BLUE ARP

Operation Manual

Corresponds to BlueARP v2.8.0

Pattern Arpeggiator / Step Sequencer / Drum Sequencer

VST/AU midi-FX plug-in for Windows & OSX

by Oleg Mikheev aka Graywolf

http://www.omg-instruments.com

January 2025

Table of Contents

INTRODUCTION **4**

SPECIAL THANKS **5**

HOW TO INSTALL BLUEARP **6**

WINDOWS VST2 VERSION 6
WINDOWS VST3 VERSION 6
MAC OSX VST2 VERSION 6
MAC OSX VST3 VERSION 6
MAC OSX MIDI-FX VERSION (FOR LOGIC PRO X) 7

HOW TO REMOVE BLUEARP **8**

SETTING UP BLUEARP IN SOME DAWS **9**

FL STUDIO (FRUITY WRAPPER METHOD) 9
FL STUDIO (PATCHER METHOD) 11
ABLETON LIVE 12
REAPER 6.X AND LATER 13
SAMPLITUDE PRO X8 14
CUBASE PRO 13 15

SIGNAL FLOW **16**

BASIC CONCEPTS **18**

SWITCHING PATTERNS ON THE FLY 18
USING BLUEARP AS A STEP SEQUENCER 19
OPERATION MODES 19

INTERFACE **20**

ELEMENTS AND NAVIGATION 20
MAIN WINDOW LAYOUT 21
BLOCK (1): TOP PANEL 22
Operation mode and State 22
Program browser 22
BLOCK (2): LEFT PANEL 23
Input filter 23
ARP Engine 27
Output filter 29
Chains 30
BLOCK (2): SETTINGS 31
BLOCK (4): MAIN MENU AND PATTERN CONTROLS 34
BLOCK (5): MATRIX EDITOR 36
BLOCK (6): VALUE LANES 36
BLOCK (7): CHAINS 39
BLOCK (8): INFO PANEL 42

DRUM SEQUENCER MODE **43**

OVERVIEW 43
LEFT PANEL 43
Input filter 43
ARP Engine 43
Drum lane 45
VALUE LANES 47

GUITAR STRUM MODE 48

OVERVIEW 48
LEFT PANEL 48
Input filter 48
ARP Engine 48
Guitar strum 49
Output filter 51
Chains 51
VALUE LANES 52

FAQ / TROUBLESHOOTING 54

INSTALLING BLUEARP 54
SYNC & TIMING ISSUES 54
RENDERING AUDIO IN FL STUDIO 54
VST3 COMPATIBILITY 55

LINKS 56

Introduction

BlueARP is a programmable pattern arpeggiator / step sequencer, it comes as a VST or MIDI-FX plug-in for Windows and OSX. BlueARP is a pure MIDI plugin, it doesn't generate any sound by itself but transforms MIDI messages. It has to be routed to either software or hardware synth in any VST/AU-enabled DAW (Digital Audio Workstation) like FL Studio, Ableton Live, Cubase, Reaper, Logic Pro, etc.

Basically, you need to program some pattern in BlueARP, then you play some chords and BlueARP transforms these chords into melodic phrases according to the pattern you programmed or selected.

BlueARP was initially designed for electronic music genres (like Trance, House, New Wave etc.), but later it was expanded with Drum and Guitar operation modes, along with lots of smaller feature updates, so now it can be used in a wide variety of genres.

From 2022, BlueARP has its hardware counterpart called BlueARP DM (Desktop Module), find out more at www.omg-instruments.com.

Compatibility info

Formats:	VST 32-bit (windows only), VST 64-bit, AU MIDI-FX 64-bit (for Logic Pro X)
OS:	OSX (10.9 or later, tested on 12.7.3), Windows XP or higher

Features

- Up to 64 steps per pattern;
- Up to 128 programs per bank;
- «Chains» feature to chain patterns together into longer «super-patterns»
- Chains can be switched on the fly with real-time quantization;
- 128 factory patterns to start with;
- Intuitive matrix editor to program patterns quickly;
- Almost all controls can be automated;
- Up to 5 input keys in a chord;
- Real-time input note quantization;
- Chord recognition, you can crease chord-based patterns;
- Input range setting for keyboard-split performances;
- Separate settings for octave and semitone per step transpose;
- Configurable color schemes (skins);
- Dedicated ‘drum sequencer’ mode since v2.5.0
- Chord-driven chain switching since v2.5.0
- Dedicated ‘guitar strum’ mode since v2.8.0

To get the idea what can be done with BlueARP, check these videos:

https://www.youtube.com/watch?v=1KOGVuElrhY

https://www.youtube.com/watch?v=retDsYjPokA

These are live performances using BlueARP with FL Studio, but the same can be done with Ableton Live and many other DAWs.

Special thanks

Thanks to community at **KVR audio forums**, there are lots of great people there who are interested in electronic music and tools and I feel this was the right place to put the first BlueARP release back in 2012: https://www.kvraudio.com/forum/viewtopic.php?t=361311,
… and as of 2024, this thread is still going.

Special thanks to **Saif Sameer** aka **phreaque**, who was one of the earliest beta testers, he devoted lots of his time to keep that KVR thread online and he was the one who encouraged me to cooperate with Image-Line team and bring BlueARP to FL Studio as a stock plugin (while I was FL Studio user for years already). Also, he coined various cool ideas, the idea of chord-driven chains which was implemented in v2.5.0 as *chain variations*.

Thanks to Image-Line team, who welcomed the idea of integrating BlueARP into their product and finally made this: now BlueARP is also VFX Sequencer, stock FL Studio plugin.

How to install BlueARP

Before installing newer version of BlueARP, it is recommended to remove the existing version first, unless you want to use both older and newer version (refer to the next chapter «How to remove BlueARP»).

Windows VST2 version

Step 1. Unzip the package, copy "BlueARP_Win_VST2_vXXX" folder to your VST plugins directory.

Normally it will be:

- C:\Program Files\Steinberg\Vstplugins\ or
- C:\Program Files (x86)\Steinberg\Vstplugins\ *(for 32-bit plug-ins on Windows 64-bit)*

Step 2. In you DAW (Cubase, FL Studio or whatever you use), re-scan VST plugins folder (refer to the respective manual on how to do this). «BlueARP» or «BlueARP.x64» (64-bit version) should appear in plugin list and it is now ready to use.

Windows VST3 version

Step 1. Unzip the package, copy "BlueARP_Win_VST3_vXXX" folder to your VST3 plugins directory.

Normally it will be:

- C:\Program Files\Common Files\VST3\ or
- C:\Program Files (x86)\Common Files\VST3\ *(for 32-bit plug-ins on Windows 64-bit)*

Step 2. In you DAW (Cubase, FL Studio or whatever you use), re-scan VST plugins folder (refer to the respective manual on how to do this). «BlueARP» should appear in plugin list and it is now ready to use.

PS. Some DAW applications may combine VST2 and VST3 version of the plugin, in some cases VST3 version may have priority over VST2 version. FL Studio, for example, has a setting "Combine VST2 and VST3 versions of the plugin".

Mac OSX VST2 version

Step 1. Unzip the package, copy «BlueARP_OSX_VST2_vXXX» folder to your VST plugins directory.

It should be one of the following:

- Hard disk/Library/Audio/Plug-Ins/VST *(for all users)*
- Hard disk/Users/<username>/Library/Audio/Plug-Ins/VST *(for <username> only)*

Step 2. In you DAW (Cubase, FL Studio or whatever you use), re-scan VST plugins folder. BlueARP should appear in plugin list and it is now ready to use.

Mac OSX VST3 version

Step 1. Unzip the package, copy "BlueARP_OSX_VST2_vXXX" folder to your VST3 plugins directory.

It should be one of the following:

- Hard disk/Library/Audio/Plug-Ins/VST3 *(for all users)*
- Hard disk/Users/<username>/Library/Audio/Plug-Ins/VST3 *(for <username> only)*

Step 2. In you DAW (Cubase, FL Studio or whatever you use), re-scan VST plugins folder. BlueARP should appear in plugin list and it is now ready to use.

Mac OSX MIDI-FX version (for Logic Pro X)

Step 1. Unzip the package, take "BlueARP.component" from the "BlueARP_OSX_MFX_vXXX" folder and copy it to your Audio Units directory.

It should be one of the following:

- Hard disk/Library/Audio/Plug-Ins/Components *(for all users)*
- Hard disk/Users/<username>/Library/Audio/Plug-Ins/Components *(for <username> only)*

IMPORTANT: just take "BlueARP.component", don't copy the entire "BlueARP_OSX_MFX_vXXX" folder, otherwise Logic won't see the plugin.

Step 2. Restart! OSX caches AU plugins and you won't see your new plugins until you restart.

If it doesn't help, try to:

1. Clearing the folder /Users/graywolf2004/Library/Caches/AudioUnitCache
2. Run in terminal:

    killall -9 AudioComponentRegistrar

3. Remove the quarantine flag by running in terminal

    sudo xattr -rd com.apple.quarantine <your components path>/BlueARP.component

4. After you start Logic, check your system -> Security page, sometimes you might see a warning there about the unverified developer/app, so you can choose 'install anyway' (this was relevant for the earlier OSX versions, not sure about the latest ones)

In you DAW (Logic Pro, Garage Band or Main Stage), re-scan Audio Unit plugins folder. BlueARP should appear in plugin list and it is new ready to use.

If it doesn't appear in the plugin list, try to reboot or logout/login. I got some complaints on this in 2022 and it seems like Logic now needs system reboot to see new installed MFX plugins.

Troubleshooting

> Since v2.3.8, BlueARP is notarized with the proper Apple notary tool. While it is not listed in the Apple store, it is still digitally verified by Apple. However, it doesn't give 100% guarantee that the installations will be flawless. The majority of the problem reports is related to Logic X Pro and MIDI-FX version «BlueARP.component».

If your Logic X or Garage Band still doesn't see BlueARP, try the following:

Remove the quarantine flag from the plugin by running in terminal:

```
xattr -rd com.Apple.quarantine BlueARP.component
```

If this doesn't help:

1. clearing the folder /Users/graywolf2004/Library/Caches/AudioUnitCache
2. running killall -9 AudioComponentRegistrar in terminal
3. Choosing Preferences on Logic -> Reset and rescan all audio units in Logic

How to remove BlueARP

BlueARP has no installer, so just remove "BlueARP_Win*" folder on Windows or "BlueARP_OSX*" folder on Mac (the one you copied during installation).

If you want to remove all traces of BlueARP in your system, also delete the following folder:

Windows: C:\Users\<user>\AppData\Roaming\BlueARP

OSX: C:/Users/<username>/Library/Application Support/BlueARP

This is the place where BlueARP stores its “ini” file with the settings like selected skin index, GUI scale. It is a small file, way below 1 Kbyte in size.

The reason I had to put these settings into separate folder is because VST/AU folder with the plugin itself often doesn’t grant write permission to the plugin, so it can’t save the settings.

Troubleshooting

When you try to delete the folder, system may give an error: “Oxanium*.ttf” files are locked by the system. This may happen because BlueARP uses these fonts for GUI rendering, they are bundled into the package. Upon loading, system locks these files and won’t allow deleting them.

To solve this problem, try the following:

- Manually delete all “Oxanuim” fonts from your system
- Reboot
- Try to delete the folder again

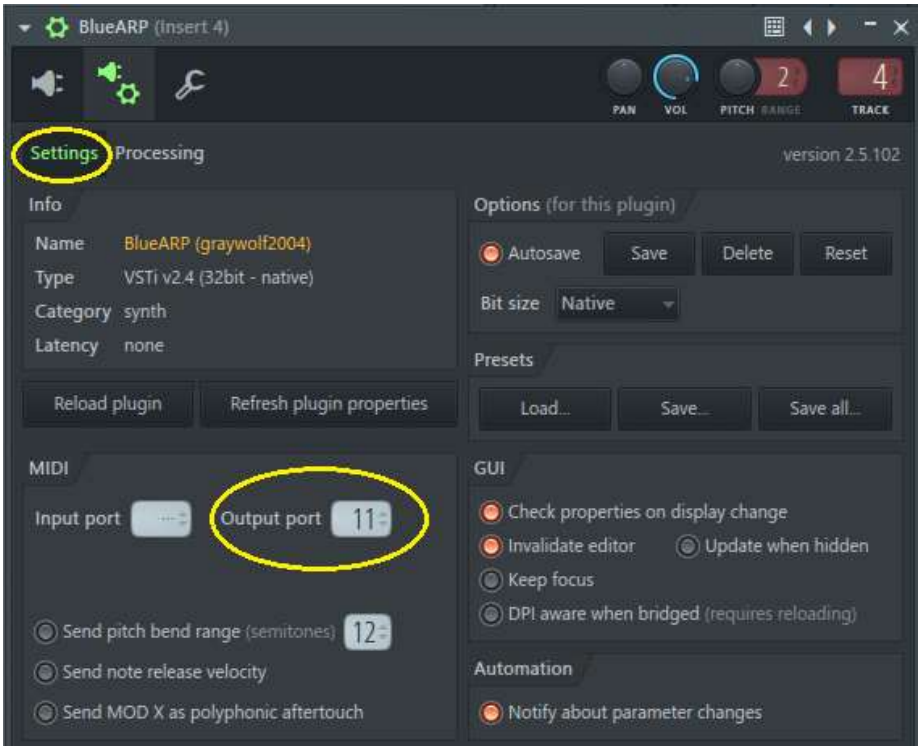
Setting up BlueARP in some DAWs

If your DAW is not present in this list, refer to other VST arpeggiator manuals like *Kirnu Cream*, *Catanya*, *Nora* or search for tutorials with keywords «***how to set up MIDI plugin in DAW ZZZ***». For BlueARP procedure should be the same as for any other MIDI plugin.

FL Studio (Fruity Wrapper method)

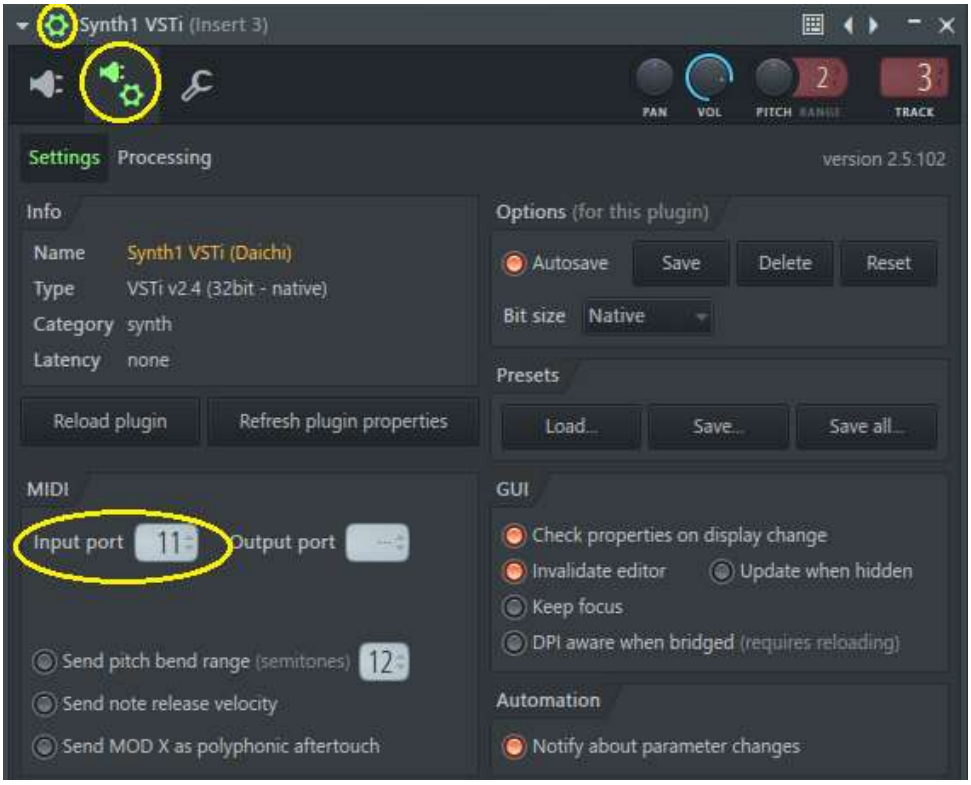
Load BlueARP, click the buttons as shown on picture:

Click «SETTINGS» tab, set «Output port» to any value, not occupied by hardware MIDI devices and memorize this value (we will need it further):



Return to main plugin window:

Go to Fruity Wrapper settings of a VST synth (Synth1 in our example), set «Input port» to the value we memorized on the previous step:

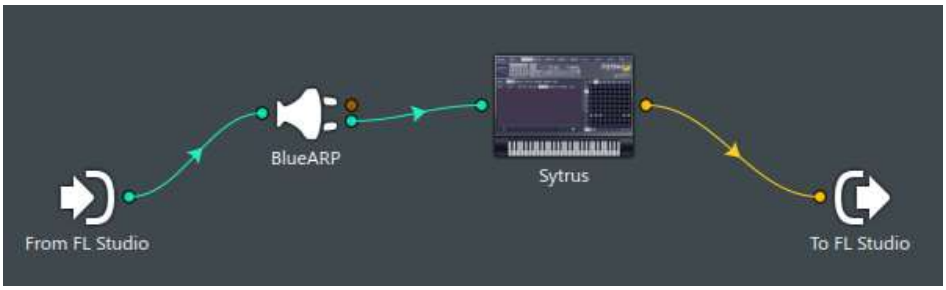


This way we tell FL Studio to route MIDI messages from BlueARP's MIDI output to Synth1's MIDI input. Just make sure this MIDI port is not occupied by hardware synths or other routings.

> **Hint**. *I usually reserve ports 1 – 10 for hardware MIDI devices and use numbers 11 and above for software routings.*

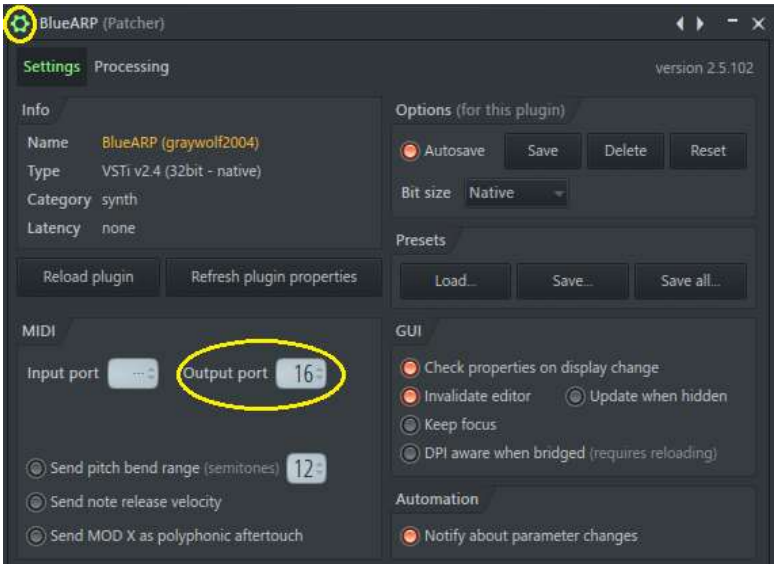
FL Studio (Patcher method)

Add «Patcher» instrument to the track, inside Patcher add BlueARP and Fruity Generator of choice (Sytrus in our example), connect them as follows:



Green arrows represent MIDI signal flow, yellow arrows - audio signal.

Double cluck BlueARP to open plugin window, go to wrapping settings and set output port to any unused number (**this is important**, otherwise it will not work).



Ableton Live

Ableton is tricky when it comes to MIDI plugins. There are 2 options.

Option 1.

Load BlueARP on one track, VST synth (Synth1 VST in our case) on another.

For Synth1 track, set MIDI From = BlueARP (both list boxes).

For BlueARP track, set Monitor = «In».

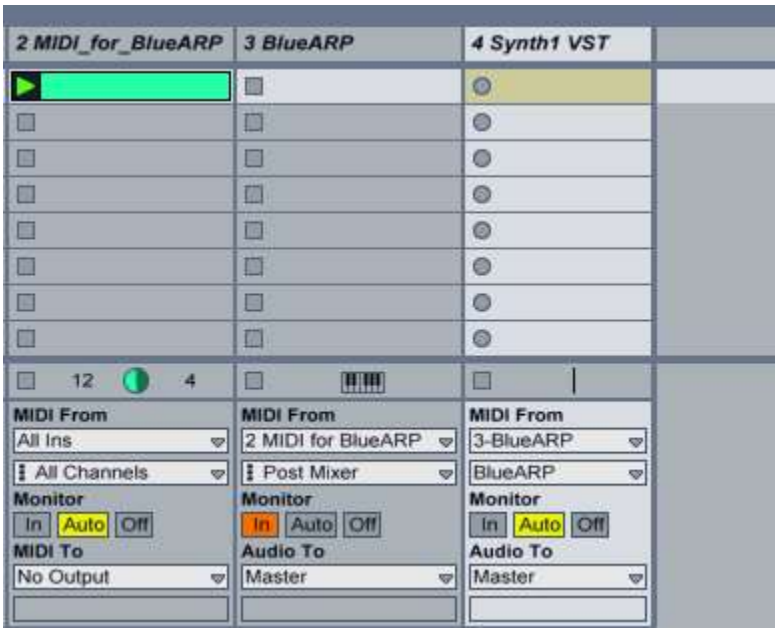
There's an issue - BlueARP will pick up MIDI from clips only when Monitor = «Auto», but it takes notes from Keyboard only with Monitor = «In». So, you have to constantly switch monitor from «In» to «Auto».

If you want to avoid it, go for Option 2.

Option 2.

Create a separate track (say «*MIDI_for_BlueARP*»), it will hold your MIDI clips.

Add 2 more tracks, one for BlueARP and another for a VST synth. Now we have 3 tracks in total:



For the track «*MIDI_for_BlueARP*», set Monitor = «Auto».

For «*BlueARP*» track, set MIDI From = «*MIDI_for_BlueARP*», Monitor = «In».

For «*Synth*» track, set MIDI From = «*BlueARP*» (both list boxes!), Monitor = «Auto».

Now, use «*MIDI_for_BlueARP*» track to record patterns and «*BlueARP*» track to play live.

If you want to drive hardware synth (connected via MIDI), use «External instrument» device (it's in «Live Devices» list) instead of a VST.

Reaper 6.x and later

Add a track with VSTi synth of choice, click IN FX setting

Pick BlueARP from the list

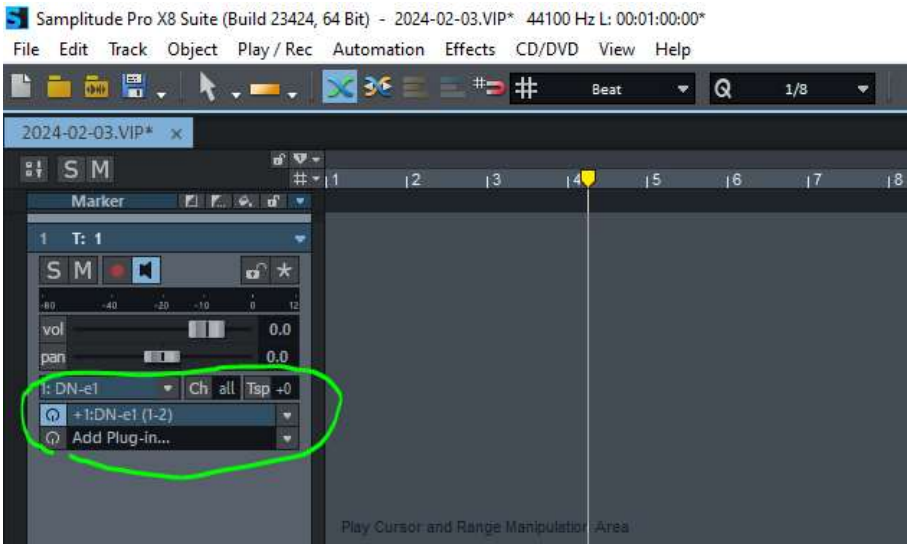
Done! IN FX button will become green

Click it to open BlueARP window.

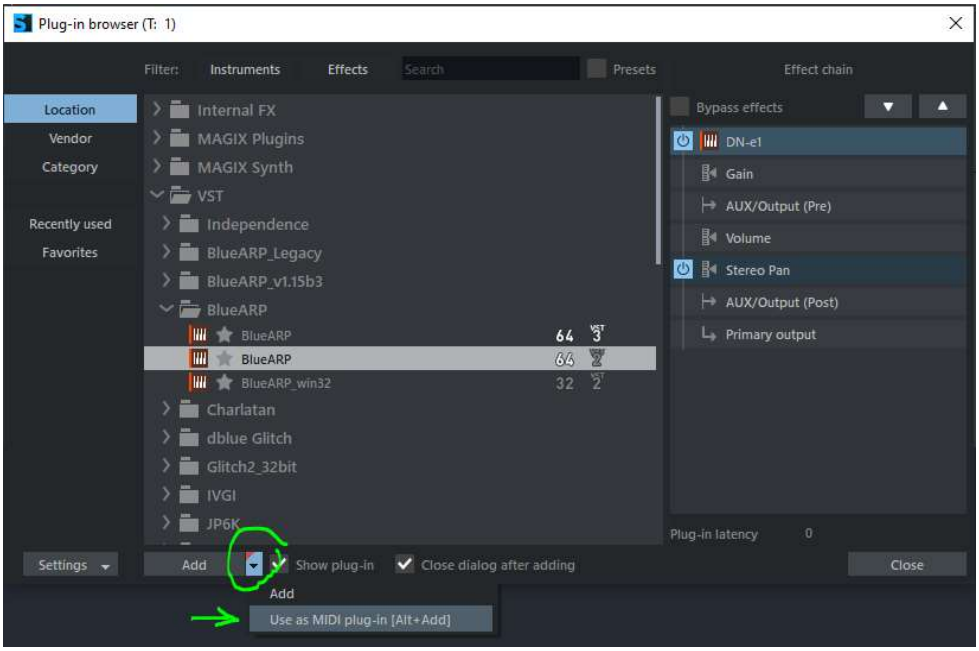
Samplitude Pro X8

Newer versions of Samplitude support MIDI plugins as a track insert (no need to add separate track for the BlueARP), this is the newer «MIDI insert» method described.

First, arm new or existing track with an instrument of choice:



Next, click «Add Plug-in…», select BlueARP from the list. Instead of clicking «Add» button below, select «Use as MIDI Plug-in» from the drop-down list.



BlueARP will appear in the plugin stack above the instrument plugin:

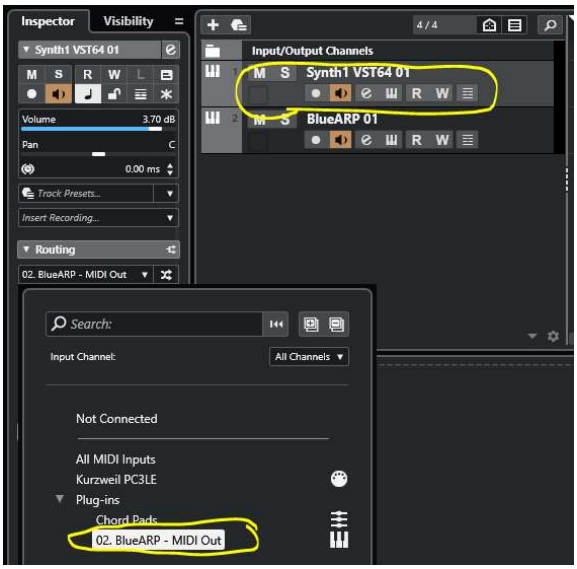
Hint: Faster way to do the same: hold ALT while selecting and adding the plugin.

Cubase Pro 13

Actually, this is relevant for another versions of Cubase as well, but the important thing to mention is that despite the fact that the latest Cubase has 'MIDI Inserts' section on the instrument tracks,

you cannot use if for the 3rd party MIDI plugins, it is only Steinberg's own arpeggiator and some other MIDI effects. They didn't open this for 3rd party developers and probably never will. You can still use BlueARP in Cubase, just another way.

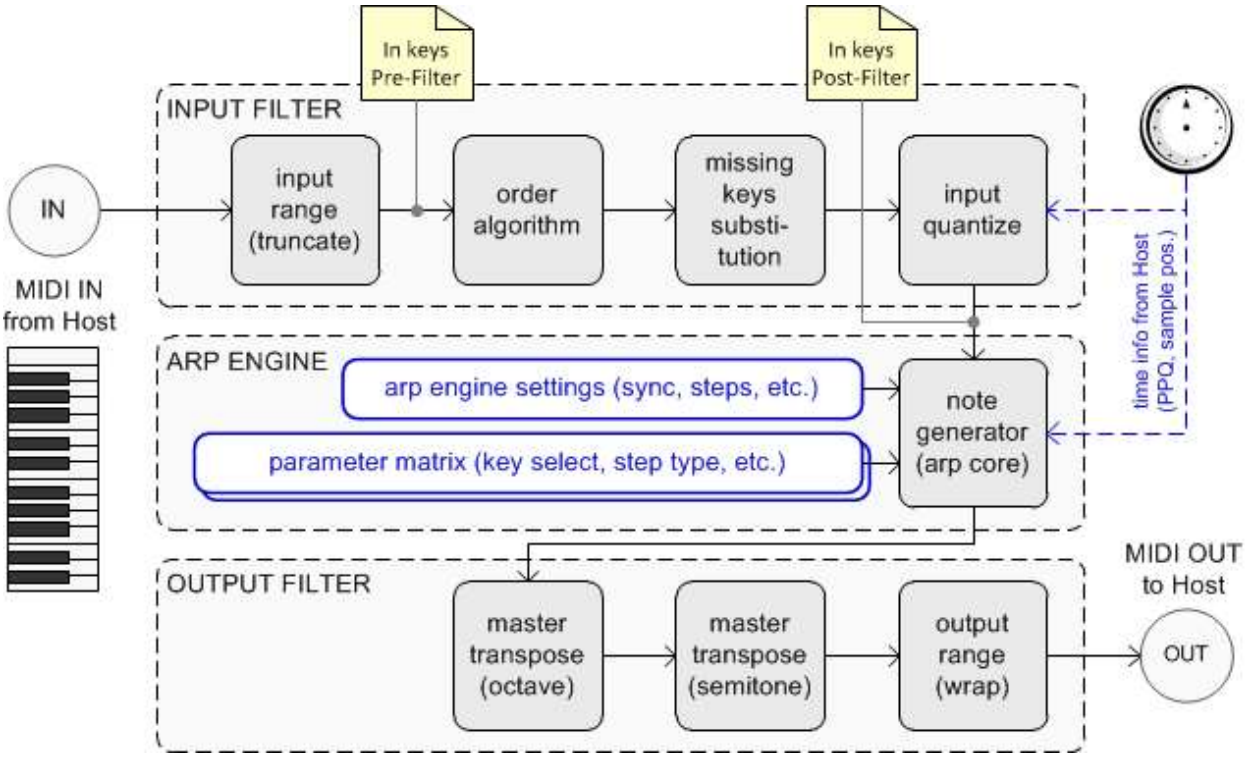
To use BlueARP in Cubase, you have to add BlueARP on a separate Instrument track. Then, go back to your synth track (the one you need to drive with BlueARP) and in the Routing section change MIDI input from your keyboard to one from BlueARP's MIDI output:



Also make sure both of your tracks are armed for either monitoring (speaker icon) or recording:

Signal flow

The picture below shows a basic data flow diagram for BlueARP. At the input BlueARP receives MIDI events from the host. These are real-time events of pressing/releasing the keys on a MIDI keyboard or events coming from the MIDI track. At the output we have the same type of events (MIDI notes), generated by the arpeggiator engine and further transposed by output filter.



pic. 1. BlueARP processing diagram.

Main blocks are «Input Filter», «Arp Engine» and «Output Filter».

> *In this manual, «keys» are actually pressed notes on the keyboard, while generated «notes» come from arpeggiator output.*

Input Filter receives MIDI events from the host - key press and release events; also, it may be pitch bend, aftertouch and controller messages. From key «on» and «off» events, it generates *Key List* – an ordered list of keys with corresponding velocities (velocity is how hard you pressed a key).

«*In keys Pre-Filter*» is a key list as it comes from Host (keys are ordered as they were pressed). «*In keys Post-Filter*» represents the same key list after ordering, missing keys substitution and real-time quantization (for further details on Input Filter, go to page **Error! Bookmark not defined.**).

«*In keys Post-Filter*» goes directly to the arp core.

You can see what's currently in both key lists on the Information panel at the bottom:

See Information panel description on page 42.

Arp Engine transforms keys coming from input filter into melodic phrases according to per-step settings in Value lanes (STEP TYPE, KEY SELECT and others). For example, «KEY SELECT» lane determines which key to take for the current step (k1 – key 1, k2 – key 2, fix – fixed key, etc.). «STEP TYPE» lane tells whether this step is a normal note (Nrm), the rest/sustaining note from the previous step (Rst) or muted (Off). Refer to page 36 for more information about Value lanes and Matrix editor.

BlueARP has unique «*missing keys substitution*» feature. It works like this: when you have, for example, 4-keys pattern and play 2-key chord, by default («*missing keys substitution*» - «*don't play*») all steps with KEY SELECT = k3, k4 or K5 will be muted, cause these keys are not present at the input. If you select other options for «*missing keys substitution*», these missing keys will be substituted with the existing ones. There are several substitution algorithms, see page 23 for details.

Output Filter adds some post-processing to generated notes – octave / semitone transposition, wrapping notes to fit the given range. See page 28 for more details.

Chains block allows you to merge several programs together to create longer patterns. You can automate current chain parameter and switch chains on the fly – it was implemented with live performances in mind. See page 39 for more details.

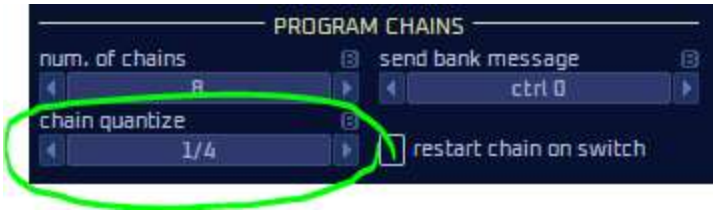
Basic Concepts

Switching patterns on the fly

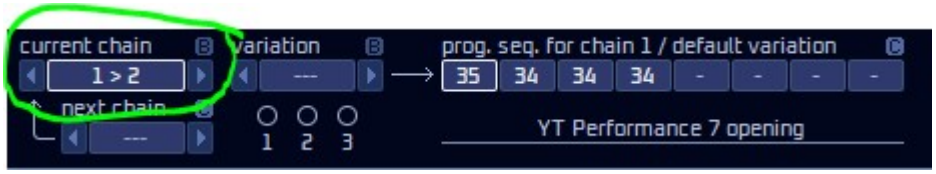
You can either switch programs or assign programs to chains and switch chains. While you can switch programs on the fly during the performance, this switching will not be quantized and you may experience some early/missing notes at the output. In BlueARP, it is better to use chains instead.

Chain is a pre-programmed sequence of programs, which can be automated and changed on the fly/ Each chain can contain just 1 program or up to 8 programs, which will play one after another.

The main advantage of chains: chain switching is quantized according to this setting on the left panel:



In this case, chain will actually switch not when you change «current chain» param, but at the start of the next beat (or 1/4 of a bar). While chain is about to switch but didn't switch yet, it is highlighted with a white frame and the numbers «1 > 2» say that it is switching from chain 1 to chain 2, but yet is on chain 1.

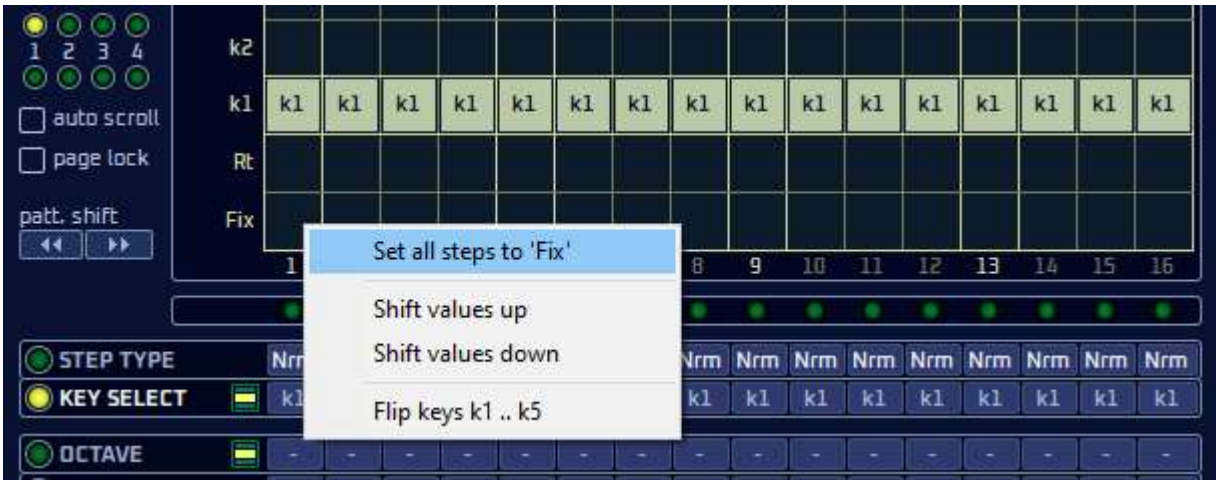


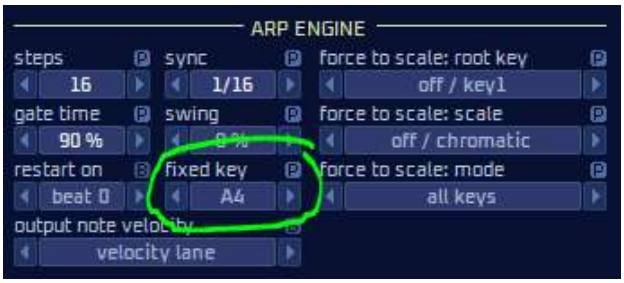
Chain variations add even more flexibility to chains: you can have up to 3 variations per chain, each with its own program sequence, triggered by a certain chord, key, root note, etc. For example, this condition is input key k1 is in range C0 .. C1:

Once this condition is true, chain variation will trigger automatically (see page 39 for details on chains and chain variations).

Using BlueARP as a step sequencer

To do this, set all steps in your program to «Fixed»: select the KEY SELECT lane, then right-click on the matrix somewhere on the «Fix» value, choose «Set all steps to 'Fix'».

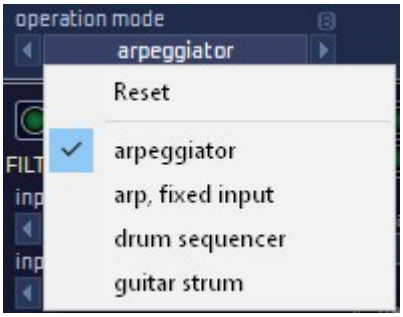




All KEY SELECT lane values will become «Fix» and now output notes will be bound to «fixed key» param in ARP ENGINE block:

Operation modes

With the concept of operation modes, BlueARP is basically a several arpeggiator plugins in one.



Except the default “arpeggiator” mode, there are:

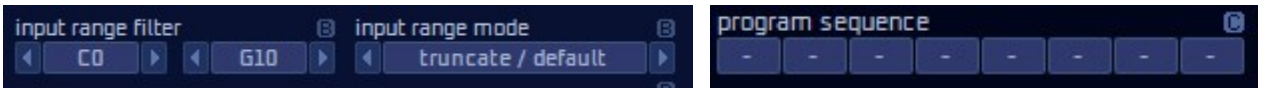
“arp, fixed input” – the same as “arpeggiator” but the input keys are pre-defined per program, for step sequencer-like use cases.
“drum sequencer” – for sequencing drum patterns.
“guitar strum” – for sequencing guitar strumming and picking pattrerns.

For more details on Drum and Guitar modes, check the chapters “Drum sequencer mode“ on page 43 and “Guitar strum mode” on page 48.

Interface

Elements and Navigation

The main GUI element is a «value box», either surrounded by arrow buttons or not:



There are several ways to adjust the value:

- left-click and hold on the box, drag it up or down;
- place the pointer over the box, use mouse wheel to adjust the value;
- click buttons to adjust the value;
- right-click on the box and select value from the popup menu

B / P / C marks next to control tell whether this particular parameter is saved with a bank **(B),** program **(P)** or chain **(C)**. Global settings are stored in BlueARP.ini file and marked as **(G)**.

When you switch programs, (**B)** or bank-related parameters stay the same.

(C) or chain-related parameters are dependent on «current chain» setting (chains are described at page 21).

Almost all editable parameters have right-click popup menus, they mostly contain list of values to select from or actions to perform (copy, paste, etc.), for example

or

Main window layout

Here are brief descriptions of GUI blocks. For more info, go to the respective chapters.

(1) **Top panel** contains arp mode, midi in channel and midi out channel. All are bank-related (B). So, when you switch programs, these settings remain the same;

(2) **Left panel** has 2 pages – ARP / MAIN and SETTINGS. ARP / MAIN page contains all step-independent arpeggiator settings like number of steps, synchronization, key sort order etc. Some are bank-related (B), some are program-related (P). SETTINGS page has midi filtering options, GUI settings and some other rarely changed stuff;

(3) **Program browser** is there to select programs and to rename them;

(4) **Main menu block** has MENU button (calls drop-down menu), page selector (for patterns longer than 16 steps), cyclic pattern shifts buttons and LEDs indicating which page is currently playing and which is being edited;

(5) **Matrix editor** represents step-related values for the selected value lane;

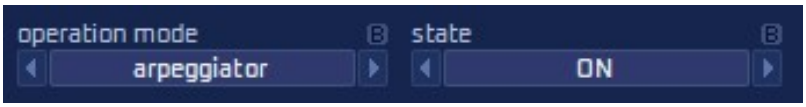
(6) **Value lanes** contain step-dependent pattern parameters. To select a lane, click on its caption. To adjust the value, drag the «value box» up and down or use mouse wheel;

(7) **Chains** allow you to chain several programs into one continuous sequence. «Current chain» parameter switches the chain, it can be automated;

(8) **Info panel** - information on current position, beat, input and output keys;

Block (1): TOP PANEL

Operation mode and State

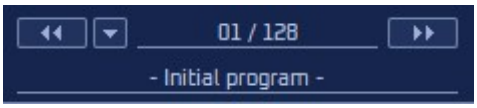


operation mode	Switching between several operation modes
values	*arpeggiator, arp. fixed input, drum sequencer, guitar strum*
comments	*arpeggiator* – default mode *drum sequencer* – some of the functions and GUI elements change to sequence drums, see page 43 for details

Warning. Switching the operation mode will erase all your current programs, it is like re-loading the plugin with the different settings (cause operation modes have different parameter mappings). So, normally you should do this only once when adding BlueARP plugin to your project. The only exception is switching between the "arpeggiator" and "arp. fixed input" modes, it will keep all the programs intact, so you can switch it back and forth whenever you need it.

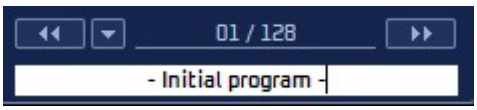
state	arp state
values	*OFF, ON, THRU, 2k auto*, 3k auto**
comments	*OFF* – switched off, ignores incoming MIDI; *ON* – ARP is active; *THRU* – pass thru for MIDI noted with respect to input range filter; *2k auto on/off* – ON when 2 or more keys pressed, OFF otherwise; *2k auto on/thru* – ON when 2 or more keys pressed, THRU otherwise; *3k auto on/off* – ON when 3 or more keys pressed, OFF otherwise; *3k auto on/thru* – ON when 3 or more keys pressed, THRU otherwise;

Program browser

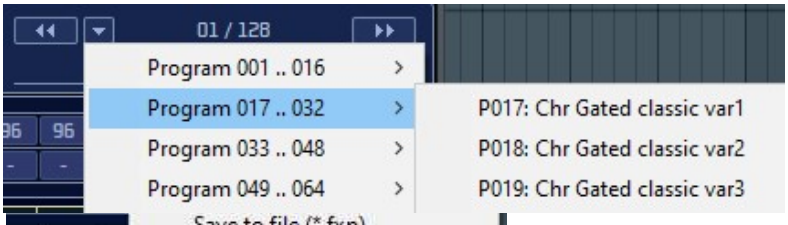


Use buttons to navigate through the programs in a current bank. Alternatively, click on the program number and drag up or down to change it (just like with the other value boxes).

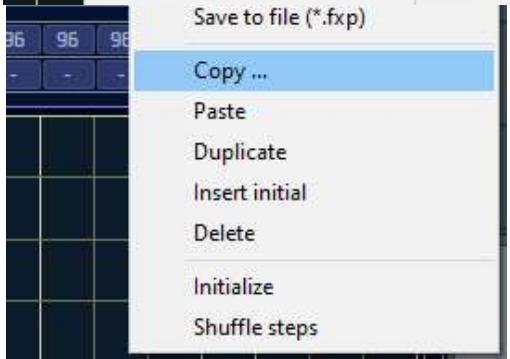
Bank contains 128 programs, so you can configure up to 128 arpeggiator patterns, they will be all saved with your project file.



To change program name, click on it, type in new name and hit enter or click somewhere outside this area.



Click drop-down button to select a program from the list:



Right click on program number label to access actions menu:

Here you can quickly copy, paste and duplicate programs, i.e. when you are experimenting and don't want to lose the

original pattern. It duplicates the «Program» submenu from the main menu, accessible via MENU button.

Note: when you duplicate, delete of insert the program, all the rest of the programs are shifted left of right and their numbers are changed, but BlueARP takes care of that and adjusts program numbers in chains, so your chains won't be spoiled.

Block (2): LEFT PANEL

Left panel has 2 pages – ARP/MAIN and SETTINGS, click these buttons to switch:

SETTINGS page contains rarely used bank-related settings, described on page 31 (you don't need to change them often, so they were moved them to a separate page to save main GUI space).

ARP/ MAIN is a primary page, it is divided into 4 blocks – «INPUT FILTER», «ARP ENGINE», «OUTPUT FILTER» and «CHAINS». Their controls are described in the following chapters.

> *In this manual, we use the word «keys» to represent what's pressed on the MIDI keyboard, while «notes» is what comes from the arpeggiator output.*

In general, left panel represents all program-related and bank-related parameters, except the pattern itself. Program-related params have **(P)** mark; they may vary from program to program (for example, number of steps or gate time). Bank-related params have **(B)** mark; they are the same for all the programs in a given bank. For example, «input range filter» is bank-related, no need to set it for each program individually and it won't change as you switch programs.

Input filter

INPUT FILTER processes the input key list before it enters the arpeggiator «core» engine. We have «Input keys post-filter» key list at the output of this block.

These keys go further into «Arp Engine» block.

input range filter	range for filtering out input notes
values	*C0 .. G10 (MIDI notes 0 .. 127)*
comments	Change it if you want this instance of BlueARP to react to MIDI keys only within a given range. All notes outside this range will be ignored. You will need this if you want to create keyboard-split performance with several instances of BlueARP. BlueARP can also pass outside-the-range notes non-arpeggiated, it's controlled by «*input range mode*» setting.

Hint. Right-click value box and select «press MIDI key...» to set the value from your MIDI keyboard.

input range mode	modifies «input range filter» behavior
values	*truncate (default), pass thru (no arping)*
comments	Sets the behavior of «*input range filter*» setting. In «*pass thru (no arping)*» mode, keys outside the range will be passed to the output non-arpeggiated.

order algorithm	ordering (sorting) algorithm for input keys
values	*by pitch, by pitch desc, as played, as played desc, by velocity, by velocity desc, chord (normalized), chord (as played)*
comments	Default setting is «by pitch» - pressed keys come into the arp engine in natural order, from left to right on the keyboard. It also means that «k1» in «KEY SELECT» lane will be the lowest key. Sometimes it's not the best way to order pressed keys. For example, if you play 1-key bass line, it's better to set order algorithm to «as played, desc». In this case «k1» will always be the last pressed key. «chord (normalized)» can be explained by example. You press C4+E4, Cmaj chord is detected. Ordered list will be C4+E4+G4 (complete Cmaj chord). If you play inverted Cmaj – G3+C4+E4, output will be the same, because chord is normalized. «chord (as played)» behaves the same way, but inverted chord will stay inverted.

input range (wrap)	range for input key «wrap-around»
values	*C0 .. G10 (MIDI notes 0 .. 127)*
comments	Unlike «input range filter», this one won't ignore notes outside the range, but will fit them into the given range by applying up or down octave transposition. Assume your set this range to C3...C4. When you press keys **A2**, C3, E3, G3, **D4**, the processed keys will be **A3**, C3, E3, G3, **D3** (bold notes were wrapped into the range C3...C4). It's sonically useful when you play chords all over the keyboard, but want your bass line to sound right, not too low or too high.

missing keys substitution	missing keys substitution algorithm
values	*don't play, cyclic, first key, last key, fixed key*
comments	When your pattern has more keys than you actually play, this setting will determine whether to mute these steps (*don't play*) or substitute missing keys with the existing ones. For example, you hold C5 and E5, while «KEY SELECT» lane has steps with «k1», «k2», «k3» and «k4». Info panel will show input keys pre-filter (before substitution) as «C5, E5, -, -, -». Key list post-filter (after substitution) will be, depending on this setting: • *don't play* «C5, E5, -, -, -» • *cyclic* «C5, E5, C5, E5, C5» • *first key* «C5, E5, C5, C5, C5» • *last key* «C5, E5, E5, E5, E5» • *fixed key* «C5, E5, G5, G5, G5» («fixed key» = G5)

missing keys transpose	additional transpose for substituted keys
values	*none, -1 octave, +1 octave*
comments	Adds additional transposition for substituted missing keys. For the example above, if we set missing keys transpose to +1 octave, post-filter key list will be: • *don't play* «C5, E5, -, -, -» • *cyclic* «C5, E5, C6, E6, C6» • *first key* «C5, E5, C6, C6, C6» • *last key* «C5, E5, E6, E6, E6» • *fixed key* «C5, E5, G6, G6, G6» («fixed key» = G5)

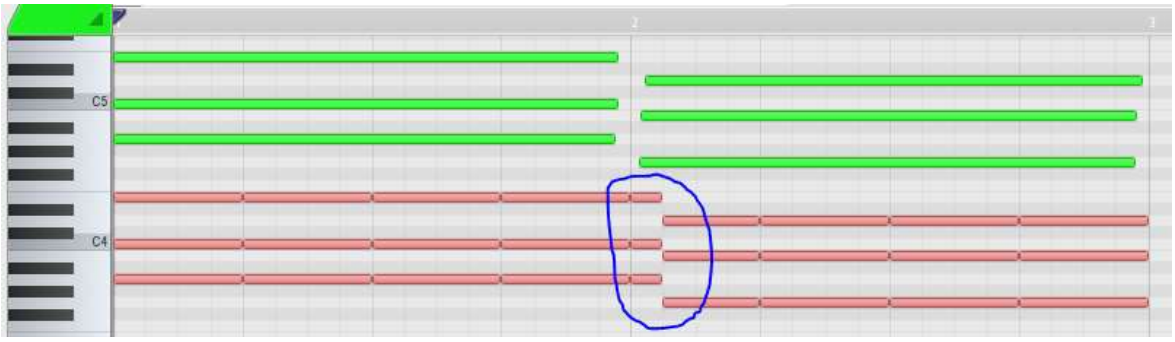
in quantize	input keys real-time quantization
values	*none, 1/16, 1/12, 1/8, 1/6, 1/4, 1/2, 1 bar, 2 bars*
Comments	Values are fractions of a bar (1/16 means 16th notes, 1/4 corresponds to 1 beat). For example, at value 1/4 BlueARP will capture pressed keys on the start of each beat.

Hint. When input quantize is on, you should press keys a little beforehand, because input keys need to be already captured when the next step/beat starts.

q. tolerance	Input quantize tolerance
values	*none, =sync, 1/64 .. 1/4*
comments	Input quantize tolerance. Useful for live playing, especially in Guitar mode. Sets the maximum time interval (in fractions of a bar), by which the input keys can come later than the quantization measure (defined by the "in quantize" setting). For such late notes, arp engine will take them at the end of the additional "q. tolerance" interval. Special value "sync" will take this interval from the "sync" parameter. See the example below to get the idea.

Example. Let's assume we play 1-bar long chords live (green notes below) and the arp generates the repeated chords each beat (pink notes); "in quantize" is set to 1/4, "q. tolerance" is "none". On the second bar input keys come late and since "in quantize" is the whole beats, this change gets into the arpeggiated noted only on the next beat:

Now let's change "q. tolerance" to 1/16 and check what changes, the same input notes:



In this case, we have late input notes, coming right after the beat (quantization measure), but not later than 1/16 (quantize tolerance). They will be taken into the arp right at out beat + 1/16, so this change will still be quantized.

arp. latch	Latch (or hold) pattern
values	*On, Off (checkbox)*
comments	When checked, BlueARP will continue to play pattern for the last pressed chord even after all input keys are released, until another key is pressed. For live performances it may be useful to assign "arp.latch" to sustain pedal, or to switch it off to free your hands from the keyboard to do some other stuff.

notes off	All notes off (button)
Values	-
comments	Works like ‘PANIC’ button in DAWs, will clear the output note buffer and send ‘All notes OFF’ midi message.

ARP Engine

ARP ENGINE takes post-filter key list from the input filter (after fitting to range, missing keys substitution, quantize, etc.) and generates note pattern at the output, referring to MIDI clock and current song position from the Host.

steps	number of steps for current program
Values	*0 .. 64*
comments	Default value is 16. You may also experiment with irregular values like 15 or 17; it will make the pattern sound less predictable which is sometimes sonically useful. ***steps = 0 and 1*** are special modes, in this case BlueARP works as a MIDI thru (0 – simple thru, 1 – quantized thru). The purpose is to use this «MIDI thru dummy» program in chains to switch between «arpeggiated» and «midi thru» scenes.

sync	Step length (as a fraction of a bar)
values	*1/64, 1/48, 1/32, 1/24, 1/16, 1/12, 1/8, 1/6, 1/4, 3/64, 3/32, 3/16, 3/8*
comments	Default value is 1/16, it means 1 step = 16th note. 1/12 is «8th triplets» or «16th dotted».

gate time	note length, relative to step
values	*1% .. 125%*
comments	Sets generated note length as a fraction of a step length.

swing	swing control
values	*-50% .. 50%*
comments	Sets relative time shift for even steps as a fraction of a step length (assuming step numbers start from 1). For example, swing = 33% means that each even step will be delayed for 33% of the step length. For negative values, it will start earlier.

restart on	pattern restart trigger
values	*beat 0, key, 1st key, play*
comments	In default «beat 0» mode step number is always aligned to the song position given by host. When your song or pattern restarts in a DAW, BlueARP pattern will also restart. «play» mode is the same but aligned to playback start position. With «key» setting, BlueARP will restart pattern each time new key/chord is pressed. In «1st key» mode pattern will start with the first key/chord pressed and will keep going until you restart playback in a DAW.

fixed key	Fixed key value
values	*C0 .. G10 (MIDI notes 0 .. 127)*
comments	In «KEY SELECT» lane, you can set any step to «Fixed», it tells BlueARP to ignore input keys and take «fixed key» value. Set all steps to «Fixed» to use BlueARP as a step sequencer.

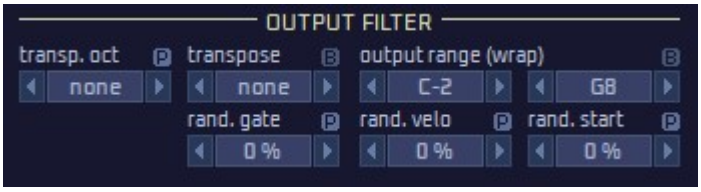
output note velocity	Sets where to take velocity for generated notes
values	*velocity lane, input key, lane + input key*
comments	«lane + input key»: BlueARP takes output note velocity from VELOCITY lane and adjusts it to input note velocity (multiplying and normalizing them)

force to scale: root key	root key for «force to scale» mode
values	*off/key1, detect from chord, C, C#, D ... Bb, B*
comments	Works together with «force to scale: scale» parameter. You can either set a fixed root for a selected scale or let BlueARP detect it dynamically from the chord you play. BlueARP recognizes basic chords and chord inversions, so if you press (E4, A4, C5 - Am inverted), your root key will be **A**.

force to scale: scale	Sets scale key for «force to scale» mode. Works together with «force to scale: root key» parameter
values	*off/chromatic, detect from chord, Major, minor, harmonic minor, melodic minor, pentatonic Major, pentatonic minor, pentatonic neutral, pentatonic blues"*
comments	If you set anything except «off/chromatic», two things will happen: 1. BlueARP will fit output notes to the given scale (either all or only semi-transposed notes, depending on «force to scale: mode» parameter); 2. «SCALE STEP» lane will transpose notes in scale steps. Say if your scale is C Major, you pressed **D4** and scale step=+1, the output note will be **E4**. With «off/chromatic» selected, «SCALE STEP» will work as a semitone transposition. With «detect from chord» selected, BlueARP will derive scale from a chord you play. From minor/major chords it will derive minor/major scales, for other chords like sus2, sus4 etc., BlueARP will try to derive an altered minor/major scale which will fit the given chord (for version 2.3.8 this feature is experimental).

force to scale: mode	how to apply semitone transposition
values	*all keys, semi-transposed*
comments	Works together with "force to scale: scale" parameter. When set to *semi-transposed*, force to scale will not be applied to the steps with SCALE STEP = “-“ (zero). This way you can still play out-of-scale notes.

Output filter



OUTPUT FILTER performs some post-processing of generated notes – octave / semitone transposition, wrapping notes to fit the given range, applying randomization.

transp. oct	output transposition, octaves
values	*-3 oct .. +3 oct*
comments	It is program-related.

transpose	output transposition, semitones
values	*-12 .. +12*
comments	It is bank-related, because there's no sense to make this setting different for different programs.

output range (wrap)	Range for output notes (wrapping)
values	*C0 .. G10 (MIDI notes 0 .. 127)*
comments	Notes outside the range will be wrapped (octave-transposed up or down to fit the range). Works just like «input range (wrap)», but for the output notes.

rand. gate	randomize output note gate time
values	*0% .. 100%*
comments	Add random value (both positive and negative) to generated note length.

rand. velo	randomize output note velocity
values	*0% .. 100%*
comments	Add random value (positive or negative) to each output note velocity.

rand. start	randomize output note start time
values	*0% .. 100%*
comments	Add random value (positive only) to generated note start time.

Chains

Relates to "**Block (7) Chains**" panel.

num. of chains	sets maximum value for «current chain» parameter
values	*1 .. 16*
comments	To switch chains with a midi controller, you need to automate «current chain» parameter. If you use a knob for this, setting «num. of chains» to the appropriate value will utilize full rotation range of this knob.

chain quantize	input quantization for chain switching
values	*none, 1/16, 1/12, 1/8, 1/6, 1/4, 1/2, 1 bar, 2 bars*
comments	When you switch chains, for better transition it should be done strictly at the start of a new beat. Chain quantize = 1/4 does exactly that and it is the default setting.

send bank message	selects bank/patch change MIDI message format
values	*ctrl 0, ctrl 32, ctrl 0+32*
comments	Relevant for controlling hardware synths, some VST synths will also react to this message. Sylenth1 does, for example. When you switch chains, BlueARP may send program/bank change to its MIDI output if «bank num» and «patch num» parameters are not empty. Hardware synths use different bank change message formats. If the default one doesn't work for you (synth doesn't switch banks, only patches), try other options.

restart chain on switch	restart chain from the beginning after chain switch
values	*On, Off*
comments	When checked, chain always starts from the beginning after chain switch (otherwise, in restart on «beat 0» mode, chain step is calculated from song position given by host)

Block (2): SETTINGS

Hit SETTINGS button on the left panel to call this page.

This page contains rarely used bank-related and global settings. They were moved to a separate page to save GUI space on the main panel. You don't need to change them often.

MIDI ROUTING block has MIDI input and output channel, moved here to save GUI space.

MIDI FILTERS block contains settings for MIDI message filtering (*).

MATRIX EDITOR block adjusts matrix editor behavior.

GUI block has some GUI-related settings.

(*) Some of these settings are not supported in VST3 version, see «VST3 compatibility» in «FAQ / Troubleshooting» section on page 55.

midi in channel	input MIDI channel
values	*all, 1 .. 16*
comments	*all* – BlueARP will take MIDI input from all MIDI channels, *1 .. 16* – only from a given channel.

midi out channel	output MIDI channel
values	*1 .. 16*
comments	Default setting is 1, because soft synths usually don't care about MIDI channel. You may need it if you have multi-timbral hardware synth connected to BlueARP or several hardware synths chained on one MIDI output port, separated by MIDI channels.

midi thru mode	optional MIDI thru
values	*disabled, all channels, except midi in ch, only midi in ch*
comments	«*all channels*» - BlueARP will pass thru all input notes to the output (for all midi channels), along with the generated notes. May be useful in apps like MainStage on Mac, where you can't run MIDI FX plugins in parallel and can only chain them. «*except midi in ch*» - all channels except midi in channel will be re-routed to the output (keeping the same midi channels); midi in channel should not be set to «any». «*only midi in ch*» - input notes from midi in channel will be copied to the output, in addition to arpeggiator-generated notes.

prog.change msg	how to respond to incoming Program Change MIDI message
values	*ignore, set own program, pass thru*
comments	«set own program»: BlueARP will set its internal program in response to Program CC message. «pass thru»: BlueARP will pass this message to its MIDI out (= to VST plugin it is connected to).

pitch bend msg	how to respond to incoming Pitch Bend MIDI message
values	*ignore, pass thru*
comments	«pass thru»: BlueARP will pass this message to its MIDI out (= to VST plugin it is connected to).

mod wheel msg	how to respond to Modulation Wheel MIDI message
values	*ignore, pass thru*
comments	«pass thru» - BlueARP will pass this message to its MIDI out.

aftertouch msg	how to respond to incoming aftertouch MIDI message
values	*ignore, pass thru*
comments	«pass thru»: BlueARP will pass this message to its MIDI out.

sustain msg	how to respond to sustain MIDI message (CC 64)
values	*ignore, pass thru, sustain, arp latch*
comments	«pass thru»: BlueARP will pass this message to its MIDI out. «sustain»: BlueARP will sustain input notes in a normal way, just like any other synth would do «arp latch»: sustain message is linked to «arp latch» parameter, with respect to «sustain polarity» value.

sustain polarity	sustain pedal polarity for «sustain msg» setting
values	*normally low (-), normally high (+)*
comments	Normally low (-) means that in released state it should be value 0.

other CC msg	sets how to respond to incoming CC MIDI messages
values	*ignore, pass thru*
comments	The same as other MIDI filters but applies to all other CC messages not mentioned before. *This setting is not working in VST3 version due to VST3 limitations*

scale step range	sets value span for SCALE STEP lane
values	*«-12...+12», «0...+12», «-7...+7», «0...+7»*
comments	Default value is «-12...+12». For touch-screens it may be better to set «0...+12», «-7...+7» or «0...+7» for easier adjustment.

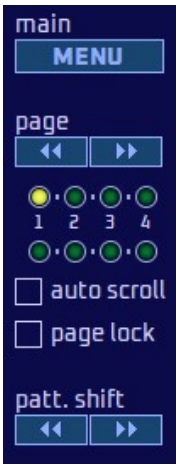
color scheme	sets skin / color theme
values	*default (blue) and others*
comments	Color schemes are stored in *.ini files in \skins sub-directory. On windows it is in plugin directory, on Mac – inside the bundle). Selected color scheme index is stored in BlueARP.ini file in user directory: **Windows**: C:\Users\<user>\AppData\Roaming\BlueARP **OSX**: c:/Users/<username>/Library/Application Support/BlueARP

Hint. When you load BlueARP for the first time, it will create this directory and BlueARP.ini inside it. Ini file it was placed here because plugin directory doesn't usually grant write permission to the plugin.

size / scaling	sets GUI size
values	*100%, 125%, 150%, 200%*
comments	Adjusts GUI size.

octave numbering	sets one of note naming conventions
values	*«C-2 .. G8 (mid C3)», «C-1 .. G9 (mid C4)», «C0 .. G10 (mid C5)»*
comments	It tells BlueARP how to display notes or which key is the middle - C3, C4 or C5. It doesn't affect the functioning, just the way note names are displayed.

Block (4): MAIN MENU and pattern controls



MENU button calls drop-down menu with Bank load/save, Program load/save and some other functions.

page buttons are necessary when you pattern is longer than 16 steps, so it doesn't fit single screen. There are 2 small LED lanes underneath, upper one shows the selected page (page being edited), lower one – page being played.

auto scroll checkbox: when checked, matrix will always show the page actually playing.

page lock checkbox: when checked, current page will cycle over and over until unchecked (useful for programming long patterns to prevent pages from switching while you edit).

Pattern shift buttons perform cyclic step shifting (pattern rotation). It's useful when your pattern doesn't match the beat and you try to align it. The shift is cyclic, so when you shift the patter right, the last step won't disappear but will «jump» to the beginning (this is why it is also called pattern rotation).

Main menu includes the following items:

Bank	bank contains entire BlueARP state, except global (G) settings
Load from file (*.fxb)	Load bank from file, current state will be overwritten
Save to file (*.fxb)	Save bank to file
Initialize	Initialize all programs in a current bank

Program	load, save and copy/paste programs
Load from file (*.fxp)	Load program from file, current program will be overwritten
Save to file (*.fxp)	Save current program to file
Copy …	Memorize the current program for «Paste» operation.
Paste	Paste program at a current location («Copy …» should be done before). Paste overwrites the target program.
Duplicate	Duplicate program at the current location: inserts empty slot after the current program and copies it there.
Insert initial	Insert initial program at the current location. Current program and all the rest will be shifted right to make space for the new program, the last (128^{th}) program will be lost.
Delete	Delete current program. The remaining programs will be shifted to the left to fill the gap.
Initialize	Initialize the current program
Shuffle steps	Randomly shuffle steps in the current program

Chain	copy/paste chains
Copy …	Memorize current chain as a source for copy/paste operation.
Paste	Paste chain at the current location («Copy …» should be done before). Paste overwrites target chain.
Duplicate	Duplicate the current chain: insert empty chain after this one and then copy current chain to the next one.
Insert initial	Insert initial chain at the current location, shifting the rest to the right.
Delete	Delete current chain.
Initialize	Clear current chain data
Initialize all chains	Clear all chains data

Chain variation	copy/paste chain variations
Copy …	Memorize current chain var. as a source for copy/paste operation.
Paste	Paste chain var. at the current location («Copy …» should be done before). Paste overwrites target chain variation.
Duplicate	Duplicate the current chain variation: insert empty chain variation after this one and then copy current chain var. onto the next one.
Insert initial	Insert initial chain var. at the current slot, shifting the rest to the right.
Delete	Delete current chain variation.
Initialize	Clear current chain variation data
Initialize all chains	Clear all chain variations for this chain

Page	copy/paste chains
Copy …	Memorize current pattern page as a source for copy/paste operation.
Paste	Paste pattern page at a current location ("Copy …" should be done before). Paste overwrites the target chain.
Initialize	Initialize all steps for the pattern page.

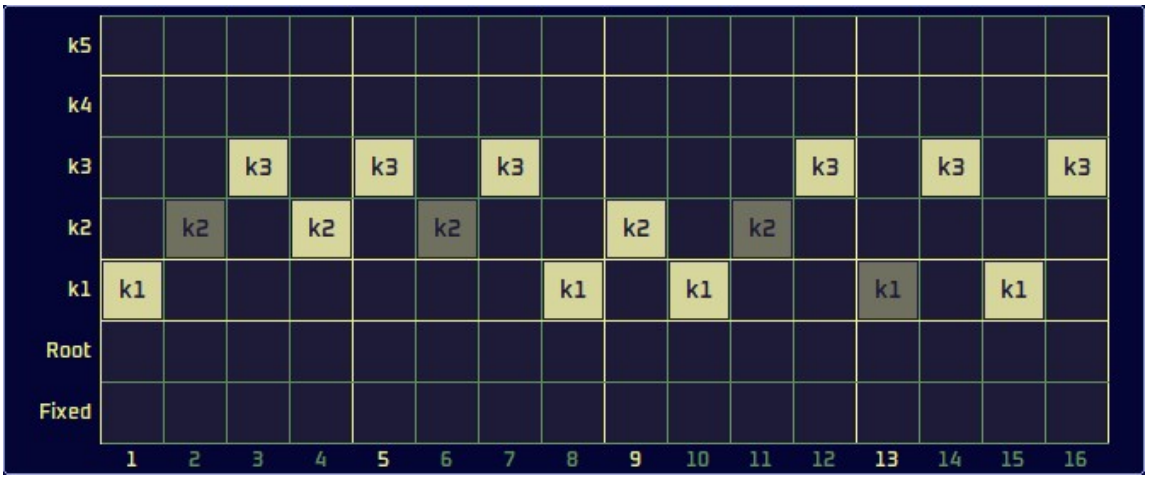
Debug info	various information
Open BlueARP.ini location	Opens BlueARP.ini file location. BlueARP.ini holds global settings like GUI scale, GUI skin index, octave numbering.
pGraphics->GetGUIAPI()	Show selected GUI API
pGraphics->PluginPath()	Show path to the plugin file
pGraphics->HostPath()	Show path to host application
ArpEngine->PatchVer_loaded	Show currently loaded bank format index

Open Manual (pdf)	available versions of the manual
English (EN)	Opens "BlueARP_Manual_vNNN_EN.pdf" file, where NNN stands for version. File should be located in the same folder as the plugin; it is included into the .zip installation package.
Other languages	Opens the manual in another language by a direct web link

Make a donation	link to support BlueARP development with a donation

Developer's website	link to developer's website with the latest updates for BlueARP

Block (5): MATRIX EDITOR



Matrix editor allows you to adjust current value lane values in a friendlier graphic way.

You can adjust step-related values either in a matrix editor or on a value lane itself.

On the matrix editor, there are several ways to change values:

- Click the cell to set the single value;
- Drag the mouse from left to right set the steps to a certain value, as you drag;
- Right click on a matrix sell and select «Set all steps to ***», where *** is the desired value, it will set all steps in a program, including those on other pages;

Grayed-out bricks mean that this particular setting doesn't affect the generated pattern. On the picture above, step 2 is set to Off, so «key select» value for this step doesn't make any difference.

Block (6): VALUE LANES

Value lanes contain step-related pattern parameters. Selected value lane is also shown in the MATRIX EDITOR. To select it, click the lane caption.

To adjust the value for a certain step:

- Click on it and drag up or down;
- Put the mouse over the box and use mouse wheel to adjust the value;
- Right-click on the box and select the value from the popup menu;

Yellow indicator (or) next to «OCTAVE» and «KEY SELECT» labels switch the lane between **monophonic** and **polyphonic** mode. In polyphonic mode, you can set several values at once in matrix editor.

«GATE TIME» can be switched to «CHANNEL» or «CHANCE» mode via 1 indicator:

See descriptions for each value lane below.

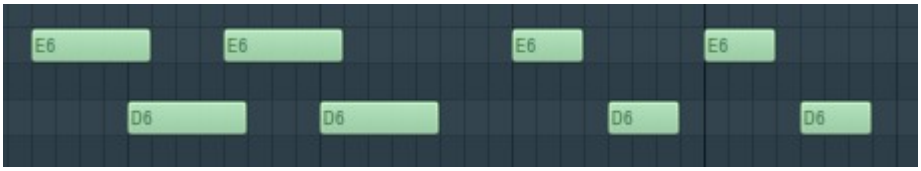
VELOCITY	Velocity value for each step
values	*0 .. 127*
comments	Default value is 96. Use it to set velocity accent for certain steps. VELOCITY values will be ignored, if you set "output note velocity" = "input key" in MENU >> Settings.

GATE TIME (mode 1)	Gate time multiplier for each step
Values	*1/16, 1/8, 1/4, 1/2, -, 2, 4x, 8x, 16x*
Comments	Multiplies gate time by a given value. "-" means no change (default value). For example, with gate = 60% and GATE TIME for a step = "2x" note length for this step will be 60% * 2 = 120% or 1.2 steps.

CHANNEL (mode 2)	Modify output MIDI channel for each step
Values	*-, 1 .. 16*
comments	When not "-", output midi channel will be changed to the specified value for a step. Use this with multi-timbral synths to create complex textures/arpeggios with different sounds for various steps.

CHANCE (mode 3)	Note trigger changce for a given chance
Values	*1% .. 100%*
comments	Defines note trigger chance on a pre-step basis. For 'Nrm' steps, decreasing the chance will attract the step to 'Off'; for 'Rst' - to 'Nrm', for 'Tie' - to 'Rst'; for other step types - to 'Off'.

STEP TYPE	Several options for output note generation
values	*Off – this step doesn't generate any note* *Nrm – Normal(default) – generates a note;* *Rst – this step will play the Rest of the previous step;* *Tie – this note will overlap with the previous one (for glides);* *Chr – Chord, or triggering all notes at once* *Rnd – Random, picks up random key from input key list*
comments	«Rst» step means that this step continues the note from the previous step. You may chain several «Rst» steps together to make longer notes. «Tie» option may be tricky and not self-describing. Its main purpose is to create «glides» between notes. But it requires configuring synth properly – set it to monophonic mode, with legato and portamento on. In this case, when you press keys with overlapping (press key1, press key2, release key1), sound pitch will glide between the notes, but not when you press them with gaps (see picture below). When you configure the synth this way, «Tie» steps will create glides.



«Tie» steps «Nrm» steps

KEY SELECT	Input key selection for the given step
values	*Fixed – use fixed key from Arp Engine settings* *Root – root key from detected chord, key1 if no chord detected* *k1..k5 – take keys №1..5 from key list (post-filter)*
comments	Tells which key to take from «post-filter key list» for the current step. Yellow label next to KEY SELECT caption (or) toggles between monophonic and polyphonic mode. In **monophonic** mode you can only select one key for a step or all keys at once with STEP TYPE = Chord. In **polyphonic** mode you can select several keys at once, like k1+k2 or k1+k3.

Hint. Fixed key doesn't depend on pressed keys, so you can set all steps to «fixed» and use BlueARP as a step sequencer; or set some steps to «fixed» to create variations.

SCALE STEP	Semitone/Scale step transposition for each step
values	*-12 .. +12*
comments	Depends on «force to scale: scale» parameter. When the latter is «off/chromatic», this will work as a semitone transposition. Otherwise, it will transpose output note with respect to the selected scale.

OCTAVE	Octave transposition for each step
values	*-3, -2, -1, 0, +1, +2, +3*
comments	It's convenient for bass lines, where the steps are usually transposed for the whole octaves. Yellow label next to OCTAVE caption (or) toggles between monophonic and polyphonic mode. In **monophonic** mode all keys for a given step are transposed by octaves. In **polyphonic** mode only key 1 is transposed. So, if you have STEP TYPE = Chord, OCTAVE = -1; 0 and press F4 + A4, output notes will be F3 + F4 + A4. (key1 = F4 is copied down an octave, but not key2 = A4)

Block (7): CHAINS

Chains deliver the possibility to chain several programs (patterns) together into a longer «super-pattern». It was implemented with live performances in mind. White chains should be switched manually, chain variations are kind of «sub-chains» that are triggered automatically, according to pre-programmed conditions like the chords you play, keys you press, etc.

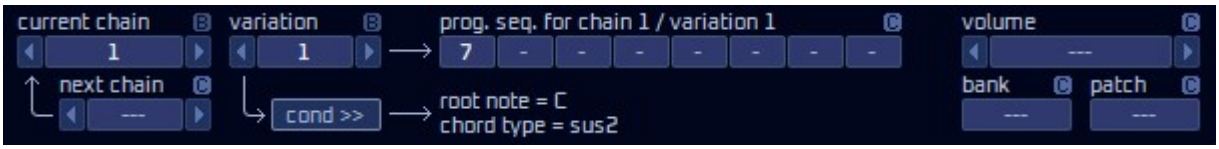
All controls to the right of the «variation» box are driven by current chain + variation: when either chain or variation switches, it will bright up another program sequence.

Chain variation «---» is the default variation or the chain itself. You may not use variations and use just chains; in this case you don't need to change chain variation box.

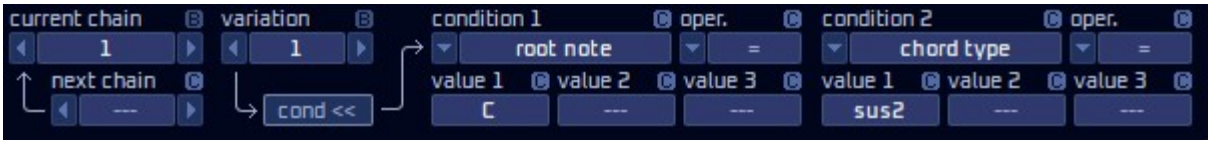
Dots and numbers below chain variation box show if any of the variations are used (non-empty):

Click the dot to jump to that variation (the same as adjusting «variation» box).

When you jump to the variation, you can program a different program sequence for that variation"



Press «cond >>» button to change trigger condition for this variation. In this example, variation 1 of chain 1 will be triggered when «Csus2» chord is detected:



Condition 1 and **condition 2** can be one of the following:

- *In.keys*: input keys post-filter (with respect to latch and quantization), you can see them on the info panel at the bottom;
- *In.notes*: the same as «input keys», but without the octave number (i.e. «A» instead of «A4»);
- *Root note*: root of the detected chord, when just one key pressed and no chord detected, root note will be equal to that pressed key;
- *Chord type*: detected chord type (like «m», «Maj», «sus2», etc.);
- *Simp. chord*: simple chords (only minor and Major chords for all notes, 24 in total);
- *Num. in keys*: number of input keys (with respect to latch and quantization);
- *k1 .. k3 (by pitch)*: respective key in the input list post-filter;
- *k1 .. k3 (note)*: the same as previous, bit without an octave number (i.e. «A» instead of «A4»);
- *lowest velocity*: lowest velocity value of all pressed keys;
- *highest velocity*: highest velocity value of all pressed keys;
- *aftertouch*: last received aftertouch value$

Operator and **value 1..3** fields define the rest of the equation, which will be checked against the entity selected in the «condition» field. Operator can be equal, not equal, greater or equal, less or equal, in range or not in range. Number of available «value» fields depends on the selected operator. For «equal», all three value fields are available, meaning «equal to any of these values». For «not equal» it means not equal to all of the given values.

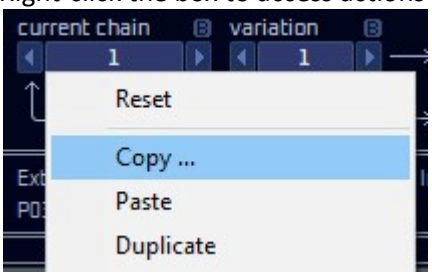
NOTE: Chain variations may have overlapping conditions; in this case the last variation that meets the criteria will be triggered. For example, you can have *root note = E* for variation 1 and *root note = E* plus *highest velocity >= 112* for variation 2, in this case pressing any E chord will trigger the chain variation 1, but when pressed harder, it will trigger variation 2. For example, it can be 2 variations of a drum fill.

Program sequence lane holds numbers of chained programs for a current chain + variation.

Right-click program sequence slot to select program for a particular chain/variation step:

current chain	current chain
values	*---, 1, 2, … number of chains (up to 16)*
comments	Current chain parameter can be automated; its maximum value is set by "num. chains" parameter on the left panel.

Hint. Pay attention to «restart chain on switch» setting on the left panel. When On, switched chain will always start from the beginning (1st step of the program sequence).
Right-click the box to access actions menu to copy/paste/duplicate chains, etc.:



variation	current chain variation
values	*---, 1, 2, 3*
comments	Adjust it to select chain variation for editing. During live performance, it is not meant to be automated / changed manually; rather you should define conditions on which it will trigger (like the chord or a certain key). When the conditions overlap for 2 or more variations within a chain, the first possible chain variation will trigger. Once no conditions are met, the default variation (or the chain itself) will trigger.

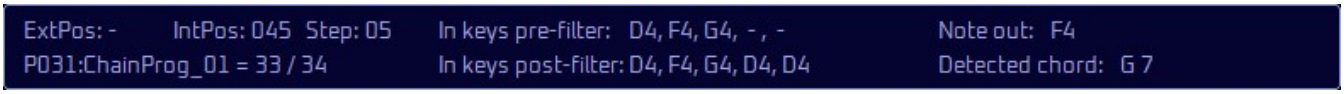
Hint. Right-click the box to access actions menu to copy/paste/duplicate chains, etc.:
(popup menu is the same way for chains and chain variations)

next chain	next chain auto-switch
values	*---, caller, caller-1, caller+1, chain 1, … chain 16*
comments	Allows you to automatically jump to another chain after current chain plays once. The options include: «caller» - switch back to the chain it was invoked from; «caller-1», «caller+1» - the same, but with the shift to the «caller» chain; «chain 1» … «chain 16» - switch to particular chain after this chain ends;

patch num, bank num	send bank\program change on chain switch
values	*---, 0 … 127*
comments	If specified, BlueARP will send program\bank change midi message to the connected synth each time current chain is changed (with respect to chain quantize).

volume	send volume change when chain switches
values	*---, 0 … 127*
comments	As previous, BlueARP will send volume change MIDI message to the connected synth each time current chain is changed.

Block (8): INFO PANEL



Shows current beat, step and some other information:

- **ExtPos**: song position, reported by host. For *restart on* = *beat 0*, it is used as a reference for step position;
- **IntPos**: internal song position (with respect to looping);
- **In keys pre-filter** - input keys, as they are pressed;
- **In keys post-filter** - input keys after «input filter» - truncated and wrapped to fit the given range, ordered, with missing keys substituted, quantized. This is what goes into the BlueARP «core» engine;
- **Note out** – generated notes;
- **Detected chord**: shows root key + chord type

Hint. Lower left label «P031: ChainProg_01 = 33 / 34» gives information about last changed parameter and associated value. First number «33» represents internal value, second number «34» – corresponding midi CC value. May be useful for automation with external controllers: for example, for checkboxes internal values will normally be 1 for On and 0 for Off, while the corresponding MIDI CC values will be 64 for On and 0 for Off. The be more precise, MIDI CC value range 0..63 will be interpreted as Off and 64..127 – as On.

Drum sequencer mode

Overview

Since version 2.5.0, BlueARP has the «drum sequencer» operation mode in addition to the default «arpeggiator» mode. It was designed mostly for the BlueARP DM (hardware BlueARP counterpart), but may be useful with the plugin as well, cause drum sequences:

- Can be chained and automated the same way, using chains and variations;
- Have some probability and randomization functions;
- Make it possible to make a full arranger-like performance driven by several BlueARPs.

To switch to the drum mode, change «operation mode» parameter one the top left:

Important note: Current bank with all its programs and chains will be initialized for the drum mode; there will be no warning dialog, so make sure to save your previous bank if needed.

LEFT PANEL

Input filter

In Drum mode, most of the settings from the ARP mode are not relevant, since generated notes don't depend on the input notes. Input range, order and missing key settings make sense only of you use special k1..k5 drum notes for the steps, which is rarely the case.

Here "in quantize" and "q. tolerance" settings work the same as in Arp mode. Although it won't affect the output notes, it will affect the start time / alignment with the beat, especially when combined with "restart on" = "key".

ARP Engine

In ARP ENGINE block, «force to scale*» params will disappear, they are not relevant for the drum sequencing.

A set of parameters for the dynamic velocity control will appear - «**velo amp**» and «**base velo**», see the description below.

Other params work the same as in the Arp mode.

velo amp.	Velocity amplify factor
Values	*0 .. 2.00*
comments	Increase or decrease velocity dynamics around the “base velo” value, see the formula below. Default value is “1.00”, it means no change to the output velocity. When above “1.00”, all velocities greater than the base value will be amplified, while velocities below the base value will be attenuated. When “base velo” = 0, all output velocities will be equal to the “base velo”, meaning no velocity dynamics at all.

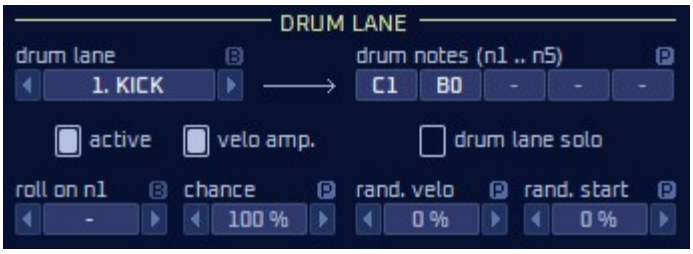
Formula for the velocity adjustment:

```
IF [NoteVelo] < [base_velo] AND [velo_amp] > 1 THEN
   [NoteVelo] = [base_velo] + ([NoteVelo] - [base_velo]) * (1 + ([velo_amp] - 1) / 2)
ELSE
   [NoteVelo] = [base_velo] + ([NoteVelo] - [base_velo]) * [velo_amp]
ENDIF
```

(the first branch means that for velo amp greater than 1 and velocities below the base value the attenuation will be 2x times less, this gives more realistic dynamics)

base velo	Base velocity for “velo amp.” Param
Values	*0 .. 127*
comments	Defines the base value for the velocity dynamics, see the formula above. When “base velo” is close to 0, output velocities will be attracted to the “base velo” value. When “base velo” increases, output velocities will be amplified with “base velo” as a relative zero-line for the amplification.

Drum lane



DRUM LANE block has all the parameters related to the selected drum lane (except step-related parameters, which are on the value lanes).

The «**drum lane**» parameter is linked to the currently selected drum lane; once you change it here, it will change the selected value lane.

drum lane	Currently selected drum lane
Values	*"1. KICK", "2. SNARE", "3. HHATS", "4. CRASH", "5. TOMS", "6. PERC"*
comments	Selected drum lane, this parameter is linked to the currently selected drum lane in the matrix editor.

drum notes (n1 .. n5)	MIDI Channel for “palm mute” note
Values	*C0 .. G10 (MIDI notes 0 .. 127)*
comments	Pre-programmed MIDI notes for each drum lane; these changes will be stored per-program. The default drum note assignment is based on the General MIDI drum map, see the table “DRUM NOTE MAPPING” on the next page.

active	Activate or mute the lane
Values	*On, Off*
comments	Use it to temporary mute certain drum lanes, this can be used as a part of the performance. Note that this param is bank-related, so the lane will remain muted even if you switch programs.

Hint. You can also mute/unmute the lane by clicking this box on the lane itself:

velo amp.	Apply “velocity amplify” to this lane
Values	*On, Off*
comments	When «**velo amp**» checkbox is checked, output note values for this lane will react to «**velo amp**» param value and its associated control «**base velo**». It is something like «accent» or «dynamics» control in some other drum machines and synths

drum lane solo	MIDI Channel for “palm mute” note
Values	*"---", 1 .. 16*
comments	When “palm mute” param is not “---”, it will define midi out channel for that “tapping sound” note. When “p. mute ch” = “---”, midi out channel will be the same as midi out on the SETTINGS tab.

Hint. You can also solo drum lane from the lane itself, «S»-mark will appear, indicating that this lane is soloed. If you switch to another lane, it will become the soloed one, unless you uncheck «drum lane solo».

DRUM NOTE MAPPING:

Default drum notes are based on General MIDI drum mapping, see the following table.

DRUM LANE	Drum note	Note №	key* / *C-2 low*	key / *C-1 low*	key / *C0 low*	
1. KICK	n1.	36	C1	C2	C3	Bass Drum 1
	n2	35	B0	B1	B2	Acoustic Bass Drum
2. SNARE	n1	38	D1	D2	D3	Acoustic Snare
	n2	40	E1	E2	E3	Electric Snare
	n3	39	D#1	D#2	D#3	Hand Clap
	n4	37	C#1	C#2	C#3	Side Stick
3. HIHATS	n1	42	F#1	F#2	F#3	Closed Hi Hat
	n2	46	A#1	A#2	A#3	Open Hi-Hat
	n3	44	G#1	G#2	G#3	Pedal Hi-Hat
	n4	54	F#2	F#3	F#4	Tambourine
	n5	56	G#2	G#3	G#4	Cowbell
4. CRASH	n1	49	C#2	C#3	C#4	Crash Cymbal 1
	n2	57	A2	A3	A4	Crash Cymbal 2
	n3	52	E2	E3	E4	Chinese Cymbal
	n4	51	D#2	D#3	D#4	Ride Cymbal 1
	n5	59	B2	B3	B4	Ride Cymbal 2
5. TOMS	n1	43	G1	G2	G3	High Floor Tom
	n2	45	A1	A2	A3	Low Tom
	n3	47	B1	B2	B3	Low-Mid Tom
	n4	48	C2	C3	C4	Hi Mid Tom
	n5	50	D2	D3	D4	High Tom
6. PERC	n1	60	C3	C4	C5	Hi Bongo
	n2	61	C#3	C#4	C#5	Low Bongo
	n3	62	D3	D4	D5	Mute Hi Conga
	n4	63	D#3	D#4	D#5	Open Hi Conga
	n5	64	E3	E4	E5	Low Conga

(*) these are key names for different naming conventions, depending on your «octave numbering» parameter value (on the SETTINGS tab)

roll on n1	Drum roll for this lane
Values	*"-", 1/4, 1/8, 1/16, 1/32*
comments	When not "-", it will generate a drum roll for this lane (using note "n1") with the specified note length (1/4 is the whole beat).

chance	Not trigger change for this lane
Values	*0% .. 100%*
comments	When below 100% (which is the default value), output notes for this lane will trigger with the specified probability.

Hint. An easy way to create simple non-repetitive percussion pattern works like this: on the «6. PERC» lane, set all steps to «Rnd» (picking random note n1..n5), probability to somewhere around 80%, «rand. velo» to 20%, then add basic kick and share – and you will get a basic pattern with a random percussion over the top, not as repetitive as a simple fixed loop.

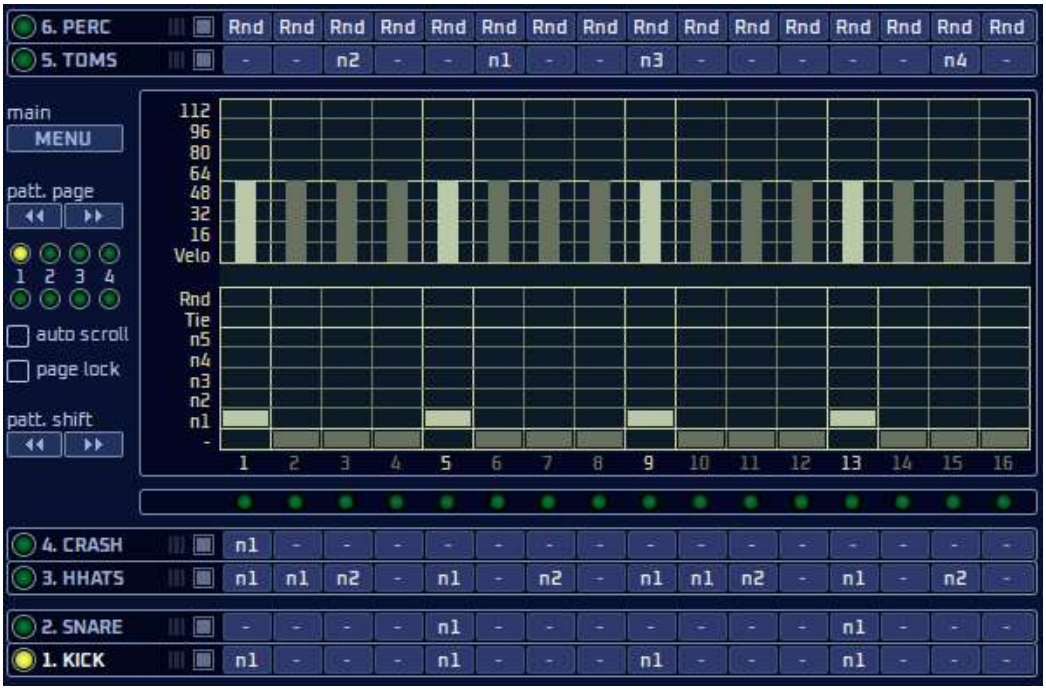
rand. velo	randomize output note velocity
values	*0% .. 100%*
comments	Add random value (positive or negative) to each output note velocity. Works only for a particular drum lane.

rand. start	randomize output note start time
values	*0% .. 100%*
comments	Add random value (positive only) to generated note start time. Works only for a particular drum lane.

VALUE LANES

In «drum sequencer» mode some of the GUI elements will change to match this new functionality.

First, value lanes will change their names and function:



Each of these lanes now can play a selected drum note per step, with varying velocity. Value «-» will mute the step, «Tie» will extend the note from the previous step, «Rnd» will pick the random note from the «n1..n5» list, «n1..n5» will play that particular note.

Lane names like «KICK», «SHARE», «HHATS» are fixed and can't be changed, but it doesn't force you to use those particular sounds for those lanes (but still it is the "default" way to use them and the way they are used in the factory bank).

> ***Info***. Since a single byte is used to encode both drum note (n1..n5) and the velocity for each step, only 5 bits left to encode the velocity, that's why velocity is adjusted in the steps of 8 (like 0, 8, 16, etc..). However, if you apply randomization, it will add fine values to the velocity.

Guitar strum mode

Overview

Since version 2.7.0, BlueARP has «guitar strum» operation mode to generate various guitar strumming and picking patterns. Of course it is not a real guitar replacement, but it is a handy tool to generate guitar rhythm tracks, working well with acoustic and electric guitar sounds and covering variety of genres.

For now, it models just the standard 6-string guitar tuning (E, A, D, G, B, E). Other tunings like open E or a 7-string guitar will be probably added later.

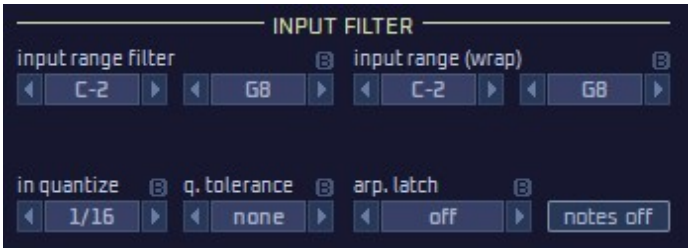
Note that this mode models not only the acoustig guitar, but electric guitar as well, there are special chord voicing modes using strings 1-3 or 1-4 only, which is usually the case for electric guitar rhythm patterns.

In the following sub-chapters, only guitar mode specific settings are described, the ones shared with the Arp mode are just mentioned.

LEFT PANEL

In the Guitar mode, about half of the parameters are shared with the arp mode, another half are guitar-mode specific.

Input filter



INPUT FILTER section in Guitar mode has less params compared to the Arp mode.

Hidden settings here are “order algorithm”, “missing key” settings and “input range (wrap)”; they are not relevant for the guitar mode, because chord voicing here is driven by the chord lookup table.

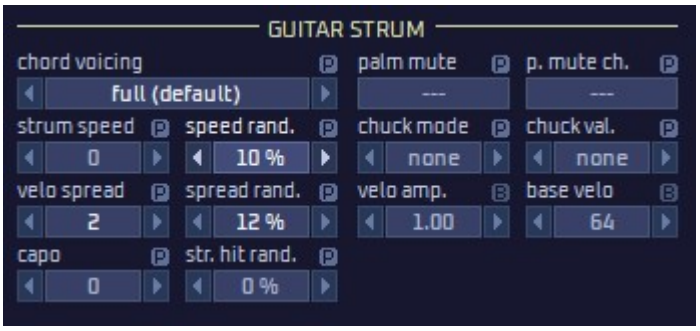
> ***Note***: “q. tolerance” param is especially useful in Guitar mode, it will make BlueARP more responsive to the input chord changes and more “forgiving” to small latencies in the input keys. Recommended settings for Guitar mode: “in quantize” = 1/4 and “q. tolerance” = 1/16.

ARP Engine

ARP ENGINE in Guitar mode has a subset of settings from the Arp mode.

“Force to scale*” and “fixed key” params are missing here, they are not relevant for the guitar mode, because chord voicing algorithm is different here.

> ***Note***: “restart on” = “key” may be more useful and expressive here, than the default “beat 0” setting. With “restart on” = “key” the sequence will restart each time you press some new chord. This add some expression if you have a full down or up strum on the step 1 (which is usually the case). Combined with “in quantize” = 1/4 and “q. tolerance” = 1/16, this is a recommended combination of settings for the guitar mode to play various pop and pop/rock songs.

Guitar strum

GUITAR STRUM block is specific to Guitar mode, it has some params to tune the chord voicing and note generation for the strumming pattern.

chord voicing	how to generate input key list for the arp
Values	*"input keys only", "full (default)", "avoid D* chords", "strings 1-4 only", "strings 1-3 only"*
Comments	*"input keys only"* – ignore the guitar strings, use exact input keys, aligned to string 1, so 3-key chord will be assigned to strings 3-2-1. *"full (default)"* – default mode, the input chord will be played with all 6 notes / guitar strings when possible. *"avoid D* chords"* – works like “full, default” mode, but D* chords will be replaced with A* chord + 5 frets, thus making them 5-stringed rather than 4-stringed. *"strings 1-4 only"* – strings 5 and 6 won’t be used. *"strings 1-3 only"*– strings 4, 5 and 6 won’t be used (these two modes are handy for clean electric guitar patterns, where you usually strum only the lower / hi pitch strings).

palm mute	Note to play on STEP TYPE = “Mute”
Values	*"---", C0 .. G10 (MIDI notes 0 .. 127)*
comments	With the default "---" value, BlueARP will mute all sounding strings on STEP TYPE = “Mute”. When it has this param set to some note, it will also play this note (some guitar patches have guitar tapping and scratching sounds on the left of the keyboard).

p. mute ch.	MIDI Channel for “palm mute” note
Values	*"---", 1 .. 16*
comments	When “palm mute” param is not “---”, it will define midi out channel for that “tapping sound” note. When “p. mute ch.” = “---”, midi out channel will be the same as the main midi out in the SETTINGS tab.

strum speed	Strumming speed
Values	*-5 .. 0 .. +5*
comments	Strumming speed is relative, the default “0” value should be most natural, increasing makes it faster and decreasing makes it slower.

speed rand.	Strumming speed randomization
Values	*0% .. 100%*
comments	Adds a random offset (with zero median) to the strumming speed.

velo spread	Velocity spread for the generated notes
Values	*0 .. 10*
comments	When you strum down on the real guitar, you normally hit the top strings harder, and vice versa when you strum up. Velocity spread models this: the greater is the “velo spread” value, the higher will be the relative velocity or the top strings and the lower the velocity of the bottom strings, while the overall loudness of the chord will stay about the same. When the value is “0”, it won’t affect the velocity. The default value is “2”, recommended values are 2 .. 4.

spread rand.	Velocity spread randomization
Values	*0% .. 100%*
comments	Adds a random offset (with zero median) to the velocity spread.

capo	Guitar capo
Values	*0 .. 12*
comments	Models a capo on the strings. Default “0” value means no capo, “1” – capo on the 1st fret and so on. For example, if you play Em chord with capo = 0, it will be normal Em, but with capo = 1 it will be Dm + 2 frets, the closest chord it can find. If you set “chord voicing” to “avoid D* chords”, it will be Am + 5 frets.

str. hit rand.	STRINGS HIT lane randomization
Values	*0% .. 100%*
comments	Adds a random offset to the STRINGS HIT lane values.

chuck mode	Chuck strumming algorithm
Values	*"none", "channel","octave","AGML2"*
comments	Makes difference when you use “ChkUp” and “ChkDn” values on the STEP TYPE lane. With the default “none” setting, it will play a short strum with the same sound. But you can also select the sound from different MIDI channel with “channel” setting or from a lower octave with “octave” setting (some guitar patches may have chuck/muted strings sound on the lower part of the keyboard). “AGML2” setting is specifically to deal with the “Ample Guitar M Lite II” free plugin, where chuck strumming is modelled by holding some low key.

chuck val.	Value for the “chuck mode” setting
Values	*"---", 1 .. 16 for* “chuck mode” = “channel”; *"---", -3 .. +3 for* “chuck mode” = “octave”
comments	Depends on the “chuck mode” setting. With “chuck mode” = “channel”, sets the output channel for chuck strum notes (“---“means use the main midi out). With “chuck mode” = “octave”, sets the octave offset for the chuck strum notes.

velo amp.	Velocity amplify factor
Values	*0 .. 2.00*
comments	Increase or decrease velocity dynamics around the “base velo” value, see the formula below. Default value us “1.00”, no change to the output velocity. When above “1.00” all velocities greater than the base value will be amplified, below the base value they will be attenuated. When “base velo” = 0, all output velocities will be equal to the “base velo”, meaning no velocity dynamics at all.

Formula for the velocity adjustment:

```
IF [NoteVelo] < [base_velo] AND [velo_amp] > 1 THEN
   [NoteVelo] = [base_velo] + ([NoteVelo] - [base_velo]) * (1 + ([velo_amp] - 1) / 2)
ELSE
   [NoteVelo] = [base_velo] + ([NoteVelo] - [base_velo]) * [velo_amp]
ENDIF
```

(the first branch means that for velo amp greater than 1 and velocities below the base value the attenuation will be 2x times less, this gives more realistic dynamics)

base velo	Base velocity for “velo amp.” Param
Values	*0 .. 127*
comments	Defines the base value for the velocity dynamics, see the formula above. When “base velo” is close to 0, output velocities will be attracted to the “base velo” value. When “base velo” increases, output velocities will be amplified with “base velo” as a zero-line for the amplification.

Output filter

OUTPUT FILTER block in Guitar mode has a subset of settings from the Arp mode.

You won’t see “output filter (wrap)” and “rand. gate” params, they are not relevant for the Guitar mode.

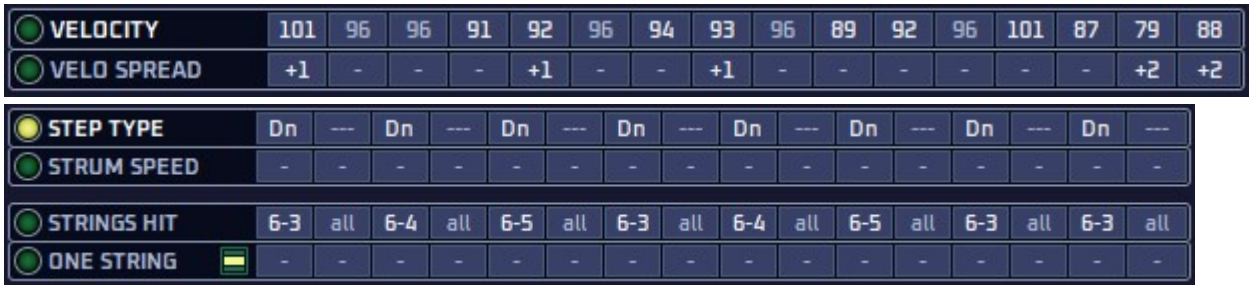
“rand. velo” param works a bit more complicated than in arp mode – it will mostly randomly decrease/increase velocities for all strings in a chord, to mimic occasional fluctuations in the strumming strength, but will also add smaller random variations to the velocities of the individual strings.

“rand. start” param works about the same, it adds start time variation to the whole chord and smaller variations to the individual strings, ensuring that strings will still play in the right order.

Chains

Chains block is the same as in the Arp mode, and works exactly the same.

VALUE LANES



Value lanes in Guitar mode are different from the Arp mode, except the “VELOCITY” lane.

VELOCITY	Velocity value for each step
values	*0 .. 127*
comments	Default value is 96. Use it to set velocity accent for certain steps, here it works about the same as in other modes.

VELO SPREAD	Velocity spread adjustment per step
Values	*-5 .. +5*
Comments	This is the same setting as “velo spread” in the “Guitar strum” block of the left panel, but here it adjusts it on a per-step basis, the additive way. For example, if we have “velo spread” = 2, having VELO SPREAD = +1 for a step will result in 2 + 1 = 3 value applied to this step.

STEP TYPE	Several options for output note generation
values	*sPk – short pick, all strings sound at once;* *Pick – all strings are picked at once and keep sounding;* *StrmUp – strum up;* *StrmDn – strum down;* *“---” – no action, strings from the previous steps keep sounding;* *Mute – mute all sounding strings;* *ChkUp – chuck strum up;* *ChkDn – chuck strum down;*
comments	For *Pick*, *StrmUp* and *StrmDn* values strings keep sounding until the “Mute” step or all input keys released. On several consecutive *Pick* / *StrmUp* / *StrmDn* steps notes will be re-triggered. This mimics the guitar behavior, where you have to palm-mute the strings to cut the sound, otherwise they keep sounding. Chuck strumming can be done with a different sound on a different midi channel, see “chuck mode” and “chuck val.” parameters in the “Guitar strum” block of the left panel.

STRUM SPEED	Strumming speed adjustment per step
Values	*-5 .. +5*
Comments	This is the same setting as “strum speed” in the “Guitar strum” block of the left panel, but here it adjusts it on a per-step basis, the additive way. For example, if we have “strum speed” = 3, having STRUM SPEED = -1 for a step will result in 3 - 1 = 2 value applied for this step.

STRINGS HIT	Restrict the strings that can be hit for a certain step
Values	*"3-2", "4-2", "2-1", "3-1", "4-1", "all", "6-2", "6-3", "6-4", "6-5", "6"*
comments	Additionally restricts the strings hit for a step. “all” is the default setting. String numbering is from bottom to top (string 6 has the lowest pitch). For example, for low-pitch power strumming on the electric guitar it will work well if you use “6-5” and “6-4” values here. Also, for some up and down strums you can restrict number of strings hit, the other strings will keep sounding (sustained) and it creates less pronounced strums, it works well on weak measures and within dense strumming patterns.

Note. This parameter has a set of string ranges instead of individual strings for two reasons. Firstly, to make it easier to adjust up and down strums, you can make them less pronounced by using a subset of strings. Secondly, it works well with the “str. hit rand” parameter, since the nearby values are close. If you want to set the individual strings to be picked for a certain step, you can use the following ONE STRING lane in the polyphonic mode.

ONE STRING	Trigger just one string
Values	*"5th", "Rt", "-", "s1", "s2", "s3", "s4", "s5", "s6"*
comments	the default setting is “-“, it doesn’t affect anything. With any other setting, it will override STRINGS HIT lane and trigger just one specified string, but only if STEP TYPE is not “---” or “Mute”. When polyphonic mode mark is on , you can select several strings at once, it will override STRINGS HIT lane as well, this is useful for complicated picking patterns.

Note. Strings 5 and 6 may have some special logic, since they are not used in some chords. For the version v2.8.0, string 6 will be substituted with string 4 for D* chords (that use only strings 1-4).

FAQ / Troubleshooting

Installing BlueARP

«Unrecognized Developer» error message when trying to run BlueARP on OSX (reported on OSX Catalina and later versions)

Since v2.3.8, BlueARP is notarized with the proper Apple notary tool. While it is not listed in the Apple store, it is still digitally verified by Apple. However, it doesn't give 100% guarantee that the installations will be flawless. The majority of the problem reports is related to Logic X Pro and MIDI-FX version «BlueARP.component». If your Logic X or Garage Band doesn't see BlueARP, try the following:

Remove the quarantine flag from the plugin by running in terminal:

```
xattr -rd com.Apple.quarantine BlueARP.component
```

If this doesn't help:

1. clearing the folder /Users/graywolf2004/Library/Caches/AudioUnitCache
2. running killall -9 AudioComponentRegistrar in terminal
3. Choosing Preferences on Logic -> Reset and rescan all audio units in Logic

Also, sometimes security settings may block new software installation, so go to System Settings -> Security & Privacy, if you see «BlueARP was blocked because…» message, press «Open anyway» to create exception for.

Sync & Timing issues

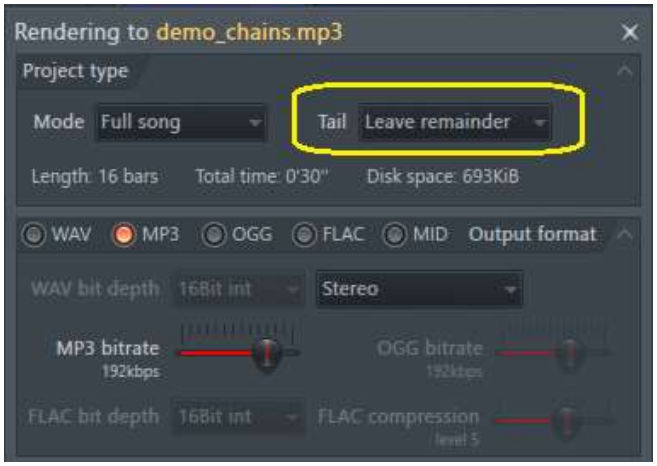
Output note timing is not perfect, like they are delayed by random values.

Check audio settings in your DAW. Your audio buffer size should be 256 samples or less, 128 is recommended. 256 samples will give maximum inaccuracy of 5ms at 48kHz (256 / 48000 ≈ 0.005s)

Rendering audio in FL Studio

When trying to render a project in FL Studio, only 1st note comes out of BlueARP, others are missing.

In rendering settings (you get there automatically, when you call Export -> mp3 or whatever) change «Tail» option to «Leave remainder»



solution provided by Saif Sameer

VST3 compatibility

«MIDI FILTERS» settings like pass thru for pitch bend and mod wheel do not work in VST3 version of BlueARP, while in VST2 version they work fine.

VST3 standard has some artificial limitations, preventing developers to create full-featured MIDI plugins. There are some «Legacy» extensions in VST3 added at some point, but still normally MIDI CC reception is not supported by VST3. If your DAW has full VST3 support (in particular, if it calls getMidiControllerAssignment() to ask plugin about supported controller messages), then almost all settings in «MIDI FILTERS» block should work. Except one - «other CC msg», it will be grayed out in VST3 version, because it is too tricky to implement (will require adding 100+ dummy params to the plugin to handle MIDI CC messages).

If this functionality is critical for you, use VST2.4 version of BlueARP instead.

Links

Developer's website:
http://www.omg-instruments.com/
http://www.graywolf2004.net/

BlueARP discussion thread at KVR Audio forums (latest updates, news):
http://www.kvraudio.com/forum/viewtopic.php?p=5080757

Video demonstrations and tutorials are available on developer's YouTube channel:
http://www.youtube.com/user/graywolf2004ru?feature=watch

1-hour long video manual for the BlueARP, 2024
https://youtu.be/3W837bBID5k

Please write bug reports and suggestions to KVR audio thread or email me at graywolf2004@gmail.com

Oleg Mikheev aka Graywolf, © 2012-2024