	Sets the amount of reverberating steam.
Saturation	Soft / Hard	Controls the balance between soft saturation and hard clipping.
	Symmetry	This parameter introduces level asymmetry into the signal. With increasing asymmetry the positive part of the signal is reduced.
Polarity	Polarity	This control inverts the polarity of the pipe, thereby changing the timbre of the sound. It interacts tonally with the [Push-Pull] section.
Mod-wheel to filter	Hi Pass 0	Sets the cutoff frequency of an additional highpass filter within the pipe to enhance formant frequencies. The formants are modified by the pressure in the pipe (not by the pipe's pitch). The pressure can be controlled by the mod wheel. At low modulation wheel values this knob is used to determine the formant frequency.
	Hi Pass 1	Sets the cutoff frequency of an additional highpass filter within the pipe to enhance formant frequencies. The formants are modified by the pressure in the pipe (not by the pipe's pitch). The pressure can be controlled by the mod wheel. At high modulation wheel values this knob is used to determine the formant frequency.
	Key-track High	Controls the amount of key-tracking applied to the highpass filter's cutoff frequency. Turn to the right for higher cutoff frequencies at high MIDI pitches.

Section	Control	Function
	Lo Pass 0	Sets the cutoff frequency of an additional lowpass filter within the pipe to enhance formant frequencies. The formants are modified by the pressure in the pipe (not by the pipe's pitch). The pressure can be controlled by the MOD wheel. At low modulation wheel values this knob is used to determine the formant frequency.
	Lo Pass 1	Sets the cutoff frequency of an additional lowpass filter within the pipe to enhance formant frequencies. The formants are modified by the pressure in the pipe (not by the pipe's pitch). The pressure can be controlled by the mod wheel. At low modulation wheel values this knob is used to determine the formant frequency.
	Key-Track Low	Controls the amount of key-tracking applied to the lowpass filter's cutoff frequency. Turn to the right for higher cutoff frequencies at high MIDI pitches.

8.17.4 Global Controls

The last section of SteamPipe 2 consists of global controls over pitch, polyphony, glide, and an output stage. You also get an Arpeggiator and a test tone generator.

Section	Control	Function
Voice Mode	Pitch Bend	Sets the range of the pitch bend wheel.
	Detune	Introduces a slight detune into the signal for a livelier sound.
	Mode	Menu for the different polyphony modes. Choose between poly, mono, unison, and three arpeggiator modes.
	Glide on / off	Toggles glide on or off.
	Glide Time	This sets the time the pitch of SteamPipe takes to follow incoming MIDI pitches, when [Glide] is on.
	Mod-Whl	This knob follows an incoming mod wheel signal. Use it when you have no hardware controller available.
	Arp Mode	This menu offers different arpeggiation modes. Choose between up (>>), down (<<), up and down (>><<), and random.
	Arp Speed	Menu for choice between different speeds relative to the global tempo.

Section	Control	Function
Output	Spread	Introduces a stereo spread into the main output.
	Gain	Main output volume control.
A440	A440 Tuning tone on / off	If on, a sine oscillator's signal is mixed into the main output. Use it to tune the pipe. The frequency is 440Hz.
	Gain	Controls the volume of the 440Hz tuning tone.

8.17.5 Space Master Deluxe

You can find this remarkable reverb module on panel B of the SteamPipe 2. Based on several diffusion delays, Space Master can produce a wide array of high-quality natural or experimental ambiences. The patch's useful set of reverb parameters include an early reflections section, a late reflections module, and a post EQ. Dials for main reverb time, control of balance between the two reflection stages, and between dry and wet signal round off the controls.

8.17.6 Input and Output Stage

You can put an initial delay into the reverb signal with the predelay [Time] dial and control the predelay's stereo position with the [Symmetry] knob. The [Early / Late Balance] slider can be used to move the source in space – more early reflections bring the signal to the front and more late reflections make it appear further back in space. At the end of the signal chain the [Dry / Wet] slider crossfades between original signal and reverb.

Section	Control	Function
Predelay	Time	Sets an initial delay for the wet signal.
	Symmetry	Introduces a difference into the delay times for the right and left predelay channels. Use this to position the signal around in the stereo field.
Mixing	Early/Late Balance	With this parameter you can set how much of the early and late reflections, respectively, can be heard in the output.
	Dry / Wet	This controls the balance between dry and wet signal.
	Power button	Switches the reverb on or off.

8.17.7 Reflections

Use the two [Size] and [Diffusion] parameters to control two stages of variable density diffused reflections. The early stage commonly represents the direct response of the virtual space, whereas the late reflections define the sound when the early reflections have died away. For dynamic reverb effects you can use the Modulation section. It offers an LFO routed to the delay times with [Rate] and [Depth] control. The LFO can enhance your reverb signal by adding liveliness.

Section	Control	Function
Early/Late Reflections	Size	Determines the amount of space generated by the early or late reflections modules by adjusting delay time of the underlying diffusion delays. Higher values give the impression of larger spaces.
	Symmetry	Introduces a stereo shift into the generated reflections.
	Diffusion	Adjusts the perceived density of the reflections generated. Use for a sparser or fuller reverb sound.
	Reverberation Time RT60	This control alters the decay time of the reverb response.
Modulation	Rate	Control of LFO frequency modulating the delay times.
	Depth	Control for the LFO's modulation depth. Higher values yield increased amplitude modulation.

8.17.8 Frequency Response

The two EQ sections serve slightly different needs. The Damping EQs are integrated into the reflection stages and influence their frequency responses. The Post EQ acts on the patch's main output and should be used to finalize the overall sound.

Section	Control	Dsecription
Damping	Low Freq	Cutoff for the low shelving filter that cuts into diffusion delay frequency response of both early and late reflections.
	High Freq	Cutoff for the high shelving filter that cuts into diffusion delay frequency response of both early and late reflections.

Section	Control	Dsecription
	Lo Damp	Amount of cut for the low shelving filter.
	Hi Damp	Amount of cut for the high shelving filter.
Equalizer	Low Freq	Cutoff for the low EQ that acts on the main output of the reverb.
	High Freq	Cutoff for the high EQ that acts on the main output of the reverb.
	Lo Boost	Cut or boost amount for the low Equalizer.
	Hi Boost	Cut or boost amount for the high Equalizer.

8.18 Steam Pipe



1.25 Steam Pipe user interface

8.18.1 Introduction

Steam Pipe is a physical-modeling synthesizer that uses a tuned resonator to create bowed, blown, and plucked sounds, as well as many strange new and hybrid sounds. SteamPipe effectively models air, or steam, being blown through a tuneable pipe. In addition to a large number

of controls for the "shape" of the pipe, and a tuned all pass filter, there is a mod wheel-controlled filter to achieve damping and breath noise effects. Giving dimension to all of this is an excellent-sounding reverb unit.

8.18.2 Quick Start

Since SteamPipe is such a potentially expressive instrument, you'll want to plug in your favorite MIDI keyboard to try it out. Step through the presets to get an idea of the different kinds of sounds this thing can make. Give the mod wheel a twirl or two and hear how this can add expression to the sound.

8.18.3 Structure and Signal Flow

SteamPipe simulates the passing of air through a pipe where the size and resonance can be modified. It uses a type of synthesis called physical modeling, which uses contoured noise passing through feedback delays that can be tuned and filtered. The ensemble is basically separated into three parts: Steam, Pipe, and R66, the reverb module. The Steam module generates shaped and filtered noise. The Pipe module gives the "wind" pitch and resonance. R66 provides the space.

You can think of the Steam module as SteamPipe's oscillator, or at least part of it. Steam provides the sound energy that will be pitch-formed by the Pipe module. To this end there are an ADSR volume envelope, modulated by velocity and keyboard scaling (meaning that the envelope can be set to be shorter or longer overall as effected by velocity), and low pass filter with key- and velocity-tracking.

The Gen section of the Steam module is where the timbral shaping of the DC/Noise source takes place. The low pass filter works in 1-pole or 2pole mode, though the resonance control only applies to the 2-pole filter. After the noise is filtered, the signal is fed into the Pipe module.

The Pipe module is made up of a number of submodules for creating pitch and resonance. The noise signal is fed from a single, tuned delay, which provides pitch, into the Allpass module for resonance. A Saturator receives the signal next and applies edge and breakup. The MW Filter finishes off the signal chain, giving one final overall tone shaping stage. The FeedBack and Push-Pull sections act on signals diverted from the main signal chain and passed back into it via feedback loops.

The Del Tune module contains the tuned delay that provides pitch to the Steam. The Tune and Fine knobs allow you to dial in the fundamental pitch of the signal. The A440 oscillator at the bottom of the ensemble is there to give a stable pitch to tune the pipe to. The Err knob introduces small amounts of detuning to the pitch, to give a more human quaver. The Delay pitch can be swept negatively or positively by the mod wheel, with the amount of modulation set by the MW knob.

The Allpass filter receives the tuned signal from the single delay. It can be turned on and off with the Power button in the Allpass section. The Allpass can be tuned to create resonant effects. You can make glassy, metallic and bell-like sounds by detuning the All pass filter against the Delay. By adjusting the Diffusion knob, you can also create a variety of reverb sounds - the simulation of air echoing along a pipe's hard surface.

The Saturation module morphs between saturation and clipping, overdriving or breaking up the signal before it hits the MW Filter.

The MW Filter, controlled by the mod wheel, features a 1-pole high pass followed by a 1-pole low pass filter. Each filter allows you to set a wheel-down and a wheel-up setting, making it possible to set up complex timbre changes and damping effects. Each filter can have its own key track setting.

The Ione Polarity switch inverts the pipe polarity, changing the timbre of the sound, often turning high frequency tones to deep ones and vice versa.

The FeedBack module processes the feedback in the signal chain. The RT knob extends or shortens the reverb generated from the feedback signal. The reverb signal can be muffled with the Damp control. Damping can be modified by K-Track amount know. High K-Track values result in more damping on higher pitches. This allows SteamPipe to emulate struck or plucked instruments like pianos, harps, and acoustic guitars.

R66

The final step in SteamPipe's processing chain is the R66 reverb module. R66 is a full featured and rich-sounding reverb with lots of parameters to adjust. First, the Del module uses predelay time and left/right panning offset to position the signal in time and space.

The Diffusion module has controls for size and reverb time and has a low pass filter and low and high frequencies damping in order to model a convincing space. There are also controls to modulate reverb time and pan position with a sine wave or random LFO, or both. The Pos knob lets you set the reverb reflections early or late, further coloring the sound of your virtual space.

8.19 SubHarmonic



4.56 SubHarmonic user interface

8.19.1 Introduction

SubHarmonic generates pad-like, atmospheric sounds and – at the same time – thick, monophonic lead patches. It consists of two independent sound generators: The [Sub Oscillator] is based on additive synthesis, and the [Formant] section performs like an oscillator with a constant frequency vowel filter. Internally, those oscillators are quite complex; the [Sub Oscillator], for example, does not use normal harmonics of the main frequency to perform additive synthesis, but produces subharmonics. However, those technical details remain hidden below the simple user interface.

8.19.2 Voice

This section deals with the voice assignment. You have the normal monophonic and polyphonic modes, and an additional unison mode that uses all available voices (like polyphonic) but sets them to the same pitch, only slightly detuned. The resulting phase interferences produce a chorus-like effect.

The amount of portamento and the influence of the MIDI pitchbend wheel onto the instrument can also be adjusted here.

Control	Function
Voice Mode	Controls the voice assignment of the instrument. [Poly] selects polyphonic, [Mono] switches to monophonic behavior; [Uni] is also monophonic, but uses all available voices which are slightly detuned to each other.
Detune	Sets the amount of detuning in [Uni] mode. Turn to the left for larger intervals between the voices.
Glide	Switches portamento on or off (see also [Speed]).
Speed	Adjusts the amount of portamento, i.e. the time that passes until a newly received MIDI note's pitch is gradually reached.
Pitchbend	Sets the range of the MIDI pitchbend wheel in semitones.

8.19.3 Vibrato

Crucial to this instrument's sound is the vibrato effect. It is produced by mapping an LFO's signal onto the instrument's pitch. The LFO waveform, its frequency, and the amount of pitch modulation can be controlled here, providing settings that differ from the normal idea of musical vibrato but generate impressive sounds nonetheless.

Control	Function
Shape	Selects the waveform of the LFO whose signal is used to modulate the instrument's pitch for vibrato effects.
Rate	Sets the frequency at which the vibrato LFO oscillates.
Width	Sets the pulse width of the vibrato LFO; turn to mid position for a symmetric waveform.

Control	Function
Amount	Sets the amount of vibrato. Turn to the left for no vibrato, to the right for a pitch modulation of the range adjusted by [Range].
Range	Controls the absolute vibrato amount in semitones (see also [Amount]).

8.19.4 Amplitude and Modulation Envelope

These two envelope generators, placed to the left (modulation envelope) and the right (amplitude envelope) of the [Voice] and [Vibrato] sections, shape the instrument's amplitude and modulate the feedback amount of the [Sub Oscillator] as well as the formant frequency of the [Formant Oscillator]. They operate as normal ADSR envelopes, but offer additional re-triggering options, key-scaling, and adjustable MIDI velocity sensitivity.

Control	Function
Mode	Selects the way the envelope generator reacts to a new gate signal when the previous gate is not yet closed. In [Leg] mode the envelope generator doesn't react to the new gate signal; in [Ret] mode it is re-triggered, using the current envelope level as a starting point; in [Rst] mode the generator is also re-triggered, but reset to its initial level. This control shows no effect if [Voice][Voice Mode] is set to [Poly].
Attack	Sets the attack time of the envelope generator.
Decay	Sets the decay time of the envelope generator.
Sustain	Sets the sustain level of the envelope generator.
Release	Sets the release time of the envelope generator.
Key	Sets the amount and polarity of key scaling applied to the envelope generator's transition times. Turn to the left for faster transitions at low pitches, to the right for slower transitions at low pitches.
Velocity	Sets the amount of MIDI velocity influence on the envelope's amplitude. Turn to the left for constant amplitudes that are independent of velocity, and to the right for full velocity sensitivity.

8.19.5 Sub Oscillator

This oscillator creates the fundamental MIDI pitch which generates four subharmonics below this frequency. The ratio of the harmonics to the main pitch and their amplitudes can be controlled individually, similar to a standard additive synthesizer. A feedback feature provides basic waveform modulation, crossfading smoothly from a sine wave to one that sounds like a sawtooth waveform.

Controls	Function
Pitch	Controls the pitch shift of the sub oscillator's main frequency; there is a coarse (top) and a fine (bottom) control. As the sub oscillator generates harmonics below the main frequency, the pitch control shifts those sub tones to a usable frequency.
Harmonic A/B/C/D	Select four sub harmonics of the main frequency. Their volume is controlled by the respective [Amplitude] control.
Amplitude A/B/C/D	Select the amplitude of the corresponding sub harmonic adjusted by the [Harmonic] control.
Feedback Amount	Sets the amount of feedback applied to the sub oscillator internally. Turn to the left for a undistorted sine-like sound, to the right for a saw-like signal.
Envelope Modulation Amount	Controls the modulation amount applied to the [Feedback amount] by the [Modulation Envelope].

8.19.6 Formant Oscillator

This particular oscillator is made up by a simple sine waveform; however, the adjustable frequency band doesn't move with the oscillator's pitch, but remains stable as a formant of the sound. By moving this frequency band you can achieve vowel filter-like effects.

Control	Function
Pitch	Controls the pitch of the main frequency; there is a coarse (top) and a fine (bottom) control.
Formant Frequency	Adjusts the frequency of the oscillator's formant; this formant is not changed by the main pitch.
Envelope Modulation Amount	Controls the amount and polarity of modulation applied to the [Formant Frequency] by the [Modulation Envelope]. Turn to the left for inverse modulation, i.e. low formant frequencies at high modulation signals, and to the right for normal modulation, i.e. high formant frequencies at high envelope levels.

8.19.7 Mix and Output

In this section the signals of both oscillators can be mixed, positioned in the stereo field, and leveled.

Control	Function
Mix	Crossfades between the sound of the sub oscillator (at the left) and the signal of the formant oscillator (at the right).
Spread	Sets the amount of displacement within the stereo field applied to the instruments' voices. Turn to the left for a mono signal, and to the right to pan each voice individually; this is particularly impressing if [Voice][Voice Mode] is set to [Uni].
Gain	Sets the output gain.

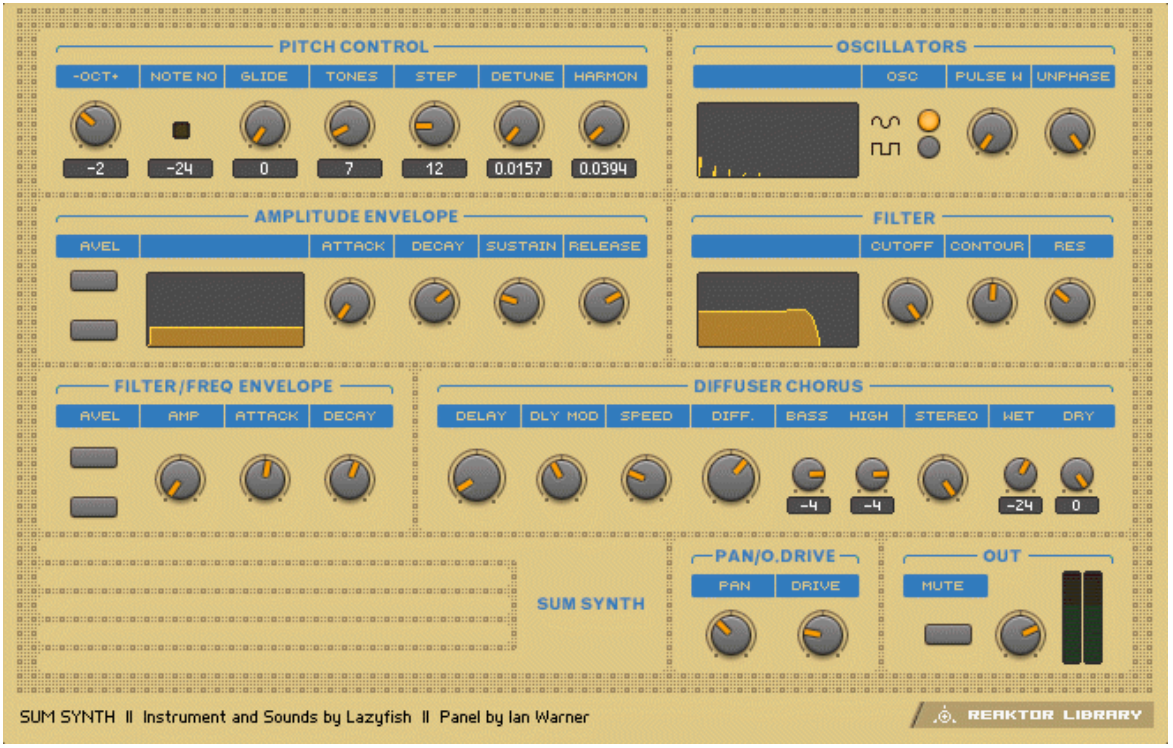
8.19.8 Reverb

The high-quality reverb unit is contained within the panel's B view. It can further enhance the sound's spatial characteristics. When not in use it should be turned off by the [Power] control to save CPU power.

Control	Function
Size	Sets the size of the virtual reverberation room.
Symmetry	Places the signal in the virtual reverberation room. Turn to the left or right to move the signal away from the center.
Diffusion	Sets the amount of reverb signal diffusion. Turn to the right for a less echoic sound.

Control	Function
Release	Adjusts the time that passes before the reverberation sound has decayed.
Spin	Sets the amount of modulation applied to the reverb. Technically, the modulation affects the delay time of the delay modules on which the reverb is built.
Frequency	Sets the rate of the LFO used as modulation source (see [Spin]).
High Cutoff	Sets the cutoff frequency of the lowpass filter that is damping the high frequencies.
High Damp	Sets the amount of damping applied to the frequencies above the [High Cutoff] frequency.
Low Cutoff	Sets the cutoff frequency of the highpass filter that is damping the low frequencies.
Low Damp	Sets the amount of damping applied to the frequencies below the [Low Cutoff] frequency.
Mix	Crossfades between the unprocessed, dry signal (at the left) and the reverberated, wet sound (at the right).
Power	Switches the reverb unit on or off. Turn off to save CPU power when the reverb is not in use.

8.20 Sum Synth



1.26 Sum Synth user interface

8.20.1 Introduction

Sum Synth is an additive synthesizer with a few tricks up its sleeve. A unique oscillator section makes it easy to control the frequencies of the oscillators and shape of the sound, while an informative graphical display shows the current pitch of each oscillator. The number of oscillators can easily be varied simply by changing the polyphony of the instrument in the instrument title bar.

8.20.2 Quick Start

Get your MIDI keyboard warmed up and check out the presets. Pay special attention to the Tones, Step, Detune, and Harmonics controls since they have a big effect on Sum Synth's sound. See how the four controls interact to change the width and quality of the sound.

8.20.3 Structure and Signal Flow

Sum Synth uses additive synthesis techniques to generate its deep tones. It has sine or square oscillator waveshapes with the capabiltity to produce up to 128 additive waves, and a sound-shaping section consisting of a stereo 4-pole lowpass filter, an overdrive effect, and the Diffuser Chorus. Two envelopes further shape the sound -- one for volume, and one for filter frequency, which can also modulate oscillator pitch.

The oscillator can be switched between sine and square (with adjustable pulse width) waveforms. The Pitch Control module contains controls to divide the oscillator pitch into detunable tones, select the initial octave, and add glide between notes. The Step knob lets you separate the tones from each other by musical intervals. Detune, will, as you can guess, detune the intervals from each other to give chorusing or atonal sounds. The Harmonics knob generates ring mod-like effects between the tones. Along with a pulse width control for the pulse oscillator, there is an Unphase knob, which causes the oscillator tones to trigger successively out of phase with each other, thickening the sound.

The oscillator feeds into the Pan/Overdrive module. The Drive knob allows you to overdrive the stereo amplifier that the oscillator is fed through. Use the Pan control to widen the stereo soundfield.

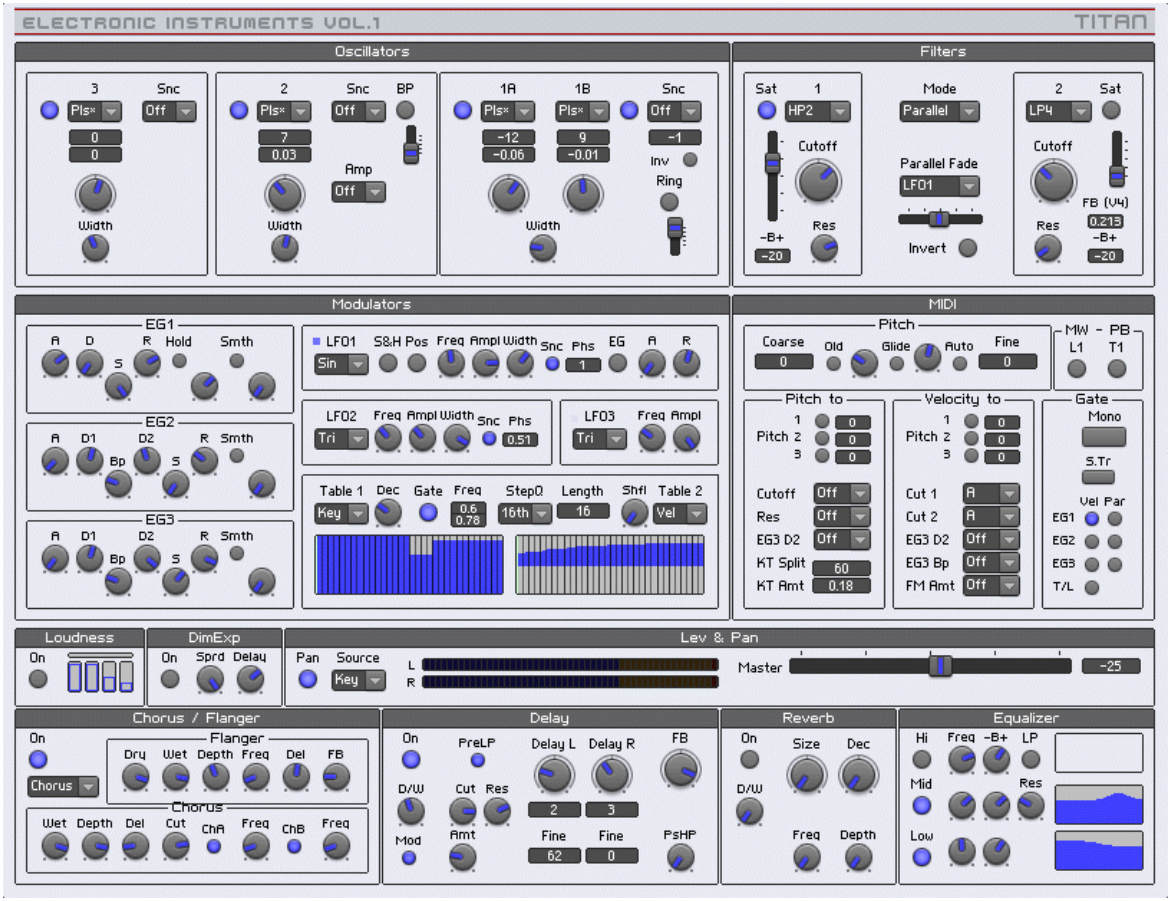
After the Overdrive circuit, the signal reaches the stereo lowpass filter. The Filter module gives control over cutoff frequency, resonance, and envelope contour amount. The filter modules are taken from the Pro-52 and give out a rich, creamy sound.

The Filter/Freq AHD envelope works the same way as the Amplitude envelope. Hold/sustain can be activated by velocity, allowing you create dynamic bass drum patterns and expressive lead lines. Velocity can also be routed to filter contour amount. The AmP knob routes the Filter envelope signal to the oscillator pitch. This can be useful for making the little blip at the beginning of those classic bass drums.

The Amplitude ADHSR envelope has a hold stage that can be activated by velocity - click the bottom button to the left of the amplitude envelope display to turn it on. Less velocity will result in more hold time, which can create interesting gating effects in your playing. Velocity can also be routed to volume by clicking the topmost of the two buttons.

The Diffuser Chorus consists of four stereo modulation delays. The main delay time is controlled by the Delay knob. Diffusion sets the inaccuracy of the delay times to "spread" the sound. The Dly Mod control adds an LFO to modulate delay time. The Speed knob sets the rate of modulation. The Stereo knob offers still one more method to spread the sound across the stereo field. A two-band equalizer lets you shape the chorus tone by attenuating the bass and treble frequencies. Clicking the buttons above the Bass and Treble faders engages the EQ bands.

8.21 Titan



2.7 Titan user interface

Introduction

Titan is a very powerful synthesizer that features three independently controlled oscillators, three envelopes, two flexible event table modulators, three LFOs, built-in effects, and a comprehensive modulation matrix.

Signal Flow

Oscillators > Filters > Loudness Corrector > Dimension Expander > Chorus > Delay > Reverb > 3-band EQ

Oscillators

The oscillator section contains three versatile oscillators (OSC1 [double OSC], OSC2, and OSC3). A different waveform can be selected for each oscillator; those marked by an asterisk

* can be pulse width modulated. All oscillators can be tuned, detuned, and mixed. The oscillators 1 and 2 contain additional features explained below.

For each oscillator, the waveform can be selected. Not all oscillators offer all possible waveforms.

- <Off> Turns the oscillator off.
- <Sin> Sine waveform.
- <Par> Parabol waveform.
- <Par*> Parabol waveform with pulse width control.
- <Tri> Triangular waveform
- <Tri*> Triangular waveform with pulse width control
- <Saw> Sawtooth waveform.
- <Pls*> Pulse waveform with pulse width control
- <BiP*> Bipolar pulse waveform with pulse width control.
- <Imp> Impulse waveform.
- <Nse> White noise.

Beside the waveform selector there are the following knobs for each oscillator:

- <P>, <Fine> Set the pitch offset of the oscillator to the main pitch in semitones and cents, respectively.
- <Amplitude> Sets volume of the oscillator.
- <Width> Sets the pulse width. This affects only those waveforms that allow pulse width control; if another waveform is selected, this knob has no effect.

OSC1 is a dual oscillator and offers dual waveform select, tune, detune, and amplitude controls. OSC1 also offers the following unique features:

- <Snc> Sets the synchronization mode: It can be off, the oscillators can be synchronized to the midi gate or to osc3.
- <Phs> Sets the phase of the waveform when the oscillator is re-triggered by the sync signal; thus, there is no effect when synchronization is off. Turn to -1 for a phase of -180¡, to 1 for a phase of +180¡.
- <Ring> Switches the ringmodulation of the oscillators osc1 and osc3 on or off. <RM*3>
- <Amount> Crossfades between the "normal" oscillator output at the left and the ringmodulated signal at the right. If <Ring> is off, this knob has no effect.

Additionally, OSC2 offers:

- <Envelope> Selects the envelope (EG1, EG2, EG3) used to modulate the amplitude of this oscillator. Its peak value is specified by the <Level> knob. If <lev> is selected, the <Level> knob output is used without envelope modulation like in all other oscillators. <<<>
- <Filter> Switches a bandpass filter on or off to modify the sound colour of this oscillator.
- <Freq> Sets the center freqeuncy of the bandpass filter switched on by <Filter>. Turn left for low freqeuncies, right for high ones.

Filters

Titan features a flexible dual filter design that can be routed in series or parallel.

(The number of poles signifies the amount of damping above (lowpass) or below (highpass) the freqeuncy adjusted by the Cut control. Each pole represents 6 dB / octave.)

- Filter 1 offers the following modes:
  - LP4: 4-pole lowpass

    - BP2: 2-pole bandpass
    - HP2: 2-pole highpass
    - HP4: 4-pole highpass
    - HP6: 6-pole highpass
    - Pk: peak equalizer
    - DFF: differentiator; brightens the colour of the sound
- Filter 2 offers the following modes:
    - LP2: 2-pole lowpass
    - LP4: 4-pole bandpass
    - J4: 4-pole lowpass with unique resonance control
    - V4: cascaded analog style lowpass with feedback
    - <FB> Sets the delay time of the <V4> filter mode: There is a feedback with very short delay times to saturate the signal and emulate an analog filter.
    - HP4: 4-pole highpass
    - Pk: peak equalizer
    - HSF: hi-shelf equalizer
- <Cut> Adjusts the filter cutoff frequency.
- <Res> Sets the resonance of the filter.
- <-B+> Sets the amount of peak equalization in <Pk> filter mode; in all other modes it has no effect.
- <Sat> Switches the Saturation on or off.
- <Amount> Sets the amount of saturation, i. e. the amount of over-amplification before the signal is saturated. <SAT>

The following controls effect both filters:

- <Mode> When on, the filters are routed in parallel. When off, the filters are routed in series: Filter 1 -> Filter 2.

- <Mod> When on, the master LFO (LFO1) fades between the two filters. Only active in parallel mode. When active, <F1/F2> has no effect.
- <F1/F2> Fades between filter1 and 2. Only active in parallel mode.
- <Inv> Inverts the phase of filter1 only. Only active in parallel mode.

Modulators

The modulator section contains three envelope generators, three lfos and two event tables. All of them are used to modify certain parameters of the audio generating sections, e. g. the pitch to apply a vibrato, the amplitude for tremolo effects and smooth fade in and out of the sound. Which modulation source is used at which destination is be controlled in the matrix section which can be edited in the instrument view. See the Modulation Matrix section below for more information.

- **Envelope EG1**: The first envelope consists of a standard ADSR-envelope with an additional minimal hold time control and smoothing. Turn the EG1 switch in the Gate Router section on to control the peak value of the envelope by the velocity of the current midi note; if that control is off, the peak is independent from the current velocity.
  - <A, D, S, R> Sets the attack, decay, sustain, and release portion of the envelope.
  - <Hold> Switches hold mode on or off. If on, the envelope is triggered by a normal midi gate event, but then it is independant of it, i. e. the gate is open according to the control on the left, then it closes and starts the release period.
  - <A, D, S, R> Sets the attack, decay, sustain, and release portion of the envelope.
  - <H> Sets the hold time, i. e. the time in which the gate is open after a triggering midi gate signal. It ranges from 1 up to 10000 milliseconds in logarithmic steps. If the Hold switch at the right is off, this control shows no effect.
  - <Smth> Activates the additional smoothing of the envelope
  - <Amt> Sets the amount of smoothing. Turn to the right for more effect.
- **Envelope EG2**: The second envelope consists of an ADBDSR-envelope with additional smoothing. Turn the EG2 switch in the Gate Router section on to control the peak value of the envelope by the velocity of the current midi note; if that control is off, the peak is independent from the current velocity. See EG1 above for information on the specific parameters.

- **Envelope EG3**: The third envelope consists of an ADBDSR-envelope with additional smoothing. The breakpoint level and the second decay time (marked by an asterisk *) can be modulated. Turn the EG2 switch in the Gate Router section on to control the peak value of the envelope by the velocity of the current midi note; if that control is off, the peak is independent from the current velocity. See EG1 above for information on the specific parameters.
- **LFO 1** supports the following waveforms:
  - <Sin> Sine waveform.
  - <Tri> Triangle waveform.
  - <Pls> Pulse waveform.
  - <Rnd> Random waveform
  - <S&H> Activates the sample and hold generator that, according to the adjusted frequency, grabs a random audio value and holds it until the next sampling of the audio signal. An oscillator with random steps is used. If this switch is on, the waveform selection has no effect.
  - <Snc> Switches synchronization of the lfo to the midi gate on or off. If off, the phase control on the left shows no effect.
  - <Phs> Sets the phase of the lfo when a re-triggering synchronization event arrives. 0 represents a phase of 0 degrees, 1 a phase shift of 180 degrees. If <Snc> is off, this control has no effect.
  - <Freq> Sets the freqeuncy of the lfo, ranging form approx. 0.08 Hz to approx 14.5 Hz in logarithmic steps.
  - <Ampl> Sets the amplitude of the lfo, i. e. the max. output value reached during oscillation.
  - <Width> Sets the pulse width of the lfo. If in Rnd or Sample and Hold mode, this control has no effect.
  - <EG> Switches the internal envelope generator on or off. If on, the lfo amplitude is shaped by an AR-envelope that is triggered by the midi gate. In Sample and Hold mode, this envelope has no effect.

- <A> Sets the attack time of the internal envelope generator, ranging from 1 up to 10000 milliseconds in logarithmic steps.
- <R> Sets the release time of the internal envelope generator, ranging from 1 up to 10000 milliseconds in logarithmic steps.
- <Psv> If on, only positve values of the lfo are passed through. All others are clipped to 0.

- **LFO2** has less features than LFO1. See LFO1 for explanation of specific controls.
- **LFO3** is the simplest of all three LFOs. It can generate a random Sample and Hold signal or a normal triangular waveform. See LFO1 for explanation of specific controls.
- **Event Table T1**: This event table table contains a data module whose information can be used in three modes (see below). 128 different patterns can be stored in the module; however, if an existing one is changed, the sound of all snapshots using this pattern will change as the data module is independant of the snapshot control.
  - <Env>In Envelope mode the data table is read out from the left to the right once with the speed adjusted by the Dec knob, triggered by a midi gate on signal. If T1/L1 in the Gate Router section is on, the starting point of the envelope is shifted the more to the right the higher the velocity of the gate signal.
  - <Lfo*> In LFO mode the data table is read out from the left to the right again and again with the freqeuncy adjusted by the Freq knob and ist fine control below. Note that the speed of the table readout can be modulated.
  - <Key> In Key mode one value is read out of the table according to the midi pitch, i. e. low pitches select a value at the left, high pitches at the right, and held until a new pitch arrives.
  - <Snc> Switches synchronization to the midi gate on or off if the LFO is in LFO mode.
  - <Freq> Sets the oscillation frequency if the tabel is in LFO mode. This is the coarse control with steps of 0.1 Hz.
  - <Fine> Sets the oscillation frequency if the table is in LFO mode. This is the fine control with steps of 0.01 Hz.
  - <Dec> Sets the decay time of the envelope mode, i. e. the time that is used to read out the data table from the left to the right.

- **Event Table T2**: See above for general information on Titan's event tables. Table T2 is simpler than T1 and offers two modes:
  - <Key> In Key mode one value is read out of the table according to the midi pitch, i. e. low pitches select avlue at the left, high pitches at the right, and held until a new pitch arrives. In this mode, the table is a pitch remapper or microtuning editor.
  - <Vel> In Velocity mode one value is read out of the table according to the actual midi velocity. In this mode, the table is a velocity shaper.

MIDI Pitch Router

- Routes the incoming midi pitch events to pre-wired, musically useful destinations.
- <03.P> If on, the midi pitch of oscillator 3 is multiplied by the value of <amt1>
- <02.P> If on, the midi pitch of oscillator 2 is multiplied by the value of <amt2>
- <01.P> If on, the midi pitch of oscillator 1 is multiplied by the value of <amt3>
- <F.Res> Selects the mode of filter resonance modulation by the midi pitch.
  - <off> resonance is not modulated by midi pitch
  - <A> Weak (Decreases the resonance of both filters at pitches between 74 and 100 by up to 15%.)
  - <B> Moderate (Decreases the resonance of both filters at picthes between 84 and 90 by up to 20%)
- <F.cut> Activates keytracking of both filters, i. e. the frequency of the filters is connected to the midi pitch.
- <Amt> Adjusts the amount of keytracking. Turn left for no keytracking, to mid position for exact keytracking, i. e. the filter freqeuncy is completely controlled by the midi pitch, to the right for overdrive effects that produce very high filter freqeuncies for high notes.
- <Split> Sets the center frequency of the keytracking algorithm. If the knob to the left is turned to the right, all notes above this center freqeuncy get much higher filter freqeuncies, all notes below much lower filter frequencies.
- <EG3.D2> Selects the mode of modulation of the second decay time of EG3 by the midi pitch.
  - <off> EG3 decay 2 is not modulated by midi pitch

- <A> Strong (Shortens the time at pitches above 60 by up to 70%.)
- <B> Moderate (Shortens the time at pitches between 65 and 88 by up to 50%.)

MIDI Velocity Router

Titan has many musically useful velocity modulations hardwired.

- <03.P> If on, the midi pitch of oscillator 3 is multiplied by the value of <amt1> and the velocity.
- <02.P> If on, the midi pitch of oscillator 2 is multiplied by the value of <amt2> and the velocity.
- <01.P> If on, the midi pitch of oscillator 1 is multiplied by the value of <amt3> and the velocity.
- <F1.C> Selects the mode of frequency modulation of Filter 1 by the midi velocity
  - <off> Filter 1 is not modulated by midi velocity
  - <A> Moderate (High velocities increase the filter freqeuncy by up to 40 semitones.)
  - <B> Strong (High velocities increase the filter frequency by up to 120 semitones.)
  - <C> Weak (High velocities increase the filter freqeuncy by up to 12 semitones.)
- <F2.C> Selects the mode of frequency modulation of Filter 2 by the midi velocity
  - <off> Filter 2 is not modulated by midi velocity
  - <A> Moderate (High velocities increase the filter freqeuncy by up to 40 semitones.)
  - <B> Strong (High velocities increase the filter frequency by up to 120 semitones.)
  - <C> Weak (High velocities increase the filter freqeuncy by up to 12 semitones.)
- <FM> Sets the mod of FM amount modulation by the midi velocity.
  - <off> FM is not modulated by midi velocity
  - <A> Moderate
  - <B> Strong
- <Brp> Sets the mode of modulation of the breakpoint of EG3 by the midi velocity.
  - <off> BRP is not modulated by midi velocity

- <A> Moderate (High velocity values increase the breakpoint level by up to 20%.)
- <B> Strong (High velocity values increase the breakpoint level by up to 60%.)

- <D2> Sets the mode of modulation of the second decay time of EG3 by the midi velocity.
  - <off> second decay time is not modulated by midi velocity
  - <A> Moderate (High velocity values increase the decay time by up to 100 milliseconds.)
  - <B> Strong (High velocity values increase the decay time by up to 100 seconds.)

Modulation Matrix

The modulation matrix is the Instrument view of Titan. By double clicking on Titan s main titlebar, the modulation matrix will pop into view. In the modulation matrix, 6 modulation sources (envelopes, lfos, etc) can be routed to 11 destination (pitch, filter, etc). The sources are:

- One of the 3 envelopes EG1 or EG2 or EG3
- LFO1
- LFO2 or LFO3
- Table 1
- Table 2
- The pitchbend or the modulation wheel

The destinations are:

- The global pitch.
- The pitch of osc2.
- The pitch of osc1.
- The frequency of the first filter.
- The frequency of the second filter.
- The pulse width of OSC1, OSC2, OSC3.
- The resonance of the first and / or the second filter. A maximum threshold can be adjusted to prevent self resonance.

- The frequency modulation of OSC2.
- The frequency modulation of OSC1
- The frequency of LFO1, LFO2, LFO.
- The main amplitude.

Each source can be switched on or off and, when on, the amount of modulation can be adjusted by two controls, one for coarse, and one for fine adjustment. The controls used to modify the signal of the different sources are explained at the end of this documentation.

Loudness Correction

After filtering, the signal is send to the Loudness correction. This section amplifies certain freqeuncy areas while attenuating others to enhance the sound due to its function as bass, lead etc.

- <On> Switches the loudness correction on or off.
- <Split> Sets the breakpoint of the loudness correction. Notes below this breakpoint are amplified, notes above this breakpoint are attenuated.
- <Amt> Sets the amount of loudness.

Effects

- **Dimension Expander**: This effect generates a stereo impression out of the mono input signal. Its controls are:
  - <On> Switches the effect on or off.
  - <Sprd> Sets the maximum stereo spread. Try different settings as the effect depends a lot on the type of input signal.
  - <Delay> Sets the max. delay time. Technically speaking, the effect is based on a four delays with different delay times whose output is mixed with slowly changing ratio - just try different settings.
- **Chorus/Flanger, Delay, Reverb, EQ**: Note that the settings of these standard effects are stored with each snapshop. For more information about the specific settings of each knob, please check the online hints. The delay time of the delay is set in sixteenth notes, synced to the system clock.



REAKTOR 6

BUILDING IN PRIMARY

The information in this document is subject to change without notice and does not represent a commitment on the part of Native Instruments GmbH. The software described by this document is subject to a License Agreement and may not be copied to other media. No part of this publication may be copied, reproduced or otherwise transmitted or recorded, for any purpose, without prior written permission by Native Instruments GmbH, hereinafter referred to as Native Instruments.

“Native Instruments”, “NI” and associated logos are (registered) trademarks of Native Instruments GmbH.

Mac, Mac OS, GarageBand, Logic, iTunes and iPod are registered trademarks of Apple Inc., registered in the U.S. and other countries.
Windows, Windows Vista and DirectSound are registered trademarks of Microsoft Corporation in the United States and/or other countries.
All other trade marks are the property of their respective owners and use of them does not imply any affiliation with or endorsement by them.

Document authored by: Adam Hanley, Jan Ola Korte

Software version: 6.3 (04/2019)

NATIVE INSTRUMENTS GmbH
Schlesische Str. 29-30
D-10997 Berlin
Germany
www.native-instruments.de

NATIVE INSTRUMENTS North America, Inc.
6725 Sunset Boulevard
5th Floor
Los Angeles, CA 90028
USA
www.native-instruments.com

NATIVE INSTRUMENTS K.K.
YO Building 3F
Jingumae 6-7-15, Shibuya-ku,
Tokyo 150-0001
Japan
www.native-instruments.co.jp

NATIVE INSTRUMENTS UK Limited
18 Phipp Street
London EC2A 4NU
UK
www.native-instruments.co.uk

NATIVE INSTRUMENTS FRANCE SARL
113 Rue Saint-Maur
75011 Paris
France

www.native-instruments.com

SHENZHEN NATIVE INSTRUMENTS COMPANY Limited
5F, Shenzhen Zimao Center

111 Taizi Road, Nanshan District, Shenzhen, Guangdong

China

www.native-instruments.com

NATIVE INSTRUMENTS

© NATIVE INSTRUMENTS GmbH, 2019. All rights reserved.

Table of Contents

1 Welcome to Building in Primary 15
1.1 The REAKTOR 6 Documentation 16
1.2 Manual Conventions 18
2 Introductory Topics 19
2.1 Browsing and Loading Modules and Macros 19
2.1.1 Using the Context Menu 19
2.1.2 Using the Searchbox 20
2.2 Connecting Modules and Macros 21
2.2.1 Creating Ports with Wires 22
2.3 Event Signals and Audio Signals 22
2.4 Mono and Poly Modes 24
2.4.1 Setting the Number of Polyphonic Voices 25
2.4.2 Setting the Mono State of a Module 26
2.4.3 Polyphony and Macros 27
2.4.4 Voice Combiners 27
2.5 Debugging 28
2.5.1 Wire Debugging 29
2.5.2 CPU Usage 30
2.5.3 Module Sorting 31
2.5.4 Event Initialization Order 32
3 Subtractive Synthesizer 34
3.1 Overview 34
3.1.1 Previous Knowledge 34
3.1.2 The Theory 34
3.2 Tutorial 35

3.2.1 Setting Up 35
3.2.2 Building a MIDI Controlled Oscillator 37
3.2.3 Adding the Filter 43
3.2.4 Modulating the Filter 46
3.2.5 Adding Polyphony 49
3.3 Going Further 52
3.3.1 Related Factory Content 53
4 Echo Effect Macro 55
4.1 Overview 55
4.1.1 Previous Knowledge 55
4.1.2 The Theory 55
4.2 Tutorial 56
4.2.1 Setting Up 56
4.2.2 Building the Simple Echo Effect 60
4.2.3 Adding Feedback 65
4.2.4 Adding Tempo Sync 67
4.2.5 Saving and Loading the Macro 69
4.3 Going Further 71
4.3.1 Related Factory Content 73
5 Basic Step Sequencer 75
5.1 Overview 75
5.1.1 Previous Knowledge 75
5.1.2 The Theory 75
5.2 Tutorial 76
5.2.1 Setting Up 76
5.2.2 Building the Basic Sequencer 77

5.2.3 Adding Tempo Sync ... 83
5.3 Going Further ... 87
5.3.1 Related Factory Content ... 87
6 Advanced Step Sequencer ... 89
6.1 Overview ... 89
6.1.1 Previous Knowledge ... 89
6.1.2 The Theory ... 89
6.2 Tutorial ... 90
6.2.1 Setting Up ... 90
6.2.2 Building the Sequencer Interface ... 90
6.2.3 Storing Values in a Snapshot ... 100
6.2.4 Adding Playback ... 102
6.3 Going Further ... 105
6.3.1 Related Factory Content ... 106
7 Additive Synthesizer ... 108
7.1 Overview ... 108
7.1.1 Previous Knowledge ... 108
7.1.2 Theory ... 109
7.2 Tutorial ... 109
7.2.1 Setting Up ... 109
7.2.2 The Sine Bank Oscillator ... 111
7.2.3 Building the Synthesizer ... 112
7.2.4 Adding Real-time Updates ... 116
7.2.5 Adjusting the Partial Ratio Spread ... 118
7.3 Going Further ... 126
7.3.1 Related Factory Content ... 126

8 Drag and Drop Sampler 128
8.1 Overview 128
8.1.1 Previous Knowledge 128
8.1.2 Theory 128
8.2 Tutorial 129
8.2.1 The Drag and Drop Area 129
8.2.2 The Waveform Display 133
8.2.3 Playback 136
8.3 Going Further 139
9 A Quick Guide to Initialization 140
10 Creating and Customizing Interfaces 143
10.1 Overview of Panel Elements 143
10.2 Customizing the Header 145
10.3 A/B Views 147
10.3.1 Examples in the Factory Library 148
10.4 Changing the Color Scheme 149
10.4.1 Editing Colors 150
10.4.2 Saving a Custom Color Scheme 151
10.5 Skinning with Custom Images 152
10.5.1 Image Formatting 152
10.5.2 Importing an Image 153
10.5.3 The Image Properties Window 154
10.6 The Panel Grid and Borders 156
10.7 Layering Panel Elements 158
10.8 Stacked Macros 161
10.8.1 Using Stacked Macros 162

10.9 Grouping Panel Elements 165
10.10 Panel Ports 167
11 Automation 171
11.1 Customizing Automation Properties 171
11.2 The Automation Module 173
11.2.1 Sending and Receiving Automation Values 173
11.2.2 Dynamically Changing Automation Names 174
11.2.3 Custom Control Example 176
12 KOMPLETE KONTROL and MASCHINE 2 Integration 180
12.1 Snapshot Visibility 180
12.1.1 Tagging Snapshots 180
12.2 Defining the Bank Name 184
12.3 Panelsets 186
12.3.1 Default and Additional Views 186
12.4 Hardware Control 187
12.4.1 Message Types 189
12.4.2 Available Note Colors 190
12.4.3 Related Macros 191
12.5 Native Map 192
12.6 Adding Ensembles to the Library 193
13 Internal Connection Protocol 197
13.1 Connecting Two Panel Elements via the IC Protocol 197
13.2 IC Send/Receive Modules 199
14 Optimization Techniques 201
14.1 Reduce the Number of Voices 201
14.1.1 Automatic Voice Reduction 201

14.1.2 Use Monophonic Processing 202
14.2 Use Events Rather than Audio signals 202
14.3 Eliminate Superfluous Events 202
14.3.1 Filtering Duplicate Values 202
14.3.2 Triggering Values 203
14.4 Use the Display Clock 203
14.5 Reduce the Control Rate 204
14.6 Use Multipliers Instead of Dividers 205
14.7 Avoid unnecessary use of Send/Receive Modules 205
14.8 Use the Modulo Module instead of Wrap options 206
15 An Introduction to Core 207
16 Table Framework 209
16.1 Tables and Samples 209
16.2 Table Reference Wires 209
16.3 Table Reference Ports 210
16.4 The Table List Module 211
16.4.1 Adding a Table to the Table List 211
16.4.2 Removing a Table from the Table List 212
16.4.3 Selecting a Table Reference to Output 212
16.5 The Sample Map Module 213
16.6 The Table Info Module 213
16.7 The Table Location 214
16.8 Drag and Drop 214
16.8.1 The Drag and Drop Process 214
16.8.2 Dragging a Table Reference from a Mouse Area 215
16.9 Table References in Core 216

16.9.1 TableRef Input 216
16.9.2 Accessing Table References 217
16.9.3 Deleting Tables 218
17 Compiled Core Cell Code Cache 219
17.1 Default Cache Location 219
17.1.1 Changing the Compiled Core Cell Code Cache Location 219
17.2 Purging the Compiled Core Cell Code Cache 220
18 Module Reference 221
18.1 Math 222
18.1.1 Constant 222
18.1.2 Basic Modules 222
18.1.3 Modifiers 223
18.1.4 Mult/Add, a*b+c 224
18.1.5 Modulo 224
18.1.6 Quantize 224
18.1.7 Comparison 225
18.1.8 Logarithms (Converters) 226
18.1.9 Power and Roots 226
18.1.10 Trigonometric Functions 227
18.2 MIDI In 227
18.2.1 Pitch 228
18.2.2 Gate and Velocity 228
18.2.3 Controllers 229
18.2.4 Transport and Clock 230
18.2.5 Program Change 230
18.2.6 Channel Message 231

18.3 MIDI Out 231
18.3.1 Note Pitch and Gate 231
18.3.2 Controllers 232
18.3.3 Transport and Clock 233
18.3.4 Channel Message 233
18.4 OSC 234
18.4.1 Send/Receive 234
18.4.2 Arrays 235
18.5 Panel 235
18.5.1 Basic Controls 235
18.5.2 Basic Displays 236
18.5.3 Pictures 237
18.5.4 Text 237
18.5.5 List 238
18.5.6 Switch 239
18.5.7 Receive 239
18.5.8 XY 240
18.5.9 Mouse Area 241
18.5.10 Advanced Displays 242
18.5.11 Scope 242
18.5.12 Stacked Macro 243
18.6 Signal Path 243
18.6.1 Selectors 244
18.6.2 Distributors 244
18.6.3 Mixers 245
18.7 Oscillators 245

18.7.1 Basic Audio Oscillators 246
18.7.2 Variants 247
18.7.3 Multi-step 249
18.7.4 Noise 250
18.7.5 Utility 250
18.7.6 Additive 251
18.8 Samplers 252
18.8.1 Basic 252
18.8.2 Granular 253
18.8.3 Beat Loop 254
18.8.4 Lookup 254
18.9 Table Framework 255
18.9.1 Table List 255
18.9.2 Table Info 255
18.9.3 Sample Map 256
18.10 Sequencer 256
18.10.1 Basic 257
18.10.2 Multiplex 16 258
18.11 LFO, Envelope 258
18.11.1 LFO 258
18.11.2 Envelopes 259
18.11.3 Multi-ramp 261
18.12 Filters 261
18.12.1 Basic 262
18.12.2 Modeled 263
18.12.3 Modal Bank 264

18.12.4 Equalizers 264
18.12.5 Differentiator, Integrator 265
18.13 Delay 266
18.13.1 Basic 266
18.13.2 Diffuse 267
18.13.3 Granular 267
18.14 Audio Modifier 268
18.14.1 Saturators and Distortions 268
18.14.2 Shapers 269
18.14.3 Modifiers 270
18.14.4 Audio Table 271
18.15 Event Processing 271
18.15.1 Counters 272
18.15.2 Random 272
18.15.3 Frequency Divider 272
18.15.4 Control Shapers 273
18.15.5 Logic 274
18.15.6 Iteration 274
18.15.7 Signal Flow 275
18.15.8 Routers 276
18.15.9 Timing 276
18.15.10 Event Table 277
18.16 Auxiliary 278
18.16.1 Audio to Event 278
18.16.2 Voice 278
18.16.3 Voice Combiners 279

18.16.4 Smoothers 279
18.16.5 Snapshot 280
18.16.6 Snap Values 281
18.16.7 System 281
18.16.8 Instrument Properties 282
18.16.9 Spread 283
18.16.10 Automation 283
18.16.11 Set Random 283
18.16.12 Tapedeck 284
18.16.13 Comment 284
18.17 Terminal 285
18.17.1 Ports 285
18.17.2 Send/Receive 285
18.17.3 IC Connections 286
18.17.4 Hardware Control 286
Index 287

1 Welcome to Building in Primary

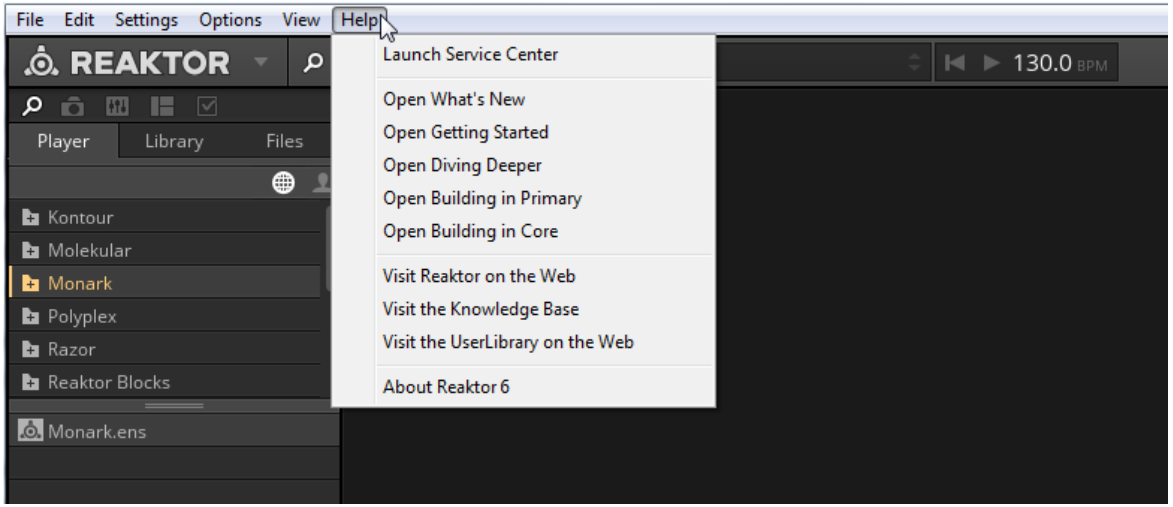
Welcome to Building in Primary, a tutorial for building in REAKTOR's Primary level. It describes the functions in REAKTOR that relate to building custom instruments, effects, or sequencers, enabling you to modify existing Ensembles or create your own.

This document assumes you are already familiar with the REAKTOR interface, as well as loading and playing with Ensembles. At the least, you should have read the chapters in the REAKTOR 6 Diving Deeper document regarding navigation in the Structure View.

This document begins with an introduction to basic functions and concepts, followed with a collection of tutorials. These tutorials introduce building concepts through common examples.

The final sections are guides to additional building features available in REAKTOR, as well as some general tutorials for general building tasks (like skinning controls, and preparing Ensembles for the KOMPLETE PLAYER Browser).

1.1 The REAKTOR 6 Documentation



The REAKTOR documentation is accessible from the Help menu

The documentation for REAKTOR 6 is divided into five separate documents, guiding you from loading and playing pre-built Ensembles to building your own Instruments.

- **REAKTOR 6 What Is New** is written for users who are already familiar with previous versions of REAKTOR and only describes the latest features in brief.
- **REAKTOR 6 Getting Started** is for new users. It is the only document needed for users who intend to use REAKTOR for loading and playing pre-built REAKTOR instruments and effects.
- **REAKTOR 6 Diving Deeper** expands on the concepts introduced in the Getting Started document. It provides more detail on subjects like Snapshots (REAKTOR's preset system), and introduces advanced topics like OSC control and combining multiple Instruments in one Ensemble.

- **REAKTOR 6 Building in Primary** shows you how to build your own Instruments in REAKTOR's Primary level. It focuses on a series of tutorials that guide you through building your first synthesizers, effects, and sequencers.
- **REAKTOR 6 Building in Core** describes the Core level of REAKTOR with its low-level building features, which can be used for implementing custom DSP algorithms. It includes reference information about the Core Macro Library, an comprehensive collection of DSP building blocks.

With the exception of the What Is New document, each of the documents listed above builds on the knowledge of the previous documents. While it is not necessary to read all of every document, some of the more advanced documents, like Building in Primary, assume knowledge from the previous documents.

1.2 Manual Conventions

This section introduces you to the signage and text highlighting used in this manual.

- Text appearing in (drop-down) menus (such as Open..., Save as... etc.) and paths to locations on your hard disk or other storage devices is printed in italics.
- Text appearing elsewhere (labels of buttons, controls, text next to checkboxes etc.) is printed in blue. Whenever you see this formatting applied, you will find the same text appearing somewhere on the screen.
- Important names and concepts are printed in **bold**.
- References to keys on your computer's keyboard you'll find put in square brackets (e.g., "Press [Shift] + [Enter]").

1. Single instructions are introduced by this play button type arrow.

⇨ Results of actions are introduced by this smaller arrow.

An indented, gray paragraph contains additional, contextual information.

Furthermore, this manual uses particular formatting to point out special facts and to warn you of potential issues. The icons introducing these notes let you see what kind of information is to be expected:



The speech bubble icon indicates a useful tip that may often help you to solve a task more efficiently.



The exclamation mark icon highlights important information that is essential for the given context.



The red cross icon warns you of serious issues and potential risks that require your full attention.

2 Introductory Topics

2.1 Browsing and Loading Modules and Macros

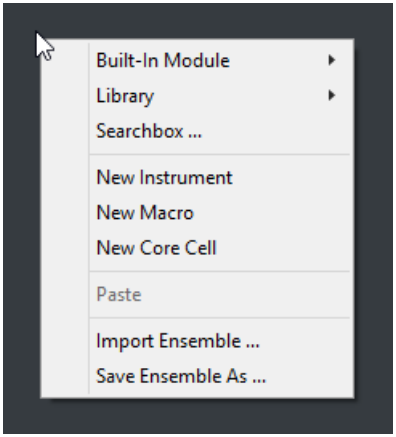
The Primary level of REAKTOR uses two types of building-blocks to create structures, these are:

- **Modules**: These are the lowest level of building-block in the REAKTOR Primary level.
- **Macros**: These are containers used to group together sub-structures of other Modules and Macros.

There are a few different ways to browse for and load Modules and Macros.

2.1.1 Using the Context Menu

Clicking the right mouse button (or [Ctrl] + click on OSX) on any empty area in the structure view will open the context menu.



The Context Menu.

For the purposes of browsing and loading Modules and Macros, you will only need to use the following entries:

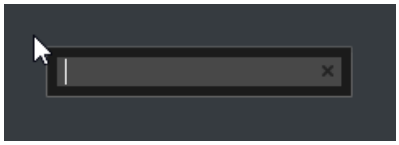
- Built-In Module: Opens a sub-menu from which you can browse for Modules. Once you have located the Module you wish to load, click on its name in the menu.
- Library: Opens a sub-menu from which you can browse for and load Macros, and Instruments.
- Open Searchbox: Opens the Searchbox, which is covered in section ↑2.1.2, Using the Searchbox ↑2.1.2, Using the Searchbox.
- New Macro: Creates a new empty Macro.

The Context Menu is a useful thing to use when you are unsure of the name of the Module or Macro you wish to load.

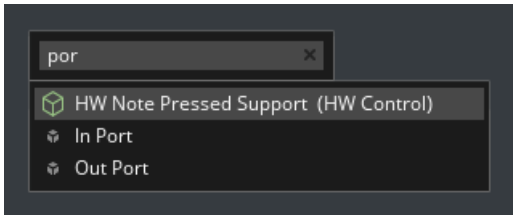
2.1.2 Using the Searchbox

When you are familiar with the Modules and Macros available in REAKTOR, you may find it faster to use the Searchbox to browse for and load content.

1. The fastest way to open the Searchbox is to press the [Enter] key when in the structure view.
2. Alternatively, you can open the Searchbox by selecting Open Searchbox in the Context Menu.
3. With the Searchbox now displayed, you can search for a Module or Macro by typing its name.



4. The Searchbox will autocomplete your entry.



5. When the Module/Macro you wish to load appears, you can load it by clicking on its name.
6. If the Module/Macro you want is the first or only entry that appears, you can load it by pressing [Enter].
7. You can also select from the list by using the arrow keys, before clicking [Enter].

When you next open the Searchbox, it will display a list of your most recently loaded Modules and Macros. You can then load any of the Modules/Macros in this list as you would normally.

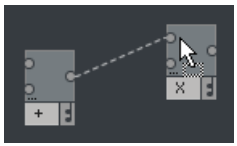
You can also search for mathematical functions by typing their symbol rather than their full name. For example, typing "*" will load the Multiply Module and typing "+" will load the Add Module.

You can also search for mathematical functions by typing their symbol rather than their full name. For example, typing "*" will load the Multiply Module and typing "+" will load the Add Module.

2.2 Connecting Modules and Macros

REAKTOR uses wires to connect the input and output Ports of the Modules and Macros.

1. Connecting two Ports is as simple as clicking on one port, dragging a wire to another Port, and then releasing the mouse to create the connection.



However, there are some rules to making connections:

- Wires can only be used to create a connection between an output Port and an input Port.
- An input can only ever accept one connection at a time, but an output can make multiple connections.
- Event outputs can connect to Audio inputs, but not vice versa.

2.2.1 Creating Ports with Wires

Some Modules and all Macros will allow you to make new ports by holding the [Ctrl]/[Cmd] key while dragging a wire.



Modules that can accept additional ports are identified by the ... below their lowest port.

2.3 Event Signals and Audio Signals

The Primary level of REAKTOR uses two main signal types:

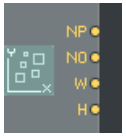
- **Audio**: Audio signals run at the audio rate (typically 44.1kHz or 48kHz). Anything that produces audio (like an oscillator), processes audio (e.g. a filter), or requires a high level of fidelity (e.g. an envelope), should use Audio signals.
- **Event**: Event signals run at a slower rate, known as the Control Rate, the default value of which is 400Hz. Since they run at a slower rate, event signals require less CPU, but have less fidelity. This makes them ideal for signals which we do not directly hear. For example, the pitch input for an oscillator uses Event signals, as our ears cannot perceive changes in pitch at rates faster than the Control Rate. Event signals are also useful when creating displays or sequencers.

Certain ports are predefined to send or receive one kind of signal. These ports are color-coded:

- Light Grey ports are Audio ports.

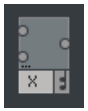


- Yellow ports are Event ports.



Some Modules can operate at either the Audio Rate or the Control Rate. Most Math Modules can operate at either rate.

- Dark Grey ports show that the Module can operate at either the Audio Rate or the Control Rate.



- The processing rate of these Modules depends on how the ports are connected:
  - If all ports are connected to other Event ports, then the Module will run at the Control Rate.
  - If any of the ports are connected to an Audio port, then the Module will run at the Audio Rate.

There are also Modules with hollow or black Ports. This shows that the Module can accept Event signals or Table References.



Table References will be explored in detail in the tutorial in section ↑8, Drag and Drop Sampler ↑8, Drag and Drop Sampler.

2.4 Mono and Poly Modes

Many Modules have two modes of operation: Monophonic (mono) and Polyphonic (poly).

These modes are a common stumbling block for users who are new to REAKTOR, as their function is not always obvious.

- Monophonic Modules will only process a single signal.
- Polyphonic Modules can process a separate signal for each voice.

The easiest way to think of these concepts is how you would think of synthesizers:

- Monophonic synthesizers can only produce a single note at a time.
- Polyphonic synthesizers can produce several notes at once.

The mono/poly modes in REAKTOR work much in the same way.

Another way to think about it is like this: a monophonic Module is a single item, whereas a polyphonic Module is like a simplified representation of several duplicate Modules, one for each voice.

You can see if a Module is set to mono or poly mode by looking at the icon to the bottom right:

- If the icon is of a single note, then the Module is monophonic.

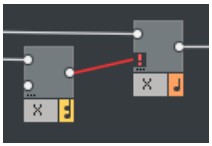


- If the icon is of two notes, one on top of the other, then the Module is polyphonic.



It is possible to connect the output of a monophonic Module to the input of a polyphonic Module, but it is not possible to connect the output of a polyphonic Module directly to monophonic Module.

Connecting a polyphonic Module to a monophonic Module will produce an error and the connection will not work. A connection like this will be colored red and an "!" will mark the input of the monophonic Module.



A Polyphonic Module Connected to a Monophonic Module

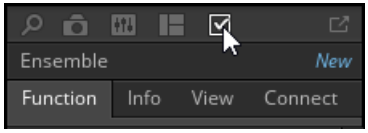
This kind of warning is reserved exclusively for this kind of error.

2.4.1 Setting the Number of Polyphonic Voices

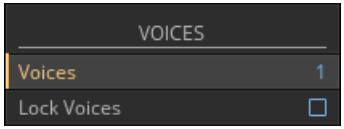
By default, the Ensemble that is created when you select File -> New Ensemble has its number of voices set to 1. This effectively makes everything in the Ensemble monophonic, regardless of the individual mono/poly settings of the Modules. This is because a polyphonic module that only uses one voice is functionally the same as a monophonic Module.

For some cases, like creating a polyphonic synthesizer, you will need to change the number of voices.

1. Create a New Ensemble.
2. Select the Ensemble by clicking in an empty space, or on the Ensemble Header.
3. Open the Ensemble's Properties in the Side Pane.

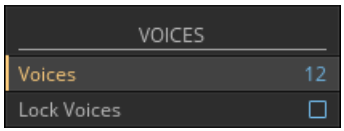


4. In the Function tab is an area called VOICES.
5. Locate the Voices parameter, which should be set to 1.



6. Click and drag on this number to change it.

7. Alternatively, you can also double-click on it and type in a specific number (in the range of 1 to 1024).



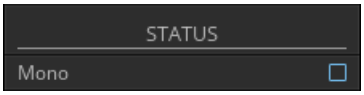
Lock Voices

The Lock Voices option below the Voices parameter disables the editing of the properties related to polyphony. If your Ensemble requires a specific number of voices to function correctly, checking this box will help to avoid accidental editing of the Voices parameter.

2.4.2 Setting the Mono State of a Module

Any Module that can be either monophonic or polyphonic will have an option in its properties for this setting.

The Mono setting is located in the STATUS section of the Function tab.



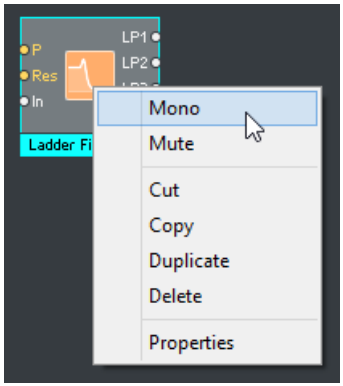
The Mono Option.

- Checking this box will set the Module to be monophonic.
- Unchecking this box will set the Module to be polyphonic.

It is also possible to set the mono/poly state of a Module without entering the properties:

1. Right-click on the Module to open a menu.

2. Select Mono to set the Module monophonic mode.



3. Re-opening the menu and unchecking the Mono option will set the Module to polyphonic mode.

2.4.3 Polyphony and Macros

Macros also have a Mono option in their menu; however the functionality is a little different.

A Macro is never truly monophonic or polyphonic, but checking the Mono option will set all Modules inside the Macro to be monophonic.

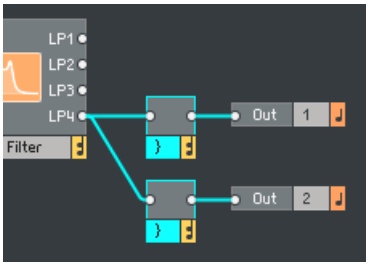
In other words, setting the mono/poly state of a Macro is a shortcut to changing the settings of all the Modules within the Macro, rather than being a state of the Macro itself.

However, it is then possible to enter the Macro and set a single Module to be polyphonic, without altering the mono/poly state of the Macro.

2.4.4 Voice Combiners

Voice Combiners are Modules that can be used to merge a polyphonic signal into a monophonic signal.

The outputs of REAKTOR are monophonic, since they can only output a single audio signal. Therefore, Voice Combiners are important to add before the final output of any Ensemble that uses polyphonic signals.



Audio Voice Combiners.

Voice Combiner Modules merge polyphonic signals into a monophonic signal.

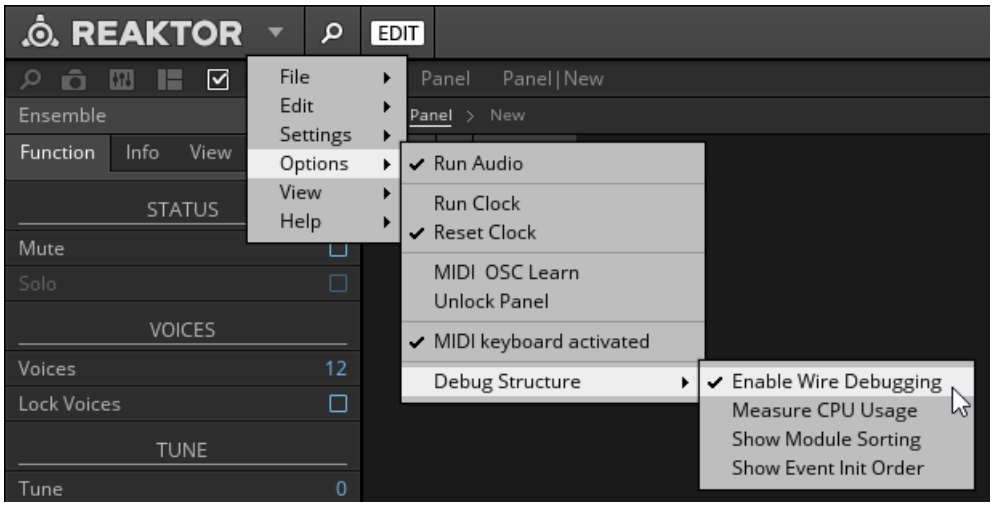


In some cases it may be necessary to use voice combiners before the Ensemble output ports, for example, if you are using effects like Delay or Reverb.

2.5 Debugging

Debugging is always an important part of development, and REAKTOR is no exception. As such, REAKTOR has a handful of built-in tools for debugging.

1. The debugging tools are located in the Options menu, under Debug Structure.



2.5.1 Wire Debugging

When debugging in REAKTOR, it will be common for you to want to see the value of a signal as it passes from Module to Module, and Wire Debugging is quick and easy way of doing this.

There are two ways of activating Wire Debugging:

1. In the structure view, click on the bug icon to the left of the breadcrumb navigation.



Or…

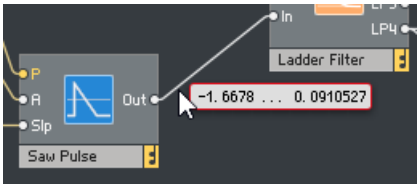
1. Enter the Options Menu.
2. Select Debug Structure to display the debugging tools.
3. Click on the Enable Wire Debugging option.

With Wire Debugging active, you can hover the mouse over any wire in the structure and see its current value.



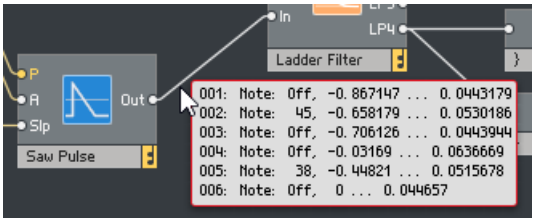
Wire Debugging.

If the value is changing faster than the display refresh rate (i.e. if it is an audio signal), then you will be shown the minimum and maximum values that pass through the wire, like this:



Debugging an Audio Signal.

If the wire is polyphonic, then you will be given a list of the values on each voice:



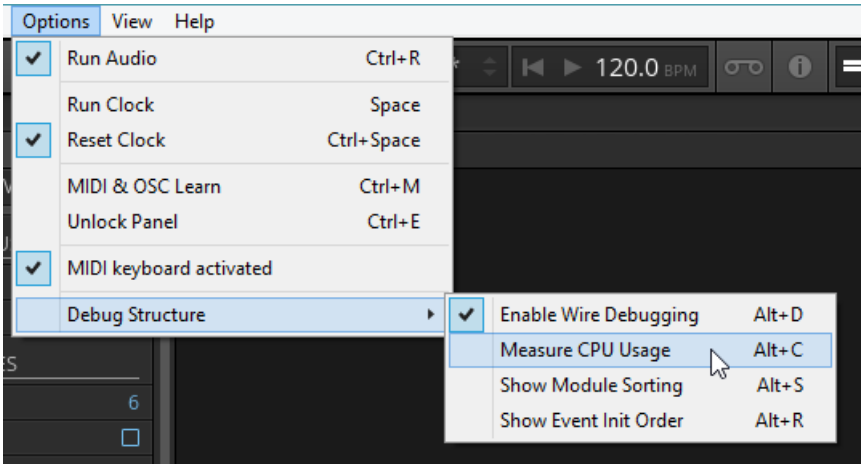
Debugging a Polyphonic Signal.

2.5.2 CPU Usage

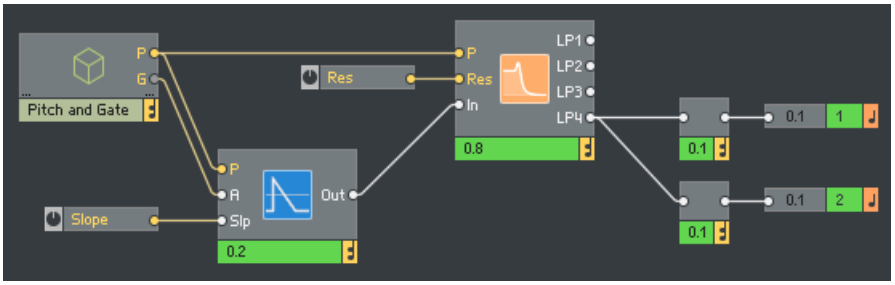
In REAKTOR, it is possible to measure the CPU usage of the individual elements in your structure. This can help you locate any areas that may be causing CPU spikes, or that could be optimized.

To toggle the Measure CPU Usage option on or off:

1. Enter the Options Menu.
2. Select Debug Structure to display the debugging tools.
3. Click on the Measure CPU Usage option.



With this option activated, you will be able to see numbers in place of the names of some of the Modules and Macros.



Measuring CPU Usage.

These numbers show the CPU usage of the Modules and Macros.

If a Module or Macro does not have a number, this means that its CPU usage is less than 0.1%

All audio outputs will be muted while the Measure CPU Usage option is active. They will return to normal when the option is deactivated.

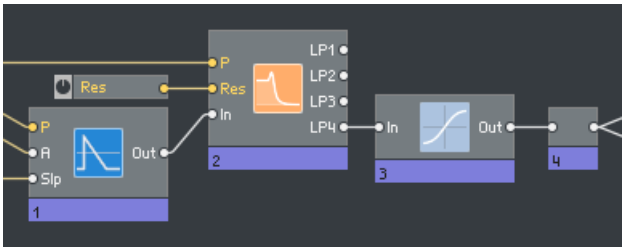
2.5.3 Module Sorting

In Primary Level when performing signal processing with Audio signals, the order of operations becomes important. This is particularly crucial if you have feedback in your Structure.

You can use the Module Sorting Debug Tool to see in which order the operations are performed on your Audio signal.

To toggle the Measure CPU Usage option on or off:

1. Enter the Options Menu.
2. Select Debug Structure to display the debugging tools.
3. Click on the Show Module Sorting option.



The Module Sorting of Audio Modules.

With this Debug Tool active, a purple display for each Module shows its place in the sequence in which the operations are performed on the Audio signal.

Note that only Audio Modules have Module Sorting numbers, Event Modules have no Module sorting number.

2.5.4 Event Initialization Order

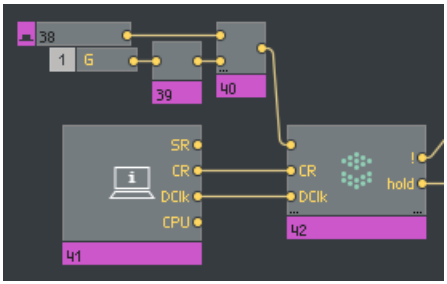
When you load an Ensemble, it goes through a process called Initialization. During this procedure, the Modules are initialized in a certain order and each only once.

At the moment of its Initialization, the Module looks at all values that have arrived at its input ports and then sends the appropriate values and/or Events from its output port(s). In more complex Structures, Event Initialization becomes a crucial part of your Structure. Some Modules require Events to arrive in a certain order at their input ports before functioning in the intended way.

To see the order in which Modules that process or send Events are initialized, use the Event Initialization Order Debug Tool.

To toggle the Event Initialization Order option on or off:

1. Enter the Options Menu.
2. Select Debug Structure to display the debugging tools.
3. Click on the Show Event Init Order option.



Modules displaying their Event Init Order.

With this Debug Tool active, a magenta display for each Module shows its place in the sequence in which Event functions are initialized.

3 Subtractive Synthesizer

3.1 Overview

REAKTOR is a visual programming environment, allowing you to build musical tools specific to your own needs. This tutorial will introduce the basic skills needed to build your own Ensembles from the components in the Macro Library using a subtractive synthesizer as an example.

3.1.1 Previous Knowledge

This tutorial assumes you have read all of the documentation up to this point and that you are familiar with REAKTOR's interface and basic functionality, including how to navigate in the Structure View.

The tutorial also assumes that you have set up your Audio and MIDI devices and are able to play REAKTOR Ensembles from the Factory Library.

3.1.2 The Theory

In its most basic form, a subtractive synthesizer is an oscillator routed through a filter. A harmonically rich waveform, like a saw wave, has its harmonics subtracted by the filter, hence the term subtractive synthesis.

However, even early synthesizer designers knew that a single oscillator and a single filter were not enough on their own and that modulation was required to change the sound over time. Amplifiers, LFOs (Low Frequency Oscillators) and Envelopes were added to give sound designers and musicians more flexibility.

In this tutorial, you will use the Macro Library to build a subtractive synthesizer, learning the basics of building in REAKTOR along the way.

3.2 Tutorial

3.2.1 Setting Up

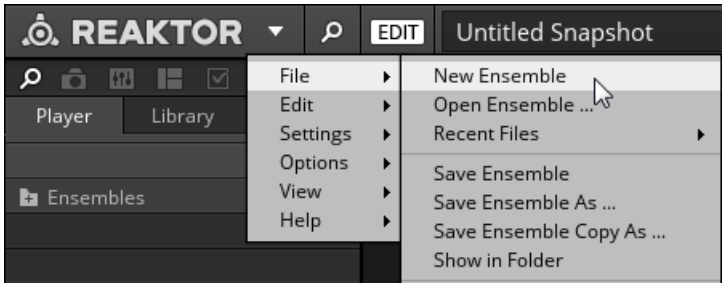
Before you begin building, you need to have REAKTOR running in Edit Mode.

1. If Edit Mode is not already enabled, activate it by clicking on the EDIT button in the toolbar.



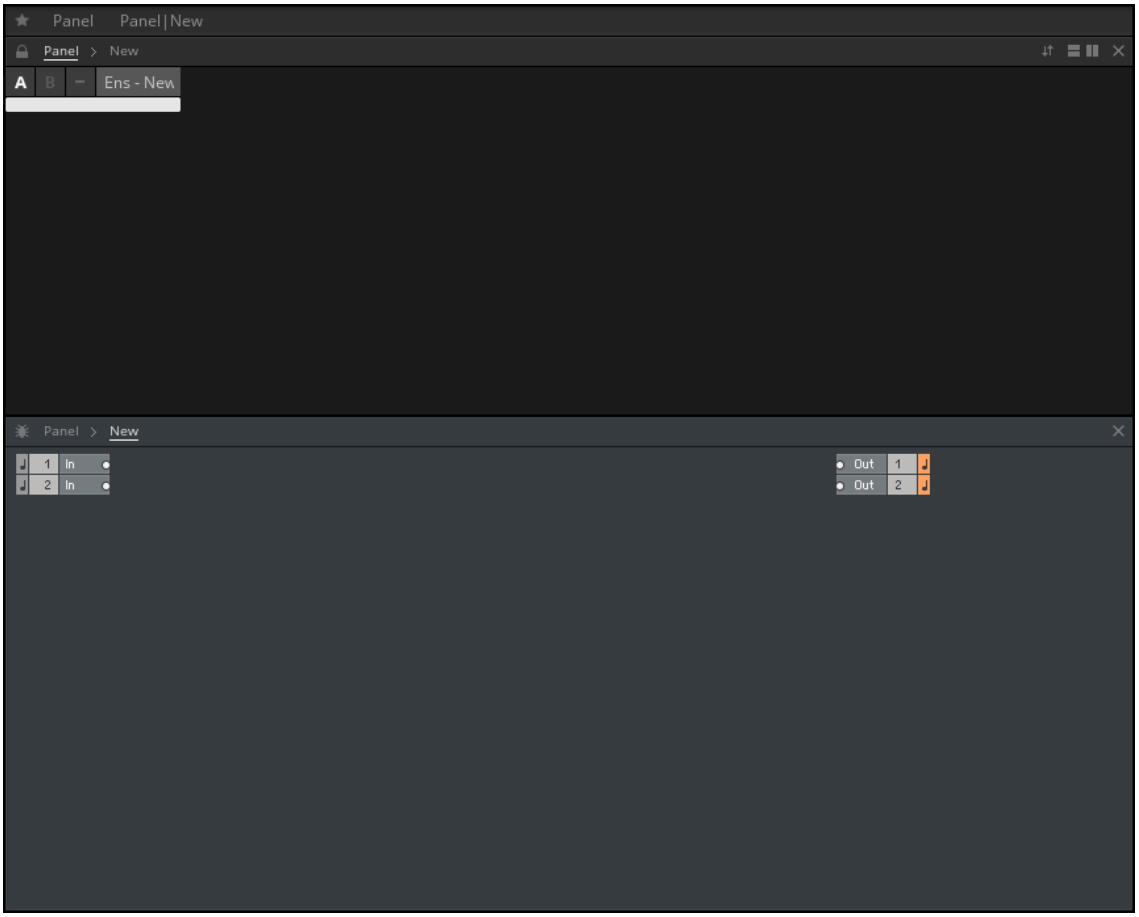
Next, you need to create a new empty Ensemble.

1. Create a new empty Ensemble by entering the File menu and selecting New Ensemble.



2. Alternatively, you can use the shortcut [Ctrl] + [N] ([Cmd] + [N] on OSX)

When you create a new Ensemble in Edit Mode, it loads with the screen already split horizontally, with the Panel View on top and the Structure View below.



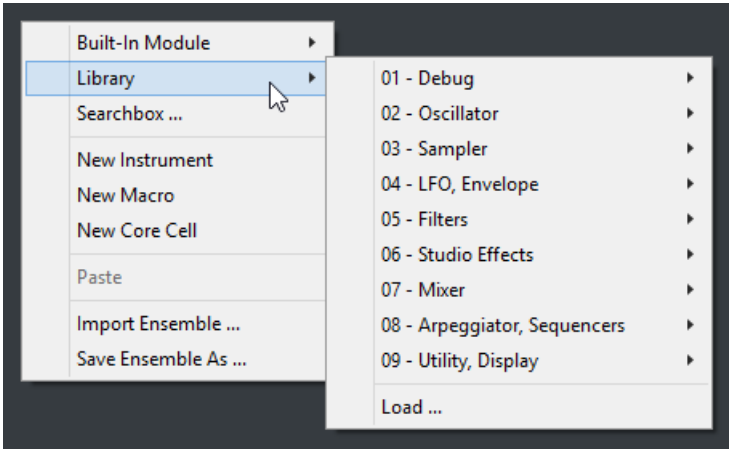
The New Ensemble, as it appears when first loaded.

This kind of split-screen setup is useful when building something like a synthesizer, as you can view the Panel and the Structure at the same time, allowing you to edit them both without navigating back and forth.

3.2.2 Building a MIDI Controlled Oscillator

For this synthesizer, there will be one oscillator, the amplitude of which will be controlled with an AHDSR envelope. So the first step towards building this will be to create these elements.

1. In the Structure View, right-click on an empty space to open the Context Menu.
2. Navigate to the Library entry to view the Macro Library.

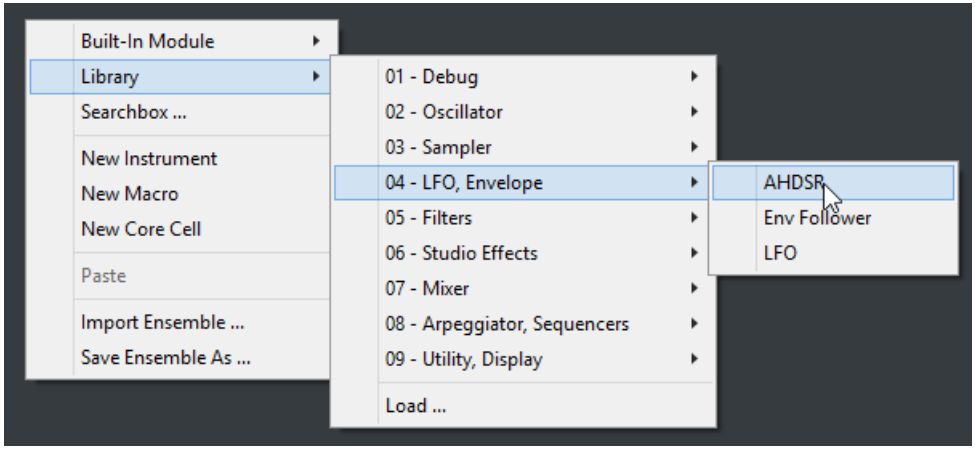


3. In the 02 - Oscillator sub-menu, select OSC Sync.

⇨ The *OSC Sync* Macro will be inserted into the Structure and will also appear on the Ensemble Panel.

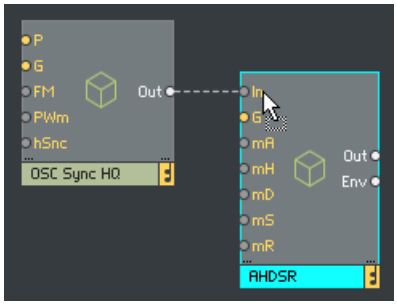


1. To create an AHDSR Envelope, repeat the above process, only this time select the AHDSR entry in the 04 - LFO, Envelope sub-menu.



With the two elements created, the next step is to wire them together.

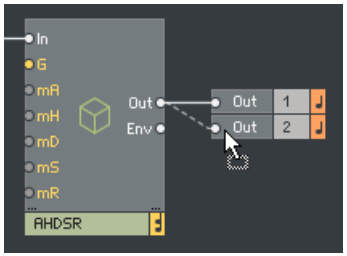
1. Click and drag with the mouse to create a wire from the Out Port of the *OSC Sync* Macro to the In Port of the *AHDSR* Macro.



The AHDSR Macro has its own internal amplifier, so connecting the oscillator to the envelope in this way means that the envelope will control the output volume of the oscillator.

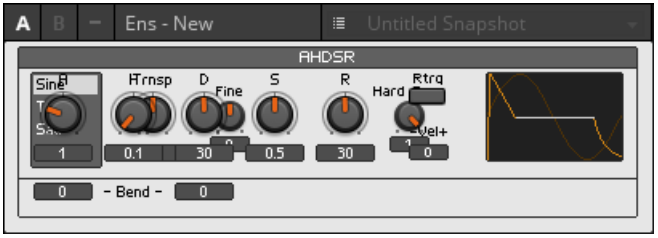
The output of the *AHDSR* Macro will be the final output of the synthesizer for now, so the next step will be to connect it to the Ensemble outputs. However, there are two Ensemble Out Ports, one for each of the stereo channels (left and right). To produce a sound from both speakers, the output of the AHDSR should be connected to both ports. Fortunately, REAKTOR allows you to make multiple connections from one output port to multiple input ports.

1. Create connections from the Out Port of the *AHDSR* Macro and to the input ports of both of the Ensemble Out Ports.



With the two Macros created and connected, it is a good time to arrange the controls on the Ensemble Panel.

Looking at the Panel view, you can see that the *AHDSR* Macro was created on top of the *OSC Sync* Macro.



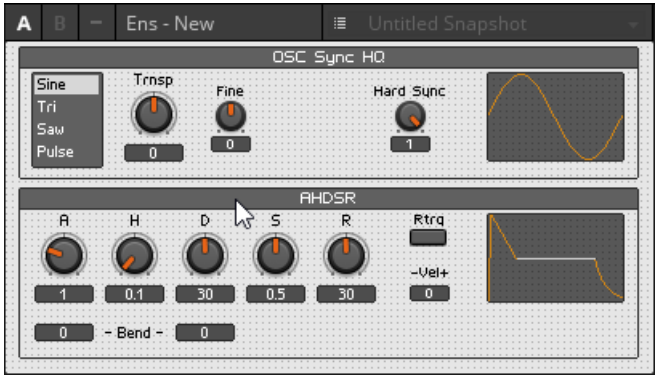
The Panel View

In order to make the Ensemble usable, you should re-arrange the Macros so that they are not obscuring each other in this way.

1. Unlock the Ensemble Panel by clicking on the padlock icon.



2. Click on the title of the *AHDSR* Macro to select the whole Macro and all of the controls contained within it.
3. Move the Macro so that it is no longer overlapping with the OSC Sync Macro.



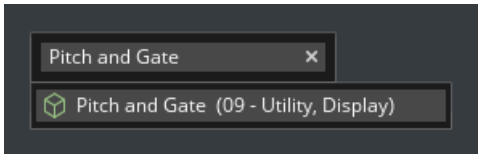
4. When you are finished, lock the Panel by clicking on the padlock again.

Whichever Macro was last selected in the Structure View will be the Macro that is on the top layer on the Panel. This means that the OSC Sync Macro could be on top of the AHDSR Macro, but the principal of moving the Macros remains the same.

With the Macros connected and the Panel arranged, the next step is to make the instrument playable. To do this you will need to connect MIDI Pitch and Gate messages to the Ensemble Structure. The MIDI Pitch will control the pitch of the oscillator and the Gate will be used to trigger the envelope.

The Macro Library includes a *Pitch and Gate* Macro that you can use for this. To speed up the process, you can use the Searchbox to load this Macro.

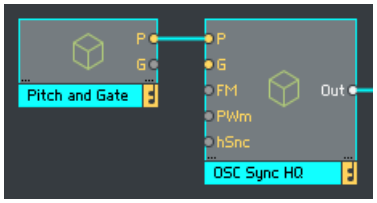
1. In the Structure View, press [Enter] to open the Searchbox.
2. Type "Pitch and Gate" into the Searchbox.



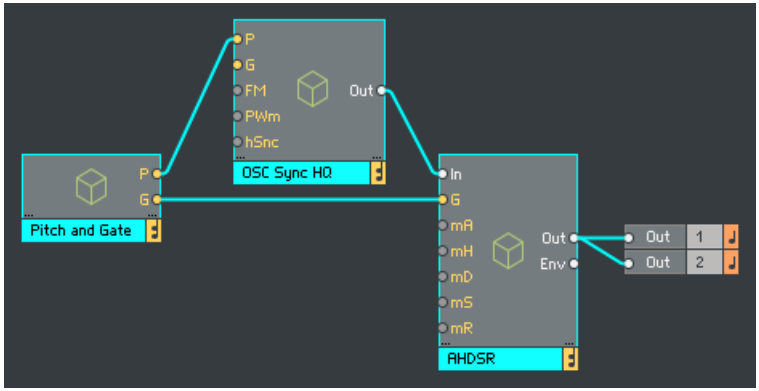
3. Press [Enter] to insert the *Pitch and Gate* Macro into the Structure.

REAKTOR's Modules and Macro Library use a number of standard abbreviations for Port names. A port named P is always for pitch, and a G Port is always for gate signals.

1. Since the MIDI Pitch will control the pitch of the oscillator, connecting the two Macros to accomplish this is a case of connecting the two P ports to each other.



2. The same logic applies to the G Ports of the *Pitch and Gate* Macro and the *AHDSR* Macro.

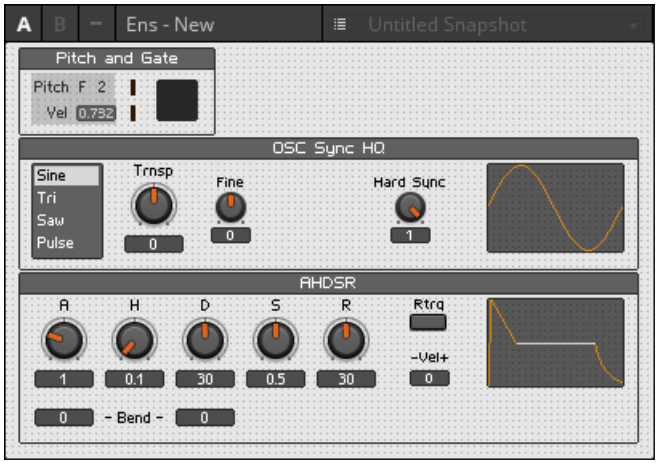


⇨ With the Pitch and Gate connections made, play the MIDI keyboard to hear a sound.

As a final step, you will need to re-arrange the Ensemble Panel to accommodate the Pitch and Gate Macro.

1. Unlock the Ensemble Panel.

2. Move the *Pitch and Gate* Macro so that it is not overlapping with the other Macros.



3. When you are finished, lock the Panel.

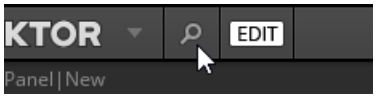
You have just created a basic synthesizer that you can play with a MIDI Keyboard.

3.2.3 Adding the Filter

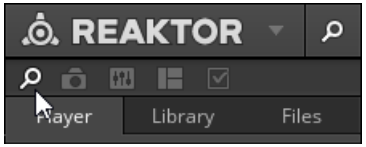
The synthesizer you have created is only the first part of a subtractive synthesizer: the oscillator. The next step is to add the second part: a filter.

For this part of the tutorial, you will use the Library tab in the Side Pane to load the filter Macro.

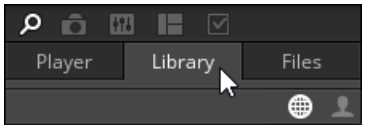
1. If the Side Pane is not already open, open it now by clicking on the magnifying glass icon in the toolbar.



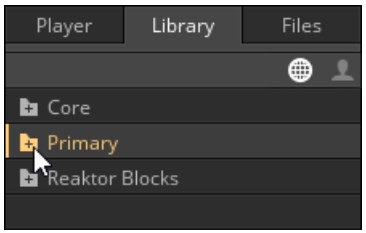
2. With the Side Pane open, click on the magnifying glass tab to open the browser.



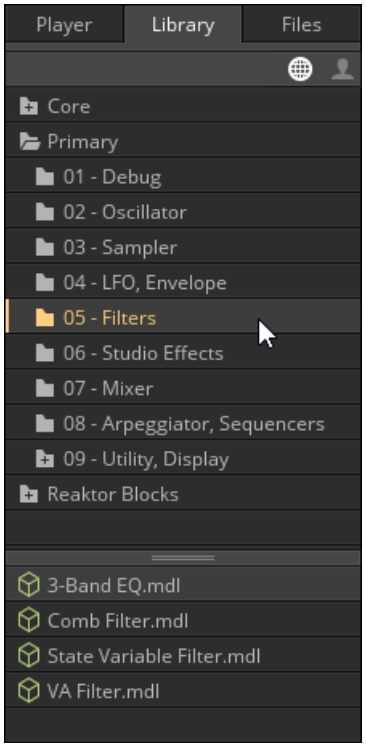
3. Click on the Library tab to display the Builder Library content.



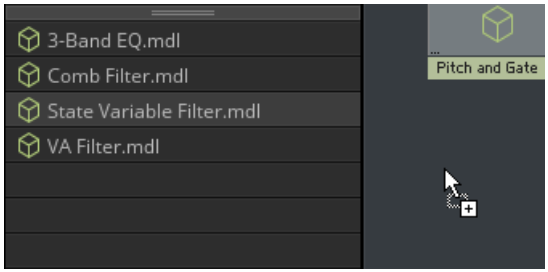
4. Open the Primary folder by clicking on the folder icon.



5. Click on the 05 - Filters folder to display its contents in the lower half of the Side Pane.



6. Click and drag the State Variable Filter into the Ensemble Structure.

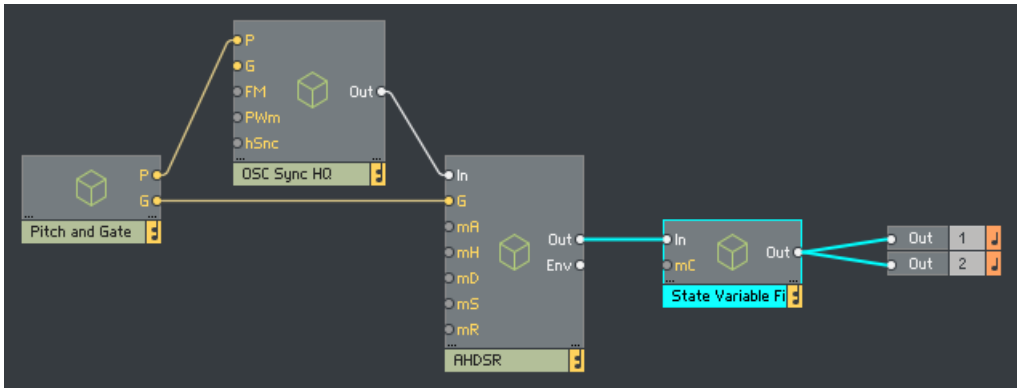


7. Release the mouse to insert the *State Variable Filter* Macro into the Structure.

⇨ The *State Variable Filter* Macro will be added wherever you dropped it.

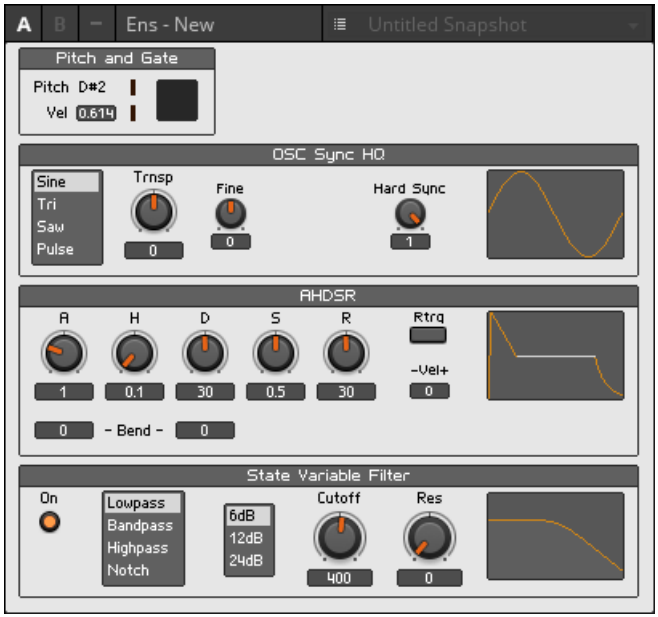
The *State Variable Filter* should be connected to the end of the Ensemble Structure, in order for it to process the audio coming from the oscillator.

1. Remove the wires connecting the *AHDSR* Macro to the Ensemble Outputs by selecting them and pressing the [Delete] key.
2. Connect the *State Variable Filter* as illustrated below.



3. Return to the Panel View and Unlock the Panel.

4. Move the *State Variable Filter* so that it is not obscuring any other Macros.



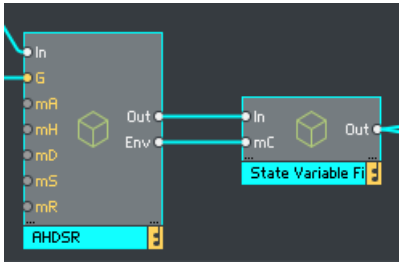
5. Lock the Panel.

⇨ Congratulations! You have built a playable subtractive synthesizer in REAKTOR.

3.2.4 Modulating the Filter

A static filter, like the one that is currently in your synthesizer, does not sound very interesting. However, with REAKTOR you can always build more! By connecting the envelope to the filter, you will be able to modulate the filter to alter the timbre of the synthesizer over time.

1. The most simple way of doing this is to connect the Env output of the *AHDSR* Macro to the mC (modulate Cutoff) input of the *State Variable Filter*.



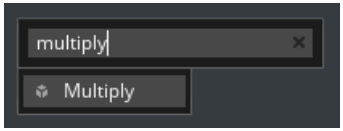
However, this doesn't allow you to control the modulation amount.

To add control for the modulation amount, you will need to start using **Modules**, which are the basic building blocks in REAKTOR.

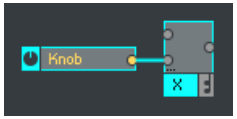
The basic skills required to use Modules are not much different from those you have developed already.

To control the modulation amount, there needs to be a multiplier between the envelope and the filter, and a knob to control the multiplication amount.

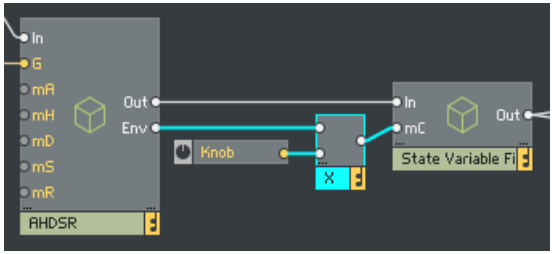
1. In the Structure View, open the Searchbox by pressing [Enter]
2. Type "Multiply" and press [Enter] to create a *Multiply* Module.



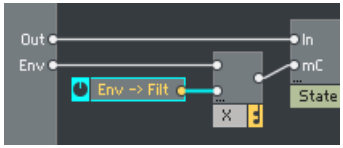
3. Open the Searchbox again and type "Knob" to create a *Knob* Module.
4. Connect the *Knob* to the *Multiply* Module.



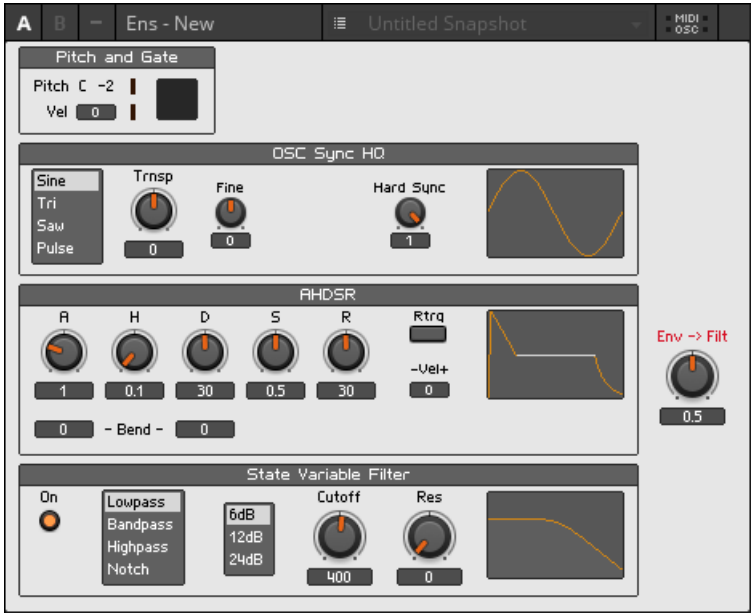
5. Connect the *Multiply* Module to the Ensemble Structure between the *AHDSR* Macro and the *State Variable Filter* Macro, as illustrated below.



6. Double-click on the *Knob* to re-name it.
7. Re-name the Knob "Env -> Filt"



8. In the Panel View, move the new Env -> Filt knob so that it is not obscuring any other controls.



⇨ Turn the Env -> Filt knob to control the intensity of the modulation from the envelope to the filter.

3.2.5 Adding Polyphony

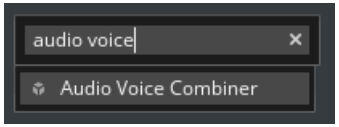
The synthesizer you have created is monophonic; it can only play one note at a time, making it impossible to play chords.

By increasing the number of Voices in the Ensemble, you can make the synthesizer polyphonic, i.e. capable of producing several notes at once. The number of Voices directly affects the number of notes you can play at once, so an Ensemble with 3-Voice polyphony can play up to three notes at any one time.

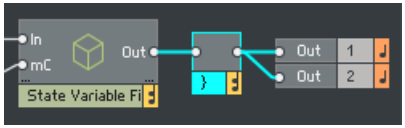
Before you change the number of voices, you will need to create an *Audio Voice Combiner*.

> The importance of Voice Combiners, as well as their use in this scenario, is described in section ↑2.4.4, Voice Combiners ↑2.4.4, Voice Combiners.

1. In the Structure View, open the Searchbox by pressing [Enter]
2. Type "Audio Voice" into the Searchbox and the *Audio Voice Combiner* Module will appear in the results.



3. Press [Enter] to insert the *Audio Voice Combiner* into the Structure. It will appear as a Module labeled with the } symbol.
4. Connect the *Audio Voice Combiner* between the *State Variable Filter* Macro and the Ensemble *Out* Ports.



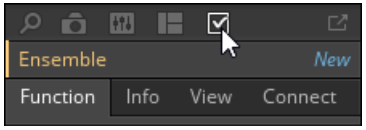
⇨ With the *Audio Voice Combiner* connected, changing the number of voices will not cause any errors.

The next step to add polyphony to the synthesizer is to increase the number of Voices in the Ensemble. This can be done in the Ensemble Properties.

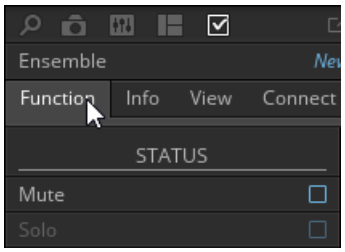
1. Select the Ensemble by either clicking on an empty space in the Structure View, or by clicking on the Ensemble Header.



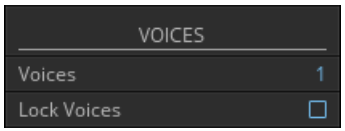
2. Click on the Properties tab (the checkbox icon) in the Side Pane.



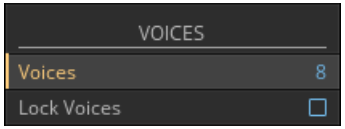
3. The Properties should open on the Function tab, but if it does not, click on the Function tab.



4. In the Function tab, locate the section called VOICES.



5. The Voices parameter in this section is currently set to 1, making the Ensemble monophonic.
6. Change the Voices parameter in this section either by clicking on the value and dragging to a new value, or by double-clicking on the value and entering a new value with the keyboard.
7. Set the Voices parameter to 8.



⇨ You can now play up 8 notes at once.

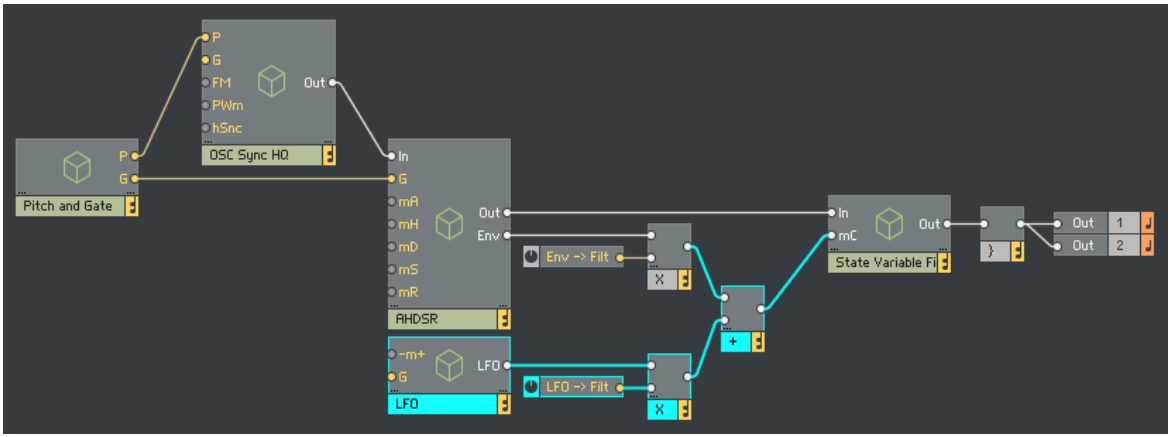


Playing several notes at once may cause REAKTOR's output to overload, so you should reduce the master output level in the Toolbar to avoid this.

Congratulations! You have created a polyphonic subtractive synthesizer in REAKTOR.

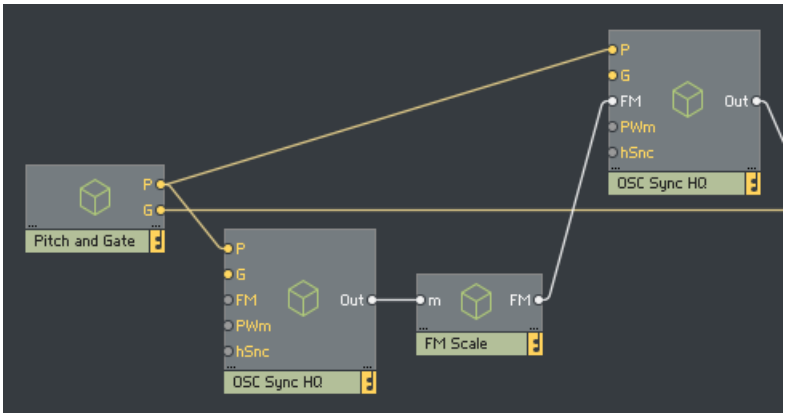
3.3 Going Further

By connecting more Macros to your synthesizer, you will be able to expand it in any number of ways. For example, you could connect an LFO to also modulate the filter cutoff.



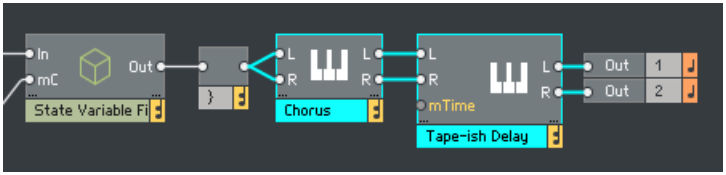
Adding LFO Modulation

You could add a second oscillator for Frequency Modulation (FM)



An FM Structure

You can even add some effects after the Audio Voice Combiner.



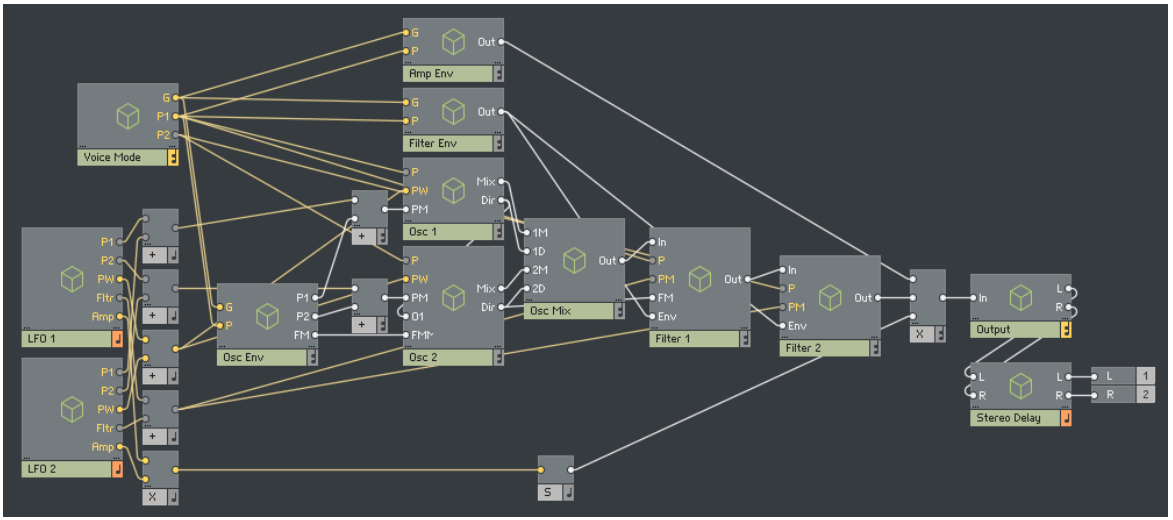
Adding Chorus and Delay Effects

By experimenting with different Macros and signal routing, you can create any number of different sound tools.

3.3.1 Related Factory Content

2-Osc is a relatively simple subtractive synthesizer Ensemble that can be found in the Factory Library in the Synthesizers folder.

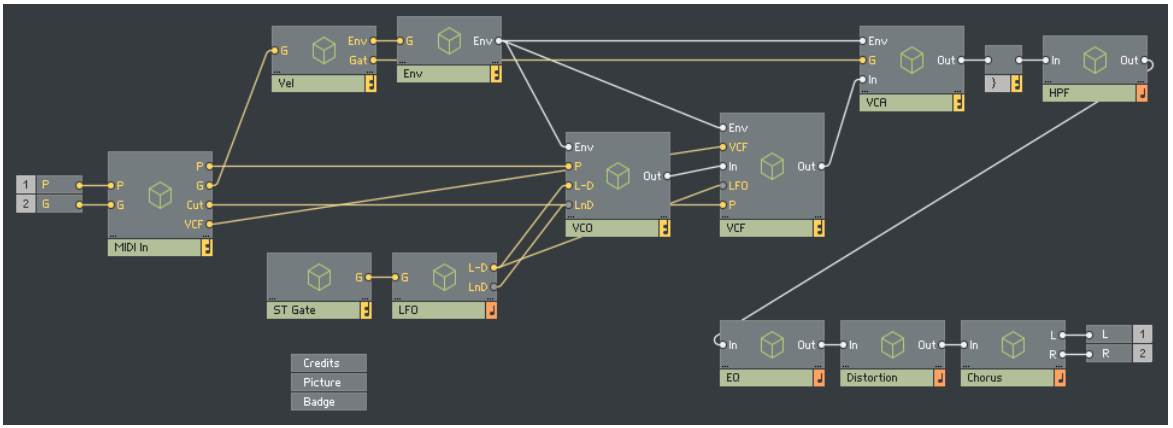
Its structure is organized into Macros like the synthesizer you have built, and these Macros are well labeled, making it easy to navigate.



The 2-Osc Structure

This not only makes it a good example for learning REAKTOR, but also makes it an easy Ensemble to customize.

Junatik is another simple subtractive synthesizer with a Structure that is also easy to read.



The Junatik Structure

4 Echo Effect Macro

4.1 Overview

Macros are a very useful way of containing sections of a larger Instrument or Ensemble project. This tutorial will lead you through the process of building your own Macros with Built-In Modules, using an echo effect as the example.

4.1.1 Previous Knowledge

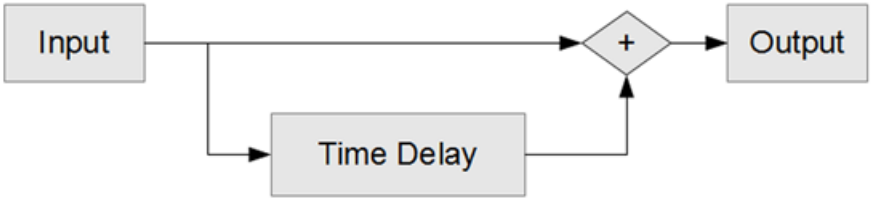
This tutorial assumes you have read the previous chapters of this document, and so are familiar with the following concepts:

- How to navigate REAKTOR's Structure View.
- The difference between Audio and Event signals.
- Polyphony in REAKTOR.
- How to create wires between ports.

4.1.2 The Theory

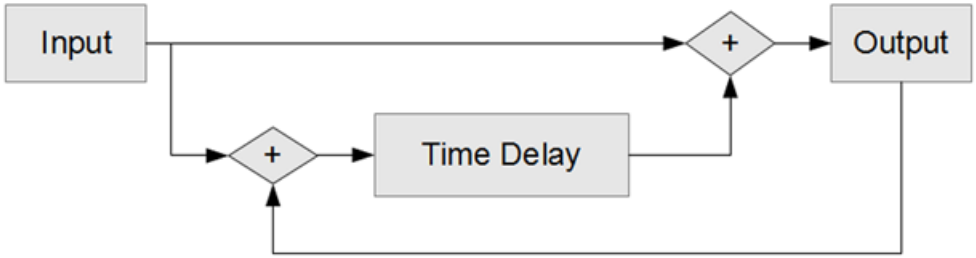
In its most simple form, an echo effect takes an audio signal and adds a time delay to it; this delayed signal is then mixed with the original signal to produce something resembling an echo.

A signal flow diagram of this effect is as follows:



The Signal Flow Diagram of a Basic Echo Effect

It is then possible to add repeating, decaying echoes by feeding the delayed signal back into the delay input, like this:



The Signal Flow Diagram of an Echo Effect with Feedback

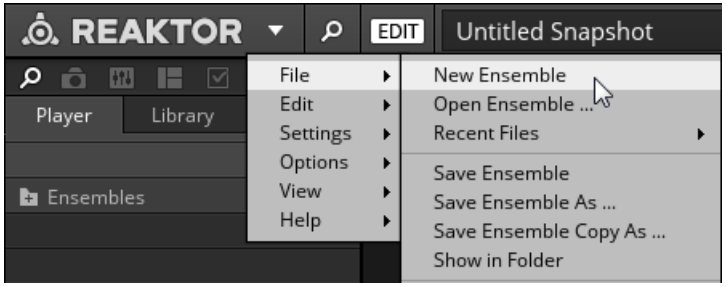
This tutorial will guide you through the process of building the simple echo effect using a delay module, before modifying it to include feedback and tempo-sync options.

4.2 Tutorial

4.2.1 Setting Up

For this tutorial you will need to start with an empty Ensemble:

1. Create a new empty Ensemble by entering the File menu and selecting New Ensemble.

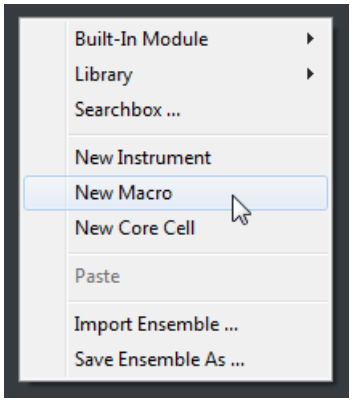


2. Alternatively, you can use the shortcut [Ctrl] + [N] ([Cmd] + [N] on OSX)

Next you can create an empty Macro inside which you will build your echo effect.

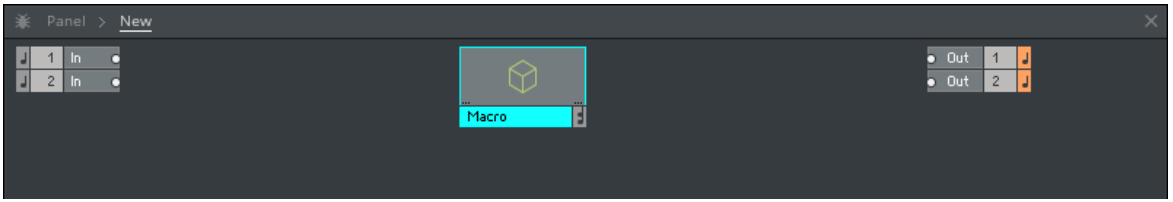
To create an empty macro:

1. Right-click in the Ensemble Structure to open the Context Menu.
2. Click on New Macro.



⇨ You have now created a new empty Macro.

Your structure should now look something like this:

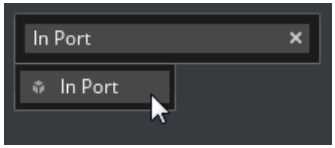


How your structure should currently look

In order for the Macro to receive and send audio signals, you will need to create input and output ports.

You can create ports in two ways:

1. Enter the Macro by double-clicking on it.
2. Press [Enter] to open the Searchbox.
3. Type "In Port" into the Searchbox.



4. Press [Enter] to create an *In Port* Module. This Module also creates an input port on the Macro container.
5. Repeat steps 2, 3, and 4, only this time enter "Out Port" into the Searchbox.

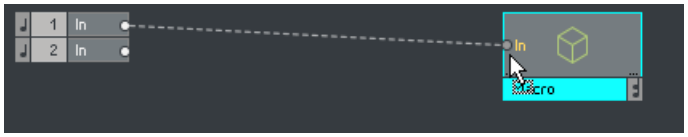


⇨ The Macro will now have one input port and one output port.

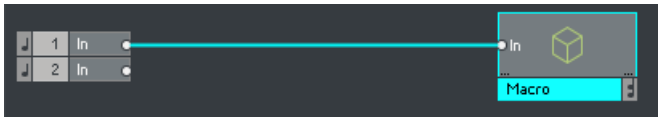
Alternatively, you can create ports without even entering the Macro:

1. If you are not already there, navigate to the top level of the Ensemble structure.
2. Click on the output of the top *In* Module.
3. While holding down both the mouse button and the [Ctrl] key on your keyboard (or [Cmd] key for OSX), drag a cable to the left side of the Macro.

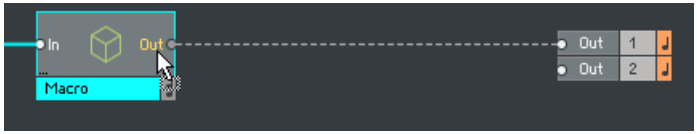
4. When you reach the edge of the Macro, an In port should appear.



5. When this happens, release the mouse button and then release the [Ctrl]/[Cmd] key. This will create a wire from the output of the In Module to a new In port on the Macro.



6. Repeat this process, only this time drag the wire from the input of the top *Out* Module to the right edge of the Macro.



⇨ The Macro will now have an input port, and output port, and wires connecting these ports to the Ensemble's first audio inputs and outputs.

1. With the Macro's ports created, Wire them to the Ensemble inputs and outputs like this:



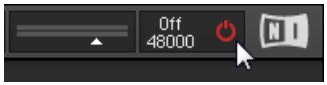
To keep things organized, re-name the Macro:

1. Double-click on the Macro's name.
2. Type in a logical name, like "Echo FX"
3. Press [Enter] or click away from the Macro to set these changes.



While working on an Ensemble's structure, it is often a good idea to deactivate the audio engine while you work. If your computer is connected to speakers and an active microphone, creating an effect could cause feedback between them.

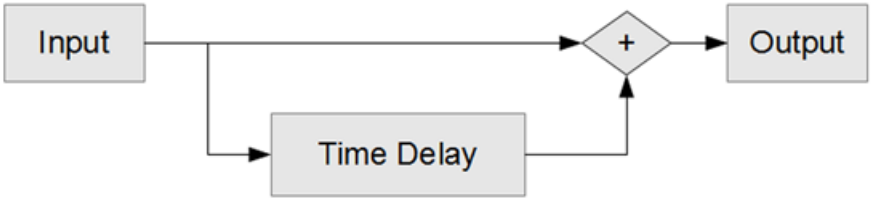
1. Deactivate REAKTOR's audio engine by clicking on the power button to the top right.



4.2.2 Building the Simple Echo Effect

1. If you aren't already in the Echo FX Macro's structure, enter it now by double-clicking on the Macro.

Since REAKTOR is a visual programming environment, you can use the signal flow diagram from earlier as a template for what you will need to build in the structure:



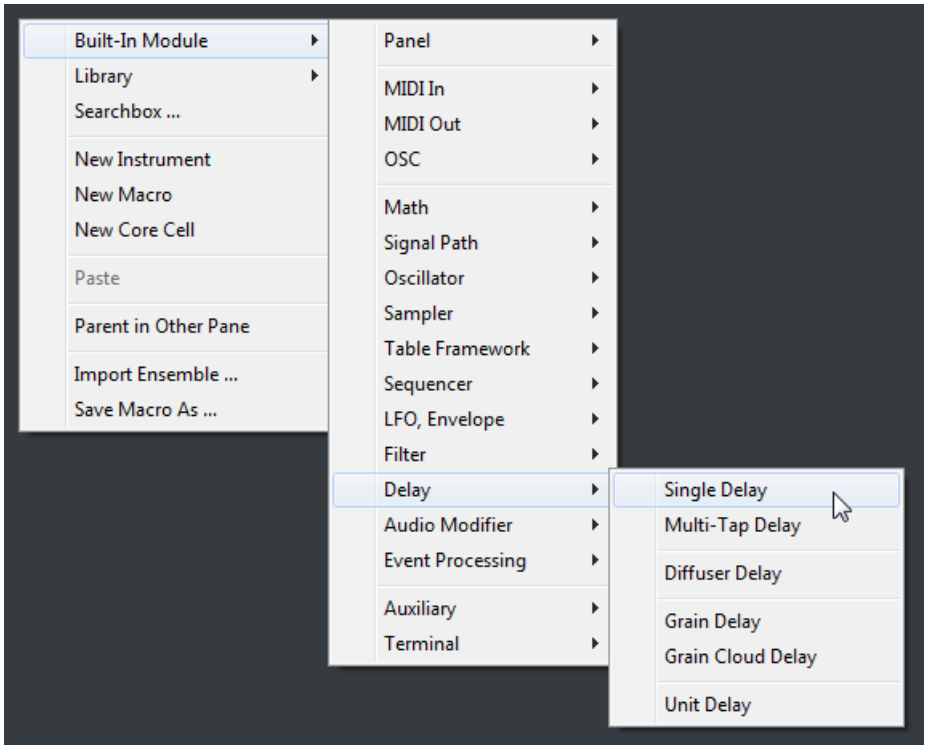
The Echo Effect Signal Flow Diagram

To build this structure you will need three Modules: *Single Delay*, *Multiply*, and *Add*.

The *Multiply* Module will be used to set the volume level of the delayed signal and the *Add* Module will be used to add the input signal to the delayed signal.

1. Right-click to open the Context Menu.
2. Navigate to Built-In Module and then select Delay.

3. Select and click on Single Delay to create a *Single Delay* Module.



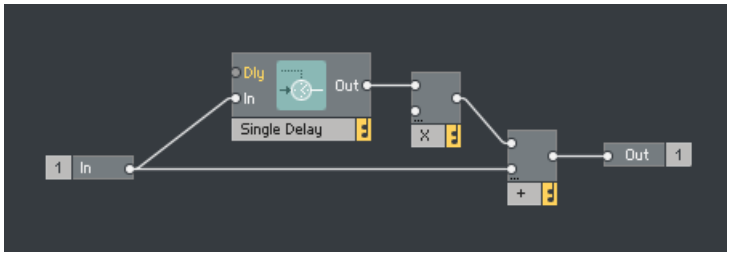
4. Open the menu again and navigate to Built-in Module followed by Math.
5. Click on Add.
6. Return to the Math section of the menu and click on Multiply.

⇨ You now have all the basic Modules you need to build the echo.

Alternatively you can open the Searchbox by pressing the [Enter] key, and then type "single delay" to locate the module directly. You can also search for mathematic modules by typing the corresponding symbol into the Searchbox, so you can type * to find the Multiply Module, and type + to find the Add Module.

If you haven't done so already, now is a good time to deactivate REAKTOR's audio engine, or mute the main output.

1. Wire the Modules together like this:

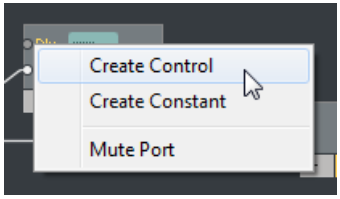


Adding Controls

You will need two knobs for this effect, one to control the delay time, and one for the delay output level.

You can create knobs in the same way you created the other Modules, either through the menu or the Searchbox, but there is also a shortcut:

1. right-click on the Dly port of the Single Delay and select Create Control.



⇨ A *Knob* Module has been created with a logical name and useful minimum and maximum values.

Although the properties of this newly created *Knob* Module are good for most cases, you may want to change the resolution of the knob to get more precise delay times.

1. Select the Delay *Knob* Module.
2. Navigate to the Properties window.
3. Select the Function tab.
4. In the RANGE section, check the Step Size parameter.

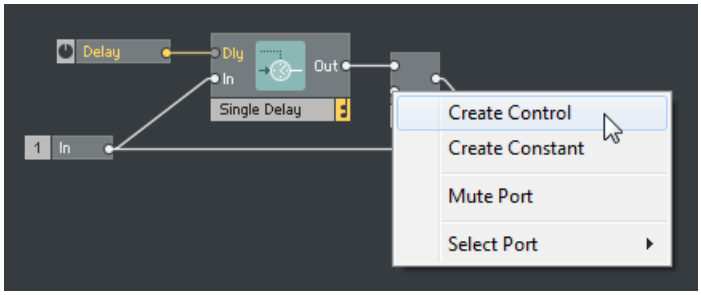
5. Double-click on this value to edit it.



6. Set it to 1.

⇨ This has decreased the step size of the control from 10ms to 1ms.

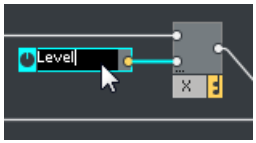
1. Use the same technique with the empty port of the *Multiply* Module.



Since the *Multiply* module ports can have many uses with many possible value ranges, the *Knob* Module we have been given here is less suited to our purposes, but we can change that.

First, re-name it to "Level":

1. Double -click on the knob and type "Level" as the new name.



2. Press [Enter] or click away from the knob to accept these changes.

Fortunately, the Max, Min, and Step Size parameters for this *Knob* Module are all set to values that work for this tutorial, so they do not need to be changed.

Finally, you'll need to arrange these controls on the Instrument Panel. Return to the Panel view and have a look at your work...



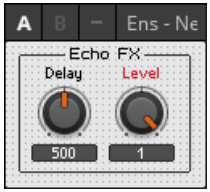
The Current Look of the Ensemble Panel

All of the knobs are stacked on top of each other, which makes them rather difficult to use. You need to move the knobs to make them user-friendly:

1. Unlock the Panel by clicking on the padlock button.



2. Move the knobs so they are no longer overlapping.



3. When you have finished positioning the knobs where you want them, lock the panel by clicking on the padlock button again.

Testing the Effect

The easiest way to test an effect is to use REAKTOR's File Player.

> If you do not know how to use the File Player, you can read the relevant chapter in the REAKTOR 6 Diving Deeper document to learn more about it.

1. Load an audio file (wav or aiff format) into the File Player (ideally something short and looped).
2. Select the loop option.
3. Press the play button.
4. If you deactivated the audio engine, or muted the main output, you will now need to reactivate the audio engine and/or unmute the main output.

⇨ You should be able to hear the audio file you loaded with a delayed signal added on top.

As the audio plays, you will be able to edit the parameters of the echo effect by using the two knobs you created:

- Move the Delay knob to control the delay time (the time between the input signal and the delayed signal).
- Move the Level knob to control the output level of the delayed signal.

4.2.3 Adding Feedback

Your echo effect is currently a simple slap-back style single delay, which is good for some uses, but by adding a feedback loop you can create repeating echoes.

Rather than introducing more *Multiply* and *Add* Modules, you can save space and keep a tidier structure by using the *Mult/Add, a*b+c* Module.

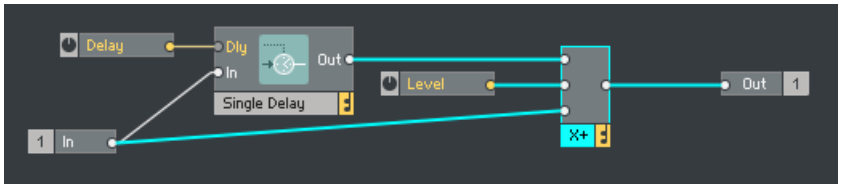


The *Mult/Add* Module outputs the result of the formula (a*b)+c, with the three input ports corresponding to the variables a, b, and c. The top port is the a value, and the bottom port is the c value.

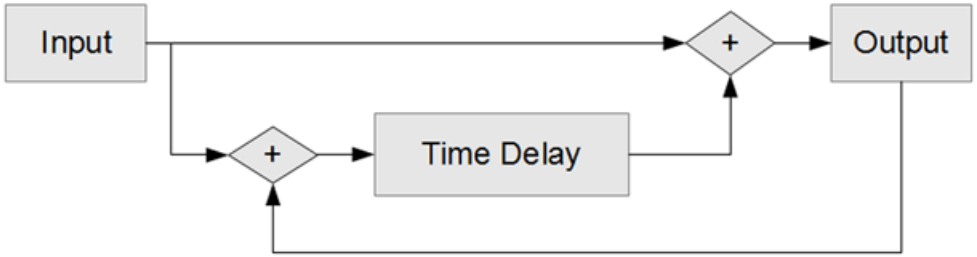
Remember that when you are editing the structure, it is advised that you deactivate the audio engine, or mute the main output.

You can easily replace the *Multiply* and *Add* Modules in your current echo effect with a single *Mult/Add* Module:

1. Enter the structure view of the Echo FX Macro.
2. Select the *Multiply* and *Add* Modules.
3. Press the [Delete] key to remove them.
4. Create a *Mult/Add* Module either by using the menu or the Searchbox.
5. Wire the *Mult/Add* Module into the structure like this:



Looking back to the signal flow diagram from earlier, you can see the modifications that need to happen to your current effect in order to add a feedback loop:



The Feedback Signal Flow Diagram

You will need to take the delayed signal, multiply it by a level control, add it to the input, and then feed the resulting signal back into the delay module. You can do this easily with another *Mult/Add* and a *Knob*:

1. Create a *Mult/Add* Module and a *Knob* Module and wire them into the structure like this:



Do not worry about the Z over the input of the Mult/Add Module, this just shows that there is feedback in the structure, and a 1 sample delay has automatically been added because of it.

2. To keep things readable, rename the new knob to "Feedback".
3. Return to the panel view and move the newly created knob so that it is no longer sitting on top of any other controls:



⇨ You have added a feedback path to your echo effect.

The newly added Feedback knob will control the amount of signal fed from the output of the delay back into its input. This effectively controls the number of repeating echoes in the effect.



Before you test the updated echo effect, make sure the feedback knob is not set too high, otherwise the effect could overload.

4.2.4 Adding Tempo Sync

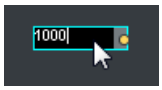
Currently, you can only set the delay time of your effect in milliseconds, but for a lot of creative echo effects the user is able to sync the delay to the host tempo. In order to do this, you will need to retrieve the tempo information somehow. Fortunately, REAKTOR includes a Built-in Module that allows you to do exactly that: the *Tempo Info* Module.

The *Tempo Info* Module outputs the host tempo in Beats per Second (BPS). You will need to convert this to Milliseconds per Beat (µPB) using the following formula:

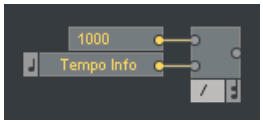
```
µPB = 1000/BPS
```

1. Create a *Tempo Info* Module.
2. Create a *Constant* Module.
3. Enter the *Constant* Module's properties and set the value to 1000.

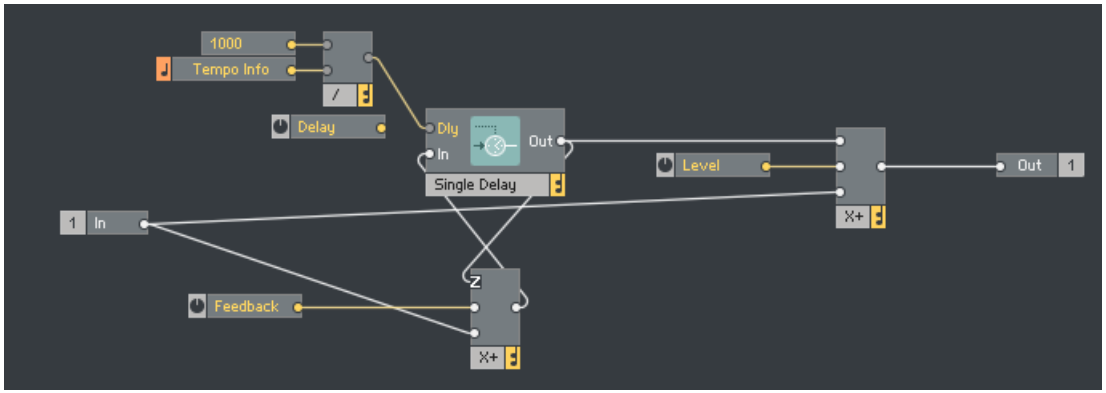
4. You can also set the value of a *Constant* by double-clicking on it, much like how you rename Macros and Knobs.



5. Create a *Divide* Module.
6. Re-create this formula in REAKTOR by using these Modules to build the following structure:



7. Connect the output of the *Divide* Module to the Dly input of the *Single Delay* Module, replacing the Delay knob.

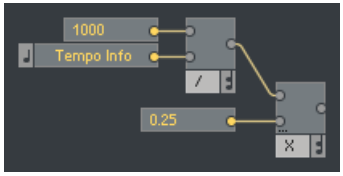


⇨ The echo time is now synched to the length of a quarter note.

Although the echo is tempo-synched, it is fixed at a single length.

You can create a delay that uses multiples of 16th note lengths quite easily:

1. Take the µPB value and multiply it by 0.25 to get the milliseconds per 16th note value.



The Multiply Module can have more than two inputs. You can attach another wire to this Module using the same technique described at the start of this tutorial, in which you used the [Ctrl]/[Cmd] keys to create ports while dragging wires.

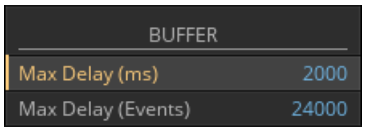
Modules and Macros that can have extra inputs created in this way are identified by the ellipsis (three dots) below its bottom input port.

If you play with the effect now, you may notice that the delay time does not change when you set the Delay knob to a value above 8. This is because the Single Delay Module has a maximum delay time buffer, which is 1000ms by default (equivalent to 2 beats at 120bpm).

As the delay essentially records the audio in order to play it back later as the delayed signal, the delay buffer assigns a section of memory for this recording. Larger buffer sizes can be useful for longer delay times, but will take up more RAM. The maximum setting for the delay buffer is 1000000ms (a little under 16.7 minutes) which equates to about 96MB of RAM for 16bit/48kHz audio.

You can change the buffer setting in the Single Delay's Properties:

1. In the Single Delay's properties, under the Function tab, find the BUFFER section.
2. In this section locate the Max Delay (ms) parameter.
3. Double-click the parameter's value and set it to 2000.



4. Press [Enter] to set this change.

Congratulations, you have just created a tempo-synced echo effect in REAKTOR!

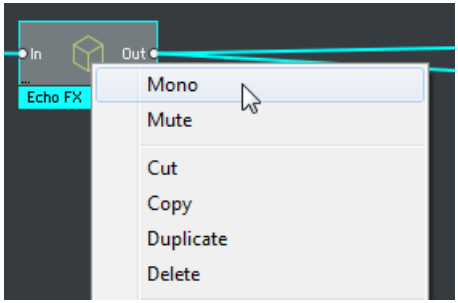
4.2.5 Saving and Loading the Macro

You can export your echo effect for use in other Ensembles by saving it as a Macro.

Before you save it, you'll need to make sure it remains as a monophonic structure, even in polyphonic Ensembles. To do this:

1. Navigate to the top level of your Ensemble.

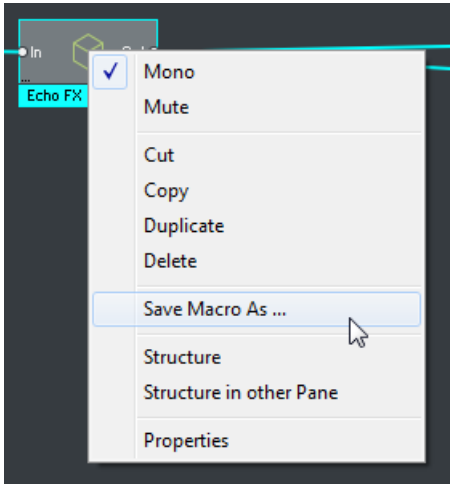
2. Select the echo Macro and right-click on it.
3. Select the Mono option from the menu.



⇨ All Modules inside the Macro have been set to monophonic, so even when you load the Macro into a polyphonic Instrument, the echo effect will remain mono.

Next, export the effect by saving it as a Macro:

1. Select the echo Macro.
2. Right-click the Macro and select Save Macro As …

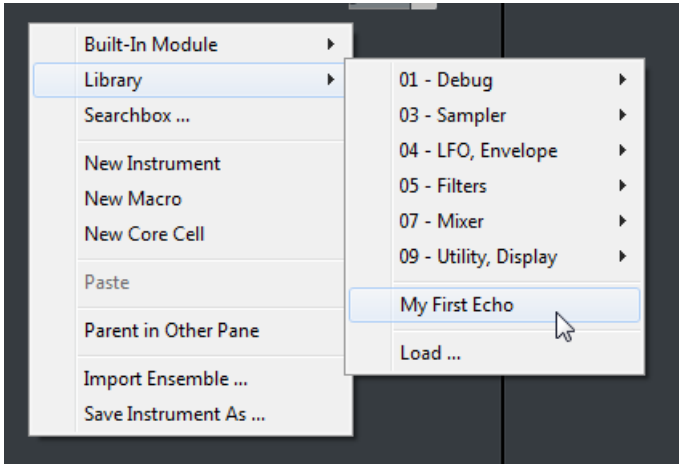


3. Select a save location for the Macro file (saving it to the default location makes it easier to locate in REAKTOR later).
4. Give it a unique name like "My First Echo".
5. Click Save.

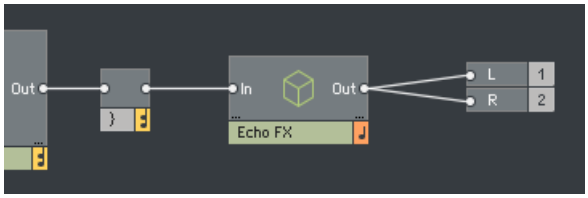
⇨ You can now load your echo effect like any other Macro.

Using the subtractive synthesizer from the previous tutorial as an example, you can load and connect your echo effect like this:

1. Load the Ensemble you wish to use and enter its structure.
2. Rick-click on an empty space in the structure and select Library.
3. If you saved your effect in the default location, it should appear at the bottom of this menu.



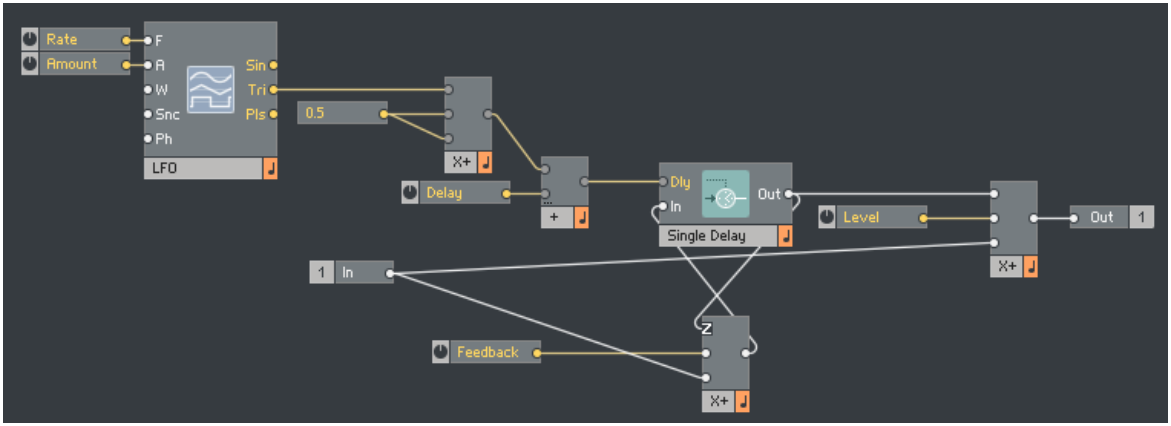
4. If you saved it somewhere else you can locate it by clicking on Load ...
5. Since the echo effect is monophonic, it will need to be placed after any *Audio Voice Combiners* in the structure, like this:



4.3 Going Further

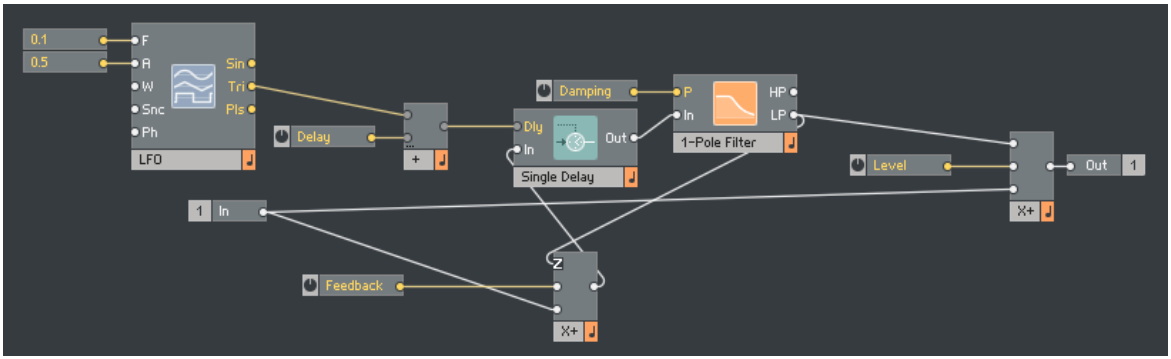
The delay effect is the basis for a few other effects including flanger and chorus.

You can create a flanger/chorus by modulating the delay time with an LFO:



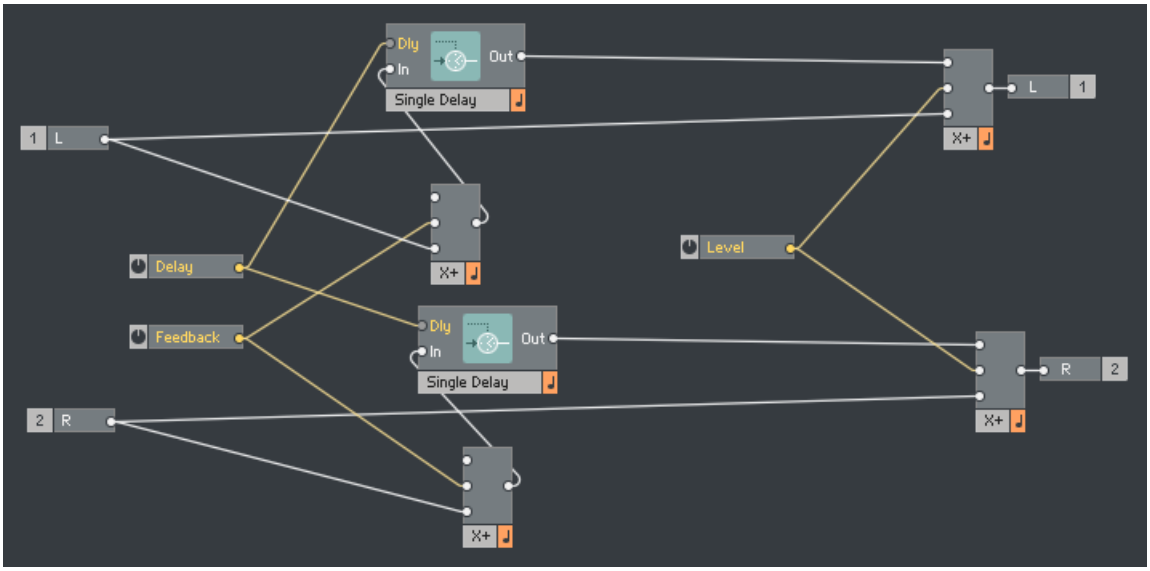
The structure of a simple chorus/flanger effect.

You could also create a vintage style delay by adding filters to the delay output/feedback chain, and slow, subtle modulation to the delay time:



The structure of a vintage style echo effect.

You can make any of these effects stereo by duplicating the macro structure:



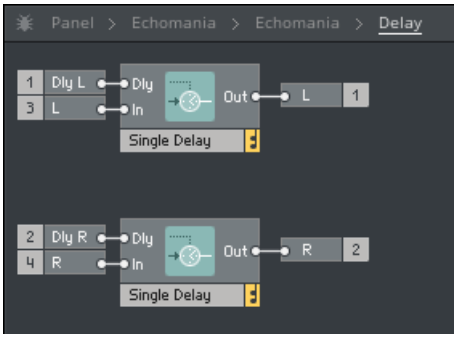
The structure of a basic stereo delay.

4.3.1 Related Factory Content

The following REAKTOR 6 Factory Library Ensembles were all built using simple delay units at their core. With what you have learnt here, you should be able to reverse engineer these Ensembles to take your knowledge further.

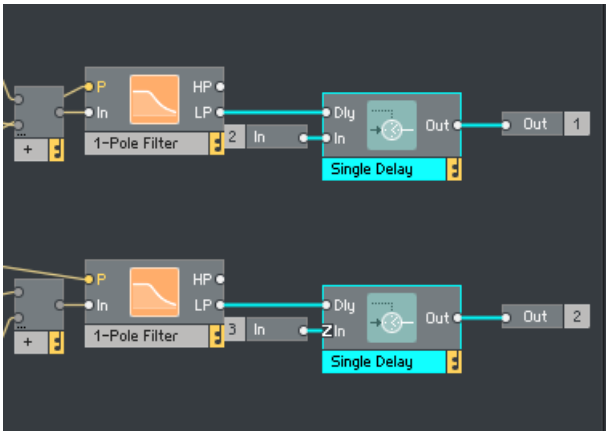
> The following delay based Ensembles can be found in the Effects section of the Factory Library.

Although it features LFOs and EQs, the *Echomania* Ensemble is ultimately driven by two *Single Delay* Modules.



The two Single Delay Modules in Echomania

Longflow is another effect that ultimately comes down to two *Single Delay* Modules.



The Single Delay Modules in Longflow

5 Basic Step Sequencer

5.1 Overview

Event signals in REAKTOR behave differently from audio signals: audio signals constantly send values at the audio rate, whereas event signals can have their timing controlled. Events can be triggered, and thus can be used to trigger other events.

This process of triggering timed events is ideal for creating step sequencers, which is what the following tutorial will cover.

5.1.1 Previous Knowledge

This tutorial assumes you have read the previous chapters of this document, and so are familiar with the following concepts:

- How to navigate around REAKTOR's Edit view
- Understanding the difference between audio and event signals
- Understanding the difference between monophonic and polyphonic signals
- How to create wires between ports
- How to create Macros and Modules

5.1.2 The Theory

A basic step sequencer sends a sequence of values at timed intervals. They are commonly used to create musical patterns by sequencing the pitch of a synthesizer's oscillators, or they can be used to modulate sound parameters to create rhythmic effects. It is also common to sync a sequencer to the tempo of a project.

Common hardware step sequencers have a row of knobs to set the sequence of values, with each knob representing a step in the sequence. A clock signal is used to progress the sequence: when a clock tick is received, the sequencer advances to the next step, outputting the current step's value.

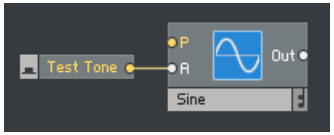
In REAKTOR, our clock signal will be created using timed events.

5.2 Tutorial

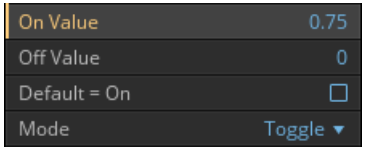
5.2.1 Setting Up

In order to test the sequencer, you will need to create a single oscillator:

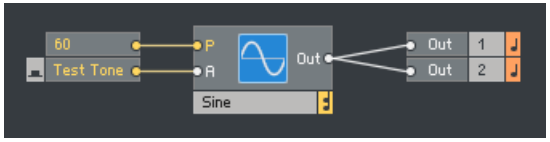
1. Create a new empty Ensemble.
2. Enter the structure and create a *Sine Oscillator* Module.
3. Create a button and attach it to the A (amplitude) input port of the *Sine Oscillator*.
4. Rename the button to "Test Tone"



5. In the button's Properties, change the On Value to 0.75. This sets the output amplitude of the *Sine Oscillator* when the button is on. An output amplitude of 1 can overload REAKTOR's output and should be avoided.



6. Create a Constant for the P (pitch) input of the oscillator by right-clicking on the port and selecting Create Constant.
7. Connect the output of the *Sine Oscillator* to the Ensemble's main Out ports.



⇨ The oscillator is now ready to be used. Pressing the Test Tone button activates the Oscillator with a MIDI pitch of 60 (middle C). We recommend to lower the volume of your loudspeakers before testing the oscillator.

Next, you will create a Macro inside which you will build your sequencer:

1. Create a new Macro.
2. Re-name it to "Sequencer".



5.2.2 Building the Basic Sequencer

As mentioned in the Overview, a sequencer uses a clock to move between steps in a sequence, so you will start building your sequencer by creating a clock source.

1. Enter the *Sequencer* Macro structure.
2. Create a *Clock Oscillator* Module.
3. Create a *Constant* at the A input port by right-clicking on the port and selecting Create Constant.
4. Create a *Knob* at the F input port using the same right-click method, but this time selecting Create Control.



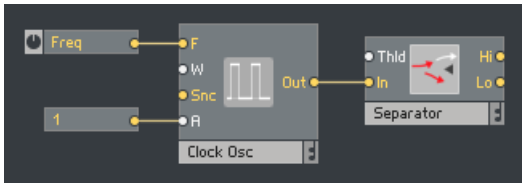
Using the right-click method to create constants and controls usually sets their Properties to commonly used values.

You have now created a clock source (the *Clock Oscillator*) which outputs events with a value of 1 at a rate specified by the frequency at the F input. Each of these events is followed by an event of 0, producing a kind of square wave with a frequency of F.

Since the sequencer structure will ignore the value of these events, only using them to trigger additional events, you will need to filter out the 0 value events, as these double the frequency of events.

1. Create a *Separator* Module. This Module splits events depending on whether or not their value is above or below a certain threshold (Thld).
2. For this case, the threshold should be 0, so you can leave the Thld port empty as it defaults to 0

3. Connect the Out port of the *Clock Oscillator* to the In port of the *Separator*.

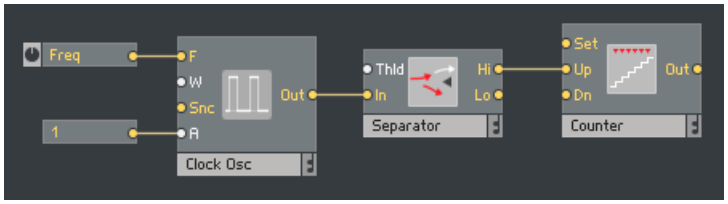


⇨ The Hi output port of the Separator Module will now act as the output of our clock.

Each event produced by this clock needs to trigger a value to increase by one. This value will be used to select the steps in the sequencer.

You can build this using the *Counter* Module.

1. Create a *Counter* Module.
2. Connect the Hi output of the *Separator* to the Up port of the *Counter*.



⇨ Every time the clock sends an event, the *Counter* will increase by one.

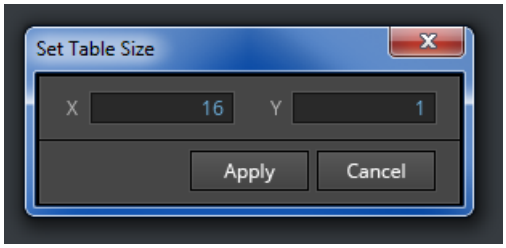
With this basic counting structure in place, you can move on to creating a Module for setting the values the *Counter* will address. For this tutorial, you will use a Module called the *Event Table*. This Module holds a table of values (if you are familiar with programming, you may like to think of it as a 2D array), and is perfect for quickly creating step sequences.

1. Create an *Event Table* Module.
2. In the *Event Table* Properties, locate the SIZE section in the Function tab.

3. Click on the Set button.



4. A window will open, asking you to set the table size for the event table.
5. You will be creating a 16 step sequencer, so set the X value to 16 and leave the Y value at 1.



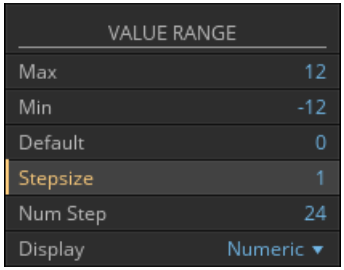
6. Click Apply.

⇨ You have created a table of 16 values.

Next, you will need to set the range of the values in the table. This sequencer will be used to control the pitch of the *Sine Oscillator* you created earlier, so the range should be +/- one octave (+/- 12 semi-tones).

1. In the *Event Table* Properties, locate the VALUE RANGE section in the Function tab.
2. Set the Max value to 12 and the Min value to -12.

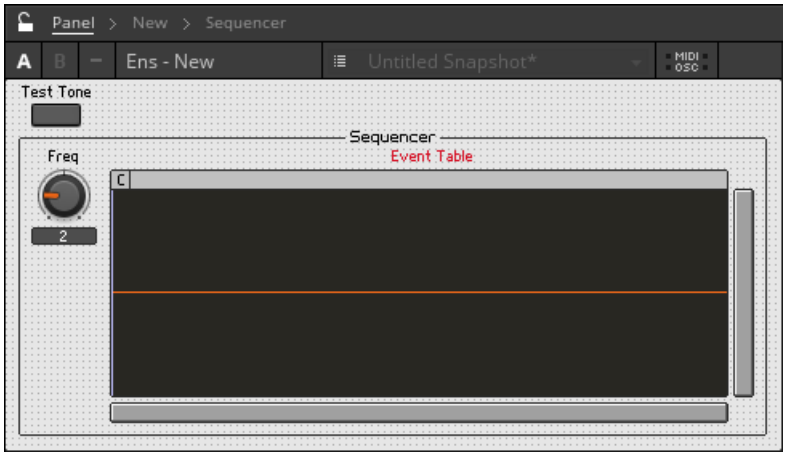
3. Set the Stepsize to 1. The Num Step property should automatically update to 24 when you do this.



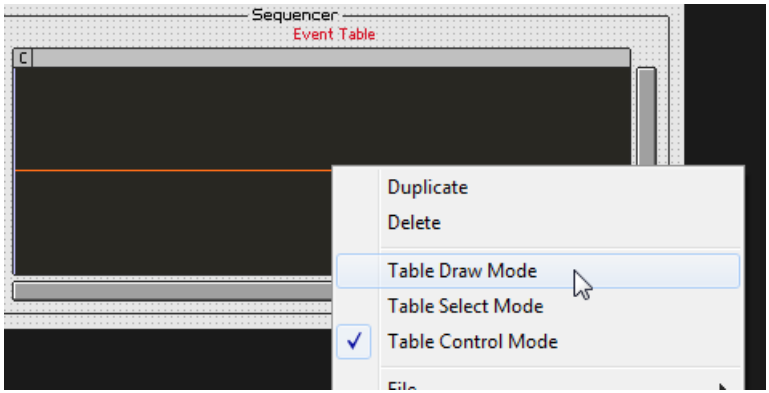
You have created a table of values with a possible range of -12 to +12. You now need to create a way of editing these values.

The *Event Table* has a panel element which shows the values in the table. With the following additional settings, you will also be able to edit them from the panel:

1. Go to the Ensemble Panel view.
2. Unlock the Panel and rearrange the controls so they are no longer on top of one another.



3. Right-click on the *Event Table* and click on the Table Draw Mode option.



4. Lock the Panel.

⇨ You will now be able to draw on the Event Table to edit the values in the sequencer.

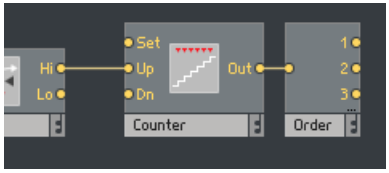
To complete the sequencer, you need to connect the *Counter* to the *Event Table.*

The output of the *Counter* will select an X position in the *Event Table* using the RX (Read X) input, and then trigger the *Event Table* to output the value at that position using the R (Read) input.

Triggering two inputs from a single output in this way can be sensitive to the ordering of events. In this case the RX input needs to receive the *Counter* output event before the R input.

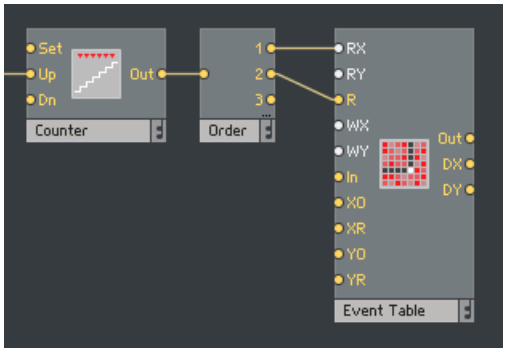
To control the order of the events, you can use an *Order* Module.

1. Navigate to the Sequencer structure.
2. Create an *Order* Module.
3. Connect the output of the *Counter* to the input of the *Order* Module.



4. Connect output 1 of the *Order* Module to the RX input of the *Event Table*.

5. Connect output 2 of the *Order* Module to the R input of the *Event Table*.



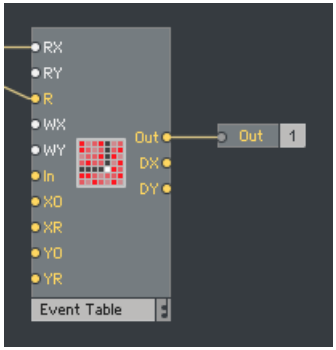
⇨ The *Order* Module now ensures that RX receives the *Counter* value before R.

Simple cases like this may not require an *Order* Module; however it is good practice to use them to ensure that events are processed as planned.

Testing the Sequencer

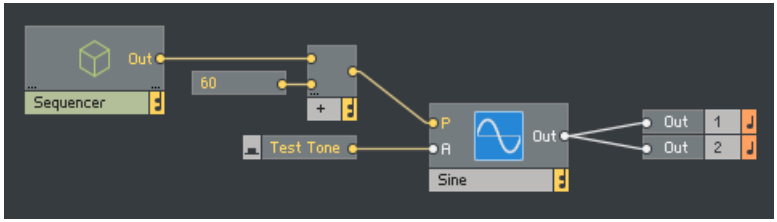
With your sequencer complete, you can connect it to the Sine Oscillator for testing:

1. Create an *Out Port* Module in the *Sequencer* Macro.
2. Connect the Out port of the *Event Table* to the input of the *Out Port* Module.

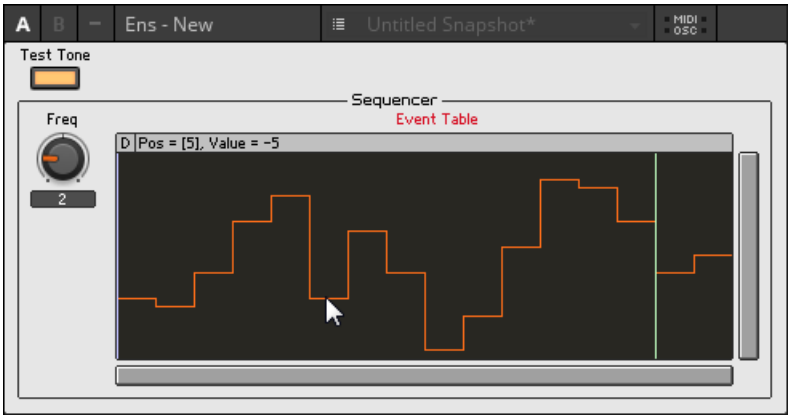


3. Navigate to the top level of the Ensemble structure.
4. Create an *Add* Module.

5. Use the *Add* Module to add the sequencer output to the *Constant* at the P input of the *Sine Oscillator*, like this:



When you return to the Ensemble Panel, you can draw a sequence by clicking and dragging on the *Event Table*. You can then listen to this sequence by activating the Test Tone button.



Editing and Testing the Sequencer

Changing the Freq value will alter the rate at which the sequence plays.

5.2.3 Adding Tempo Sync

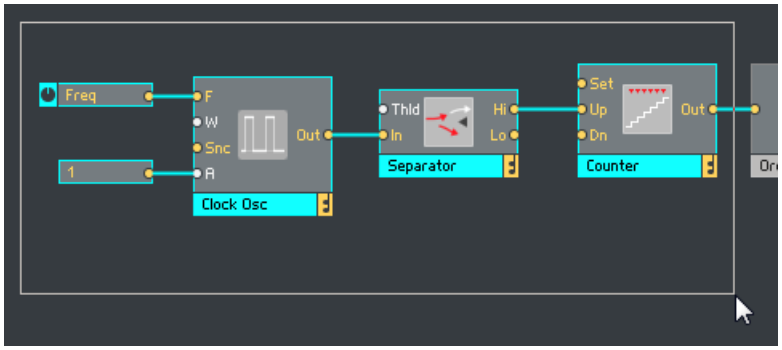
It is common for step sequencers like the one you have just created to be synchronized to the host tempo. This can be further improved by synchronizing to the song position, not just the tempo.

REAKTOR has a *Song Position* Module that allows you to access the current playback position of the host in 1/96 note resolution.

In this next step of the tutorial, you will replace the clock you previously created with this *Song Position* Module.

Before you build anything new, remove all of the Modules that are a part of the clock functionality:

1. Enter the structure of the *Sequencer* Macro.
2. Select the Modules pictured below. You can select them all at once by clicking and dragging with the mouse to create a box around them.



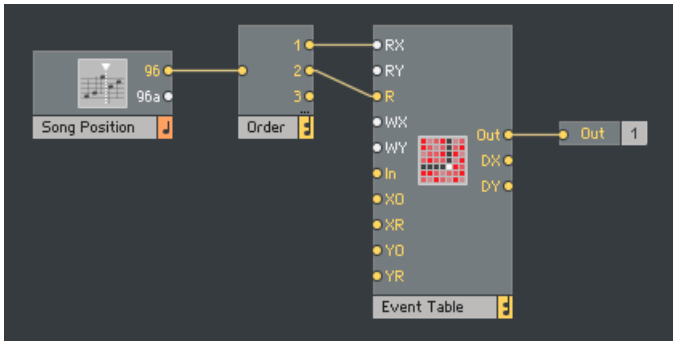
3. Delete them by pressing the [Delete] key on your keyboard.

Now you can replace the old clock structure with a *Song Position* Module.

1. Create a *Song Position* (MIDI In) Module.

The *Song Position* Module has two outputs: the top one outputs event signals and the lower one outputs an audio signal. Both outputs send the host song position at 1/96 note resolution.

2. Connect the event (top) output to the input of the *Order* Module. You are using the event output as it sends a single event with each song position value, and thus can also be used as a clock source.



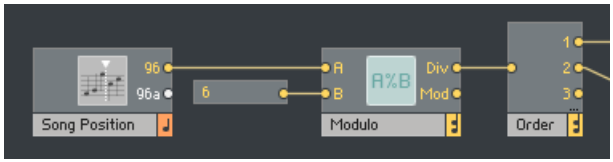
When you return to the panel and test the sequencer, you may notice that it is not running. This is because the song position will not change unless the transport is running, and if the song position does not change then the *Song Position* Module will not output any events.

1. Start the transport by pressing the play button in the Toolbar, or by pressing [Space] on your keyboard.

With the transport running, the sequence will start playing. However, the sequence is running at a fast 1/96 rate, so you need to adjust the structure to control this.

By using a divider that truncates (rounds down), you can control the resolution of the *Song Position*, and thus the rate of the sequencer.

1. Create a *Modulo* Module. This Module outputs the truncated result of a division function.
2. Wire the event output of the *Song Position* Module to the A input of the *Modulo* Module.
3. Create a *Constant* and connect it to the B input of the *Modulo* Module.
4. In order to get a 1/16 rate, you need to divide the 1/96 position by 6 (96/6 = 16), so set the *Constant* to 6.
5. Connect the Div output of the *Modulo* Module to the input of the *Order* Module.



⇨ The sequencer now runs at 1/16 rate.

Even though you have reduced the resolution of the song position to 1/16 notes, the Modulo Module is still outputting events at a 1/96 rate. So instead of outputting the following values and events:

```
0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, etc…
```

It is now outputting the following at the same rate:

```
0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, etc…
```

To optimize the structure, the sequencer should filter out repeated values. Fortunately REAKTOR includes a *Step Filter* Module to do just that.

1. Create a *Step Filter* Module and connect it between the *Modulo* and *Order* Modules, like this:



The Tol (tolerance) input of the Step Filter can remain disconnected since the default value works with this structure.

With the *Step Filter* connected, a new event will only be sent to the *Event Table* when the value changes.



Although this optimization did not change the final output of the Ensemble in any noticeable way, this technique is very important. If a structure uses events to trigger sounds, filtering repeated values ensures that the sounds are only triggered when the value changes.

The final step in creating this sequencer is to add control over the rate. Changing the value at the B input of the Modulo Module will do this, however a knob is not ideal as you do not need a large range of values, only those that produce commonly used musical note divisions.

The *List* Module allows you to create a menu with any number of entries, each with specified names and values. Using this Module, you can create a menu of musical note divisions, but output the number required to get that rate from the *Song Position* Module.



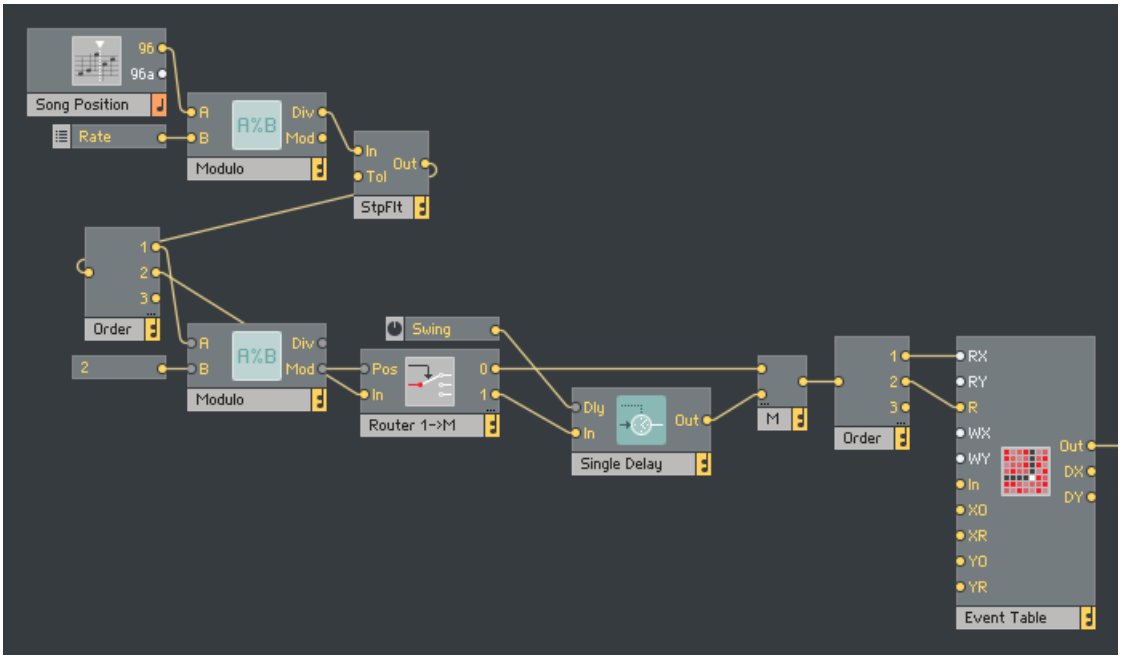
To find the values needed, just use the simple formula: value = 96*rate. For example, if you want quarter notes the value = 96 * (1/4) = 96/4 = 24.

- You can now use the Rate Menu to specify the rate of the sequencer using musical note divisions.

5.3 Going Further

You can use a sequencer to control any parameter, not just pitch. Try saving the *Sequencer* Macro and importing it into other Ensembles. Be sure to set the Macro to Mono, and to change the Range values of the *Event Table* accordingly.

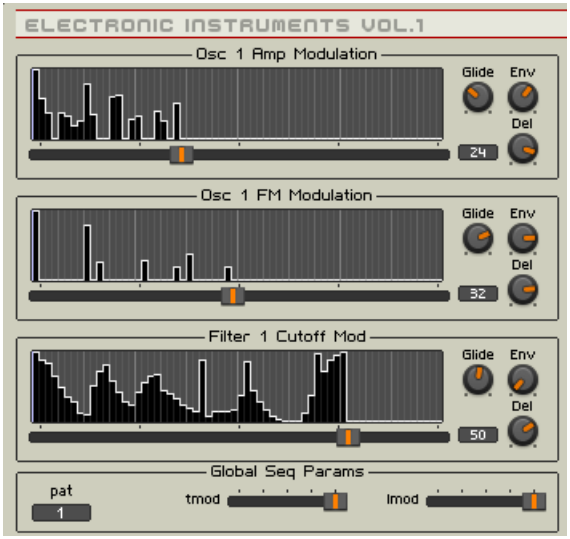
You can use further event handling and a *Single Delay* Module to create swing:



A Sequencer with Swing

5.3.1 Related Factory Content

Any sequencer in the Factory Library will be built on the basic concepts you have just learnt. However, there are two Ensembles in particular that uses Event Table Modules in their sequencers: *Atmotion* and *Vierring*. Both of these Ensembles can be found in the Sequenced Synthesizers section of the Factory Library.



The Step Sequencers of Atmotion

6 Advanced Step Sequencer

6.1 Overview

REAKTOR offers a number of Modules that you can combine to create custom controls.

Building on the ↑5, Basic Step Sequencer tutorial, this tutorial will guide you through using the advanced Panel Modules and techniques to create your own Step Sequencer interface.

6.1.1 Previous Knowledge

As well as assuming a basic understanding of building in REAKTOR, this tutorial also assumes that you have completed the tutorial in section ↑5, Basic Step Sequencer ↑5, Basic Step Sequencer.

The following will not only build on the Ensemble created in that tutorial, but will also assume all knowledge acquired to that point.

6.1.2 The Theory

By this stage you should have already created a ↑5, Basic Step Sequencer, however certain aspects of this sequencer are limited - the information in the sequencer is stored in the Ensemble, and not in the Snapshots.

Building on the ↑5, Basic Step Sequencer also makes a good starting point for learning some advanced concepts about REAKTOR's GUI (Graphical User Interface) capabilities.

Until now you will have been using simple, pre-made Panel Modules like knobs and buttons. While these are useful, sometimes you'll need a control with an option that is unavailable in those on offer, and this is why REAKTOR also offers additional advanced Panel Modules.

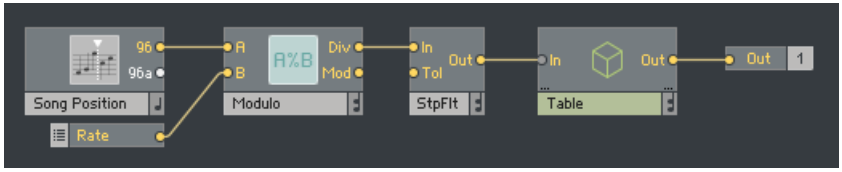
In this tutorial, you will replace the *Event Table* Module used in the ↑5, Basic Step Sequencer with your own table, built from scratch. This new table will allow you to save sequences in Snapshots, and not just in the Ensemble.

6.2 Tutorial

6.2.1 Setting Up

This tutorial will build on the ↑5, Basic Step Sequencer, focusing on replacing the *Event Table* Module.

1. If you do not have the Step Sequencer Ensemble loaded, open it now.
2. Enter the Sequencer Macro structure and delete the *Event Table* Module and the *Order* Module.
3. Create a new empty Macro.
4. Rename it to "Table".
5. Attach the Table Macro to the structure as follows:



6.2.2 Building the Sequencer Interface

Before you build any of the step sequencer functionality, you will create the interface elements needed.

1. Enter the Table Macro structure.
2. Create a *Mouse Area* Module.



The *Mouse Area* Module allows you to access the state and position of the user's mouse when it is over a defined area on the Panel. Because of this, the *Mouse Area* is a very useful Module when creating custom interfaces.

The *Mouse Area* sends the actions of the user to our structure, but this is only one part of the functionality of an interface control; the sequencer will also need a display to show the values of the sequencer to the user.

To ensure the display stays under the *Mouse Area* (so as not to block the user's input) you should create a new Macro to contain the display.

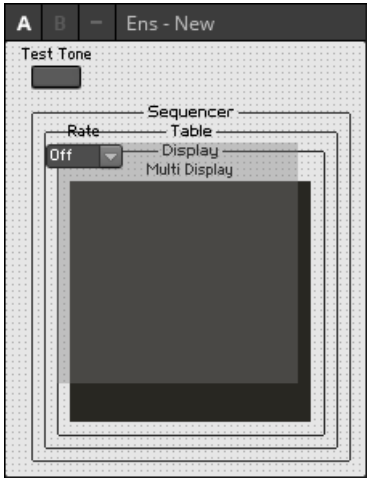
1. Create a new empty Macro.
2. Name this Macro "Display".
3. Enter this Macro and create a *Multi Display* Module.

⇨ Because the *Multi Display* is contained in a Macro one level below the *Mouse Area*, it will be placed under of the *Mouse Area* on the panel.

The *Multi Display* Module is a Panel element that can contain a number of objects, which can be made of basic shapes, like lines and rectangles. In this tutorial, the *Multi Display* will be used to create the bars of the sequencer.

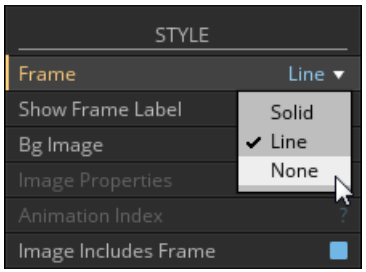
The *Mouse Area* and the *Multi Display* will be the only Panel elements you'll need for this tutorial, so now is a good time to adjust the Panel to make sure everything is in the right place.

1. Return to the Ensemble Panel and Unlock the Panel.
2. When you unlock the Panel, you will notice that the Mouse Area is shaded so that you can see where it is located.

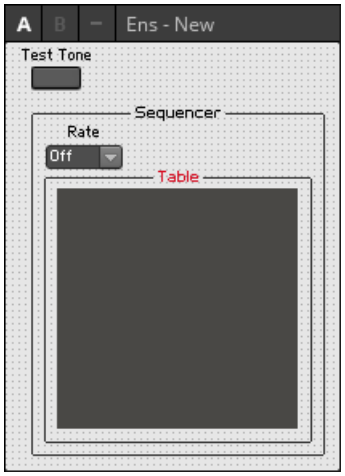


3. Select the *Multi Display* to show its properties.
4. Navigate to the View tab and uncheck the Show Label option to hide the label.
5. Select the Display Macro to show its properties.

6. Navigate to the View tab and set the Frame parameter to None. This will hide the frame of the Macro without hiding its contents.



7. Move the Table Macro so that it is no longer overlapping with the rate control.



The interface is starting to come together, but the default square shape of the Table Module is not ideal for a 16-step sequencer.

To re-size the Table:

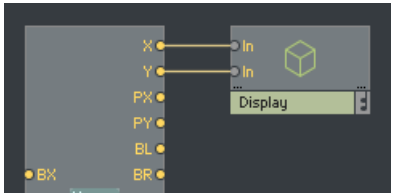
1. Select the *Mouse Area* Module.
2. Navigate to the View tab of its Properties.
3. Change the Width parameter to 300.
4. Repeat this process for the *Multi Display*.

⇨ The Table is now a 300 x 150 rectangle.

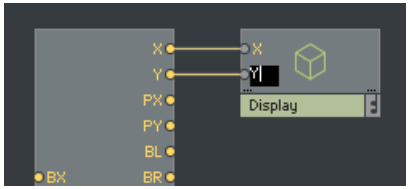


With the panel organized, now is time to connect the *Mouse Area* to the *Multi Display*:

1. Enter the Table Macro structure.
2. Connect the X and Y outputs of the *Mouse Area* to the Display Macro.



3. Rename the ports on the Display Macro to match the corresponding ports on the *Mouse Area*. You can do this by double-clicking on the ports and typing in the new names.



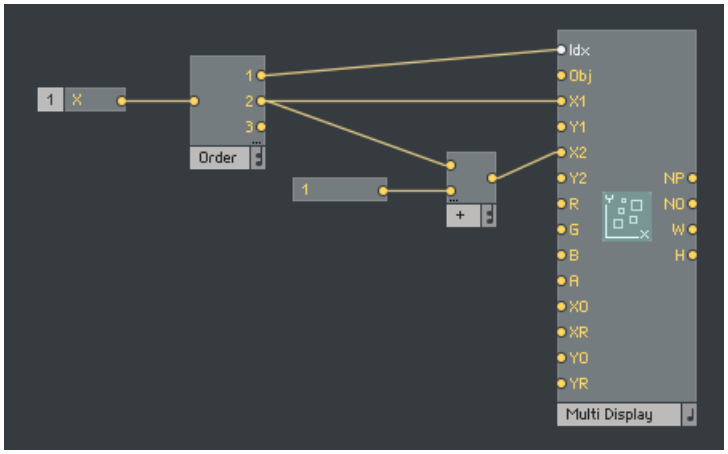
When the user interacts with the sequencer interface, the X position of the mouse will correspond to the step in the sequence, and the Y position will give the value at that step.

When using the *Multi Display*, you need to select which object you are editing by using the Idx (index) port. Once you have selected an object, you can set its parameters using the other input ports.

Therefore, the X position of the mouse should select the object index, and that the Y value will define the object height.

Using this information, you can connect the X and Y ports to the various input ports on the *Multi Display* Module:

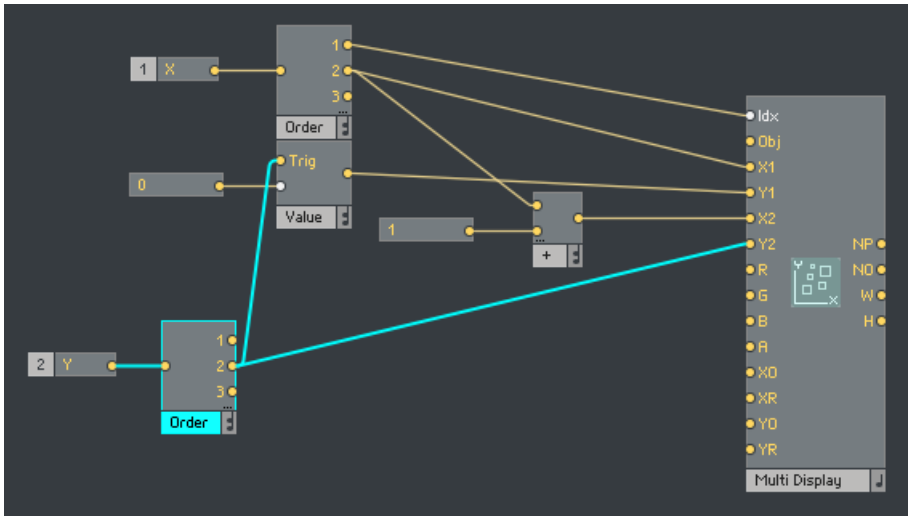
1. Enter the Display Macro.
2. Connect the X port to the *Order* Module. The *Order* Module ensures that the object index is set before the X co-ordinates are set.
3. Connect output 1 of the *Order* Module to the Idx (index) input of the *Multi Display*.
4. Connect output 2 to the X1 input
5. Create an *Add* Module with a *Constant* of 1 attached to the bottom input.
6. Wire the Modules as follows:



With the Modules connected in this way, the index will be selected, and then an object will be drawn from X1 to X2. X2 will be the same as X1 of the next object, so the bars should be placed next to each other.

The Y1 position will always be the center of the display, and therefore should always have a value of 0. Y2 will define the height of the bar.

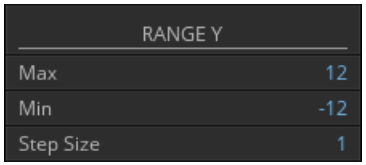
1. To ensure the Y values arrive after the Idx value is set, you will need to create another *Order* Module.
2. Create *Value* Module. When the *Value* Module receives an event at the Trig input, it outputs a new event with a value defined at the lower input.
3. Create a *Constant* and set it to 0.
4. Connect the Modules as follows:



You will notice that you have used the second output of the *Order* Module for the Y values. This is to ensure that the Y values are also processed after the Index is set.

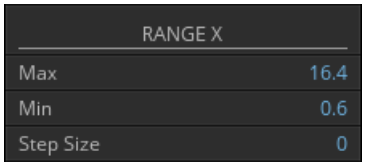
With everything wired up, you will need to set the properties of the *Mouse Area* and the *Multi Display* to produce the desired results.

1. Enter the *Mouse Area* properties.
2. The Y range will set the value of each step, which you'll want to be -12 to +12. So in the RANGE Y section, set the Max value to 12 and the Min value to -12.
3. Set the Step Size in the RANGE Y section to 1.

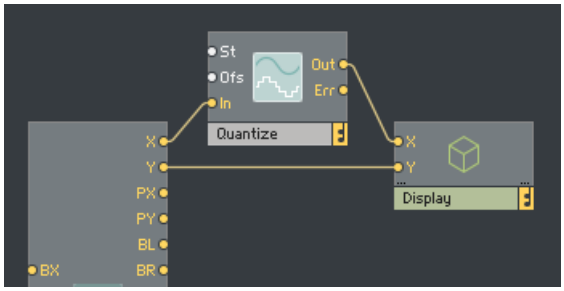


The X range will select the bar in step in the sequencer, so you will want a range of 1 to 16. However, this will not align with the display, as and X value of 1.5 will round up to 2 and select the next step even though you are still in the first section of the *Mouse Area*.

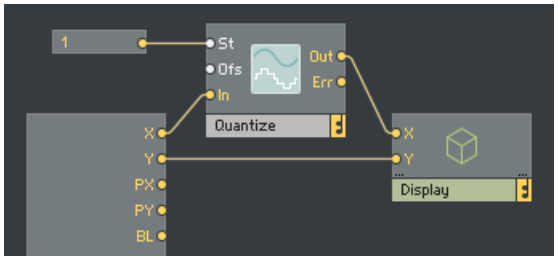
1. To compensate for this, set the Min value of the RANGE X section to 0.6 and the Max value to 16.4. Leave the Step Size at 0.



2. Then enter the Table structure and create a *Quantize* Module.
3. Connect the *Quantize* Module like so:



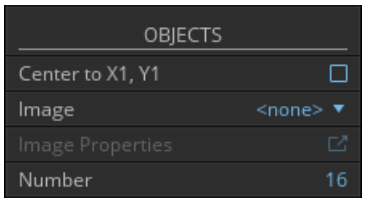
4. The *Quantize* Module needs to round the X values to the nearest whole number, so set the St (steps) to 1 by connecting a *Constant* to that port.



Now that the *Mouse Area* properties are defined, it is time to edit the *Multi Display*:

1. Enter the Display Macro and select the *Multi Display* to show its properties.

2. For the Table you'll need 16 objects in the display (one for each bar), so locate the OBJECTS section and set the Number to 16.

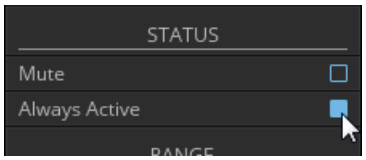


3. In the RANGE section, set the X Origin to 1 and the X Range to 16.
4. Set the Y Origin to -12 and the Y Range to 24 (the minimum value is -12 and the maximum is 12, this is why the range is 24).



The final step here is very important, and is a common stumbling block for first time builders who are unfamiliar with REAKTOR.

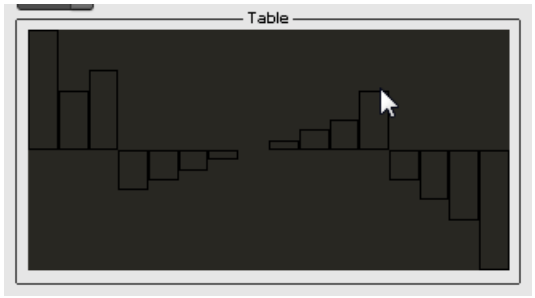
1. Check the Always Active option in the Multi Display properties.



Checking this option ensures that the *Multi Display* will function, even though none of its outputs are connected to anything.

It is now time to return to the Panel and check your work.

1. Click and drag the mouse on the table to draw in a sequence.



The bars of the Table are black outlines, which are not ideal, but you can change this easily:

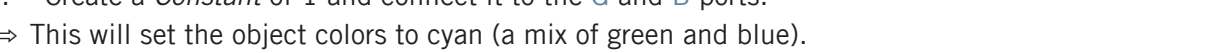
1. Return to the *Multi Display* Module in the structure.
2. The *Multi Display* has 3 inputs for setting the object color, labeled R, G, and B for Red, Green, and Blue. They all accept values between 0 and 1.
3. Create a *Constant* of 1 and connect it to the G and B ports.

⇨ This will set the object colors to cyan (a mix of green and blue).

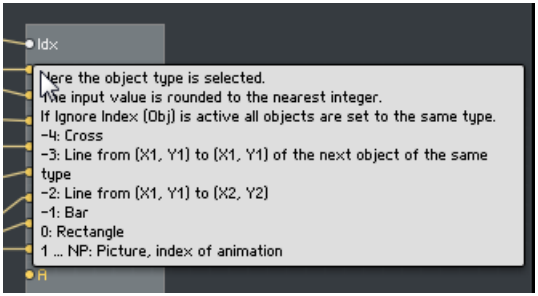
The *Multi Display* also has an input for setting the object type (Obj). Without anything connected to this input, the current object is a Rectangle.

1. Create a *Constant* with a value of -1.
2. Connect this to the Obj port of the *Multi Display*.

⇨ The objects are set to solid bars.

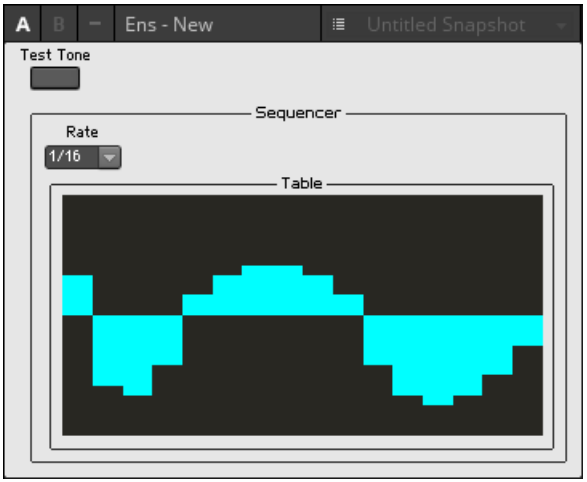


You can use Info Hints on ports to see the available options and value ranges. Using Info Hints on the Obj port gives you a list of the available object types and their corresponding values.



Using Info Hints on the Obj Port

Return to the Panel and take another look at the Sequencer.



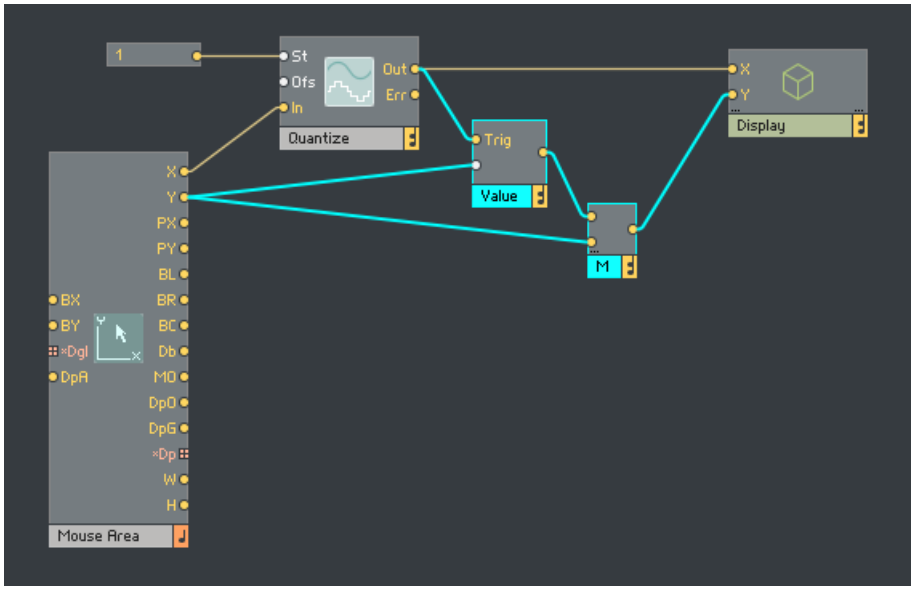
The Current Look of Your Sequencer

Before you move on to the next section, you need to do a little optimization of your sequencer. At the moment, dragging the mouse along the X axis, without moving in the Y axis, doesn't update the values of the steps of the sequencer; this is because the *Mouse Area* is not sending any events from the Y output, since the Y value is not changing.

You can fix this using a *Value* Module and a *Merge* Module:

1. Enter the Table Macro structure.
2. Create a *Value* Module and a *Merge* Module.

3. Connect these Modules to the structure like this:



⇨ Now the Y value will send an event when either it changes or when the X value changes.

6.2.3 Storing Values in a Snapshot

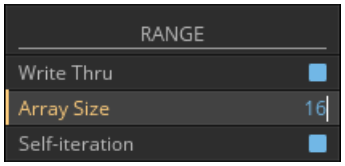
The sequencer you have created looks good and the interface is working, but it does not function like a sequencer yet.

Before you get the sequencer running, you will need a way of storing the values of the sequencer in a Snapshot.

REAKTOR includes a *Snap Value* Module that stores a value when a Snapshot is saved, and outputs that value when the Snapshot is recalled. For the Sequencer you will need to store 16 values in the snapshot, and this requires a *Snap Value Array*, which has similar functionality to the *Snap Value* Module, but can store an array of values, rather than just one.

1. Enter the Table Macro structure and create a *Snap Value Array* Module.
2. Open the *Snap Value Array*'s properties.

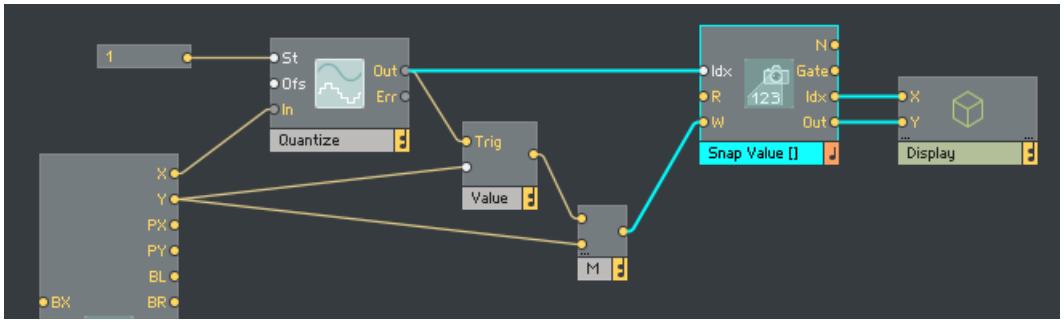
3. In the RANGE section, set the Array Size to 16.



⇨ You now have a *Snap Value Array* that can store 16 values.

The *Snap Value Array* needs to store the values sent by the *Mouse Area*, and send them to the Display.

1. Connect the *Snap Value Array* in the structure like this:



As with the *Multi Display* Module, the *Snap Value Array* will use the X value of the *Mouse Area* to set the index (Idx), and the Y value to write (W) a value into this index.

You do not need an *Order* Module in this case because the additional Modules in the structure ensure that the X value reaches the *Snap Value Array* before the Y value.

When you set the Array Size, you may have noticed the options Write Thru and Self-iteration, which were both checked.

- The Write Thru option means that when a new value is written into the *Snap Value Array*, the index and value are automatically sent to the Module's corresponding outputs.
- Self-iteration means that the *Snap Value Array* will automatically send all of its values when a Snapshot is recalled, or when the Ensemble is initialized.

For this sequencer, both of these options should remain checked, as they allow the Display to update when a Snapshot is loaded and when the user edits the values of the sequencer from the interface.

6.2.4 Adding Playback

The Sequencer is nearly there: the interface is in place and its values can be stored and recalled with Snapshots.

However, the sequencer is neither connected to the clock, nor is it outputting its values to the Sine Oscillator.

Unfortunately, you cannot connect the clock directly to the *Snap Value Array*, as the single Idx port will cause it to interfere with the events coming from the *Mouse Area*. So, to add playback, you'll need to create a new *Event Table*.

This *Event Table* will not be accessed by the user, and is only used to temporarily hold the sequencer values for the clock to trigger. Accordingly, you will need to remove the *Event Table* from the panel.

1. In the Table Macro, create an *Event Table*.
2. In the *Event Table*'s properties, navigate to the View tab.
3. At the bottom of the tab, in the VISIBLE section, uncheck the On option.

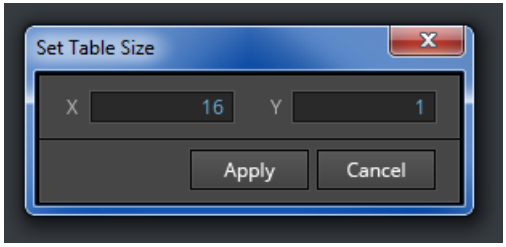
With the Event Table now hidden, you need to set it up to hold 16 values with a range of -12 to +12:

1. In the *Event Table* Properties, locate the SIZE section in the Function tab.
2. Click on the Set button.



3. A window will open, asking you to set the table size for the event table.

4. You will be creating a 16 step sequencer, so set the X value to 16 and leave the Y value at 1.



5. Click Apply.
6. Navigate to the VALUE RANGE section in the Function tab.
7. Set the Max value to 12 and the Min value to -12.

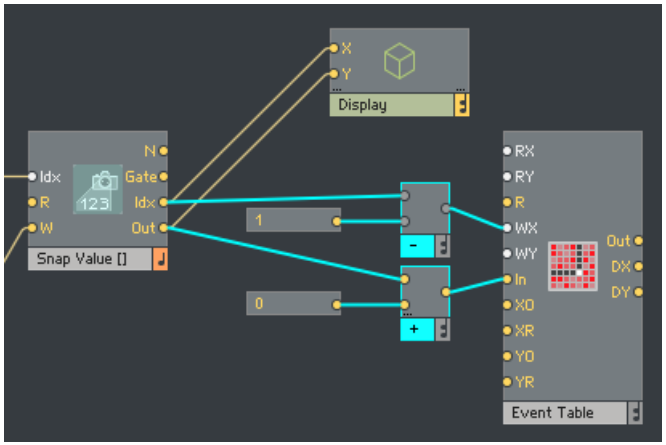
The *Snap Value Array* and the *Multi Display* use 1 indexed arrays, this means that the first array value is accessed with an index of 1. The *Event Table* uses a 0 based indexing system, so the first value is accessed with an index of 0.

When you connect the *Event Table* to the *Snap Value Array*, you will need to compensate for this.

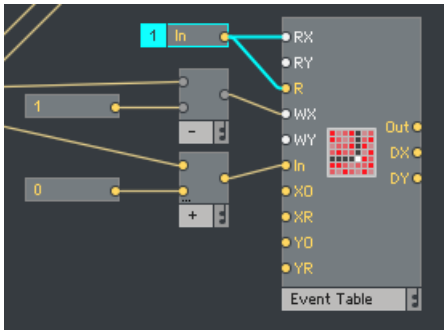
1. Create a *Subtract* Module and connect it to a *Constant* of 1 to its lower input.
2. Create another *Add* Module and connect it to a *Constant* of 0.

The -1 function will be used to adjust the index from the *Snap Value Array* to the *Event Table*. The +0 function is a dummy function used to ensure that the indices and the values have the same number of modules between the *Snap Value Array* and the *Event table*. This means that their timing will remain consistent.

1. Wire these Modules into the structure as follows:

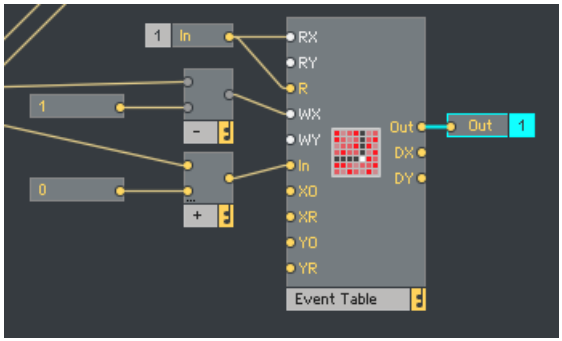


2. Next, you can connect the clock so that it reads values from the *Event Table*.



An *Order* Module is not needed here. When the RX and R ports receive events at the same time, as they will do here, the *Event Table* automatically applies them in the correct order.

1. Finally, connect the *Event Table* to the *Out* port of the Macro.

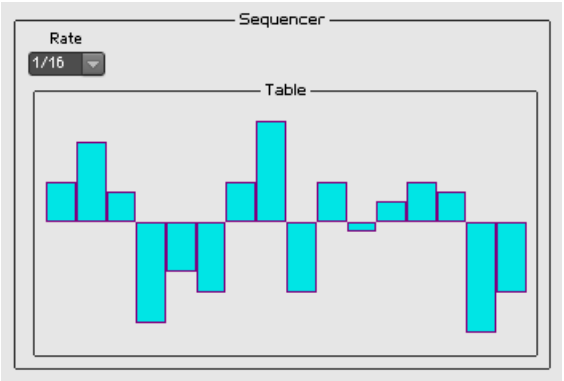


Congratulations! With the *Event Table* connected, the sequencer will be fully functional. You can now return to the Panel and use it as before.

6.3 Going Further

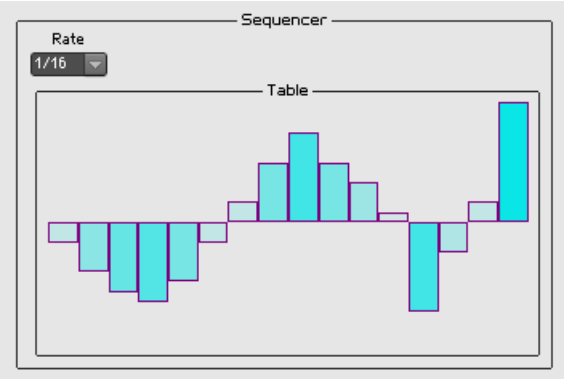
Although you only created a series of solid bars, the techniques you have learnt will allow you to create your own interface elements, with their own unique look.

For example, by using a transparent image as the background of a *Multi Display*, you can layer multiple *Multi Displays* to create more complex interfaces:



An Interface created by layering Multi Display Modules.

You can also access the color and opacity of individual objects in the *Multi Display* to create more responsive displays.



An Interface created by layering Multi Display Modules and changing the opacity of individual objects.

6.3.1 Related Factory Content

Although they tend to use *Poly Display* Modules rather than *Multi Displays*, any of the Ensembles from the Sequencers section of the Factory Library will use techniques similar to the ones you have learnt in this tutorial.



The SQX Ensemble

7 Additive Synthesizer

7.1 Overview

Using event signals in REAKTOR not only lets you control their timing, but also allows you to use a process called iteration.

> The idea of iteration was introduced in the previous tutorial with the *Snap Value Array* and its Self-iteration property.

For those of you with previous programming experience, iteration can be thought of as REAKTOR's version of a loop (with the *Iteration* Module functioning like a For Loop).

In REAKTOR, iteration can be used to generate a large number of events that are processed simultaneously, but are treated as if they came one after the other. This technique is very useful when using Modules which behave like arrays (like the *Multi Display* seen in the previous tutorial, ↑5, Basic Step Sequencer ↑5, Basic Step Sequencer)

This tutorial will teach you about iteration using the example of building an additive synthesizer.

7.1.1 Previous Knowledge

As well as assuming a basic understanding of building in REAKTOR, this tutorial also assumes that you have completed tutorial ↑5, Basic Step Sequencer ↑5, Basic Step Sequencer.

The following will not only build on the Ensemble created in that tutorial, but will also assume all knowledge acquired to that point.

7.1.2 Theory

An additive synthesizer creates sounds by combining large numbers of sine waves with different frequencies, amplitudes and phases. In the context of an additive synthesizer these sine waves are called partials or harmonics (throughout the rest of the tutorial they will be referred to as partials).

REAKTOR offers a Module created specifically for this purpose: the *Sine Bank*.

Rather than creating a large number of individual *Sine Oscillator* Modules, this one Module allows you to generate many sine waves with only a single Module. It is also optimized for this function, and is thus a more CPU-friendly option.

The *Sine Bank* Oscillator is at the heart of REAKTOR synthesizers LAZERBASS and RAZOR.

The best way to control the *Sine Bank* is to use iteration, as it can be used to set the properties of many sine waves in a single process.

For this tutorial, you will be guided through the process of building an additive synthesizer that uses 64 partials, the amplitudes of which are edited from the interface via a table similar to the one built in tutorial ↑5, Basic Step Sequencer ↑5, Basic Step Sequencer.

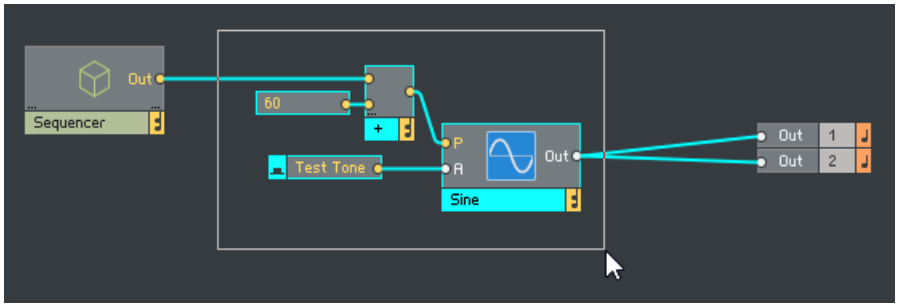
7.2 Tutorial

7.2.1 Setting Up

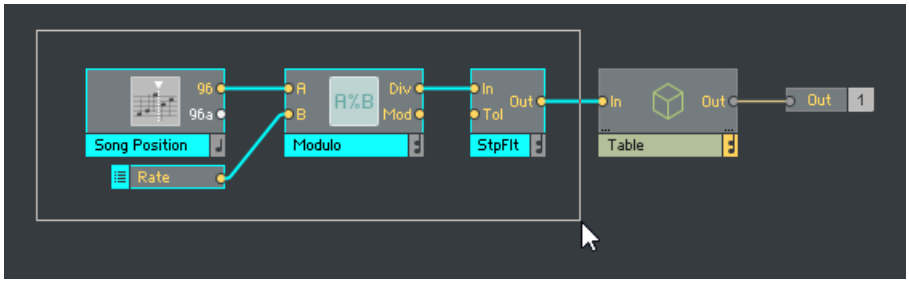
This tutorial will build on the ↑5, Basic Step Sequencer, as it already uses many of the Modules you will need. However, it also has a lot of Modules that you will not need, so the first thing to do is delete them:

1. If you do not have the Step Sequencer Ensemble loaded, open it now.

2. Enter the structure and delete the Modules highlighted below:

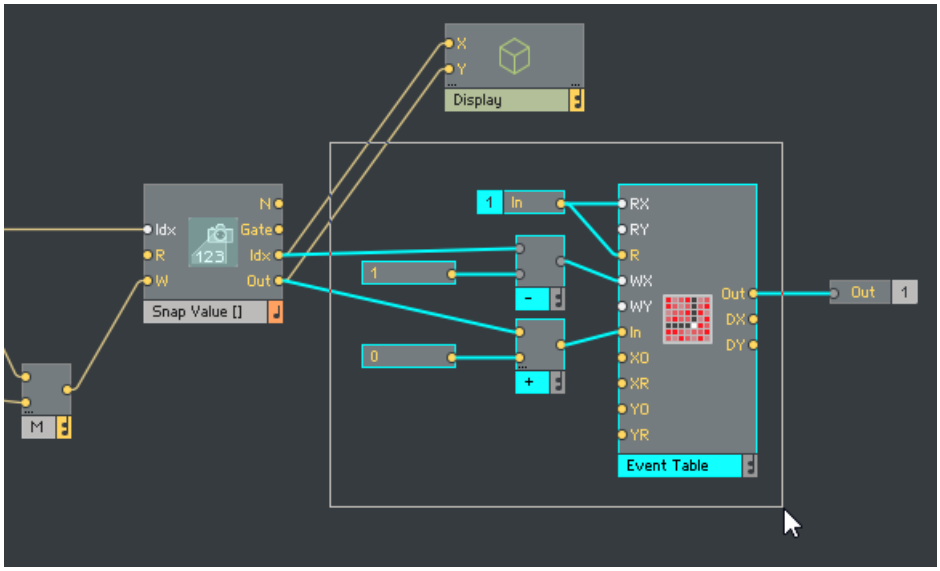


3. Enter the Sequencer Macro and delete all of the Modules that are used to generate the clock signal.



4. Rename the Sequencer Macro to "Additive Synth".

5. Enter the Table Macro and delete the *Event Table* and the Modules highlighted below:

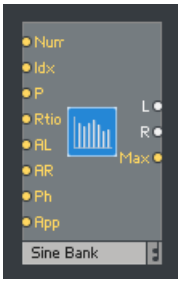


6. Rename the Table Macro to "Partials".

⇨ The Ensemble is now ready to be converted to an additive synthesizer.

7.2.2 The Sine Bank Oscillator

The *Sine Bank* can be a daunting Module, but it works on the same principals as the *Multi Display*, only with audio partials in place of visual objects.



The Sine Bank Module

Like the *Multi Display*, you need to select a partial using its index (Idx). With the index selected you can then set a variety of parameters for that partial. In the case of the sine bank, the available parameters are:

- Rtio: Sets the ratio of the partial. The frequency of the partial will be the main frequency (defined by the pitch (P) input) multiplied by the number set at this port.
- AL: Sets the amplitude of the partial in the left channel.
- AR: Sets the amplitude of the partial in the right channel.
- Ph: Sets the phase of the partial.

Where the *Sine Bank* differs from the *Multi Display* is the inclusion of the App (apply) port. When you set parameters for the partials in the *Sine Bank*, the changes need to be applied by sending an event to the App port before they take effect.



Although the P (pitch) input isn't set per partial, it is also not applied until an event is sent to the App port.

So, when creating this Ensemble, you will need to set the parameters for all of the partials, and then apply these settings. The App port will also need set every time the user plays a note, in order to update the pitch of the Module.

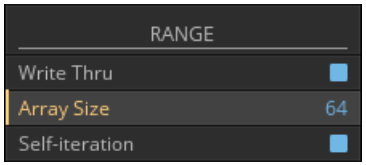
7.2.3 Building the Synthesizer

The first section of this tutorial for building the 64-partial additive synthesizer will explain how to update the table that you created in the ↑5, Basic Step Sequencer tutorial from 16 bars to 64 bars.

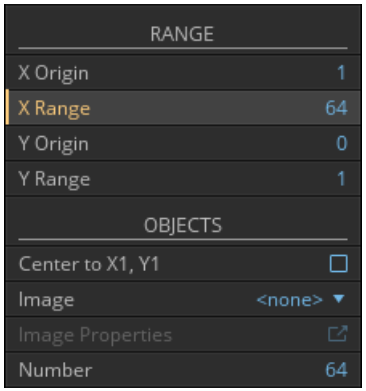
1. Select the *Mouse Area* Module to display its Properties.
2. Set the Max parameter in the RANGE X section to 64.4



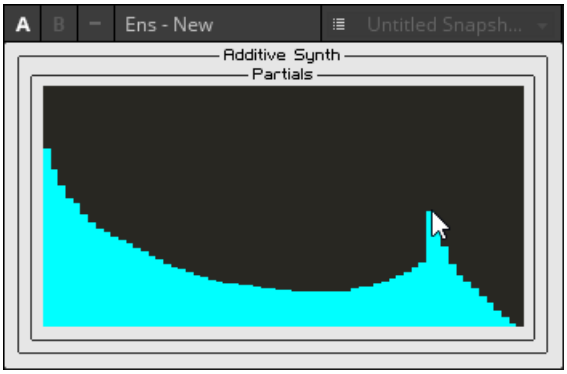
3. Select the *Snap Value Array* Module to display its Properties.
4. Set the Array Size to 64



5. Enter the Display Macro and select the *Multi Display* to show its Properties.
6. In the OBJECTS section, set the Number parameter to 64
7. In the RANGE section, set the X Range to 64



If you now return to the Panel view, you will see that the table has many more bars in it, but still behaves as before.



The Table now with 64 Bars

With the interface updated, you can now connect this to the *Sine Bank* Module.

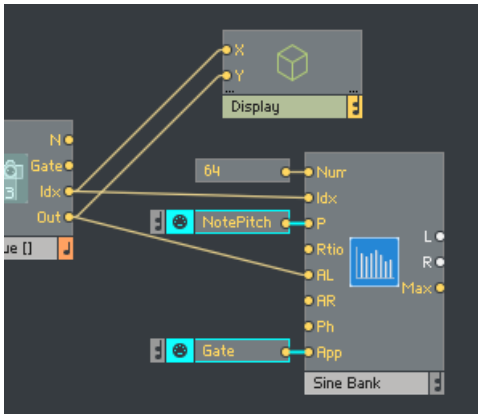
1. In the structure view, navigate to the Partials Macro.
2. Create a *Sine Bank* Module.
3. Create a *Constant* of 64 and attach it to the Num input.
4. Connect the *Snap Value Array* to the *Sine Bank* as shown below.

You only need to connect to the AL input as the additive synthesizer you are creating will be mono, and therefore only requires one output channel.

In order to make the *Sine Bank* playable, you'll need to add pitch and gate MIDI inputs.

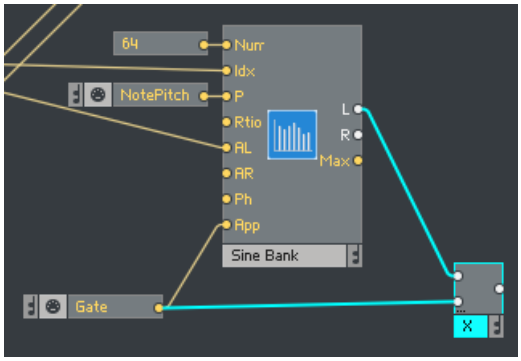
1. Create a *Note Pitch* (MIDI In) Module and a *Gate* (MIDI In) Module.

2. Connect these Modules to the *Sine Bank* as shown below.



Now the MIDI pitch input will define the pitch of the *Sine Bank*, and the MIDI gate will apply the settings. However, you still need a way of controlling the output with velocity.

1. Create a *Multiply* Module.
2. Use it to multiply the *Sine Bank* output with the Gate output, as illustrated below.



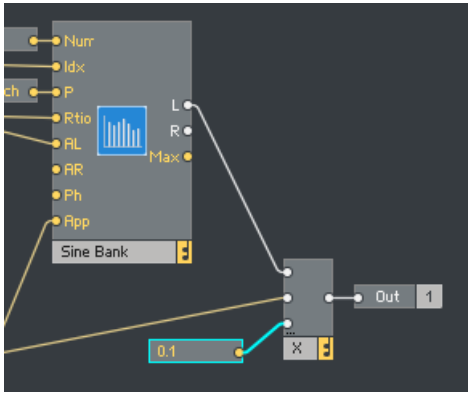
3. Attach the output of the *Multiply* Module to the Macro *Out* port.

⇨ When you play a MIDI keyboard, you should now be able to hear the output of the Sine Bank.

Return to the Panel and edit the table of partials. Each time you play a new note, you will hear the sound update according to the settings of the table.

The Output of this synthesizer may be quite loud, so you should adjust the output level of the synthesizer accordingly:

1. Create a Constant of 0.1 and connect it to the Multiply Module at the synthesizers output.

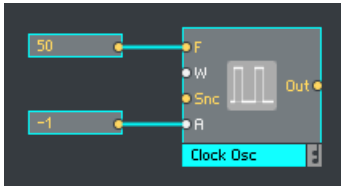


7.2.4 Adding Real-time Updates

The additive synthesizer you have built has one flaw: its sound does not update in real-time, it only updates when the user plays a new note.

To resolve this, you can add a clock that triggers the App port at regular intervals while the user's mouse is interacting with the interface.

1. Enter the structure and navigate to the Partials Macro.
2. Create a *Clock Oscillator*.
3. Create *Constants* at the F and A ports.
4. Set the *Constant* at the F Port to 50.
5. Set the *Constant* at the A Port to -1.

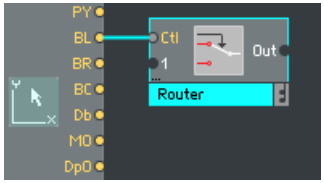


⇨ You now have a clock that will oscillate between -1 and 0 at a rate of 50 times per second.

The reason you want the clock value to be negative is that when you send an event higher than 0 to the App input of the Sine Bank, the phases of the partials will update. If the phases update while a sound is being played, then this will cause an audible click. Sending an event of 0 or lower to the App input will update all of the partial parameters except the phase, which should produce a less noticeable transition.

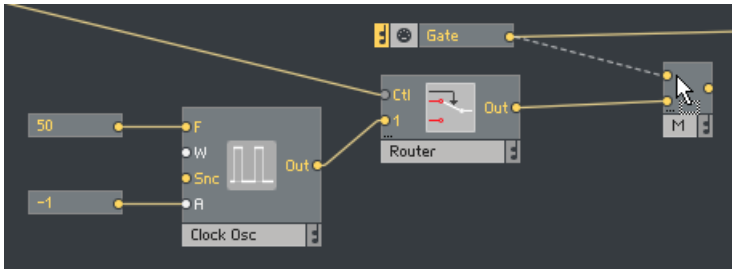
Next you will create a way to make sure the clock only updates the *Sine Bank* while the user's mouse is pressed on the table.

1. Create a *Router 1,2* Module.
2. Connect the BL output of the *Mouse Area* to the Ctl input of the *Router*.



The *Router* Modules act as switches that control the flow of signals. With the BL port of the *Mouse Area*, which outputs the pressed state of the left mouse button, connected to the Ctl input of the *Router*, the signal from the 1 input will only be sent to the output when the left mouse button is pressed.

1. Connect the *Clock Oscillator* to the 1 port of the *Router*.
2. Create a *Merge* Module.
3. Connect the output of the *Router* to the input of the *Merge* Module.
4. While holding the [Ctrl]/[Cmd] key, connect the *Gate* Module to the *Merge* Module in such a way that it creates another port in the *Merge* Module.



5. Connect the output of the *Merge* Module to the App input of the *Sine Bank*.

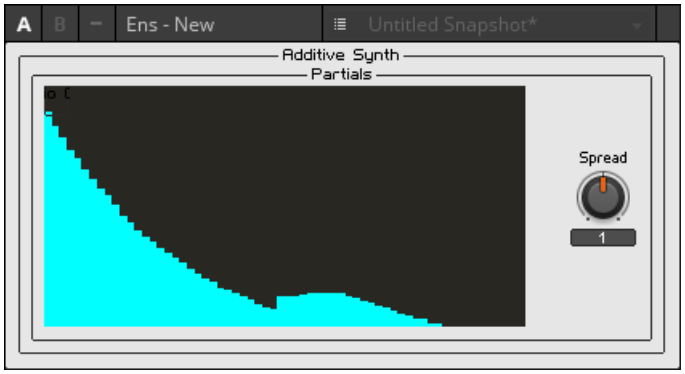
⇨ When you return to the Panel view, you will be able to hear the changes to the partials as you edit the values in the table.

7.2.5 Adjusting the Partial Ratio Spread

By now you should have an additive synthesizer that you can edit in real-time, but you have not yet needed to use iteration.

This final section will add a knob that will alter the harmonic spread of the partials. Since this single knob will need to edit the ratio parameter of all of the partials, it will require iteration to do so.

1. Enter the structure of the Partials Macro.
2. Create a *Knob* Module.
3. In the *Knob's* properties, set the Max value to 2.
4. Re-name the *Knob* to "Spread".
5. Go to the Panel view and move the Spread knob so that it is not overlapping with the table.



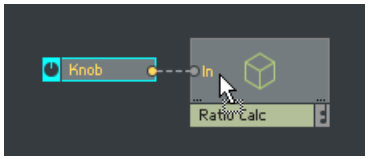
At the moment the ratios of the partials are set to their default settings, which is the Harmonic Series (partial 1 has a ratio of 1, partial 2 has a ratio of 2, and so on).

The Spread knob that you just created will be used to multiply the ratio values of the partials to control the spread of the harmonics.

The process that the knob will trigger will be rather complex. You can use a Macro to keep the structure of your synthesizer organized.

Creating the Ratio Calculation Macro

1. Create a New Macro.
2. Re-name it to "Ratio Calc"
3. While holding the [Ctrl]/[Cmd] key, connect the Spread knob to the Macro in such a way that it creates an input port in the Macro.



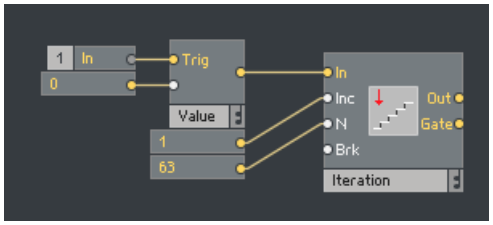
4. The Ratio Calc Macro will not have any controls, so it can be hidden from the Ensemble's Panel. Click on the Ratio Calc Macro to display its properties.
5. Click on the View Tab.
6. In the VISIBLE section, uncheck the On option.



The Iteration Loop

The process that needs to occur in the Ratio Calc Macro each time the user uses the Spread knob is a loop that generates values from 0 to 63, these values will then access the indices of the partials of the *Sine Bank*, and then also be used to calculate the ratios of the partials. You can build this with an *Iteration* Module:

1. Enter the Ratio Calc Macro.
2. Create an *Iteration* Module and a *Value* Module.
3. Connect these Modules with some additional *Constants* as illustrated below.



What will happen in the structure you have just created is the following:

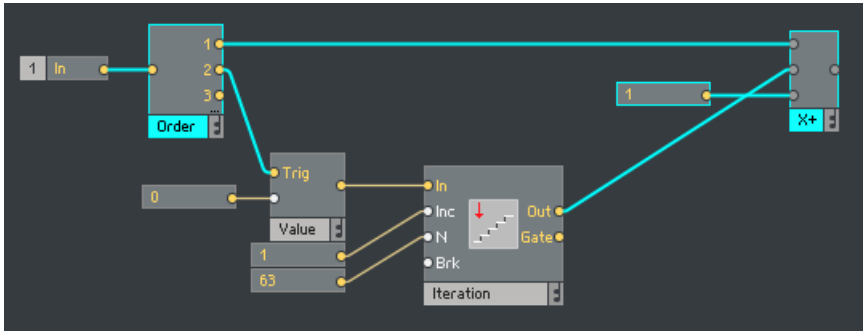
- When the Spread knob sends an event into the Macro, the *Value* Module will output an event with a value of 0.
- This event will trigger the *Iteration* Module to start its process.
- The iteration process will create 63 additional events (specified by the *Constant* at the N port).
- Each additional event will be of a value 1 higher than the last event (specified by the *Constant* at the Inc port).

So the *Iteration* Module has effectively become the loop that generates the values of 0 to 63 every time the knob is moved.

The Ratio Calculation

Now you can add the harmonic ratio spread calculation:

1. Create the structure illustrated below.



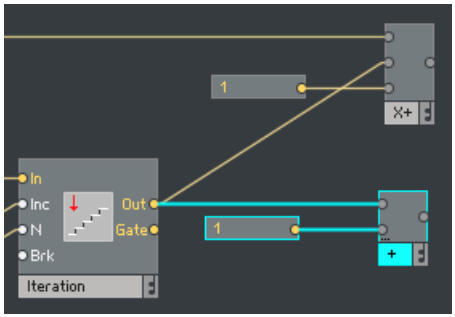
The calculation is very simple; it only needs one *Mult/Add* Module. The power of this process is that the calculation happens 64 times at an effectively instantaneous rate.

Note that you also need to add an *Order* Module. This ensures that all of the values coming from the *Iteration* Module are processed with the same value from the Spread knob.

Calculating Partial Indices

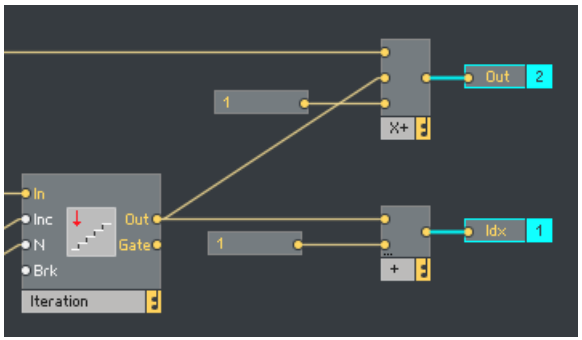
An additional calculation is needed to send the corresponding indices for the partials:

1. Add the following Modules to your structure and connect them as illustrated below:



The final step in this Macro will be to create the outputs needed to send these values to the *Sine Bank*.

1. Create 2 *Out Port* Modules.
2. Re-name the first one "Idx".
3. Connect the *Out Ports* to the structure as follows:

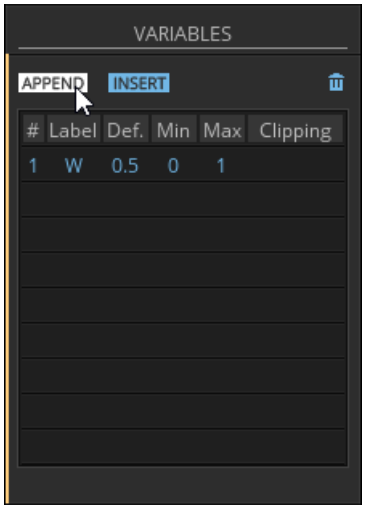


Snapshot Storage

The values generated by the Spread knob will be stored in the Snap Value Array as another variable.

1. Navigate to the Partials Macro.
2. Select the *Snap Value Array* to display its Properties.

3. In the VARIABLES section, click the APPEND button.

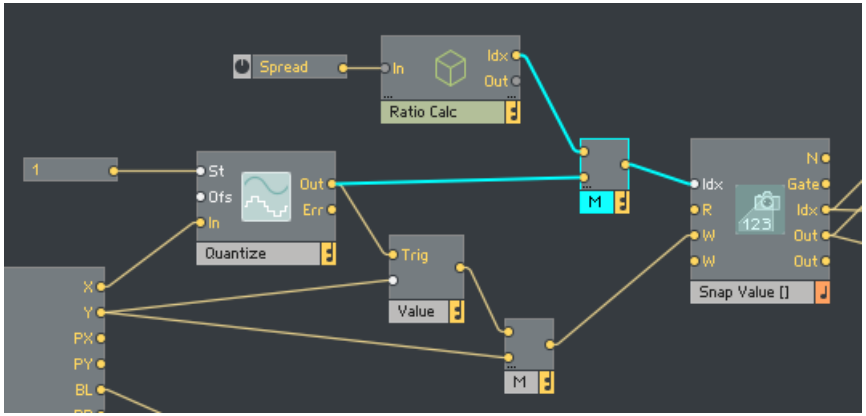


⇨ A new variable has been created and a new W input has been added to the *Snap Value Array* Module, coupled with a new Out port.

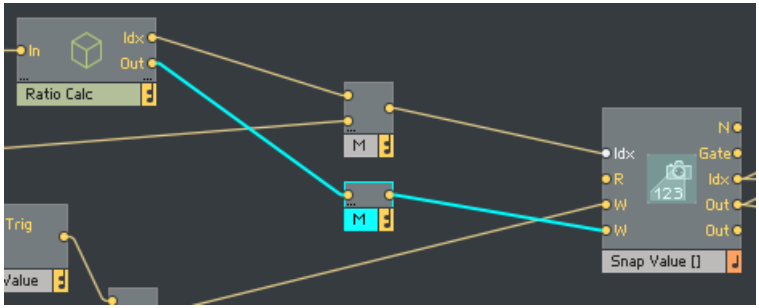
Connecting the Ratio Values to the Synthesizer

The next step is to connect the Ratio values that are generated in the Ratio Calc Macro to the *Sine Bank* via the *Snap Value Array*.

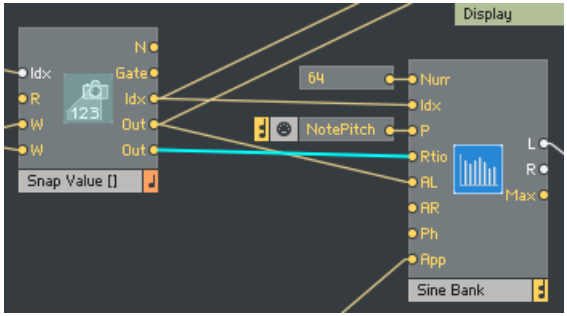
1. Connect the Idx output of the Ratio Calc Macro to the Idx input of the *Snap Value Array*, using a *Merge* Module to combine it with the current connection.



2. Connect the Out port of the Ratio Calc Macro to the second W input of the *Snap Value Array*, using a second *Merge* Module as a dummy to keep the event timings correct.



3. Connect the second Out port of the *Snap Value Array* to the Rtio input of the *Sine Bank* Module.

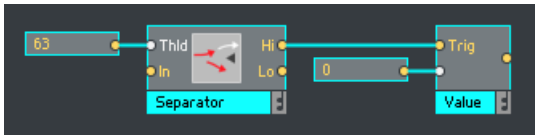


Applying the Ratio Setting Changes

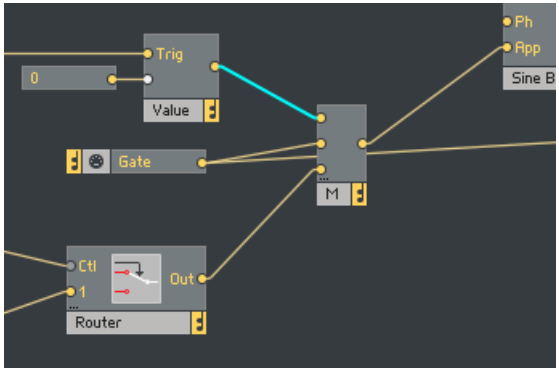
There needs to be an event sent to the App port of the *Sine Bank* in order for these changes to be heard in real-time.

The apply event should happen after the iteration has taken place. You can use a *Separator* Module to split the Idx signal from the Ratio Calc Macro so that the final Idx event from the *Iteration* Module triggers a 0 valued event to the App port.

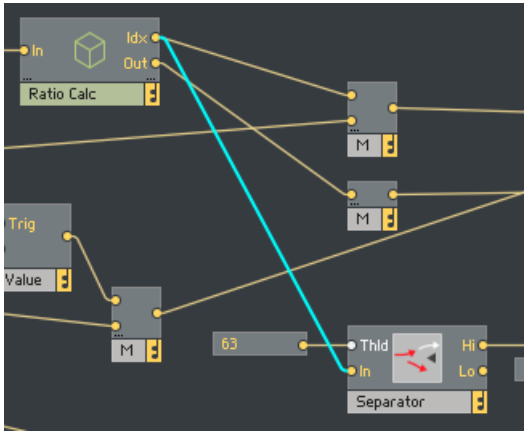
1. Create a *Separator* and set the threshold (Thld) to 63.
2. Create a *Value* Module and attach a *Constant* of 0 to its lower input.
3. Connect the Hi output of the *Separator* to the Trig input of the *Value* Module.



4. Connect the output of the *Value* Module to the *Merge* Module that leads to the *Sine Bank*'s App port.



5. Connect the Idx output of the Ratio Calc Macro to the In port of the *Separator*.



⇨ When the final index value (64) comes from the *Iteration* Module, it will also trigger an event with a value of 0, which will then apply the ratio changes in the *Sine Bank*.

Congratulations! You have created an additive synthesizer with controls for setting the amplitude of each partial, and for setting the harmonic spread of the partials, and all in real-time.

7.3 Going Further

The *Modal Bank* Filter Module shares many similarities with the *Sine Bank*. It has many of the same input ports and shares the same input paradigm; it simply creates a bank of resonant band-pass filters rather than a bank of sine wave oscillators.

The Modal Bank is the Module at the heart of REAKTOR PRISM and MIKRO PRISM.

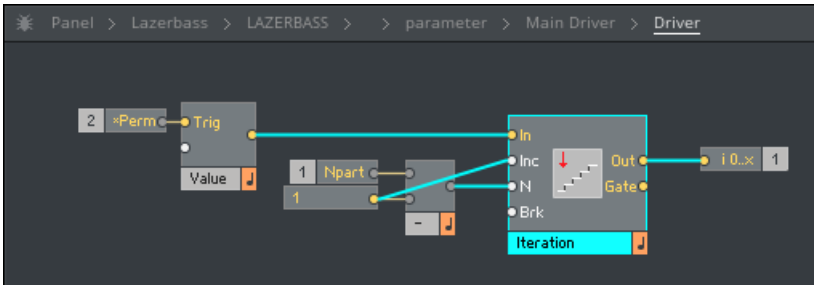
Additive synthesis and modal synthesis are both very powerful. By adding more calculations to the iteration process, you can create complex sounds that are not possible with any other forms of synthesis.

Try adapting your additive synthesizer to use the *Modal Bank*. Note that the *Modal Bank* requires an "exciter" signal (like a short click) to produce sound.

7.3.1 Related Factory Content

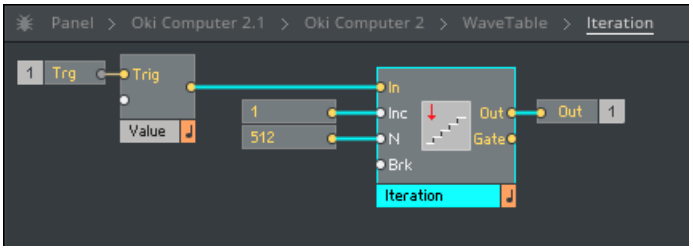
The *LAZERBASS* synthesizer uses a *Sine Bank* as its main sound generator. Almost all of the available controls on the interface apply calculations to parameters of the partials in the *Sine Bank*.

Although it is a large structure, you will be able to find *Iteration* Modules driving many of the parameter calculations.



An Iteration Module inside LAZERBASS.

As an example of how iteration can be used outside of additive synthesis, it is also used in *Oki Computer 2* to drive the calculations that create the wavetables.



The Iteration Module that drives the Wavetable creation.

8 Drag and Drop Sampler

8.1 Overview

Table References are a signal type that facilitates flexible and efficient sharing of data in the structure. Modules and Macros that make use of Table References are part of the Table Framework.



Reference information about the Table Framework can be found in section ↑16, Table Framework.

In the following tutorial, you will use Macros that are part of the Table Framework to create a drag and drop sampler with a waveform display. You will also look at the structure of these Macros to learn how Table References can be used.

This tutorial will also introduce Core Cells and how they interact with the Primary level of REAKTOR.

8.1.1 Previous Knowledge

This tutorial assumes you have read the tutorials up until this point. You should already know how to navigate the Structure View and should be familiar with Event processes.

8.1.2 Theory

A table is a two-dimensional array of data, and Table References allow you to access this data anywhere in the structure. These properties make the Table Framework ideal for working with samples.

Table References exist in the Primary level of REAKTOR, but are utilized by Core Cells.

In this tutorial, the Table Reference will be an audio sample. The Ensemble you create will use the Primary structure to load the sample, which will then have its data sent to Core Cells for playback and to calculate the display.

8.2 Tutorial

8.2.1 The Drag and Drop Area

To start, you will need a blank canvas:

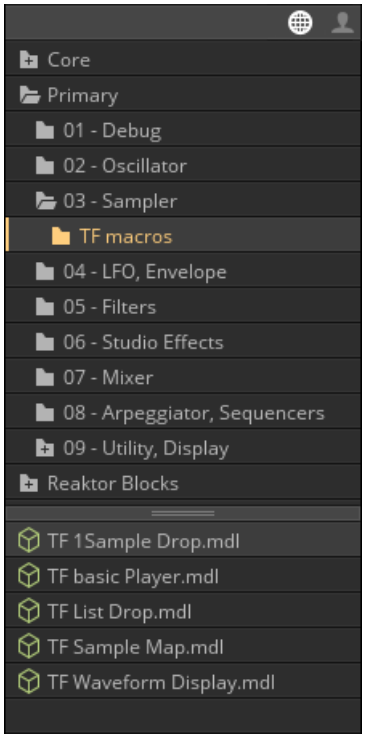
1. Create a new Ensemble.

All of the Library Macros that make use of the Table Framework can be found in the **03 - Sampler** section of the Library. In this section is a sub-section called **TF-macros** that contains a number of building-blocks for creating your own samplers.

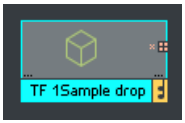
As a first step, you will create the Drag and Drop Area:

1. Open the Library Browser in the Side Pane.
2. Open the Primary folder, followed by the 03 - Sampler folder.

3. Click on the TF macros folder to display its contents in the lower half of the Side Pane.



4. Drag the *TF 1Sample Drop* Macro into the Structure.



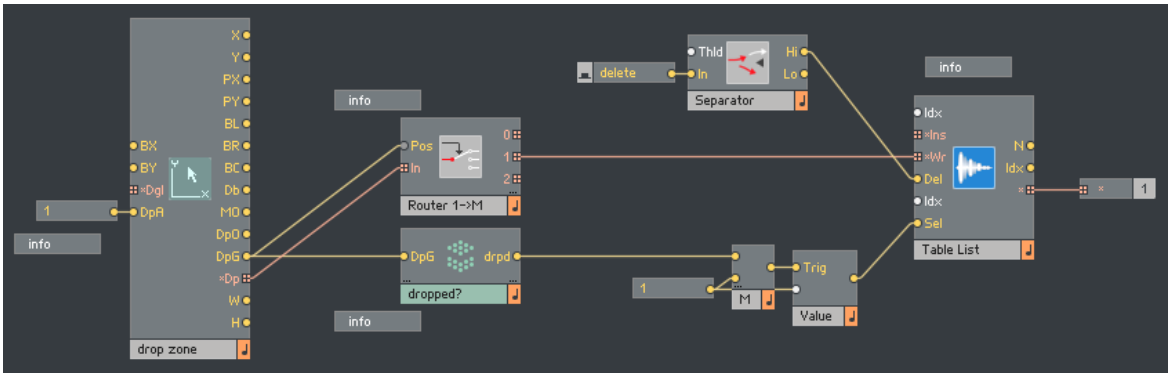
The TF 1Sample drop Macro

Looking at the Macro you have just loaded, you will notice a new port type. This kind of port is a Table Reference port and is used to send Table References (in this case audio sample data) around the Structure.

Table References do not send signals with single values in the same manner as Audio or Event ports, but rather send a reference to a table of data.

If you think of a sample as a table of data that is store in RAM, the Table Reference is a pointer to that data.

1. Enter the *TF 1Sample drop* Macro by double-clicking on it.



The Structure of the TF 1Sample drop Macro

In the Structure of the Macro you will notice a number of Modules called info. These Modules contain information regarding the Modules to which they are closest.

To view the information in these Modules:

1. Click on the Module to select it.
2. Open the Properties View in the Sidepane.
3. Click on the Info tab.

⇨ The information in the Module will be displayed in the Properties.



While the Info tab is open, you can click on other Modules and the information will update accordingly.

As a brief overview of the Macro there are three important Modules:

The Mouse Area

The *Mouse Area* Module (renamed here to drop zone) is used to define the area where the user can drop an audio file.

- The 1 at the DpA input tells the Module that it should accept audio files for drag and drop.
- The *Dp (Drop) output sends the Table Reference whenever an audio file is dragged over the Mouse Area and whenever it is dropped.
- The DpG (Drop Gate) output is used to send the drag and drop state of the audio file.

The values from the *Mouse Area* are then sent to the next important Module...

The Router 1->M

Since the *Dp output sends Table References when an audio file is dragged over the *Mouse Area*, as well as when it is Dropped, the *Router 1->M* Module is used to control which *Dp Table Reference is sent to the rest of the Structure.

1. The *Router* only allows a *Dp Reference through when the value of the DpG is at 1.
2. The DpG outputs a 0 value generally, a 2 when an audio file is dragged over it, and a 1 when it is dropped.
3. Thus, the *Router* blocks the *Dp References until the file is definitely dropped on the *Mouse Area* by the user.

The Table List

The *Table List* is the Module that holds the Table data.

As the name suggests, it can hold several Tables in a list. The different Tables are selected using indices. But in this instance, it only holds a single Table, so the indices are left blank.

- The Table Reference is sent from the *Router* into the *Table List*, which stores it.
- A Core Cell (named dropped?) is used to trigger the output of the Table Reference from the *Table List* by sending a value to the Sel input of the *Table List*.
- The Table Reference is then sent out of the Macro and into the rest of the Structure.
- Note that an Event will also be sent to the Sel input during initialization because of the *Constant* attached to the *Merge* Module. This will ensure that the Table Reference will be sent to the Structure during initialization.

8.2.2 The Waveform Display

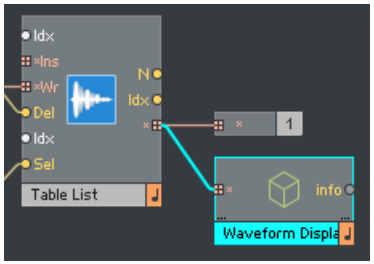
For the purposes of layering, you need to place the Waveform Display Macro inside the drop Macro.

1. Insert the *TF Waveform Display* Macro into the Structure of the *TF 1Sample drop* Macro.



The *TF Waveform Display* has an input for Table References. It then uses the Table data to draw a waveform using a *Multi Display*.

1. Connect the Table Reference output of the *Table List* to the Table Reference input of the *Waveform Display* Macro.



The Structure of the *Waveform Display* is more complicated than the *1Sample drop*, but the basic theory is that when a Table Reference is sent to its input, an iteration process is triggered. This process reads the data in the Table which is then converted it into signals that can be used by a *Multi Display* Module.

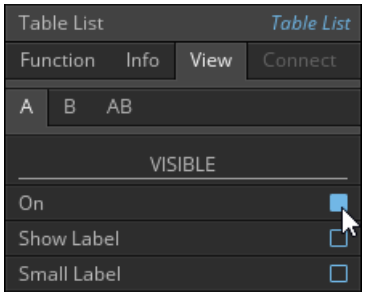
This is the same basic principal behind the Step Sequencer from section ↑6, Advanced Step Sequencer.

Next, you need to align these Macros on the Panel.

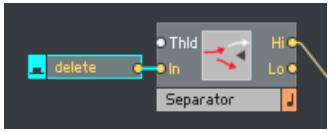
You do not need the *delete* button or the Table List on the Panel, so they should be hidden.

1. Select the *Table List* Module.
2. Open the View tab of its Properties.

3. Deactivate the On option in the VISIBLE section.



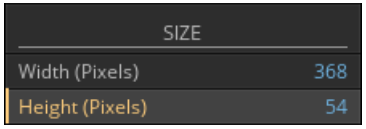
4. Repeat this process with the *delete* Button.



Resizing the Mouse Area

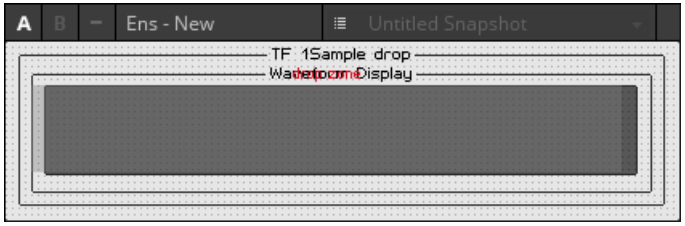
Next, you will enlarge the *Mouse Area* so that it covers the whole Waveform Display.

1. Select the *Mouse Area* Module.
2. Open the View tab of its Properties.
3. Set the Width (Pixels) parameter to 368.
4. Set the Height (Pixels) parameter to 54.



⇨ The *Mouse Area* is now the same size as the Waveform Display.

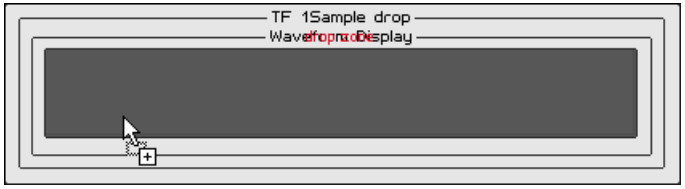
1. Unlock the Panel to see the Mouse Area as a grey shadow.



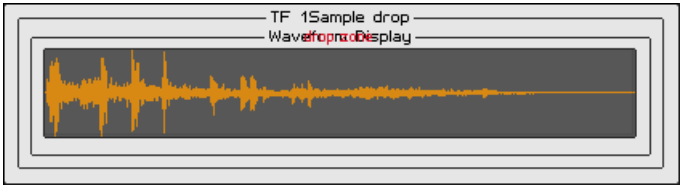
2. Move the Mouse Area until it covers the Waveform Display.
3. Lock the Panel.

The Panel elements of the Drag and Drop Sampler are complete.

1. Drag and drop an audio file (wav or aiff) onto the Mouse Area.



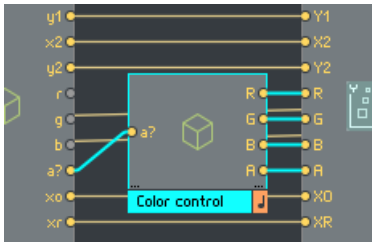
⇨ The Waveform of the audio file will be displayed on the Panel.



Changing the Waveform Color

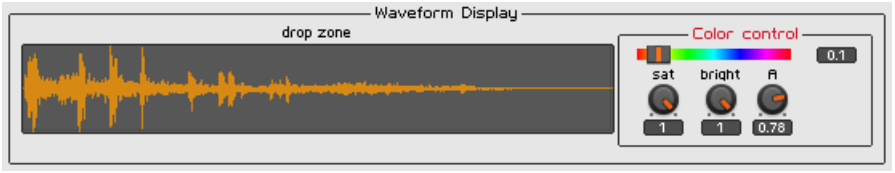
It is possible to change the color of the display by opening some hidden controls.

1. Enter the *Waveform Display* Macro Structure.
2. Select the *Color control* Macro.



3. Open the View tab of its Properties.
4. Activate the On option in the VISIBLE section.

⇨ Color controls will appear on the Panel.



1. Edit the color of the waveform using these controls.
2. When you are finished, return to the View tab of its Properties and deactivate the On option in the VISIBLE section.

8.2.3 Playback

The Sampler can load samples and it can display their waveforms, but it cannot yet play the samples. You will need one final Macro for that.

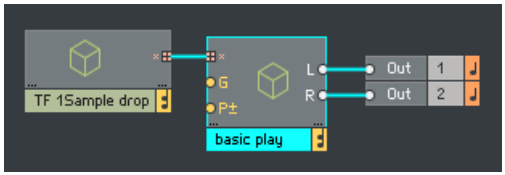
1. Navigate to the top level of the Ensemble.
2. In the TF macros sub-section of the Library, load the *TF basic Player* Macro.



The TF basic Player Macro

The Basic Player plays the audio data from a Table when it receives a gate signal.

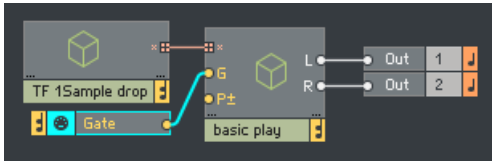
1. Connect the *TF basic Player* Macro as shown below.



⇨ The Table Reference from the Table List in the *TF 1Sample drop* Macro will feed into the *TF basic Player* Macro.

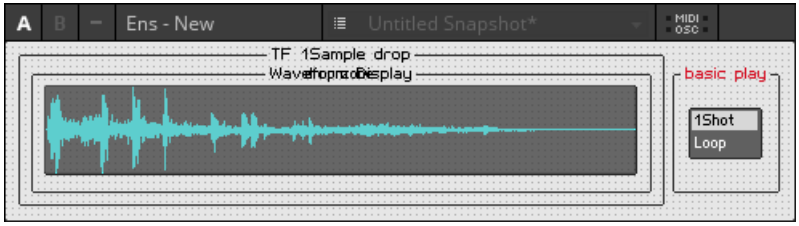
Next, you need to connect a gate to the *TF basic Player* Macro so that the user can trigger playback of the sample.

1. Create a *Gate (MIDI In)* Module.
2. Connect it to the G input of the *TF basic Player* Macro.



⇨ Play a MIDI key to trigger the sampler.

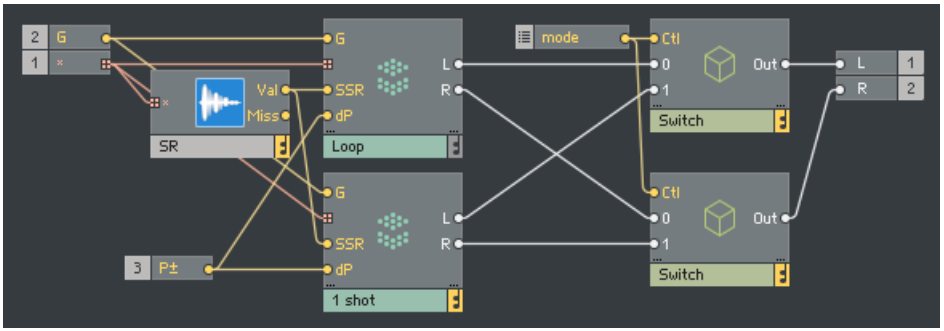
1. In the Panel View, move the basic play Macro so that it is not obscuring the Waveform Display.



Basic Player Structure

Now is a good time to take a look into the Structure of the basic Player Macro.

1. Double-click on the *basic Player* Macro to enter its Structure.



The Structure of the basic Player Macro

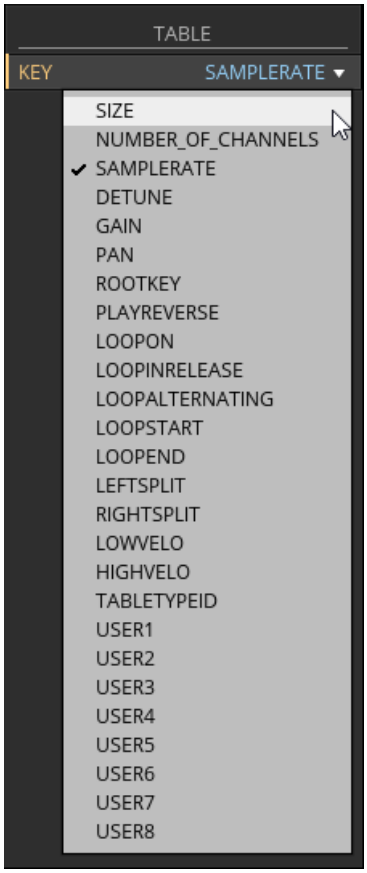
There are two Core Cells in this Structure, one called Loop and one called 1 shot, each one handling a different mode of sample playback.

This tutorial will not cover the contents of the Core Cells, their contents would be close to meaningless without some understanding of Core (read the REAKTOR 6 Building in Core document to learn about building Core Cells).

The Module that will be explored is the *Table Info* Module (renamed in this Structure to SR).

The *Table Info* Module reads meta-data from a Table and exports that data as an Event value. In this Structure, the *Table Info* Module is retrieving the sample rate (SR) of the audio file and sending that information to the Core Cells.

1. The meta-data that the *Table Info* Module retrieves is set in its Properties.



8.3 Going Further

The Table Framework allows you to manipulate audio samples in your own custom structures, but it requires Core Cells for this, and so now is a good time to jump into the Core level of REAKTOR (see section ↑15, An Introduction to Core).

You can also read section ↑16, Table Framework for more detailed information about the Table Framework.

Related Factory Content

Everything in the **03 - Sampler** section of the Library uses the Table Framework. You can load in any of the Samplers and check out their Structures.

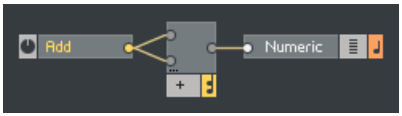
As a starting point, however, it is advised that you begin by looking into the other **TF macros**. These Macros are annotated with *info* Modules and are much easier to navigate and read.

9 A Quick Guide to Initialization

Initialization was mentioned briefly in section ↑2.5.4, Event Initialization Order ↑2.5.4, Event Initialization Order. It is the process that occurs immediately after you load an Ensemble.

Initialization is particularly important for Events signals and Event Modules, especially in complex structures. It is important to be mindful of the initialization process - an Ensemble that works fine while it is being built, may experience odd behavior when it is closed and re-loaded.

As an example, in normal circumstances the *Add* Module sends an Event when an Event arrives at either input port. Even in the example shown below, the *Add* Module will output two Events when the *Knob* sends one, although they will happen almost simultaneously.



However, during initialization, the *Add* Module will preemptively look at all the values at its inputs and send a single Event from its output port.

Initialization of the Order Module

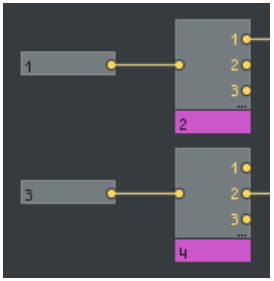
The *Order* Module exhibits special behavior during initialization.

You may have already used the *Order* Module in the earlier tutorials, but they did not explain initialization.

The *Order* Module only sends an Event from its 1 output port during the actual initialization procedure. The Events from the 2 and 3 output ports come later.

It is also important to note that this affects Modules that come before the *Order* Module.

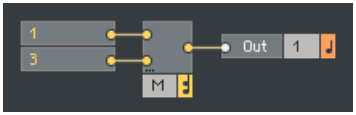
With the Show Event Init Order option active, you can see this in effect:



Initialization of the Merge Module

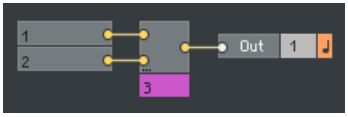
Another Module that has special behavior at initialization is the *Merge* Module. As with the *Add* Module example, it sends only one Event during initialization. As a result, the *Merge* Module cannot behave like it does during normal operation, that is, to forward all Events to its output. It only forwards the Event from the bottommost input port to its output port.

In the Structure shown below, the *Merge* Module sends the value “3” to its output during initialization.



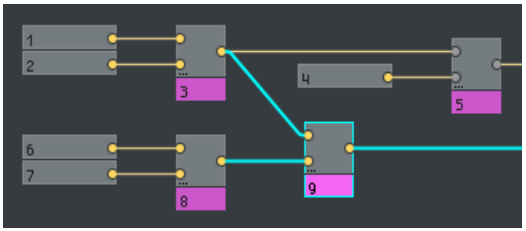
Another special property of the *Merge* Module is that its ports determine the initialization order of the Modules connected to them.

The rule is that the Modules are always initialized from top to bottom. So if you look at the example Structure again with the Show Event Init Order option active, you will see this illustrated:



Even though the *Merge* Module will only let the Event from the bottommost input port through, the initialization order can affect other parts of the Structure as well if the Modules lying upstream of the *Merge* Module have other connections as well.

In the following structure, the *Merge* Module is highlighted and Show Event Init Order is active.



It shows that everything connected to the top port of the *Merge* Module gets initialized first, and then the Modules connected to the lower port are initialized.

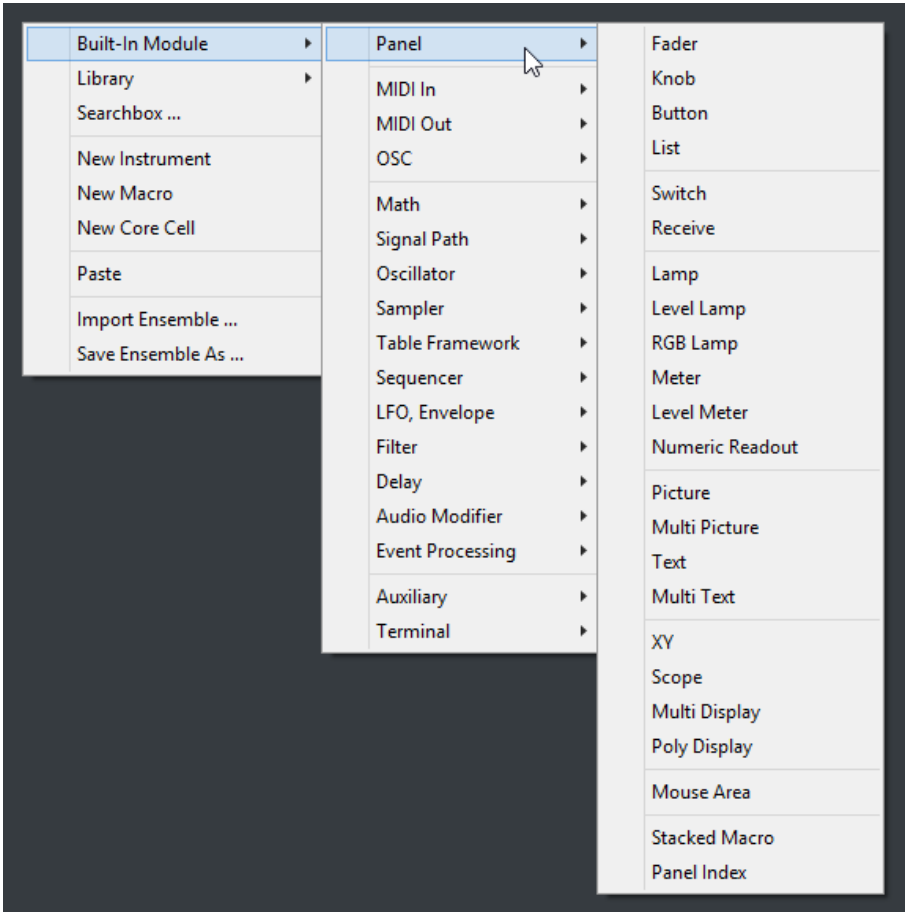
For simple structures, this behavior should not cause any problems, but should issues arise with the timing of Events during initialization, this is something to look out for.

10 Creating and Customizing Interfaces

REAKTOR gives you several ways to design and customize the graphical user interface (GUI) of your creations, from simply setting a custom color scheme to importing your own custom skins for controls. This section will cover all topics in this area.

10.1 Overview of Panel Elements

Any Module designed for use on the instrument panel can be found in the Panel section of the Built-In Modules.



The Panel Modules.

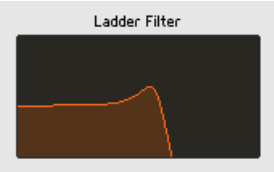
There are many Panel Modules, all of which are listed in the Module Reference in section ↑18.5, Panel ↑18.5, Panel.

There are 3 main types of Panel Modules:

- **Simple Controls** like *Knobs* and *Buttons*. These Modules are controls designed for common uses. *Knobs* and *Buttons* appear on many instrument and effect interfaces, and these Modules are the quickest and easiest way to create these controls.

- **Simple Displays** like the *Level Meter*. These Modules display information, but are not a control. In other words, they have inputs but no outputs.
- **Elements** like the *Mouse Area* or *Multi Display*. These Modules generally cannot be used on their own; they need to be combined with other modules to create a control or display. They allow for more flexibility, but are thus more complicated to use.

Some other Modules also have a display option. For example, the Filter Modules can be set to visible in their properties to display a frequency curve.



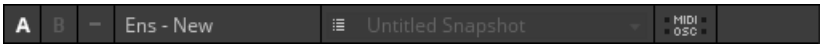
The Panel Display of the Ladder Filter.

The following categories of Modules have optional panel displays:

- **Samplers** display the waveform of the currently playing sample.
- **Envelopes** display the envelope plot.
- **Filters** display the frequency response of the filter.
  - This includes EQ Modules, but excludes the Modal Bank, Differentiator, and Integrator.
- The **Event Table** and **Audio Table** display the contents of the tables.

10.2 Customizing the Header

At the top of every new Ensemble or Instrument you create is the Header.



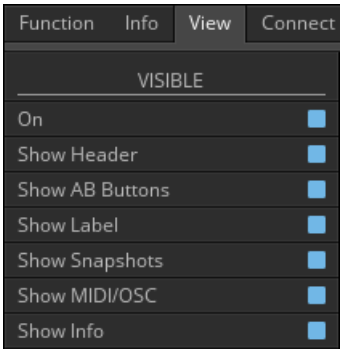
The Header of a New Ensemble.

By default it contains the following (from left to right):

- A/B view and minimize buttons.

- The Instrument/Ensemble name.
- Snapshot menu.
- MIDI and OSC input/output indicators.
- Polyphony voice display.

You can customize the contents of the Header in the View tab of the Properties.



The Header Display Options.

In the VISIBLE section of the View tab there is a list of checkboxes, each one controlling the visibility of each of the elements listed above.

- Show Header: Toggles the visibility of the entire Header on/off.
- Show AB Buttons: Shows/hides the A/B and minimize buttons.
- Show Label: Toggles Shows/hides the Instrument/Ensemble name label.
- Show Snapshots: Shows/hides the Snapshot menu.
- Show MIDI/OSC: Shows/hides the MIDI and OSC input/output indicators.
- Show Info: Shows/hides the Voices display.

If the number of Voices is set to 1, the Voices display is automatically hidden.

10.3 A/B Views

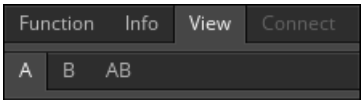
Each Instrument created in a REAKTOR Ensemble can have two panel views, labelled A and B.

1. You can locate the A/B view controls in the Instrument Header.



The A/B views can be very useful for creating a main interface panel and something like an options or settings panel.

In every Module or Macro with a panel element, in the View tab of its Properties you will see 3 tabs: A, B, AB.



The A/B View Tabs.

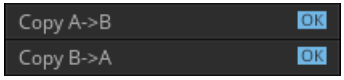
These allow you to give each Module a different set of panel properties for each panel view.

- Selecting A means you will be editing the values for the A view only.
- Selecting B means you will be editing the values for the B view only.
- Selecting AB means you will be editing the values for both the A view and the B view.

You have many options to completely change the look and position of a control between the A and B view. You can even make a control only visible in one view.

In some cases you may want the A and the B view to be the same. In this case you can copy the settings for all panel elements from one view to the other from the Instrument Properties:

1. Enter the Instrument Properties and select the View tab.
2. Find the COPY section at the bottom of the View tab. It holds two functions: Copy A->B and Copy B->A.

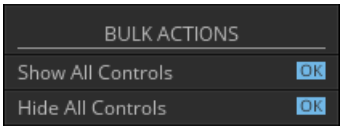


3. Click on the OK button beside the function you wish to perform.

⇨ The View settings from one panel will be copied to the other.

Every individual Module/Macro also has this option, so you can perform this function on a single control or display.

The View tab of the Instrument/Ensemble Properties also gives you BULK ACTIONS to hide or show all controls.



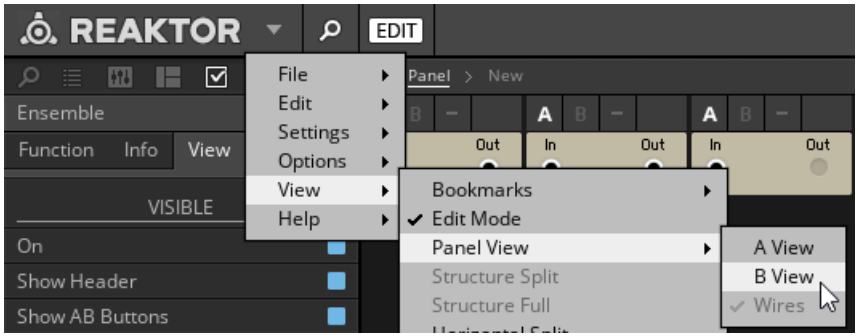
The Bulk Actions.

You can use these to hide all controls on the B view, and then go through the structure and only make a handful of Modules visible on the panel.

Switching Views for Multiple Instruments

You can switch between the A and B view of all Instruments contained in an Ensemble at the same time.

1. Open the REAKTOR main menu, go to **View** and select **A View** or **B View** under **Panel View**.



10.3.1 Examples in the Factory Library

Some of the Ensembles in the Factory Library make use of the A/B Views. Here are a handful of examples:

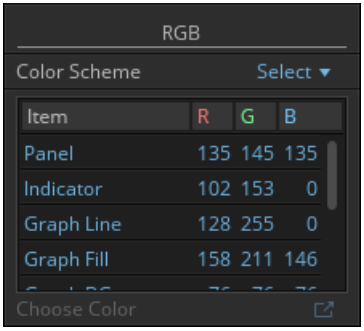
- The *Steam Pipe 2* and *SubHarmonic* synthesizers both use the B view to display the reverb effect controls.

- The *Green Matrix* synthesizer uses the B view to display the modulation matrices.

10.4 Changing the Color Scheme

Any control with a built-in panel display can have its colors customized at the Ensemble/Instrument level. This means that for any Instrument in the Ensemble, you can set the indicator color for all knobs to a single setting, but you cannot set the color of an individual knob.

The Color Scheme settings are located in the View tab of the Instrument Properties in a table labeled RGB (Red Green Blue).



The RGB Controls.

Here, you can select the color of the following elements:

- Panel: The main background color of the Instrument.
- Indicator: Sets the color of the indicator of the knobs and sliders, as well as the on state of the buttons and switches.
- Graph Line: Sets the color of lines drawn in any element that uses a graph or plot (eg. Filter curves and Tables)
- Graph Fill: Sets the color of the fill area under the Graph Line.
- Graph BG: Sets the background color of the graph.
- Grid: The Event and Audio Tables can have a value grid overlaid on top of the graph. This parameter selects the color of the grid.

The following parameters only apply to Tables which have their Graph property set to 2D Color:

- 2D Table Min: Sets the color to represent the minimum value.
- 2D Table Max: Sets the color to represent the maximum value.
- 2D Table Default: Sets the color to represent the default value (0).

10.4.1 Editing Colors

You can specify colors for any of the available elements in the table in two ways:

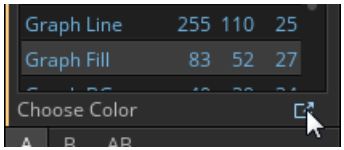
- Entering RGB values.
- Selecting a color using the Choose Color window.

To manually enter an RGB value:

1. Double-click on the value you wish to edit.
2. Enter a value between 0 and 255.
3. Click away or press [Enter] to apply the changes.

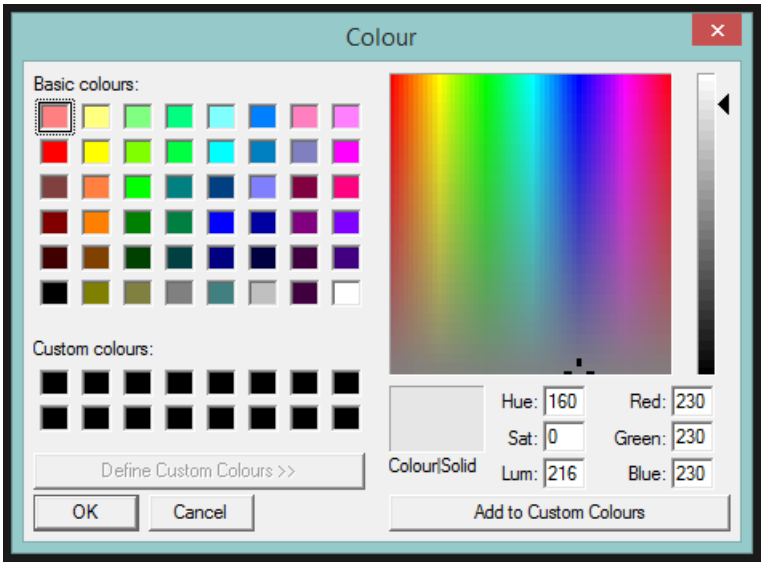
To select a color using the Color window:

1. Select the element you wish to edit from the RGB table.
2. Click on the open window icon next to Choose Color below the RGB table.



⇨ The Color window will open.

The Color window displays a number of options that allow you to select colors in a more intuitive way than entering numbers into a table.



The Color Window as it appears on the Windows Operating System

1. Select a color of your choice and confirm the selection by clicking OK.

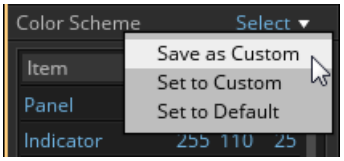
⇨ The values in the RGB table will update to those of the chosen color.

10.4.2 Saving a Custom Color Scheme

REAKTOR allows you to store a custom color scheme that you can store and recall for future creations.

Once you have created a color scheme you like you can store it by following these steps:

1. Open the Select menu next to Color Scheme.
2. Select Save as Custom.



⇨ The current color scheme will be stored as the Custom color scheme.

To recall the Custom color scheme:

1. Open the Select menu again.
2. Select Set to Custom.

⇨ The color scheme for the Instrument will be set to the last color scheme you saved as the Custom color scheme.

10.5 Skinning with Custom Images

Many of the Panel Modules can have their look altered by importing image files to skin the control or display.

You can also add custom images as the background of the Instrument.

10.5.1 Image Formatting

REAKTOR can accept BMP, PNG or TGA file formats for images.

It is not possible for REAKTOR to process images (for example: rotating them or adjusting their transparency) and so all frames of an animation should be rendered as images and then stitched together to form one long image, like a reel of film.

The frames of the animation can be arranged either horizontally or vertically, with the lowest value state in the 1st frame to the left/top of the file. As an example, an animation for a knob could look something like this:



An example of a knob animation in a filmstrip style image.

When an image like this is loaded into REAKTOR, you must specify the number of frames, and then REAKTOR can divide the image accordingly.

Faders are an exception as they can accept a single image, which is then used to replace the fader handle.

For creating knob animations like the one above, you can use a free program called KnobMan by g200kg.

Formatting Images for Buttons

Buttons can accept 2 or 4 state animations. A 2-state animation will give you the basic on/off states, but a 4-state animation will also allow you to customize mouse-pressed states.

A 2-state button animation will have its frames interpreted like this:

- Frame 1 = off
- Frame 2 = on

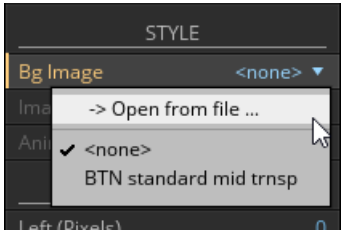
A 4-state button animation will have its frames interpreted like this:

- Frame 1 = off
- Frame 2 = on
- Frame 3 = off, mouse pressed
- Frame 4 = on, mouse pressed

10.5.2 Importing an Image

As a first example, you can import an image as a custom background for an Instrument:

1. Select the Instrument.
2. In the Properties window, select the View tab.
3. In the section labelled STYLE, open the Bg Image menu (which should currently display the text <none>)
4. Select -> Open from file …



5. A dialog window will open allowing you to browse your file system for BMP, PNG and TGA image files.
6. Locate and select an image and click Open.
7. The Image Properties window will open, but you can ignore this for now.

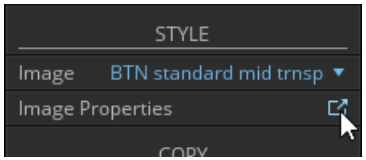
8. Click Ok at the bottom of the Image Properties window.

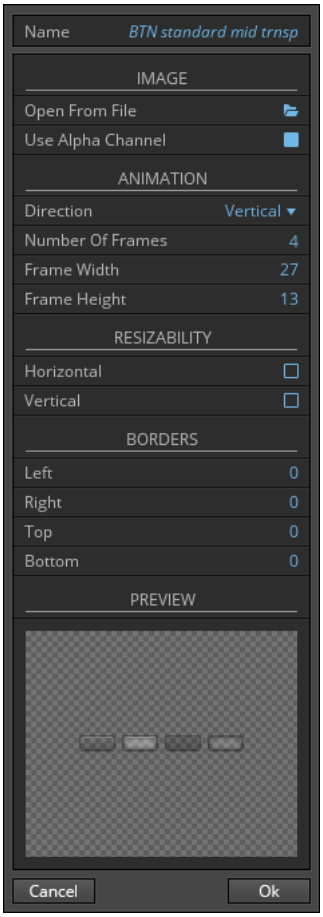
⇨ Your image will now be displayed as the instrument background.

10.5.3 The Image Properties Window

When loading an image for the first time, you will be presented with the Image Properties window. This is where you tell REAKTOR how to read the image file.

1. You can also edit the properties of an image that has already been imported by clicking on the open window icon next to Image Properties in the STYLE section of the Properties' View tab.





The Image Properties Window

The Image Properties window has the following options:

- Name: By default the name of the imported image will be the name of the file you loaded, however you can change this by entering new text into this area.
- IMAGE:

  - Open From File: If you have made adjustments to an image that is used in multiple places throughout an Instrument, you do not have to replace each one individually. You can use this option to open an image file to replace whichever image you had selected when you opened the Image Properties window.
  - Use Alpha Channel: This checkbox tells REAKTOR whether or not the imported image uses an alpha channel (i.e. transparency).

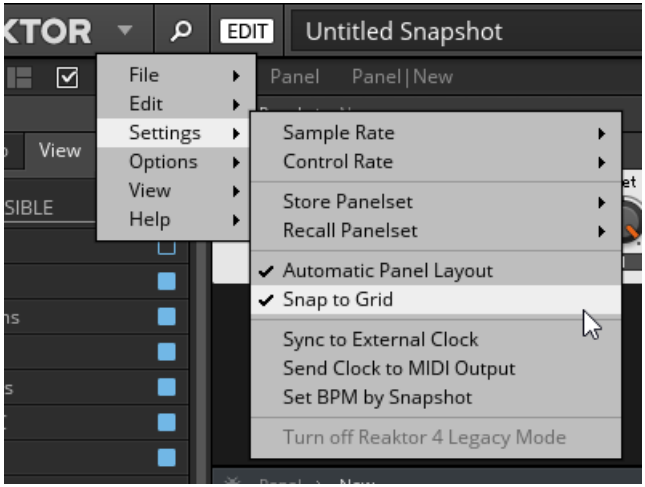
- ANIMATION:
  - Direction: Tells REAKTOR how the frames of the animation are orientated.
  - Number Of Frames: You can enter the number of frames in the animation here. If the imported image is not an animation, this should be set to 1.
  - Frame Height: Displays the pixel height of each frame in the animation. If the animation is vertical, you can manually enter the height of each frame and REAKTOR will calculate the number of frames.
  - Frame Width: Displays the pixel width of each frame in the animation. If the animation is horizontal, you can manually enter the width of each frame and REAKTOR will calculate the number of frames.
- RESIZABILITY: Certain Panel Modules can be resized by REAKTOR (for example, Macros and the backgrounds of the Multi Display and Poly Display). This section contains checkboxes to tell REAKTOR if the image is permitted to be resized in such circumstances.
- BORDERS: If an image is set to be resizable in any way, you are able to specify borders which will not be resized. This is especially useful for any image used as a background.
- PREVIEW: This area displays a preview of the image after it has been processed by REAKTOR.

10.6 The Panel Grid and Borders

By default, REAKTOR places controls on a 4 pixel grid. This can be useful for creating clean layouts with the built-in panel elements, but when working with custom skins, this restriction could be undesirable.

To turn the 4-pixel grid off:

1. Click on Settings in the menu bar.
2. Click on Snap to Grid to uncheck this option.



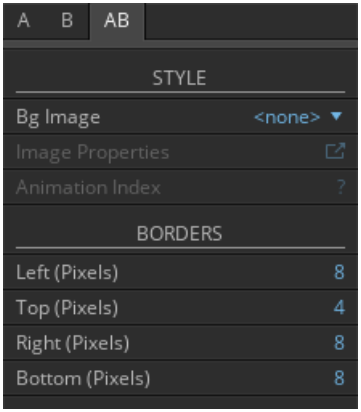
⇨ You will now be able to place controls anywhere on the panel.

Most controls have their position on the grid calculated from the top left corner of the element; however the Knob element is measured from its center.

Borders

The borders at the edges of an Instrument panel can be adjusted. This way you can define how close panel elements can be placed to the edges.

The thickness of the borders can be adjusted under **BORDERS** in the **VIEW** tab of the Properties.



BORDERS in the **VIEW** Properties

10.7 Layering Panel Elements

Sometimes it is necessary to layer one panel element on top of another; a common example would be placing a *Mouse Area* Module on top of a *Multi Display*.

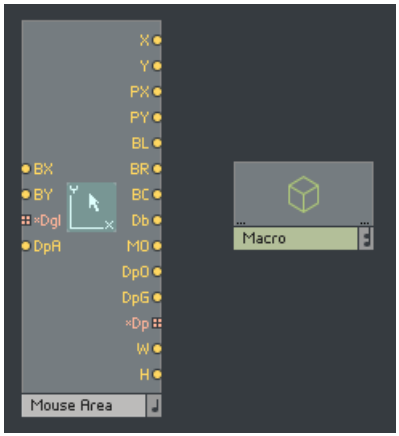
Layering Using Macros

When layering panel elements, REAKTOR uses the Macro order to decide which element should be placed on top of the others.

So, in the example case of placing a *Mouse Area* on top of a *Multi Display*, you would need to:

1. Create a Macro and place a *Mouse Area* Module inside it.

2. Create a new Macro inside the current Macro.



3. Place a *Multi Display* inside the new Macro.

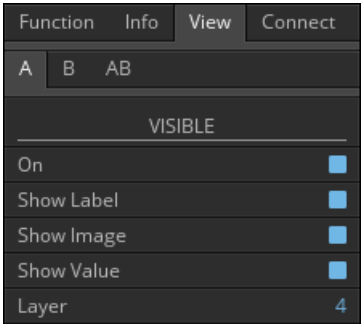
⇨ Because the *Multi Display* is contained in a Macro one level below the *Mouse Area*, it will be placed under of the *Mouse Area* on the panel.



This technique is used in the ↑6, Advanced Step Sequencer ↑6, Advanced Step Sequencer tutorial.

Layering within Macros

Every Panel Element has a Layer property in the VISIBLE section of the View tab.



The Layer Property of a Knob

With this option you can specify the layering of Elements within a Macro.

There are seven available layers that you can assign to the elements, with layer 0 being the lowest layer, and layer 6 being the top layer.

Sub-Macros are also assigned a layer, with all of the Elements contained within that Macro being placed in the assigned layer in the parent Macro, and then sorted by their own layering settings.

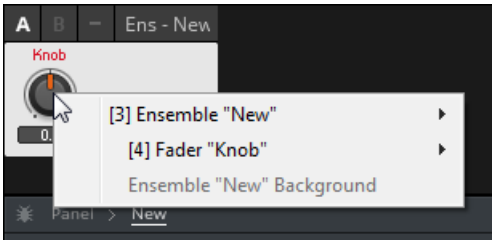
So the previous layering example could also be accomplished using this technique:

1. Create a *Mouse Area* Module and a *Multi Display* Module.
2. In the Properties of the *Mouse Area*, navigate to the View tab.
3. Set the Layer property of the *Mouse Area* to 5.
4. In the *Multi Display* Module, set the Layer property to 3.

⇨ The *Mouse Area* will be placed on top of the *Multi Display*.

Editing Layering from the Panel

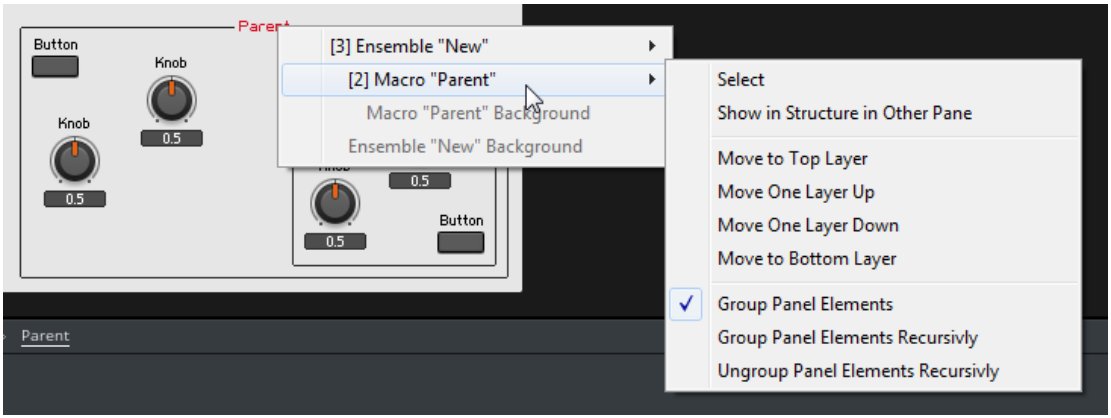
All controls and Macros have a secondary menu which can be accessed by holding [Ctrl] + [Shift] ([Cmd] + [Shift] on OS X) while right-clicking.



The Secondary Context Menu

From this menu you can select several layering commands.

1. To view these commands, move the mouse over the Panel Element's name in the menu.



The available layering commands are:

- Move to Top Layer: sets the selected Module's Layer Property to 6
- Move One Layer Up: moves the selected Module up one layer
- Move One Layer Down: moves the selected Module down one layer
- Move to Bottom Layer: sets the selected Module's Layer Property to 0

If you do not specify the order of Modules within a Macro, REAKTOR will give top level priority to whichever Module is currently selected. This means that the panel element on top could change during use, which is often undesirable.

10.8 Stacked Macros

Stacked Macros are a special kind of Macro that allows you to dynamically show and hide panel elements.

Each Macro placed inside a Stacked Macro can be considered as a page or tab on the panel. You can only view the panel elements contained in one of these Macros at any one time.

How you select which Macro is currently on display is by using the **Panel Index** Module.

10.8.1 Using Stacked Macros

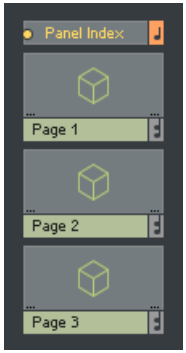
This short guide will show you how to use a Stacked Macro to create multiple pages of controls.

Firstly, you will need to create a few items in the structure:

1. In an empty Ensemble, create a *Stacked Macro*. It is located in the Panel submenu of the Built-In Module menu.

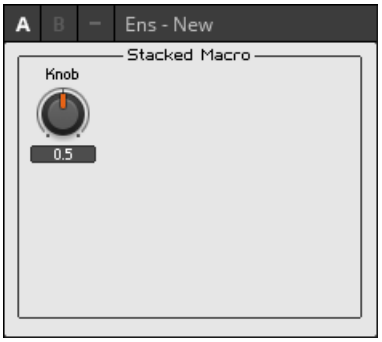


2. Enter the *Stacked Macro* and create a *Panel Index* Module.
3. Create 3 empty Macros inside the *Stacked Macro* and call them Page 1, Page 2, and Page 3.



4. Inside the Page 1 Macro, create a *Knob* Module.
5. Inside the Page 2 Macro, create a *Fader* Module.
6. Inside the Page 3 Macro, create a *Button* Module.

When you return to the panel view, you should only be able to see the knob you created in the Page 1 Macro.

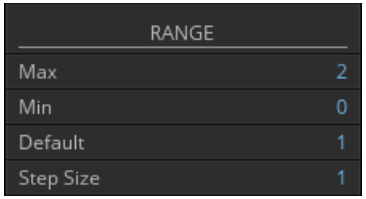


The Ensemble Panel

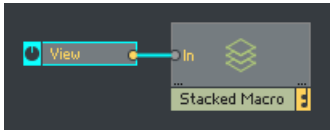
The *Fader* and *Button* are hidden because they are in different Macros.

To select between these Macros we need to create a control:

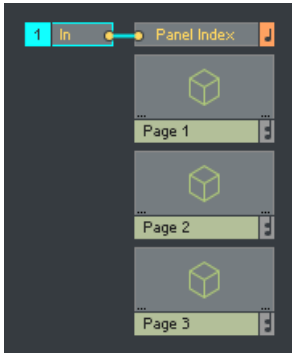
1. Enter the Ensemble structure and create a *Knob*.
2. In the Function tab of the *Knob* Properties, set the Min value of the knob to 0 and the Max value of the knob to 2.
3. Set the Step Size to 1.



4. Rename the knob to "View".
5. Connect the *Knob* to the *Stacked Macro*.



6. Inside the *Stacked Macro*, connect the *In Port* from the *View* knob to the input of the *Panel Index* Module.



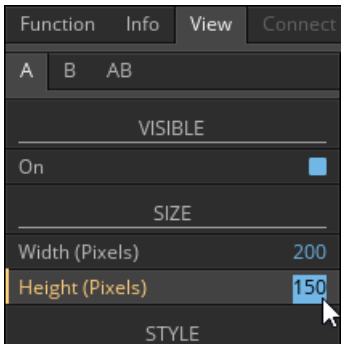
⇨ Upon returning to the panel view, you should be able to change which Macro, and therefore which control type, is displayed by using the View knob.



Note that when using the View knob, the order of the Macros displayed is the same as the order in which they were created.

When using the Stacked Macro you might notice that the Macro frame is much larger than required for the controls you have placed inside it. This is because a Stacked Macro may contain several Macros with different numbers and sizes of elements, and it would not be ideal for the frame to constantly resize as the user browses through the Macros.

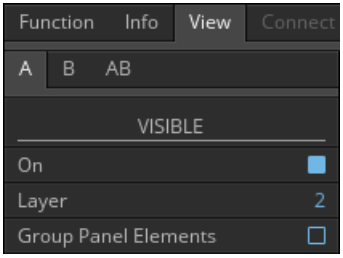
1. The size of the Stacked Macro frame is set in the View tab of the Stacked Macro Properties.



10.9 Grouping Panel Elements

By default, if you want to move the Panel Elements contained within a Macro as one, you need to click and drag on the Macro frame. This was fine for some circumstances, but if the Macro frame is not being used, it can be awkward moving back and forth between visibility settings for the Macro frame.

To make things easier, all Macros have a Group Panel Elements option, located in the VISIBLE section of the View tab, which allows you to group the Panel elements within a Macro and move them together as one.



The Group Panel Elements Property

When Group Panel Elements is active, you can move all Panel elements contained within the Macro by clicking and dragging only one of the elements on the Panel. This is especially useful for layered elements or elements that should be treated as one thing (for example, the modulatable knobs in the REAKTOR BLOCKS framework).

Note that if you have nested Macros, the elements of sub-Macros are not automatically grouped with elements of the parent Macro.

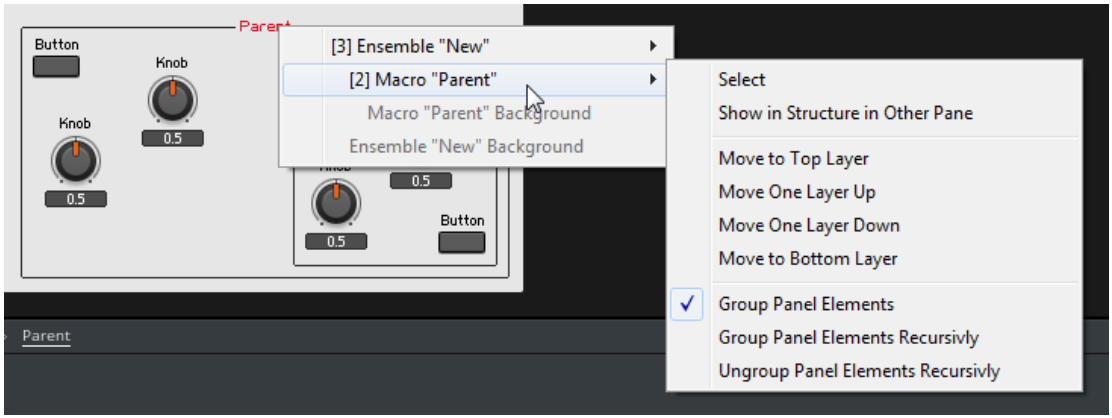
Note that if you have nested Macros, the elements of sub-Macros are not automatically grouped with elements of the parent Macro.

A context menu is available in the Macro panel element that allows you to group and un-group all Macros that are nested within a Macro.

1. To access this menu, hold [Ctrl] + [Shift] ([Cmd] + [Shift] on OS X) while right-clicking on the Macro header.

To Group panel elements of all nested Macros:

1. On the Panel View, hold [Ctrl] + [Shift] ([Cmd] + [Shift] on OS X) and right-click on the parent Macro header.
2. Highlight the parent Macro's name in the menu to display the options sub-menu for the Macro.



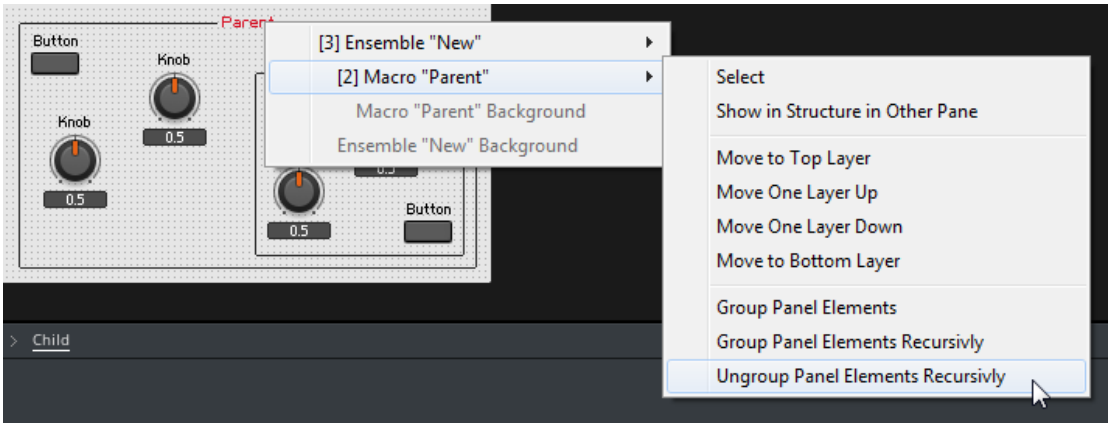
3. Select Group Panel Elements Recursivly.

⇨ This will activate the Group Panel Elements property of the parent Macro and all Macros contained within it.

If you want to ungroup all Panel Elements:

1. Open the menu again.

2. Navigate to the Macro's sub menu and select Ungroup Panel Elements Recursivly.



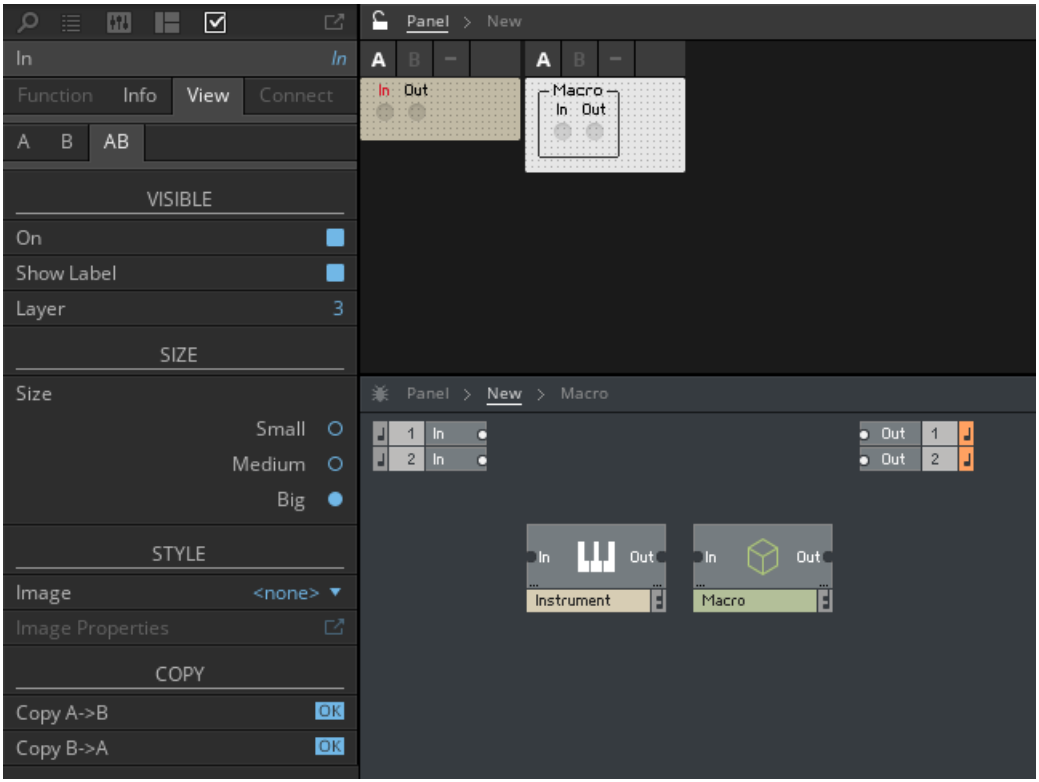
⇨ This will deactivate the Group Panel Elements property for the parent Macro and all Macros within it.

10.10 Panel Ports

Ports of Instruments and Macros can be made visible and used on the panel. This way you can add modular-style patching to your Instrument.

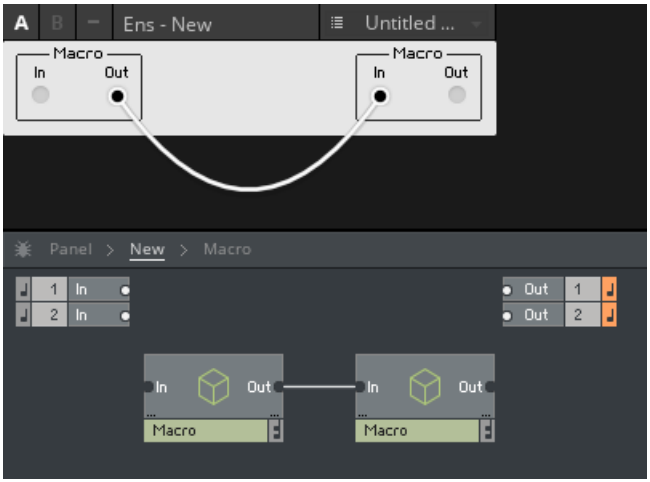


Other than REAKTOR Racks, Ensembles do not store connections made between ports in the host project (when using REAKTOR as a plug-in). The same applies to Snapshots. Connections must be saved with the Ensemble.



Ports of an Instrument and a Macro made visible on the panel

Connections made between panel ports have the same functionality as connections made in the Structure View and are mirrored accordingly.



Connections are mirrored on the panel and in the Structure View

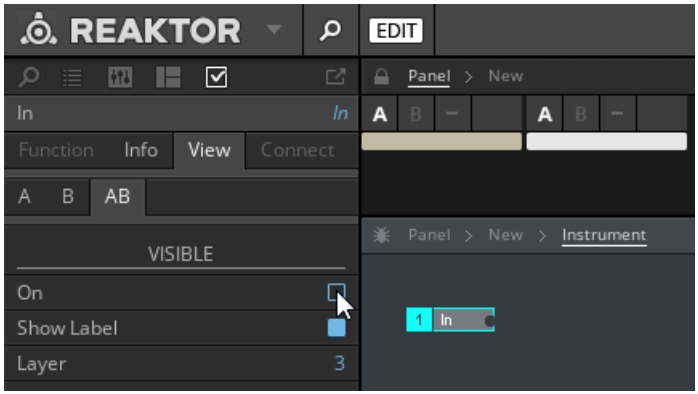
You can use the A and B view of Instruments to create views with and without ports. When using multiple Instruments in the same Ensemble, the View menu allows you to conveniently switch between the A and B view of all Instruments at the same time. For more information, refer to ↑10.3, A/B Views.

Creating Panel Ports

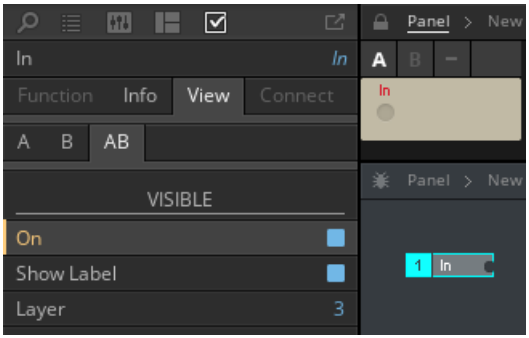
To make a port visible on the panel:

1. Click on the port to select it.
2. Go to the Properties in the side pane and open the View tab.

3. Click on the checkbox for **On** in the section called **VISIBLE**.



⇨ The port is visible on the panel.



Macros and Instruments containing panel ports must be placed on the same structural level in order to allow for interconnectivity.

The following additional settings are available for panel ports in the **VIEW** Properties:

- **VISIBLE** / **Show Label**: Shows or hides the label of the port.
- V**ISIBLE** / **Layer**: Places the port on one of the seven visuals layers on the panel.
- **STYLE** / **Image**: Allows for customization of the port's appearance.
- **COPY** / **Copy A->B**, **Copy B->A**: Copies the port between A and B views.

11 Automation

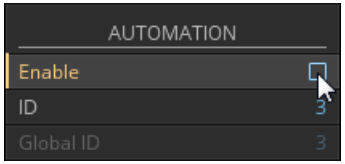
Automation is a way of controlling the parameters of an instrument or effect from a host program. When any new control (like a knob or button) is added to an Ensemble, it is automatically given an Automation ID and added to the automation list.

11.1 Customizing Automation Properties

Disabling Automation

In some cases, you may wish to remove the ability to automate a control (for example, if the control is used to change the display in some way).

1. Click on the control you wish to edit and open its properties.
2. Click on the Connect tab.
3. At the bottom of the Connect tab area is a section called AUTOMATION.
4. Click on the Enable checkbox so that it is empty.



⇨ The control will no longer be automatable and will be removed from the automation list.

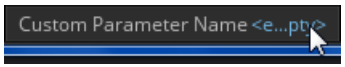
Automation Name

By default, the automation name for a control will be the name of the control combined with its location in the Instrument structure.

However, this is not always the best way to identify a parameter, so it is possible to change this name.

1. Open the control's properties and click on the Connect tab.
2. Navigate to the AUTOMATION section.

3. Click on the text in the section called Custom Parameter Name (it should currently say <empty>).



4. Enter the name you wish to give the control.



5. Once you are finished, press [Enter].

⇨ The control will now use the Custom Parameter Name as its automation name.

Automation ID

As well as an automation name, each parameter has an automation ID. This is a number used to identify the parameter and to order the parameters.

An ID is assigned when a new control is created, and it will automatically be the next number after the ID of the last control that was created.

Like the automation name, this ID number can be changed.

1. Open the control's properties and click on the Connect tab.
2. Navigate to the AUTOMATION section.
3. The parameter called ID will be displaying the current automation ID of the control.



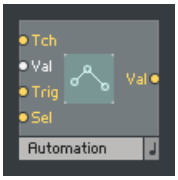
4. Double-click on this number to enter a new number.



5. Once you have finished, press [Enter]

⇨ The newly entered number will now be used as the automation ID

11.2 The Automation Module



The Automation Module

The Automation module is a module that sends and receives automation data to/from the host. It can be used to add custom automation to an existing control, or can be used as part of a custom control element you may want to create from scratch.

The Automation Module also gives you the ability to dynamically change the automation name for the Module. This means that you could use this as part of a control whose function changes depending on a setting in the instrument. For example, a distortion effect might have a control that functions as a Drive control, but the effect may be able to change to a Bitcrusher, and the Drive control becomes a Bit-depth control.

11.2.1 Sending and Receiving Automation Values

The Automation Module has 3 input ports that relate to sending automation values, which are ordered from top to bottom in the order you should use them:

- Tch (touch): sends the touch-state of the parameter, in other words: this lets the host know whether or not the user is moving the connected control.
  - A value greater than 0 should be sent if the touch-state is active.
  - A value of 0 or less should be sent if the touch-state is inactive.
- Val (value): sets the current value of the module to be sent when an event arrives at the Trig input.
- Trig (trigger): an event at this input triggers a send of the value currently at the Val input. To send, the Tch input also needs to be set to active state. If Tch input is in inactive state, then this Trig input has no effect.

The module has a single output port, which sends a value event whenever it receives automation values from the host.

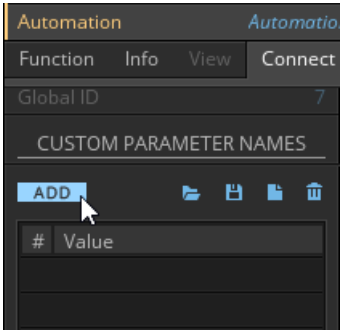
11.2.2 Dynamically Changing Automation Names

By default, the automation name for the Automation Module is the name given to the Module itself. Re-naming the Module will change the automation name.

However, there are some cases in which you may want to change the automation name dynamically.

First of all, you will need to create the name entries:

1. Select the Automation Module and open its properties.
2. Click on the CONNECT tab.
3. In the CUSTOM PARAMETER NAMES table, click the ADD button to add an entry, or click on the trashcan icon to delete an entry from the list.

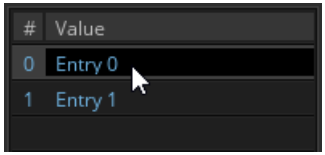


Each entry has an index (#) and a value (Value) field:

- The index is automatically set as an integer, starting from 0 for the first entry.
- The value is the automation name that is to be sent to the host.

To change a value:

1. Double-click on the text you want to edit and then type the text you want to be associated with the corresponding index.



The Sel input selects which entry from the list is used as the automation parameter label:

- If the Sel input is not connected, then entry 0 is used.
- If the list is empty, then the original default automation name will be used (i.e. the module's name and its macro path). The Sel input will be ignored.
- If the Sel input is out of bounds, then it is clipped to 0 → n-1, where n is number of items in the list.

You can create and edit lists of automation parameter labels outside of REAKTOR, which will considerably speed up the process of creating and saving large lists. To open, save, or clear lists of automation parameter labels, use the CUSTOM PARAMETER NAMES table's additional buttons. From left to right these are:

- **Read from File**: Click the folder icon to open a file selection dialog that allows you to select a plain text file. The list of automation parameter labels will be filled with the content of this file.



This action will replace any existing list content.

- **Save to File**: Click the disk icon to open a file save dialog that allows you to save the current list of automation parameter labels as plain text file.



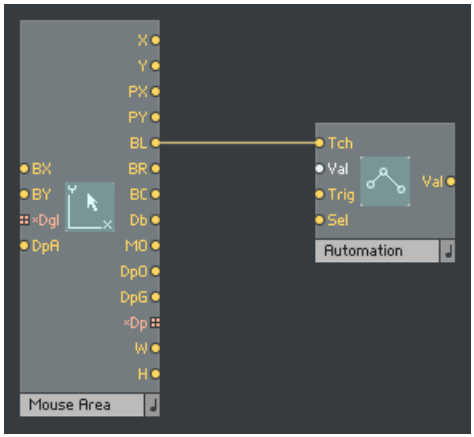
- **Clear List**: Click the document icon to clear the entire list of automation parameter labels.

The file format used by REAKTOR is plain ASCII text (*.txt or *.asc) with one line per entry. An entry index number is not required, as REAKTOR will create these numbers.

11.2.3 Custom Control Example

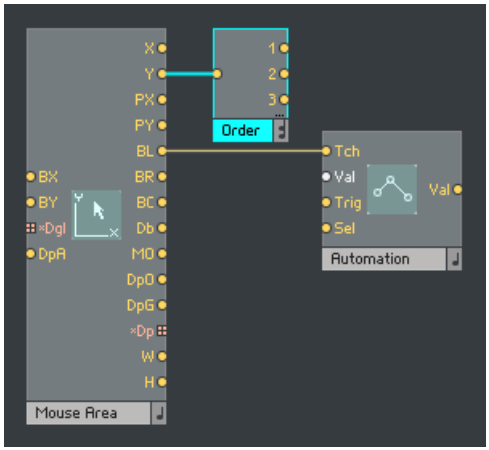
The following tutorial is a quick example designed to introduce you to the Automation Module. You will create a custom control using the Mouse Area Module and a Meter to mimic a fader.

1. Create a *Mouse Area* Module and an *Automation* Module.
2. The control will have its touch-state activated when the left mouse button is pressed, so connect the BL (button left) output of the *Mouse Area* Module to the Tch (touch) input of the *Automation* Module.



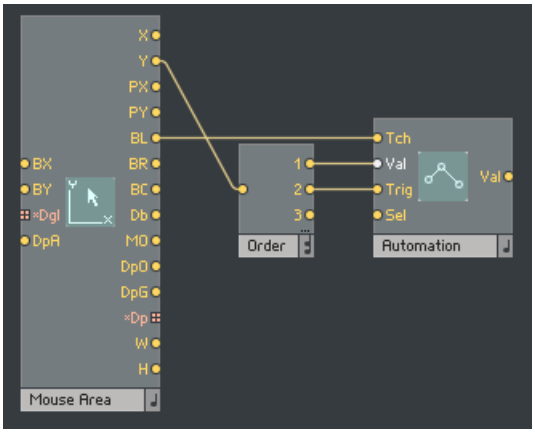
3. When using a software fader, the Y position of the mouse on the fader sets the value of the control. Use the Y output of the *Mouse Area* to send the value. However, the Y output also needs to trigger the value of the *Automation* Module and send it to the host. To insure a correct event order, create an *Order* Module.

4. Connect the Y output of the *Mouse Area* to the input of the *Order* Module.



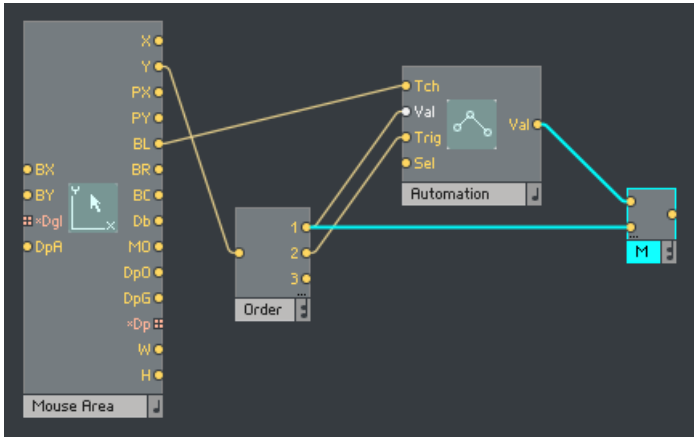
5. Connect output port 1 of the Order Module to the Val input of the Automation Module.
6. Then connect output port 2 to the Trig input.

⇨ The Structure should look like this:



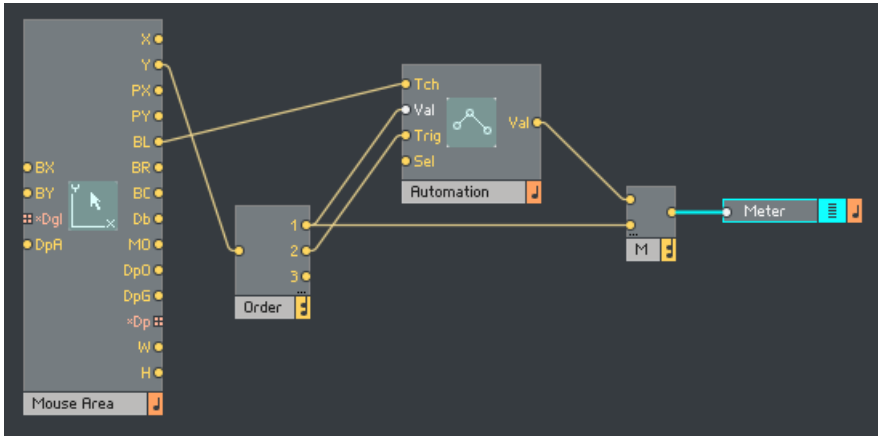
Since both the *Mouse Area* interaction and the automation received from the host will be used to drive the fader value, the Y output of the *Mouse Area* will need to be merged with the output of the *Automation Module*. This can be done be using a *Merge* Module.

1. Create a *Merge* Module and connect output port 1 of the *Order* Module and the Val output of the *Automation* Module to its inputs.

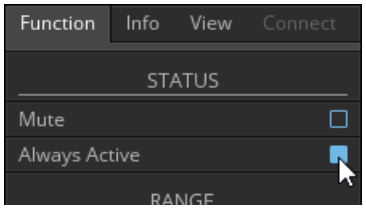


The final stage will be creating the fader itself, and for this you can use a *Meter* Module.

1. Create a *Meter* Module
2. Connect the output of the *Merge* Module to the input of the *Meter*.



3. In the *Meter*'s properties, make sure the Always Active option is checked.



⇨ Clicking and dragging in the *Mouse Area* will now change the level of the *Meter*.

Now that the control is working, you can save the Ensemble and open it in a host. If everything is connected correctly, writing and reading automation values will function as it does for any other REAKTOR control.

12 KOMPLETE KONTROL and MASCHINE 2 Integration

The following chapter will cover the features that relate to integrating your REAKTOR Ensembles into MASCHINE 2 and KOMPLETE KONTROL. This includes visibility in the sound browsers, and optimizing the panel.

Although this chapter will mostly refer only to KOMPLETE KONTROL, the same principals can all be applied to MASCHINE 2.

12.1 Snapshot Visibility

When using REAKTOR, you can browse for a number of different things: Ensembles, Modules, Snapshots, etc. Generally a Snapshot is considered to be a REAKTOR sound preset.

Since the KOMPLETE KONTROL Browser is used primarily to navigate sounds, it has been designed to scan for Snapshots within a REAKTOR Ensemble file. However, Snapshots need to be tagged so that KOMPLETE KONTROL can integrate them correctly.

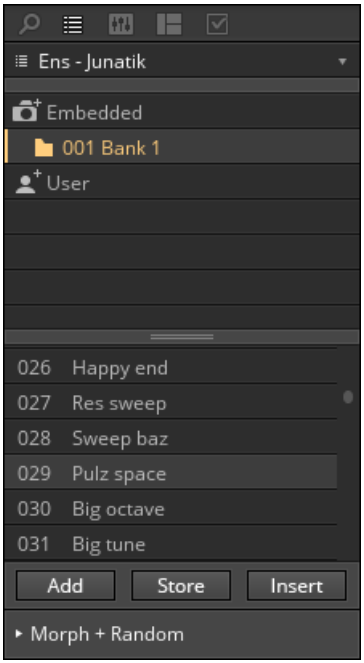
12.1.1 Tagging Snapshots

Since a REAKTOR Ensemble can be an instrument, an effect, or even both, we need to communicate which of these the Snapshot is classed as so KOMPLETE KONTROL will know where to place it in the Browser.



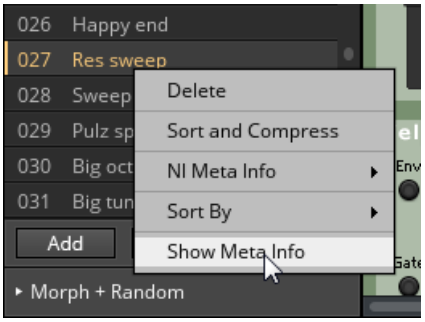
Note that KOMPLETE KONTROL can only open Snapshots tagged as instruments; whereas MASCHINE 2 can open both instrument and effect Snapshots.

Tagging is done in the Presets tab.

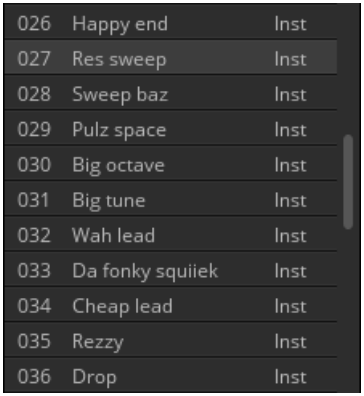


The Presets Tab

1. To display Snapshot tags, right-click on a Snapshot and select Show Meta Info.



⇨ A new column will appear in the Presets browser displaying any tags applied to the Snapshots.



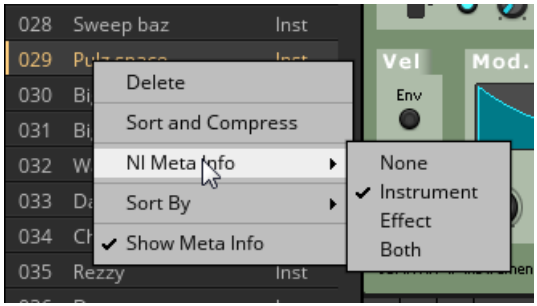
There are four kinds of tags:

- None: The sound will be hidden from the KOMPLETE KONTROL browser. This can be useful if your Ensemble includes an "Init" (i.e. blank) snapshot, which is not something you necessarily want to be displayed in the KOMPLETE KONTROL browser.
- Inst: An Instrument sound. Any Snapshot that generates audio, but does not alter incoming audio should be tagged as an Instrument sound.
- FX: An Effect sound. Any Snapshot that relies on, alters, or reacts to incoming audio should be tagged as an Effect sound.
- Both: A sound that can be either an Instrument or Effect. Any Snapshot that generates audio independently, but can also react to incoming audio somehow, should be tagged as Both.

Tagging a Single Snapshot

1. Right-click on the Snapshot in the Snapshot browser.

2. Enter the NI Meta Info sub-menu to display the available tags.



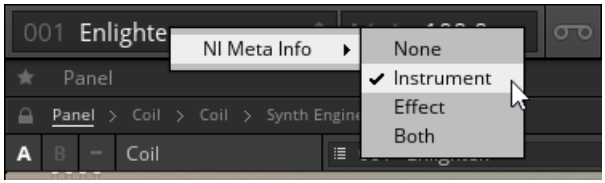
3. Click on the relevant tag for the Snapshot.
4. Save the Ensemble file to make the changes permanent.

Single Preset Files

If you save a Preset file in REAKTOR the saved file will take the tag information from the last selected Snapshot.

If the Snapshot was saved without a tag, you can tag it by using the Snapshot menu in REAKTOR's toolbar:

1. Right-click on the Snapshot menu.
2. Select the relevant tag for the Snapshot.

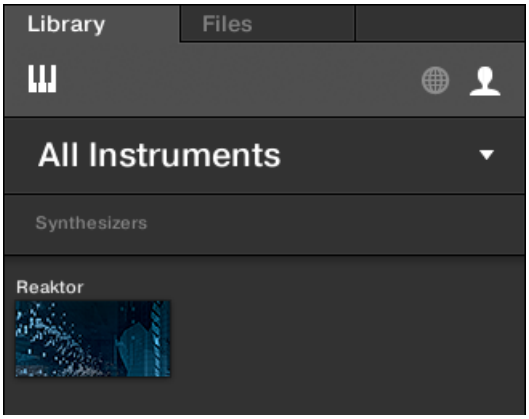


3. Re-save the preset file.

Tagging a preset using this method will not change the tag of the original Snapshot.

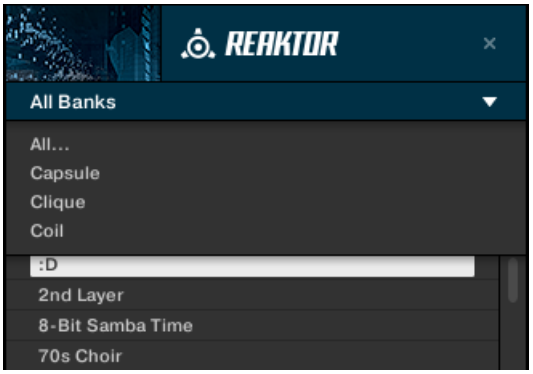
12.2 Defining the Bank Name

Any Ensemble created in REAKTOR containing tagged Snapshots will be located in the REAKTOR product when it is added to the KOMPLETE KONTROL library (see section ↑12.6, Adding Ensembles to the Library ↑12.6, Adding Ensembles to the Library for a walkthrough of how to do this).



The User Tab of the Komplete Kontrol Browser

In order to make locating the sounds in the KOMPLETE KONTROL Browser easier, you can define a Bank name.

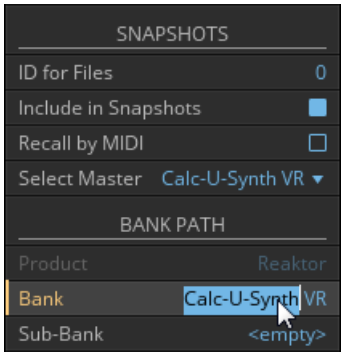


Displaying the Banks

The Bank name is taken from the Ensemble Bank Path property.

To change the Bank name:

1. Select the Ensemble and open the Properties tab.
2. Navigate to the Function sub-tab.
3. Scroll to the area called BANK PATH.
4. Enter the name you wish to use in the Bank property.



⇨ When KOMPLETE KONTROL scans the Ensemble, it will use this as the Bank name.

Note that this also changes the folder name for Preset saving and loading.

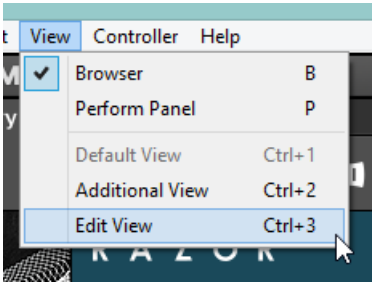
12.3 Panelsets

Even if no action is taken, your REAKTOR Ensemble's interface will appear in the KOMPLETE KONTROL interface when a sound from it is loaded. The appearance will be the same as the last saved state.

However, it is possible to make some optimizations to make using your Ensemble in KOMPLETE KONTROL a better experience.

12.3.1 Default and Additional Views

KOMPLETE KONTROL has three ways of viewing a REAKTOR Ensemble's interface:



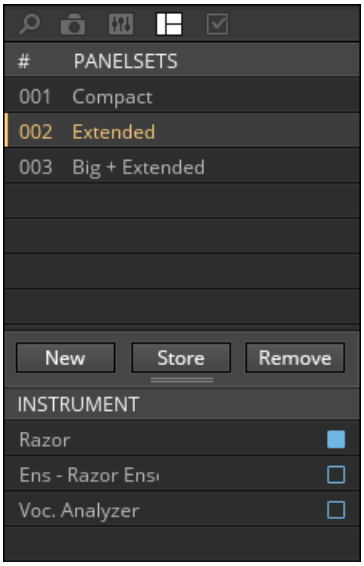
The 3 Views in the View Menu

- Default View: The first view displayed when you load the Ensemble.
- Additional View: An optional second view.
- Edit View: Displays the Ensemble in the full REAKTOR interface.

The first two views are taken automatically from the Ensemble's Panelsets.

In order for KOMPLETE KONTROL to recognize a Default and Additional view, you need to create two Panelsets:

- The first Panelset will be the first view displayed - the Default View.
- The second Panelset will be the alternate view - the Additional View.



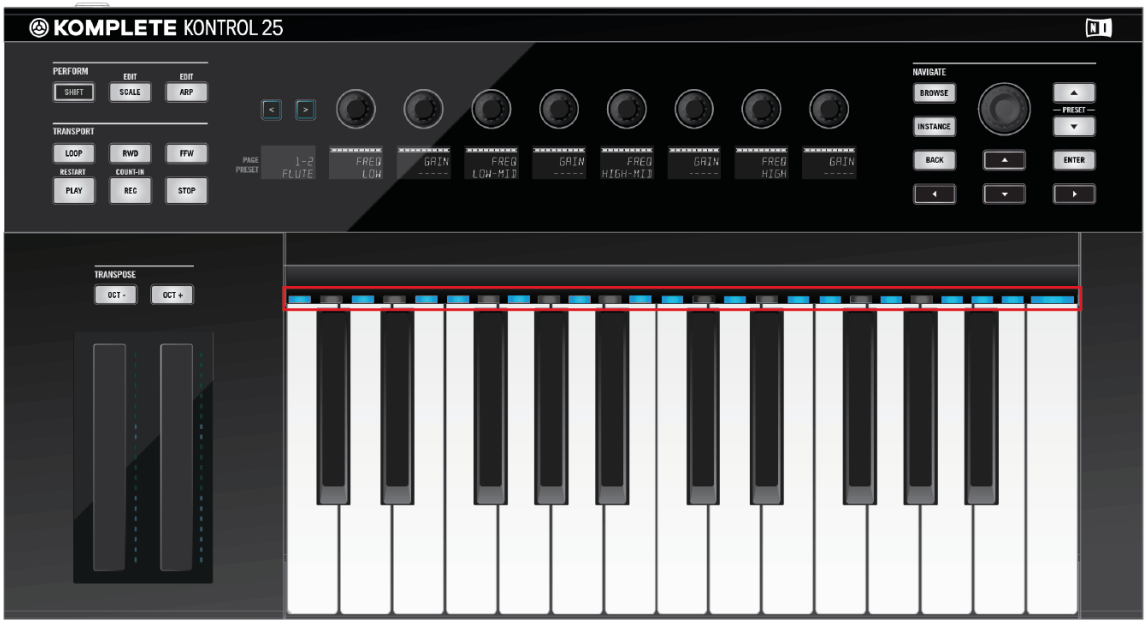
The Panelsets for RAZOR

Any further Panelsets will be ignored by KOMPLETE KONTROL. They will not interfere with the KOMPLETE KONTROL integration, but they will not be available either.

12.4 Hardware Control

The Hardware Control Module (listed in the menus as HW Control) allows greater integration with NATIVE INSTRUMENTS hardware products. The module uses an open design so that it can be updated to support any new hardware NATIVE INSTRUMENTS releases in the future.

At the time of writing, this Module allows custom control over certain features of the KOMPLETE KONTROL S-SERIES hardware, namely the Light Guide system.

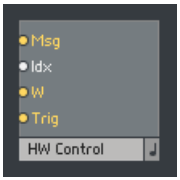


The Light Guide on the KOMPLETE KONTROL S-SERIES keyboard

REAKTOR must be used as a plug-in in the MASCHINE or KOMPLETE KONTROL software in order for the Hardware Control messages to be sent. Hardware Control messages are not sent when REAKTOR is used in stand-alone mode.

The Hardware Control module has four inputs:

- Msg (Message Type): Selects the message type to be sent by the module.
- Idx (Index): Sets the index of the array into which data is to be written.
- W (Write): Writes a value into the currently selected array index.
- Trig (Trigger): Triggers the contents of the module's array.



The Hardware Control Module

To use it correctly, you must send the messages in the following order:

1. Select the Message Type you want to use via the Msg port.
2. Use the Idx and W ports to fill the array with data.
3. Once the necessary data has been written into the array, send an event to the Trig input to send the data to the KOMPLETE KONTROL software.

⇨ KOMPLETE KONTROL will take the data and apply the necessary changes to the KOMPLETE KONTROL S-SERIES hardware.

12.4.1 Message Types

Message Type	Msg Input Value
Set single note attributes	10
Set 128 note colors	11
Set note pressed support	40

Set Single Note Attributes

For this message type, the first index of the array (index 0) sets the MIDI note number, and the following indices are used to set various attributes for that note:

Array[0]	Array[1]	Array[2]	Array[3]
MIDI Note Number	Note Color	Pressed State (0 or 1)	Note Type (0 or 1)
0 ... 127	0 ... 17	0: Unpressed 1: Pressed	0: Default 1: Control

The Note Type tells KOMPLETE KONTROL whether this is a standard playable note, or whether it is a control note used for special functions in instruments. Control notes will be excluded from any performance functions, such as the arpeggiator.

Sending any value outside of the value ranges specified above will cause the message to be ignored, and the relative setting will remain unchanged.

The pressed state will only be sent if Note Pressed Support is set to be active.

Set 128 Note Colors

This message type allows you to send the note colors for all MIDI notes in a single array. With this message type, the array index represents the MIDI note, and the value of that index sets the note color.

Array[0]	Array[1]	...	Array[126]	Array[127]
MIDI Note 0 Color (0...17)	MIDI Note 1 Color (0...17)	...	MIDI Note 126 Color (0...17)	MIDI Note 127 Color (0...17)

Set Note Pressed Support

A value of 40 at the Msg input sets the module to send the Note Pressed Support state. A single

yes/no (0/1) value is needed, and is sent in array index 0:

- : No - REAKTOR will not send Note Pressed messages; this is to be handled by the host.
- : Yes - REAKTOR will send Note Pressed messages, taking control of this from the host.

12.4.2 Available Note Colors

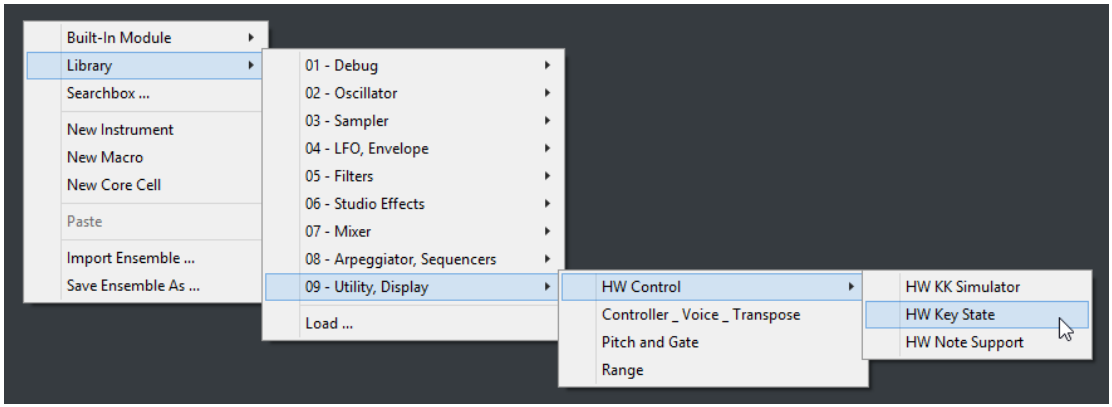
Value	Color
0	Red
1	Orange
2	Light Orange
3	Warm Yellow
4	Yellow

Value	Color
5	Lime
6	Green
7	Mint
8	Cyan
9	Turquoise
10	Blue
11	Plum
12	Violet
13	Purple
14	Magenta
15	Fuchsia
16	Default
17	Inactive

12.4.3 Related Macros

The Library contains a number of tools designed to help with support the Light Guide system, even if you do not have any KOMPLETE KONTROL S-SERIES hardware to test your Ensembles.

1. These tools are found in the 09 - Utility, Display -> HW Control submenu.



For debugging, the *HW KK Simulator* Instrument takes the same messages given to the HW Control Module, and uses them to display a simulation of how the KOMPLETE KONTROL S-SERIES hardware would interpret them.

12.5 Native Map

The Native Map technology used in KOMPLETE KONTROL scans the automatable controls in an Ensemble and maps them to the knobs on the KOMPLETE KONTROL or MASCHINE hardware.

As such, it is best to prepare an Ensemble's automation data to use intelligible names, and to be ordered in such a way as the most important controls appear first.

The controls are mapped according to their automation IDs, and the names published are taken from the automation name.

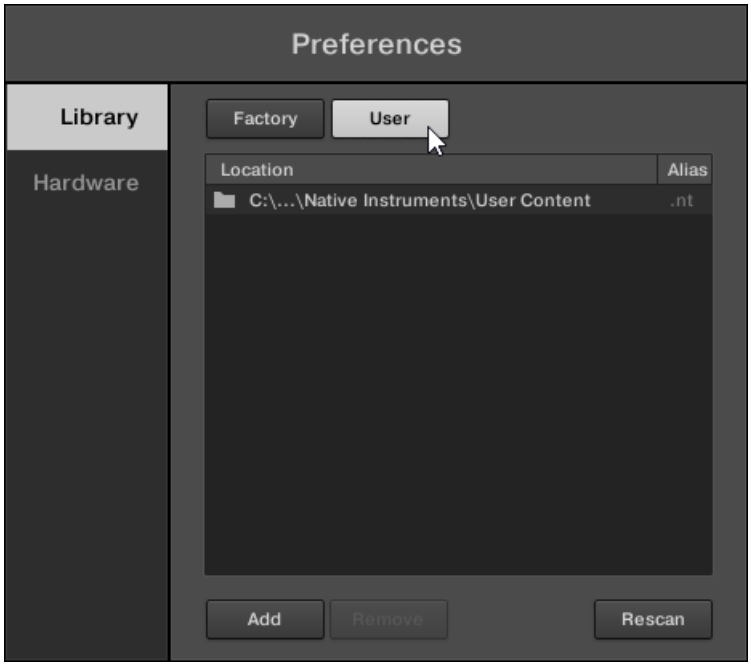
Section ↑11, Automation ↑11, Automation gives all the information you will need regarding editing automation properties in your Ensemble.

12.6 Adding Ensembles to the Library

All NATIVE INSTRUMENTS products are automatically integrated into the KOMPLETE KONTROL Browser during installation.

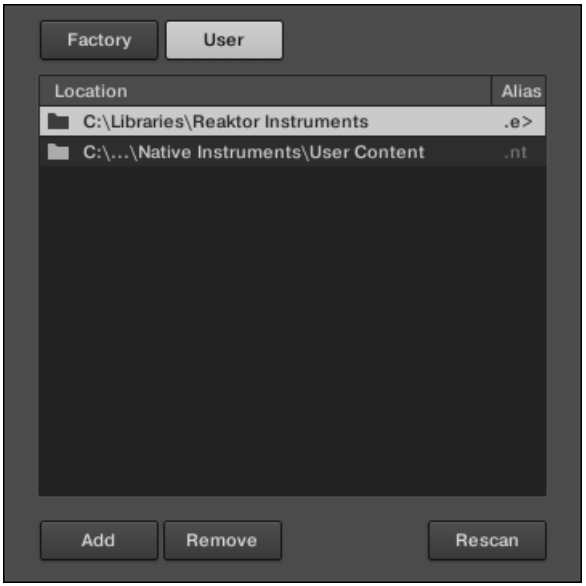
Since any Ensemble you create lacks an installation, it will not automatically appear in the Browser. However, it is possible to tell KOMPLETE KONTROL to scan a certain folder for content:

1. Open the Preferences window by selecting Preferences… from the Edit menu.
2. In the Library tab (which should be selected by default), click on the User button.



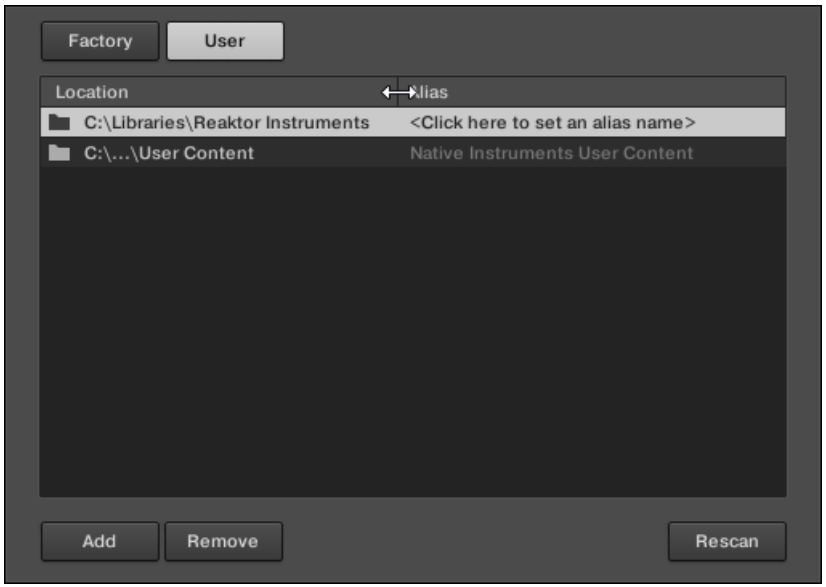
3. Click on the Add button.
4. A window will open, from which you can browse for a folder to add to the KOMPLETE KONTROL User Content.
5. When you have located the folder you wish to add, select it and click OK.

⇨ The selected folder will now appear in the Location table.



With the folder added for KOMPLETE KONTROL to scan, we can change the Alias of this folder to give it a name that is easier to identify.

1. If necessary, you can resize the Preferences window and the Location table columns to get a better view of the content.



2. Click on the entry in the Alias column of the Location table.
3. Type the name you want to give this location.

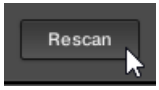


4. When you are finished click away or press [Enter] to apply the changes.

⇨ The Location will now have a custom Alias.

With the folder added and named, there is only one final step left: KOMPLETE KONTROL needs to scan the folder for appropriately tagged Ensembles.

1. Start the scan by clicking on the Rescan button.



⇨ After the scan has finished, the contents of the newly added folder(s) will be added to the KOMPLETE KONTROL browser.



If there are a large number of folders and files to scan, this process could take some time.

13 Internal Connection Protocol

The Internal Connection protocol (IC protocol, in short) is a feature that lets you send and receive Events to and from different parts of the structure without the need of wires. It also allows you to directly connect panel elements to each other, or to other IC Modules.

13.1 Connecting Two Panel Elements via the IC Protocol

First of all, you will need to create two controls:

1. Create a new Ensemble.
2. Unlock the Panel.
3. Right-click the Panel area and choose the Built-In Module > Knob menu entry.
4. Repeat the previous step to create a second *Knob*.
5. Move the Knobs so that they do not overlap.
6. Double-click on each knob's label and rename the knobs to “Knob A” and “Knob B”.

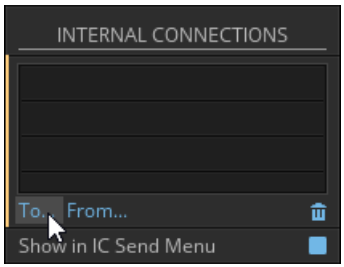


7. Lock the Instrument Panels by pressing the Panel Lock button.

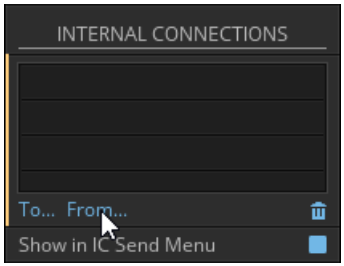
With the knobs in place, you can now use Internal Connections to control Knob B with Knob A:

1. Select *Knob A* to view its properties and navigate to the Connect tab.
2. At the bottom of the Connect tab is an area called INTERNAL CONNECTIONS.

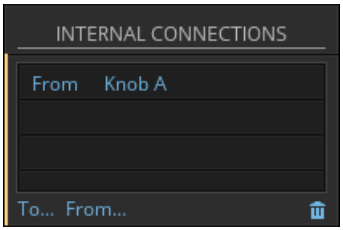
3. In this area is a button labelled To..., click on this to activate *Knob A* as a potential IC Send Module.



4. Go to the Connect tab of *Knob B* and find the INTERNAL CONNECTIONS area.
5. Click on the From... button to activate *Knob B* as a potential IC Receiver.



⇨ Since *Knob A* was already activated as a potential IC Send, a "wireless" connection is made between *Knob A* and *Knob B*.



If you now move *Knob A*, *Knob B* will mirror the value of *Knob A*. However, if you move *Knob B*, *Knob A* will remain as it was. Note that this holds true even if *Knob B* has a different value range than *Knob A*.



You can also set up an additional Internal Connection with *Knob B* as the sender and *Knob A* as the receiver. REAKTOR handles this in such a way as it will not cause any infinite feedback loops.

13.2 IC Send/Receive Modules

As well as being able to connect controls to each other, REAKTOR also includes *IC Send* and *IC Receive* modules for other uses.



The IC Send and IC Receive Modules

To give an example of how this works, this tutorial will show you how to create a knob that deactivates a button whenever it is set to zero.

1. Create a new Ensemble.
2. Enter the Ensemble structure and create a *Knob*, a *Button*, and an *IC Send* Module.



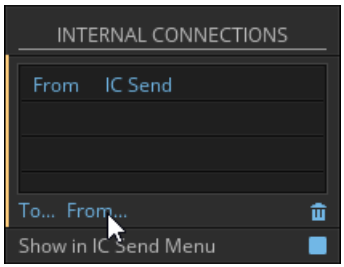
You will now need a way of sending a zero event only when the knob is at zero.

1. Create a *Separator* Module (Built-In-Module > Event Processing) to split the events from the *Knob* depending on their value.
2. Connect the output of the *Knob* to the In port of the *Separator*, and then connect the Lo out port of the *Separator* to the *IC Send* Module.

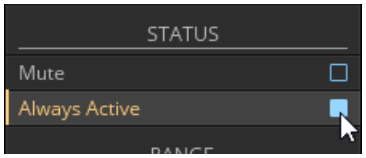


3. The Thld (threshold) port of the *Separator* does not need a *Constant* attached, since it defaults to zero, which is the value needed in this example.

4. You can connect the *IC Send* module to the *Button* in the same way as described in the previous section.



5. In order for everything to work as desired, you will need to activate the Always Active option in the Function tab of the IC Send properties.



⇨ When you return to the Panel, you will be able to active the *Button* by clicking on it, and then deactivate it by turning the *Knob* to 0.

On the panel you will also notice that the *IC Send* has a Panel element. This control allows you to make or break Internal Connections with any IC enabled Module from the Panel.

The settings made with this control are not saved in the Snapshot, but are saved in the Ensemble. As such, this control is generally reserved for helping with prototyping, and is often hidden when releasing an Ensemble to the public.

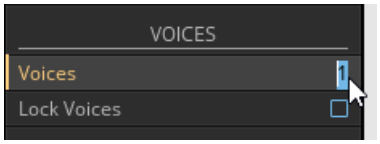
14 Optimization Techniques

14.1 Reduce the Number of Voices

Although a single oscillator with 8 voices will not be as heavy on the CPU as 8 monophonic oscillators, the general principal still applies: More voices leads to a greater load on the CPU.

If the instrument you are creating does not need a high level of polyphony, try setting the number of voices to just the number you need.

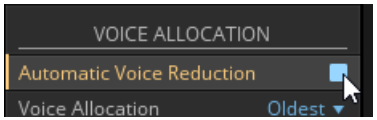
1. Reduce the number of Voices in the Ensemble Properties.



14.1.1 Automatic Voice Reduction

Automatic Voice Reduction reduces the number of voices in an Ensemble until the CPU level is below the Max CPU value (which is defined in the Preferences window).

1. Click on the Automatic Voice Reduction option in the Ensemble Properties to activate Automatic Voice Reduction.



⇨ REAKTOR will reduce the number of Voices until the CPU level drops below the Maximum threshold. It will continue to do this until the Automatic Voice Reduction option is unchecked.

14.1.2 Use Monophonic Processing

You should only use polyphonic Modules when it is a necessity. If you are building an FX unit, or a mono-synth, you do not need 16 voices all doing the same thing. And even for a poly-synth, you may only need the oscillators, envelopes and filters to be polyphonic; any LFOs or effects could be processed as mono signals.



See section ↑2.4, Mono and Poly Modes ↑2.4, Mono and Poly Modes for more information about polyphony.

14.2 Use Events Rather than Audio signals

When processing at the audio rate, calculations are made constantly at the sampling rate. Events are only processed when a value is triggered, and are also processed at a slower rate. If the process you are performing does not directly interact with the audio stream, consider processing with events.

14.3 Eliminate Superfluous Events

In REAKTOR it is sometimes possible to generate Events even if the value is not changing, or to generate multiple Events from a single process.

14.3.1 Filtering Duplicate Values

If you see an area in which this could occur, try using a *Step Filter* Module.



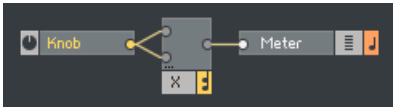
The Step Filter Module

This Module blocks all Events which carry a value that is within a certain tolerance interval from the value carried by the last Event at its input. The tolerance interval is set at the Tol input port. With the value 0 at the Tol input port, the Step Filter Module blocks all Events that carry the same value as the previous Event.

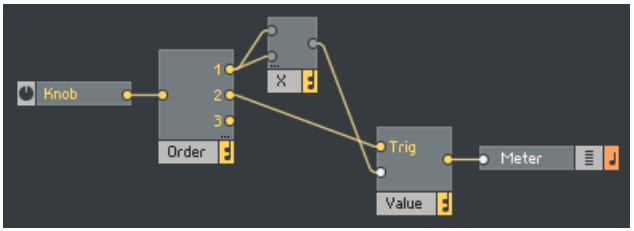
14.3.2 Triggering Values

Another way to filter unwanted Events is to use a combination of the *Order* Module and the *Value* Module.

In the following Structure, moving the *Knob* will produce two Events at the *Multiply* Module's output.



By adding an *Order* Module and a *Value* Module as illustrated below, you can effectively block the output of the *Multiply* Module until it has finished its calculations.

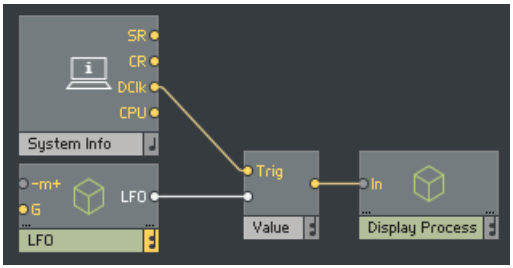


14.4 Use the Display Clock

If your ensemble has a complex display element that updates frequently, it can be beneficial to use the Display Clock.

REAKTOR's GUI processes function at around 25 frames per second (though this value can fluctuate depending on the instrument that is loaded). This means that, at most, the panel updates itself 25 times in a second.

The *System Info* Module includes an output called DClk (Display Clock) that sends an Event before each display update. You can use this clock to trigger any GUI processes in your structure, thereby only calculating them when necessary.



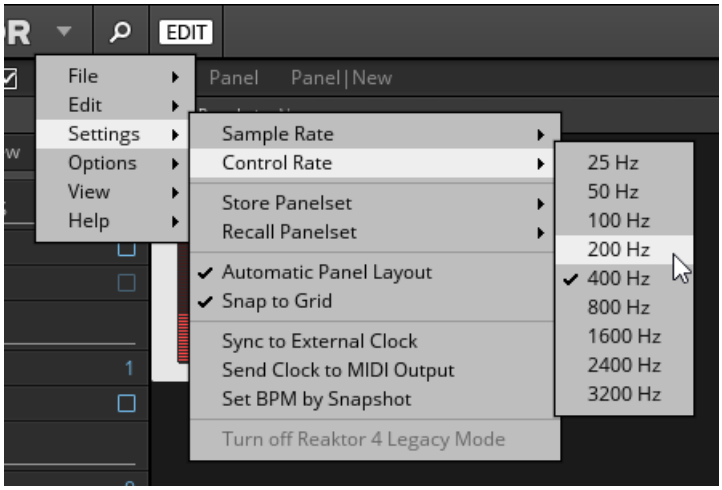
An Example of Using the Display Clock

In the example above, the LFO is driving a display process, but this process does not need to happen at the Audio Rate at which the LFO runs. The Display Clock is used to effectively down-sample the LFO value at the Display Rate, making the display process more efficient.

14.5 Reduce the Control Rate

The Control Rate is the rate at which Events are processed. By default this is set to 400Hz (400 times a second). Reducing the Control Rate means that Event signals will be processed more slowly, reducing the strain on your CPU.

1. You can reduce the Control Rate from the Settings Menu:



14.6 Use Multipliers Instead of Dividers

Division is a more expensive process for a CPU to handle than a multiplication. For example, it would be a lot more efficient to multiply by 0.5 rather than divide by 2, even though the result is the same.

14.7 Avoid unnecessary use of Send/Receive Modules

Send and *Receive* Modules can be used to make wireless connections in your structure. This can be appealing for many uses, including improving the readability of your structure.



The Send and Receive Modules

However, overusing these Modules can cause your Ensemble to have a long loading time. They also make it difficult to debug your structure, so use them where necessary, but try not to use them frivolously.

14.8 Use the Modulo Module instead of Wrap options

Some Modules that use indices (like the *Multi Display* or the *Event Table*) contain an option in their properties to wrap indices that are out of bounds. So if you had an *Event Table* with a table of 5 elements, and you sent an index value of 6 to the RX input, the Module would wrap this value around to become a 1.

In some cases, like a step sequencer, this behavior is desired. Unfortunately, this process is CPU Intensive and it is much more efficient to use the *Modulo* Module instead.



The *Event Table* and the *Audio Table* both have the wrap option selected by default.

15 An Introduction to Core

Core Cells are in some ways like a hybrid between a Module and a Macro.



A Core Cell

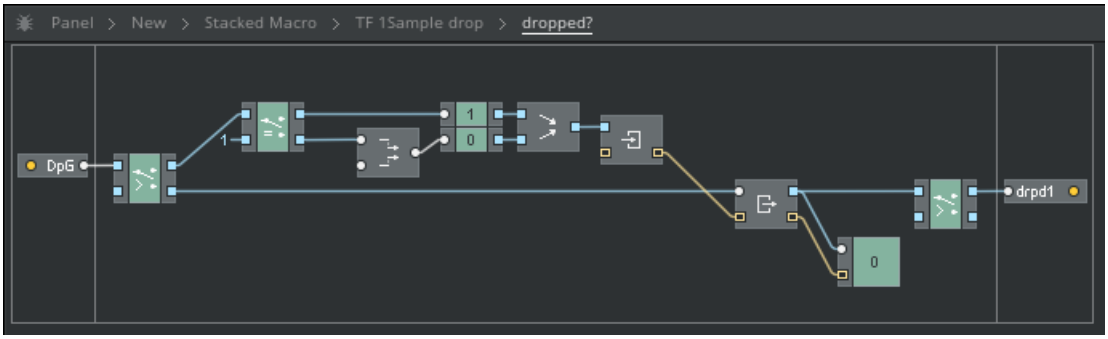
Core Cells can be identified by the icon of a cube made of dots.

The Primary level of REAKTOR treats Core Cells like Modules, but you can enter the Structure of a Core Cell like a Macro.

Although Primary and Core share the basic REAKTOR building paradigm of connecting blocks with wires to create new things, Core uses different signal types and has its own set of Modules that make it more optimized for creating low level DSP functions.

Another way to think about it is like this:

- The Primary level of REAKTOR is where you can connect an oscillator Module to a filter Module.
- The Core level is a programming environment where you would create your own oscillator and filter algorithms.



The Structure of a Core Cell

Even though Core is a deeper programming environment, Primary is still necessary for building user interfaces and receiving MIDI signals. In other words, Core requires the Primary level in order to communicate with the "outside world".

To learn how to program in Core, read the REAKTOR 6 Building in Core document.

Learning Core is not mandatory for building in REAKTOR, there are a number of Core Cells and Core Macros in the Factory Library for you to use. However, if you want to dive deeper into building and customizing REAKTOR instruments and effects, learning Core can be a very useful tool.

16 Table Framework

The tutorial in section ↑8, Drag and Drop Sampler introduces the basics of the Table Framework in a practical application. This section is a more technical reference, giving you information regarding the fundamental functionality of the Framework.

This section will also cover aspects of the Core level of REAKTOR, and thus it is advantageous to have experience with Core, or to read the REAKTOR 6 Building in Core document, before proceeding with this section.

16.1 Tables and Samples

A Table is a two-dimensional array of numbers which, on a technical level, is the same as a stereo audio file. Digital audio is essentially a sequence of numbers, each representing a sampled amplitude value. Taken out of this audio context, each sample is like an element in an array and the amplitude values are just numbers in the array.

The Table Framework allows you to share the data from an audio sample throughout an Ensemble Structure with ease. It also allows samples to be read as data in Core Cells, which enables the possibility of building sample-based DSP in Core.



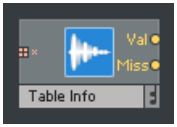
Currently the Table Framework is designed primarily for the sharing of audio data, and thus only accepts wav or aiff files; however, there are plans to expand the Table Framework in the future to make it more flexible.

16.2 Table Reference Wires

As suggested by its name, the Table Reference wire does not send the contents of a Table along it; rather, it passes a unique reference to a Table's memory location. This makes it unlike the other wires in REAKTOR, which transmit values. Table Reference wires simply join Modules and Core Cells so that they share the same Table data.

16.3 Table Reference Ports

Table Reference Ports can be identified by their orange square appearance.



The Table Info Module has a Table Reference input

For Built-in Modules, these ports are always labelled with a *.

Certain other Modules can accept Table Reference Connections:

- Merge
- Order
- Routers (all types)
- Value
- Switch
- To/From Voice

Ports that can accept Table Reference connections can be identified by their dark and hollow appearance.



The Order Module can accept Table Reference connections

16.4 The Table List Module



The Table List Module

This Module is the main Table manager and lets you maintain and select from a list of currently active Tables. It is the only Table Framework Module with Snapshot support and a Panel element.

The Panel view of the Table List is the same as the List Module, and contains a list of the currently active Tables.



The Table List Panel Element

16.4.1 Adding a Table to the Table List

There are two methods for adding a Table to the Table List:

- A Table Reference Event at the *Ins input will insert the referred Table at the index set at the Idx input and pushes any existing tables down the list.
- A Table Reference Event at the *Wr input will overwrite any Table at the index set at the Idx input. If the index is greater than the highest index currently used, then the new Table will be added to the end of the list.

16.4.2 Removing a Table from the Table List

To remove a Table from the Table List:

1. Send a value to the Idx input to select a Table index.
2. Send an event to the Del input to delete the Table at the selected index.

⇨ Upon deletion, the N and, if necessary, the Idx output values are updated.



If the currently selected Table was deleted, then the * Table Reference output is set to Null. In this case the Idx output is set to 1.

16.4.3 Selecting a Table Reference to Output

There are two methods for selecting a Table Reference:

From the Structure

1. Send a value to the Idx input to select a Table index.
2. Send an event to the Sel input to trigger the selection.

⇨ The Idx output will update and the selected Table Reference will be sent from the * out port.

When a selection is made from the Structure, the *Table List*'s Panel element will be updated to show which Table has been selected.



Idx in port values should be kept between zero and max index. Anything outside this range is ignored.

From the Panel

The Panel view of the *Table List* can be used to select a table.

1. Click on a Table Reference in the *Table List* to select it.
1.

⇨ The Idx output will update and the selected Table Reference will be sent from the * out port.

16.5 The Sample Map Module

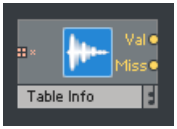


The Sample Map Module

The *Sample Map* Module allows you to send sample data from a Sample Map into a Core Cell or Table Framework Module as a Table.

- The sample is selected using the pitch (P) and velocity (Vel) inputs.
- The sample is then sent to the output as a Table when an Event is sent to the Trig input.

16.6 The Table Info Module



The Table Info Module

The *Table Info* Module outputs a specific piece of information (meta-info) about the current table, such as Sample Rate, Table Size, Loop Points, etc.

- The table meta-info to be read is selected in the Module Properties.
- If the meta-info is not present in the selected table, then the Miss output sends a value of 1.
- If the meta-info is present, then the value is sent from the Val output.

16.7 The Table Location

The location of a Table depends on its origin:

- When using the *Sample Map* Module, the location of the Table is the same as that of the sample. So if the sample is embedded, then the Table will also be embedded in the Ensemble; if the sample is not embedded, then the Table maintains a reference to the location on the disk.
- A Table that originated from an embedded sample in the *Sample Map* Module remains embedded when it is stored in a *Table List* Module.
- A sample that was dragged onto the *Mouse Area* Module and then stored in a *Table List* maintains a reference to the location on the disk, like a non-embedded sample.

16.8 Drag and Drop

The *Mouse Area* Module supports the dragging and dropping of Table References.

16.8.1 The Drag and Drop Process

To give the builder full control over which samples are accepted and when, the following sequence happens when a user drags a sample or a group of samples over the *Mouse Area*:

- When the user drags a sample or a group of samples over the *Mouse Area* this is a drop request and the DrpG out port is set to 2.
- The *Drp out port iterates through the Table Reference of all samples being dragged.
- When the Table Reference iteration is finished, the DrpG out port is set to 0 and the *Drp output is set to Null.
- If DpA is set to 1, then the samples may also be dropped by the user by releasing the mouse button.
- When the user releases the mouse button, the DrpG out port is set to 1 (indicating drop receive).

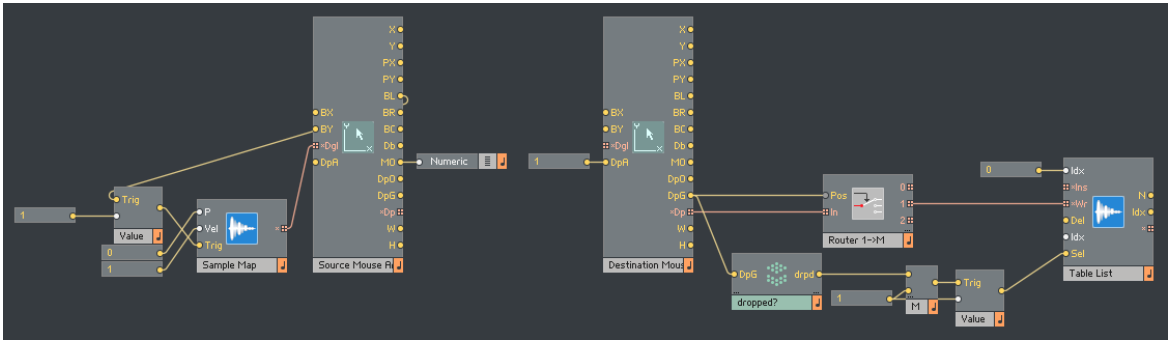
- The *Drp out port iterates through the Table Reference of all samples being dropped.
- When the Table Reference iteration is finished, the DrpG out port is set to 0 and the *Drp output is set to Null.

16.8.2 Dragging a Table Reference from a Mouse Area

You can also drag a Table Reference out of a *Mouse Area* and into another one.

When a drag event happens on the Mouse Area, the *DgI in port listens for any incoming Table References and these are added to the dragged object.

Example: Dragging a Table Reference from one Mouse Area to Another



An Example Structure Illustrating Drag and Drop between Mouse Area Modules.

In the above example, when the user clicks to start dragging from the Source *Mouse Area*, the *DgI in port is fed with a Table Reference to the sample at note 0 in the *Sample Map*.

The Table Reference can then be dragged and dropped into the Destination *Mouse Area*, where it is subsequently stored in a *Table List* Module.



The Example Structure's Panel View.

16.9 Table References in Core

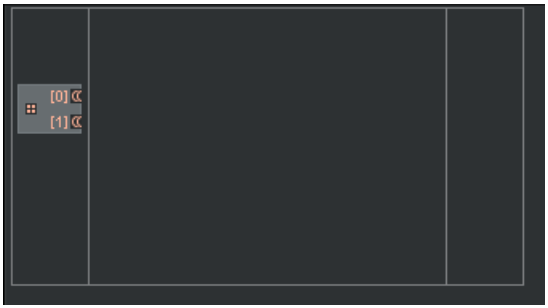
16.9.1 TableRef Input

A Table Reference can be passed into a Core Cell by using the *TableRef* input type.

1. Insert a *TableRef* input by right-clicking on the input area to the left and selecting New -> TableRef from the menu.

1.

⇨ A *TableRef* input will be created.



By default, the *TableRef* input will have 2 outputs, one for each channel of a stereo sample.

- Output [0] references the data in the left channel.

- Output [1] references the data in the right channel.
- If the audio sample is mono, then both outputs reference the same mono data.

The number of outputs from the *TableRef* can be set in its Properties:

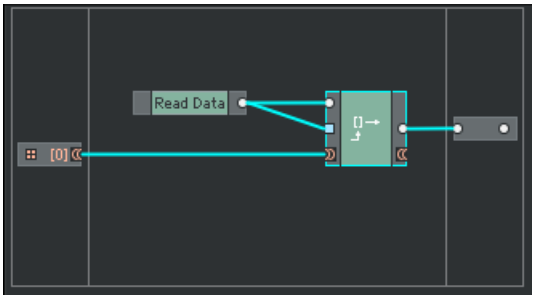
1. Select the *TableRef* input and open its Properties.
2. Navigate to the Function tab.
3. Use the Output Count parameter in the FUNCTION section to set the number of outputs.
4. Currently the output count can only be 2 or 1, so set the Output Count parameter to 1 to see how the *TableRef* input changes.

⇨ The *TableRef* input will be reduced to 1 output.



16.9.2 Accessing Table References

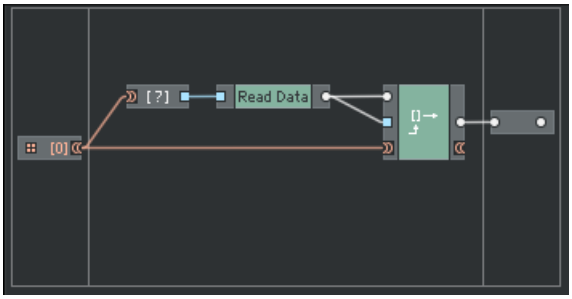
In Core, a Table Reference is accessed and treated like a normal Core Array or Table. The data in the Table can be accessed using the *Index* and *Read* Modules or the *Read []* Macro.



An example of a Core Structure that reads the data from a Table Reference

At the time of writing Tables are read only; data cannot be written into a Table Reference.

You can access the size of the Table using the *Size []* Module.



Reading the Table size using the Size [] Module

The *Size []* Module will also output an Event any time the Table Reference is updated from the Primary level. So the *Size []* Module can be used to trigger any processes that need to happen after a new Table is loaded.

16.9.3 Deleting Tables

When a Table is deleted from the *Table List* Module and the Table Reference becomes Null, the *Size []* Module in Core will not send out a new Event. This means that deleting a Table will not be registered in Core in the same way as changing a Table. Therefore, it is important to build your own structure in Core to deal with the deletion of Tables.



Although precautions have been put in place so that REAKTOR should not crash, reading data from a deleted table is a behavior that is undefined and should be avoided.

17 Compiled Core Cell Code Cache

When opening an Ensemble with large Core Cell structures in REAKTOR, a cache file of the compiled Core Cell code is saved on your hard drive. The next time you load the Ensemble, REAKTOR will access this cache file and skip the compiling stage, which significantly speeds up the loading process.

- The Compiled Core Cell Code is only used when REAKTOR is in Play Mode.
- Switching to Edit Mode when using Ensembles with large Core Cell structures causes a re-load of the Core Cells, which can cause REAKTOR to pause momentarily.

17.1 Default Cache Location

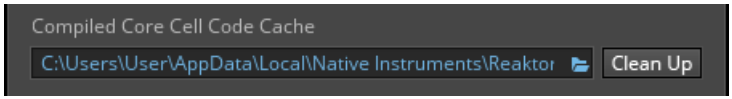
The default location of the Compiled Core Cell Code Cache is:

- Windows: C:\Users\<your user name>\AppData\Local\Native Instruments\Reaktor6\Cache \
- OS X: Macintosh HD/Users/<your user name>/Library/Application Support/Native Instruments/Reaktor6/Cache/

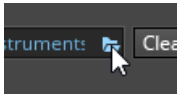
17.1.1 Changing the Compiled Core Cell Code Cache Location

To change the Compiled Core Cell Code Cache location:

1. Open the Preferences window by clicking on Preferences... in the File menu or by pressing [Ctrl] + [,] on Windows or [Cmd] + [,] on OS X.
2. Navigate to the Directories tab. At the bottom of the Directories tab you will see an area labeled Compiled Core Cell Code Cache.



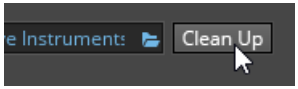
3. To change the Cache location, either click on the browser button on the left to locate the target directory on your system, or type in the location manually.



4. Click Close to close the window.
5. If your operating system prompts you, allow REAKTOR to make changes to your hard drive.

17.2 Purging the Compiled Core Cell Code Cache

1. To delete all unused or corrupt files from the Cache directory, click on the Clean Up button under Compiled Core Cell Code Cache in the Directories tab of the Preferences.



The following files will be deleted:

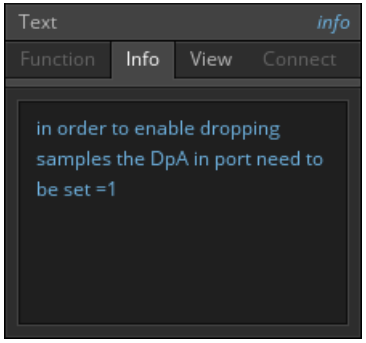
- Any Compiled Core Cell Code file for which the parent Ensemble can no longer be found
- Any corrupt Compiled Core Cell Code files

You can remove all of the files in the Cache manually, but REAKTOR will re-create the Compiled Core Cell Code files the next time you load Ensembles containing large Core Cell structures in Play mode.

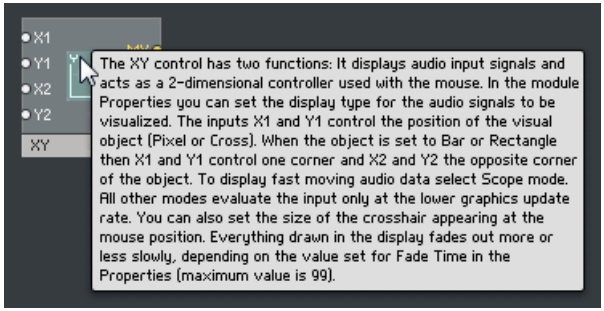
18 Module Reference

The following is a reference of all of the Primary level Modules; it will give you an overview of everything available.

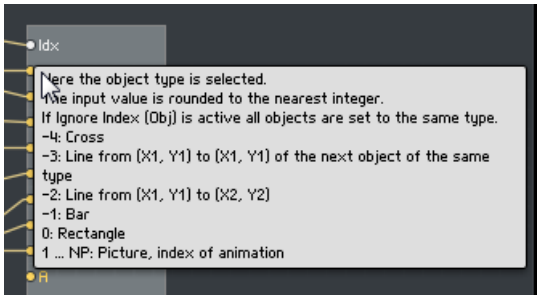
- For more information on each individual Module, you can reference the Module information from the Info tab of the Module Properties.



- You can also access this information by activating Info Hints and hovering your mouse over the Module.



- You can also get information on the Module Ports by placing the mouse over the Port.



18.1 Math

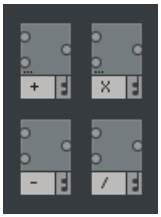
18.1.1 Constant



The Constant module holds a single value, which it outputs at initialization.

The value of the Constant module can be set in the module properties or in the structure by double clicking on the module.

18.1.2 Basic Modules



The basic math Modules cover the fundamental mathematical functions:

- Add

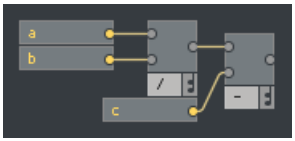
- Multiply
- Subtract
- Divide, x/y

These Modules all accept two inputs to produce a single result based on their functions. This result will be output whenever an event enters either the top or bottom input.

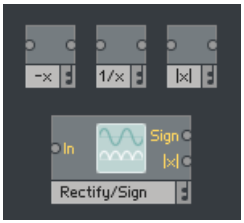
The top input will be the first number of the function and the bottom input will be the second. So, in REAKTOR format, the following formula:

```
(a / b) - c
```

Would be created like this:



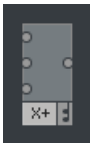
18.1.3 Modifiers



These Modules all take a single input and perform a function to produce a modified value.

- Invert, -x: Inverts the input (the equivalent of multiplying it by -1).
- Reciprocal, 1/x: The output of this module is the result of dividing 1 by the input value.
- Rectify, |x|: Can also be referred to as an 'absolute' function. The input has its sign removed, so that it is always a positive number.
- Rectify/Sign: The same process as the Rectify module, but with an additional output port which outputs a value (-1 or 1) depending on whether or not the input value is negative.

18.1.4 Mult/Add, a*b+c



Outputs the result of the formula:

```
(a * b) + c
```

The inputs from top to bottom correspond with the variables in the formula from left to right. So the top input is `a` and the bottom is `c`.

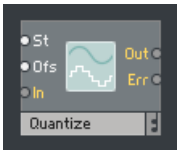
18.1.5 Modulo



This Module takes two input values and outputs two values:

- The Div port outputs the truncated division of the inputs, so input values of A=5 and B=2 would produce an output of 2.
- The Mod port outputs the remainder of the division, so the input values of A=5 and B=2 would produce 1.

18.1.6 Quantize



Takes the input value and quantizes it according to the settings.

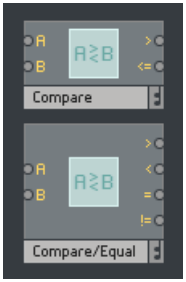
The module outputs the quantized value to the Out port, and outputs the difference between the original input value and the output value from the Err (error) port.

The settings can be set from the St and Ofs inputs:

- St (step size): sets the size of the steps to which the input should be quantized. So with an input of 3.2 and a St setting of 1, the Out value would be 3.
- Ofs (offset): applies an offset to the quantized value. So if the quantized value is 3, and the Ofs is set to 0.1, the Out value will change to 3.1.

This module can be used to create a simple bit-crusher effect.

18.1.7 Comparison



There are two comparison Modules that apply logic operations to two numeric inputs. If the operation is true, the output port sends a value of 1, if it is false it outputs a 0.

The modules are:

- Compare: Outputs two integer values based on testing the > (greater than) and <= (less than or equal) logic operations.
- Compare Equal: Outputs the results of 4 logic operations:
  - > (greater than)
  - < (less than)
  - = (equal to)

- != (not equal to)

As with all the mathematical modules, the top input is placed to the left of the logic operation, and the bottom input is placed to the right. Thus input A is always compared against input B in the following manner:

```
A > B; A<=B; etc…
```

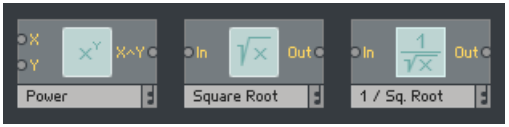
18.1.8 Logarithms (Converters)



The logarithmic Modules in REAKTOR are primarily used to convert from one unit to another. There are 4 different modules for performing different conversions:

- Expon. (A): Converts level values in decibels (dB) to amplitude factor values. An input of -6 will produce an output of 0.5
- Expon. (F): Converts MIDI pitch values to frequency values in hertz (Hz). An input of 69 (A3) will produce an output around 440Hz.
- Log. (A): Converts amplitude factors to dB. An input of 1 will produce an output of 0.
- Log. (F): Converts frequency values (in Hz) to a MIDI pitch number. An input of 220 will produce an output of 57 (A2)

18.1.9 Power and Roots

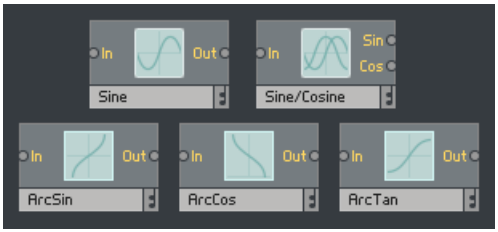


There are three mathematical Modules relating to powers and square roots.

- Power: Outputs the result of the formula `X^Y`.

- Square Root: Calculates the square root of the input value.
- 1 / Sq. Root: Calculates the reciprocal square root. The equivalent of dividing 1 by the output of the Square Root Module.

18.1.10 Trigonometric Functions



These Modules all take a single input and perform a trigonometric function to produce an output value.

The following Modules and functions are available:

- Sine
- Sine/Cosine

Inverse Trigonometric Functions

- ArcSin
- ArcCos
- ArcTan

The inverse trigonometric Modules have their inputs and/or outputs limited by necessity.

- ArcSin and ArcCos clip their inputs when they exceed the range -1 to +1
- ArcTan clips its output to the range -0.25 to +0.25

18.2 MIDI In

18.2.1 Pitch



There are two MIDI In Modules that relate to pitch:

- Note Pitch: Outputs the MIDI note value of any note message received by the Ensemble.
- PitchBend: Outputs the value of the PitchBend controller. The range of the output is set in the Properties.

18.2.2 Gate and Velocity



REAKTOR includes several Modules relating to Velocity and Gate.

Gate signals are used to communicate whenever a MIDI key is pressed and released. Velocity is the value of the strength at which these actions happen.

Gate

- Gate: Outputs the value of the note on velocity when a key is pressed, and a value of 0 when it is released.
- ST Gate (Single Trig. Gate): A monophonic variant of the Gate Module. This Module outputs a single value with the first note on and ignores subsequent note on messages. It then outputs a 0 with a note off and ignores any subsequent note off messages. Essentially this Module can be used to see whether or not the keyboard in use.
- Sel. Gate (Sel. Note Gate): Sends the gate signal from a single note, which is selected in the Module Properties.

Velocity

- On. Vel (On Velocity): Outputs Note On Velocity values, ignoring Note Off messages.
- Off Vel. (Off Velocity): Outputs Note Off Velocities, ignoring Note On messages.



Most MIDI Keyboards do not send Note Off Velocities, as such the Off Vel. Module is not guaranteed to output values.

18.2.3 Controllers



These Modules can be used to access MIDI controllers (CC) as well as aftertouch. They output a value when they receive their relative MIDI message.

- Controller: Outputs a MIDI controller value. The MIDI CC number can be selected in the Module Properties.
- Chan. AT (Ch. Aftertouch): Outputs the channel aftertouch value.
- Poly AT (Poly Aftertouch): Outputs the polyphonic aftertouch values.
- SelPolyAT (Sel. Poly Aftertouch): Outputs the polyphonic aftertouch of a single key (selected from the Module Properties).

18.2.4 Transport and Clock



There are four Modules that relate to the transport and clock (i.e. playback).

Transport

- The Start/Stop Module sends a gate signal that is set to 1 when the transport starts playback and 0 when it is stopped.
- Song Position outputs the current song position in 1/96 resolution. It has two Outputs
  - 96: An Event signal output. Since events can have their timing controlled, this output can be useful when building sequencers.
  - 96a: An Audio signal output, for sample-accurate timing.

Clock

- Clock (1/96 Clock): Outputs an event on every 1/96th note (i.e. 24 times per beat).
- Sync Pls. (Sync Clock): A MIDI synchronized clock. Its rate, length and values can all be set in the Module Properties.

18.2.5 Program Change



This Module outputs the value of MIDI Program Change messages.

18.2.6 Channel Message



The Channel Message Module outputs all MIDI messages that relate to MIDI channels:

- Note On
- Note Off
- Controllers (including Pitchbend and Aftertouch)
- Program Changes

The outputs send the MIDI message information:

- St (Status): The type of MIDI message.
- Ch (Channel): The MIDI channel.
- Nr (Number): For Note On messages, this would be the note number; for controllers, this is the CC number; and so on.
- Val (Value): The value of the message.

18.3 MIDI Out

18.3.1 Note Pitch and Gate



This Module is used to send MIDI Note Messages to other MIDI devices. The MIDI message will be sent when an event arrives at the G input.

- P: Sets the MIDI Note number.
- G: Sets the MIDI Note On velocity. A value of 0 will send a Note Off message.

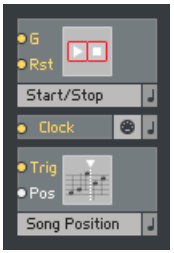
18.3.2 Controllers



These Modules can be used to send various MIDI Controller messages.

- Pitchbend: Sends Pitchbend messages.
- Controller: Sends MIDI Controller messages. The Controller Number is set in the Module Properties.
- Chan. AT (Ch. Aftertouch): Sends Channel Aftertouch messages.
- Poly AT (Poly Aftertouch): Sends Poly Aftertouch messages. The P input is used to set the Note Number, in the same manner as the Note Pitch and Gate Module.
- SelPolyAT (Sel. Poly Aftertouch): Sends a Poly Aftertouch message for a single note. The Note Number is set in the Module Properties.

18.3.3 Transport and Clock



These Modules can be used to send MIDI Transport and Clock messages.

- Start/Stop: Sends Transport Start/Continue/Stop Messages.
- Clock (1/96 Clock): An Event at the input sends a 1/96 clock MIDI message.
- Song Position: Sends a 1/96 resolution Song Position value when an Event with a positive value arrives at the Trig input.

18.3.4 Channel Message



The Channel Message Module can send any MIDI message that relates to MIDI channels:

- Note On
- Note Off
- Controllers (including Pitchbend and Aftertouch)
- Program Changes

The inputs set the MIDI message information:

- St (Status): The type of MIDI message.
- Ch (Channel): The MIDI channel.

- Nr (Number): For Note On messages, this would be the note number; for controllers, this is the CC number; and so on.
- Val (Value): The value of the message.



An Event at the St (Status) input triggers the MIDI message, so be sure to set this value last.

18.4 OSC

18.4.1 Send/Receive



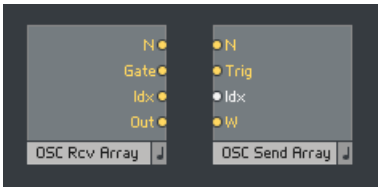
These Modules can be used to Send and Receive OSC messages.

- The Address Pattern is set in the Connect Tab of the Module Properties.
- If the OSC message is an array, these Modules can have multiple inputs/outputs for each of the indices of the array.



When using array messages it might be better to use the OSC Array Modules.

18.4.2 Arrays



These Modules are used to send/receive OSC arrays. Rather than each index having an individual output, these Modules use the principals of iteration to send arrays of data.

- The Address Pattern is set in the Connect Tab of the Module Properties.

The tutorial ↑7, Additive Synthesizer ↑7, Additive Synthesizer covers the principals of how iteration works in REAKTOR.

18.5 Panel

18.5.1 Basic Controls



There are three basic Panel Controls. They are designed to include common functions found on GUI elements of music instruments (like MIDI Learn and Automation).

The three controls are:

- Fader
- Knob

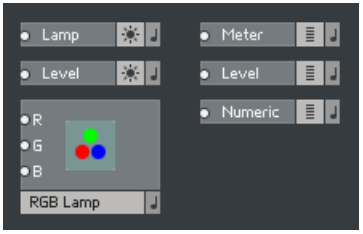
- Button



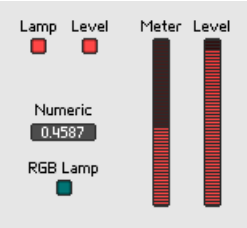
The Basic Controls on the Panel in their default states.

- All of these Modules can be given custom graphics.
- The Fader can have its orientation changed in its Module Properties.

18.5.2 Basic Displays



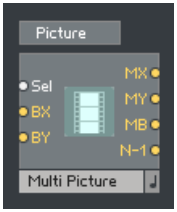
- Lamp: A small colored box, the brightness of which is controlled by the input value.
- Level Lamp: Like the Lamp, but the input uses the dB scale and is smoothed to respond better to audio signals.
- RGB Lamp: A Lamp with three inputs for setting the Red, Blue, and Green color values of the Lamp.
- Meter: Displays the input value as a bar.
- Level Meter: Like the Meter, but the input uses the dB scale and is smoothed to respond better to audio signals.
- Numeric Readout: Displays the input value.



The Basic Display Modules on the Panel in their default states.

The colors of the Lamps and Meters are set in the Module Properties.

18.5.3 Pictures



The Picture Modules are used to display images on the Panel.

- Picture: Displays a single static image.
- Multi Picture: Can be used to read an animation-style image file. It also has outputs for mouse interaction and thus can be used as a custom control.

18.5.4 Text



The Text Modules are used to display text on the Panel.

- Text: Displays static text.
- Multi Text: Contains a list of text options; the displayed is selected from the input.

The text displayed by these Modules is set in the Info tab of the Module Properties.

These Modules have several font options, which are set in the Module Properties. There are 9 available fonts, the color and size of which are fully customizable.



The Available Fonts

18.5.5 List

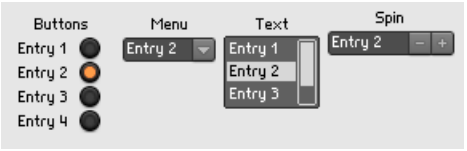


The List Module offers the user a selection of entries that they can choose from.

The entries and their values are set in the Module Properties.

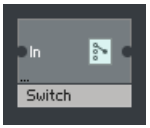
This Module can have four different appearances:

- Radio Buttons
- Dropdown Menu
- Text List
- Spin



The Different List Appearance Options

18.5.6 Switch



In the Structure, the Switch functions like an M -> 1 Router. It is used to select between multiple inputs and then routes the selected input to the output.

The main benefit of using a Switch is that it deactivates any Modules attached to the unselected inputs. Thus the Switch can be used as a CPU saving feature.

The other benefit is that the Switch already includes a Panel element, in the form of a Radio Button List.

18.5.7 Receive



The Receive Module can connect to any Send Module and can be used to create wireless connections.

On the Panel, the Receive Module is presented like a List.

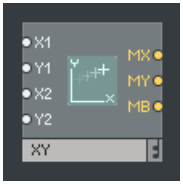
This Module is the same as the Receive Module located in the Terminal section of the Built-in Modules.



See section ↑18.17.2, Send/Receive ↑18.17.2, Send/Receive for information on the Terminal Modules.

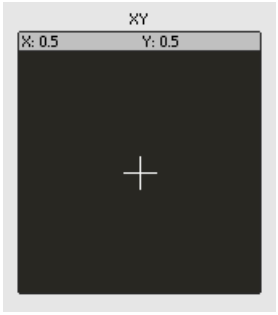
Note that Send/Receive Modules should be used in moderation due to their potential performance overhead. See section ↑14.7, Avoid unnecessary use of Send/Receive Modules ↑14.7, Avoid unnecessary use of Send/Receive Modules for more details.

18.5.8 XY



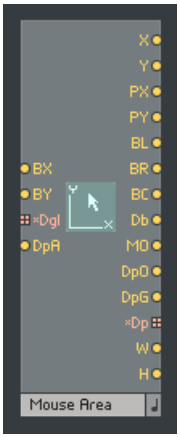
The XY Module is an area that outputs the mouse X and Y positions while the mouse button is pressed.

This Module also has a display element: it can accept up to two coordinates and draw an object in that space. This feature can even be used by audio signals as a scope.



The XY Module on the Panel

18.5.9 Mouse Area



This Module sends detailed information about the state of the mouse.

As well as X, Y coordinates and button states, the Mouse Area can also access Double-clicking and the mouse-over state.

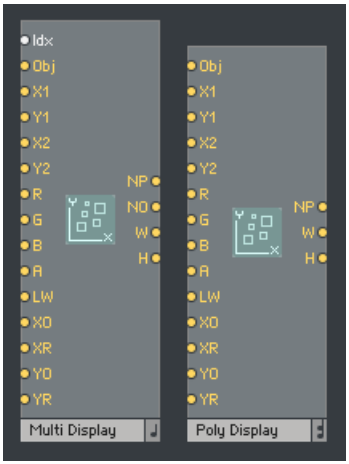
All of these features make the Mouse Area an important asset when creating custom interface elements.

The Mouse Area is used to create a custom control in the tutorial in section ↑6, Advanced Step Sequencer ↑6, Advanced Step Sequencer.

The Mouse Area can also be used in conjunction with the Table Framework to accept audio files for drag and drop.

The Drag and Drop feature of the Mouse Area is examined in section ↑8, Drag and Drop Sampler ↑8, Drag and Drop Sampler.

18.5.10 Advanced Displays



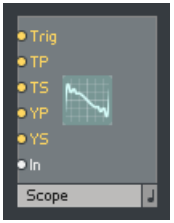
The Multi Display and the Poly Display and be used to create various numbers of objects.

These objects can be simple shapes (like lines and rectangles) or they can be custom images.

- The Multi Display can have the number of objects defined in its Properties.
- The Poly Display creates an object for each polyphonic voice in the Instrument.

The tutorial in section ↑6, Advanced Step Sequencer ↑6, Advanced Step Sequencer shows how to use a Multi Display.

18.5.11 Scope

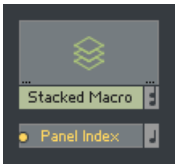


The Scope Module can be used to create oscilloscopes.

It samples the signal at the In input and then draws a trace of the input over time when a positive Event arrives at the Trig input.

The ranges and offsets of the time axis and Y axis can be set using the inputs.

18.5.12 Stacked Macro



Stacked Macros can be used to create dynamic control panels.

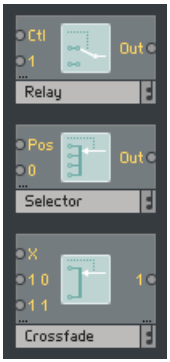
- The Stacked Macro can contain any number of Macros, but it will only display the Panel of one of them at any one time.
- The displayed Macro is selected using the Panel Index, which must be placed inside the Stacked Macro.



Section ↑10.8, Stacked Macros ↑10.8, Stacked Macros describes the use of Stacked Macros in more detail.

18.6 Signal Path

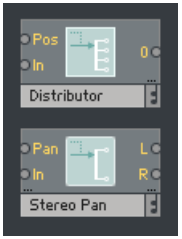
18.6.1 Selectors



There are three Signal Path Modules that can be used to select between multiple signals to output a single signal.

- Relay: A switch style selector.
- Selector: Can be used to blend between different inputs.
- Crossfade: Like the Selector, but it pairs the signals and crossfades them as a stereo pair.

18.6.2 Distributors



There are two distribution Modules. They take an input signal and route it to multiple outputs.

- Distributor: This Module is like the Selector in reverse. It takes an input and routes it to different outputs.
- Stereo Pan: A variation of the Distributor that takes an input and blends it between two outputs (one for each of the stereo channels).

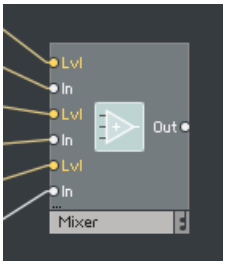
18.6.3 Mixers



These Modules act like audio mixing amplifiers.

- Mixer: Takes an input signal (In) and alters its level using the decibel scale (Lvl).
- Stereo Mixer: An expansion of the Mixer Module, with two outputs (one for each of the stereo channels) and a Pan input to blend the input between these outputs.

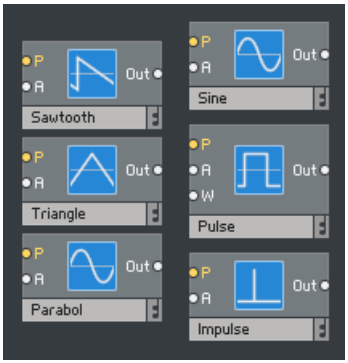
You can add multiple inputs to these Modules, meaning that a single Mixer Module can be used to mix together several signals into one.



The Mixer with multiple inputs

18.7 Oscillators

18.7.1 Basic Audio Oscillators



The basic audio oscillators produce an audio rate waveform based on the two input parameters:

- P - MIDI Pitch
- A - Amplitude

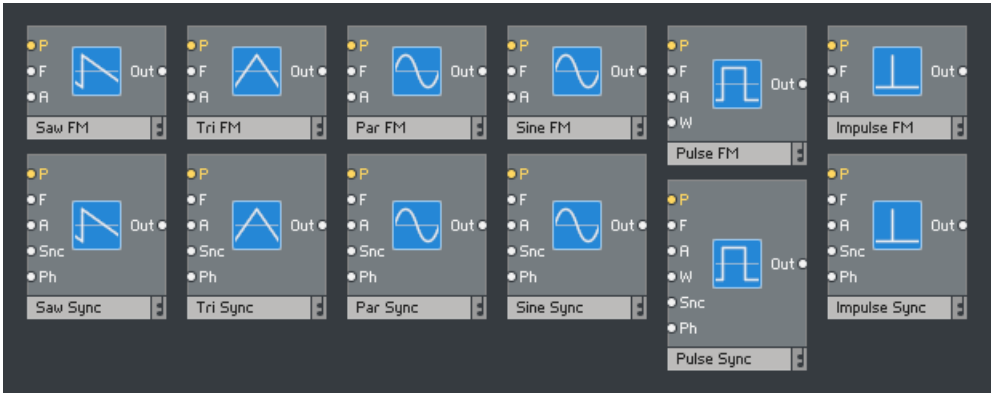
The basic oscillator modules are available in the following waveforms:

- Sawtooth
- Triangle
- Parabol
- Sine
- Pulse
- Impulse

The Pulse waveform has an additional input (W) that controls the width of the pulse cycle.

FM and Sync variants

All of the basic audio oscillators are all also available as FM and Sync oscillators.



The FM oscillators add an additional input, which allow them to be used for audio rate Frequency Modulation (FM):

- F - Linear frequency control, which is added to the frequency of the P input.

The Sync oscillators add two inputs to the FM oscillator:

- Snc - when the input rises above zero, the oscillator re-starts at the phase set by the Ph input.
- Ph - sets the re-starting phase point.

The FM and Sync variants will have a heavier impact on the CPU, so only use them when you need the additional functionality.

18.7.2 Variants

Some of the Basic Audio Oscillators also feature unique Variants. These offer extra control inputs that affect the shape of the waveform in some way.

Saw



- Saw Pulse: Provides control over the steepness of the saw slope.
- Bi-Saw: Provides control over the width of the saw.

Triangle/Parabol

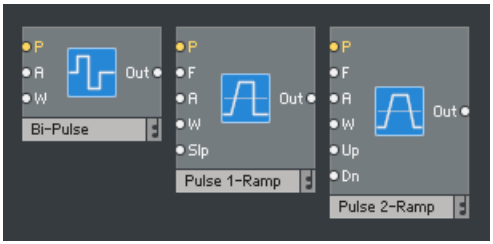


- Tri/Par Symm: Includes two outputs for Triangle and Parabol waveforms. The Symmetry of the waveforms can be controlled via the W (width) port.

This Module gives you the ability to morph between a Triangle wave and a Saw wave.

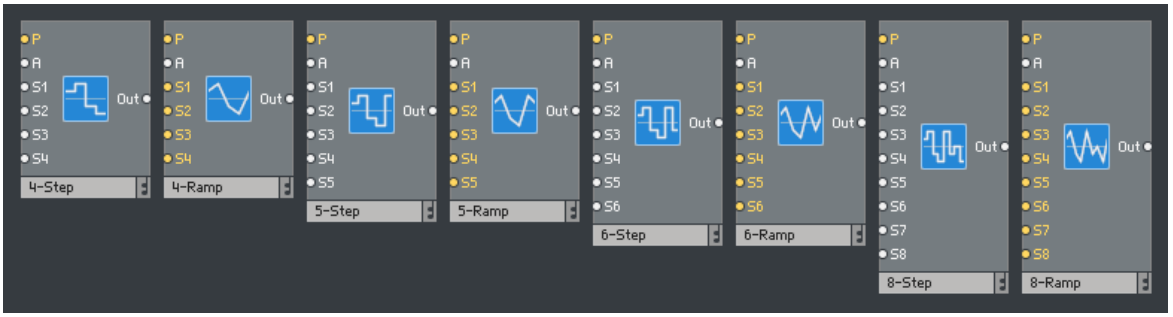
- Par PWM: A Parabol waveform that can have its width controlled. The resulting shapes are different from the Par output of the Tri/Par Symm Module.

Pulse



- Bi-Pulse: The Width (W) input controls the width of the whole pulse waveform, rather than one side of it.
- Pulse 1-Ramp: Provides control over the slope steepness of the rising edge of the waveform.
- Pulse 2-Ramp: Provides control over the slope steepness for both the rising and falling edges of the waveform.

18.7.3 Multi-step



The Multi-Step Oscillators enables you to draw your own waveforms by entering values for multiple steps in the waveform. The oscillator reads through these values like a table.

They come in two main types with varying numbers of steps:

- **Step**: Values change instantaneously, producing a blocky waveform.
  - 4-Step
  - 5-Step
  - 6-Step
  - 8-Step
- **Ramp**: Values change gradually, producing a smoother waveform.
  - 4-Ramp
  - 5-Ramp
  - 6-Ramp

- 8-Ramp

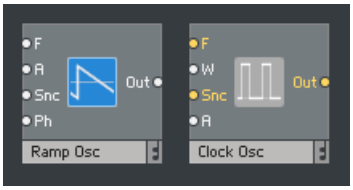
18.7.4 Noise



There are three Modules that use random numbers to produce a signal:

- Noise: Randomly outputs a value of -0.5*A or 0.5*A at the Sample Rate. Filter Modules can be used to change the color of the noise.
- Random: Produces a random number at a rate defined by the P input, and in a range defined by the A input.
- Geiger: Outputs clicks (a value of 1 followed instantly by a value of 0) at definably random intervals.
  - The rate of the clicks is controlled by the P input and the randomization is controlled by the Rnd input.
  - The Geiger Module also has an Event output, which generates Events at the same random interval. For example, it can be used to trigger Envelopes at random intervals.

18.7.5 Utility



The Utility Oscillators are not designed to be used as Audio signals, but are tools that behave like oscillators.

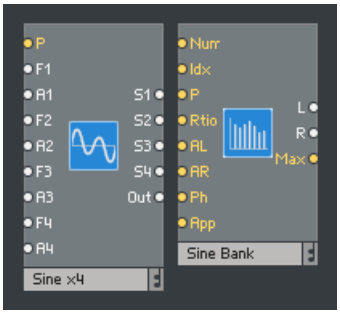
- The Ramp Osc ramps from 0 to the value defined at input A over the period defined by the F input.

The Ramp Osc is a useful driver for wavetables oscillators.

- The Clock Osc produces an Event of value A followed by an Event with a value of 0 at a rate defined at the F input.

The Clock Osc is used in the tutorial in section ↑5, Basic Step Sequencer ↑5, Basic Step Sequencer

18.7.6 Additive



Additive synthesis uses large numbers of sine waves to create complex sounds.

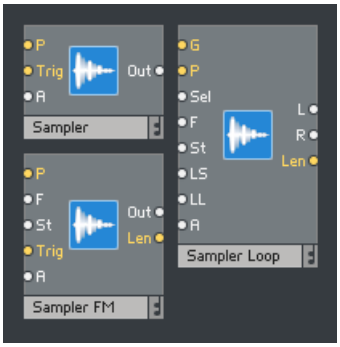
REAKTOR has two oscillators designed to use this technique:

- The Sine x4 Module essentially stacks four Sine Oscillator Modules inside one Module.
- The Sine Bank is optimized to produce large numbers of sine waves. Therefore, rather than having individual inputs for the parameters of each of the sine waves, the parameters are set as values in an array.

The Sine Bank is one of the main subjects in the tutorial in section ↑7, Additive Synthesizer ↑7, Additive Synthesizer.

18.8 Samplers

18.8.1 Basic

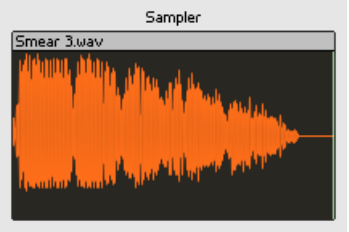


The Basic Samplers play back audio samples from a Sample Map.

- Sampler: Plays back the samples as they are mapped.
- Sampler FM: Provides the ability to modulate the frequency of the sample playback, as well as the option to define the sample start offset.
- Sampler Loop: Like the Sampler FM Module, but with inputs for settings the loop start and end points for the sample.

Sample Maps and the Sample Map Editor are described in more detail in the REAKTOR 6 Diving Deeper document.

Sampler Modules all have a Panel element, which is a display of the current sample waveform, as well as overlays for the loop points and the playback head position.



The Sampler Module Panel Element

18.8.2 Granular



The Granular Samplers cut the samples into small sections (called grains) and process each grain separately. This can produce choppy effects, or be used to individually control pitch and time.

- Resynth: The standard granular sampler, with control over the grain settings and loop points.
- Pitch Former: Like the Resynth, but with a Formant Shift (FS) input.

- Grain Cloud: Creates a number of grains (i.e. a cloud), with control over the envelope of the grains, and independent control over the distance between the grains and their length. Several of the parameters have Jitter inputs that control a randomization of their respective partner.

18.8.3 Beat Loop



Beat Loop is a granular sampler that aims to keep the tempo of the sample locked to the tempo defined by the Clk (1/96 clock) input.

This Module requires that the sample is an exact number of bars, preferably an even number like 1, 2, 4, 8, 16, etc...

18.8.4 Lookup



The Sample Lookup Module allows you to scrub through the waveform of the sample by manually setting the playhead position using the Pos input.

Unlike the other Sampler Modules, this Module does not use the Sample Map Editor. It can only hold one sample at a time and that sample is set in the Module Properties.

18.9 Table Framework

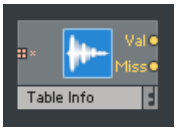
18.9.1 Table List



This Module is the main Table manager and lets you maintain the list of active Tables, reorder and select between them.

The Table List is the only table reference module with Snapshot support.

18.9.2 Table Info



The Table Info outputs a specific piece of information (meta-info) about the current table, such as Sample Rate, Table Size, Loop Points, etc.

- The table meta-info to be read is selected in the Module Properties.
- If the meta-info is not present in the selected table, then the Miss output sends a value of 1.
- If the meta-info is present, then the value is sent from the Val output.

18.9.3 Sample Map



This Module takes sample information from the Sample Map and outputs it as a table reference. It similar in function to the Sampler Module (see section ↑18.8, Samplers ↑18.8, Samplers), only it outputs the selected sample as a table reference rather than as an audio signal.

- The sample is selected using the pitch (P) and velocity (Vel) inputs.
- The sample is then sent to the output as a table when an event is sent to the Trig input.

18.10 Sequencer

18.10.1 Basic



The Sequencer Modules allow you to create sequences of values, which are read at a rate defined by the Clk input.

- Every time a rising value passes 0 at the Clk input, the sequence progresses to the next step. When the sequence passes the last step, it wraps around again.
- You can reset the sequence position using the Rst input.

There are 4 Sequencer Modules, each defined by their length:

- 6-Step
- 8-Step
- 12-Step
- 16-Step

18.10.2 Multiplex 16



The Multiplex 16 Module is more like a value selector than the Sequencer Modules.

- You can set the length of the sequence using the Len input and then set the sequence position using the Pos input.
- A value is read every time the Pos input receives an Event, so the Pos input also acts as the clock input.

18.11 LFO, Envelope

18.11.1 LFO

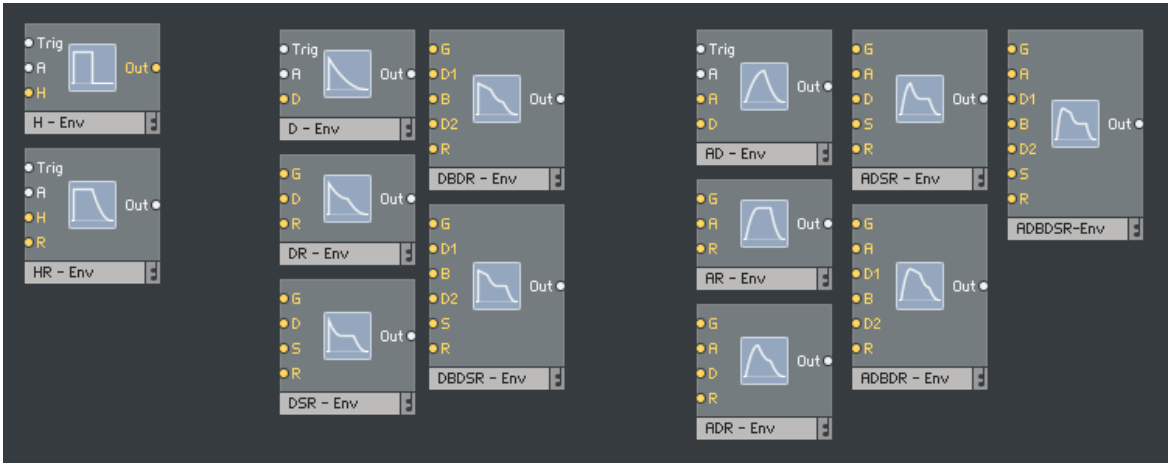


LFOs are Low Frequency Oscillators, i.e. Oscillators that are designed to run at a frequency lower than the range of human hearing.

The LFO Modules are Oscillators that output Events rather than Audio signals.

- LFO: A standard LFO with three different waveforms available from individual outputs.
- Slow Random: Produces a random value at the Control Rate. The values are smoothed by an internal filter, the cutoff of which is defined at the F input.

18.11.2 Envelopes



There are 13 different Envelope Modules, each with a different set of parameters that affect the envelope shape and behaviors in different ways. They all use the following abbreviations to show how they function:

- A: Attack - A fade-in at the start of the envelope.
- D: Decay - A gradual reduction in level.
- H: Hold - Keeps a value level for a set amount of time.
- B: Break - A Break-point level in the envelope. Usually this is a level between two decay phases.
- S: Sustain - The level at which the envelope remains for as long as the gate is active.
- R: Release - The fade-out time after the gate is deactivated.

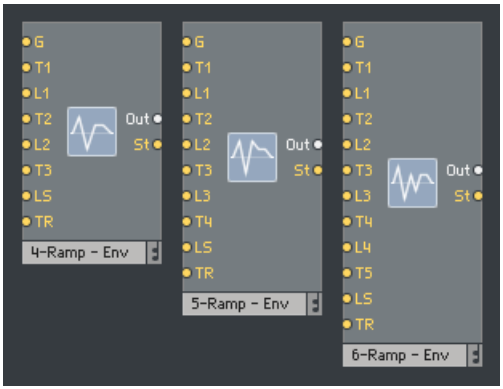
The available Envelopes mix and match these functions.



The most common Envelope type for synthesizer is the ADSR Envelope.

- With the exception of the HR Envelope, all of the Envelopes with a release phase (R) use a Gate (G) input as this tells the envelope whether or not the note is being held. The positive Gate value also sets the peak value of the Envelope.
- The Other Envelopes use an A input to set the peak value of the Envelope, and a Trig input to trigger the start of the envelope.

18.11.3 Multi-ramp



The Multi-ramp Envelopes allow you to define the shape of the Envelope.

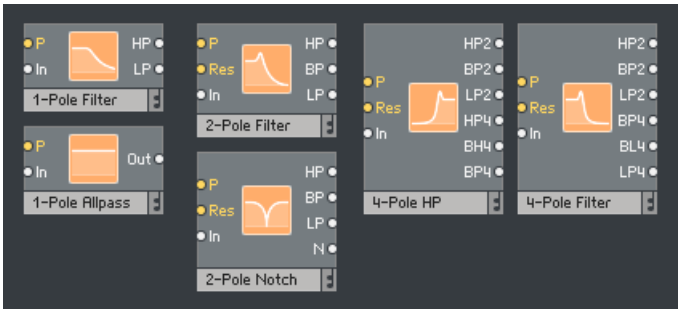
- Each point is defined by the time it takes to get there (T) and its level (L).
- These Envelopes also have a sustain level (LS) and a release time (TR).

There are three Multi-ramp Envelopes, defined by the number of ramps available (note that the release phase counts as a ramp):

- 4-Ramp
- 5-Ramp
- 6-Ramp

18.12 Filters

18.12.1 Basic



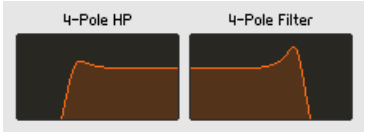
The Basic Filter Modules are designed to filter audio signals.

There are three steepness levels available:

- -Pole (6dB/Octave). The 1-Pole filters do not have resonance options.
- -Pole (12dB/Octave)
- -Pole (24dB/Octave)

As the number of poles increases, so do the number of available filter outputs. Note that the 4-Pole filters both contain the options available in the 2-Pole Filter.

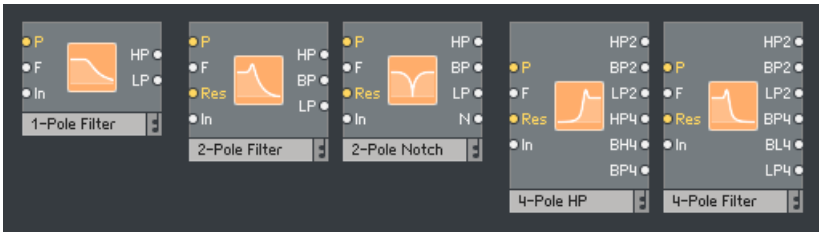
The Filters also have a panel element that can be activated from the View tap of the Module Properties.



The Panel Display of the Filter Modules

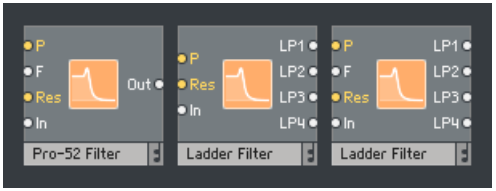
Note that the plot shown is that of the primary filter type. So the 4-Pole HP Filter will always show a high-pass filter plot, even if you are using the low-pass or band-pass outputs.

FM



The Basic Filters also have FM variants, which can have their cutoff frequency modulated at the audio rate.

18.12.2 Modeled



There are three Filter Modules that are based on hardware filters:

- Pro-52: This is an emulation of the Prophet 5 filter that was used in the now discontinued Pro-52 software.
- Ladder: An emulation of the classic Ladder Filter design. It has four outputs for different numbers of poles (i.e. the filter steepness).
- Ladder FM: Similar to the Ladder Module, but with an F input for frequency modulation.

18.12.3 Modal Bank

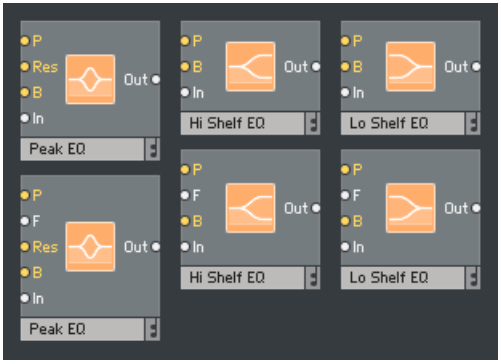


The Modal Bank creates a large number of resonant band-pass filters. When these filters are excited by an input signal, they produce a tone. This process is called modal synthesis and is generally considered to be good for physical modeling (although it can take some fine-tuning to get the right results.

The Modal Bank functions similarly to the Sine Bank, in that the parameters of the individual band-pass filters are set in an array.

The tutorial in section ↑7, Additive Synthesizer ↑7, Additive Synthesizer goes into more detail about using the Sine Bank. The same principals apply to the Modal Bank.

18.12.4 Equalizers



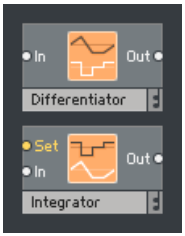
Equalizers (or EQ for short) are used to alter the levels of a select band of frequencies.

There are three EQ types available:

- Peak EQ
- Hi Shelf EQ
- Lo Shelf EQ

Each type comes with and FM variant.

18.12.5 Differentiator, Integrator



- Differentiator: A signal slope detector. Calculates and outputs how fast the input signal is rising or falling.
- Integrator: A ramp generator. Creates a ramp with a slope defined by the value at the input.

18.13 Delay

18.13.1 Basic



The Delay Modules take an input signal and delay it in time.

There are three Basic Delay Modules:

- Unit Delay (T): Delays the input signal by one sample. This Module is useful in feedback paths.

If you do not insert a Unit Delay in your feedback paths, REAKTOR will do it automatically. The location of the delay is marked by a Z.

- Single Delay: Delays the input signal by the time specified at the Dly Port.

The Single Delay is the main Module in the tutorial in section ↑4, Echo Effect Macro ↑4, Echo Effect Macro.

- Multi-Tap Delay: Can create up to 8 different delayed signals from a single input.

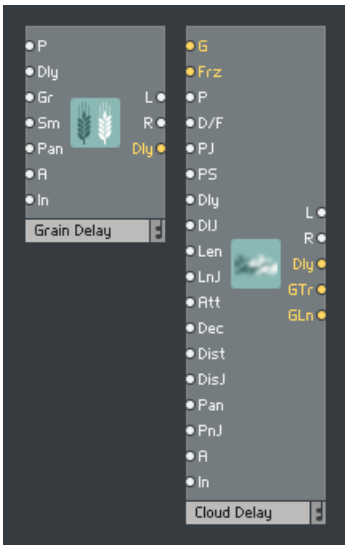
18.13.2 Diffuse



The Diffuser Delay Module is an all-pass filter containing a delay line with feedback. Its function is to smear (decorrelate) the input signal without emphasizing any particular frequency component. The delay time can be continuously modulated by an Audio signal.

These features make the Diffuser Delay a good component for a reverb effect.

18.13.3 Granular



The Granular Delay Modules cut up the input signal into small sections (grains). Essentially these Modules are effect counterparts to the Granular Sampler Modules.

- Grain Delay: This module allows you to modulate the delay time without affecting the pitch, and vice versa.
- Grain Cloud Delay: Gives you much more control over the parameters of the grains. This Module is an effect version of the Grain Cloud from the Samplers.

18.14 Audio Modifier

18.14.1 Saturators and Distortions



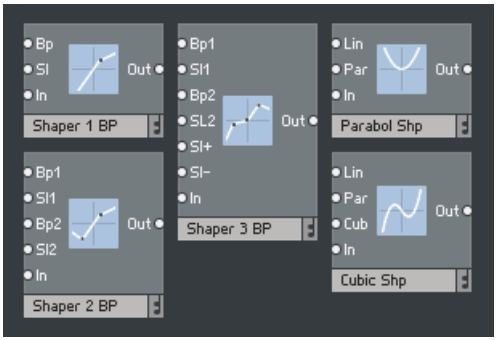
These Modules distort the input signal.

There are seven Modules available:

- Saturator: A soft saturation Module. The saturation amount depends on the level of the input.
- Saturator 2: A more advanced version of the Saturator. Gives you access to additional parameters that affect the character of the saturation.
- Clipper: Clips the input when it goes beyond certain threshold levels, which are defined at the Max and Min inputs.

- Mod. Clipper: A variation on the Clipper, optimized for modulation of the clipping threshold.
- Mirror 1: Rather than clipping the signal, this Module "reflects" any signal that passes above the threshold level.
- Mirror 2: Similar to the Mirror 1 Module, but is able to reflect signals at both an upper limit and a lower limit.
- Chopper: When the value at the M Port is positive, the signal at the In Port is multiplied by the value at the X Port. When the M value is negative, the In signal is left unchanged.

18.14.2 Shapers



The Shaper Modules also distort the input signal, but use waveshaping techniques.

There are three Break-Point (BP) based Shapers:

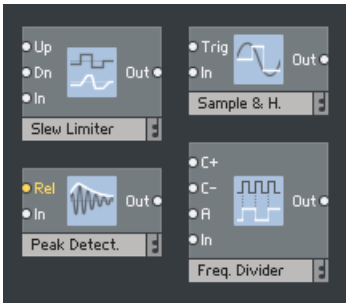
- Shaper 1 BP: When the input rises above the break-point (Bp), its slope is adjusted by the Sl value.
- Shaper 2 BP: Like the Shaper 1BP, but with an additional lower break-point (Bp2), below which the slope is affected by the Sl2 value.
- Shaper 3 BP: An expansion on the Shaper 2BP Module, with control of the slope of the signal between zero and Bp1, and control over the slope between zero and Bp2.

There are two mathematical Shapers:

- Parabol Shp (Shaper Parabolic): A Shaper with a parabolic curve. It allows you to mix the undistorted signal (Lin) and the parabolic signal (Par).

- Cubic Shp (Shaper Cubic): Adds cubic distortion to the Parabol Shp Module.

18.14.3 Modifiers



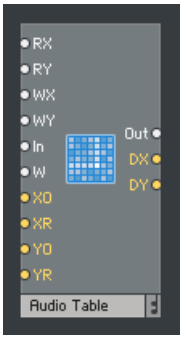
These Modules take the input signal and use it as the basis of a new output signal.

- Slew Limiter: Follows the input signal, but at a limited rate of change. The upward rate and downward rate can be set independently.
- Peak Detect. (Peak Detector): The input signal is rectified and then smoothed by an adjustable release time (Rel). The attack time is fixed at 0ms.
- Sample & H. (Sample + Hold): The input signal is sampled and held each time the Trig input rises above 0.
- Freq. Divider (Frequency Divider): The zero-crossing slopes of the input signal are counted and the output signal only changes when the counters reach the values set at the C+ and C- inputs.



The Frequency Divider Module is the same as the Frequency Divider found in the Event Processing section.

18.14.4 Audio Table



The Audio Table holds a two-dimensional table of values, which can be read and written at the audio rate.

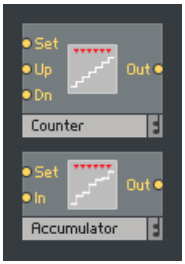
The Audio Table can be used to create a crude sampler or wavetable.

The Audio Table functionality is very similar to the Event Table, which is used in the tutorial in section ↑5, Basic Step Sequencer ↑5, Basic Step Sequencer.

The Audio Table is not part of the Table Framework.

18.15 Event Processing

18.15.1 Counters



There are two Event Counters in REAKTOR:

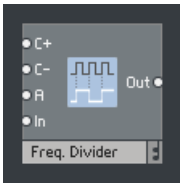
- Counter: Adds or subtracts 1 from an internal value each time an Event arrives at Up or Dn (Down) inputs. The internal value can be set to a specific value by using the Set input.
- Accumulator: Adds the value of the input Event to an internal value. The internal value can be set to a specific value by using the Set input.

18.15.2 Random



The Random Module modifies the input Event value by a random deviation. The deviation amount is set at the Rng input.

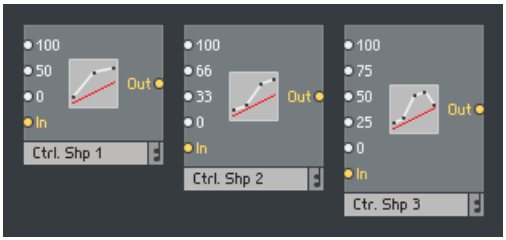
18.15.3 Frequency Divider



The zero-crossing slopes of the input signal are counted and the output signal only changes when the counters reach the values set at the C+ and C- inputs.

This Module is the same as the Frequency Divider found in the Audio Modifier section.

18.15.4 Control Shapers



The Control Shapers allow you to alter the slope or curve of Event signals. Unlike the Audio Shapers, these Modules remap specific values and thus alter the control curve.

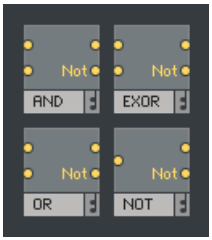
To define the slopes, you set the values of several points; Input values are changed to fit their respective location on the lines these points draw.

Note that the remap values as they are written at the Ports are presented as percentages, but the input requires a range of 0 to 1. So the 50 Port defines the value remap of a 0.5 input and the 100 Port defines the remap of a 1 input.

- Ctrl. Shp 1 (Ctrl. Shaper 1 BP): Allows you to alter the minimum point, a midpoint value, and the maximum point, creating two different slopes.
- Ctrl. Shp 2 (Ctrl. Shaper 2 BP): Allows you to alter the minimum point, the maximum point, and two midpoints (33 and 66), creating three different slopes.
- Ctrl. Shp 3 (Ctrl. Shaper 3 BP): Allows you to alter the minimum point, the maximum point, and three midpoints (25, 50, and 75), creating four different slopes.

Control Shapers can be useful for changing the scaling of a Knob or Fader.

18.15.5 Logic



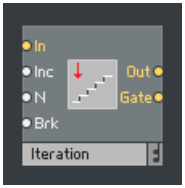
The Logic Modules are used to perform logic (Boolean) processes.

- When an input is greater than 0 it is considered to be **True**.
- When an input is less than or equal to 0 it is considered to be **False**.

The available Modules and processes are:

- AND
- OR
- EXOR
- NOT

18.15.6 Iteration



The Iteration Module is REAKTOR's equivalent of a loop.

- The input Event is set to the output along with N additional Events. The value of each Event increments upon the previous Event by the amount set at the Inc input.
- The iteration process can be set to a limited speed in the Module Properties.

- The iteration process can be stopped by sending a positive Event to the Brk input.

The Iteration Module is one of the main topics covered in the tutorial in section ↑7, Additive Synthesizer ↑7, Additive Synthesizer.

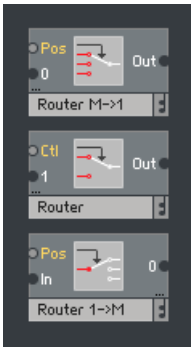
18.15.7 Signal Flow



These Modules control the flow of Event signals.

- Order: An Event at the input is set to the outputs in a defined order.
- Separator: Separates an Event to one of the two outputs depending on whether it is above or below the Threshold (Thld) value.
- Value: An Event at the Trig input will send an Event with the value of the signal at the lower input. Note that this Module can also be used to trigger the transmission of Table References.
- StpFlt (Step Filter): An incoming Event is only passed to the output if its larger or smaller than the previous value by an amount defined at the Tol input.
- M (Merge): This Module is used to merge multiple signals into one. Whichever input received an Event last will have its value sent to the output. If multiple Events arrived simultaneously, the lowest input will be used.

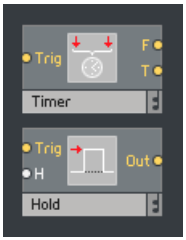
18.15.8 Routers



These Modules are used to route Events to/from different parts of the Structure.

- Router M->1: A "many to one" router. The Pos input is used to select which input signal is sent to the output.
- Router: This Module is a precursor to the Router M->1 Module. It can only accept 2 signals, and switches to the 1 signal when the Ctrl input is >=1
- Router 1->M: A "one to many" router. The Pos input is used to select to which output the input signal will be sent.

18.15.9 Timing

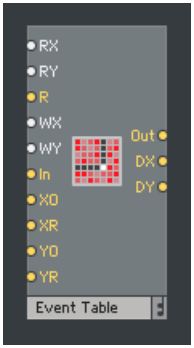


These Modules can be used to measure or manipulate the timing of Events.

- Timer: The T Port outputs the number of milliseconds between an Event at the Trig input and the previous Event at that input. The F Port outputs the frequency of Events (by calculating 1/T).
- Hold: When an Event arrives at the Trig input, it is held at the output for an amount of time defined at the H input. Once the hold time ends, the output returns to zero.

The timing of the Hold Module is course, as it uses the Control Rate clock for better CPU performance. For finer control over the Hold time, use the H Envelope from the LFO, Envelope section, as this uses the Sample Rate clock.

18.15.10 Event Table

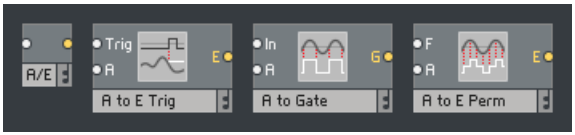


The Event Table holds a two-dimensional table of values.

The tutorial in section ↑5, Basic Step Sequencer ↑5, Basic Step Sequencer makes extensive use of the Event Table.

18.16 Auxiliary

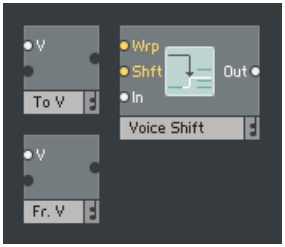
18.16.1 Audio to Event



These Modules are all designed to convert Audio signals into Event Signals.

- A/E (A to E): Simply resamples the incoming Audio signal at the Control Rate.
- A to E Trig: The A value is sampled when the Trig input has a rising edge.
- A to Gate: Converts the Audio signal into a Gate. A rising edge opens the gate, and a falling edge closes it. The on value of the gate is set at the A input.
- A to E Perm: Resamples the value A at a rate set at the F input.

18.16.2 Voice

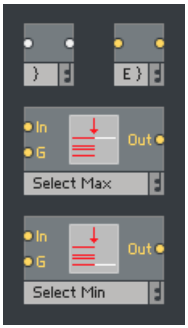


These Modules provide you with the ability to alter or interact with the polyphonic voices.

- To V (To Voice): Sends a monophonic signal to a specific voice in a polyphonic signal. The voice number is specified at the V input.

- Fr. V (From Voice): Outputs a monophonic signal by selecting the signal from a single polyphonic voice. The voice number is specified at the V input.
- Voice Shift: Can be used to shift a signal from one voice to another.

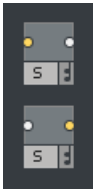
18.16.3 Voice Combiners



Voice Combiners are used to convert polyphonic signals into monophonic signals.

- } (Audio Voice Combiner): Sums the voices of a polyphonic audio signal into a monophonic audio signal.
- E } (Event V.C. All): Merges the voices of a polyphonic Event signal. The value of the last Event to arrive at the input will be the value at the output.
- Select Max (Event V.C. Max): Selects the voice with the current highest value to output as a monophonic signal.
- Select Min (Event V.C. Min): Selects the voice with the current lowest value to output as a monophonic signal.

18.16.4 Smoothers



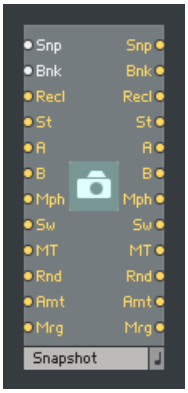
Smoothers slow the rate of change in values, effectively smoothing transitions.

The smoothing time is set in the Module Properties.

There are two Smoothing Modules:

- Audio Smoother: Takes an Event input and creates smoothed transitions at the Audio Sample Rate. This Module is better for higher quality smoothing.
- Event Smoother: Creates smoothed transitions at the Control Rate. This Module is better for performance.

18.16.5 Snapshot



The Snapshot Module has inputs and outputs that relate to almost all of the controls in the Snapshot tab of the Sidepane.

Not only does this Module output the current Bank and Snapshot numbers, but it also sends Events on Snapshot recall (Recl) and storage (St), which can be useful when initializing the instrument after a Snapshot loads.

18.16.6 Snap Values



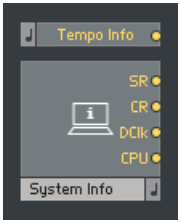
These Modules hold values that are stored and recalled with Snapshots.

- Snap Value: Stores and recalls a single value with the Snapshot.
- Snap Value [] (Snap Value Array): Stores and Recalls an array of values with the Snapshot.



The Snap Value Array is used in the tutorial in section ↑6, Advanced Step Sequencer ↑6, Advanced Step Sequencer.

18.16.7 System



The System Modules output information about REAKTOR.

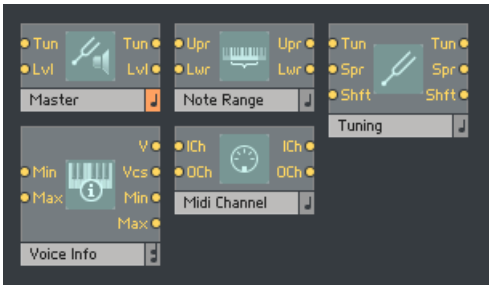
- Tempo Info: A source for the current tempo in Beats per Second.
- System Info: Contains several outputs relating to the REAKTOR engine:
  - SR (Sample Rate): The audio engine Sample Rate.

- CR (Control Rate): The Control Rate for Event signals.
- DClk (Display Clock): Sends an Event before the interface refreshes, which should happen around 25 times per second.
- CPU: A measurement of the current CPU load. Whether this measurement is smoothed or raw output can be set in the properties.
- GUI: A measurement of the GUI thread load.



The Display Clock is a useful tool when optimizing custom interfaces. See section ↑14.4, Use the Display Clock ↑14.4, Use the Display Clock for more information.

18.16.8 Instrument Properties



These Modules can be used to retrieve or define certain settings in the Instrument (or Ensemble) Properties.

The available Modules are:

- Master: Accesses the main output volume and tuning.
- Note Range: Accesses the MIDI Note range.
- Midi Channel: Accesses the MIDI input and output channels.
- Tuning: Accesses the tuning properties.
- Voice Info: Accesses the polyphony settings, including an output (V) that sends the Voice ID on that Voice.

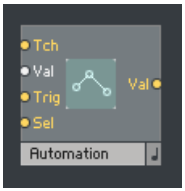
18.16.9 Spread



The Spread Module is only of use when the Ensemble it is inside is polyphonic.

This Module generates a different offset value for each voice, which can be used to create unison spread in other Modules.

18.16.10 Automation



This Module is used to send and receive automation to/from a host program.



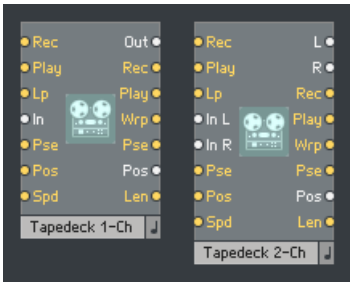
Using this Module is described in more detail in section ↑11.2, The Automation Module ↑11.2, The Automation Module.

18.16.11 Set Random



Sets the seed value for the random number generation within the Ensemble.

18.16.12 Tapedeck



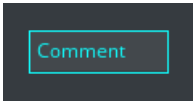
The Tapedeck Modules essentially mimic the behavior of the Player and Recorder, but as Modules in the structure.

You can use them to record audio, save it, and play it back.

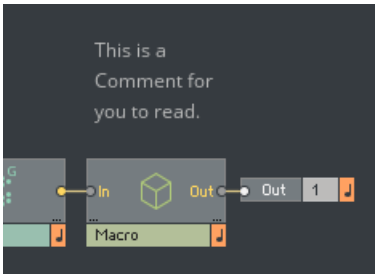
The Tapedeck is available in two versions:

- Tapedeck 1-Ch: A monophonic audio recorder.
- Tapedeck 2-Ch: A stereo audio recorder.

18.16.13 Comment



The Structure *Comment* Module is a Module that allows you to annotate your Ensemble's Structure. It appears as floating text within the Structure View of REAKTOR.



An Unselected Comment as It Appears in the Structure

See Improving Readability for more information on how to use the *Comment* Module.

18.17 Terminal

18.17.1 Ports



Ports are an essential part of building in REAKTOR. They allow the contents of one level of the Structure to send values to the level above.

The final Ports are the Ensemble Ports, which send audio to the host software or computer soundcard.

18.17.2 Send/Receive



The Send and Receive Modules can be used to make wireless connection in the Ensemble.

The Receive Module listed here is the same as the Receive Module from the Panel section.

Caution is advised when using these Modules, as they can impact performance and loading times. See section ↑14.7, Avoid unnecessary use of Send/Receive Modules ↑14.7, Avoid unnecessary use of Send/Receive Modules for more details.

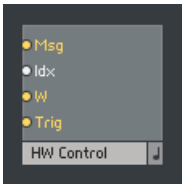
18.17.3 IC Connections



The IC Send and IC Receive Modules use the Internal Connection (IC) Protocol to wirelessly send signals.

The IC Protocol and its use is documented in section ↑13, Internal Connection Protocol ↑13, Internal Connection Protocol.

18.17.4 Hardware Control



The Hardware Control Module (HW Control) is used to communicate with NATIVE INSTRUMENTS hardware.

The use of the Hardware Control Module is documented in section ↑12.4, Hardware Control ↑12.4, Hardware Control.

Index

Numerics

2-Osc [53]

A

always active [97]

animation [152]

array [103]

Atmotion [87]

audio engine [59]

audio signals [22]

- converting to event signals [278]
- routing [243]

automation [171] [192] [283]

- ID [172]
- name [171] [174]

B

bit-crusher [225]

BPM (Beats Per Minute) [67]

C

clock [75] [77] [116] [230] [250]

color scheme [149]

connections [21] [58]

- IC (Internal Connection) [197]
- wireless [205] [285]

constant [222]

Context Menu [19]

- Built-In Module [20] [60]
- Library [20] [37]
- New Macro [20]
- Open Searchbox [20]

control rate [204] [282]

control shaper [273]

core [207]

- building [139]
- CCCCC (Compiled Core Cell Code Cache) [219]
- core cells [128] [137]

counter [272]

CPU (Central Processing Unit)

- measurement [30] [282]

D

Debug Structure [28]
Enable Wire Debugging [29]
Measure CPU Usage [30]
Show Event Init Order [32]
Show Module Sorting [31]
debugging [28]
decibel
converting [226]
delay [55] [266]
buffer [69]
differentiation [265]
distortion [268]

E

Echomania [73]
edit mode [35]
Envelope [259]
envelope follower [270]
EQ (Equalizer) [264]
event signals [22] [75] [202]
filtering [86] [202]
initialization [32]
ordering [81] [140] [275]
processing [271]
routing [117] [124] [275] [276]
timing [276]

F

factory library
ensembles [53] [73] [87]
macros [37] [43] [129] [191]
feedback [65] [266]
file player [65]
filter [43] [261]
FPS (Frames Per Second) [203]
frequency divider [270] [273]

G

g200kg KnobMan [152]

granular [253] [268]

GUI (Graphical User Interface) [89] [143]

display clock [203] [282]

H

hertz

converting [226]

host

automation [171]

song position [83] [230]

tempo synchronization [67] [83]

transport [230]

I

IC (Internal Connection) protocol [197]

modules [199] [286]

image

Image Properties [154]

importing [153]

skinning [152]

info hints [221]

initialization [32] [140]

Instrument Properties [282]

integration [265]

iteration [108] [274]

J

Junatik [54]

K

Komplete Kontrol [180]

browser [184] [193]

Light Guide [187] [286]

Native Map [192]

Komplete Kontrol S-Series [187]

L

Lazerbass [126]

LFO (Low Frequency Oscillator) [258]

logarithms [226]

logic [225] [274]

Longflow [74]

loop

- for loop [108]

loopSee iteration [119]

M

macro [19] [55]

- browsing and loading [20] [37] [41] [43] [71]
- creating [57]
- properties [91]
- re-naming [59]
- saving [69]
- stacked macro [161] [243]

macros

- polyphony [27]

Maschine [180]

mathematics [222]

MIDI (Musical Instrument Digital Interface

- MIDI Out [231]

MIDI (Musical Instrument Digital Interface)

- MIDI In [41] [114] [227]
- notes [226]

Mikro Prism [126]

mixer [245]

module [19] [47]

- browsing and loading [20] [47] [60]
- deleting [110]
- info [131]
- properties [62]
- reference [221]

re-naming [64]
sorting [31]
mono [26]

N

noise
generator [250]

O

Oki-Computer 2 [126]
optimization [86] [201] [239]
OSC (Open Sound Control) [234]
oscillator [37] [245]
oscilloscope [243]

P

panel
A/B view [147]
color scheme [149]
customization [89] [161]
customizing [134]
editing [40] [64] [91]
header [145]
layering [158]
modules [86] [89] [106] [143] [235]
panel grid [156]
skinning [152]
text [237]
panelsets [186]
playback [85] [230]
polyphony [24] [49] [69] [202]
ports [22] [42] [58] [69] [130] [285]
Create Control [62]
Preferences [219]
Directories [219]
preset [183]
Pro-52 [263]

Q

quantize [96] [224]

R

random [272] [283]

Reaktor Prism [126]

S

S&H (Sample and Hold) [270]

sample rate [138] [281]

sampler [128] [252]

saturation [268]

searchbox [20] [41] [47]

sequencer [75] [256]

Settings

- Snap to Grid [157]

skinning [152]

smoothing [280]

snapshot [100] [280]

- tagging [180]
- value [100] [121] [281]

SQX [106]

synthesis

- additive [109] [251]
- FM (Frequency Modulation) [52] [246]
- modal [126] [264]
- subtractive [34]

T

table [78]

- audio [271]
- event [277]
- Table Draw Mode [81]

table framework [128] [209] [255]

table references [128] [130] [209]

tapedeck [284]

tempo [67] [83]

transport

- play [85]

trigonometry [227]

truncation [85]

U

unison [283]

V

Vierring [87]

voices [25] [50] [201]

- Automatic Voice Reduction [201]
- lock voices [26]
- modules [278]
- voice combiners [27] [50] [279]

W

waveshaper [269]


REAKTOR 6
BUILDING IN CORE

The information in this document is subject to change without notice and does not represent a commitment on the part of Native Instruments GmbH. The software described by this document is subject to a License Agreement and may not be copied to other media. No part of this publication may be copied, reproduced or otherwise transmitted or recorded, for any purpose, without prior written permission by Native Instruments GmbH, hereinafter referred to as Native Instruments.

“Native Instruments”, “NI” and associated logos are (registered) trademarks of Native Instruments GmbH.

Mac, Mac OS, GarageBand, Logic, iTunes and iPod are registered trademarks of Apple Inc., registered in the U.S. and other countries.
Windows, Windows Vista and DirectSound are registered trademarks of Microsoft Corporation in the United States and/or other countries.
All other trade marks are the property of their respective owners and use of them does not imply any affiliation with or endorsement by them.

Document authored by: Vadim Zavalishin

Software version: 6.2.2 (06/2018)

NATIVE INSTRUMENTS GmbH
Schlesische Str. 29-30
D-10997 Berlin
Germany
www.native-instruments.de

NATIVE INSTRUMENTS North America, Inc.
6725 Sunset Boulevard
5th Floor
Los Angeles, CA 90028
USA
www.native-instruments.com

NATIVE INSTRUMENTS K.K.
YO Building 3F
Jingumae 6-7-15, Shibuya-ku,
Tokyo 150-0001
Japan
www.native-instruments.co.jp

NATIVE INSTRUMENTS UK Limited
18 Phipp Street
London EC2A 4NU
UK
www.native-instruments.co.uk

NATIVE INSTRUMENTS FRANCE SARL
113 Rue Saint-Maur
75011 Paris
France

www.native-instruments.com

SHENZHEN NATIVE INSTRUMENTS COMPANY Limited
203B & 201B, Nanshan E-Commerce Base Of Innovative Services

Shi Yun Road, Shekou, Nanshan, Shenzhen
China

www.native-instruments.com

NATIVE INSTRUMENTS

© NATIVE INSTRUMENTS GmbH, 2018. All rights reserved.

Table of Contents

1 Welcome to Building in Core 9
1.1 The REAKTOR 6 Documentation 9
1.2 Manual Conventions 11
1.3 Core Updates in REAKTOR 6.0.1 12
1.4 Core Updates in REAKTOR 6.2.2 13
2 REAKTOR Core Fundamentals 15
2.1 An Introduction to Core Cells 15
2.2 Core Structures 17
2.3 Built-in Modules 19
2.4 Core Connection Types 20
2.4.1 Scalar 21
2.4.2 OBC 22
2.4.3 BoolCtl 24
2.4.4 Bundle 24
2.4.5 Connection Defaults 25
2.5 Core Cells in Detail 26
2.5.1 Inputs 26
2.5.2 Outputs 27
2.5.3 Properties 29
2.6 Core Macros 31
2.6.1 Inputs 33
2.6.2 Outputs 35
2.6.3 Properties 35
2.7 Saving Core Cells and Core Macros 37

3 Additional Connectivity Features 40
3.1 QuickConsts 40
3.2 QuickBuses 41
3.3 Scoped Buses 45
3.3.1 Definitions 45
3.3.2 Simple Scoped Access 47
3.3.3 Scoped Connection Errors 50
3.3.4 Name Collisions and Definition Overriding 51
3.3.5 Parent Mode Accessors 52
3.3.6 Solid Areas 53
3.3.7 Name Normalization and Escaping 55
3.4 Bundles 56
3.4.1 Bundle Pack 56
3.4.2 Bundle Unpack 59
3.4.3 Definition Conflicts and Pickup Errors 60
3.4.4 Scoped Bus Subfiber Pickups 62
3.4.5 Bundle Merging 62
3.4.6 Bundle Splitting 65
4 Processing Model of Core 67
4.1 Events 67
4.2 Processing Order 70
4.3 Object Bus Connections (OBC) 72
4.4 Initialization 74
4.5 Clocks and Latches 76
4.6 Modulation Macros 79
4.7 Arrays 81

4.7.1 Array Indexing ... 82
4.7.2 Read [] and Write [] Macros ... 84
4.7.3 Table Specifics ... 85
4.7.4 Advanced OBC Ordering ... 87
4.8 Routing and Merging ... 88
4.9 SR and CR Buses ... 91
4.9.1 Connections to the Clock Buses ... 92
4.9.2 Clock Gating ... 92
4.9.3 SR and CR redefinition ... 94
4.9.4 CR rate change ... 94
4.9.5 SR Rate Change ... 99
4.9.6 Internal Structure of Clock Buses ... 100
4.10 Feedback Connections ... 102
4.10.1 Automatic Resolution ... 102
4.10.2 Manual Resolution ... 103
4.10.3 Nonsolid Macros and Feedback Loops ... 104
4.10.4 Resolution Mechanism ... 105
5 Building Practices and Conventions ... 107
5.1 Expression Computation ... 107
5.2 Generation of Audio and Other Regularly Clocked Events ... 108
5.3 Processing of Audio and Other Regularly Clocked Events ... 110
5.4 Rounding ... 112
5.5 Denormals and Other Bad Numbers ... 113
5.6 Optimization hints ... 115
5.6.1 Core Cell Handlers ... 115
5.6.2 Latches and Modulation Macros ... 116

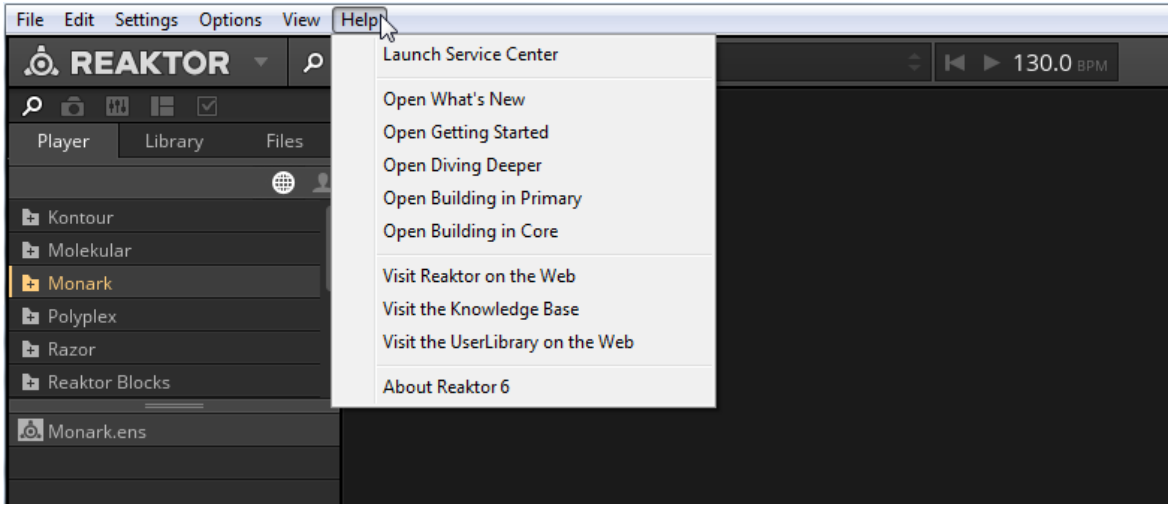
5.6.3 Routing and Merging 118
5.6.4 Numerical Operations 124
6 Macro Reference 127
6.1 Low-level Macros 127
6.1.1 Math 127
6.1.2 Math Mod 127
6.1.3 Clipping 128
6.1.4 Saturator 128
6.1.5 Convert 128
6.1.6 Memory 128
6.1.7 Flow 128
6.1.8 Event 128
6.1.9 Logic 128
6.2 Audio 129
6.2.1 Oscillator 129
6.2.2 Filter 133
6.2.3 FX 140
6.2.4 Transfer Functions 143
6.3 Control 145
6.3.1 Envelope 145
6.3.2 LFO 149
6.3.3 Smoother 153
6.4 Toolkits 153
6.4.1 LFO Toolkit 153
6.4.2 Envelope Toolkit 154

6.4.3 ZDF Toolkit 157
6.4.4 TF toolkit 162

1 Welcome to Building in Core

Welcome to Building in Core, a manual describing the Core level in REAKTOR with its low-level building features. Using a run-time optimizing compiler, this graphical audio programming language can be used for implementing your own custom DSP algorithms. This document also includes reference information about the Core Macro Library, a comprehensive collection of DSP building blocks.

1.1 The REAKTOR 6 Documentation



The REAKTOR documentation is accessible from the Help menu

The documentation for REAKTOR 6 is divided into five separate documents, guiding you from loading and playing pre-built Ensembles to building your own Instruments.

- **REAKTOR 6 What Is New** is written for users who are already familiar with previous versions of REAKTOR and only describes the latest features in brief.

- **REAKTOR 6 Getting Started** is for new users. It is the only document needed for users who intend to use REAKTOR for loading and playing pre-built REAKTOR instruments and effects.
- **REAKTOR 6 Diving Deeper** expands on the concepts introduced in the Getting Started document. It provides more detail on subjects like Snapshots (REAKTOR's preset system), and introduces advanced topics like OSC control and combining multiple Instruments in one Ensemble.
- **REAKTOR 6 Building in Primary** shows you how to build your own Instruments in REAKTOR's Primary level. It focuses on a series of tutorials that guide you through building your first synthesizers, effects, and sequencers.
- **REAKTOR 6 Building in Core** describes the Core level of REAKTOR with its low-level building features, which can be used for implementing custom DSP algorithms. It includes reference information about the Core Macro Library, an comprehensive collection of DSP building blocks.

With the exception of the What Is New document, each of the documents listed above builds on the knowledge of the previous documents. While it is not necessary to read all of every document, some of the more advanced documents, like Building in Primary, assume knowledge from the previous documents.

1.2 Manual Conventions

This section introduces you to the signage and text highlighting used in this manual.

- Text appearing in (drop-down) menus (such as Open..., Save as... etc.) and paths to locations on your hard disk or other storage devices is printed in italics.
- Text appearing elsewhere (labels of buttons, controls, text next to checkboxes etc.) is printed in blue. Whenever you see this formatting applied, you will find the same text appearing somewhere on the screen.
- Important names and concepts are introduced in **bold**. Furthermore, **bold** text is used to stress important statements in the text.
- References to keys on your computer's keyboard you'll find put in square brackets (e.g., "Press [Shift] + [Enter]").

1. Single instructions are introduced by this play button type arrow.

⇨ Results of actions are introduced by this smaller arrow.

An indented, gray paragraph contains additional, contextual information.

Furthermore, this manual uses particular formatting to point out special facts and to warn you of potential issues. The icons introducing these notes let you see what kind of information is to be expected:



The speech bubble icon indicates a useful tip that may often help you to solve a task more efficiently.



The exclamation mark icon highlights important information that is essential for the given context.



The red cross icon warns you of serious issues and potential risks that require your full attention.

1.3 Core Updates in REAKTOR 6.0.1

Here is a brief list of what has been changed in Core compared to REAKTOR 5.

- Two new major features, *Scoped Buses* (see section ↑3.3, Scoped Buses) and *Bundles* (see section ↑3.4, Bundles) have been introduced. Particularly the former SR.C and SR.R connections are now a part of the Scoped Buses/Bundles framework (see section ↑4.9, SR and CR Buses).
- Unification of audio and event Core Cells. Now there is only one Core Cell type. The input and output ports of the Core Cells can be switched between audio and event types (see sections ↑2.5.1, Inputs and ↑2.5.2, Outputs).
- SR.C does not send an initialization event anymore (see section ↑4.9.6, Internal Structure of Clock Buses).

  Upon importing the older Core Cells special adapter Macros are automatically inserted at the top level of the Structures to simulate the REAKTOR 5 behavior of SR.C.

- Nonsolid Macros and Structures now have a dedicated look (see section ↑2.6.3, Properties).
- The feedback loop highlighting now highlights the entire loop instead of just the resolution point (see section ↑4.10.1, Automatic Resolution).
- The Macro library has been completely restructured (see section ↑6, Macro Reference).
  - No more Expert / Standard parts.
  - New oscillators, filters, FX, LFOs, envelopes and smoothers.
  - Dedicated set of Macros for audio and control rate handling (see section ↑4.9, SR and CR Buses).
  - The former Expert > Modulation part is now found under Math Mod.
  - The logic signals have been slightly reworked (see section ↑6.1.9, Logic).
  - The Flow submenu contains a number of value-controlled routers.

- The *z^-1* Macros have been reworked. Now they are intended to be used exclusively for audio (see section ↑4.10.2, Manual Resolution). For other one-clock delay purposes use Memory > Latch[-1] or Event Proc > Event[-1] Macros.
- The *~exp* and *~log* Modules used inside the Macros (such as e.g. Convert > P2F) are typically set to the highest precision. Set them to a lower precision if the highest precision is not needed.

1.4 Core Updates in REAKTOR 6.2.2

In REAKTOR 6.2.2 the Macro Library has received significant updates.

New Additions

- (Non-transposed) nonlinear Sallen–Key filter
- 8-pole ladder filter
- 6-pole and 8-pole band-pass Butterworth filters
- 2nd kind (resonating) Butterworth filters (as described in The Art of VA Filter Design by Vadim Zavalishin)
- Completely reworked set of shelving filters
- Completely reworked set of crossover Macros
- Configurable notch count for phasers
- Thru-zero phaser and flanger, barber pole phaser and flanger, harmonic phaser
- Asymmetric overdrive, anti-aliased versions of symmetric and asymmetric overdrives
- Bit and sample rate reduction effects
- Compressor
- Tempo and transport position synced LFO
- LFO toolkit
- Zero-crossing detector event processing Macro

Updates and Improvements

- Cleaned up the internal structure of the transposed Sallen–Key filters. The *TSK NLA* filter sound is slightly changed.
- Improved CPU consumption of the *Delay* and *Delay (ZDF Toolkit)* Macros.

Fixes

- Wrong phase deviation value at the output of the *Phase Splitter* Macro inside the *Freq Shift* Macro (this issue did not affect the sound, but provided wrong information).
- Some Math Macros were sending initialization when they should not.
- *xy2polar* and *toPolar* Macros did not handle zero input.

2 REAKTOR Core Fundamentals

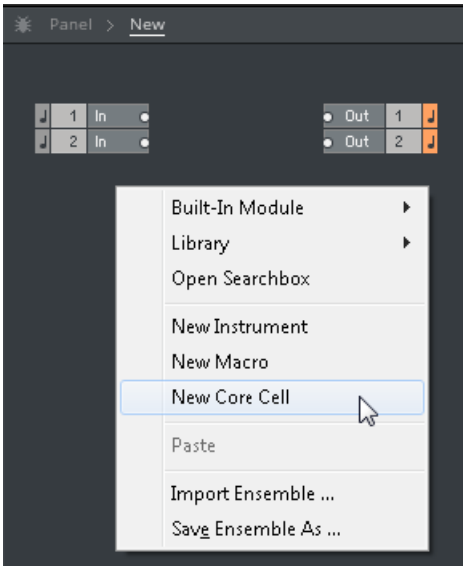
2.1 An Introduction to Core Cells

REAKTOR Core is a special level of functionality within REAKTOR with a distinct set of features. Similarly to the main part of REAKTOR (which we will refer to as the **Primary level** of REAKTOR), the **Core level** is also using a graphical signal flow paradigm. However, there are a number of important differences in the details between the Primary and the Core levels:

- REAKTOR Core is using an integrated run-time compiler, allowing efficient processing of highly detailed low-level Structures.
- The low-level Structure building features of REAKTOR Core make it specifically suitable for implementing custom DSP algorithms.
- The details of the rules of signal processing in REAKTOR Core are different from Primary level and are tailored to the needs of the DSP algorithm design.
- REAKTOR Core is using a different set of basic Modules and a different Macro library.

The REAKTOR Core Structures exist inside special wrapper objects called **Core Cells**. Core Cells can be considered as a special kind of Primary level Macros, where the difference is that inside Core Cells there is a different world of REAKTOR Core.

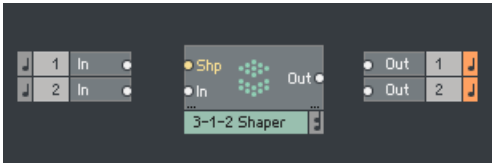
In order to create a new blank Core Cell right-click somewhere in the background of the REAKTOR Structure and select New Core Cell from the context menu:



Creating a new Core Cell from the context menu.

The prebuilt Core Cells from the factory and user libraries can be found in the Library menu among Primary Macros and Instruments.

Core Cells look like a special type of the Primary level Modules:

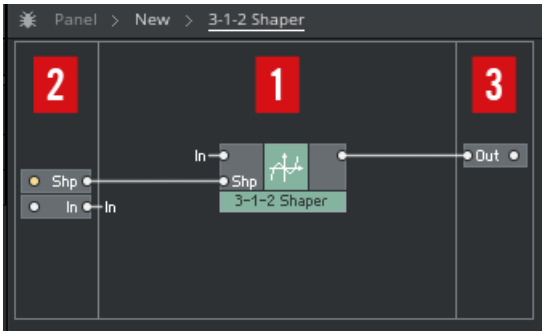


A Core Cell.

Similarly to Primary level Macros, Core Cells have internal Structures, which can be navigated to by double-clicking the Core Cell.

2.2 Core Structures

Core level Structures (or simply 'Core Structures') are the Structures contained inside Core Cells and Core Macros. These Structures look like follows:



The internal Structure of a Core Cell.

The most obvious difference from the Primary Structure is the frame with three different areas (1), (2) and (3).

(**1**) **Normal Modules area**: The bigger area in the middle (referred to as the **normal Modules area** or simply **normal area**) is where the majority of the Structure's Modules (more specifically, the so-called **normal Modules**) are typically located. The Core level Macros are also inserted here among the other normal Modules.

(**2**) **Input area**: The smaller area on the left is where the input port Modules are located.

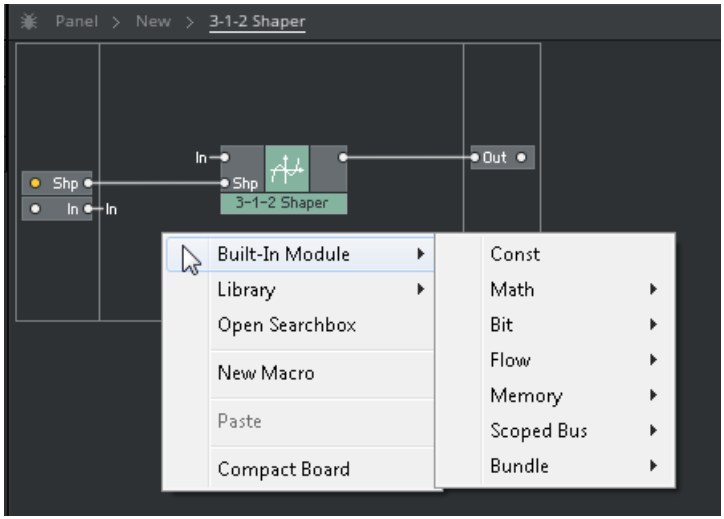
(**3**) **Output area**: The smaller area on the right is where the output port Modules are located.

The three areas can contain only the Modules of the respective **flavor**, that is the normal area can contain only the normal Modules and the input and output areas can contain only the input and output port Modules respectively.

The normal Modules can be moved freely in both horizontal and vertical directions. The input and output port Modules can be moved only vertically. Their vertical positions specify the order in which they appear on the outside of the Core Cell or a Core level Macro. Changing their relative vertical positions changes their outside order.

In order to create a new Module in a Core Structure, right-click on the background inside of one of these three areas. **Depending on which of the three areas is clicked, a context menu with a different list of available Modules appears.**

1. For example, right-click on the normal area to access the normal Modules:



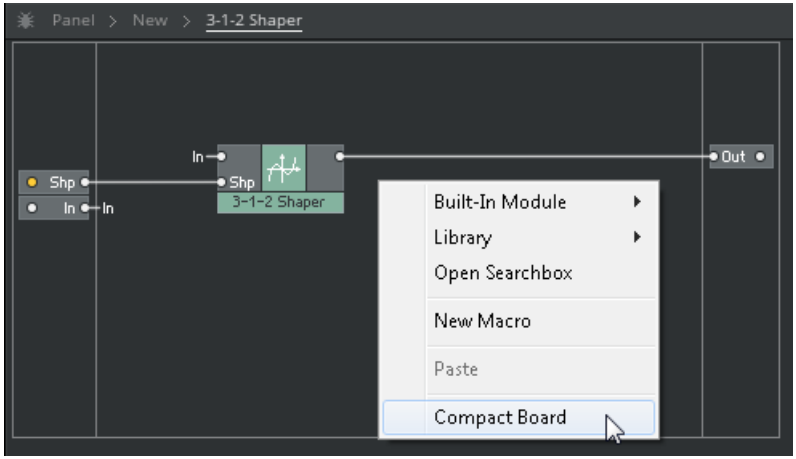
Pretty much any editing done to a Core Cell Structure will cause it to recompile. For small and simple Core Cells the compilation typically happens very quickly, but for large or complicated Core Cells it might take noticeable amounts of time. During the compilation, the progress is displayed in REAKTOR's toolbar by an orange gradient moving over the CPU load display:



Structure Shrinking

As more Modules are inserted into the Structure and/or moved around within the Structure, the enclosing frame will grow to accommodate the Modules. The frame never shrinks back though.

1. In order to make the frame shrink back, right-click on the background somewhere within the Structure and select Compact Board from the context menu.



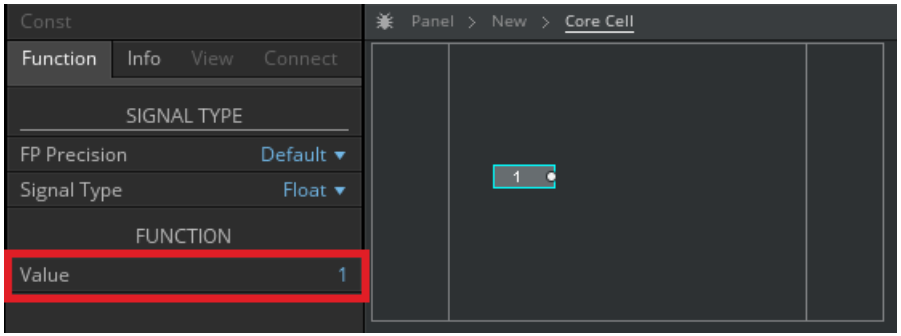
The shrinking will not change the distances between the Modules (except possibly the distances from the inputs and the outputs to the normal Modules). There is also some hard-coded minimum size which the shrinking will not go beyond.

2.3 Built-in Modules

The ideas and conventions behind the groups of Modules available in the Built-In Module submenu of the normal area context menu are discussed in different places of the manual according to their function. The information about specific Modules can be found directly within the software in the Module's info.

- Const: The Core constants are very similar to the Primary level constants. See section ↑4.4, Initialization for additional details.

The value of the Core constants cannot be changed in the name field of the Properties, one has to use the dedicated value property field:



- Math: See sections ↑4.1, Events and ↑4.2, Processing Order.
- Bit: This group is simply an extension of the Math group dedicated to the integer bit arithmetic.
- Flow: See section ↑4.8, Routing and Merging.
- Memory: See sections ↑4.3, Object Bus Connections (OBC) and ↑4.7, Arrays.
- Scoped Bus: See section ↑3.3, Scoped Buses.
- Bundle: See section ↑3.4, Bundles.
- Macro: See section ↑2.6, Core Macros.

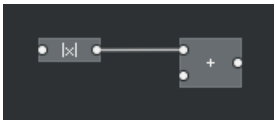
2.4 Core Connection Types

REAKTOR Core Structures are using several different types of connections. Respectively there are different types of the input and output ports of the Modules used inside Core. These types fall into several different classes.

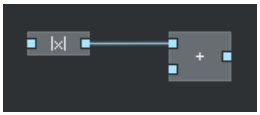
The connections between ports of different classes are not possible. Whether the connections between different port types within the same class are possible varies from one class to another.

2.4.1 Scalar

Scalar is the most commonly used class. Scalar connections can be either of **float** (floating-point) or **int** (integer) type, of which float is more commonly used.

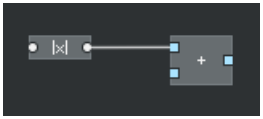


A connection between float ports.



A connection between int ports.

All scalar types are connection-compatible to each other (meaning it is possible to connect a float output to an int input or vice versa). The values of one type will be automatically converted into another wherever necessary:



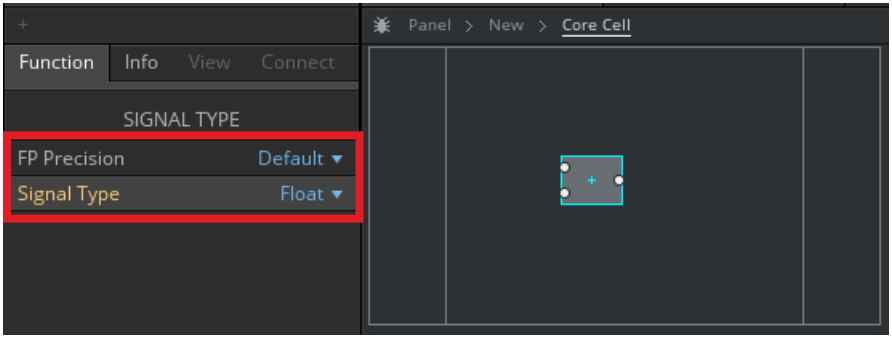
A float output connected to an int input.

Upon conversion from float to an int the value is rounded to the nearest integer.

For the values which are exactly in between, the rounding direction is conceptually undefined. For example, 1.5 can be rounded to 1 or to 2. It is not advisable to rely on a specific direction of that rounding for any of such 'exactly-in-between' values, even if it seems fully consistent and reliable in the current REAKTOR version. The rounding could change e.g. on a different hardware architecture or for whatever other reason. As a general rule one should not rely on the implementation details which are undefined by the specification, especially if they are explicitly mentioned as undefined.

Type Configuration

For many of Core's built-in Modules, particularly for those which perform mathematical operations, the scalar type can be reconfigured in the Properties:



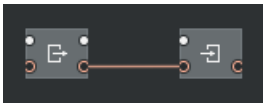
Scalar configuration properties of an adder.

The Signal Type property can be switched between Float and Int. The FP Precision setting (applicable only to the float mode) can be Default, 32 bit or 64 bit. Normally it should be left at Default, which means that the Module will use the default floating point precision of the given Structure, where the latter (unless reconfigured by the builder) is 32 bit. The 64 bit precision is needed in exceptional cases. The precision is not reflected visually in the look of Modules, ports or connections. The precision does not affect the compatibility of connections either, automatic precision conversion will occur wherever necessary.

> Conceptually, a 32 bit float precision setting does not guarantee that it will have exactly 32 bit precision. It means that the precision will be at least 32 bits. The same holds for 64 bit.
>
> The 32 bit float is the type used by Primary audio and event connections.

2.4.2 OBC

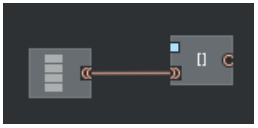
The OBC class is used for identification of the memory storage items (see the discussion in section ↑4.3, Object Bus Connections (OBC)), where the items can be of **latch** (for single variables) or **array** type. Each of these types can be float or integer.



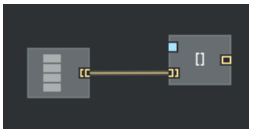
A connection between float OBC ports of type latch.



A connection between int OBC ports of type latch.



A connection between float OBC ports of type array.

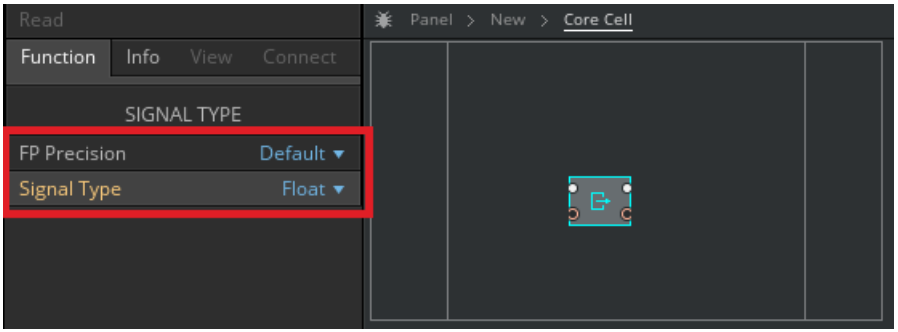


A connection between int OBC ports of type array.

Differently from the scalar connections, the OBC connections of different types are not compatible to each other.

Type Configuration

Similarly to the scalar processing Modules, most of the OBC Modules feature the possibility of type configuration in the Properties:



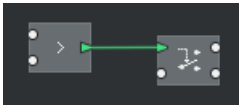
OBC configuration properties of a Read Module.

Besides affecting the OBC port types, these properties also affect the type of the memory storage associated with the OBC Module.

Differently from scalar processing Modules, the float OBC connections of different precision settings are not compatible to each other. Particularly, the Default precision setting is not compatible to the others (e.g. it is not compatible to 32 bit setting even if the actual precision corresponding to the Default setting is 32 bit). Normally the Default precision setting should be used. See sections ↑2.5.3, Properties and ↑2.6.3, Properties for further details of precision control.

2.4.3 BoolCtl

The BoolCtl class contains a single type, which is also called BoolCtl (Boolean control). These connections are used for router control signals (see section ↑4.8, Routing and Merging).

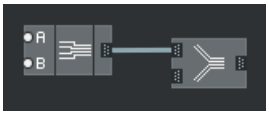


A connection between BoolCtl ports.

The BoolCtl type does not have any further configuration parameters.

2.4.4 Bundle

Bundles are 'multiwire cable' connections. They are simply containers for the nested connections.



A connection between Bundle ports.

The Bundle type itself does not have any configuration parameters. However the contents of the cable vary depending on its source. In principle any Bundle output can be connected to any Bundle inputs, but the connection may still be considered erroneous if the contents of the incoming Bundle do not match the expectations of the connection destination's owner. See section ↑3.4, Bundles for further details about using the Bundle connections.

2.4.5 Connection Defaults

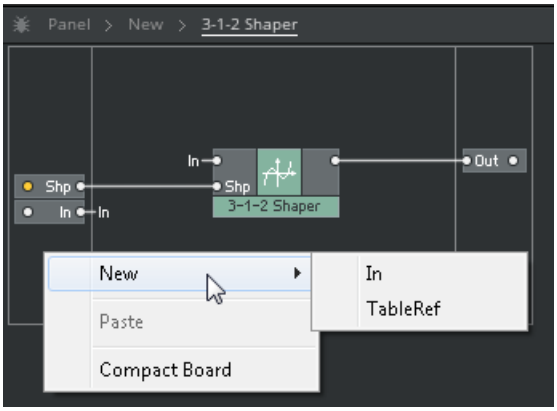
Inputs which are left unconnected still normally have a defined functionality. This functionality can be specifically defined by each Module for each of its inputs separately. Particularly, a Macro builder can specify the meaning of the disconnected Macro inputs for each of its inputs (see section ↑2.6.1, Inputs). In practice, however, there are most commonly used defaults. These defaults are used for the Macro inputs, unless specifically overridden by the Macro builder, and are also used for the built-in Modules, unless the Module's documentation says otherwise.

- **Scalar** inputs default to a zero constant. This means that the input behaves exactly as if there was a zero constant connected to it, including the initialization behavior (see section ↑4.4, Initialization).
- **Non-array OBC** inputs default to a unique storage location. This rule is abundantly used in storage manipulation (see section ↑4.3, Object Bus Connections (OBC)).
- **Array OBC** inputs, unless explicitly specified otherwise, should not be left disconnected. A disconnected array OBC input has undefined semantics, except that it is guaranteed not to affect the correctly connected OBC memory.
- **BoolCtl** inputs default to a false signal.
- **Bundle** inputs default to an empty Bundle.

2.5 Core Cells in Detail

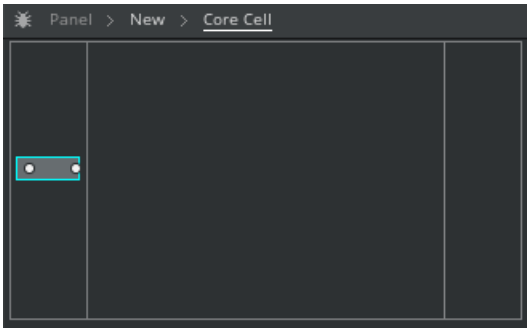
2.5.1 Inputs

There are only two different input port Module types available in the Core Cell's input port area context menu: *In* and *TableRef*.



The Core Cell's input port area context menu.

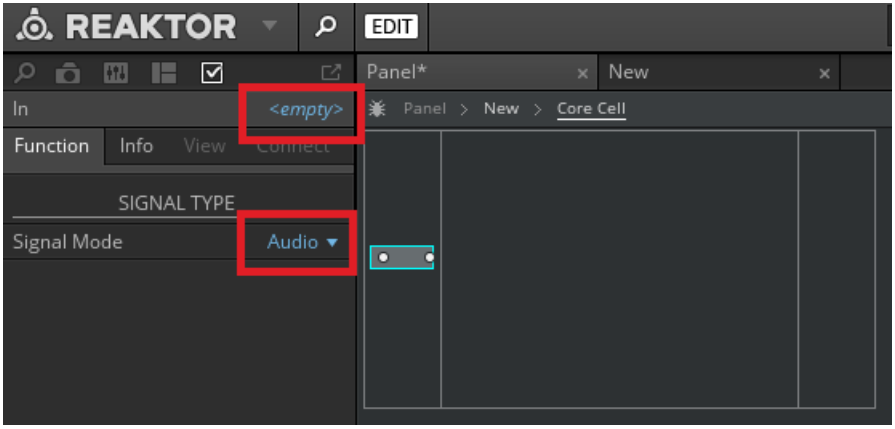
The *TableRef* input port type is described separately in the Building in Primary document. The Building in Core document deals exclusively with the Core Cell input ports of type *In*:



A Core Cell input port Module.

Input Properties

The Properties of a Core Cell input port allow you to specify the port's name and select between the **Audio** and **Event-mode** of the port:



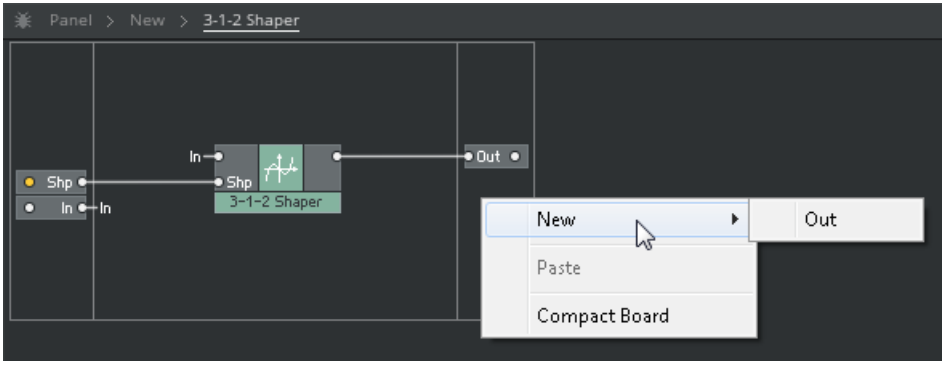
Properties of a Core Cell input port Module.

The Audio/Event-mode specifies whether the input will appear as a Primary level audio or event port on the outside of the Core Cell.

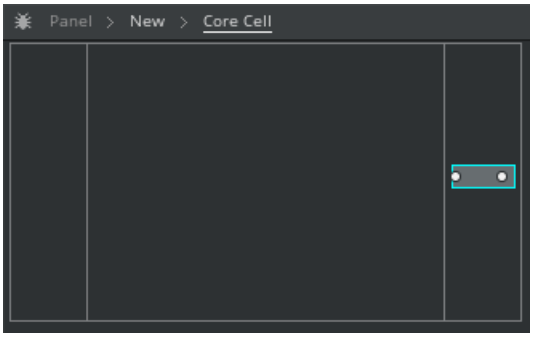
In the Info Properties tab one can specify the hint text for the port.

2.5.2 Outputs

There is only one output port Module type available in the Core Cell's output port area context menu: *Out*.



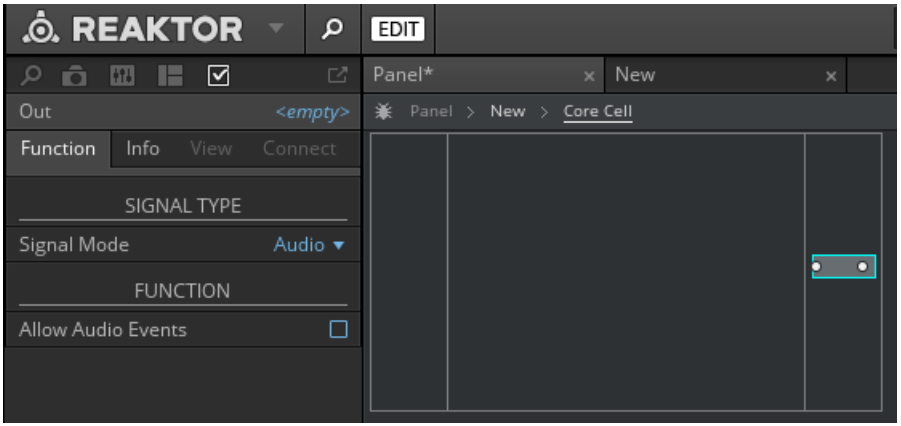
The Core Cell's output port area context menu.



A Core Cell output port Module.

Output Properties

The Properties of a Core Cell's output port are similar to those of the Core Cell input port:



Properties of the Core Cell output port Module.

The Audio/Event-mode specifies whether the output will appear as a Primary level audio or event port on the outside of the Core Cell.

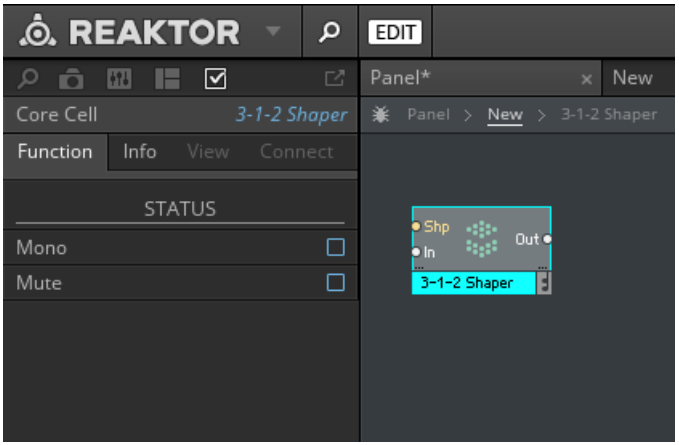
> The additional Allow Audio Events checkbox is an advanced option for the event-mode outputs. By default, the event-mode outputs block internal Core events that are originating from an audio source (those are the events obtained directly or indirectly from audio-mode inputs of the Core Cell and from the default internal clock sources of the Core Cell). This is important in order to avoid events being sent inadvertently from the Core Cell's outputs to the Primary level at audio rate, which is quite CPU-intensive. If you enable this checkbox, the event-mode output will not block any events. This option should be used with care. For the audio-mode outputs this check-box does not have any effect.

In the Info Properties tab one can specify the hint text for the port.

2.5.3 Properties

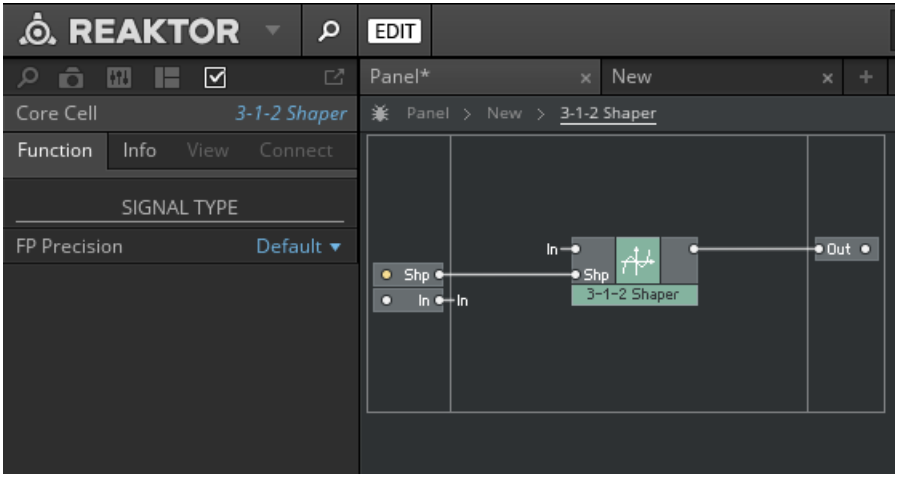
The Core Cells feature two distinct sets of Properties: the 'inside' (Core level) and the 'outside' (Primary level) ones.

The outside Properties of a Core Cell are accessible when the Core Cell is selected within the Primary Structure: They contain the Primary level configuration settings of the Core Cell.



The outside Properties of a Core Cell.

The inside Properties of a Core Cell are accessible by clicking the background of the Core Cell's internal Structure:



The inside Properties of a Core Cell.

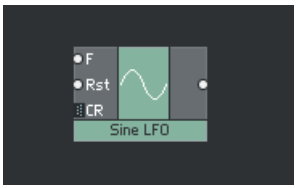
The Core Cell name and (editable) Info properties are shared between the inside and the outside Properties and can be equally accessed from either side.

The FP Precision property selects the meaning of the 'default' setting of the float type objects (Modules and ports) within the Core Cell's Structure. Simply put, the floating-point computations inside the Core Cell will be performed at this precision, unless explicitly overridden for certain elements within the Structure.

- Default: use the system-default setting, whatever that is. Technically this is 32 bit, the same as within Primary level. This setting should be used whenever possible (that is, unless there is a special reason to use a different setting).
- 32 bit: use 32 bit floats by default
- 64 bit: use 64 bit floats by default

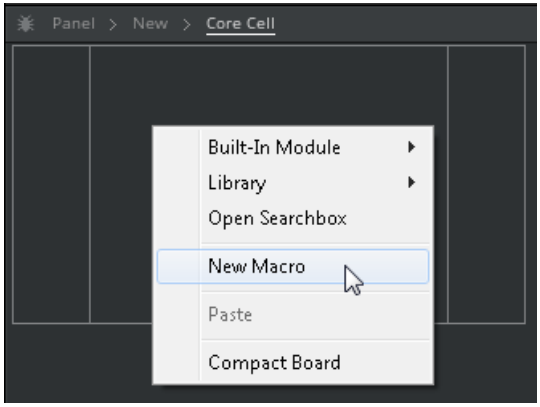
2.6 Core Macros

Conceptually seen, a **Core Macro** is not different from a Primary level Macro. This is simply a container Module which has another Core Structure inside:

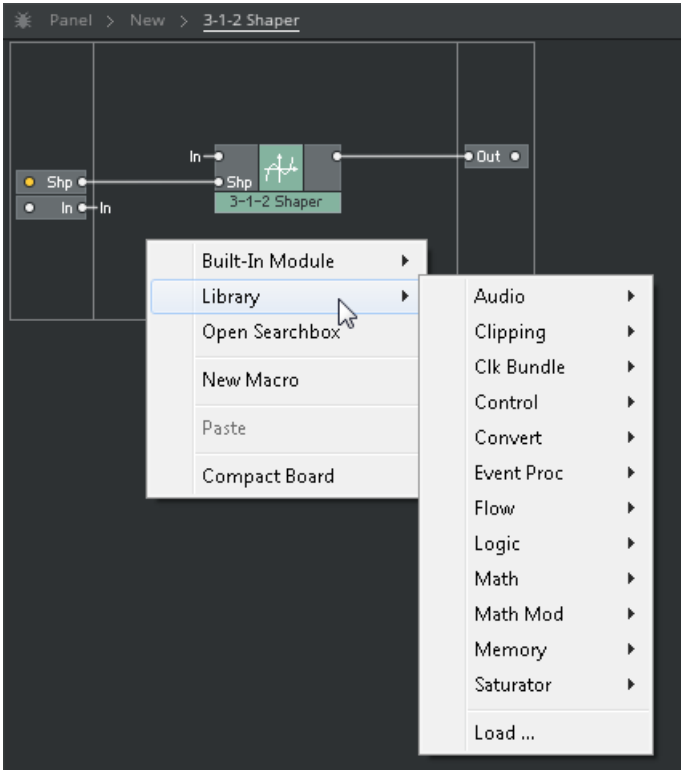


A Core Macro.

1. To create a new blank Macro, right-click in the normal area of a Core Structure and select New Macro from the context menu.



In the normal area context menu next to the Built-In Module submenu there is a Library submenu containing the factory library and the user library Macros:

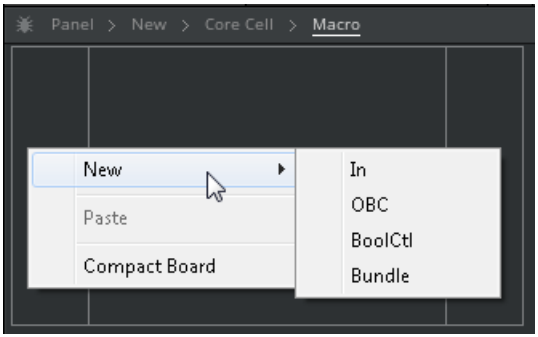


The Library submenu in the Core Macro's normal area context menu.

The internal Structures of Core Macros are very much like the ones of Core Cells, except that the available input and output port types are different.

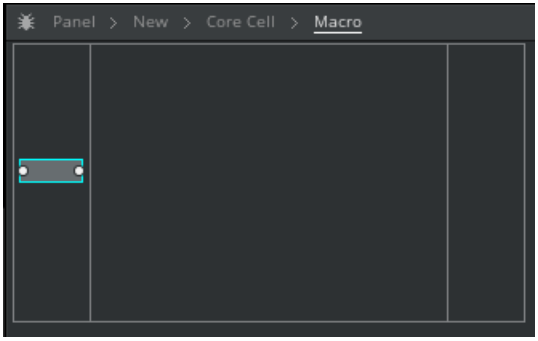
2.6.1 Inputs

The input types available in the input port area context menu of a Macro Structure correspond to different connection classes available in Core (see section ↑2.4, Core Connection Types for the connection classes and type discussion).



The Core Macro's input port area context menu.

- In: Creates a scalar type port. This is the most commonly used port type, which can be configured to float or int, of which float is most commonly used.
- OBC: Creates an OBC type port.
- BoolCtl: Creates a BoolCtl type port.
- Bundle: Creates a Bundle type port.

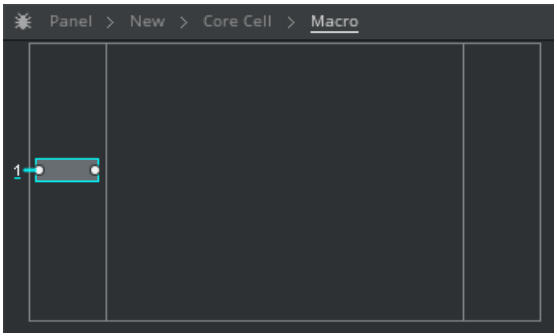


A Core Macro input port Module of float type.

The set of available properties of an input port Module varies depending on the Module type. Here one can switch the signal type between float and integer, control the floating point precision etc.

Default Connection

Differently from Core Cell port Modules, which have only an output, the Macro input port Modules also feature an input. This input is typically left disconnected and is only used if one wants to override the default behavior of the corresponding port on the outside of the Macro. Technically, if a Macro input (on the outside of the Macro) is disconnected, the input connection of the corresponding input port Module inside the Macro will be used instead. So, if there is e.g. a constant of 1 connected to the 'default' input of the input port Module, then a disconnected Macro port will default to 1 rather than 0:



A Core Macro input port Module with a non-zero default signal.

2.6.2 Outputs

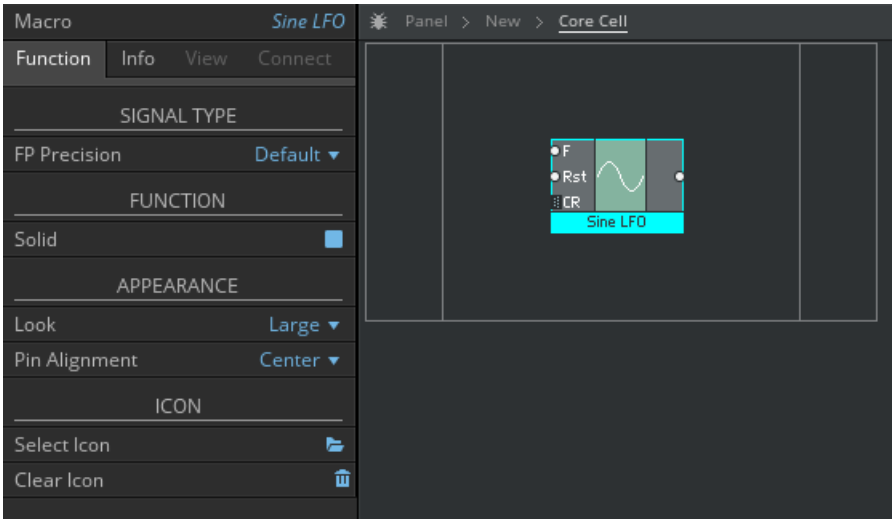
The output types available in the output port area context menu of a Macro Structure are exactly the same as the input types. Except for the absence of the 'default connection' functionality, the output port configuration is exactly the same as that of the inputs.

2.6.3 Properties

The Properties of a Macro can be accessed in two different ways:

1. Select the Macro from the outside, from the Structure it is contained in.
2. Left-click on the Structure background inside the Macro.

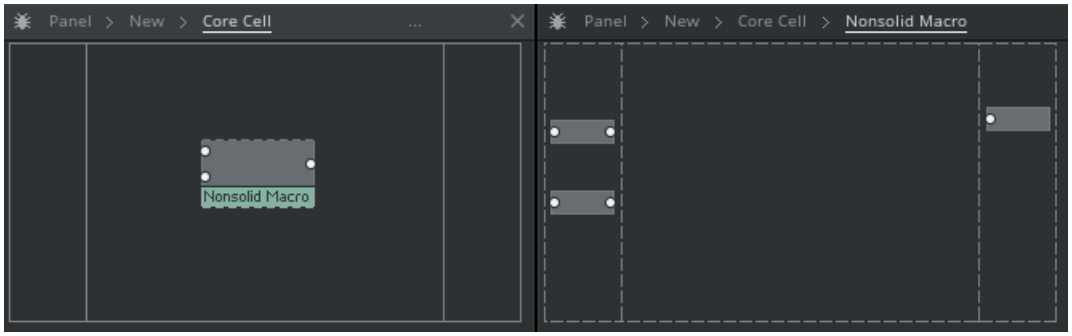
In other words, the inside and the outside Properties of a Macro are identical.



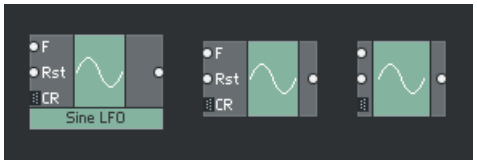
Properties of a Core Macro.

- FP Precision: controls the default precision of the float-type objects (Modules and ports) inside the Macro. It allows to specify the precision of the floating point computations inside the entire Macro (including the nested Macros), except for the areas which are explicitly configured to different precision settings. Usually this property should be left in the Default setting.
  - Default: use the same precision as the Structure that contains the Macro.
  - 32 bit: objects configured to default precision will use 32 bit precision floats (notice, that formally these objects are still considered to be configured to the default precision!) The respective float ports (scalar and OBC) on the outside of the Macro will appear as explicitly configured to 32 bit. In particular, a float OBC port which is configured to the default precision setting will appear as a default-precision port on the inside of the Macro and as a 32 bit precision port on the outside of the Macro.
  - 64 bit: the same as 32 bit, except the precision is 64 bit.
- Solid: controls the solidity of the Macro. Affects the resolution of the feedback loops (see section ↑4.10.3, Nonsolid Macros and Feedback Loops) and the visibility of the Scoped Buses (see section ↑3.3.6, Solid Areas). This property must be left in the 'On' state unless the Macro explicitly needs to be non-solid, otherwise it may make the Structure more

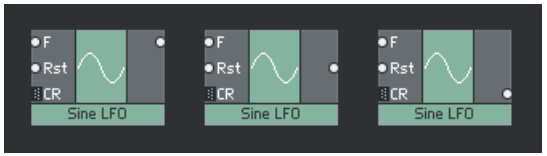
prone to difficult to find building errors and sometimes unnecessarily increase compilation times. The nonsolid Macros can be told visually by ragged edges, while their internal Structures can be told by dashed frames:



- Look: controls the outside look of the Macro. There are large, medium and small settings:



- Pin Alignment: controls whether the ports of the Macro should be aligned to the top, center or bottom in the outside look:

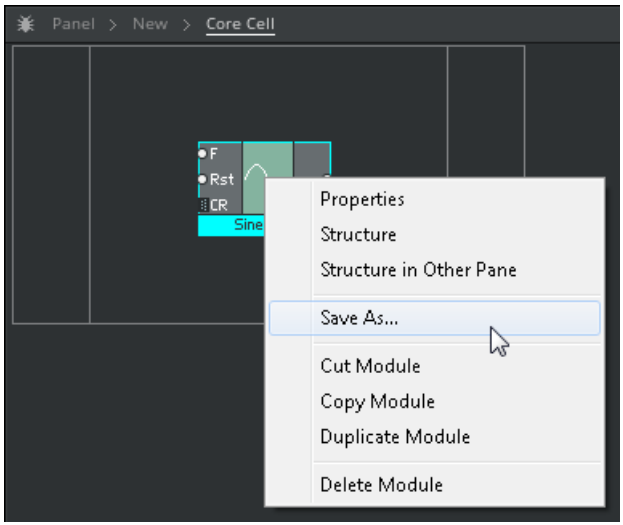


- Select Icon / Clear Icon: allows to load or unload a picture to be displayed in the outside look of the Macro. It is recommended to use a picture with an alpha channel (in PNG or TGA format).

2.7 Saving Core Cells and Core Macros

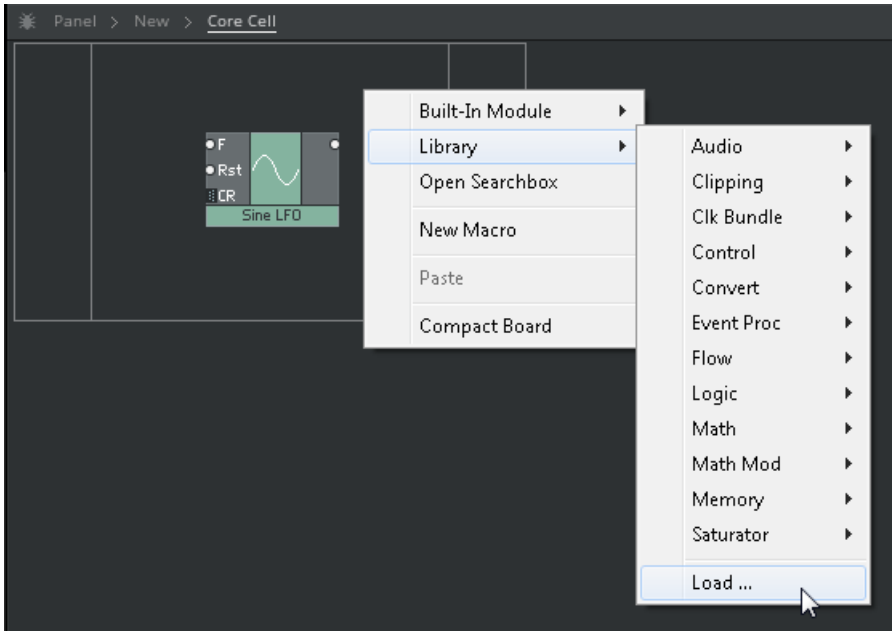
Similarly to the Primary Macros it is possible to save Core Cells and Core Macros as individual files as well as load them back into the Structure.

1. To save a Core Cell or a Core Macro, right-click it and select Save Core Cell As... or Save As... respectively.



The Save As... function can also be used for built-in Core Modules. They are saved to the same file format as Core Macros. One can use this to save built-in Core Modules with preconfigured Properties, if desired.

1. To load Core Cell and Core Macro files back, right-click on the background of the Primary or Core Structure (for Core Macros/Modules the right-click has to be within the normal area) and select Library > Load....



Similarly to the user library folder "Primary" for Primary level objects, there is a folder called "Core" for Core level objects. The folders and the Core objects located in this folder will show up in the Library submenu after a separator.



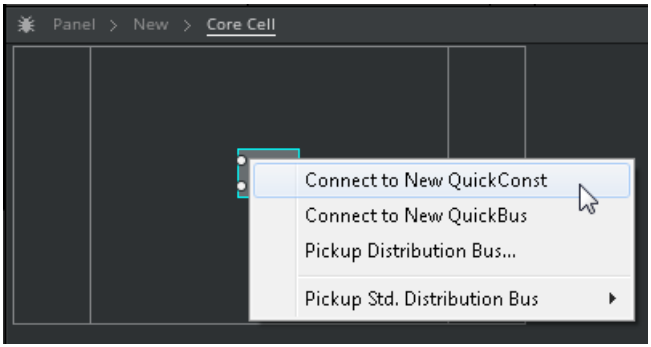
Notice that while Core Macros and Modules belong to the "Core" folder, the Core Cells belong to the "Primary" folder, since they are inserted from the Primary level and respectively need to be available in the Primary level context menu.

3 Additional Connectivity Features

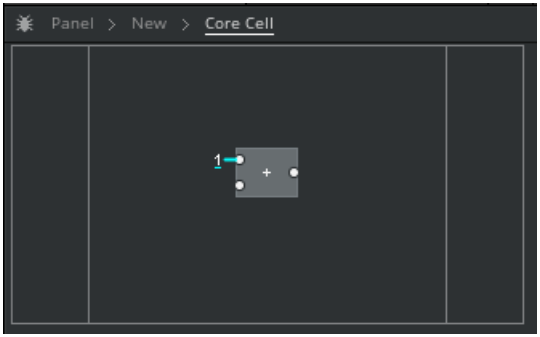
3.1 QuickConsts

QuickConsts are a lightweight alternative to *Const* Modules. To create a QuickConst do the following:

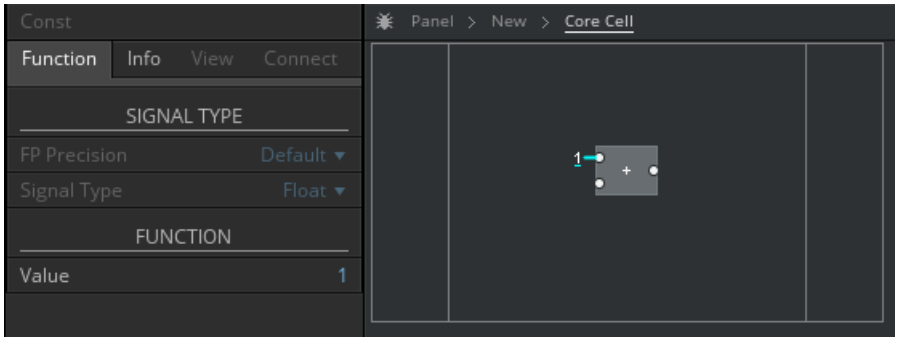
1. Right-click on a scalar input and select the Connect to New QuickConst.



2. A new QuickConst is created (if the input is already connected, the old connection will be deleted).



3. The value and the type of the QuickConst can be adjusted in the QuickConst's Properties.



The Properties show up whenever the QuickConst is selected.

1. To select a QuickConst, simply click on the QuickConst.
2. To delete a QuickConst, you can either select it and press [Del] or [Backspace], right-click on it and select Delete Connection, or click-and-drag from the input port into the Structure background.

Functionally the QuickConsts are fully identical to the ordinary *Const* Modules. However, unlike the *Const* Modules, they cannot be copied and pasted. Additionally, multiple connections to a QuickConst are not possible.

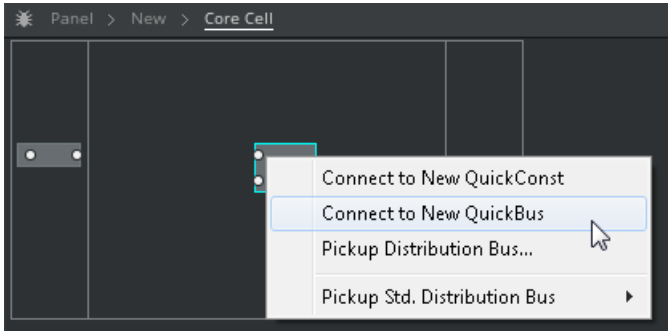
3.2 QuickBuses

QuickBuses are an 'invisible' alternative to the wires and can be used to reduce the connection clutter within a Structure.

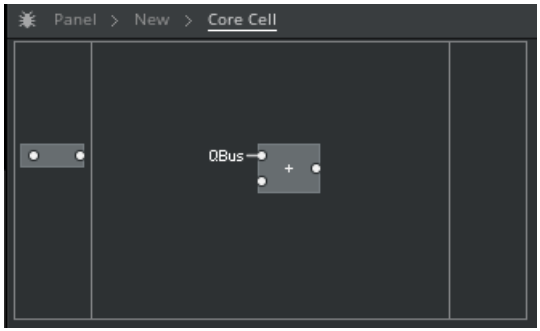
QuickBuses do not support cross-Structure connectivity. For cross-Structure connectivity use Scoped Buses (see section ↑3.3, Scoped Buses).

To create a QuickBus, do the following.

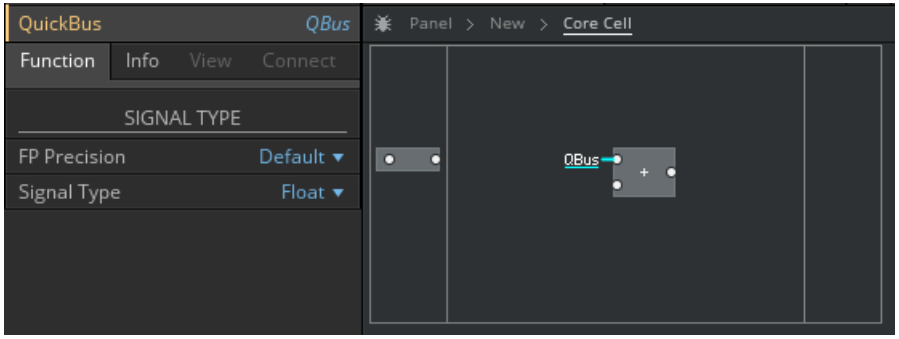
1. Right-click on an input or an output and select Connect to New QuickBus.



2. A new QuickBus with a default name (in this case "QBus") is created and the corresponding **QuickBus accessor** is connected to the port.



3. The name and the type of the QuickBus can be adjusted in the QuickBus's Properties.

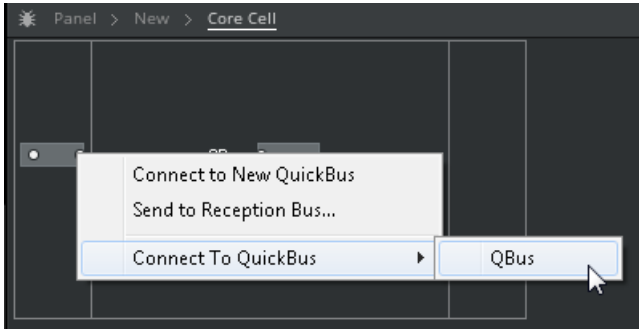


The QuickBus itself is not visible in the Structure. What is connected to the port is not the QuickBus itself, but the **QuickBus accessor**. Selecting an accessor (by clicking on it) switches the Properties to the associated QuickBus.

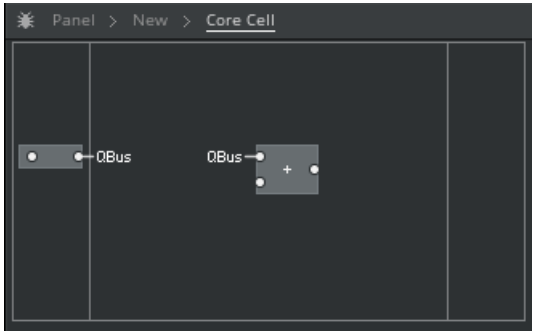
QuickBuses are supported for all port types, not just the scalar ports as QuickConsts.

In order for the QuickBus to establish a connection between an output port and an input port it has to be connected to both an input and an output. To connect a port to the existing QuickBus, do the following.

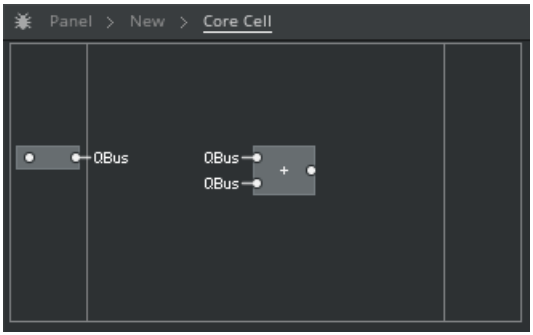
1. Right-click on the port and select the respective QuickBus's name in the Connect to QuickBus submenu.



2. Another accessor is created, thereby establishing a connection between the output and the input. Both accessors are referring to the same QuickBus object and thus are connected to each other.



The same QuickBus can be used to connect several inputs to one output. Simply right-click on an input and select the QuickBus from the Connect to QuickBus submenu to establish an additional connection to the same QuickBus:



Two connections established via a single QuickBus.

While it is possible to connect several inputs to a single QuickBus, at most one output can be connected to a given QuickBus at any given time. An attempt to connect a second output to the same QuickBus will disconnect the previously connected output from the QuickBus.

A QuickBus that is not connected to an output will be 'default-connected' according to the conventions explained in section ↑2.4.5, Connection Defaults. An input connected to a QuickBus accessor is formally a connected input (regardless of whether there is an output connected to the QuickBus or not), thus the default connection mechanism for this input is not active in this case. The explicit usage of QuickBuses which are not connected to an output is not advisable in finished Structures.

QuickBuses can have the same name within the Structure, even though giving each QuickBus a unique name is recommended. Duplicate names are automatically extended with extra indices "(1)", "(2)", "(3)" etc. These extra indices are not a part of the name and are merely a visual display feature, allowing you to distinguish between identically named QuickBuses. In principle these indices are intended to be used exclusively in temporary situations, such as after a QuickBus name collision upon copying and pasting. QuickBus name collisions in finished Structures are not recommended.

QuickBus connections can only be used between ports within the same Structure. Therefore, identically named QuickBuses from different Structures (if existing) represent completely different QuickBus objects and do not generate a collision either. In the terminology of programming languages, the QuickBus namespaces are per-Structure.

A QuickBus exists as long as there is at least one accessor associated with the QuickBus. The accessors are deleted in the same way as other connections and QuickConsts.

Disconnecting accessors by clicking and dragging to the background works only for the accessors at the input ports.

3.3 Scoped Buses

Scoped Buses allow invisible connections across several Structure layers. Unlike QuickBuses, Scoped Bus connections are asymmetric in the following sense:

- A QuickBus connection is established between two (lightweight) accessors.
- A Scoped Bus connection is established between a single (heavy-weight) **definition point** and one or multiple (lightweight) accessors.

The accessors connected to a definition point must be located within the **definition scope**. The definition scope includes the same Structure as the definition point and the Macros contained within that Structure (including the Macros contained within those Macros, etc.).

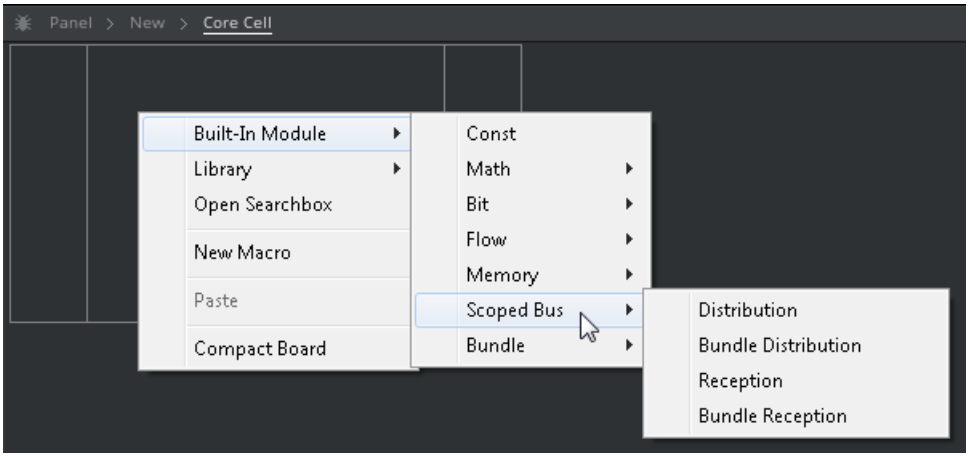
More strictly speaking the definition scope is the same **solid area** and all nested ones. See section ↑3.3.6, Solid Areas for more details.

3.3.1 Definitions

The definition points are provided in the form of built-in Modules and come in two different flavors:

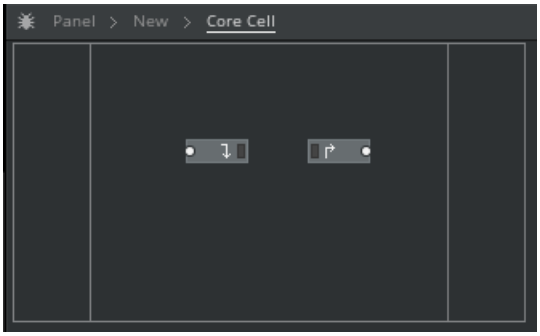
- **Distribution Scoped Buses**: The definition point acts as the output side of a connection (the signal flows from the definition point to the accessors). The input side of a connection is represented by a **scoped pickup accessor**. There can be multiple scoped pickup accessors connected to a single distribution bus. This is the most useful flavor of the two.
- **Reception Scoped Buses**: The definition point acts as the input side of a connection (the signal flows from the accessor to the definition point). The output side of a connection is represented by a **scoped send accessor**. The current implementation of REAKTOR Core allows only **single-reception Scoped Buses**, which do not support more than one connected scoped send accessor.

The connection classes supported by Scoped Buses in the current implementation of REAKTOR Core are scalar and Bundle. The Built-In Module > Scoped Bus menu allows you to create the definition points of distribution and reception flavors within these connection classes.



Scoped Bus submenu for creating distribution and reception definition Modules

A distribution definition Module has an input which is supposed to be connected to a signal source. A reception definition Module has an output which is supposed to be connected to a signal's destination:



Newly created scalar scoped distribution (left) and reception (right) Modules.

A newly created definition does not have a name and thus does not define a Scoped Bus. In order to define a Scoped Bus, the definition must be given a nonempty name in its Properties.

Both the distribution definition and reception definition Modules have a connection status indicator that turns green when one or more accessors are connected to this definition. It turns red in case of a definition conflict (see section ↑3.4.3, Definition Conflicts and Pickup Errors.).



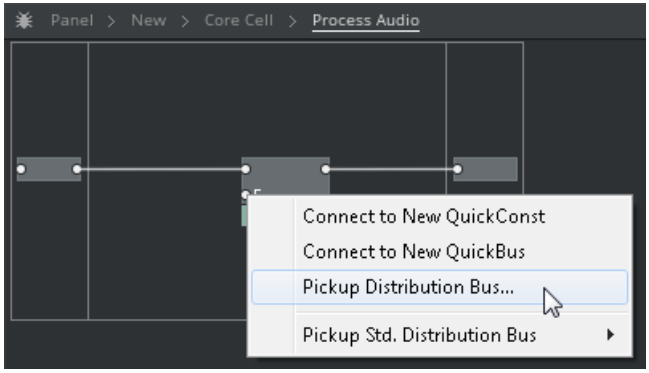
A distribution definition of a Scoped Bus named "Bus" with a green ('connected') connection status indicator.

3.3.2 Simple Scoped Access

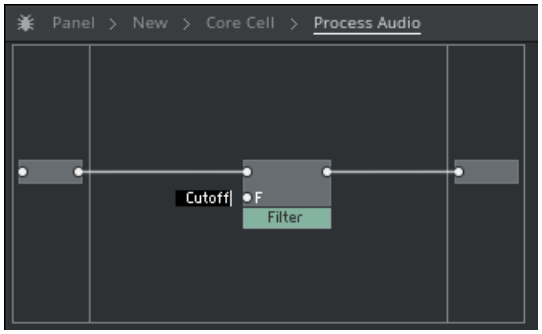
In order to make a scoped connection to a definition point an accessor needs to be created within the definition scope. For distribution buses a scoped pickup accessor (or simply 'pick-up') needs to be created, for reception buses a scoped send accessor (or simply 'send') needs to be created.

To create a scoped pickup accessor do the following:

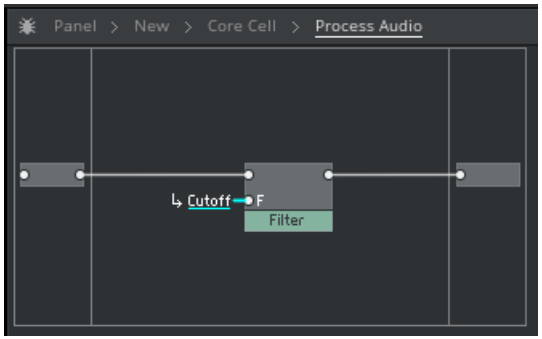
1. Right-click on an input of the scalar or Bundle type and select Pickup Distribution Bus....



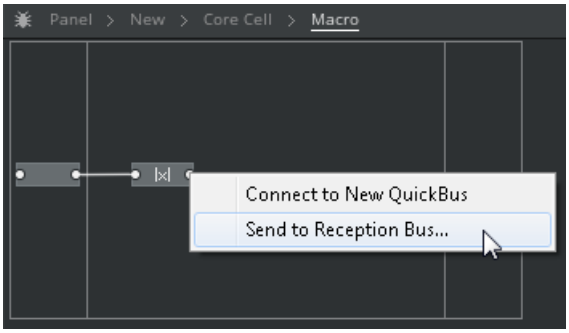
2. A text entry field appears next to the input. Enter a nonempty name for the scoped pickup accessor.



3. Press [Enter] to create a pickup with the name you just typed in. If you want to abort the creation of the pickup, press [Esc].

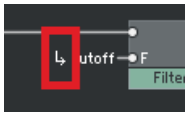


Scoped sends are created in the same way, except that you have to right-click on an output port:



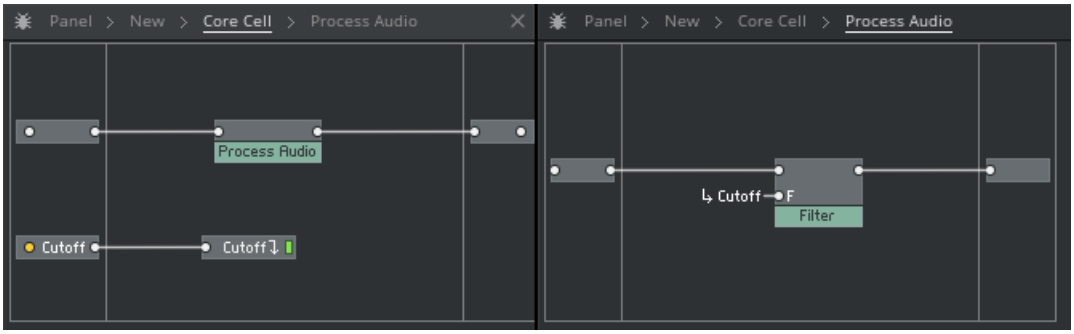
Creation of a scoped send.

The scoped accessors can be distinguished from QuickBus accessors by the 'broken' arrow next to them:



A scoped accessor is looking for an identically named definition, within the scope of which the accessor is located. The search is started in the same Structure (more correctly, in the same **solid area**, see section ↑3.3.6, Solid Areas) where the accessor is located. If an identically named definition is found, the accessor connects to that definition. If such definition is not found, the accessor looks in the parent Structure (parent solid area), and so on. If the definition is not found at the topmost (Core Cell) level, the accessor fails to connect.

The following example demonstrates a pickup named "Cutoff" connecting to the identically-named definition one level higher:



A scoped connection.

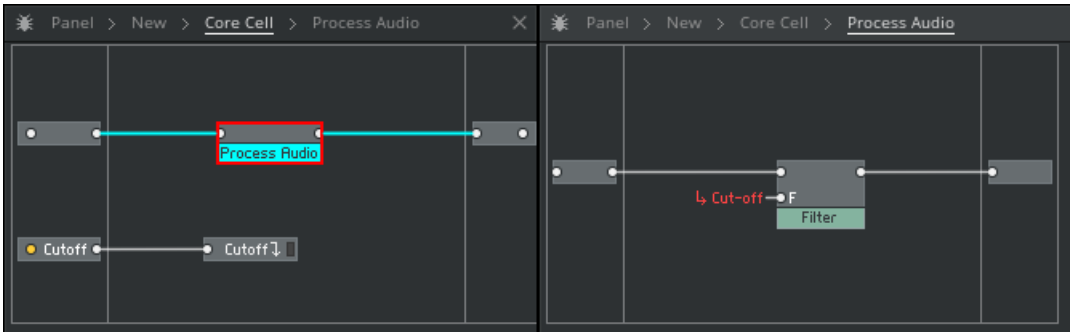
A scoped accessor can be selected by a single left-click. Differently from QuickBuses, accessor selection does not provide access to the bus's Properties.

A scoped accessor can be renamed by double-clicking it. Renaming is confirmed by pressing [Enter] or aborted by pressing [Esc], just like when giving an accessor a name upon creation. An accessor cannot have an empty name. If renamed to an empty name, the accessor is deleted. Otherwise, the deletion of scoped accessors can be done in the same way as for other connections and QuickBus accessors.

Differently from QuickBus accessors, which always stay connected to the same QuickBus and where the rename function renames the bus itself and not the accessors, the name of a scoped definition or of an accessor applies only to this specific definition or the accessor. If a scoped accessor or definition is renamed, it gets reconnected to a different definition or to a different set of accessors according to the new name.

3.3.3 Scoped Connection Errors

The scoped name lookup completely disregards the definition flavor and type (in programming language terminology, all Scoped Buses share the same namespace). If a scoped accessor finds a definition with a matching name that it cannot connect to (because the flavor and/or type are incompatible), it will fail to connect. Equally a scoped accessor fails to connect if it does not find a definition of a matching name at all. Such accessors are highlighted in red:



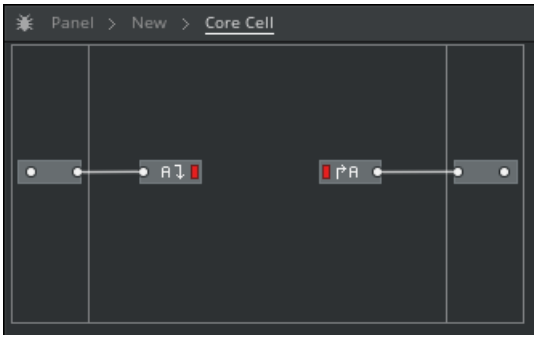
A scoped pickup accessor that fails to connect.

Aside from highlighting the accessor itself, the Core compiler highlights the parent Macros containing this accessor. The highlighting of Macros goes up in the Macro hierarchy as far as the 'error locality area' reaches.

The error locality area is (approximately) defined as the area that can affect the accessor error. E.g. if an accessor does not find a definition of a matching name, the parent Macros will be highlighted all the way up to the Core Cell's Structure. If, on the other hand, the error is due to a type or flavor incompatibility, the highlighting will go up only to the level of the respective bus definition.

3.3.4 Name Collisions and Definition Overriding

Two identically named definitions within the same Structure produce a name collision, regardless of their types and flavors. The connection status indicators of the conflicting definitions turn red:



Two conflicting definitions.

It is however possible to have identically named definitions if one Structure is a (direct or indirect) parent of the other. In this case the lower-level (the 'more nested') definition **overrides** the higher level one. Overriding means that from this Structure on, the new definition will be used by the respectively named accessors.

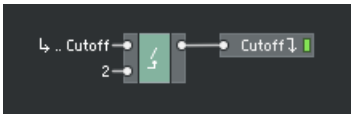
> The overriding effect is naturally following from the described earlier lookup process for a name-matched definition from the accessor. Clearly, the lookup process will find the 'most nested' definition above (or at) the accessor's level.

Identically named definitions in 'unrelated' Structures (that is, when neither of two different Structures is the parent of the other) do not produce a conflict, as their name visibility scopes do not overlap.

3.3.5 Parent Mode Accessors

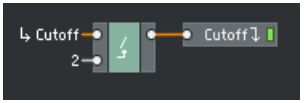
Sometimes, when overriding a definition, the new definition needs to be specified 'in terms' of the old definition. This makes it necessary (or at least highly desireable) to be able to access the old definition within the Structure of the new definition. This can be done by using a **parent-mode** accessor.

The accessor parent mode is specified by typing two dots in front of the accessor's name (when creating or renaming the accessor). The parent mode causes the accessor to ignore the definition within the same Structure and start the lookup for the matching definition one level higher, from the parent Structure. So, in case of a definition override, a parent-mode accessor will connect to the old rather than the new definition.



An example of parent-mode accessor usage

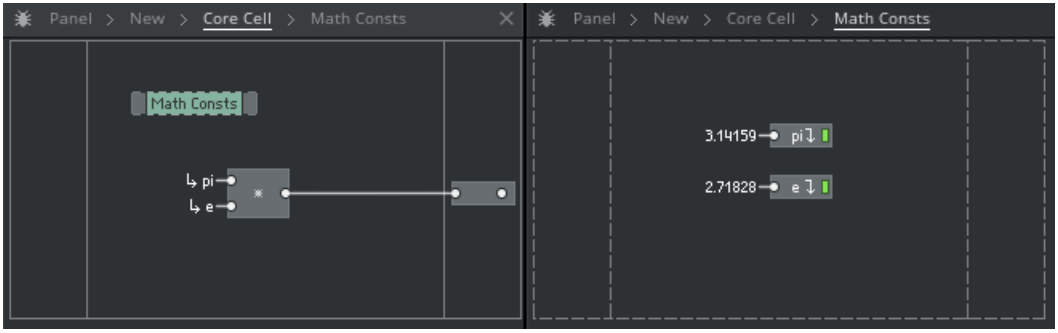
In the above example there is an (overriding) redefinition of the "Cutoff" Scoped Bus to a halved value. If the scoped pickup accessor on the left had not been set to parent mode, it would have connected to the new definition, creating a feedback loop (see section ↑4.10, Feedback Connections for a discussion of feedback loops):



An example of a logical error caused by the missing parent mode of the accessor.

3.3.6 Solid Areas

If a Scoped Bus is defined inside a Macro, the definition scope does not include the Macro's parent Structure. This makes it impossible to create a Macro the purpose of which is to define a Scoped Bus. This restriction can be lifted by turning off the Solid property of the Macro. In this case the Scoped Bus definitions contained directly within the Macro become visible on the parent level:



Using a non-solid Macro to define Scoped Buses for the parent Structure.



The Macro solidity also affects the resolution of the feedback loops (see section ↑4.10.3, Nonsolid Macros and Feedback Loops).

Whether using the non-solid Macros to define Scoped Buses within the parent Macro's context is good practice or not is questionable, since in this case the definitions are not visible at the top Structure of the definition scope. In any case this possibility should be used with care.



The description of the Scoped Bus functionality up to this point in the manual is approximate. It is only correct assuming that all Macros are solid. This assumption was done for simplifying the material presentation. A precise description of the Scoped Bus functionality follows below.

A Precise Description of Scoped Bus Rules

The Scoped Bus mechanism of REAKTOR Core works in terms of the **solid areas** rather than in terms of Structures. A **solid area** is the area bounded by the boundaries of solid Macros. If a Macro is solid and so are all directly nested Macros, then the Macro's Structure constitutes a solid area. If all Macros in a Core Cell are solid, then there is no difference between Structures and solid areas. If in such a Core Cell one Macro is made non-solid, then the solid boundary between this Macro and its parent Structure disappears, meaning that the parent Structure and the Macro's Structure now constitute one solid area, etc.

Consequently, the following applies:

- The scope of a definition starts at the solid area of the definition and extends to all nested solid areas, unless overridden by another definition in a nested solid area.
- Two identically named definitions within one solid area produce a conflict.
- A parent-mode accessor looks up for the matching definition starting from the parent solid area, rather than the parent solid Structure.

3.3.7 Name Normalization and Escaping

Normalization

Since scoped connections are established by name, it is very important that the names are visually distinguishable in the Structure. For that reason REAKTOR Core automatically collapses all whitespace within the names:

- Whitespaces in the middle of a name are collapsed to a single space.
- Whitespaces at the beginning and at the end of a name are eliminated completely.

This process is referred to as **normalization**.

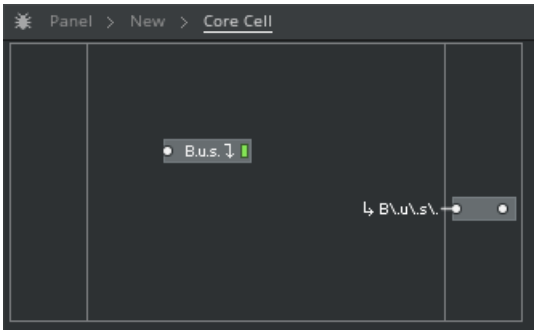
Escaping

The dot symbol has a special meaning in the scoped accessors. It is used to specify the accessor parent mode. It is also used as a fiber separator in the Bundle subfiber pickup accessors (see section ↑3.4.4, Scoped Bus Subfiber Pickups). At the same time it is still possible to define a Scoped Bus with one or more dots within its name. In order to be able to access such definition the accessor must be able to specify that a certain dot should not be treated as a special syntactical element (parent mode specifier or fiber separator), but an ordinary dot. This possibility is implemented in the form of the so called **escaping**.

In order to specify that a certain dot in the accessor name is an ordinary dot, the dot must be prefixed by a backslash.

The backslash is referred to as the **escape character**.

The backslash thereby also obtains a special syntactical meaning and thus needs to be escaped itself if an ordinary backslash character is being meant.



Dot escaping in an accessor.

In the above case the Scoped Bus name ("B.u.s.") contains three dots. In the accessor each of these dots needs to be prefixed by a backslash.

Escaping is only applied to accessors, not to definitions.

If some special symbol within an accessor name is used at an inappropriate position (e.g. three dots are used in a row at the beginning of an accessor), then REAKTOR Core will not 'approve' it as a special symbol and will automatically escape it, explicitly highlighting its non-special meaning.

3.4 Bundles

Bundles are a connection type, in which each 'cable' has multiple wires internally. These internal wires are referred to as the Bundle **fibers**.

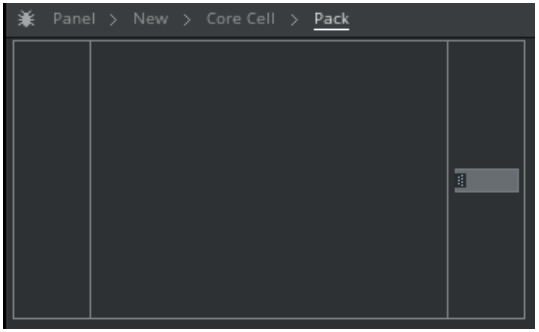
3.4.1 Bundle Pack

In order to put several fibers into one Bundle use a Built-In Module > Bundle > Pack:



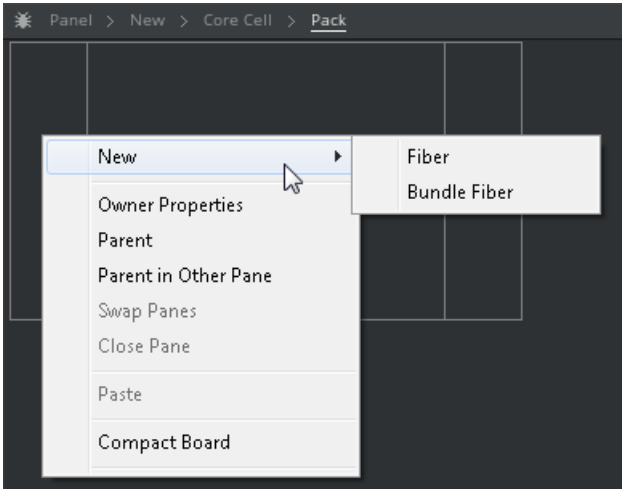
A newly created Bundle Pack Module.

A newly created Bundle *Pack* Module has no inputs and one Bundle output. Having no inputs means that the outgoing Bundle has no fibers. In order to add fibers to the Bundle, navigate into the internal Structure of the *Pack* Module by double-clicking on it, in the same way as it is done with Macros. The *Pack* Module's internal Structure will show up:



The empty internal Structure of a newly created Pack Module.

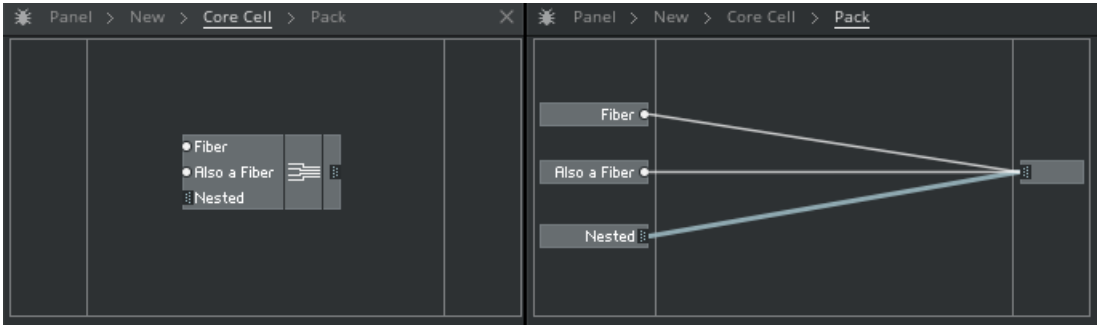
The only editable part of the Pack Module's internal Structure is the input area on the left. The inputs correspond to the Bundle fibers and are referred to as **fiber definitions**. Currently, only scalar and Bundle fiber definitions are supported:



The Pack Module's input area context menu.

The fiber definitions of Bundle type allow to nest Bundles.

The fiber definitions created in the *Pack* Module's internal Structure show up as inputs on the outside of the Module:



Fiber definitions and the corresponding inputs of the Pack Module.

Each fiber definition has to be given a name. Definitions with empty names cannot be picked up on the other side of the Bundle.

Similarly to the Scoped Bus names, the fiber names are automatically normalized.

Most commonly the output of the Pack Module is connected to:

- a Macro output port from the inside of a Macro (if the *Pack* Module is contained within a Macro and the Macro output interface is employing a Bundle), or
- a Macro input from the outside of a Macro, or
- a scoped distribution definition.

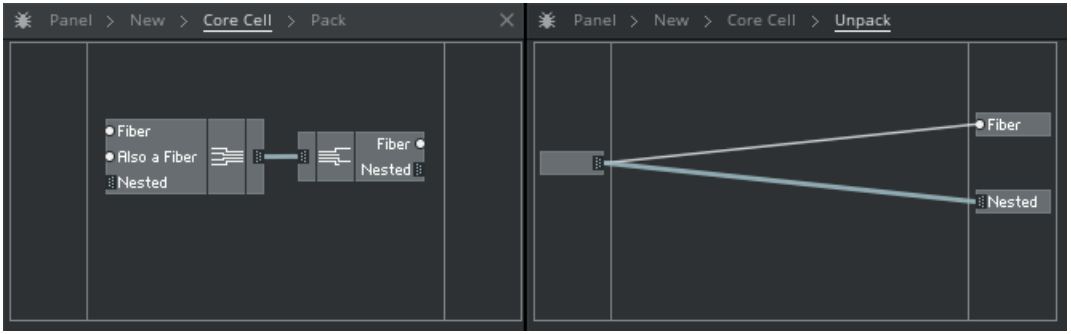
3.4.2 Bundle Unpack

In order to access individual fibers within a Bundle use Built-In Modules > Bundle > Unpack:



A newly created Bundle Unpack Module

A newly created Bundle *Unpack* Module has a Bundle input and no outputs. The Bundle input is supposed to be connected to the Bundle, the fibers of which need to be accessed. The outputs of the *Unpack* Module correspond to the fibers which need to be accessed and are created as outputs in the *Unpack* Module's internal Structure. They are referred to as **fiber pickups**.



Fiber pickups and the corresponding outputs of the Unpack Module.

The fiber pickups need to be given names identical to the names of the fibers that are being picked up. They also need to have a compatible type.

It is not necessary to pick up all the fibers of the Bundle in the *Unpack* Module. Only a subset of the fibers may be picked up. E.g. in the above example the "Also a Fiber" fiber is not being picked up. The pickups do not have to be arranged in the same order as definitions either.

3.4.3 Definition Conflicts and Pickup Errors

Definition Conflicts

The fibers within a Bundle are getting distinguished by their name. Thus, a Bundle should not have two identically named fibers. If it does, then these fibers generate a conflict error:



A definition conflict created by two identically named fibers.

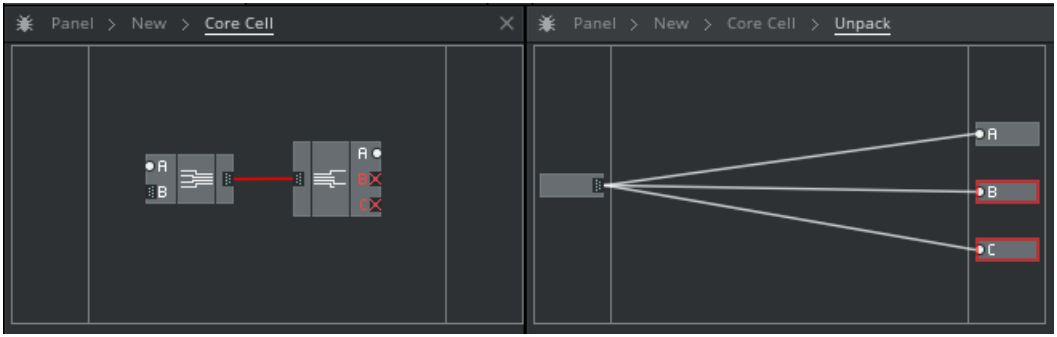
A conflicting definition does not generate a fiber in the Bundle.

More precisely, all conflicting definitions of the same name generate a **phantom** fiber in the Bundle. Phantom fibers cannot be picked up, but are participating in Bundle merging (see section ↑3.4.5, Bundle Merging) and splitting (see section ↑3.4.6, Bundle Splitting).

Definitions with empty names are considered void and do not conflict with each other.

Pickup Errors

An attempt to pick up a missing fiber, a phantom fiber or an incompatible fiber will generate a pickup error:



Pickup errors generated by missing and incompatible fibers.

The above picture shows two errors:

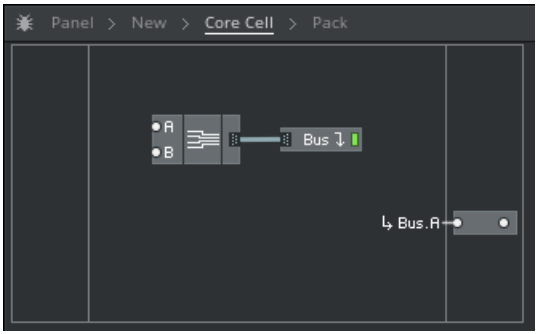
- The error in the pickup "B" is due to the type incompatibility (since the fiber "B" is of Bundle type, while the pickup is scalar).
- The error in the pickup "C" is due to the missing fiber (since the fiber "C" is not present in the incoming Bundle).

Pickup errors cause the Bundle wire to be highlighted in red all the way up to the definitions. This is done for use cases such as connecting together two library Macros by a Bundle wire. If the Bundle output of one Macro is incompatible to the Bundle input of the other Macro, the wire between the Macros will be highlighted in red. The wire highlighting also helps tracing the erroneous connection all the way between the definitions and the pickups in both directions.

> This highlighting approach follows the same idea of the error locality discussed in section ↑3.3.3, Scoped Connection Errors.

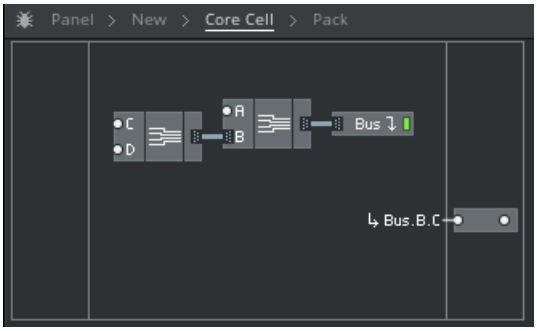
3.4.4 Scoped Bus Subfiber Pickups

As a convenience feature, when accessing a Bundle Scoped Bus, the scoped pickup accessor can access individual fibers of a Bundle. To do this, simply type the fiber name after the bus name, separated by a dot in between:



Scoped subfiber pickup.

In case of nested Bundles, subfiber pickups can be chained within one scoped pickup:

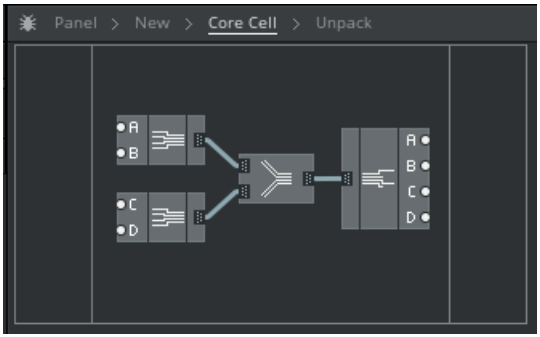


Scoped nested subfiber pickup.

There is no corresponding feature for scoped send accessors.

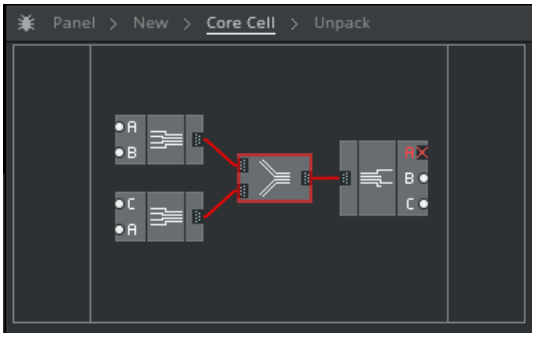
3.4.5 Bundle Merging

As a means of Bundle manipulation, two Bundles can be merged together into a single Bundle by the Built-In Module > Bundle > Merge:



Merging two Bundles into a single one.

The merged Bundles should not have identically named fibers, otherwise a merge conflict will occur. The conflicting name is turned into a phantom fiber in the output Bundle and cannot be picked up:

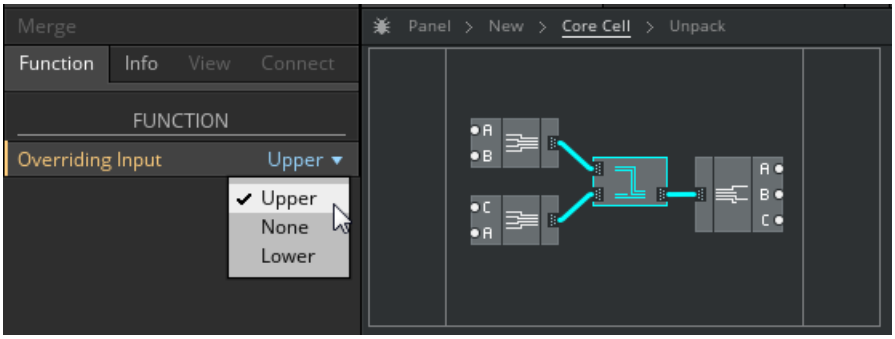


A Bundle merge conflict.

> A phantom fiber can generate further merge conflicts downstream if merged with identically named fibers.

Override Merging

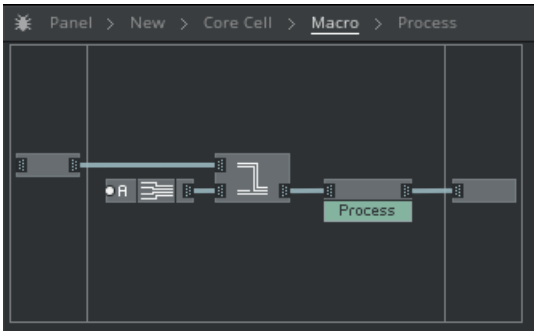
By selecting an overriding input in the Bundle *Merge*'s Properties the Bundle *Merge* Module can be switched into override mode:



An override mode Merge.

This causes the fibers at the **overriding input** to take precedence over the fibers of the other input in case of a name conflict.

The overriding mode is also useful as a means to provide default definitions for optional Bundle fibers in a library or framework Macro:



Using the override mode to provide defaults for optional fibers.

In the above example the incoming Bundle is expected to contain certain fibers, among which the "A" fiber is optional. If the "A" fiber is not present in the incoming Bundle, a default scalar fiber of zero value is inserted into the Bundle before it goes further into the "Process" Macro.

In the 'providing the defaults' use case the main Bundle has to go into the overriding input of the *Merge* Module, while the defaults should be connected to the other ('lower priority') input.

3.4.6 Bundle Splitting

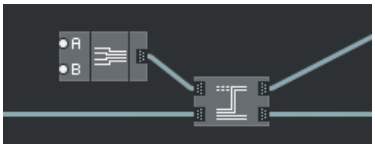
The reverse of Bundle merging is Bundle splitting, done by Built-In Module > Bundle > Split:



A Bundle Split Module.

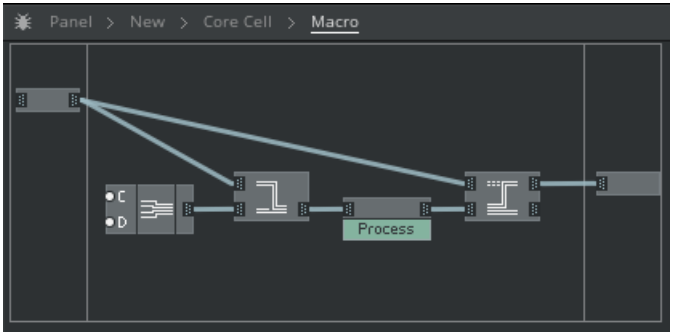
The Bundle that needs to be split should be connected to the lower input of the *Split* Module. The upper input of the *Split* Module is the **template** input. It controls the routing of the split fibers. The routing control is purely name-based. The fibers in the lower input for which there is an identically-named fiber in the template are routed to the upper output. The remaining fibers are routed to the lower output.

One can explicitly 'route off' certain fibers from a Bundle by connecting a 'dummy' *Pack* Module to the template input:



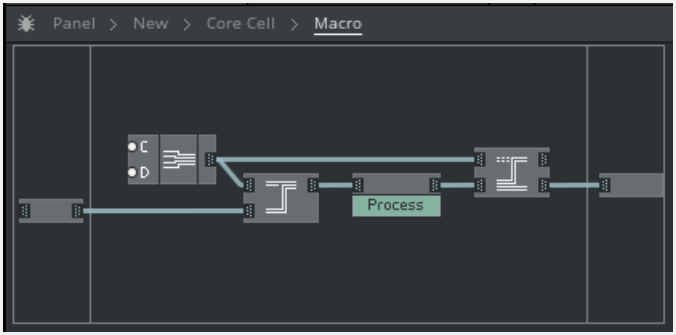
'Routing off' of fibers "A" and "B" from the Bundle.

Another use case is to remove the fibers that have been 'temporarily' merged into the Bundle:



Removing the temporarily merged 'default fibers'.

In the above example it is important that the *Split* Module is using the original Bundle as the template. A possible alternative implementation using the defaults as the template will not work correctly if the default fibers are already present in the input Bundle of the Macro, because these fibers are routed off by the *Split*:



4 Processing Model of Core

4.1 Events

A fundamental concept of signal processing in Core is the **event**. Similarly to the Primary level, an event within REAKTOR Core simply means that at some point in time a value has been generated at some output port. This value is immediately transmitted further downstream along the wires (or connections of other kinds), which is referred to as 'an event is being sent from the output to the connected inputs.' The downstream Modules then usually generate further events at their outputs in response to the incoming events.

The default event triggering convention in Core is the following: **a Module sends an output event in response to an incoming event at any of its inputs**.

This convention is most commonly used for the math processing Modules such as the ones found in Built-In Module > Math and Built-In Module > Bit menus. See section ↑5.1, Expression Computation for an example of efficient use of this convention. Nevertheless, there are a number of other different triggering conventions used for certain groups of Modules or individual Modules (constants, Core Cell inputs, Modulation Macros, audio processing Macros, audio generation Macros etc.). These conventions are described in the respective areas of the manual.



Differently from Primary, within REAKTOR Core Structures there is no distinction between audio- and event-mode connections. In Core everything is an event. An audio signal is simply consisting of events regularly sent at the sampling rate.

Conversion between Primary and Core Signals

The conversion between Primary audio/event signals and Core event signals is done according to the following:

- The audio-mode input ports of Core Cells produce events at the sampling rate of the Primary Ensemble.
- The event-mode input ports of Core Cells forward Primary level events arriving on the outside of the Core Cell to the Core Cell's internal Structure.

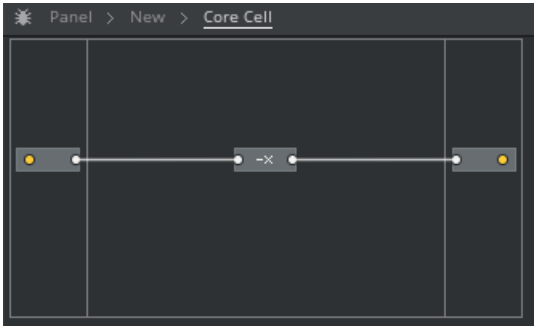
Respectively:

- The audio-mode output ports of Core Cells simply make their internal (Core level) value available to the external Primary level. This internal value is simply the value of the most recently arrived event.
- The event-mode output ports of Core Cells simply forward the internally arriving events to the external Primary level.

The Primary level does not support simultaneous events (see section ↑4.2, Processing Order for the discussion of simultaneous events in Core). If several Core events arrive simultaneously at the event-mode outputs of a Core Cell, they will be sent to the outside Primary Structure consecutively, one after the other. The order of sending is 'top-to-bottom', according to the visual ordering of the Core Cell output ports. The events are sent immediately upon the completion of the respective handler (see section ↑5.6.1, Core Cell Handlers). Particularly, if the events are generated from the audio handler, they will be sent before any further audio processing is done in the Primary Structure.

Event Processing and Triggering Example

The following example Core Cell Structure implements an event signal inverter Core Cell:



A simple audio inverter Core Cell.

The Structure consists of an event-mode input port Module, an event-mode output port Module and a Core level inverter Module connected in between, where the latter can be found under Built-In Module > Math > -x. Since in Core there is no difference between audio and events, the built-in Core level inverter Module does not need to be configured (neither automatically nor manually) to an audio or an event-mode.

The processing of an incoming Primary level event by the above Core Cell is happening like follows:

1. A Primary level event is arriving at the Core Cell's event input
2. The Core Cell's event input is converting the Primary event into the Core event, which is then sent from the output of the input port Module inside the Core Structure.
3. The event reaches the input of the *-x* Module.
4. The *-x* Module is inverting the value and sends the result as its output event.
5. The event reaches the input of the Core Cell's output port Module.
6. The Core Cell's output port is converting the Core event into the Primary event and sends it further within the Primary level.

The above Core Cell can be turned into an audio inverter Core Cell by simply reconfiguring the input and output port Modules of the Core Cell to the audio-mode. In this case the input port Module will send the events regularly at the audio rate. Configuring the Core Cell's ports to an event-mode input and an audio-mode output is also possible.

One also can configure the input to the audio-mode and the output to the event-mode. In this case the expected behavior of the Core Cell is to send the Primary level output events at an audio rate. However, since the Primary level handles the events differently from Core, such a dense event stream on the Primary level will typically cause a noticeable CPU overhead. In order to avoid accidentally doing this by mistake, the Core Cell's output port has the Allow Audio Events property, which is disabled by default. When disabled, it will prevent the events originating at the Core Cell's audio-mode inputs (or another audio rate source, such as SR.C and CR.C) from being sent to the Primary level, blocking the events completely. The property has no effect on audio-mode ports.

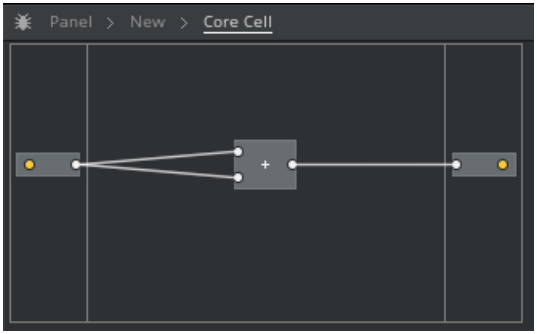
> Events derived from an audio rate source by routing (e.g. custom CR.C of a lower rate) will be equally blocked. On the other hand, the events derived from the event-mode Core Cell ports are not blocked. The initialization event (see the discussion of the initialization event in section ↑4.4, Initialization), including the initialization event obtained from the audio-mode Core Cell ports, is not blocked either.

4.2 Processing Order

As long as a Core event is processed exclusively in a serial fashion, there is no difference between Primary and Core event processing. The event is simply propagated downstream from one Module to the other. There is however a fundamental difference if the event propagation path splits into several branches.

On the Primary level an event that is sent to more than one destination is forwarded to each of these destinations in turn, one destination after the other. If the event paths later merge back again, then, instead of one event arriving at the mergepoint, there are two or more, depending on how many parallel paths the event was split into before merging back (for details about Primary level event processing, see the Building in Primary document).

In Core (logically seen) an event is sent to all such parallel destinations **simultaneously** on all paths. The following Core Cell Structure illustrates this:



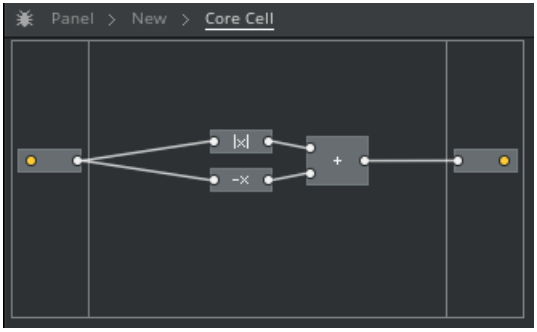
A Core Cell Structure illustrating the simultaneity of parallel events in Core.

The purpose of this Core Cell is simply to double the values of the incoming events. Instead of multiplying the values by 2, the signal's value is added to itself by using the adder Module (Built-In Module > Math > +).

Each incoming event is sent to the two inputs of the adder on two parallel paths. In Primary, the incoming event would first arrive at one input of the adder, producing an output event at the adder's output, and later it would arrive at the other input of the adder, producing another output event at the adder's output. Thus, for each incoming audio event the adder would have

generated two output events. This would lead not only to a doubling of the value, but also to a doubling of the count of the incoming event. In Core, the event arrives **simultaneously** at both inputs of the adder, so the adder produces a **single** output event.

The simultaneity of parallel events is of course extended to more complicated Structures, e.g. like the following one:



Another Core Cell Structure illustrating the event parallelism in Core.

In this Structure the event is going through the absolute value Module on the upper path and through the inverter on the lower path, and then both paths are merged by the adder. Of course, internally one of the two parallel Modules (the absolute value and the inverter) will be processed before the other. However, both the output event of the absolute value Module and the output event of the inverter Module will arrive at the adder's inputs simultaneously, thereby producing one event at the adder's output.

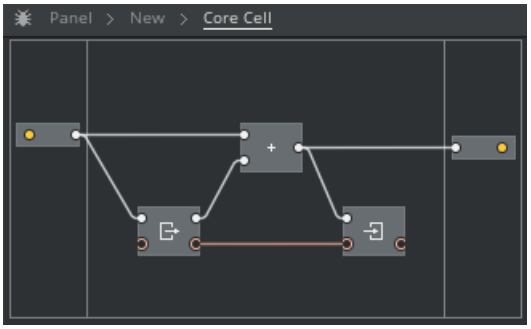
Thus, logically seen, it is impossible to tell which of the two parallel Modules (the absolute value and the inverter) has been processed first. From the formal point of view, both Modules can be considered as being processed simultaneously.

The processing order rule in Core is the following. If two Modules are connected by a unidirectional path (that is one can start at one of the Modules and reach the other one by always going only 'downstream' or only 'upstream') then the upstream Module is processed before the downstream one. If there is no unidirectional path between the two Modules, then their relative processing order is undefined. Such unordered Modules can be considered as 'simultaneously processed' ones.

The 'simultaneous processing' paradigm is not applicable to the explicit accesses to the memory by the OBC Modules. Memory accesses from the unordered Modules are serial and unordered.

4.3 Object Bus Connections (OBC)

Explicit memory storage is handled in Core by the *Read* and *Write* Modules performing the memory reading and the memory writing operations respectively. These Modules can be found under the Built-In Module > Memory menu. For the purpose of identifying the storage items referred to by these Modules, Core uses a dedicated connection class, called OBC (Object Bus Connection):



A simple event accumulator Core Cell.

In the above Core Structure there is a *Read* Module on the left, followed by an adder, followed by a *Write* Module. The direct connection between a *Read* Module and a *Write* Module is an OBC connection. It means that the *Read* and the *Write* Modules access the same storage (or the same variable, in terms of text-based programming languages). This storage is implicitly created by the mere fact that the *Read* and the *Write* Modules want to access some storage. Had there been no OBC connection between the *Read* and *Write* Modules, they would have accessed two independent storage items.



All Read and Write Modules connected together by OBC connections access one and the same memory storage (as long as there are no special OBC Modules, like e.g. the *Index* Module in between them).

The above Structure implements a simple event accumulator and works as follows:

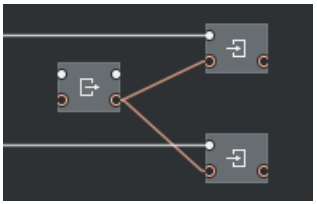
1. An incoming event from the Primary level is converted into a Core event by the input Module and is sent to the *Read* Module and to the adder.

2. The event arrives at the upper input of the *Read* Module. This is the clock or triggering input of the *Read* Module. In response to this clock event the *Read* Module reads the value from the memory and sends it from its upper output (the initial value of the memory is zero).
3. The event from the *input Module* and the event from the *Read* Module simultaneously arrive at the inputs of the adder. The adder computes the sum and sends it from its output.
4. The output event from the adder is sent to the Core Cell's output and also to the *Write* Module. The *Write* Module stores the new value of the accumulated sum in the memory, while the output forwards the same value to the Primary level.

The arrays and array connections which are a special kind of OBC are discussed separately in section ↑4.7, Arrays.

Memory Access Ordering

In typical cases the memory accesses are naturally ordered by the OBC connection between the *Read* and the *Write* Modules. However, it is possible to construct Structures where these accesses are not ordered relative to each other:



Unordered memory accesses.

In the above picture the two *Write* Modules are ordered relative to the *Read* Module, but are not ordered relative to each other. If an event comes simultaneously on both wires, then it is not defined which value will be written first and which second. **Such Structures are generally not recommended, unless the access order is explicitly unimportant.**



As in the other cases of unspecified behavior, it is highly unrecommended to rely on the de-facto ordering of conceptually unordered OBC Modules, even if this ordering seems consistent.



As a general rule, it is recommended to ensure the correct ordering of the OBC Modules (accessing the same memory) by arranging them serially along one and the same OBC wire.



Macro boundaries by themselves do not ensure any ordering (please see section ↑4.7.4, Advanced OBC Ordering for a more detailed explanation).

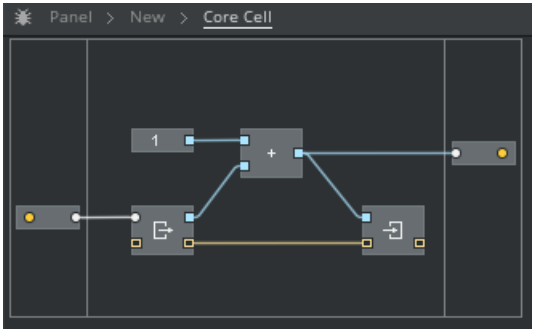
4.4 Initialization

Upon loading, before the regular processing begins, the Core Structures are getting initialized. The Core Cell initialization is performed as a part of the Primary level initialization process (please see the Building in Primary document for more information).The Core Cell initialization runs in the following steps.

- Firstly, all writable memory of the Core Cell is reset to zero. This includes the states of the output ports and the OBC storage.
- Then, an initialization event is sent simultaneously from all initialization event sources, which include the following.
  - Audio-mode Core Cell inputs. These always send the initialization event. The value of the event is equal to the value available on the outside of the Core Cell at the moment of the Core Cell's initialization.
  - Event-mode Core Cell inputs. These send the initialization event if and only if there is a Primary level initialization event coming from the outside at the moment of the Core Cell's initialization. The event's value is equal to the value of the incoming Primary level event.
  - Constants and QuickConsts (also including unconnected inputs without special default meaning as well as scoped and Bundle pickups which failed to connect). These always send the initialization event. The value of the event is equal to the constant's value. The constants never send any other events afterwards.
- After that, the Primary level initialization process might send additional events after the Core's initialization event is finished. From a Core perspective these events are regular events, not initialization events.

Initialization Example

The following Core Cell Structure demonstrates the effects of the initialization process (**the Structure has a bug, which is embedded there for the demonstration purposes**):



A (faulty) event counter Core Cell.

At the first glance, this Structure should implement an event counter (notice the integer mode of the Modules used in the Structure). However, this Structure has a bug. If no Primary events have ever been sent to the Core Cell's input, the output value of the Core Cell (and the value stored internally in the OBC memory of the counter) is still one rather than zero. Furthermore, even though there is no incoming initialization event on the Primary side, the event counter Core Cell sends an initialization event (of value 1) from its Primary level output.

The reason for this are the details of the Core Structure initialization which runs as follows:

1. The shared OBC storage of the *Read*/*Write* Module pair and the output port states are zeroed.
2. The initialization event is sent from the constant "1" Module. Assuming there is no incoming initialization event from the Primary level at the Core Cell's input, there are no other initialization event sources in the Structure.
3. In response to the incoming event from the constant Module the adder computes the sum of the input values, where the lower input value is obtained from the just zeroed output of the *Read* Module. Thus, the adder outputs a value of 1.
4. The value of the sum is written into the OBC storage and is sent to the Core Cell's output.

Such problems can be avoided by using latches (discussed in section ↑4.5, Clocks and Latches) and/or Modulation Macros (discussed in section ↑4.6, Modulation Macros).

4.5 Clocks and Latches

A factory library *Latch* Macro is available under Library > Memory > Latch:



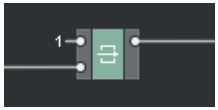
A Latch Macro.

> An integer version of the latch is available in the same menu under the name ILatch.

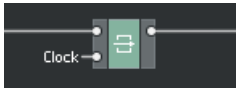
The upper input of the *Latch* Macro is the value input. The lower input of the *Latch* Macro is the event or the **clock** input. The output combines both into one event where the value is taken from the upper input and the clock or the triggering is taken from the lower input.

The *Latch* Macro can be used for two different purposes:

- Replacing a value of an event with another value:



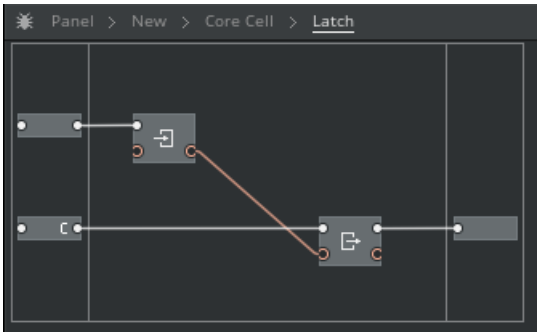
- Delaying a sent value until a certain event (clock) occurs:



Both purposes simply represent two different viewpoints on one and the same process.

> The second usage purpose (delaying a value) is the reason for the *Latch* Macro's icon and for the alignment of the *Latch* Macro's ports.

The internal Structure of the *Latch* Macro is very simple:



The internal Structure of the Latch Macro.

In this Structure it is imperative that the *Write* Module precedes the *Read* Module in the OBC chain. This ensures that if the value and the clock events arrive simultaneously at the *Latch* Macro's inputs, the value will be used to generate the output event.

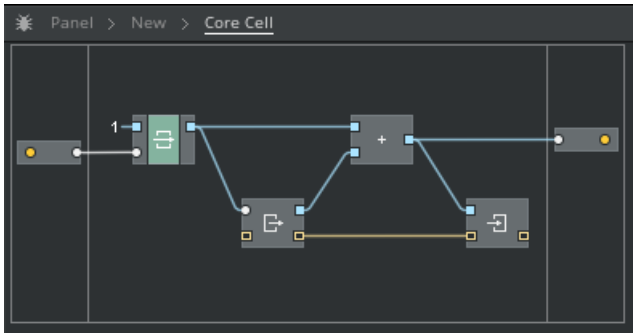
> The opposite ordering (the *Read* Module preceding the *Write* Module) is used to build the one-sample delay (z^{-1}) Macro discussed in section ↑4.10.4, Resolution Mechanism.

The compiler recognizes the '*Write* Module followed by a *Read*' Module OBC pattern and treats it in special, optimized way. Depending on the circumstances, using a *Latch* Macro at a given position in a Structure will produce more efficient code than without using it. As a general rule, the Core Structures should employ latches where logically appropriate without attempting to minimize their amount.

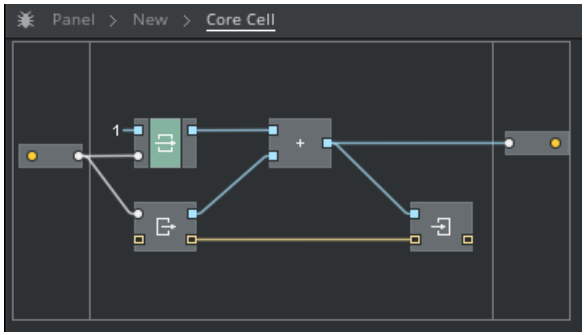
Usage Examples

The faulty event counter discussed in section ↑4.4, Initialization could have been fixed in one of the two following ways using the latches.

- Use an integer event accumulator Structure (similar to the float event accumulator discussed in section ↑4.3, Object Bus Connections (OBC)) and use an integer latch (*ILatch*) Macro to replace the values of the incoming events with 1:



- Start with the faulty Structure discussed in section ↑4.4, Initialization and use a *Latch* Macro to make sure that the output event of the constant Module does not trigger the addition:



Notably, the two Structures are pretty much identical, differing only by the specific pickup points of the clock signals for the *Latch* and the *Read* Module.

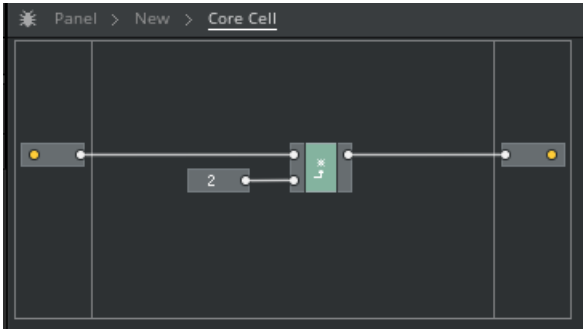
> The clock inputs completely ignore the incoming signal's value. The Core compiler is aware of that and drops out the respective computations in the incoming signal's path if these values are not needed otherwise. Particularly, the compiler will drop a value conversion from int to float if an int signal is used at a float clock input. For that reason it does not matter whether a clock input is configured to int or float type. The convention used in Core however is that all clocks are float.

The most appropriate way to fix the event counter is by using Modulation Macros, discussed in section ↑4.6, Modulation Macros.

4.6 Modulation Macros

The basic math operation Modules follow the standard Core triggering convention: any input triggers an output event. Sometimes a different convention is desired, where some inputs should not trigger the output. This convention is implemented in the Modulation Macros, found under Library > Math Mod (for Math Modulation).

The following alternative implementation of the event doubler Core Cell from section ↑4.2, Processing Order illustrates the concept of Modulation Macros:



An alternative implementation of the event doubler Core Cell using a modulation multiplier.

The Macro in the middle is the modulation multiplier, available under Library > Math Mod > x mul a. If one of the usual built-in multipliers had been used instead, the Structure would have had the same bug (for exactly the same reason) as the event counter in section ↑4.4, Initialization. With the modulation multiplier, the constant Module does not trigger the addition, thus the Structure does not have a bug.

The name 'modulation multiplier' comes from the use case where the amplitude of an event stream is being modulated by some other 'gain' signal. Clearly, in this case it is not desired that the events of the 'gain' signal trigger the multiplication:

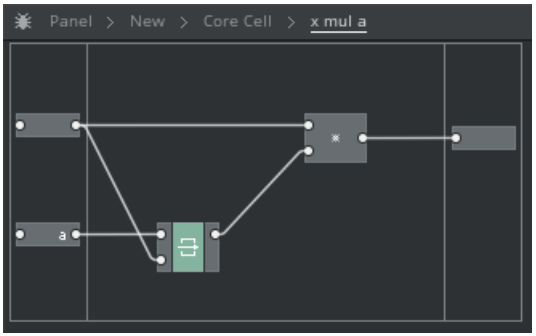


Modulation of the amplitude of an event stream

This use case is also the reason for the icon and the port alignment of the Macro. The arrow in the icon corresponds to the modulation signal input. Additionally, in the name of the Macro and in the labels of the ports the modulation inputs are denoted with the letters from the beginning of the Latin alphabet ("a", "b", ..).

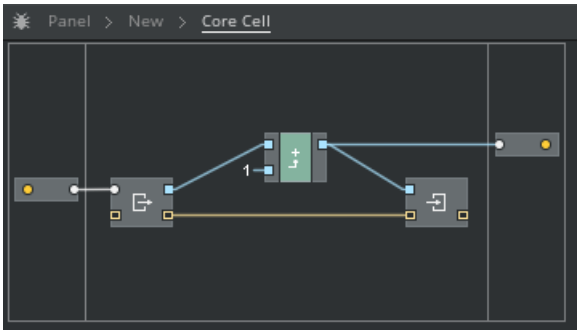
Occasionally the opposite to the modulation concept is used: the *tapping*. While the modulation inputs are inputs that are explicitly declared as non-triggering, a tapping input is explicitly declared as triggering (whereas the remaining inputs are assumed non-triggering by default).

The internal Structure of the modulation multiplier is using a *Latch* Macro to prevent the lower input from triggering the computation:



The internal Structure of the modulation multiplier.

Another good illustration of Modulation Macro usage is yet another way of fixing the event counter of section ↑4.4, Initialization, besides the two bug-fixing approaches discussed in section ↑4.5, Clocks and Latches. The fix is using an integer modulation adder, which is available under Library > Math Mod > Integer > Ix + a:



A bug-fixed event counter using a modulation adder.

4.7 Arrays

REAKTOR Core supports arrays as a special kind of OBC storage. While ordinary OBC storage is implicitly defined by the mere fact of using it (by reading or writing), the arrays need to be explicitly defined by placing an array Module into the structure. There are two array flavors:

- Writable array (available under Built-In Module > Memory > Array): The number of array elements is specified in the Properties. As with all writable memory, writable arrays are zero-initialized upon initialization.



- Read-only array or *Table* (available under Built-In Module > Memory > Table): The table is supposed to be prefilled by the Structure builder using the Table Editor available from the *Table* Module's Properties.

The array memory is allocated per voice. Thus an array of 1024 floats in a 16 voice setting needs a total memory for 16*1024=16384 floats, which, considering that each float needs 4 bytes of storage, is equal to 64 Kbytes.

The read-only tables are allocated just once and are not multiplied with the number of voices. Furthermore, multiple tables sharing exactly identical sets of data are internally combined into a single table. Thus having multiple copies of the same table Module containing a large table does not noticeably increase the memory usage.

The output port of the *Array* or *Table* Modules is of a special **array OBC** type (which is further subdivided into int and float types with variable float precision). Thus a *Read* or a *Write* Module cannot be directly connected to an array. In order to be able to read or write into an array, a single array element needs to be selected first for access.

A Macro port of the Latch class can be configured to the array connection mode in the Properties.

4.7.1 Array Indexing

The array element selection is performed by using the *Index* Module (Built-In Module > Memory > Index):



An Index Module.

- The lower input of the *Index* Module is of the array OBC type and is supposed to be connected (directly or via some Macro ports and other connectivity features) to the array to be accessed.
- The output of the *Index* Module is of the usual (latch) OBC type and is supposed to be connected, directly or indirectly, to one or more *Read* and/or *Write* Modules.
- The upper input of the *Index* Module is of scalar int type. An incoming event at this input causes a selection of an array element of the corresponding index for the OBC output of the Index Module.

The indexing is zero-based. That is the valid index range is from 0 to N-1, where N is the array size.

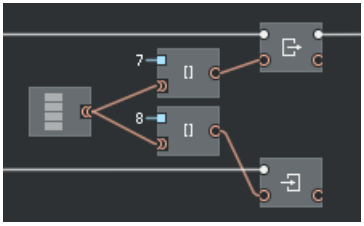
The result of accessing the array elements out of the valid index range is unspecified, except that such accesses should not cause memory corruption or crash REAKTOR.

In the current implementation of the Core compiler the safety of the out of range index accesses is ensured by internally clipping the index value (in the unsigned integer mode). However, as with other unspecified behavior, the specific details of this safety mechanism should be never relied upon. Correctly built REAKTOR Core Structures should never use out of range index values.

Sometimes it is convenient to be able to automatically determine the size of an array. This function is provided by the Built-In Module > Memory > Size [] Module. The Module outputs the total number of the array elements.



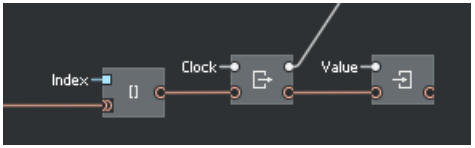
The *Index* Module rarely should be used directly. Typically *Read []* and *Write []* Macros should be used instead. For the sake of illustration of the Index Module's functionality the following two cases are presented:



Usage of raw Index Modules to access array elements (unrecommended).

In the above picture the two *Index* Modules are used to access the array elements with the indices 7 and 8 respectively, where the 7th element is accessed for reading and the 8th element for writing.

In principle, it is possible to attach a *Read* and a *Write* Module to the same *Index* Module, e.g. in series. In this case they are obviously accessing the same array element.



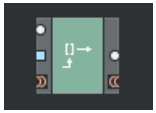
A Read and a Write Module sharing the same Index Module (untypical and unrecommended).

Neither of the two cases is a recommended way to access the array.

4.7.2 Read [] and Write [] Macros

The array accesses typically should be done by using the *Read []* and *Write []* Macros available under Library > Memory.

- The *Read []* Macro has a clock input, an index input and an array OBC input. The clock input triggers the array reading operation at the specified index, sending the read value at the upper output of the Macro.

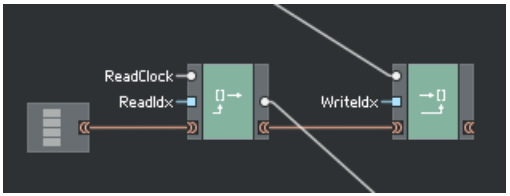


- The *Write []* Macro has a value input, an index input and an array OBC input. An incoming value at the write input causes a write operation at the respective index.



It would have been possible to implement the *Read []* Module without the clock input, by using the event at the index input as the triggering signal for the reading operation. Besides being inconsistent with the *Write []* Macro this would have caused unnecessary CPU overhead if the reading is performed without an actual change of the index. With the index input being separate, the (rather small) CPU overhead associated with the indexing is occurring only in response to the events at the index input.

The array OBC inputs and outputs of the *Read []* and *Write []* Macros are supposed to be used to order the array accesses relatively to each other:



An example of the ordering of read and write accesses to an array.

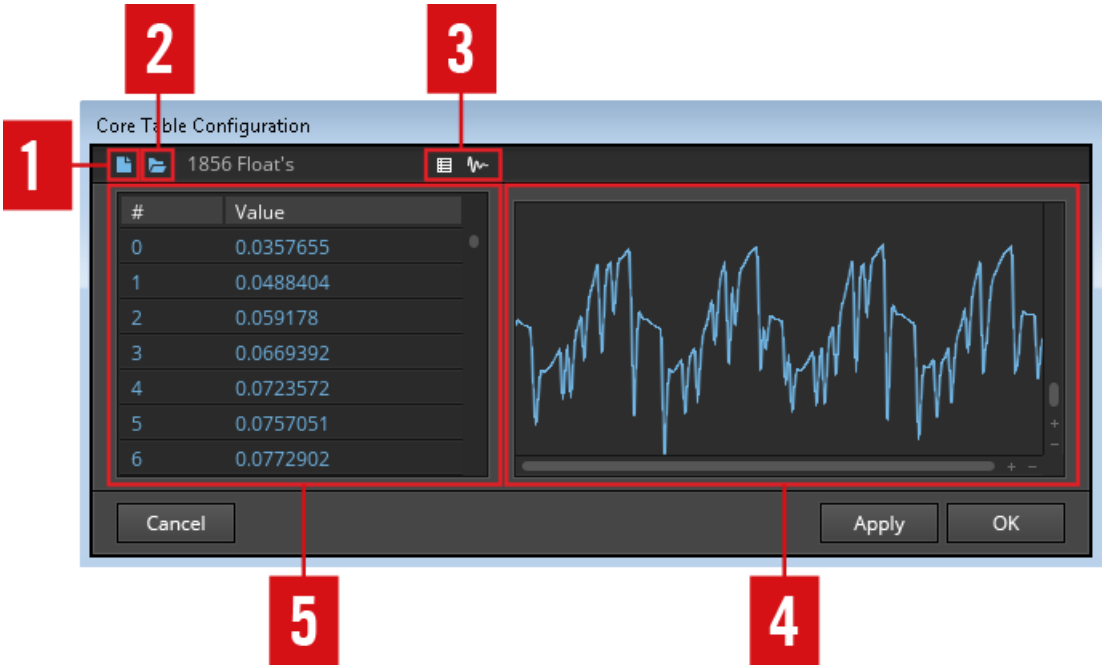
4.7.3 Table Specifics

The Table Module works pretty much as an usual array, except that writing into the table is not possible. The tables also have specific properties.

- FP Precision: In case the table contains float data, this configures the precision of the outgoing float OBC connection type (and thus the precision of the compatible *Read* accesses). The precision of values internally stored in the table is not affected by this setting.

  The *Read []* and *Write []* Macros can be connected to a table with a non-default FP precision by changing the FP precision setting of the *Read []* and *Write []* Macros.

- Table Editor: Calls up the table editor, which allows to fill the table with values.



The Table editor.

(**1**) **Create/Resize Table:** This button allows you to configure the table's data type and size and to pre-fill it with a specified initial value.

(**2**) **Import Table Data**: This button allows you to load the table from a file. A number of audio and text file formats are supported, where the latter can be used to import special numeric tables prepared in other software.

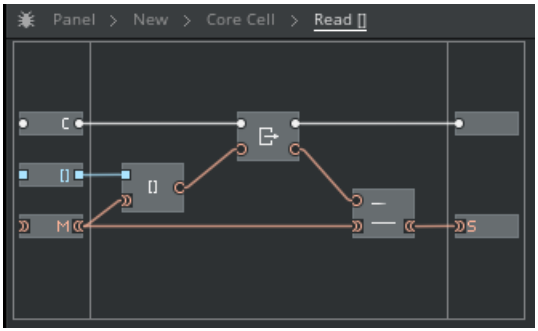
(**3**) **Layout**: These buttons turn the numeric display and the wave display on or off.

(**4**) **Wave display**: The wave display displays the contents of the table as a wave.

(**5**) **Numeric display**: The numeric display can also be used to edit the individual values in the table.

4.7.4 Advanced OBC Ordering

Some advanced OBC ordering topics can be nicely illustrated by the internal Structure of the *Read []* Macro:

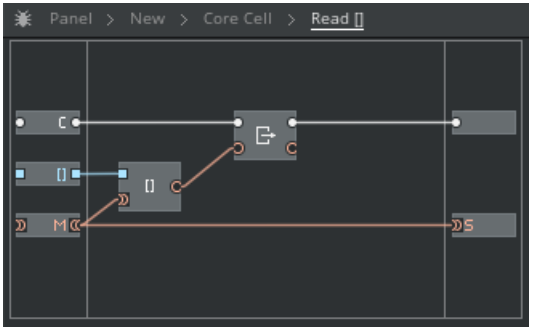


The internal Structure of the Read [] Macro.

The function of the *Index* and *Read* Modules is clear. The third Module (the one next to the OBC output of the Macro) is the *R/W Order* Module (available under Built-In Module > Memory > R/W Order).

> The *R/W Order* Module is also used inside the *Write []* Macro at the similar position.

Upon the first look this Module might seem to be completely redundant, because the *Read []* Macro could have been implemented without it:



A faulty implementation of the Read [] Macro without R/W Order.

However, in this case the OBC output of the Macro is not ordered relatively to the *Read* Module inside the Macro. This means that Modules and Macros (such as e.g. a *Write []* Macro) connected to this OBC output will not be ordered relative to the reading operation.

The *R/W Order* Module inside the *Read []* Macro does nothing more than providing a formal ordering between the *Read* Module and the OBC output of the Macro, due to the mere fact that the *Read* Module is connected to its upper input. The lower input of the *R/W Order* Module simply forwards the incoming OBC connection to the output.

The *R/W Order* Module can also be used for explicit ordering of non-array OBC connections, although there such need is much less likely to occur.

The need for the explicit ordering by the *R/W Order* Module is not affected by the setting of the Solid property of the containing Macro. This might be improved in future versions of REAKTOR Core (without loss of compatibility with older Structures).

4.8 Routing and Merging

The *Router* Module (Built-In Module > Flow > Router) is used to direct the event flow to one of two alternative paths, where the event flow direction can be switched at runtime:



A Router Module.

Depending on the router state (controlled by the upper input) the events arriving at the lower input are directed either to the upper output or to the lower output. The upper input (the control input of the router) is of BoolCtl type and can be in the **true** or **false** state.

- If the control input is in the **true** state then the events are routed to the upper output.
- If the control input is in the **false** state then the evens are routed to the lower output.

The BoolCtl connection does not transmit any events. It only transmits the state but no triggering information. Therefore it is not a signal in the same sense as the scalar type class.

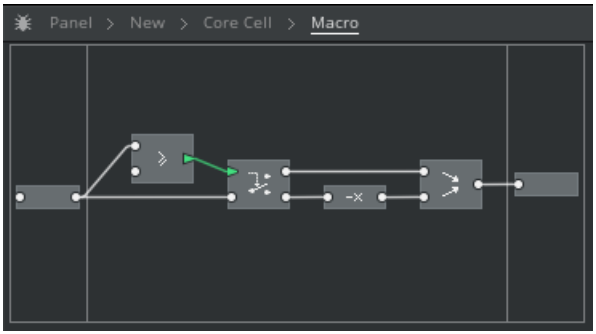
The BoolCtl sources are provided by *Compare*, *Compare Sign* or *ES Ctl* Modules in the Built-In Module > Flow menu.

The comparison criterion of the *Compare* and *Compare Sign* Modules can be adjusted in their Properties.

The *Compare Sign* Module is useful for detecting zero (and other thresholds) crossings.

There are no Boolean operation built-in Modules to combine different BoolCtl connections together. Use the Macros from Library > Logic (at the cost of a somewhat higher CPU load) if you need standard Boolean logical operations.

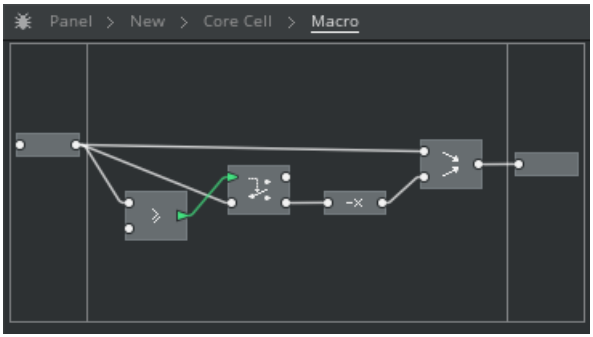
A signal split by a *Router* Module can be merged back by the Built-In Module > Flow > Merge:



Merging the two paths back. The Macro Structure implements the absolute value functionality.

The *Merge* Module simply lets all incoming events go through to the output. If the events arrive simultaneously at both inputs, the lower input has priority.

Instead of merging the two paths of the split signal, another common pattern is to merge back with the original signal:



Merging back to the original signal. The Macro Structure implements the absolute value functionality.

Since the lower input of the *Merge* Module has higher priority, in case the signal goes through the inversion path, the inverted signal at the lower input of the *Merge* Module has priority over the original (unrouted) non-inverted signal.

The *Merge* Module does not need to be used exclusively in combination with routers. One can merge arbitrary events.

In the current implementation of the REAKTOR Core technology this kind of 'chaotic' merging (as compared to merging the split paths back together or to their original source) might overload the compiler. Therefore 'chaotic merging' needs to be used with some care. In cases of large amounts of chaotic merging, the compilation time may grow excessively. Event reclocking by using latches and Modulation Macros often helps (see section ↑5.6.3, Routing and Merging).

The math processing Module such as the built-in adder, multiplier etc. technically perform event merging. The difference to the *Merge* Module is solely in how the output value is computed. Therefore the math processing Structures are subject to the same 'chaotic merging' considerations.

The *Merge* can be made to have more than 2 inputs by configuring the number of inputs in its Properties. The priority rule for simultaneous events is the same: the priority is highest at the lowest input. Intuitively this can be imagined as the events being locally ordered from the top input

to the bottom one, so that the lowest event is the one which arrives latest of all and thus overrides the value prepared for the output.



4.9 SR and CR Buses

A number of signal processing Modules need to be clocked by some regular trigger sources. The typical examples are oscillators, filters, envelopes and LFOs. REAKTOR Core provides default clock sources for such Modules. There are two sources provided in the form of Bundle Scoped Buses with two different names:

- **SR** (for Sampling Rate): The bus with this name is the source of the audio clock.
- **CR** (for Control Rate): The bus with this name is the source of the control clock.

The SR bus is intended to be used for audio processing Modules like oscillators and filters, while the CR bus is intended to be used for control processing Modules like envelopes and LFOs. However, this SR/CR distinction is purely conventional and is made with the sole purpose of being able to provide different audio and control clocks 'by default'. There is always a possibility of overriding the default connection and/or providing even more clocks by defining further buses with different names and the same Structure.

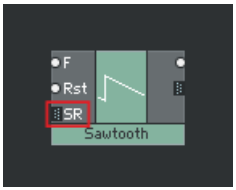
The default CR source in REAKTOR Core is identical to SR signal-wise, runs at the audio rate and has nothing to do with the Control Rate source in the REAKTOR Primary. If desired, the Primary Control Rate can be 'imported' into the CR bus by using the Library > Clk Bundle > Control > CR From Prim Macro.

Both SR and CR bus Bundles have identical internal Structure and theoretically can be used in place of each other. Practically this is usually not recommended. Particularly, the conversion between different clock rates needs to be different for audio and control signals.

The default built-in SR and CR buses are provided 'one level higher' than the top Structure of the Core Cell, thereby allowing to redefine them already within this top-level Structure without causing a definition conflict. If such redefinition is done, the 'previous' (default) bus has to be picked up using the parent-mode scoped accessors, as usually. See ↑4.9.4, CR rate change and ↑4.9.5, SR Rate Change for the details of redefining SR/CR.

4.9.1 Connections to the Clock Buses

A Core Macro Library Module that needs an SR or a CR connection has by convention a Bundle input, labelled SR or CR respectively:



The SR input of an oscillator.

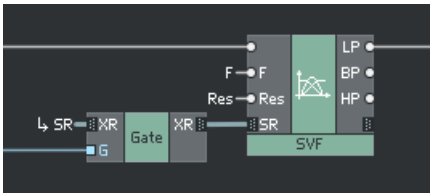
The SR/CR input convention is not used for the low-level library Macros. Particularly, the *z^-1 ffd* and *z^-1 fbk* Macros connect themselves to the SR bus, even though they do not have an SR input.

If the SR or CR input is not connected, it automatically picks up the SR or CR bus. In the case of the above oscillator, in the default unconnected state of the SR input, this input will be automatically picking up the SR bus. Should this default connection be undesired, some other clock Bundle source needs to be manually connected to this input. This automatic connection convention is employed for effectively all signal generating and processing Modules in the Core Macro Library.

4.9.2 Clock Gating

Sometimes it is desired to turn off audio or control processing within a given part of the Structure. This can be achieved by **gating** the clock.

In the following Structure the audio clock is gated for a 2-pole SVF filter:



SR clock gating.

The *XR Gate* Macro (Library > Clk Bundle > XR Gate) on the left interrupts the SR clock as long as the gate signal G is zero. As soon as G is turned to 1 the clock begins to run.

> Upon turning from 0 to 1, the *XR Gate* sends a reset signal (see section ↑4.9.6, Internal Structure of Clock Buses) along the clock Bundle, causing the connected Modules to reinitialize their internal states if necessary. Should the reset signal be undesired in this case, use an *XR Freeze* Macro instead.

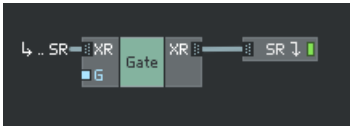
The "XR" stands for either "SR" or "CR". That is, the XR Macros are equally usable for both SR and CR Bundles. Some other clock Bundle Macros are usable only for SR or only for CR Bundles. Those Macros are located under Library > Clk Bundle > Audio and Library > Clk Bundle > Control respectively.

The *XR Gate* Macro only gates the clock, but does not cause the outputs of the connected audio- and control-processing Macros, such as the LP output of the filter in the above picture, to be zeroed (this is done for the CPU efficiency reasons, associated with the event merging issues discussed in section ↑5.6.3, Routing and Merging). It is the builder's responsibility to avoid picking up the 'frozen' output values by some other running parts of the Structure downstream.

The automatic default connection of SR/CR input is implemented only for the signal generating and processing Modules like oscillators, filters, envelopes, LFOs, etc. The Modules dedicated to the handling of SR/CR bus themselves (such as clock gates, dividers, etc. (located in Library > Clk Bundle > Control) always need to be connected explicitly.

4.9.3 SR and CR redefinition

If a whole area of the Structure is being controlled by a gated clock, then instead of wiring up the output of the *XR Gate* Macro to each of the clocked Modules, it is convenient to put all such gated Modules into one Macro Structure and locally redefine the clock bus within this Structure:

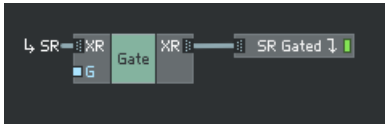


Redefining the SR bus.

Notice the parent-mode accessor usage at the input of the *XR Gate*, which ensures that the higher-level SR bus, rather than the one defined on this level, will be picked up.

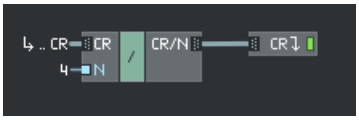
Another version of the same approach is discussed in section ↑5.2, Generation of Audio and Other Regularly Clocked Events.

Some kind of an 'in-between' approach is to define a Scoped Bus (or a QuickBus) with a different name and explicitly connect to this bus. Notice that there is no need for the parent-mode accessor at the *XR Gate*'s input in this case.



4.9.4 CR rate change

In certain advanced Structure building scenarios one control rate is insufficient. Or the default control rate, which is equal to the audio rate, may be considered too high for the builder's purposes. In this case the CR rate division may be employed to generate lower control rates. E.g. the following Structure fragment employs the *CR Div* Macro (Library > Clk Bundle > Control > CR Div) to locally redefine the CR bus to a 4 times lower rate:

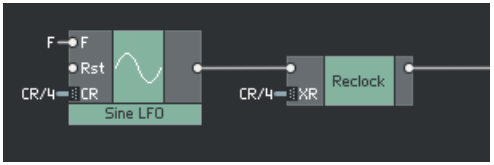


Redefining the CR bus to a 4 times lower rate.

Of course, the other discussed possibilities to connect to the new CR (including explicit wiring, QuickBuses and differently named Scoped Buses) can be used as well.

Smoothing

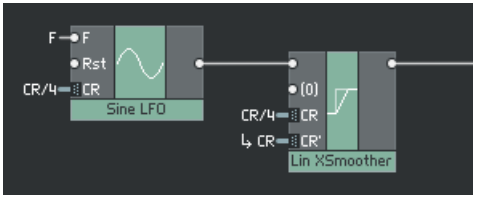
Converting a signal from a higher control rate to a lower control rate can be done by the *XR Reclock* Macro (Library > Clk Bundle > XR Reclock):



Reclocking control signal to a lower rate.

> In principle the *XR Reclock* Macro can be also used for control signal upclocking or audio up- or downclocking. Usually, however, this will create undesired artifacts that can be avoided by using other means instead.

Conversion from a lower control rate to a higher control rate is achieved by **smoothing**. A linear cross-control rate smoother is provided by the *CR Lin XSmoother* Macro (Library > Clk Bundle > Control > CR Lin XSmoother):



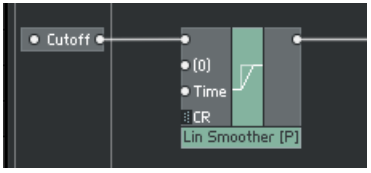
Smoothing from a lower control rate to a higher one.

The *CR Lin XSmoother* Macro is designed to be used by hard-locked control rates, the ratio of which is an integer number. E.g. in the above picture the CR/4 control rate is obtained by applying the CR Div Macro to the CR control rate. **If this condition is not met, the smoother will not work correctly.**

The "(0)" input of this and other smoothers is intended to provide an initial value for the smoother. If the input signal of a smoother is clocked, then according to the clocking conventions (see sections ↑4.9.6, Internal Structure of Clock Buses, ↑5.2, Generation of Audio and Other Regularly Clocked Events and ↑5.3, Processing of Audio and Other Regularly Clocked Events) the input signal cannot arrive during the initialization or a Structure reset. Thus, it will be not possible to use the input signal to specify the initial state of the smoother. One way around it is to merge in a dedicated initial value signal using a *Merge* Module in front of the smoother's input. However this is a kind of hidden violation of the mentioned clocking convention. Instead, one should connect the initialization signal to the "(0)" input of the smoother. If the input signal is not following the clocking convention (like e.g. a signal which comes from a knob) and provides a reasonable initial value setting for the smoother, then the "(0)" input can be left disconnected. The smoother will use the input signal as the initial value source in this case.

Irregular Permanent Smoothing

Smoothing can also be applied to convert an irregularly clocked signal (or a signal which comes completely sporadically, such as a stream of values from a panel knob) to remove the steps in the signal. Such smoothing will produce a stream of values clocked at some control rate. Depending on the specifics of the particular smoother, however, the signal events might not be sent all the time, but only until the destination value is reached by the smoother. E.g. in the following Structure the "Cutoff" knob's value at the Macro's input is being smoothed by a *Lin Smoother [P]* Macro (Library > Control > Smoother > Lin Smoother [P]):



Permanent smoothing of an irregular control event stream.

The Library > Control > Smoother > Lin Smoother [P] Macro is not considered a CR-handling Macro, since it does not modify the CR Bundle or explicitly convert from one rate to another. For that reason it is located under Library > Control rather than Library > Clk Bundle > Control. For the same reason it automatically connects to the CR scope bus.

The *Lin Smoother P* Macro generates a permanent stream of output events at a rate defined by CR bus ("P" for Permanent). Whenever there is a new event coming from the "Cutoff" input port Module, the smoother generates a ramp of duration specified by the Time input (in seconds) from the old value to the new value.

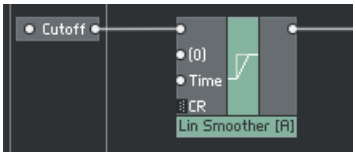
If the Time input is not connected, there is some preconfigured default. The "(0)" input has the same function as for *CR Lin XSmoother*.

Irregular Automatic Smoothing

Imagine the control signal processing Structure downstream from a smoother. The processing in this Structure may follow different conventions, such as the following:

- The control signal processing Structure is clocked by the CR clock.
- The control signal processing Structure is unclocked; the processing is triggered by the incoming control signal events.

In the latter case it might be undesirable that the smoother is generating the output events all the time, thereby causing the control signal processing to run all the time as well. Rather, the smoother should only generate the output events until the target value is reached and then stop. This behavior is implemented by the *Lin Smoother [A]* Macro (Library > Control > Smoother > Lin Smoother [A], "A" for Automatic), which differs from *Lin Smoother [P]* only in exactly this aspect:

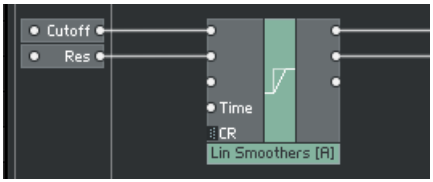


Automatic smoothing of an irregular control event stream.

Irregular Batch Smoothing

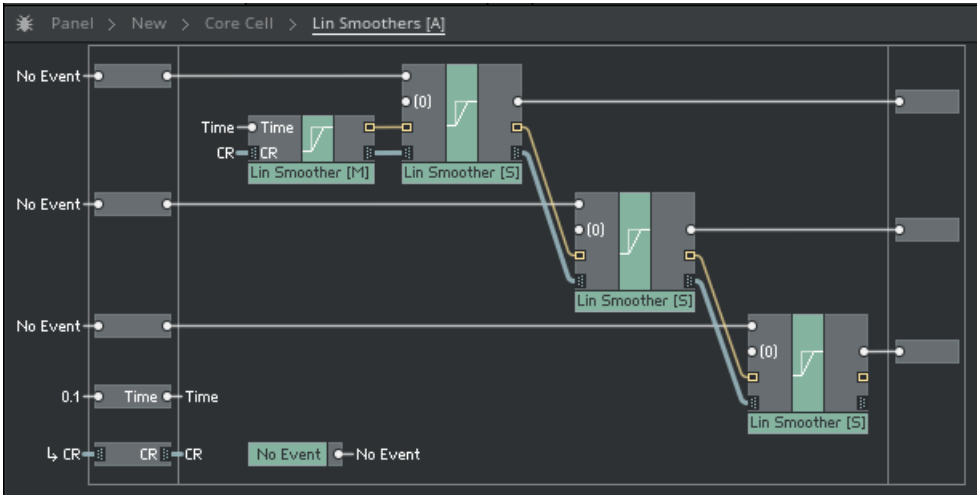
The automatic stop option however still requires a runtime check on every control clock in order to determine whether the smoother is active or not. Such runtime check also consumes CPU time. Therefore it is a good idea, if one has a group of several of such smoothers, to group them together, performing one single runtime check for the entire group.

This approach is implemented by the *Lin Smoothers [A]* Macro (Library > Control > Smoother > Lin Smoothers [A]):



Batch smoothing of two irregular control event streams.

Internally the *Lin Smoothers [A]* Macro contains a 'master' smoother (containing the master check) and a chain of 'slave' smoothers:



The internal Structure of Lin Smoothers [A] Macro.

The slave smoothers are the ones doing the smoothing itself and can be added or removed by the user of the Macro according to the number of signals which need to be smoothed. The "No Event" default for the Macro inputs prevents the unused inputs from initiating a smoothing ramp upon reset. The maximum number of slave smoothers supported by the current implementation is 32, which should be sufficient for most purposes.

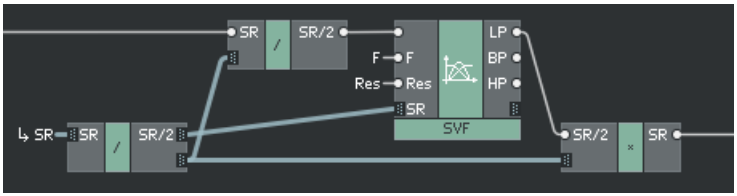
> Sticking too many smoothers together in this fashion will produce some small overhead, due to the fact that the per-smoother runtime checks will be performed for all smoothers as long as at least one of the smoothers is active.

4.9.5 SR Rate Change

A conversion between audio signals of different rates is done differently from control signals. While control signals can be naively downsampled and smoothed upon upsampling, the audio signal resampling requires special kinds filtering techniques.

> In the current version of REAKTOR Core the clock rates can be only lowered, not raised. Thus, oversampling is currently not possible. The rate conversion Macros are of course available in both directions: downsampling (used upon entering the downsampled area) and upsampling (used upon leaving the downsampled area).

Thus, a set of library Macros provided for the SR rate change is different from those for CR:



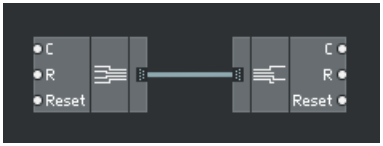
A locally downsampled SVF (State-Variable Filter)

The *SRdiv2 Master* Macro (Library > Clk Bundle > Audio > SRdiv2 Master) on the very left is the downsampling master, which is driving the downsampling converter *SRdiv2 In* on the left and the upsampling converter *SRdiv2 Out* on the right. Thus the filter in between is running at halved sampling rate. You can drive multiple *SRdiv2 In* and *SRdiv2 Out* converters with the same master.

Often it might be reasonable to put the entire downsampled area inside a dedicated Macro. In this case it is useful to locally redefine the SR bus to the downclocked one using the same trick as it was explained for the CR bus in ↑4.9.4, CR rate change. Note that you still need to explicitly resample all incoming and outgoing audio signals of the downsampled area by using SRdiv2 In and SRdiv2 Out Macros.

4.9.6 Internal Structure of Clock Buses

A clock bus is a Bundle containing three different fibers named "C", "R" and "Reset". For the convenience of the builders the Macro library contains preconfigured Bundle *Pack* and *Unpack* Modules for the SR and CR buses under Library > Clk Bundle > Audio > XR Pack and XR Unpack:



Preconfigured Pack and Unpack Modules for the SR/CR Bundles from the library

The details of each of the fibers are described below. The examples of the low-level usage of the clock buses can be found in sections ↑5.2, Generation of Audio and Other Regularly Clocked Events and ↑5.3, Processing of Audio and Other Regularly Clocked Events.

"C" Fiber

This is the clock fiber which sends regular events at the clock rate. The audio processing Macros are supposed to send the output audio events only in response to the clock events. Same holds for regularly clocked control events.

The convention requires that no event is sent on the C fiber during the initialization. This particularly means that the audio signal outputs and clocked control outputs of Core Macros are not sending during the initialization.

The value of the C fiber is unspecified and should not be relied upon.

"R" Fiber

This fiber transmits the clock rate in Hz.



The convention requires that the R event is sent during the initialization. Afterwards it is sent as needed.

The reason for this convention is to simplify the event processing in the use cases of the R fiber. Since the R fiber is guaranteed to send the initialization event, it can be safely used as a divisor in the computation of oscillator increment steps, normalized filter cutoffs, and so on.

The events on the R fiber may cause additional computation overhead in the consumer Structures, as they are reconfiguring themselves to a new rate, and should not be sent indiscriminately, but rather only when the rate changes or is likely to change (filtering out the duplicate values sent on this fiber is probably a little excessive, while sending events here at a regular rate is a highly questionable practice).

"Reset" Fiber

This fiber transmits a zero-value reset event. The consumers of the SR Bundle are supposed to reset their state (if necessary) in response to this event. E.g. a filter should reset its state, while a free-running oscillator does not have to do anything in response to this event.



The convention requires that the Reset event is sent during the initialization. Afterwards it is sent as needed.



The convention requires that the Reset event always has a zero value.



The convention specifies the Reset event semantics as 'starting again without the past'. E.g. a Reset event can be sent upon resuming a paused clock to cause the Structures (such as e.g. filters) to reinitialize their states.

The reason for sending the Reset event during the initialization is to simplify the initialization of the Structures, which otherwise might need to take special measures to make sure that the same initialization steps are performed during the initialization and during further resets. The reason for the zero value of the Reset event is that a typical response to the Reset event is to zero the states, therefore the Reset event can be directly stored to the state memory of the respective Structures.

4.10 Feedback Connections

4.10.1 Automatic Resolution

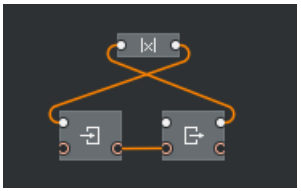
REAKTOR Core allows the usage of the feedback loops in the Structures, provided each feedback loop contains at least one scalar (float or int) wire. The loop will be handled by automatically placing an implicit one-sample audio delay somewhere within the loop,

'Audio' means that this implicitly inserted delay will be clocked by the SR.C clock obtained from the current scope. The delay also responds to SR.Reset by zeroing its memory.



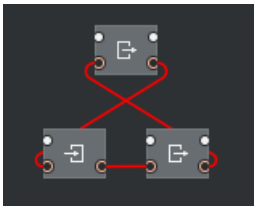
It is not guaranteed that REAKTOR Core will use at maximum one implicit one-sample delay per detected feedback loop. In some cases it is possible that more delays will be inserted. For exact control over the feedback loop resolution, one needs to use *z^-1 fbk* Macros explicitly (see section ↑4.10.2, Manual Resolution).

REAKTOR Core highlights the feedback loop visually by turning the connection color to orange:



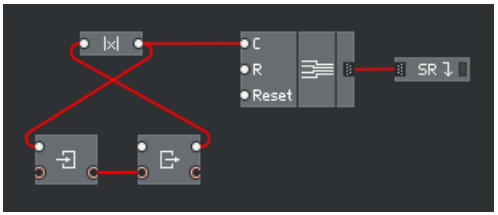
A highlighted feedback loop.

If the loop does not contain any scalar wires then it is an error, which is highlighted in red:



A purely non-scalar feedback loop.

Equally, if an SR bus definition is contained in or depends on a feedback loop, such loop is an error, even if it does contain scalar wires:

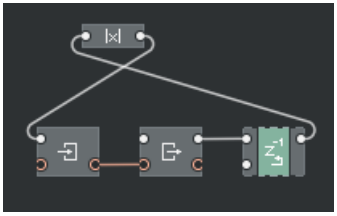


SR bus dependent on the feedback loop

> The reason for this is that the implicit audio delay used for the feedback loop resolution needs access to SR.C and SR.Reset, which creates another feedback loop after the resolution of the first one. See section ↑4.10.4, Resolution Mechanism for the details of the delay's Structure.

4.10.2 Manual Resolution

If exact control over the feedback loop resolution is needed, a *z^-1 fbk* Macro (Library > Memory > z^-1 fbk) has to be used. In this case the compiler does not highlight the loop anymore:



The use of the z^-1 fbk Macro to explicitly resolve the feedback loop.

The *z^-1 fbk* Macro implements the same one-sample delay Structure which is used by the automatic resolution. Differently from the automatic resolution, by using this Macro the builder can exactly specify the position of the one-sample delay.

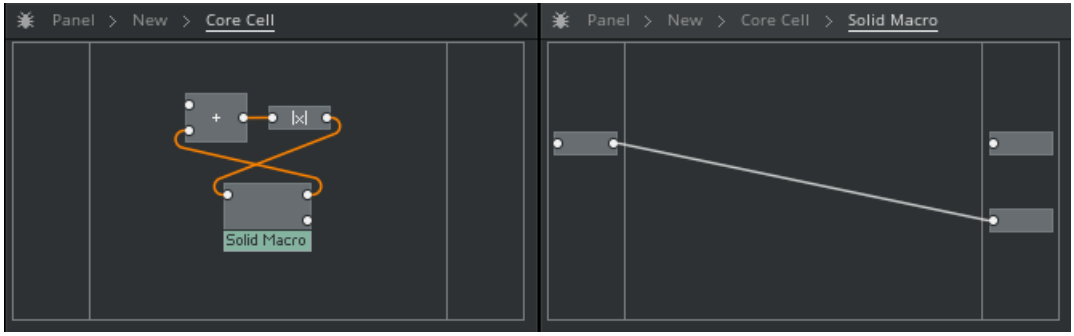
The *z^-1 ffd* Macro (Library > Memory > z^-1 ffd) is not intended and cannot be used for the feedback resolution. Its intention is implementation of feedforward 1-sample audio delays.

Neither of the above two Macros is supposed to be used for other purposes than audio rate 1-sample delays. For delaying other kinds of events by one tick use the *Latch[-1]* and *Event[-1]* Macros.

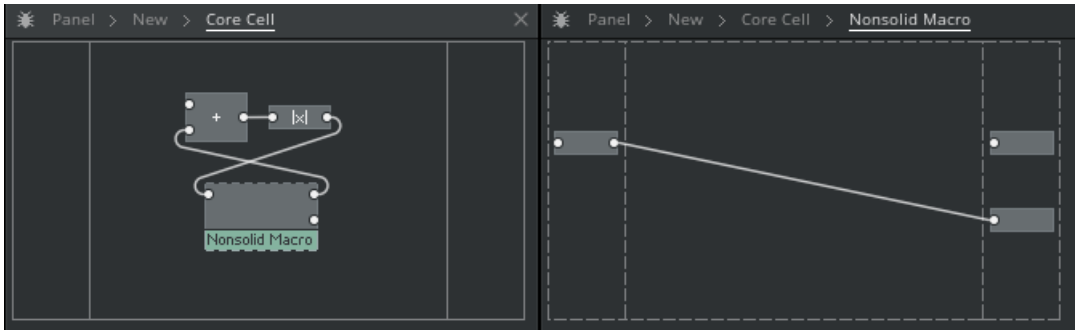
4.10.3 Nonsolid Macros and Feedback Loops

The Solid property setting of the Macro is affecting how the Macro is treated in respect to the feedback loops.

- **Solid Macros** 'cannot be entered' by feedback loops. This means that, regardless of the internal contents of the Macro, a feedback loop around this Macro is always a feedback loop. A Macro is treated as if it didn't have any internal Structure and was just a single solid block:



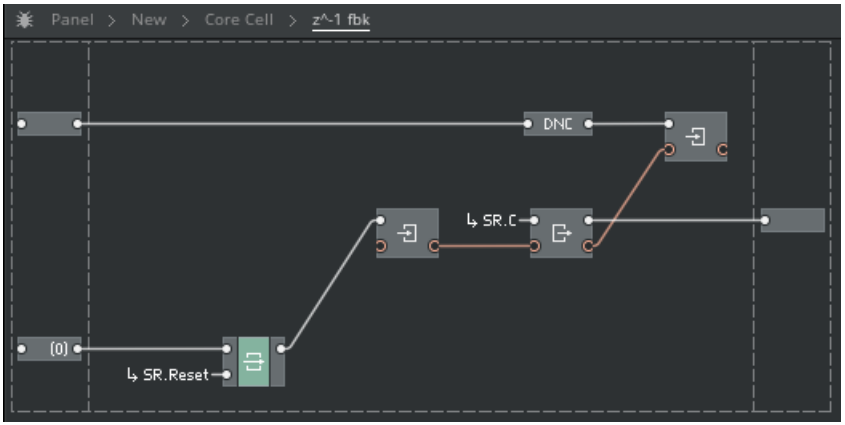
- The boundaries of **nonsolid Macros** are transparent to the feedback loops. If there is no *unidirectional* connection from an input to an output internally inside a nonsolid Macro, there will be no loop:



In particular, the *z^-1 fbk* Macro needs to be nonsolid in order to be able to resolve the feedback.

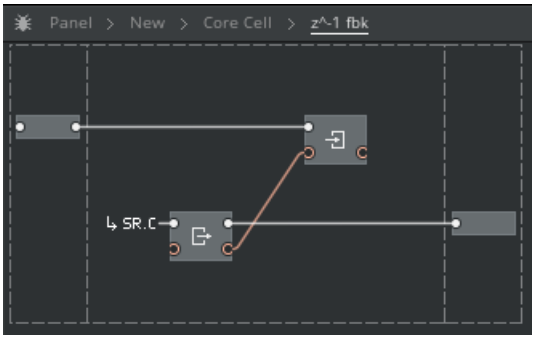
4.10.4 Resolution Mechanism

Internally the z^-1 fbk Macro loops like follows:



The internal Structure of the z^-1 fbk Macro.

The *Latch* and the *Write* Modules on the left are responsible for handling the SR.Reset, where the "(0)" input can be used to specify the value to reset to. The *DN Cancel* Module takes care of denormals (see section ↑5.5, Denormals and Other Bad Numbers). The essential part of the Structure is based on the remaining two Modules:



The essential part of the z^-1 fbk Macro's internal Structure.

Since the *z^-1 fbk* Macro is nonsolid, the following two Structures are equivalent:



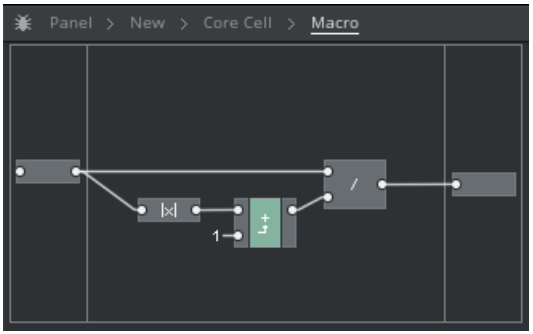
An equivalent expansion of the z^-1 fbk Macro (the nonessential parts of the z^-1 fbk's internal Structure are omitted).

So, formally from the compiler's perspective, there is no feedback loop at all (and thus no highlighting either).

5 Building Practices and Conventions

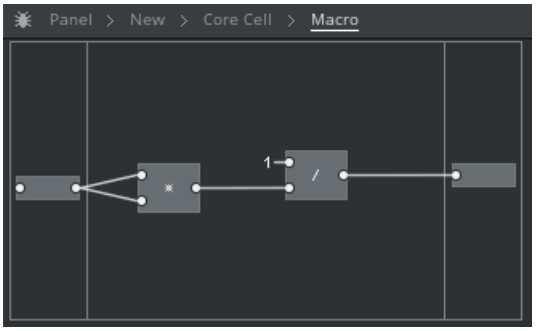
5.1 Expression Computation

The simultaneity of event processing in REAKTOR Core makes the computation of single-argument mathematical expressions quite straightforward. One thing to keep in mind is that, as a general rule, it is better to connect the constants to the Modulation Macros:



Computation of the x/(1+|x|) formula by a Core Macro.

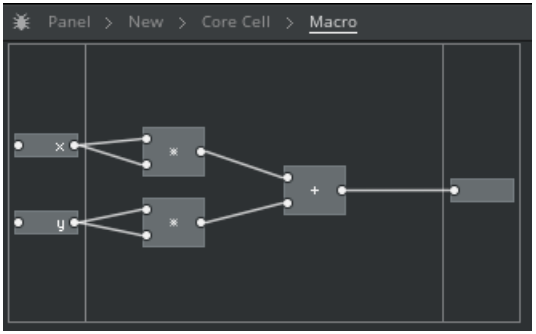
The reason for using Modulation Macros is clearly illustrated by the following Structure:



A questionable implementation of the 1/x^2 formula.

The problem is that if there is no input event during the initialization, the constant at the upper input of the divider will trigger the computation. Since no event has ever arrived at the lower input of the divider, though, the value at this input is zero. Therefore a division by zero will be performed, causing an *INF* value to be sent further downstream (see section ↑5.5, Denormals and Other Bad Numbers for further details on why this should rather be avoided) However, if it is certain that the initialization event will arrive at the lower input of the divider, the above Structure can be used.

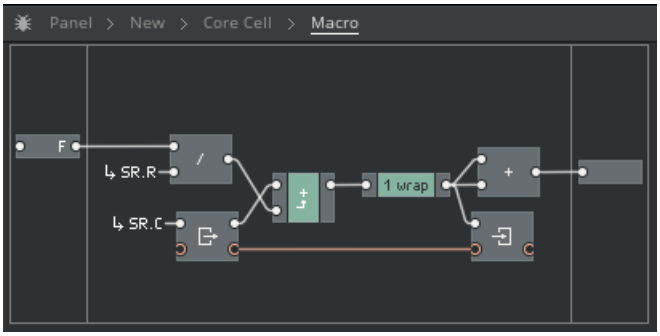
The computation of multiple argument expressions is fairly straightforward as well, except that one needs to watch for potential initialization issues.



Computing x^2+y^2.

5.2 Generation of Audio and Other Regularly Clocked Events

The audio-generating Structures typically should be internally clocked by the SR.C:



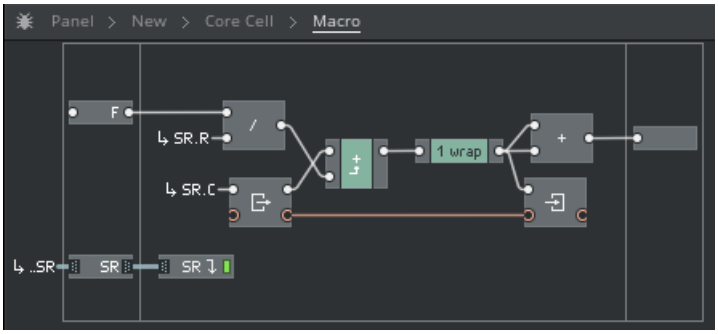
A naïve (non-antialiased) sawtooth generator Macro.

The *1 wrap* Macro used in the above Structure is found under Library > Math.

There is a general convention for the audio generating and processing Modules that they should perform their 'sample tick' computations and send output events **in and only in** response to the SR.C. The usage of the modulation adder in the above sawtooth oscillator ensures that no additional events (from the F input) will cause iteration triggering and/or sending of an output event.

The same convention applies for CR-clocked processing and is a good candidate for the application to other regularly clocked Structures.

Rather than binding the Macro to the usage of the SR bus available in the context, it might be useful to allow an SR Bundle to be connected explicitly to a dedicated input of the Macro. A useful pattern for this is illustrated in the following picture:



A Macro with a dedicated SR input port.

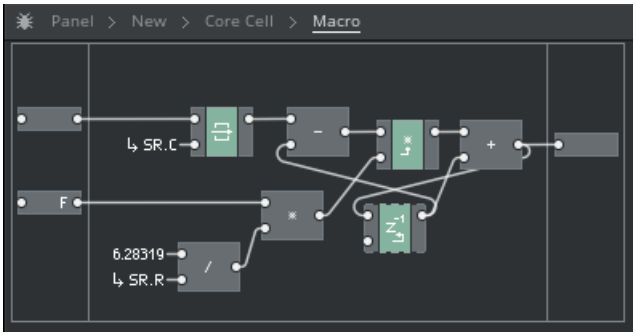
The Macro employs the Scoped Bus overriding technique to redefine the SR bus locally. If there is no connection at the SR input of the Macro, the redefined bus will simply be connected to the SR bus from the Macro's parent context, so the Macro will work exactly in the same way as if the SR bus was not redefined at all. However, if there is an explicit connection at the SR input of the Macro, the Macro will use the explicitly connected SR Bundle instead.

> An audio generating Structure might also need to perform some specific actions in response to the SR.Reset. For a free-running oscillator, however, the Reset event can be completely ignored.

The above trick involving the local redefinition of the SR Scoped Bus will not work correctly inside a non-solid Macro, since the bus redefinition will also affect the outside Structure. In this case the internal wiring of the SR.C, SR.R and SR.Reset needs to be done using normal wires and/or QuickBuses.

5.3 Processing of Audio and Other Regularly Clocked Events

The processing of audio and other regularly clocked events is similar to the generation. Besides the usage of Modulation Macros in places where the control signals are mixed with audio events, it is a good idea to reclock the audio input with SR.C just in case that a non-audio signal is connected to the input from the outside (one particularly common case of this is leaving an input disconnected, which implies a zero constant):



A naïve 1-pole lowpass filter.

Discrete-time difference LTI systems can be directly implemented in REAKTOR Core by simply wiring up the system's block diagram using the *z^-1 fbk* and *z^-1 ffd* Macros for z^{-1} blocks, modulation multipliers for gains, and adders for summation points. The above picture provides an example.

In the above Structure the *Latch* and the *z^-1 fbk* Macros serve as SR.C-clocked signal sources, which are then combined in various ways. The only other signal which is coming in (the normalized cutoff control signal derived from the input "F") is connected via a Modulation Macro. This ensures that no other clocks or trigger sources can accidentally generate an output event.

The same convention as for the audio generators is applied here: the output signal events must be produced **in and only in** response to SR.C. The same convention applies to CR.C and potentially to other regular clocks.

The SR.Reset fiber is automatically handled by the internal Structure of the *z^-1 fbk* Macro, so that the filter's state is zeroed in response.

In the above filter Structure the processing of the cutoff control input "F" is not done by the Modulation Macros. If the F signal is clocked by CR.C, the filter Structure might seem to violate the above guidelines of processing the clock signals, however in fact this does not.

The clocking guidelines apply to entire Library or framework Macros, not to some parts of their internal Structure.



Imagine a modulation multiplier had been used to combine F with the 6.28319/SR.R value. Furthermore, imagine there was change of the sampling rate without a change of the F input's value at the same time. In this case, the multiplier output would not have been updated, which would have clearly been a bug.

As to whether one should have used a modulation divider to compute 6.28319/SR.R, this does not matter since SR.R should be guaranteed to arrive during the initialization.



Had this guarantee not been in place, the divider could have produced an INF value at the initialization, which would most likely result in a NaN value stuck in the filter feedback loop (see section ↑5.5, Denormals and Other Bad Numbers).

5.4 Rounding

There is only one type of rounding available in Core by default: rounding to the nearest integer. The other rounding modes (rounding down, rounding up and rounding towards zero) are not provided. There are a number of technical reasons behind this, related to the precision and efficiency of the generated code.

While in principle the other rounding modes could be implemented as Core Macros, often the same Structures can be implemented just using the round-to-nearest mode, thereby not losing the efficiency. For example, an interpolated read from a buffer in a classical implementation rounds the non-integer position *x* down to an integer *n* and then performs a linear interpolation between the values at *n* and *n*+1, where the interpolation weights are 1-(*x*-*n*) and *x*-*n*. However, instead one could round (*x*-0.5) to the nearest integer. Since for values with a fractional part of exactly 0.5 the round-to-nearest direction is not specified, sometimes, for an exact integer *x*, the value *x*-0.5 will be rounded to *n*=*x*-1 rather than *n*=*x*. However, for the interpolation it does not matter, since the interpolation weights in this case will be 0 and 1 instead of 1 and 0, producing the same result.

One might need to ensure that the buffer has at least one 'extra sample' on each side of the region, which is read by the interpolator (this is usually a good idea regardless of the rounding mode used, since there can be precision losses in the computations). These extra samples can be set to an arbitrary value (within the same value range as the typical values in the buffer), since the interpolation weights for the extra samples will be very close to zero.



In situations where a division result needs to be rounded down, usually an integer division is meant semantically and this is what normally should be used as well. Note that the integer division rounding mode is 'towards zero'.

The reason that the integer division rounding mode is 'towards zero' rather than 'down' is that this is the mode supported by modern processors. Implementing a rounding down mode instead would produce a higher CPU load. However, there is no difference between the two modes for non-negative integers.



The rounding down mode of the integer division seems to be generally more useful in signal processing applications. Therefore, if future processors support this mode natively in an efficient way, it is possible that Core integer division will also employ this mode. While in this case Core will probably implement a legacy mode switch, it is recommended to stay away from relying on specific rounding direction when there are negative integers involved in the division.

5.5 Denormals and Other Bad Numbers

The IEEE floating point representation, which is typically used on modern computers, has several different classes of special values, which are often processed at a much higher CPU time cost. A single value of this sort occasionally popping up in the real-time data should not cause noticeable harm. The problem is that quite often these values tend to stick in the state memory for quite long periods of time, if not forever (that is until the memory reinitialization).

Denormals

In 32 bit floating point, denormals are the values approximately in the absolute magnitude range between 10^{-37} and 10^{-45}. The values below this range cannot be represented in this format and are replaced by zeros.

For 64bit floating point the denormal range is approximately between 10^{-307} and 10^{-324} in magnitude.

One most critical situation where denormals can appear is exponentially decaying feedback loops (such as feedback loops in the filters, feedback delays, reverbs, etc.). Denormals appear when the signal level decays to 10^{-38} and stays there for a while until it drops below 10^{-45}, which, depending on the decay speed, can take quite a while.

Denormals are also generated in some other situations as well.

In order to mitigate the above problem, REAKTOR Core offers a *DN Cancel* Module whose purpose is to kill the denormals. In the current implementation it is done by adding a small value to the incoming signal.

- If the incoming value is within a typical signal value range (approximately above 10^{-10}), then the addition of the denormal compensation value will not change the incoming value at all, due to the limited precision of the floating point.
- If the incoming value is lower (including the denormal range), it will be possibly modified, but, due to the limited precision of the floating point, will also never produce a denormal result.



Because the denormals are so likely to appear in feedback loops, the library *z^-1 fbk* Macro includes a *DN Cancel* internally.

An additional side effect of the *DN Cancel* embedded into *z^-1 fbk* is that the *z^-1 fbk* acts as an internal excitation source for the self-oscillating filters, like circuit noise in an analog filter. In absence of the *DN Cancel* such filters would not produce signals on their own without an external excitation signal.



Denormals are rarely a problem in feedback loops that are not exponentially decaying. For example, typical oscillator Structures include some kind of feedback loop, which is implicitly formed between the *Read* and the *Write* Modules in the oscillator's phase increment Structure. However, since this Structure is not exponentially decaying, but rather produces a repeating ramp signal, the denormals are not a problem here.

Modern CPUs can often be configured in such way that denormals do not cause extra CPU load (typically in this mode the CPUs simply flush them to zero). So, denormals have become less of a problem in recent years. Nevertheless, it is probably reasonable to take measures against them anyway because:

- we do not know how the situation with handling of denormals by CPUs will develop in the future.
- at some point REAKTOR Core might support a different processor architecture, which does have problems with denormals.
- *DN Cancel* can serve as an internal excitation source for self-oscillating filters.

Infinities

Dividing a non-zero value by zero will produce an infinity (INF). INFs have a tendency to 'stick' in the processing path for a long time, and they also have an increased risk of being converted into a **NaN**.

NaNs

A NaN (not-a-number) is typically produced when some mathematical operation cannot deliver a valid result. If combined with another value a NaN would normally produce another NaN, which results in the tendency of NaNs to 'stick' in the processing paths, similarly to the INFs.

> Initialization problems in filter Structures are one typical source of stuck NaNs, quite likely with some infinities generated along the way.



There is no embedded way in REAKTOR Core to kill NaNs (or infinities). It is the responsibility of the builder to prevent their appearance.

5.6 Optimization hints

5.6.1 Core Cell Handlers

When compiling a Core Cell, the compiler generates a number of different functions, or 'handlers', which correspond to the Primary level internal Module API:

- **Initialization handler:** This handler is invoked once at the very beginning of processing and also upon a 'hard reset' of a Core Cell.

- **Event handlers:** Each of the event-mode inputs of the Core Cell has a corresponding event handler. The event handler unconditionally sends the event from the respective input into the Core Cell's Structure.
- **Audio handler:** This handler is invoked on each audio tick. The audio clock event is unconditionally sent from all respective sources inside the Core Cell.

Each of the handlers is compiled and optimized separately.

- The compiler identifies the areas of the Structure which do not receive any events within the particular handler and does not generate the respective code.
- The compiler identifies the events which are being sent unconditionally within the particular handler (e.g. an audio clock within the audio handler) and does not generate any runtime checks for these events.
- For events that are sent conditionally the compiler generates the runtime checks, which determine whether a particular event is being sent or not.

5.6.2 Latches and Modulation Macros

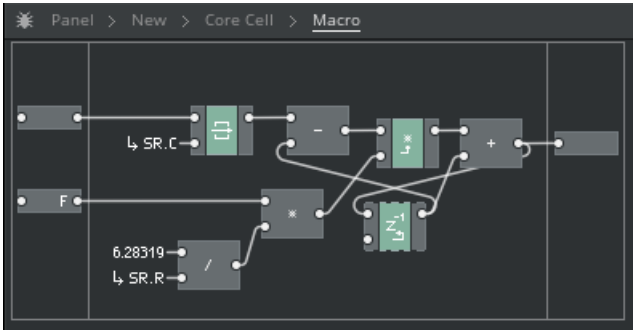
Often in Structure building there is a question of whether a Latch should be used at a particular position. In answering this question the logical function of the Latch and the associated CPU consumption can be considered. As it was already mentioned, the compiler is treating latches (and the 'read followed by a write' pattern in general) in a special optimized way. Thus, the relationship between the usage of a Latch at a particular place and the associated change in the CPU load is not straightforward.

- Generally, if it is not necessary to store the value into the memory (because the value is immediately read afterwards anyway), the compiler will not do so. In such situations a Latch by itself will not add to the CPU load.
- On the other hand, not placing a Latch on some signal path may result in a more complicated triggering logic of the downstream Structure and thus in a higher CPU load produced not by the Latch itself, but by this downstream Structure.

Thus, there is no general rule whether using a Latch will increase the CPU load or decrease it. It is best to simply use latches wherever logically appropriate.

> As Modulation Macros are simply shortcuts for the mathematical operations combined with Latches, the same applies to the Modulation Macros.

The already familiar naive 1-pole lowpass filter Structure illustrates:



A naïve 1-pole lowpass filter.

Depending on the clocking of the signal at the upper input there are the following two typical situations:

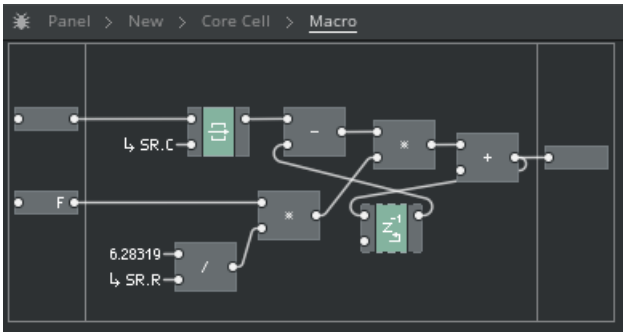
- If the input signal is already clocked at the audio rate, the latch at the input is fully transparent and is not going to produce any compiled code.
- If the input signal is not an audio rate signal, the latch will store its value upon the arrival of the input event, the value being read back upon every audio clock.

The compiler will also handle a number of more complicated situations in regards to the input signal clocking relative to the audio clock.

As for the cutoff control signal at the F input, the same situations are handled in the following ways:

- If the cutoff is an audio rate signal, the output of the first multiplier will be directly used by the internal multiplier of the Modulation Macro during the audio clock processing.
- If the cutoff is not an audio rate signal, the internal latch of the modulation multiplier will store the value, and read it back upon the following audio clock tick.

On the other hand, the following Structure illustrates the case where a usual multiplier had been used instead of the modulation multiplier:



A naïve 1-pole lowpass filter with a buggy use of a non-modulation multiplier.

In the above Structure the following problems would have arisen if a non-audio event arrives at the F input.

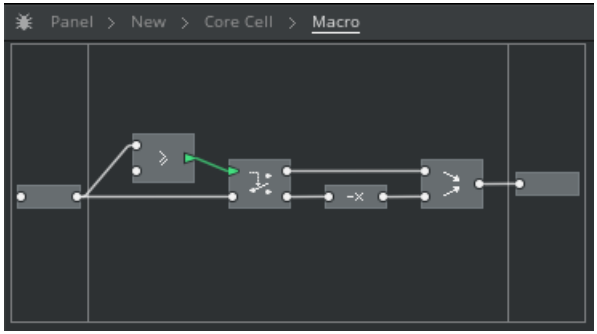
- The non-audio event at the F input will trigger the computation and the sending of the new output value at an inappropriate (non-audio-clock) time moment.
- The subtraction, which is normally done on the audio clock, will not be triggered during this non-audio computation, since neither the *z^-1 fbk* Macro nor the input latch send an event. However, the second multiplier and the adder will be triggered and the computation result will be stored in the *z^-1 fbk* Macro, overriding the previously stored state. Thus, the filter will have performed a 'halfway update cycle', basically destroying its own state.
- The triggering conditions of the second multiplier and of the adder will be a mixture of the audio clock and the cutoff event. If the audio clock is not the default clock (which is unconditionally sent during the audio handler), but something else (e.g. downsampled or gated audio clock), the mixture will require the compilation of a runtime check before the second multiplier, causing a higher CPU load.

Thus a removal of the *Latch* from the Modulation Macro in the above 1-pole filter Structure could have caused both incorrect functioning and a higher CPU load.

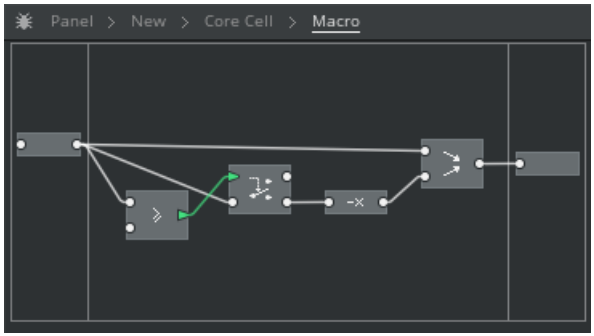
5.6.3 Routing and Merging

The compiler recognizes the following routing/merging patterns:

- Two signal branches from a router are merged back afterwards. Such merging is treated like an endpoint of the splitting:

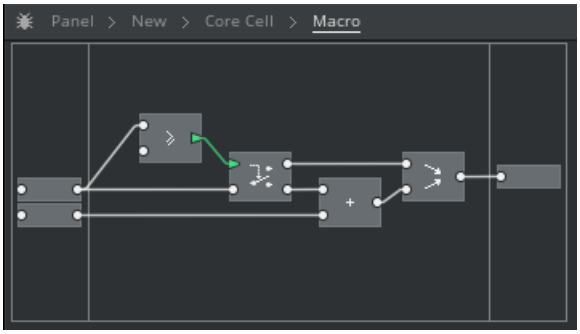


- A branch is being merged back to the pre-router signal. This is also a kind of endpoint of the splitting:



Triggering-wise, most of the arithmetic Modules are equivalent to the *Merge* Modules and are also subject to a similar recognition by the compiler.

The above patterns imply that no additional triggering events have been merged into the branch (e.g. the branch is consistently employing Modulation Macros to mix in further signals). The following Structure illustrates this:



An incorrect merging of two branches.

The Structure is supposed to add the lower input to the negative values of the upper input, while the non-negative values should be left untouched. The Structure however neither works as expected nor the mergepoint really merges the two signal branches back (triggering-wise).

- If the events arrive simultaneously at both inputs of the Macro, the event at the lower input will trigger the adder. The adder will use the latest negative value of the upper input (which is available at the lower output of the router) as the other addend. The result of the addition will be sent through the *Merge* Module to the Macro's output regardless of the sign of the value at the upper input.
- If an event arrives at the lower input only, the situation is similar. The adder will use the latest negative value as the other addend and send it through the *Merge* to the output.
- If an event arrives at the upper input only, the Structure works as expected. The value is forwarded to the output untouched.

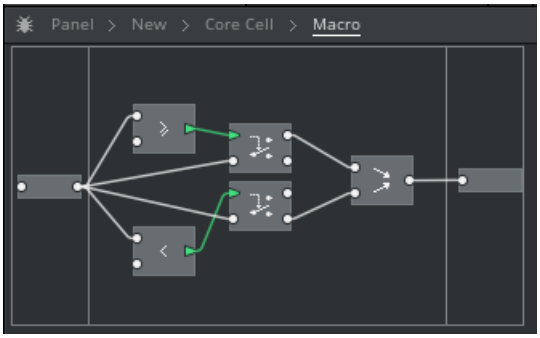
Regarding the triggering conditions of the output, rather than being identical to those of the input, they are a complicated mixture of the triggering conditions of the first and the second inputs and the sign of the value at the upper input. Thus, no splitting endpoint occurs triggering-wise.

Replacing the adder with a modulation adder will fix all of the above problems.



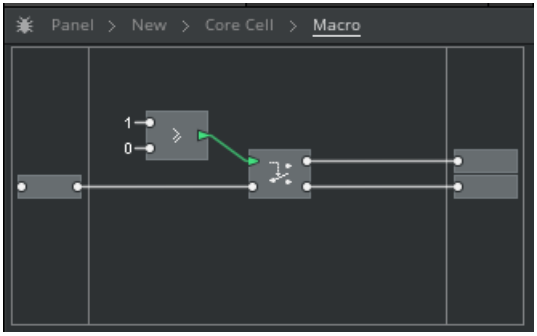
The above example and other kinds of merging do not correspond to the splitting endpoints, because the post-mergepoint triggering information is not identical to the pre-router triggering information. Therefore there is additional CPU overhead at the merging point due to the necessity of checking the runtime triggering condition for the subsequent signal path.

The compiler does not recognize identical or related conditions originating from different BoolCtl sources. A merging of such signals will produce additional runtime overhead as any merging of unrelated signals:



A merging of two formally unrelated signals.

Neither does the compiler identify 'always true' or 'always false' conditions from numerical comparisons:



The compiler will not recognize that the event is always routed to the upper output.

Disconnected BoolCtl ports **are** recognized as providing constant control conditions.

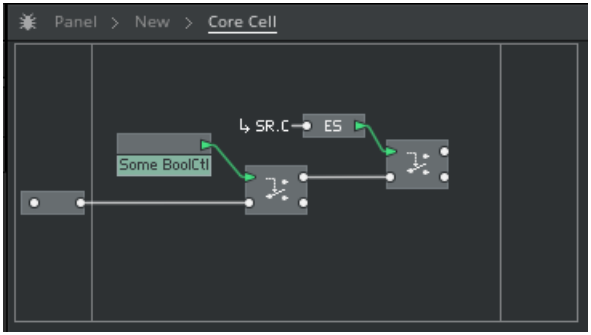
ES Ctl

The BoolCtl signal produced by an *ES Ctl* Module is in the true state when there is an incoming event at the *ES Ctl* Module's input at the very same,moment, and in the false state otherwise. Thus, the *ES Ctl* Module allows you to check whether some event and the event at the Router's input are arriving simultaneously.

The routing controlled by *ES Ctl* Modules is subject to further special optimization, where a router can be found to be in an 'always true' or an 'always false' state. This is particularly detected in the following cases:

- The signal at the *ES Ctl* Module is the 'triggering parent' of the input signal of the Router.
- The event at the ES Ctl Module is mutually exclusive with the event at the Router's input, **because it lies on a different branch of some other Router or because the event simply never arrives within a particular handler.**

This is illustrated by the following picture:



ES Ctl controlled by a 'parent' trigger.

Within the audio handler both the SR.C and the audio-mode Core Cell input (in the inputs area on the left) are triggered unconditionally. Therefore the SR.C and the audio-mode input are considered as the same (unconditional) clock source. So, within the audio handler, the audio rate clock from the Core Cell's input is being sent through the first router. Thus, the input of the second Router is some subset of the audio rate events and there is always an SR.C coming to the ES Ctl simultaneously with the second Router's input events. Due to the 'parent-child' relationship between these two events (one being obtained by routing the other), the compiler is aware of this relationship and will deduce that the second Router can be always considered to be in the **true** state.

Within the initialization handler the compiler will detect that the SR.C will never come at all and will deduce that the second Router is always in the **false** state.

> Actually, compared to the event and especially the audio handlers, it is not crucial to ensure that the initialization handler is optimized, because the initialization handler is executed rarely. The REAKTOR Core compiler also performs less optimization for the initialization handler, often for the sake of being able to perform better optimization for the event and audio handlers.

Reclocking

At each mergepoint the compiler attempts to detect whether a splitting endpoint occurs. In cases where 'chaotic routing' (when the routing branches are not merged back together, but rather some signals with unrelated triggering sources are mixed) has been used in excessive amounts, the analysis time can grow drastically. In the worst cases this can cause the compiler to appear to 'hang indefinitely' (the compiler's progress bar stops completely).

> NI is aware of this issue and is looking for a solution.

In a 'chaotic merging' situation practically each Module represents a new mergepoint with a new set of triggering conditions. While the analysis of these triggering conditions consumes the compilation time, the respective runtime check eventually generated by the compiler consumes the runtime.



'Chaotic merging' by itself is not necessarily bad. The builder simply needs to be aware of the associated CPU overhead. Such overhead can be completely acceptable e.g. in processing of a rarely sent control events. The mentioned growth of the compilation times is another problem requiring the builder's awareness.

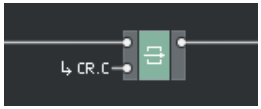
To kill the problem of 'chaotic merging' at its very root the builder can do the following:

- Use Modulation Macros at all logically appropriate places. The Modulation Macros copy the triggering information from their triggering input (as long as they have only one triggering input, which is true for the majority of such Macros), thus the signal stays within the same routing/triggering branch.

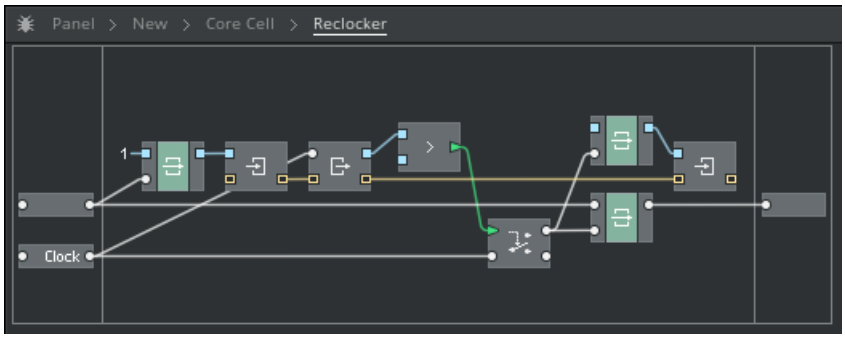
- Use control and audio clocks to make sure that the merged signals are either identically clocked or at least clocked with a small number of different clocks. It is even better if the clocks are derived from some common master clock in a hierarchical fashion, because merging a 'subclock' to its 'parent' will simply produce the 'parent' clock again.
- Use the technique of **reclocking**.

Reclocking means that the builder places Latches at a number of points within a 'chaotically merging' structure. The Latches should be clocked by events with a relatively simple triggering. Some candidates for such clocks are the following.

- A control or an audio clock signal:



- A control or an audio clock signal, put through a router, which is controlled by a Boolean variable, where the latter is being set upon any arriving merged event and reset upon the event passing through the Router:



Other strategies based on the specifics of given Structures can be used.

5.6.4 Numerical Operations

When computing mathematical expressions, various issues need to be taken into account.

Relative Cost

The relative CPU cost of various operations can vary from one processor to the other. However, broadly speaking, the following hints can be kept in mind:

- The cheapest operations are float/int addition, float/int subtraction, float multiplication. Negation, absolute value and integer bitwise operations are equal or slightly more expensive. *DN Cancel* in the current implementation is using an addition internally.
- Float/int division and integer multiplication are noticeably more expensive.
- Float exp and log Modules are the most expensive.
- The cost of runtime condition checks (occurring in routing and non-split endpoint merging) can noticeably vary depending on the circumstances. At any rate they are probably less desirable than a couple of cheap mathematical operations such as addition, so whenever possible, the latter might be preferred.
- The *Read* and *Write* operations are comparable in cost to the cheapest mathematical operations. They might be marginally more expensive for the arrays (this applies to the array memory access, array indexing is more expensive).
- Array indexing internally contains a safety clipping mechanism, which is essentially a runtime check. So avoid sending unnecessary events to the array *Index* Module. Equally, consider sharing an array *Index* Module between a *Read* and a *Write* in such situations.

Relative Order

Check the relative order of the expression computation and consider how often various input events arrive. For example, the following Structure is taking into consideration that the sampling rate changes much more rarely than the frequency control signal, therefore the division will be done only upon sampling rate changes, otherwise only the much cheaper multiplication will be performed:



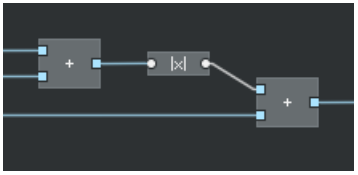
Computation of the normalized frequency for an oscillator.

Had the above Structure been implemented just by dividing F by SR.R, the division would have been performed on each F event, potentially causing a noticeable increase in the CPU load.

Type Conversions

Conversions between floats and integer and **between floats of different precisions** are generally comparable in cost to the cheapest mathematical operations, but this could drastically vary depending on the platform (and used to in the past). Therefore, as a general rule, unnecessary conversions should be avoided.

The following Structure for example does two unnecessary conversions between float and int:



Unnecessary implicit type conversion.

Even though this Structure should typically perform correctly (unless the integer values exceed the range where the integers are exactly representable by floating point), the CPU load of the Structure is higher than necessary due to the type conversions implicitly occurring on both sides of the absolute value Module.

Avoid Bad Numbers

Various special types of values causing higher CPU load have been mentioned in section ↑5.5, Denormals and Other Bad Numbers. These have to be avoided. Since these values can be particularly produced when mathematical operations are computed for the out-of-range inputs, care needs to be taken to keep the values in the supported range.

Even if a particular out-of-range value does not result in a 'bad number' in a given implementation of REAKTOR Core, this might change in the future versions or on different hardware. The result of processing an out-of-range value should not be relied upon.

6 Macro Reference

6.1 Low-level Macros

6.1.1 Math

The Macros for computing various mathematical operations, which are not implemented by the built-in Modules: maximum, minimum, reciprocation, square root, rounding, wrapping etc. As with math built-in Modules, each input is a triggering one.

Trig-Hyp

Trigonometric and hyperbolic functions and their inverses.

Shaper

Functions for control signal shaping.

BLT Prewarp

Frequency pre-warping for the bilinear transform (used in digital filter construction). Various argument ranges and precisions.

6.1.2 Math Mod

Modulation Macros for computing various mathematical operations. See section ↑4.6, Modulation Macros for the explanation of the modulation concept.

The 'tap' Macros are a special kind of Modulation Macros, where the triggering is done by the 'control' input rather than by the 'signal' inputs.

Shaper

Modulation versions of control signal shaper functions.

6.1.3 Clipping

Modulation Macros for clipping the values to specified ranges. The clipping threshold inputs are non-triggering.

6.1.4 Saturator

There are some 'hard saturators', which ensure that the output signal level does not exceed a certain threshold and some 'soft saturators', which do not have a saturation threshold.

6.1.5 Convert

Dedicated mathematical Macros for conversion between different value scales (for example pitch/frequency, gain/decibel etc.).

6.1.6 Memory

Scalar and array storage handling Macros. Latches, array read/write, z^-1.

6.1.7 Flow

Additional router Macros, which employ a threshold input instead of a BoolCtl one. The routing is done depending on how the input signal compares to the threshold value.

6.1.8 Event

Macros specifically focusing on the processing of events (value + triggering). Filtering duplicates, raw clock handling, delaying the events.

6.1.9 Logic

Boolean value handling. The Boolean values are represented by integer type signals.

The built in BoolCtl signal type does not implement true signals in the same sense as scalar types, because it does not transmit triggering information. A related issue is that they do not support classical Boolean operations like AND and OR. If Boolean algebraic computations are required, the *Logic* Macros can be used instead. Particularly they contain:

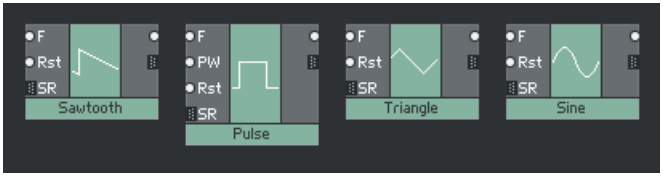
- Comparison operations producing logical signals.
- Classical Boolean AND, OR, NOT and XOR.
- Dedicated float and integer signal routers controlled by a logical signal.
- Conversion to and from BoolCtl.

6.2 Audio

The Library > Audio menu contains a number of low-level audio processing Macros (delays, crossfaders, panners). Further audio processing Macros can be found in the submenus.

6.2.1 Oscillator

Classical analog waveforms



Classical analog waveform Macros.

The inputs are the following:

- F: The oscillator frequency in Hz. Negative frequencies can be specified as well.
- PW: The width of the pulse waveform.
  - 0 means 0% width
  - 1 means 100% width
  - 0.5 means 50% width (default)

- Rst: Oscillator restart trigger. The value of the restart event specifies the restart phase:
  - 0 restarts to the beginning of the cycle
  - 1 restarts to the end of the cycle (which is the same as the beginning, since the waveform is fully periodic)
  - 0.5 restarts to the middle of the cycle
  - 0.3 restarts to the 30% of the cycle
  - Values different by a whole integer restart to the same position, for example 1.3 restarts to the 30% of the cycle

The classical analog waveform oscillators (as well as their slave versions) have a latency of 1 sample. This may need to be taken into account if you need sample-precise control of your structures.

The Rst input is intended for occasional oscillator restarts, for example in response to the MIDI gate signal and should not be used to implement audio oscillator sync. For oscillator sync use the slave versions of the same oscillators.

The SR input is the standard SR Bundle connector explained in section ↑4.9, SR and CR Buses.

The upper output of the oscillators delivers the oscillator signal. The lower (Bundle) output is for connection of the slave oscillators.

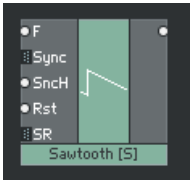
A special *4-Wave* oscillator Macro delivers all four waveforms simultaneously. The waveforms produced by this oscillator are phase locked to each other.



The 4-Wave oscillator.

'Slave' Oscillators

The 'slave' oscillators are the audio-synced versions of the ordinary oscillators. Compared to the ordinary oscillators, they have two additional inputs:

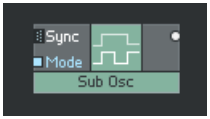


A 'slave' sawtooth oscillator.

- Sync: Can be connected to the lower (Bundle) output of a master oscillator to produce the audio sync effect. Only ordinary (non-slave) oscillators can serve as master oscillators.
- SncH: Controls the sync hardness:
  - 1 = hard sync
  - 0 = no sync
  - 0 to 1 = various degrees of soft-sync

Sub Oscillator

The sub oscillator is a special kind of 'slave' pulse oscillator, which is phase-locked to the master.



Sub oscillator.

The idea of the sub oscillator is that it changes its output from -1 to 1 or back whenever there is a transition (a cycle completion) in the master, thereby producing a square waveform one octave below the master oscillator's frequency. By leaving out some of the master transitions out, further sub oscillator modes are available:

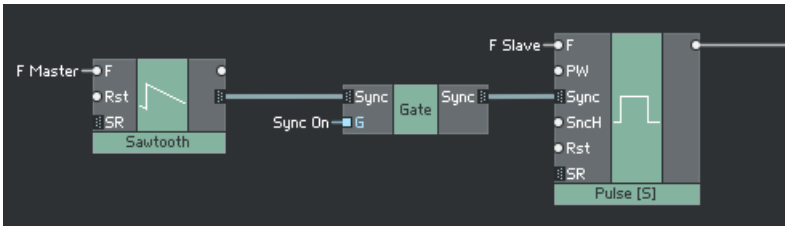
- 0 = -1 octave, 50% pulse width (use every master transition)
- 1 = -2 octaves, 50% pulse width (use every other master transition)

- 2 = -2 octaves, 75% pulse width (use 1st, 4th, 5th, 8th etc. master transitions)

The sub oscillator is fully driven by the master oscillator. Therefore it stops whenever the master is stopped.

Sync Gate

The *Sync Gate* Macro allows to interrupt the sync signal from the master to a slave or a sub oscillator.



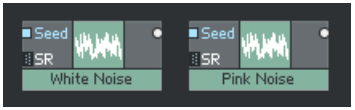
Using a sync gate to stop the audio sync.

The signal at the G input should be either 0 or 1. In the 0 state the sync signal does not come through, effectively disabling the synchronization of the slave to the master.

A sub oscillator would be completely stopped if the gate is off.

Noise Oscillators

These generate different kinds of noise.



Noise oscillators.

If several identical noise oscillators are used, then by default they will produce fully identical signals. This can be avoided by connecting different constants to their Seed inputs, which then initializes their internal (pseudo) random number generators with different values.

6.2.2 Filter

The Core Macro Library contains a selection of ZDF filters. The filters are built using the *ZDF toolkit*, which is also part of the Library (see section ↑6.4.3, ZDF Toolkit).

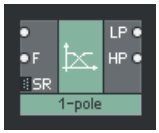
> The filters built using the ZDF toolkit are optimized for Structure readability, not for CPU efficiency. A number of filters have been redone with CPU efficiency in mind. These can be told by the "(opt)" suffix (standing for optimized) at the end of their names.

Linear Filters

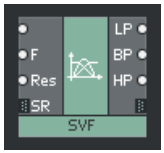
Linear filters only change the frequency spectrum of the signal and do not introduce any distortion. A number of these filters can resonate but cannot self-oscillate (at least not reliably). The maximum resonance setting of these filters, where applicable, is therefore limited to (approximately) 1.

The most important Macros implementing the linear filters are the following.

- 1-pole > 1-pole: A multimode linear 1-pole filter producing a low-pass and a high-pass signals with a -6dB/octave roll-off.



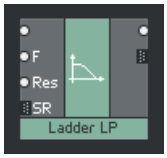
- 2-pole > SVF: A multimode 2-pole resonating filter producing a low-pass, a band-pass and a high-pass output. The roll-off of low-pass and high-pass is -12dB/octave. The roll-off of the band-pass is -6dB/octave on each of the two slopes. Modal pickups (see below) can be used to obtain further modes.



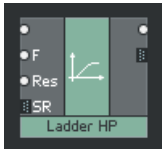


Intuitively, although not fully correctly, one could imagine each of the filter's poles to be responsible for a -6dB/octave roll-off: one pole is responsible for the -6dB/octave slope to the left of the filter's center frequency, and the other pole is responsible for the -6dB/octave slope to the right of the filter's center frequency.

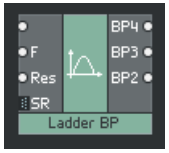
- 4-pole > Ladder LP: A 4-pole resonating low-pass filter, emulating the classical transistor ladder filter design. The roll-off is -24dB/octave. At higher resonance settings this filter naturally produces a drop in the bass frequencies. This drop can be avoided by using the modal pickups (see below).



- 4-pole > Ladder HP: A 4-pole resonating high-pass filter. The roll-off is -24dB/octave. The *Ladder LP* modal pickups (see below) can be used with *Ladder HP*, producing 'flipped' filter shapes (high-pass instead of low-pass etc.).



- 4-pole > Ladder BP: A 4-pole resonating band-pass filter. The roll-off of each of the two slopes is -12dB/octave. The filter does not provide the modal pickup connector, but two further modes with lower pole counts (BP3 and BP2) are provided directly by the Module outputs. At zero resonance the BP3 mode has a -12dB low-pass roll-off and -6dB high-pass roll-off, while BP2 has both roll-offs at -6dB/octave



The F input specifies the filter cutoff frequency in Hz.

The Res input, if available, controls the filter's resonance. The range is from 0 to approximately 1. In fact, the maximum resonance value is clipped slightly below 1, because at too high resonance settings linear filters may produce very loud resonance peaks. For higher resonance settings use nonlinear filters (see below).

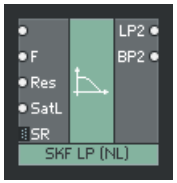
> The resonance control of the filter Macros is implemented by linear mapping of the resonance parameter to the filter's internal parameters (such as feedback amount or damping). This does not necessarily provide the best resonance control curves. Use control signal shapers (such as those found in Math Mod > Shapers) to change the resonance control curve. For example, an *SVF* resonance control knob can be shaped with a *Hyp Shaper* whose bending parameter is set to 0.3. This will make the resonance control of the *SVF LP* feel similar to the one of the *Ladder LP*.

Nonlinear Filters

Nonlinear filters, besides the purely filtering effect, apply some nonlinear processing to the signal, causing it to saturate internally in the filter. As a result, the resonance of such filters can be driven to excessive values (above 1) without causing the filter to 'explode'. At these excessive resonance settings the filters may produce sound even in the absence of the input signal, which is referred to as the self-oscillation effect.

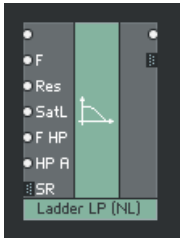
The most important nonlinear filters in the Core Macro Library are the following.

- 2-pole > SKF LP (NL): Nonlinear Sallen–Key low-pass filter (with an additional band-pass output). In the linear range (low signal levels) this filter sounds identically to the *SVF* and therefore can be thought of as a nonlinear counterpart of the latter. The SatL input controls the internal saturation level of the filter. The smaller is the saturation level setting, the stronger the filter will saturate (default value is 1).



> For a larger variety of output signals use nonlinear TSK filters, which have dedicated pickup Macros. The Core Macro Library also contains a nonlinear SVF, but SKF and TSK are the recommended option for nonlinear filters.

- 4-pole > Ladder LP (NL): A nonlinear counterpart of *Ladder LP*. The SatL input controls the internal saturation level of the filter. The smaller is the saturation level setting, the stronger the filter will saturate (default value is 1). The F HP and HP A inputs control the cutoff (in Hz) and the amount (from 0 to 1) of the internal high-pass filtering of the feedback signal. One particular effect of this high-pass filtering is prevention of the loss of bass frequencies. When used together with normalized DC gain pickups (such as *Ladder LP -> LPn*), it can produce an extra bass boost.

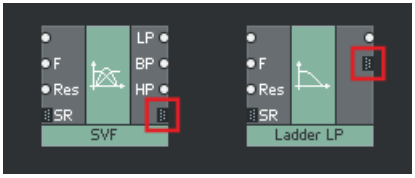


The Core Macro Library also contains 8-pole nonlinear ladder filters. The 8-pole band-pass ladder filter stands out among those by having two resonance peaks in the negative resonance self-oscillation range, which can generate quite special sounds.

In order to better balance the behavior of the saturating nonlinear filters in the low-resonance and self-oscillation it is recommended to connect an additional saturator (see section ↑6.1.4, Saturator) at the filter's output.

Modal Pickups

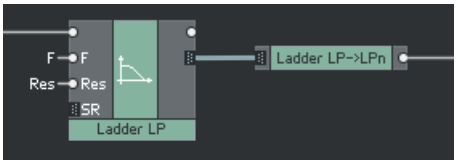
With many filter designs (including *Ladder*, *SVF* and *TSK*) a large number of different modes can be picked up from the filter. In order to keep the number of outputs low, many filter Macros in the Core Macro Library provide only the main mode (or a small number of 'main modes') directly as their outputs. The remaining modes can be picked up from the dedicated modal Bundle output:



Modal Bundle outputs of *SVF* and *Ladder LP* filters.

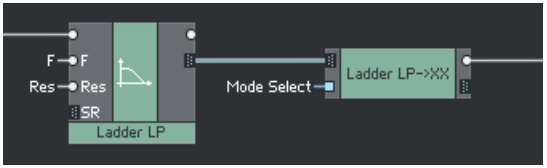
The pickup can be done by using the modal pickup Macros, located in the Pickup submenus of the respective menus. For example, modal pickups for the 2-pole filters are located in Audio > Filter > 2-pole > Pickups. The pickup Macros are categorized as follows.

- **SVF pickups:** Can be used with linear and nonlinear 2-pole *SVF* filters.
- **TSK LP pickups:** Can be used with linear and nonlinear *TSK LP* filters.
- **Ladder LP pickups:** Can be used with linear and nonlinear *Ladder LP* filters. The same pickups can also be 'abused' with the *Ladder HP* filter, producing 'flipped' modes (for example HP modes instead of LP modes etc.).



A modal pickup picking up a normalized-DC-gain low-pass mode of a *Ladder LP*.

There is a special XX pickup category. These pickups do not pick up a specific mode, but rather allow dynamic control over which specific mode is being picked up. The control is performed via a selector input. The specific meanings of the selector values can be taken from the information of the respective pickup Macros.



Dynamic modal pickup.

The upper output of such dynamic pickup Macros provides the picked up modal signal.

> The lower Bundle output of "XX" pickups provides the modal mixing coefficients for the selected mode. These coefficients can be particularly used by the TF modal pickup Macros from the TF Toolkit (see section ↑6.4.4, TF toolkit).

Multinotch

Contains multi-notches with a low notch count and a comb filter.

Butterworth

Butterworth filters are a classical signal processing filter type. By design, they are intended to statically cut a frequency band from a signal, therefore they do not have any resonance setting. Nevertheless, their parameters can be modulated, as with the other filters in the Core Macro Library.

> The *Butterworth 2* filter group contains the 2nd kind (resonating) Butterworth filters as described in The Art of VA Filter Design by Vadim Zavalishin.

Shelving

Each of the shelving filters simultaneously produces three different responses at the respective three outputs: low-shelving, high-shelving and tilting (simultaneously low- and high-shelving in opposite directions and at half amounts). Due to technical reasons the high-shelving output has inverted control of the gain (for example if Gain input is set to +12, the high frequencies will be lowered by 12dB). As a convenience means for controlling the high-shelving gain, there is an inverting Gain input.



An 8-pole shelving filter.

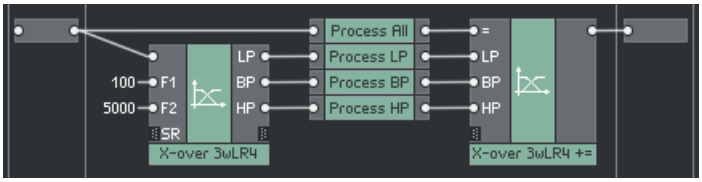
Shelving filters have an additional Slope input which controls the slope steepness. The range is 0 to 3 for the 8-pole shelving filter. Shelving filters with lower pole counts cover a smaller range of Slope value, the maximum supported slope setting being 2 for 4-pole, 1 for 2-pole and 0 (therefore there is no slope input) for the 1-pole.

The shelving group also contains band-shelving filters, which also have three outputs: band-shelving, “notch”-shelving and band-tilting. The most important output is the band-shelving one, the other two are provided mostly for completeness, as an analogy to the low-shelving/high-shelving/tilting outputs. The “notch” shelving is an “inverted” version of the band-shelving (the variable-gain shelves are on the left and on the right of the center band) and is controlled by an inverted gain signal. The band-tilting is a combination of band- and notch-shelv-

ing done in opposite directions at half amounts. The BW input controls the shelving bandwidth in octaves. The range of the Slope parameter for band-shelving filters is 0 to 2 for the 8-pole, 0 to 1 for the 4-pole and 0 to 0 (no Slope input) for the 2-pole.

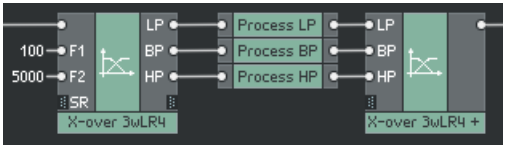
Crossovers

The crossover Macros come in groups of four to be used together. In the simple case you will need only two of the respective Macros. For example, in the picture below a signal is being split into three bands by a 4th-order Linkwitz-Riley crossover. The original signal and the three bands are processed by four independent processors and then are merged together, where you need to use a special dedicated Macro for merging (there is a different Macro for each kind of crossover). The reason you need this Macro is that there are phase differences between the signals and they need to be phase corrected before the merging.



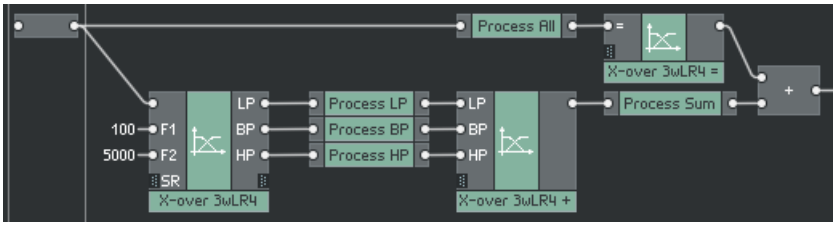
A usage example of a 3-band crossover with band thresholds at 100Hz and 5kHz

If you don't need to merge the original signal back, a somewhat less CPU-intensive Macro can be used instead:



A cheaper mixer for the 3-band crossover.

On the other hand, if you merge the original signal later, you need to use a separate Macro for the phase compensation:



Phase correction of the unsplit signal.

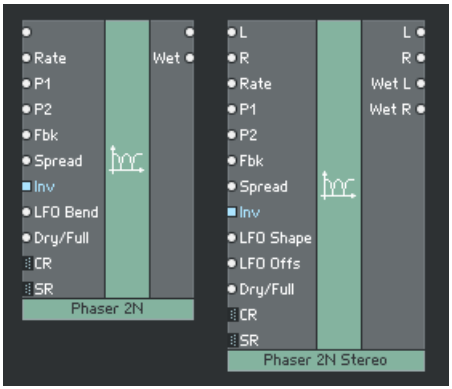
6.2.3 FX

A selection of various effects.

Modulation Effects

The modulation effects include phaser, flanger, chorus, tremolo and autopan. Except for autopan, there are mono and stereo versions of these effects. Most of their control parameters are identical or similar.

The phaser Macros are discussed below as typical examples. Further details about the effect parameters which are different from those of the phasers can be found in the information of the respective Macros.



Stereo and mono versions of 2-notch phaser.

The following inputs are available:

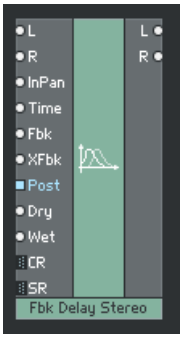
- L, R (stereo) or the top nameless input (mono): Input signal.
- Rate: The rate of internal LFO in Hz.
- P1, P2: The boundary pitches of the modulation range. The center cutoff frequency of the internal multinotch filter is modulated within this pitch range.
- Fbk: Feedback amount. Range -1 to 1. The feedback setting affects the width of the notches, positive values making the notches wider, negative values making them smaller.
- Spread: Adjust the spreading of the notches. Negative values increase the notch density, positive values spread them wider apart.
- Inv: The notch inversion mode switch. Swaps the positions of the peaks and the notches. 0 = off, 1 = on.
- LFO Bend (effects with exponential LFO): Controls the bending of the exponential LFO segments. 0 = linear, +1/-1 = rectangular jumps.
- LFO Shape (effects with triangle-sine morphing LFO): Controls the morphing amount from the triangle LFO shape to the sine LFO shape.
- Dry/Full: Controls the mixing amount of the dry and wet signals. 0 = dry signal only, 1 = 'strongest sounding' mixture of dry and wet signals.

The following outputs are available:

- L, R (stereo) or the top nameless output (mono): Output signal.
- Wet L, Wet R (stereo) or Wet (mono): Pure wet signal. To be used with the 'send' style of effect usage, when several sounds are sent to the same effect with different 'send' amounts, producing a common 'wet' signal which is then returned back to the entire mix.

Delay Effects

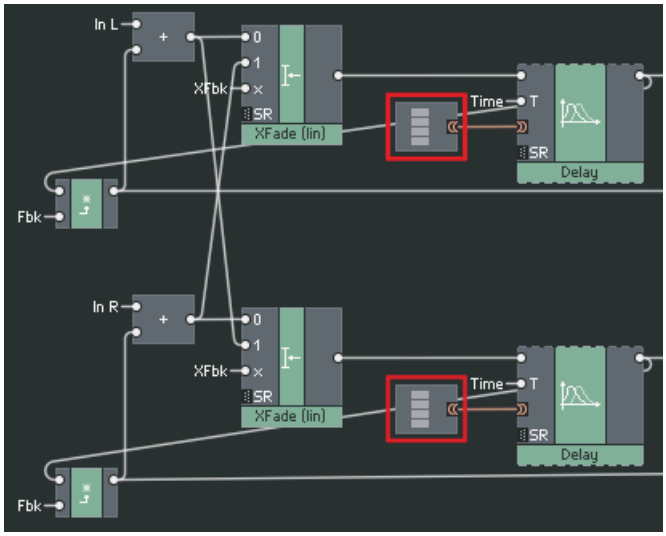
The delay effects are using audio delays to produce echo sounds. A stereo feedback delay Macro illustrates:



Stereo feedback delay.

The following inputs are available:

- L, R: Signal inputs.
- InPan: Controls stereo balance of the input signal. Range -1..1.
- Time: Specifies the delay time in seconds. Maximum delay time at 192 kHz sampling rate is 1 sec. At lower sampling rates the maximum time is larger. It can be further increased by modifying the delay buffers inside the Macro:



- Fbk: Feedback amount. Range -1..1.

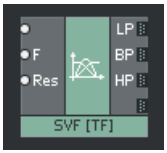
- XFbk: Cross-channel (left/right) feedback amount. Range -1..1.
- Post: The 'post-feedback-gain' output mode. If this input is set to 1, the output signal is picked up after the feedback gain has been applied (if the feedback setting is zero, there will be no wet output signal in this mode).
- Dry: Amount of dry signal to be sent to the outputs
- Wet: Amount of wet signal to be sent to the outputs.

The following outputs are available:

- L, R: Output signal.

6.2.4 Transfer Functions

A set of Macros for computing the transfer functions (more precisely, frequency responses) of the filters is located under Library > Audio > Filters > Tranfer Func. These are the counterparts to the respective filters. They follow one and the same set of conventions. The *SVF* transfer function Macro illustrates:



An *SVF [TF]* Macro.

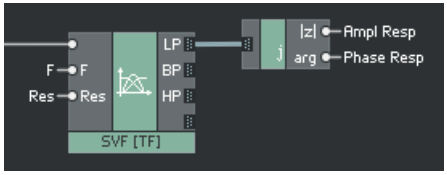
The inputs and the outputs of a *TF* Macro are roughly identical to those of the respective filter Macro with some differences.

- The upper input of a *TF* Macro receives a signal frequency value in Hz instead of the audio signal. The Macro will output the value of the filter's frequency response at this frequency.

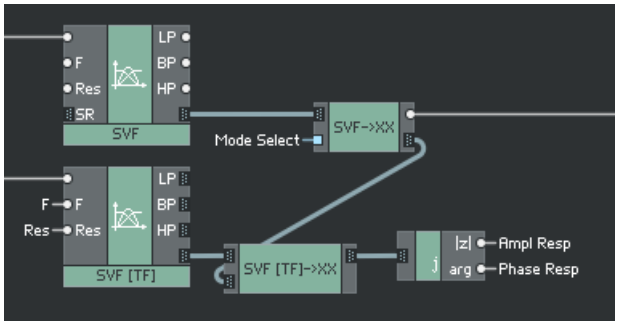
  In order to build a graphical display of filter's amplitude or phase response, send an iterated range of signal frequency values to this input and pick up the respective output values.

- Other inputs of a *TF* Macro receive the parameter values identical to those of the respective filter Macro.

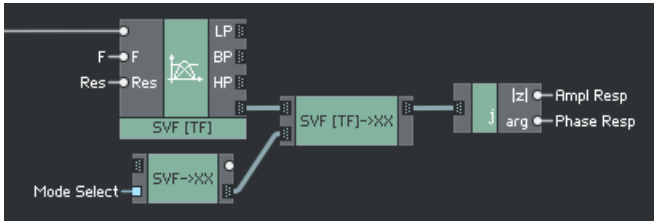
- The 'audio signal' outputs of the *TF* Macro (those corresponding to the audio signal outputs of the filter) produce the frequency response value at the frequency specified by the upper input. These are complex values which can be converted into the amplitude and phase response parts by using the *to Polar* Macro from the *TF* toolkit (see section ↑6.4.4, TF toolkit):



- The bottom 'modal Bundle' output contains a Bundle of modal *TF* values, which can be picked up by a dedicated *TF* pickup Macro. Except for the all-pass mode, the modal *SVF* frequency response pickup is done by using the *SVF [TF] to XX* Macro. The *SVF [TF] to XX* Macro does not have an integer modal selector input, in place of it is expecting a modal coefficients Bundle from the *SVF* to XX selector:



The *SVF to XX* Macro in the above picture does not need to be connected to an audio filter:





The *TF* Macros compute the ideal analog frequency responses, not taking into account the discrete-time warping of the frequency axis.

Some of audio filters do not have *TF* counterparts. However, often there is a *TF* counterpart, which is not immediately obvious. Below are some examples.

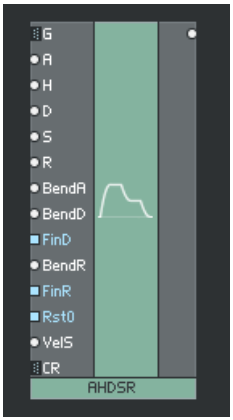
- There are no dedicated *TF* Macros for nonlinear filters (since frequency responses in unstable range do not really make sense anyway). Use the corresponding linear *TF* Macros.
- The frequency response of *TSK* filters is identical to those of *SVF*, so simply use an *SVF TF* Macro to obtain *TSK* frequency response.
- The *Ladder LP [TF]* Macro has the F HP and HP A inputs available only for the *Ladder LP (NL)* Macro. To obtain the frequency response of a low-pass ladder filter without a high-pass in the feedback these inputs can be simply left disconnected.

6.3 Control

6.3.1 Envelope

A small selection of prebuilt envelopes of varying complexity. If the selection is not sufficient, the existing envelopes can be easily extended by the builder, because they are built using the Envelope toolkit (see section ↑6.4.2, Envelope Toolkit).

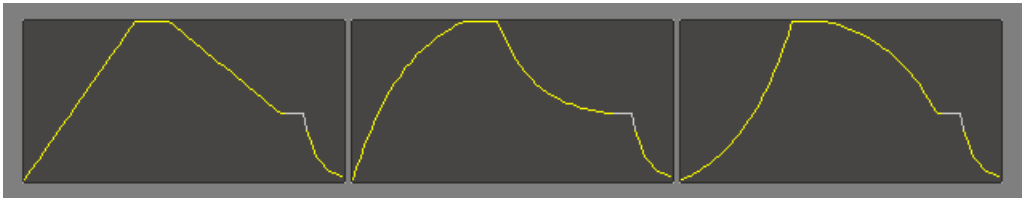
The envelopes follow one and the same set of conventions. The AHDSR (attack-hold-decay-sustain-release) envelope is used here as an example to illustrate these conventions:



AHDSR envelope.

The envelope inputs are the following.

- G: Should be connected to a gate Bundle (see below for the explanation of Core's gate Bundle conventions). It is responsible for triggering the attack (upon the 'gate on' event) and release (upon the 'gate off') modes of the envelope and for sending the keypress velocity information, if any.
- A, H, D, S, R: The 'main' control parameters. A, H, D and R specify the time constants for the attack, hold, decay and release stages. S specifies the sustain level, which normally should be in the range 0 to 1, since the attack peak level is always 1.
- BendA, BendD, BendR: The bending of the attack, decay, and release stage curves. The bending applies only to the finite-time curves (see the finite-time mode explanation below). The Bend parameter range is -10 to 10. Zero bending produces linear curves, negative bending produces 'slowing down' curves, positive bending produces 'speeding up curves'. The following pictures illustrate the attack and decay shapes for zero, negative and positive bends respectively:



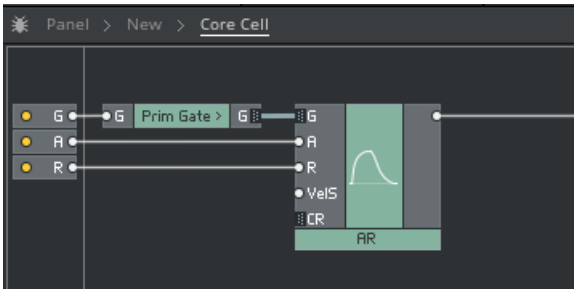
- FinD, FinR: The finite-time mode of the decay and release stages. When the mode is set to 1 (finite), the respective stage has the duration as specified by the D or R input. When the mode is set to 0 (infinite), the respective stage decays infinitely to its target value (sustain level or zero).
- RstO: The zero-reset mode of the envelope. When set to 1 the attack always starts from zero level. When set to 0 the attack starts from the current level of the envelope.
- VelS: Velocity sensitivity, the range is 0 to 1. If set to 0, the envelope ignores the velocity information in the gate Bundle completely. If set to 1, the envelope scales its level according to the velocity (provided the velocity information is present in the Bundle, see the discussion of the gate Bundle below). At intermediate settings the envelope partially scales its level according to the velocity.

The CR input is the standard CR Bundle connector explained in section ↑4.9, SR and CR Buses.

Gate Bundle

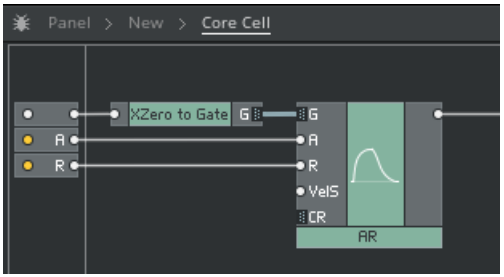
The envelope gate connections are employing a special gate Bundle. The Bundle internally transmits the information about the gate on/off events and the associated velocities.

The gate Bundle can be obtained by converting from a Primary level gate event (forwarded to Core via an event-mode Core Cell port). The conversion is implemented by the *Gate from Prim* Macro (Library > Control > Envelope > Gate > Gate from Prim) and includes the gate on/off events and the 'on' velocity (no 'off' velocity):



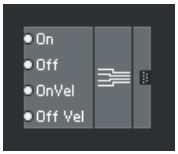
AR envelope controlled by a Primary gate event signal.

The *XZero to Gate* Macro (Library > Control > Envelope > Gate > XZero to Gate) sends gate events in response to positive and negative zero crossings of the input signal (positive zero crossings produce 'gate on' events, negative zero crossings produce 'gate off' events):



AR envelope controlled by zero crossings of the incoming audio signal.

Internally the gate Bundle consists up to four fibers:



A Bundle Pack for all four fibers of the gate Bundle.

The conventions for the gate Bundle are the following.

- On: Transmits the 'gate on' trigger. The consumers should ignore the value of this fiber. The producers should always send zero values on this fiber.
- Off: Optional fiber. Transmits the 'gate off' trigger. The consumers should ignore the value of this fiber. The producers should always send zero values on this fiber. If this fiber is absent, it means that the 'gate off' trigger events are never sent.
- On Vel: Optional fiber. If present, specifies the current 'on velocity'. The standard range is positive numbers less than or equal to 1, however, larger values are allowed as well. Generally speaking, this value should be sent only simultaneously with 'gate on' events (or sent during the initialization, if the value is constant). If an 'on velocity' event comes at any other moment, the consumers may respond to it or ignore it until the next 'gate on'. Velocity-sensitive consumers should treat the missing "On Vel" filber as a value of 1.
- Off Vel: Optional fiber. If present, specifies the current 'off velocity'. The convention is the same as for the 'on velocity', except that the 'off velocity' events should be sent together with the 'gate off' events.

A number of preconfigured Bundle *Pack* and *Unpack* Modules with different numbers of fiber definitions/pickups are available under Library > Control > Envelope > Gate.

The *Add Gate Defaults* Macro (Library > Control > Envelope > Gate > Add Gate Defaults) automatically adds Off, On Vel and Off Vel to the Bundle, in case these fibers are missing (the added velocities are equal to 1). It is intended to be used in implementations of gate Bundle consumers and other Structures.

When constructing a gate Bundle, never leave the triggering fibers On and Off disconnected. Disconnected fibers imply a zero constant, which will send a trigger event upon initialization. Simply leave the optional fiber away from the Bundle, or connect it to a *No Event* Macro if you do not want any events sent.

6.3.2 LFO

A number of LFOs of classical shapes. These are intended to be used as modulation sources, not as audio signal sources. Particularly, the LFO Macros do not contain any anti-aliasing.

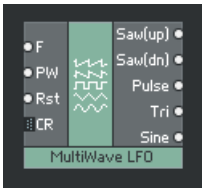


Various LFOs available in the Core Macro Library.

The inputs are similar to those of the oscillators (see section ↑6.2.1, Oscillator).

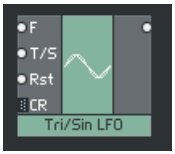
The CR input is the standard CR Bundle connector explained in section ↑4.9, SR and CR Buses.

The *MultiWave LFO* provides the same waveforms (except the parabolic one) phase-locked to each other.



MultiWave LFO.

The *Tri/Sin LFO* can be smoothly morphed between the triangular and sinusoidal shapes. The T/S input controls the morphing (0 = triangle, 1 = sine, other values in between specify intermediate shapes).



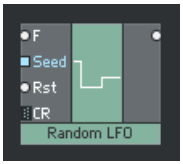
Tri/Sin LFO.

The *Exp LFO* is generating analog-like slowing down exponential curves. The slowdown amount varies from a purely linear segment to a discontinuous jump, thereby morphing between triangular and rectangular shapes at the zero and maximum settings. Instead of slowing down shapes, the speeding up shapes can also be generated. The Bend parameter controls the segment shape (0 = linear, 0..-1 = slowdown, -1 = rectangular, 0..1 = speedup, 1 = rectangular). The PW parameter controls the 'pulse width'.



Exp LFO.

The *Random LFO* is delivering the signal which would be produced by a white-noise driven S&H. The Seed parameter is similar to the one of the noise oscillators (see section ↑6.2.1, Oscillator).



Random LFO.

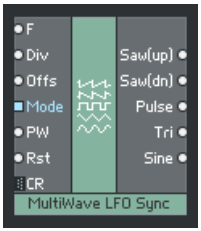
The triangle, sine and parabolic LFO shapes are also delivered in stereo versions. Compared to mono versions, these have an additional Offs input, which controls the offset between the left- and the right-channel phase-locked LFO shapes. At zero Offs both left and right channel signals are identical. For Offs between 0 and 0.5 the left channel LFO is 'preceding' the right channel one ('movement from left to right'). For Offs between 0 and -0.5 the right channel is 'preceding' the left one ('movement from right to left'). At ±0.5 the channels have opposite phases (both +0.5 and -0.5 settings have identical effect). The Offs value can exceed the ±0.5 range, so that at ±1 the channels are again in sync and so on.



Stereo LFOs.

Tempo and transport position syncing

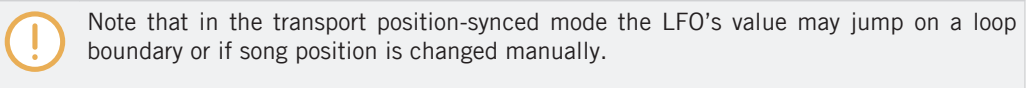
The *Multiwave LFO Sync* Macro is capable of syncing to the tempo and/or transport position:



Tempo- and transport-syncable LFO

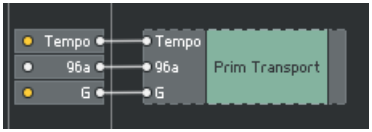
The Mode input controls the syncing where you can independently enable tempo and transport position syncing (see the Macro's info for the details).
If tempo syncing is enabled, instead of the F input specifying the LFO rate in Hz, the Div input is used, specifying the division of the whole note. For example, Div=8 sets the duration of one LFO cycle equal to the duration of an 8th note.

If transport position syncing is enabled, the LFO doesn't freely run but strictly follows the song position. This guarantees that the LFO will always have one and the same value at each given song position, therefore LFO works exactly the same each time, like an automation curve. If the transport is stopped, the LFO can continue to run freely or stop, like an automation curve would do (this choice is also separately controlled by the Mode parameter).



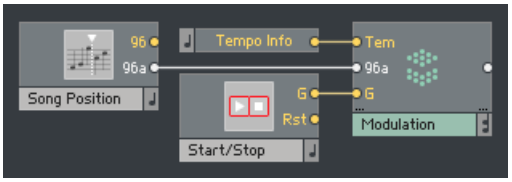
Note that in the transport position-synced mode the LFO's value may jump on a loop boundary or if song position is changed manually.

The tempo and transport position syncing require the transport information to be available. This can be achieved by using the *Prim Transport* Macro (Library > Control > LFO > Helpers > Prim Transport) which defines a number of scoped buses based on primary level signals.



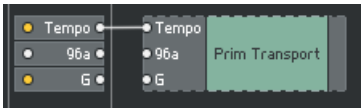
Prim Transport Macro providing the transport information.

The input signals for the *Prim Transport* Macro need to be obtained from the respective primary modules.



Obtaining signals for the *Prim Transport* Macro.

If you need only tempo- or only transport position-syncing, you need to provide only the respective signals to the *Prim Transport* Macro.



Obtaining only the tempo information.

Other tempo and transport position synced LFOs (for example the ones with only tempo or only transport position syncing, or providing only one output waveform) can be built, by stripping down and/or modifying the internals of the *Multiwave LFO Sync* Macro. For doing such modifications using the LFO toolkit Macros is recommended.

6.3.3 Smoother

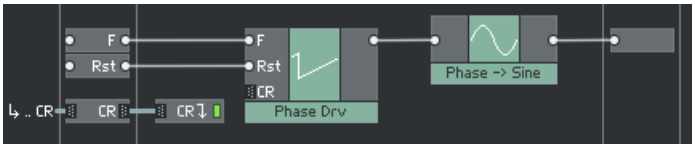
Please refer to section ↑4.9.4, CR rate change for the details of using smoothers.

6.4 Toolkits

6.4.1 LFO Toolkit

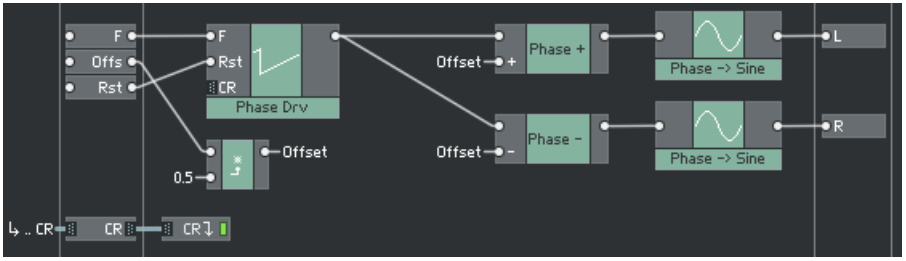
The LFO toolkit consists of two parts: Helpers (Library > Control > LFO > Helpers) and Phase Shapers (Library > Control > LFO > Phase Shapers). The Helpers part concerns itself with generation of the LFO phase signals. The Phase Shapers part contains Macros for converting the phase signal into LFO waveforms.

As a simple example of the usage of the toolkit Macros consider the internal structure of the *Sine LFO* Macro from the Core Macro Library:



Here the phase driver is directly connected to the shaper, the later converting the phase signal to the sine waveform.

The *Sine LFO Stereo* Macro additionally uses two phase offset helper Macro to shift the left and right channels symmetrically in the opposite directions:



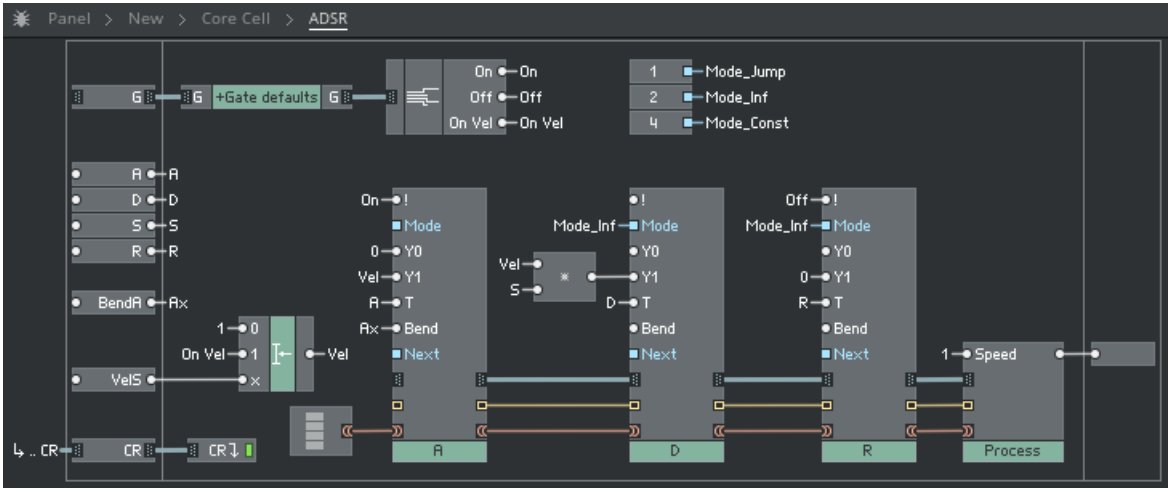
To build LFOs with tempo and/or transport position syncing features it is easiest to use the *Phase Drv Sync* helper Macro (which is the core of the *MulltiWave LFO Sync* Macro). Lower-level helpers such as *Position Sync* and *Div to F* Macros can also be used.

6.4.2 Envelope Toolkit

The envelope toolkit (Library > Control > Envelope > Toolkit) contains the building blocks of the envelopes in the Core Macro Library. In fact there are only two Macros (*Stage* and *Process*) in the toolkit and each envelope is built using these two Macros, where the *Process* Macro is used exactly once in each given envelope. Therefore in order to build a custom envelope one could simply start off with the existing envelope and duplicate the Stage Macro within that envelope as necessary.

> In fact this duplication approach should be even preferred, because due to the development of the Core Macro Library, different versions of the *Stage* and *Process* Macros may become incompatible with each other.

The implementation of the ADSR envelope found in the Core Macro Library illustrates this:



The internal Structure of an ADSR envelope.

The *A*, *D* and *R* Macros are simply the renamed copies of the *Stage* Macro. There is no dedicated Macro for the sustain stage, because in this envelope the decay stage is infinitely decaying. Thus there is really no sustain stage, only the sustain level as the target level of the decay stage.

The *Process* Macro on the right is the one which generates the real envelope signal, the *Stage* Macros only being responsible for setting the stage parameters. The upper input of the *Process* Macro specifies the speed coefficient for the envelope. This can be useful if the envelope toolkit is used to build an LFO, otherwise this coefficient should be set to 1, so that the stage durations are exactly as specified by the respective stage's parameters.

The three parallel connection paths at the bottom are used to connect the stages together in their natural order. The *Process* Macro must be at the end of the chain. For the first stage Macro these three inputs must be left disconnected, except for the lowest of them. This should be connected to a 64-bit precision float array. The number of elements of the array must be equal (or at least equal) to the number of the stage Macros times 6.

The other inputs of the stage Macros are as follows.

- !: This is the explicit triggering input of the stage. An event at this input will force the envelope to immediately jump to this stage. Notice that the 'Gate On' event is connected to the triggering input of the attack stage, while the 'Gate Off' event is connected to the triggering input of the release stage.
- Mode: A bitwise OR of stage modal flags:
  - 1 = jump mode. If specified, this flag causes the stage to start (when explicitly triggered or automatically switched to) at the level specified by the Y0 input. Otherwise the stage starts at the current output level of the envelope (no jump).
  - 2 = infinite mode. If specified, this flag causes the stage to infinitely decay from the starting level (according to the jump mode and Y0 settings) to the end level specified by Y1. The infinite decay is exponential, the decay characteristic time being controlled by the T input.
  - 4 = constant mode. If specified, this flag causes the stage to stay at its starting level (according to the jump mode and Y0 settings) infinitely. The segment end level specified by Y1 and duration specified by T are ignored.
- Y0: The starting level of the stage. Used only if the jump mode is specified.
- Y1: The end level of the stage. Used unless the constant mode is specified.
- T: The segment duration in seconds. In the infinite mode this input specifies the segment's characteristic time. In the constant mode this input is ignored.
- Bend: The exponential bending of the segment. Does not apply to infinite or constant segments. Refer to section ↑6.3.1, Envelope for more details about this parameter.
- Next: The zero-based index of the 'next' segment (the segment which should be switched to after this segment runs out). If disconnected, assumes the 'natural' next segment, according to the order of connection of the stage Macros.

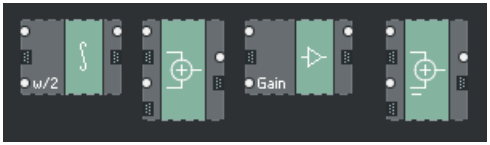


The parameters can be modulated in real-time. Particularly the modulation of Y1, T and Bend parameters of the currently running segment will cause real-time changes of the envelope behavior.

6.4.3 ZDF Toolkit

The ZDF toolkit (Library > Audio > Filters > ZDF Toolkit) contains the building blocks of the filters in the Core Macro Library. The purpose of the toolkit is to allow the implementation of ZDF (zero delay feedback) filters in an intuitively readable manner.

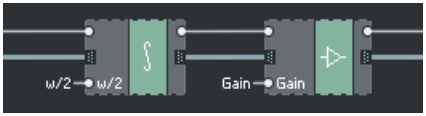
The following building blocks correspond to the fundamental elements of the continuous-time ('analog') filter block diagrams:



ZDF toolkit Macros implementing the fundamental continuous time audio processing primitives.

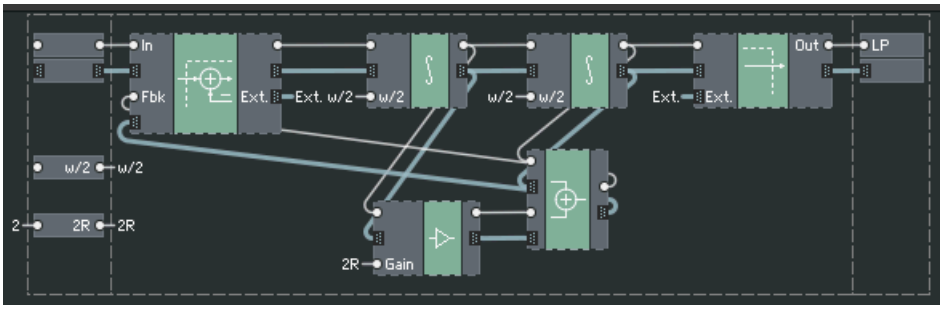
- **Integrator:** A fundamental 'analog memory element'. Roughly corresponds to the z^{-1} blocks in the digital filter block diagrams, but is not functionally equivalent to the latter. The w/2 input should be connected to the pre-warped half-cutoff value
- **Adder:** Adds to ZDF signals together.
- **Gain:** Multiplies a ZDF signal by a specified gain factor. The gain input is non-triggering, therefore this gain Macro is similar to the modulation multiplier.
- **Subtractor:** Subtracts one ZDF signal from another. A convenience shortcut for a combination of an adder and a -1 gain.

Each 'audio signal connection wire' in the ZDF toolkit needs to be implemented by two parallel wires:



A single audio path implemented by two parallel ZDF wires.

Each feedback loop needs to be implemented by a single **nonsolid** Macro. The following *SVF* Structure illustrates.



A nonsolid Structure implementing the main part of a low-pass *SVF*.

The guidelines are the following.

- Each Macro must implement exactly one feedback loop. Any other feedback loops (if existing) should be contained in the nested Macros built according to the same guidelines.
- The Macro may have exactly one audio input. This input must be directly connected to an *Input -Fbk* (as in the picture above) or an *Input +Fbk* Macro from the ZDF toolkit. This *Input ±Fbk* Macro is the point where the input signal is 'injected' into the feedback loop.

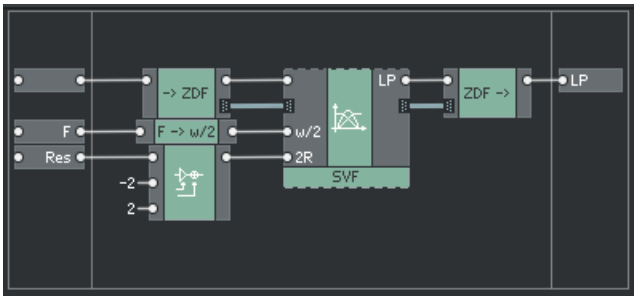
  The difference between the two Macros is simply that *Input -Fbk* inverts the feedback signal. One could use a ZDF gain Macro to invert the signal instead.

- The audio path inside the feedback loop may split and merged back (for example in the picture above the path splits after the first integrator and is merged back at the adder). However no further feedback loops are allowed, except in the nested Macros built according to the same guidelines.

  Sometimes this limitation can be worked around by treating two feedback loops as one loop 'nested' into the other. However, there are situations where the inner feedback loop is 'instantaneously unstable' (see the instantaneous instability discussion below) on its own, and only the presence of the outer feedback loop stabilizes it back. In this case such nesting decompositions will not be handled correctly by the ZDF toolkit, as it requires each feedback loop to be stable (in the instantaneous sense only) on its own, without outer stabilization factors. This is for example the reason why a non-transposed version of the Sallen–Key filter cannot be (directly, without additional manual 'hacks') implemented by the ZDF toolkit.

- The Macro may have several outputs. However each output needs to be 'implemented' by a dedicated *Output* Macro from the ZDF toolkit. The Ext input of the *Output* Macro needs to be connected to the Ext output of the *Input* Macro of the Structure.

The outermost Macro of a ZDF implementation should be (normally) a solid Macro and implement the conversion between the ordinary and ZDF audio signals (using *to ZDF* and *from ZDF* Macros from the toolkit):



The outermost Structure of a ZDF low-pass *SVF* implementation.

Besides the audio conversion, this Structure is using an *F to w/2* Macro from the ZDF toolkit to convert the cutoff in Hz to a pre-warped half-cutoff for the ZDF integrators. It also converts the resonance parameter into the damping gain 2R.

The above guidelines can be violated in principle, as long as the following hard restrictions are taken into account: Each feedback loop must have only a single input point implemented by an *Input ±Fbk* Macro and correctly employ the ZDF *Output* Macros. The guidelines simply provide a framework for following these hard restrictions.

Saturators

In order to build filters which can self-oscillate one needs to introduce saturating nonlinearities into the filters. The ZDF toolkit provides two different versions of a tanh saturator and a saturating integrator:



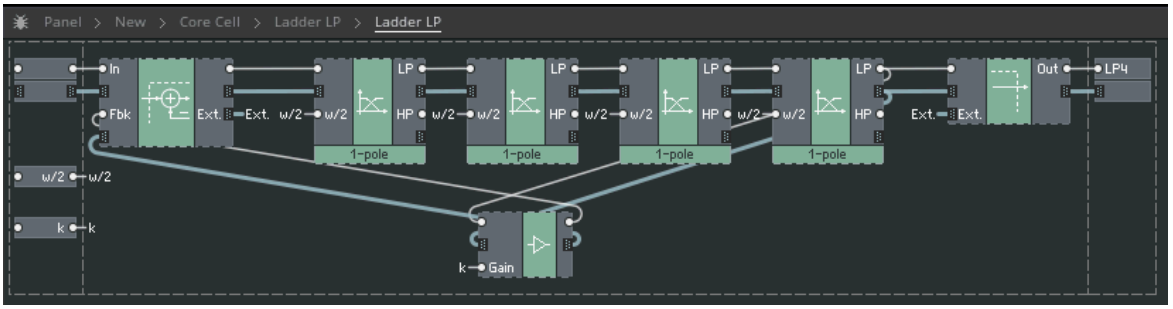
Saturating ZDF primitives.

The L input controls the saturation level (the horizontal asymptote of the saturation curve).

Filter Primitives

Often filters are used as building blocks for more complex filters. The Filter Primitives submenu of the ZDF toolkit contains some of such low-level filters.

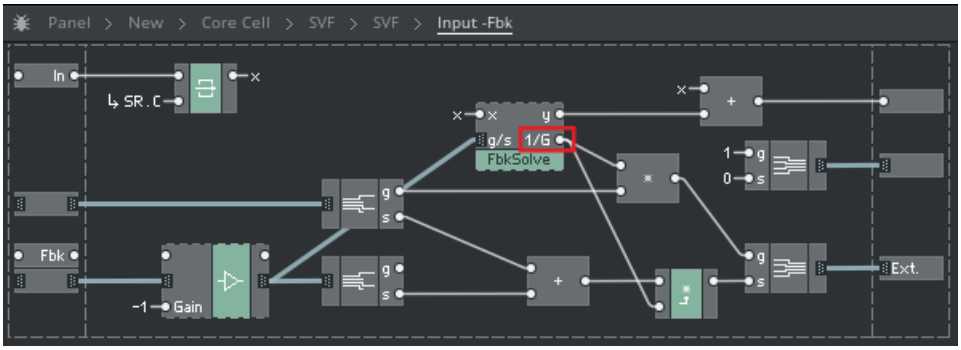
The following example illustrates how a ladder filter is built from four 1-pole filters.



Ladder filter Structure.

Instantaneous Instability

At certain extreme parameter settings (typically, in cases of excessive positive feedback) the routine application of the ZDF approach will not deliver correct results anymore. The mathematical reasons for this can be found in the corresponding literature. What this means practically is that the denominator of the zero delay feedback equation becomes zero or negative, which can be checked by monitoring the denominator value inside the feedback solver, which is located inside *Input ±Fbk* Macros:



The feedback solver inside the Input -Fbk Macro. The denominator value is available at the 1/G port of the solver.

The instantaneous instability normally happens far in the seloscillation range of the filters or in some untypical designs or usage cases. Some of the examples are the following.

- *Ladder* filter with excessive unclipped negative resonance settings (since the ladder filter employs an inverted feedback, negative resonance means positive feedback).
- *SVF* with excessive unclipped resonance settings (in the self-oscillation range the internal damping of the *SVF* is negative, which together with the inversion of the feedback signal results in a positive feedback).
- An attempt to implement a non-transposed Sallen–Key filter by feedback loop decomposition. The Sallen–Key filter has two feedback points. Since each feedback point needs to be implemented by a dedicated *Input ±Fbk* Macro, and since there can be only one such Macro per feedback loop, it is not implementable in a straightforward manner using the ZDF toolkit. One could attempt a workaround, by decomposing it into two feedback loops, one nested into the other. However, the inner feedback loop employs positive feedback, which at certain (not excessive at all) settings can become instantaneously unstable on its own. It is the presence of the outer loop which stabilizes the inner feedback again. The instantaneous instability of the inner loop results in the ill-conditioning of the entire Structure, and the filter may produce INFs and NaNs at its output and in its state as the result.

  This problem with the non-transposed Sallen–Key filter is not the limitation of the ZDF approach as such, but the one of the toolkit. In principle, with some manual 'hacking' the toolkit also could allow a non-transposed Sallen–Key filter implementation.

It is the Structure builder's responsibility to clip the filter parameters, if necessary, so that the instantaneous instability does not occur.

6.4.4 TF toolkit

The 'transfer function' toolkit (located under Library > Audio > Filter > Tranfer Func > TF Toolkit) contains the building blocks of the transfer function Macros in the Core Macro Library and can be used to modify the existing transfer function Macros or build the new ones.

The absolute majority of the Macros in the toolkit are simply implementing the complex math operations. The complex values themselves are represented by a Bundle containing *Re* and *Im* fibers.

The special 'input' and 'output' Macros of the toolkit are the following Macros.

- **s**: This Macro takes the specified frequency *f* and the specified cutoff frequency *F* and produces an imaginary value *j f / F*. This complex value specifies the point on the *s*-plane where the (unit cutoff) transfer function needs to be evaluated.



- **to Polar**: This Macro converts a complex value *z* to its absolute magnitude |*z*| and argument arg *z*. Normally used to obtain amplitude and phase responses from the frequency response value.

