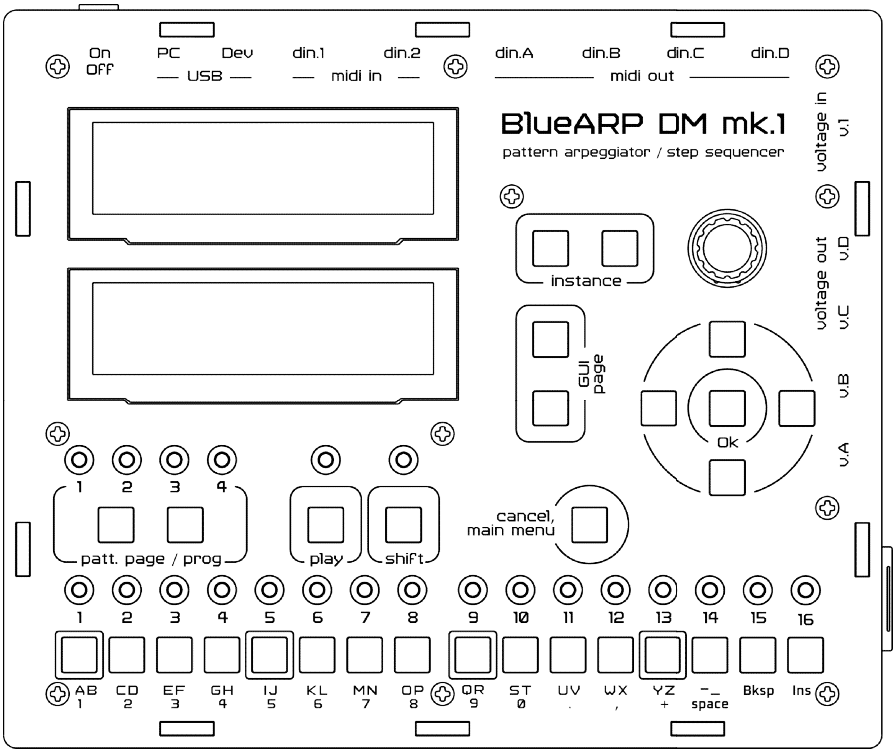
BlueARP DM mk.1

8-channel pattern arpeggiator / step sequencer

Operation Manual

October 2021



www.omg-instruments.com

Safety and Operating conditions

Volume levels

The device may be able to produce sound levels in other devices that could cause hearing damage or loss when used in combination with synthesizers, headphones and/or speakers.

DO NOT operate at high volume levels.

Power and grounding

Make sure to use the power supply with galvanic isolation from the wall plug, delivering 5V and at least 600 mA of current. Inputs/outputs of the device share ground plane with the power source. Using an incorrect (non-isolated) power supply may cause an electric shock and permanent damage to the device.

Operating conditions

Use the device in a dry environment within a room temperature range of 21–32 °C (70–90 °F). Do not use it near water or moisture, such as a bathtub, sink, swimming pool or similar place. Please note the enclosure is not water proof.

Do not expose the device to hot sunlight.

It is not recommended to use it in a smoking environment as it may decrease the lifespan of the device.

Do not place heavy objects on top of the device as this may damage the front panel.

Cleaning

To clean the device, only use a soft damp cloth. Do not let the liquid to seep inside while cleaning.

Only use water, alcohol or a mixture of both as a cleaner. Other cleaning substances may cause a chemical reaction with the surface materials.

Welcome

BlueARP was initially developed as a VST plugin and released to the public in 2012. You can still find the thread at KVR audio forums, which I started back then and still update to this day.

The idea of BlueARP was a fusion of several features I liked in arrangers, groove boxes and hardware synthesizers. The goal of making the plugin was to create a tool for real-time electronic music performances; similar to arrangers, but not quite. The idea was to separate the pattern information and the actual notes/pitches, and be able to control the harmonic part of the track in real-time, while having the ability to switch patterns on the fly. And more importantly, to have something that would allow more real-time flexibility and encourage live performing.

In 2010 I made a first dirty prototype in SynthMaker (now DSP Robotics* FlowStone). It had all the basic functionality, however there were issues with MIDI timing due to SynthMaker's limitations at the time. I switched from SynthMaker to coding, and in 2012 the first Windows version of BlueARP plugin was released to the public.

In 2014 BlueARP was initially released for OSX, and by 2016 it reached maturity.

In 2016 I started to experiment with embedded development and gradually realized that it is possible to make BlueARP in hardware, the device I dreamed of from the very beginning.

In 2017 I posted my 1st video with the hardware BlueARP prototype: it was just a development board and a jumble of wires, but for me it was a very exciting moment and a milestone in development! I managed to run the actual C code from the plugin on the embedded system, on that tiny little board, without a computer.

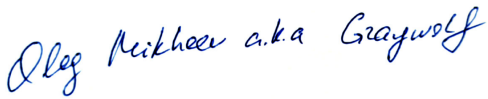
Now the plugin and the hardware device share the same core code written in C, one of the things I'm really proud of. This code was perfected over the years with substantial help from the great community at KVR audio forums.

The philosophy behind BlueARP DM is a bit nostalgic: I wanted this thing to feel "old-school", a callback to early music computers and hardware sequencers. This is one of the reasons I use white-on-black screens and old-fashioned "clicky" buttons.

Still, I wanted to follow "best of both worlds" approach and implemented a software control interface, fully resembling the BlueARP plugin.

The BlueARP DM is a result of my labor of love and passion for years, and I hope you'll enjoy making music with BlueARP and it finds a good place in your setup!

Regards,



* DSP Robotics is a division of Outsim Ltd.

Special Thanks

Marshal Arnold for being the very first beta-tester of the unit and giving me lots of valuable feedback on how to improve things.
https://www.marshalarnold.com/

Randal Adamson aka **Ranzee** for hosting BlueARP beta-testers chat, spreading the word online and being one of the first beta testers.
https://ranzee.com/

Saif Sameer for supporting BlueARP plugin for years; for keeping the BlueARP thread at KVR alive. His efforts played the role in making this possible as well.
https://www.youtube.com/c/SaifSameer

Seaton McDonald aka **MPhrasor** for meticulously testing the unit; for helping me to find and fix some nasty bugs.

Lina Glover aka **Sakura** for supporting me from the very early stages of prototyping.

Andrew Mee for helping me with MIDI protocol and improving the manual.

All the other great people from **Ranzee**'s community.

Last but not least, my wife **Sasha** for putting up with all the countless weekends and vacations with me lost in the laptop and late night working hours.

Table of contents

Introduction

Concepts

BlueARP DM (Desktop Module) is an advanced multi-channel MIDI pattern arpeggiator, able to run up to 8 independent arpeggiators, called **Instances**, in parallel.

BlueARP DM can be either a brain of a hardware setup or an addition to another hardware sequencer. BlueARP DM can control up to 8 synthesizers in real-time from a MIDI keyboard, computer (via USB port) or a hardware sequencer.

BlueARP creates repeating MIDI output, which depends on two crucial elements: **Pattern** and **Key Ordering Algorithm.**

A **Pattern** in BlueARP is a set of step-related parameters defining:

- Whether to generate the note for a certain step, mute or sustain it;
- Which note to take from the **Input Key List**;
- How much octave and semitone transposition to apply;
- Whether to change length and velocity of the generated note;

BlueARP DM is different from sequencers and conventional arpeggiators. In a sequencer, a melodic phrase is recorded and played back afterwards. In a conventional arpeggiator an algorithm is selected such as “Up”, “Down”, “Up-Down”, and then input keys are arranged in a respective manner and repeatedly passed to the output.
BlueARP’s key difference is that is takes key sequence from a pre-programmed pattern and applies incoming notes to this pattern, adding per-step controls like octave and semitone shift, velocity change and others.

A **Pattern** doesn’t hold actual note pitches; they are taken on the fly from the **Input Key List**, which is an ordered set of input notes. It can be ordered in several ways, depending on the selected **Key Ordering Algorithm:** by pitch, as played, by steps of the detected chord, ascending or descending.

Input Key List defines the harmony of the generated melodic phrase.

Pattern and **Key Ordering Algorithm** parameters are part of the **Program**, along with other parameters such as input ranges and swing. Each arpeggiator **Instance** holds up to 128 programs, making up a **Bank**. Programs can be arranged into **Chains** to create longer super-patterns.

Finally, **Chains** can be included into **Meta-Chains**; the latter allow changing the programs for all 8 **Instances** with a single press of a button.

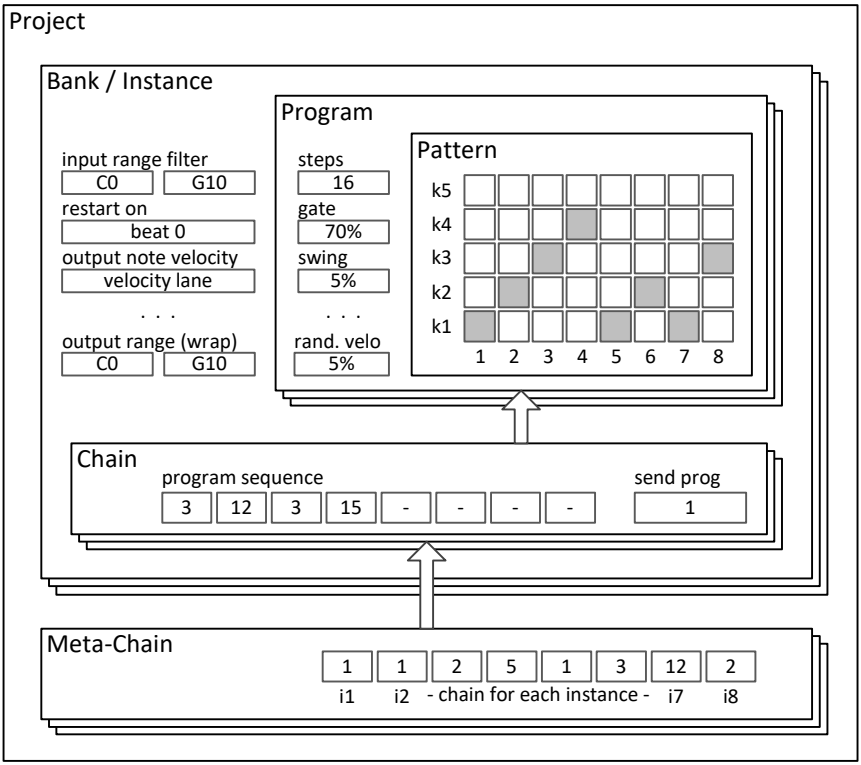


Figure 1. BlueARP data structure showing relations between these elements and how they are nested inside the project file (.bap).*

The **Pattern** is a set of step-related parameter lanes, like **KEY SELECT** (k1, k2 … k5) or **OCTAVE** transposition for a step. Together with step-independent parameters like “steps” and “gate” they make a **Program**.

A set of 128 programs makes a **Bank**, also incorporating program-independent settings like “input range filter” and “restart on”.

Chains are also part of a Bank; it holds up to 16 of them.

Project is a top-level container that holds 8 **Banks** (a **Bank** for each of 8 **Instances**) and a set of 16 **Meta-Chains**. Meta-chain is a lane holding Chain numbers for all 8 Instances. Thus, by switching a Meta-chain it will switch Chains for all the Instances simultaneously.

See the Conceptual model chapter on page 22 for more details on the core concepts.

BlueARP DM core functionality is the same as the plugin (moreover, it shares the same C code), so it is recommended to familiarize yourself with the plugin before using BlueARP DM, it will make it easier to grasp the hardware interface.

Features

BlueARP will transform incoming notes into melodic phrases according to pre-programmed patterns. Additionally, it can:

- Switch patterns on the fly using **Chains** and **Meta-Chains**
- Control various synth parameters via CC / RPN / NRPN with the help of message routing / transcoding functionality

There are various ways to connect BlueARP DM to the rest of a setup. It can either be a MIDI Clock source or follow an external MIDI clock.

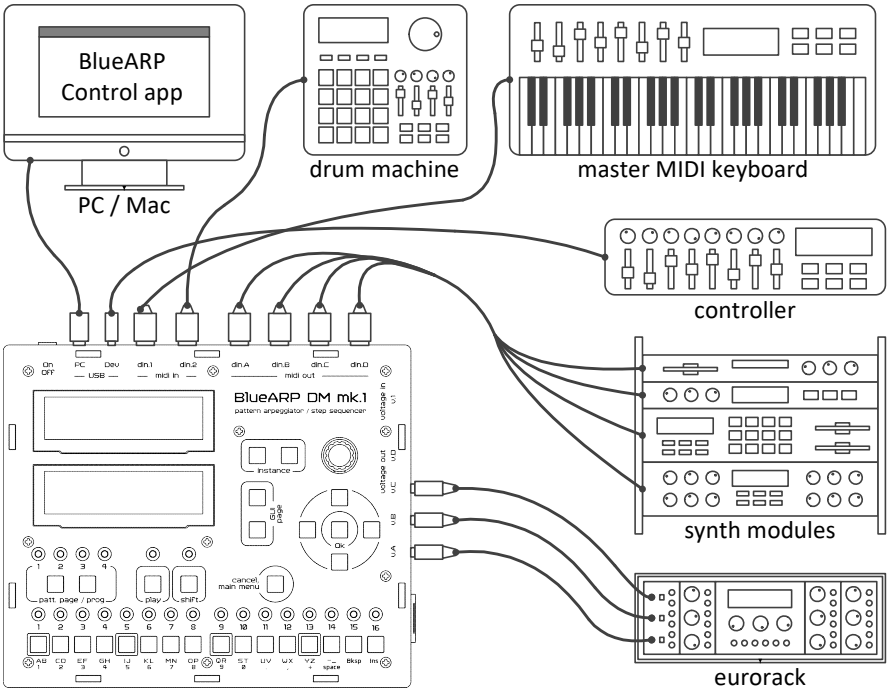


Figure 2. BlueARP DM possible connections

On this picture:

- USB PC port (PC / Mac): supplies power for the unit and runs BlueARP control software to edit BlueARP DM, much like the BlueARP plugin
- USB Dev port: MIDI controller to automate BlueARP parameters, or control the synths via CC/RPN/NRPN transcoding functionality
- din.1 port: drum machine and provides external MIDI clock
- din.1 port: master MIDI keyboard
- din.A..C ports: synth modules
- v.A..C ports: eurorack modules (vA is pitch, vB is gate, vC is velocity)

It is possible to connect external pulse clock signal to **v.1** analog input. The BlueARP can follow the external clock and re-distribute it to other outputs.

All the patterns, programs, chains and other settings can be saved to SD card as a single *.bap project file (*.bap stands for **b**lue**a**rp **p**roject). It is possible to save/load banks (*.fxb) and programs (*.fxp) as well, they are compatible with BlueARP plugin.

A bank is loaded to the active (selected) instance, so the BlueARP DM can load up to 8 different banks in total.

When connected to a computer or a device with an embedded USB host, BlueARP DM will be recognized as a class-compliant MIDI device.

BlueARP DM by the numbers:

- 8 arpeggiator Instances running in parallel
- Up to 128 programs for each Instance (each program holds a Pattern)
- Up to 64 steps per program
- Up to 16 Chains per Instance
- Each Chain can link up to 8 programs
- Up to 16 Meta-Chains, each one holds the Chain numbers for each Instance
- 6 MIDI inputs and 12 MIDI outputs in total
- 4 voltage outputs with ±10V range (each act as a CV or gate)
- 1 voltage input to receive pulse clock

How to read this manual

If you are eager to start using BlueARP DM, go to Quick Start chapter on page 12. It will give you the essential instructions to start.

This document is presented in a linear fashion. It is recommended that you read the “Layout” and “Interface” chapters for the hardware overview and interface layout. It will help you to understand the interface basics.

The “Conceptual Model“ chapter has several diagrams to get basic top-level understanding of the device design and workflow. While the “Tips and Tricks“ section on page 68 explains some common workflow routines.

The “GUI Pages” chapter may be referred to later, if any parameter on a page is unclear.

Formatting conventions

Parameter names, GUI page names and port names and always **marked with this font** to improve readability.

Parameter description looks as follows:

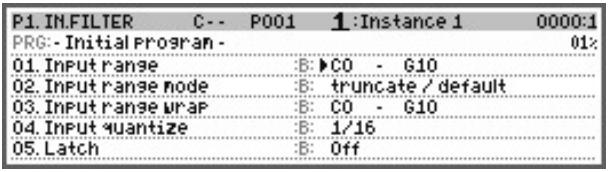
parameter name

Parameter description text.

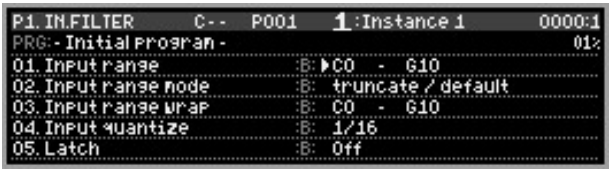
Hints look like this:

Hint text gives some additional information, additional explanations or tips and tricks.

Pictures are inverted for better readability on paper, for example:



On the BlueARP DM the color is white on black and looks like this:



Quick Start

Basic operation

To power up the unit:

- Insert USB type B cable into **USB PC** port on the back panel. Cable can be connected to either PC or a wall adapter with USB-A port, delivering at least 600 mA of current
- Press the power button on the back panel to turn the unit on

The startup page will display for about 3 seconds before the **P1 IN. FILTER** page.

By default, the unit starts into the following state:

- Factory bank loaded for all 8 **Instances**
- **Instance** 1 is active, others are muted (**Arp Mode** = Off)
- For **Instance** 1, **din.1,** channel 1 is selected as a MIDI input and **din.A**, channel 1 as a MIDI output
- For other muted instances output MIDI channels are set to 2…8 respectively (this can be adjusted on the **P7. ROUTING** page)

Use two DIN-5 MIDI cables to:

- Connect MIDI out from a keyboard to **din.1** port on the back panel
- Connect **din.A** port on the back panel to MIDI input port of a synth

Connect the synth to speakers. Please reduce the volume on the synth to make sure it doesn't get too loud.

If everything is connected properly, rolling 16th notes should be heard when pressing any key on the keyboard. This is BlueARP's "- Initial program -" running.

Make sure the synth is configured to receive MIDI notes on the given MIDI input port and channel.

Use **patt. page / prog** buttons to browse the programs. Current program name is displayed under the header, **P005** label indicates the program number:

Use **GUI page** buttons to move through GUI pages. Use the cursor buttons with **Ok** in the middle to navigate through elements on the page, rotate the encoder (wheel) to change the selected value.

Refer to sub-chapters in "GUI Pages" chapter to get more information on the GUI pages and the parameters they contain.

Layout

Top Panel

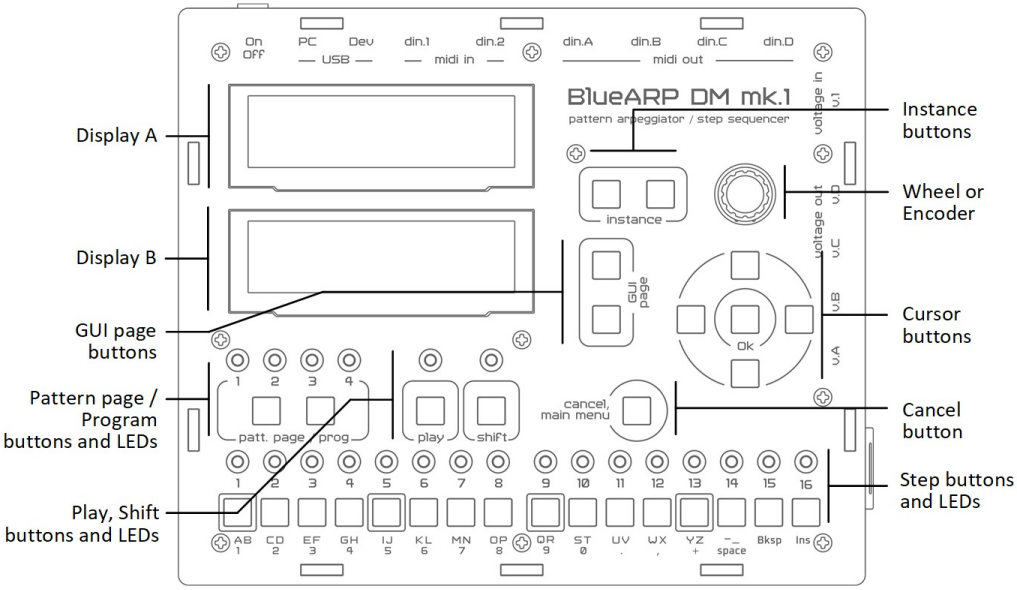


Figure 3. BlueARP DM Top Panel

Displays A and B work as a single display split into 2 halves.

All the buttons except step buttons work the same on all GUI pages, see “Buttons“ chapter on page 17 for further details.

Step buttons and LEDs may behave differently depending on GUI page. For example, on most pages the green step LEDs represent the current pattern step, while on page **06. META-CHAIN** they represent the current beat in a reference loop.

For more details, check “GUI Pages” chapter on page 28. Each subchapter describes each GUI page and has information about step buttons / LEDs behavior.

Back Panel

Back panel has the power input via USB B (**USB PC**) port and all the MIDI connectivity.

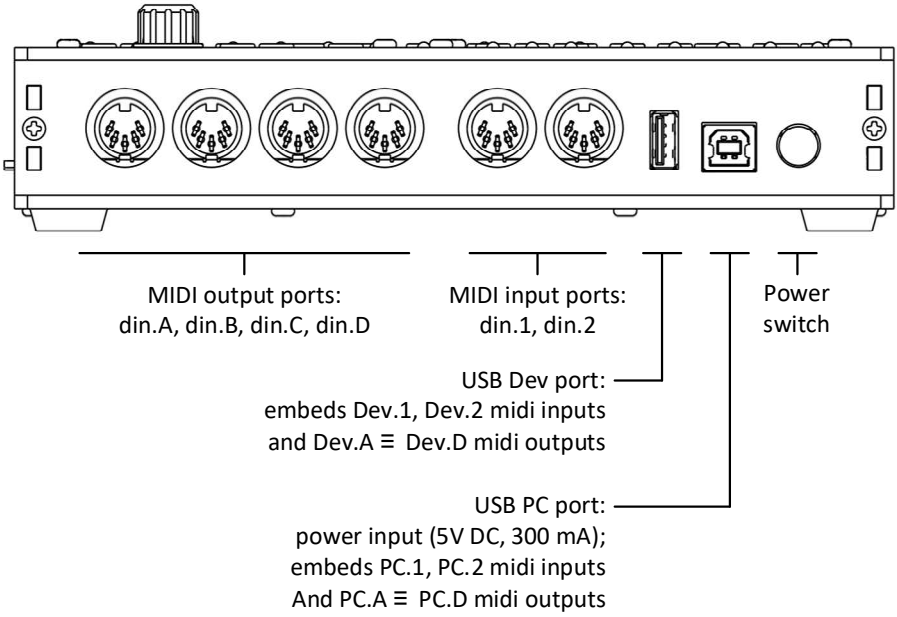


Figure 4. BlueARP DM Back Panel

MIDI output ports, MIDI input ports

Since BlueARP DM has 4 MIDI outputs, separate MIDI cables can be used to connect up to 4 synths, without chaining the devices or using MIDI splitters.

Two MIDI inputs are available to connect 2 MIDI keyboards or a separate MIDI-clock source, such as a drum machine or hardware sequencer.

USB Dev. port

Accepts any USB-MIDI compliant device such as a MIDI-keyboard, controller, synthesizer (USB pen-drives are not supported, use SD Card instead).

For a bus-powered device, supply current will be drawn from the **USB PC** port and will add to BlueARP's own current consumption (300 mA). Make sure that power supply limits are not exceeded. When the current exceeds the limit, shutdowns, reboots or instability may occur.

The USB-MIDI specification has the concept of virtual MIDI cables, allowing several MIDI ports connections over a single USB wire. A single USB-MIDI device may have several MIDI ports.

For example, on the BlueARP side a typical MIDI keyboard will be presented as 2 MIDI input ports (which are outputs from the keyboard perspective). First one will transmit keyboard events like note on/off messages; second one will represent physical MIDI input port on the keyboard. In BlueARP DM, they will be labelled as **Dev.1** and **Dev.2** MIDI inputs.

The same happens with embedded MIDI outputs **Dev.A** … **Dev.D**. For a typical synth module, first MIDI input (output from BlueARP side) will be the synth engine, while the second will likely represent physical MIDI output port on the synth module. In BlueARP DM, they will be addressed as **Dev.A** and **Dev.B** MIDI outputs respectively.

USB PC port

When **enable USB PC port** setting on page **10. SYSTEM** is "On" and the unit is connected to the PC, it will be recognized as a generic USB-MIDI class-compliant device.

From the PC side, a "BlueARP DM" device should be seen with 2 MIDI outputs and 4 MIDI inputs. BlueARP MIDI input will be seen as a MIDI output from the PC side and vice versa.

This means that 2 MIDI outputs on the PC side correspond to MIDI input ports **PC.1** and **PC.2** inside BlueARP DM.

The 4 MIDI inputs on the PC side relate to MIDI output ports **PC.A**, **PC.B**, **PC.C,** and **PC.D** inside the BlueARP DM.

Right Panel

Right panel contains SD Card slot and voltage I/O ports.

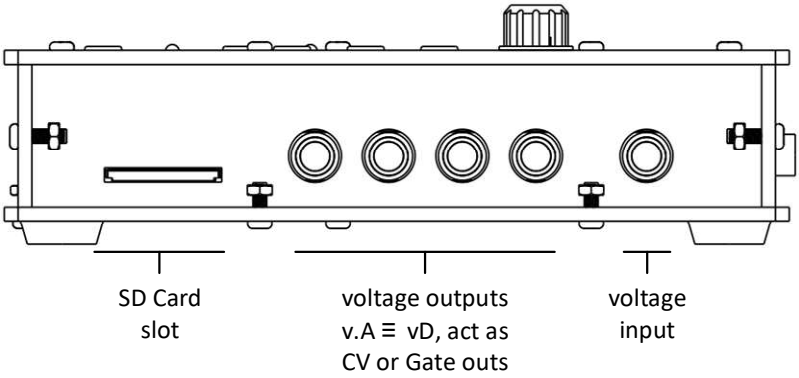


Figure 5. BlueARP DM Right Panel

SD Card slot

Accepts SD (classic), SDHC (high capacity) and SDXC (extended capacity) cards.

SD Card should be FAT or exFAT formatted. NTFS and other file systems are not supported.

voltage outputs

Each voltage output can deliver voltage in a range ±10V.

Operational amplifiers at the output have short-circuit protection, so shorting the voltage output shouldn't damage the unit. However, voltage at the other outputs will drop, because shorting will overload the internal voltage converter.

Voltage outputs should be connected to a high impedance load (1 kΩ or higher).

Each voltage output acts as a CV, gate or clock output, depending on the configuration, see chapter "16. CV/GATE" on page 63 for more details.

voltage input

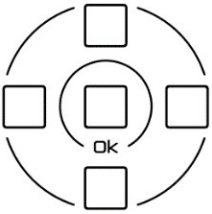
Works as a pulse clock input, making it possible to synchronize BlueARP DM to modular. This connection accepts positive pulse clock signal with amplitude between 2V and 10V.

Voltage input has internal pull-up resistor, so it is possible to connect switch pedal and detect its state (feature reserved for future updates).

Interface

Buttons

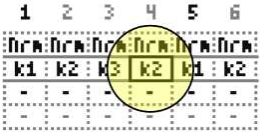
Cursor buttons



Cursor buttons will navigate through elements on any GUI page. Pay attention to the selector mark:

- editing mode, rotate wheel to change the value
- navigation mode, rotate wheel to navigate

OK button also brings up action menu when in editing mode (see page 19).



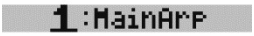
On **04. STEPS** page there is no selector mark because elements are too small, selected matrix cell is highlighted with a solid border.

On many pages, pressing and holding any step button will switch to navigation mode, try it and see if the selector mark changes. On **04. STEPS** page, hold any step button and rotate the wheel to change lanes.

13. MIDI MON, **14. KEY MON** and **15. USB MON** pages do not have selectable GUI elements. On these pages, cursor buttons will only select elements on the header.

Instance buttons

BlueARP DM runs 8 **Instances** in parallel, but only one is visible on the screen at a time:



Selected instance number is shown as s a big digit in the header line (top display).

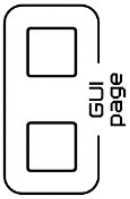
This can be renamed via action menu: select 1 in the header, press **Ok** and select **rename**.

Use **Instance** buttons to change the active instance.

Alternatively, digit in the header can be selected with cursor buttons and using the wheel to change the instance.

When **02. Arp Mode** = Off (page **P2. ARP**), instance number will be greyed out.

GUI page buttons



Use **GUI page** buttons to navigate through GUI pages.

Current page name and number are always displayed in the header like this:

P6. META-CHAIN

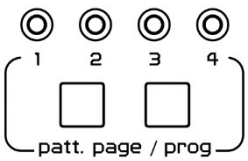
Alternative and faster way to navigate through GUI pages: press **cancel, main menu** button to bring up the MAIN MENU:

Select the desired page: using the wheel, cursor buttons or pressing step button **1...16**.

Pages above 16 can't be selected with step buttons.

Depending on which page is active, **cancel, main menu** button may first cancel the current operation (for example, exit the action menu) and then will bring up the page list.

Patt. page / prog buttons



On all pages (except **04. STEPS)** these buttons will browse programs.

Current program number is shown in the header like this:

P008

On page **04. STEPS**, logic is a little different:

- **patt. page / prog** buttons will browse through pattern pages first when program has more than 16 steps
- on the last pattern page right button will switch to the next program
- on the first pattern page left button will switch to the previous program

Pnnn element can be selected in the header (**nnn** is the program number) and rotate wheel to change it.

Action menu / Ok button

Action menu shows up when clicking **Ok** on some GUI elements. Action menu is like a right-click popup menu in many PC applications, providing additional option for the current element.

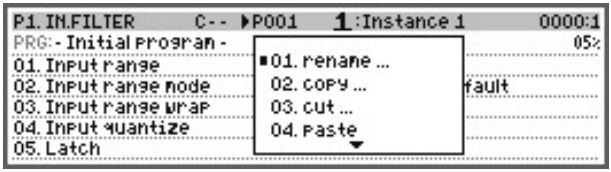
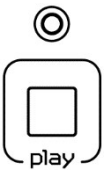


Figure 6. Action menu upon clicking P001 element on the header

There are number of ways to navigate action menu:

- Use Up/Down buttons to select an option and press **Ok** to confirm
- Use wheel to scroll through options and press **Ok** confirm
- Press corresponding step button to confirm (this is a shortcut)
- Press **cancel**, **left** or **right** to quit the action menu

Play button



Will start or stop playback.

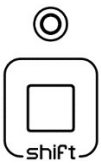
When playing, LED above the button will blink red on the start of each bar and green on the start of each beat.

Bar:Beat value on the top right shows the current song position e.g.

BlueARP DM responds to playback start/stop system MIDI messages.

When pressing some keys without starting playback, BlueARP will start “fake playback”, a behavior inherited from the plugin. If **Latch** is set to On, BlueARP will continue running after all the keys are released. Pressing **play** in such case will stop ”fake playback” first.

Shift button



When in “shift” mode, the LED above will light up and some buttons will alter their functions.

shift + patt. page / prog	Cyclic shift (rotate) pattern, the same as “patt. shift” buttons on the plugin.
shift + step button	Brings up QuickEdit screen. A fast way to change essential parameters without going to another GUI page. See chapter “QuickEdit param window” on page 21 for details

Common GUI elements

GUI Page Header

Each GUI page has a header with the same structure (see Figure 7).

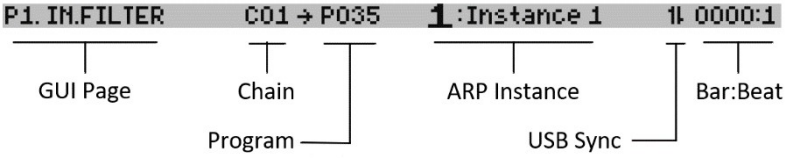


Figure 7. Page header structure

Header has the following elements:

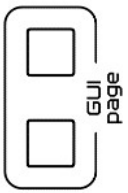
GUI Page	Number and name of the current GUI page.
Chain	Current chain number, starting from 1, or “C - -“ if no chain selected.
Program	Current program (1 … 128). If chain is selected, arrow indicates that current program is selected by the chain engine
ARP Instance (1 ... 8)	One of the 8 **Instances**. Most of the settings on GUI Pages are related to the selected arp **Instance**. If [arp mode] = Off, instance number will be greyed out.
USB Sync icon	This icon will appear when USB connection to PC is connected; BlueARP Control application is running and synchronized with the BlueARP DM unit.
Bar:Beat	Current song position.

TextEdit window

Text editing window displays when to change file name, program name or another object name.

Use **Left/Right** cursor buttons to navigate through characters.

Use wheel to change current character, **Up/Down** arrows will jump to next/previous character block (for example, pressing **Up** will capitalize lowercase letter).



GUI Page Up will move cursor to the 1st character, **GUI Page Down** will move to the end of the string.

Pressing **OK** will confirm the action, **Cancel** will abort the operation.

Use step **buttons 1 – 14** for faster text input. Step **button 14** is a space / underscore, **button 15** – backspace (delete the character to the left), **button 16** inserts an empty character.

QuickEdit param window

QuickEdit windows allows the editing of parameters such as **gate** and **swing** without switching to another GUI page.

To open QuickEdit window, press **shift** (LED above will light up) and press one of the step buttons, each one is associated with a certain parameter.

Conceptual Model

Signal Flow Diagram

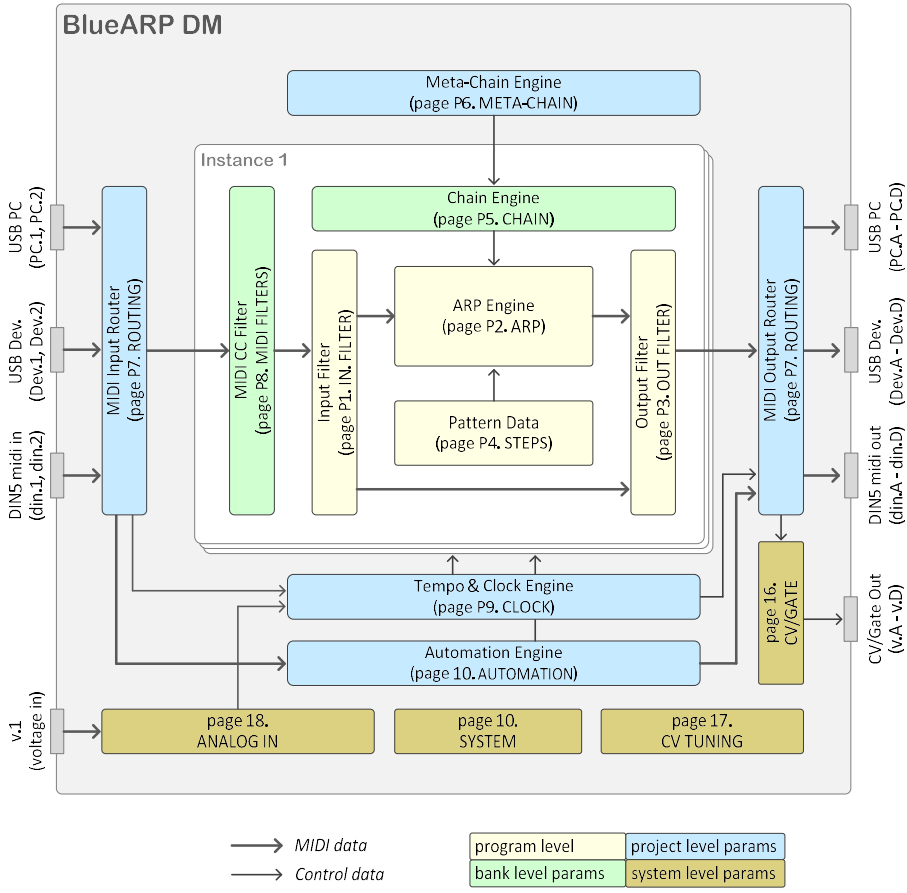


Figure 8. BlueARP DM Basic Signal Flow Diagram

MIDI signal flows from the left to the right, from MIDI inputs to MIDI outputs.

Each block shows a reference to its associated GUI page. See the “Project data structure” chapter on page 26 for more details.

MIDI Input Router redirects incoming MIDI data to ARP Instances 1-8 according to settings on page **P7. ROUTING.**

Instance 1 block can be thought of as a BlueARP plugin running inside the unit, along with seven other instances (2 to 8).

MIDI CC Filter processes incoming CC messages and can ignore them or pass them to the next block.

Input Filter oversees preparing an ordered key list for the **Instance** by performing input range filtering, real-time quantization, and missing keys substitution.

ARP Engine is the heart of the BlueARP DM – it takes ordered key list from the Input Filter and generates the note pattern according to Pattern Data programmed on page **P4. STEPS**.

Output Filter performs post-processing of the generated notes such as master transpose, output range filtering, and randomization.

CV/Gate Engine converts MIDI note on/off messages into voltages on v.A – v.D outputs, according to the configuration on page **16. CV/GATE**.

MIDI Output Router routes data to MIDI output ports according to the settings on page **P7. ROUTING**.

Tempo & Clock Engine generates clock ticks for all ARP instances, using internal or external MIDI clock, as declared on **P9. CLOCK** page.

Automation Engine works much like CC automation in a DAW – it processes incoming CC midi messages, checks them against automation table on page **12. AUTOMATION** and, if found, changes the linked parameter value or transmits CC / RPN / NRPN message to a specified output port. CC transcoding functionality is available to modify one type of MIDI Control Message into another.

Chain Engine works when **Chain** is selected. It changes programs according to the sequence programmed on **P5. CHAIN** page.

Meta-Chain Engine finally sends chain or program switch commands to all ARP Instances, with a single touch of a button (see the next chapter for details on **Meta-Chains**).

Workflow and Meta-Chains

BlueARP DM, much like the original BlueARP plugin, is designed primarily as a performance tool.

Let's consider that we want to create a live performance with 4 tracks:

- *arp* (main arpeggiated track)
- *bass* (bass track)
- *seq* (fixed sequence track)
- *drum* (drum loop).

Using **BlueARP plugin**, it works in the following way:

- Create a project in a DAW with 3 instances of the BlueARP plugin

We won't use BlueARP for drums, assume we have a fixed drum loop in our DAW.

- Connect each BlueARP to the VST synth of choice
- Create programs and chains for each BlueARP instance
- Assign **current chain** parameter to a physical knob or a row of buttons for each BlueARP instance

Since we have 3 instances of BlueARP and each one has its own chains, there are two possible ways to automate chain switching:

1) Use 3 knobs or button rows to automate chain for each BlueARP Instance
2) Use 1 knob to change all chains at once

Option 1 is better for studio use to try out different combinations and search for happy accidents.
Option 2 is better for live performance.

Option 2 provides less options. For example, when Chain 2 is selected, this will select Chain 2 for all Instances. However, only one physical control is needed instead of 3.

Option 2 also requires more deliberate chain programming, as chain numbers for each BlueARP instance should be in line with your performance idea.

For example, on Chain 1 may just have an arpeggiated intro, on Chain 2 bass comes in. To achieve this, Chain 1 on the *bass* Instance should be linked to the program "Silent Dummy" (factory program 2, where all steps are set to Off).

With **BlueARP DM**, the idea is the same, with the significant improvement of **META-CHAINS**.

Meta-chain is the central performance element in BlueARP DM. It is basically a Chain over Chains and is used as a tool to change chains for all 8 instances with a single button or knob.

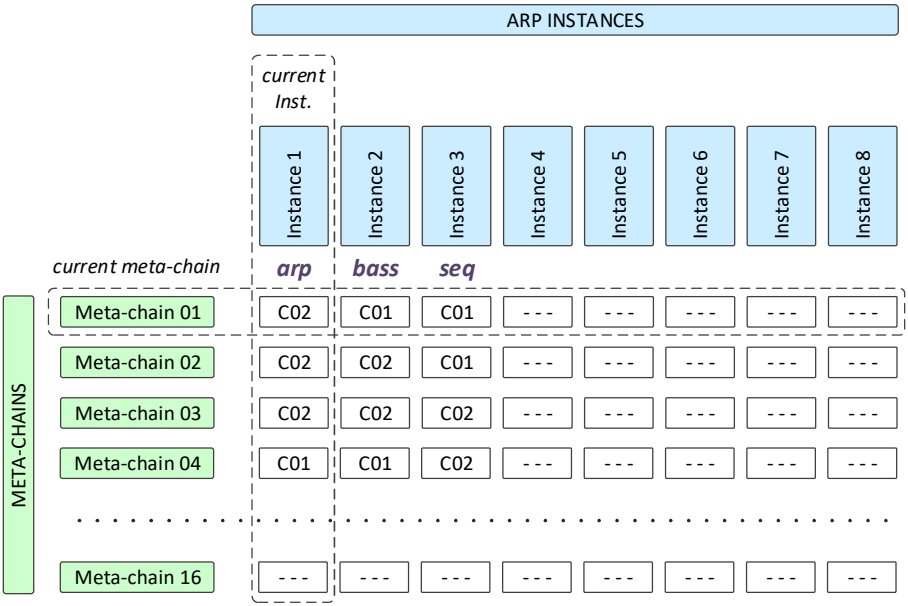


Figure 9. Meta-Chains concept

Columns are **Instances**. They are the equivalent of 8 BlueARP plugins running inside a DAW. The Current instance is what is seen on the screen, other instances remain running in the background.

Rows are **Meta-Chains**; each Meta-Chain is a set of chain numbers for each instance ("C01" means Chain 01 as so on).

Instead of chain number program number can also be set. Scroll to values below "- - -", it will give program numbers like P001, P002 … P128.

When the Meta-Chain is switched, the Meta-Chain engine sends Chain switch command to all Instances.

On the **P6. META-CHAIN** GUI page, pressing step button will switch the meta-chain.

To set up Meta-Chains using *arp, bass, seq* and *drum* tracks from the example on page 24:

For Instances 1-3 Chains are configured in the same way: Chain 1 points to "Silent Dummy" program with all steps having **STEP TYPE** = Off.

Chain 2 points to a program with the sequence.

The Meta-Chain configuration on *Figure 98* works like this:

- meta-chain 01: we have *arp* sequence playing, *bass* and *seq* tracks are muted
- meta-chain 02: *bass* comes in
- meta-chain 03: *seq* comes in, now all 3 tracks are playing
- meta-chain 04: *seq* keeps playing, *arp* and *bass* are muted

For live performances, Meta-Chains can be switched in two ways:

1. Directly from the unit: go to **P6. META-CHAIN** GUI page, press one of step buttons to switch to the respective Meta-Chain
2. Assign "meta-chain" control to the external knob or slider: click Ok on the **meta-chain** control to open the action menu, select **midi learn...** and move the control afterwards. The assignment will be created, this can be later edited on the **12. AUTOMATION** page.

Project data structure

BlueARP project (*.bap file) contains the BlueARP state and all settings. A few parameters such as CV/Gate calibration are stored in battery-backed RAM and not in the *.bap file.

Project (*.bap) file embeds 8 bank (*.fxb) files for each **Instance**. Each bank embeds up to 128 programs (*.fxp files). Banks and programs are compatible with the BlueARP plugin and can be exchanged between the plugin and the BlueARP DM unit.

Projects can be loaded into BlueARP DM only. To load project data into the plugin, save a bank (*.fxb) from BlueARP DM and then load this bank into BlueARP plugin.

Figure 109 shows data blocks inside the project file and how they are nested.

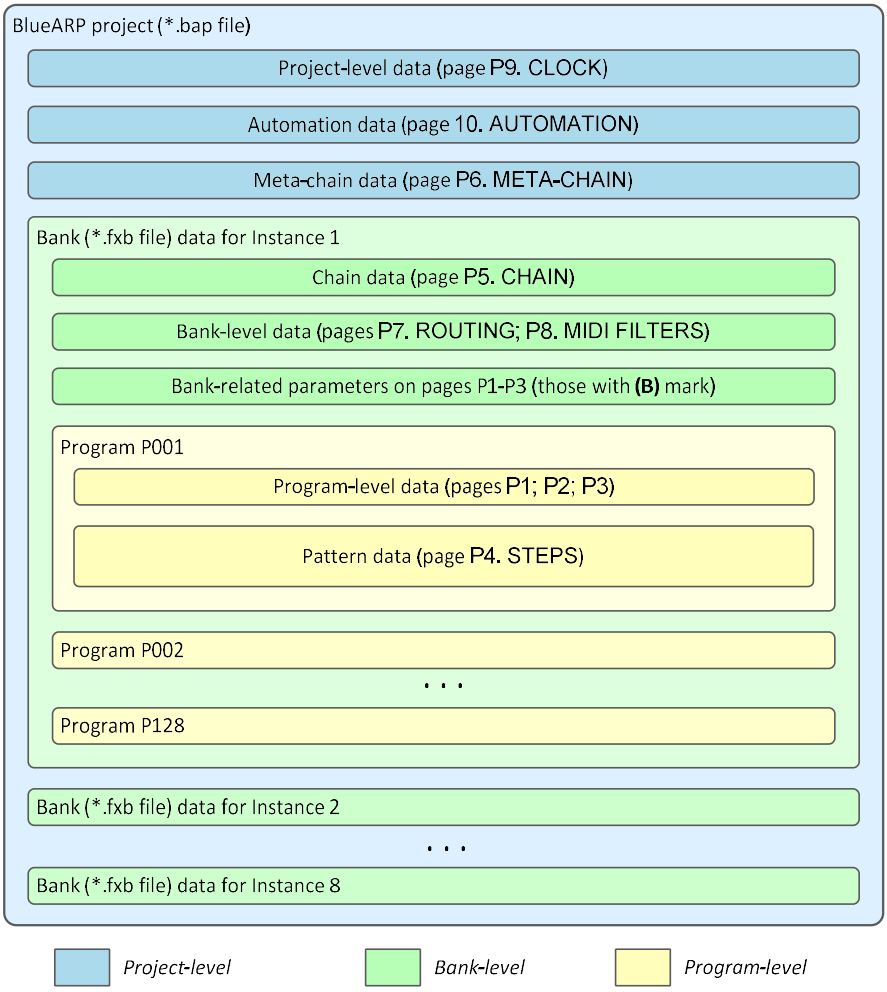


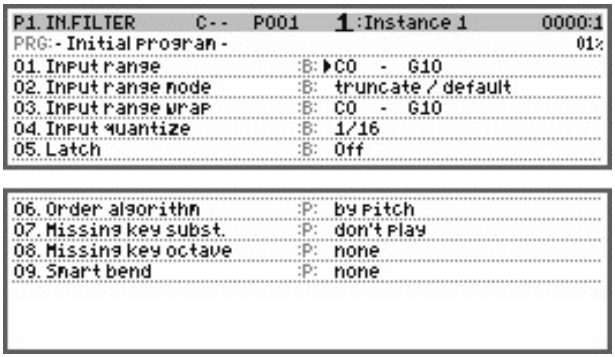
Figure 10. BlueARP project (.bap) data structure*

GUI Pages

The following sub-chapters describe each GUI page.

P1. IN. FILTER

This page is the equivalent of the **INPUT FILTER** block in BlueARP plugin.



Input Filter is responsible for sorting, quantizing and modifying input notes before they enter the Arp Engine block.

LEDs \ STEP BUTTONS

Step buttons	Short press to select a parameter (01 – 09). Hold + wheel to navigate through parameters.
LEDs	Green: step being played, Red: selected parameter number.

ACTION MENU

set default	Reset parameter value to default
midi learn...	Assign parameter value to CC, RPN or NRPN. Assigned link will appear on page **12. AUTOMATION** (see on page 54).
midi unlearn	Unassign automation for this parameter
note input...	Enter key from a MIDI keyboard (for note/key params only)

PARAMETERS

01. Input range: range for filtering input notes

BlueARP will react to MIDI keys only within a given range. All notes outside this range will be ignored. This may be used to create keyboard-split performances using several BlueARP Instances.

Click OK button for context menu, select "**note input...**" to set the value from a MIDI keyboard.

02. Input range mode: adjusts “Input range” behavior

BlueARP may also pass notes outside-the-range as non-arpeggiated.

truncate (default)	Keys outside the range will be ignored
pass thru (no arping)	Keys outside the range will be passed to the output non-arpeggiated

03. Input range wrap: range for input key wrap-around

Unlike 01. Input range setting, this one will not ignore notes outside the range, and instead wrap them into the given range by applying an up or down octave transposition. For example, assume the range is set to C3...C4. Upon pressing 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).

This is sonically useful when chords are played all over the keyboard, however keeps the bass line to sounding right and not too low or too high.

04. Input quantize: input keys real-time quantization

Values are fractions of a bar (1/16 means 16th notes, 1/4 corresponds to 1 beat). For example, at the value 1/4 BlueARP will capture pressed keys on the start of each beat.

When input quantize is on, keys should be pressed a little beforehand, as input keys need to be already captured when the next step/beat occurs.

05. Latch: Latch (hold or sustain) input keys

When checked, BlueARP will continue to play pattern for the last pressed chord even after all input keys are released, and until another key is pressed.

It works much like a sustain pedal, but with one difference – there’s no pedal, input keys are latched and re-scanned automatically.

06. Order algorithm: ordering (sorting) algorithm for input keys

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, to 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)” is best explained by an example. Pressing C4+E4, Cmaj chord is detected. Ordered list will be C4+E4+G4 (a complete Cmaj chord). If an inverted Cmaj – G3+C4+E4 is played the output will be the same, because the chord is normalized.

“chord (as played)” behaves the same way, but inverted chord will stay inverted.

07. Missing key subst.: how to substitute missing keys

When a pattern has more keys than is actually played, this setting will determine whether to mute these steps (i.e., don't play them) or substitute missing keys with the existing ones.

For example, hold C5 and E5, while **KSEL** lane has steps with "k1", "k2", "k3" and "k4".

Page **14. KEY MON** 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 here)

08. Missing key octave: octave transposition for the substituted missing keys

In the example above, if missing keys is set to 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)

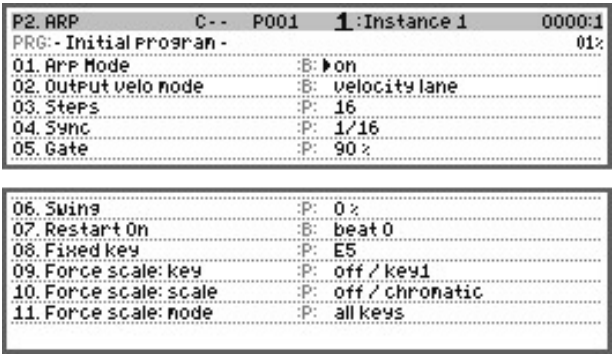
09. Smart bend: input key transposition with a few extra features

This parameter is meant to be assigned to a pitch bend or mod wheel. It will transpose the input key up or down with respect to the selected scale and 'input quantize' setting. The intended usage is to make fast synth lead solos, while keeping it musical: real-time quantized and within a scale.

This feature is experimental in firmware 2.3.8 and may change later.

P2. ARP

This is the same as **ARP ENGINE** block in the BlueARP plugin.



This block takes post-filter keys from the **INPUT FILTER** block and generates the note pattern at the output, according to per-step data on **P4. STEPS** page, with respect to MIDI clock and current song position.

LEDs \ STEP BUTTONS

Step buttons	Short press to select a parameter (01 – 11). Hold + wheel to navigate through parameters.
LEDs	Green: step being played. Red: selected parameter number.

ACTION MENU

set default	Reset parameter value to default
midi learn	Assign parameter value to CC, RPN or NRPN. Assigned link will appear on page **12. AUTOMATION** (see on page 54).
midi unlearn	Unassign automation for this parameter

PARAMETERS

01. Arp Mode: turn arpeggiator On or Off

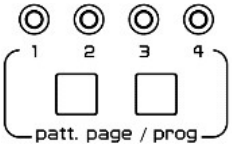
off	BlueARP is inactive, all input keys are ignored
on	BlueARP is enabled, normal mode
thru	BlueARP passes midi notes from input to output without arpeggiating. Some settings still work such as input range, output range, transpose, force to scale. In thru mode the BlueARP can be used as a real-time MIDI transpose tool and/or keyboard range filter.
2k auto on/thru 2k auto on/off 3k auto on/thru 3k auto on/off	Arping will start automatically as soon as at least 2 or 3 keys are pressed. When less keys are pressed, the arp will be muted (on/off modes) or work as pass thru (on/thru mode).

02. Output velo mode: where to take velocity for generated notes

velocity lane	Take from VELOCITY lane.
input key	Take from input key.
lane+input key	Take from VELOCITY lane and adjust to input note velocity (multiply and normalize values).

03. Steps: number of steps

When number of steps is greater than 16, the pattern is divided into pages (4 pages maximum), providing up to 64 steps for a single program.



On “patt. page” led block the current page is shown on the screen (red led) and being played (green led).

Steps = 0 and Steps=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 “non-arpeggiated” scenes.

04. Sync: Step length, as a fraction of a bar

Default value is 1/16, it means 1 step = 16th note. 1/12 is “8th triplets” or “16th dotted”.

05. Gate: output note length, as a fraction of a step length

Varies between 1% and 125%. For 100% and above, output notes will overlap.

06. Swing: swing control

Varies between -50% ... 50%. 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.

07. Restart On: pattern restart trigger

beat 0	Step number is always aligned to the song position. If playback is started from beat 0, pattern will start from the beginning.
key	BlueARP will restart pattern each time new key/chord is pressed, after all previous keys were released.
1st key	Pattern will start with the first key/chord pressed and will keep going until playback is restarted.
play	The same as “beat 0” but aligned to playback start position.

08. Fixed key: input key value for steps with KEY SELECT = “Fix”

In **KSEL** lane, any step can be set to “Fixed”, it tells BlueARP to ignore input keys and take a fixed key value from this parameter.

Set all steps to “Fixed” to use BlueARP as a step sequencer.

09. Force to scale: key: root key for “force to scale” mode

Works together with **10. Force to scale: scale** parameter.

Either set a fixed root for a selected scale or let BlueARP detect it dynamically with “detect from chord” option.

BlueARP recognizes basic chords and chord inversions, so if pressed (E4, A4, C5 - Am inverted), the root key will be A.

10. Force to scale: scale: set scale for “force to scale” mode

Works together with **09. Force to scale: key** parameter.

By setting 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 **11. Force to scale: mode** parameter)
2. “SCALE STEP” lane will transpose notes in scale steps. For example, if the scale is C Major, and D4 is pressed, 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 the chord played. Detection is limited, however it will give “Major/minor” scales for Major/minor chords.

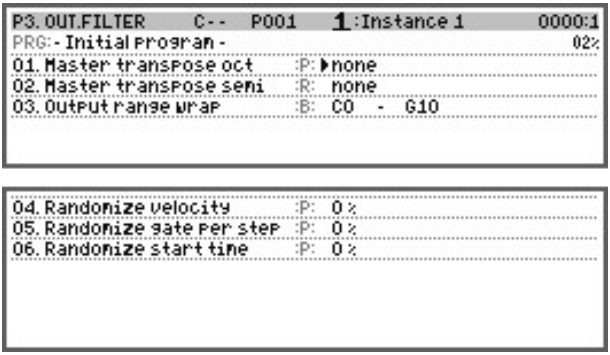
11. Force to scale: mode: how to apply semitone transposition

Works together with **10. Force to scale: scale** parameter.

When set to semi-transposed, force to scale will not be applied to the steps with **SCALE ST** = “-” (zero). This way out-of-scale notes can be still played in force to scale mode.

P3. OUT. FILTER

This is the same as **OUTPUT FILTER** block in the BlueARP plugin.



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

LEDs \ STEP BUTTONS

Step buttons	Short press to select a parameter (01 – 06). Hold + use to navigate through parameters.
LEDs	Green: step being played. Red: selected parameter number.

ACTION MENU

set default	Reset parameter value to default
midi learn	Assign parameter value to CC, RPN or NRPN. Assigned link will appear on page **12. AUTOMATION** (see on page 54).
midi unlearn	Unassign automation for this parameter

PARAMETERS

01. Master transpose oct: output note transposition, octaves

Varies between -3 and +3 octaves.

This setting is program-related, while semitone transposition is bank-related.

02. Master transpose semi: output note transposition, semitones

Varies between -12 and +12 semitones.

This parameter is bank-related, as there is no musical sense to apply different semitone transpositions for different programs.

03. Output range wrap: range for output notes (wrapping)

Notes outside the range will be wrapped (octave-transposed up or down to fit the range). It works like **03. Input range wrap** setting on page **P1. IN. FILTER**, however for the output notes.

04. Randomize velocity: randomize output note velocity

Add a random value (can be positive or negative) to the generated note velocity.

05. Randomize gate per step: randomize output note gate time

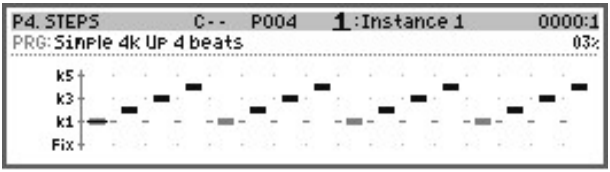
Add random value to the generated note length.

06. Randomize start time: randomize output note start time

Add random delay to the generated note start time.

P4. STEPS

This is similar to the **Matrix Editor** block in the BlueARP plugin.



Value lanes (display B) contain step-related pattern parameters. Selected value lane is shown on the top display.

See descriptions for each value lane below.

LEDs \ STEP BUTTONS

Step buttons	Short press to select a step (1 – 16). Hold + wheel to navigate up and down through lanes. Hold + left/right arrows to move a step
LEDs	Green: step being played. Red: selected step (the one being edited).

ACTION MENU

When a step number is clicked:

Copy ...	Copy a step (memorizes the current step for a future paste operation).
paste	Paste the step overwriting the current one.
delete	Deletes the current step, shifts the rest to the left to fill the gap.
insert initial	Inserts the initial step, shifts the rest to the right.
shuffle steps	Randomly shuffles steps in a current program.

When a "value box" is clicked:

set default	Reset parameter value to default
midi learn	Assign parameter value to CC, RPN or NRPN. Assigned link will appear on page **12. AUTOMATION** (see on page 54).
midi unlearn	Unassign automation for this parameter

PARAMETERS

VELOCITY lane: set velocity for each step

Default value is 96. Use it to set velocity accent for certain steps.

VELOCITY values will be ignored, if the **02. Out velo mode** parameter from **P2. ARP** page is set to “input key”.

GATE lane: gate time multiplier for each step

This lane can be either “gate per step” or “channel per step”, depending on this selector:

GATE - multiplies gate time by a given value on per-step basis.
CHAN – changes output MIDI channel on per-step basis.
“-” means no change (default value) for both modes.

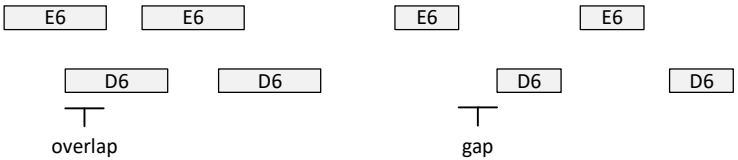
For example, with gate = 60% and **GATE** = “2x” note length will be 60% * 2 = 120%, or 1.2 steps.

STEP TYP lane: several options for output note generation

Off	This step doesn’t generate any note
Nrm	Normal (default) – generates a note
Rst	This step continues to play the note from the previous step. Several “Rst” steps may be chained together to make longer notes.
Tie	This note will overlap with the previous one (for glides)
Chr	Chord, triggers all notes at once
Rnd	Random, picks up a random key from input key list

The “Tie” option’s main purpose is to create “glides” between the notes, however it requires the synth to be configured correctly. The synth should be set to monophonic mode, with legato and portamento on.

When pressed keys overlap (e.g., press key1, press key2, release key1) the pitch of the sound will glide between the notes, but not when pressed with gaps (see the picture below). When the synth is configured in this way the “Tie” steps will create glides, while “Nrm” steps won’t.



KSEL (KEY SELECT) lane: input key selection for a given step

Choose which key to take from the post-filter key list for the current step.

Fixed	Take **08. Fixed key** value from **P2. ARP** page
Root	Take the root key from the detected chord, or key1 if no chord detected
k1...k5	Take keys №1...5 from the post-filter key list

The control to the left from **KSEL** label toggles the lane between monophonic and polyphonic mode. Use cursor buttons to navigate to this element.

The monophonic mode can only select one key for a step, or all keys at once with **STEP TYP** = “Chr” (chord). In polyphonic mode several keys can be selected at once, for example k1+k2 or k1+k3.

OCTV (OCTAVE) lane: add octave transposition for a given step

Varies between -3 and +3 octaves. This is suitable for bass lines, where the steps are usually transposed by the whole octaves.

The control to the left from **OCTV** label toggles the lane between monophonic and polyphonic mode. Use cursor buttons to navigate to this element.

In monophonic mode, all keys for a given step are transposed by the number of octaves. In polyphonic mode only key 1 is transposed.

For example, if **STEP TYP** = “Chr”, **OCTV** = “-1; 0”, and F4 + A4 is pressed; the output notes will be F3 + F4 + A4. (key1 = F4 is copied down an octave, but not key2 = A4)

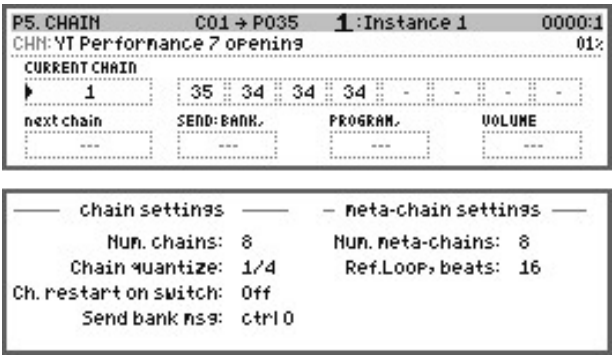
SCALE ST lane: add semitone / scale step transposition for a given step

The **SCALE ST lane** parameter is dependent on “force to scale: scale” parameter. When the latter is set to “off/chromatic”, this lane will work as a semitone transposition. Otherwise, it will transpose the output note with respect to the selected scale.

For example, if force to scale = C Major, press C4 and set **SCALE ST** = +1; the output note for this step will be D4 (not C#4, because force to scale and this lane works in scale steps, not semitones).

P5. CHAIN

This is like the "**current chain**" panel and **PROGRAM CHAINS** block in the BlueARP plugin.



Program chains have the ability to chain several programs together into a longer "super-patterns".

This has been implemented with live performances in mind.

LEDs \ STEP BUTTONS

Step buttons	Short press to change current chain. Hold + wheel to navigate through parameters.
LEDs	Green: pattern step playing. Red: LEDs 1...8: chain playing; LEDs 9..16 – chain step.

ACTION MENU

When "current chain" box is clicked and current chain is value is not "- - -"):

rename ...	Rename current chain
copy ...	Copy current chain (memorize for further paste operation).
cut ...	Cut current chain (memorize for further cut-paste operation).
paste	In copy-paste mode it will overwrite the current chain. In cut-paste mode it moves the chain, removing it from the source location and inserting it into target place.
insert initial	Insert empty chain at the current location, shifting the rest to the right. For example, inserting at chain 3 will shift chain 4 to 5, 5 to 6 and so on.
delete	Remove current chain; shifting the rest to the left to fill the gap.
initialize	Clear current chain.
initialize all	Clear all chains data.
midi learn...	Assign "current chain" parameter to a CC / RPN / NRPN.
midi unlearn	Remove assignment for "current chain" parameter.

When any other parameter is clicked on:

set default	Reset parameter value to default
midi learn	Assign parameter value to CC, RPN or NRPN. Assigned link will appear on page **12. AUTOMATION**.
midi unlearn	Un-assign automation for this parameter

PARAMETERS

current chain: set current chain

"**---**" means no chain selected, 1 – chain 1, etc. Maximum possible value depends on **Num. chains** setting on the same page.

Pay attention to "**Ch. restart chain on switch**" setting on the left panel. If set to "On" the switched chain will always start from the beginning of the 1st step of the program sequence.

program sequence lane: a sequence of program numbers for the current chain.

Use cursor buttons to navigate through program sequence slots.

Program sequence is linked to the **current chain** parameter, switching current chain calls up another program sequence. Chain-related (C) – parameters are only valid with current chain is selected (i.e., current chain value is not "**---**").

next chain: option for next chain auto-switch

If set to anything except "**---**", the BlueARP will automatically jump to another chain after current chain plays once. The options include:

---	Off
caller	Switch back to the chain it was invoked from
caller-1, caller+1	Switch back to the chain it was invoked from, but with the shift to the "caller" chain
chain 1 ... chain 16	Switch to a particular chain after this chain ends

send: bank, program, volume: send MIDI data on chain switch

When specified (not "**---**"), BlueARP will send program\bank change and\or volume change midi messages to the connected synth. This will happen each time on a chain switch. For a given chain, these messages will be sent when this chain is switched from any other chain.

Num. chains: sets maximum value for **current chain** parameter

When automating the current chain parameter, setting **Num. chains** to the appropriate value will utilize the full range of the knob / slider.

Chain quantize: input quantization for chain switching

For a better transition when switching chains it should be done strictly at the start of a new beat. This is achieved using the default setting of chain quantize = 1/4.

Values are fractions of a bar (1/16, 1/8, 1/4, etc.). Value = “none” means no quantization, chain will switch immediately after **current chain** changes value.

Ch. restart chain on switch: restart chain after chain switch

When checked, chain always starts from the beginning after chain switch. Otherwise, when **P2. ARP / 07. Restart On** = “beat 0”, chain step is calculated from song position and chain may start somewhere in the middle of a pattern.

Send bank msg: selects bank/patch change MIDI message format

Works with **SEND: BANK, PROGRAM** setting.

When switching chains, the 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 value “ctrl 0” doesn’t work (synth doesn’t switch banks, only patches), try other options.

Num. meta-chains: sets maximum value for **meta-chain** parameter on page **P6**.

This setting is related to page **P6. META-CHAIN**, it is moved here to save GUI space.

Ref.Loop, beats: reference loop length in beats.

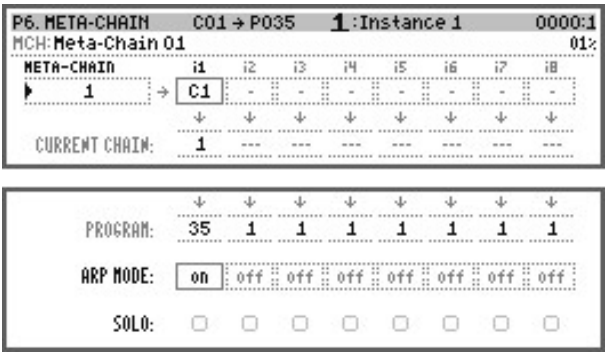
This setting is related to page **P6. META-CHAIN**.

For meta-chains, reference loop is a tool to indicate where the performance loop is right now. When on **P6** page, it will be indicated with the step and pattern page LEDs.

This should be defined manually as the BlueARP will not check if it is set correctly. For example, if there are several instances with chains, having lengths 4, 8 and 32 beats; the reference loop length must be set to 32 beats.

P6. META-CHAIN

These settings are not present in BlueARP plugin.



Meta-chain is the top-level performance element in BlueARP DM.

P6 page is the main page to use during live performances. Here is where meta-chain are switched, switching the chains for all the instances.

LEDs \ STEP BUTTONS

Step buttons	Short press to change meta-chain. Hold + wheel to navigate through parameters.
LEDs	Green: reference loop beat being played. Red: meta-chain being played.

ACTION MENU

When meta-chain parameter is clicked on:

rename ...	Rename current meta-chain
copy ...	Copy meta-chain (memorize for further paste)
cut ...	Cut meta-chain (memorize for further paste)
paste	In copy-paste mode it will overwrite the current meta-chain. In cut-paste mode it will indeed move it removing from the source location and inserting into target place.
Insert initial	Inserts initial meta-chain at the current location.
delete	Deletes meta-chain, shifting the rest to the left.
initialize	Clear current meta-chain.
init. all	Clear all meta-chains.
midi learn...	Assign “meta-chain” parameter to a CC / RPN / NRPN.
midi unlearn	Remove assignment for “meta-chain” parameter.

When any other parameters is clicked on:

set default	Reset parameter value to default
midi learn	Assign parameter value to CC, RPN or NRPN. Assigned link will appear on page **12. AUTOMATION** (page 54).
midi unlearn	Unassign automation for this parameter

PARAMETERS

meta-chain: set current meta-chain

"---" means no meta-chain, 1 – meta-chain 1, etc. Maximum possible value depends on **Num. meta-chains** setting on page **P5. CHAIN** page.

chains for meta-chain lane:

This lane defines which chains to trigger for each **Instance** within a given meta-chain:

"i1" means "Instance 1" and so on.

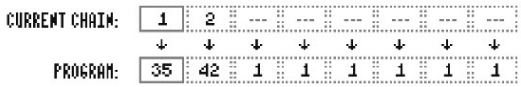
Important: i1...i8 boxes depend on the meta-chain value. When Meta-Chain is switched it will bring up another set of 8 chain values.

Use Meta-Chains to switch all Chains value instead of individually switching 8 chains for each of 8 instances.

To bypass chains and set program numbers directly just decrease the values below "---" and select one of "Pnnn" values, where "nnn" stands for program number.

current chain and program lanes

Chain and program for each **Instance** (currently playing). These are not editable and this is only an indication.



Meta-Chain changes Chains for all Instances, and Chain sets the program (the top-down arrows indicate this dependency).

Current Chain here may differ from the Chain in the previous lane (which is next to Meta-Chain setting) when the Chain is switching but hasn't switched yet. This lane will change only when the Chain switches with respect to "chain quantize" setting.

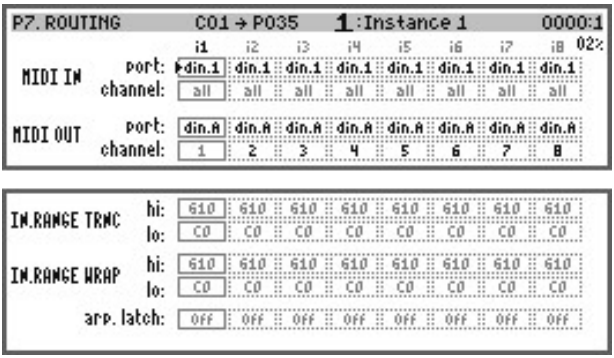
arp mode lane: set arp mode (on, off, thru) for each Instance

This is a shortcut for convenience and is the same as **01. Arp Mode** setting on **P2. ARP** page.

solo lane: solo current Instance

Sets **arp mode** = "on" for the current Instance and "off" for all other Instances.

P7. ROUTING



This page has essential MIDI settings, including input and output ports, channels, ranges.

Parameters on display B are shortcuts, they are also present on pages **P1**, **P2**.

LEDs \ STEP BUTTONS

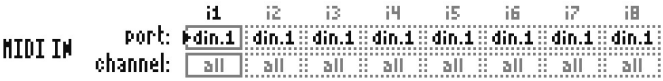
Step buttons	Short press to select instance. Hold + wheel to navigate through lanes.
LEDs	Green: step being played. Red: selected instance.

ACTION MENU

set default	Reset parameter value to default
midi learn	Assign parameter value to CC, RPN or NRPN. Assigned link will appear on page **12. AUTOMATION**.
midi unlearn	Un-assign automation for this parameter

PARAMETERS

MIDI IN port, channel: set MIDI input for the selected Instance



port:

din.1, din.2	DIN 5 MIDI input ports
Dev.1, Dev.2	Virtual MIDI ports for “USB Dev.” (type A) port. For MIDI keyboards, Dev.1 port will likely be the keyboard itself, Dev.2 – external MIDI input.
PC.1, PC.2	Virtual MIDI ports for “USB PC” (type B) port (*). From the PC side, these will be MIDI OUTs to BlueARP, for BlueARP itself they are MIDI inputs.

(*) To use this port, turn on **enable USB PC port** setting on **10. SYSTEM** page. Otherwise BlueARP DM will only use the USB port for power.

channel:

all	Respond to all MIDI channels
1 ... 16	Respond to selected MIDI channel only

MIDI OUT port, channel: set MIDI output for the instance

port:

din.A ... din.D	DIN 5 MIDI output ports
Dev.A ... Dev.D	Virtual MIDI ports for "USB Dev." (type A) port. For synths modules, Dev.A port will likely be the synth itself, Dev.B – external MIDI output.
PC.A ... PC.D	Virtual MIDI ports for "USB PC" (type B) port (*). From the PC side, these will be MIDI inputs from BlueARP, for BlueARP itself they are MIDI outputs.
cvAB / cvAC	Either a CV pitch/gate pair (v.A + v.B outputs) or extended pitch/gate/velocity (v.C for velocity). Depends on "cv mode" setting on page **16 CV/GATE** (see page 63)
cvCD	Another CV pitch/gate pair. Only available when "cv mode" is set to "AB+CD" on page **16 CV/GATE**.

(*) To use this port, turn on **enable USB PC port** setting on **10. SYSTEM** page.

channel:

1 ... 16	Send generated data to selected MIDI channel

IN.RANGE TRNC: input range (truncate)

Duplicates **01. Input range** parameter from **P1. IN. FILTER** page. Normally, incoming keys outside the range will be ignored.

IN.RANGE WRAP: input range (wrap)

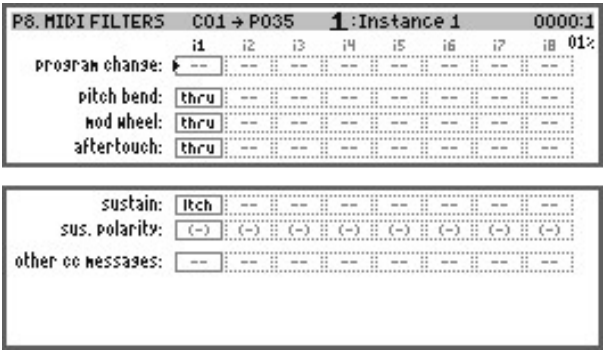
Duplicates **03. Input range wrap** parameter from **P1. IN. FILTER** page. Incoming keys outside the range will be octave-transposed to fit this range.

arp.latch: latch (hold or sustain) incoming keys

Duplicates **05. Latch** parameter from **P1. IN. FILTER** page. Incoming keys will be sustained until another key combination is pressed.

P8. MIDI FILTERS

This is the same as “MIDI FILTERS” block / “SETTINGS” tab in the BlueARP plugin.



This page contains MIDI filtering options and declares how the BlueARP should react to various incoming MIDI CC messages.

LEDs \ STEP BUTTONS

Step buttons	Short press to select instance. Hold + wheel to navigate through lanes.
LED	Green: step being played. Red: selected instance.

ACTION MENU

set default	Reset parameter value to default
midi learn	Assign parameter value to CC, RPN or NRPN. Assigned link will appear on page **12. AUTOMATION** (see on page 54).
midi unlearn	Unassign automation for this parameter

PARAMETERS

program change: how to react to incoming program change message

- -	Ignore
set	Change BlueARP program for a given instance (set own program)
thru	Pass to MIDI output, so the connected synth will receive it

pitch bend: how to react to pitch bend message

- -	Ignore
thru	Pass to MIDI output (to the connected synth)

mod wheel: how to react to modulation wheel message

- -	Ignore
thru	Pass to MIDI output (to the connected synth)

aftertouch: how to react to aftertouch message

this setting applies to both poly and channel aftertouch

- -	Ignore
thru	Pass to MIDI output (to the connected synth)

sustain: how to react to sustain pedal message

- -	Ignore
thru	Pass to MIDI output (to the connected synth)
sust	Common sustain logic. As long as the pedal is held, all incoming notes are sustained. This is same way a piano works.
ltch	Sustain message is linked to “arp.latch” setting (**05. Latch** parameter on **P1. IN. FILTER** page)

sus. polarity: sustain pedal polarity

(-)	normally low
(+)	normally high

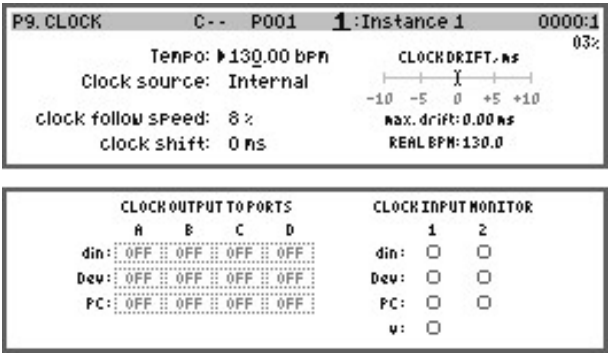
other cc messages: how to react to other CC messages

- -	Ignore
thru	Pass to MIDI output (to connected synth)

P8. MIDI FILTERS settings may interfere with page **12. AUTOMATION**. Be careful to enable CC pass thru if automation is enabled as well. If only a certain MIDI CC messages need to pass, then it is recommended to use automation.

P9. CLOCK

This page has no comparative function in the BlueARP plugin.



This page contains project-related settings, such as Tempo, input and output clock.

This page also shows external clock drift. BlueARP DM can be used to measure external clock stability.

LEDs \ STEP BUTTONS

Step buttons	Short press to select a parameter. Hold + wheel to navigate through parameters.
LEDs	Green: step being played. Red: n/a.

ACTION MENU

set default	Reset parameter value to default
midi learn	Assign parameter value to CC, RPN or NRPN. Assigned link will appear on page **12. AUTOMATION** (see on page 54).
midi unlearn	Unassign automation for this parameter

PARAMETERS

Tempo: set project tempo

Valid when **Clock source** = Internal. For external clock, the tempo will be shown below the **CLOCK DRIFT** indicator.

Clock source: reference clock for song position

Internal	Use an Internal clock source. Clock starts when “play” button is pressed.
Auto	BlueARP will attempt to use clock from MIDI inputs. If no clock present, internal clock will be used.
din.1, din.2	Take clock from DIN5 MIDI input.
Dev.1, Dev.2	Take clock from USB device (via USB Dev. port)
PC.1, PC.2	Take clock from PC (USB PC port)
v.1 (1 ppq) ... v.1 (24 ppq)	Take pulse clock from v.1 analog input (rising edge). Pulses should be at least 2V high, up to 10V is safe.

clock follow speed

If the Clock source is set to external, this parameter will define how fast BlueARP's internal clock will sync to an external clock. Higher values mean faster adaptation to rapid clock changes, but lower values will give better clock stability. In practice, values between 2% and 10% should be fine.

clock shift

Use this setting to compensate delays in multi-device setups when BlueARP follows an external clock. When positive, BlueARP will delay its internal clock by a given number of milliseconds (1 millisecond = 0.001 seconds). When negative, BlueARP's internal clock will come earlier. Clock ticks sent by BlueARP will be delayed / shifted accordingly.

CLOCK DRIFT, ms

If the Clock source is set to external, this indicator shows the de-synchronization between the external and internal clock in milliseconds. Upon receiving an MIDI clock input tick, the BlueARP calculates external song position and compares it with the internal one, and gives the clock drift difference.

When the Clock source is set to PC, the drift indicator will usually swing between zero and 3ms. This drift is normal, as MIDI clocks on PC are known for poor stability. On powerful PCs it may be as good as 1ms. On hardware it is normal to have a drift as low as 0.05ms (5 microseconds).

The BPM value below the drift value is the actual BPM, calculated from the external clock. When synched to PC, it may drift up to 0.2 BPM away from the DAW's BPM value due to clock instability.

max. drift

Indicates maximum drift value in milliseconds.

REAL BPM

The BPM dynamically calculated from input clock. This may differ from tempo on the master device due to clock drift, normally this difference is below 0.2 BPM.

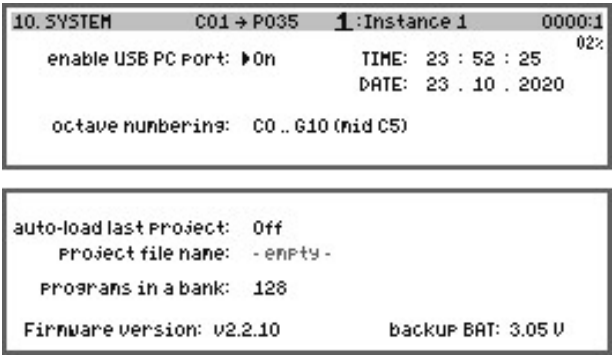
CLOCK OUTPUT TO PORTS

The matrix contains all 12 MIDI output ports, including virtual MIDI outs over USB connections. Set it to "PLAY" to transmit clock only on playback, or "ON" to transmit clock all the time.

CLOCK INPUT MONITOR

Indicates the MIDI clock presence on input ports.

10. SYSTEM



These are system-wide settings stored in battery-powered backup memory.

LEDs \ STEP BUTTONS

Step buttons behavior	n/a.
LEDs behavior	Green: step being played. Red: n/a.

ACTION MENU

set default	Reset parameter value to default

PARAMETERS

enable USB PC port

On	BlueARP DM will be recognized by PC as a generic MIDI class-compliant device (i.e., it will respond to protocol messages via its USB type B port).
Off	BlueARP DM will only draw power from USB type B port and won't be recognized by PC.

octave numbering: set MIDI note naming convention

Values are "C-2 ... G8 (mid C3)", "C-1 ... G9 (mid C4)", "C0 ... G10 (mid C5)".

This tells BlueARP how to display notes or which key is the middle - C3, C4 or C5. This is only used to display note names, it doesn't affect how the ARP engine works.

DATE and TIME

Date and Time are used to write the correct timestamp to saved files (banks, programs and projects). BlueARP engine itself doesn't need a real date and time for operation.

backup BAT

Indicates backup battery voltage. A battery replacement should be considered when below 2.5V. This is CR2032 coin-cell, located on the main board under the “11” and “12” step keys.

Backup battery powers the internal RTC (real-time clock) and a battery-backed RAM, which stores some system-wide settings such as voltage calibration, “auto-load last project” and “enable USB PC port’. Without backup battery BlueARP DM will keep working, but it will lose these settings on each power-down.

auto-load last project

When “On” the BlueARP memorizes the last loaded or saved project (file name will be shown below, “- empty -” otherwise). On power-up, BlueARP will try to load this project from SD Card. To ensure this works, make sure the SD Card is inserted.

programs in a bank

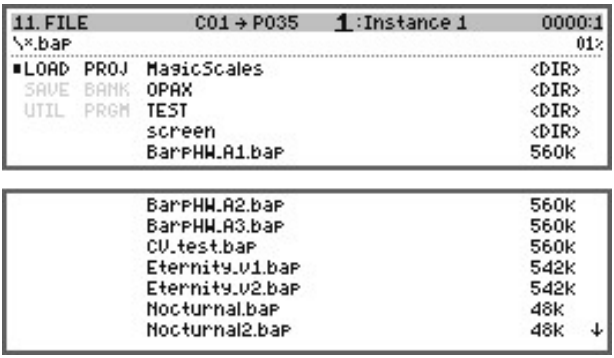
Changing this setting will limit number of programs in a current bank. Use this to speed up the save/load process. This will also speed up USB sync with the control interface.

Firmware version

Indicates the firmware version. Version numbering is shared with the plugin.

To make sure the BluARP DM is 100% compatible with the plugin, use the BlueARP plugin with the same version.

11. FILE



This page enables loading and saving of projects (*bap), banks (*.fxb) and programs (*fxp).

WARNING: Only SD Cards are supported. Pen drives attached to USB port will not work; The **USB Dev** port only accepts USB-MIDI devices.

LEDs \ STEP BUTTONS

Step buttons behavior	n/a.
LEDs behavior	Green: step being played. Red: n/a.

NAVIGATION

Use cursor buttons or wheel to navigate through items. First, select the mode:

LOAD PROJ
SAVE BANK
UTIL PRGM

Select action (LOAD, SAVE, or UTIL to delete files or create folders)

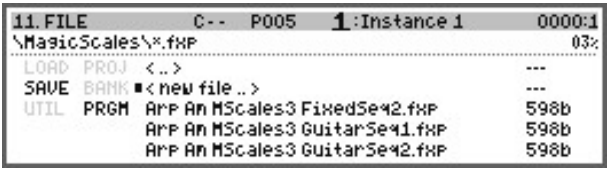
Select file type (PROJ - *.bap, BANK - *.fxb, PRGM - *.fxp)

Navigate to the desired item and click OK to select it; other items will be grayed out.

LOADING FILES

Navigate to the file to load (using cursor or wheel) and click OK.

SAVING FILES



Navigate to <new file...> item and click OK to create new file. To overwrite existing file, navigate to this file and click OK.

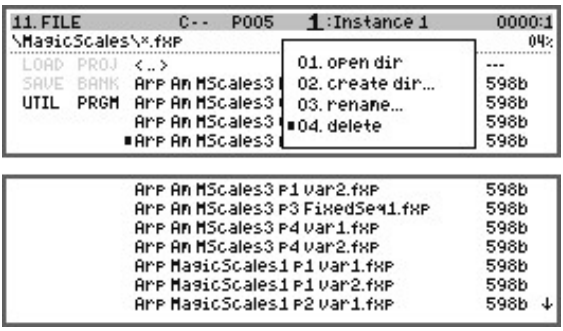
Text editing window will show up when changing the file name. If necessary, click OK to confirm. Cancel will abort the operation.

DELETING AND RENAMING FILES

Switch to UTIL mode first (select UTIL item and click **Ok**).

Then select file type to work with (PROJ, BANK, PRGM).

Locate file in the list to rename or delete, click **Ok**:

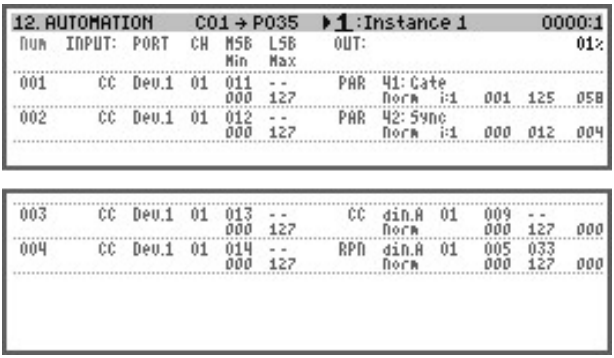


Action menu will appear. Select the desired option and click **Ok** again.

If **03. Rename** is chosen a TextEdit window will appear. Change the file name and click Ok to confirm.

Deleting a file or creating an empty directory can be achieved here as well.

12. AUTOMATION



Automation here works in a similar way to most DAW's.

External controllers can be assigned to change BlueARP's parameters or convert incoming Control Change messages.

LEDs \ STEP BUTTONS

Step buttons behavior	Hold + wheel to navigate through parameters.
LEDs behavior	Green: step being played. Red: n/a.

OVERVIEW

Note for Experts: BlueARP DM can take any CC \ RPN \ NRPN message at any MIDI input, linear-transform it into another CC \ RPN \ NRPN message and send it to any MIDI output according to the settings seen on the screen (ports, channels, input ranges, output ranges, etc.).

Automation engine **doesn't care** about P7. ROUTING settings. This allows for using one port for key input (according to ROUTING settings) and any other port for automation.

This GUI page contains automation records, with each record defining how to process a certain incoming message. This table is complex as the parameters seen depend on the record type. It is important to pay attention to **INPUT** and **OUT** columns.

INPUT defines what to take at the input (i.e., CC, RPN or NRPN message), then **OUT** defines what is done with it. This can either be used to set an internal BlueARP parameter or convert it to another CC \ RPN \ NRPN message and send it to an output port.

Background information

This page frequently uses the terms MSB and LSB. MSB stands for Most Significant Byte, and LSB is least significant byte. Note that in MIDI, these are truncated bytes, they are 7 bits long and take values between 0 and 127 respectively (while a normal 8-bit byte has 0 – 255 value range).

The simplest case is CC message, it has a 7-bit number (MSB only) and a 7-bit value (MSB only as well).

RPN's and NRPN's are technically an ordered set of CC messages. They always have 14-bit number (MSB and LSB), however the value can be either 7-bit (MSB only) or 14-bit (MSB and LSB).

14-bit numbers and values can be denoted as 2 separate numbers (MSB and LSB) or as a single 14-bit number. They have a mathematical relation:

*Value14bit = ValueMSB * 128 + ValueLSB*

Example: MSB = 20, LSB = 35, 14-bit value = 20*128+35 = 2595.

RPN's and NRPN's increase the number of steps to make fine grained adjustments (14-bit value gives 16384 steps, versus 128 steps for 7-bit).

If the above is not clear, please take some time to read the resources at https://midi.org or look online.

ACTION MENU

add new...	Create new automation assignment: "waiting for midi CC input" message will appear, move knob / slider or button on the controller to be assigned. New lane will be added to the end of the list.
reassign...	Reassign current automation lane to another input CC, RPN or NRPN message. Target CC / Parameter assignment will stay intact.
clone	Duplicate current automation lane
delete	Remove current automation lane
delete all	Remove all automation lanes

PARAMETERS

INPUT:

Incoming MIDI message type. Can be:

CC	Regular CC message, it has 7-bit number (MSB only) and 7-bit value (MSB only as well).
CC-W	Wide CC message*, it has 7-bit number (MSB only) and 14-bit value, MSB and LSB. This is an ordered pair of CC messages, first one carries MSB value, second one - LSB value. Second message CC number always has +32 number offset.
RPN	RPN message, it has 14-bit number (MSB and LSB) and 7-bit value (MSB only).
RPN-W	Wide RPN message, it has 14-bit number (MSB and LSB) and 14-bit value (MSB and LSB as well).
NRPN	NRPN message, it has 14-bit number (MSB and LSB) and 7-bit value (MSB only).
NRPN-W	Wide NRPN message, it has 14-bit number (MSB and LSB) and 14-bit value (MSB and LSB as well).

(*) To be compliant with MIDI 1.0, use numbers 1 – 31, except number 6 (which is used as a part of RPN / NRPN messages).

PORT, CH

Input port and channel. Can be set to “any” or to specific values.

MSB, LSB (input)

Incoming CC \ RPN \ NRPN number.
For CC messages, LSB is not applicable; it is grayed out and not editable.
For CC-W messages, LSB is calculated automatically (as MSB+32) and is also not editable.

Min, Max (input)

Input range for filtering incoming CC \ RPN \ NRPN messages by their MSB value. Default range is 0 – 127, it means no filtering. If set to 0 – 63, then incoming values greater than or equal to 64 will be ignored. This is useful to make one knob to do several things. For example, 1st half of the rotation range changes filter cutoff while the second half adds reverb.

OUT:

Tells BlueARP, what to do when the automation lane “fires”:

NONE	Ignore, do nothing. Use it to suspend automation lane.
PAR	Change BlueARP’s internal parameter
CC	Regular CC message: MSB number, MSB value
CC-W	Wide CC message*: MSB number, MSB+LSB value
RPN	RPN message: MSB+LSB number, MSB value.
RPN-W	Wide RPN message: MSB+LSB number, MSB+LSB value.
NRPN	NRPN message: MSB+LSB number, MSB value.
NRPN-W	Wide NRPN message: MSB+LSB number, MSB+LSB value.

(*) To be compliant with MIDI, use numbers 1 – 31, except number 6 (which is used as a part of RPN / NRPN messages).

MODE

Defines how BlueARP will change the output parameter/value:

Norm	Normal, linear transform of the input value to the output. This is the default option to use with the physical knobs at the input.
Dec1	Each time BlueARP gets the input message, it will decrease the output value by 1. This is good to use with the physical buttons at the input. Normally a pair of buttons are configured to increment and decrement.
Inc1	The same as Dec1, but BlueARP will increase the output value by 1.
Togl	Toggles the output value between Min and Max output values. This is good to control binary parameters (like something On\Off) with a single physical button.

PARAM

Only available when OUT = PARAM

BlueARP’s internal parameter to be changed.

Parameter numbers and names are shared with those in the BlueARP plugin.

INS

Only available when OUT = PARAM

Parameter can be changed for the certain instance (i1 – i8), for the current instance, or for all instances at once.

PORT, CH

Only available when OUT = CC, CC-W, RPN, RPN-W, NRPN, NRPN-W

Port and channel for the output message

MSB, LSB (output)

Only available when OUT = CC, CC-W, RPN, RPN-W, NRPN, NRPN-W

Sets MSB and LSB number for the output MIDI message. For CC and CC-W messages, LSB is not editable.

Min, Max (output)

Sets value range for the output message (or for the parameter value).

For wide (14-bit valued) messages, applies to MSB value only.

Use this feature to limit the target value range. For example, when automating filter cutoff, narrowing the range will prevent the value from going too low and closing the filter completely.

Cur (output)

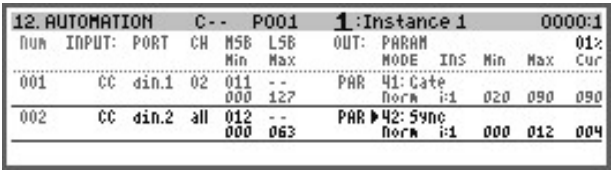
Current output value. For CC \ RPN \ NRPN, BlueARP memorizes the last sent value to avoid duplicate CC values and the output.

Normally, this is not meant to be changed. However if this is changed the BlueARP will immediately send the corresponding MIDI message, or will change the target parameter.

EXAMPLES

Here are some examples of the automation lanes and how to read them.

Example 1. Automating parameter change.



Lane 001:

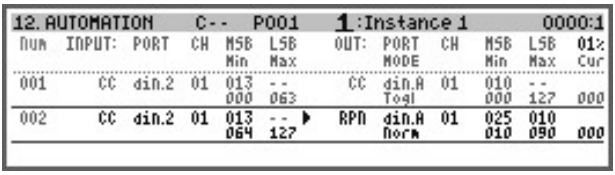
Input message CC#11 at port din.1, channel 2.
Will change Gate parameter within range 20% - 90%.

Lane 002:

Input message CC#12 at port din.2, any channel, with a value = 0 ... 63*.
Will change Sync parameter within range 0 - 12.

(*) The same CC message with values 64 ... 127 will be ignored.

Example 2. Automating other synths (CC transcoding).



Lane 001:

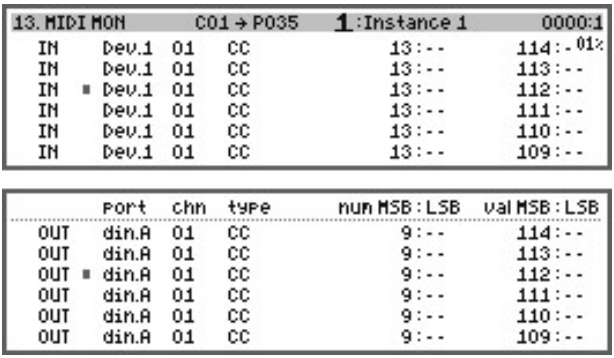
Input message CC#13 at port din.2, channel 1, with a value = 0 ... 63.
Will trigger output message CC#10 at port din.A, channel 1, value will be toggled between 0 and 127 each time input message is received.

Lane 002:

Input message CC#13 at port din.2, channel 1, with a value = 64 ... 127.
Will trigger output message RPN 25:10 (MSB number = 25, LSB num = 10) with values within range 10 - 90.

Since the input range here is 64...127, the value will be linear-transformed. Input value 64 will trigger output value 10, input value 127 >> output value 90, the values in between will be transformed proportionally.

13. MIDI MON



This is an indication page and doesn't have any settings.

It shows input and output activity on all MIDI ports.

LEDs \ STEP BUTTONS

Step buttons behavior	n/a.
LEDs behavior	Green: step being played. Red: n/a.

OVERVIEW

Display A shows MIDI input activity, while display B shows MIDI output.

Use this page to test MIDI connections. For example, if connected to a MIDI keyboard via **USB Dev** port, "NoteOn" and "NoteOff" input messages should display as soon as keys are pressed on this keyboard.

This page shows activity on all MIDI ports, it doesn't depend on input port and range settings. To check if the ARP is receiving the incoming notes, check **Keys post-filter** value on **14. KEY MON** page.

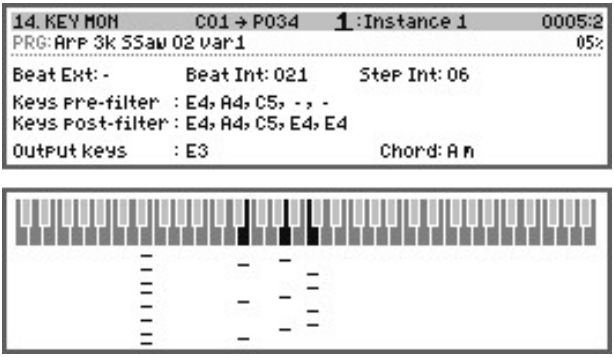
Use this page to check which CC number each knob on the controller is transmitting. This page will also show Note events and all midi controller messages such as CC, RPN and NRPN messages. This does not show SysEx and real-time messages like play, stop and MIDI clock.

Display B will show not only generated notes, but also CC messages that were passed thru according to the settings on **P8. MIDI FILTERS** page.

MIDI MON page will show RPN and NRPN as a single message, not as a series of CC messages.

14. KEY MON

This page is similar to the Information Panel in the BlueARP plugin.



This is an indication page and doesn't have any settings.

It shows input and output keys for the selected instance, current beat, and step position.

LEDs \ STEP BUTTONS

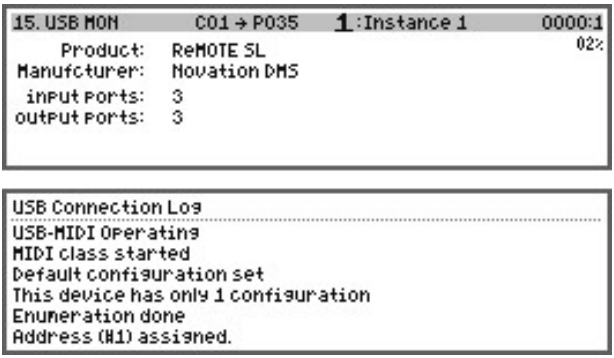
Step buttons behavior	n/a.
LEDs behavior	Green: step being played. Red: n/a.

ELEMENTS

Beat Ext	Song position in beats from the reference clock (depends on "Clock source" setting on **P9. CLOCK** page).
Beat Int	Adjusted song position for step calculation. For example, in restart on key mode Beat Ext will start running with playback. Beat Int clock will only start running when first key is pressed.
Step Int	ARP step (usually it is also indicated with green LED)
Keys pre-filter	Input keys before filtering and quantization. Check if input ports are configured correctly and BlueARP receives incoming MIDI notes.
Keys post-filter	Keys after "Input Filter" block, i.e., after range filtering, quantization and missing keys substitution.
Output keys	Notes generated by the ARP
Chord	Detected chord, (?) if failed to detect

On the keyboard below, input keys are highlighted, falling dots represent output keys.

15. USB MON



This is an indication page and doesn't have any settings.

Shows basic information about USB-MIDI device connect to the **USB Dev** port.

LEDs \ STEP BUTTONS

Step buttons behavior	n/a.
LEDs behavior	Green: step being played. Red: n/a.

ELEMENTS

Product, Manufacturer

Values reported by the connected USB device.

input ports

Number of input ports USB device reported. These ports are addressed as **Dev.1** and **Dev.2**. BlueARP DM supports 2 input ports maximum. If the device has more than only the first two can be used.

output ports

Number of output ports USB device reported. These ports are addressed as **Dev.A**, **Dev.B**, **Dev.C** and **Dev.D**. BlueARP DM supports 4 output ports maximum. If the device has more only the first four can be used.

USB Connection Log

Shows USB device detection progress. It may be useful for troubleshooting: if there are some problems with a device the log will show which step it is stuck on.

On success, the last topmost message should be "USB-MIDI Operating".

16. CV/GATE

OVERVIEW

This page has settings related to voltage output ports (**vA .. vD**)

Background information

CV/GATE was the common way of controlling hardware synthesizers before MIDI took over in 1980's. Many older synths have CV/GATE inputs. Now CV/GATE is widely accepted in modular synths, including popular eurorack format.

Typical CV/GATE connection to a synth employs the following voltage signals:

Pitch CV	Note pitch voltage. There are 3 common standards: • **1V/oct** (1 volt per octave, works like 0V: A1 note, 1V: A2, 2V: A3); • **1.2V/oct** (1.2 volts/octave, 0.1V for a semitone); • **Hz/V** (to rise the pitch up an octave, the voltage should be doubled, like 1V: A1, 2V: A2; 4V: A3; 8V: A4)
Gate	Voltage for note on/off event. Simply speaking, gate is on while the note is sustained. Typical voltages are: 5V for gate on, 0V for off.
Velocity CV (optional)	Voltage for note velocity. The harder the key is pressed, the higher is the voltage. Often it is linked to filter cutoff or envelope gain, so higher velocities produce brighter or louder sound.

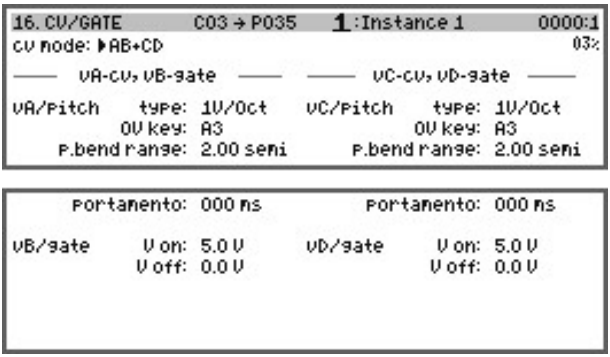
CV/GATE interface in BlueARP DM is monophonic, so, polyphonic features will not work when using CV/GATE output.

BlueARP DM has 2 possible CV/Gate configurations via **cv mode** setting:

AB + CD	2 CV/Gate pairs: v.A is a pitch CV + v.B is a Gate, v.C is a pitch CV + v.D is a Gate
AC + D	1 CV/Gate/Velo triple: v.A is a pitch CV, v.B is a Gate, v.C is velocity CV; v.D becomes a pulse clock output.

Other settings will depend on **cv mode** value.

cv mode = AB + CD



There are identical sets of parameters for **AB** and **CD** CV/Gate pairs.

type

Selects the pitch voltage standard:

1V/Oct	One volt represents 12 semitones (one octave). So, the pitch produced by CV = 0V is one octave lower than that produced by a voltage of 1V. Negative voltages are possible as well. -1V will be 2 octaves lower than 1V.
1.2V/Oct	One octave is 1.2 volts, each semitone is 0.1V.
Hz/V	One volt represents a fixed frequency difference in Hertz. To rise the pitch up one octave, the voltage should be doubled, like 1V: A1, 2V: A2; 4V: A3; 8V: A4.

0V key

Sets the reference key for a pitch voltage. For **V/Oct** standards, this key will give 0 volts at the pitch CV output. For **Hz/V** standard, negative voltages are not possible (and 0V is practically not possible too), so the reference key corresponds to 1.0 volts pitch CV.

p.bend range

Pitch bend range in semitones. Pitch bend for CV will only work if pitch bend = ”thru” on page P8. MIDI FILTERS

portamento

Portamento time in milliseconds to raise the pitch by 1 octave (zero value means no portamento).

Gate, V on

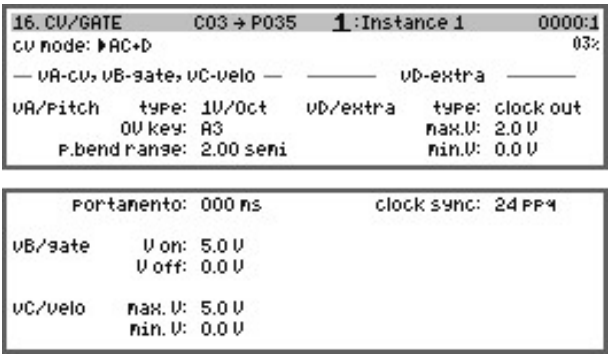
Gate ON voltage. For eurorack modules, default value 5V should be fine. Some older synths may require voltages up to 10V.

Gate, V off

Gate OFF voltage. Default value 0V should be fine for most cases.

The gate signal can be inverted if the **V on** is set to 0V and **V off** for 5V, for example. These can be set to negative as well.

cv mode = AC + D



For **AC** mode there are additional setting for velocity CV and clock output at vD.

All other settings are the same as in previous mode.

vC/velo: max, V and **min, V**

Voltage range for velocity. Default range 0V ... 5V should be fine for eurorack environment. The widest possible range is -10V…+10V.

vD/extra: type

Defines vD output type:

clock out	**v.D** becomes clock pulse output with a frequency defined by clock sync setting (cycle length as a fraction of a bar).

No other alternatives yet, may be expanded in the future updates.

vD/extra: min.V, max.V

Voltage range for vD output. For type = “clock out”, sets minimum and maximum voltage for the clock pulse signal. The signal can be inverted by setting max.V=0 and min.V=5.

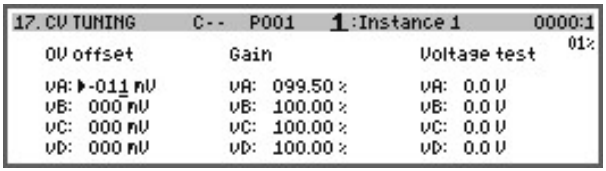
vD/extra: clock sync

Output clock relative frequency in PPQ (Pulses Per Quarter note). Pulse amplitude is defined by vD/extra: min.V and max.V parameters.

17. CV TUNING

OVERVIEW

Due to some imperfections of the analog circuitry, real output voltage on analog outputs may differ from the expected value by up to 1-2%. To have perfect voltage on the analog outs, it is recommended to use calibration. Especially this is important for **v.A** and **v.C** outputs, which are used as a pitch CV.



Calibration only needs to be done once. The voltage offset and gain settings (shown on the page above) will be stored in coin-cell powered memory.

Perform the following procedure for each of **v.A ... v.D** outputs:

1) Connect a cable to **v.A** output, prepare a multi-meter to measure DC voltage within a range around -10V ... +10V
2) When **V.test, v.A** = 0.0V, the voltage at **v.A** port should be exactly 0.0V. If this is not the case, adjust **Ofs, v.A** value to get actual voltage as close to 0 as possible
3) Change **V.test, v.A** to 3.0V, adjust **Gain, v.A** to get the voltage as close to 3.0V as possible
4) Repeat the procedure for outputs **v.B**, **v.C**, **v.D**.

For step 3), other voltages can be used as well, depending on the voltage range . Voltage outputs have some non-linearity over a full voltage range.

Make sure the multi-meter is set to measure DC voltage and not AC otherwise the wrong readings will occur.

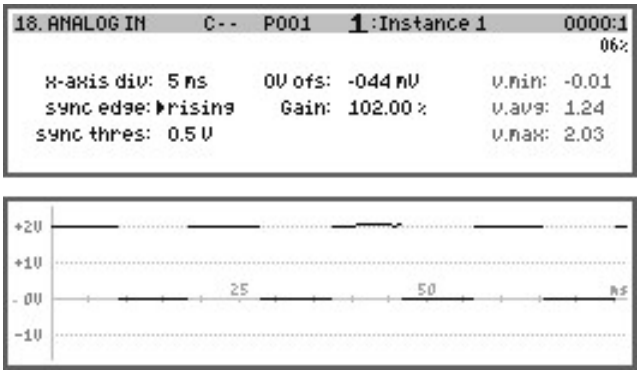
Note that multi-meters are not 100% precise and some fine adjustments made by ear may be required with the synth connected via CV\Gate. One way to do this while staying on this page:

1) Set **vB** to 5V (or whatever gate voltage is enough for a connected synth)
2) Use a reference tone for 0V pitch, adjust **vA** offset to tune the synth to the reference tone
3) Adjust **vA** by a whole volts and check it against the reference tone again. Adjust vA Gain to get the best possible match within a few octaves

18. ANALOG IN

OVERVIEW

BlueARP DM has one analog input for input pulse clock synchronization and possible future extensions. It has a range from -1V to +2V, maximum safe voltage is ±15V. The analog input is good for reliable pulse clock detection, however may be too noisy for input control voltages.



v.1 input has internal pull-up resistor, so when left floating, it will measure about +1V (and 0V when shortened). This makes possible to connect a pedal switch directly to detect state.

PARAMETERS

x-axis div: horizontal axis time division.

Values from 1 to 10 milliseconds.

sync edge: synchronize display to the input signal egde

none	no sync
rising, falling	synchronize to rising or falling edge of the input signal (this setting only affects display). Works together with sync thres setting.

sync thres: threshold voltage for the edge detection

This sets the voltage level for edge detection when sync edge is set to “rising” or “falling”.

0V ofs: input calibration, zero voltage offset.

Use this setting if 0V at the input doesn’t give an exact 0V reading, the 0V point can be shifted with this setting.

0V ofs: input calibration, gain adjustment.

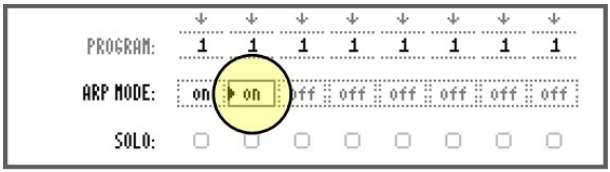
Use this setting to adjust input gain for more precise voltage reading.

Tips and Tricks

Running multiple arp instances

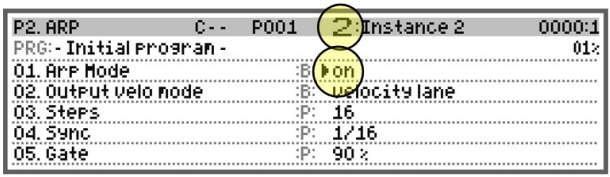
BlueARP runs 8 Instances in parallel. By default, when the unit is powered up without an SD card inserted, Instance 1 will be active, and all others are muted.

To un-mute other Instances, go to **P6. META-CHAIN**, bottom display:



Set ARP MODE to “on” for the Instances needing to be un-muted. Make sure SOLO is off (i.e., none of the boxes from SOLO lane is checked).

Alternatively, go to **P2. ARP** page, select the desired Instance with “instance” buttons and change “Arp Mode” to “on”:



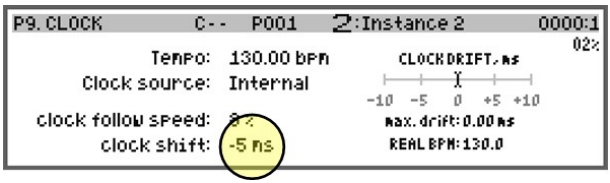
After that, set the appropriate MIDI input and output ports/channels for each instance on **P7. ROUTING** page. Set the input ranges if necessary (input range settings on **P7. ROUTING** page are shortcuts to those on **P1. IN. FILTER** page).

Dealing with latency issues

Shifting internal clock against external clock.

When BlueARP follows the external MIDI clock, external clock can compensate for latency with the “clock shift” parameter. This may be necessary when synchronizing BlueARP DM to the PC or Mac, and the master clock is received from a DAW such as Ableton or Logic Pro.

On **P9. CLOCK** page, set Clock source to the respective input port and adjust “clock shift” value:



This will make BlueARP clock come 5 milliseconds earlier than the external clock.

Using DIN5 MIDI outs.

The generic advice here is to avoid midi device chaining and use separate MIDI out ports when possible.

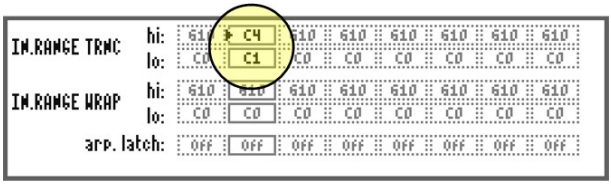
Keep in mind that MIDI protocol is quite slow; it is basically a serial protocol with baud rate of 31500. Sending one note on/off event takes around one millisecond, 10 msec latency may be quite audible in some cases.

Or at least use separate MIDI out ports for the synths needed to be strictly in time (like bass lines and fast arp sequences). For the synth lines like pads and drones additional few milliseconds delay won't be audible and it is OK to chain them.

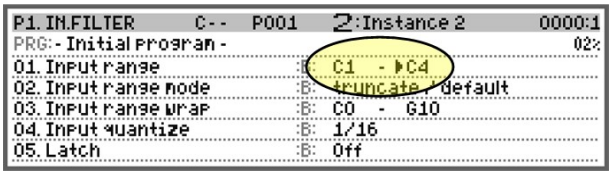
Keyboard splitting and layering

After setting up multiple Instances, additional filtering can be added with the "Input range" parameter.

There are two ways to do this: either set "IN. RANGE TRNC" (input range / truncate) parameter on **P7. ROUTING** page, bottom screen:



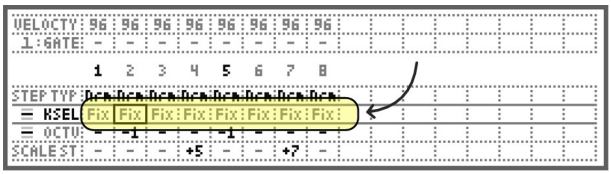
Or select the desired Instance with "instance" buttons and change "Input range" setting on **P1. IN.FILTER** page:



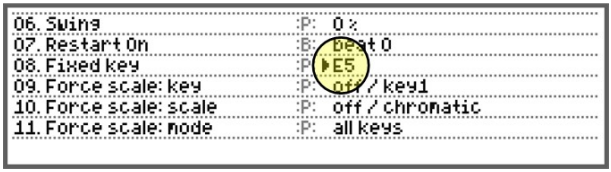
Also pay attention to "Input range mode" and "Input range wrap" parameters (see page 28).

Using BlueARP as a step sequencer

On **P4. STEPS**, set KEY SELECT lane values to “Fix” for all steps:



The generated output will not depend on the input keys. Instead it will depend on “Fixed key” parameter from **P2. ARP** page:



When playback is On, BlueARP will generate output notes even when no input keys are present in the way step sequencers behave.

Appendices

Specifications

Physical specs

DIMENSIONS	202 (W) x 40 (D) x 170 (H) mm 7.95 (W) x 1.57 (D) x 6.69 (H) inches
WEIGHT	0.7 kg / 1.5 lbs. (unit itself) 1.0 kg / 2.2 lbs. (packed)
POWER	5V DC via USB type B port Current: 600 mA (*)

(*) When BlueARP DM is powered from the PC, it reports itself as a generic USB-MIDI device with 500 mA current consumption. It is safe to connect USB devices to BlueARP which will draw up to 200 mA current, otherwise it is better to power BlueARP from a wall adapter.

Connectivity

DIN5 MIDI ports	2 MIDI IN ports, 4 MIDI OUT ports
USB type B (USB-MIDI to PC)	1 USB type B port, 2 virtual MIDI IN ports (from PC to BlueARP), 4 virtual MIDI OUT ports (from BlueARP to PC)
USB type A (USB-MIDI to device)	1 USB type A port, 2 virtual MIDI IN ports (from USB device to BlueARP), 4 virtual MIDI OUT ports (from BlueARP to USB device)
v.A ... v.D voltage outs, 6.35” mono jacks	4 voltage output ports with range -10 ... +10V, 16-bit precision and short-circuit protection. Each output acts as a single CV or Gate, depending on settings.
voltage in, 6.35” mono jack	1 voltage input with range -2.5 … +2.5V and 12-bit precision, tolerating up to ±15V.
SD Card	1 SD Card slot, supporting SD, SDHC and SDXC cards with FAT or exFAT file system

Technical specs

Buttons	30 tactile buttons
LEDs	21 dual color red-green LEDs, 1 white LED
Displays	Two 3.12” OLED displays with resolution 256x64 pixels, 16-level grayscale
Processing Unit	STM32F407 MCU running at 120 MHz
Memory	128 Kbytes on-chip SRAM memory, 1 Mbyte external SRAM memory 1 Mbyte on-chip Flash memory

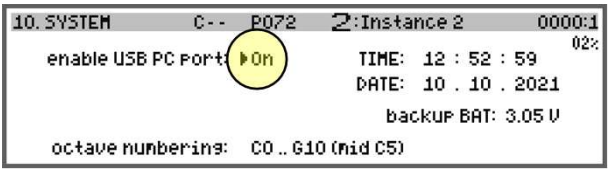
BlueARP Control Software

BlueARP Control Software is a PC/Mac companion application for BlueARP DM. It allows for the editing of patterns and parameters in the same way as in the plugin, using almost the same GUI, but with a few extra panels and controls.

It helps if you have used BlueARP plugin before. If not, pay attention to BlueARP plugin manual: it is relevant for the control software as well, as the GUI is almost the same.
BlueARP plugin manuals are available for download here:
https://omg-instruments.com/wp/?page_id=46

Establish connection

Make sure “enable USB PC port” is set to “on” on page **10. SYSTEM**:

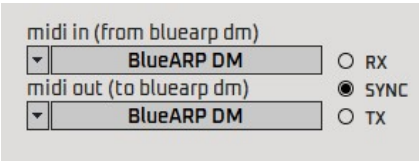


Otherwise, the unit will only draw the power from USB port and won’t be recognized as a USB-MIDI device.

Pre-conditions:

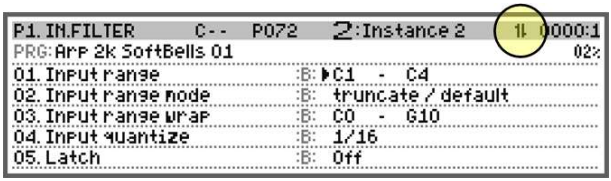
1. BlueARP DM should be connected to PC/Mac via USB cable
2. On the PC side, BlueARP MIDI ports “PC.1” and “PC.A” should be free (not occupied by any DAW or other application). The BlueARP Control software will use these ports to transmit SysEx configuration data
3. It is recommended to have the same version of the software/firmware inside BlueARP DM and the control application (version numbering matches).

Launching the BlueARP Control application will try to select the appropriate MIDI ports automatically:

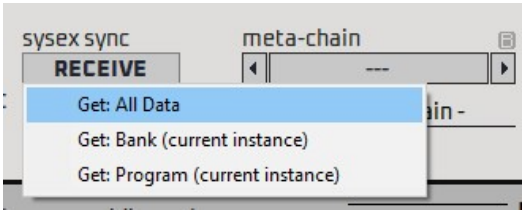


“BlueARP DM” should be selected in both boxes.

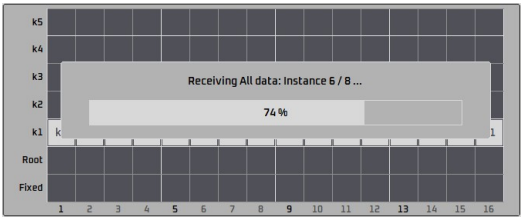
Once the two-way synchronization is established, “USB sync” should display indicator on the top display of the unit:



The next step is to synchronize the state between the unit and the control software. On the top panel of the application by selecting RECEIVE: Get All Data:



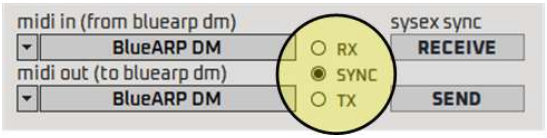
The gauge indicating data transfer progress should now show:



It should take about 5 seconds. BlueARP DM will send all of its machine state to the control software.

The BlueARP DM and the Control software should now be in sync. The synchronization is bi-directional allowing for the adjustment of controls in the application or the DM unit, where they will be immediately duplicated on the other side.

There are 3 indicators on the top panel:

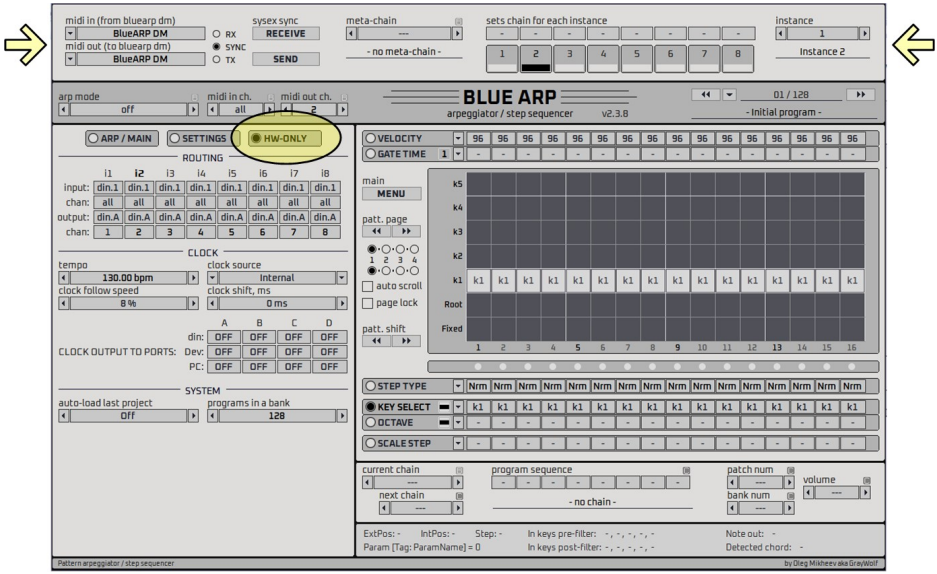


“SYNC” shows the link is alive (the app polls the unit few times a second to check whether it is responding).

“RX” indicates data reception on the app side; “TX” indicates sending data from the app to the unit.

Interface overview

BlueARP control application looks similar to the BlueARP plugin, with a few extra panels and controls:

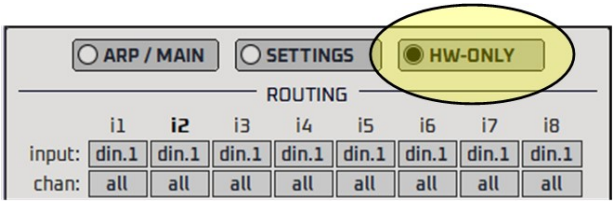


A new panel has been added to the top. The panel allows the setting of MIDI ports, initiate SysEx transfer (send or receive), set Meta-Chain, and selection of the Instance.

Clicking on one of these labels will change the active Instance:

Everything below the top panel will be updated according to the selected Instance. The 8 independent BlueARP plug-ins (or Instances) are located here, however only one Instance is editable at a time.

An extra “HW-ONLY” tab is located on the left panel:



This tab contains hardware-related settings found on pages **P7. ROUTING**, **P9.CLOCK** and **P10. SYSTEM**.

This tab contents doesn’t depend on the selected Instance.

Saving and recalling projects

The projects can be saved and loaded in two ways:

1. On the unit itself to the SD Card inserted into the unit
2. In the control software, via MENU -> Project -> Load/Save

The current software limitation is that projects cannot save to the SD Card via the control software. This can only be done from the unit itself.

If projects loaded/saved in the control software work similarly to the plugin. These projects can be saved to SD Card and loaded from the unit. Individual banks and programs can be loaded as well, just like in the plugin.

Once anything is loaded inside the app, it will be automatically transferred to the unit, to make sure they stay in sync. The same will happen when anything is loaded inside the unit while USB sync is active.

If a project is loaded in the control app, it will be automatically transferred to the unit and will overwrite whatever was there.

Known issues and limitations

MIDI port issues on Windows 10.

Control software relies on Windows MM libraries which aren't officially supported by Microsoft in Windows 10. This may interfere with some other MIDI devices connected via USB.

For example, while beta testing it was reported that BlueARP control software doesn't work together with Roland MX.

Some controls aren't present in the control software

These pages are only available on the unit itself:

1. Page **11. FILE** (load/save files to SD Card)
2. Page **12. AUTOMATION** (edit automation data)
3. Page **16. CV/GATE** (CV/Gate ports configuration)
4. Page **17. CV TUNING** (CV/Gate voltage calibration)
5. Page **18. ANALOG IN** (v.1 analog input configuration)

Some of these pages may be implemented in future updates, depending on the user feedback.

Updating the embedded firmware

Firmware

BlueARP DM firmware can be updated via SD Card.

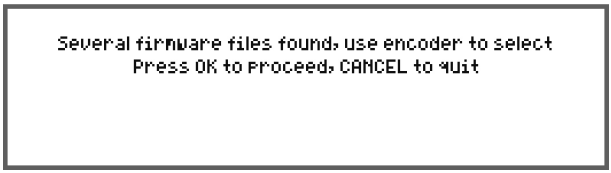
To check current firmware version, go to page **10. SYSTEM**:

To update the firmware, proceed as follows:

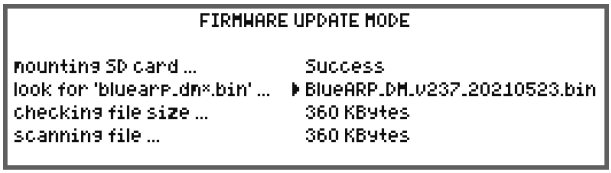
1) Download the latest firmware from https://omg-instruments.com/wp/?page_id=84
2) Place the "bluearp_dm*.bin" file from the package in to the SD Card's root folder
3) Insert SD Card into BlueARP DM
4) Power up the unit, holding down **SHIFT** key. This should load the boot loader screen:

5) If there are several files matching "bluearp_dm*.bin" pattern, select the right one with the encoder. Click **Ok** when done:

The procedure can be safely cancelled by switching off the BlueARP DM at this stage.

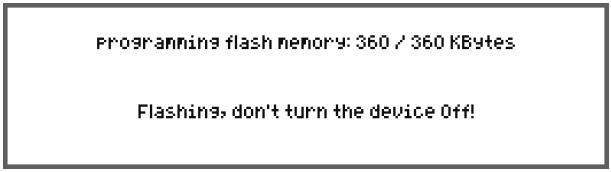


6) If the file is good, press **Ok** to start the actual flashing. "press OK to proceed" message will blink:



This is the last stage where it is safe to quit the update process.

7) Flashing takes about 20 seconds. First it will erase the sectors and then program the flash memory. Progress will look like this:



8) When finished successfully the following will be displayed:

In case of problem during update, try to repeat the procedure. Even if the update failed, the **firmware update mode** screen should still be able to be loaded.

If it didn't help and the device doesn't boot, contact support, contact information here: https://omg-instruments.com/wp/?page_id=92.

Bootloader

Bootloader is a small program that allows firmware updating. Bootloader always starts on power-on and it checks if the “shift” key is pressed. If it does, it passes control to the firmware update routine, otherwise jumps to the main application.

The boot loader itself can be updated from the main application, as it can’t update itself. The procedure is pretty much like the firmware update.

To check boot loader version, enter firmware update mode first (power on the unit holding down **SHIFT** key), it will be shown on the 2nd line.

Press **Cancel** to continue normal boot.

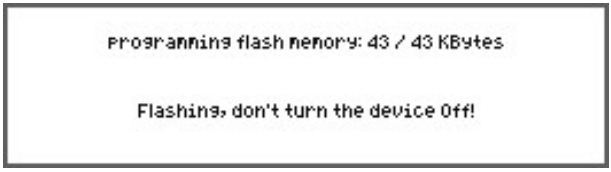
1) Download the latest bootloader from https://omg-instruments.com/wp/?page_id=84
2) Place “bootldr_dm*.bin” file from the package to the root folder of SD Card
3) Insert SD Card into BlueARP DM
4) Power up the unit holding down **UP** key, it enters the bootloader update mode:

5) If there are several files matching the “bootldr_dm*.bin” pattern, select the right one with the encoder. Press **Ok** when done:

6) If *.bin file check gave no errors, press **Ok** to start the actual flashing ("press OK to proceed" message will blink):

This is the last stage where it is safe to quit the update process .

7) Flashing takes about 10 seconds. First it will erase the sectors and then program the memory. Progress will look like this:



8) When finished successfully the following will be displayed:

If the update went wrong and the unit doesn't boot, please check the service manual at https://omg-instruments.com/wp/?page_id=84 for other options or contact support.

Troubleshooting

Can't get the arp working

This is the most common issue. Most likely this is because of routing (either physical cable routing or port\channel settings on **P7. ROUTING** page):

1. Go to **13. MIDI MON** page, press some keys and check if for incoming note on\off events from the keyboard.
   If there is no input logged – something is wrong either with the keyboard configuration or the cable is plugged into the wrong port.
2. Check **P7. ROUTING** page to make sure the right input and output MIDI ports and channels are set up. BlueARP will ignore incoming events on ports and channels other than the configured ones for the instance.
3. Check **P2. ARP** page to make sure **01. Arp Mode** is set to **On**.
4. Check **13. MIDI MON** page again. If output note on\off events are seen, but don't hear any sound, it means that the synth is likely expecting the notes on a different port and\or channel. Double check cable routing and connected synth settings.

SD Card is not recognized by the unit

Symptoms

When the SD Card is inserted, it gives and error message.

Reason

May be a file system issue or card type issue.

Solution

Make sure the SD is FAT- or exFAT-formatted. Also make sure the card type is one of the following: SD, SDHC or SDXC. There may be issues with modern cards of 128Gb size of larger. In this case, try to use 64Gb or smaller card.

Stuck notes when using Arturia Keystep over USB.

Symptoms

Sometimes pressed chords and notes get stuck and they keep sounding after releasing the keys. This happens more often when pressing several keys rather than a single key. This doesn't happen if Keystep is connected to BlueARP DM via DIN cable. This also doesn't happen when connecting the Keystep to a computer via USB.

Reason

Keystep has some bugs in the firmware, which affect embedded hosts like BlueARP DM and other MIDI devices with USB-MIDI connectivity.

Solution

Update the Keystep to the latest firmware. Version 1.1.0.18 from 22-Jul-2019 is known to work.

Can't get any CC, RPN or NRPN messages from Novation Remote.

Symptoms

Novation Remote controller (SLII or any of this line) connected to BlueARP and note on\off messages display on **13. MIDI MON** page, however incoming controller data when adjusting the knobs or sliders is not displayed.

Reason

Probably the Novation controller is in Automap mode. Automap mode is designed to work with a DAW, not with hardware devices like BlueARP DM.

Solution

Switch the Novation controller to advanced mode.