

# StudioFactory 0.42

## Software based desktop audio/video studio

### User & Reference Manual

Peter Wendrich  
pwsoft@syntiac.com

February 25, 2007

## Contents

<b>1</b>	<b>Introduction</b>	<b>5</b>
1.1	The desktop audio/video studio . . . . .	5
1.2	Downloading and installing StudioFactory . . . . .	5
1.3	Quick tour . . . . .	5
1.4	Midi routing and ports . . . . .	5
<b>2</b>	<b>Program components</b>	<b>6</b>
2.1	The Project Browser . . . . .	6
2.2	Time line . . . . .	7
2.2.1	Tracks . . . . .	7
2.2.2	Patterns . . . . .	8
2.2.3	Songs . . . . .	8
2.2.4	Phrases . . . . .	8
2.3	Midi event editor . . . . .	8
2.4	SynFactory patch editor . . . . .	8
2.4.1	Introduction . . . . .	8
2.4.2	Adding modules . . . . .	8
2.4.3	Connecting the cables . . . . .	8
2.4.4	Changing values . . . . .	9
2.4.5	Moving modules . . . . .	9
2.4.6	Scrolling the screen . . . . .	10
2.4.7	Scrolling with zoom out . . . . .	10
2.4.8	Adding labels . . . . .	10
2.4.9	Mouse and keyboard control . . . . .	10
2.5	Configuration and settings . . . . .	11
2.5.1	Language . . . . .	11
2.5.2	Display patch cables as . . . . .	11
2.5.3	Colors . . . . .	11
2.5.4	Audio input device . . . . .	11

2.5.5	Audio output device	11
2.5.6	Number of buffers	11
2.5.7	Buffer size	12
2.5.8	Latency	12
2.5.9	Midi input devices	12
2.5.10	Midi output devices	12
<b>3</b>	<b>Menus</b>	<b>12</b>
3.1	File menu	12
3.1.1	New...	12
3.1.2	Open... (Ctrl+O)	12
3.1.3	Add/Merge	12
3.1.4	Close	12
3.1.5	Close Project	12
3.1.6	Save (Ctrl+S)	13
3.1.7	Save as	13
3.1.8	Save All	13
3.1.9	Exit program	13
3.2	Edit menu	13
3.2.1	Cut (Ctrl+X)	13
3.2.2	Copy (Ctrl+C)	13
3.2.3	Paste (Ctrl+V)	13
3.2.4	Delete (Del)	13
3.2.5	Select All (Ctrl+A)	13
3.2.6	Clone (Ctrl+D)	13
3.2.7	Select cable color	14
3.3	View menu	14
3.3.1	Project Browser (F2)	14
3.3.2	Color Selector	14
3.3.3	Transport Control	14
3.3.4	Audio Output Scope	14
3.3.5	Audio Mixer	14
3.3.6	Video Preview	14
3.3.7	Midi Scope	14
3.3.8	Midi Activity Monitor	14
3.3.9	Console and messages window (F11)	14
3.4	Play mode menu	14
3.4.1	Rewind	14
3.4.2	Stop	14
3.4.3	Play	14
3.4.4	Record	14
3.4.5	DC Blocking Filter	15
3.4.6	DSP runtime compiler	15
3.5	Configuration menu	15
3.5.1	Global settings	15
3.5.2	Patch settings	15

3.5.3	Save Settings As . . . . .	15
3.6	Help menu . . . . .	15
3.6.1	Context help (F1) . . . . .	15
3.6.2	Editor help (Shift+F1) . . . . .	15
3.6.3	Menu help . . . . .	15
3.6.4	Introduction . . . . .	15
3.6.5	About StudioFactory . . . . .	16
<b>4</b>	<b>Building patches</b>	<b>16</b>
<b>5</b>	<b>SynFactory Modules</b>	<b>16</b>
5.1	Generators . . . . .	16
5.1.1	OSC - Multifunction oscillator . . . . .	16
5.1.2	MOS - Mini oscillator . . . . .	17
5.1.3	RND - White and colored noise source . . . . .	17
5.1.4	Formanzzz - The formant oscillator . . . . .	17
5.1.5	SineBank - Multiple octave Sine wave oscillator . . . . .	18
5.1.6	PerlinOsc - Perlin-noise based oscillator . . . . .	18
5.1.7	PM Osc - Oscillator with phase modulation . . . . .	19
5.1.8	Quad Osc - Quadrature oscillator with phase modulation . . . . .	20
5.2	Filters and amplifiers . . . . .	20
5.2.1	MMF - Multi Mode Filter (12/18/24 dB) . . . . .	20
5.2.2	OB VCF - 12dB OB-type filter . . . . .	21
5.2.3	ECF - Envelope Controlled 12 dB OB-type Filter . . . . .	22
5.2.4	VCA - Amplifier with variable gain . . . . .	22
5.2.5	ECA - Envelope Controlled Amplifier . . . . .	23
5.3	Modulation . . . . .	23
5.3.1	CTR - Signal controller . . . . .	23
5.3.2	ENV - ADSR Envelope generator . . . . .	24
5.3.3	AHD - Attack/Hold/Decay envelope generator . . . . .	24
5.3.4	S/H - Sample and Hold . . . . .	25
5.3.5	GLI - Glide/Portamento . . . . .	25
5.3.6	TRIG - Delayed gate/trigger/one-shot . . . . .	26
5.3.7	SEQ - 8 steps sequencer . . . . .	26
5.3.8	SEQ - 16 steps sequencer . . . . .	26
5.3.9	BinSeq - Binary input 8 steps sequencer . . . . .	27
5.3.10	Scanner - 9 input analog style multiplexer . . . . .	27
5.4	Math modules . . . . .	28
5.4.1	ADD - 2-8 input adder . . . . .	28
5.4.2	MUL - Multiplier/Ringmodulator . . . . .	28
5.4.3	SUB - Subtractor . . . . .	28
5.4.4	SPLIT - Positive and negative signal splitter . . . . .	29
5.4.5	BETWEEN - Window comparator . . . . .	29
5.5	Logic Modules . . . . .	30
5.5.1	OR - 2-8 input logic OR gate . . . . .	30
5.5.2	AND - 2-8 input logic AND gate . . . . .	30

5.5.3	NOT - Logic NOT gate . . . . .	30
5.5.4	XOR - Logic XOR gate . . . . .	31
5.5.5	ADC - Digitize signals (9 bits resolution) . . . . .	31
5.5.6	DAC - Binary to signal (9 bits resolution) . . . . .	31
5.5.7	Pulsar - Pulse generator/clock divider . . . . .	31
5.5.8	BIN - 8 bit Binary counter . . . . .	32
5.5.9	DeMux - 1 to 2 demultiplexer . . . . .	32
5.5.10	DeMux - 2 to 4 demultiplexer . . . . .	33
5.5.11	DeMux - 3 to 8 demultiplexer . . . . .	33
5.5.12	DeMux - 4 to 16 demultiplexer . . . . .	33
5.6	Song Modules . . . . .	34
5.6.1	SCRIPT - Script controlled note player . . . . .	34
5.7	Effect Modules . . . . .	35
5.7.1	FLA - Flanger . . . . .	35
5.7.2	DLY - Delay line . . . . .	36
5.7.3	SyncDly - Syncable Delay line . . . . .	36
5.7.4	BIT - Bit reducer This module is also known as a quantizer. . . . .	37
5.7.5	Clippy - NoiseGate / Clipper / Distortion . . . . .	37
5.7.6	PAN - Signal controlled panner . . . . .	38
5.8	Mixer modules . . . . .	38
5.8.1	MIX - 3 channel Mixer . . . . .	38
5.8.2	MIX - 4 channel Mixer . . . . .	39
5.8.3	MIX - 8 channel Mixer . . . . .	39
5.8.4	CROSS - Cross fade . . . . .	40
5.8.5	MXO - Output with mixer control . . . . .	40
5.8.6	OUT - Output . . . . .	41
5.8.7	OUT - Output with surround . . . . .	41
5.8.8	PANOUT - Output with panning control . . . . .	41
5.8.9	INPUT - Audio Input . . . . .	42
5.9	Interface Modules . . . . .	42
5.9.1	Joystick - Joystick controlled module . . . . .	42
5.9.2	Scope - 4 channel scope . . . . .	42
<b>6</b>	<b>Midi implementation chart (SynFactory Patch)</b>	<b>44</b>
<b>7</b>	<b>File Formats</b>	<b>45</b>
7.1	SYN file format (SynFactory 1.x) . . . . .	45
7.1.1	Introduction . . . . .	45
7.1.2	Layout . . . . .	45
7.1.3	Header . . . . .	46
7.1.4	Filepath section . . . . .	46
7.1.5	Settings section . . . . .	46
7.1.6	Windows section . . . . .	47
7.1.7	Samplebank section . . . . .	47
7.1.8	Samples section . . . . .	47
7.1.9	Sample section . . . . .	48

7.1.10	Sampledata encoding . . . . .	48
7.2	STF file format (StudioFactory) . . . . .	48
7.2.1	Introduction . . . . .	48
7.2.2	Layout . . . . .	48
7.2.3	Encoding . . . . .	48
7.2.4	Header . . . . .	49
7.2.5	StudioFactory configuration data . . . . .	49
7.2.6	Project data . . . . .	49
7.2.7	Sample and instrument data . . . . .	49
7.2.8	SynFactory patch data . . . . .	49
7.2.9	Bitmap data . . . . .	50
7.2.10	Sequencer data . . . . .	51
7.2.11	Embedded file data . . . . .	51
7.3	BMP file format (bitmap) . . . . .	51

## 1 Introduction

### 1.1 The desktop audio/video studio

StudioFactory is based on SynFactory (the modular software synthesizer) and can load any file created with it. In addition to editing and playback of virtual analog synthesizer patches, the system has a very advanced MIDI sequencer with audio support. The output of the sequencer can be send to the internal synthesizer or to external MIDI equipment. The SynFactory based synthesizer is extended to support multi timbral and polyphonic playback. There is no predefined program limitation on the number of tracks or synthesizer patches active in a single project (of course there is a limit what the computer can handle).

### 1.2 Downloading and installing StudioFactory

### 1.3 Quick tour

### 1.4 Midi routing and ports

StudioFactory has many features to work with MIDI (Musical Instrument Digital Interface) messages. The MIDI protocol allows electronic devices (usually synthesizers, but also computers, light show controllers, VCRs, multi-track recorders, etc.) to interact and work in synchronization with other MIDI compatible devices.

Each MIDI connection supports up to 16 channels, communication goes in a single direction. Each channel has it's own instrument attached with it's own note and control events. The timing pulses and clocks when using a sequencer is global for all 16 channels however. Because most of the modern MIDI keyboards have many voices (62+) and are multi timbral (can play multiple sounds at the same time), a single MIDI port can become insufficient quite fast.

Depending on the setup a single midi connection can become too limited. Therefore StudioFactory can support up to 26 ports, each port has its own midi clock and 16-MIDI channels for a total of 416 channels and 26 clocks. MIDI cables run in a single direction and can only support one connection, so to fully connect an instrument 2 cables are necessary. The ports in StudioFactory are different and support multiple sources, so multiple devices can drive the same port and multiple devices can listen on the same

port. This allows the same features as on a MIDI-patchbay, like merging and distributing signals from multiple sources to multiple destinations without any limits (up to the maximum of 416 channels and 26 clocks). For example to make a 'soft-thru' connection, specify the same port on both the MIDI-In and the MIDI-Out and all messages received on the input port will be re-transmitted to the output port.

Ports are identified with a single capital letter. Two numbers indicate the channel when required, so A01 is the first channel on port A. Because StudioFactory doesn't limit the direction of messages in one direction, all MIDI-in and MIDI-out ports should be connected to their own port unless a permanent soft-thru is required. Therefore in practice the maximum number of channels which can be used is less as the absolute maximum of 416 (unless every channel on every MIDI-in and -out port is used, which is very unlikely).

For specific routing and merging operations the MIDI-router component can be activated. This component supports filtering and translation of MIDI-events. The MIDI activity monitor and MIDI scope components can be used to get information about the MIDI-events flowing inside StudioFactory.

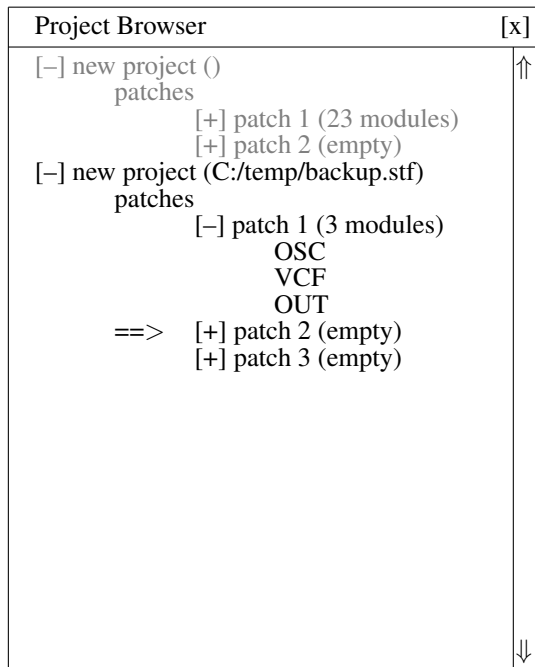
## **2 Program components**

### **2.1 The Project Browser**

The project browser gives a overview of all projects and files loaded into StudioFactory. There is only one project active at the same time. This project is shown in black the inactive projects are light gray.

One project can contain many other objects and sub-projects. By clicking on the [+] icon the project can be unfolded and the contents of the project is shown. Items from the unfolded view can be selected. The currently selected items are highlighted in yellow. Multiple items can be selected by holding the Ctrl key while making the selection. Also items, modules and tracks can be selected from the associated editor screens. The selection made there is reflected in the project window also.

Pressing DEL while the project browser is the current window will delete all selected items. This can include modules, samples but also complete projects. The clipboard shortcuts Ctrl+X, Ctrl+C and Ctrl+V for cut, copy and paste can also be used in the project browser.



## 2.2 Time line

### 2.2.1 Tracks

The time line shows a number of tracks. Each track supports one task or activity at a time (when one activity is finished on a track, another one can start). These tasks or activities can be one of the following:

- Playing a sample.
- Playing MIDI notes and other MIDI data.
- Playing a video clip.
- Showing a picture.

Each track has a stereo audio output, a midi port and a video layer attached. The audio output can be connected either to the audio mixer or to a SynFactory module in the patch editor. The module has a left and right audio output and a few control signals. During sample playback there are some extra commands available used for emulating MOD-tracker like effects. The commands control both the sample being played and the extra outputs on the SynFactory module.

If the track is used for MIDI playback the MIDI data is send to one of the 26 internal MIDI ports. This way the tracks can control either the patches or external equipment (or a combination of the two). The track has a channel overrule option so MIDI data is send to this channel even if the original stream was directed to other channels.

### 2.2.2 Patterns

### 2.2.3 Songs

Songs are buildup from a sequence of patterns.

### 2.2.4 Phrases

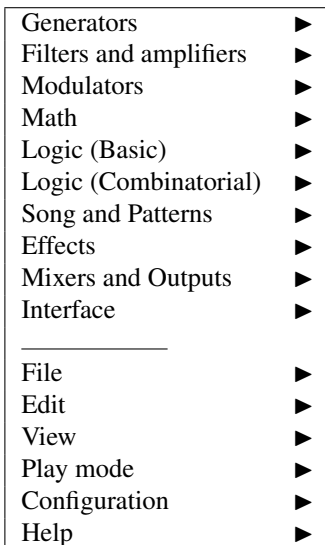
## 2.3 Midi event editor

## 2.4 SynFactory patch editor

### 2.4.1 Introduction

### 2.4.2 Adding modules

Modules are added by pressing the right mouse button (or pressing the 'enter' key on the keyboard). A popup menu becomes visible. By selecting a module from the menu it is inserted in the patch on the location where the mouse was on the moment the right mouse button (or 'enter' key) was pressed.

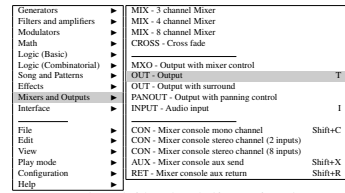
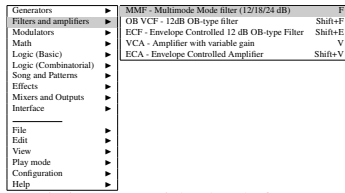
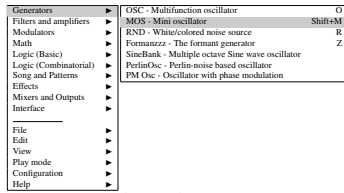


The modules are grouped together depending on the main function of the particular module. There are currently ten groups defined. Of course the grouping is only an indication and actual use is not limited in any way. For example the delay lines are found in the effects group because they are normally used for making echo and reverb, but for physical modelling they can be used as oscillators.

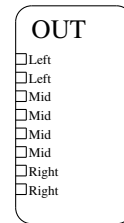
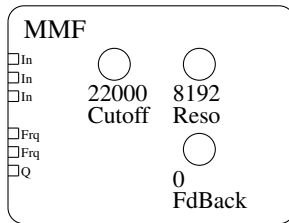
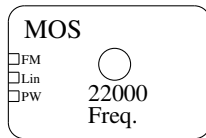
There are about 80 different modules available in the system, see chapter 5 for a complete overview of the modules. There is no limit for the number of any particular module. Any combination of modules can be added to a patch until the computing power is used up or the program limit of 1022 modules is reached. The 1022 module limit is for a project (all patches together).

### 2.4.3 Connecting the cables

First we insert some modules to play with. Select a MOS from the oscillator menu, a MMF from the filter menu and a OUT from the Mixers and Outputs menu.



Now select the Saw output of the Mini OSC with the left mouse button and while holding it drag a cable to one of the In ports of the MMFilter and release the mouse button. An other way to connect cables is clicking on the output (it will become black) releasing the mouse button and clicking on an input. If the input already has a connection it will be disconnected.



!!!

Now in a similar way connect the MMFilter LP12 or BP outputs to the OUT module. Which output to which input is at this stage not really important. It can always be changed at a later time. The result so far should look something like this:

!!!

Disconnecting a cable is done by clicking left on the place where it is connected to an input. A single output can be connected to any number of inputs, but an input can only be connected to one output at a time. As can be seen in the picture the MMFilter has multiple In ports to make it easier for connecting multiple oscillators to it. If three ports is not enough however an ADD module can be placed before the MMFilter.

## 2.4.4 Changing values

Changing the values of the virtual knobs is done with the mouse. Click and hold the left mouse button when the mouse is above a knob. Move the mouse up or down to increase or decrease the current value. If the value changes too fast, pressing and holding the right mouse button (together with the left button) while moving the mouse up and down changes the value 16 times slower.

The value of the knob can be set to zero by pressing left mouse button and at the same time double clicking with the right button.

## 2.4.5 Moving modules

Press and hold down the left mouse button when the cursor is above the name of the module (the part that turns white on selected modules). Moving the mouse moves the module to a new location. When multiple modules are selected, holding down the Ctrl key allows moving all selected modules at the same time.

## 2.4.6 Scrolling the screen

Pressing and holding the right mouse button makes it possible to scroll the screen with the mouse. This works also when dragging or selecting modules or drawing cables. Just press the right mouse button while holding the left down also. This is very useful when the modules which you want to connect are far away from each other.

## 2.4.7 Scrolling with zoom out

When scrolling with only the right mouse button pressed, pressing the left mouse button together with the right enables the overview mode. The screen zooms out 5 times in both horizontal and vertical direction. This enables a good overview of the patch and makes it easy to navigate fast across big patches.

Because this overview uses both mouse buttons it is not available when dragging cables or drawing the selection rectangle. Only the normal scrolling is possible in this situation.

## 2.4.8 Adding labels

To make navigating in large patches easier and make identification of your modules easier, each module can have a label. Select the module(s) you want to give a label and select 'hide/show label' from the 'edit' menu. The label is displayed above the module. Text can be entered in the label by typing text while the mouse is hovering above the label. If the label is hidden (by selecting the menu option) the text will be saved. So if the label is showed later, the stored label text will be restored. The label will follow the module when it is moved. Cloning the module(s) will also clone the label text.

## 2.4.9 Mouse and keyboard control

The table below gives an overview of all the mouse button combinations and their usage within the patch editor.

Mouse operation	Effect
Left click on module	Select module
Ctrl key + left click on module	Select multiple modules
Left double click on module	Open extra edit window (only for some modules)
Left click and hold on module name	Select and move module
Ctrl key + left click and hold on module name	Move multiple modules
Left click and hold on background in editor	Select all modules inside the rectangle
Ctrl key + left click and hold on background	Add all modules inside rectangle to selection
+ Right click and hold	Scroll screen until right mouse button release
Left click on output	Select output as active (for new connections)
Left click on output, hold and release above an input	Drag new cable from output to input
+ Right click and hold	Scroll screen until right mouse button release
Left click on not connected input	Connect cable from active output to this input
Left click on connected input	Disconnect cable
Left click and hold on knob	Change value
+ Right click and hold	Change value more slowly
+ Right double click	Reset value to zero
Right click	Access popup menu
Right click and hold	Scroll view
+ Left click and hold	Zoom out 5 times (large overview)
Hover over knob. Type 0-9, '-' or backspace	Change value/position of knob.
Hover over knob. Type '[' , ']' , '{' or '}'	Increase/decrease value with 1 ('[' ']'') or 10 ('{' '}'')
Hover over label. Type characters or backspace	Change label.

## **2.5 Configuration and settings**

This window shows all configuration settings and makes it possible to change them. The changes take effect immediately so there is no confirmation box. All the settings can be changed at any time. Even when audio playback is active.

### **2.5.1 Language**

Sets the language the program uses for menus and help texts. Not all languages are fully translated. English texts will be used for the parts that are not translated yet.

### **2.5.2 Display patch cables as**

Selects the way connections between modules in the patch editor are displayed. Black lines will show all connections as thin black lines unaffected by the cable color. Colored lines shows the connections as straight lines in the color that was selected when cable was drawn. The setting virtual patch cables will render all connections as hanging cables, giving the patches an analog synthesizer look.

### **2.5.3 Colors**

Some of the colors that are used in StudioFactory can be customized. The color selection is done with 3 bars. The top bar controls the amount of red, the middle the amount of green and the bottom bar the amount of blue. Black lines below the color bars show the currently selected amount of red, green and blue. The bars are divided in 16 blocks and each block have the color that will result when the block is selected with the mouse. So moving the black line in the red bar (by clicking on one of the blocks) changes the green and blue bars to show the color that will result when the black lines are moved in these bars. The bar blocks which have the black lines below them show the currently active color.

If the black lines are directly above each other, the result is an amount of gray (black when fully left or white when fully right).

### **2.5.4 Audio input device**

Selects the device used for audio input. This can be on the same audio card used for output if the card is full duplex or it can be on a different audio card. The audio input stream is synced to the audio output. If the input and output sample frequencies are slightly different this can create small artifacts in the input stream. Therefore StudioFactory tries to match the streams over a longer period of time to reduce these artifacts. It is recommended to stop audio playback before changing this setting.

### **2.5.5 Audio output device**

Selects the device used for audio output. The use of the DirectSound driver is recommended because of the lower latency attainable with this driver. The WaveMapper setting will use the default audio device assigned by Windows.

### **2.5.6 Number of buffers**

Sets the number of buffers used for sending audio data to the sound card. Observe the latency values for determining the effect of this setting.

### **2.5.7 Buffer size**

Sets the size of the buffers used for sending audio data to the sound card. Observe the latency values for determining the effect of this setting.

### **2.5.8 Latency**

The latency readout shows the effect of the 'number of buffers' and 'buffer size' configuration settings. It shows the maximum delay between making changes in the patch and hearing it at the audio output. A low latency gives smoother operation, but is especially important when controlling StudioFactory from external MIDI equipment. Two values are shown. The first value is the latency of a single audio buffer and the second value (if present) is the maximum total latency of all buffers. The latency becomes better by lowering the number of buffers and/or the buffer size. The attainable latency depends on the system configuration. On modern systems it can be as low as 50 ms for the WaveMapper and 15 ms for the DirectSound (sf\_dsound.dll) driver. Using many small buffers is better as having a few big ones.

### **2.5.9 Midi input devices**

This is a list with all MIDI-hardware and software drivers in the system. For each interface there is a selection which StudioFactory port it should use. When set to [off] the device is not used. All MIDI data received on this device is ignored by StudioFactory. Setting it to one of the ports (A-Z) opens the device and all received data is sent to the internal port (only exception is sysex data which is not supported). The mapping between real devices and device drivers and the internal ports of StudioFactory makes it possible to design synthesizer patches and sequences and play them on complete different hardware configurations without changing the patches.

### **2.5.10 Midi output devices**

Midi output is not supported in the current development version.

## **3 Menus**

### **3.1 File menu**

#### **3.1.1 New...**

#### **3.1.2 Open... (Ctrl+O)**

Load a file from disk and create a new project for it's contents.

#### **3.1.3 Add/Merge**

Load a file from disk and add content to the current active project.

#### **3.1.4 Close**

#### **3.1.5 Close Project**

Remove the active project from memory.

### **3.1.6 Save (Ctrl+S)**

Store the project, overwriting original file used to load the project. If it is a project created with the 'new' command no filename is assigned yet, so a 'Save as' is executed instead.

### **3.1.7 Save as**

This command is used to save the current project under a new name. If the new name already exists a confirmation box will be displayed asking if the old file should be replaced with the new one.

### **3.1.8 Save All**

Save all modified projects. For each new project a 'Save as' is executed.

### **3.1.9 Exit program**

This will exit the program (same as clicking on the 'X' on the right top of the patch window). The current projects are automatically stored in a file called 'autoload.stf' together with the current program settings. This file is reloaded on startup of StudioFactory. This makes it possible to leave at anytime without losing any unsaved work.

## **3.2 Edit menu**

### **3.2.1 Cut (Ctrl+X)**

Move all selected items to the clipboard. The items will be removed from the project(s).

### **3.2.2 Copy (Ctrl+C)**

Place a copy of all selected items on the clipboard.

### **3.2.3 Paste (Ctrl+V)**

Place a copy of the modules stored on the clipboard to the current project or patch.

### **3.2.4 Delete (Del)**

Remove the selected item(s) from the project(s). All cables running to and from deleted module(s) are also removed from the patch.

### **3.2.5 Select All (Ctrl+A)**

Select all modules in the current patch.

### **3.2.6 Clone (Ctrl+D)**

Clone all selected modules. This duplicates the modules inclusive all the settings of the controls and the wiring within the set of selected modules. The newly created module(s) are selected (so they can be moved to a new location together. Hold CTRL key while dragging to keep the modules selected).

### **3.2.7 Select cable color**

This selects the color used for new patch cables. The cables already connected will keep their color. Its like you have eight boxes with patch-cables sorted on color and select which box to use. This menu is only available if 'Patch cable display' in the config-submenu is NOT set to mode 0.

## **3.3 View menu**

### **3.3.1 Project Browser (F2)**

### **3.3.2 Color Selector**

### **3.3.3 Transport Control**

### **3.3.4 Audio Output Scope**

### **3.3.5 Audio Mixer**

### **3.3.6 Video Preview**

### **3.3.7 Midi Scope**

### **3.3.8 Midi Activity Monitor**

### **3.3.9 Console and messages window (F11)**

## **3.4 Play mode menu**

### **3.4.1 Rewind**

Starts playback from the beginning. All sequencers and synthesis modules are reset to the default value and start position. This button can be pressed both in Stop and Play state.

### **3.4.2 Stop**

Stops playback. The scope window freezes on the last generated audio information.

### **3.4.3 Play**

Starts playback of the generated audio.

### **3.4.4 Record**

Opens a file dialog to set the filename and type of the capture file. Possible recording file types available depend on the installed drivers. StudioFactory supports WAV with 16 bits resolution and 44.1 KHz sampling rate, MP3 at 128 kbit/sec or OGG-Vorbis. StudioFactory uses external DLLs for MP3 and OGG recording so these need to be present and correctly installed. Capturing will start immediately after closing the dialog with OK. If the filename already exists a confirmation box will appear asking if the original file must be overwritten. During capture the sound will also be played through the sound card. Selecting this option again will close the capture file and stops further capturing. If the record button (on the playback control window) and this menu is not 'checked' after selection of the filename, an error has occurred with opening the file (either the file could not be created or the required external drivers are not present).

### **3.4.5 DC Blocking Filter**

If this menu is checked, the DC blocking filter is switched on. Frequencies below 5 Hz are suppressed for the signals sent to the audio card. This removes any DC offset and centers the signal around 0. Each output channel has its own independent DC blocking filter.

### **3.4.6 DSP runtime compiler**

If this menu is checked, the DSP compiler is enabled. The compiler generates specific x86 instructions from the patch(es). It replaces generic functions with specialised sequences of opcodes that are optimised for the current use of modules in the patch. It looks for unconnected inputs or outputs and makes smaller code that skips calculation of these connections. The result is reduced CPU load in many cases, while the generated audio is identical to the more generic implementations.

## **3.5 Configuration menu**

### **3.5.1 Global settings**

Open a window for changing settings effecting all projects or the program.

### **3.5.2 Patch settings**

Open a window for changing settings related to a single patch.

### **3.5.3 Save Settings As**

The current program settings (including window positions) are saved in a new file. This can be used to maintain multiple configurations and window locations for the program.

## **3.6 Help menu**

### **3.6.1 Context help (F1)**

Help window will open displaying help corresponding with current context. This is most of the time the current selected item or the item directly below the cursor.

### **3.6.2 Editor help (Shift+F1)**

Help window will open showing help for current active editor screen.

### **3.6.3 Menu help**

Help window will open showing an information page about the menus.

### **3.6.4 Introduction**

Help window will open showing a small introduction into the basics of using StudioFactory.

### 3.6.5 About StudioFactory

Help window will open showing detailed licensing and version information of StudioFactory.

## 4 Building patches

## 5 SynFactory Modules

### 5.1 Generators

#### 5.1.1 OSC - Multifunction oscillator

This is the standard oscillator, it can provide 5 different waveforms selectable with a block of icons (there are 6 icons but one is not currently active). The 5 waveforms are: Triangle, Saw, Pulse, Sine and Triggered Noise. The Pulse waveform has a variable pulse width. Use the *PWidth* control and/or the *PW* inputs to change the pulse width. With the control at 0 gives a 50% waveform. At 32767 (or -32768) the wave is fully DC. The Triggered Noise waveform is comparable with a white noise source followed by a Sample and Hold gate. The trigger speed of the gate is set by the oscillator speed. The effect is useful as stepped random control at low speed, or at higher speeds to generate special effects like spaceships, storms and sea-waves.

#### Controls:

**Freq** - Sets frequency, this is a logarithmic control. To make life easier the actual frequency is displayed below the control in Hertz (number of cycles a second).

**PWidth** - Sets the pulse width, only functional when the Pulse waveform is selected.

**Ampl** - Sets the amplitude of the output signal. When set to 0 there is no signal. The value can also be negative and reverses the waveform (useful for the Saw waveform).

#### Buttons:

**AA** - Toggle button to switch the anti-aliasing filter on and off. It should be on when the oscillator is used to generate audio as it gives the cleanest result. It should be off when the oscillator is used as modulator or LFO source.

**0** - Select normal frequency range.

**+1** - Frequency range of the oscillator is shifted one octave up.

**+2** - Frequency range of the oscillator is shifted two octaves up.

#### Inputs:

**FM** (2x) - (frequency modulation) modulates the oscillator frequency in a logarithmic (musical) way.

**Lin** - Also a frequency modulation input but with a linear response, usefull for vibrato and FM synthesis.

**PW** (2x) - Pulse-width modulation, sets the width of the Pulse waveform (only active when the Pulse waveform is selected).

**AM** - Amplitude modulation input, it is added to the amount set by Ampl control. The amplitude control is linear.

**Sync** - Hard-sync, can be used to let two or more oscillators run synchronized to each other, even if both run at different frequencies. The trigger point is when the signal on this input becomes positive.

#### Outputs:

Single output providing the currently selected waveform.

#### Available:

SynFactory: All versions.

StudioFactory: All versions.

### 5.1.2 MOS - Mini oscillator

This is a more simple oscillator as the OSC. It has less inputs and controls. It provides only 3 waveforms, but these are available simultaneously. This can be useful when combined with a selector or switch so the waveform can be changed by signals available in the patch.

Controls:

**Freq** - Sets frequency, this is a logarithmic control.

Inputs:

**FM** - (frequency modulation) modulates the oscillator frequency in a logarithmic (musical) way.

**Lin** - Also a frequency modulation input but with a linear response, useful for vibrato and FM synthesis.

**PW** - Pulse-width modulation, sets the width of the Pulse waveform.

Outputs:

**Tri** - Triangle wave output

**Saw** - Saw-wave output

**Pls** - Pulse wave output (only this one can be controlled with the *PW* input)

Available:

SynFactory: Version 1.00 or higher.

StudioFactory: All versions.

### 5.1.3 RND - White and colored noise source

White noise source, with an additional 6db/oct low-pass filter.

Controls:

**Color** - Sets the cutoff frequency of the low-pass filter.

**Ampl** - Sets the peak amplitude of the resulting noise.

Inputs:

**AM** - Controls the peak amplitude of the resulting noise.

Outputs:

Single output providing the (filtered) noise.

Available:

SynFactory: Version 0.99 or higher.

StudioFactory: All versions.

### 5.1.4 Formanzzz - The formant oscillator

The formanzz module generates sound with very distinct sound spectra. It is usefull as raw material to filter or otherwise process. As pure sound source it is less usefull due to its harsh sound. It can be used to emulate electric guitars or other sounds with much overtones. The formant control selects one of the available waveforms and must be set to something different as zero (0) to generate sound. There are over 30000 different sounds spectra to choose from.

Controls:

**Freq** - Sets frequency, this is a logarithmic control (same scale as OSC and MOS oscillators).

**Formant** - Selects the generated waveform. Set this to something different as zero (0) otherwise there is no sound.

**Ampl** - Sets the amplitude of the output signal. When set to 0 there is no signal. The value can also be

negative and reverses the waveform.

Inputs:

**FM** - (frequency modulation) modulates the oscillator frequency in a logarithmic (musical) way.

**Lin** - Also a frequency modulation input but with a linear response, useful for vibrato and FM synthesis.

**Frmt** - Sets the formant, this value is added to the formant control.

**AM** - Amplitude modulation input, it is added to the amount set by Ampl control. The amplitude control is linear.

Outputs:

Single output providing the generated waveform.

Technical details:

The formanzzz module is actually a pseudo random noise generator, but it gets retriggered by a buildin oscillator. Due to the repeating character of the noise a tone with specific spectra is generated. The noise source is specifically 'badly' designed so its repetitive, something a normal noise source shouldn't be. The formant control sets the start point for the pseudo random number generator (seed) and is multiplied by a internal value to spread the actual available sounds over a huge range. Its undo able to describe all 30000+ spectra so experiment yourself.

Available:

SynFactory: Version 0.99 or higher.

StudioFactory: All versions

### 5.1.5 SineBank - Multiple octave Sine wave oscillator

Oscillator generating 9 sinewaves at the same time, each one higher in frequency. This oscillator can be used to experiment with additive synthesis (useful to make organ and bell like sounds).

Controls:

**Freq** - Sets frequency, this is a logarithmic control. To make live easier the actual frequency is displayed below the control in Hertz (number of cycles a second).

**Base** - Sets amplitude for sinewave of base harmonic.

**Base\*2..Base\*9** - Sets amplitudes for sinewaves of higher harmonics (number gives multiplication to base frequency).

Inputs:

**FM (2x)** - (frequency modulation) modulates the oscillator frequency in a logarithmic (musical) way.

**Lin** - Also a frequency modulation input but with a linear response, useful for vibrato and FM synthesis.

**FB..F9** - Control input for amplitude modulation of each of the 9 sinewaves.

Outputs:

Single output providing the sum of the 9 generated sinewaves.

Available:

SynFactory: Version 1.12 or higher.

StudioFactory: All versions

### 5.1.6 PerlinOsc - Perlin-noise based oscillator

This module is provided to replace the Formanzzz oscillator. It is a pseudo random number generator but with interpolation filters. So it goes smoothly from one value to the next. With the harmonize knob extra harmonics can be added to the sound spectra.

Because of the smoothing filters, it can be used either as complex waveform oscillator, or at low frequencies as a complex control signal source. Different waveforms can be selected but they all sound much alike,

but small harmonic variations are possible. The waveform selector also has a smoothing function so PWM like sounds can be generated.

Controls:

**Freq** - Sets frequency, this is a logarithmic control. An exact frequency can not be displayed because this oscillator generates a complex waveform.

**Wave** - Select one of the possible waveforms.

**Harm** - Higher values of this setting will add more harmonics to the generated waveform.

**Ampl** - Sets the amplitude of the output signal. When set to 0 there is no signal. The value can also be negative and reverses the waveform.

Inputs:

**FM** - (frequency modulation) modulates the oscillator frequency in a logarithmic (musical) way.

**Lin** - Also a frequency modulation input but with a linear response, usefull for vibrato and FM synthesis.

**Wave** - Select one of the possible waveforms.

**Harm** - Higher values of this input will add more harmonics to the generated waveform.

**AM** - Amplitude modulation input, it is added to the amount set by Ampl control. The amplitude control is linear.

Outputs:

Single output providing the generated waveform.

Available:

SynFactory: Version 2.00 or higher.

StudioFactory: All versions

### 5.1.7 PM Osc - Oscillator with phase modulation

This oscillator provides a sine wave output and has phase modulation inputs. This oscillator can be used to build the carrier oscillator for classic 'FM'-Synthesis. Modulating the phase instead of the frequency has two important advantages. It allows selfmodulation (feedback) and small DC offsets on the PM inputs don't change the oscillator frequency. To increase the range of available sounds there is also normal FM and hardsync inputs.

Controls:

**Freq** - Sets frequency, this is a logarithmic control. An exact frequency can not be displayed because this oscillator generates a complex waveform.

**FBack** - Sets the amount of feedback modulation. It modulates the phase with the output waveform. Higher settings transform the sine output to triangle waveform.

**Ampl** - Sets the amplitude of the output signal. When set to 0 there is no signal. The value can also be negative and reverses the waveform.

Inputs:

**FM (2x)** - (frequency modulation) modulates the oscillator frequency in a logarithmic (musical) way.

**Lin** - Also a frequency modulation input but with a linear response, usefull for vibrato.

**PM (2x)** - Phase modulation input. Allows phase modulation based 'FM'-synthesis.

**fdb** - Feedback modulation input.

**AM** - Amplitude modulation input, it is added to the amount set by Ampl control. The amplitude control is linear.

**Sync** - Hard-sync, can be used to let two or more oscillators run synchronized to each other, even if both run at different frequencies. The trigger point is when the signal on this input becomes positive.

Outputs:

**out** - Output of oscillator scaled by AM modulation.  
**full** - Waveform before it is amplitude modulated by *Ampl* and *AM*.  
Available:  
SynFactory: not available.  
StudioFactory: All versions

### 5.1.8 Quad Osc - Quadrature oscillator with phase modulation

Oscillator with multiple sine wave outputs. Each sine-wave has a 45 degree phase shift in relation to the previous output. This oscillator can be used for panning or cross-fading control. It also has 2 phase modulation inputs so can be used for fm-synthesis as substitute for the PM-Osc.

Controls:

**Freq** - Sets frequency, this is a logarithmic control (same scale as OSC and MOS oscillators).

**Ampl** - Sets the amplitude of the output signal. When set to 0 there is no signal. The value can also be negative and reverses the waveform.

Inputs:

Outputs:

**0** - Sine-wave output.

**45** - Sine-wave output with 45 degrees phase shift.

**90** - Sine-wave output with 90 degrees phase shift.

**135** - Sine-wave output with 135 degrees phase shift.

**180** - Sine-wave output with 180 degrees phase shift.

**225** - Sine-wave output with 225 degrees phase shift.

**270** - Sine-wave output with 270 degrees phase shift.

**315** - Sine-wave output with 315 degrees phase shift.

Available:

SynFactory: not available.

StudioFactory: All versions

## 5.2 Filters and amplifiers

### 5.2.1 MMF - Multi Mode Filter (12/18/24 dB)

This is a 4 stage filter with multiple outputs. It has a very natural character and can be used for most filtering purposes. Because of its 4 stages it can sound a little dull, use the feedback control to 'lighten up' the sound.

Controls:

**Cutoff** - Sets cutoff frequency for the filter. All frequencies above the cutoff frequency will be filtered.

**Reso** - Sets the filter resonance amount. It amplifies frequencies around the cutoff point. Generates a distinct peak in the audio spectrum. At high settings the filter will self oscillate.

**FdBack** - Enhances the higher tones in the spectrum. Higher feedback settings give a brighter sound.

Inputs:

**In** (3x)- Audio input, this is the signal which is filtered.

**Frq** (2x) - Cutoff control input. Same scale as Cutoff control.

**Q** - External control of resonance, input is combined with Reso control.

Outputs:

**L12** - Lowpass output, the audio is filtered with a 12 db/oct slope.

**L18** - Lowpass output, the audio is filtered with a 18 db/oct slope.

**L24** - Lowpass output, the audio is filtered with a 24 db/oct slope.

**BP** - Bandpass output.

**HP** - Highpass output.

Technical details:

The filter has 4 stages with 6 db/oct filters. The BP and HP are generated by taking the difference between some outputs of the stages. This result in only between 6 to 12 db/oct for the BP and HP outputs. More strong filtering would make these outputs too dull. Use the OB-VCF filter if a little more heavy BP filtering is necessary. The OB-VCF has very strong feedback and therefore more steep filtering. The Feedback control is accomplished by sending part of the lowpass filtered output back to the input, because these will be out of phase they cancel each other out. This results in LP outputs which sound brighter.

Available:

SynFactory: Version 1.00 or higher.

StudioFactory: All versions

### 5.2.2 OB VCF - 12dB OB-type filter

This filter emulates the sound of the Oberheim voice-expansion filter. It is a two stage filter with much feedback, so it gives a very strong sound. Use it for those classic analog sounds like leads and strings. Because of its resonance characteristic, its also useful for noise filtering in percussion sounds. Due to the power of the filter, there is a build in limiter to prevent 'lockup' at high signal inputs.

Controls:

**Cutoff** - Sets cutoff frequency for the filter. All frequencies above the cutoff frequency will be filtered.

**Reso** - Sets the filter resonance amount. It amplifies frequencies around the cutoff point. Generates a distinct peak in the audio spectrum. At high settings the filter will self oscillate.

Inputs:

**In** (3x)- Audio input, this is the signal which is filtered.

**Frq** (2x) - Cutoff control input. Same scale as Cutoff control.

**Q** - External control of resonance, input is combined with Reso control.

Outputs:

**LP** - Lowpass output, the audio is filtered with a 12 db/oct slope.

**BP** - Bandpass output.

**HP** - Highpass output.

Technical details:

The filter has 2 stages with 6 db/oct lowpass filters. Due to a feedback loop from the last stage to the input, the filter will operate as a multimode filter providing LP, BP and HP outputs simultaneous. This filter has some special characteristics. The resonance peak lies higher as the cutoff frequency. The BP and HP outputs are much stronger as compared to the LP (except when self oscillating). Although the MMF-filter has a feedback loop, using it will not result in the same sound. The MMF-filter has the resonance peak around the cutoff, while this filter has it about 2 octaves higher.

Available:

SynFactory: Version 1.00 or higher.

StudioFactory: All versions

### 5.2.3 ECF - Envelope Controlled 12 dB OB-type Filter

This module is a combination of 3 other modules. It contains a 3 channel audio mixer, an envelope generator and an Oberheim type filter.

Controls:

**Lev 1** - Control the mixing level of the In 1 input (linear response).

**Lev 2** - Control the mixing level of the In 2 input (linear response).

**Lev 3** - Control the mixing level of the In 3 input (linear response).

**Cutoff** - Sets cutoff frequency for the filter. All frequencies above the cutoff frequency will be filtered.

**A** - Sets the attack rate for the envelope generator.

**D** - Sets the decay rate for the envelope generator.

**S** - Sets the sustain level for the envelope generator.

**R** - Sets the release rate for the envelope generator.

**Reso** - Sets the filter resonance amount. It amplifies frequencies around the cutoff point. Generates a distinct peak in the audio spectrum. At high settings the filter will self oscillate.

Inputs:

**In1** - Audio input 1.

**In2** - Audio input 2.

**In3** - Audio input 3.

**Mod** - Amount control input.

**Frq (2x)** - Cutoff control input. Same scale as Cutoff control.

**Q** - External control of resonance, input is combined with Reso control.

**Gate** - When gate is high ( $> 0$ ), Attack/decay stage is started. On gate is low ( $\leq 0$ ) release stage is entered.

Outputs:

**LP** - Lowpass output, the audio is filtered with a 12 db/oct slope.

**BP** - Bandpass output.

**HP** - Highpass output.

Available:

SynFactory: Version 1.00 or higher.

StudioFactory: All versions

### 5.2.4 VCA - Amplifier with variable gain

This module which is named after its analog counterpart (Voltage Controlled Amplifier), is used to amplify a signal depending on the value of the control inputs. There is also a front panel control to set the default gain of the amplification. The range of the gain is 0% ( $\leq 0$ ) to 400% (32767).

Controls:

**Gain** - Initial gain (range 0% to 400%).

Inputs:

**In** - Input signal.

**Exp** - exponential control input (same as Gain knob).

**Lin** - Linear control input, sometimes useful for control by an ADSR. Because this module has 2 quadrant modulation only, it can't be used for ringmodulation. Use a MUL module instead.

Outputs:

Single output providing the attenuated or amplified signal.

Technical details:

Output = (In \* (Lin + SCALEDEXP(Exp + Gain))) / 8192

Available:

SynFactory: Version 1.00 or higher.

StudioFactory: All versions

### 5.2.5 ECA - Envelope Controlled Amplifier

This module is a combination of 3 other modules. It contains a three channel mixer, an envelope generator and an amplifier with variable gain. In most analog synthesizer a module like this one is used as last stage before it goes to the output mixer.

Controls:

**Lev 1** - Control the mixing level of the In 1 input (linear response).

**Lev 2** - Control the mixing level of the In 2 input (linear response).

**Lev 3** - Control the mixing level of the In 3 input (linear response).

**Volume** - The mixed signal of the 3 inputs are scaled by the output curve of the envelope. This Knob sets the maximum amplification level for the envelope.

**A** - Sets the attack rate for the envelope generator.

**D** - Sets the decay rate for the envelope generator.

**S** - Sets the sustain level for the envelope generator.

**R** - Sets the release rate for the envelope generator.

Inputs:

**In1** - Audio input 1.

**In2** - Audio input 2.

**In3** - Audio input 3.

**Vol** - External control for the Volume setting.

**A** - Attack rate control input.

**D** - Decay rate control input.

**R** - Release rate control input.

**Gate** - When gate is high ( $> 0$ ), Attack/decay stage is started. On gate is low ( $\leq 0$ ) release stage is entered.

Outputs:

Single output providing the mixed and enveloped signal.

Available:

SynFactory: Version 1.00 or higher.

StudioFactory: All versions

## 5.3 Modulation

### 5.3.1 CTR - Signal controller

This module can change the gain and offset of a control signal. The gain can change the amplitude of the control signal from -200% to 200%. The offset setting will add an additional static level to the signal (DC offset).

Controls:

**Gain** - Change gain (amplification) of the input signal.

**Offset** - Add DC offset to the output signal.

Inputs:

**In** - Signal input.

**Mod** - External control of gain, input is added to value of Gain control.

**Offs** - External control of offset, input is added to value of Offset control.

Outputs:

Single output providing the modified control signal.

Technical details:

Output = (In \* (Gain + Mod)) / 16384 + Offset + Offs

Available:

SynFactory: Version 0.99 or higher.

StudioFactory: All versions

### 5.3.2 ENV - ADSR Envelope generator

This is the classic attack-decay-sustain-release envelope generator. Next to it use as envelope generator for the output signal it can modulate filters, create frequency sweeps for the oscillators or control parameters from effect generators.

Controls:

**A** - Sets the attack rate for the envelope generator.

**D** - Sets the decay rate for the envelope generator.

**S** - Sets the sustain level for the envelope generator.

**R** - Sets the release rate for the envelope generator.

**Amount** - Set the maximum level the envelope will reach.

Inputs:

**Mod** - Amount control input.

**A** - Attack rate control input.

**D** - Decay rate control input.

**S** - Sustain level control input.

**R** - Release rate control input.

**Gate** - When gate is high (> 0), Attack/decay stage is started. On gate is low ( $\leq 0$ ) release stage is entered.

Outputs:

Single output with envelope signal.

Available:

SynFactory: Version 0.99 or higher.

StudioFactory: All versions

### 5.3.3 AHD - Attack/Hold/Decay envelope generator

Small envelope generator which uses a trigger input. It has a hold setting, because normally the sustain time is determined by a gate input, but this envelope generator has only a trigger input.

Controls:

**Attack** - Set total attack time.

**Hold** - Set time the envelope will wait after attack phase before starting decay phase.

**Decay** - Set total decay time.

**Amount** - Set the maximum level the envelope will reach.

Inputs:

**Mod** - Amount control input.

**A** - Attack time control input.

**H** - Hold time control input.

**D** - Decay time control input.

**Trig** - Trigger input, envelope starts on positive trigger. It will retrigger even if A, H or D stage is not yet completed.

Outputs:

Single output with envelope signal.

Available:

SynFactory: Version 1.13 or higher.

StudioFactory: All versions

#### 5.3.4 S/H - Sample and Hold

The sample and hold module will take the value of the In input on each low-to-high transition of Trig and will hold this value until the next transition. It will generate stepped control signals from continuous signals.

Inputs:

**In** - Signal input, this signal is sampled on each trig transition.

**Trig** - Trigger signal.

Outputs:

**Out** - Stepped signal output.

Technical details:

This module can be used for a lot of different purposes. In combination with a RND as source it generates a new random value on each trigger point. Feeding it continuous signals generates stepped control signals. Unlike its analog counterpart, this module's use is not restricted to control signals, it works also on audio signals, generating all sorts of artifacts when both Trig and In are connected to two different audio signal sources, in this mode it functions as a distortion unit.

Available:

SynFactory: Version 0.99 or higher.

StudioFactory: All versions

#### 5.3.5 GLI - Glide/Portamento

The output of this module follows the input, but it can be slowed down with the speed setting. It finds its use after Sample and Hold gates or sequencers to 'glide' the sound. In combination with a 'splitter' module it can function as an envelope follower. This module is not very useful for processing audio signals (output doesn't follow the input fast enough).

Controls:

**Speed** - Sets the follow speed 0 slowest and at 32767 the output will follow input directly.

Inputs:

**In** - Input signal which is filtered and sent to the output.

**Spd** - Control input for the speed setting.

Outputs:

Single output providing the filtered/followed input signal.

Available:

SynFactory: Version 0.99 or higher.

StudioFactory: All versions

### 5.3.6 TRIG - Delayed gate/trigger/one-shot

This module delays a trigger signal or stretches a gate signal. Both modes can be used at the same time, delaying both the trigger moment and the gate release times.

Controls:

**T-On** - Sets the delay time between the activation of the gate input and the outputs becoming active (32767).

**T-Off** - Sets the delay time between the deactivation of the gate input and the gate output becoming inactive (0) and it also sets the length of the one shot pulse.

Inputs:

**T-on** - Control input for the on time delay setting.

**T-off** - Control input for the off time delay setting.

**Gate** - Gate signal input. Defined as  $\leq 0$  is inactive and  $> 0$  is active.

Outputs:

**Gate** - Gate signal delayed from the input gate. Length of the pulse depends on the length of the input pulse and the settings of the controls.

**1shot** - The pulse length is independent on the length of the gate input.

Available:

SynFactory: 1.00 or higher version.

SynFactory: 1.12 or higher version for module with 1shot output.

StudioFactory: All versions

### 5.3.7 SEQ - 8 steps sequencer

This is a eighth step analog style sequencer. It needs a clock input to step through all the knob values. If it encounters a negative value it will reset to the first position.

Controls:

**Step 1-8** - Value for each step of the sequencer.

Inputs:

**Clk** - On each clock transition from low to high ( $\leq 0$  to  $> 0$ ) the sequencer steps one step further.

**Rst** - Restart sequencer to the first step when this input is high ( $> 0$ ). It will stay at step 1 as long as the input is high.

Outputs:

**Out** - Value of the current active step of the sequencer.

**Rst** - This output is active when sequencer is on step 1.

Available:

SynFactory: Version 0.99 or higher.

StudioFactory: All versions

### 5.3.8 SEQ - 16 steps sequencer

This is a sixteen step analog style sequencer. It needs a clock input to step through all the knob values. If it encounters a negative value it will reset to the first position.

Controls:

**Step 1-16** - Value for each step of the sequencer.

Inputs:

**Clk** - On each clock transition from low to high ( $\leq 0$  to  $> 0$ ) the sequencer steps one step further.

**Rst** - Restart sequencer to the first step when this input is high ( $> 0$ ). It will stay at step 1 as long as the

input is high.

Outputs:

**Out** - Value of the current active step of the sequencer.

**Rst** - This output is active when sequencer is on step 1.

Available:

SynFactory: Version 0.99 or higher.

StudioFactory: All versions

### 5.3.9 BinSeq - Binary input 8 steps sequencer

This is an eighth step analog style sequencer. The active step is selected with a four bits binary input.

Controls:

**Step 1-8** - Value for each step of the sequencer.

Inputs:

**S0** - Select the active step (S0 to S2 form a 3 bits value selecting a step). This is bit 0 (step offset 1).

**S1** - Select the active step (S0 to S2 form a 3 bits value selecting a step). This is bit 1 (step offset 2).

**S2** - Select the active step (S0 to S2 form a 3 bits value selecting a step). This is bit 2 (step offset 4).

**-OE** - Disables the output (makes it zero) if this input is positive.

Outputs:

**Out** - Value of the current active step of the sequencer.

Available:

SynFactory: ??? or higher version.

StudioFactory: All versions

### 5.3.10 Scanner - 9 input analog style multiplexer

The magnitude of the 'scan' input determines which input(s) are available on the output. Each input has a 16k value range that overlap with one higher and one lower input. If the 'scan' input is in a overlapped range the inputs are mixed.

Inputs:

**In32k** - .

**In24k** - .

**In16k** - .

**In8k** - .

**In0** - .

**In-8k** - .

**In-16k** - .

**In-24k** - .

**In-32k** - .

**scan** - .

Outputs:

**Out** - .

Available:

SynFactory: not available.

StudioFactory: All versions

## 5.4 Math modules

### 5.4.1 ADD - 2-8 input adder

Adds the signals on all inputs together and send it to the output. It accepts all types of signals and simply adds them together. Useful if you want to feed multiple outputs into one input. Some modules already have multiple inputs when this is common (like the filters). Adding multiple high volume signals together increases the chance for distortion. Use one of the mixers instead, because these have controls to lower the volume on each input.

Inputs:

2 to 8 signal inputs which will be added together.

Outputs:

Single output providing the summed value of all inputs.

Available:

SynFactory: Version 0.99 or higher.

StudioFactory: All versions

### 5.4.2 MUL - Multiplier/Ringmodulator

Calculates the product of the two inputs and scales this down to the normal signal range. If both inputs have maximum amplitude, the resulting product will have the same amplitude. This module can be used for multiple purposes. It can function as a ringmodulator, a amplitude modulator with linear response or a distortion unit (with same signal connected to both inputs).

Inputs:

Two signal inputs.

Outputs:

Single output providing scaled product of the two inputs.

Technical details:

Normal signal range is from -32768 to 32767. Multiply these together and the range becomes much larger. The resulting product is scaled down by dividing by 32768. This is a very fast operation because it can be done by shifting the result 15 bits to the right.

Available:

SynFactory: Version 0.99 or higher.

StudioFactory: All versions

### 5.4.3 SUB - Subtractor

Calculates the difference between two signals. Because a not connected input is zero, it can also be used as an inverter.

Inputs:

**inv** - Inverted input, this signal is subtracted from zero before added.

**noninv** - Non inverting input, this signal is added unmodified.

Outputs:

Single output providing the difference between the two inputs.

Technical details:

Output = noninv - inv

Available:

SynFactory: Version 0.99 or higher.

StudioFactory: All versions

#### 5.4.4 SPLIT - Positive and negative signal splitter

This module splits the input in 3 different signals depending on the sign of the input signal.

##### Inputs:

Single input signal which is splitted on sign.

##### Outputs:

**Pos** - Only the positive values of the input signal are send to this output. If the input signal is negative this output is set to zero.

**Neg** - Only the negative values of the input signal are send to this output. If the input signal is positive this output is set to zero.

**Abs** - This is the absolute value of the input signal. Negative signals are converted to positive signals, zero or positive signals are unchanged. The effect is a signal with the same amplitude as the input signal, but the signal is always positive (or zero).

##### Available:

SynFactory: Version 0.99 or higher.

StudioFactory: All versions.

#### 5.4.5 BETWEEN - Window comparator

This module performs two functions. It checks if a signal is in a range set with two set-points. Two logical outputs provide information about the input signal. The signal is also send out unmodified as long its in the allowed range. Outside the range its clipped to one of the set-points. This module can therefore also be used as a distortion unit.

##### Controls:

**Point1** - Set clipping point 1.

**Point2** - Set clipping point 2.

##### Inputs:

**inp** - Signal input.

**pnt1** - Control input for setting first limit point.

**pnt2** - Control input for setting second limit point.

##### Outputs:

**out** - Output is the same as the input signal but clipped to the set-points when the signal is outside the allowed range.

**btwn** - This logic output is active (32767) when the signal is within the range (doesn't clip).

**clp'd** - This logic output is active (32767) when the signal is outside the range (it is clipped).

##### Available:

SynFactory: Version 1.13 or higher.

StudioFactory: All versions.

## 5.5 Logic Modules

### 5.5.1 OR - 2-8 input logic OR gate

The OR module performs a logic function. The output is active (32767) when one or more of the inputs are active. When all inputs are smaller or equal to zero the output is also zero.

Inputs:

Two to eight logic inputs. When the input value is  $\leq 0$  the input is inactive, when value is  $> 0$  the input is active.

Outputs:

Single output providing the result of the logic OR function. This output is active (32767) when one or more inputs are active ( $\neq 0$ ).

Technical details:

!!!TODO

Available:

SynFactory 1.00 or higher version.

StudioFactory: All versions.

### 5.5.2 AND - 2-8 input logic AND gate

The AND module performs a logic function. The output is active when all of the inputs are active.

Inputs:

Two to eight logic inputs. When the input value is  $\leq 0$  the input is inactive, when value is  $\neq 0$  the input is active.

Outputs:

Single output providing the result of the logic AND function.

Technical details:

Technical details:

!!!TODO

Available:

SynFactory: Version 1.00 or higher.

StudioFactory: All versions.

### 5.5.3 NOT - Logic NOT gate

The output is active (32767) when the input is inactive ( $\leq 0$ ). The output is inactive (0) when the input is active ( $> 0$ ).

Inputs:

Single input for NOT gate.

Outputs:

Single output providing the logic NOT function.

Technical details:

!!!TODO

Available:

SynFactory: Version 1.00 or higher.

StudioFactory: All versions.

#### 5.5.4 XOR - Logic XOR gate

Inputs:

2 inputs for XOR gate.

Outputs:

Single output providing the logic Exclusive OR function.

Technical details:

!!!TODO

Available:

SynFactory: Version 1.00 or higher.

StudioFactory: All versions.

#### 5.5.5 ADC - Digitize signals (9 bits resolution)

Converts a continuous signal to a 9 bit digital representation.

Inputs:

Single signal input

Outputs:

**b7..b0** - 8 bit digital representation of absolute value of input.

**neg** - negative flag is 32767 when signal is  $\leq 0$  otherwise it is 0.

Available:

SynFactory: not available.

StudioFactory: All versions.

#### 5.5.6 DAC - Binary to signal (9 bits resolution)

Converts a 9 bit digital representation of a signal to a (511 steps) signal output.

Inputs:

**b7..b0** - 8 bit digital representation of absolute value of output.

**neg** - negative flag is 32767 when signal is  $\leq 0$  otherwise it is 0.

Outputs:

Single output providing the converted signal.

Available:

SynFactory: not available.

StudioFactory: All versions.

#### 5.5.7 Pulsar - Pulse generator/clock divider

Pulsar acts like a counter but the outputs give only a very short pulse. This module can be used to trigger SyncDelay modules or control counters and other pulsar modules. Triggering envelope generators can be tricky because the pulse is too short for the envelope generator to follow. Use a TRIG module to increase the pulse length.

Inputs:

**Clk** - Clock input, Pulsar counts on rising edge of clock signal.

**Rst** - Internal counters are resetted.

Outputs:

**:2** - Clock divided by two (a pulse is generated for each 2 clocks).

- :3** - Clock divided by three (a pulse is generated for each 3 clocks).
- :4** - Clock divided by four (a pulse is generated for each 4 clocks).
- :5** - Clock divided by five (a pulse is generated for each 5 clocks).
- :6** - Clock divided by six (a pulse is generated for each 6 clocks).
- :7** - Clock divided by seven (a pulse is generated for each 7 clocks).
- :8** - Clock divided by eight (a pulse is generated for each 8 clocks).
- :16** - Clock divided by sixteen (a pulse is generated for each 16 clocks).

Available:

SynFactory: Version 1.01 or higher. Although it is available in some earlier versions, this module doesn't work correctly in any version of SynFactory before 1.01.

StudioFactory: All versions.

### 5.5.8 BIN - 8 bit Binary counter

Digital counter. Clock signals are counted binary wise. The counter is 8 bits wide so divides the clock signal by 256. The counter triggers on a negative clock input, this makes daisy-chaining multiple counter easy. Simply connect the :256 output of the first counter with the -Clk input of the second. They will act as a 16 bit counter that way dividing the clock by 65536.

Inputs:

**-Clk** - Clock input for counter. Single step after each positive to zero or negative transition of the clock.

**Rst** - Making this input high will reset the counter to 0 (all outputs low).

Outputs:

**:2** - Counter bit 0, clock divided by 2 output.

**:4** - Counter bit 1, clock divided by 4 output.

**:8** - Counter bit 2, clock divided by 8 output.

**:16** - Counter bit 3, clock divided by 16 output.

**:32** - Counter bit 4, clock divided by 32 output.

**:64** - Counter bit 5, clock divided by 64 output.

**:128** - Counter bit 6, clock divided by 128 output.

**:256** - Counter bit 7, clock divided by 256 output.

Available:

SynFactory: Version 1.00 or higher.

StudioFactory: All versions.

### 5.5.9 DeMux - 1 to 2 demultiplexer

A demultiplexer makes one of its outputs high (32767) and the rest is low (0). Which of the outputs is active depends the input signals. The inputs represent a binary number that selects the active output.

Inputs:

**S0** - Binary input to select one of the two outputs.

Outputs:

**Gt0-1** - Only one of these outputs is active.

Technical details:

!!!TODO

Available:

SynFactory: Version 1.00 or higher.

StudioFactory: All versions.

#### 5.5.10 DeMux - 2 to 4 demultiplexer

A demultiplexer makes one of its outputs high (32767) and the rest is low (0). Which of the outputs is active depends the input signals. The inputs represent a binary number that selects the active output.

Inputs:

**S0** - Binary input to select one of the four outputs (weight of S0 is 1).

**S1** - Binary input to select one of the four outputs (weight of S1 is 2).

Outputs:

**Gt0..3** - Only one of these outputs is active (active output calculate with  $S1*2+S0$ ).

Technical details:

!!!TODO

Available:

SynFactory: Version 1.00 or higher.

StudioFactory: All versions.

#### 5.5.11 DeMux - 3 to 8 demultiplexer

A demultiplexer makes one of its outputs high (32767) and the rest is low (0). Which of the outputs is active depends on the input signals. The inputs represent a binary number that selects the active output.

Inputs:

**S0** - Binary input to select one of the eight outputs (weight of S0 is 1).

**S1** - Binary input to select one of the eight outputs (weight of S1 is 2).

**S2** - Binary input to select one of the eight outputs (weight of S2 is 4).

Outputs:

**Gt0..7** - Only one of these outputs is active (active output calculate with  $S2*4+S1*2+S0$ ). Technical details:

!!!TODO

Available:

SynFactory 1.00 or higher version.

StudioFactory: All versions.

#### 5.5.12 DeMux - 4 to 16 demultiplexer

A demultiplexer makes one of its outputs high (32767) and the rest is low (0). Which of the outputs is active depends the input signals. The inputs represent a binary number that selects the active output.

Inputs:

**S0** - Binary input to select one of the sixteen outputs (weight of S0 is 1).

**S1** - Binary input to select one of the sixteen outputs (weight of S1 is 2).

**S2** - Binary input to select one of the sixteen outputs (weight of S2 is 4).

**S3** - Binary input to select one of the sixteen outputs (weight of S3 is 8).

Outputs:

**Gt0..15** - Only one of these outputs is active (active output calculate with  $S3*8+S2*4+S1*2+S0$ ).

Technical details:

!!!TODO

Available:

SynFactory: Version 1.00 or higher.

StudioFactory: All versions.

## 5.6 Song Modules

### 5.6.1 SCRIPT - Script controlled note player

This module is a note sequencer with a builtin scripting engine. It can perform multiple functions depending on the script. Main output is a note with gate signal, but six additional control outputs are available. The outputs can be used for example to set note volume (velocity) or change the filter cutoff and resonance.

Double click on the script module with the left mouse button to open the script editor. After required changes are made, close the editor window with the close icon in the right corner of the window. If the patch is running, the scripting module will react as if the Rst input is activated (the script will start from the beginning). Take note: Errors in the script will be skipped without any visual feedback!

#### Inputs:

**-Clk** - Clock input. With the Pcount command the script can be paused for one or more clock ticks (trigger on falling edge).

**Rst** - If 1 reset the module, script will restart from beginning (Note: outputs are not resetted).

**Inp1 - Inp6** - 6 extra inputs which can be used to control the flow of the running script.

#### Outputs:

**Freq** - Frequency output (use commands 'C', 'C#', 'D', 'D#', 'E', 'F', 'F#', 'G', 'G#', 'A', 'A#' or 'B' to control this output).

**Gate** - Gate signal (use commands [ and ] to control this output).

**u-z** - Six extra control outputs (use commands u-z to control these outputs). Outputs can function as control signal or extra note outputs.

#### Scripting language:

{ **comment** } - Place comment in the script, this will be skipped by the scripting engine. Use it to make the scripts more readable.

**C4** - Play note C from 4th octave. Last octave is remembered so if multiple notes from the same octave are played only the first one needs the octave number.

**C** - Play note C from current octave.

**C#** - Play note C-sharp from current octave.

**D D# E F F# G G# A A# B** - Play one of the other notes from the octave.

[ - Turn gate signal on (32767).

] - Turn gate signal off (0).

:label - Define subroutine label. The : command will only set a label, executing the command will not skip the subroutine behind it. Therefore place subroutines at the end of the script, so the normal script flow is not disturbed.

; - Return from subroutine, back to the point just behind the Mlabel command. If there are no return points, it will behave as the R command.

Jlabel - Jump to label, its like a 'goto' command in basic. The ; command can not return after a jump, use Mlabel if the return point is needed.

Mlabel - Call subroutine with specified label.

Pcount - Wait for count number of clock cycles. **P0** doesn't wait on clock but waits only one sample, this is usefull for generating trigger signals. The sequence W32767 P0 W0 P0 will generate a short pulse on the w output.

**R** - Reset module, script will restart from beginning (Note: outputs are not resetted, use the sequence U0 V0 W0 X0 Y0 Z0 ] P0 to reset all outputs).

Uvalue - Set u output to specified value.

Unote - Set u output to specified note (for example UC#2).

**V**value - Set v output to specified value.  
**V**note - Set v output to specified note (for example VC#2).  
**W**value - Set w output to specified value.  
**W**note - Set w output to specified note (for example WC#2).  
**X**value - Set x output to specified value.  
**X**note - Set x output to specified note (for example XC#2).  
**Y**value - Set y output to specified value.  
**Y**note - Set y output to specified note (for example YC#2).  
**Z**value - Set z output to specified value.  
**Z**note - Set z output to specified note (for example ZC#2).

Example script:

```
{ This script will loop fast through multiple notes, use W output to trigger base drum sound }  
C5 W1 P0 W0 P2 D# P2 E P2 D P2  
C5 W1 P0 W0 P2 D# P2 E P2 D P2  
C5 P2 D# P2 E P2 D P2  
C5 P2 D# P2 E P2 D P2 R
```

Available:

SynFactory: Version 1.13 or higher.  
StudioFactory: All versions.

## 5.7 Effect Modules

### 5.7.1 FLA - Flanger

Strictly speaking a flanger/chorus is nothing more as a variable delayline. This delayline has filters around it, so it's possible to modulate the delayline length without introducing noise. When combining the input and delayed signal, phase differences will generate the flanger/chorus effect. The exact effect depends on modulation speed, modulation amplitude and the waveform that is used (best use triangle or sine waves as modulator). The delayline is very short, use a normal delay module if longer delaytimes are needed (but the length of the normal delayline is more difficult to modulate without artifacts). Controls:

**InpLev** - Sets the gain of the input signal. The delayline can clip if the sum of the feedback and input signal is too high.

**FdBack** - Part of the delayline output can be send back to the input.

**Mix** - Sets the mixing level between the input signal and delayed signal.

Inputs:

**Inp** - Audio input signal.

**Pos** - Controls the position of tap in the delayline. This changes the effective length of the delayline. Modulate with a low-frequency triangle or sine wave.

**Fbd** - Control input for the feedback setting.

**Mix** - Control input for the mix setting.

Outputs:

**mix** - This output provides the sum of input signal with the delayed signal (depending on the setting of the mix control).

**dly** - Provides the output of the delayline without mixed input (this is also the internal signal used as feedback signal).

Technical details:

The flanger is a delayline with a interpolation filter at the output. The lower 4 bits are used to interpolate

between the nearest samples in the delayline. This gives a 16 times higher modulation precision as the samplerate normally permits, but it sacrifices the maximum available delayline length to get these four extra bits.

Available:

SynFactory: Version 0.99 or higher.

StudioFactory: All versions.

### 5.7.2 DLY - Delay line

Delayline with single input and output (single tap). For convenience there is builtin support for a feedback signal. Use it as a simple reverb or connect multiple together to build a multi-tap delay effect. Although the length of delayline can be modulated, there is a high chance for clicking noise. Use a Flanger module instead because it has better suppression of transition noise (but also shorter delay times).

Controls:

**InpLev** - Sets the gain of the input signal. The delayline can clip if the sum of the feedback and input signal is too high.

**Length** - Sets the length for the delayline (in samples). Maximum length is  $32767/44100 = 0.74$  seconds.

**FdBack** - Part of the delayline output can be send back to the input.

Inputs:

**Inp** - Audio input signal.

**Len** - Control input for the delayline length setting.

**Fbd** - Control input for the feedback setting.

Outputs:

**mix** - This output provides the sum of input signal with the delayed signal.

**dly** - Provides the output of the delayline without mixed input (this is also the internal signal used as feedback signal).

Available:

SynFactory: Version 0.99 or higher.

StudioFactory: All versions.

### 5.7.3 SyncDly - Syncable Delay line

Delayline with single input and output (single tap). For convenience there is builtin support for a feedback signal. The delay length is derived from a clock source. Maximum delay length is little over 1 second, it can take a few seconds before the delay 'locks' to the input clock. Use it as a rhythmic delay or connect multiple together to build a multi-tap delay effect.

Controls:

**InpLev** - Sets the gain of the input signal. The delayline can clip if the sum of the feedback and input signal is too high.

**FdBack** - Part of the delayline output can be send back to the input.

Inputs:

**Inp** - Audio input signal.

**Clk** - Input for clock signal, the frequency of the input clock determines delayline length.

**Fbd** - Control input for the feedback setting.

Outputs:

**mix** - This output provides the sum of input signal with the delayed signal.

**dly** - Provides the output of the delayline without mixed input (this is also the internal signal used as

feedback signal).

Available:

SynFactory: Version 1.00 or higher.

StudioFactory: All versions.

#### **5.7.4 BIT - Bit reducer This module is also known as a quantizer.**

It reduces the possible values a signal can be. All other values are rounded down to the nearest allowed value. High settings will generate extreme quantization noise, which might be useful for special sound effects. It can also process control signals to make 'stair case' waveforms. Not unlike a Sample and Hold gate, but without the need for a trigger signal.

Controls:

**Bits** - Resolution of the quantizing steps. Higher values give bigger steps and more noticeable effect.

Inputs:

**Inp** - Signal input which will be quantized.

**Bits** - Resolution control.

Outputs:

Single output providing the quantized output.

Technical details:

If this module would really do a bit-reduction only 15 different control settings would be available. To make a finer control range, a rounding function is performed. The signal is divided, rounded to the nearest integer and then multiplied again by the same value. The result is the same as a bit-reduce but with much finer control.

Available:

SynFactory: Version 0.99 or higher.

StudioFactory: All versions.

#### **5.7.5 Clippy - NoiseGate / Clipper / Distortion**

This module does multiple sound effects at the same time. It's a weird combination of a limiter, distortion unit and a noise-gate. Unlike normal noise-gates it doesn't remove small signals by opening and closing a (virtual) switch, but it subtracts them from the signal. The result is a crossover distortion generating high-frequency sounds at zero crossings. After that comes an amplifier with a hard limiter. When the sound hits the limiter threshold rich overtones are generated (like a pulse wave).

Controls:

**Lev** - Initial amplitude level. Higher settings will increase the limiter effect. Lower settings will increase the noise-gate effect.

**Gate** - Sets the minimum level of the signals which are let through. Higher settings (1024+) give huge distortion.

**Drive** - Boosts the signal to get the amplitude needed to activate the limiter. Works together with the Lev and Gate controls. Because these can lower the amplitude of the signal. Drive can restore or even increase the level of the signal.

Inputs:

**Inp** - Audio signal input.

**lev** - External control input for the Lev setting.

**gate** - External control input for the Gate setting.

**drv** - External control input for the Drive setting.

Outputs:

Single audio output providing the resulting audio signal after the effects.

Available:

SynFactory: Version 1.00 or higher.

StudioFactory: All versions.

### 5.7.6 PAN - Signal controlled panner

Give an audio source a position within the stereo signal.

Controls:

**Pan** - Manual panning (balance), sets the centre of the sound.

**CtrlDly** - Delays the 'Ctr' input. Fast changes can cause clicks, this can be prevented by slowing down fast transitions. If this control is at 32767 the 'Ctr' input is directly setting the panning position, lower values will filter the signal, at 0 there is no change at all.

Inputs:

**In** - Audio input.

**Ctrl** - Panning control. -32768 is left, 0 is center and 32767 is right.

**Dly** - This input is added to the value of the CtrlDly control.

Outputs:

**Left** - Left channel audio output.

**Right** - Right channel audio output.

Available:

SynFactory: Version 1.10 or higher.

StudioFactory: All versions.

## 5.8 Mixer modules

### 5.8.1 MIX - 3 channel Mixer

Module to mix 3 input signals to one. Level for each input can be set independently. The level controls are logarithmic. This means like a real mixing desk, two times higher level setting means two times perceived higher volume. This module can also be used to mix control signals, but because of the logarithmic level controls it can be tricky to get the correct amplitude for the resulting sum signal.

Controls:

**Lev 1** - Control the mixing level of the In 1 input (logarithmic).

**Lev 2** - Control the mixing level of the In 2 input (logarithmic).

**Lev 3** - Control the mixing level of the In 3 input (logarithmic).

Inputs:

**In 1** - Signal input 1.

**In 2** - Signal input 2.

**In 3** - Signal input 3.

Outputs:

Single output providing the mixed signal.

Available:

SynFactory: Version 0.99 or higher.

StudioFactory: All versions.

### 5.8.2 MIX - 4 channel Mixer

Module to mix 4 input signals to one. Level for each input can be set independently. The level controls are logarithmic. This means like a real mixing desk, two times higher level setting means two times perceived higher volume. This module can also be used to mix control signals, but because of the logarithmic level controls it can be tricky to get the correct amplitude for the resulting sum signal.

Controls:

**Lev 1** - Control the mixing level of the In 1 input (logarithmic).

**Lev 2** - Control the mixing level of the In 2 input (logarithmic).

**Lev 3** - Control the mixing level of the In 3 input (logarithmic).

**Lev 4** - Control the mixing level of the In 4 input (logarithmic).

Inputs:

**In 1** - Signal input 1.

**In 2** - Signal input 2.

**In 3** - Signal input 3.

**In 4** - Signal input 4.

Outputs:

Single output providing the mixed signal.

Available:

SynFactory: Version 0.99 or higher.

StudioFactory: All versions.

### 5.8.3 MIX - 8 channel Mixer

Module to mix 8 input signals to one. Level for each input can be set independently. The level controls are logarithmic. This means like a real mixing desk, two times higher level setting means two times perceived higher volume. This module can also be used to mix control signals, but because of the logarithmic level controls it can be tricky to get the correct amplitude for the resulting sum signal.

Controls:

**Lev 1** - Control the mixing level of the In 1 input (logarithmic).

**Lev 2** - Control the mixing level of the In 2 input (logarithmic).

**Lev 3** - Control the mixing level of the In 3 input (logarithmic).

**Lev 4** - Control the mixing level of the In 4 input (logarithmic).

**Lev 5** - Control the mixing level of the In 5 input (logarithmic).

**Lev 6** - Control the mixing level of the In 6 input (logarithmic).

**Lev 7** - Control the mixing level of the In 7 input (logarithmic).

**Lev 8** - Control the mixing level of the In 8 input (logarithmic).

Inputs:

**In 1** - Signal input 1.

**In 2** - Signal input 2.

**In 3** - Signal input 3.

**In 4** - Signal input 4.

**In 5** - Signal input 5.

**In 6** - Signal input 6.

**In 7** - Signal input 7.

**In 8** - Signal input 8.

Outputs:

Single output providing the mixed signal.

Available:

SynFactory: Version 0.99 or higher.

StudioFactory: All versions.

#### **5.8.4 CROSS - Cross fade**

Mixes two signals together. The cross fade control and cross input control the balance between the two signals. Setting cross fade at 0 will mix both inputs at 50% each. Making the cross input higher as 0 or rotating the cross control clockwise increases the amplitude of the inp H input and decreases the amplitude of inp L. When this input or control is maximum only inp H is send to the output at 100%. Making the cross input lower as 0 reverses the process and increases the amplitude of the inp L input until it is 100% at fully anti-clockwise setting of the control (or -32768 at the cross input).

Controls:

**Cross** - Sets mix between inp H and inp L. Clockwise increases inp H input, anticlockwise decreases inp H and increases the amplitude of inp L.

Inputs:

**inp H** - Input that is mixed with inp L will become louder when cross fade setting is higher as 0.

**cross** - Control input for the cross fade.

**inp L** - Input that is mixed with inp H and will become louder when cross fade setting is less as 0.

Outputs:

Single output providing a mix between inp L and inp H.

Available:

SynFactory: not available.

StudioFactory: All versions.

#### **5.8.5 MXO - Output with mixer control**

This module mixes its inputs together and sends the result directly to the soundcard.

Controls:

**L1** - Set level for first left input.

**L2** - Set level for second left input.

**M1** - Set level for first mono input

**M2** - Set level for second mono input

**M3** - Set level for third mono input

**M4** - Set level for fourth mono input

**R1** - Set level for first right input

**R2** - Set level for second right input

Inputs:

**L1** - first left input

**L2** - second left input

**M1** - first mono input

**M2** - second mono input

**M3** - third mono input

**M4** - fourth mono input

**R1** - first right input

**R2** - second right input

Available:

SynFactory: ?.

StudioFactory: All versions.

### 5.8.6 OUT - Output

This module mixes its inputs together and sends the result directly to the soundcard. The mono (mid) inputs are mixed both to the left and right channel. The signal is reduced to 50% of the original amplitude before send to the soundcard. This allows some headroom if multiple signals are added. There can be multiple OUT modules in the patch, the signals from the various OUT modules are simply added together.

Inputs:

**Left** (2x) - Signals on these inputs are send only to the left speaker.

**Mid** (4x) - Signals on these inputs are send both to the left and right speaker (mono).

**Right** (2x) - Signals on these inputs are send only to the right speaker.

Available:

SynFactory: All versions.

StudioFactory: All versions.

### 5.8.7 OUT - Output with surround

This module mixes its inputs together and sends the result directly to the soundcard. The mono (mid) inputs are mixed both to the left and right channel. The signal is reduced to 50% of the original amplitude before send to the soundcard. This allows some headroom if multiple signals are added. There can be multiple OUT modules in the patch, the signals from the various OUT modules are simply added together.

Inputs:

**Left** (2x) - Signals on these inputs are send only to the left speaker.

**Mid** (4x) - Signals on these inputs are send both to the left and right speaker (mono).

**Right** (2x) - Signals on these inputs are send only to the right speaker.

**Surrnd** (2x) - Signals on these inputs are send both to the left and right audio channels, but are send in anti-phase. A dolby decoder will put these signals onto the rear speakers.

Available:

SynFactory: Version 1.10 or higher.

StudioFactory: All versions.

### 5.8.8 PANOUT - Output with panning control

This module mixes its inputs together and sends the result directly to the soundcard. The mono (mid) inputs are mixed both to the left and right channel. In addition this module has both a panning and volume knob (and a control input for each) to set volume and balance of the left and right channel.

Controls:

**Pan** - Set difference between the amount of signal send to the left and right channels.

**Volume** - Total amount of signal send to the output.

Inputs:

**Left** - Signals on this input are send only to the left speaker.

**Mid** - Signals on this input are send both to the left and right speaker (mono).

**Right** - Signals on this input are send only to the right speaker.

**Pan** - Control input for panning.

**Vol** - Control input for setting volume.

Available:

SynFactory ??? or higher version.

StudioFactory: All versions.

### 5.8.9 INPUT - Audio Input

This module takes audio samples from a microphone or line-in and makes them available in the patch. StudioFactory can be used in this way as a powerful reverb or echo chamber. In combination with other modules more complex effects can be build like a vocoder.

Outputs:

**Left** - Left channel of microphone or line input.

**Right** - Right channel of microphone or line input.

Available:

SynFactory: not available.

StudioFactory: All versions.

## 5.9 Interface Modules

### 5.9.1 Joystick - Joystick controlled module

If an (analog) joystick is connected to the machine, this module can be used to extract position information from the joystick to control various parameters. Joysticks with upto 6 axis and 4 buttons are supported.

Outputs:

**X** - Returns position on the horizontal axis (range -32767 to 32767).

**Y** - Returns position on the vertical axis (range -32767 to 32767).

**Z** - Returns position on the third axis, this is often the throttle control (range -32767 to 32767).

**R** - Returns position on the fourth axis (range -32767 to 32767).

**U** - Returns position on the fifth axis (range -32767 to 32767).

**V** - Returns position on the sixth axis (range -32767 to 32767).

**B1** - Returns state of button 1 (32767 if held down, 0 if not pressed).

**B2** - Returns state of button 2 (32767 if held down, 0 if not pressed).

**B3** - Returns state of button 3 (32767 if held down, 0 if not pressed).

**B4** - Returns state of button 4 (32767 if held down, 0 if not pressed).

Available:

SynFactory 1.15 or higher version.

StudioFactory: All versions.

### 5.9.2 Scope - 4 channel scope

This is a seperate scope, use it to debug the patch or visualize one or more parameters. Double click on the module to open the scope Window. It can display upto 4 channels at the same time. The 3 numbers represent the running average and the lowest and highest peak value seen in the current timeslot.

Inputs:

**Ch 1** - First signal input. Data will be shown in left-top corner of scope window.

**Ch 2** - Second signal input. Data will be shown in right-top corner of scope window.

**Ch 3** - Third signal input. Data will be shown in left-bottom corner of scope window.

**Ch 4** - Fourth signal input. Data will be shown in right-bottom corner of scope window.

Available:

SynFactory: Version 1.15 or higher.

StudioFactory: All versions.

## 6 Midi implementation chart (SynFactory Patch)

Function		Transmitted	Recognized	Remarks
Basic Channel	Default	X	1 – 16	Memorized
	Changed	X	1 – 16	
Mode	Default	X	2, 4	Memorized
	Messages	X	X	
	Altered	*****	X	
Note Number	True Voice	X	0 – 127	
		X	0 – 112	
Velocity	Note On	X	1 – 127	Scaled to 0 – 32767
	Note Off	X	0 – 127	
After Touch	Key's	X	X	
	Ch's	X	O	
Pitch Bender		X	O	14 bits resolution, scaled to -32768 – 32767
Control Change	1	X	*	Modulation
	2	X	*	Breath
	3	X	*	
	4	X	*	FootPedal
	5	X	*	Portamento Time
	7	X	*	Volume
	8	X	*	Balance
	9	X	*	
	10	X	*	Pan
	11	X	*	Expression
	12	X	*	Efx Control 1
	13	X	*	Efx Control 2
	14	X	*	Efx Control 3
	15	X	*	Efx Control 4
	16 – 19	X	*	Slider 1 - 4
	41 – 48	X	*	Knob 1 - 8 (eg AN1x)
	65, 66, 67, 68 or 69	X	Sustain (patch)	Sustain
	70	X	*	Sound Variation
	71	X	*	Sound Timbre
	72	X	*	Sound Release Time
	73	X	*	Sound Attack Time
	74	X	*	Sound Brightness
	75	X	*	Sound Control 6
	76	X	*	Sound Control 7
	77	X	*	Sound Control 8
	78	X	*	Sound Control 9
	79	X	*	Sound Control 10
0 – 127	X	*	User programmable	
Prog Change	True	X	X	
		X	X	
System Exclusive		X	X	
System Common	Song Pos	X	X	
	Song Sel	X	X	
	Tune	X	X	
System Realtime	Clock	X	O	
	Commands	X	O	
Aux	Local On/Off	X	X	
	All Notes Off	X	X	
	Active Sense	X	X	
	Reset	X	X	
Notes	* Control signals can be wired to any desired function within the patch.			

## 7 File Formats

### 7.1 SYN file format (SynFactory 1.x)

#### 7.1.1 Introduction

There is only a single file format for all versions of SynFactory. This makes it easy to transfer files between different versions of the program. Although the file format is the same, there are some subtle difference in the files generated by the different version of SynFactory. The only format differences between 1.x and 2.0beta versions of SynFactory is an extra section and the possibility to have multiple patches in one file. Although the beta versions are not officially supported, StudioFactory can load both the SYN (1.x) and SYF (2.0) files.

The file format is designed to provide full compatibility between different versions, newer versions can always load older syn-files, but the reverse isn't always true. Because new modules will be added, using these modules in patches can make it impossible to load these files in older versions of SynFactory. A module with three questions marks can appear in the patch if an unsupported module was used. Most incompatible files are rejected with an error message.

The patch files are saved in ASCII format. Most information is identified by a ASCII based name token. As the file can contain information not understood by all versions of SynFactory. All tokens not supported are simply skipped on load. The end of a command or piece of information can be determined by a line-break (CR+LF on windows machines).

During parsing of .syn files be prepared to deal with very long lines (>1000 chars). Don't assume 80 or so characters, because some commands can have many parameters and would overflow your storage buffers (this is one of the most common bugs in many programs). You have been warned!

#### 7.1.2 Layout

Patch files contain multiple sections. A section is identified by the special sequence {{{ and the end with }}}. Sections can be nested inside each other and are identified with a name placed behind the {{{ and the }}} tokens.

The order of the sections in the file is not defined and the same type of section can occur multiple times. Which sections are present in the file depends on the patch saved and the way it is saved. The 'autoload.syn' file for example has a number of extra sections for storing configuration and settings, these are normally not found in other patch files.

All files begin with a fixed header to identify it as a .syn-file (this header is *not* stored in a section). After the header come zero or more sections each identified with special section token. Currently the following section tokens are in use:

- {{{FILEPATHS – Names of loaded files and other file system related information.
- {{{SETTINGS – Section to store user setting for the operation of SynFactory.
- {{{WINDOWS – Section which contains all window locations to reposition them on restart.
- {{{SAMPLEBANK – This is a container for sections dealing with samples and sampling.
- {{{SAMPLES – Alias for SAMPLEBANK (2.0+ only).
- {{{SAMPLE – Description for a single sample.
- {{{PATCHES – Section which contains zero or more PATCH sections (2.0+ only).
- {{{PATCH – This is a container for other sections specifying details about the complete patch.
- {{{MODULE – Description for a single synthesizer module. Describes position, layout and settings of the

module.

### 7.1.3 Header

The header consists of 3 lines which have a fixed layout. Changing only a single character in this header will result in a reject by SynFactory to load your file.

```
FILETYPE SYN
PROGRAM SynFactory
VERSION 1.00
```

Or for SynFactory 2.0 and above:

```
FILETYPE SYN
PROGRAM SynFactory
VERSION 2.00
```

After the header is the path used to store this file. While this is not used by SynFactory and is not an official part of the header, most files saved by SynFactory will have this as the 4th line in the file. After these four lines come the sections. There is no footer or other file terminating sequence. The loading ends when EOF (End Of File) is detected.

```
NAME C:\SynFactory\Shepard\_01.syn
```

### 7.1.4 Filepath section

On startup SynFactory will redisplay the patch which was active while SynFactory was closed. From SynFactory 1.12 onward the filename of the patch is also restored. The following setting is used to save this filename:

```
CURRENTFILE C:\SynFactory\Shepard_01.syn,Shepard_01.syn
```

### 7.1.5 Settings section

All flags and settings in this section can be changed from within synfactory so it is not recommended to change these values from within another tool. Noted here for reference only.

```
TOOLBAR <flag>
```

<flag> can be 0 or 1. If 0 then toolbar is off, when 1 then toolbar is visible. The default value is 1.

```
MAINMENU <flag>
```

<flag> can be 0 or 1. If 0 then main menu is invisible. The default is 1 and is recommended setting.

```
PATCHCABLES <type>
```

<type> can be 0, 1 or 2.

```
BUFFERSIZE <size>
```

<size> is one of 256, 512, 1024, 2048, 4096, 8192 or 16384.

This controls the buffer size for the DSP engine. I recommend to change this value by use of SynFactory only. Incorrectly setting of this value can cause undesired misbehavior of SynFactory or your system sound drivers (blue screen bye windows :-).

NROFBUFFERS <count>

Number of buffers used to drive the audio output. Only used for some audio drivers. Valid range depends on SynFactory version but is normally 3 to 10.

SCOPEMODE <mode>

<mode> can be 0 or 1. This setting controls the way the Output Scope displays its waveforms. 0 will display the channels are above each other. 1 will display the channels next to each other.

JOYSTICKMODE <mode>

<mode> selects which joystick input is used.

### 7.1.6 Windows section

Stores location and size of the tool windows. There are two possible commands for the scope-window. Only one is present in the file. From version 1.13 the scope window is resizable, so 2 extra parameters need to be stored. The SCOPEWINDOW2 token is added to not break backwards compatibility. Only problem is when loading a 1.13+ autoload.syn file in an older version the scope window will not be displayed and the stored location and size will be lost (because the SCOPEWINDOW command is not present).

SCOPEWINDOW <flag>, <x>, <y>

<flag> can be 0 or 1. If 0 then the scope window is not visible, when 1 the scope window is visible.  $x_0$  is the horizontal location of the window. <y> is the vertical location of the window.

SCOPEWINDOW2 <flag>, <x>, <y>, <xs>, <ys>

<flag> can be 0 or 1. If 0 then the scope window is not visible, when 1 the scope window is visible. <x> is the horizontal location of the window. <y> is the vertical location of the window. <xs> is the width of the window in pixels. <ys> is the height of the window in pixels.

TRANSPORTWINDOW <flag>, <x>, <y>

<flag> can be 0 or 1. If 0 then the transport window is not visible, when 1 the transport window is visible. <x> is the horizontal location of the window. <y> is the vertical location of the window.

MIXERWINDOW <flag>, <x>, <y>, <xs>, <ys>

<flag> can be 0 or 1. If 0 then the mixingdesk window is not visible, when 1 the mixingdesk window is visible. <x> is the horizontal location of the window. <y> is the vertical location of the window. <xs> is the width of the window in pixels. <ys> is the height of the window in pixels.

### 7.1.7 Samplebank section

Container section for storing SAMPLE sections.

### 7.1.8 Samples section

Container section for storing SAMPLE sections.

### 7.1.9 Sample section

This section stores information for a single sample. These samples are used in the Tracker-sequencer. Because sample data can contain any character they are encoded to remove any ASCII control characters which might give cross platform inconsistency.

### 7.1.10 Sampledata encoding

The sample data is coded in a 'base-85' encoding. 4 bytes are packed to 5. If the file is not a multiple of 4, extra zero's are added during encoding phase but are not counted in the length fields. These always give the real length of the sample. Four bytes are taken as an unsigned 32bits long (filling from MSB to LSB) and are 5 times divided by 85. The modulus (% operator in C) is used to get the remainder of the division and with 33 added this is written to the file. Two small improvements are made to save some of the space, because this code gives 25% overhead. If the 32 bits integer is 0 we do not write '!!!!' but 'z'. This saves 4 chars on each 32 bits integer which is zero (and zero is very common in samples because it is silence!). And if current 32 bits integer is same as previous 'x' is written instead of the full encoding. The 'z' does NOT reset the state of the previous storage value! So multiple zero's are written as 'zzzzz' and not as 'zxxxx'!

## 7.2 STF file format (StudioFactory)

### 7.2.1 Introduction

This is the StudioFactory native file format. It follows basically the same logic as the .SYN file format, but is binary format instead of an ASCII format. Because of it's binary format, it can store samples and other binary data much more efficiently.

### 7.2.2 Layout

The file consist of a header followed by one or more chunks with data. Each chunk is identified with a name which determines the type. Each chunk also contains a field that specifies the length of the chunk. This way the file format can be extended without breaking older software. The new chunks or extra data fields are simply ignored by older software. The rule is only adding new fields after the old ones. The software should always use the length given in the chunk and never assume anything on the chunk name. There is no predefined contents of the files, all depends on the specific data stored. The order of the chunks however might be important for the application, so external tools should try to preserve the chunk order when that is possible.

### 7.2.3 Encoding

Integers are stored as 4 bytes with the MSB first (big endian). Strings start with an integer (4 bytes, MSB first) with the length of the string counted in characters, followed by the characters themselves. No terminating 0, padding or aligning characters are added. The next field can therefore start on an odd offset in the file. Text strings have a length field at front and no embedded terminate characters and can therefore also used to store binary data like samples and bitmaps. Each data chunk starts with the name of the chunk (stored as STF string with 4 bytes for the length followed by the characters of the name). The minimum chunk name length which should be supported by applications is 31 characters. The name is followed by an integer that tells the length of the data in the chunk (excluding chunk name and this integer). So for

an empty chunk this integer is zero. The data inside each chunk is application depended. For chunks not recognized by a particular application the length field can be used to skip to the next chunk.

#### 7.2.4 Header

The header is an empty chunk (24 bytes total length) with the name "STUDIOFACTORY100" (hex 00 00 00 10 53 54 55 44 49 4F 46 41 43 54 4F 52 59 31 30 30 00 00 00 00). This header must be the first chunk in the file starting at offset 0. This marker enables StudioFactory to quickly identify the format of the file by inspecting only the first few bytes.

#### 7.2.5 StudioFactory configuration data

ProgramLanguage

menu and help language (string: DE — EN — FR — IT — LT — NL)

PatchCableType

cable type (integer 0 blacklines — 1 colored lines — 2 virtual patch cables)

MAINWINDOW

HELPWINDOW

COLORSELECTWINDOW

TRANSPORTWINDOW

SCOPEWINDOW2

AudioMixerWindow

MIDIACTIVITYWINDOW

MIDISCOPEWINDOW

PROJECTWINDOW

CONFIGWINDOW

MidiInDevice

device (integer 1 to maximum devices in system)

port (integer 0 to 25 for port A-Z) MidiOut-

Device

device (integer 1 to maximum devices in system)

port (integer 0 to 25 for port A-Z)

#### 7.2.6 Project data

ProjectBegin projectname (string) ProjectPath absolute path with filename (string) Project-Modified (empty chunk only used in autoload.stf, because this file can contain unsaved projects) Samples (see !!!) SynFactory patches (see !!!) Sequencer data (see !!!) ProjectEnd (empty chunk)

#### 7.2.7 Sample and instrument data

SampleBegin (chunk) sample id (integer) sample name (string) SampleRate SampleEncoding SampleLoopPoints SampleData SampleEnd (empty chunk)

#### 7.2.8 SynFactory patch data

PatchBegin (chunk)

patch id (integer)

patch name (string)

PatchMidi (chunk)  
 MIDI port (integer 0 off, 1 .. 26 for port A .. Z)  
 MIDI channel (integer 0 all, 1 .. 16 single channel)  
 MIDI sustain CC number (integer 0 off, 64, 65, 66, 67, 68 or 69)

ModuleBegin (chunk)  
 module id (integer 1..n. It must be unique for a patch. Gaps in the numbering are allowed.)  
 module type (string, refer to the source code for possible values)

ModulePosition (chunk)  
 x position (integer)  
 y position (integer)

ModuleLabel (chunk)  
 label visible (integer 0: label is not visible, 1: label is visible)

ModuleName (chunk)  
 string with either text when it is a 'text' module or the label when any other module.

ModuleCable (chunk)  
 input number (integer 0..31)  
 module where cable comes from (integer 1..n)  
 output on module where cable comes from (integer 1..32)  
 cable color index (integer 0..7)

ModuleKnob (chunk)  
 knob id (0..31)  
 value (-32768..32767)

ModuleMode (chunk)  
 mode (integer 0..5 for OSC, not used for other modules)

ModuleScript (chunk)  
 script (string)

ModuleBuffer (chunk)  
 buffer (binary string)

ModuleSize (chunk)  
 x size (integer)  
 y size (integer)

ModuleColor (chunk)  
 foreground color (integer 24 bit RGB)  
 background color (integer 24 bit RGB)

ModuleEnd (empty chunk)  
 PatchEnd (empty chunk)

## 7.2.9 Bitmap data

BitmapBegin  
 bitmap id (integer)  
 bitmap name (string)

BitmapSize  
 x,y,colors,aspect x,y

BitmapPalette

BitmapEncoding  
BitmapPlanes  
BitmapEnd

#### **7.2.10 Sequencer data**

TrackBegin  
    track id (integer) SectionBegin  
SectionEnd  
TrackEnd

#### **7.2.11 Embedded file data**

ObjectBegin  
ObjectName  
ObjectPath  
ObjectEncoding  
ObjectData  
ObjectEnd

### **7.3 BMP file format (bitmap)**

This file format is not yet supported.