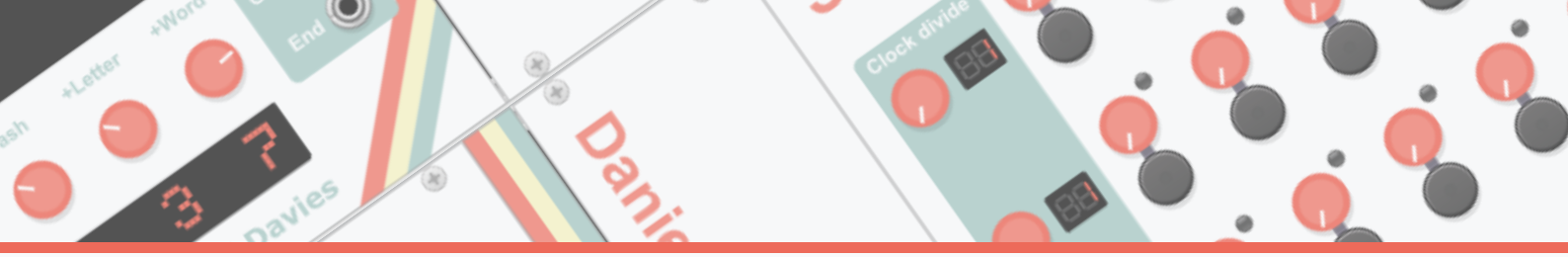
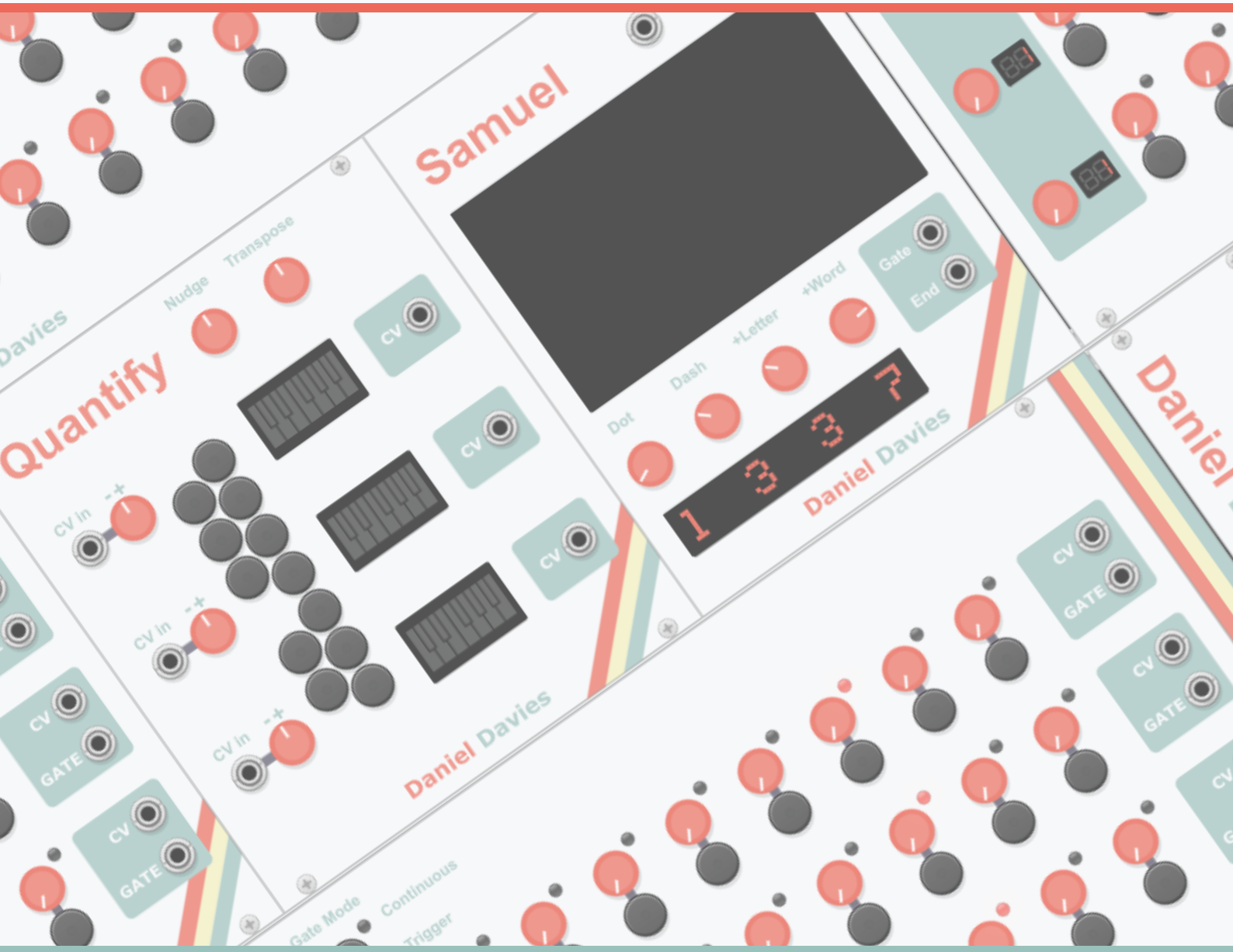


Daniel Davies

Modules for VCV Rack
Manual



Samuel

Morse code based rhythm generator - 20HP

From words, to rhythms. Named after Samuel Morse, the creator of Morse code. Samuel takes text input and constructs natural sounding rhythmic sequences using Morse Code.

Special thanks to Paul Gatt, for both the initial idea, and testing of Samuel

Panel

A. Clock input

B. Reset input

C. Message input screen

D. Dot length control

E. Dash length control

F. New letter length control

G. New Word length control

H. Gate output

I. End of sequence output

J. Length indicator screen

How it works

Samuel uses international morse code:

A .-	G --.	M --	S ...	Y -.--	4 ....-
B -...	H	N -.	T -	Z --..	5 .....
C -.-.	I ..	O ---	U ..-	0 -----	6 -....
D -..	J .---	P .--.	V ...-	1 .----	7 --...
E .	K -.-	Q --.-	W .--	2 ..---	8 ---..
F ..-.	L .-..	R .-.	X -..-	3 ...--	9 ----.

- The length of a dot is one unit
- A dash is three units
- The space between parts of the same letter is one unit
- The space between letters is three units
- The space between words is seven units

Explanation:

Samuel requires 2 things before it will do anything useful:
1. A clock input
2. Some text input

To provide text input to Samuel, click anywhere within the text input screen **(C)**, you can then type using your computer's keyboard (until you click anywhere outside of the text input screen)

Note: currently only letters A-Z and numbers 0-9 are supported

Samuel treats one unit of time as the time between two clock inputs recieved via the clock input **(A)** because of this, fast clocks tend to work best.

Once you have entered some text, and hooked up the clock input **(A)** to a clock source you can then use the gate output **(H)** to trigger drums, envelopes, Nuclear Armageddon etc.

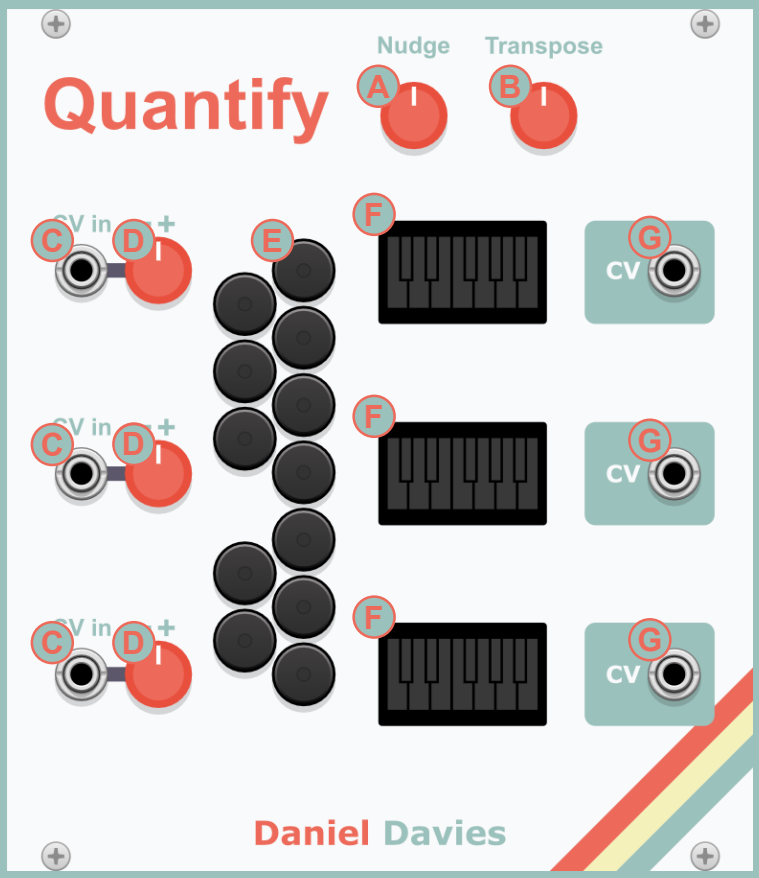
Knobs **(D - G)** can be used to vary the length of dots, dashes, new letters, and new words. Altering these values will change the characteristics of the resulting rhythms.

Quantify

3x Quantizer with nudge and transpose controls - 23HP

Designed to work seamlessly with my Sequel range of sequencer modules, this quantizer can be used to build 3 part melodies within a scale/chord of your choosing. Quantify gives you extra fine control over the range of your melodies in the form of attenuverted inputs and the built in *nudge* control allows you to shift your melody within your scale.

Panel



A. Nudge control

B. Transpose control

C. CV inputs

D. Input value attenuverters

E. Note select buttons (notes C - B)

F. Current note indicator

G. CV outputs

How it works

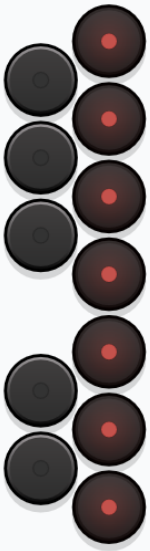
Explanation

Quantify works by rounding your CV inputs to the nearest available note in your selected scale.

To select a scale/chord, click on the note select buttons **(E)** that relate to your intended scale. You can think of the note select buttons as a single octave piano keyboard that has been rotated 90 degrees.

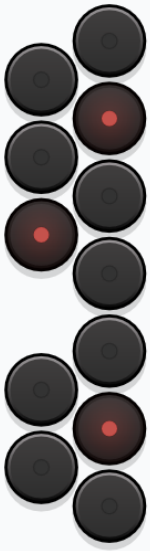
Note: when no notes are selected, Quantify will round all CV inputs to the nearest C.

C Major/A Minor scale



C Major chord

D Major chord



Quantify will display the note that is currently being output on note displays **(F)**

You can change the range of values Quantify will output per-row by altering the attenuvert knobs **(D)**. while these knobs are set to the 12 o'clock position, only one note will play for that row.

The nudge knob **(A)** will shift your CV outputs up or down within the scale, while the transpose knob **(B)** will shift the CV output values up/down by semitone values by a maximum of one octave in either direction.

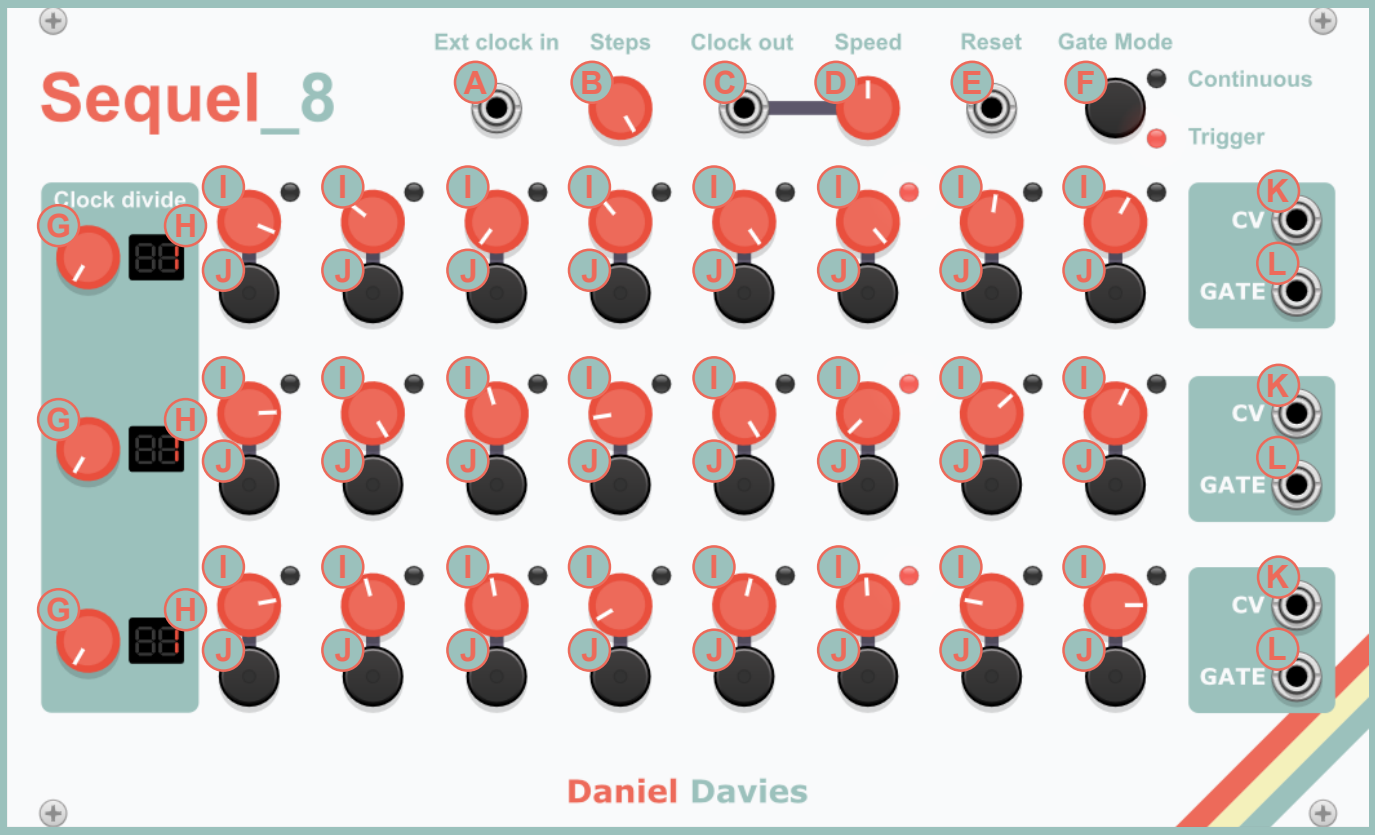
Note: currently the note screens **(F)** do not show the transposed output. This is likely to change in a future update.

Sequel 8 & Sequel 16

Three row 8/16 step sequencer - 48/73 HP

Powerhouse step sequencers with built in clock divide functionality (Sequel 8 and Sequel 16 are identical aside from number of steps). Each row of Sequel has both a gate output and a CV output, this allows Sequel to be used as a powerful drum sequencer, melody generator or controller.

Panel



A. External clock input

B. Step count knob

C. Clock output

D. Speed control knob

E. Reset input

F. Gate mode select

G. Clock divide knobs

H. Clock divide displays

I. CV control knobs

J. Gate on/off buttons

K. CV outputs

L. Gate outputs

How it works

Explanation

Sequel is a CV & Trigger sequencer, inspired by hardware analogue sequencer modules.

Each step of each of Sequel's 3 rows has a CV Knob **(I)**, a trigger button with an led indicator **(J)**, and an active step led. The voltage outputted by each row's CV output **(K)** is equal to the value of the CV knob **(I)** for the currently active step. Similarly the gate output **(L)** is controlled by switching on or off the gate button **(J)** for the currently active step.

There are two methods for controlling the speed of Sequel. You can use the internal clock controlled by the speed knob **(D)**, or you can feed an external clock source into Sequel via the external clock in port **(A)**.

Clock Divide:

One of the most powerful features of Sequel is it's per-row clock divide functionality.

Set a clock divide value for each row using the clock divide knobs **(G)**, the value is indicated via the clock divide displays **(H)**. The clock divide value determines how many clock inputs are needed before that row will progress to a new step. This means that Sequel's rows are able to become out of sync with eachother, allowing for the creation of interesting polyrhythms.

Gate Modes

Sequel is capable of two different gate modes. The active gate mode is controlled by the gate mode select button **(F)**:

Trigger mode: Gates output 10V for a duration of 1ms.

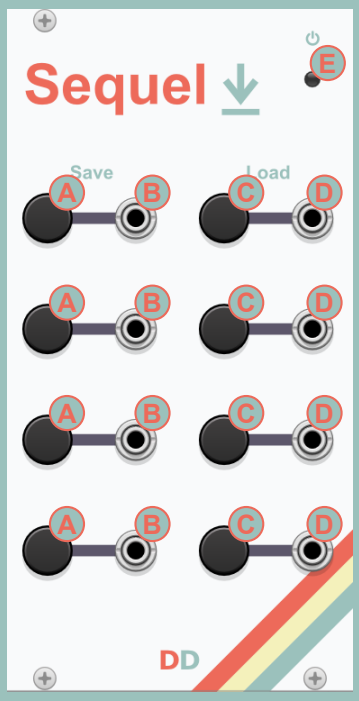
Continuous mode: Gates output 10v for as long as gate buttons **(J)** are toggled on.

Sequel Save

Save state expander module for Sequel 8 & 16 - 13 HP

Get even more out of Sequel 8 & 16 with the Sequel Save module. Save the state of Sequel to one of 4 slots and then restore that setting via CV or button controls..

Panel



A. Save buttons

B. Save trigger inputs

C. Load buttons

D. Load trigger inputs

E. Active LED

Explanation

To active Sequel Save, place it directly to the right of Sequel 8 or 16, the Active LED **(E)** will light when positioned correctly.

Sequel Save has 4 slots that can be saved to, using either the Save Buttons **(A)** or the Save trigger inputs **(B)**. Upon saving, the state of all the panel parameters of Sequel are remembered.

Use either the Load Buttons **(C)** or the Load trigger inputs **(D)** to load a previously saved state.

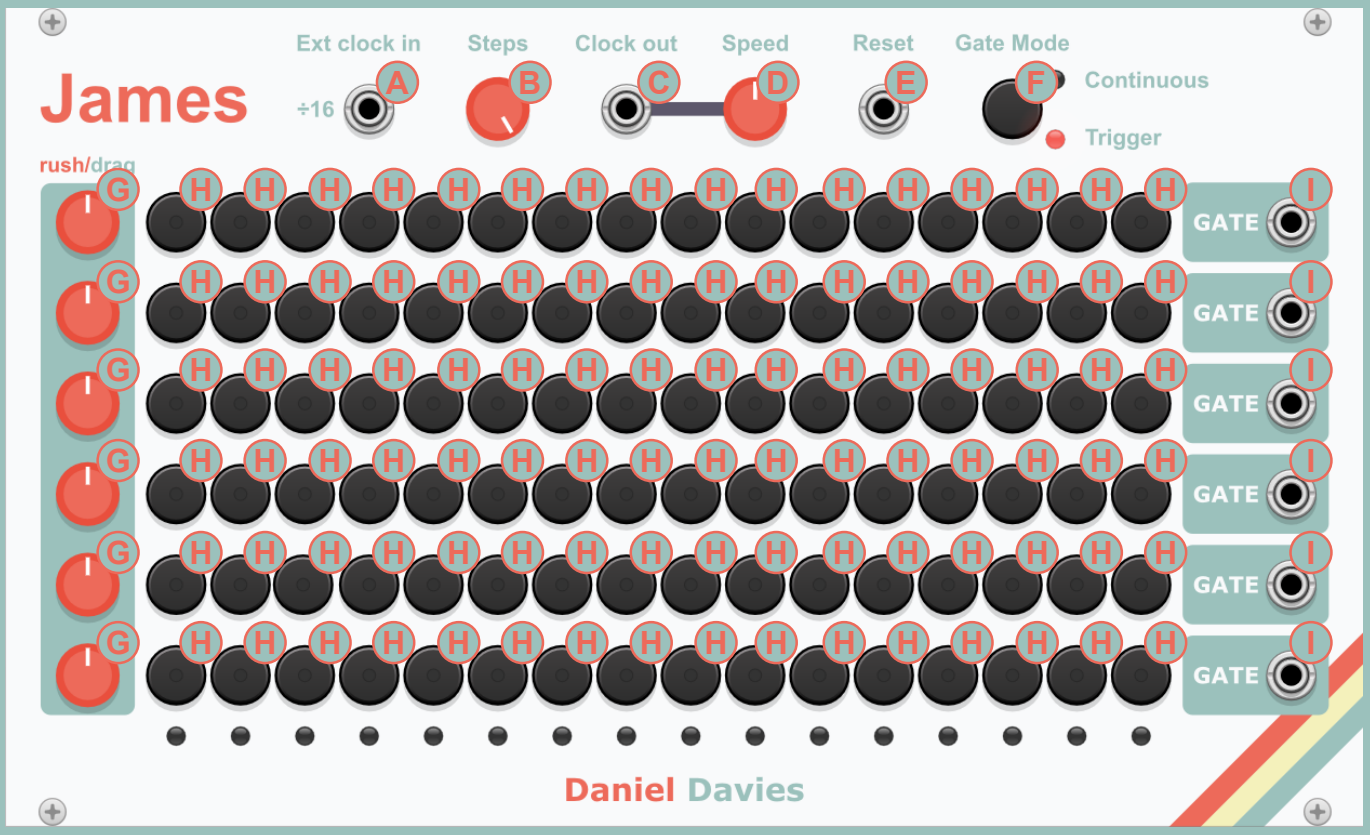
James

Six row 16 trigger sequencer - 42 HP

Inspired by the ryhtmic techniques of legendary producer and songwriter James Dilla (J Dilla). James is a 6 row drum sequencer with per-row rush/drag controls.

Special thanks to Paul Gatt, for both the initial idea, and testing of James

Panel



A. External clock input

B. Step count knob

C. Clock output

D. Speed control knob

E. Reset input

F. Gate mode select

G. Rush/Drag knobs

H. Gate on/off buttons

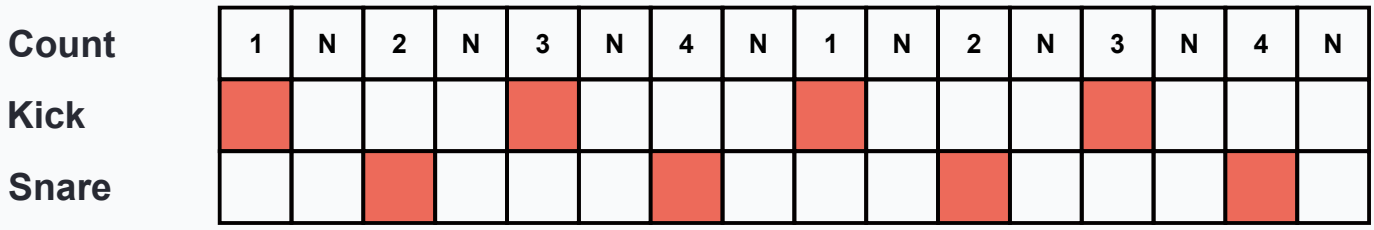
I. Gate outputs

How it works

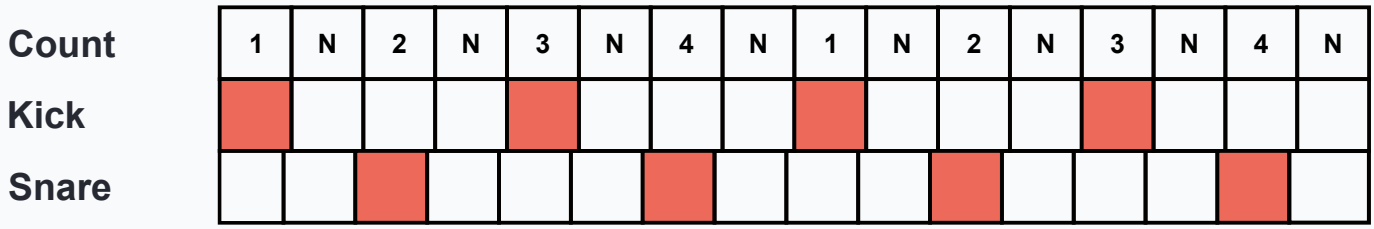
Explanation - Concept

James is a unique step sequencer that allows each of it's 6 rows to be rushed or dragged independently of the other rows. See the below diagram for an explanation of how the rush/drag feature of James works.

No rush/drag:



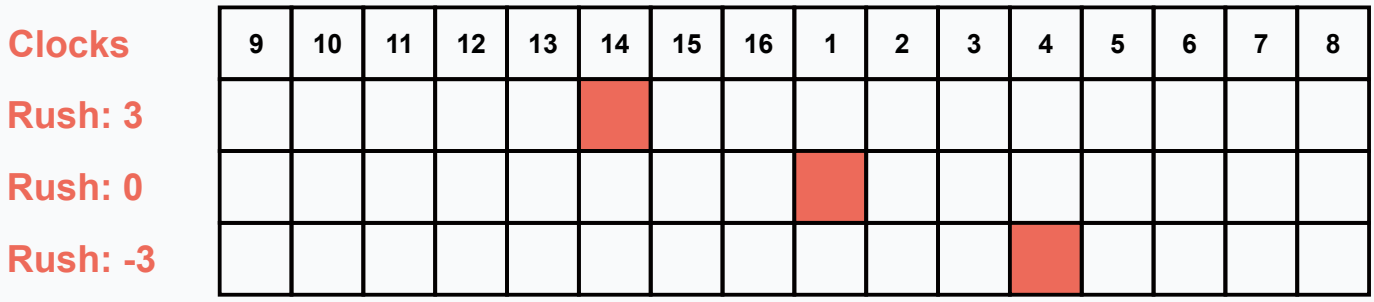
Rushed snare:



By introducing rush/drag for specific elements of a beat, you can create conflict with the straight beat.

Explanation - James

James works by dividing it's clock input **(A)** by 16, and then using the rush/drag knobs **(G)** to rush/drag each element by up to 15 clocks forward or backwards. For every 16 clock inputs sent to James, the currently active step will advance by 1.

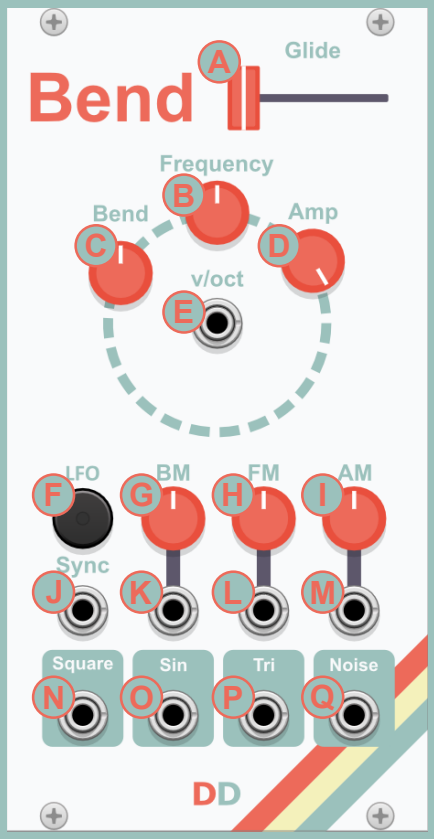


Bend

Phase modulation oscilator - 13 HP

Wavetable oscilator with a twist. Bend is a polyphonic VCO and LFO, featuring phase modulation, and glide controls. A labour of love, this is my first sound generating module.

Panel



A. Glide time slider

B. Frequency knob

C. Bend amount knob

D. Amp amount knob

E. V/Oct input

F. LFO mode toggle switch

G. Bend mod knob

H. Frequency mod knob

I. Amp mod knob

J. Sync input

K. Bend mod input

L. Frequency mod input

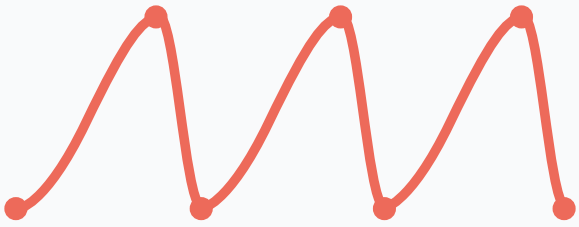
M. Amp mod input

N. Square wave output

O. Sin wave output

P. Triangle wave output

Q. Noise output



How it works

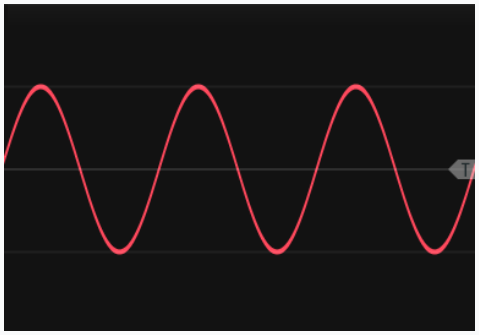
Bend uses phase modulation:

Phase modulation is a complex topic that has been explained in great depth elsewhere. Rather than try and detail everything here, if you're new to the subject I reccomend taking a look at this youtube video by Groovy DSP:

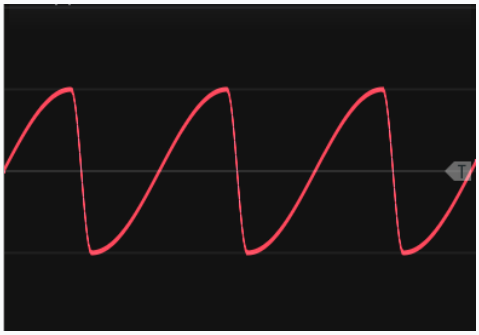
https://www.youtube.com/watch?v=EW12RYc7QRA

If you're not too interested in what phase modulation is and you simply want to know what it can do in the context of Bend, here is a very brief synopsis. As you apply phase modulation (bend) to a sin wave or a triangle wave, the resulting wave becomes closer and closer to a saw wave. This can be used to create a similar sounding effect to a low pass filter:

Bend: 0.00

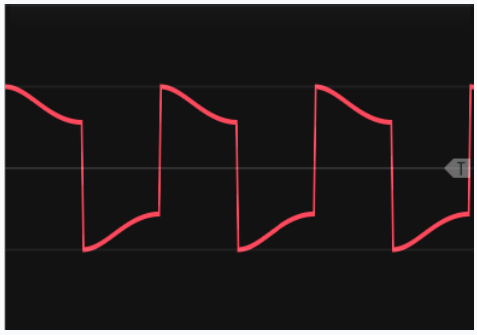


Bend: 0.8



Applying phase modulation to a square wave is akin to pulse width modulation:

Bend: 0.00



Bend: 0.98

Additional features

Glide:

The glide slider **(A)** controls portamento time when control voltages are sent to the v/oct input **(E)**. This smooths the transition between note changes, rather than being instantaneous.

Noise:

The noise output **(Q)** outputs white noise which can be used to add texture to a voice or, when paired with a sample & hold module, to generate random/ unpredictable control voltages.

LFO Mode:

LFO Mode is enabled by toggling the LFO mode switch **(F).** When enabled, the fequency ranges of the square/sin/tri **(N, O, P)** wave outputs are drastically reduced, allowing frequencies well below the audible range. This is turns Bend into a flexible modulation source.

Note: When LFO mode is enabled, the glide functionality of Bend is disabled.

Sync:

Bend features a hard sync input **(J)**. Try hooking the square wave output of another oscillator to this input and varying the two oscillators frequencies independently of eachother for some interesting sonic cababilities.

Bend, Frequency, and Amp modulation:

Bend features 3 modulation inputs:

Bend: by modulating the bend amount using the BM input **(K)** you can simulate a filter sweep effect on the tri/sin wave outputs, and PWM on the square output.

Frequency: Bend features the classic frequency modulation that is commonplace on most oscillators via the FM input **(L)**.

Amplitude: Modulating the amplitude of a wave can drastically alter the characteristics of the resulting sound created by an oscillator. This is why I decided to include this parameter which is often missing from other oscillator modules. Try hooking another VCO or LFO to the AM input **(Q)** For some very interesting results.