

cheat codes



v1.1.1.1.1.1.1.1

cheat codes is a live + pre-recorded sample playground for norns.



It's called cheat codes because it's sorta like being transported 20 minutes into a cranes session, where things start to get meaty. Instead of having to play through, you can just start at the boss level.

It's also called cheat codes because it uses ~25 different grid key finger combos to manipulate three Softcut voices. Softcut is the sampling engine built directly into norns – cheat codes does not require any additional engines to be installed.

cheat codes can also record gestures on grid + arc to navigate changes to a bunch of different parameters (sample windowing, playback position, rate + direction, filter, which sample is playing, one-shot/loop, panning, volume, etc). These gestures are stored as patterns and can be quantized to a clock (internal or external, MIDI or CV), saved between sessions (each bank has 8 save slots), and sequenced using a built-in meta-sequencer similar to Kria.

### **requirements:**

- norns (191201 or later)
- grid highly suggested – full functionality is harder to access without grid
- arc strongly encouraged – arc provides fine-tuned control over parameters and gestures can be recorded for playback on top of grid patterns

This guide will be updated as folks ask questions, challenge my assumptions, and share their work.

### dynamic help:

Included in cheat codes is a dynamic help menu, which can be accessed by selecting [?]. In this menu, you can hit any key on an attached grid to learn more about it - this includes both its primary function and how it works in cooperation with other keys:

```
help
bank: 2 | pad: 13
pads recall parameters:
- rate: 1.00 - pan: 0.81
- start: 7.31 - end: 7.96
- loop: false etc.
...
```

```
help
4th row action: 124
- 2x pad's current rate
- upper limit: 4x

rate: 2.0x          try: 134
...
```

The help menu can be accessed at any point and does not interrupt your current session. Every action you take in the help menu executes in real-time, so you can jump in and out of playing and learning.

---

### Table of contents:

grid overview -----> p 3  
-----banks + pads -----> p 4  
-----zilchmo -----> p 5  
-----buffers -----> p 7  
Patterns + [timing] -> p 9  
-- meta-sequencing -> p 13  
Buffers + [loops] -> p 16  
-----gridless-----> p 18  
[levels] -----> p 19  
[panning] -----> p 20  
[filters] -----> p 21  
[delay] -----> p 22  
arc -----> p 23  
crow -----> p 24  
save/load -> p 25

```
cheat codes

[ loops ]   filters
levels      delay
panning     timing

?
```

All illustrations made by Zach Spindler, founder of [Zilchmo's Za](#) (a Creative Studio)

When you plug grid into cheat codes, turn it so it's 16 tall and 8 wide (versus the standard 8 tall and 16 wide) and this is what you'll see:

```
x x x x | 1 2 3 -
x x x x | L C - P
x x x x | * - Z Z
x x x x | - Z Z Z
- - - - | Z Z Z Z
x x x x | 1 2 3 -
x x x x | L C - P
x x x x | * - Z Z
x x x x | - Z Z Z
- - - - | Z Z Z Z
x x x x | 1 2 3 -
x x x x | L C - P
x x x x | * - Z Z
x x x x | - Z Z Z
- - - - | Z Z Z Z
A 1 2 3 | p p p m
```

cheat codes is broken up into three banks of two main playing surfaces:

### banks and pads

```
x x x x
x x x x
x x x x
x x x x
```

and...

### Zilchmo's right-angle slice

```
- - - P: pattern recorder
- - Z Z: level + play/pause
- Z Z Z: panning
Z Z Z Z: start/end points, rate, direction
```

There are some additional functions in-between and underneath, which we'll cover later on, but those two sets of grid keys will be the ones you use most.

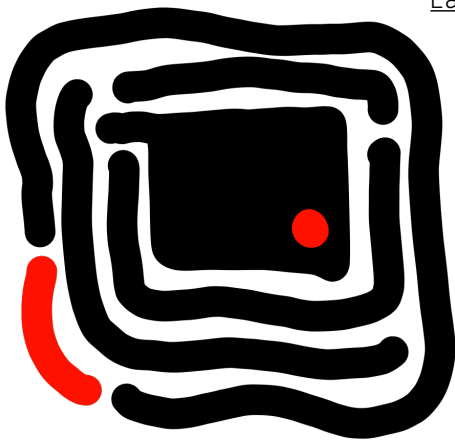
In the bottom-left corner of the grid is a momentary **ALT** button, which can be held to change the scope of a control from local to global, or vice versa. Any time a control has an ALT behavior, we'll mention it!





## banks and pads

There are three banks, a b and c. Each bank has 16 pads (x's). Pads within the same bank cut each other off when triggered (think "choke group"), but they do not affect pads in the other banks - cheat codes has 3 voice polyphony.



Each pad contains its own individual settings for the following parameters:

- which buffer to use (Live or Clip)
- which buffer segment to use (there are three Live segments and three Clip segments)
- playback start + end points
- playback rate / pitch + direction
- whether to play as a loop or 1-shot
- level / gain
- envelope settings
- panning position
- filter tilt
- filter ease timing + style

When you load a fresh session, these are the defaults for each bank's pads:

- *which buffer to use*: Live buffer 1
- *which buffer segment to use*: segment 1 of Live buffer 1
- *playback start + end points*: each pad will step through an even distribution of the 8 second buffer, eg:
  - pad 1 starts at 0, ends at 0.5
  - pad 2 starts at 0.5, ends at 1
  - pad 7 starts at 3, ends at 3.5
- *playback rate / pitch + direction*: each pad plays the recorded audio at 1x rate, forward
- *whether to play as a loop or 1-shot*: each pad is set to loop
- *level / gain*: each pad has gain 1.0
- *envelope settings*: no envelope is applied to any pads
- *panning position*: each pad is panned to center
- *filter tilt*: each pad is set to neutral, with a mild resonance
- *filter ease timing + style*: each pad has a 0.5s filter frequency glide and each pad is set to "cont(inuous)" changes

## copy/paste

Holding ALT while pressing pads lets you copy/paste parameter information from one pad to another, even across banks. To copy/paste:

- hold ALT
- select a pad to copy
- select a pad to overwrite

## Zilchmo's right-angle slice

My friend Zach is starting a pizza company. It's called Zilchmo's and its specialty is square-shaped pizzas, cut diagonally into two right-angle slices.

In cheat codes, the big right triangle next to every bank is a Zilchmo, split into four rows of quick-control keys which affect each pad's parameters:

- - - P: pattern recorder
- - Z Z: level + play/pause
- Z Z Z: panning
- Z Z Z Z: start/end points, rate, direction



The keys in each Zilchmo row each have their own function, but you can also combine keys horizontally for additional functions. This is where cheat codes starts to feel like mashing buttons in a video game to unlock hidden features. The paradigm was initiated by Rod Constanzo's Block Party, which was goddamn gold.

Starting from the bottom row:

### Z Z Z Z: start/end points, rate, direction

1	2	3	4	function
x	-	-	-	set pad's start (s) point to 0
-	x	-	-	restore default (s) + (e) points (based on pad ID)
-	-	x	-	1/16th @ bpm (s) + (e) points (based on pad ID)
-	-	-	x	set pad's end (e) point to 8
x	x	-	-	random (s) point
-	-	x	x	random (e) point
-	x	x	-	random window, (s) + (e) move together
x	-	x	-	double the loop length
-	x	-	x	halve the loop length, (s) + (e) move inward
x	x	x	-	loop sync across banks: a = b, b = c, c = a
-	x	x	x	loop sync across banks: a = c, b = a, c = b
x	x	-	x	2x current rate (4x max)
x	-	x	x	0.5x current rate (0.125x min)
x	-	-	x	toggle reverse playback
x	x	x	x	1.5x current rate (raise a fifth)

### Z Z Z: panning

-	1	2	3	function
-	x	-	-	hard-pan pad L
-	-	x	-	hard-pan pad C
-	-	-	x	hard-pan pad R
-	x	x	-	nudge pad's panning to L
-	-	x	x	nudge pad's panning to R
-	x	-	x	reverse pad's current panning
-	x	x	x	random panning

### Z Z: levels + play/pause

-	-	1	2	function
-	-	x	-	reduce pad's level by -0.125
-	-	-	x	increase pad's level by +0.125
-	-	x	x	toggle pad's playback (bright is paused)

*nb. Though cheat codes can parse multi-finger presses within a right-angle slice, it can only parse presses on a single right-angle slice at a time. This won't limit playability, as you'll often use your left hand to play pads in the banks and your right hand to change parameters in the right-angle slice. You can even execute multiple commands on the same right-angle slice at the same time. Just don't get worried if you try to execute multiple commands on two separate right-angle slices at the same exact time and one of them doesn't happen.*

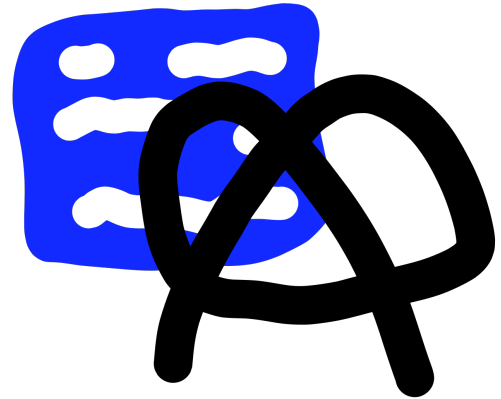
### bank-wide changes

Holding ALT while performing a Zilchmo gesture will change the scope of the action to affect all the pads in the corresponding bank. For example, hold ALT and execute 2x current rate to multiply every pad's rate by 2x (1 -> 2, 2 -> 4, 0.25 -> 0.5, etc)

cheat codes has 6 buffers: 3 loopers for live recording + 3 static clips to load pre-recorded samples. To make things simple, these buffers are capped at 8 seconds of audio each.

Buffers can be switched and managed from grid directly, using the 1 2 3 and L C keys in the first and second row located between the bank and the Zilchmo right-angle slice:

```
x x x x | 1 2 3 -  
x x x x | L C - P  
x x x x | * - Z Z  
x x x x | - Z Z Z  
- - - - | Z Z Z Z
```



### L(ive) and C(lip)

Using the L and C keys, you can tell the selected pad to apply its stored parameters to either the Live input or the loaded Clip.

If you load a Clip and play a Live instrument in the same tonality, a nice performance trick is to create a pattern of pad presses and dynamically change whether the pad is playing from the pre-recorded Clip or from the Live audio.

### 1 2 3

After you set a pad's audio stream (Live or Clip), use these keys to switch between the three buffers for that stream. For example: load 3 Clips in the PARAMS and for each of three pads in a bank, select Clip and choose a unique buffer for each. This allows you to play audio from three different pre-recorded samples in a single bank.

### the 1 2 3 at the bottom of the grid

The bottom row of the grid interface looks like this:

```
A 1 2 3 | p p p m
```

Let's just focus on the 1 2 3 keys. This is how you can change which buffer the Live loop is recording into. By default, cheat codes is set to record into buffer 1. You can change this to record into buffer 2 or 3 with just a key press!

### clearing your Live buffers

To clear the audio between the Live buffer's loop points, hold ALT and press the 1 2 3 key corresponding to the buffer you wish to clear. This will also put the record head into a disabled state, so you can choose when to record anew.

Let's change how the record head moves between the loop points!

Head to PARAMS, where you can switch the **live rec behavior** to *loop* or *1-shot*.

## loop

- Live buffer will loop between the start and end points
- When you press any of the 1 2 3 keys, you will switch to a new Live buffer and immediately begin recording; the loop window stays the same length
- If you press any of the 1 2 3 keys twice, you will toggle punch-in / punch-out (the grid LED will be bright for punch-in, dim for punch-out)

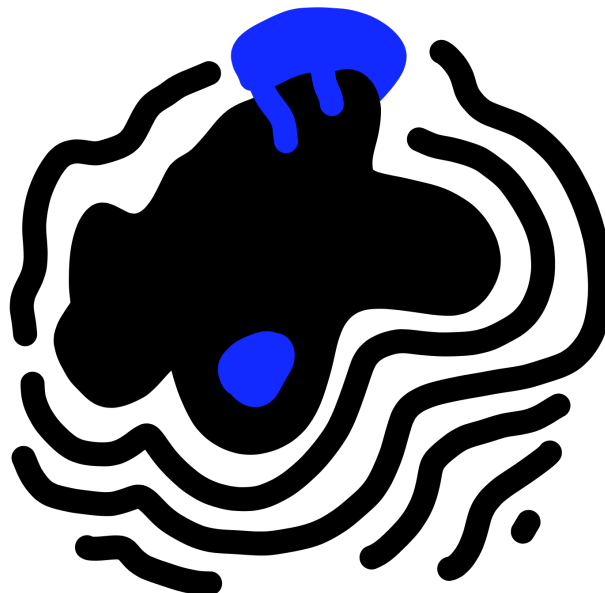
## 1-shot

- Live buffer will perform a single recording pass from the start point to the end point
- When you press any of the 1 2 3 keys, you will switch to a new Live buffer and immediately begin recording; recording will stop when the end point has been reached and the loop window will stay the same
- If you press any of the 1 2 3 keys twice before the record head has reached the end point, the record head will jump back to the beginning of the Live buffer
- When the record head is recording, the grid LED will be bright; when the record head reaches the end point, it will stop recording and the grid LED will dim

By adjusting the Live buffer's loop points and behavior, you can easily structure a fun playground.

## World 1: Recording Live

- Set the live rec behavior to 1-shot
- On the screen, set unique playhead loop points for the pads in banks (a), (b), and (c) so they each cover their own slice of the Live buffer
- Narrow the Live buffer's loop points and navigate to one of the sections that has a playhead
- Play your external sound source and trigger the 1-shot recording
- Navigate the Live buffer's loop to another section with a different playhead
- Play your external sound source and trigger the 1-shot recording
- Rinse + repeat to build a galaxy of micro-loops inside of the Live buffer



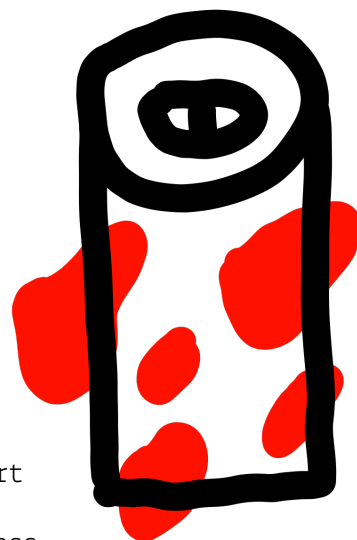
## Patterns: an introduction

At the far (top) corner of each Zilchmo right-angle slice is a pattern key, which records pad playing for the Zilchmo's corresponding bank. Patterns will not record presses from other banks, only the bank to the left of the Pattern key. This creates opportunity for asynchronous pattern looping, where pad presses in each bank phase in and out of time.

Pattern recording has two modes: classic and rad sauce. Switch between them in the PARAMS menu under pattern rec style.

**classic:** v1 behavior. Record pad presses in the bank, play them back. Zilchmo modifications change the pads parameters as the Pattern plays back. Great for morphing looping patterns + injecting quick shifts.

**rad sauce:** Pad presses and gestures on the fourth row of the Zilchmo right-angle slice are recorded. When the Pattern plays back, it will recall a pad's state both pre and post-Zilchmo. Really nice for capturing + repeating performances.



There are various LED indicators:

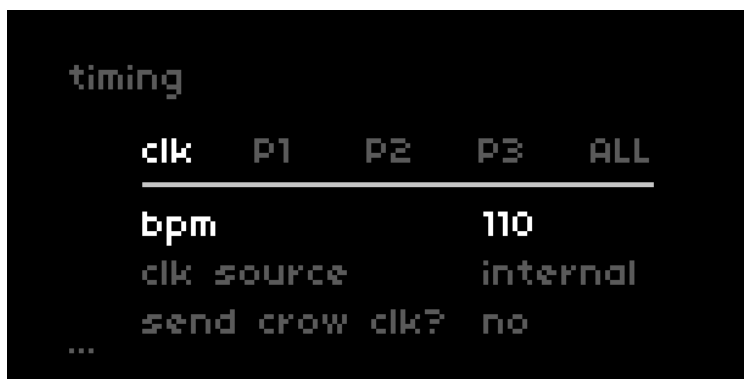
- base-light: No pattern recorded. Press the Pattern key to start recording!
- blinking light: Pattern recording. Play some pads and then press the Pattern key to stop recording + start playback.
- bright solid light: Recorded Pattern is playing. Press the Pattern key to stop playback.
- mid-level solid light: Pattern is recorded, but not playing. Press to the Pattern key to start playback.

## clearing Patterns

Holding ALT while pressing any Pattern key will clear out that recorded Pattern.

## [timing]

If you've recorded a pattern but things just don't *flow* the way you'd hoped (or you want to keep your flow but use an external clock source to play your Pattern back OR you want to subvert your flow altogether), open [timing] from the main screen.



[timing] holds a number of powerful + quickly accessible controls. Use E1 to navigate across the different

domains, E2 to choose a submenu item, and E3 to change it.

The *P*(attern) domains feature some K3 commands without any E3 options and some E3 options which must be executed with a K3 press (eg. tap tempo on *bpm*):



	clk	P1	P2	P3	ALL
linearize				[K3]	
snap to bars				1 [+K3]	
crow output				pads	
...					

The *ALL* domain gives global control over linearizing and in-the-moment quantizing:



	clk	P1	P2	P3	ALL
linear recording?					no
quantize pads?					no
quant resolution					1/16
...					

### linearization, quantization, and snap to bars

cheat codes uses three methods to make adjustments to time:

- *quantization* is a performance parameter which suppresses pad press actions until the next clock event
- *linearization* is a post-processing tool to micro-nudge the events in your Patterns into bpm-aware timing, based on the specified *quant resolution*
- *snap to bars* is a post-processing tool to macro-adjust the events in your Patterns to fit inside of a bpm-aware amount of time

If 'linearization' and 'snapping to bars' are performed one after the other, unexpected results can occur. Changes are sequential – linearization will overwrite the original timing values, so the result of snapping to bars after linearization will be different than if you snapped to bars without linearization. For example:

```
bpm: 120
linear recording?: no
quant resolution: 1/16
```



Let's say we have 4 pad presses recorded with these durations:

```
1      0.47999095916748 sec
2      0.49550008773804 sec
3      0.33601188659668 sec
4      0.4467670917511 sec
total  1.7582700252533 sec
```

*linearizing* these will gently nudge each duration to the nearest *quant resolution* multiple of the current *bpm*. At 120 *bpm* and 1/16, this is a multiple of 0.125:

```
1      0.5 sec = 1/4 note
2      0.5 sec = 1/4 note
3      0.375 sec = dotted 1/8 note
4      0.5 sec = 1/4 note
total  1.875 sec
```

That dotted eighth might be unexpected or undesirable, even though it is a musical division of the current *bpm*. If we went with a *quant resolution* of 1/8, we'd get:

```
1      0.5 = 1/4 note
2      0.5 = 1/4 note
3      0.25 = 1/8 note
4      0.5 = 1/4 note
total  1.75 sec
```

Reducing *quant resolution* to 1/4, we'd get:

```
1      0.5 = 1/4 note
2      0.5 = 1/4 note
3      0.5 = 1/4 note
4      0.5 = 1/4 note
total  2.0 sec = 1 bar
```

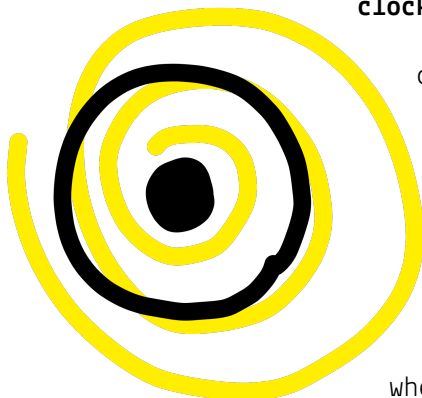
*nb. If linear recording? was set to yes, these changes would have happened automatically after recording the Pattern.*

Alternatively, we can use *snap to bars* to retain the relationships of each of the notes to each other, but stretch their values to equal a full bar of time. Returning to our original timing, here's what happens when we *snap to bars* (set to 1 bar):

```
1      0.47999095916748 sec -> 0.54598093839236 sec
2      0.49550008773804 sec -> 0.56362228852381 sec
3      0.33601188659668 sec -> 0.38220737630817 sec
4      0.4467670917511 sec -> 0.50818939677566 sec
total  1.7582700252533 sec -> 2.0 sec (= 1 bar!)
```

Notice that the individual pad presses do not line up as even divisions of the *bpm*. Instead, what we've done is retained the swing of the presses and stretched the total time of the Pattern so that it fits within a single bar of 120bpm. This approach is perhaps unexpected but is very helpful in the context of cheat codes – since you are often working within three time domains (your sample's bpm, the pace of your pad presses, and the global bpm), you can achieve compelling musical results by using *snap to bars* to lock down your "1" and letting the rest flow freely.

### clocking



cheat codes has an internal clock, which is controllable using *bpm* on the *clk* page. cheat codes can also be clocked externally, either by MIDI (24 PPQ) or by a rising-edge voltage pulse through a connected crow's second input. If you'd like to tap tempo, simply highlight *bpm* in the *clk* domain and tap K3.

Even when it's clocked externally, cheat codes never fully ignores the *bpm* parameter. *bpm* carries influence when linearizing and snapping to bars.

## World 2: Shaping Time

*nb. Please set up audio for your banks and pads ahead of this section. It might be best to just load a pre-recorded sample.*

These three sub-worlds will help you explore the possibilities of incorporating varied clock sources, linearization, snapping, and quantization into your cheat codes sessions. For the best results, reset changes between sub-worlds.

### 2.1: Internal Clock

- Set a *bpm* in the *clk* domain of [timing]
- Arm Pattern recording on grid (notice the Pattern pad flashes at the specified *bpm*)
- Play a pattern on the corresponding bank's pads
- End the Pattern recording and the Pattern will play + loop
- In the [timing] menu, use E1 to navigate to the corresponding *P* domain
- *linearize* the Pattern (notice the subtle adjustment to your timing)
- Use E1 to head back to the *clk* domain
- Choose a new *bpm*
- Use E1 to head back to the *P* domain
- *linearize* again (notice the timing between pad presses remains the same, but the pace of the presses matches the new *bpm*)

### 2.2: MIDI Clock

- Connect a MIDI clock source to norns
- In the *clk* domain of [timing], adjust your *clock source* to *midi*
- Use E1 to navigate to the *ALL* domain
- Use E2 to select *quantize pads?* and E3 to specify *yes*
- Set your MIDI clock source to 86 bpm (maybe set up a drum loop or clock tick to hear the tempo)
- Press a pad and notice that though the pad switches, it does not execute until it receives a MIDI clock tick
- Set *linear recording?* to *yes* (this will linearize your Pattern immediately after recording)
- Record a 2-bar Pattern and play it back (you should be able to get a pretty clean loop if you play close to tempo)
- In the corresponding *P* domain, set *snap to bars?* to 2 and hit K3 (this will snap your recording to a clean 2-bar loop, in case you had any timing kerfuffles)

### 2.3: *crow Clock*

- Connect *crow* to *norns*
- In the *clk* domain of [timing], adjust your *clock source* to *crow*
- Connect a CV clock pulse to *crow*'s second input
- Record a pattern (button mash a bit, independent of the clock)
- After ending the recording, the Pattern will play back at the CV clock rate
- While the Pattern is playing, navigate to the *clk* domain and set a new *bpm* (this will not affect the Pattern, which is being clocked through *crow*)
- Navigate to the corresponding *P* domain and *linearize* the Pattern to the *bpm*
- The Pattern itself will morph as it linearizes the original *bpm* to match the new *bpm*, but since the clock is external the changes will just "work"
- Use this to inject new life into old patterns

### Patterns: meta page

The bottom-right grid key toggles grid's display between our main performance page and a special Pattern meta page. Here, you can save up to 8 grid Patterns per bank and sequence them using a meta-sequencer similar to Kria.

This page is essentially three instances of this configuration (one for each bank):

```
[P] [P] [P] [P] [P] [P] [P] [P]
[c] [c] [c] [c] [c] [c] [c] [c]
[~] [~] [~] [~] [~] [~] [~] [~]
[~] [~] [~] [~] [~] [~] [~] [~]
[d] [d] [d] [d] [d] [d] [d] [d]
```



### **P: Pattern save/load slots**

If a bank has a Pattern recorded, that Pattern can be saved for long-term storage in any of the slots in this section's first row. Once a Pattern is saved, it can be erased from the play page and re-loaded from the meta page.

*To save a Pattern in a slot:*

- record a Pattern on the play page
- switch to the meta page
- press and hold any slot to save the corresponding Pattern there
- the slot's LED brightness will increase to show a Pattern is saved there

*To load a Pattern from a slot:*

- on the meta page, press any slot that has a saved Pattern to load that Pattern immediately
- Loading a saved Pattern slot will overwrite any currently unsaved Pattern on the play page

*To erase a slot:*

- while holding ALT, press and hold a Pattern slot that has saved data
- after one second, the Pattern slot will be erased

### **c: meta-sequence clock divider**

Each bank's meta-sequencer can be set to divide the clock (internal or external) by integers 1 through 8. Each click of the clock divider pushes the **d** (duration) counter.

### **~: the meta-sequencer**

Each bank's meta-sequencer can trigger up to 16 Pattern switching events.

*To assign a Pattern slot to a meta-sequencer step:*

- press and hold a meta-sequencer step
- while holding, select a Pattern slot from the section's top row
- after selecting, release all keys

*To clear a meta-sequencer step:*

- press and hold ALT
- press the meta-sequencer step you wish to clear

*nb. changes you make to a step's duration or Pattern loop state will remain*

*To see which Pattern slot a meta-sequencer step is assigned:*

- press and hold a meta-sequencer step and its corresponding Pattern slot will illuminate

### **d: step duration counter**

The step duration counter is driven by the meta-sequence clock divider. You'll notice that the default step duration is 4 clock ticks. When the step duration counter reaches its end, it will move the meta-sequencer to the next step.

*To set the duration of a step:*

- press and hold a meta-sequencer step
- while holding, select a new end point for the current step's duration counter
- hold ALT while adjusting the duration counter to adjust all steps at once

*To see a step's duration:*

- press and hold a meta-sequencer step
- while holding, you'll see a change in LED levels to indicate the selected step's duration counter end point

## Bottom Row

Along the bottom of the meta page is a series of handy controls, some currently implemented and some planned:

[A] [t1] [t2] [t3] - - [l] [m]

We've already covered ALT and the meta page toggle.

### toggles on/off

t1, t2, and t3 act as multi-function toggles.

*meta-sequencer toggles:*

- when no other key is held, they toggle their respective meta-sequences on/off (which will pause the respective meta sequencer)
- hold ALT and press any of these toggles to reset the meta-sequencer counters and steps to the beginning of the loop

*per-step Pattern loop toggles:*

- press and hold one of the steps and you'll notice the toggle indicators change intensity
- if the key is illuminated, this means that the currently held step is set to loop its pattern
- press the toggle to turn this looping behavior on/off

### meta-sequence loop points

Hold the *l* button in the bottom row to perform loop mods. While *l* is held, your first press will always establish the start point for the meta-sequence's loop and your second press will always establish the end point. If you set an end point that's before the start point, you will get a single-step loop.

## World 3: Remember

- Record some audio into the Live buffers
- Set Zilchmo's style to **rad sauce**
- On the play page, record a Pattern of pad changes and Zilchmo gestures
- After you record a Pattern, switch to the meta page and save this pattern in an empty slot
- Switch back to the play page and erase the Pattern (don't worry, it's saved in the slot on the meta page)
- Record a new Pattern of pads + gestures with the same bank as before
- Save this pattern in an empty slot
- Repeat until you have four pattern slots saved
- On the meta page, switch between your saved Patterns (you'll hear them start playing)
- Figure out an order for your Patterns and enter some as steps in the meta-sequencer
- Use the *l* mod to change loop points

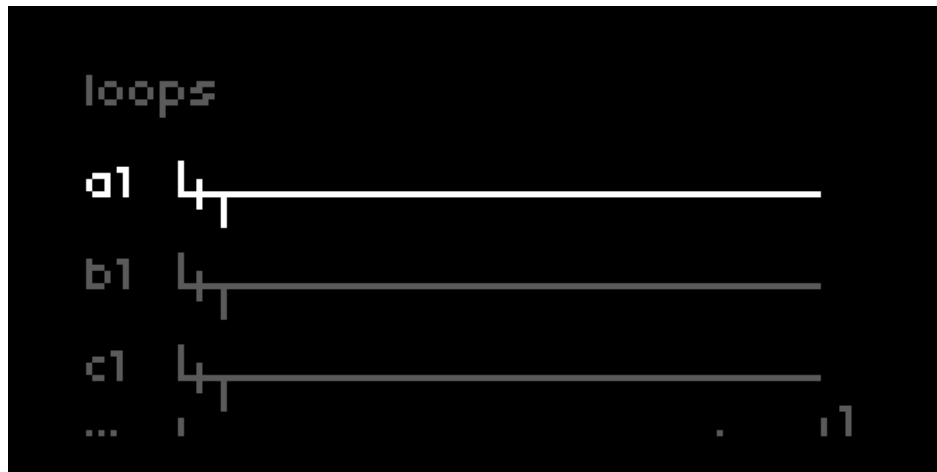
cheat codes: buffers + [loops]

---

## **buffers + [loops]**

As mentioned earlier, cheat codes has 6 buffers: 3 loopers for live recording + 3 static clips to load pre-recorded samples. To make things simple, these buffers are capped at 8 seconds of audio each.

When you load a fresh session of cheat codes, Live buffer 1 is already recording in a loop. To see the record head's position, navigate to the [loops] menu:



The record head is the small dot at the bottom of the screen.

Using K3, you can navigate down to the Live buffer's loop and adjust its start and end points the same way you adjust the other loops':

- E2: start point
- E3: end point
- E1: move loop window

## **PARAMS**

In the PARAMS menu, you'll find a number of "set it and forget it" parameters that don't require an in-script shortcut:

### **feedback**

By default, the record head has a 25% feedback setting - this means that when it passes over previously recorded audio, the previous audio's level will reduce by 75%. Adjust to taste.

### **loop encoder resolution**

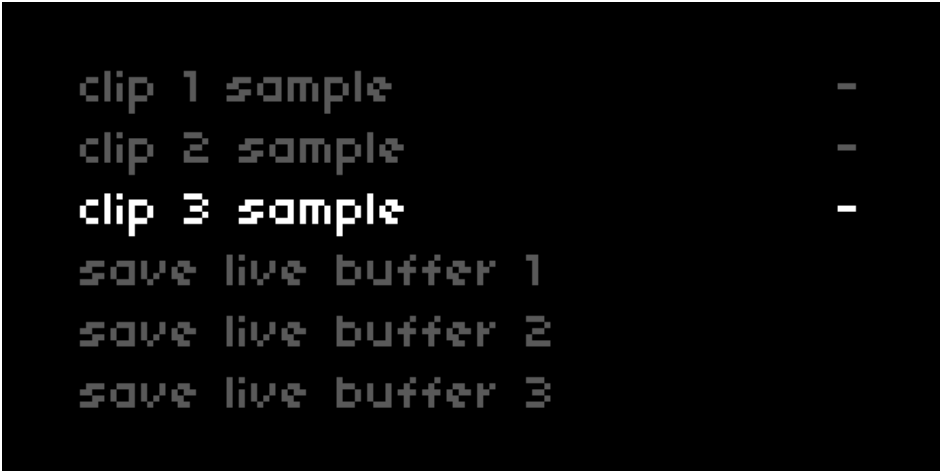
By default, the encoders on [loops] will scroll at 100ms intervals. This can be changed to 10ms intervals by adjusting this parameter to 0.01

cheat codes: buffers + [loops]

---

### clip [x] sample

Load a sample into one of the three static clips or save the audio in the Live buffers as a clip:



```
clip 1 sample -
clip 2 sample -
clip 3 sample -
save live buffer 1
save live buffer 2
save live buffer 3
```

*nb. norns + cheat codes expect .wav files @ 48khz. You can load files with other sample rates, but their pitch will be wonky.*

### save live buffer [x]

If you've recorded something into the Live buffer of cheat codes and wish to save this audio as a Clip for future manipulation, highlight this parameter and hit K3 to write the audio into a special folder, located at `we/dust/audio/cc_saved_samples`. The file will be named as `cc_YYMMDD_TIME-buffBUFFERNUMBER.wav`

## World 4: Gaming the System

- Load a sample into clip 1 and use grid to switch a bank to reference it
- Set Pattern recording to **rad sauce**
- Set the pads to **loop**
- Arm Pattern recording, but don't touch any pads yet (*nb. Pattern recording doesn't actually start until a pad is pressed*)
- Navigate to the [loops] page and use E1 to move the loop window, then press the corresponding pad
- Move the loop window and press the same pad again
- Again (and as many times as you'd like – same pad, but using E1 to change loop windows)
- Play the pattern back

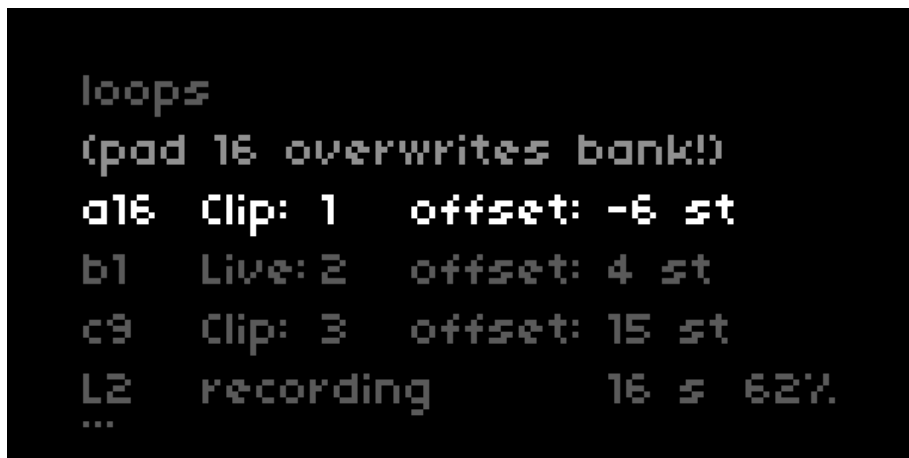
You'll notice that even though we didn't change what pad we were hitting, the loop window's loop points were recorded with each press. This is because **rad sauce** mode records not only which pad was pressed, but also its parameters' states.



Though the grid interface for cheat codes provides the most comprehensive control over its features, gridless play is totally possible and may provide a nice challenge to shake up your muscle memory.

Through the [—] menus, it's easy enough to make choices and changes, but one facet of play had previously been locked to the gridless: selecting, swapping, and manipulating the six buffers.

In [loops], hold K1 to access an alt menu:



Use K3 to move down the screen.

For the first three lines:

- E1: switch between pads, 1-16, across each bank (*nb. changes made to pad 16 will map to all pads in the bank, as an approximation of grid's handy ALT functions*)
- E2: switch which buffer the selected pad references, Live (1-3) then Clip (1-3)
- E3: add a semitone offset to the selected pad

For the last line:

- E1: switch between the three Live buffers, to record unique audio into each
- E2: enable or disable recording into the selected Live buffer
- E3: extend the maximum Live buffer recording time

**A note about extending the maximum Live buffer recording time:**

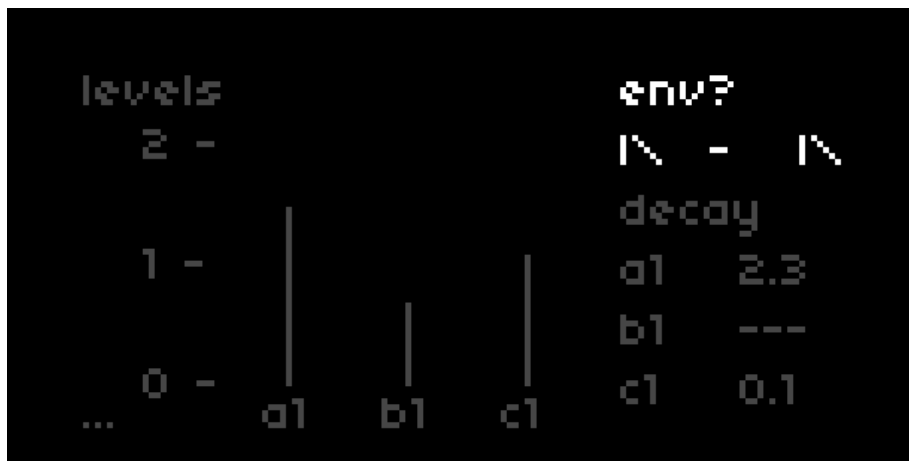
This control slows the rate of the Live buffer record head, so it's still only traveling along an 8 second buffer, but it's doing it at half (or a quarter) of the normal rate. This introduces a nice sprinkling of grit but also an immediate chipmunk effect. For "normal" play with a slower Live buffer, you can use the offset controls to subtract octaves: a 16s Live buffer corresponds to a -12st offset, 32s corresponds to -24st.

cheat codes: [levels]

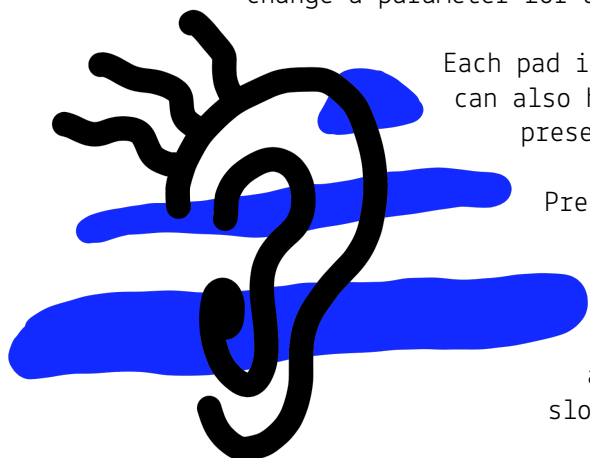
---

### [levels]: volume and envelopes

When you navigate to the [levels] menu, you'll find controls for volume and envelopes:



By default, these controls only affect the currently selected pad. Use ALT (or K1) to change a parameter for all the pads in the bank.



Each pad in each bank can have its own volume, but each pad can also have its own decaying envelope to shape its presence in your mix.

Press K3 to navigate to the **env?** submenu and use the encoders to enable/disable the envelope for each bank's current pad. Press K3 again and use the encoders to choose a decay length for each bank's current pad. Envelopes can be as short as 100ms for snappy perc, or as long as 60sec for slow fades.

Envelopes work with both 1-shot and looping pads:

- Try setting a pad to a super-short loop, double its rate, enable the pad's envelope, tune it to a 0.9 sec decay, and trigger the pad
- Try setting a pad to 1-shot with a long loop, enable the pad's envelope, tune it a 2.3 sec decay, and trigger the pad

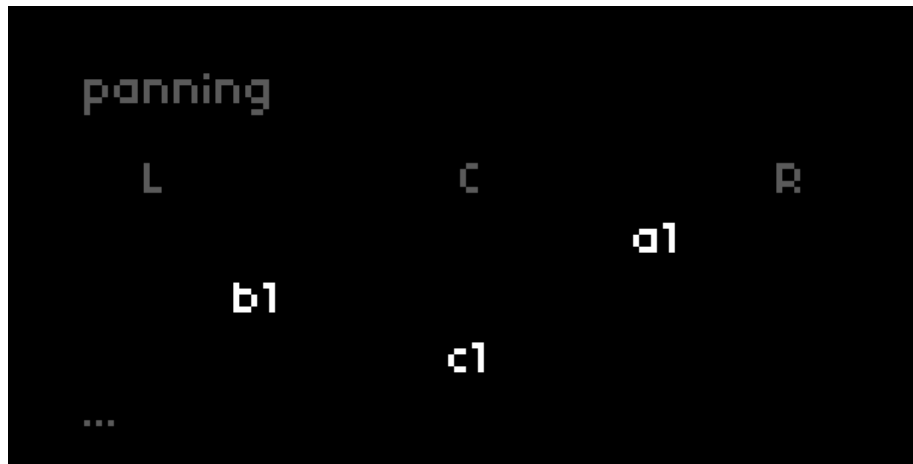
**ALT** is particularly helpful on this page, as it will allow you to fade in an entire bank's worth of pads, or evenly adjust bank-wide envelope decay. When Patterns are in full-swing, this can be very handy.

cheat codes: [panning]

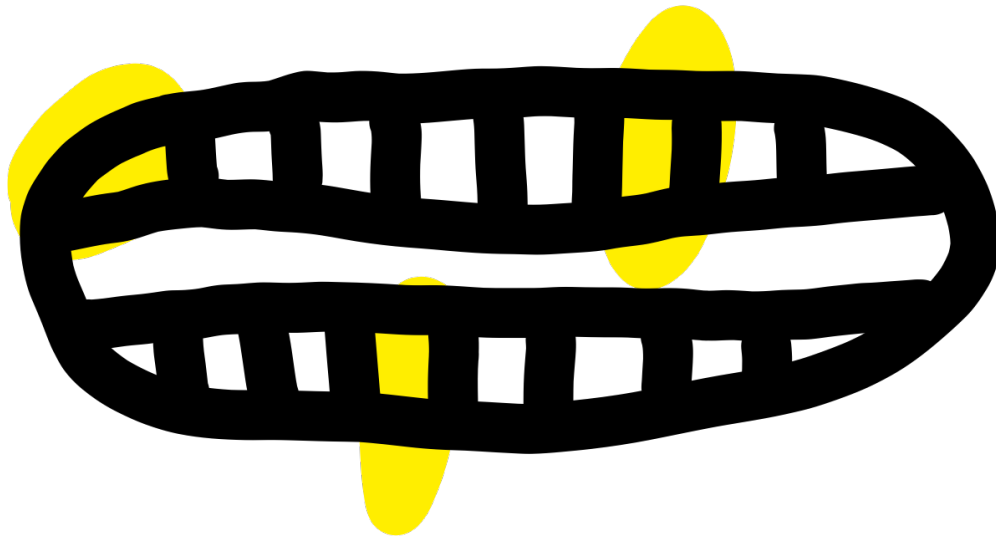
---

**[panning]: movement in the field**

To help layer your worlds, utilize per-pad and bank-wide panning:



Each encoder controls the panning position of the currently-selected pad in each bank. Hold K1 to adjust all of the pads in the bank uniformly and from their current position – eg. if pad 1 is set slightly L and pad 2 is set C, a K1-held clockwise encoder adjustment would bring pad 1 to C and pad 2 slightly R.

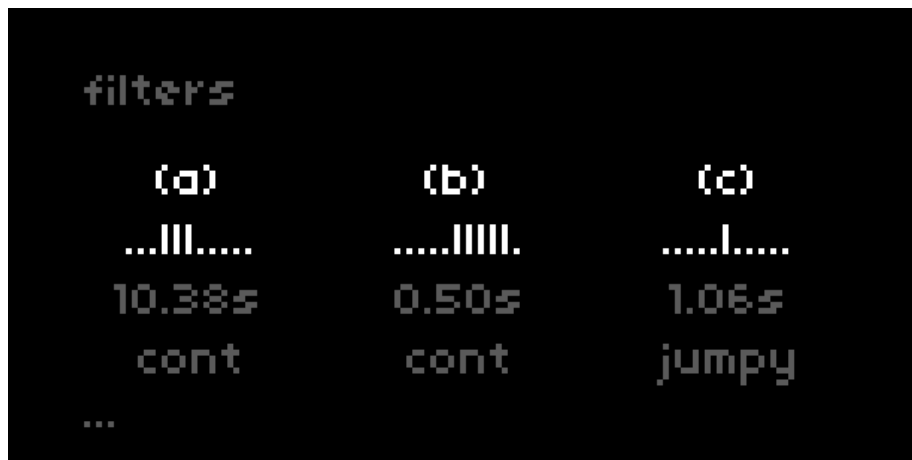


cheat codes: [filters]

---

### [filters]: timbral shaping

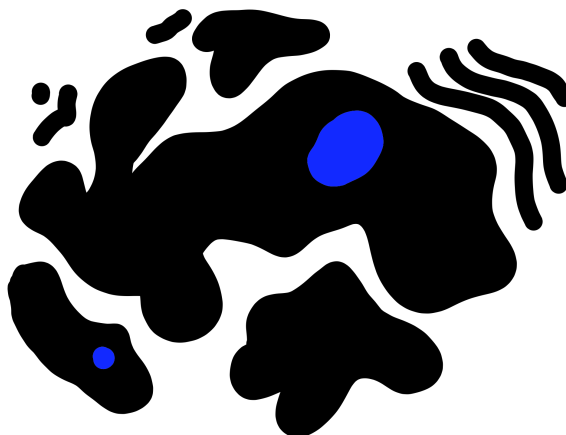
When you navigate to the [filters] menu, you'll find a set of streamlined LP/HP dj-style filters:



By default, these controls affect the entire bank of pads at once. This makes it easy transform the frequency content of a playing Pattern. Hold ALT to change the filter parameters of a currently selected pad.

Use K3 to switch between:

- *filter tilt*: CCW for LP, CW for HP, with a neutral middle
- *ease timing*: if two pads have different *tilts*, *ease timing* defines the length of filter sweep between the two values
- *transition character*: if the *tilt* is *easing* between two values and a new pad is pressed, should the *ease* simply pivot toward the new value (*cont*) or should it jump to its original destination and then *ease* toward the new value (*jumpy*)?



If you have individual *tilt* values for each pad, performing a global adjustment will add or subtract from the individual values accordingly. There is no limit, however – so it's entirely possible to spin an encoder for a while and min/max-out your individual values. This may be desirable in some cases, but it might be unexpected in others.

cheat codes: [delay]

---

### [delay]: in stereo

cheat codes comes with its own super-utilitarian stereo delay:

delay			
/1 1/3	rate	x1 1/3	
32	feed	92	
7787	cutoff	7202	
0.36	q	0.09	
...	level	0.71	

Use E1 to navigate the rows, E2 and E3 to change left and right, respectively.

- *rate*: adjust the multiplier/divisor for unique delay rhythms
- *feed*: feedback control
- *cutoff*: cutoff param for a LP filter
- *q*: lower values = higher resonance
- *level*: how present should each channel be in the mix?

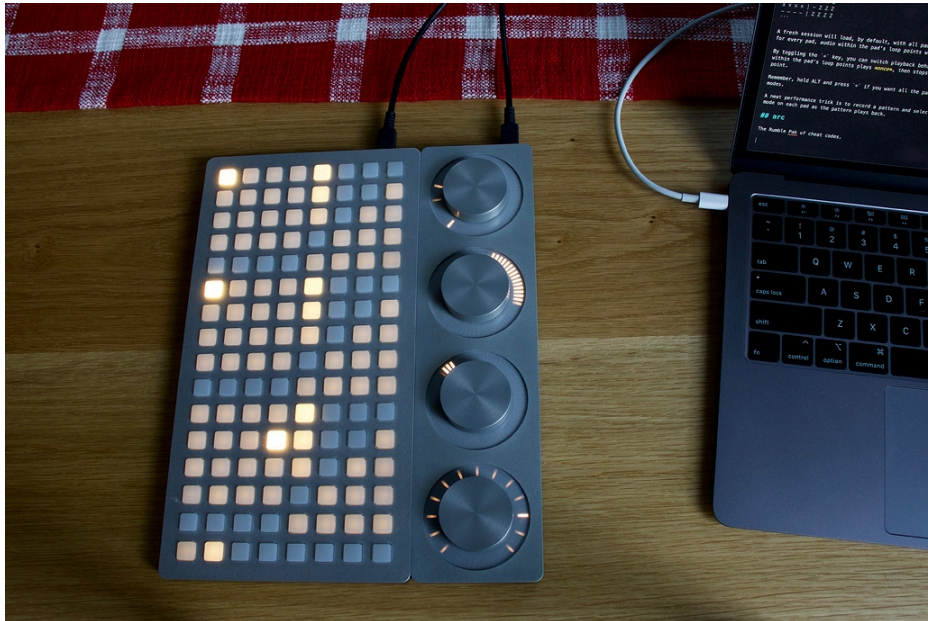
By default, all banks feed into the delay lines. You can turn bank sends on/off in the system PARAMS, under *delay L/R: [x] send*. Map a MIDI fader to quickly toggle 'em!



cheat codes: arc

---

The Rumble Pak of cheat codes.



## parameters

Using the four keys underneath each bank, you can toggle between arc control over the following parameters:

- loop window
- loop start
- loop end
- filter cutoff

Each arc encoder linearly corresponds to each bank; 1:a, 2:b, 3:c. The fourth arc encoder controls both the Left and Right delay rates. By default, it displays and controls the Left channel - hold ALT to display and control the Right channel.

## arc patterns

There are also three pattern recorders just for arc!  
Remember those p's in right of the grid's bottom row?

A 1 2 3 | p p p m

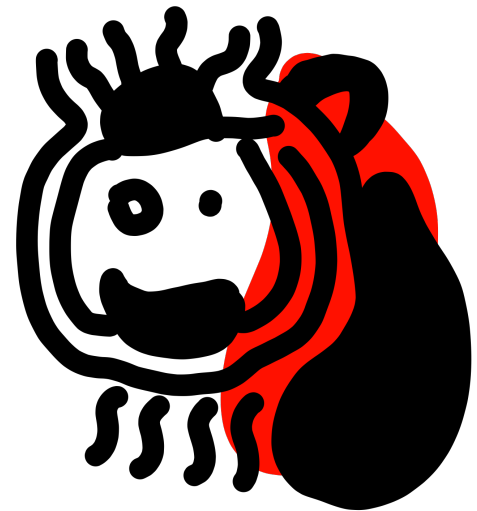
Use these to control pattern recorders for each arc encoder. These will record both arc encoder turns as well as arc parameter changes. Record + playback follows the exact same process as the grid patterns. Similarly, you can use ALT to clear the arc patterns.

*nb. At this time, there is no long term storage for arc patterns.*

Through the [timing] page, you can customize how crow interacts with cheat codes:

```
timing
```

clk	P1	P2	P3	ALL
bpm			110	
clk source			internal	
send crow clk?			no	
...				



There are two primary modes, which can be mixed and matched:

- *clk*: send steady divisions of the clock (whether the clock is internal, MIDI, or from crow itself)
- *pads*: any time a bank's pad is pressed, send a pulse to the corresponding crow output (very fun, especially once you start sequencing Patterns)

### clk

If you wish to send divisions of the clock through crow's outputs, you'll first need to enable *send crow clk?* in the *clk* [timing] domain. As soon as you enable this mode, crow output 4 will emit a steady pulse. crow output 4 has a steady 1/1 division.

Now, enter any of the *P*(attern) domains to specify the division for the other crow outputs.

If *crow output* is already set to *pads*, use E3 to change to *clk*. Then, use E2 to navigate over to the divisor selector and choose a division 1-16:

```
timing
```

clk	P1	P2	P3	ALL
linearize			[K3]	
snap to bars			1 [+K3]	
crow output			clk (/1)	
...				

```
timing
```

clk	P1	P2	P3	ALL
linearize			[K3]	
snap to bars			1 [+K3]	
crow output			clk (/3)	
...				

### pads

In this mode, presses on a bank's pads will send a quick trigger signal through the corresponding crow output. If a Pattern is playing with this mode enabled, playback will also fire off triggers. Combine with other [timing] actions for unique rhythms.



## **collections**

Everything you change in a cheat codes session – banks, pads, Patterns, meta-sequences, params, audio in the buffers (Live and Clip), etc etc – can be wrapped up and saved into a single collection.

Navigate to the system PARAMS menu for cheat codes and you'll see a **collection** selector up top. You can save up to 100 collections, so don't feel bashful – if you land on a nice thing, feel free to save it and revisit it later.

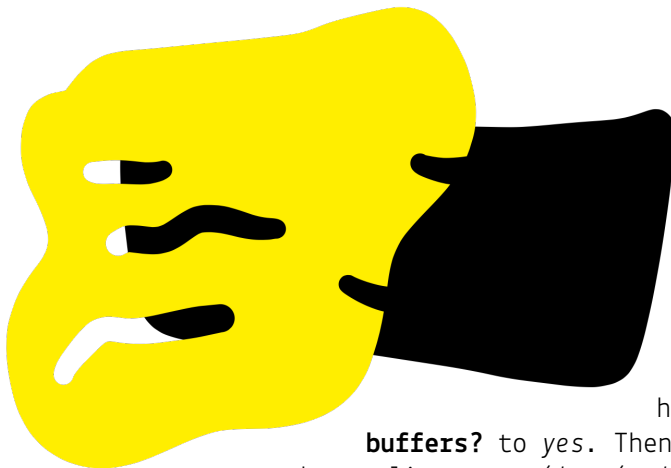
Once you decide on a number, highlight *save* and hit K3.

When you decide you want to re-open a collection, simply select the number that corresponds, highlight *load* and hit K3.

If you make changes to a loaded collection, you'll need to *save* the collection again in order for cheat codes to commit the changes. Otherwise, you can just reload the collection and your session will restore to its previous state.

If you load a fresh cheat codes session, nothing will be reinstated from your previous session. You always have to *load* a collection.

As you add to your collections, please be careful when saving – there is no undo!



### **how does audio save?**

When you load a collection, cheat codes will always restore the pre-recorded audio in each of the Clip slots (provided that you haven't moved/deleted them). If you've recorded something into the Live buffer of cheat codes and wish to restore this audio into the Live buffers whenever you load this collection in the future, head to PARAMS and set **collect Live**

**buffers?** to yes. Then, save your collection. This will save three clips at `we/dust/audio/cc_collection-samples/`  
`COLLECTIONNUMBER` as `cc_COLLECTIONNUMBER_BUFFERNUMBER.wav`. When you load that collection in the future, these samples will be loaded into the Live buffers and Live recording will be disabled (so the record head doesn't overwrite your restored audio).

Additionally, you can simply save whatever audio is in your Live buffers without having to save a collection (let's say you don't need the other settings to persist, but you dig what you sampled). Head to PARAMS, highlight **save live buffer [x]** and press K3. This will write the audio into a folder at `we/dust/audio/cc_saved_samples`. The file will be named as `cc_YYMMDD_TIME-buff[x].wav`