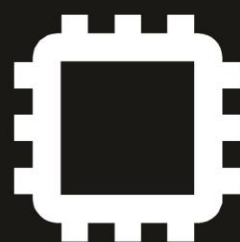


Soundigy



MIDI PROCESSOR
v 1.1

operation manual

The information in this document is subject to change without notice and does not represent a commitment on the part of Soundigy. The software described by this document is subject to a License Agreement and may not be copied to other media. No part of this publication may be copied, reproduced or otherwise transmitted or recorded, for any purpose, without prior written permission by Soundigy. All product and company names are trademarks of their respective owners.

And also, if you're reading this, it means you bought the software rather than stole it. It's because of people like you that we can continue to create great tools and update them. So, thank you very much.

© Soundigy, 2015. All rights reserved.
Second Edition, February 2015



Soundigy
os. Oswiecenia 37/11
31-636 Krakow
Poland
info@soundigy.com
www.soundigy.com

By using this software you agree to the terms of any license agreement accompanying it.
VST is a trademark of Steinberg Soft – und Hardware GmbH. All other trademarks are the property of their respective owners.

Table of contents

1. INTRODUCTION	5
1.1. WHAT IS MIDI PROCESSOR?	5
1.2. DEMO MODE	5
1.3. VST AND STANDALONE	5
1.4. TROUBLESHOOTING AND REPORTING BUGS	5
2. INSTALLATION PROCESS	6
2.1. SYSTEM REQUIREMENTS AND RECOMMENDATIONS	6
2.2. SOFTWARE INSTALLATION	6
3. PRODUCT AUTHORIZATION	7
3.1. WHAT IS THE PRODUCT AUTHORIZATION?	7
3.2. CONDUCTING THE PRODUCT AUTHORIZATION	7
3.3. REGISTRATION SUPPORT	8
4. FIRST STEPS IN MIDI PROCESSOR	9
4.1. THE MAIN WINDOW	9
4.2. SIGNAL AND CONTROL FLOW	10
4.3. LOADING AND SAVING PRESETS (SCENES)	11
5. MESSAGES AND EVENTS	13
5.1. CHANNEL MESSAGE	13
5.2. WAIT EVENT	14
5.3. BANK/PROGRAM CHANGE EVENT	15
5.4. CHANGING CHANNEL MESSAGE	15
5.5. STOP CHANGING CHANNEL MESSAGE	16
5.6. SYSTEM EXCLUSIVE MESSAGE	16
5.7. GENERIC MIDI MESSAGE	17
5.8. MIDI FILTER/TRANSFORM MESSAGE	18
5.9. CANCEL MIDI FILTER/TRANSFORM MESSAGE	19
5.10. SEND GROUP MESSAGE	19
5.11. VARIABLE EVENT	20
6. OTHER FEATURES	21
6.1. LIST OF MESSAGES	21
6.2. VARIABLES	22
6.3. TRIGGERS	23
6.4. GROUPS	24
6.5. LINKS	25
6.6. PASS ON CH/MAKE CH	26
6.7. PANIC BUTTONS	27
7. GENERAL OPTIONS	28
7.1. GLOBAL OPTIONS	28
7.2. SCENE TRANSFER	29
7.3. SCENE'S SETTINGS AND BUTTONS	30
8. STANDALONE VERSION	32
8.1. PRESETS SECTION	32

9. ADDENDUM..... 35

9.1. WORKING WITH SCENES AND PRESETS FROM OLDER VERSIONS..... 35

9.2. SHARING SCENES BETWEEN VST AND STANDALONE 35

1. Introduction

1.1. What is MIDI PROCESSOR?

MIDI PROCESSOR is a very powerful tool for manipulating MIDI messages in real time. It combines a few ideas in one place – manager, merger/splitter, filter, transformer, generator and synchronizer. It supports multiple MIDI inputs and outputs simultaneously, however it is also possible to run many instances of MIDI PROCESSOR in the same time. It allows you to process incoming MIDI messages as well as generate new MIDI messages and send them into MIDI output(s). All settings can be loaded or saved as presets. It contains many useful types of events that you can use in your configurations. There are also some additional features (groups, links, variables and triggers) provided for

1.2. Demo mode

Before buying the software you can try it for free by downloading an installer from our website (www.soundigy.com/download.php). Then it will work in demo mode (all features are available, but messages are filtered out randomly and from time to time you may see a message informing you about the demo mode). All these annoying limitations will disappear after registering the product.

1.3. VST and standalone

The software comes and is installed in two versions: VST plugin and standalone executable. This manual has been created in respect of the VST version. There are many differences between these two versions, however the standalone version has been described in chapter 8.

1.4. Troubleshooting and reporting bugs

Despite our best efforts to provide for you software free of bugs, it is actually almost impossible to test any software and say it is certainly reliable in all cases. If any problems occurs when using our software, in first place please use our forum at <http://www.soundigy.com/forum> and try to find answer there or create an article asking other users if they could help. To report bugs/errors in the software, contact us at support@soundigy.com directly.

2. Installation process

2.1. System requirements and recommendations

To use the MIDI PROCESSOR software, you need a computer with the following minimum specifications:

Hardware

- Pentium III 1 GHz/ Athlon XP 1.33 GHz
- 512 MB RAM
- A sound card compatible with Windows XP

Software

- Microsoft Windows XP
- Microsoft .NET Framework 2.0

2.2. Software installation

- Download the MidiProcessorSetup (x86).msi or MidiProcessorSetup (x64).msi file from www.soundigy.com/download.php.
- Open the folder containing the downloaded setup file.
- Start the installation by double-clicking the file. The wizard will guide you through the installation process. You will be asked to accept End-User License Agreement (EULA).
- The setup will suggest [OS HDD]\Program Files\Steinberg\VstPlugins\Soundigy\MidiProcessor\ as the path for the destination folder for a .dll file (VST plugin). You may also choose another folder if you wish. On the 64-bit machines the default location is [OS HDD]\Program Files (x86)\Steinberg\VstPlugins\Soundigy\MidiProcessor\ for the .dll. The EULA and examples will be also copied into that folder.
- The setup will also install a standalone version of the software in the default location [OS HDD]\Program Files\Soundigy\MidiProcessor\. This version is just a single executable file MidiProcessor.exe so you may copy it into any place you want later.

Note: If more than one host program for VST 2.0 plug-ins is installed on your computer, you may want to use multiple VST-folders. After installing the software, you can simply copy the “MidiProcessor.dll” file into the VST plug-ins folders of these programs. Optionally, you can set up your programs so that they all use the same VST plug-ins folder.

3. Product Authorization

Part of the MIDI PROCESSOR installation is a Product Authorization which has to be fulfilled in order to use the product without the demo version limitations.

3.1. What is the Product Authorization?

The Product Authorization includes a full registration. After having completed the authorization, you can make use of all the features without any limits. On the Soundigy website you can read online tutorials, download files from the Soundigy Online Library, get technical support, participate in the Soundigy forums and download updates. The Product Authorization is part of the license agreement.

As long as the software runs unauthorized a message at every program start reminds you that the authorization has not been completed yet and runs in demo mode.

The Product Authorization is performed inside the application by clicking on the Soundigy logo in the right bottom corner of the application's window and then on the Register button. The registration process contains a few steps.

First, after making a purchase, you will receive your unique Serial Number. After putting a valid Serial Number in MIDI PROCESSOR and clicking Generate Activation Code, the so called Activation Code is generated which serves as request code for receiving a License Key. The Activation Code is based on some hardware components of your computer system, the operating system version and the serial number you have entered before.

Note: Exchanging your audio card, MIDI interface or external equipment will not start the request for a new Authorization Code. Only exchanging a basic hardware component (e.g. CPU, HDD) in your computer or installing a new operating system might produce a new Product Authorization request. In this case you have to repeat the Product Authorization. Updating the software to newer version (i.e. v1.1 to v1.4) doesn't require repeating the Product Authorization. However installing new edition of the software (i.e. v1.4 to v2.0) does, as usually you have to buy new license, because editions of even the same software are treated as different software and are licensed separately.

The Authorization Code has to be sent to Soundigy in order to receive the License Key which allows the unlimited use of MIDI PROCESSOR.

3.2. Conducting the Product Authorization

Soundigy has set a high value on making the authorization procedure as easy and comfortable as possible. Below we put all steps needed to successfully authorize the product.

- Open a Registration form of the MIDI PROCESSOR (by clicking on the Soundigy's logo in the right bottom corner of the main MIDI PROCESSOR's window and then Register button). If you don't see the Register button, it means you have already authorized MIDI PROCESSOR.



- Put a valid Serial Number in the right field. Serial Number can be only obtained after buying MIDI PROCESSOR.
- Put your or your company's name in the field below. It may contain letters and digits only.
- Click Generate Activation Code button on the bottom of the form. If the Serial Number is correct, the Activation Code should be generated. This step should be performed on the computer on which you will use the software.
- Send the generated code by email (license@soundigy.com), fax (+48 12 6489834) or conventional mail service to our address:

Soundigy
 os. Oswiecenia 37/11
 31-636 Krakow
 Poland

- The License Key will be sent to you using the same method as used by you, i.e. email, fax or mail.
- After receiving the License Key, paste it into the proper field on the Registration form and click Register using License Key button below (you don't have to put Serial Number and Name again). This step should be performed on the computer on which you will use the software. If the License Key is correct, your MIDI PROCESSOR is registered and all limitations are gone. Thank you and congratulations!

3.3. Registration support

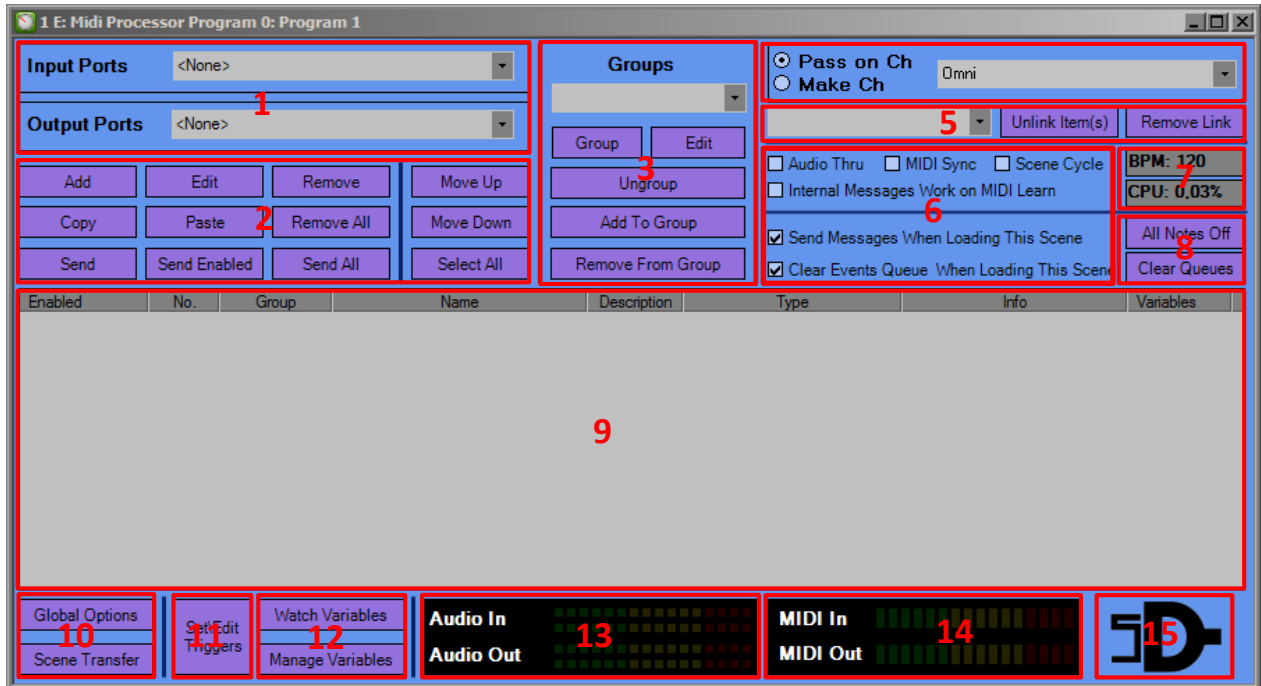
If you run into problems during the Product Authorization the Soundigy registration support team will be happy to help you. In this case send a support request to support@soundigy.com.

Please describe the occurred problem as accurate as possible and provide the registration support team with the necessary details to solve the problem.

4. First steps in MIDI PROCESSOR

The purpose of this chapter is to make you familiar with the basics of the operation and the functionality of MIDI PROCESSOR and how to program it.

4.1. The main window



Note: The look may be slightly different for various VST hosts.

The main window of MIDI PROCESSOR is divided into few sections.

- | | |
|--|-------------------------------------|
| 1) MIDI sources and destinations | 9) List of Messages |
| 2) Operations on messages | 10) General operations and settings |
| 3) Groups section | 11) Triggers manager |
| 4) Channel filter | 12) Variables operations |
| 5) Links section | 13) Audio signals indicator |
| 6) Miscellaneous scene's (preset's) settings | 14) MIDI signals indicator |
| 7) Information displays | 15) About/Registration link |
| 8) Panic buttons | |

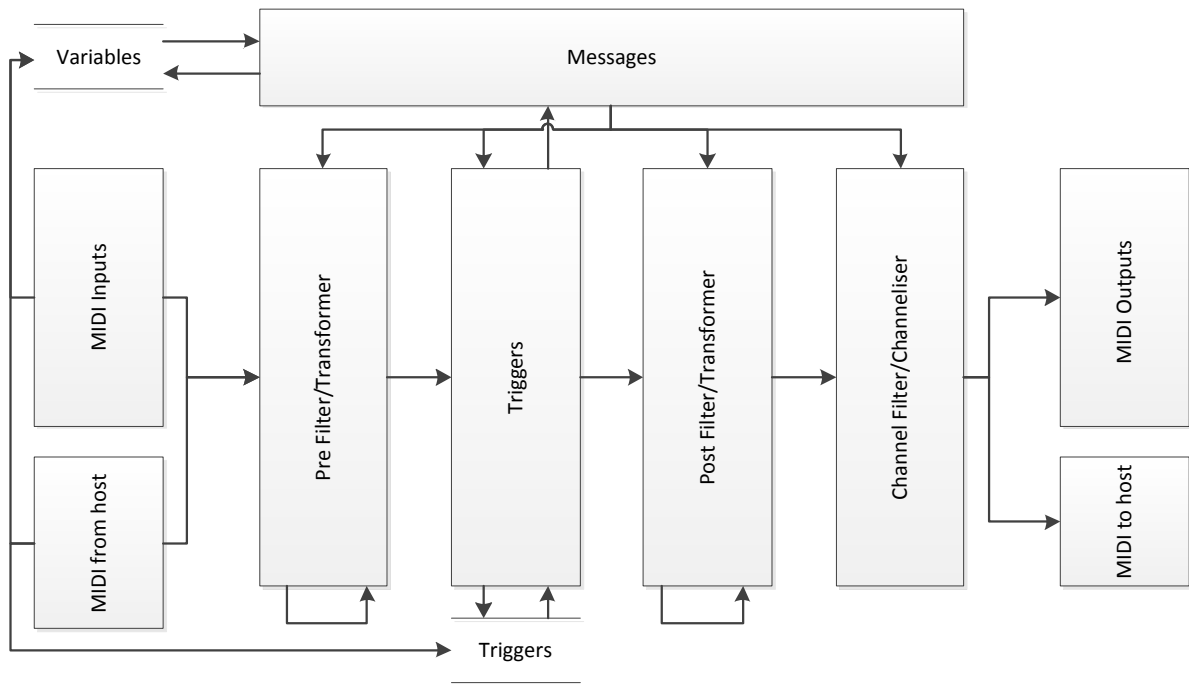
You will learn how and when to use above sections in the next chapters. If you want to create and send your first message, please follow the steps below:

- 1) Click Add button.
- 2) Choose Bank/Program Change message type.
- 3) As a name put "Set program to 10", set the Program field value to 10 and click OK button.
- 4) Choose at least one output MIDI port where you want to send the program change message (on the Output Ports list you should see both physical and virtual MIDI ports and <To Host> option, which routes signal to the VST host).
- 5) Click Send All button.
- 6) You should now see LEDs' short flash on the MIDI Out indicator and current program to be set to 10 on all devices/instruments connected to the previously chosen MIDI outputs.
- 7) If you want to change anything in the program change message, simply double click it with your left mouse click and you should see the message edit window again.

4.2. Signal and control flow

MIDI PROCESSOR signal processing engine is the core unit of the software. It is a very powerful and flexible element. Thus in some situations it might be quite hard to find out how the signal flows thru the engine. Recalling three rules should help then.

- 1) Enabled messages are sent from top to bottom. However, some messages may only turn on/off some functions/features, other may force engine to send other messages.
- 2) Sent messages are put in a queue. Messages from the queue are then sent forward.
- 3) The engine contain a few subunits (i.e. triggers and filters). The following diagram shows how they all are connected.



4.3. Loading and saving presets (scenes)

One of the main ideas behind the creation process of MIDI PROCESSOR was using it with VST hosts during live performances on stage to control and quickly reconfigure your rack setup. Scenes (or presets, using Steinberg's nomenclature) can be saved/loaded by VST host (using so called plugin persistence). But you can also save/load scenes to/from files from the plugin's side. You may think of a single scene as a unit (a file) with all settings and data saved in it.

Loading scene from sample file

Note: After loading new configuration from a file old settings are gone.

- 1) Click the Scene Transfer button in the left bottom corner of the MIDI PROCESSOR window.
- 2) Click Read Scene Settings From File.
- 3) Navigate to a folder with sample files for MIDI PROCESSOR (default location: C:\Program Files\Steinberg\VstPlugins\Soundigy\MidiProcessor\Examples\).
- 4) Choose Example1.sst and click OK.

After performing these steps you should see new configuration – 4 messages. As there are many different MIDI interfaces available, this example doesn't contain any MIDI port configuration (however it is possible of course). If you turn on some MIDI output port and click Send All button you should see the effect if there is any device connected to the selected port.

Saving scene settings to new file

- 1) Make sure there isn't any messages selected on the List of Messages (if there is any, click an empty area inside the List of Messages).

- 2) Add new message of Wait type for example. To do it, click Add button and choose Wait as a type.
- 3) In the field “ms” put a value 4000, so before any other message there will be 4 seconds break.
- 4) Click the Scene Transfer button in the left bottom corner of the MIDI PROCESSOR window.
- 5) Click Save Scene Settings To File.
- 6) Choose any location you want.
- 7) Put any name you want, i.e. Example1Mod and click OK
- 8) To check if all the settings were saved successfully, change something in the current configuration and then try to load settings from the file saved before.

5. Messages and events

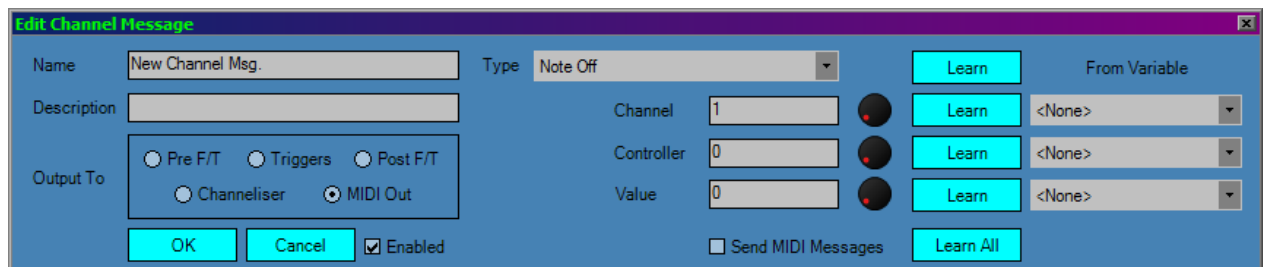
In this chapter we will present and discuss all message and event types in the current version of MIDI PROCESSOR. As the software evolves this list may become longer, and hopefully will. To add even more fuel to the fire we can reveal you a secret – we are currently working on a few completely new and innovative types of events, including a very sophisticated and complex Audio To MIDI unit.

5.1. Channel Message

MIDI channel (or MIDI channel voice) messages are the most basic and common MIDI messages. They are used to send a single control signal. There are seven types of channel messages:

- 1) Note Off,
- 2) Note On,
- 3) Poly Pressure (or Poly Aftertouch),
- 4) Control Change,
- 5) Program Change,
- 6) Mono Pressure (or Mono Aftertouch),
- 7) Pitch Wheel Change.

Below you can see this message type window. We will discuss it now more deeply as other types of messages have similar look and options.



Name: Identifier of a message (doesn't have to be unique) – when referencing this message from other places (i.e. triggers), name is used to identify the message.

Description: Additional piece of information for the message.

Output To: The place in the engine where the message is sent. As you can see on the signal and control flow diagram, signal from messages may be routed to different subunits. The meaning of these subunits will be explained later.

Enabled: If checked, this message will be sent when loading this scene (if Send Messages When Loading This Scene on the main window is checked) or clicking Send Enabled on the main window.

Type: Type of the channel message (one of the seven types mentioned above).

Channel: MIDI channel number of the message. Can be any integer value between 1 and 16 (some other types of values in MIDI PROCESSOR start from 0, not 1; however in terms of MIDI status byte,

channel values are scaled to integer numbers 0-15 and this way they are treated when used in connection with variables). Can be set using text field or knob.

Controller: Number of controller or note to be sent. Legal values are integers between 0 and 127. Can be set using text field or knob.

Value: Value of the controller to be sent. Legal values are integers between 0 and 127. Can be set using text field or knob.

Send MIDI Messages: If checked, changing Type, Channel, Controller or Value field will cause a new MIDI message to be composed from current values of these fields and sent immediately to MIDI output ports directly.

Learn: When clicked, the field's on the left side value will be set to the appropriate value taken from incoming MIDI messages. Listening to incoming messages is performed on all MIDI input ports selected on the Input Ports list on the main screen of MIDI PROCESSOR.

Learn All: Works like Learn, but for all the four values simultaneously (Type, Channel, Controller and Value).

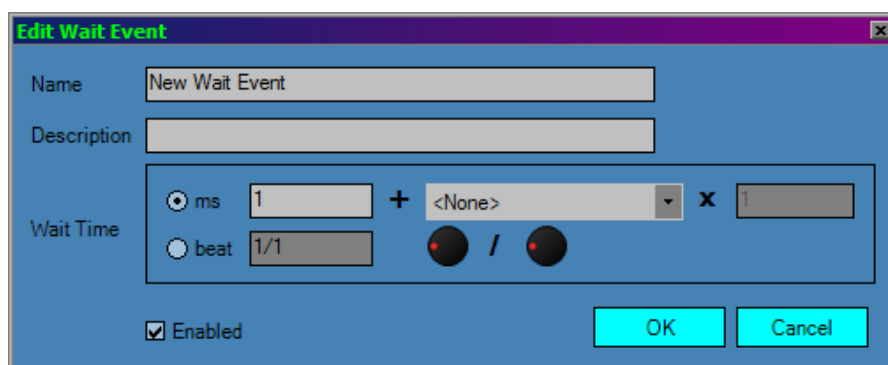
From Variable: For the Channel, Controller or Value field there can be additionally a variable chosen. If there is any, then the current value of the part of the message is composed as a sum of a value in text field and current variable's value (for channel number and all other fields with indexed from 1 it's a sum of text field value minus 1 and current variable's value).

OK: Adds new message or accepts and saves all changes and closes the Edit Channel Message window.

Cancel: Doesn't add new message or forgets all changes made inside the window and closes the Edit Channel Message window.

5.2. Wait event

This type of event is used to make a break before sending another (following) events. The break time can be expressed in milliseconds and depend on a variable or can be based on current BPM (tempo). The screenshot below presents this type of window with default settings.



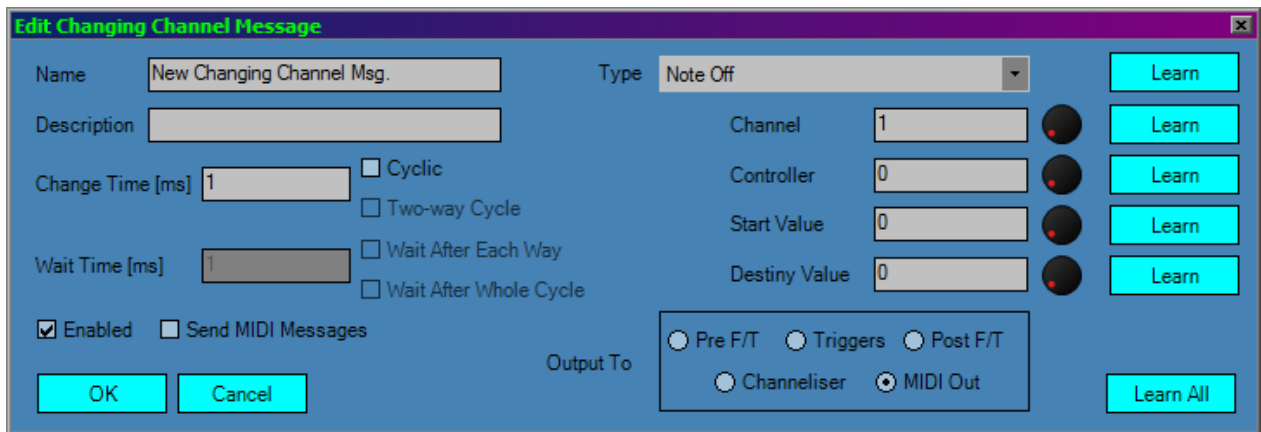
5.3. Bank/Program Change event

Using this type of event you are able to build a bank and/or program change messages in one place. You can set MSB and LSB parts of bank number and define if you want to send only program or bank number or both. All values can be variable-dependent.

The screenshot shows the 'Edit Bank/Program Change Event' dialog box. The title bar is purple and contains the text 'Edit Bank/Program Change Event'. The main area is blue. At the top, there is a 'Name' field with the text 'New Bank/Program Change Msg.' and an empty 'Description' field. Below these is the 'Output To' section with five radio buttons: 'Pre F/T', 'Triggers', 'Post F/T', 'Channeliser', and 'MIDI Out' (which is selected). The 'Bank' section is in the center, containing 'MSB' and 'LSB' fields, each with a numeric input box (both set to '1'), a red LED indicator, and a dropdown menu (both set to '<None>'). To the left of the Bank section is the 'Channel' field with a numeric input box (set to '1'), a red LED indicator, and a dropdown menu (set to '<None>'). To the right is the 'Program' field with a numeric input box (set to '1'), a red LED indicator, and a dropdown menu (set to '<None>'). Below each of these fields is a cyan 'Learn' button. A larger cyan 'Learn All' button is centered below the MSB and LSB fields. At the bottom left, there are four checkboxes: 'Send MIDI Messages' (unchecked), 'Send Bank Change Message' (unchecked), 'Send Program Change Message' (checked), and 'Enabled' (checked). At the bottom right, there are two cyan buttons: 'OK' and 'Cancel'.

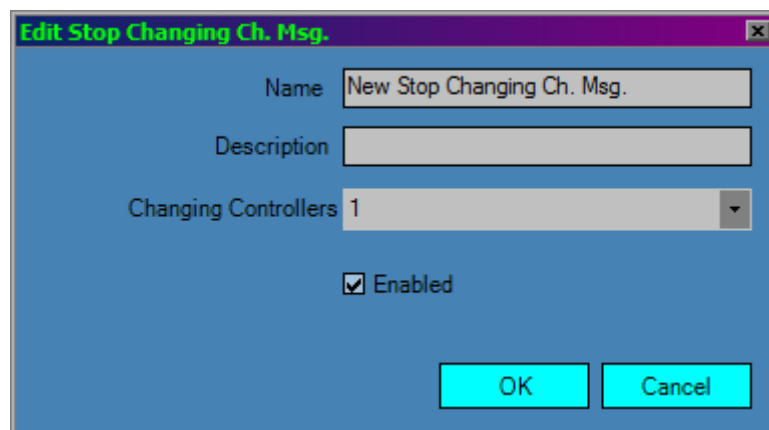
5.4. Changing Channel Message

This is a message type very similar to the first one, however it works in a different way. Sending one message like this may cause many MIDI messages to be sent further. It's because it works like a generator of messages. It allows you to define two boundary values – start and destiny. Then it will generate MIDI messages smoothly in steps and there will be a break between these steps (Change Time). If all other options are left as default, it will generate the series of messages only once. However you can also set it to run cyclically. You should then check Cyclic option. Then message will be generated from Start to Destiny value, then again from Start to Destiny and again infinitely or until stopped (i.e. by Stop Changing Channel Message event). You can also make going from Start to Destiny and back to Start and so on. Then you will have to check Two-way Cycle option too. There are also two options so there can be a break before going again from Start or from Destiny (Wait After Each Way; way is from Start to Destiny or from Destiny to Start) or alternatively you there can be a break before whole cycle (cycle is Start to Destiny plus Destiny to Start).



5.5. Stop Changing Channel Message

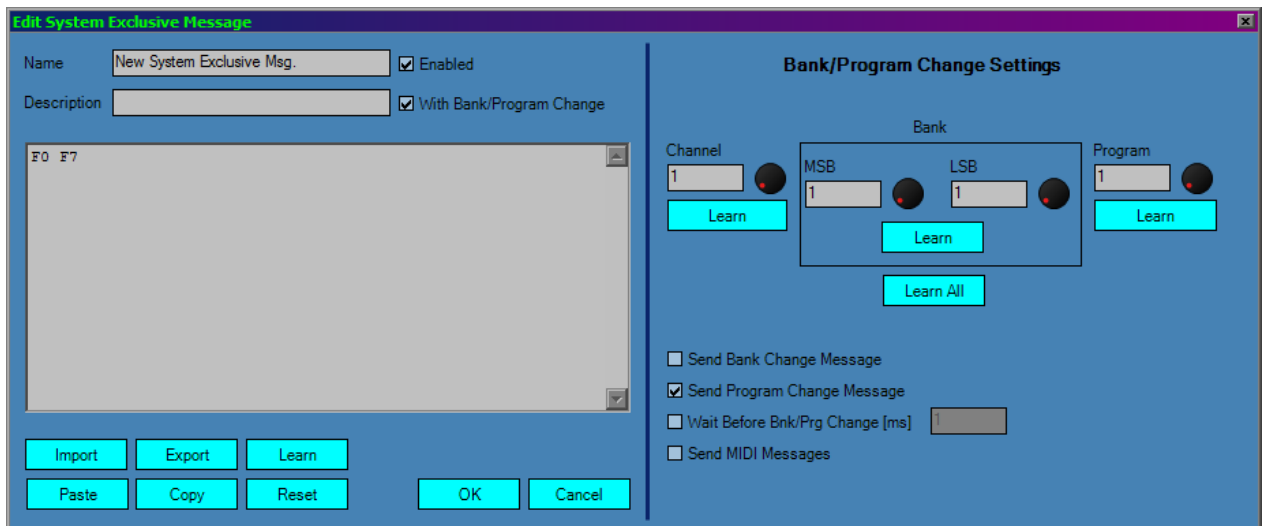
This message type can be added only if there is at least one Changing Channel Message on the list. It is used to stop on or more running message of the previous type. If you try to stop not running message nothing happens (so there won't be any error message). If you delete all Changing Channel Messages selected on the list inside this type of message, it will be automatically removed from the List of Messages.



5.6. System Exclusive Message

Here you can define a MIDI SysEx message. Optionally you can also include Bank/Program change messages after the SysEx. For SysEx message, there are some additional buttons available: Import/Export (from file), Paste/Copy (from clipboard), Learn (listens for incoming SysEx messages) and Reset (clears the whole edit window for SysEx message).

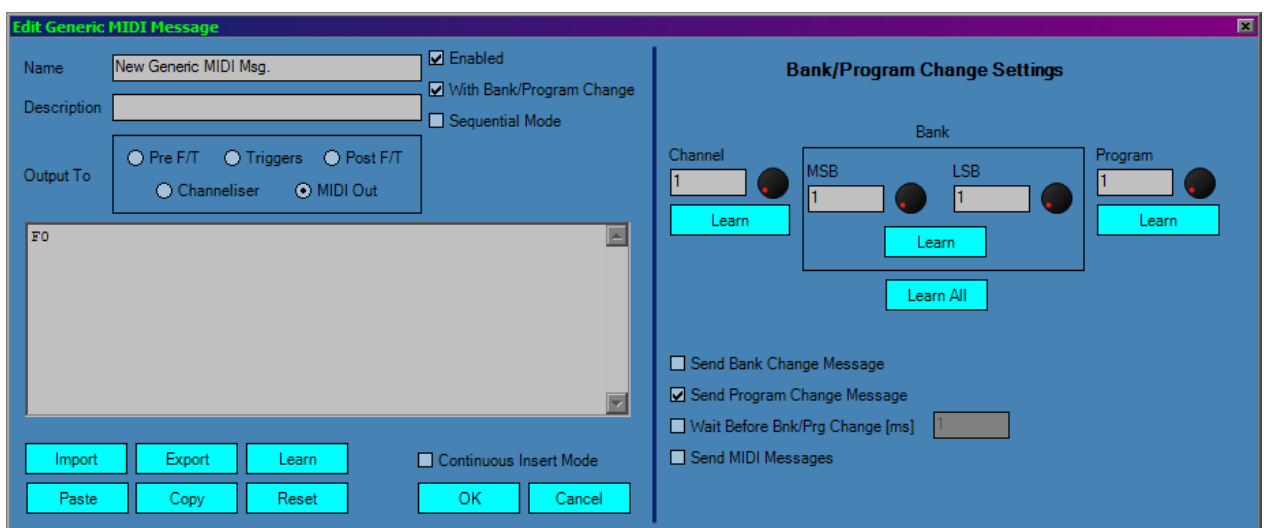
Note: Some hosts don't pass SysEx messages from MIDI inputs to VST(i) plugins. In this case you might need to use direct MIDI input on MIDI PROCESSOR.



5.7. Generic MIDI Message

It works in a very similar way to the previous type, but has a few more features:

- it supports any type of valid MIDI message, not only SysEx (it even supports few SysEx messages at once),
- if used in Sequential Mode, it can contain a series of messages, i.e. MIDI channel messages,
- you can decide which subunit of the MIDI PROCESSOR's engine should receive this message/messages,
- if in learn mode, incoming MIDI messages might be added (not overwritten) into message edit window; you should then check Continuous Insert Mode option.



5.8. MIDI Filter/Transform Message

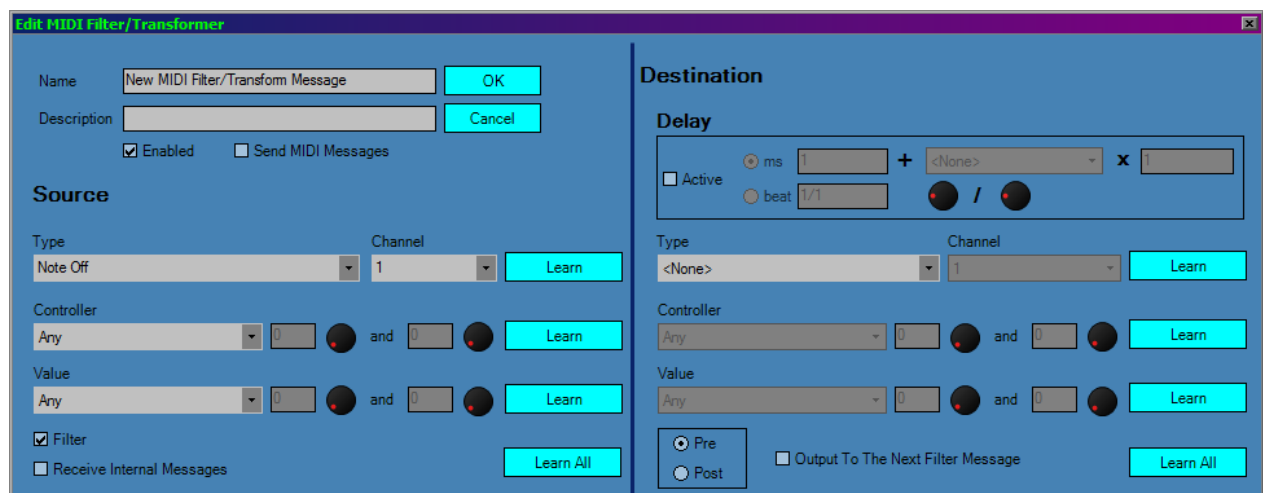
These messages are a very powerful part of MIDI PROCESSOR. Basically, if any of the incoming MIDI messages satisfies the condition on the Source's side, then it is passed to the Destination part and mapped in a way described there. If Filter option is checked, then original (incoming) MIDI messages won't be passed through. Otherwise, it will be passed to the next MIDI Filter/Transform (if there is any). Additionally, it may work before (Pre option) or after (Post option) the Triggers subunit. If Output To The Next Filter Message is checked, then outgoing (transformed) MIDI message will be passed to the next (if there is any) MIDI Filter/Transform. An order of MIDI Filter/Transform messages is how they were sent. Transformed message might be delayed using Delay panel options (it works like Wait event).

Source->Destination transformation may scale values in some situations.

From the Controller and Value lists on you can choose:

- Any
- Equal To
- Between
- Less Than Or Equal To
- Greater Than Or Equal To
- Not Equal To
- Not Between
- Add
- Subtract
- Multiply
- Divide

The last 4 options are present only on the Destination side.



5.9. Cancel MIDI Filter/Transform Message

Once a message of the previous type is sent, it works (so it transforms or filters) until stopped. Cancel MIDI Filter/Transform Message is used to stop one or more messages of the previous type. Stopping not running message doesn't produce any error. You can add this type of message if there is at least one message of the previous type. If you delete all MIDI F/T Messages selected on the list inside this type of message, it will be automatically removed from the List of Messages.



The screenshot shows a dialog box titled "Edit Cancel MIDI F/T Message". It has a blue background and a purple title bar. The dialog contains the following fields and controls:

- Name:** A text input field containing "New Cancel MIDI F/T Message".
- Description:** An empty text input field.
- MIDI F/T Messages:** A dropdown menu showing the number "2".
- Enabled:** A checked checkbox.
- Buttons:** Two buttons labeled "OK" and "Cancel" at the bottom right.

5.10. Send Group Message

This type of message may be used to send grouped messages at once. You can read more about groups in chapter 6.4. You can add this type of message if there is at least one group. If you remove all groups selected on the list inside this type of message, it will be automatically removed from the List of Messages.

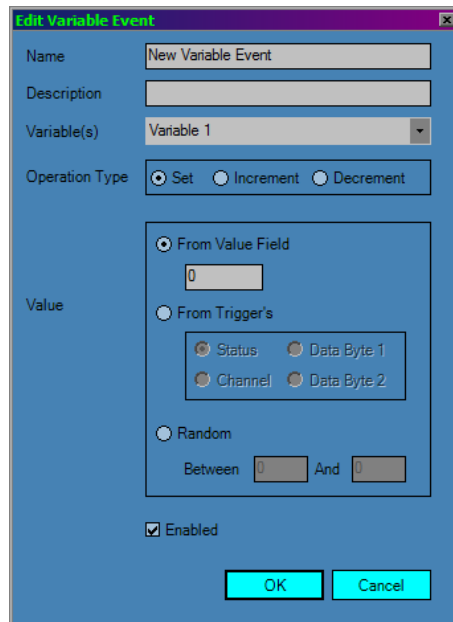


The screenshot shows a dialog box titled "Edit Send Group Message". It has a blue background and a purple title bar. The dialog contains the following fields and controls:

- Name:** A text input field containing "New Send Group Message".
- Description:** An empty text input field.
- Groups:** A dropdown menu showing the number "1".
- Enabled:** A checked checkbox.
- Buttons:** Two buttons labeled "OK" and "Cancel" at the bottom right.

5.11. Variable event

It's used to modify one or more variables' value. You can read more about variables in chapter 6.2. Variable's value may be set, incremented or decremented by a value specified in the Value panel. The value can be set to fixed value, value taken from the part of the MIDI message taken from a trigger which caused the Variable event to be sent or set to the random value within a given range.



The image shows a dialog box titled "Edit Variable Event" with a close button (X) in the top right corner. The dialog has a blue background and contains the following fields and controls:

- Name:** A text input field containing "New Variable Event".
- Description:** An empty text input field.
- Variable(s):** A dropdown menu showing "Variable 1".
- Operation Type:** Three radio buttons: "Set" (selected), "Increment", and "Decrement".
- Value:** A section with three radio buttons:
 - From Value Field:** Selected. Below it is a text input field containing "0".
 - From Trigger's:** Unselected. Below it is a sub-panel with four radio buttons: "Status", "Data Byte 1", "Channel", and "Data Byte 2".
 - Random:** Unselected. Below it are two text input fields labeled "Between" and "And" with a small downward arrow on each.
- Enabled:** A checked checkbox.
- Buttons:** "OK" and "Cancel" buttons at the bottom right.

6. Other features

6.1. List of Messages

The list contains messages and events that may be sent to MIDI outputs or internal MIDI engine. The list with some sample messages/events is shown below.

Enabled	No.	Group	Name	Description	Type	Info	Variables
<input checked="" type="checkbox"/>	[T]	1	In 1 Group(s)	New Channel Msg.	Channel Message	Type Note Off Ch 1 Num 0 = 0	
<input checked="" type="checkbox"/>		2	In 1 Group(s)	New Wait Event	Wait	1 ms	
<input checked="" type="checkbox"/>	[Link 1]	3		New Channel Msg.	Channel Message	Type Note Off Ch 1 Num 0 = 0	CV
<input checked="" type="checkbox"/>		4	In 2 Group(s)	New Changing Channel Msg.	Changing Controller	Type Note Off Ch 1 Num 0 = ...	
<input checked="" type="checkbox"/>	[R]	5		New MIDI Filter/Transform Message	MIDI Filter/Transformer		
<input checked="" type="checkbox"/>		6		New Stop Changing Ch. Msg.	Stop Changing Ch. Msg.	Msgs. To Stop: 4	
<input checked="" type="checkbox"/>		7	In 1 Group(s)	New Cancel MIDI F/T Message	Cancel MIDI F/T Msg.	Msgs. To Stop: 5	
<input checked="" type="checkbox"/>		8		New Send Group Message	Send Group Message	Groups To Send: 1	
<input checked="" type="checkbox"/>		9		New Variable Event	Variable	Variables: 1, Set: 1	
<input checked="" type="checkbox"/>	[Link 1]	10		New Channel Msg.	Channel Message	Type Note Off Ch 1 Num 0 = 0	CV

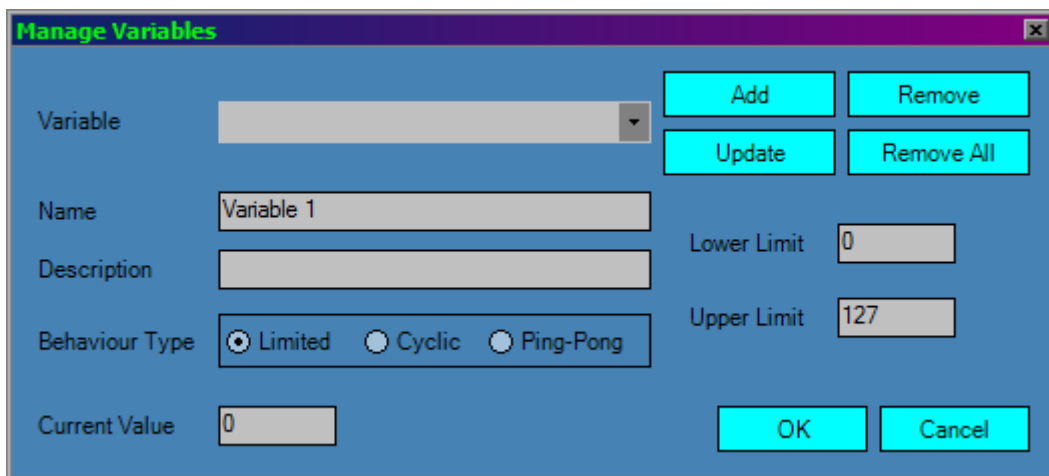
Each column corresponds to some information on messages or events.

- 1) **Enabled** – it contains a checkbox, meaning if a message is sent when loading this scene (preset) or pressing Send Enabled button, but occasionally there might occur some additional signals:
 - a. **[T]** – it means there is a trigger set for this variable,
 - b. **[R]** – appears only for MIDI Filter/Transform Messages; it informs a user that this filter/transformation was sent and is active,
 - c. **[Link ...]** – it tells a user that this message is linked with another message (so it was copied and pasted using Link, not Clone method); links are numbered 1, 2, 3,
- 2) **No.** – an ordinal number of a message or event.
- 3) **Group** – information on in how many groups is the message included.
- 4) **Name** – a label of message, by which the message is identified on lists etc.
- 5) **Description** – additional place for a comment or description of a message.
- 6) **Type** – type descriptor of a message.
- 7) **Info** – detailed information on a message, containing all the settings defined for the message in a brief form.
- 8) **Variables** – a list specifying if there are any variables used within the message. Possible values depend on the message type:
 - a. Channel Message:
 - [Ch] – for channel number,
 - [C] – for controller number,
 - [V] – for value (the second data byte of MIDI message).
 - b. Bank/Program Change Event:
 - [Ch] – for channel number,
 - [Bm] – for most significant byte value,
 - [Bl] – for least significant byte value,
 - [P] – for program number.
 - c. Wait Event:
 - [W] – for number of milliseconds.

6.2. Variables

Variables are used to add more variety to messages being sent and more automation to the application itself. Variables are kind of memory cells that can be used to store, modify and use numeric values. Current version of MIDI PROCESSOR supports messages for Channel Messages, Bank/Program Change and Wait Events. Possible values of variables are limited to nonnegative integer numbers, but are not bounded. If you try to set some MIDI message data byte like controller number to a number taken from a variable and greater than 127 (or 128 in some cases where possible values are 1-128 not 0-127) it will be converted to satisfy requirements (so its value will be 127 or 128).

To set or modify variables' value you use Variable event described in chapter 5. There are two additional windows used when working with variables: Manage Variables and Watch Variables.

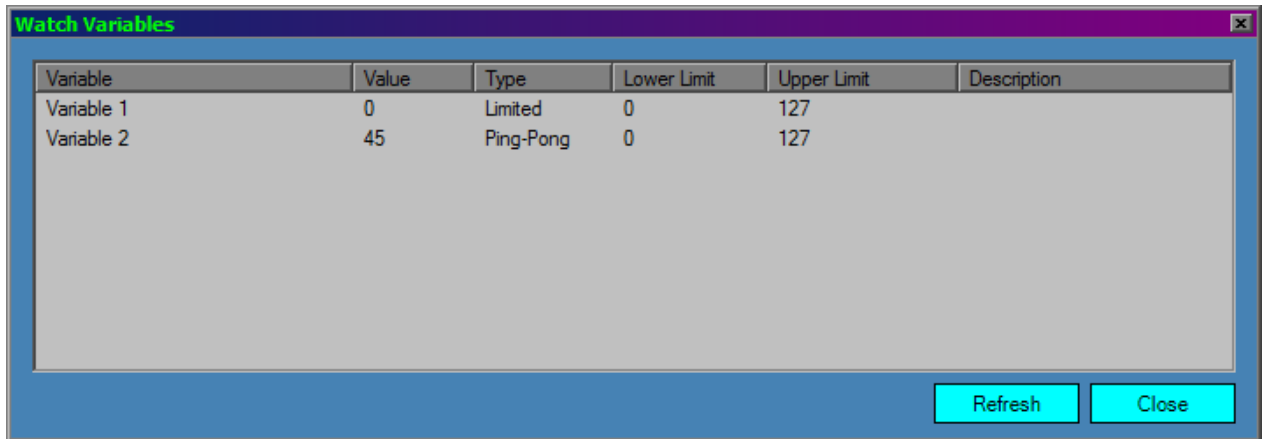


The Manage Variables window is where you can basically add, remove or update variables. If you delete a variable here, it will be removed from all other places where used. If you update it here, all references in other places (e.g. messages) will be updated as well.

Here you can set lower and upper bounds for variables and their current (initial) values. Behaviour Type defines what should be done with variable's value when it's increased or decreased out of its bounds (using Variable event). There are three options:

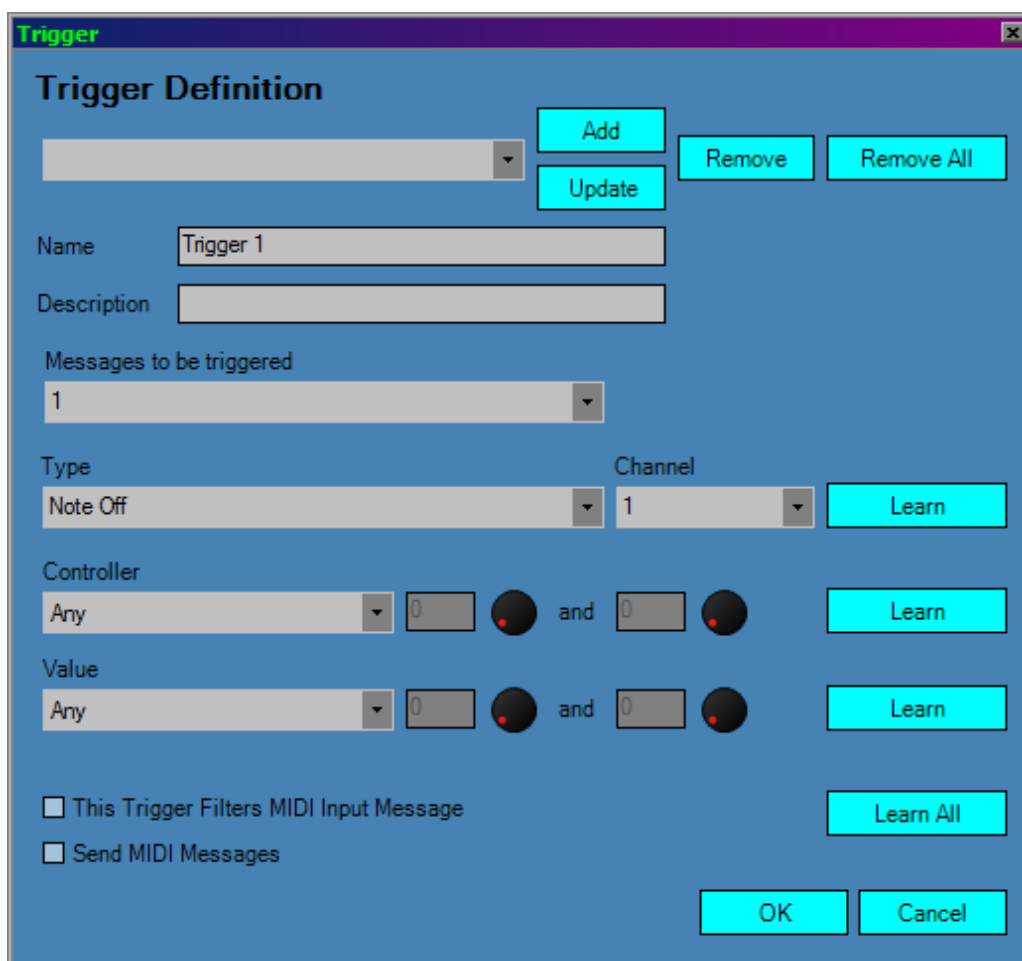
- 1) **Limited** – if value exceeds any of the bounds, its value will be set to the lower or upper limit respectively,
- 2) **Cyclic** – if value exceeds the lower bound, the decrease process will be continued from the upper limit; on the other hand, if value exceeds the upper bound, the increase process will be continued from the lower limit,
- 3) **Ping-Pong** – even when decreased or increased by a Variable event, its value will go up, down, up, down (or vice versa) and so on.

Another interesting window is Watch Variables. It's like a monitor showing all variables and their current values and properties in one place. The window should be refreshed automatically (if not, use Refresh button).



6.3. Triggers

Triggers are special objects waiting for some messages and being able to send other messages or fire some events. There are two types of triggers in MIDI PROCESSOR: message triggers, group triggers and scene triggers. The first type mentioned are triggers used to send chosen messages or events from the List of Messages and they are described below. The second type are triggers used to send whole groups of messages/events (described in chapter 6.4). The last type are triggers used to perform some actions on w whole scene (described in chapter 7.1).



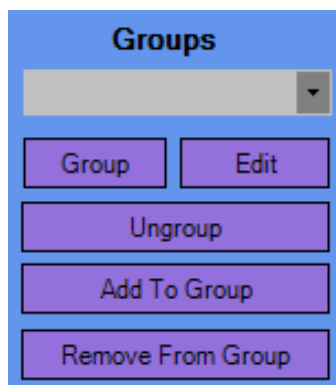
To show this window you have to press Set/Edit Triggers button on the main window of MIDI PROCESSOR. This window will appear only if there is at least one message/event on the List of Messages. Messages defined in the trigger as Messages to be triggered are sent if a message incoming to the trigger subunit of the MIDI PROCESSOR engine satisfies the condition defined in this window. As you can see above, as a trigger condition you can define message's type, channel number and range of the first (Controller) and the second (Value) data byte of MIDI message. Additionally you can check the This Trigger Filters MIDI Input Message if you don't want the message that fired the trigger to be sent to the further part of the MIDI PROCESSOR engine.

Note: An order of trigger subunits is: scene triggers, group triggers, message triggers. Within a single subgroup triggers are ordered as on the dropdown lists inside Trigger or Group Settings window.

6.4. Groups

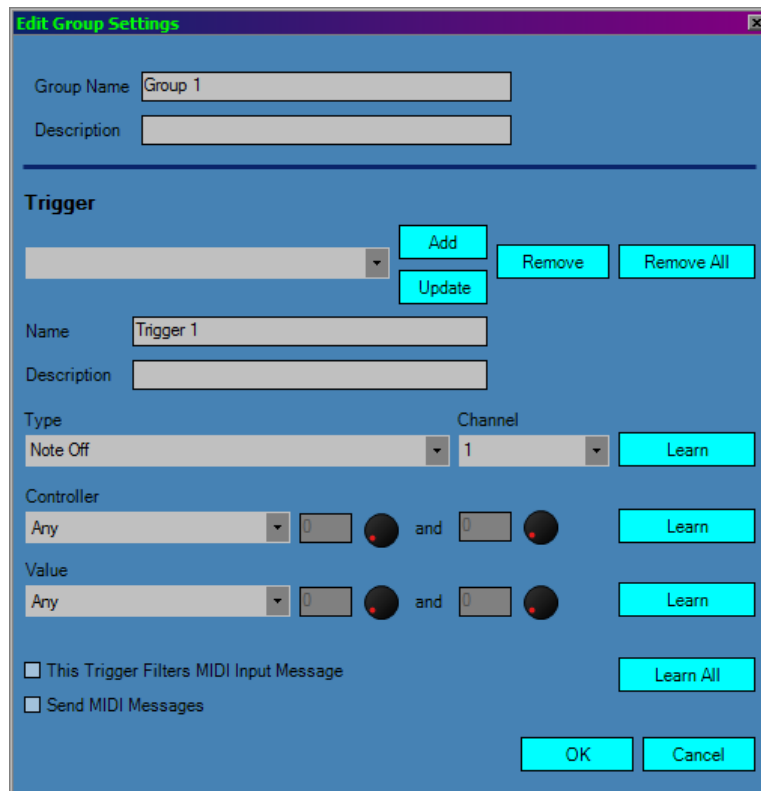
Messages or events that appear inside the List of Messages can be grouped. It may be useful for several reasons:

- to make it easier to perform some actions on many message/events at once quickly (e.g. copy or remove),
- to be able to send many messages/events at once,
- to set the same trigger for many messages/events.



Messages can be grouped by selecting one or more messages from the List of Messages on the main window. You can select multiple messages with Control (for nonadjacent messages) or Shift (for adjacent messages) key. Then you should use the Group button from the Groups panel on the main window (as shown below).

Then you will see the Edit Group Settings window.



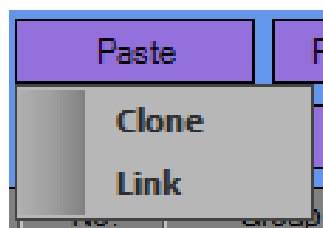
This windows allows you to set a name for a group, add some comment (Description) and – optionally – add one or more triggers for the group (in the same way as described in the previous chapter).

Using other buttons from the Groups panel on the main window you can:

- edit selected group's settings, using Edit button,
- ungroup messages from the selected group (delete a group from the list), using Ungroup button,
- add selected messages to the selected group, using Add To Group button,
- remove selected messages from the selected group, using Remove From Group button.

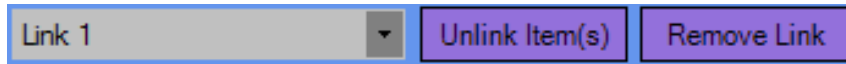
6.5. Links

In MIDI PROCESSOR there are two ways to copy messages/events – using Clone or Link (applicable after pressing Paste button after copying some messages/events using Copy button).



If Clone option is chosen, then pasted message are independent (changing any property of the original message doesn't affect its copy). However in some situations it might be convenient to use Link paste method instead. Then any change performed on the original message causes the same change to be executed on all linked messages. There can be more than one message linked to the original message.

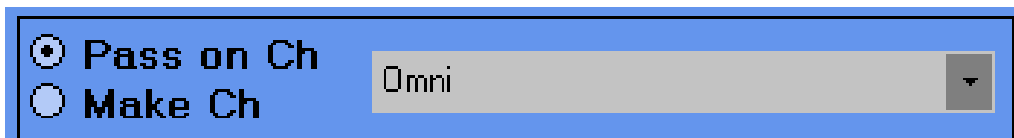
On the main MIDI PROCESSOR window there is also a special panel for managing links.



When a link is selected from the dropdown list, all messages linked using this link are selected. If you want to remove links from the **selected link**, use Remove Link button. Then all messages linked using this link will be converted into cloned messages. If you want to remove links from the **selected messages**, use Unlink Item(s) button. Then, again, these messages will be transformed into cloned messages.

Note: Links are always named and numbered Link 1, Link 2, ..., Link n, where n is number of links on the list.

6.6. Pass on Ch/Make Ch

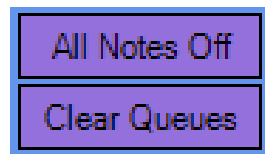


This subunit can work in two modes: as a channel filter or channel maker.

In the Pass on Ch mode, if some sent message is put before it in the messages flow, it is passed to MIDI outputs only if its channel number is one from the selected numbers on the dropdown list. If Omni is selected, then messages on all channels are passed to MIDI outputs. If <None> is selected, everything is filtered out and no messages will be present on any output.

Sometimes it might be useful to use the second mode – Make Ch. In this mode, all the messages' (sent before this subunit) channel numbers will be made to be equal to the numbers selected on the dropdown list. For example, if you send a single MIDI CC message and there are channels 1, 2 and 3 selected on the list, then there will be three MIDI CC messages generated (for channel numbers 1, 2 and 3). If Bypass is selected then all messages are canceled (filtered). All Notes Off works in the same way as in the previous mode.

6.7. Panic buttons



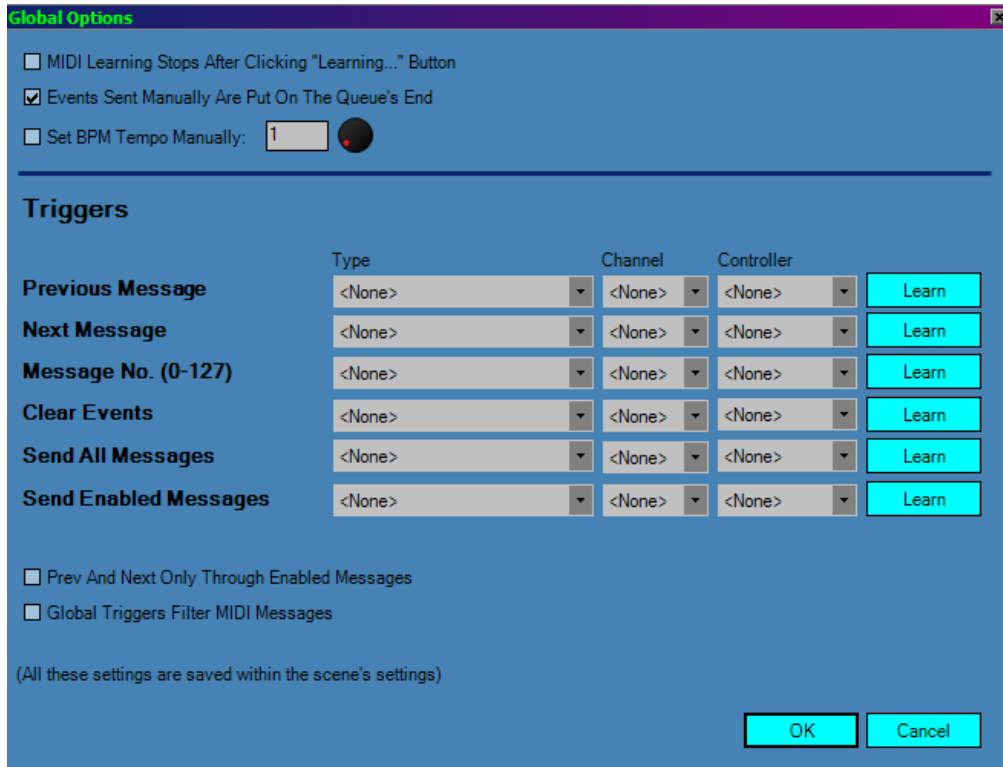
This buttons might be very useful in situations like loops or unneeded messages.

All Notes Off should be used to send Note Off MIDI messages for all notes (0-127) plus All Sound Off and All Notes Off. These messages are sent for all active (in the meaning of the previous chapter) MIDI channels.

Clear Queues button removes all messages waiting or just being somewhere inside the MIDI PROCESSOR's engine.

7. General options

7.1. Global options

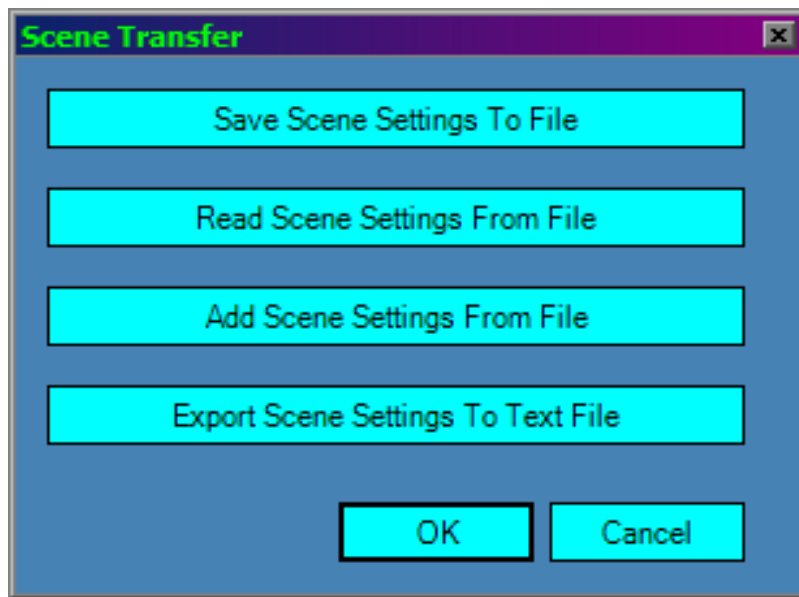


This window contains many useful settings for the whole MIDI PROCESSOR plugin. As you can read on the bottom of the window, all these settings are saved with scene's settings (you will learn more on saving scenes in chapter 7.2). Settings visible above are the default ones.

- 1) **MIDI Learning Stop After Clicking "Learning..." Button** – corresponds to using Learn and Learn All buttons in various windows of MIDI PROCESSOR. When disabled, MIDI learning process is stopped immediately after receiving first message on the input. To continue the process, user has to click Learn button again. If you enable this option, learning process will last continuously until user clicks Learn or Learn All button again – then it's stopped.
- 2) **Events Sent Manually Are Put On The Queue's End** – this setting controls where in the MIDI PROCESSOR engine's queue messages sent manually should be put. Manually in this context means sent by clicking Send, Send Enabled or Send All button on the main MIDI PROCESSOR window.
- 3) **Set BPM Tempo Manually** – by default BPM tempo of the plugin and therefore the sync MIDI messages are synchronized to the tempo of VST host (option disabled). If you check this option, you can set the BPM tempo manually here (legal values are 1-999).
- 4) **Triggers** – there 6 so called global triggers which control some general operations on messages and events of MIDI PROCESSOR: loading previous message, loading next message, loading message of a number taken from incoming MIDI message defined as the trigger, removing events from the queue, sending all messages and sending enabled messages. They behave analogically to the triggers described in the previous chapters.

- 5) **Prev And Next Only Through Enabled Messages** – it defines the behaviour of the first two global triggers. If unchecked, these triggers iterate through all messages or events on the List of Messages (even disabled). If checked, they go through enabled messages or events only.
- 6) **Global Triggers Filter MIDI Messages** – it controls whether a message that triggered a global trigger should be passed to the next trigger subunits and MIDI engine or not. Global Triggers are caught just before Pre F/T subunit.

7.2. Scene transfer



This window contains file operations on the MIDI PROCESSOR settings and messages, both scene's and global. You can save scene's and global settings like a preset to a binary file (or read these settings from this kind of file). You can also generate a report of all current settings and messages/events in a human-readable form (.txt file).

- 1) **Save Scene Settings To File** – it saves all scene's and global settings and message/events, variables, triggers, groups, links and selected MIDI input and output ports to a binary file (.sst).
- 2) **Read Scene Settings From File** – it reads the data mentioned in the previous clause from a binary file (.sst) and replaces ALL current settings, messages/events etc. with these data.
- 3) **Add Scene Settings From File** – it works like the previous clause, but it merges data rather than replacing it (global settings read from a file are omitted, messages/events, triggers, links, variables and groups are added).
- 4) **Export Scene Settings To Text File** – it generates and saves a text file containing a report of all current scene's and global settings and all other things (like triggers, messages/events etc.).

7.3. Scene's settings and buttons

In this chapter we will describe remaining panels from the main MIDI PROCESSOR's window (listed in chapter 4.1).

MIDI sources and destinations

These two dropdown lists are used to choose on which sources to listen to incoming MIDI messages (physical and virtual MIDI inputs plus <From Host>) and where to send outgoing MIDI messages (physical and virtual MIDI outputs plus <To Host>).

Operations on messages

- 1) Add – adds new message/event of chosen type,
- 2) Edit – edits a message/event selected on the List of Messages (the first one if more than one item is selected),
- 3) Copy – copies selected message(s) into the clipboard,
- 4) Paste – pastes the messages/events from the clipboard into the List of Messages; the messages are pasted in the same order as when copied; if nothing is selected on the List of Messages, then they are pasted below the last message/event present on the list; otherwise they are put above the selected item; message can be pasted as clones (independent copies) or links (described in chapter 6.5),
- 5) Remove – deletes selected items from the List of Messages,
- 6) Remove All – deletes all items from the List of Messages,
- 7) Send – sends selected messages/events,
- 8) Send Enabled – sends all enabled messages/events,
- 9) Send All – sends all messages/events from the List of messages,
- 10) Move Up – moves all selected items up in the order of messages on the List of Messages,
- 11) Move Down - moves all selected items down in the order of messages on the List of Messages,
- 12) Select All – selects all messages/events on the List of Messages.

To unselect item on the List of Messages, click any empty area on the List of Messages (for example below the last item on the list).

Miscellaneous scene's (preset's) settings

- 1) Audio Thru – if checked, audio signal from the input to the plugins is just passed to the outputs; it supports stereo only,
- 2) MIDI Sync – if checked, MIDI PROCESSOR sends MIDI BPM tempo synchronization messages to the active (selected on the Input Ports list) MIDI output ports,
- 3) Scene Cycle – if selected, enabled messages and events from the List of Messages will be sent one by one and after last item is sent the process will begin from the first item and so on.
- 4) Internal Messages Work on MIDI Learn – if checked, internal messages (not from MIDI inputs ports, so generated somewhere inside the MIDI PROCESSOR's engine) will work when on MIDI learn (when Learn or Learn All is checked for example in some message's or event's edit window).

- 5) Send Messages When Loading This Scene – all enabled messages/events from the List of Messages will be sent automatically when loading this scene (preset).
- 6) Clear Events Queue When Loading This Scene – the internal MIDI PROCESSOR engine’s queues will be cleared automatically when loading this scene (preset).

The default settings are shown on the screenshot in chapter 4.1.

Information displays

- 1) BPM – shows current BPM tempo, received from the VST host or set manually in the Global Options window,
- 2) CPU – shows current CPU load taken by this instance of MIDI PROCESSOR.

Audio signals indicator

It monitors incoming and outgoing (if Audio Thru option is checked) stereo audio signal.

MIDI signals indicator

It monitors incoming to and outgoing from MIDI PROCESSOR MIDI messages.

About/Registration link

Clicking the Soundigy logo opens about window. If MIDI PROCESSOR isn’t registered, it allows you to register the software.

8. Standalone version

There are only slight differences between the VST and standalone version of the software:

- there is a Presets section in the standalone,
- only manual BPM tempo in the standalone,
- currently no audio support in the standalone,
- currently processing timer in the standalone version works in constant 2 ms intervals.
- no <From Host> nor <To Host> on the MIDI inputs/outputs lists.

Because of some of these differences, the main window looks a little bit different.



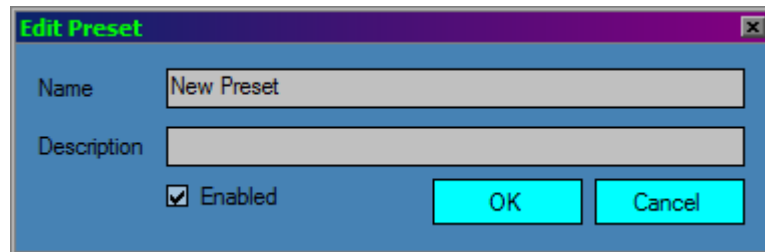
As you can see in place of the Audio In/Out indicators there is a Presets section.

8.1. Presets section

In VST Hosts presets (also known as programs or scenes) support is usually part of the host functionality (but doesn't have to). To preserve this very important feature (especially for MIDI PROCESSOR's idea), presets have been added to the standalone version.

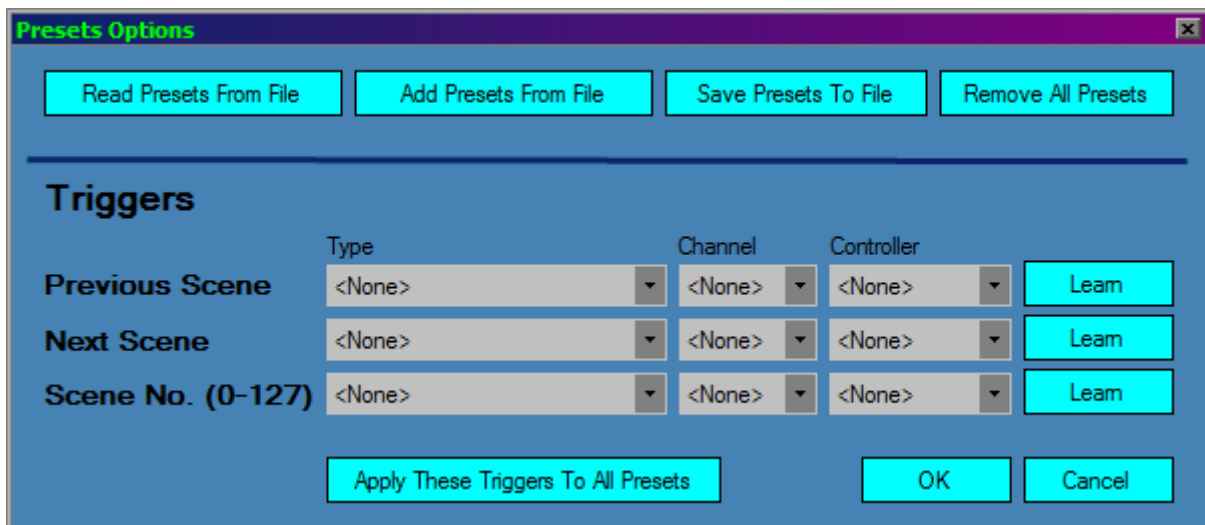
Operations on presets (scenes)

- 1) Add – adds and saves a new preset under given name; you can also add a description for the preset and make it enabled or disabled; Enabled checkbox might be important when activating scenes using triggers (will be discussed later in this chapter),



- 2) Edit – opens the same window as above and allows you to edit presets name, description and make it enabled or disabled; it has to be mentioned that clicking OK in Edit Preset window overwrites all the data saved with the preset with current settings,
- 3) Remove – deletes a preset chosen in the dropdown list,
- 4) Up – moves selected preset up in the order of presets on the dropdown list,
- 5) Down – moves selected preset down in the order of presets on the dropdown list,
- 6) Act – activates (loads) a preset selected in the dropdown list,
- 7) Opt – opens Presets Options window.

Presets Options



This window allows you to run some operations on presets and setup triggers for them.

- 1) **Read Presets From File** – reads and loads presets into MIDI PROCESSOR from a binary file (.ssa) and replaces ALL current presets. It doesn't activate any preset.
- 2) **Add Presets From File** – works like the previous clause, but it merges presets rather than replacing them.
- 3) **Save Presets To File** – saves all presets to a binary file (.ssa).
- 4) **Remove All Presets** – deletes all presets.
- 5) **Triggers** – allows you to define three triggers for presets (in a very similar way as for global triggers described in chapter 7.1.). These are triggers for: loading previous preset, loading next

preset and loading preset of a number taken from the last byte of incoming MIDI message defined as the trigger (might load only presets 1-128). Loading previous/next preset works on enabled presets only.

- 6) **Apply These Triggers To All Presets** – sets above triggers settings for all presets.

Global triggers

The Apply These Triggers To All Presets command is also available for global triggers in the Global Options window (in standalone version only).

9. Addendum

9.1. Working with scenes and presets from older versions

After updating Midi Processor to higher version it may be necessary to update scenes and presets files into recent format. Follow these steps:

1. Load old file version scenes or presets into new version of Midi Processor,
2. Set features added in new version,
3. Save scene/preset into file.

9.2. Sharing scenes between VST and standalone

Since Midi Processor v1.1, loading and saving scenes between VST and standalone is fully supported.