DROID blue-7

DROID Forge 1.7

Release notes

Hell has frozen over. Droid got a display!



These are the release notes for the DROID blue-7 firmware for your DROID master and for the DROID Forge 1.7. Please always update both at the same time.

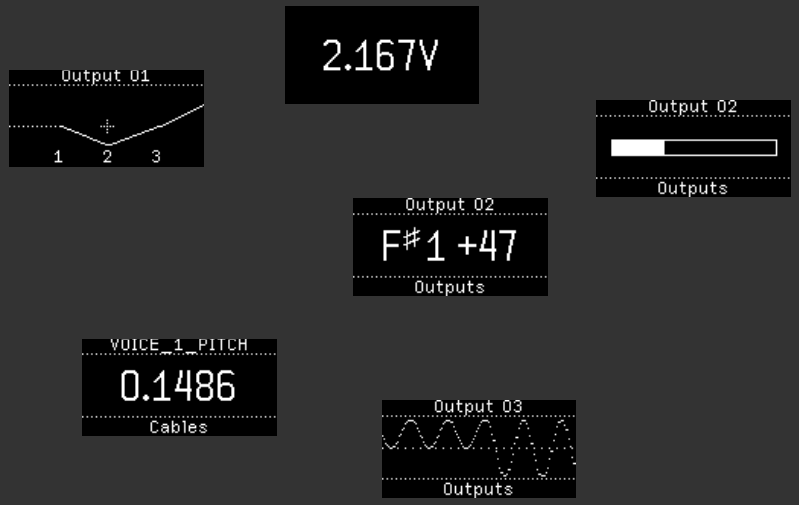
Contents

General improvements....2
New features....3
Bug fixes....5
Update your patches....9
Maintenance udpate for the M4....10
DROID Forge 1.7....11
Motor Fader Performance Sequencer (MFPS)....15
Hints for updating....18

GENERAL IMPROVEMENTS

Hell has frozen over. **Droid got a display!**

The outstanding new feature of blue-7 is the support for the new DB8E display controller. When you add this to your patch, it automatically shows information about what is going on. In addition you can display user specific information from your patch if you like.



Please consult the manual of blue-7 about all the details around the new and shiny DB8E!



NEW FEATURES

- Support for the new DB8E controller
- [**display**]: **New circuit** for sending custom texts and numbers to DB8E display,
  and also for controlling settings like the screensaver.
- [**watch**]: **New circuit** for detecting edges and slopes in an input signal. This
  circuit has four operation modes for various detection tasks. Please consult
  the new manual for details.
- [**clockedtrigger**]: **New circuit** for synchronizing a trigger (probably a button press) with a clock. You press a button and the the press is delayed until the next clock
  tick (or start-of-sequence trigger or whatever). This helps you to build things like
  delaying the switching of a preset exactly at the next cycle of a sequence. This
  feature has been built for @eising in order to spare him all the trouble building this
  with other circuits, like [**notebuttons**], that are not really suitable for this purpose.
- [**switch**]: New inputs “shuffle” and “sort”. A trigger to “shuffle” creates a (new) random mapping between the inputs and outputs. “sort” resets this. A shuffled mapping can still be shifted around with “forward”, “backward” and “offset”. The background: @milgauss suggested a new circuit for the job of doing this random mapping, which is a great idea! But integrating this into the existing switch circuit makes things even more flexible.
- [**nudge**]: New trigger output “changed”, which triggers whenever the value is changed by nudging.
- [**pot**]: new parameter "outputoffset"
- [**multicompare**]: increase number of branches from 8 to 16.
- [**math**]: New output “log2” for computing the binary logarithm (as suggested by @trevormeier).
- [**motorfader**]: new parameters "outputscale" and "outputoffset"
- [**droid**]: New parameters “m4touchgain1” … “m4touchgain8” for adjusting the sensitivity of the touch plates on up to eight individual M4 controllers (needs M4 firmware upgrade).
- [**algoquencer**]: New input “startaccents” for setting how the accents should be preset when the circuit is cleared or starts new (as suggested by @effiksmusic).
- [**motoquencer**], [**encoquencer**]:
  - New parameter “bulkedit”. If you set this to 1 and move a fader or touch a button, all other faders or buttons at the right of this step (later in the sequence) take over the same fader or button value. You want to wire this input to a button. When you hold this button while editing one aspect of a step, this aspect will be immedately be copied to all steps at the right. Do this on step 1 to edit the whole sequence at once. This idea is by @Effiksmusic and I think it's really brilliant. It makes editing your sequence on the fly super quick and efficient. Jam along the gate length of all steps at once – here we go. etc.
  - New parameter “stepcopy” allowing for copy and paste of indiviual steps (suggested by Mylar Melodies).

- New parameter “doublerange” for doubling the current sequence in length and copying the first part to the second part (suggested by Mylar as well).
- New parameter “setendstep” that has the same effect as manually modifying the end step.

- [`quantizer`]: New input parameter “histeresis” for dealing with noise on the input signal. This feature is a special gift to @vpme.
- [`minifonion`]: As suggested by several users there is now an input called “direction”, which selects the direction in which to search for the next allowed semitone. There is “upwards”, “downwards” and “nearest”. Previously the search was always upwards.
- [`midiin`], [`midifileplayer`]: Add “notegatevelocity1” … “notegatevelocity16” to give access to the velocity when using note gates. This feature was desperately requested by @Panda.
- The status dump file now tells you the time it needs for a button press or pot movement on a controller to make it through to the master. There is now a line like this:

  Avg controller cycle: 14.597 ms

  In this example it takes the master 14.6 ms to speek to all of the attached controllers, one by one. So if you press a button it might take up to 14.6 ms to get noticed by your patch running in the master. Likewise it takes the same amount of time to change the light of on LED on a controller. This status change also needs up to 14.6 ms to travel via the 6-pin ribbon cable to the controller. The controller latency is dependend on the number and type of controllers that you have attached. Now you can use the status dump (or the display of the DB8E) to get this information.

BUG FIXES

- MASTER18: Fix MIDI clock, start and stop messages coming in via MIDI USB. At least on the Mac the MIDI implementation sends certain MIDI messages in special USB blocks that were not handled by the MASGTER18 at all. This way the important 1-byte-commands for synchronization didn't make it through to your circuits. No clock could be received via USB MIDI. Neither worked START / STOP. This has been fixed.
- MASTER18: Fix crash at boot time when there is MIDI input on TRS1/2 right during startup.
- When loading patches there was an imcomplete check for input numbers when using non-abbreviated names. For example you could use "compare9" in [**multicompare**] even if there is just compare1 ... compare8 defined. This could lead to strange undefined behaviour. Forge users where not affected by this bug since the Forge checks the numbers correctly before activating a patch.
- **droidpatch**: This Linux helper script for sending patches via USB-MIDI now removes all non-Ascii characters from the patch. This avoids invalid MIDI data.
- [**motoquencer**], [**encoquencer**]:
  - In blue-6, when root or degree was changed, the faders would adapt immediately to the new scale but the pitch CV output would stay the same until the next step is triggered. This could cause a sync problem in some situations, however. If a switch to a new root note or scale happened just slightly after the clock trigger, the current step would still play a note from the previous scale, so the root/scale sync was effectively one clock tick too late.
    This has been fixed. Any change to root or degree is now detected and immediately affects the current step - even if already triggered.
  - In compose mode the output gate length is now forced to exactly 10 ms. This ensures that every note gets a new attack, even if you move the fader fast and lots of notes are played in a short time. Previously sometimes you couldn't hear the notes because the gates merged together.
  - Recording did not work in a linked sequencer (if you didn't patch the clock there, too). This is fixed. Found by @Effiksmusic.
  - When changing the root / scale / note selection, a fader of your existing melody is moved into new positions if the current position does not reflect an allowed note anymore. Previously this was only done *upwards*. Now if there is a nearer allowed position *downwards*, this direction is preferred. In some cases this preserves triads better, especially if your note selection is 1/3/5 and you change the scale from major to minor, you still get a triad.
  - The parameter “defaultcv” did not work after a factory reset. The start CV was also zero in this case. Just when “clear” or “clearall” was used, it worked. This has been fixed. “defaultcv” now always works.
  - The parameter “defaultcv” was slightly off when “quantize” was set to 0. The CV was quantized to the nearest scale note, but should not, because quantization is off. This has been fixed.
- [**encoquencer**]: better guess a good default value for “numfaders” if zorder is 2 or 3.
- [**motoquencer**]: When you move a fader in CV mode while pitch quantization is on - in other words when you move the fader to change a note value - the gate is automatically switched on now. This makes entering melodies in an empty sequencer much faster. There is not much a point in changing the pitch of a step that is not played, is it? If you think otherwise, please tell us.
- All circuits with the input “selectnoteshift” (such as [**motoquencer**], [**minifonion**], etc) had a bug that if you just selected one of the allowed

scale notes, even a small shift amount shifts you a lot and you get easily out of the allowed note range of -24 ... 144 semitones. Here the bug was that if you went beyond this limit, the note was shifted by semitones instead of octaves and you got out of your scale. This has been fixed. And the lower range has been extended to -36 (3 octaves below 0).

- [`logic`]: The abbreviation of threshold has changed from "t" to "th". If you use this parameter you might need to send your patch to your master, again, with the Forge, so that it uses the new correct abbreviation.
- [`encoder`], [`encoderbank`]:
  - Fix an issue where a slight CV ramp (glide, smoothing) happens when the encoder snaps into the virtual notch at the position 0.5. Now in this case smoothing is disabled and you immediately jump to exactly 0.5.
  - The inputs “encoder” and “encoderbank” would not react to changes but just where examined just once at the beginning. This has changed. Now it's possible to map them to dynamic things like an internal patch cable the originates from a buttongroup. This way you can more easily create adaptive user interfaces where the users selects which encoder to use by pressing a button or the like.
- [`algoquencer`]:
  - If you *removed* a “mutebutton” or “unmutebutton” input from your patch while the algoquencer was muted, it stayed muted after loading the new patch. And it would stay so forever since you were unable to unmute it again for lack of the button. This took me half an hour of frustrating wonder myself. So now an algoquencer that does neither have a mute button nor unmute button always starts unmuted and forgets the “mute” state from the last time.
  - Fix the impact of “dejavu” when it is neither 0 nor 1. Let's assume dejavu = 0.5. This means that half of the random decisions are truely random (will have a different outcome every turn of the sequence), and the other half are pseudo random (will have the same once decided outcome all the time). The decision *which steps* get true random decisions and which ones get pseudo random decisions, also needs to be randomized. In blue-6 this was always done with pseudo random decisions. So once and for all it was determined, which steps should be in “dejavu” mode and which in random mode. This could lead to dejavu being not effective in some situations where “activity” disabled the steps with true randomness. Now in blue-7 the algorithm has changed: Now we use *true* random for the decision on which steps dejavu should be applied and do this differently every turn of the sequence. The result is that “dejavu” works more like you would expect it. This also means, however, that algoquencer now behaves slightly differently with some settings. This problem was reported by @NVU.
- [`recorder`]:
  - The sampled mode didn't work. This means that if you patched “sample”, the recording was always empty - the recorded length considered to be 0. @trevormeier found this bug. Now it's fixed and the sample mode works as it should.
  - Fix the input “scrubposition”. It was inverted. Now its correct and 0.0 gives you the start of the recording and 1.0 the end. Found by @trevormeier.
- [`contour`]: The pitch tracking (input “pitch”) now does not affect the attack phase anymore. This never made sense and I consider this as a bug. The pitch tracking simulates the ringing of instruments like a glockenspiel or marimba, where the larger bars (lower notes) ring longer than the shorter ones. But the attack is still the same since it is caused by the impact of the mallet and not by the bar. In the very unlikely case that you still want pitch tracking for the attack phase: You can reconstruct this by some extra circuits that modify the value you send to “attack” input based on the note's pitch.

- [`nudge`]: When a nudge up or down comes within at most 20 ms after a "clear", the button press is ignored now. This is similar to ignoring a clock when it comes within 5 ms after a reset in many circuits in the DROID. The idea is that even if in your patch the "clear" and "nudge" happen simultenously, there is no such thing as "at the same time" in the digital world. That leads to issues where *sometimes* the nudge circuit is immedately increased after a "clear" but should not. I hope this fixes the issue reported by @nando.
- [`vcotuner`]: Fix impact of the "smooth" input and change the default value from 0.01 to 0.5 to give some reasonable mild smoothing.
- [`buttongroup`]:
  - Consider the case that you have a buttongroup with "minactive = 1" and "maxactive = 4". An example would be a selection of scale notes for an up to four voiced chord. The number of active buttons (where the LED lights) is always 1, 2, 3 or 4. Now assume that one button is active. And you hit this button and another button *at the same time*. What you expect is that the first button goes off and the second one goes on. The bug was that the first button stayed on in this special case. This has been fixed. So now you can deselect one button and select another one *in the same instant*. The bug only occurred if the number of active buttons was at the minimum at this simultanous pressing of buttons.
  - Fix preserving the saved state (after power cycle) if "maxactive" is not fixed but dynamic (provided by an internal patch cable). In this case, when a patch is loaded and started, the contents of the cable is still 0 for a very short time. This caused the set buttons to be reset and the saved state to be lost. This has now been fixed. Using cables for maxactive can be useful for creating dynamic user interfaces, for example to temporarily impose a single note selection on a buttongroup that normally selects up to seven scale notes.
  - "maxactive" is not allowed to be 0 anymore. This "un-feature" is required for fixing problems when maxactive is dynamic (a patch cable). I'm sorry for that. You can use [`select`] to switch off unused LEDs instead.
- [`midiin`], [`midithrough`]: Fix a bug in the input port selection. The manual says, that if you specify a specific port or autodetection for either trs, or usb, the autodetection on the other port is disabled. This was not the case. Example:

  ```
  [midiin]
     trs = 1
  ```

  This circuit should just accept MIDI events from TRS, but ignore everthing from USB. Up to blue-6 this was not the case but USB events would also be processed unless you specified usb = 0. This has been fixed. Now no events of the other type are processed if you specify a non-zero value for trs / usb.

  This should also fix unwanted MIDI loops in some cases.

  Note: Your patch might rely on this bug. If your patch stops processing MIDI events after updating to blue-7, please check you midiin and midithrough circuits. You might want to add usb = 10 or trs = 10 to re-enable auto detection for the other non-specified port to bring back the previous behaviour.
- [`midiout`]: The value of the inputs "cc1" … "cc8" is mapped from the range 0.000 … 1.000 to the MIDI CC values 0, 1, 2, … 127. This is done by multiplying the input value by 127, so 0.000 is mapped to a CC value of 0 and 1.000 is mapped to 127. The subtle bug was that hereby after the multiplication the value was simply truncated and the fraction part removed, so if the value would be 50.9 after the multiplication, a CC of 50 was sent. This has now been changed to be rounded to the nearest integer, so 50.9 now converts into a CC value of 51.

- The register **X1** can now be used as an input. Previously this caused a parse error. One can certainly debate the usefulness of using X1 as an input, since it is intended as a diagnostic output. However, given that output registers like O1 can generally be used as inputs as well, it was still a bug.
- Fix minor LED brightness glitch in the encoder LED ring, especially noticable when the LED is set to 1 (LED ring shines white).

UPDATE YOUR PATCHES

If you are coming from blue-6, you need to check two minor things in your patches:

1. If your patch relies on a bug in [**midiin**] / [**midithrough**], that has been fixed in blue-7, you might need to adapt your patch. The bug is described in the BUGS chapter. If your patch stops to process MIDI input events after an update to blue-7, please see the bug's description above.

2. The [**buttongroup**] circuit does not allow “maxactive = 0” anymore. This was described in the manual to switch off LEDs in cases where no circuit is selected that uses them. You can use indivual [select] circuits for that task, instead:

   Previously:

```
[buttongroup]
   select = _SOME_SELECT
   maxactive = 0
   led1 = L1.1
   led2 = L1.2
```

   Now:

```
[select]
  select = _SOME_SELECT
  input = 0
  output = L1.1

[select]
  select = _SOME_SELECT
  input = 0
  output = L1.2
```

If you update from version earlier than blue-6, please consult the release notes of the versions that you have skipped. You might need to tweak a few things in your patches.

MAINTENANCE UDPATE FOR THE M4

This time there are some bug fixes for the M4. If you want to apply these fixes, you need to update your M4s' firmwares to **orange-4** (which you find in the firmware subdirectory of the release ZIP file). Please consult the manual for how to do this.

When you power on an M4 and the first three of the four LEDs below the fader blink orange fast for a second or so, you have the orange-3 firmware. If all four are blinking, you are already on orange-4 and can skip the upgrade.

Note:

- *If you don't experience any of these problems or are not annoyed enough by them anyway, you can savely skip this update. The previous firmware orange-3 is fully compatible.*
- *If you upgrade your Motor Fader Performance Sequencer to have a DB8E instead of a P2B8 and did **not** get your M4s from the new Moto Kit 2, we encourage updating the M4s. This avoids a couple of minor glitches that come to light in the display.*

The following fixes are contained in orange-4:

- When notches = 1, the center value was not *exactly* 0.5, but sometimes just in the range 0.49 ... 0.51. Now when you let the fader go and glide into the middle, it's value is exactly 0.5.
- When notches = 1 the fader sometimes would vibrate for some time when you suddenly released it while it was pushed out of the center. This has been fixed.
- Fix bug where a M4 fader position would alter by a super tiny amount every time you switch the fader to a new postion. This only happened when the fader was in the upper half. The change was about 1/4096 every time the fader switched back to its stored position.
- Make touch buttons a little less sensitive. This avoids spurious "fake" touch detections when faders move. Also support a new feature: There is now a setting in the [**droid**] circuit for adjusting the sensitivity of the touch buttons. Look for the parameter m4touchgain1 ... 8.
- When you switch a fader to a new preset or anything else, and the new fader configuration is notched, any movement of the fader in the new configuration was not effective unless you released the fader. For example in the MFPS: if you switched from "velocity" to "pitch" in a [**motoquencer**] circuit, and then immediately move the fader, no change in the pitch was effective unless you released the fader (after that, when you moved the fader again, it immediately reacted). This bug lead to non-responsive behaviour in the compose mode. This has been fixed.
- When a fader was at the top or bottom and the motor of another fader still running, there *could* be some noise in the reading of the fader position and sometimes it would be regarded as 0.001 instead of exactly 0, for example. Now there is a an improved algorithm that detects this and more reliably outputs exactly 0 or exactly 1 at both ends of the fader.
- If you powered up your M4 but your patch did not use it, the last green LED from the “charging” animation still was lit. This has been fixed (just a cosmetic fix).

DROID FORGE 1.7

Operating system support

- We are very sorry that we have to **drop the support for Mac with Intel CPUs**. This was not an arbitrary choice but due to the fact that Qt, the toolkit that we use and which is the foundation of everything in the Forge, has dropped support for some older Mac framework and my Intel Mac cannot be updated to a new enough Mac OS version that supports the new version of the framework. This is a shame since my Intel Mac still works like a charm, is fast enough and has lots of RAM.

  Since the Forge is open source (GPL) if you have a more recent Intel Mac you might succeed in building a working version from the sources. You find the Forge on github: https://github.com/Zarkuun/droidforge.

New features

- **Massive performance improvement!** The Forge now reacts much quicker to your actions. Most of the sluggishness is gone. It's fun to work with it, again. Finally! I really should've done this ages ago. Sorry for testing your patience ;-)
- New action for finding the next occurrance of a circuit. If your cursor is positioned on the title of a circuit, press '?' or call *Jump to next circuit of same type* from the menu and the cursor jumps to the next circuit of the same type in your patch (if there is any).
- New action for finding the next occurrance of a parameter. If your cursor is positioned on the name of a parameter, for example on "decay", and you press ?, it will move to the next occurrance of a parameter with the name "decay" somewhere in your patch (if there is any).
- When hovering with the mouse over a register that has a label with a description, the description is now shown in a small popup (tooltip), so you can read it without opening the labelling dialog.
- New global option "Sync cloned patch sections". This feature helps you managing more complex patches in situations where you have multiple "instances" of something. Let's assume your patch does the same thing, but for four different synth voices or tracks.

  Create one section for each voice. Make them completely identical except for prefixes in the cable names. For example each cable in the section for voice 1 starts with VOICE1_. Global cables like CLOCK do not have these prefixes, of course. Remove all inputs and ouputs and control numbers that are specific to a voice out of these sections somewhere else.

  You can create cloned sections very easily by duplicating the first section, selecting all circuits and then applying "Rewrite cable names...". If you have setup your patch in this way and enabled the global option we talk about, any change in one of your sections will automatically applied to all its clones, so that they always keep in sync. In the section list, all sections that have a clone are marked with a lightning bolt.
- New status bar element for viewing status dumps. As soon as you load a status dump, a new element in the bottom right of the status bar appears. It has buttons for switching through the various status dumps (@trevormeier suggested keyboard shortcuts, but I came up with this idea instead).
- New menu option and button in the new status dump element for displaying the source contents of a status dump (which shows useful information such as the current cycle time).
- The action "expand parameter array" (CMD E) is now more intelligent when it comes to choosing a new cable name for the added parameter. It now detects numbers within a cable name and computes the successor of STEP1_ROOT to be STEP2_ROOT.

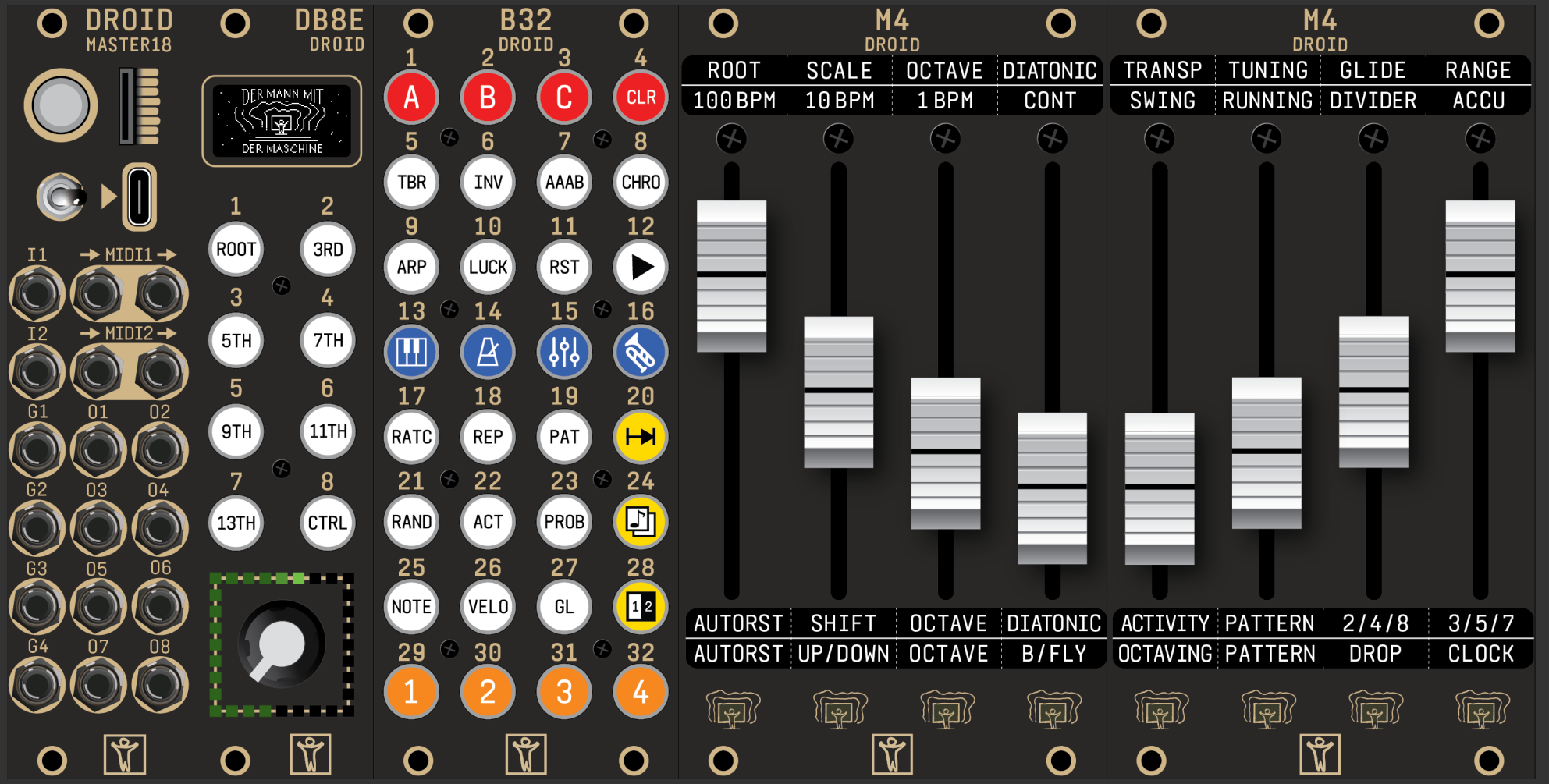
- New item in the help menu: "*Show all hints again*". This will reenable all hints that you have disabled by unchecking "*Show this hint again next time*".
- Convenience feature: The Forge now remembers the directory from which you have opened a patch most recently and offers the same directory when opening a patch next time. This way it's easier to use your own directory for patches without always having to navigate there when you open the Forge.
- New icon for ejecting the Droid SD card. This is useful if you want to remove the SD card without deploying the patch to it. You can do with the file browser of your Windows or Mac, but the new icon is much more convenient.
- New quick way for showing the user manual of a module: right click on a module in the rack view and you will find the new entry "Show user manual of ...". This brings you directly to the chapter about this module (master, X7, G8, any controller).
- You can now click on the RAM usage / patch size information in the status bar and open the detailed RAM analysis (same as "Analyse patch RAM usage" in the File menu).

Bug fixes

- When saving a patch the Forge now checks if the contents of the target file have changed since the patch has been loaded from that file. Maybe a second copy of the Forge running that edits the same file. Or some cloud service pushes an edited or old version from a different PC there or whatever. So if the Forge detects a spurios change of the original file, it issues a warning so you can abort the saving and choose another file name or check what's going on.

  This change addresses a bug report of @trevormeier where he reported that his patch has not been saved correctly over a longer period and he lost hours of work. Before the Forge marks your patch as "unmodified" it double checks the success of the saving to disk by reopening the file after saving and comparing its contents byte-by-byte to the contents it should have. Only if this exactly matches, the saving is reported to be OK. So my only guess is that the file was saved correctly but always overwritten with the previous content by some other process.
- Fix crash that happened when you pasted one or more parameter lines from one circuit into another and in the other circuit lines with those parameters already existed and the cursor was at the end of the circuit.
- Small cleanup in the "instant cable" feature when holding Alt while clicking on a different cell: This now only works if both the cursor and the place where you click are places to hold a value. Previously it also worked when the cursor was in the first column on the name of the parameter. Clicking would then produce only "half" a cable.
- The keyboard shortcut for disabling / enabling a circuit or parameter line has been changed from **#** to Shift 3. With an English keyboard layout this is the same key. For German and maybe other layouts this changes. This is the only way I found to fix the problem that in German layout in the menu the shortcut was displayed as '3' instead of '#'. I'm sorry for the inconvenience imposed on non-US layout users.
- Raise the window of the user manual to the front if it's still there but hidden in the back und you select a menu entry for showing some manual chapter (or press CMD-T).
- Fix possible crash when holding the command key and clicking somwhere in the patch, when the cursor is *not* on a suitable position for value entry (i.e. when instant copy does not make sense).
- Fix size and placement of labels on P8S8 controllers (as suggested by @nando).
- Fix "Memory Analysis": Disabled circuits were not excluded, but are now (found by @AgentTerror).

- Prevent the cursor from jumping to the end when typing in the mid of a cable name in the atom editor dialog.
- Don't show problems about a cable conntected to more than one output at the *inputs* that are connected to this cable (which is just confusing and not helpful).
- When you entered the parameter selector (for adding a parameter or changing the name of one) and in the search there was still something typed from the last time and no parameter was found (because you were now in a different circuit), the description of the previously selected parameter was still lingering on, even if it was from a different type of circuit. This has been fixed.
- When you paste one or more parameter definitions into an existing circuit, possibly existing definitions for these parameters are automatically removed now so that you don't end up with duplicate definitions. They are not removed if they are disabled (greyed out).
- Allow longer labels for controls, so that the existing space in the white rectangle can fully be used.
- Fixed a bug in the computation of free memory. Some times the actual free value could differ by 72 bytes from what the Forge showed.
- Fix the faceplate image of the P8S8. It's sliders have amber LEDs, not red and blue ones as were depicted previously.
- Dialogs used to remember the position where you moved them last time. This cause trouble in some cases so I've removed that feature. Dialogs are now positioned by Windows / Mac in the standard way (in the middle of the screen). The size is still remembered.
- When you did “New patch with same rack” and the type of master was MASTER18, it got changed to MASTER in the new rack. This has fixed and now it stays MASTER18 as it should.
- The function “Integrate other patch” did not correctly copy the labels of the controllers from the integrated patch. This has been fixed. Found by @Effiksmusic.


DROID
MASTER18
I1
MIDI1
I2
MIDI2
G1 O1 O2
G2 O3 O4
G3 O5 O6
G4 O7 O8
DB8E
DROID
DER MANN MIT
DER MASCHINE
1 ROOT
2 3RD
3 5TH
4 7TH
5 9TH
6 11TH
7 13TH
8 CTRL
B32
DROID
1 A
2 B
3 C
4 CLR
5 TBR
6 INV
7 AAAB
8 CHRO
9 ARP
10 LUCK
11 RST
12
13
14
15
16
17 RATC
18 REP
19 PAT
20
21 RAND
22 ACT
23 PROB
24
25 NOTE
26 VELO
27 GL
28
29 1
30 2
31 3
32 4
M4
DROID
ROOT
SCALE
OCTAVE
DIATONIC
100 BPM
10 BPM
1 BPM
CONT
AUTORST
SHIFT
OCTAVE
DIATONIC
AUTORST
UP/DOWN
OCTAVE
B/FLY
M4
DROID
TRANSP
TUNING
GLIDE
RANGE
SWING
RUNNING
DIVIDER
ACCU
ACTIVITY
PATTERN
2/4/8
3/5/7
OCTAVING
PATTERN
DROP
CLOCK

MOTOR FADER PERFORMANCE SEQUENCER (MFPS)

The Motor Fader Performance Sequencer is a ready to use patch that implements are versatile melody sequencer optimized for interactive live performances.
You find the newest version of this patch in the folder **motokit** of the firmware release ZIP file. There are two versions of the patch:

motokit.ini: Uses a P2B8 as first controller
motokit2.ini: Uses the new DB8E as the first controller.

For details consult the file **README.txt** that is contained in that folder.

You can also use the Forge and create the patch yourself with the the patch generator “Motor Fader Performance Sequencer”. Don't forget to update the Firmware of your MASTER18, as well.

New Features

- Support for the new DB8E display controller. You can use it instead of (or in addition to the P2B8). The display always shows you what's going on when you do something. It shows you which scale and root note you selected, which clock speed you set, the note value of a step you've just changed etc. Instead of the two pots of the P2B8 you have one encoder. This is used to take over the pot's functions (repeat/retachet ARP, arpeggio base and range). Hold CTRL while turning the encoder to select the second funtion.
- You can now have 32 steps even with the standard Moto Kit configuration of two M4s. Previously the problem was, that you would need four buttons for switching to the pages 1, 2, 3 and 4 of eight steps each. So without disabling three of the existing features the patch generator would run into an error.
  As requested by several MFPS users now just one PAGE button is needed. Press it multiple times to cycle through all four pages. This is not as immediate as having four individual buttons, but now extending the standard Moto Kit sequencer from 16 to 32 steps is as easy as setting *Maximum number of sequencer steps* to 32.
- Moving a fader in note mode automatically enables the gate.
- When you hold the CTRL button while editing a step (moving a fader or touching a step button), *all other steps after the edited one* will immediately take over the same value for the edited aspect. When you hold CTRL and change a pitch, voila – all later steps get the same pitch. Or hold CTRL while turn on the gate of a step: all later gates will be turned on, as well. This makes changing your sequence as a whole much more efficient and fast – and thus also “jammable” in real time. Want to jam along with the gate length and improvise long/short patterns? Just hold CTRL and jam with the first of the step faders (while in GL mode). This ingenious idea is from @effiksmusic!
  Note: The feature for keeping the number of steps constant when moving a fader in the REP mode while holding CTRL is now accessable by holding the REP button while moving a fader.
- When you hold the CTRL button while switching on ARP (the arpeggiator), it now starts in “Auto stop” mode. In this mode, the ARP LED blinks and the arpeggiator is running exactly until the sequence of the track reaches step 1 (The sequence is proceeding in the background while the ARP is running, but not audible, of course). The musical idea behind this: Using the ARP for creating fills gets easier. The fill automatically ends at the start of the sequence. This “fire and forget” approach is easier than if you have to switch off the ARP in time youself.
- There are new different ways how the copy button operates. Per default is now does copy & paste of single steps: hold the copy button, touch the step to copy, touch one or more steps to paste into, release the copy button button. Pressing the copy button without touching a gate doubles the active range e.g. from 8 to 16 steps and copies all steps from the old

part to the new second half of the sequence. See the manual for more details.

- New option "just extra gate (using fader)" for the velocity. This uses the velocity fader to just set an extra gate on or off (two fader positions). This is useful in situations where the receiving module just accepts an on/off velocity or accent.
- You now can set a minimum and maximum value for the velocity output (both effective for MIDI and CV output). Especially for CV output this is handy. You can set a nice and musically useful range for the faders that matches the ideal modulation range of your target synth voicde without the need of and external attenuator or CV mixer.
- We have cleaned up the MIDI input configuration. It is now in the tab “MIDI”. You can select a MIDI input source for clock/start/stop, either by selecting one specific port or allowing all ports. The default is to listen on all ports for a clock. The tab “Connectivity” does not contain any MIDI options anymore. There is just CV / gate integration.
- New option “Send start/stop/clock to MIDI” in MIDI section for sending a MIDI clock and running state to one or all MIDI output ports.
- New option “VCO tuner on input I1” in the tab “Features”. This adds a VCO tuner to input I1. You need a MASTER18 and DB8E for this to work. Patch a basic waveform output such as square, triangle, sawtooth or sine into I1 and you immediately get a tuner display that shows you the exact current pitch and note of the VCO.
- New option to receive root and scale from MIDI. This is done in the “MIDI” tab and you turn it on by selecting the MIDI port to listen on. The root note is set via a note event on channel 16. The scale is set with CC#1 on channel 16. The CC values 0 … 107 reflect the various scales that the degree input of several circuits expect (like [**minifonion**]). For example a value of 0 means lydian and 10 is a diminished scale. Note: If you use a [**midiout**] circuit of another Droid to send this CC, you need to convert the scale number into a value in the range 0.0 … 1.0. Here is an example:

```
[midiout]
    pitch1      = _ROOT / 120
    gate1       = _CLOCK
    ccnumber1   = 1
    cc1         = _SCALE / 127 + 0.003937
    updateccs   = _CLOCK
```

This example forces an update of both the scale and the root note once every clock cycle. Otherwise [**midiout**] would just send updates on a change and this might lead to missing updates when you plug in your cable after the Droid is already running or if you restart your MFPS.

- synchronize the effect of global and per-track octave switches with the clock so that nasty ugly glitches during a clock cycle cannot happen anymore. Yes, if you set odd clock divisions like x2 and /3 at the same time, the switch could happen during a note. But if we would synchronize to the actual gates, the switch would stop working when no gates happen (e.g. during very long notes) and that could confuse the musician.

Bug fixes

- Fix external USB MIDI clock and start / stop when using MASTER18 (this is a fix of the MASTER18 firmware but the bug was reported mainly by MFPS users, so I repeat it here). It simply didn't work.
- The internal clock is now synced with the START button. When you hit it, the internal clock is reset and immediately restarts. In the version from blue-6 the clock was running freely and after hitting START you needed to “wait” until the next clock tick. You can feel the change best when the clock is very slow. Also it fixes a problem where the MIDI output clock is out of phase with the sequencer. This issue was found and the fix suggested by @born a kid.

- Fix timing issue when using START from a MIDI input port. Sometimes the first note of the sequence would not be played but instead the note where the sequencer stopped last was repeated.
- Compose mode: force a short gate time of 10 ms so that you always get a new note attack even if you move the fader fast.
- Fixed left-over colored LED in unused fader 7 in clock menu. It's now black as it should be. Previously, when you switched into the clock menu from somewhere else (and have an unused fader 7 because you didn't activate an external utility clock with division), the LED below fader 7 would still have the color it had before. This was confusing.
- Sync the effect of the ARP button to the clock of the track. This fixes nasty audible issues if you hit the ARP button while a note is being played: the pitch would immediately change between the ARP/NON-ARP pitches inbetween clock cycles. Now the effect of switching the arpeggiator on/off is delayed until the next track input clock (which is identical with the master clock unless you changed the clock divider for the track).
- In compose mode the gate is now output even if the sequencer is stopped.
- The button TBR (transpose by root) now also works in the arpeggiator. It didn't and I consider this as a bug.

HINTS FOR UPDATING

In the firmware ZIP file there is a subdirectory called **manual** with the current user manual for Droid. Please carefully read the chapter about how to update your firmware. You also find the manual included in the Droid Forge. In addition here are some hints:

- There is **no** new firmware for the X7 in this release. It is the same as in blue-3.
- There is **no** new firmware for the E4 in this release. It is the same as in blue-4.
- **There is a new firmware for the M4 in this release. It fixes minor issues and is useful but not mandatory.**

Hint: The Forge has an easy firmware upgrade feature. You can use this, if your master has at least blue-2. You find it in the menu “File / Upgrade firmware of master”.

Update your Motor Fader Performance Sequencer (MFPS): If you are using the Moto Kit with the standard patch and no Droid Forge, you find and updated patch in the directory **motokit** of the ZIP file. Check out README.txt in this directory.

Hints for users of very old firmwares (until 2022) of the Droid MASTER:

- If you are still on the old **green-**8 firmware, you must first update to **blue-**1, then to **blue-2**.
- If you are one **blue-1**, you must first upgrade to **blue-2**.
- You may skip blue-3, 4, 5 and 6 and update directly from blue-2 to blue-7, if you like. Please read the release notes of the skipped releases. There are some things you might need to adapt in your patches.
- **If you start the firmware upgrade and all LEDs of the master flash magenta, your firmware file is not accepted. In this case:**
  - **Put the firmware file (“droid.fw”) from your current firmware onto the SD card (that from blue-2)**
  - **Do an update of the bootloader as described in the user manual.**
  - **After that continue with upgrading.**

Patch does not load or stop working?

Important note: If your patch does not load or run anymore after the update, chances are that you need to redeploy the patch with the new version of the Forge. Either to a *Save to SD* or *Activate!* to your updated master. This is neccessary if you have enabled *Detect and share duplicate input values for inputs* in your Forge. In this case the **droid.ini** file that is on your SD card does not work with the new firmware and needs to be regenerated.