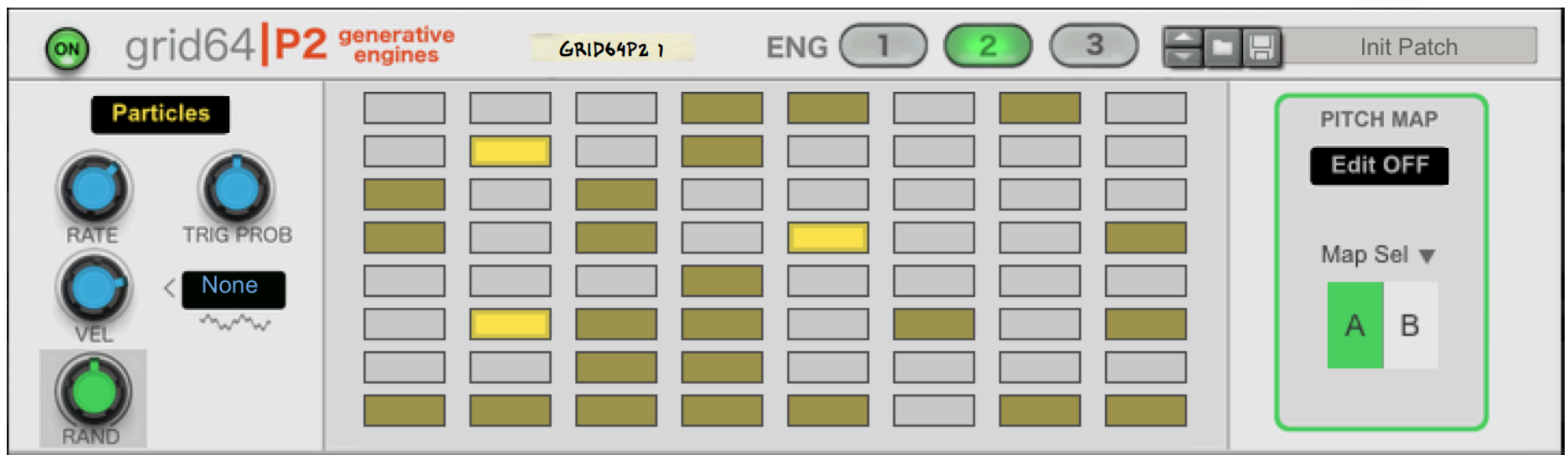


Grid64 Player

Model P2: Generative Engines

Rack Extension for Reason



USER MANUAL
version 1.0.4

Table of Contents

1. Introduction	4
1.1 Product Details	5
2. Overview	6
3. Usage	7
3.1 Engines	8
3.1.1 Engine 1	8
3.1.2 Engine 2	10
3.1.3 Engine 3	12
3.1.4 Programming Blocks	14
3.1.5 Capturing a generative performance	17
3.1.6 Input Mode Selection	18
3.2 Creating a note layout	19
3.2.1 Editing of Single Pads	19
3.2.2 Row and Column Edit menus	20
3.2.3 Map Edit menu	24
3.3 Setting up your grid controller in Reason (standalone)	26
3.3.1 Install the Remote files	26
3.3.2 Creating Control Surfaces in Reason (standalone)	27
3.3.3 Using the Control Surfaces in Reason (standalone)	29
3.3.4 "What if I don't have a grid controller?"	30
4. MIDI Implementation	31

5. Remote Implementation	32
6. Version History	37

1. Introduction

The grid64 Player series of rack extensions takes inspiration from the tools developed over the years by the user community of grid MIDI controllers, and wants to bring some of those workflows to the Reason environment.

The focus of the model P2 player is on generative music. There are three different engines which take inspiration from physical systems to generate notes. The user can interact with these systems to control various aspects of the generative process:

- Engine 1 mimics the motion of particles moving in a maze of blocks. As the particles randomly make their way thru the maze, notes are being generated based on the grid cells that they have visited. The user can control the starting position of the particles, how many there are, and the geometry of the maze
- Engine 2 mimics the motion of particles bouncing off the walls of container. This time, notes are generated when a particles visits a cell where a block exists. Like for Engine 1, the user can control the starting position of the particles, how many there are, and the distribution of the blocks over the grid
- Engine 3 mimics the evolution of a 1 dimensional cellular automata system. For each generation, some cells live on while other dies. Cells which are alive trigger notes if they happen to have a block at their location. The user can distribute the blocks on the grid and select the "rule" which determines the evolution of the system over time.

The grid itself is fully programmable. Notes can be assigned freely to each cell and saved into one of two map locations. Several edit functions make it easy to customize the maps in different ways.

The device can be used with an 8x8 grid MIDI controller. The models supported are the Novation Launchpads, NI Maschine Jam, Ableton Push and the Akai APC Mini. For those, you can download custom Remote files which enable two-way communication with Reason*.

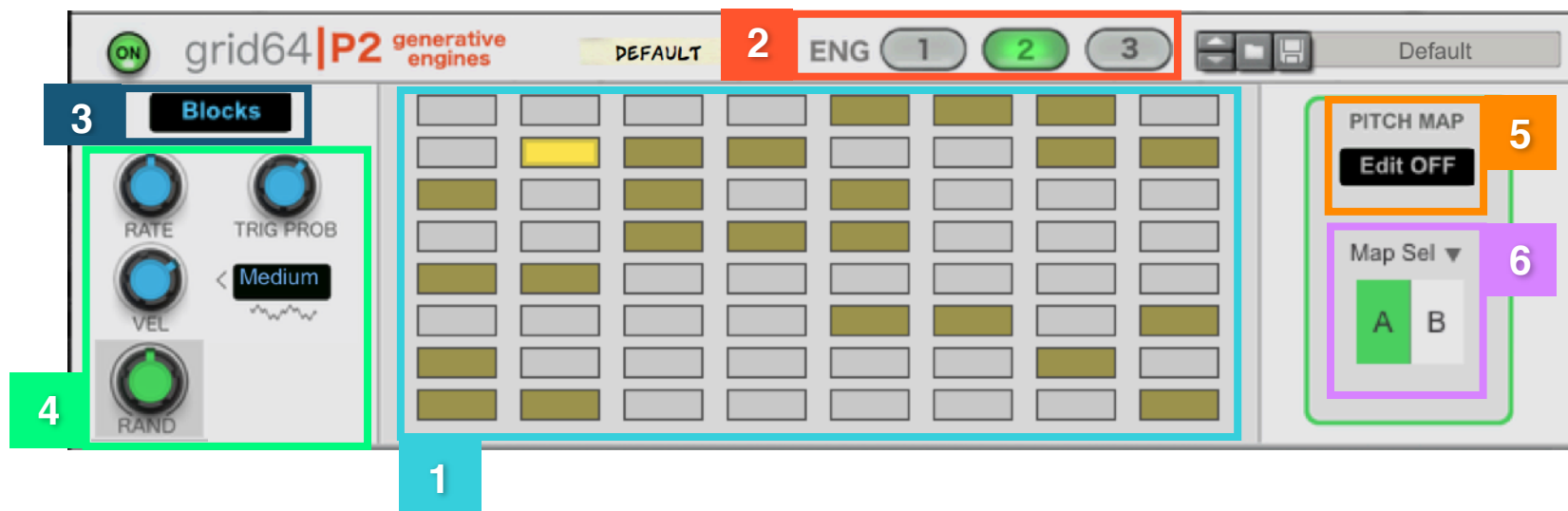
* Reason standalone mode only, Reason Rack plugin does not support Remote.

1.1 Product Details

- Interface for 8x8 grid MIDI controllers featuring two-way communication with Reason (in standalone mode, after setting up the control surfaces for the supported models, Remote files are provided)
- Notes are generated once the sequencer is running. Note generation depends on the engine type selected
- Three engine types which generate notes by emulating physical processes. Each engine allows for "live" input from the user
- Engine 1 emulates the random motion of particles inside a maze. The user has control over the initial position of the particles (up to 3), the geometry of the maze, the rate at which the particles move, how probable it is for a particle to trigger a note, and the velocity of the notes.
- Engine 2 emulates the motion of particles moving inside a four sided container. The user has control over the initial position of the particles (up to 3), the distribution of the blocks, the rate at which the particles move, how probable it is for a particle to trigger a note, the velocity of the notes, and the probability of a particle changing its direction after hitting a wall
- Engine 3 emulates a 1 dimensional cellular automata system. The user has control over the number of live seed cells at the start, the rate at which the cells evolve, how probable it is for a live cell to trigger a note, how spread out are the cells, and the rule which governs the cell evolution over time, and the distribution of the blocks
- The pitch layout is fully programmable, a specific pitch can be assigned to each grid cell
- Row or column edit menus with the ability to copy/paste, transpose, rotate, shuffle and randomize all the pitches in a given row or column
- Map edit menu with the ability to copy/paste, transpose, shuffle, randomize and assign presets to all the pitches in the selected map
- There are two configurable map variations A and B per patch which can be switched during play

2. Overview

Here is a quick overview of the main interface elements. For more details on each section, refer to later parts of this manual.



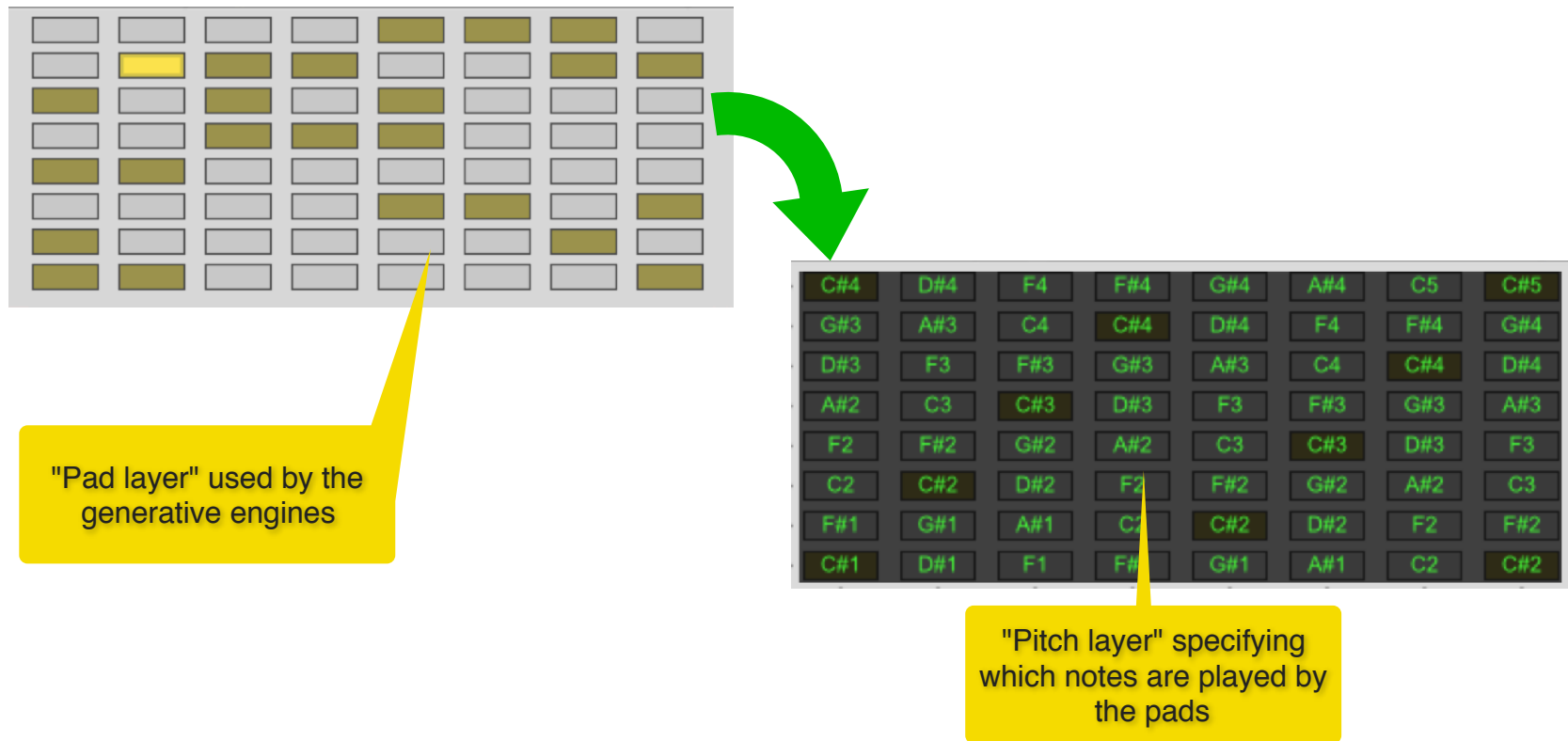
1. Grid layout which provides the interface for note generation. When "Pitch Map" edit mode is "ON", the interface allows you to assign a specific note to each grid cell
2. Engine select buttons
3. Input mode selection: if "Blocks" mode is selected, clicking on a grid cell creates a block at that position. If a block already exists there, it removes it. If Engine 1 or 2 is engaged and "Particles" mode is selected, clicking on a grid cell creates a particle at that position. If Engine 3 is engaged and "Rule" mode is selected, clicking on a grid cell selects one of the randomly assigned "rules" for the evolution of the cellular automata. If "Off" is selected, the grid does not register input from mouse clicks or MIDI messages
4. Controls for the 3 engines, including rate, trigger probability, note velocity and velocity variation. Engine 2 and Engine 3 have additional controls which appear when engaged
5. "Pitch Edit" mode selection: when "ON", click on a cell to assign a specific pitch from a pop up menu
6. A and B map variations which can be freely switched to access more notes. Several editing functions make it possible to quickly duplicate maps and alter maps in various ways for more advanced uses

3. Usage

grid64IP2 is a player device and hence it needs to be instantiated on top of an instrument. This can be a synth, a sampler, a drum machine or anything which receives notes and is able to make noise!

Before going into the details of programming the device, let's get some basics out of the way. The pads grid layout is basically an interface layer sitting on top of the underlying "pitch map" layer. A generative engine triggers whichever note is programmed in the underlying "pitch map" layer. This "pitch map" layer is fully programmable and it can be edited and altered in many different ways. Furthermore, you can switch between two different "pitch maps" while playing the device by using the Map select buttons.

For more information on how to edit the pitch map, please go to section 3.2 of this guide.



3.1 Engines

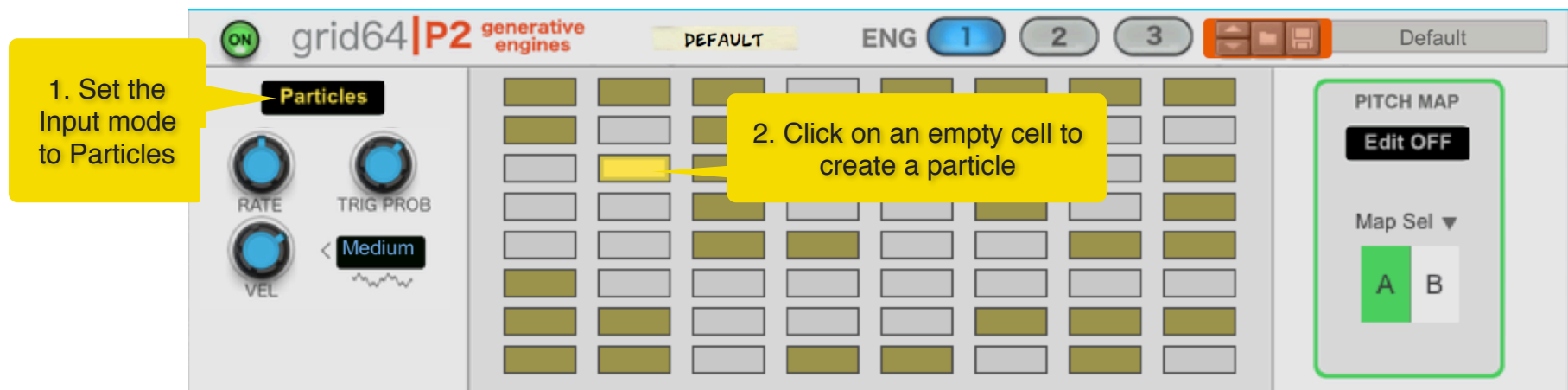
There are three generative engines which use different methods of playing notes. These are explained in the following sections.

3.1.1 Engine 1

Engine 1 emulates the random motion of particles inside a maze of blocks. As the particles move over the grid, they can only visit cells which are free from blocks (for more information on blocks, see section 3.1.4). By doing so, they trigger the cell's underlying note as assigned in the pitch map.

To activate the Engine, start the Reason sequencer. Then click on the Input mode display to enter "Particles" mode. At this point click on any empty cell to create a particle which starts moving immediately. Up to 3 particles can live at the same time, moving about the grid and triggering notes. Clicking directly on an existing particle removes it from the grid.

The speed at which particles move is determined by the "Rate" parameter. The "Trigger Probability" knob affects the likelihood that a particle triggers a note as it moves into a free cell. The velocity knob and the Velocity Variation settings affect the velocity of the triggered notes.



Engine 1 Parameters



Sets the speed at which particles are moving about the maze. This is indicated in musical intervals, from 1/64 to 1 bar notes



Sets the probability that a note is triggered when a particle visits a free cell, ranges from 0% to 100%



Sets the velocity at which notes are triggered

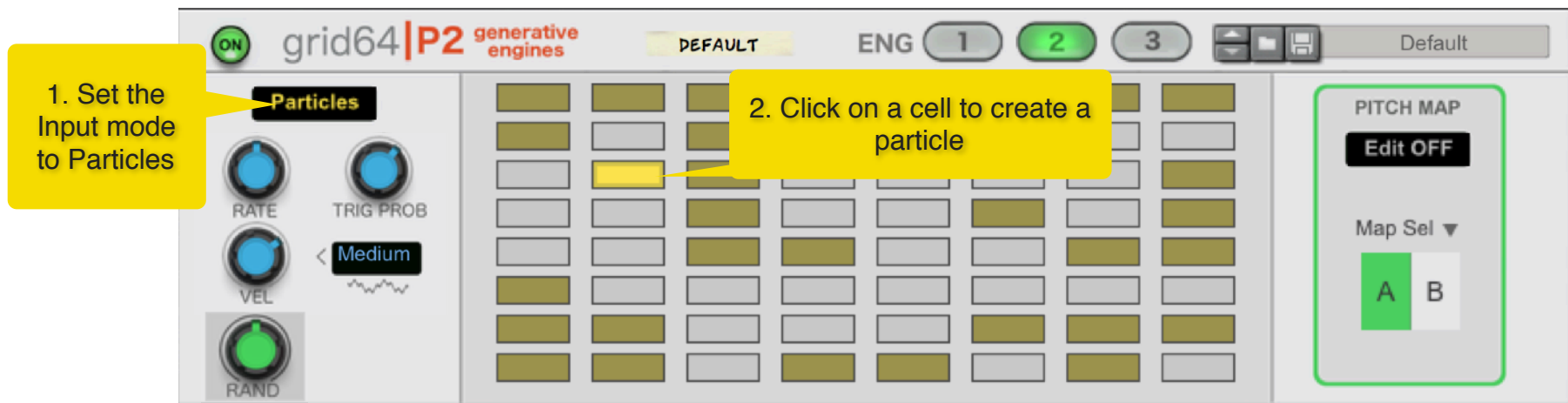


Sets the amount of random velocity variation from the value set by the Velocity knob. There are four levels: "None", "Light", "Medium" and "Heavy"

3.1.2 Engine 2

Engine 2 emulates the motion of particles inside a four wall container. The particles move around and when they hit a wall, bounce off. As they move over the grid, they trigger a note whenever they move over a cell which contains a block (for more information on blocks, see section 3.1.4).

To activate the Engine, start the Reason sequencer. Then click on the Input mode display to enter "Particles" mode. At this point click on any cell to create a particle which starts moving immediately with a randomly assigned direction. Up to 3 particles can live at the same time, moving about the grid and triggering notes. Clicking directly on an existing particle removes it from the grid.



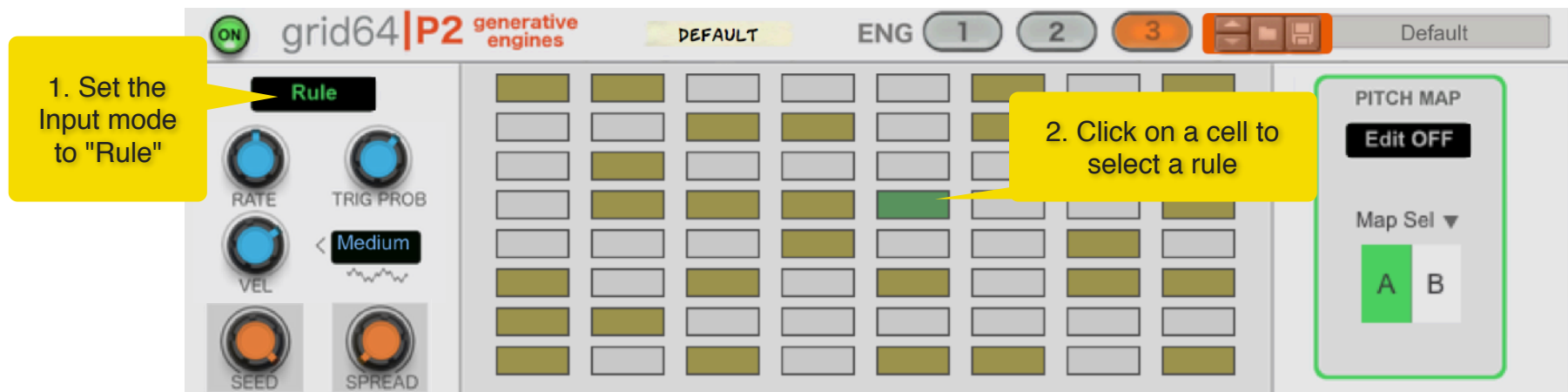
Engine 2 Parameters

	Sets the speed at which particles are moving about the maze. This is indicated in musical intervals, from 1/64 to 1 bar notes
	Sets the probability that a note is triggered when a particle visits a free cell, ranges from 0% to 100%
	Sets the velocity at which notes are triggered
	Sets the amount of random velocity variation from the value set by the Velocity knob. There are four levels: "None", "Light", "Medium" and "Heavy"
	Sets the chance that after hitting a wall the particle changes direction of travel

3.1.3 Engine 3

Engine 3 emulates the evolution of a one dimensional cellular automata system. The cells evolve over time according to a set of rules based on the states of neighboring cells. A cell has two such states, "on" or "off". If a cell is in the "on" state and it visits a grid position which contains a block, then it triggers a note (for more information on blocks, see section 3.1.4).

To activate the Engine, start the Reason sequencer. The first generation of cells is created automatically. The initial number of cells with an "on" state is dictated by the value of the "Seed" parameter. From there on, the system evolves according to the currently selected rule. To access the "Rules" selection interface, click on the Input mode display until it shows "Rule". The currently selected rule is shown in green color. To select a different rule, click on any other grid cell. *Please be aware, each time you create an instance of the player, a different set of rules is randomly selected for the 64 cells.* Basically, any time you reopen a patch or a song, a new set of rules is randomly assigned to the player. Therefore, if you want to save the player's output it's important to capture the "performance" by using the "Direct Record" method as explained in section 3.1.6.

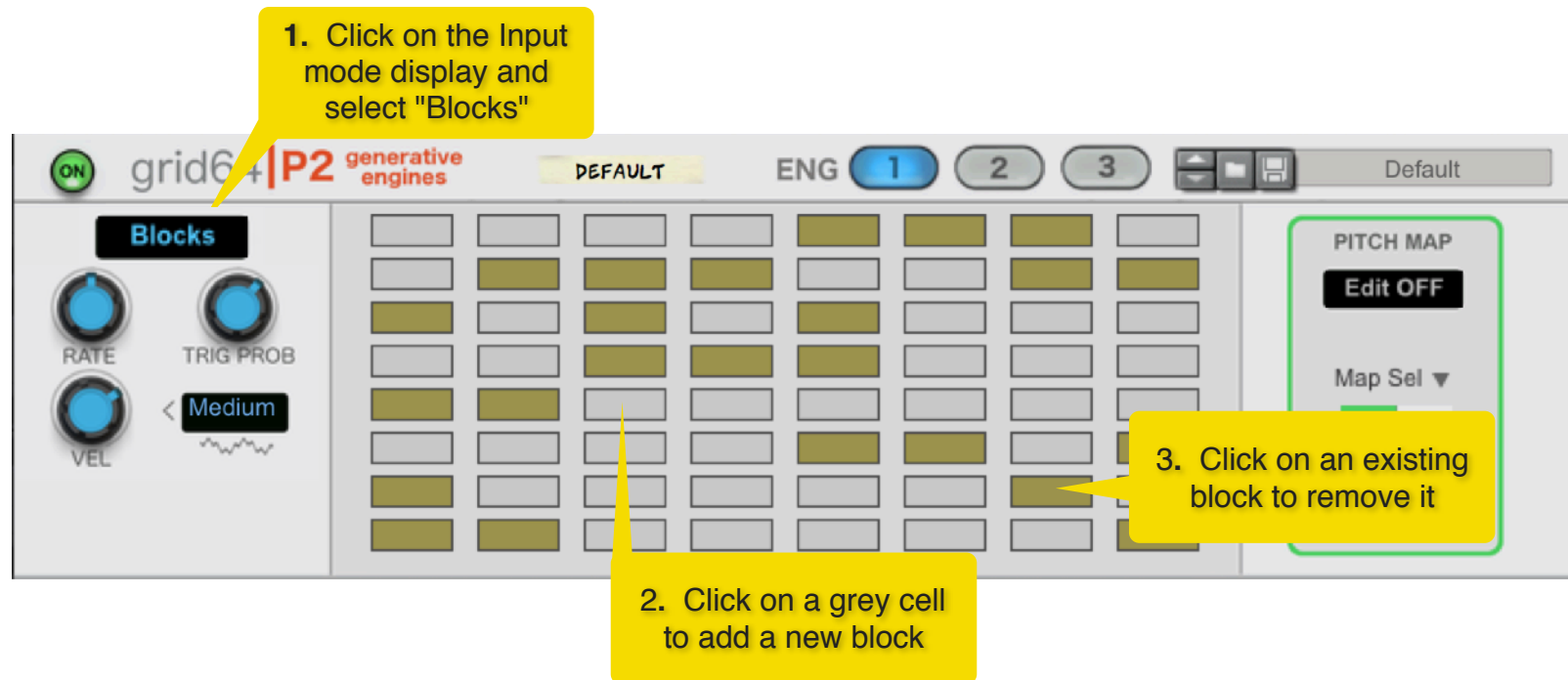


Engine 3 Parameters

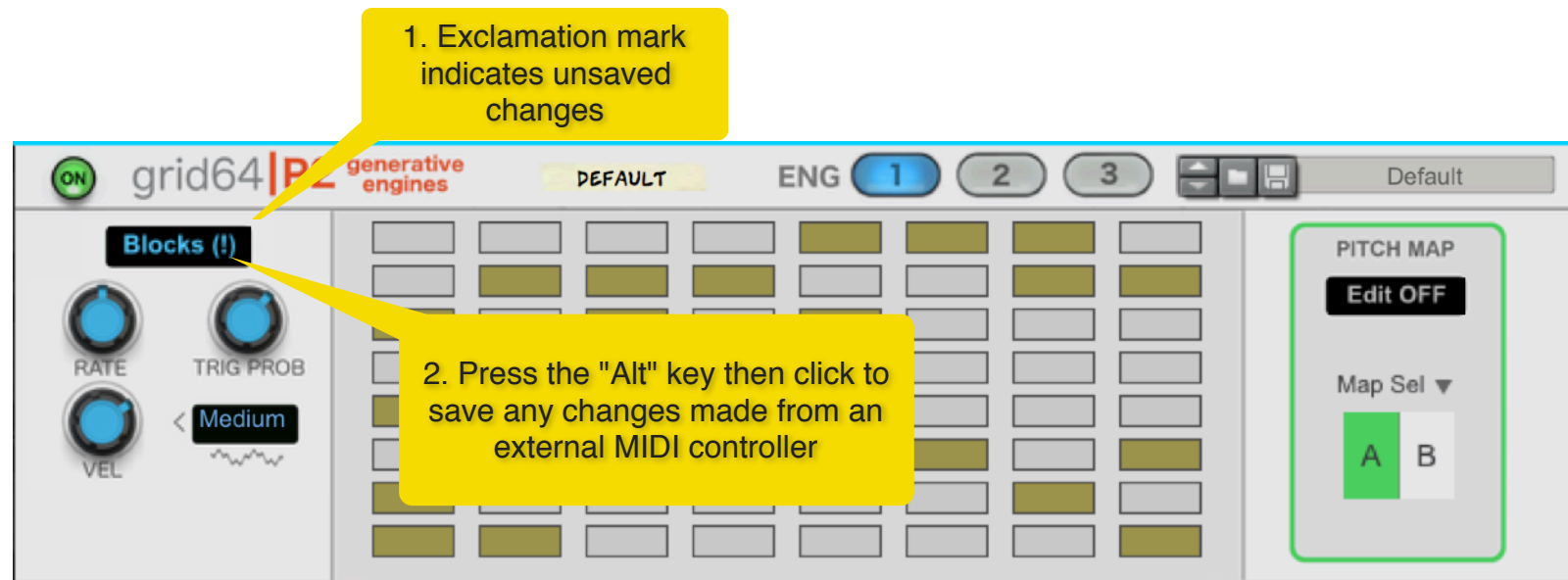
	Sets the speed at which particles are moving about the maze. This is indicated in musical intervals, from 1/64 to 1 bar notes
	Sets the probability that a note is triggered when a particle visits a free cell, ranges from 0% to 100%
	Sets the velocity at which notes are triggered
	Sets the amount of random velocity variation from the value set by the Velocity knob. There are four levels: "None", "Light", "Medium" and "Heavy"
	Sets the number of cells in the "on" state when the sequencer starts to play. There are 9 options, the last one is "Random" which means that the number of "on" cells is selected randomly every time the device starts to play.
	Sets the amount of spread for the cells. If Spread is set to zero, then all cells evolve over time in the same row. As Spread is increased, cells from the same generation evolve in different rows.

3.1.4 Programming Blocks

To create blocks, select "Blocks" as the Input Mode. Then click on an empty cell to create a new block or click on an existing block to remove it. Block cells are displayed in a dark yellow color. You can freely add or remove blocks while the device is running, thus allowing a live interaction with the Engines.



If you have configured an 8x8 grid MIDI controller, then you can program blocks directly by pressing its pads. Please note that once you program blocks via a MIDI controller, the changes are not automatically saved in the device. This is indicated by an exclamation mark "!" in the Input mode display as shown below. In order to save the changes you made, you can either just click on the Input mode display and change to another mode, or alternatively you can click on the display while keeping pressed the "Alt" key on the computer keyboard. When the changes have been correctly saved, the exclamation mark goes away.



There is a handy edit menu for modifying blocks, as shown below. You can access it by pressing "Cmd/Ctrl" on the keyboard and clicking in the Input mode display.



Shuffle: moves the existing blocks around to create a new configuration

Invert: converts the existing empty cells into blocks and the existing blocks into empty cells

Randomize: creates a random configuration of blocks. There are options for randomizing at 25%, 50%, 75% and 100% (fill)

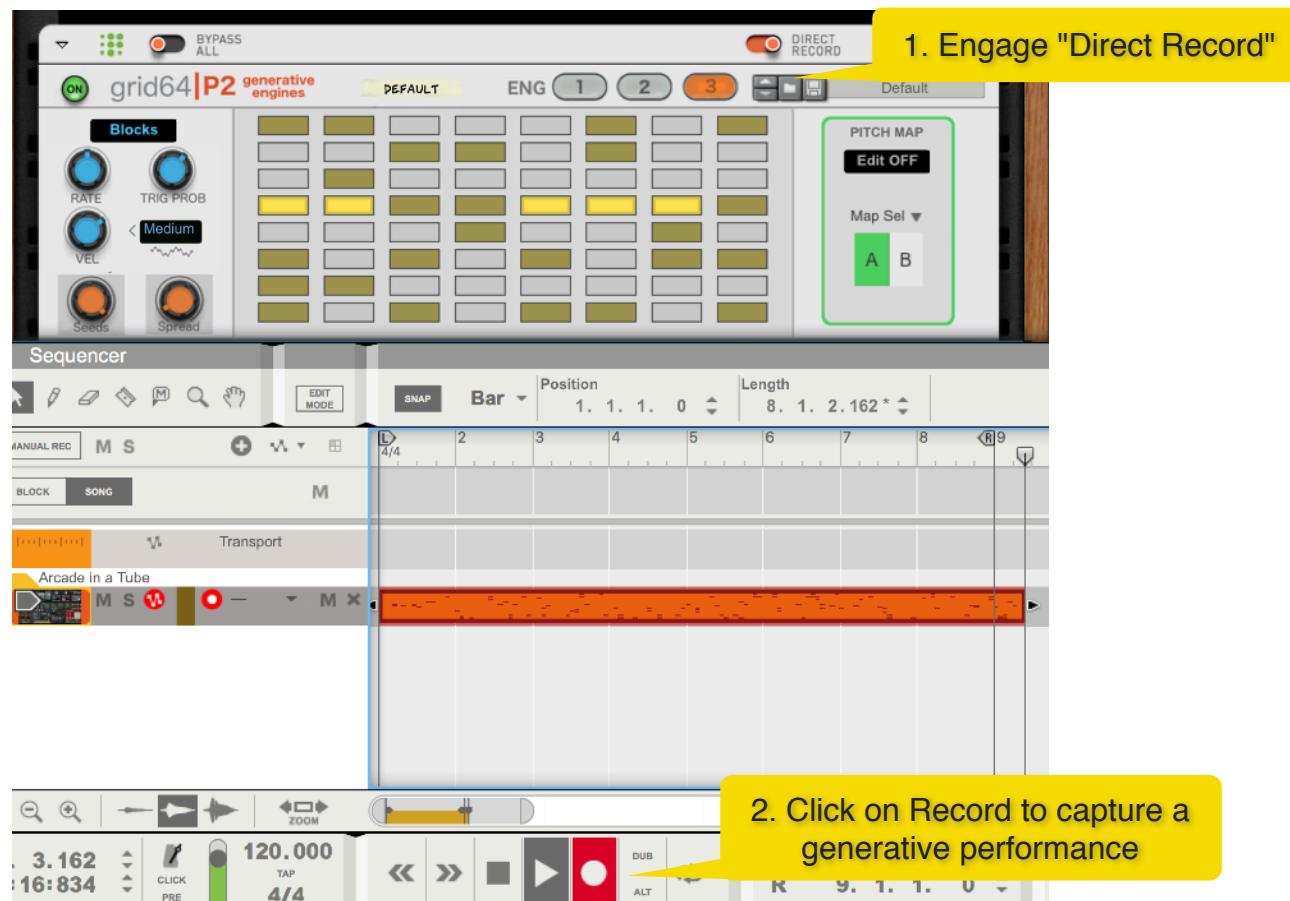
Fill Row: fills the selected row

Fill Column: fills the selected column

Reset: clears all blocks

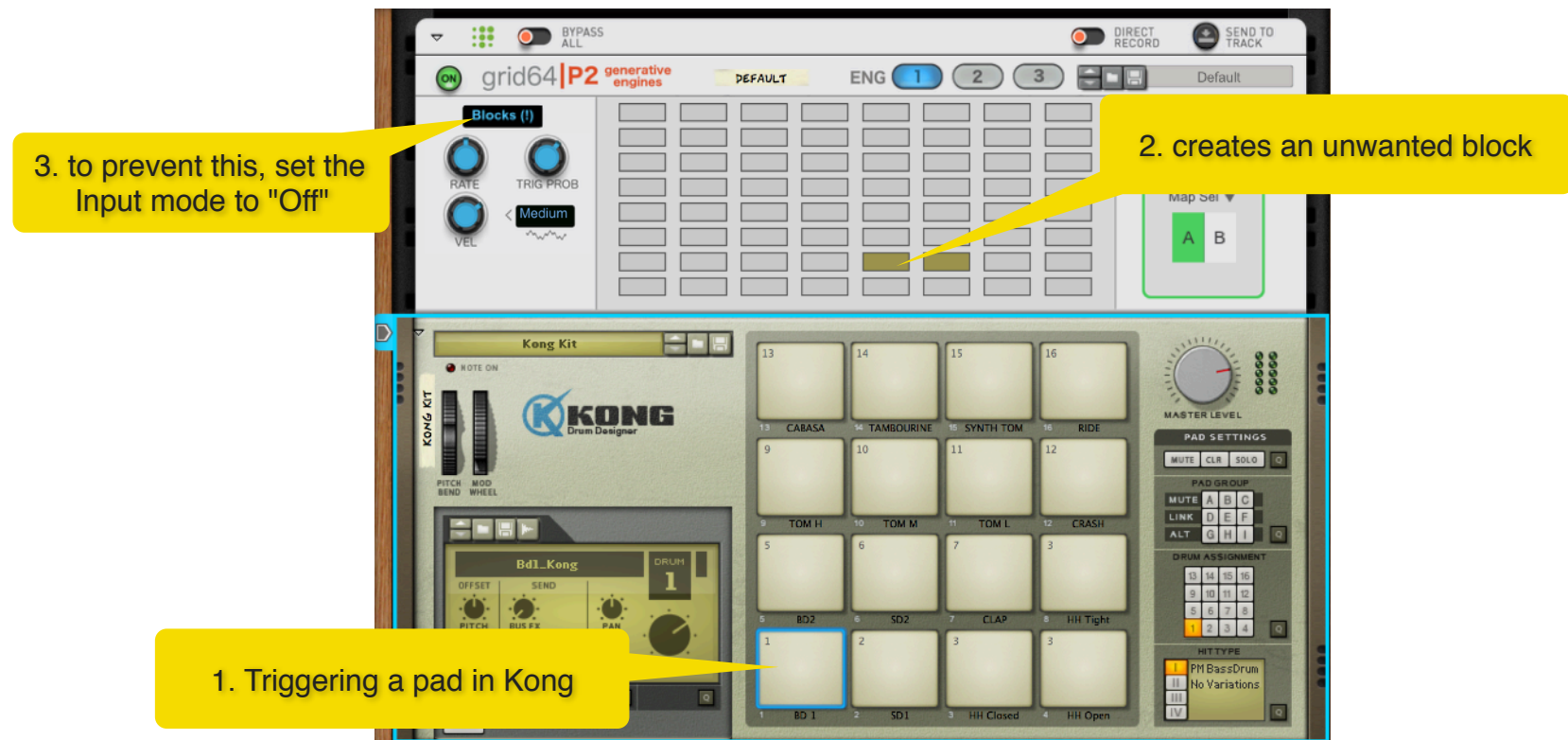
3.1.5 Capturing a generative performance

The player supports patches and there are several parameters which are stored and retained when re-opening a patch or when reopening a song. For example, the pitch map, the configuration of blocks, and the Engine parameters. However, some other crucial things are not stored, like the "Particles" position for Engine 1 and 2, and the cell states and Rules selection for Engine 3. It's therefore important to record a generative performance if the results want to be retained and used again. Luckily, this is easy to do by using the "Direct Record" facility in Reason.



3.1.6 Input Mode Selection

Most of the programming for the three Engines is done when the input mode is set to "Blocks" or "Particles/Rule", as explained in the previous sections. There is however a third "Off" mode which is useful to avoid "accidental" editing when receiving MIDI input from a connected device or from an external MIDI controller. As explained in section 3.3.4, the player responds to incoming MIDI notes which in some cases might be undesired. The following example shows a scenario where the Grid64P2 player is connected to a Kong device. If the Input mode is not set to "Off", triggering a pad in Kong create a block in the player as shown below where both Pad1 and Pad 2 were triggered and caused two blocks to be created. To prevent this behavior, put the Input Mode to "Off" when you are done programming the player.

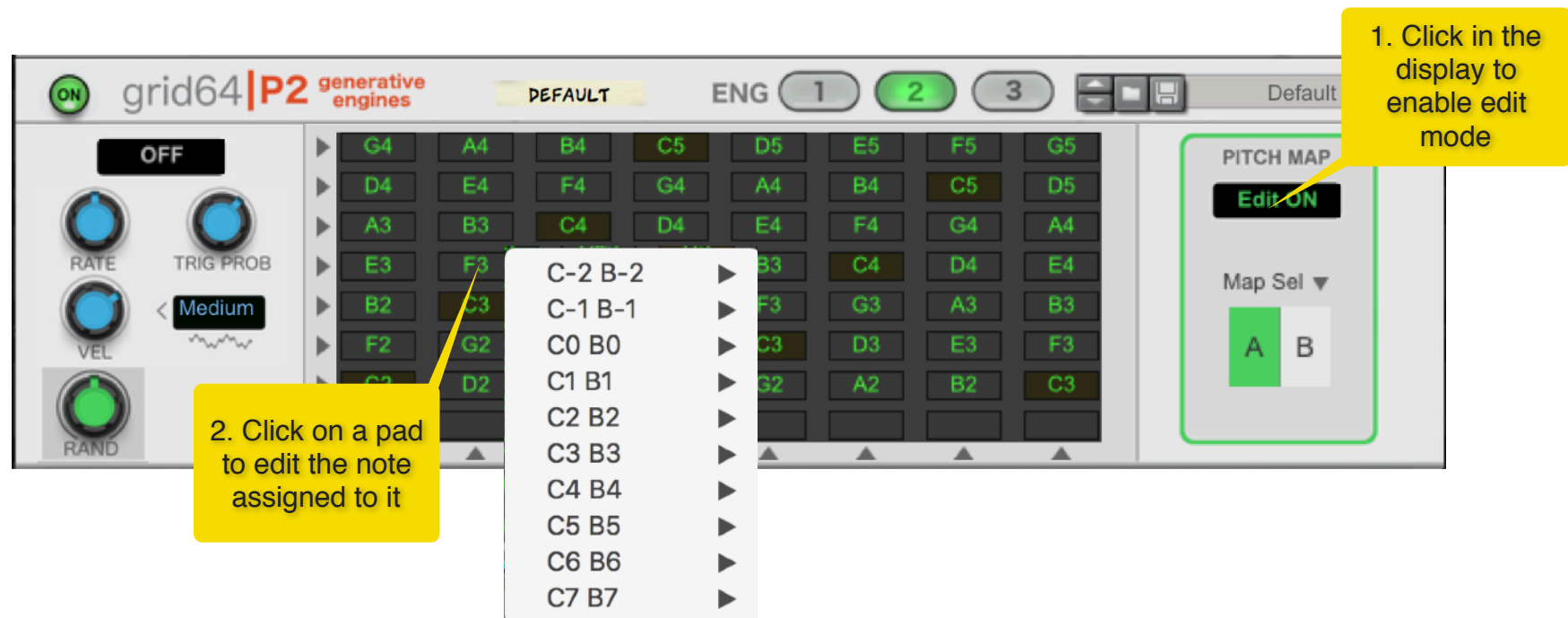


3.2 Creating a note layout

The pads can be customized to contain any specific pitch. The sections below illustrate how to assign a pitch to a single cell and how to perform operations to the entire pitch map.

3.2.1 Editing of Single Pads

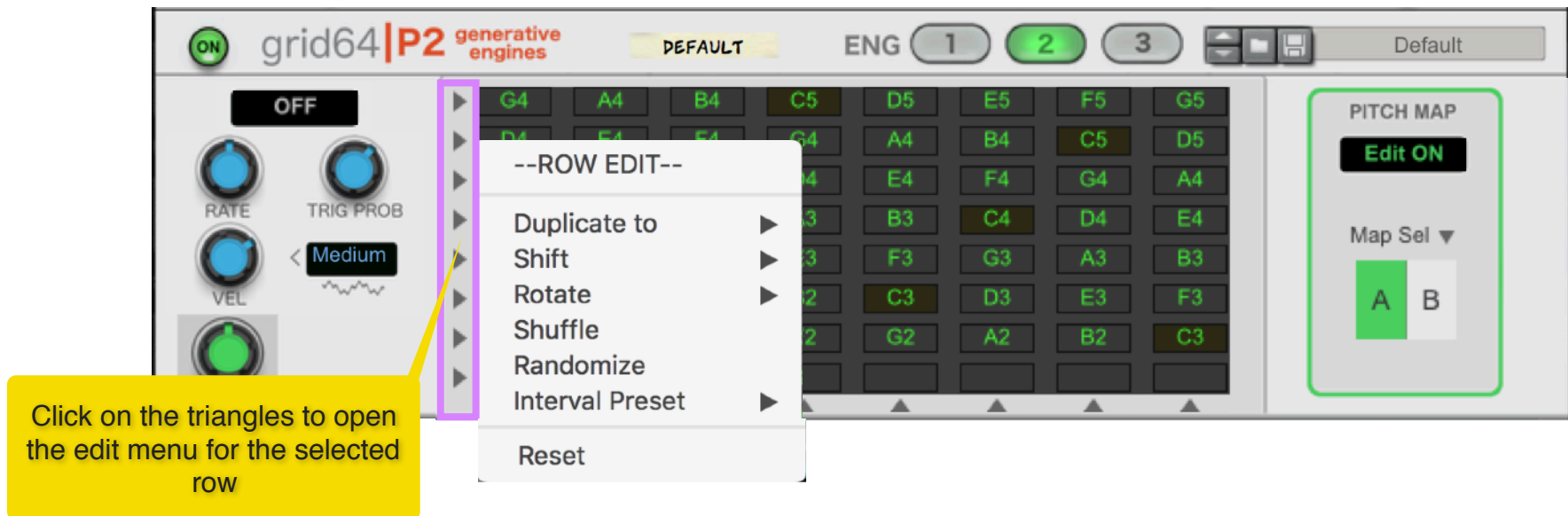
To create your custom mappings, enable the pitch map edit mode by pressing in the display area as shown below. Once edit mode is active, just click on a pad with a note name to change its value by choosing a different note from the context menu. To change the color of the pad, press "Alt" on the keyboard, then click with the mouse on the pad and select a color from the context menu.

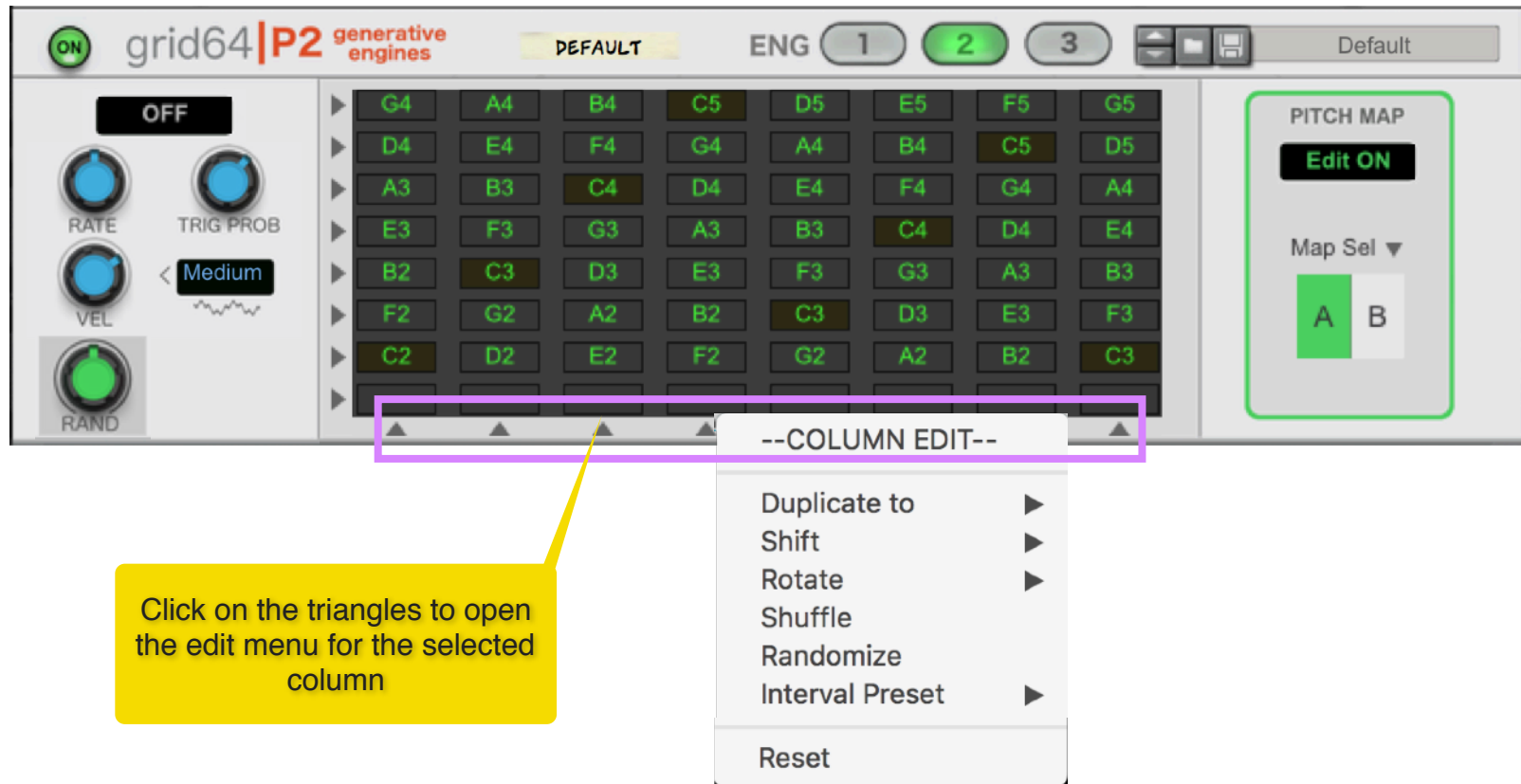


So far, we have discussed editing of single pads. What if we want to edit multiple pads at once? This is possible, and there are 3 ways to do that. You can edit at once single columns, single rows, or the entire map. These options are illustrated below.

3.2.2 Row and Column Edit menus

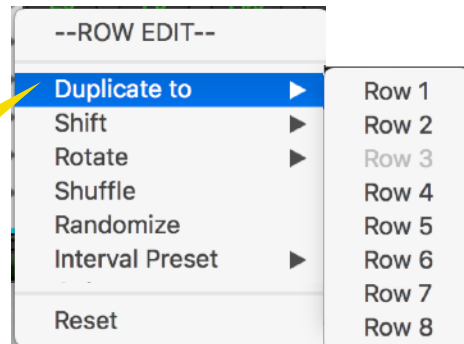
To edit an entire row or column of pads, click on the corresponding triangle to open the Edit context menu.



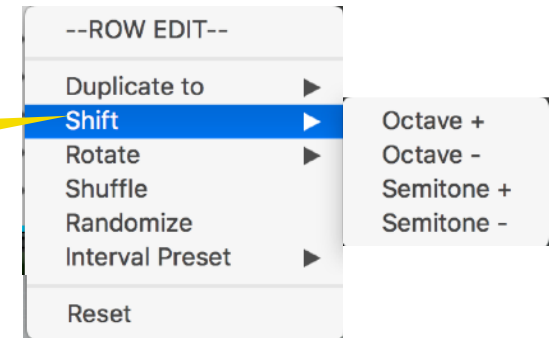


Both the Row and Column edit menus offer several editing options and these are explained below:

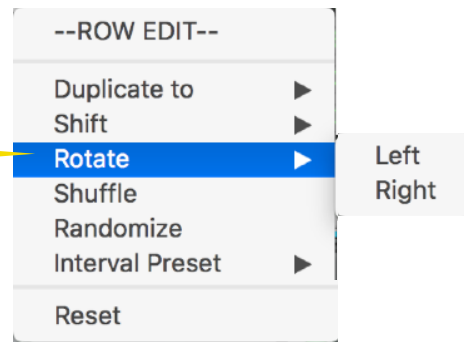
Duplicate the selected Row/Column to another Row/Column



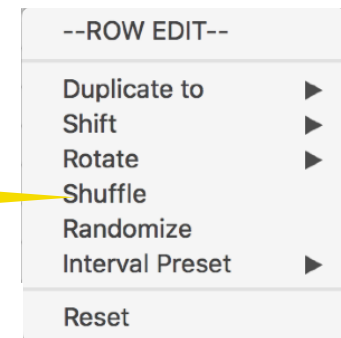
Shifts the notes in the selected Row/Column by octave or semitone



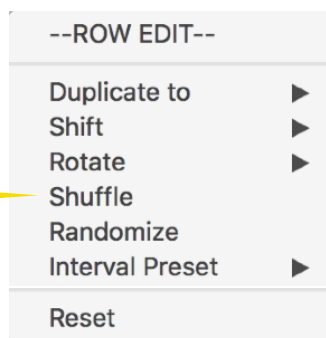
Move the notes by one position left or right for Rows, and up or down for Columns



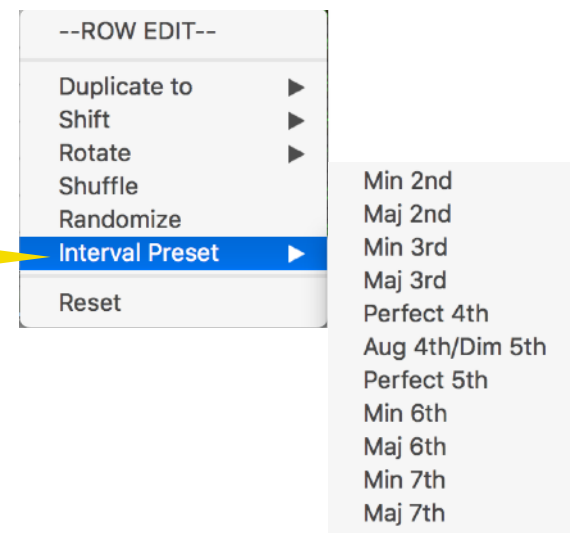
Shuffle around the positions of the notes



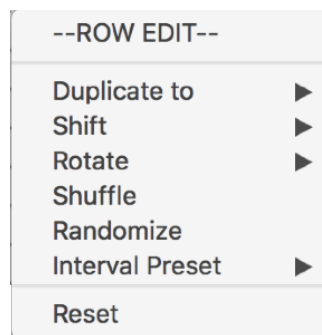
Randomize the values of the notes in the Row/Column



Assign notes with each successive pad being the selected interval apart from its predecessor. The first note in the Row/Column is the starting note in the sequence

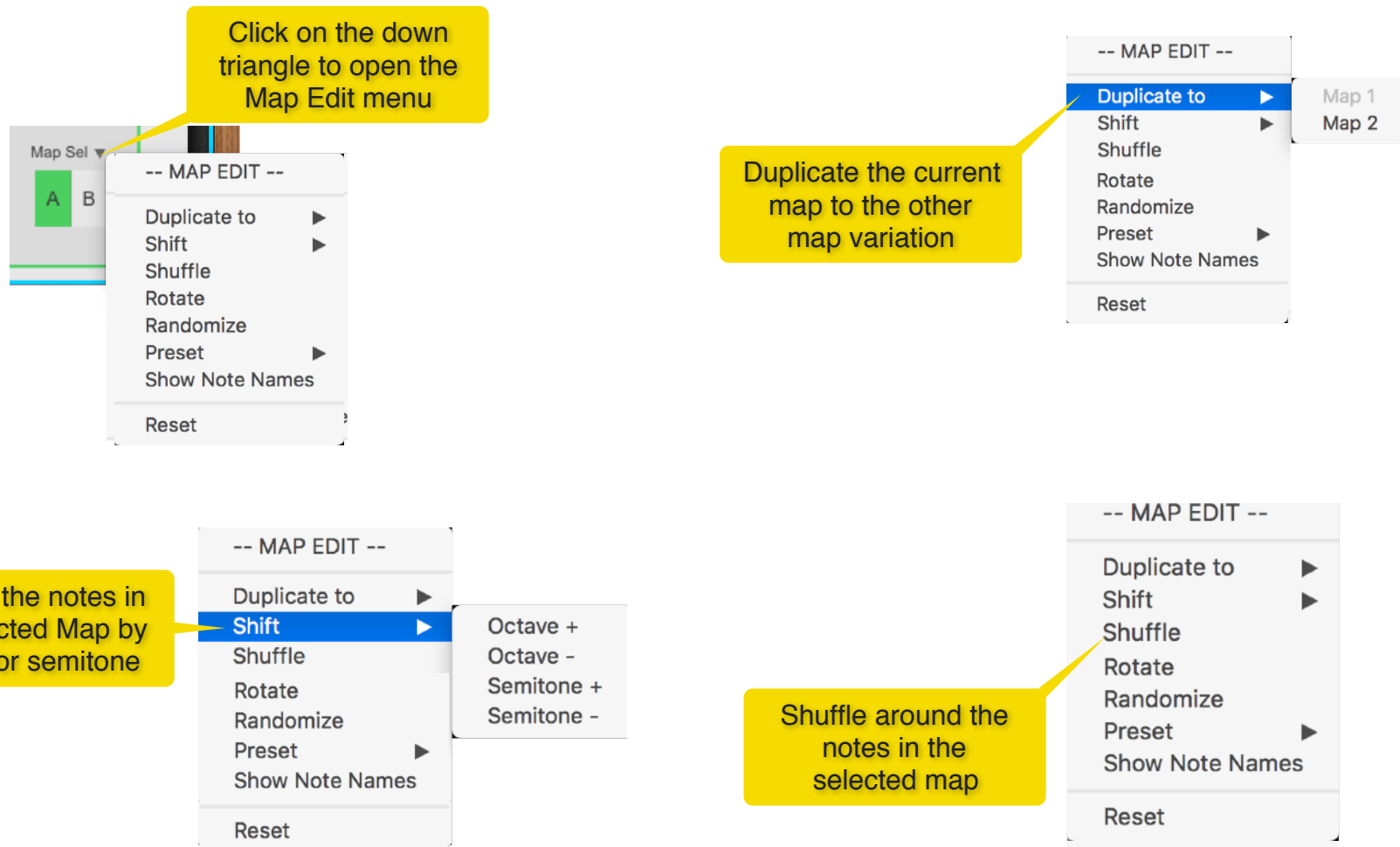


Resets all pads in the Row/Column to default values

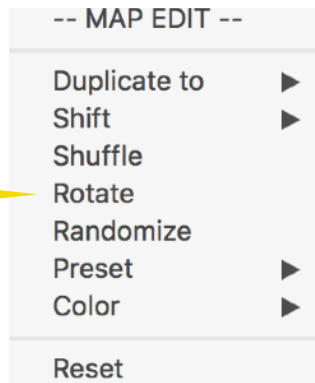


3.2.3 Map Edit menu

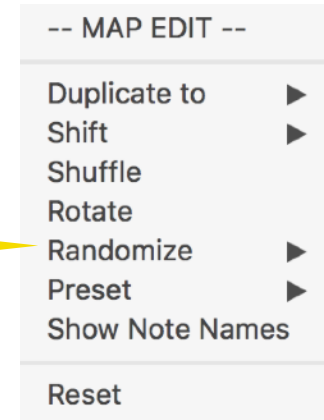
In addition to the Row/Column edit menus, you can also perform edit functions which affect all the pads for the selected Map. You access these functions by clicking on the down triangle in the "Map Sel" area, as shown below.



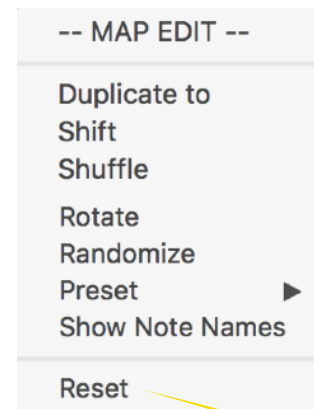
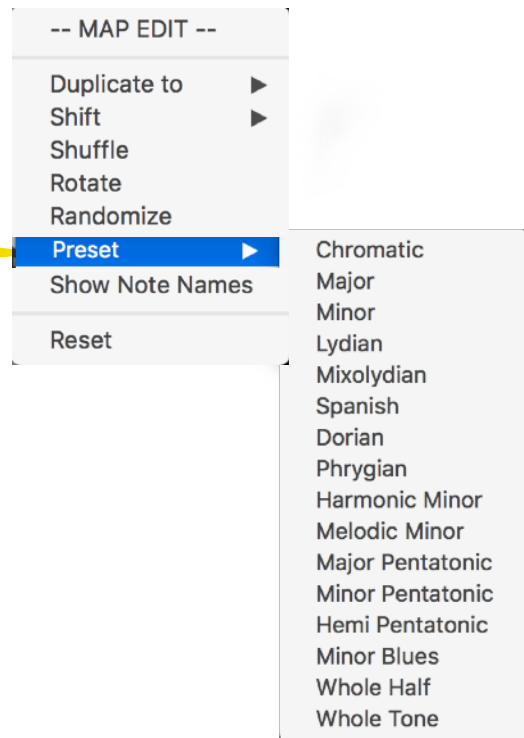
Rotate clockwise the notes in the pitch map



Randomize all the notes in the pitch map. There are options for selecting the octave range for randomization



Create a mapping of the notes according to the selected scale, starting from the leftmost pad on the 1st row as the root note



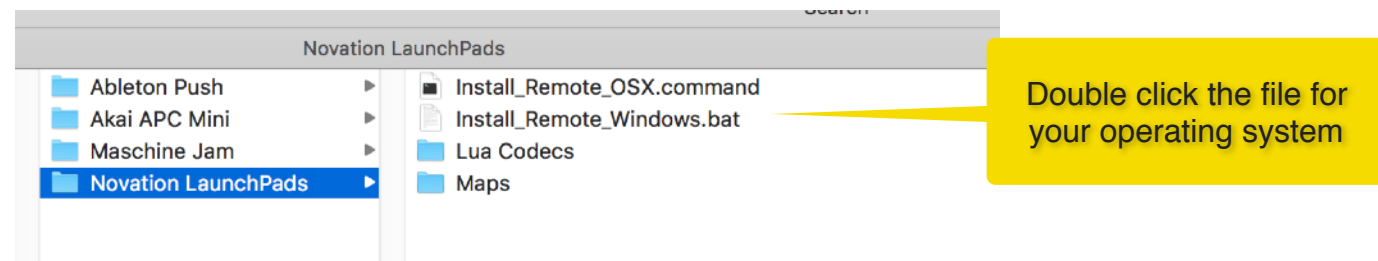
Show the note names for each pad

Reset the map to the default values

3.3 Setting up your grid controller in Reason (standalone)

3.3.1 Install the Remote files

You can download the remote files from the grid64P2 product page in the Reason Studios Shop. Once you have downloaded and unzipped the files, navigate to the folder for your midi controller. In there, you will find two installer files, one for Mac and one for Windows. Double click the installer for your operating system. On Mac OS X, you will see the terminal window open when the process is completed. On Windows, you will see the console flash quickly and then closing.



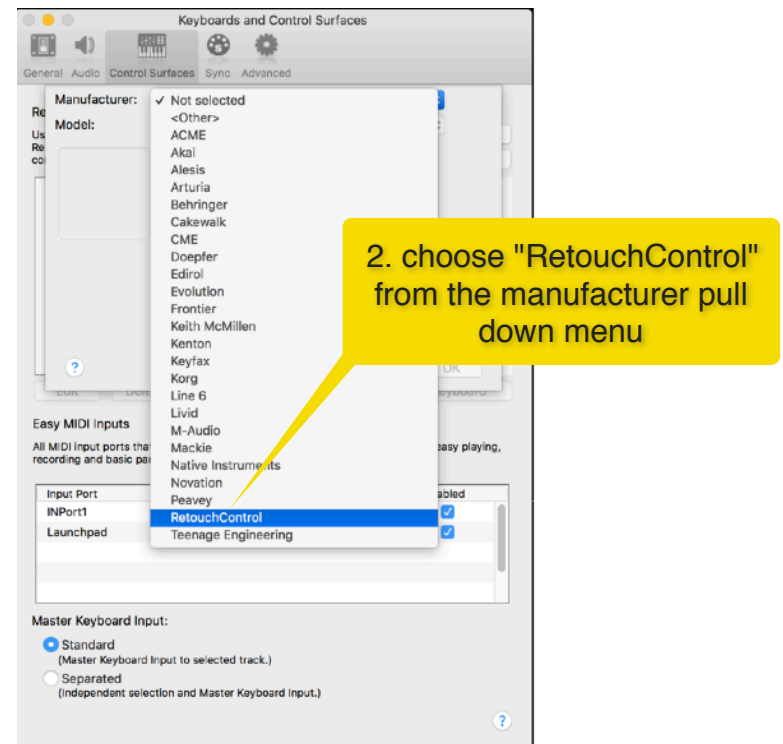
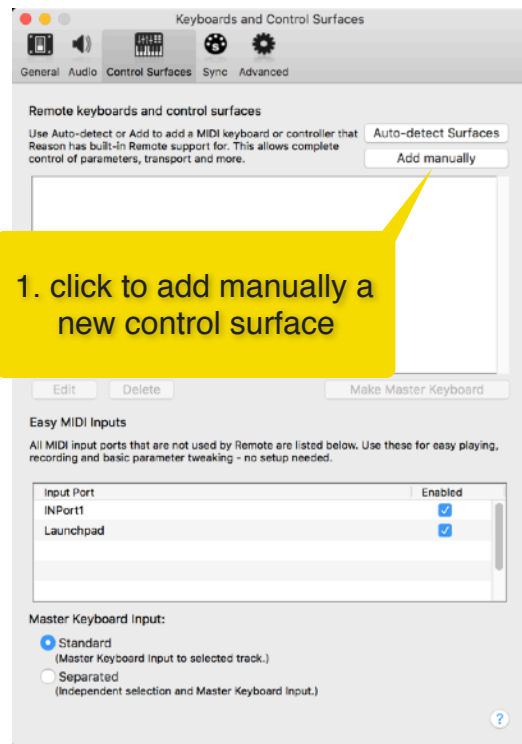
Alternatively, you can copy and paste the remote files manually. This is what you do:

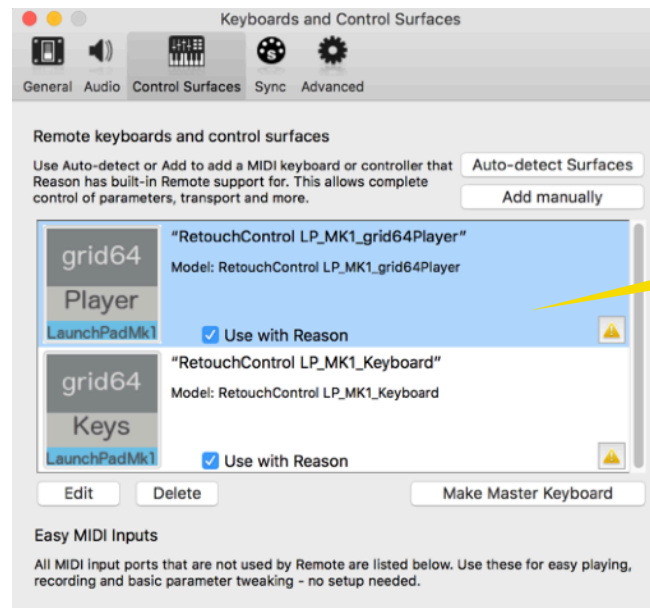
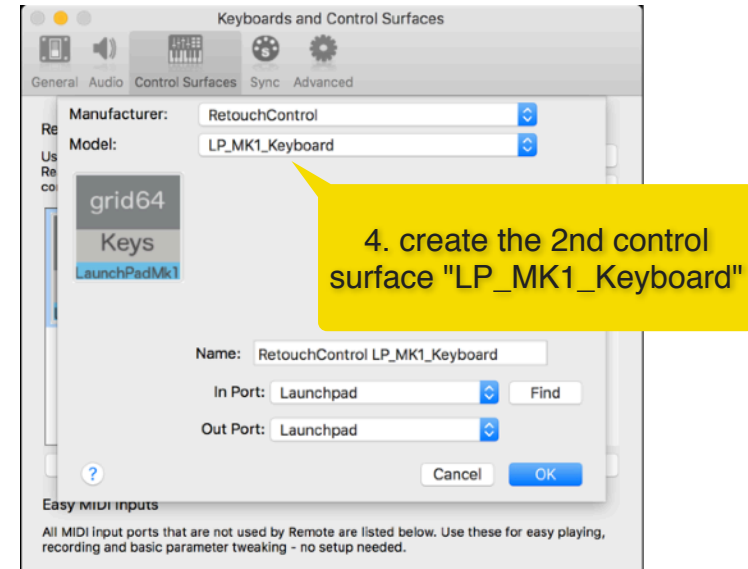
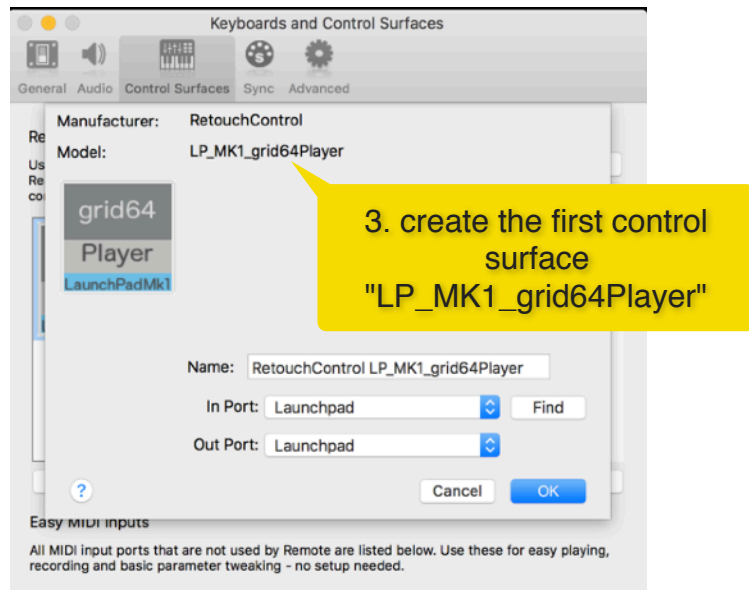
1. Go to the download, open the folder for your controller, then open "Lua Codecs", copy the entire folder in there and paste it at this location:
Mac -> Macintosh HD/Library/Application Support/Propellerhead Software/Remote/Codecs/Lua Codecs
Windows -> C:/ProgramData/Propellerhead Software/Remote/Codecs/Lua Codecs
2. Go back to the download, open the folder for your controller, then open "Maps" copy the entire folder in there and paste it at this location:
Mac -> Macintosh HD/Library/Application Support/Propellerhead Software/Remote/Maps
Windows -> C:/ProgramData/Propellerhead Software/Remote/Maps

Please note, on Windows the directory "ProgramData" is hidden by default. You need to enable "Show hidden files". See how to do it here: <https://support.microsoft.com/en-us/help/14201/windows-show-hidden-files>

3.3.2 Creating Control Surfaces in Reason (standalone)

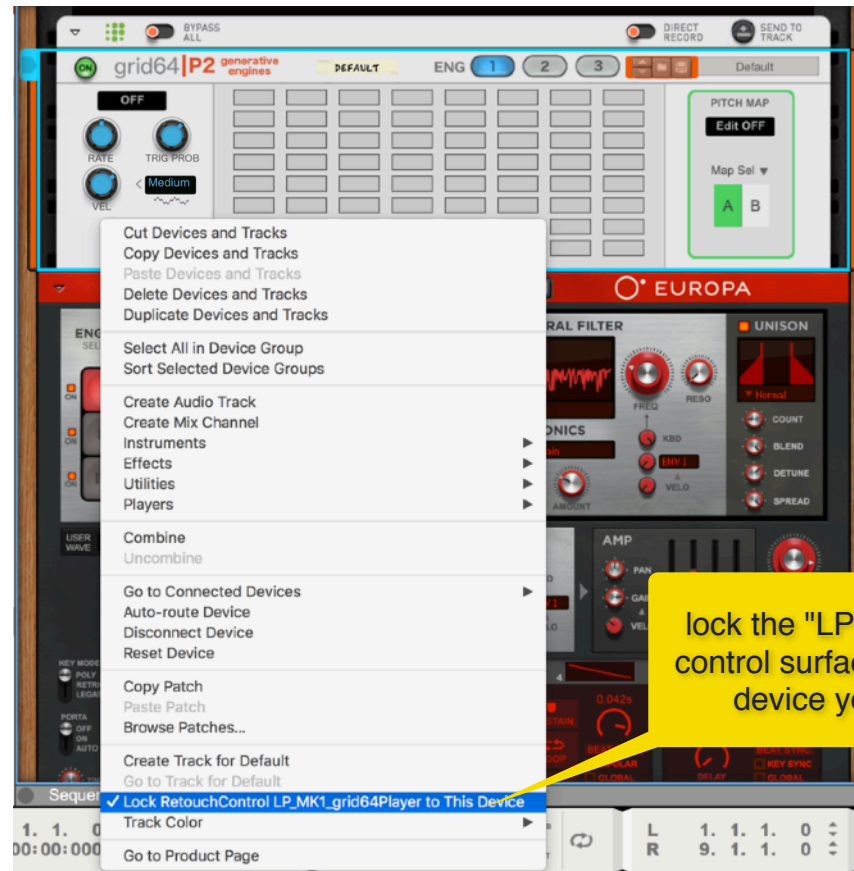
Once the remote files have been installed, start Reason. Go to "Preferences" -> "Control Surfaces" and click on "Add Manually" to create a new control surface. In the example below, we are showing how to configure a Launchpad MK1 as a controller. You will need to create two control surfaces. For each, you use as In and Out ports the midi ports of the Launchpad.





3.3.3 Using the Control Surfaces in Reason (standalone)

Now that the control surfaces have been created, you will be able to play the grid64 Player directly from your controller, but first, you need to lock the "LP_MK1_grid64Player" control surface to the player instance that you want to control, as shown below.



3.3.4 "What if I don't have a grid controller?"

If you don't have one of the supported grid controllers, you can use your MIDI keyboard to trigger the pads. See the following drawing for the details of the midi notes triggering each pad. The notes names shown below are based on the standard note assignment in Reason where C3 corresponds to midi note number 60. Please note, the device does not respond to MIDI messages if the grid Input Mode is set to off. For more information, please see 3.1.6

G#4	A4	A#4	B4	C5	C#5	D5	D#5
C4	C#4	D4	D#4	E4	F4	F#4	G4
E3	F3	F#3	G3	G#3	A3	A#3	B3
G#2	A2	A#2	B2	C3	C#3	D3	D#3
C2	C#2	D2	D#2	E2	F2	F#2	G2
E1	F1	F#1	G1	G#1	A1	A#1	B1
G#0	A0	A#0	B0	C1	C#1	D1	D#1
C0	C#0	D0	D#0	E0	F0	F#0	G0

4. MIDI Implementation

MIDI CC - Parameter

- [12] = Engine,
- [13] = Map Variation
- [14] = Velocity
- [15] = Rate
- [16] = Trigger Probability
- [17] = Engine2 Random
- [18] = Engine3 Spread
- [19] = OnOff

5. Remote Implementation

To obtain the complete list of all the available parameters which are controllable via Remote, use the "Extract Device Remote Info" from the File menu in Reason.

Manufacturer		Model			
Retouch Control		com.retouchcontrol.grid64P2			
Remotable	Min	Max	Input type	Output type	
On/Off	0	1	Toggle	ValueOutput	
Grid 1	0	127	Value	ValueOutput	
Grid 2	0	127	Value	ValueOutput	
Grid 3	0	127	Value	ValueOutput	
Grid 4	0	127	Value	ValueOutput	
Grid 5	0	127	Value	ValueOutput	
Grid 6	0	127	Value	ValueOutput	
Grid 7	0	127	Value	ValueOutput	
Grid 8	0	127	Value	ValueOutput	
Grid 9	0	127	Value	ValueOutput	
Grid 10	0	127	Value	ValueOutput	
Grid 11	0	127	Value	ValueOutput	
Grid 12	0	127	Value	ValueOutput	
Grid 13	0	127	Value	ValueOutput	
Grid 14	0	127	Value	ValueOutput	
Grid 15	0	127	Value	ValueOutput	
Grid 16	0	127	Value	ValueOutput	
Grid 17	0	127	Value	ValueOutput	
Grid 18	0	127	Value	ValueOutput	
Grid 19	0	127	Value	ValueOutput	
Grid 20	0	127	Value	ValueOutput	
Grid 21	0	127	Value	ValueOutput	
Grid 22	0	127	Value	ValueOutput	
Grid 23	0	127	Value	ValueOutput	
Grid 24	0	127	Value	ValueOutput	

Grid 25	0	127	Value	ValueOutput
Grid 26	0	127	Value	ValueOutput
Grid 27	0	127	Value	ValueOutput
Grid 28	0	127	Value	ValueOutput
Grid 29	0	127	Value	ValueOutput
Grid 30	0	127	Value	ValueOutput
Grid 31	0	127	Value	ValueOutput
Grid 32	0	127	Value	ValueOutput
Grid 33	0	127	Value	ValueOutput
Grid 34	0	127	Value	ValueOutput
Grid 35	0	127	Value	ValueOutput
Grid 36	0	127	Value	ValueOutput
Grid 37	0	127	Value	ValueOutput
Grid 38	0	127	Value	ValueOutput
Grid 39	0	127	Value	ValueOutput
Grid 40	0	127	Value	ValueOutput
Grid 41	0	127	Value	ValueOutput
Grid 42	0	127	Value	ValueOutput
Grid 43	0	127	Value	ValueOutput
Grid 44	0	127	Value	ValueOutput
Grid 45	0	127	Value	ValueOutput
Grid 46	0	127	Value	ValueOutput
Grid 47	0	127	Value	ValueOutput
Grid 48	0	127	Value	ValueOutput
Grid 49	0	127	Value	ValueOutput
Grid 50	0	127	Value	ValueOutput
Grid 51	0	127	Value	ValueOutput
Grid 52	0	127	Value	ValueOutput
Grid 53	0	127	Value	ValueOutput
Grid 54	0	127	Value	ValueOutput
Grid 55	0	127	Value	ValueOutput
Grid 56	0	127	Value	ValueOutput
Grid 57	0	127	Value	ValueOutput
Grid 58	0	127	Value	ValueOutput
Grid 59	0	127	Value	ValueOutput
Grid 60	0	127	Value	ValueOutput

Grid 61	0	127	Value	ValueOutput
Grid 62	0	127	Value	ValueOutput
Grid 63	0	127	Value	ValueOutput
Grid 64	0	127	Value	ValueOutput
Map	0	1	Toggle	ValueOutput
Engine	0	2	Value	ValueOutput
Input Mode	0	2	Value	ValueOutput
Velocity	0	127	Value	ValueOutput
Trigger Probability	0	4194304	Value	ValueOutput
Engine2 Random	0	4194304	Value	ValueOutput
Engine3 Seed	0	8	Value	ValueOutput
Engine3 Spread	0	4194304	Value	ValueOutput
Rate	0	12	Value	ValueOutput
Grid LED 1	0	7	-	ValueOutput
Grid LED 2	0	7	-	ValueOutput
Grid LED 3	0	7	-	ValueOutput
Grid LED 4	0	7	-	ValueOutput
Grid LED 5	0	7	-	ValueOutput
Grid LED 6	0	7	-	ValueOutput
Grid LED 7	0	7	-	ValueOutput
Grid LED 8	0	7	-	ValueOutput
Grid LED 9	0	7	-	ValueOutput
Grid LED 10	0	7	-	ValueOutput
Grid LED 11	0	7	-	ValueOutput
Grid LED 12	0	7	-	ValueOutput
Grid LED 13	0	7	-	ValueOutput
Grid LED 14	0	7	-	ValueOutput
Grid LED 15	0	7	-	ValueOutput
Grid LED 16	0	7	-	ValueOutput
Grid LED 17	0	7	-	ValueOutput
Grid LED 18	0	7	-	ValueOutput
Grid LED 19	0	7	-	ValueOutput
Grid LED 20	0	7	-	ValueOutput
Grid LED 21	0	7	-	ValueOutput
Grid LED 22	0	7	-	ValueOutput
Grid LED 23	0	7	-	ValueOutput

Grid LED 24	0	7	-	ValueOutput
Grid LED 25	0	7	-	ValueOutput
Grid LED 26	0	7	-	ValueOutput
Grid LED 27	0	7	-	ValueOutput
Grid LED 28	0	7	-	ValueOutput
Grid LED 29	0	7	-	ValueOutput
Grid LED 30	0	7	-	ValueOutput
Grid LED 31	0	7	-	ValueOutput
Grid LED 32	0	7	-	ValueOutput
Grid LED 33	0	7	-	ValueOutput
Grid LED 34	0	7	-	ValueOutput
Grid LED 35	0	7	-	ValueOutput
Grid LED 36	0	7	-	ValueOutput
Grid LED 37	0	7	-	ValueOutput
Grid LED 38	0	7	-	ValueOutput
Grid LED 39	0	7	-	ValueOutput
Grid LED 40	0	7	-	ValueOutput
Grid LED 41	0	7	-	ValueOutput
Grid LED 42	0	7	-	ValueOutput
Grid LED 43	0	7	-	ValueOutput
Grid LED 44	0	7	-	ValueOutput
Grid LED 45	0	7	-	ValueOutput
Grid LED 46	0	7	-	ValueOutput
Grid LED 47	0	7	-	ValueOutput
Grid LED 48	0	7	-	ValueOutput
Grid LED 49	0	7	-	ValueOutput
Grid LED 50	0	7	-	ValueOutput
Grid LED 51	0	7	-	ValueOutput
Grid LED 52	0	7	-	ValueOutput
Grid LED 53	0	7	-	ValueOutput
Grid LED 54	0	7	-	ValueOutput
Grid LED 55	0	7	-	ValueOutput
Grid LED 56	0	7	-	ValueOutput
Grid LED 57	0	7	-	ValueOutput
Grid LED 58	0	7	-	ValueOutput
Grid LED 59	0	7	-	ValueOutput

Grid LED 60	0	7	-	ValueOutput	
Grid LED 61	0	7	-	ValueOutput	
Grid LED 62	0	7	-	ValueOutput	
Grid LED 63	0	7	-	ValueOutput	
Grid LED 64	0	7	-	ValueOutput	
Device Name	0	0	-	TextOutput	
Patch Name	0	0	-	TextOutput	
Select Patch Delta		0	0	Delta	TextOutput
Select Previous Patch		0	0	Trig	TextOutput
Select Next Patch		0	0	Trig	TextOutput

6. Version History

Version 1.0.0: initial release

Version 1.0.1:

- after pressing play, the first note is now quantized according to the selected rate. For example, for 1/16 rate, the first note is quantized to the next 1/16 grid division. For 1/8 rate, the first note is quantized to the next 1/8 grid division, and so on.
- added new Randomize options in the Block edit menu, at 25%, 50%, 75% and 100% randomization (the last one fills the grid entirely)
- added new Fill Row and Fill Column options in the Block edit menu to quickly fill the selected row or column
- added new Randomize options in the Map edit menu. It is possible to randomize over the full range of notes, or only 2 octaves, 3 octaves or 4 octaves.

Version 1.0.2: Reason 12 HD optimizations

Version 1.0.4: added On/Off button for automation and remote control