



# **Automation of REAKTOR in KORE**

**-**

## **A Tutorial**

# Table Of Contents

<b>Introduction .....</b>	<b>3</b>
<b>Automation in REAKTOR.....</b>	<b>3</b>
Automation IDs as Interface between Host and Plug-in .....	3
Automation ID Management in REAKTOR .....	4
Automation Modules in REAKTOR.....	6
<b>REAKTOR in KORE.....</b>	<b>7</b>
The KORE Update of the REAKTOR Factory Library .....	7
Pre-defined Controller Pages and User Pages .....	8
Preparing an Ensemble for Use with KORE.....	9
Additional Hints .....	11

# Introduction

REAKTOR is seamlessly integrated into KORE. All factory sounds of REAKTOR 5 are included within the KoreSound database and provide Easy Access Pages and Default Pages to use the KORE controller with the REAKTOR instruments. The fundament of this integration is the capacity of REAKTOR to be automated from outside, i.e. from KORE. This document covers questions that arise if you want to create individual User Pages for REAKTOR instruments, or if you want to use your own instruments within KORE. It explains the basic mechanisms used to automate REAKTOR, and focuses on the collaboration of both applications.

Please read the tutorial completely to cover all aspects of it.

## Automation in REAKTOR

### Automation IDs as Interface between Host and Plug-in

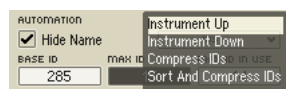
This chapter is not specific to REAKTOR or KORE. All programs that are used within a host as a plug-in have to conform to a standardized communication protocol like VST or AU. The protocol defines in which way both applications interact, e.g. in which way MIDI messages or audio signals are interchanged.

One part of those protocols deals with the control of a plug-in's sound parameters from outside. Therefore, a plug-in reports a list of parameters to the host application. The host can then send information to the plug-in to set one of the list's parameters to a specific value. This is basically the automation.

In KORE, for instance, the hardware controller tells the software that one of its buttons has been pressed by the user. The software then looks up the current Controller Page definition to determine the parameter which is assigned to that button and sends the controller's information to the plug-in to change that parameter's value.

The parameter labels, however, are only relevant for displaying information to the user; internally, the parameters are identified by numbers, the automation IDs. Most applications only display the labels, as neither those labels nor the ID values can be changed: They cannot be disconnected and are therefore identical. REAKTOR, however, provides full control over label and ID. This is a very powerful feature to create custom instruments and to specialise their integration into automation contexts.

## Automation ID Management in REAKTOR



REAKTOR 5 as a plug-in, reports one list of automatable parameters to the host application, e.g. KORE. This list is the ensemble's global parameter list; each parameter listed here can be automated from outside. It consists of the subsequent local lists of the ensemble's instruments. The integration of the instruments' lists into the ensemble's list is handled within the instruments' properties.

The **BASE ID** value represents the ID of the instrument's first parameter within the global automation list. Beginning with that ID a user-definable number of IDs is reserved for that instrument within the ensemble's list. This number can be adjusted with the instrument's **MAX ID** value. In the given figure the instrument is placed at the beginning of the ensemble's global parameter list due to its BASE ID value of 0. It then covers 285 parameter entries within the global list. The way those entries are used is specified by the instrument's local parameter list that will be explained later.

The next instrument automatically uses a BASE ID of  $\text{BASE ID} + \text{MAX ID} = 0 + 285 = 285$ . One can change the order of the instruments within the global list by using the IDS drop-down menu and selecting the **Instrument Up** or **Instrument Down** commands.

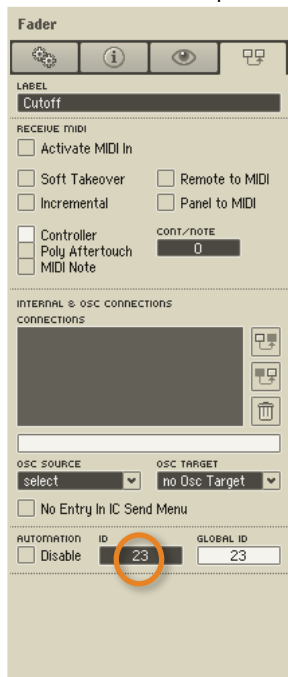
---

Here it is important to remember that the communication between host and plug-in is based on these IDs. If the order of instruments is changed also the single entries' order is altered. This, however, is transparent to the host application: It continues to send values to the parameter at a given entry although the parameter at this position has changed. Changes of the automation IDs almost always ruin existing automation connections. In KORE, for instance, the Easy Access Pages won't work anymore if an instrument is added to an ensemble, the instrument's order is changed, and so on.

---

The **MAX ID IN USE** value shows the number of IDs that are used by controllers within that instrument. If this number is higher than the MAX ID value, the instrument's list is cut at that ID; if the MAX ID is higher than the actually used number of IDs the entries within the global parameter list are left empty.

The instrument's local list is managed in all the individual controller properties. Each controller has a unique automation **ID**. It defines the position within the instruments local parameter list. Within the given figure the controller uses



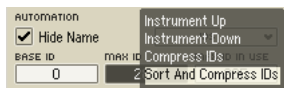
The screenshot shows the 'Fader' controller interface. It includes a 'LABEL' field with 'Cutoff', a 'RECEIVE MIDI' section with checkboxes for 'Activate MIDI In', 'Soft Takeover', 'Incremental', 'Remote to MIDI', and 'Panel to MIDI', and a 'CONT/NOTE' field with '0'. Below is an 'INTERNAL & OSC CONNECTIONS' section with a 'CONNECTIONS' list, 'OSC SOURCE' and 'OSC TARGET' dropdowns, and a 'No Entry In IC Send Menu' checkbox. At the bottom, the 'AUTOMATION' section has a 'Disable' checkbox, an 'ID' field with the value '23' circled in orange, and a 'GLOBAL ID' field with the value '23'.

the local list's 23<sup>rd</sup> entry. The **GLOBAL ID** equals the local ID plus the instrument's BASE ID (which is 0 in this case, obviously). So this controller is also placed at position 23 within the ensemble's global parameter list.

However, if the instrument's MAX ID was smaller than 23, the controller would not appear within the global list – and could therefore not be automated from a host application. Additionally, if the **Disable** box was checked the controller would keep its place within the local and global lists, but the host application could not use it. (This might be relevant is a control automates another control within REAKTOR; multiple automation can cause technical and logical problems.)

If the ID is changed manually, it can happen that the new place within the list is already occupied by another controller. In this case both controllers are exchanged. This means that changing an ID manually possibly might change not only the currently edited controller but also another controller.

The list of local IDs can become very chaotic while an instrument is built within Reaktor: Each time a new controller is created a new ID is assigned to it. Then old controllers are deleted and leave gaps within the instrument's ID list. You can re-assign the IDs manually by entering a number into the ID field, but this might be time-consuming for large instruments. Therefore you can also use the **Compress IDs** and **Sort and Compress IDs** commands that can be found within the instrument's properties.



Compressing the IDs results in filling all gaps within the local list. For instance, if the IDs 0 to 5 and 8 to 10 are used by controllers while the list's 6<sup>th</sup> and 7<sup>th</sup> entry is empty, compressing the list shifts

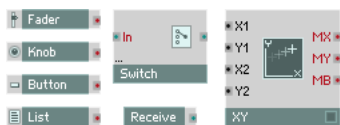
the controllers 8, 9 and 10 to the entries 6, 7 and 8.

Sorting the IDs also places all controllers that are located within one REAKTOR macro at subsequent entries. Within the macro the entries are ordered alphabetically according to the controllers' labels.

Both actions completely re-organize the instrument's local automation list and have to be used carefully. A parameter can become non-automatable if its ID is above the instrument's MAX ID value afterwards. Even more important, any existing automation based on the old IDs will be broken, similar to the process described above regarding the Instrument Down and Instrument Up commands.

## Automation Modules in REAKTOR

Only a few number of REAKTOR's modules can be automated. This is parallel to the fact that only some modules are displayed by the graphical interface. See the figure below for a list of the available automation modules. They can all be found in the Panel section of the module list.



Except the XY module all of the modules displayed react similar to automation messages. Like MIDI, automation messages from a host have a fixed range of values. This range is mapped automatically onto the module's

internal range. For instance, imagine a **fader** or a **knob** that ranges from 0 to 15. To set this controller to its highest possible value, the host sends the highest possible automation message – and not 15. If the internal range is smaller than the external range, several external values are mapped onto one internal value. This is also true if the module can only be set to two states, like a **button**: All automation messages below the middle automation value set the button to its default state; all messages above the middle invert its state. If the module provides more than two states, but not a continuous range (like the **switch**, **receive** and **list** modules), a similar “quantization” takes place. The continuous automation messages are divided into a given number of sub-ranges according to the module. For example, if a switch module provides 3 buttons, low automation values will set it to the first state, middle automation values will change to the second state, and high values will activate the third state.

The **XY module** is particular. It automatically reserves two entries in the local and global automation lists, to automate its horizontal value as well as its vertical value. The X settings can be controlled with the first entry whose ID can be set within the module's properties. The Y setting is controlled by the subsequent ID, so there is no independent ID control. However, if the ID values are edited manually, the vertical ID can be overwritten by another controller without exchanging both controllers completely. (This might be subject to change in future versions of REAKTOR.)

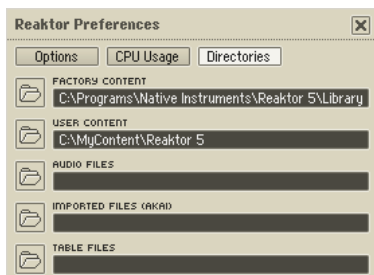
There are also modules that can be displayed on the panel without being subject to automation, e.g. the **mouse area** and the **multi picture** module, used by some instruments to build customized panel controls. Parameters based on these modules cannot be automated and need a customized automation control, parallel to the customized snapshot value management.

## REAKTOR in KORE

### The KORE Update of the REAKTOR Factory Library

The Factory Content of REAKTOR 5 has been updated for compatibility with KORE. The updated Library is installed with KORE. It will also be part of all future updates of REAKTOR 5, e.g. it is already included in the NI KOMplete 3 package.

The update aims to make automation possible for every parameter in all ensembles. Technically speaking, the MAX ID value has been increased for every ensemble to its MAX ID IN USE value. The sort and compress commands have not been used to preserve the original parameter IDs that might be used in any given automation context. If a parameter was excluded from automation by a checked Disable switch this was not changed: The option was activated in the original ensemble for technical reasons that did not change. Also, some ensembles use the mouse area module as customizable controller (e.g., most of the sequencers): These are excluded from the update, as they cannot be automated.



The KoreSound files don't contain the complete ensembles. They reference them relatively to the paths set within the REAKTOR preferences. If the ensemble is moved after the KoreSound and its reference is saved, the KoreSound won't find the ensemble. (This is similar to a sampler that needs its samples at a pre-defined place.)

---

To keep the Factory Content working, don't move the files. This is also true for custom KoreSounds and custom ensembles.

---

## Pre-defined Controller Pages and User Pages

All **pre-defined controller pages** of a KoreSound are based on the Factory Content ensembles' automation lists. If these lists are changed by the user the pre-defined controller pages won't work correctly.

---

All ensembles within the Factory Content folder of REAKTOR must not be changed to ensure proper interaction of KORE and REAKTOR.

---

The pre-defined controller pages are loaded when a plug-in is loaded into KORE. For nearly all applications this is the perfect solution: FM7, for instance, will present the same list of parameters every time it is loaded into KORE, and thus this forms a stable fundament for the controller pages. REAKTOR, however, presents parameter lists that can change during operation, simply by loading a different ensemble. This complicates the usage of pre-defined controller pages.

The preferred way to use REAKTOR within KORE is loading a KoreSound based on a REAKTOR ensemble.



Genre	Meta Information	
Avantgarde	Author	Stephan Schmitt
Orchestral/Classical	Company	NI
Film Music	Bankname	SoundSchool Analog
Ambient/Electronic	Color	White
Drum&Bass/Breakcore	Rating	●○○○○○
House	Comment	
Techno/Electro		
Industrial		
Dance/Trance		
HipHop/Downbeat		
Funk/Soul		
Reggae/Dub		
Latin/Afro-Cuban		
Rock		
Pop		
Jazz		
Folk/Country		
Ethnic/World		
	Plugins	Reaktor5

Each KoreSound contains a field called **Bankname** in its Meta Information. Here the name of the ensemble file is placed which the KoreSound uses within REAKTOR. This entry determines the pre-defined controller pages to be loaded when loading the KoreSound. If REAKTOR is loaded blankly without using a KoreSound (e.g. by using the Browser's Plugins tab) this information is not present, and the pre-defined pages cannot be loaded. If the ensemble (SoundSchool Analog in the given figure) is then loaded manually within REAKTOR, REAKTOR cannot tell this to KORE due to limitations of the VST/AU protocols.

---

So, if a new KoreSound, built from scratch with a Factory Content's ensemble, is meant to use the pre-defined controller pages of that ensemble, the ensemble file's name has to be entered manually into the Bankname field of the KoreSound.

---

Custom **User Pages**, however, are always part of the KoreSound, they are not saved with the ensemble or its snapshots. They can only be re-loaded if the KoreSound is loaded – re-assembling the sound by subsequently loading REAKTOR, the ensemble and the snapshot won't recall any User Page.

## Preparing an Ensemble for Use with KORE

The ensembles that can be downloaded from the REAKTOR User Library have not been optimized for KORE. New ensembles built from scratch are also not optimized for KORE automation, of course. With the techniques described above, preparing an ensemble for use in KORE is straightforward.

Imagine an ensemble that has been downloaded from the User Library. If its builder designed it for use with REAKTOR standalone (where automation is

of no importance) all automation parameters will be at random values. The incorporated controls like faders and buttons will fill the local parameter lists in the order of their creation, and the MAX ID value will have no relation to the number of available parameters. If there are several instruments within the ensemble, the order of the local lists within the global list might also be not optimal. The automation has to be adapted to the needs of the particular ensemble. This should be done in REAKTOR standalone, for convenience.

As a first step, the local list(s) of the instrument(s) have to be prepared. Execute the **Sort and Compress IDs** command from the instrument's properties. Then set the **MAX ID** value to the number of MAX ID IN USE value to make sure that no parameter is masked from the global list.

After doing this for each instrument within the ensemble use the **Instrument Up** and **Instrument Down** commands to position the instruments within the global list. Perhaps it might be useful to use the ensemble's signal flow as a template for the instrument's order within the global parameter list. Finally check whether the most important parameters use **GLOBAL ID** values below 1000, as this is the maximum number of parameters supported by most plug-in protocols, e.g. VST.

Here, once more it is important to remember that changing the IDs causes incompatibility to prior automation assignments within a host. The changed ensemble should be saved with a **new filename**.

Now close REAKTOR standalone and start KORE. Create a new Source Channel in the Performance Level, switch to the Sound Level, create a new Source Channel there, and load REAKTOR into its main slot. Open the plug-in interface and load the prepared ensemble. With the plug-in interface open, switch to KORE's Global Controller, create a new User Page, and press the Assign Button. While touching one of the hardware's controllers move one of the ensemble's controllers with the mouse. This connects both elements within the User Page. Continue linking controls, or click the Assign button again to leave this mode. A dialog with further options will open. Please refer to the KORE manuals for details on these options.



As mentioned above, User Pages are saved with the KoreSound – not with the ensemble. Go to the File menu and select the Save Sound As entry; perhaps you might want to add attributes to the KoreSound first. By re-opening this sound, you can always return to your custom REAKTOR automation.

Remember that the KoreSound references the ensemble file. If the path to the ensemble is changed, KORE won't find it.

## Additional Hints

Although this document is about automation, some advanced general information concerning the interaction of REAKTOR and KORE will be listed shortly. This information is based on the KORE manual where you can find detailed explanation.

- The **sensitivity** of the KORE controller's knobs can be adjusted individually. Within the Assignment Properties dialog, the Full Range can be controlled in degrees. (For instance, if you want to fade a value from 0% to 100% when the knob is turned one time, use a value of 360. If you want a fine resolution, the value can be set to 720, so it has to be turned twice before the full range is reached.) In REAKTOR, this movement is mapped again, depending on the automated parameter. If a fine resolution is set within KORE, the controller within REAKTOR should also provide a fine resolution (i.e. a high number of steps) to make use of the detailed controller data.
- REAKTOR provides a long list of **inputs**. Currently, KORE only supports the first pair of them. This is sufficient to use REAKTOR as a powerful stereo effect unit, e.g. placed within a send channel.
- REAKTOR provides a long list of **outputs**. KORE can connect to eight of them. When REAKTOR is loaded as the main plug-in of a KoreSound, go to the Sound level, open the context menu by right-clicking on the channel's header (command-click on the Mac) and select Add Extra Output. A new source channel is added which is linked to REAKTOR's output ports 3 and 4. This can be repeated until four separate stereo source channels are available.
- The MIDI Player of KORE is a powerful feature. Each Player can have its individual, looped Clock Track to which the **song position** module of REAKTOR is connected.