unimidi

UMX

Virtual modular MIDI controller and MIDI profiler

Instructions manual ver. 1_4_2

Welcome to Unimidi UMX User Guide

Acknowledgment

Thank you for choosing the **Unimidi UMX** as your MIDI solution. Your feedback is important to us. Should you have any queries or require assistance, don't hesitate to reach out to our customer support at support@unimidi.com. For additional resources, please visit our official website at Unimidi.

Introduction

This user guide is intended to assist you in operating the **Unimidi UMX midi_unifier**. This software package is designed to be compatible with any Digital Audio Workstation (DAW), software, plug-in, instrument, or device that adheres to the MIDI standard.

Disclaimer

The information herein is accurate at the time of publication. **Unimidi** reserves the right to modify any aspect of the product or this manual without prior notice.

Dedicated to my Father

Conventions and Security Guidelines

Understanding MIDI

MIDI (Music Instruments Digital Interface) is the universal standard for communication between electronic musical instruments and music software. A rudimentary understanding of the protocol nomenclature and functionalities is essential for utilizing this software effectively. Additional information related to MIDI will be interspersed throughout this manual. For an in-depth understanding, we recommend visiting the official MIDI website: https://midi.org.

Caution Symbol Guidelines

The presence of this symbol is intended to draw your attention to scenarios that may arise while using **UMX**. These scenarios could result in malfunctions or loss of work. Exercise caution and read the accompanying text carefully when you encounter this symbol in the manual.

Tips Symbol Explanation

This symbol denotes helpful suggestions or recommended best practices for working with **UMX**. Whenever you see this symbol, consider the accompanying advice to optimize your experience and workflow.

Cybersecurity Notice

UMX is designed with your privacy and security in mind. It neither opens ports on your local network and it does not collect any user information. Should you encounter any issues or have concerns about security, kindly contact **Unimidi** support for assistance.

Summary

The UMX Concept: Introducing the midi_unifier

What UMX is not

UMX is not an instrument in the conventional sense; you will not find any built-in sounds. Rather, consider **UMX** as a supplementary component that integrates with your existing hardware/software musical setup whether you are a keyboardist, guitarist, saxophonist, or vocalist.

What UMX is

UMX is a **virtual modular MIDI controller** and **MIDI profiler** — a flexible environment for routing, profiling, and high-volume MIDI data transmission. Not only can UMX control your setup, it can also reshape its behavior in real time as you play.

Complexity of Live Performances

Live performances can become intricate when utilizing a variety of sounds, effects, and configurations. This complexity amplifies when incorporating both hardware and software simultaneously. During a song, transitions from one sound or effects setting to another often require manual actions such as pressing a pedal, turning a knob, or clicking a button.

Simplifying Live Performances with **UMX**

With **UMX**, these transitions can be dramatically simplified. The device is designed to handle all controls that support the MIDI protocol. You can pre-set your sounds and complex auditory scenarios and navigate through them using two simple buttons: '*Next*' and '*Prev*'.

Configurable Interface

UMX offers a highly customizable interface that unifies commands across multiple machines, effects, software, and accessories. It remembers your configurations, allowing you to recall them instantly.

Uncompromised Performance

UMX lets you focus entirely on your performance, giving you the flexibility to introduce subtle sonic variations. With a single press of a button, you can instantly switch to the next preset scenario, delivering studio-quality sound precision to live performances.

UMX Software Features

Software Architecture

The **UMX** software architecture operates on two essential levels:

1. **Performances**: These are collections of configuration parameters for the set you want to use. A performance can correspond to a song or an entire concert, depending on your needs. **UMX** allows for a maximum of 128 performances per project.

2. **Patches**: Stored within each performance, and limited to 32 per performance, patches allow you to save and recall sets of adjustments (e.g., instrument volumes, effect depths) instantaneously. Each patch can represent a sonic section (intro, verse, chorus, etc.) of your song.

Together, performances and patches enable the storage of up to 4096 sonic scenarios in a single project. The software can generate an unlimited number of projects, and rapidly switch between the last 10 used files.

Patch Messaging

Loading a patch will transmit a bundle of MIDI messages based on the slots that populate each performance. In the version **1_4_2 *pro*** the maximum values using the standard_slot type include:

- 48 fully configurable *control-changes* (CC)
- 16 *program-changes* (PC), each preceded by a Bank Select messages (MSB and LSB)
- Transport MMC CC: *play, stop, and continue*
- Special controls quitting a patch

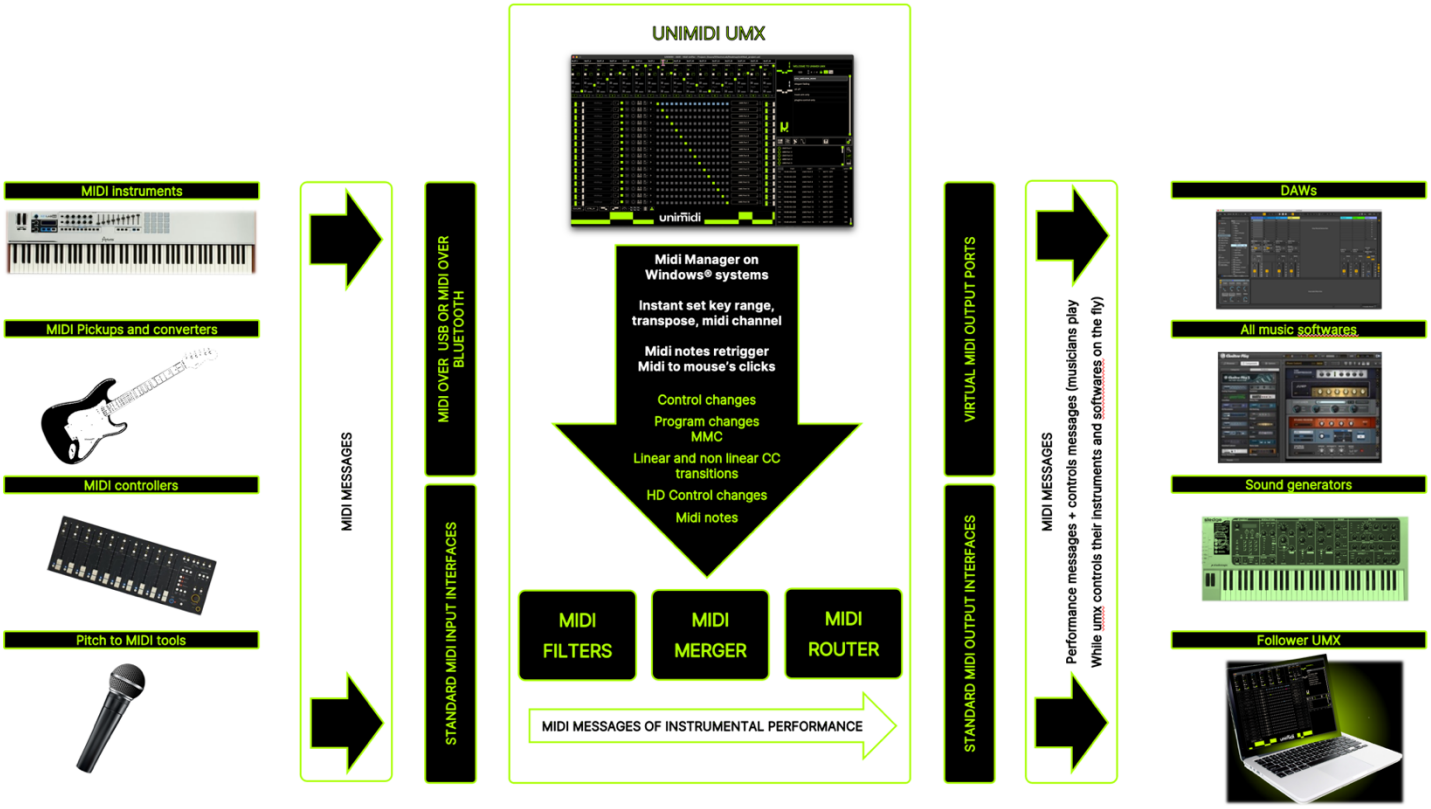
Using other types of slots allows for the composition of different bundles. For example, by using 16 program_changer slots, **UMX** can transmit 64 *program-changes* at once per patch.

MIDI Profiling

UMX is not only a modular MIDI controller, but also a MIDI profiler. It allows real-time transformation, filtering, and redirection of MIDI messages through a powerful pass_through system.

Each MIDI input/output pair in UMX is internally managed as a 'trunk'—a flexible signal path that lets you apply filters, remap data, and reshape controller curves on the fly. This profiling capability makes it possible to adapt MIDI behavior to your instrument or performance context.

Patches are transmitted through a maximum of 16 physical or virtual MIDI ports (the latter supported by specific operating systems or drivers, as detailed in OS-specific sections).



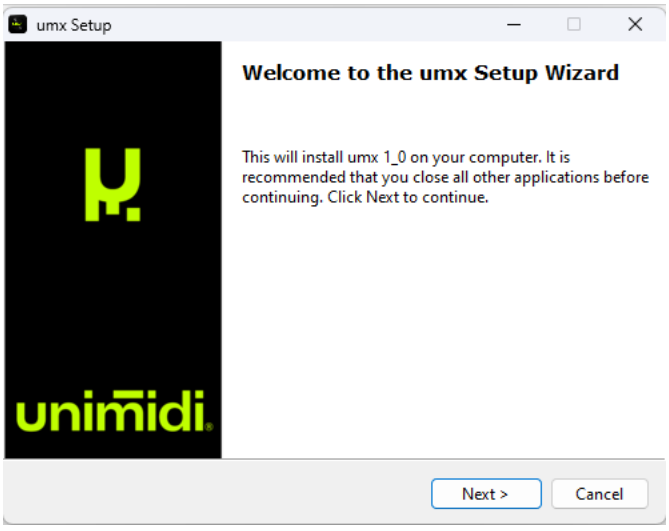
Software Installation

macOS®:

- Double click on **unimidi_umx_installer.dmg**
- Drag and drop **UMX** icon into the application folder

Windows®:

- Double-click on the **unimidi_umx_1_4_2_installer** package and follow the instructions. During the installation, you have the option to customize the installation folder:



If you have purchased an **Unimidi** device, you can locate the **UMX** folder and other useful files on the provided USB stick. Alternatively, you can download the software directly from the **Unimidi** website download section.

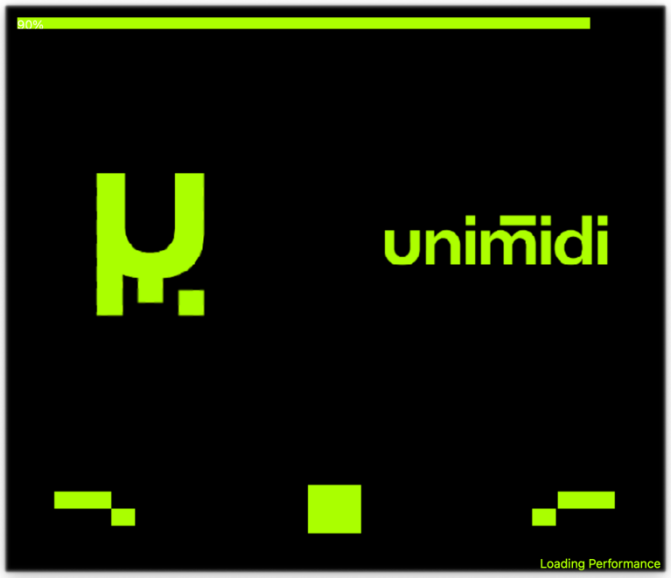
First-time Software Launch

Launch the **UMX** software program.

Please carefully read the end-user license agreement (EULA), it is mandatory to accept these terms to use the software, for further clarifications do not hesitate to contact the *Unimidi* team.

In the trial version, **UMX** has the **pro** configuration and includes a workspace with 16 control slots and 16 MIDI trunks. This demo version is fully functional and allows up to 64 application restarts. Once all authorized restarts have been used, **UMX** will switch to the free version, which features a 3x3 matrix. Both the trial and free versions can~~n~~ot load projects created with another free or trial version.

You have the option to activate your **Unimidi** license at any time (see licensing).



Project Setup

On the first launch, you will be prompted to create a new project:

- Choose your desired save directory and file name for the project.

Depending on your security settings, you may also be prompted to authorize the program access to the selected directory. Please grant this access to proceed.

Warning: If you do not grant access to one or more directories as prompted, the **UMX** software will not launch. Should an error occur, you'll need to authorize the program from your system preferences before attempting to relaunch it.

Subsequent Launches

Upon subsequent launches, the software will automatically load the last-used project file. If no such file exists, you'll need to create a new project. The software allows quick toggling between the last 10 projects.

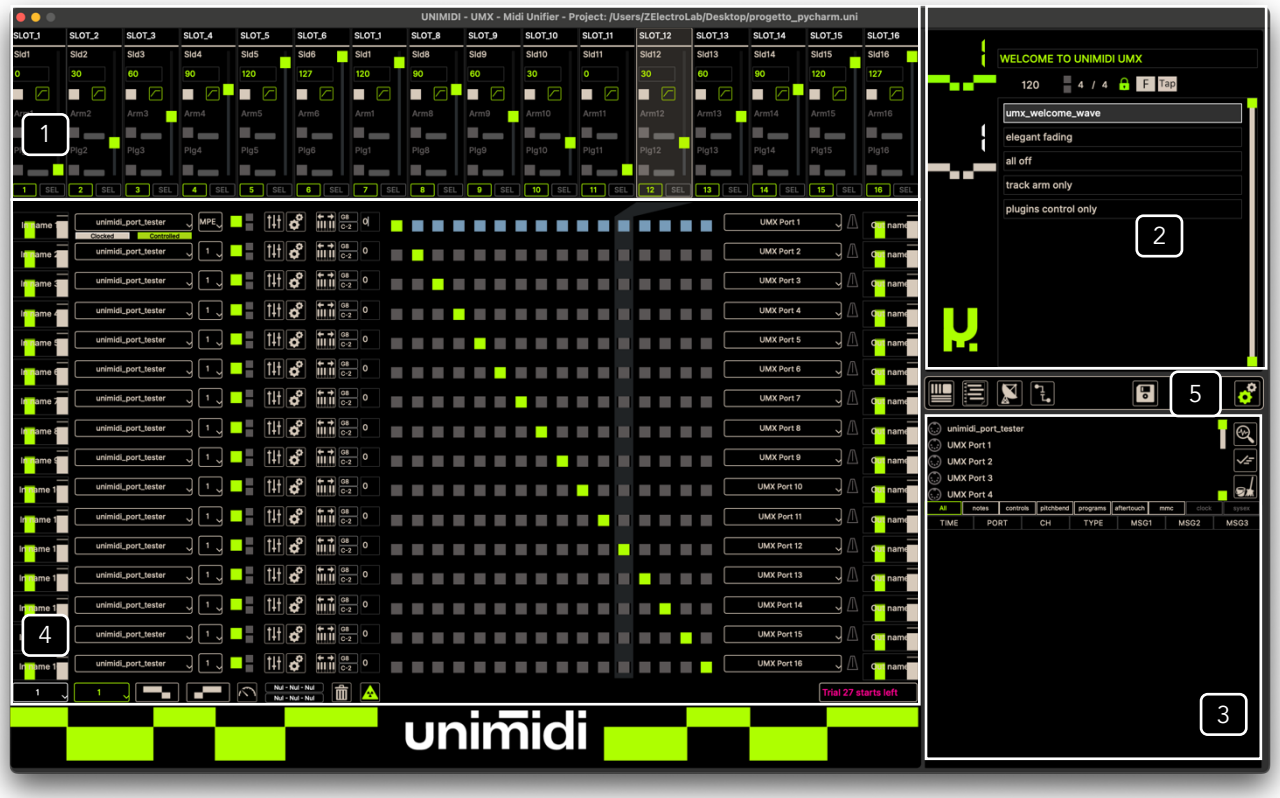
Extended Interface

Initial Launch

Upon launching, the program loads a default file containing a series of sample performances and patches. The extended interface is displayed by default. You can switch to the reduced interface at any time using the button located in the control_area (see below):

Help Indicators

All **UMX** controls come with help indicators. Simply hover over a control or field to display the helper tooltip.

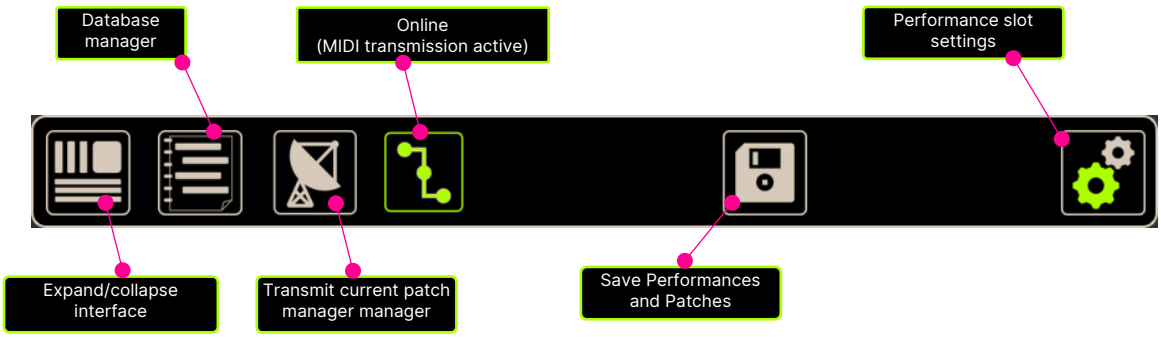


Interface Sections

1. **Performance Area:** This area contains slots that send MIDI messages based on their configurations. The software has nine types of slots that cover a wide range of possibilities. Additional slots will be added.
2. **Navigation and Slot Configuration Area**: Allows you to view the title of the current performance and navigate to adjacent ones. It also displays a list of patches contained in each performance.
3. **MIDI Monitor Area**: Allows you to see all the MIDI ports detected by the system and view transmitted messages.
4. **MIDI Merger Area**: Enables routing of incoming MIDI messages from instruments to both hardware and software sound generators.
5. **Controls Area**: Contains buttons for major actions and updates based on the current work context.

Controls Area in Both Interfaces

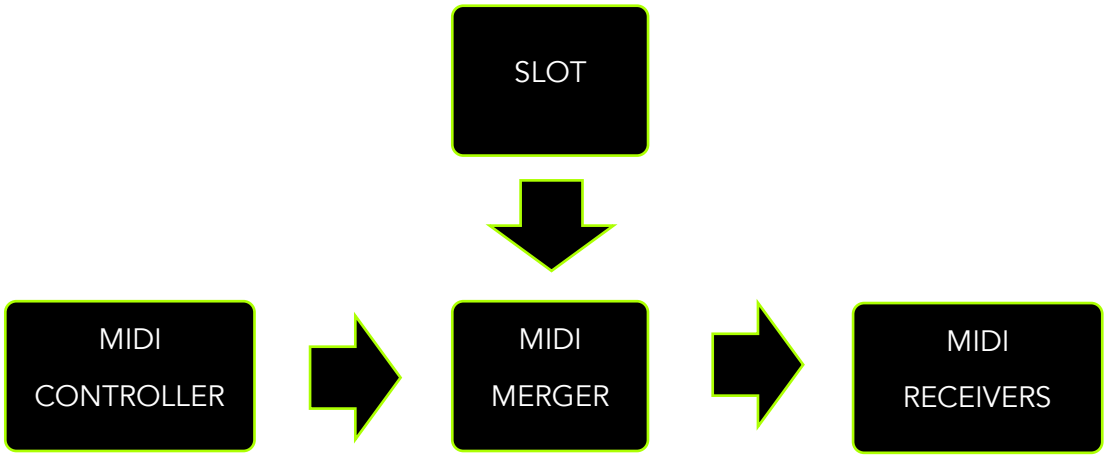
The controls_area is consistent whether you are using the extended or compressed interface.



When necessary, the software displays a notification indicating that there are unsaved changes. The save button will blink, and a 'SAVE' label will appear on the right side of the interface. It is important to note that this notification will be cleared if you switch to another performance or patch. The decision to implement a non-blocking policy was made with live performances in mind, ensuring an uninterrupted workflow.

Performance Area

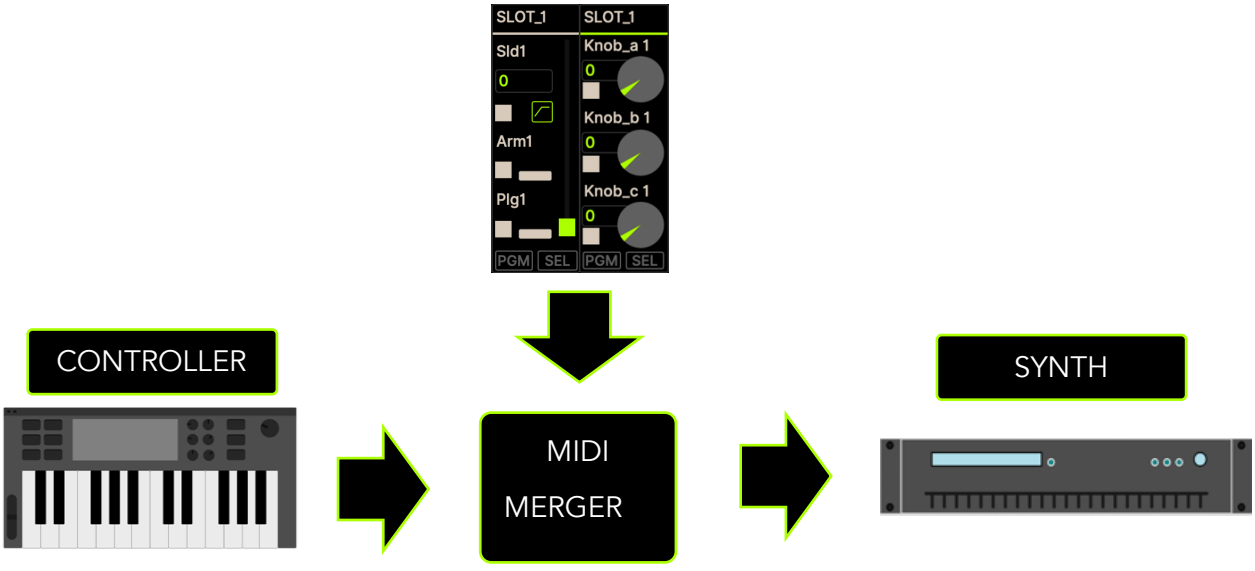
Working Principle:



In **UMX**, you can configure up to 16 MIDI message generators—such as keyboards, master keyboards, MIDI pickup-equipped guitars, pitch-to-MIDI software, wind controllers, and more. These performance messages first pass through the **midi_merger**, where they can be filtered or transposed before reaching virtual or physical MIDI outputs.

Along the way, additional MIDI messages—including *control-changes*, *program-changes*, and standard MMC transport controls—are injected by the slots that make up an UMX performance. These messages are stored within each patch.

Thanks to its integrated **pass_through** module, UMX can also perform real-time MIDI profiling: incoming data can be dynamically filtered, remapped, or reshaped using custom curves:



Simple and Versatile Configuration

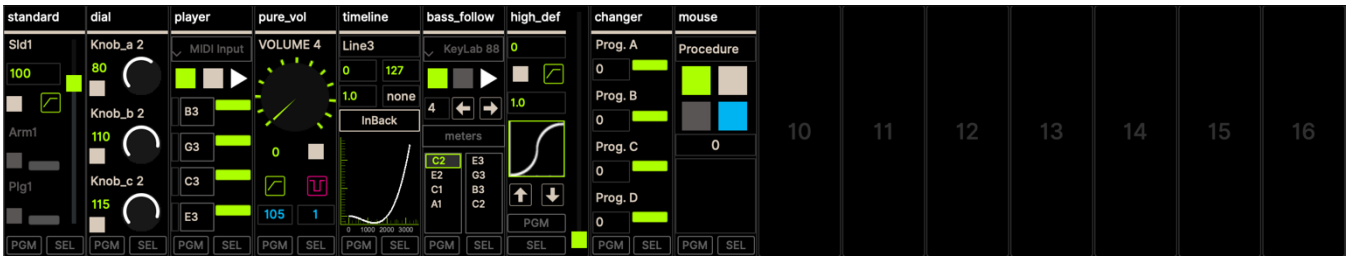
In this setup, the controller sends MIDI messages through the midi_merger to the connected synthesizer. This process is both intuitive and instantaneous, as the software instantly recognizes any available MIDI port on the system. The two slots in the performance can be configured to control various parameters of the synthesizer, such as volume, filter cut-off frequency, and effect depth.

Smooth Transitions

One of **UMX** key features is its ability to create smooth transitions when switching between saved patches during a performance. This prevents abrupt changes or interruptions, ensuring a natural musical experience.

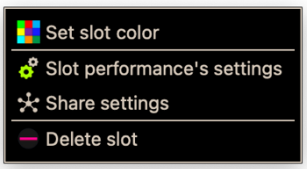
Extensive Control and Modularity

The software allows up to 16 slots in a performance, offering a high level of versatility. You can freely configure the MIDI channel, the number of controllers, and the values transmitted by the slots. The 16 midi_trunk that make up the midi_merger further extend this versatility. The software comes with nine types of slots covering a wide range of transmission functions. As **UMX** is modular, new slots will be added in future versions to further enrich the software communication capabilities and musical expression. The performance_area is designed to accommodate the slots:



The slots are numbered from 1 to 16, and each slot has two levels of configuration. The first level pertains to the performance; the parameters assigned here will be common to all the patches. The second level is specific to each of the patches created for the performance.

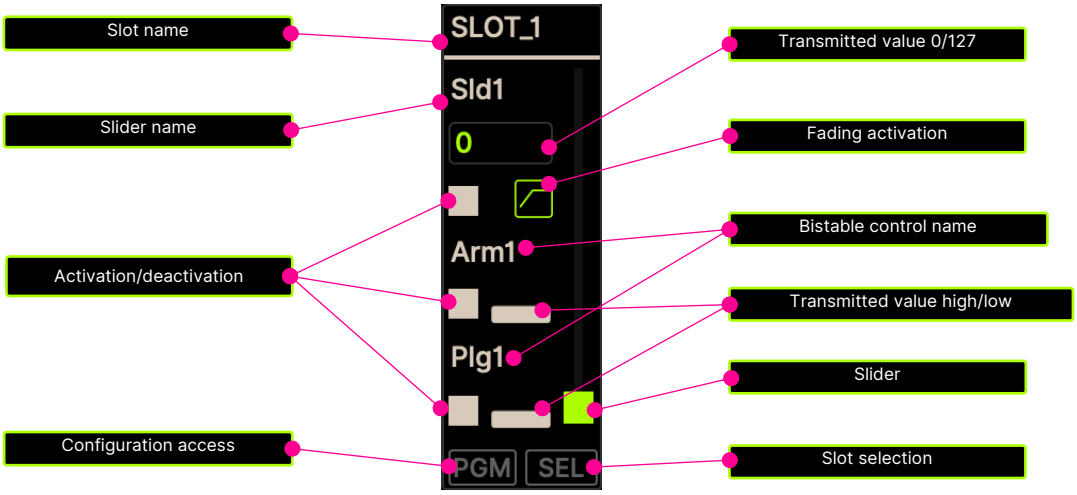
A context menu can be recalled by right click for each slot:



Set slot color – Colors help to recognize slots and their association with an instrument or a plugin. Click to open the system color manager.
Slots performance settings – Jump to the performance layer slot settings.
Share settings – Open the slot sharing settings window (replicate slot settings to multiple patches; see below).
Delete slot – The slot will be deleted.

The Standard Slot

The standard_slot allows for the transmission of up to 3 *control-changes* values. The first, represented by a slider control, allows for the transmission of values from 0 to 127, covering the entire possible range. The second and third, which are bistable, allow for the transmission of two values high and low.



The control named *Slider* is suitable for transmitting continuous values, which can be associated with parameters such as volume, stereo panning, effect depth, the volume of a secondary audio sends, the cut-off and resonance values of a synth filter, etc.

The bistable controls, named *Arm* and *Plg*, are suitable for transmitting on/off type values. These can be associated with parameters like track arming in a *DAW* (*Digital Audio Workstation*), activating/deactivating a *plugin*, an effect, launching clips, etc.

The MIDI controls for a slot (generally valid for all types of slots) are loaded and immediately transmitted when a patch is called up. For example, the standard slot can activate a track in your DAW, insert an effect, and adjust the volume in less than a millisecond, all while your performance continues.

The standard_slot can also transmit a *program-changes* message upon loading, as well as MMC *(Midi Machine Control)* transport controls like *play*, *continue*, *stop*. There are also two controls that are transmitted when the patch is abandoned.

Access to the configuration parameters for the performance slot is available via the button:

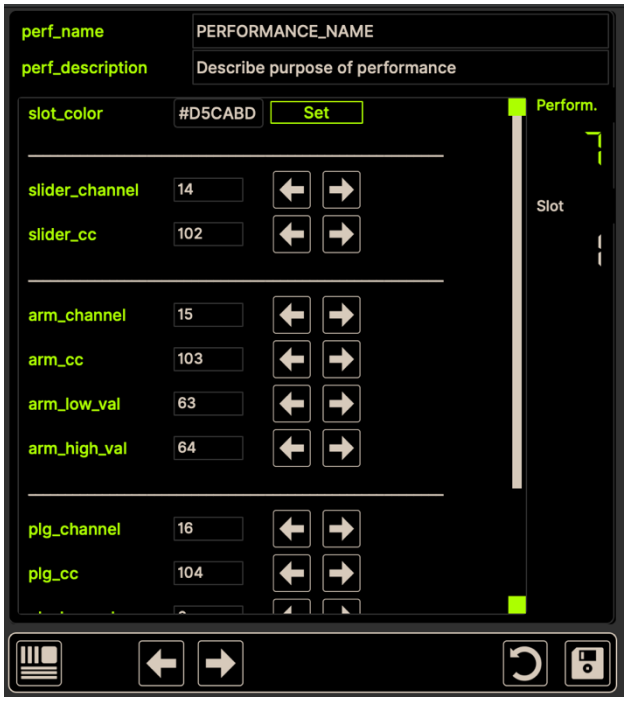
The controls_area section, located immediately below the navigation_area, updates and displays the relevant commands:

The performance parameters page primarily allows you to enter a name and a brief description. The performance name could roughly be the title of the track.

By pressing the slot selection button, the slot is highlighted in a different color, and the list of its parameters is displayed. It is also possible to move 'horizontally' through the nearest slots parameters using the arrow keys.

Please note that any modifications made to the parameters need to be saved before moving to another slot. If you fail to save, you will be prompted to save before switching to another slot or leaving the slot configuration page. The save button will blink whenever changes are made to the project that require saving. By default, the autosave option is activated (see system options).

The standard slot has the following possible performance configurations:



slot_color: Press the *SET* button to choose a color for the slot.

slider_channel: Allows setting the MIDI transmission channel for the slider value.

slider_cc: the *control change* number transmitted by the slider (0/127).

arm_channel: Setting the MIDI channel for the first bistable control.

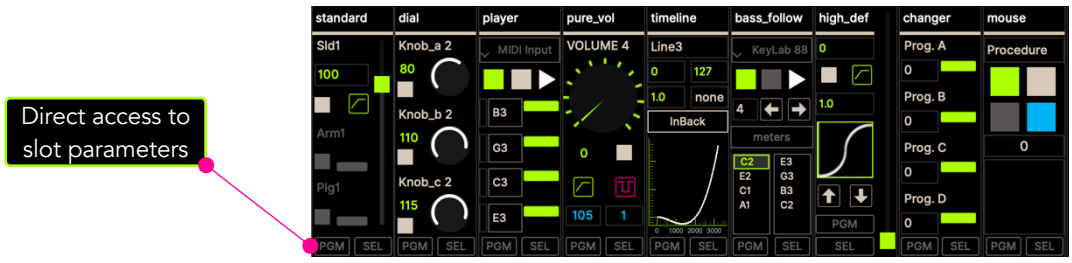
arm_low_value and **plg_high_value**: The respective high and low values transmitted (0/127).

plg_channel: Setting the MIDI channel for the second bistable control.

plg_low_value and **plg_high_value**: The respective high and low values transmitted (0/127).

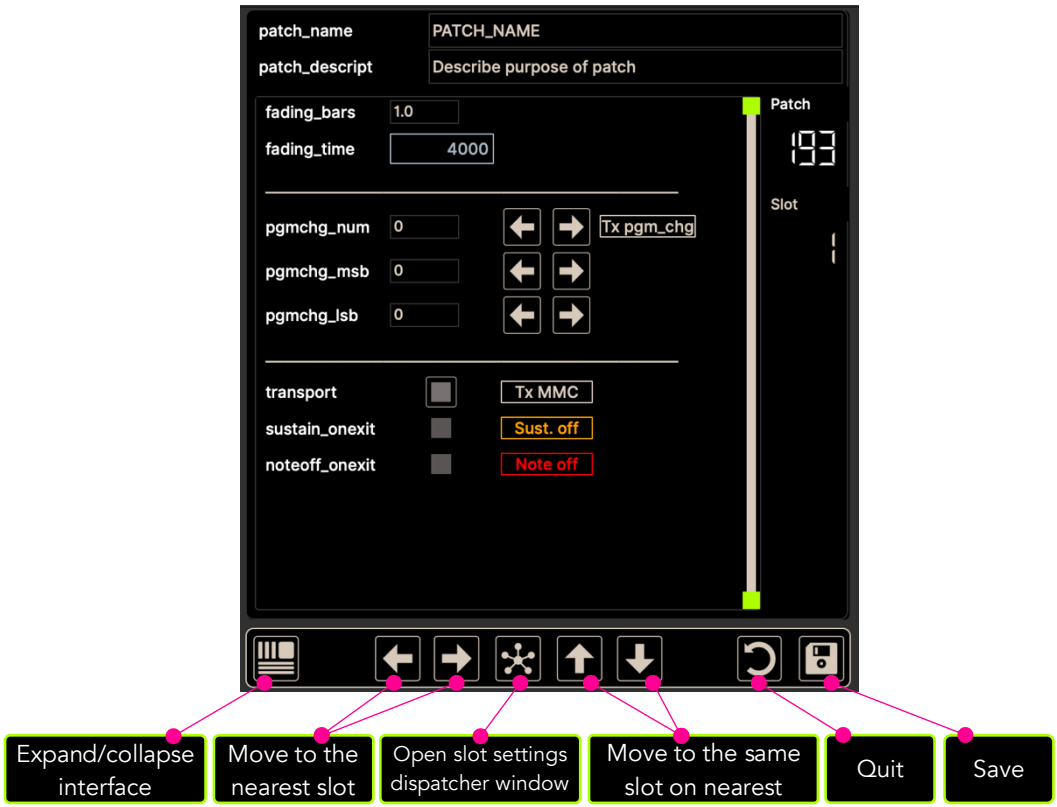
pgmchg_channel: The MIDI channel through which the slot transmits *program-changes.*

All slots also have a second configuration page, which can be accessed by pressing the 'PGM' button on the slot itself:



The parameters on this page can be modified for each individual patch contained in the current performance.

The available parameters for the standard slot for individual patches are:



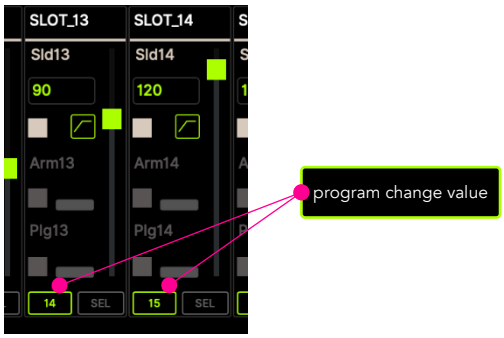
fading_bars: This parameter is related to the slider control and indicates how much time (expressed in music measures) will be required for the final control change value to be reached when the patch is invoked. For example, if we associate our slider with the volume of a track in our DAW and have a first patch where the set value is 0 (-infinite dB), when we call up the next patch, for which the slider set value is 100, the program will transmit a series of progressive values from 0 to 100. The transition will during the settled number of bars. Acoustically, the result will be a gradual increase in volume rather than an abrupt jump from 0 to 100.

Transitions are appropriate for all sound parameters, such as volume, effect amount, filters, etc. The adjustable transition duration avoids abrupt changes and also allows the creation of fade-in and fade-out or crossfading effects.

fading_time: This field is automatically updated by the program and shows the during time in ms of the settled transmission period in the **fading_bars** field.

pgmchg_num: Enter the *program change* number that will be transmitted by the slot when the patch is loaded. The values range from 1 to 128 (MIDI transmitted values are 0 to 127). A set value of 0 indicates no transmission. If a slot is configured to transmit a *program change*, this is also visible without entering the configuration page.

Clicking on the program change number at the bottom of the slot takes you directly to the configuration page:



pgmchg_msb and **pgmchg_lsb** are the transmitted values (Most Significant Byte and Least Significant Byte) that allow **UMX** to send a bank select message in accordance with MIDI standards. These values, which are two **control_changes** respectively number 32 and number 0, allow you to select the sound bank or memory group to be called up in the receiving instrument. **UMX** always transmits in the sequence *MSB, LSB, PC*.

On the right side of the three input fields (*PC, MSB, LSB*), there is a **Tx pgm_chg** button available. When pressed, a program change command based on the values entered in the fields will be sent to all MIDI outputs. However, it is important to activate the corresponding node for your slot in the **midi_merger** for MIDI *program-changes* to ensure they are properly transmitted.

Please note that the program change command will be transmitted even if the system is not online (see below for online/offline explanations). This allows users to test the entered values without needing to activate the global MIDI transmission

transport allows the slot to transmit a standard transport **MMC (*Midi Machine Control*)** command to start, stop, and resume the playback of a sequencer, arpeggiator, or DAW. These are the classic play, stop, and restart commands. Transport commands are sent immediately upon loading the patch.

The transport command is based on a sequential button that takes on a different appearance depending on the choice made: no transport, stop, play, continue:

To verify the settings for the transport controls, click on the *Tx MMC* button. Remember to activate the corresponding node for MIDI transport of your slot in the **midi_merger** to ensure the functionality is enabled correctly.

On_Exit Functions

During instrumental performance, in the event of sudden changes, it is possible that the musician calls up a new patch when there are still notes being played or maintained by any sustain pedal typically used by pianists and keyboard players. If the new patch changes parameters so that control is transferred to another instrument, inconsistencies may occur with notes that were played before the change and that do not stop playing after it. The standard_slot provides two MIDI transmission functions that are carried out by the program when exiting the patch:

sustain_on_exit, if we were playing an instrument and did not release the sustain pedal before switching to the next patch, activating this command **UMX** will transmit the release of the pedal CC for us before switching to the next or previous patch.

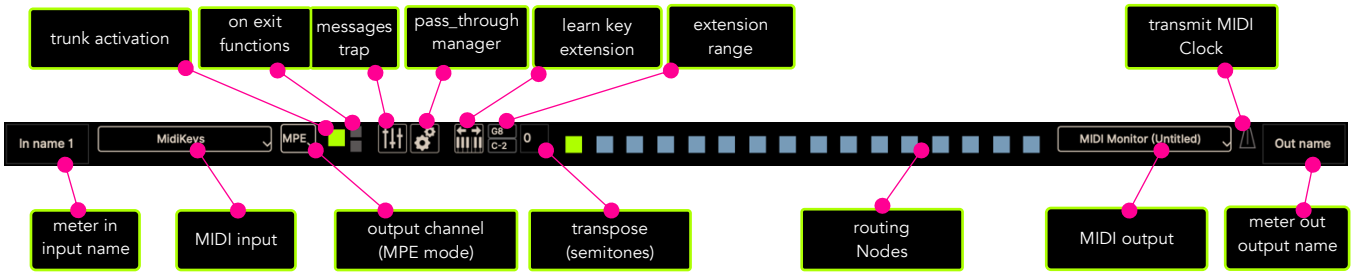
To verify the settings for this function, click on the *Sust. off* button. Remember to activate the corresponding node for MIDI transport of your slot in the midi_merger to ensure the functionality is enabled correctly.

noteoff_onexit, if some notes may remain 'played' when switching patches (plugins or machines not perfectly stable), **UMX** can transmit an *all notes off* CC command before loading a new patch.

To verify the settings for this function, click on the *note-off* button. Remember to activate the corresponding node for MIDI transport of your slot in the midi_merger to ensure the functionality is enabled correctly.

Midi Merger

The MIDI merger can handle up to 16 sources; any equipment or software capable of generating MIDI messages can be used for one or more trunks. A trunk is a 'tunnel'; messages enter **UMX**, can be modified or filtered, and are then retransmitted to the specified output. Incoming and outgoing messages are continuously monitored by two VU-meter-like groups. The structure of a trunk includes:



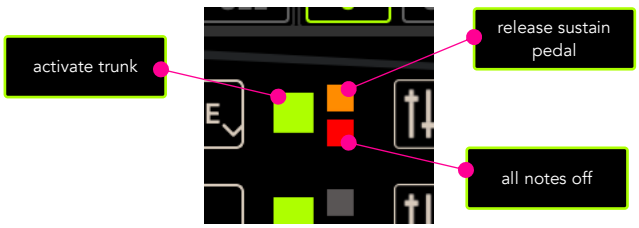
MIDI Input: Any hardware or virtual MIDI source (keyboard, pickup-midi, wind controller, pitch-to-midi software, etc.). The software constantly updates the list of available MIDI sources; any instrument or MIDI interface connected to the computer is made immediately available.

Single Opening of MIDI Ports: The software opens each MIDI port only once and then duplicates the signal for use on multiple trunks simultaneously.

Output Channel and MPE mode: Once an input is selected, **UMX** receives on all 16 MIDI channels simultaneously; the messages are retransmitted to the output on a single channel. It's possible to use the same input on multiple trunks and redirect the output to different ports and channels. It is also possible to configure the MIDI input as **MPE (*MIDI Polyphonic Expression*).**

Trunk activation: when a trunk is active, incoming *note-on/note-off* MIDI messages are forwarded from the input to the output. If the trunk is inactive, these messages are blocked. However, an inactive trunk can still receive messages for routing via the **pass-through** *manager* (see below).

On-exit funtion: UMX supports quick reset of your MIDI instruments in switching scenarios. Each trunk can transmit a standard *all notes off* or *sustain pedal release* message just before jumping to a new patch.



Trap: MIDI sources (controllers in the language of the standard) can transmit many parameters, some of which may be undesired. For example, some keyboards send a *program change* when a sound is selected. If you're using that machine as a controller to play a plugin in your DAW, the *program change* might reach the plugin and cause unwanted sound changes. The **Trunk Trap** lets you quickly block specific types of incoming messages, such as *control-changes* (*CC*) or *program-changes* (*PC*), preventing them from reaching the destination. It's a fast and effective way to keep your MIDI flow clean and under control.

The trap still allows MIDI controls related to performance to pass:

sustain-pedal CC, *pitch-bend*, and *aftertouch* are retransmitted to the output. You can verify their proper functioning thanks to the midi_meters.

Pass through manager: (see dedicated chapter) While the Trap acts as a filter, the pass_through Manager allows you to take full control of how a MIDI controller behaves—essentially re-profiling its output in real time. Using the UMX MIDI profiler, you can apply:

- Filtering of specific message types
- Controller curves, to reshape how your controller responds
- Dynamic redirection of incoming data to different outputs or destinations

With these tools, a standard MIDI controller can be adapted to behave exactly the way your setup requires—whether it's live performance, studio work, or hybrid workflows. Profiling is applied per trunk, meaning each input/output path can be configured independently, allowing for powerful modular setups.

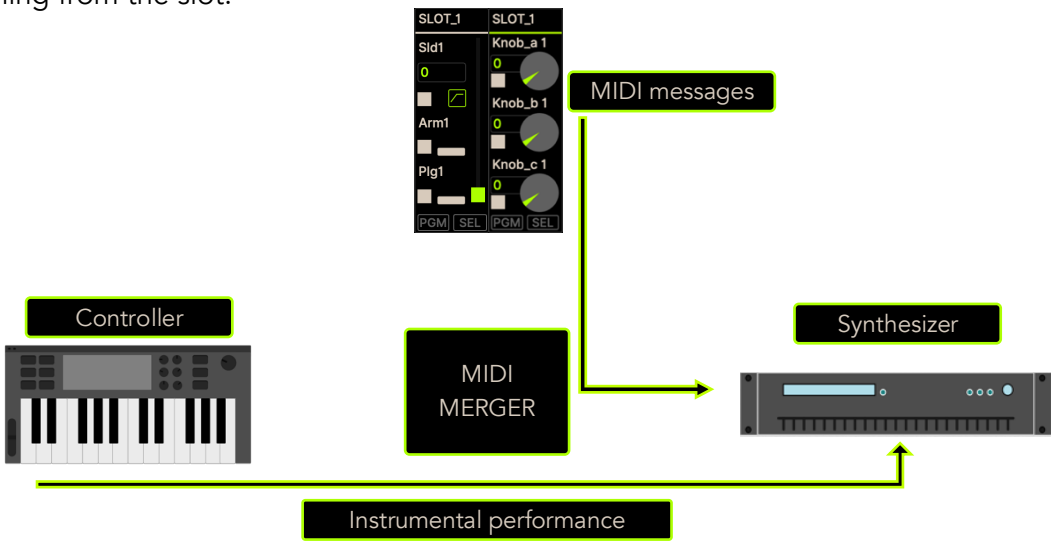
Notes range extension: For each patch, it's possible to set the note range that will be retransmitted. This allows you to use the same MIDI controller to drive different sounds with specific ranges. To set a keyboard range, press the button:

The button flashes to indicate learning mode. Play the lowest and highest notes you wish to transmit through the trunk on your controller (the order doesn't matter). The names of the notes will immediately appear to the right of the learning button:

You can reset the extension values with a right-click on the learning button.

Transposition: For each patch, you can set a note transposition. Select the text field and type a transposition value in semitones (12 transposes an octave up, -12 an octave down).

Node Matrix: It's possible to assign the transmissions of each slot in the performance to a trunk. The destination instrument will receive both the notes played from the input controller on the trunk and the messages coming from the slot.

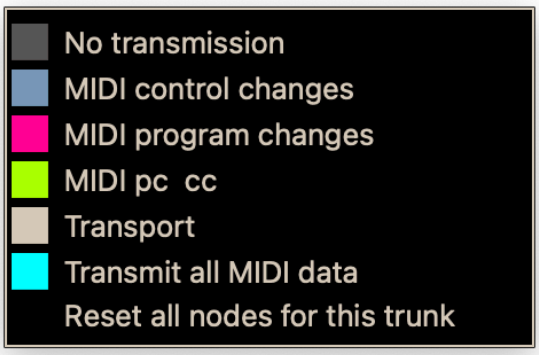


When you hover your mouse over the nodes in your trunk, the corresponding slots will be highlighted in the performance_area. Selecting a slot using the *SEL* button will display which nodes are related to the given slot in the merger space. These features help identify which node to activate to inject a slot transmission into a trunk. The software merger displays up to 256 nodes.

Activation Levels

As we have seen, each slot has the ability to transmit MIDI signals. Slots are common to all trunks, and thanks to node activation, it is possible to select which MIDI signal is used on which trunk. For example, a standard_slot can trigger a *program change* to one MIDI output, send a *play* MMC command to another output, and transmit a *control change* (CC) to yet another output, all by recalling a single patch.

The functions of a node depend on which slot is used in that particular position. Right-clicking on a node will show the slot-related menu. Clicking on a node multiple times toggles through different transmission levels. The available options for a standard_slot are:



Complexity and Recommendation

The matrix system for trunks, coupled with the ability to enable or disable the transmission of each element in the slots, allows for highly customized but also complex MIDI transmission scenarios. It's advisable to start experimenting with simpler setups, such as using just one controller and one slot.

Transmit MIDI Clock: **UMX** can act as MIDI clock sending a standard synchronization signal to all outputs (see tempo management).

MIDI Meters (In/Out): Next to the MIDI input and output buttons, two MIDI meters are displayed to show trunk activity. Clicking on a meter allows you to assign a custom name to the corresponding input or output port, making it easier to identify how your instruments are connected.

The pass_through manager (MIDI profiling)

Starting with version 1.4.2, MIDI trunks are equipped with a powerful new feature: the **pass_through** manager, accessible for each trunk via its dedicated icon.

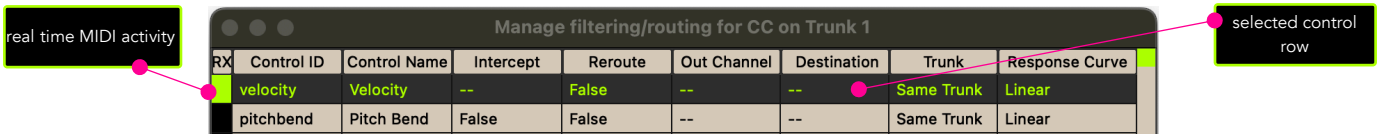
The **pass_through** manager is now displayed:

Three main functionalities are available:

MIDI controls can be intercepted and blocked, forwarded with values modified by a response curve, or routed to another control and/or to a different trunk.

Each of the 16 MIDI trunks can handle more than 130 simultaneous controls in real time.

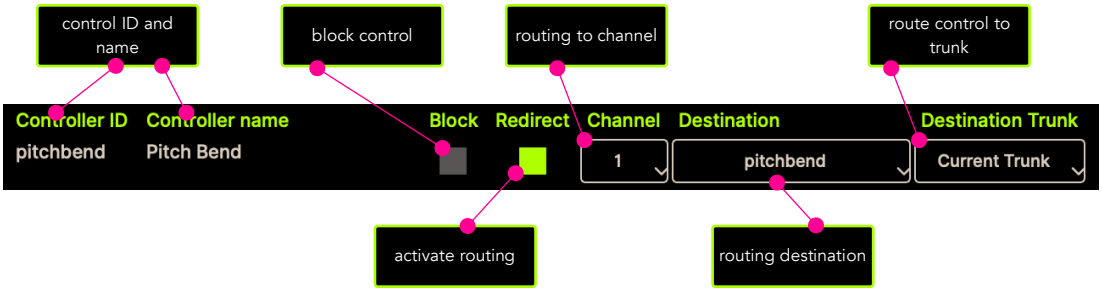
Select a control in the list:



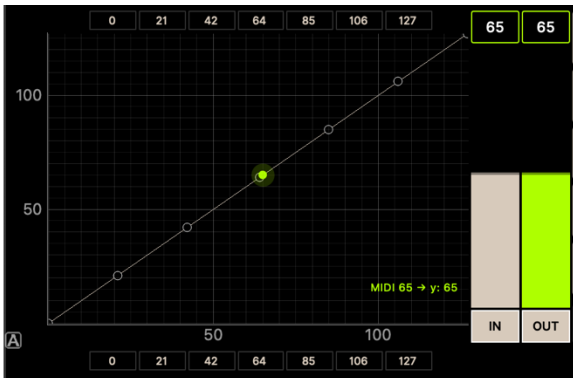
Controls refer to performance parameters such as *velocity*, *channel aftertouch*, and other MIDI *control change* messages.

An activity indicator shows whether the trunk is receiving MIDI messages, grouped by type.

Once a control is selected the control_editor displays the available settings,



The curve_area also displays the current response curve assigned to the selected the control:



Not all controls behave in the same way, some options are not available for specific control types.

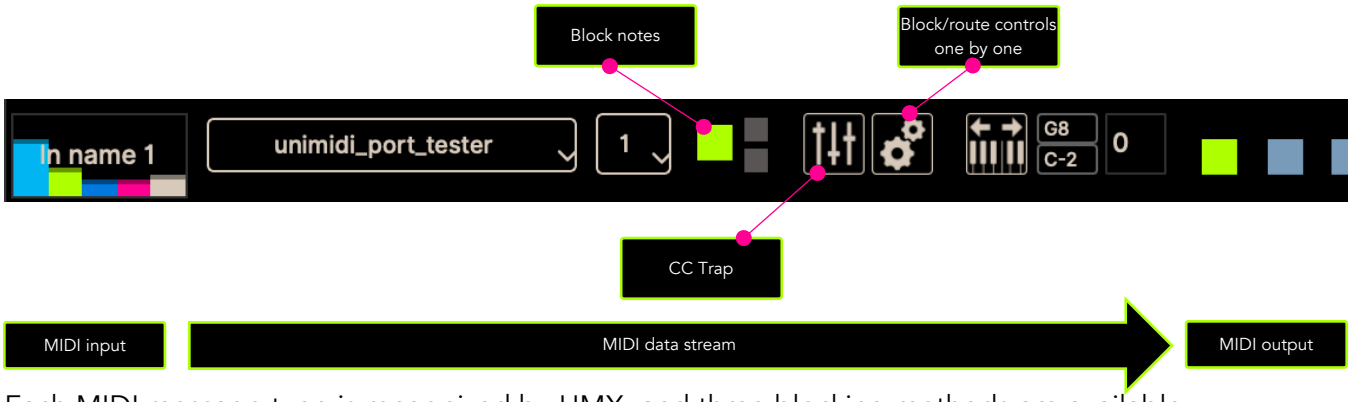
For example, velocity cannot be blocked, since this MIDI value is always transmitted as part of a *note-on* message. Blocking it would require blocking the entire message. However, *note-on* messages can be blocked globally using the trunk activation settings.

Similarly, not all controls support a response curve.

For instance, *sysex*, *MMC*, and other system-level messages cannot be processed with a response curve, as modifying them would alter the original MIDI data and compromise its intended function.

Blocking a control

When a MIDI source is selected as the input of a trunk, UMX starts listening to all incoming messages from it:



Each MIDI message type is recognized by UMX, and three blocking methods are available:

Trunk Activation Button

This control allows you to block or allow note messages. Notes are received but not retransmitted.

CC Trap

This function blocks all standard MIDI control messages (*control-changes, program-changes, sysex,* etc.) while allowing performance-related messages (*notes, pitch bend, aftertouch*) to be retransmitted.

These functions let you quickly turn on or off a MIDI flow to an audio plugin, software instrument, or physical port, or selectively block unwanted messages like *program-changes* without interfering with the musical performance messages.

pass_through Manager

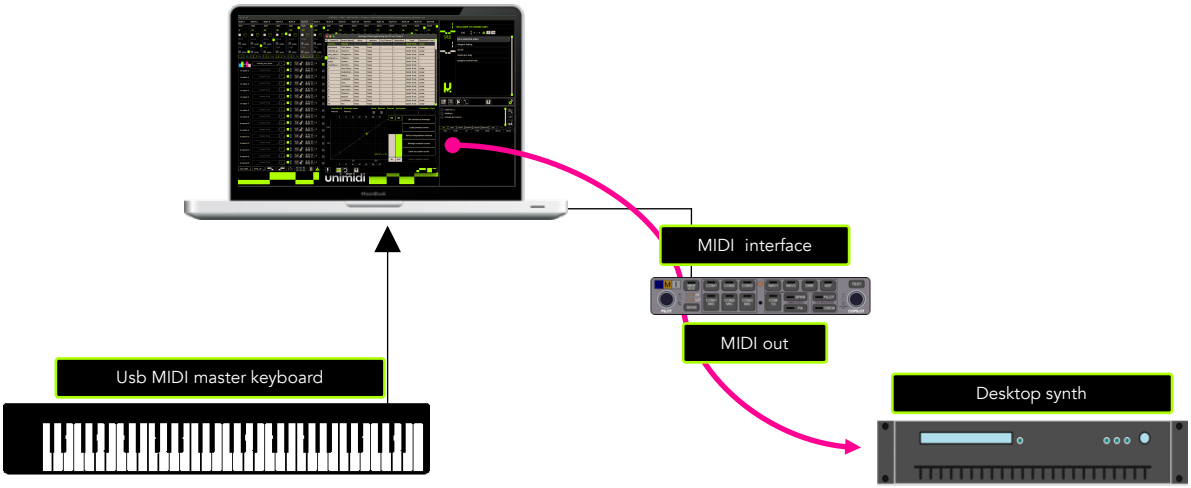
The third method, allows you to precisely define which MIDI messages are allowed to pass to the output.

Please note that when a control is processed by the pass_through manager, it is excluded from CC Trap handling.

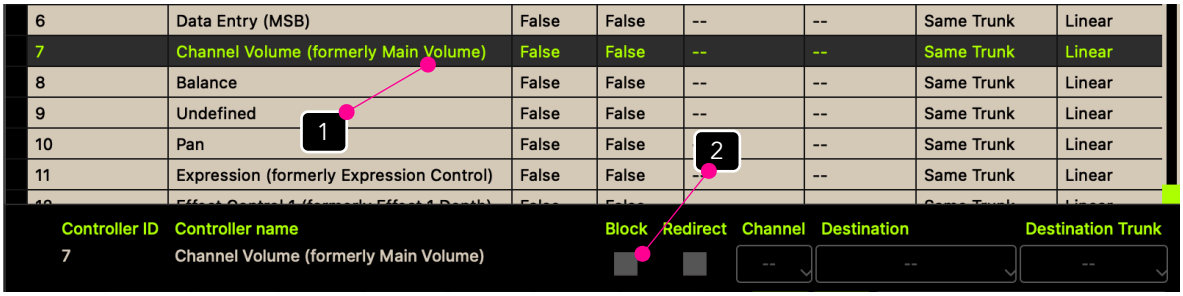
For instance, if a control change message is being routed or shaped by a response curve, CC Trap will not be able to block it anymore.

Let's look at a complete blocking procedure for a specific MIDI controller: *cc7 global volume*

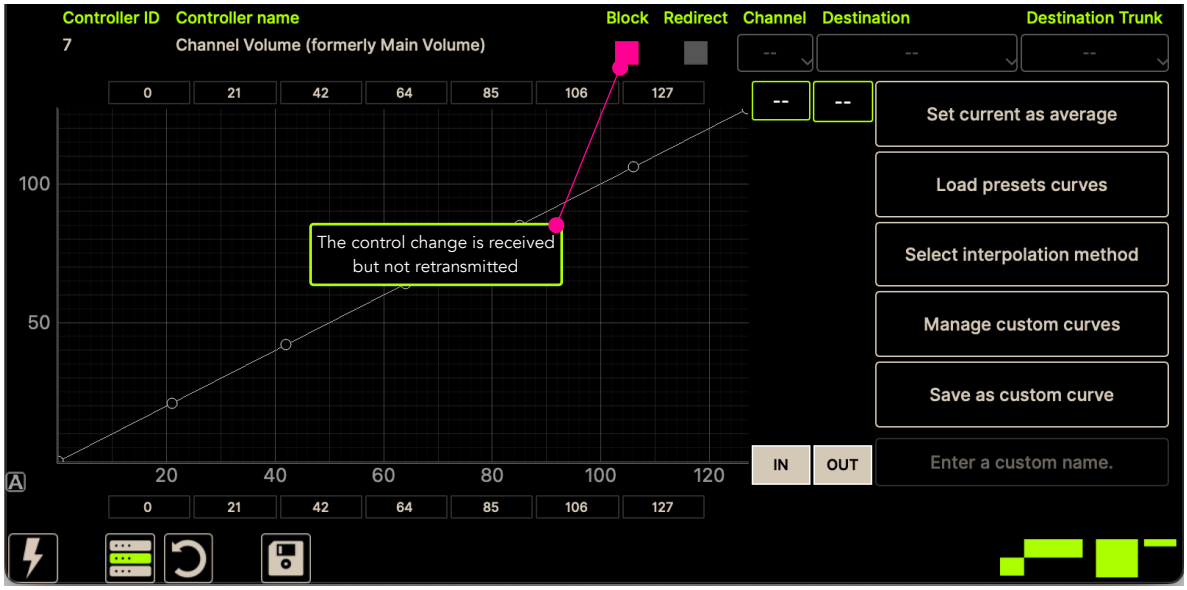
We have a master keyboard that sends unwanted volume control messages to a desktop synthesizer, both devices are connected to UMX via physical MIDI ports or through MIDI on USB or Bluetooth MIDI:



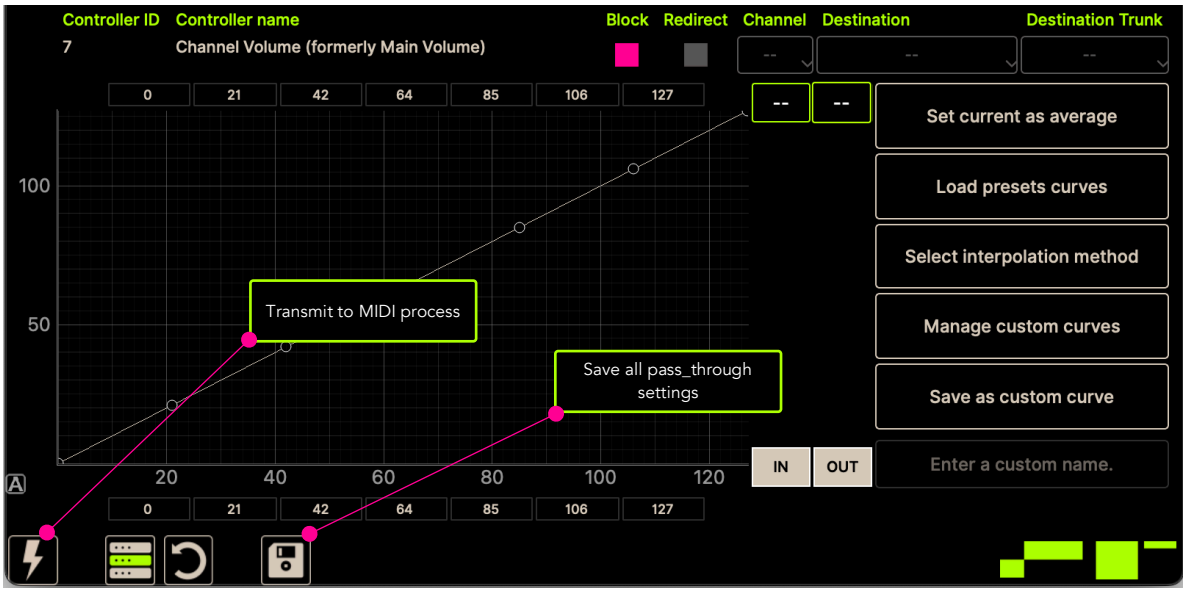
Open the pass_through manager and click on the control: *CC7 global volume*, then click to intercept and block it:



Changes are immediately reflected in the pass_through manager interface but they have no effects on the actual MIDI transmission yet:



To apply these settings, you need to transfer them to the MIDI process by clicking the *apply* button



Now, *cc7* is filtered out by UMX and will no longer reach the desktop synth, even though all other MIDI messages are still allowed to 'pass through'.

You will notice that the pass_through manager icon in the main window has changed, indicating that the trunk has been configured with one or more rules

Once the pass_through manager is launched, it temporarily stores all trunk settings.

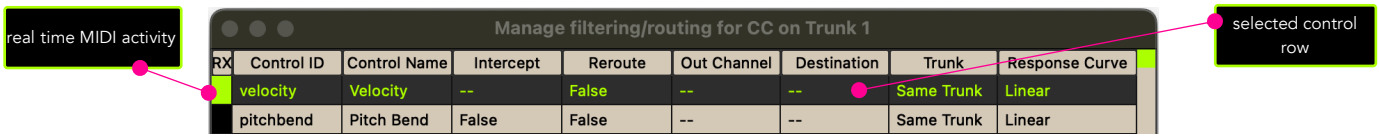
You can freely experiment and apply new configurations to the MIDI process in real time, observing the results by ear, through the MIDI meters, or using the embedded MIDI monitor.

If you're satisfied with the outcome, you can permanently save the settings by clicking the *save* button.

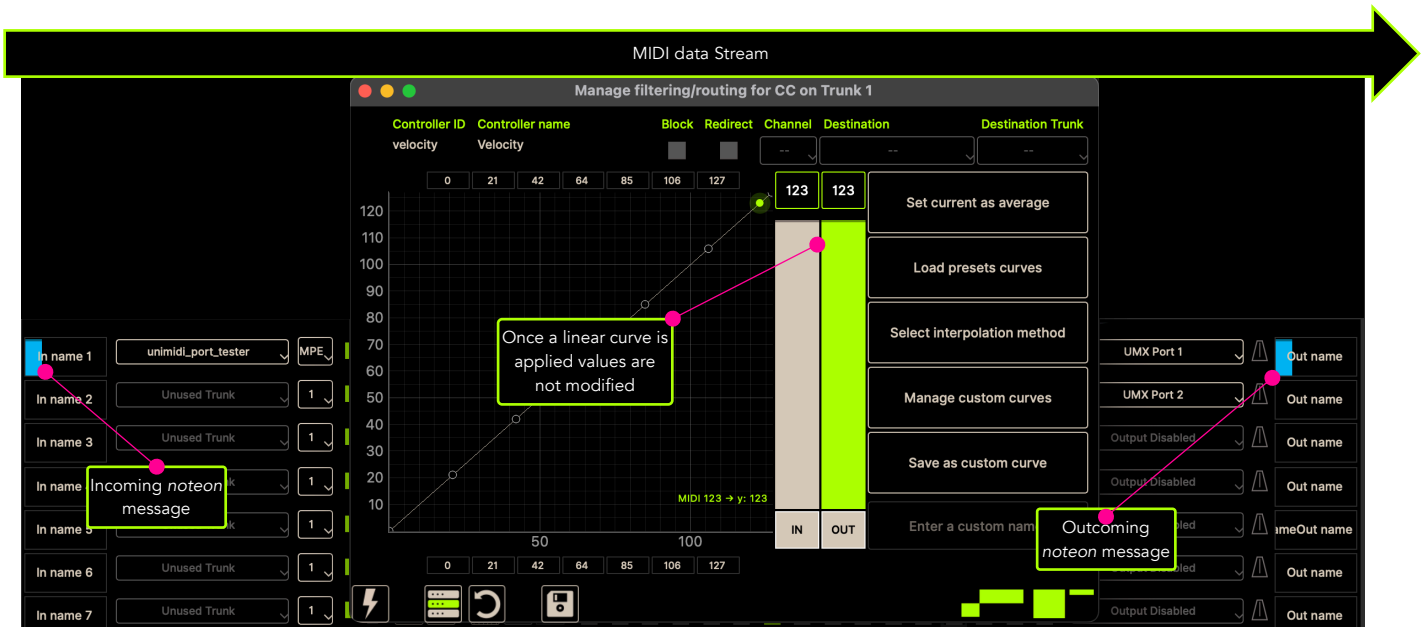
Alternatively, simply close the pass_through window — UMX will prompt you to either save the changes or discard them and revert to the initial state.

Setting a control response curve

One of the key features of the pass_through manager is the ability to apply a response curve to incoming MIDI messages, once a control is selected:

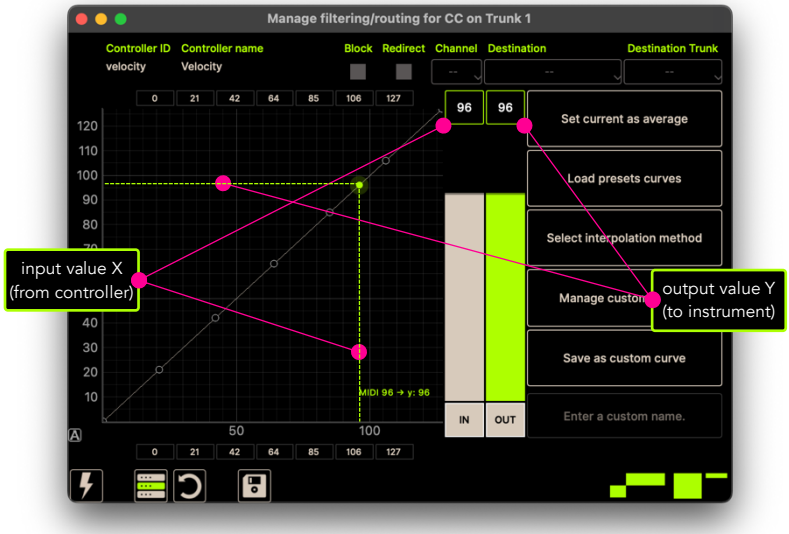


The corresponding curve is also displayed, if one exists:



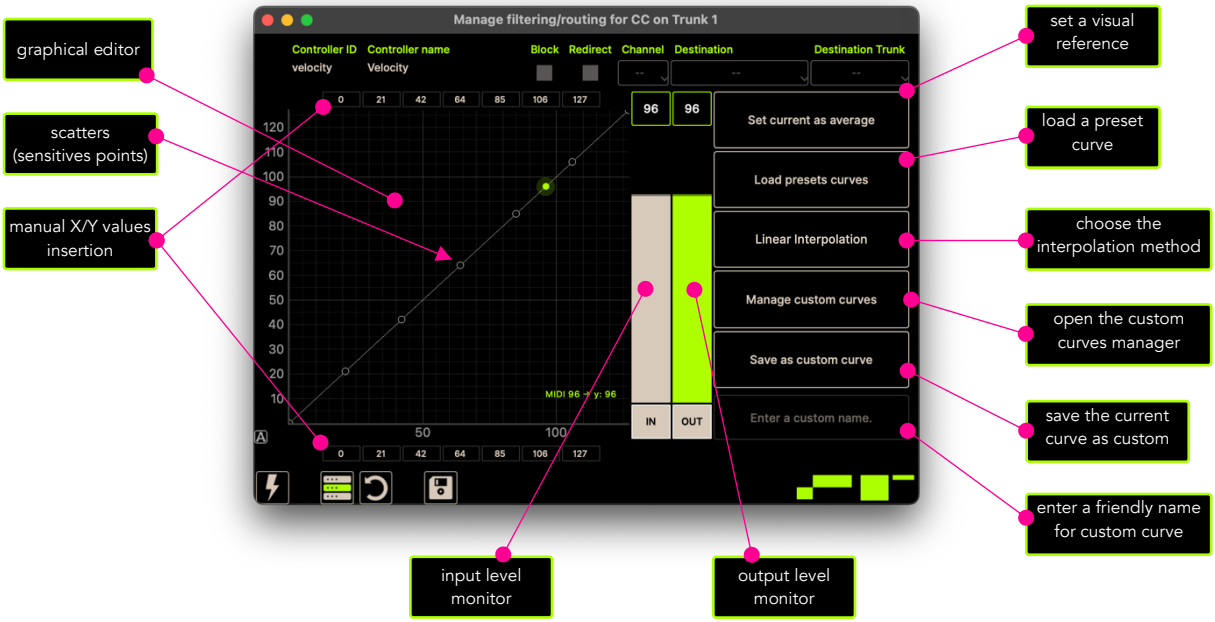
For example, if we want to modify how the *velocity* of a MIDI controller affect the sound, we can select it in the pass_through manager, the current curve (which is *Linear* by default) will be displayed.

By playing some notes on the controller, we will see value passing through the curve editor:



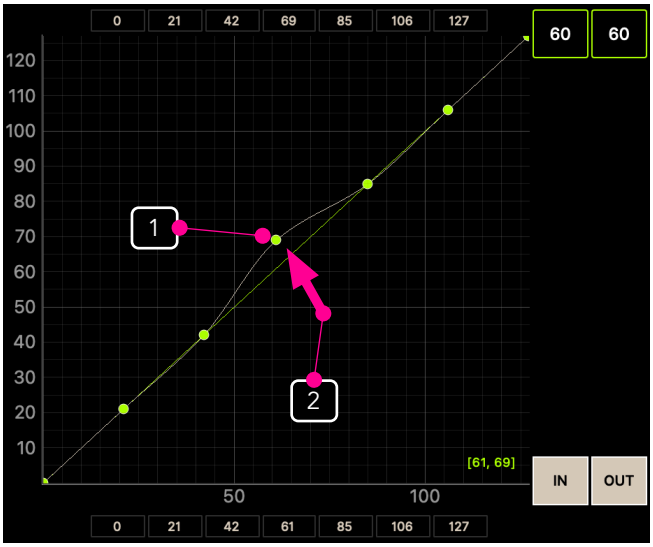
Obviously, with a *Linear* curve, there will be no difference between input and output values.

The curve editor features are:



Curves are defined by seven scatter points and an interpolation method that calculate intermediate values, users can manually draw custom curves by clicking and dragging the scatter points or manually entering X/Y values in their fields and pressing the enter key.

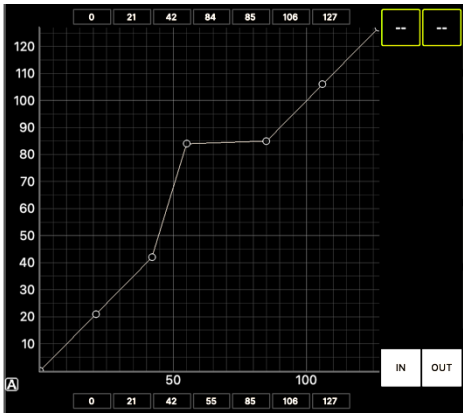
To move a scatter point:



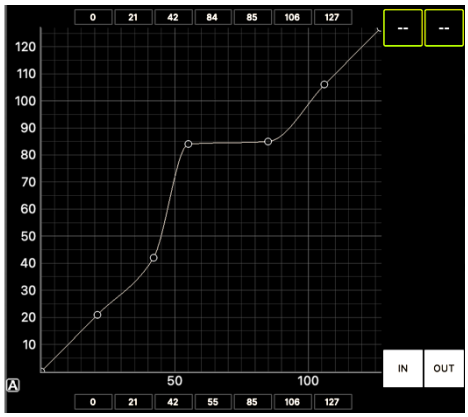
1) Click on a scatter point: all points will be highlighted to indicate that you are in edit mode.
2) Move your pointer: the curve will follow movements, to release the scatter point simply release de mouse button or click on the final position if you are using a trackpad.

Click the *transmit to process* button to apply changes to the actual MIDI transmission.

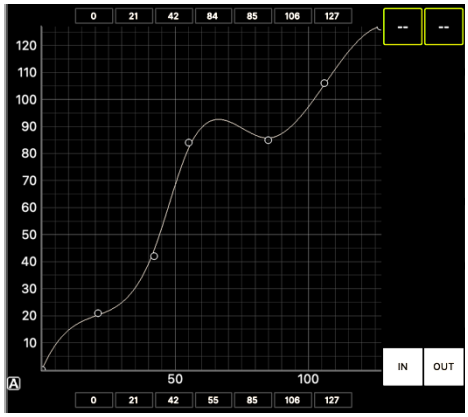
Scatter points allows you to define a complete curve using only seven pairs of X/Y coordinates, to reconstruct the entire curve UMX uses one of three interpolation methods:



Linear interpolation



Cubic interpolation



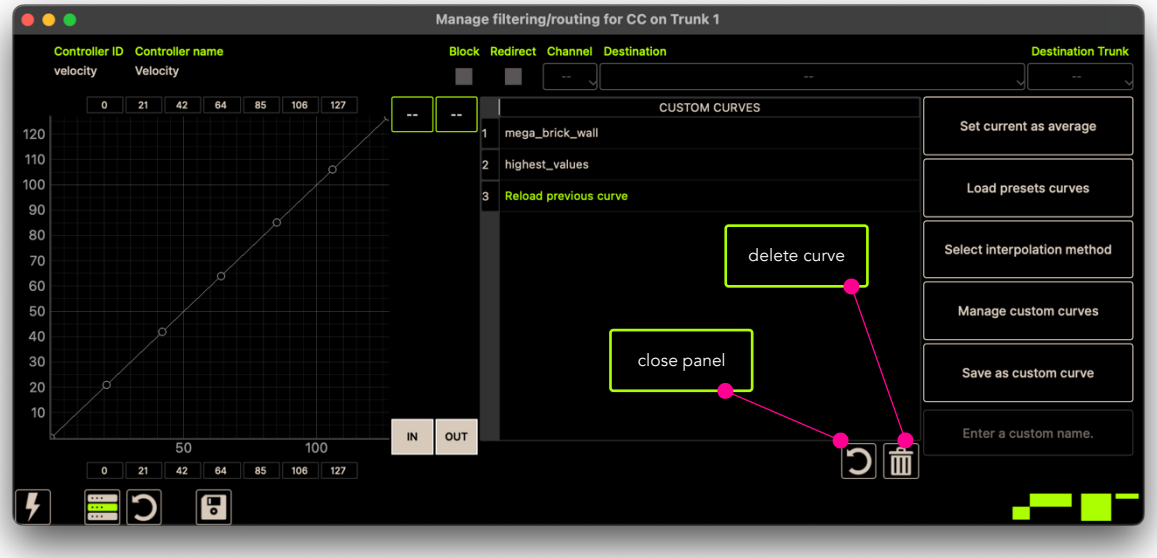
Spline interpolation

- **Linear interpolation** simply fills in the X/Y values between one scatter point and the next.
- **Cubic interpolation** creates a smooth transition between points.
- **Spline interpolation** apply a polygonal computation between scatter points, allowing for the smoothest value transitions.

You can use these methods to define your own curves. Once you're satisfied, choose a unique curve name and save the curve. All custom curves are available across all the trunks and in all the UMX performances.

Manage custom curves

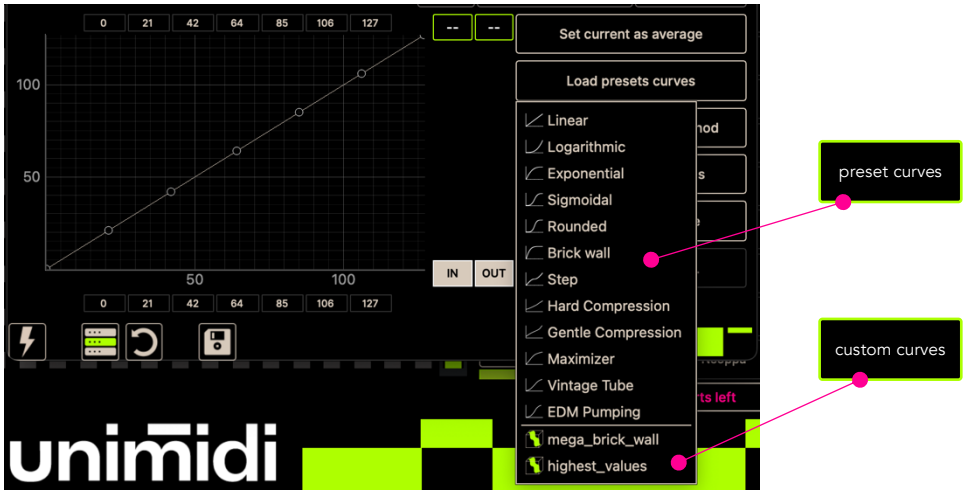
Clicking on the manage custom curves button displays an additional panel in the curve manager:



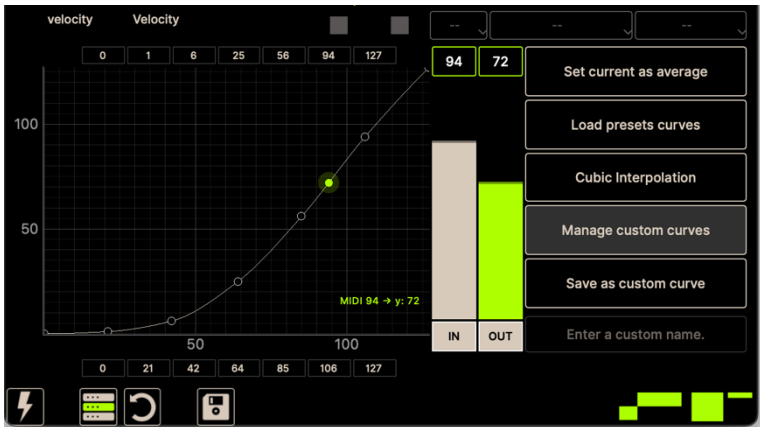
Selecting a custom curve name will load it immediately. You can reload the previously used curve at any time. Curves can also be deleted.

Load a curve preset

UMX includes a set of presets, by clicking the *load presets curves* button, a menu will appear, simply select one of the available presets or a custom curve.



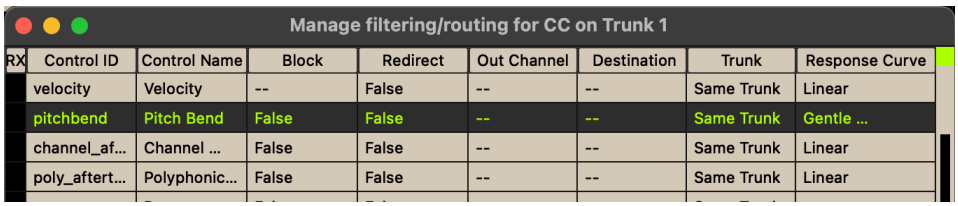
Once a curve is loaded, it immediately starts simulating how output values are affected by the curve compared to the input values:



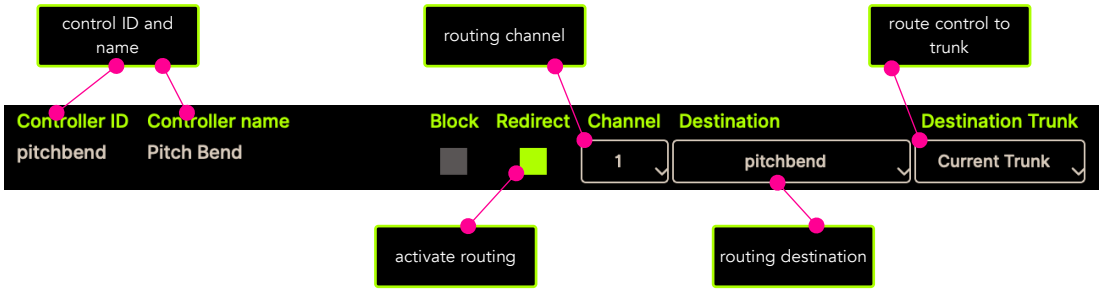
Remember to click the *transmit to process* button to apply the new curve to the current MIDI transmission.

Routing a control

As mentioned, the pass_through manager also allows you to routing an incoming MIDI message to another. Any control from the control list can be used as the routing source:

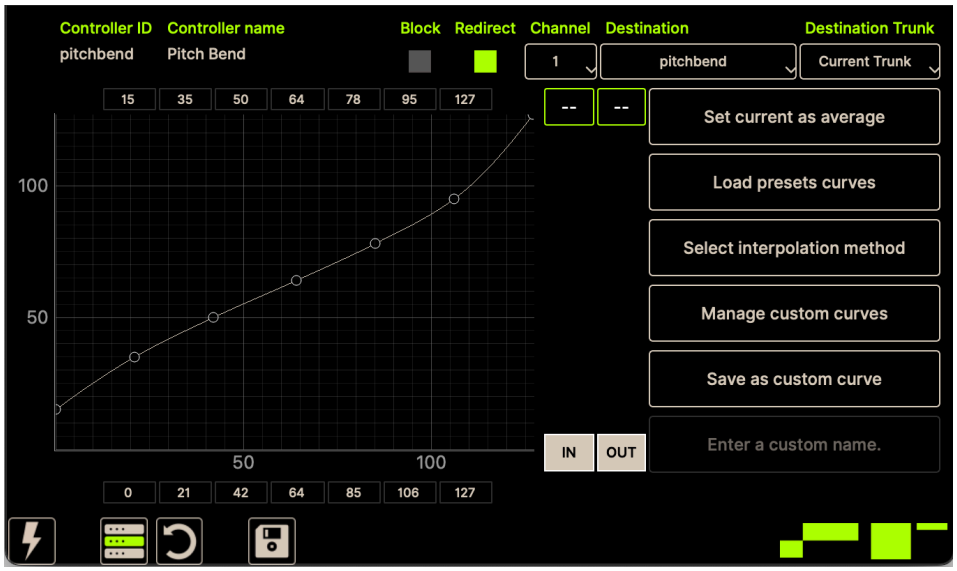


The routing options are:



For example, routing can be activated for pitch bend control.

The parameters channel, destination and destination trunk become immediately active and can be configured:



Channel – as mentioned, a trunk receives on all midi channel simultaneously. Use this menu to select which channel will be used for the redirected control.

Destination – all controls can be redirected to these output midi signals:

control	name	control	name	control	name
channel_aftertouch	Channel Aftertouch	40	Portamento Control	85	Undefined
poly_aftertouch	Polyphonic Aftertouch	41	Undefined (LSB)	86	Undefined
program_changes	Program Changes	42	Undefined (LSB)	87	Undefined
sysex	System Exclusive	43	Expression Controller (LSB)	88	High Resolution Velocity Prefix
machine_control	Machine Control MMC	44	Effect Control 1 (LSB)	89	Undefined
0	Bank Select (MSB)	45	Effect Control 2 (LSB)	90	Undefined
1	Modulation Wheel or Lever	46	Undefined (LSB)	91	Effects 1 Depth (Reverb Send Level)
2	Breath Controller	47	Undefined (LSB)	92	Effects 2 Depth (Tremolo Depth)
3	Undefined	48	General Purpose Controller 1 (LSB)	93	Effects 3 Depth (Chorus Send Level)
4	Foot Controller	49	General Purpose Controller 2 (LSB)	94	Effects 4 Depth (Detune Depth)
5	Portamento Time	50	General Purpose Controller 3 (LSB)	95	Effects 5 Depth (Phaser Depth)
6	Data Entry (MSB)	51	General Purpose Controller 4 (LSB)	96	Data Increment (Data Entry +1)
7	Channel Volume (formerly Main Volume)	52	Undefined (LSB)	97	Data Decrement (Data Entry -1)
8	Balance	53	Undefined (LSB)	98	Non-Registered Parameter Number (NRPN) LSB
9	Undefined	54	Portamento Control (LSB)	99	Non-Registered Parameter Number (NRPN) MSB
10	Pan	55	Undefined (LSB)	100	Registered Parameter Number (RPN) LSB
11	Expression (formerly Expression Control)	56	Undefined (LSB)	101	Registered Parameter Number (RPN) MSB
12	Effect Control 1 (formerly Effect 1 Depth)	57	Undefined (LSB)	102	Undefined
13	Effect Control 2 (formerly Effect 2 Depth)	58	High Resolution Velocity Prefix (LSB)	103	Undefined
14	Undefined	59	Undefined (LSB)	104	Undefined
15	Undefined	60	Undefined (LSB)	105	Undefined
16	General Purpose Controller 1	61	Undefined (LSB)	106	Undefined
17	General Purpose Controller 2	62	Undefined (LSB)	107	Undefined
18	General Purpose Controller 3	63	Undefined (LSB)	108	Undefined
19	General Purpose Controller 4	64	Damper Pedal On/Off (Sustain) (MSB)	109	Undefined
20	Bank Select (LSB)	65	Portamento On/Off (MSB)	110	Undefined
21	Modulation Wheel or Lever (LSB)	66	Sostenuto On/Off (MSB)	111	Undefined
22	Breath Controller (LSB)	67	Soft Pedal On/Off	112	Undefined
23	Undefined (LSB)	68	Legato Footswitch	113	Undefined
24	Foot Controller (LSB)	69	Hold 2	114	Undefined
25	Portamento Time (LSB)	70	Sound Controller 1	115	Undefined
26	Data Entry (LSB)	71	Sound Controller 2	116	Undefined
27	Channel Volume (LSB)	72	Sound Controller 3	117	Undefined
28	Balance (LSB)	73	Sound Controller 4	118	Undefined
29	Undefined (LSB)	74	Sound Controller 5	119	Undefined
30	Pan (LSB)	75	Sound Controller 6	120	All Sound Off
31	Expression (LSB)	76	Sound Controller 7	121	Reset All Controllers
32	Effect Control 1 (LSB)	77	Sound Controller 8	122	Local Control On/Off
33	Effect Control 2 (LSB)	78	Sound Controller 9	123	All Notes Off
34	Undefined (LSB)	79	Sound Controller 10	124	Omni Mode Off
35	Undefined (LSB)	80	General Purpose Controller 1	125	Omni Mode On
36	General Purpose Controller 5	81	General Purpose Controller 2	126	Mono Mode On
37	General Purpose Controller 6	82	General Purpose Controller 3	127	Poly Mode On
38	General Purpose Controller 7	83	General Purpose Controller 4		
39	General Purpose Controller 8	84	Portamento Control		

Understanding MIDI message basics

MIDI messages are made of one or more bytes. Most messages follow a standard format:

STATUS BYTE		DATA BYTES	
1st Byte Value Binary |Hex| Dec	Function	2nd Byte	3rd Byte

Status Byte: Indicates the type of message (e.g., *note-on*, *control change*, *pitch bend)* and the MIDI channel.

Data Bytes: Contain values such as note number, velocity, controller number, or pitch bend amount.

Common Message Types

Status byte		Message type	Message format	example	notes
Hexadecimal range	Decimal range				
0x80–0x8F	128–143	*note off*	*[status byte, note, velocity]*	*[128, 60, 100]*	*note off, middle C, (velocity can be omitted)*
90–0x9F	144–159	*note on*	*[status byte, note, velocity]*	*[144, 60, 100]*	*note on, middle C, velocity 100*
0xA0–0xAF	160–175	*polyphonic key pressure*	*[status byte, note, value]*	*[160, 50, 90]*	--
0xB0–0xBF	176–191	*control changes*	*[status byte, controller number, value]*	*[176, 7, 90]*	*Used for knobs, sliders, pedals, etc.*
0xC0–0xCF	192–207	*program changes*	*[status, program number]*	*[192, 10]*	*Selects a patch or sound; only one data byte*
0xD0–0xDF	208–223	*channel aftertouch*	*[status, value]*	*[208, 120]*	--
0xE0–0xEF	224–239	*pitch bend*	*[status, lsb, msb]*	*[224, 64, 64]*	*Uses two data bytes for high-resolution pitch changes*
0xF0–0xF7	240–247	*system common messages*	*System Exclusive (SysEx): [start, byte, byte, … , end]*	--	*SysEx messages start with 0xF0 and end with 0xF7 (EOX – End Of Exclusive). They can contain any number of data bytes and are used for:* *Manufacturer-specific feature* *Device configuration* *Bulk data transfer (e.g., patch dumps)*
0xF8–0xFF	248–255	*system real-time messages*	*MIDI Machine Control (MMC): [status, none, none]*	--	*Play, Stop, Continue*

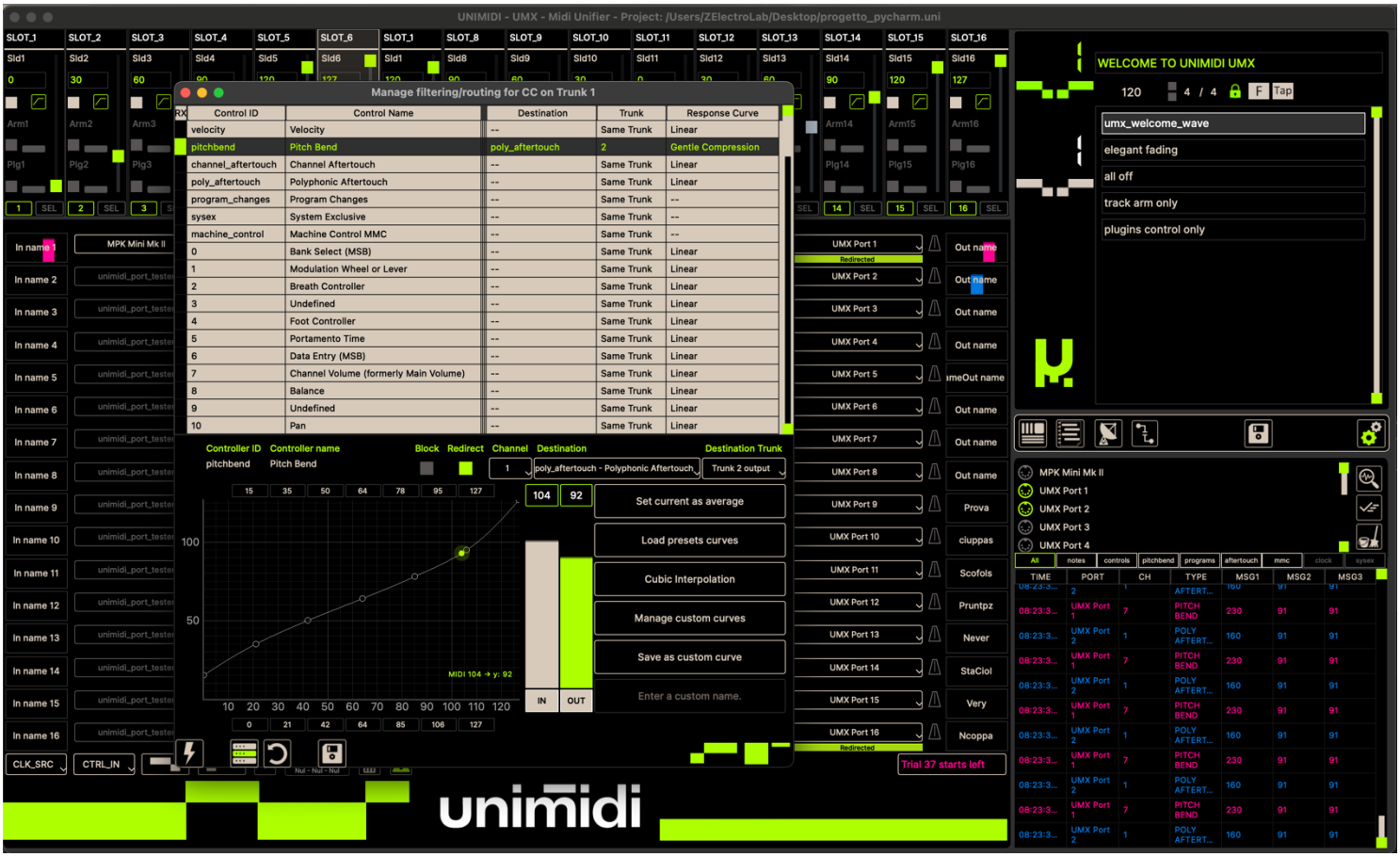
UMX allows to redirect almost everything, *pitch bend* can become *aftertouch* or *CC7 volume* or virtually anything else. In general, the UMX routing algorithm try to adapts data formats to ensure coherent transmission. However, this is not always possible and you may encounter creative applications or unexpected behaviors.

Be mindful about messages compatibility!

Routing to another trunk

All the controls can be redirected to the same trunk or to a different one, if a control is redirected to another trunk a small label is displayed next to the output port name:

Below an example of the *pitch bend* control, incoming on trunk one, being redirected to trunk two as *polyphonic aftertouch*, with a gentle compression curve applied:

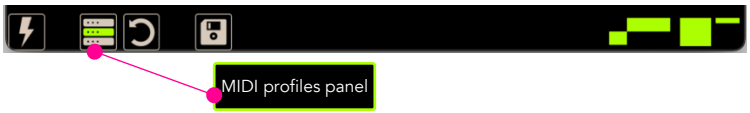


A trunk that routes MIDI messages to other trunks still requires a valid output port, even if that port is not used.

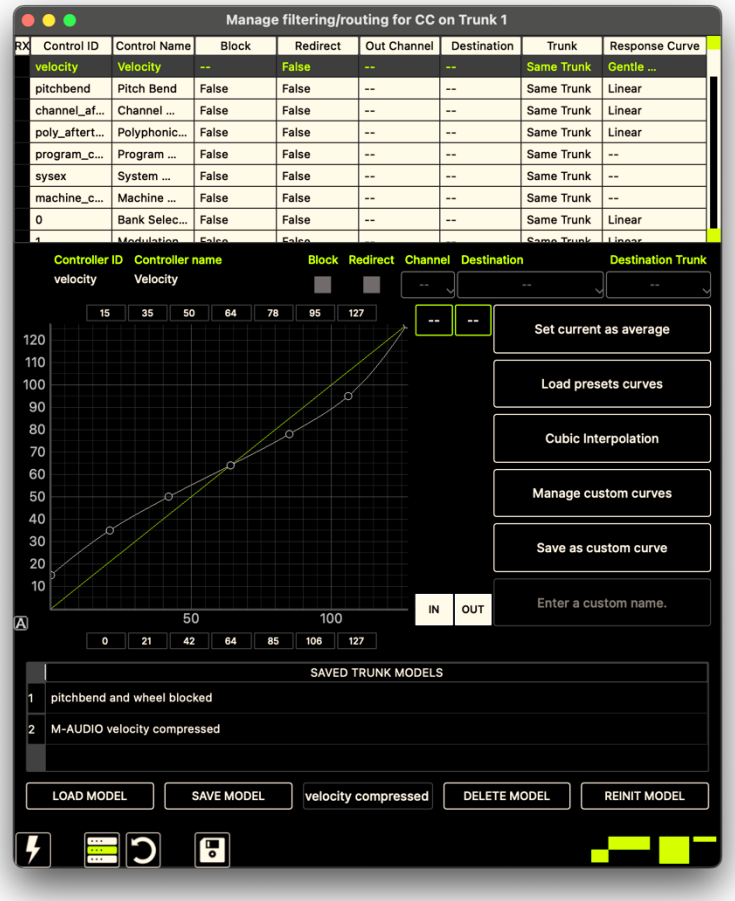
MIDI instruments profiles

For each trunk, the pass_through manager allows you to apply filters, remap controls, and reshape data using a response curve. The connected instrument receives a fully redefined MIDI output in real time, even while playing.

The pass_through manager lets you save a MIDI profile that includes all blocking filters, redirections, and response curves. To access saved profiles, click the *profiles* button at the bottom of the window:



The profiles panel is displayed:



load profile – Select a MIDI profile and load it into the current trunk.

save profile – Enter a name for the current configuration and save it as a new profile.

delete profile – Remove the selected profile from the list.

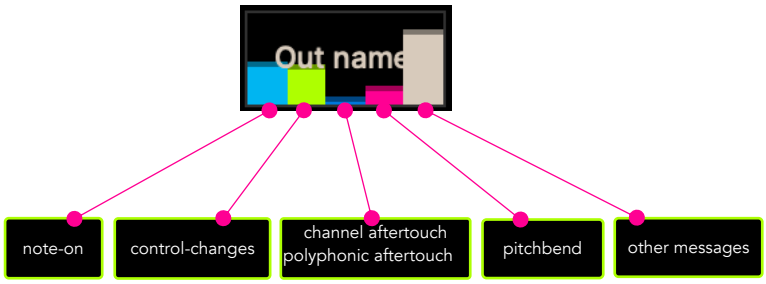
reinit profile – Resets all controls to their default configuration.

The profile is instantly loaded into the pass-through trunk. However, as always, you need to transmit the settings to the MIDI process for them to take effect.

MIDI Meters

The software is equipped with a set of midi_meters to monitor incoming and outgoing MIDI messages.

Each meter has a group of five bargraph-style indicators:



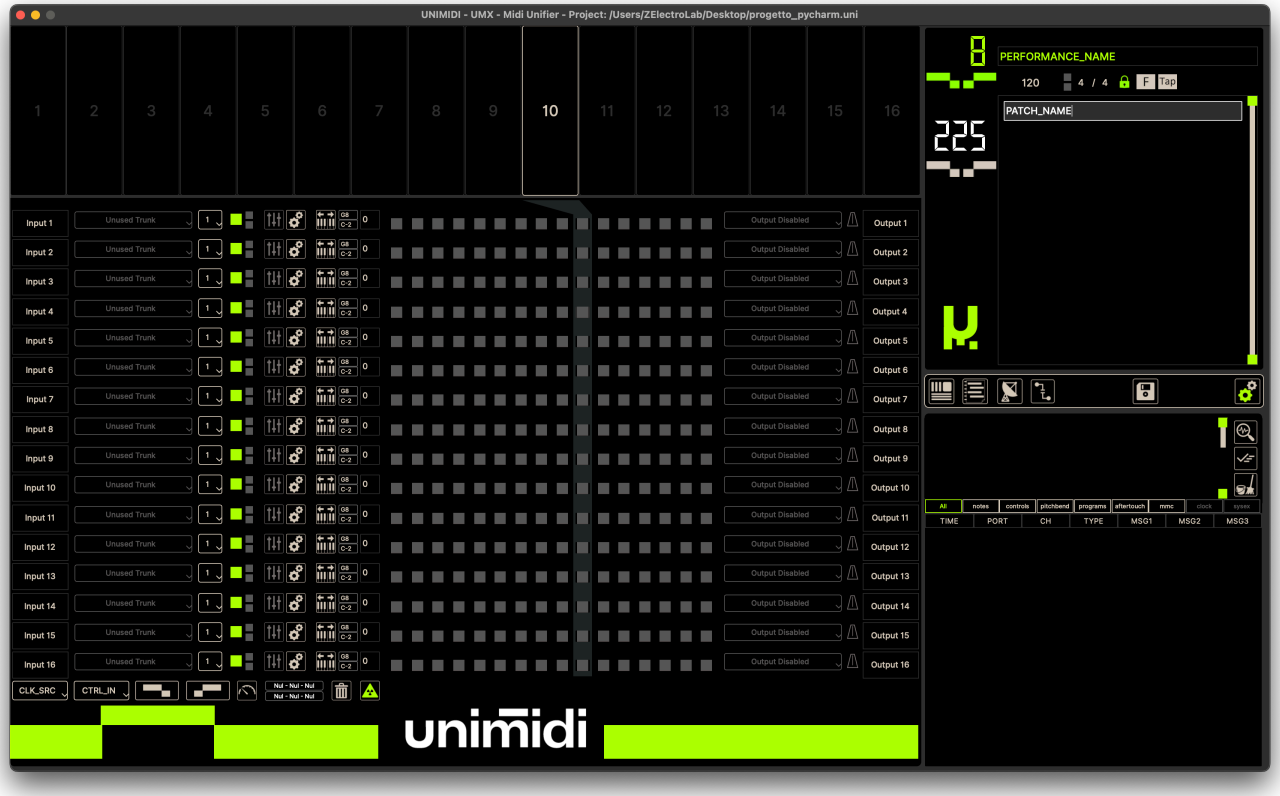
- **Note-On Messages:** The height of the bar represents the note velocity.
- **Control-Changes:** The bar height shows the current control value.
- **Aftertouch Messages:** Indicates the level of aftertouch (channel or polyphonic).
- **Pitch-Bend Messages:** Displays the pitch bend amount.
- **Other Messages:** Represents all other message types. In general, the bar displays the last meaningful data byte of the message.

In the extended interface, the input and output meters for the merger are separately available, while in the compressed interface, only the output meters are displayed.

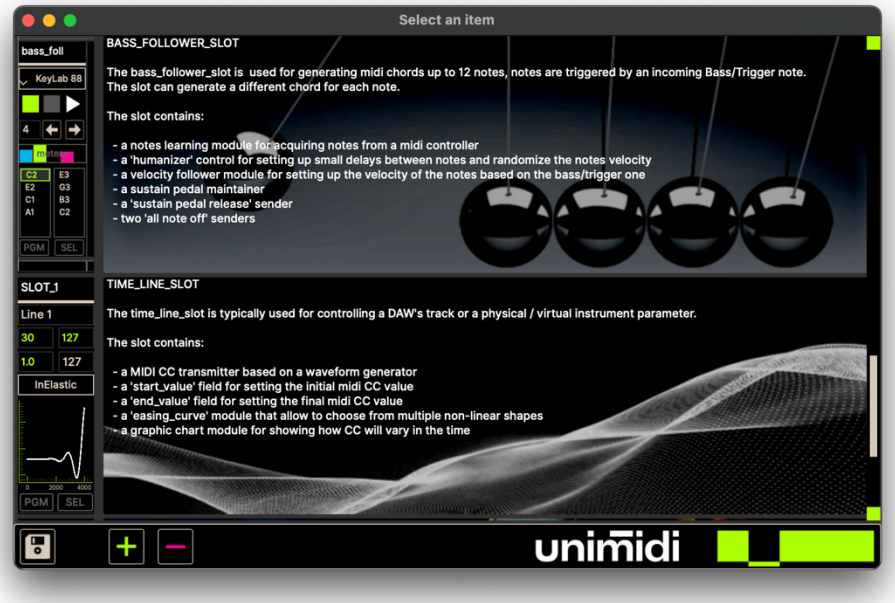
Specific colorations appear for some configurations, such as transport controls and outgoing patch controls, to indicate successful transmission:

Creating a Performance

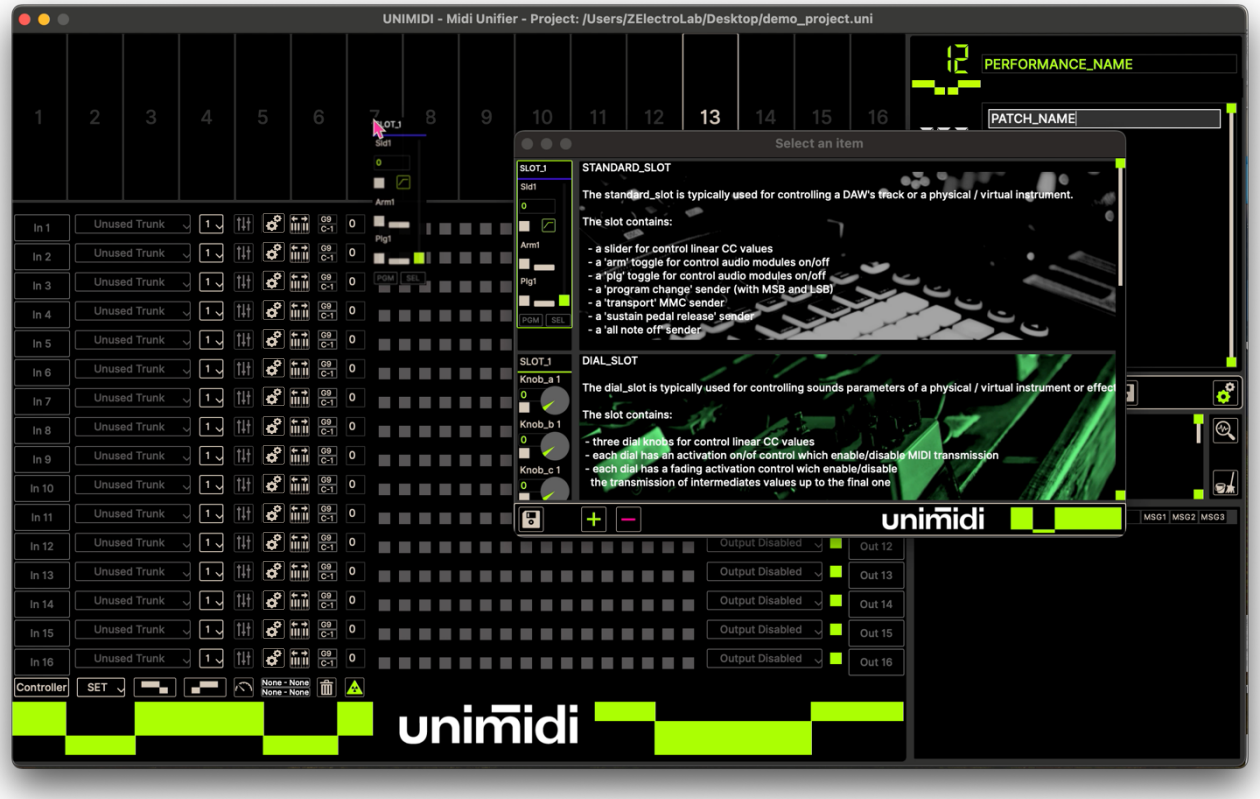
To start using the software, you first need to create a performance tailored to your instruments. Click on the *Performance* menu and select *New Performance*:



A new empty performance space is created. Right-click anywhere on the empty slots and select *Add Slot*. The slot selector panel will be displayed:



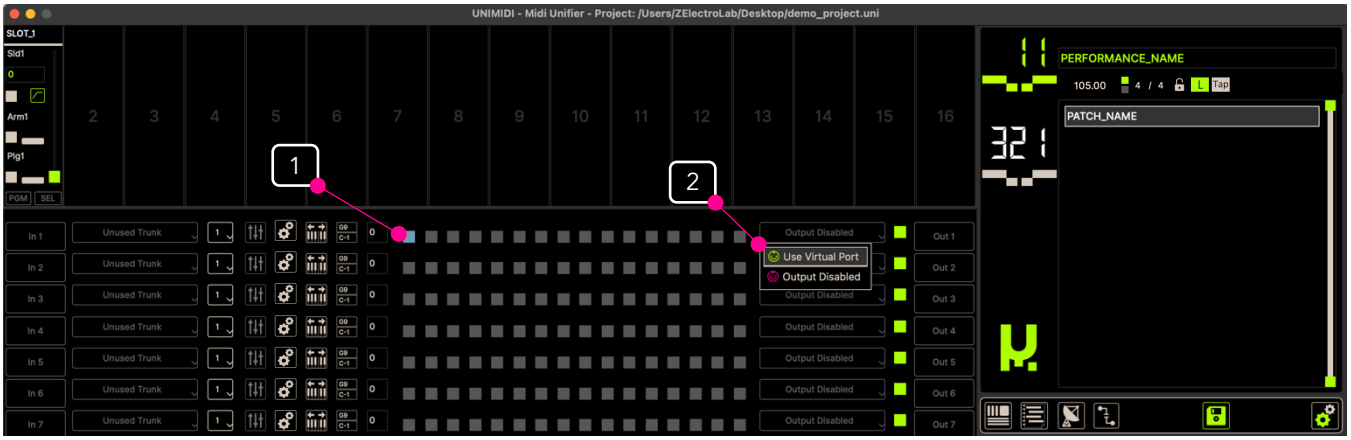
You can scroll through all the available slot types. Take the standard_slot, which is first in the list, and drag it into the performance_area:



You can move it by selecting it, pressing the *SEL* button, and then dragging it to the desired position. Once you're satisfied, hit the Save button (you can also click on the blinking *SAVE* label or just use the shortcut CTRL+S (Windows®) or CMD+S (macOS®).

Performance Configuration

Your performance is now ready for additional settings related to inputs, nodes, filters, etc.

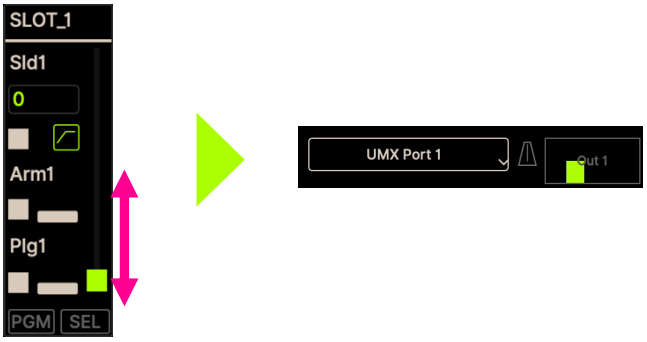


MIDI Output and DAW Configuration

1. On trunk 1, the first from the top, activate the node that correspond to the slot position (staying on a node will highlight the related slot),
2. On the same trunk, select the virtual output port (for macOS® or Linux®; for Windows®, see the section on OS-specific options).

Your slot is now ready to communicate with the external world through virtual MIDI output 1 (or another port you've selected).

If all the settings we done are corrects, and the UMX trunk is well settled to transmits MIDI messages through a virtual or physical ports, then the midi_meters shown a transmission activity when we move the slot slider.

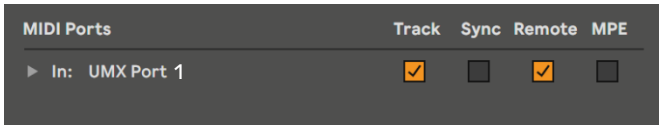


The next step is to set up a 'receiver' for our communications. The destination for the transmitted MIDI controls can be any MIDI device, such as a synthesizer, a DAW, or a lighting system. For this example, we'll control a DAW.

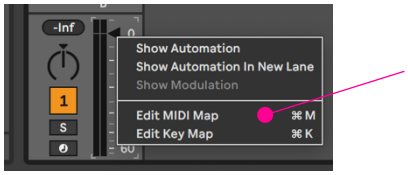
1. Launch your DAW—whichever you prefer. In our case, we'll use **Ableton®.**
2. Create a new track:

DAW Preferences and MIDI Mapping

1. Go into the software preferences (options) and locate the MIDI input configurations or 'control surfaces.'
2. The virtual output you set for our performance, '**UMX Port 1**,' should be available. Activate it and ensure it can control the software (in Ableton®, activate 'track' and 'remote'):



3. Return to the track you've just created in your DAW and activate MIDI learn or MIDI mapping for the track volume slider.
    a. In Ableton®, right-click on the volume slider and select 'Edit MIDI Map':



4. Now, moving the slider on slot 1 in the newly created UMX performance will be reflected in Ableton®, showing that the track volume slider has learned the corresponding MIDI CC. After mapping, any movement of the UMX slot slider will similarly adjust the track volume in the DAW.

UMX Slot Features and DAW Control

UMX provides different types of slots, including slider controllers, on/off switches, rotary potentiometers, note generators, and more. This versatility allows for the construction of complex performances with multiple slots, leading to complete automated control of your musical setup.

1. **Switch-type Controls**: Suitable for tasks such as arming tracks, activating or deactivating a group, plugin, or effects.
2. **Slider-type Controls:** Ideal for controlling parameters like track volume.
3. **Rotary Controls**: Designed for managing parameters such as audio sends, stereo panning, effect depth, or specific parameters within an effect or instrument.

This configuration offers a high degree of flexibility and precision, enabling the user to create nuanced and dynamic performances using **UMX** and a DAW.

Editing a Performance

An existing performance can be modified by adding and deleting slots or moving them. Use the right-click slot menu or the *Edit Performance* command from the Performance menu.

- Changes Propagation: Any modifications made to a performance (such as adding, deleting, or moving a slot) are propagated to all the patches contained within that performance upon saving.

Automapping

- Automatic Control Mapping: The software features an automatic mapping system for controls. If you create a performance containing all 16 possible slots, they will be configured with MIDI parameters relative to their creation position. For example, slot 1 will be assigned controller numbers 102, 103, 104, transmitted on channels 14, 15, and 16, respectively. This means that the entirety of the slots can be transmitted over a single MIDI port, avoiding duplicates and overlaps.

Editing Existing Performances

- Parameter Retention: If you initially created slot 1 and then move it to position 2, it will retain all its parameters (changing them would mean losing control of the target instrument and redoing part of the work).
- New Slot Addition: Adding a new slot at position 1 will inherit the same automatic mapping, resulting in two slots (1 and 2) configured in the same way.
- Potential Conflicts: If the two slots are transmitted on different ports, there will be no conflicts. However, if they are directed to the same output port (typical in DAW control where all slots are transmitted over the same port), there will be an overlap of transmitted values, with the same control-changes on the same channels.

Caution and Verification

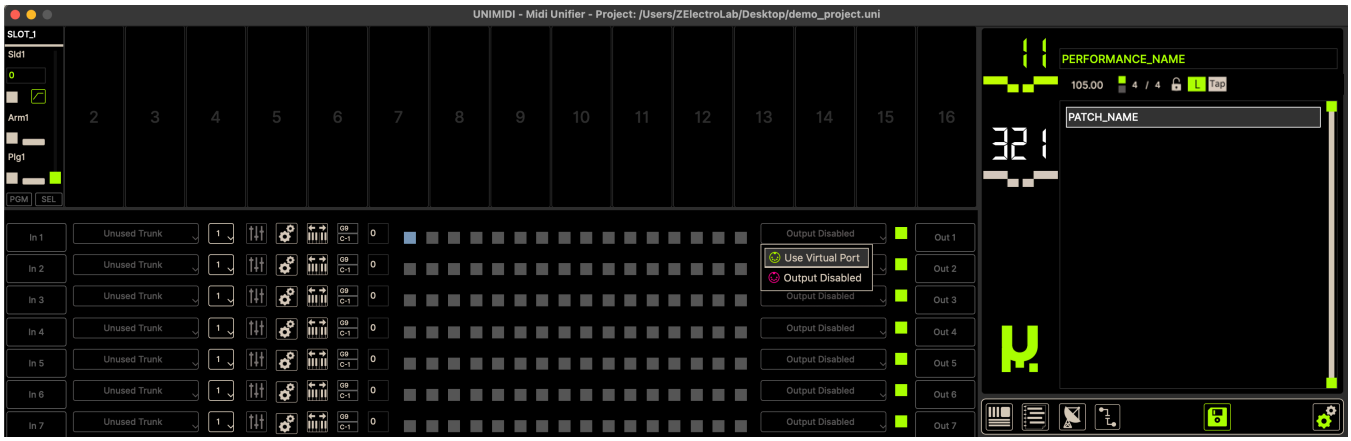
- Configuration Attention: Pay close attention to your configurations. Use the midi_monitor and the midi_meters to verify your settings. Deactivate the nodes and activate them one by one to check the values you have set.

Creating Patches in a Performance

After successfully creating a starting performance, which contains a single slot and controls one track in your DAW, the next step is to generate patches.

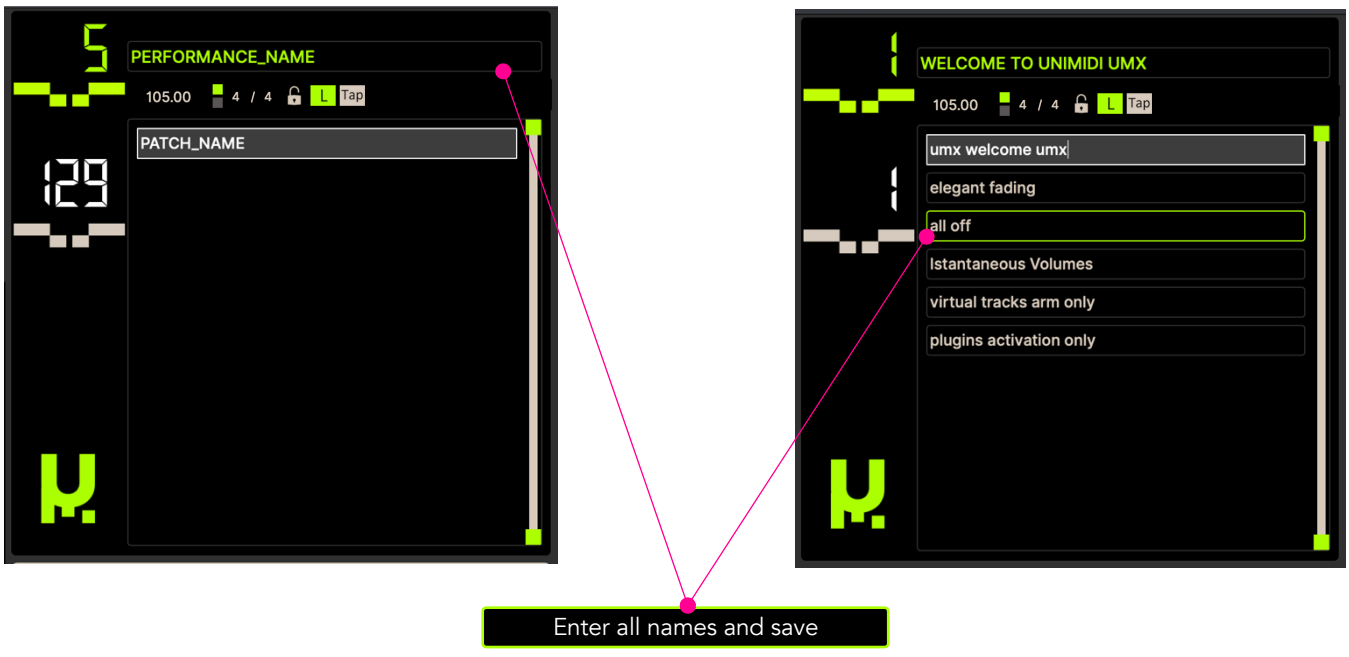
Initial Patch Generation

When you create a performance, a first patch is automatically generated:

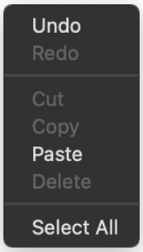


Naming performances and patches

Performances and patches can be freely named in the main windows. Just click on them and edit the name:

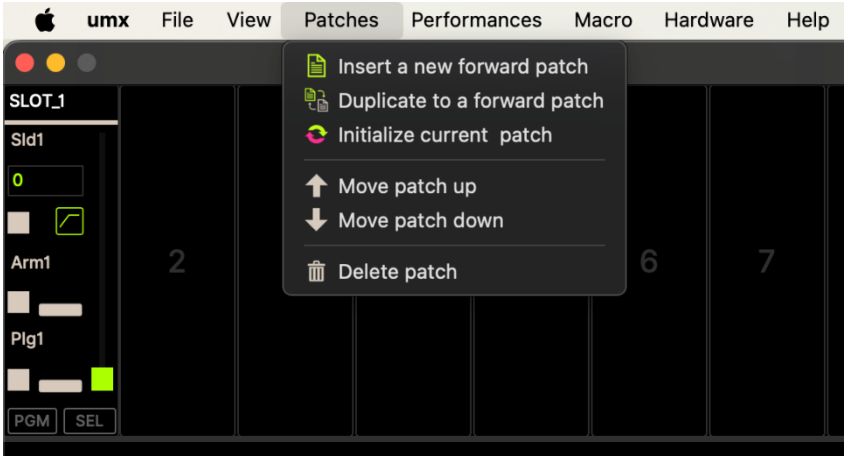


Initial Patch Strategy: A good practice is to use the first patch in a performance to silence all instruments. Ensure to set the track volume to zero (move the slider of the slot) and save the settings using the save button:



UMX text fields come with all the standard text editing functionalities that users commonly expect, (warning: Undo/Redo only work until you leave the field!).

UMX includes a dedicated menu for actions related to patches. This menu provides various options and functionalities, enabling users to:



Insert a new forward patch - Creates a new patch immediately following the current one. By default, all values of all slots (sliders and rotary controls) are set to 0, and switch-type controls are set to their low position.

Duplicate to a forward patch - Arguably one of the most useful features, this command duplicates the current patch to the next memory location while preserving all its values. The newly created patch serves as an excellent starting point for adjusting levels differently from the previous one.

Initialize current patch - Resets all values of the current patch to 0/off. This is particularly useful when you want to start afresh with a patch configuration.

Move patch up and **move patch down** -These commands allow you to move the current patch to the previous or next location, respectively. This is helpful for reorganizing the order of patches within a performance.

Delete patch Removes the current patch from the performance. This function is crucial for maintaining a streamlined and relevant set of patches.

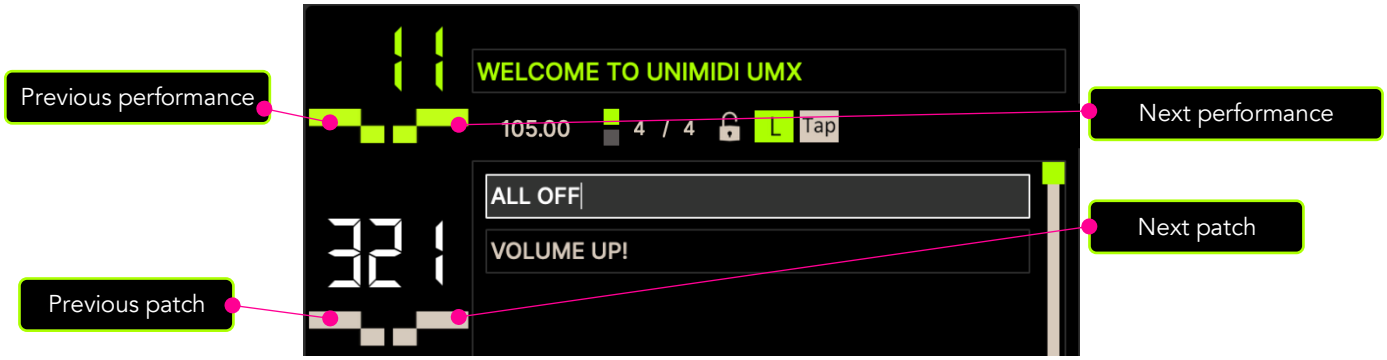
Duplicate the Current Patch: A new patch will appear following the current one.

This duplication process carries over all settings from the previous patch.

Adjusting Slot Values: Move the slider of your single slot to its maximum value and save the changes.

You will notice that each time you make changes to performance and patch names, these are immediately updated on your **umx_100** device if present. This feature ensures that your physical control is always in sync with the software configurations.

Now your performance contains two patches. As you move from one patch to the other using the keys on your **UMX** device or the corresponding keys in the software, the stored values are transmitted. This results in variations in the settings within the DAW.

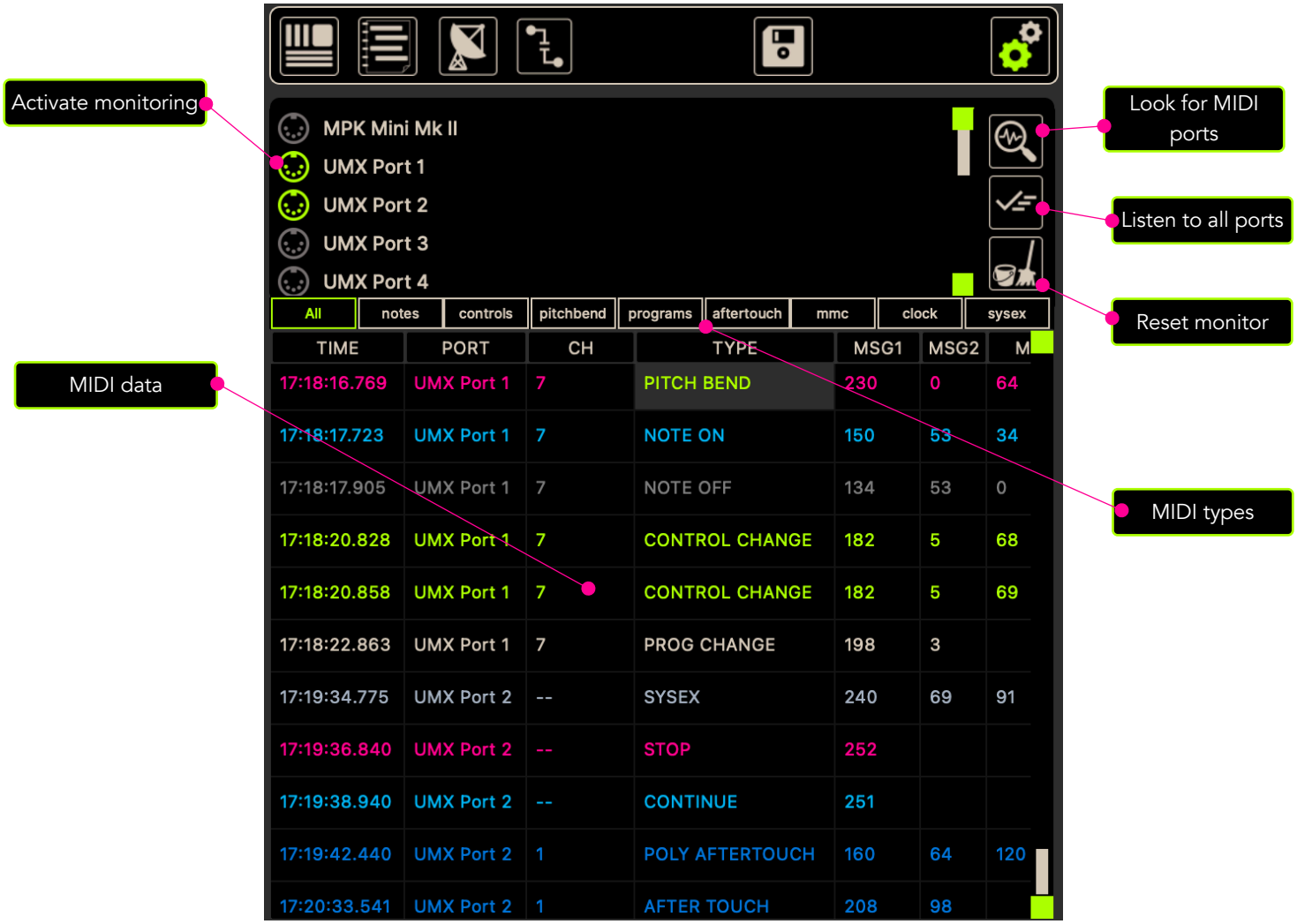


To load a patch, simply double-click on its name in the list. It will be immediately loaded, or loaded and transmitted if the system is online.

Online and offline mode: the terms online and offline are used here to indicate that MIDI transmissions are activated or deactivated, and not to refer to any activity on the Internet:

Once the *Go Online* button is pressed in UMX, the program becomes active, and all actions on slots or loading a patch will immediately generate the corresponding MIDI transmissions on the designated output ports. The offline mode is useful for freely moving between patches without impacting any controlled instrument or program. Once you are satisfied with your settings, go online to perform.

The Midi_Monitor Feature in UMX



Monitoring MIDI Transmissions

- **Capability**: The midi_monitor area in **UMX** mx is capable of displaying all MIDI transmissions across all ports detected by the operating system. Additionally, it can monitor its own virtual outputs.
- OS Compatibility:
  - **macOS® and Linux®**: On these systems, monitoring is always possible, providing comprehensive visibility of MIDI activities.
  - **Windows®**: In this environment, the ports to be analyzed must not be occupied or already connected to other software, unless a specific driver is used. This limitation is due to the way Windows® handles MIDI ports and connections.

For detailed information on handling MIDI ports in different operating systems, refer to the section dedicated to OS-specific options in the manual. This section provides insights and instructions on how to manage MIDI ports effectively, depending on the operating system you are using.

Midi types

You can select which kind of messages are displayed by the midi_monitor using the types buttons. The message list will not be modified, types displaying only change after selection.

Importance of the Midi_Monitor

The midi_monitor is a crucial tool for troubleshooting and fine-tuning your MIDI setup. It allows you to see real-time data flow, helping to identify and resolve issues with MIDI transmissions or to confirm that your MIDI devices are communicating correctly with **UMX** and your DAW.

The database_manager in UMX

The *Patch* and *Performance* menus in **UMX** provide functions for managing the software memory. For more advanced management, **UMX** features a dedicated manager, accessible through the *View* menu under *Manager window*, via the dedicated button, or by using the CTRL+M (CMD+M) shortcut:

database_manager Interface:

1. **Performance List**: The left area of the database_manager window displays a list of available performances. The loaded performance is highlighted.
2. **Patch List**: The right area shows the list of patches within the selected performance.

- **Loading Performances and Patches**: You can load a new performance or patch with a double-click. The descriptions saved for the selected performance and patch are always displayed at the bottom, even when they are not loaded.
- **Selection for Actions**: To perform an action on a performance or patch, simply select it. They do not need to be actively loaded for this.

Performance Functions

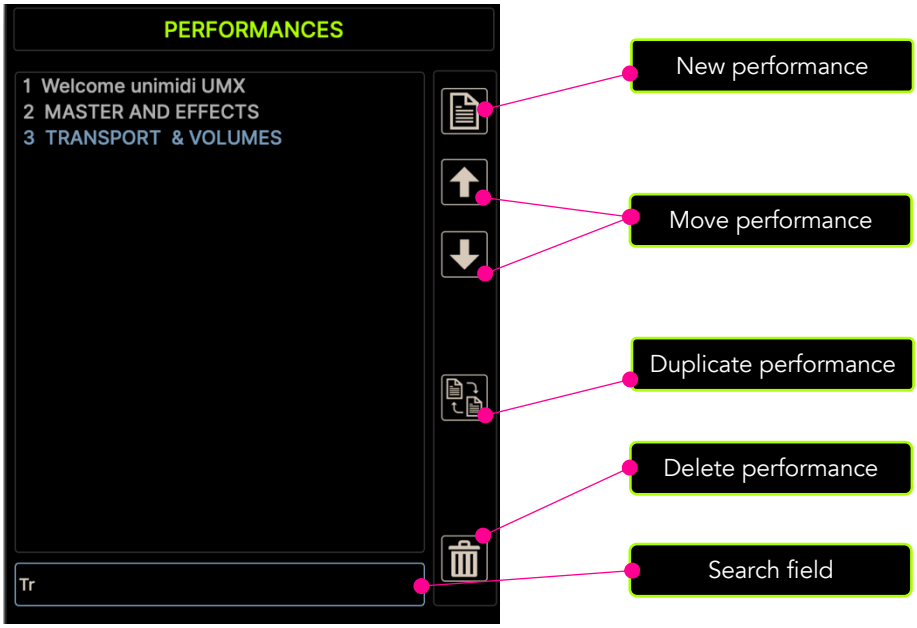
New Performance - Switches to the interface for creating a new performance.

move up and **move down -** Moves the selected performance to the previous or next memory location. Patches and performances can also be moved quickly using the drag-and-drop method.

Duplicate performance - Duplicates the performance and all its patches to the next memory location.

Delete performance - Deletes the selected performance along with all its patches.

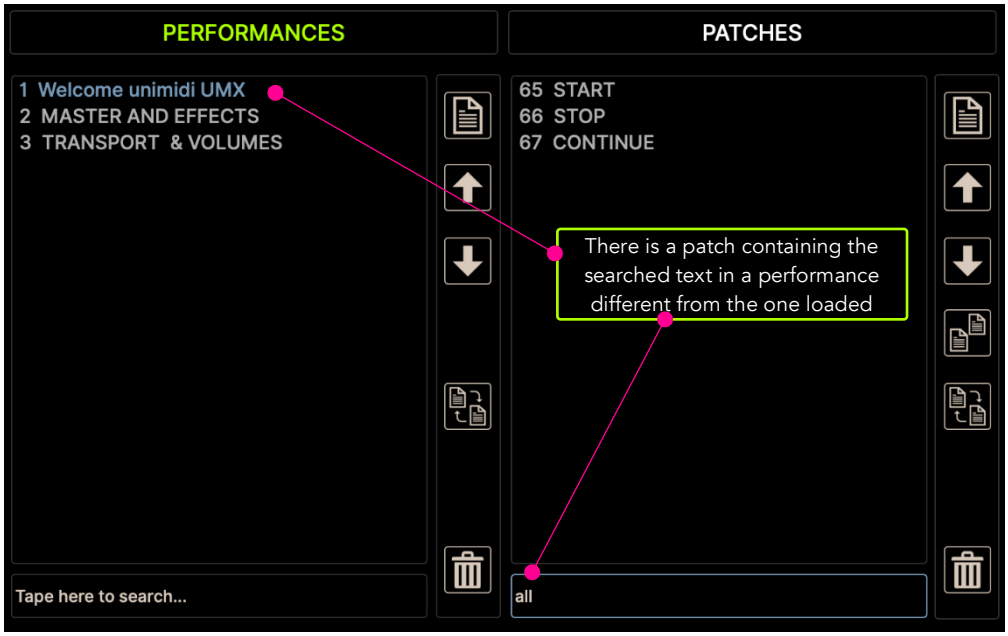
Search Field - Typing text in this field will highlight any matching names in the list.



Patch Functions and Search

- The search field also applies to patches. However, the search function works differently for patches; it looks for the text in all patches of the currently loaded performance and in all other performances. This feature allows you to find a patch contained in any performance.

This database_manager tool enhances the management and organization of performances and patches, making it easier for users to navigate, modify, and optimize **UMX** setups.



In UMX patch management, there's an additional copy function, which is accessible through a specific button:

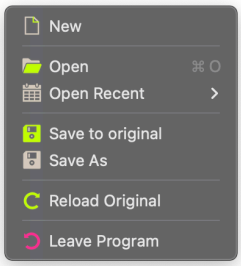
- Copy Indicator: When you press this button, it will flash to indicate that it is waiting for the copy destination to be selected.
- Copying within the Same Performance: You can select another patch within the same performance. The copied patch will replace the selected one in its current position.
- Copying to Another Performance: If you select a different performance as the destination, the copied patch will be added to the end of the list of existing patches in the target performance.

Adaptation During Copying

- Adaptation to Target Performance: When copying a patch from one performance to another, it is adapted to fit the target performance. Unnecessary slots are removed, and missing ones are added.
- Slot Matching: If the slots match between the source and target performances, all values of the patch, including slot settings, are also copied.

UMX System Menus Structure

FILE Menu



New – Creates a new project. Select the directory and name, and the file is immediately loaded.

Open – Opens an existing project. Choose the directory and file to load.

Some system parameters, for example the window size, require the program to be restarted. If you open a project that has settings different from the current one, the program will restart automatically.

Open Recent – Allows you to load the ten most recent projects.

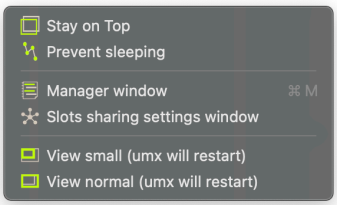
Save to original – Saves the current project. UMX opens projects by creating a temporary copy on disk. All actions and modifications are made on the working file. Modifications are saved to the original file only upon project closure. The save command saves the current state to the original file, ensuring that if your system crashes for any reason, changes will not be lost.

Save As – Saves the current project in a new directory with a new name. The project is not automatically opened; this is essentially an export of your file in its current state.

Reload Original – Reloads the original project. Discards the current working copy file and reloads the initial project. Be cautious as unsaved changes will be lost.

Leave Program – Exits the program without saving. Upon exiting, **UMX** saves the working file with all modifications to the original location. This command closes **UMX** without saving; be cautious as all changes will be lost.

VIEW Menu



Stay on Top – The UMX main window will always remain visible on top of other windows.

Prevent sleeping – UMX generates random micro-movements of the mouse pointer, preventing the computer screen from switching off or entering screen saver mode. Be aware of this feature, as it can lead to high battery consumption on laptops!

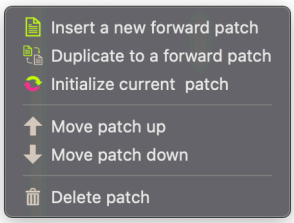
Manager window – Open the database manager window.

Slots sharing settings window – Open the sharing slots settings tool window.

View small – The app size is set to 1280x760 (WxH). The program is then restarted.

View normal – The app size is set to 1680x996 (WxH). The program is then restarted.

PATCHES Menu



insert a new forward patch - Creates a new patch in the next location, setting all slot values to 0 (both sliders and rotary controls) and switch-type controls to their low position.

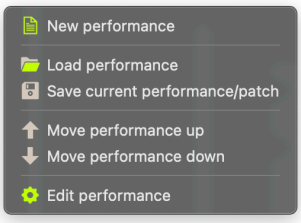
Duplicate to a forward patch - Arguably the most useful command, it duplicates the current patch to the next memory location while preserving all its values. The new patch serves as an ideal base for setting different levels from the previous one.

Initialize current patch - Resets all values of the current patch to 0/off. This is useful for starting over with a clean slate in a patch.

Move patch up e **move patch down** - Moves the current patch to the previous or next location, respectively. This is handy for reorganizing the order of patches within a performance.

Delete patch Removes the current patch from the performance.

PERFORMANCES Menu



New Performance - Create a new empty performance that is loaded immediately.

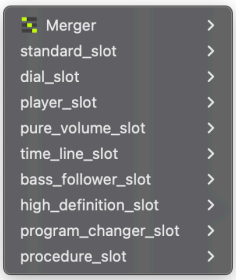
Load performance - Allows you to directly load a performance by knowing its memory location number.

Save current performance patch - Saves the current performance and patch, equivalent to the save button.

move up e **move down -** Moves the selected performance to the previous or next memory location.

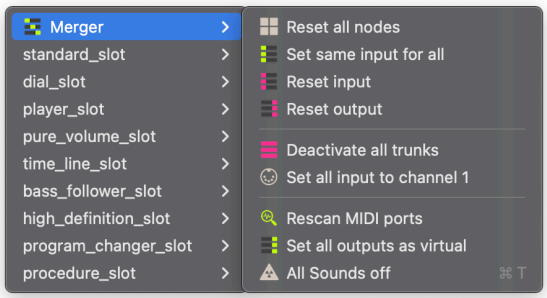
Edit performance - Opens the slots selector tool window.

MACRO Menu



The MACRO menu in UMX provides quick access to settings related to the MIDI Merger and the slots. It is divided into multiple sub-menus: the Merger sub-menu is fixed, while other sub-menus are loaded depending on the types of slots included in the performance.

- Merger - actions related to midi_merger:



Reset All Nodes - Resets all node settings of all trunks simultaneously. To reset nodes of a single trunk, right-click on any node of the concerned trunk and select the corresponding function.

Set same input for all - Uses the controller set on input 0 for all other trunks.

Reset Inputs - Sets all trunk inputs as unused.

Resets Outputs - Disables all trunk outputs ports.

Deactivate all trunks - Disables all trunk input retransmission.

Set all input to channel 1 – sets the output channel of all trunks to channel 1. It also deactivates all MPE modes.

Rescan MIDI ports - If the **UMX** encounters any issues while opening a specific port, that port will be marked as faulty and cannot be used. However, if the problem is resolved, such as a previously busy port becoming available, you can utilize this command to rescan and detect all currently available ports again.

Set All Outputs as Virtual - Opens all virtual output ports of all trunks.

All sounds off – This is a panic function. UMX immediately transmits the following on all open output ports and MIDI channels: MMC *stop*, *all notes off*, and *sustain pedal off*.

- standard_slot - actions that affect all standard slots present in the performance:

sliders_zero: Moves all standard slot sliders to 0.

sliders_fade: Changes the state of all transition activators.

sliders_active: Changes the state of all slider controller activations.

arm_low: Sets all bistable 'arm' controls to low position.

arm_high: Sets all bistable 'arm' controls to high position.

arm_active: Toggles activation state of all 'arm' controls.

plg_low: Sets all bistable 'plg' controls to low position.

plg_high: Sets all bistable 'plg' controls to high position.

plg_active: Toggles activation state of all 'arm' controls.

reset_transport: Removes all transport commands from all slots in the patch.

reset_onexit Function: Removes all exit commands from all slots in the patch.

Once a slot is loaded in the current performance a dedicated macro submenu will appear in the main macro menu. Refer to slot description for more information.

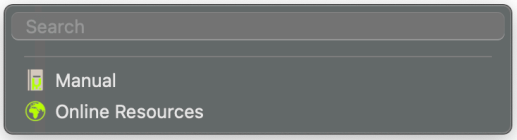
Hardware Menu

Stop Device: the device is stopped.

Start Device: this function allows you to restart the device. This action is necessary after performing a firmware update.

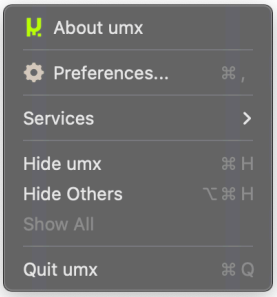
Update Firmware: The UMX software always includes the latest firmware version for Unimidi devices. If needed, you can perform a complete update of the firmware. The update function will always check for version compatibility and suggest the best choice for you.

HELP Menu



- **Search:** Type text to search for a function.
- **Manual:** Opens the instruction manual in your system PDF viewer.
- **Online Resources:** click to open the **UMX** dedicate section of Unimidi website, many resources are available.

UMX Menu



This menu only exists on macOS® systems, the relevant options are:

About UMX: Open the program info panel, which is also accessible by clicking on the Unimidi logo in the expanded interface. This panel allows you to activate your app with a license file.

Preferences: Opens the global settings panel, see *system options in UMX.*

On Windows® systems these commands are also available:

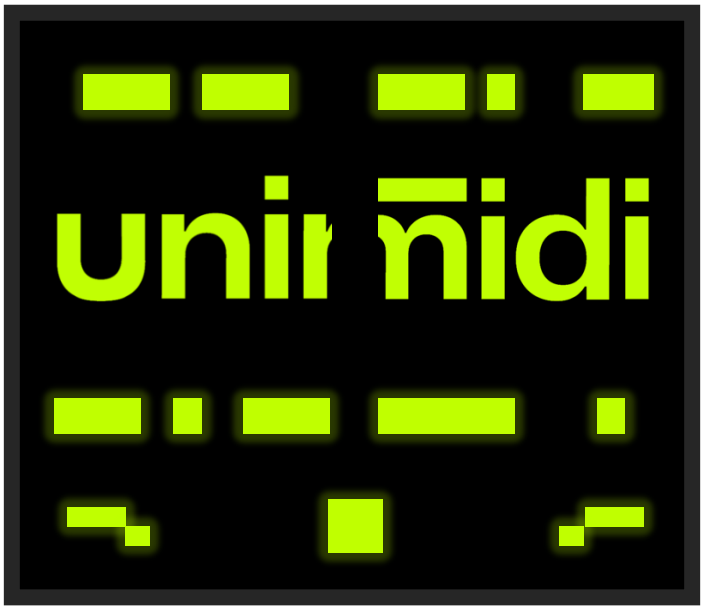
About UMX => Help menu

Options windows => View menu

UMX_100 Device Overview

The **UMX_100 remote launcher** acts as a remote control for the software.

Connect it to any USB port on your computer. Shortly after connection, you'll see a '*waiting for host*' message, indicating that it's ready for the software to be launched.



Modes of Operation

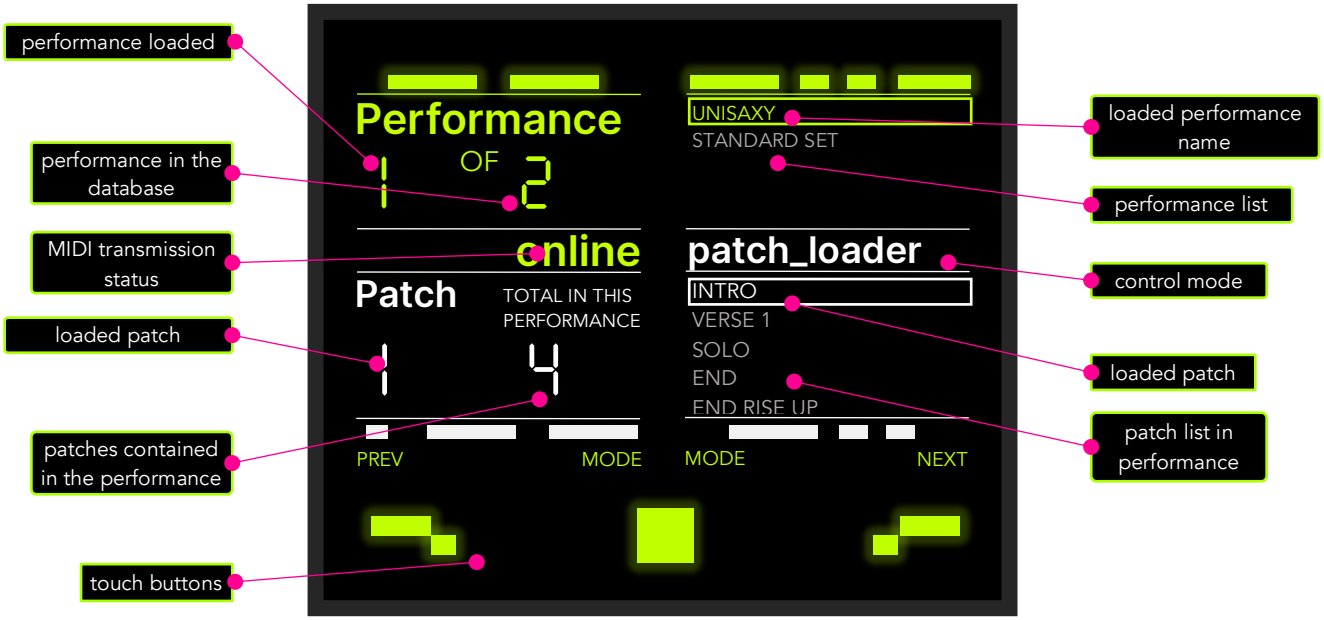
The **UMX_100** device has three operational modes, which can be cycled through by pressing the central button. The button colour changes depending on the mode:

Perf Loader (Offline[1]): Upon loading, the device starts in performance navigation mode. You can scroll through performances using the **next** and **prev** buttons. Loading a performance takes a few seconds, depending on the number of slots and MIDI ports used. The device will display '*please wait **UMX** is loading*' during this process.

Patch Loader (Offline): After loading the desired performance, switch to patch loading mode by pressing the mode button. In this mode, you can scroll through patches. While in offline mode, patches are loaded, but no MIDI transmission occurs, allowing free navigation without altering the configurations of your instruments.

Patch Loader (Online): To transmit a selected patch, press the mode button again to switch to online mode. In this mode, patches are transmitted immediately upon selection. This is the mode used during live performances. You can freely navigate through available patches for the performance, and they will be transmitted as soon as they are loaded.

[1] the terms online and offline are used here to indicate that MIDI transmissions are activated or deactivated, and not to refer to any activity on the Internet.



Device Connectivity and Inactivity

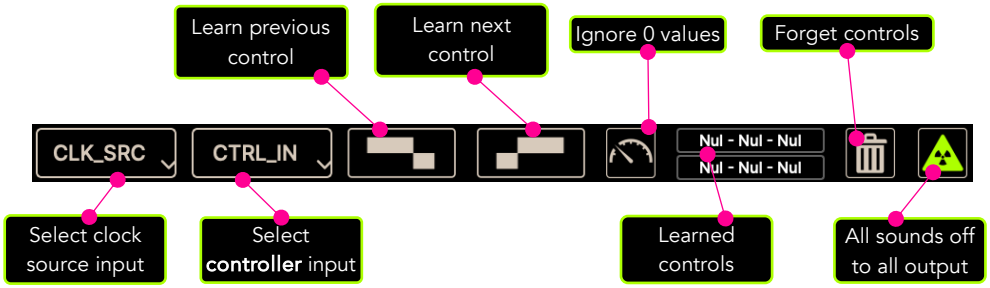
- **Reconnecting the Device**: If you disconnect and reconnect the device while the software is running, it will automatically update after a few seconds.

- **Standby Mode**: After a period of inactivity, the device enters standby mode, displaying random graphical routines. Any action in the software or touching a command on the device will exit standby mode. Pressing a button like *next* during standby will not perform the associated action; press it a second time to resume control.

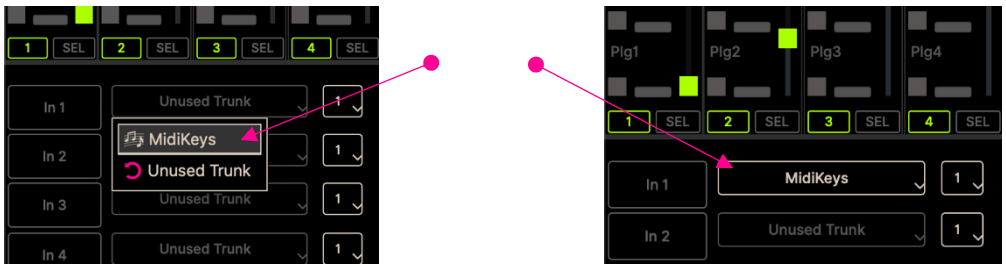
Controls UMX with a standard MIDI controller

Setting up a standard MIDI controller

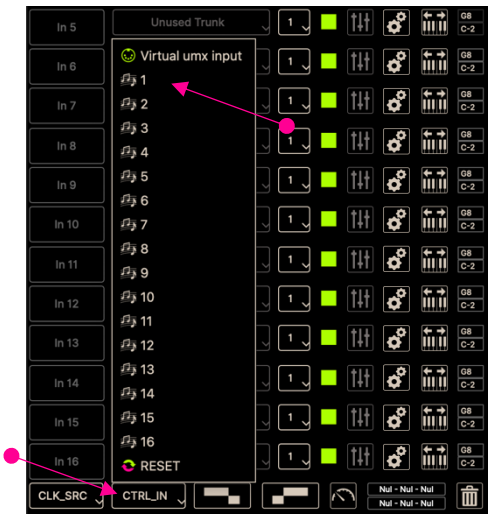
UMX_100 is the perfect companion to the **UMX** Midi Unifier software: switch patches, browse your performances and keep an eye on the essential information. Moreover, you can recall performances and patches from all MIDI controllers. In the bottom section of the midi_merger you find a dedicated area for external controller settings:



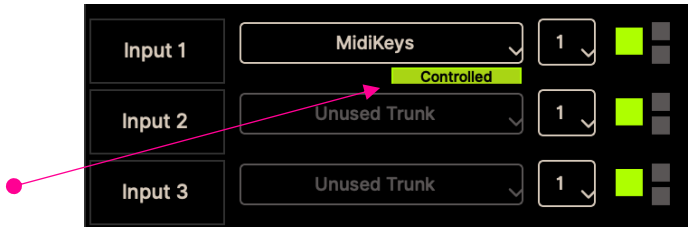
Connect your MIDI master keyboard or controller, and select is as an input within **UMX**:



Click on the *CTRL_IN* button and select the MIDI trunk you just set as the input:



The currently selected input is now being used as a controller:

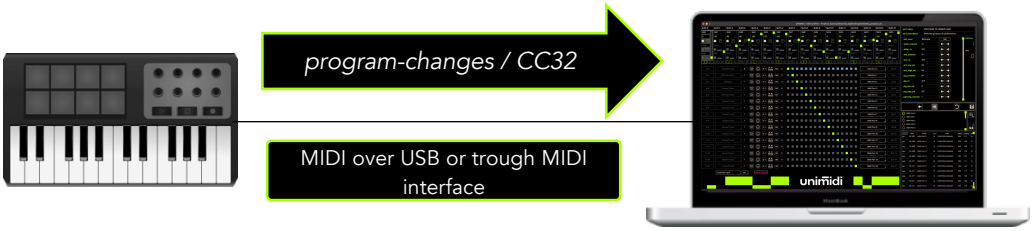


After these steps, the connected MIDI controller can recall performances by sending a standard *control change* (CC) message on *CC32*. The transmitted value will instantly recall the corresponding performance number (for example, value 0: performance 1, value 1: performance 2, and so on).

Similarly, the controller can send a standard *program change* message to recall patches within the performances. For example, program change 0 would select patch 1, program change 1 would select patch 2, and so on.

The MIDI controller allows you to jump from one performance to another and from one patch to another.

The simple interconnection diagram is as follow:



To configure controls to send *program-changes* within the range of 0-31 for directly accessing a specific patch in performances, or to send CC32 messages within the range of 0-x for loading a performance, follow these steps:

1. Ensure that your MIDI master keyboard/controller is connected and recognized by the **UMX** software.

2. Access the control settings within the dedicate editor software or interface of your MIDI master keyboard/controller.

3. Assign the desired controls (such as buttons) to send *program change* messages within the range of 0-31. Refer to the documentation of your MIDI master keyboard/controller for information on how to configure *program-changes* for specific controls.

4. Similarly, assign controls to send CC32 messages within the range of 0-x to load a performance. Again, consult the documentation of your MIDI master keyboard/controller for instructions on how to configure CC32 messages for your controls.

5. By following these steps, you should be able to set up your controls to send *program-changes* and *CC32* messages within the specified ranges, with the input configured to capture these messages within the UMX device.

6. The **UMX** input configured as controller will capture all incoming *program-changes* and *CC32* which aren't retransmitted to the output.

Please make sure to use a control on your MIDI controller that can send fixed values for *CC32*. Avoid using knob-like controls as they generate a large number of value transmissions. Instead, consider utilizing buttons, keys or switches that can send specific and fixed values for *CC32*.

In a setup that involves sequences or tracks from a DAW or sequencer, you can automate the switches on **UMX** to synchronize with your song.

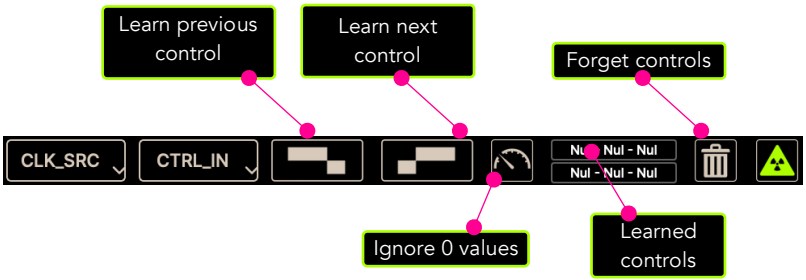
UMX can receive bank change and program change controls through any of its 16 inputs, allowing you to switch to a specific performance or patch.

To ensure correct operation, send a *CC32* message to recall a performance. Alternatively, you can manually select and load the performance through the software interface. Please note that you need to wait approximately 1-3 seconds before sending a program change message and recalling a patch, as this allows sufficient time for the performance layer to load completely.

Map a MIDI controller

UMX also provides the functionality to use two standard MIDI messages to trigger *next* and *previous* commands, allowing you to navigate to the next or previous patch number, respectively.

Once your controller is connected click on the *learning next control* or *learning previous control* button:



On your controller, press the control you want to use for switching patches forward or backward. The message will be acquired, and the learning buttons will light up, indicating that a control message has been stored.

The detected values are also displayed. You can use various types of MIDI messages, including *program-changes, control-changes,* and even *note-on* messages.

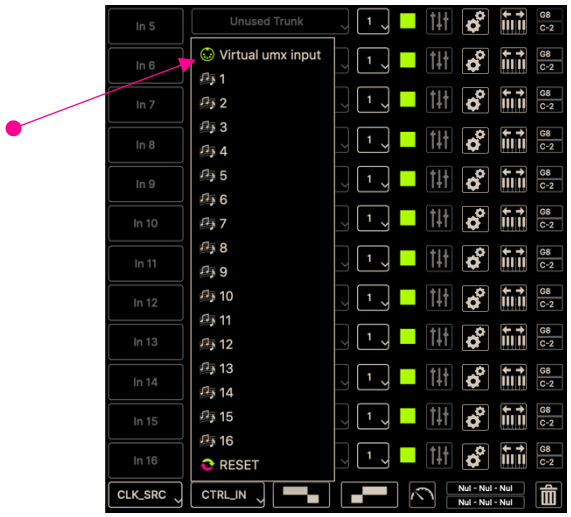
Do not use knob-like controls for mapping *next* and *previous*!

If you are using *note-on* messages to trigger patches, you might encounter a double switching on some MIDI controllers. This occur if the controller transmits a *note-on* value with a *velocity* value of zero to stop the note itself. You can easily check what your controller is transmitting using the **midi_monitor**.
In this case, you can activate the *ignore 0 values* filter for proper functioning.

If an input is used as a controller, it can still be utilized for other purposes. However, it won't allow *bank-change* and *program-change* values to transition to the trunk output as they are intercepted by the system for controlling performances and patches. If you require this functionality, you can use the MIDI controller on two separate merger inputs, allowing you to have one input dedicated to controller functions and the other for sending *control-changes* and *program-changes* messages to the trunk output.

Use the virtual input

On macOS® and Linux® systems, **UMX** has the capability to open a virtual MIDI input port. To use the virtual input, click on the *Select controller input* button and choose *Virtual UMX Input* from the list of available options. This allows you to use the virtual MIDI input for controlling UMX.



All MIDI-capable software running on your computer will detect a new MIDI port named *UMX Virtual Input* when using the virtual input feature. This virtual port can be used to receive *CC32* and *program change* messages for switching performances and patches, as previously explained.

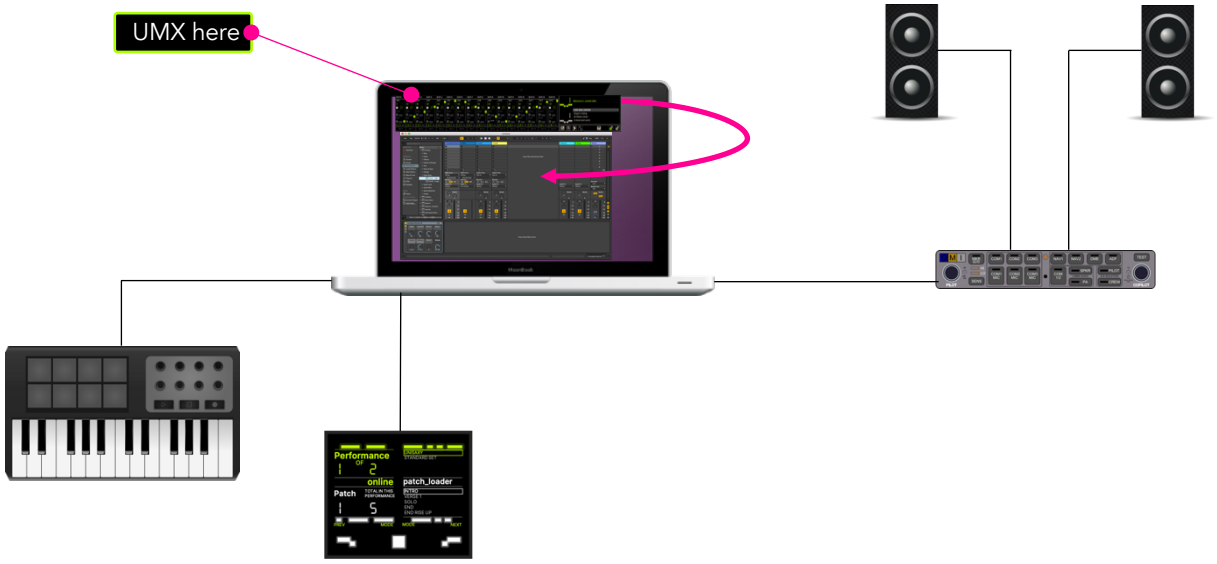
Mapped next and previous controls can also be utilized. Please note that this option is not available for Windows® users, please refer to the section on OS-specific options to ensure the correct configuration for your system.

Designing Your Setup Around UMX

Interconnecting **UMX** with Other Software

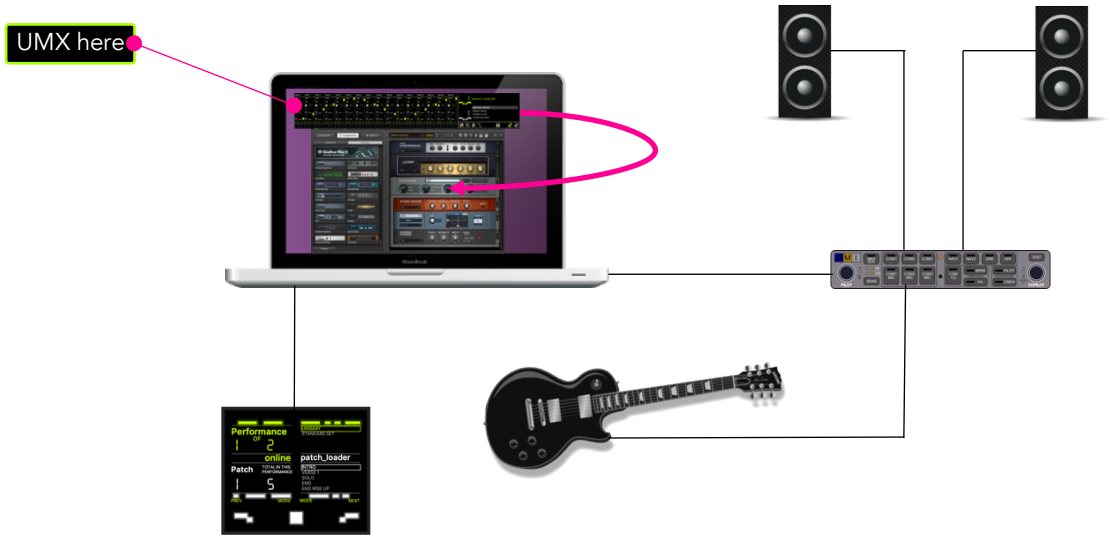
UMX, through the MIDI protocol, can modify and store the 'sound output' of your setup. The stored settings can be sequentially recalled using an optional double pedal or device buttons. Let's look at a typical connection diagram.

Basic Music Performing Setup:



- The computer runs a Digital Audio Workstation (DAW).
- A USB (or MIDI) keyboard is used for playing, recording, etc.
- The sound is output through an audio interface.
- **UMX** can be used to switch between sounds (e.g., activating/deactivating different virtual instruments) by moving from one patch to another.

Guitar Setup:



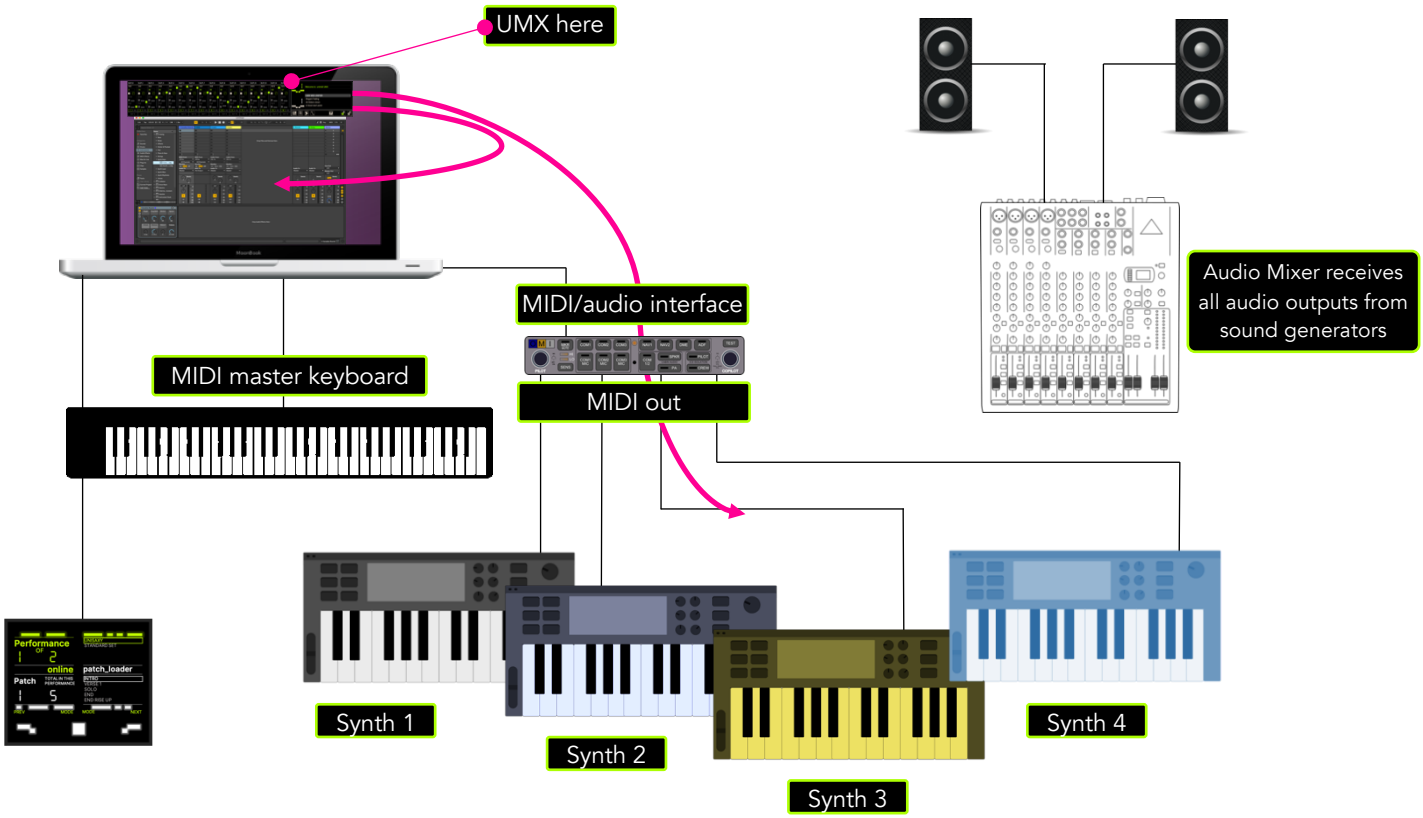
- The computer runs effects rack software.
- The guitar signal enters through a USB audio interface, is processed by virtual effects, and then outputs to amplification.
- **UMX** can be used to switch between effects (or groups of effects) or to adjust values like distortion, delay time, tube saturation, etc.
- **UMX** transition capability is crucial in effect management applications, allowing smooth changes without abrupt sound alterations.

In both scenarios, **UMX** acts as a central control unit, integrating with various elements of the music performance setup. Its ability to store and recall settings, as well as control various parameters, makes it an invaluable tool for musicians.

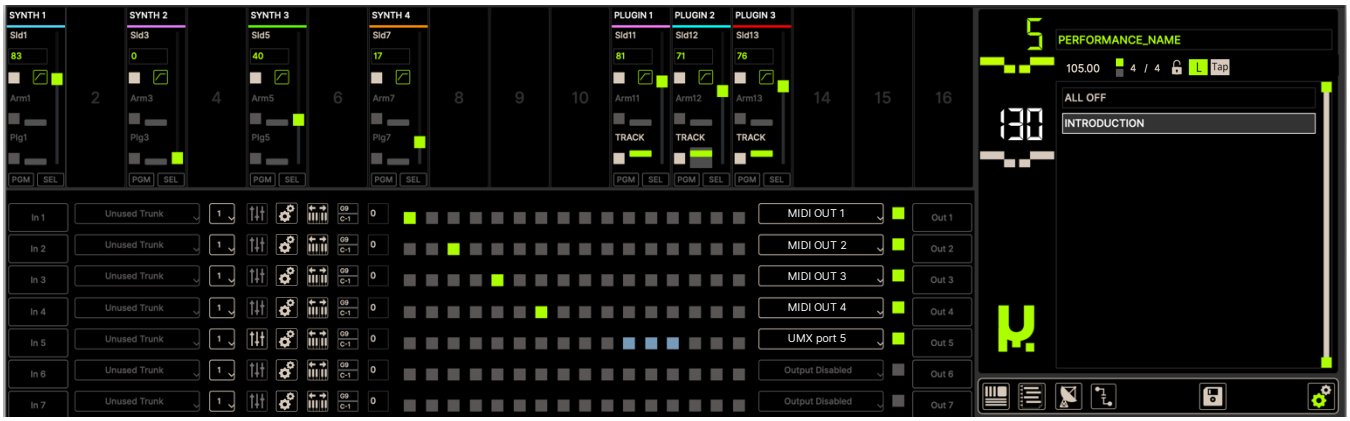
Multi-Instrument Control

UMX allows for the configuration of complex sets with multiple instruments, both physical and virtual. It can utilize any MIDI interface connected to your computer, extending beyond virtual environments to control physical devices such as synthesizers, light mixers, and smoke machines.

Let's imagine a hybrid instrumental setup, illustrating the potential complexity and versatility of **UMX**:



This could ideally be our performance setup:

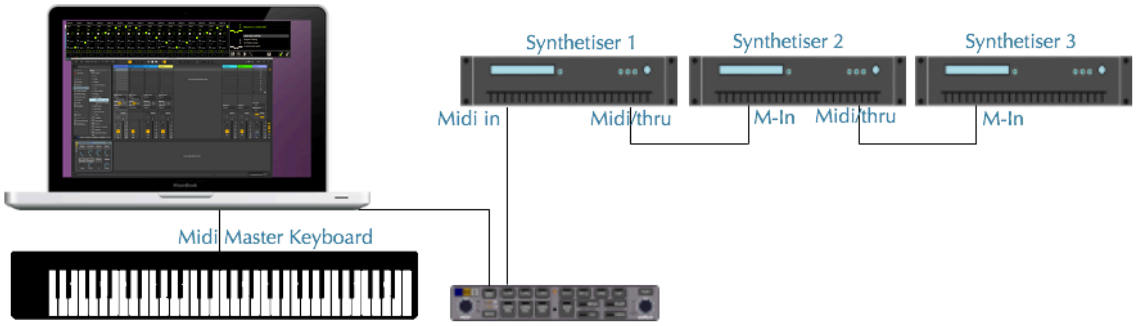


Slots for Physical Instruments:

- The first 4 slots are each directed towards a sound generator through a physical MIDI port. The corresponding node is activated in MIDI trunks 1 to 4 with *control-changes* + *program-changes* settings. For example, 'SYNTH 1' receive from *MIDI out 1*, and so forth.

Slots for Virtual Instruments in DAW:

- The other three slots control three tracks with plugins mounted in the DAW. The corresponding nodes are activated to send *control-changes* through a single virtual MIDI output used as a remote controller.
  - Each patch can load different sounds on the four synthesizers simultaneously. The same slot transmitting the *program-change* can also adjust the volume of the machine (typically *control-change* no. 7 as per MIDI protocol).
- At the same time, you can manage the volumes of the tracks in the DAW, activate and deactivate effects as desired, and more.

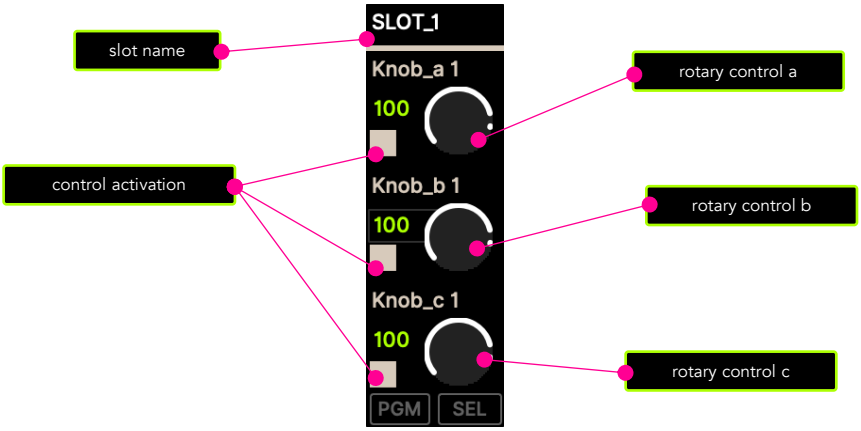


Since **umx** CC and PCh can be transmitted from only one Midi out, Midi In-thru transmission lines can be created!

Other Types of Slots

dial_slot

So far, we have primarily discussed the standard slot. The second type of slot available in **UMX 1_4_2** is the dial_slot:



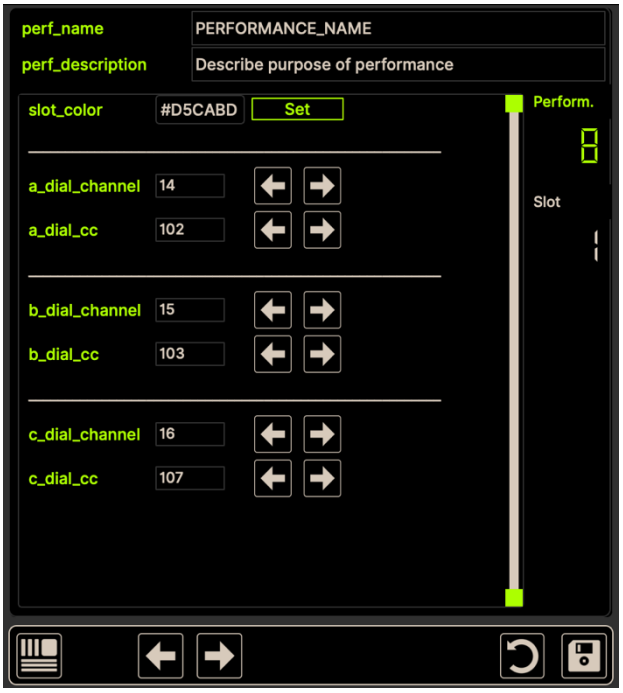
Functionality: The dial_slot can transmit 3 control-changes with continuous variation from 0 to 127. Its functionality is comparable to the slider in the standard slot. The three controls operate identically, transmitting MIDI CC values from 0 to 127 and including the feature of progressive transition from one value to another.

Applications: This slot is designed to control volumes, send audio to effects or groups, equalizers, sounds generators and effects parameters. It offers nuanced control suitable for fine-tuning audio and effects and instruments in real-time.

Performance Settings for **dial_slot**

These settings are available for each slot and are unique for each performance and common to all patches within a performance. To access these settings, hit the performance slot settings button in the control area, then select a slot using the SEL button of the desired slot, or by the right click menu available for each slot too.

Here are the possible settings:



slot_color: Choose a colour for the slot for easy identification. Press the *SET* button to select a colour for the slot.

a_dial_channel: Sets the MIDI channel for transmission of Control A.

a_dial_cc: Specifies the control-change value transmitted by Control A.

b_dial_channel: Sets the MIDI channel for transmission of Control B.

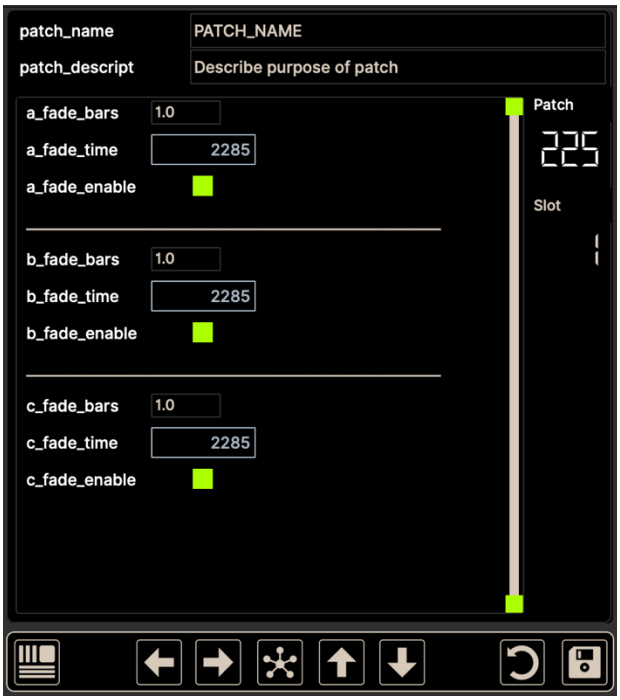
b_dial_cc: Specifies the control-change value transmitted by Control B.

c_dial_channel: Sets the MIDI channel for transmission of Control C.

c_dial_cc: Specifies the control-change value transmitted by Control C.

By adjusting these settings, users can tailor the software to their specific needs, enhancing the interactivity and efficiency of their musical workflow.

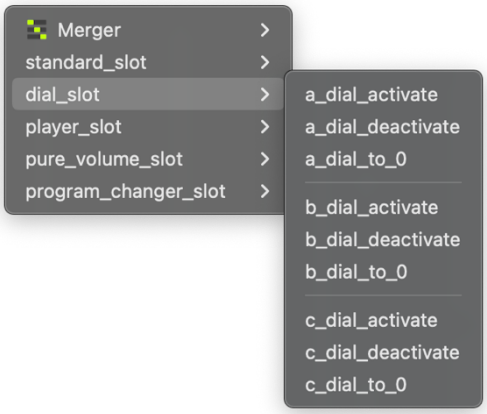
Patch Settings for **dial_slot**



In **UMX**, each patch within the dial_slot has a set of parameters that can be configured to control how the slot behaves when switching between patches. Here are the configurable parameters for each patch:
a_fade_bars: Sets the transition duration in bars from the previous value (from the previous patch) for Control a.
a_fade_time: This field is automatically updated by the program and shows the during time in ms of the settled transmission period in the **fading_bars** field.
a_fade_enable: Enables/disables the progressive transition from the previous value for Control A.
Same settings are available for control B and C.

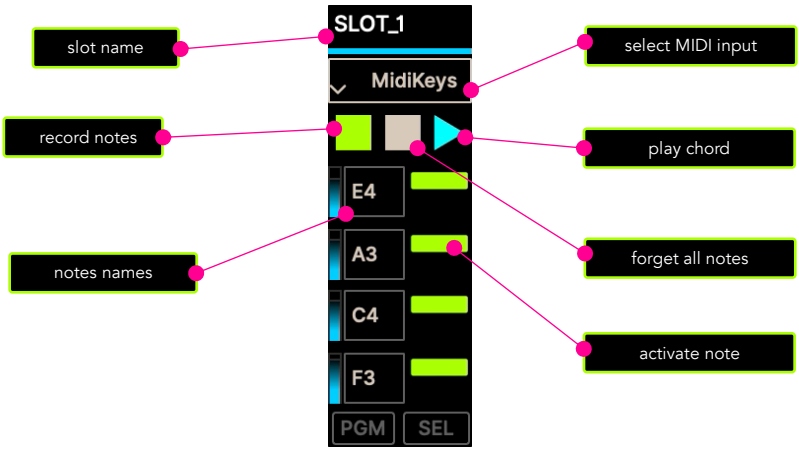
These settings are particularly useful for applications where gradual changes are necessary to avoid abrupt shifts in sound or effect parameters. By carefully configuring these parameters, you can ensure that your transitions between patches are smooth and musically cohesive.

Macro Menu actions for **dial_slot**



Dial A/B/C _activate: Activates the transmission of 'a', 'b', or 'c' controls.
Dial A/B/C _deactivate: Deactivate the transmission of 'a', 'b', or 'c' controls.
Dial A/B/C _set_to_0: Moves 'a', 'b', or 'c' controls to position 0.

player_slot



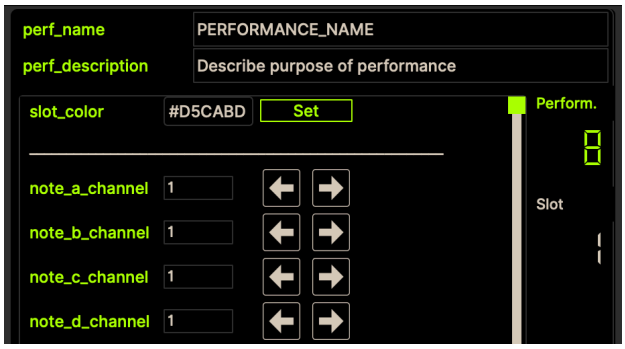
Functionality: The player_slot can record up to four notes and replay them immediately when a patch is recalled. Musicians can have 'one more hand' triggering chords through multiple MIDI outputs (which means multiple instruments or plugins).

Applications: This slot is designed to enrich your live performance with chords, or special effects dynamically triggered by musicians when the patch is loaded, it allows for complex live performances without the typical constraints of playing over backtracks that impose staying within a fixed time grid.

Slot controls for **player_slot**

- **Activate note buttons:** Chose which notes will be played by the slot when the actual patch is loaded. If deactivated notes are not played, not recorded, not reset. If the activation note button is related to an existing note, clicking it will play the note.
- **Select MIDI input button:** Chose the controller you use for recordings notes that will be reproduced by the slot.
- **Record notes button:** Once clicked, the button will flash, indicating that the recording process is active, the slot start recording the chord session waiting for up to 4 notes, once all the actives notes are detected the session ends automatically.
- **Forget notes button:** all the active notes are reset to a value of 0 (meaning the C-1 note). Clicking this button will also abort any ongoing recording session.
- **Play chord:** Hit this button to trigger the transmission of all the active notes through the MIDI ports as configured in the **midi_merger**.

Performance Settings for **player_slot**



slot_color: Press the *SET* button to choose a color for the slot.

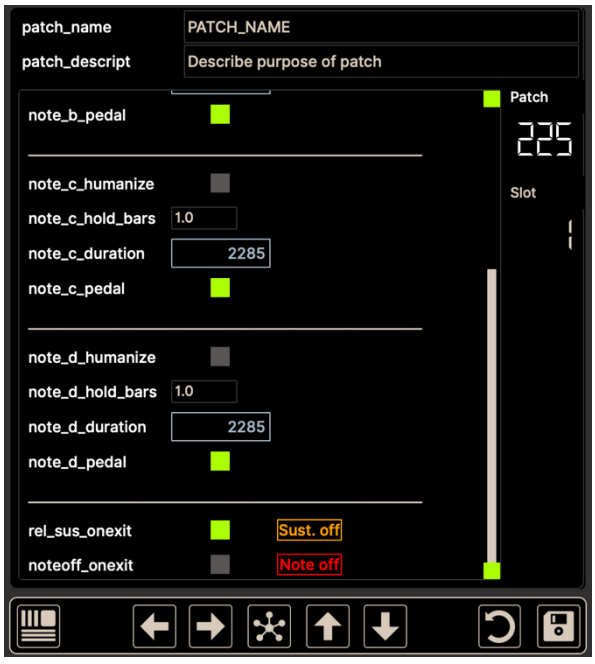
note_a_channel: Allows setting the MIDI transmission channel for the A note.

note_b_channel: Allows setting the MIDI transmission channel for the B note.

note_c_channel: Allows setting the MIDI transmission channel for the C note.

note_d_channel: Allows setting the MIDI transmission channel for the D note.

Patch Setting s for **player_slot**



note_a_humanize: If activated, the note is played with a random delay of up to 200 ms, introducing a 'human' imperfection. This effect is particularly noticeable when using percussive sounds like piano.

note_a_hold_bars: Sets the duration in bars of the triggered note. When the duration elapses, a MIDI *note-off* message is transmitted. This only works if **note_a_pedal** is set to false.

note_a_duration: This field is automatically updated by the program and shows the note holding time in ms of the settled period in the **note_a_hold_bars** field.

note_a_pedal: If true, the duration is ignored. The MIDI note is triggered, and immediately after, a MIDI sustain pedal message on (CC64 value 127) is transmitted, causing the note to be sustained indefinitely.

These parameters are available for the four notes managed by the slot.

rel_sustain_onexit: If true, a MIDI sustain pedal off (CC64 value 0) message is transmitted upon exiting the current patch. If any notes are being sustained by the pedal, they will be naturally released when a new patch is recalled.

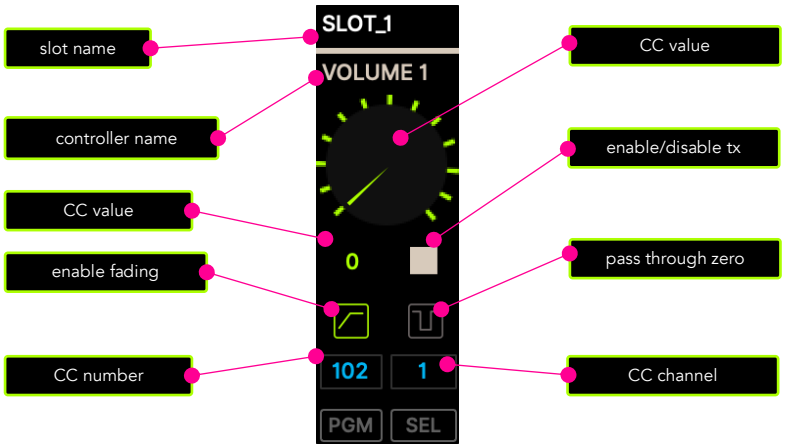
noteoff_onexit: If true, a MIDI *all notes off* message is transmitted upon exiting the current patch. This is similar to the **rel_sustain_onexit** command, but the notes are interrupted immediately rather than being smoothly released by the pedal.

Macro Menu actions for **player_slots**

all_notes_off: Switch off the four note-generator modules. the slot will transmit nothing on the current patch.

reset_all_notes: All slots forget all learned notes, which are set to a value of 0 (representing the note C-1, the lowest C note).

pure_volume_slot



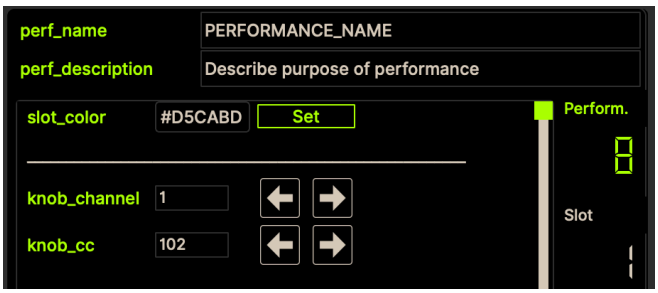
Functionality: The pure_volumeI_slot can transmit a single control change with continuous variation from 0 to 127. Its functionality is comparable to the slider in the standard slot. It offers a clear environment that manages only one CC, with the CC number and CC channel always visible.

Applications: This slot is designed to control volumes, send audio to effects or groups, equalizers, sounds generators and effects parameters. It offers nuanced control suitable for fine-tuning audio, effects and instruments in real-time.

Slot controls for **pure_volume_slot**

- **CC value:** Set a value between 0 and 127 and hit enter. The CC is immediately transmitted, and the knob position is set accordingly.
- **Enable fading:** similarly, to the standard_slot and dial_slot smooth transitions can be activated for the cc controlled by this slot.
- **Pass through zero:** if activated the pass_through_zero feature is enabled for this slot. Others slots can 'ask' this one to transmit a CC value of 0. For example, a slot which sent a program change to an instrument can ask to the pure_volume_slot to switch off the instrument volume for 200ms to avoid commutation noises. This can also be useful when switching effects.
- **Enable/Disable MIDI transmission:** hit this button to switch on or off the MIDI transmission of the CC managed by the slot when a patch is invoked.

Performance Settings for **pure_volume_slot**



slot_color: Press the *SET* button to choose a color for the slot.

knob_channel: Sets the MIDI transmission channel for the control change. This parameter is also available directly in the slot interface.

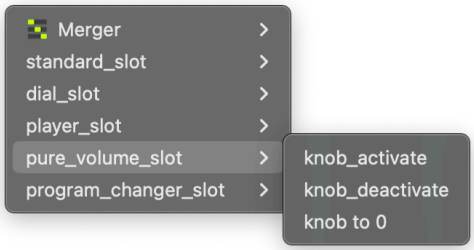
knob_cc: Sets the MIDI CC number for the control change. This parameter is also available directly in the slot interface.

Patch Settings for **pure_volume_slot**

fading_bars: This parameter indicates the amount of time (expressed in music bars) required for the sets control change value to be reached when the patch is invoked.

fading_time: This field is automatically updated by the program and shows the during time in ms of the settled transmission period in the **fading_bars** field.

Macro Menu actions for **pure_volume_slot**

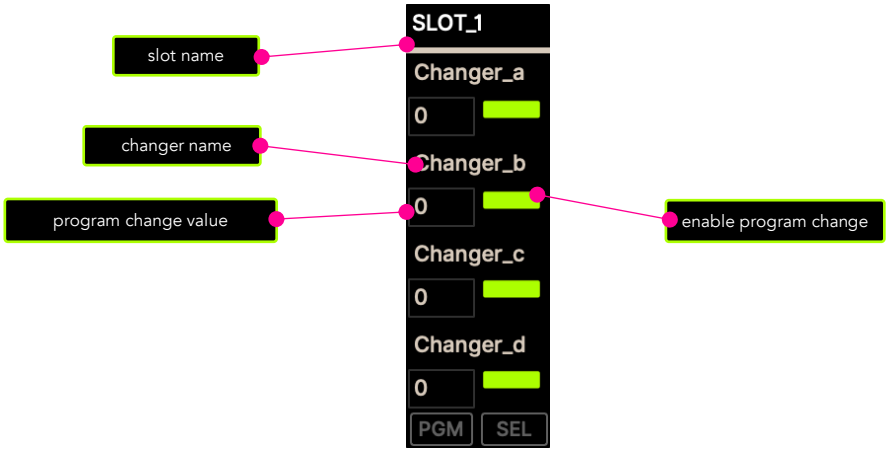


knob_activate: Activate the knob for MIDI transmission.

knob_deactivate: Deactivate the knob. The slot will transmit nothing on the current patch.

reset_all_notes: Set a value of 0 for the knob.

program_changer_slot



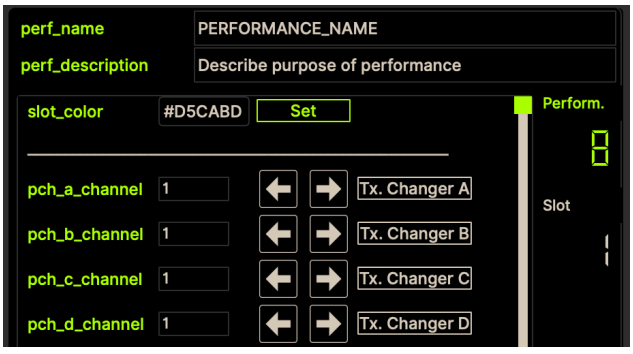
Functionality: The program_changer_slot can transmit up to four standard MIDI *program-changes* on up to four output ports. Each program change is preceded by bank changes messages MSB, LSB (CC0 and CC32).

Applications: This slot is designed to recall pre-sets on physical instruments or plugins, typically sound generators or effects.

Slot controls for **program_changer_slot**

- **Changer name:** Each program changer can be named for ease recognized.
- **Program change value:** Enter a program change value to be transmitted when a patch is recalled. MSB and LSB values must be set in the slot patch settings.
- **Enable program change:** The program change transmission can be enabled for each patch. Moving this control also generates the program change transmission for testing purposes.

Performance Settings for **program_changer_slot**

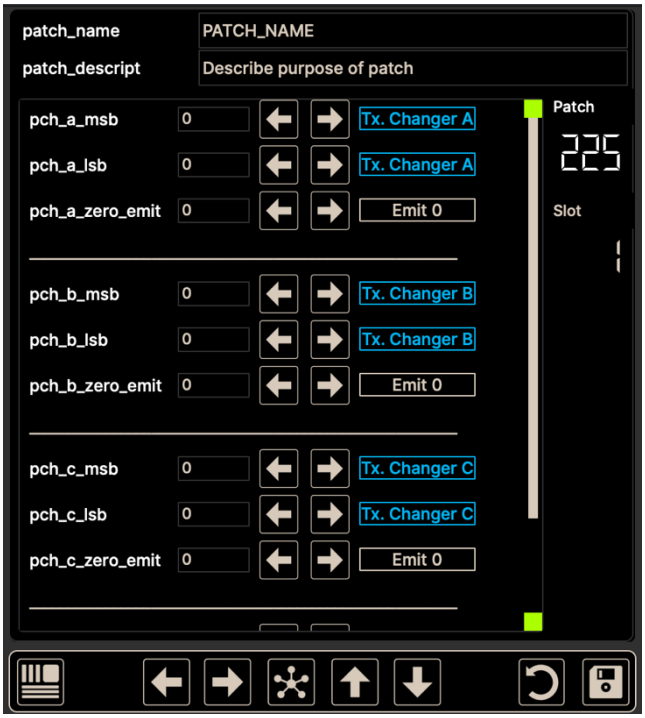


slot_color: Press the *SET* button to choose a color for the slot.

Program_a_channel: Sets the MIDI transmission channel for the program changer A.

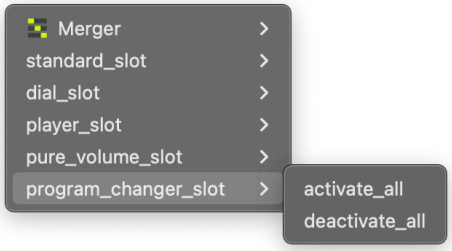
The same parameter is available for program changers B, C, and D.

Patch Settings for **program_changer_slot**



pch_a_msb: sets the bank change MSB value, the related Tx. Changer A button allow to directly test the transmission of the program change.
pch_a_lsb: sets the bank change LSB value, the related Tx. Changer A button allow to directly test the transmission of the program change.
pch_a_zero_emit: sets the slot number to send an internal **pass_trough_zero** message which is transmitted by this slot just before a program change. The destination slot must exist and must be configured to accept these kinds of messages (see **pure_volume_slot**). The Emit 0 button can be used to check these settings.
All these settings are available for program changers B, C, and D.

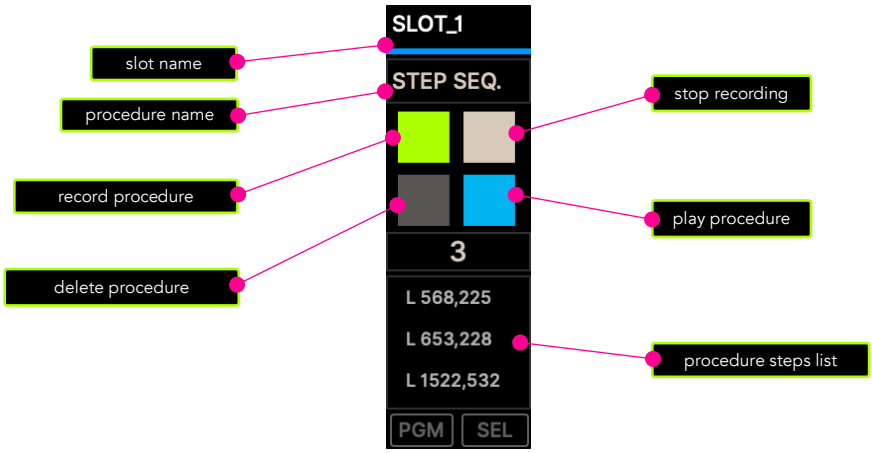
Macro Menu actions for **program_changer_slots**



activate_all: Activate the four program change modules for MIDI transmission.

Deactivate_all: Deactivate the four program change modules. the slot will transmit nothing on the current patch.

procedure _slot



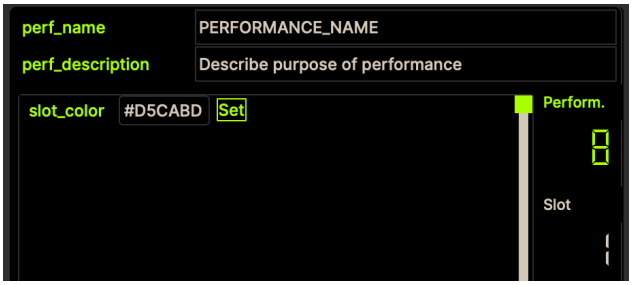
Functionality: The procedure_slot can simulate a series of mouse clicks recorded as the user executes them. The slot stores clicks at absolute screen coordinates and can recall a different procedure for each patch.

Applications: This slot can be used to automate various actions. Mouse clicks are triggered when a patch is invoked, enabling interaction with software that does not support MIDI event mapping.

Slot controls for **procedure _slot**:

- **Procedure name:** Assign a name to the stored procedure for easier recognition.
- **Record procedure:** Start recording a new procedure. Save after recording to store it in the current patch.
- **Delete procedure:** Delete the stored procedure. Resave after deletion.
- **Stop recording procedure:** End the procedure recording session.
- **Play procedure:** Replay the procedure for verification.
- **Procedure step list:** Displays all recorded steps. Double-click on a step to move the mouse to the recorded coordinates.

Performance Settings for **procedure _slot**



slot_color: Press the **SET** button to assign a color to the slot.

Patch Settings for **procedure _slot**

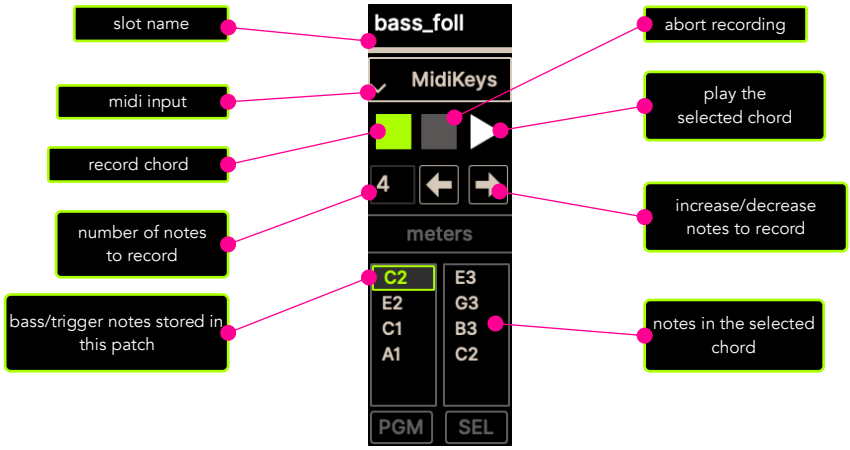
pause_time: Defines the delay between two consecutive mouse clicks in the procedure.

Macro Menu actions for **procedure _slot**

Forget_learned: Deletes the recorded and stored procedure.

To control pointer actions and clicks, the procedure_slot requires computer control privileges. Typically, your system will request these permissions the first time the slot attempts to move the pointer or generate clicks. If you deny authorization, the slot will not function. You will need to manually grant the necessary permissions in your system settings.

bass_follower_slot

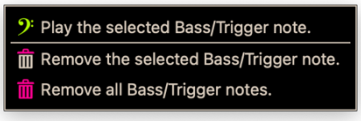


Functionality: The bass_follower_slot plays previously recorded chords when a specific incoming note (bass/trigger note) is detected on the input MIDI port.

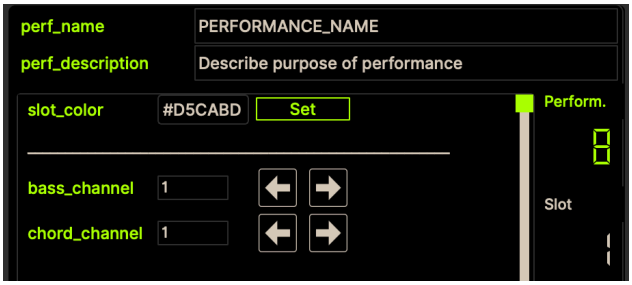
Applications: This slot is designed to generate complex harmonies with up to 12 notes per chord. It includes multiple humanization options and can be used to trigger bass-related 'left-hand' chords or arpeggiator textures.

Slot controls for **bass_follower_slot**

- **Select MIDI input button:** Chose the controller you use for recordings notes that will be reproduced by the slot.
- **Record chord button:** the slot start recording chord session, once clicked the button will flash, indicating that the recording process is active. The slot will wait and record up to 12 notes (before starting recording you have to set up the number of notes in the current chord in the dedicate field). When the settled number of notes is reached the recording session ends automatically.
- **Abort recording button:** Ends the current recording session without saving any data.
- **Play chord:** Triggers the transmission of the currently selected chord. Notes are sent according to the **midi_merger** nodes configuration.
- **Bass/trigger list:** Displays all bass notes linked to recorded chords. When a bass note is received from the selected MIDI port, its name is highlighted, and the corresponding chord notes appear in the chord notes list.
  - Click a bass note to select it.
  - Play the corresponding chord by pressing the **Play chord** button or by double-clicking the note.
  - A right-click menu provides additional options.



Performance Settings for **bass_follower_slot**



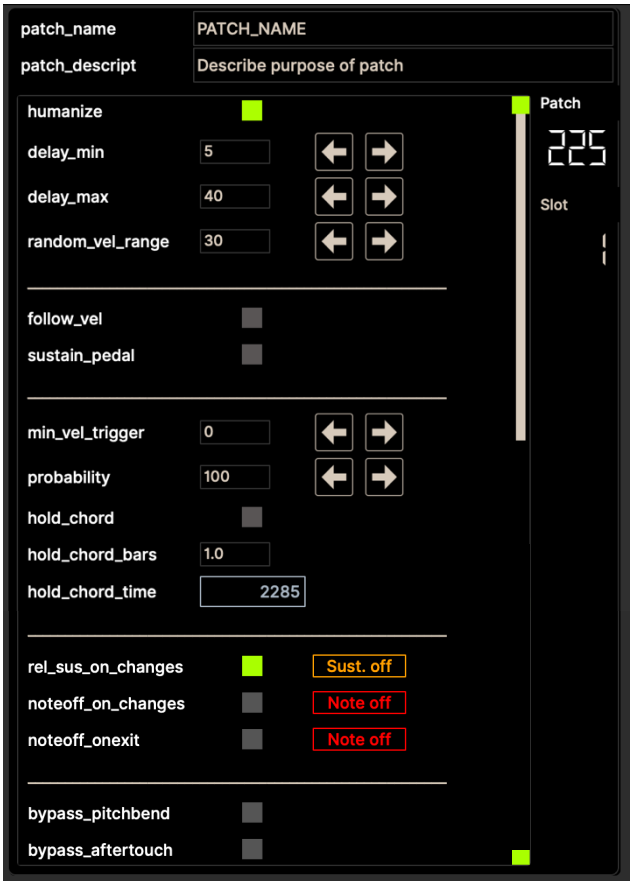
slot_color: Press the **SET** button to assign a color to the slot.

Bass_channel: the slot separates outcoming bass notes and chord notes that can be resent on different MIDI output and channels. Use this field to define the bass MIDI channel.

chord_channel: Functions identically to **bass_channel**. Use this field to define chord MIDI channel.

The bass_follower slot features full MIDI implementation. It allows all messages from its input port to pass through and reach the trunk's MIDI output, provided the corresponding node is active. This makes it possible to use the slot as a MIDI injector, effectively merging two sources into a single output.

Patch Settings for **bass_follower_slot**



humanize: When enabled, chords in this patch are played with a slight random delay and velocity variation, as defined by the **delay_min**, **delay_max**, and **random_velocity_range** parameters.

follow_vel: Chord notes inherit their velocity from the incoming bass/trigger note intensity. If humanize is enabled, additional randomization is applied. Combining both options enhances expressiveness, making the playback feel more natural and dynamic, human-like feel.

sustain_pedal: When enabled, a sustain pedal message (CC64) is sent immediately after the chord notes are played.

min_vel_trigger: Defines the minimum velocity required for a bass/trigger note to activate its assigned chord. Notes played below this threshold will not trigger a chord.

probability: Controls the likelihood of the chord being triggered when the corresponding bass note is detected. At 100%, the chord always plays. Lower values introduce randomness, making performances feel less predictable and more human-like.

hold_chord: When enabled, the triggered chord is sustained for a duration set by the **hold_chord_bars** parameter. During this period, the same chord cannot be retriggered.

release_sus_on_changes: When enabled, a sustain pedal release (CC64) message is sent before playing a new chord, preventing overlapping sustained notes.

noteoff_on_changes: When enabled, an *all notes off* (CC123) message is sent before playing a new chord.

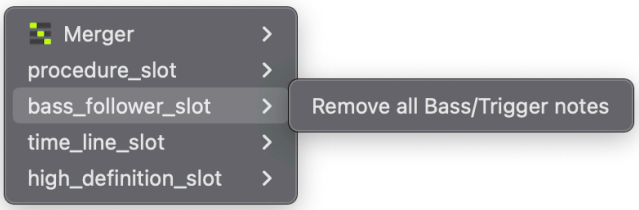
noteoff_onexit: Unlike **release_sus_on_changes** and **noteoff_on_change**, which function within a patch, **noteoff_onexit** sends an *all notes off* (CC123) message when switching to a new patch.

bypass_aftertouch: When enabled, incoming aftertouch messages from the MIDI input are passed through to the bassline output but do not affect triggered chords.

bypass_pitchbends: Functions the same way as **bypass_aftertouch**, ensuring that incoming pitch bend messages only influence the bassline, leaving the chords unchanged.

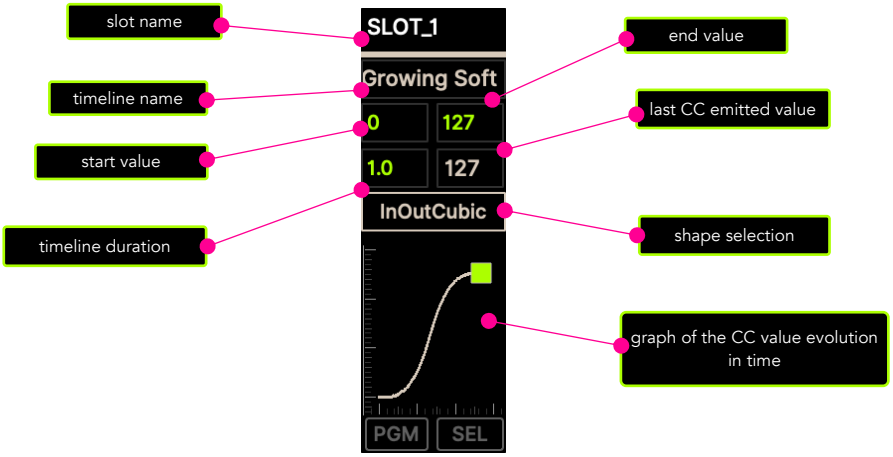
These settings apply to all chords within the current patch.

Macro Menu actions for all **bass_follower_slots**



Remove all Bass/Trigger notes: Erases all recorded bass/trigger notes along with their corresponding chords, resetting the slot to an empty state.

timeline_slot



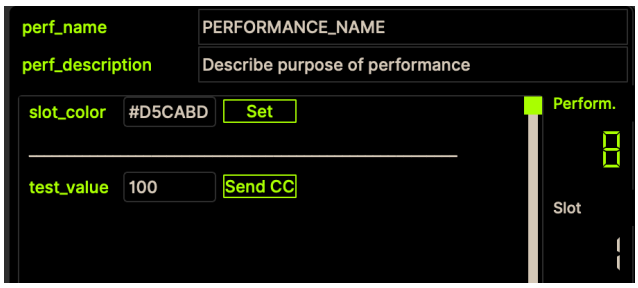
Functionality: The timeline_slot generates MIDI CC transitions, smoothly moving from a start value to an end value over a configurable time span (measured in musical bars). The transition curve can be selected from various waveform shapes.

Applications: This slot is ideal for automating volume adjustments, routing audio to effects or groups, modifying equalizers, shaping sound generators, and tweaking effect parameters. By using non-linear waveform shapes, the transition can be tailored to achieve musically natural and expressive results.

Slot controls for **timeline_slot**

- **Start value:** Defines the initial MIDI CC value at the start of the transition.
- **End value:** Sets the final MIDI CC value at the end of the transition.
- **Timeline duration:** Determines the length of the transition, measured in musical bars.
- **Shape selection:** Opens the waveform selection panel to customize the transition curve.
- **Graph:** Provides a visual representation of how the MIDI CC value changes over time during the transition.

Performance Settings for **timeline_slot**



slot_color: Click the **SET** button to assign a color to the slot for easy identification.

test_value: Allows the slot to send a test MIDI CC value (range: 0 to 127). Press the **Send CC** button to transmit this value to all configured MIDI outputs. (Ensure that nodes and MIDI outputs are properly set up in the **midi_merger**.) This feature is useful for mapping the slot MIDI CC to a specific destination.

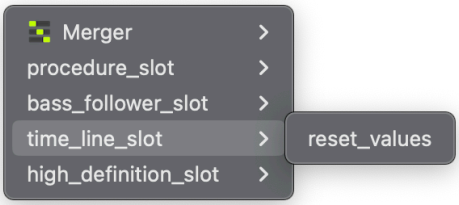
Patch Settings for **timeline_slot**

line_channel: Specifies the MIDI channel on which CC values are transmitted.

line_cc: Defines the MIDI CC number assigned to this line.

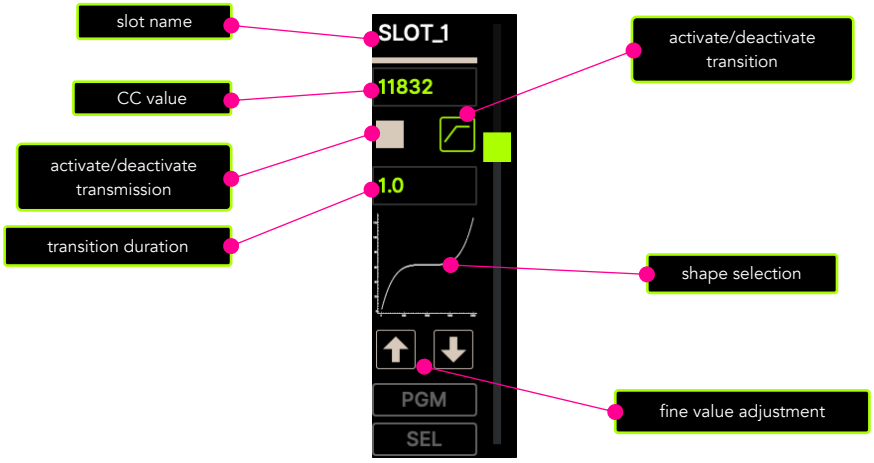
fading_bars: Sets the duration (in bars) for CC value fading, controlling how smoothly transitions occur over time.

Macro Menu actions for all **timeline_slots**



Reset_values: Resets the **start**, **end** and **duration** to their default settings.

high_definition_slot



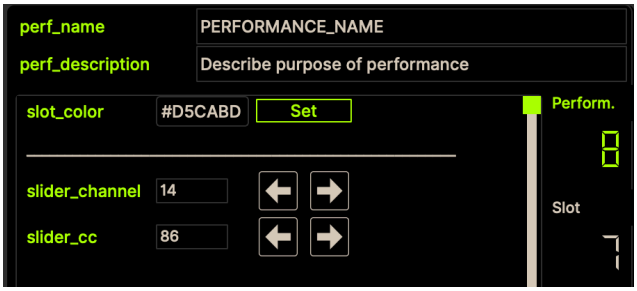
Functionality: The **high_definition_slot** transmits high-resolution **14bit NRPN 16384-step** MIDI CC values, allowing ultra-precise parameter control. It supports smooth transitions with adjustable progression time (measured in musical bars) and offers a selection of transition waveforms for customized shaping.

Applications: This slot is ideal for controlling parameters that demand fine precision, such as **tempo**, **frequencies**, and **volume**, as well as routing audio to effects or groups, adjusting equalizers, and modulating sound generators or effect parameters. With non-linear waveform shaping, it ensures musically natural and expressive transitions.

Slot controls for **high_definition_slot**

- **CC value:** Enter the desired MIDI CC value and press **Enter** to confirm.
- **Activate/deactivate transmission:** When enabled, the CC value (with or without transition) is sent immediately when the patch is loaded.
- **Transition duration:** Defines the length of the transition in musical bars.
- **Activate/deactivate transition:** When disabled, only the final CC value is transmitted, skipping the transition from the previous value.
- **Shape selection:** Opens the waveform selection panel to define the transition curve.
- **Fine value adjustment:** Due to the high CC resolution (**16,384 steps**), the slider does not provide precise tuning. Use these buttons for fine-grained adjustments.

Performance Settings for **high_definition_slot**



slot_color: Press the SET button to assign a color to the slot for easy identification.

slider_channel: Defines the MIDI channel used by the slider during the current performance.

slider_cc: Assigns the MIDI CC number to the slider. High-resolution MIDI transmission requires a dual CC system (CC_1 = MSB, CC_2 = LSB), meaning the CC number must be within the 0 – 94 range.

Patch Settings for **high_definition_slot**

fading_bars: Defines the duration (in bars) over which the CC value fades smoothly from its initial to final value.

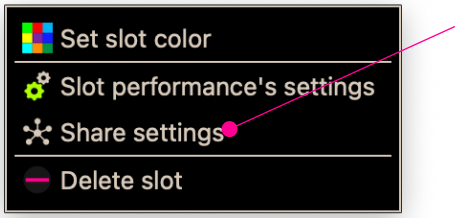
Macro Menu actions for **high_definition_slots**

reset_values: Resets value and duration to their default settings.

Slot sharing settings tool

In most cases, building a performance is a process that evolves alongside the artistic work. During this process, we need to add, delete, or move slots. Additionally, since our slots will control 'musical' parameters, it is not uncommon to need to modify the set values, adjust volumes, or alter the depth of effects, etc. Often, we will need to modify a parameter across multiple patches. If a performance contains many patches, it can be tedious to modify these one by one.

UMX includes a slot sharing settings panel that has been designed to simplify this task. To use it, right-click on a slot in your performance and choose the option *'Share Settings'*:



The panel can also be recalled by the *View* menu:

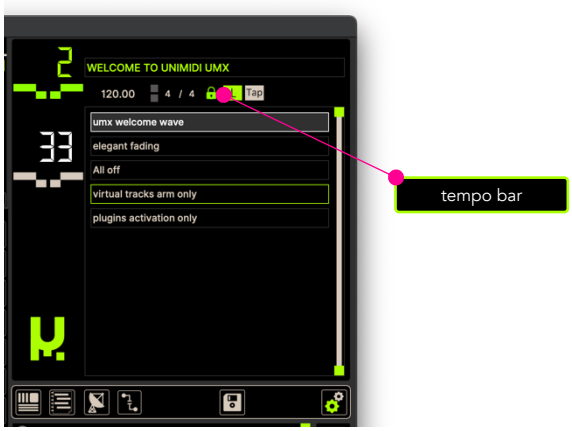
The panel immediately shows the selected slot with its settings for the current patch. All the available settings are displayed in the parameters list, and all the existing patches are visible on the right side of the panel.

We can switch to another slot by selecting it in the performance area or by using the panel arrows. We can also switch to another patch by double-clicking on its name.

For all the slots, we can select one or more parameters and then choose which patches we want to copy these settings to. Hit the save command to perform a partial copy of your slot to all the selected patches.

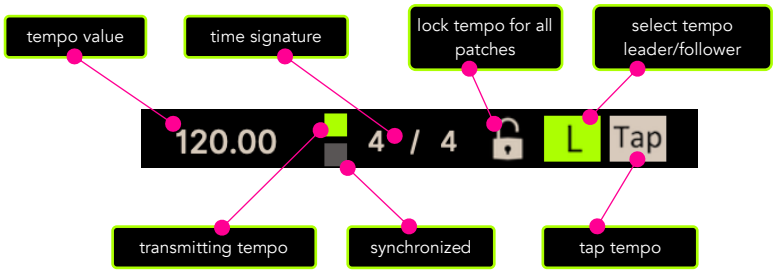
Tempo management in UMX

Starting from version 1_4 UMX includes a full tempo management system, the tempo bar is always displayed in the **navigation_area**:



UMX now supports **MIDI CLOCK** output on all MIDI ports, transmitting continuous tempo ticks at a resolution of **24 ticks per beat**, ensuring synchronization with external devices.

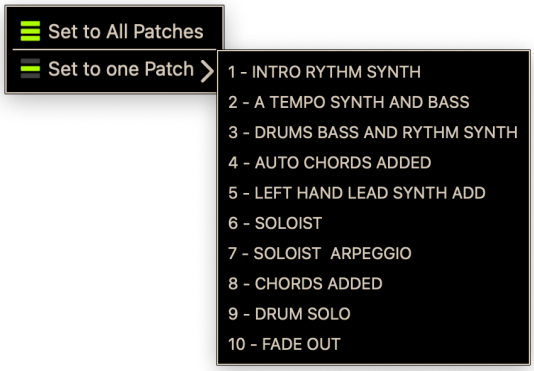
Controls for the tempo bar are:



- **Tempo value:** Defines the tempo in **beats per minute (BPM)**, with a valid range of **40 – 400 BPM**. Floating-point values are supported for precise adjustments. Enter the desired value and press **Enter** to apply.

- **Time signature:** The numerator (**beats per measure**) can range from **1 to 99**, while the denominator (**measure division**) can be set to **1, 2, 4, 8,** or **16**. Enter the desired values and press **Enter** to confirm.

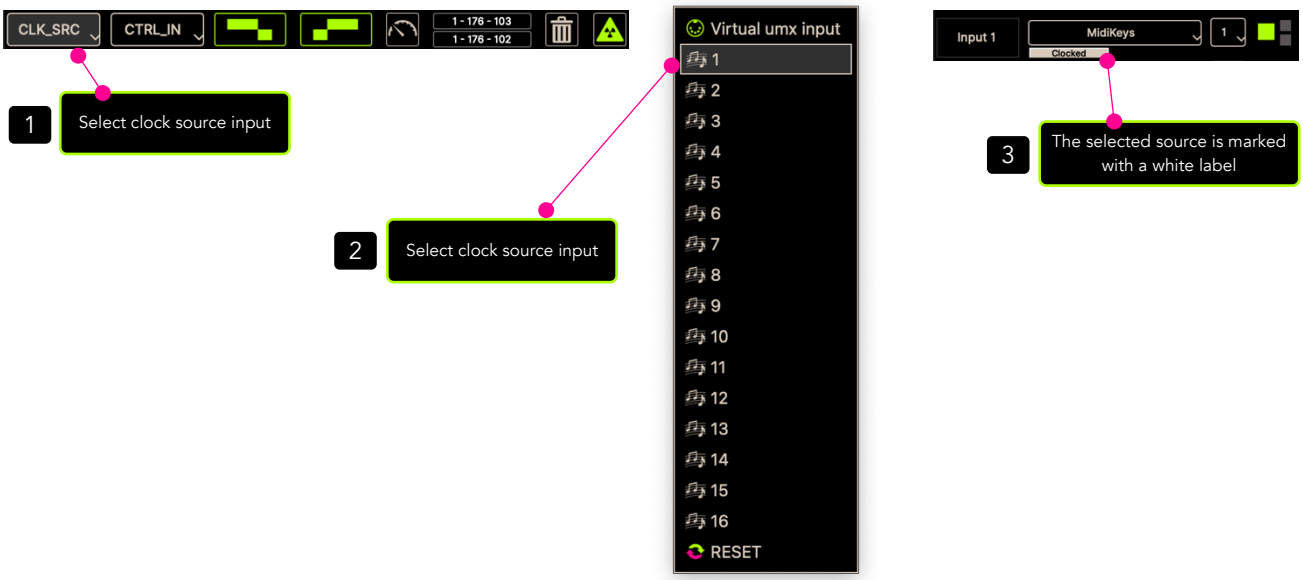
- **Lock tempo:** When enabled, all patches within the current performance use the same tempo and time signature. Any changes will apply globally. When disabled, each patch can have independent tempo and time signature settings. In this mode, **right-clicking** on the **tempo** or **time signature** fields opens a context menu with additional options.



  - **Set to All Patches:** Applies the current tempo and time signature to all patches within the active performance.
  - **Set to one Patch:** Assigns the current tempo and time signature to a specific patch selected from the submenu.

- **Select tempo leader/follower:** UMX can function as a **MIDI clock master** (emitter) or a **follower**, synchronizing with an external hardware or software clock source. When acting as the **tempo master**, UMX sends MIDI clock signals to all configured MIDI clock outputs.

When set as a **tempo follower**, the external MIDI clock source can be selected from the **midi_merger** controls area (bottom-left corner).

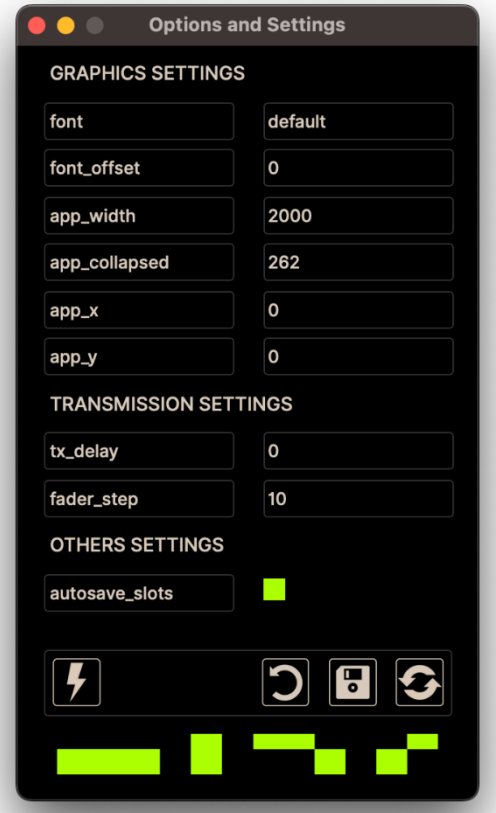


On macOS® and Linux®, UMX includes a **virtual MIDI input**, allowing synchronization with other software clock sources.

- **Tap tempo:** Click multiple times to set a new tempo based on the timing of your clicks. This function can also be assigned to an external MIDI controller using the right-click menu.

System options in UMX

The System Options window in **UMX** is accessible through the *Open options* command in the *View* menu on Windows systems and via the *Preferences* command in the *UMX* menu on macOS®. This



window contains system preferences for the current project:

1. Font
   - **font**: Choose a font from your system to be used for all text fields in the program. Leave as 'default' for the original font.

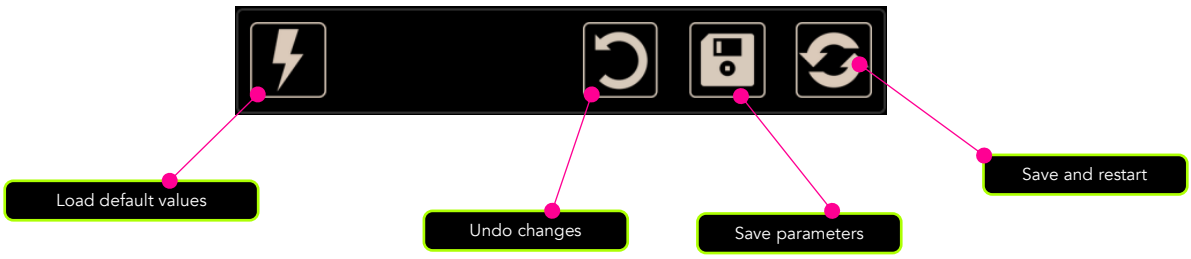
- **font_offset**: Enter a numerical value from -5 to +5 to modify the size of all text fields in the program. This is useful for custom fonts or improving readability.

2. Application Size and Position
   - **app_width**: **UMX** automatically adapts the window size to your screen. For specific needs, you can set a custom width in pixels. If the set size exceeds the screen, **UMX** will automatically adjust.
   - **app_collapsed**: Set the height of the interface. Like width, you can set a custom height, affecting the extended interface height.
   - **app_x** or **app_y**: For screens with unusual ratios (32/9, 24/9, etc.), you can position the application at a specific screen region upon opening. Enter the offset value from the left (X) and top (Y) borders in pixels.

3. MIDI Transmission Settings
   - **tx_delay**: **UMX** generates a massive transmission of MIDI data when loading each patch. Some receivers, like DAWs, can't handle simultaneous reception of numerous control-changes. This setting allows inserting a delay between transmissions of *control-changes*, measured in milliseconds. Experiment with small values and increase until you find perfect functionality.
4. Autosave for Slots
   - **autosave_slots:** When using the setting pages for slots (both performance and patch settings), remember to save each slot for changes to take effect. Enabling this parameter will make the program automatically save settings when switching from one slot to another. Remember to save the last slot before exiting the editing menu.

These system options provide significant customization and optimization possibilities, allowing users to tailor **UMX** to their specific workflow and hardware setup.

The options page has a dedicated control area:



The parameters relating to graphics require restarting the program, you can use the dedicated button, the other values only need saving to be taken care of.

Specific OS Options for UMX

UMX is designed to be compatible with different operating systems, each having its own specific requirements and features.

macOS® and LINUX®

- OSX and Linux next release: These environments do not require any special considerations for UMX operation. The MIDI support implemented allows the use of both virtual and physical ports, even with multiple simultaneous connections to the same source/destination.

WINDOWS®

Windows lacks a robust MIDI management system, leading to several limitations:

1) Device Name Changes: Windows® automatically assigns a numeric suffix to connected MIDI devices, modifying their names dynamically. For instance, a device named Midi_keyboard might appear as Midi_keyboard id (id being an integer). This suffix changes whenever the device is reconnected to a different USB port or the system is restarted. As a result, software relying on static device names may fail to recognize the MIDI device consistently, requiring users—especially those on laptops—to manually reconfigure their setup after each reconnection.
2) Single MIDI Port Access: Windows® restricts each MIDI port to a single connection at a time. If a MIDI device is in use by one program, it cannot be accessed by another. Attempting to connect to an occupied port often results in errors or software crashes.
3) No Native Virtual MIDI Ports: Unlike other operating systems, Windows® does not support the creation of virtual MIDI ports, limiting advanced routing and multi-application MIDI workflows.

UMX as MIDI manager

UMX also functions as a MIDI manager on Windows® systems, providing the following features:

True name device manager

UMX consistently displays the true device name, ensuring that all connected devices used in a project remain recognized after a restart or wiring modification.

Multiple connection to a same port

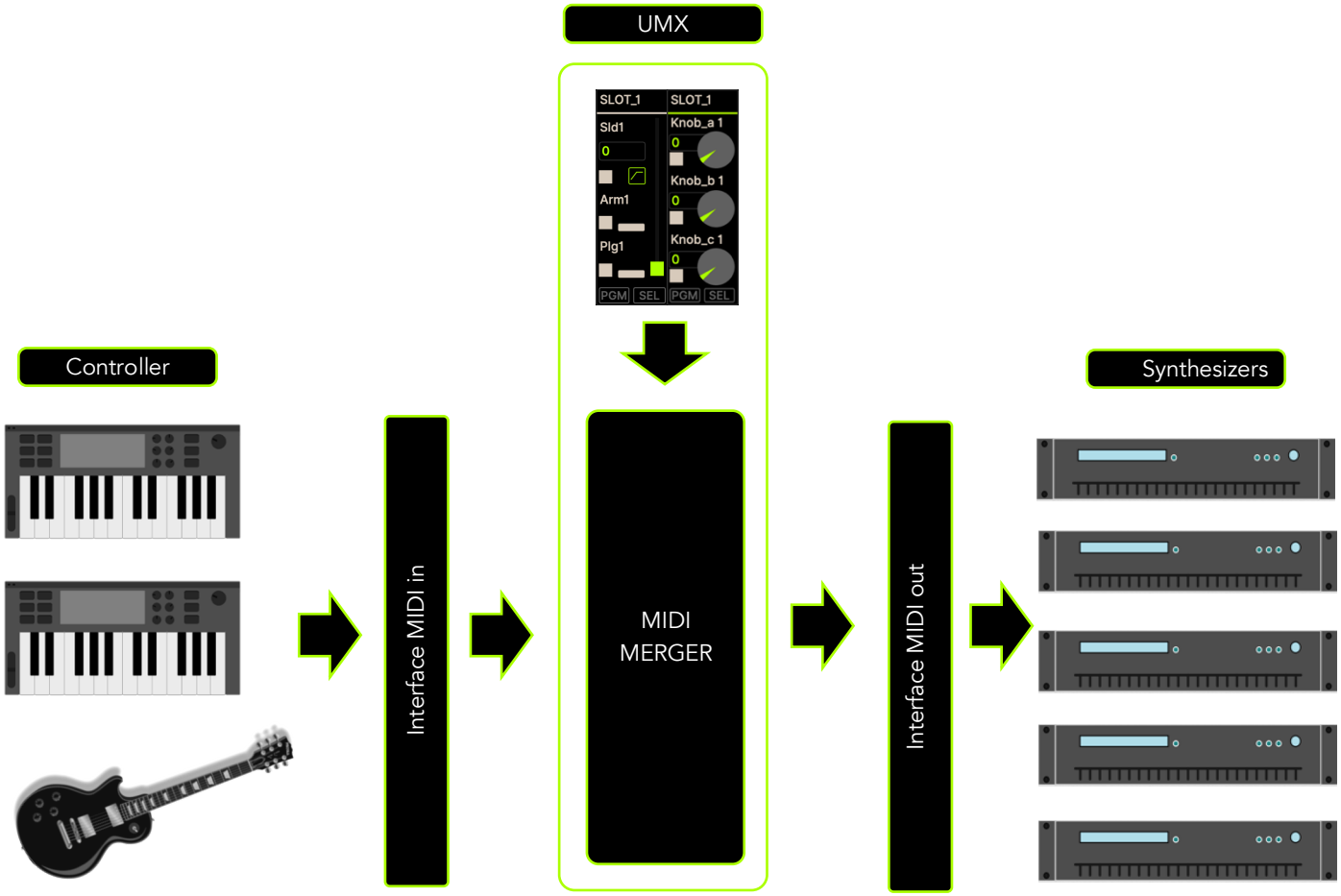
UMX optimizes MIDI port management by opening each port only once and handling all connection requests internally. This allows:

- Each output port to remain available across all inputs (16 trunks and any slots requiring the same port).
- MIDI merging, enabling all UMX input ports to be used on the same output.

It is recommended to launch UMX first, as it will automatically capture all MIDI instruments and ports settled in the performance. This process is highly efficient introducing no noticeable latency, with UMX trunks operating at microsecond-level.

Physical Instrument Setup

If your setup primarily involves physical instruments and resembles the basic schematic, Windows does not pose any limitation. UMX expands the functionality of input ports, allowing the same controller to be retransmitted on multiple outputs, each with its own slots.



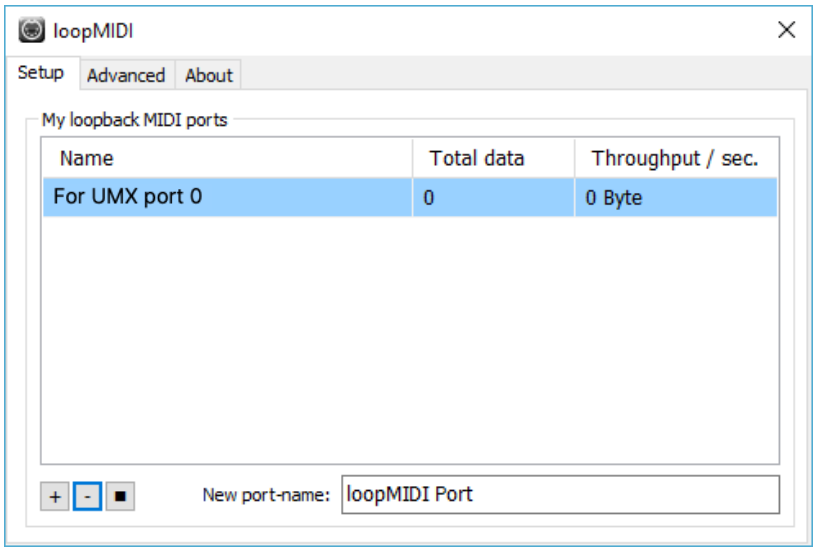
Midi Virtual ports

In the 1_4_2 version UMX always need a third-party driver for MIDI virtual ports (see below), however an Unimidi virtual driver is on its way of development and will be released soon.

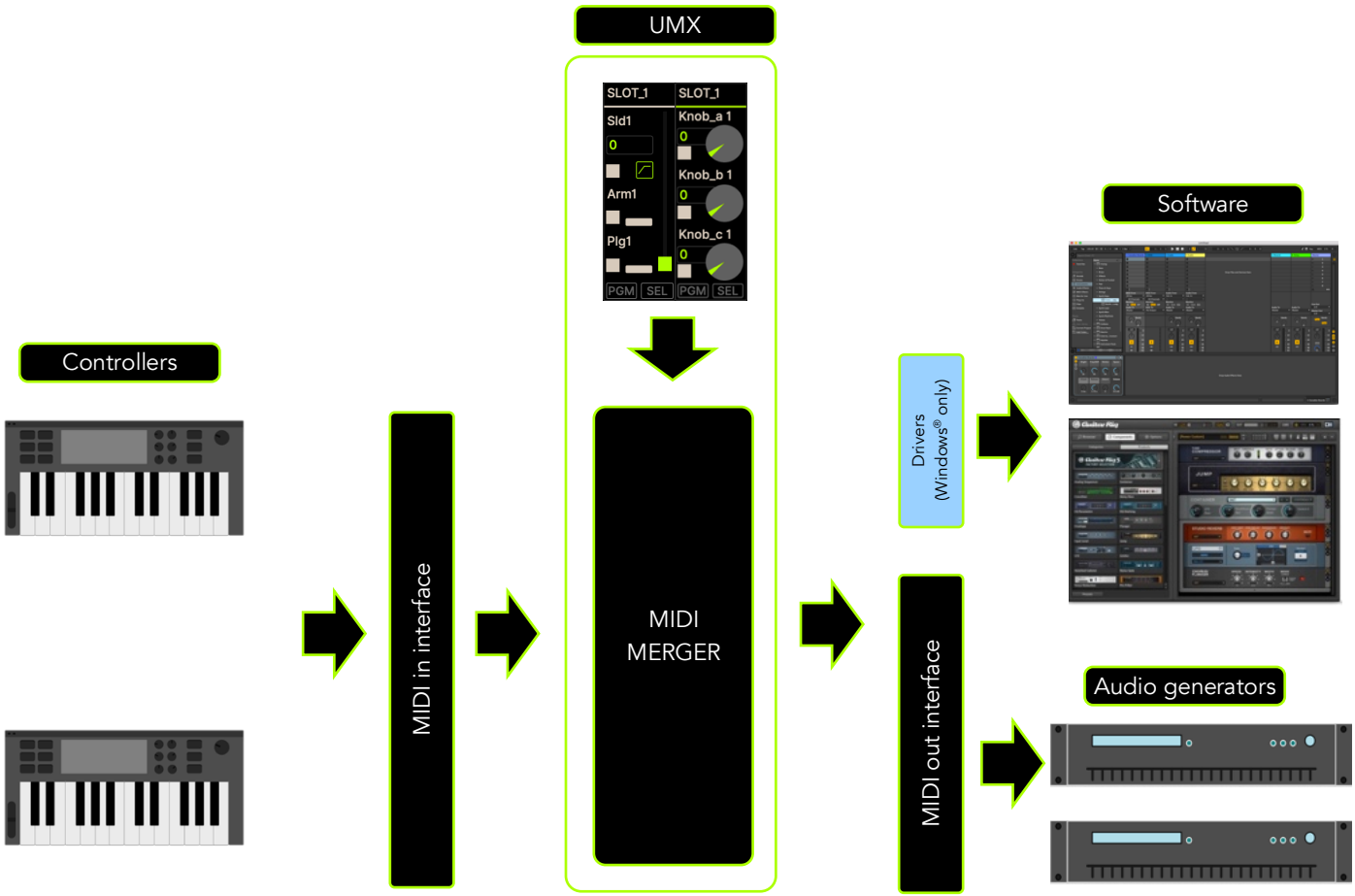
Integration with Other Software in Windows®

- For setups involving other software such as DAWs, software synthesizers, a MIDI driver is required. Various solutions are available online.
- **Example Using Tobias Erichsen's Driver**: A popular option is the driver developed by Tobias Erichsen. You can learn more and download the free personal version from Tobias Erichsen's website.

- **Installation**: Install the driver on your system and launch the program:



- **Setting Up a Virtual Port**: Create a new virtual port using a name of your choice. Enter the name in the 'new port-name' field and press the '+' button.
- **Configuration Completion**: Once set up, UMX will recognize the created input port, and other programs will see the same port as an output source. This setup allows for the creation of MIDI connection scenarios with other programs.

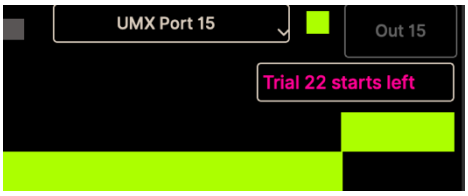


You can create as many virtual ports as you need and take full advantage of the potential of **UMX**.

Software licensing

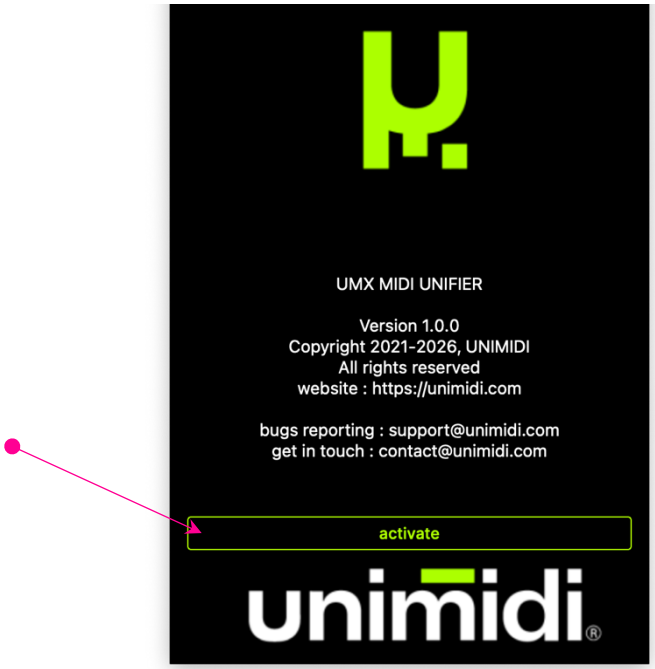
As stated, the UMX software always starts in the trial version after installation. This trial version comes with a limited number of software restarts (64), allowing you to fully experience and benefit from all the features during the trial period. This approach was chosen instead of the conventional 30-day activation method to ensure you have ample time to explore the software capabilities and make the most of the trial version.

During the trial period, a message will always be displayed at the bottom of the user interface:



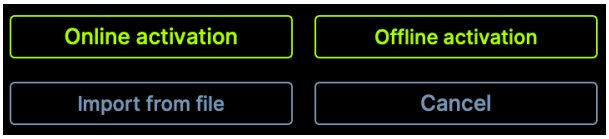
Once you have purchased one of our **UMX** licenses, you will receive an activation code.

To activate the software in the expanded view, click on the **Unimidi** logo in the navigator zone or select *about* from the UMX menu (macOS®) or from the help menu (Windows®). The *about* windows is



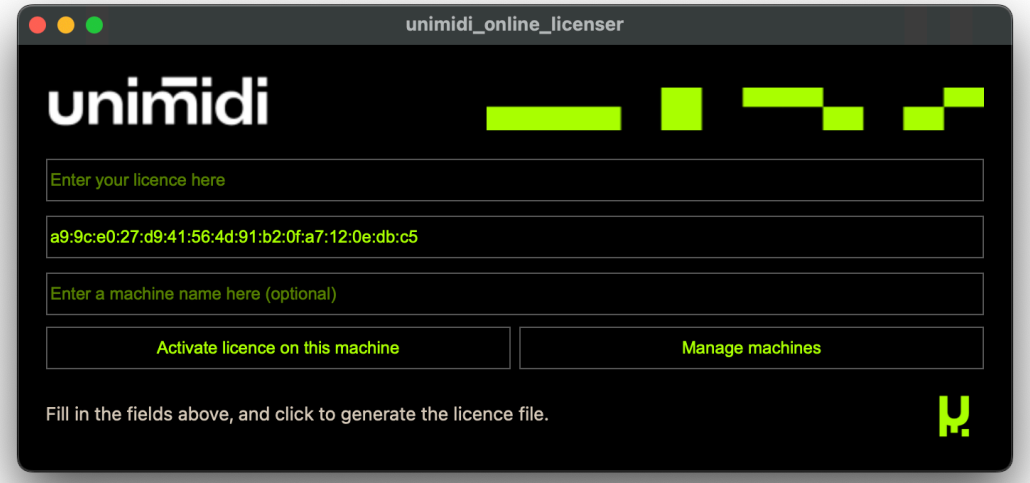
displayed:

Click the *activate* button, a new sub window with four buttons is displayed:



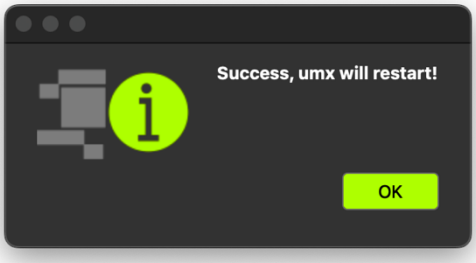
Online activation

If your computer is connected to the internet, choose **Online Activation**. The **Unimidi** licenser module will be displayed:



1. Enter your **license code** in the **License** field.
2. Assign a **friendly name** to easily identify your machine (the **Machine ID** field is automatically filled with your computer fingerprint).
3. Click **Activate License on This Machine**.

If the information is correct, a confirmation message will appear. Close the online_licenser and let UMX restarts.



After restarting the program, return to the **About** window. The **Activate** button will now be replaced with **See Current Subscription**. Click it to display your license details.

Offline activation

If your target machine is not connected to the internet, select **Offline Activation**. A file dialog box will prompt you to choose a location to save the **umx_offline_licenser** tool folder (e.g., your desktop or a USB stick).

Follow these steps:

1) Copy the '**umx_offline_licenser**' folder to a machine connected to the internet.

2) Launch the offline licenser:

Locate the file for your operating system:

Windows: **windows_unimidi_online_licenser.zip**

Mac: **mac_unimidi_online_licenser.zip**

Extract (unzip) the file. Inside, you will find the offline licenser:

Windows: **umx_offline_licenser.exe**

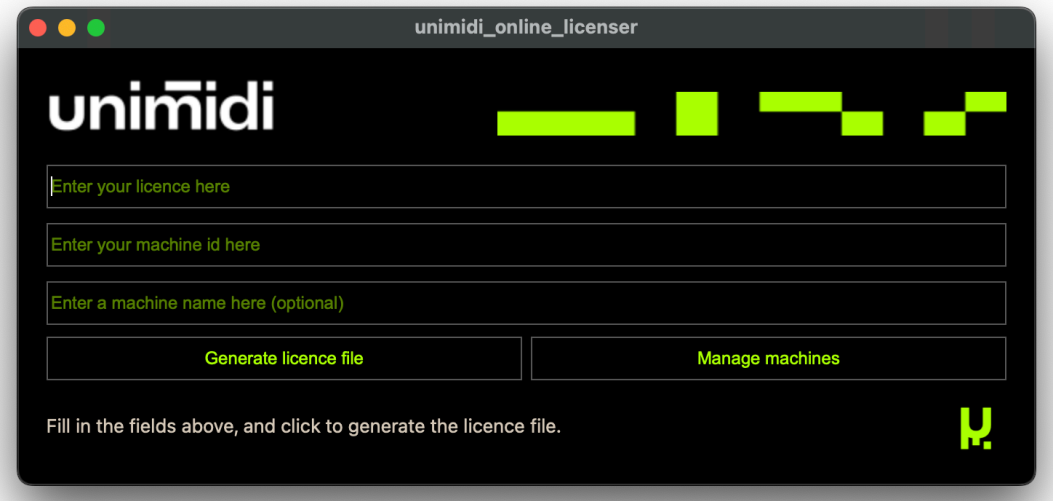
Mac: **umx_offline_licenser.app**

3) Enter your **license code** in the License field. Enter the **Machine ID** from the **IMPORTANT_READ_ME_UMX_OFFLINE_ACTIVATION.txt** file into the Machine ID field. Set a friendly name for easily recognize your machine.

4) Click on 'Generate license file'. The program will validate your license and generate a '**umx_authorize**' file.

5) Save the '**umx_authorize**' file to your desktop or directly to a USB stick.

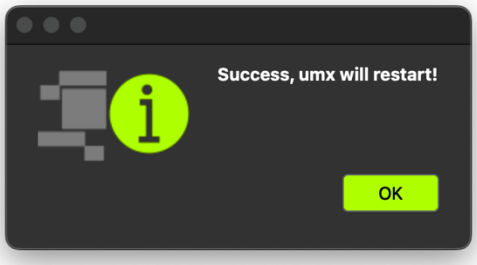
6) Copy the '**umx_authorize**' file to the machine running UMX that is not connected to the internet.



7) Launch the **UMX** software, click on the Unimidi logo or select the 'about' from the main menu

8) Select the 'Import from file' option and import the file you just copied.

If the information is correct, a confirmation message will be displayed. Close the online_licenser and let UMX restarts.



After restarting the program, return to the **About** window. The **Activate** button will now be replaced with **See Current Subscription**. Click it to view your license details.

UMX uses a **floating license** system, allowing installation and authorization on up to **three computers** with the same license code. If you exceed this limit, the software will display a notification. You can manage the devices associated with your license directly in the **Unimidi online licenser.**

For machines management you need to fill both license code and Machine ID fields and the press the 'Manage machines' button.

Appendix A – MIDI CC mapping from protocol

Refer to the list below for your configurations:

control	name	control	name	control	name
0	Bank Select (MSB)	43	Expression Controller (LSB)	86	Undefined
1	Modulation Wheel or Lever	44	Effect Control 1 (LSB)	87	Undefined
2	Breath Controller	45	Effect Control 2 (LSB)	88	High Resolution Velocity Prefix
3	Undefined	46	Undefined (LSB)	89	Undefined
4	Foot Controller	47	Undefined (LSB)	90	Undefined
5	Portamento Time	48	General Purpose Controller 1 (LSB)	91	Effects 1 Depth (Reverb Send Level)
6	Data Entry (MSB)	49	General Purpose Controller 2 (LSB)	92	Effects 2 Depth (Tremolo Depth)
7	Channel Volume (formerly Main Volume)	50	General Purpose Controller 3 (LSB)	93	Effects 3 Depth (Chorus Send Level)
8	Balance	51	General Purpose Controller 4 (LSB)	94	Effects 4 Depth (Detune Depth)
9	Undefined	52	Undefined (LSB)	95	Effects 5 Depth (Phaser Depth)
10	Pan	53	Undefined (LSB)	96	Data Increment (Data Entry +1)
11	Expression (formerly Expression Control)	54	Portamento Control (LSB)	97	Data Decrement (Data Entry -1)
12	Effect Control 1 (formerly Effect 1 Depth)	55	Undefined (LSB)	98	Non-Registered Parameter Number (NRPN) LSB
13	Effect Control 2 (formerly Effect 2 Depth)	56	Undefined (LSB)	99	Non-Registered Parameter Number (NRPN) MSB
14	Undefined	57	Undefined (LSB)	100	Registered Parameter Number (RPN) LSB
15	Undefined	58	High Resolution Velocity Prefix (LSB)	101	Registered Parameter Number (RPN) MSB
16	General Purpose Controller 1	59	Undefined (LSB)	102	Undefined
17	General Purpose Controller 2	60	Undefined (LSB)	103	Undefined
18	General Purpose Controller 3	61	Undefined (LSB)	104	Undefined
19	General Purpose Controller 4	62	Undefined (LSB)	105	Undefined
20	Bank Select (LSB)	63	Undefined (LSB)	106	Undefined
21	Modulation Wheel or Lever (LSB)	64	Damper Pedal On/Off (Sustain) (MSB)	107	Undefined
22	Breath Controller (LSB)	65	Portamento On/Off (MSB)	108	Undefined
23	Undefined (LSB)	66	Sostenuto On/Off (MSB)	109	Undefined
24	Foot Controller (LSB)	67	Soft Pedal On/Off	110	Undefined
25	Portamento Time (LSB)	68	Legato Footswitch	111	Undefined
26	Data Entry (LSB)	69	Hold 2	112	Undefined
27	Channel Volume (LSB)	70	Sound Controller 1	113	Undefined
28	Balance (LSB)	71	Sound Controller 2	114	Undefined
29	Undefined (LSB)	72	Sound Controller 3	115	Undefined
30	Pan (LSB)	73	Sound Controller 4	116	Undefined
31	Expression (LSB)	74	Sound Controller 5	117	Undefined
32	Effect Control 1 (LSB)	75	Sound Controller 6	118	Undefined
33	Effect Control 2 (LSB)	76	Sound Controller 7	119	Undefined
34	Undefined (LSB)	77	Sound Controller 8	120	All Sound Off
35	Undefined (LSB)	78	Sound Controller 9	121	Reset All Controllers
36	General Purpose Controller 5	79	Sound Controller 10	122	Local Control On/Off
37	General Purpose Controller 6	80	General Purpose Controller 1	123	All Notes Off
38	General Purpose Controller 7	81	General Purpose Controller 2	124	Omni Mode Off
39	General Purpose Controller 8	82	General Purpose Controller 3	125	Omni Mode On
40	Portamento Control	83	General Purpose Controller 4	126	Mono Mode On
41	Undefined (LSB)	84	Portamento Control	127	Poly Mode On
42	Undefined (LSB)	85	Undefined		

Appendix B – Keyboard shortcuts

ESC: deselect all slot

CTRL+O (CMD+O): Open file

CTRL+',' (CMD+','): Open system settings

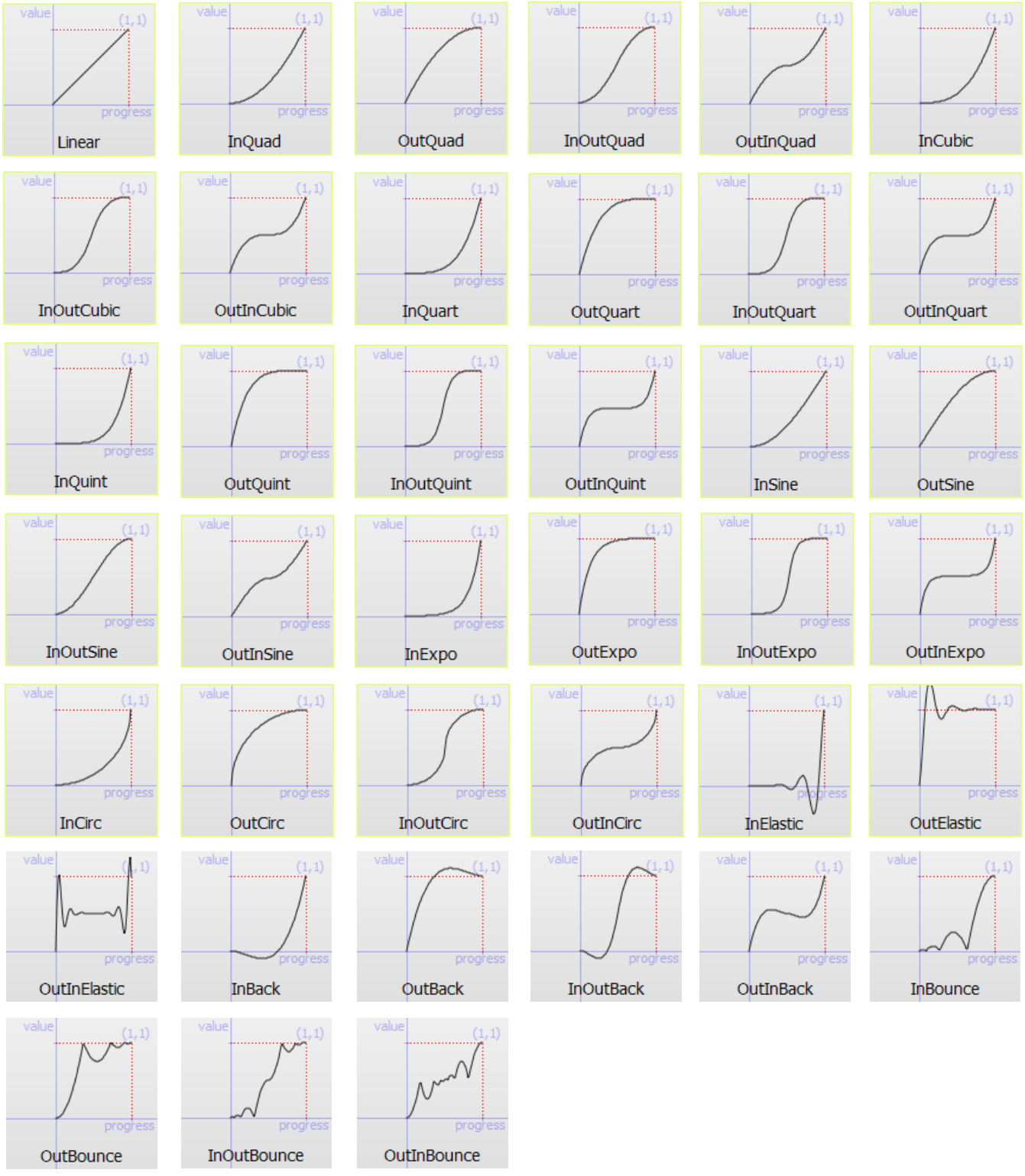
CTRL+M (CMD+M): Open database manager

CTRL+S (CMD+S): Save all layers

CTRL+T (CMD+T): Transmits the following on all open output ports and MIDI channels: MMC, *stop*, *all notes off*, and *sustain pedal off*

CTRL+Z (CMD+Z): Undo: Reverts changes in text and numeric fields to the previous value. *(Note: This only works while the field remains selected or until the current patch is saved.)*

Appendix C – available shapes for non-linear controls



Appendix D – UMX software specifications.

UMX —virtual modular MIDI controller and MIDI profiler

UMX is a modular virtual MIDI environment designed for routing, profiling, and high-volume MIDI data transmission.

Core features

- Massive MIDI transmission of *control-changes*, program-changes, MIDI notes, linear and nonlinear *control-changes* value transitions, HD *control-changes*, MMC (*MIDI Machine Control*).
- Instantly assign key range, transpose settings, and midi channel to your instruments.
- Midi merger / Midi dynamic routing.
- Retrigger up to 12 notes per slot to multiple midi outputs, generate mouse actions from any MIDI instrument.

- Full MIDI profiling per input/output path (trunk), including message filtering, curve shaping, remapping, and redirection of MIDI data.
- Midi Manager on Windows® systems: resolve midi names, connect multiple instruments and software to the same midi source/destination.

umx is available in three versions:

- Performance edition featuring an 8x8 matrix, offering 8 Slots and 8 Trunks.
- Pro Edition featuring a 16x16 matrix (16 Slots and 16 Trunks).
- Lite Edition (free) featuring a 3x3 matrix (3 Slots and 3 Trunks).

Massive MIDI transmission

UMX allows bulk MIDI data transmission with a single action. This can be triggered via the remote launcher umx_100, directly through the software interface, or by an incoming MIDI standard message.

MIDI data bulks are composed by various signals, depending on the configuration. The **standard_slot** allows the simultaneous transmission of up to:

- 48 *control-changes*
- 16 *program-changes*
- 16 MMC CC (*play, stop, continue*)
- 16 *releases sustain pedal* CC
- 16 *all note-off* CC

Using other types of slots allows for the composition of different bundles. For example, by using 16 **program_changer** slots, UMX can transmit 64 *program-changes* at once per patch.

- **output ports**: MIDI data bulks are transmitted through up to 16 MIDI ports. Ports can be physical, for connected devices, or virtual, generated by the software[2].

- **data customization:** *program-changes* (MIDI channel, value, MSB, LSB), control-changes (MIDI channel, cc number, value), HD control-changes (NRPN 14bit format values from 0 to 16383), notes (MIDI channel, duration and sustain pedal) are transmitted through one or more ports.

- **slot automapping:** when slots are created, they will map each controller in the MIDI control change range 102-119 on MIDI channels 14, 15, and 16. By default, a full

[2] Virtual output ports are available on macOS® and next Linux® releases, on Windows® a thirty-party driver is required (tested)

performance offers up to 48 control-changes without any overlap in a free control change interval, which is ready to be acquired or remapped.

- **smooth transitions**: MIDI control change values transmitted in bulk can either apply instantly or gradually over a programmable duration.

- **tempo management**: UMX now supports MIDI CLOCK transmission across all output ports. It can also function as a tempo follower, synchronizing with external hardware or software clock sources. All transitions align with the global tempo, whether emitted or received.

- **bulks structure**: MIDI controls are displayed in slots to be configured and managed. Slots are organised into patches and performances. Invoking a patch immediately trigger the MIDI bulk transmission.

- **db format:** Each project file can store up to 128 performances, with each performance containing up to 32 patches. A single project supports a maximum of 4096 MIDI data bulks. The number of projects is unlimited, and the program can rapidly switch between the last 10 used projects.

- **workflow:** musicians can create and edit performances by adding and manipulating different types of slots. All slots have a performance configuration panel for customization. Once the performance level has been defined according to the instrument setup, patches can be generated for the transmission of MIDI messages. The patch and performance layers offer various management functions for copying, duplicating, inserting, and organizing.

- **Controls identification:** All slots can be named, users can choose a slot colour for easier identification, all controls within slots can be individually named for each performance.

- **modularity:** the 1_4 version of UMX has nine slot types that cover a wide range of possibilities. The software has been designed to host slots like a MIDI DAW. Many more types of slots with both typical and atypical MIDI control functionalities will be added.

- **adaptive interface:** UMX features an extended graphical interface providing access to all modules. The collapsible UI optimizes screen space for use alongside other software. Users can freely configure size and positioning via the options panel.

- **MIDI compatibility:** UMX recognizes all MIDI devices connected to the computer, whether they are MIDI class-compliant or require a specific driver. If your operating system can recognize the device, UMX can too. **MPE** *(Midi Polyphonic Expression)* controllers can be used on all input ports.

- **extensive MIDI metering:** UMX is equipped with midi_meters that constantly show rx/tx activity, MIDI messages are displayed and classified by their types.

Midi merger

The embedded **midi_merger** redirects and blends MIDI slot controls with instrumental musical performance. Musicians can trigger the transmission of control messages to a wide number of instruments during their performance.

- **input ports**: the software can open 16 MIDI ports as inputs, the same port can be used for all inputs. Each port is opened once, and all MIDI instrument ports can be used with multiple MIDI destinations, even in the Windows architecture (macOS and Linux natively support multiple accesses to the same MIDI ports).

- **latency:** incoming MIDI messages are retransmitted to available MIDI outputs in the same millisecond (zero latency).

- **MIDI filters**: MIDI messages can be dynamically filtered (transposition, keyboard extension range, allow/block cc and pc), filters can be set for each patch individually.

- **matrix:** the node matrix dispatches the messages of slots to output ports based on their types. The software can control plugins, DAWs, standalone music software, and all MIDI devices.

- **external controls:** performances and patches can also be recalled via MIDI. One of the 16 inputs can be set as a controller, allowing UMX to be controlled by a standard MIDI launchpad. DAWs can trigger UMX transmissions in sync with audio and MIDI tracks.

- **output ports control:** 16 MIDI ports can be opened as merger outputs, transmission to these ports support MPE and single-channel MIDI.

- **dynamic routing:** output Port activation is controlled by each patch to dynamically select which instrument is played by input controllers.

MIDI Profiling

UMX introduces per-trunk MIDI profiling through its built-in pass_through manager, enabling real-time transformation of more than 130 incoming MIDI messages types. Each trunk—representing a connection between one input and one output—can apply:

- Filtering of unwanted messages (e.g., block PC/CC from hardware controllers)

- Curve shaping to modify input response (linear, logarithmic, inverted, custom etc.)
- Redirection of incoming MIDI data to different output ports and/or message types
- Saving a complete MIDI profile to your instrument profiles list

This makes UMX not just a routing tool but a dynamic layer of MIDI adaptation, perfect for both live and studio contexts. MIDI profiling settings are saved in trunk models and recalled automatically with each performance.

Other tools

- **monitor module:** this module will monitor all MIDI communications on multiple ports simultaneously. It will provide information such as the timestamp, port, channel, message type, and message values. The monitor module will also instantly detect all available ports.

- **database manager:** this dedicated tool enables users to manage performances and patches within projects. It provides various functions for copying, moving, duplicating, and creating new items. The search functions help quickly locate performances and patches, even if they are contained in non-loaded performances. Additionally, a powerful adaptive copy function allows users to copy patches from one performance to another, even if the structures are different.

- **Slot sharing settings panel:** this tool enables users to copy some parameters of a slot from one patch to other selected patches, facilitating the management and synchronization of settings.

Minimum requirements

Windows®

- Windows 10 (version 22H2) or Windows 11 (version 22H2 or higher)
- 5th generation Intel® Core™ i3 processor or AMD
- Memory 4 GB RAM
- **Note:** Windows ARM chips are not always compatibles.

macOS

- macOS 11 Big Sur or higher
- Intel® Core™ i5 processor or Apple Silicon
- Memory 4 GB RAM

For all systems: download and install the trial version before to purchase umx, get in touch with the unimidi team for further information.

www.unimidi.com