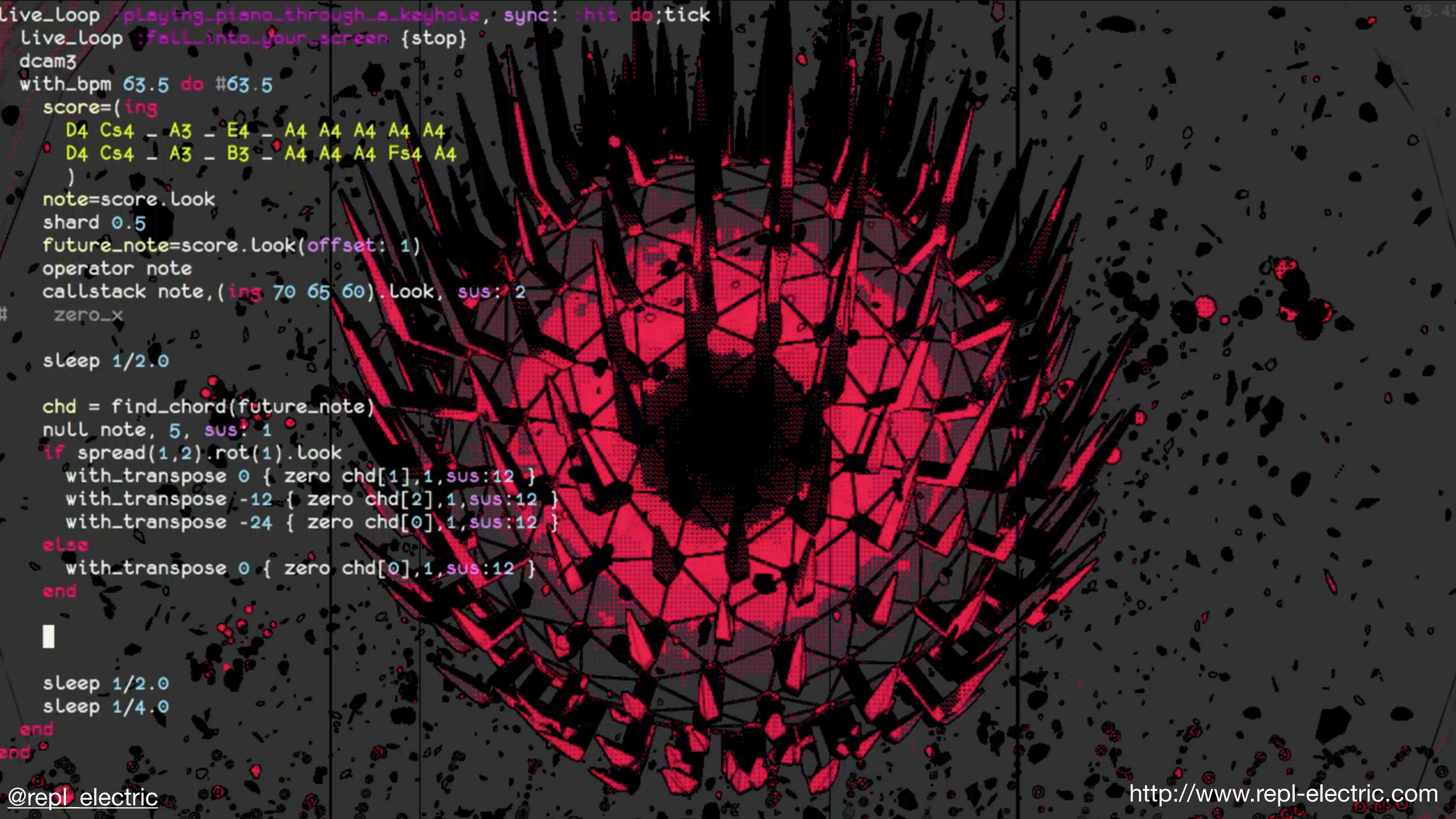
Crafting Instruments In Code

@josephwilk

https://en.wikipedia.org/wiki/Silver_Apples


live_loop :playing_piano_through_a_keyhole, sync: :hit do;tick
live_loop :fall_into_your_screen {stop}
dcam3
with_bpm 63.5 do #63.5
score=(ing
D4 Cs4 _ A3 _ E4 _ A4 A4 A4 A4 A4
D4 Cs4 _ A3 _ B3 _ A4 A4 A4 Fs4 A4
)
note=score.look
shard 0.5
future_note=score.look(offset: 1)
operator note
callstack note,(ing 70 65 60).look, sus: 2
zero_x
sleep 1/2.0
chd = find_chord(future_note)
null note, 5, sus: 1
if spread(1,2).rot(1).look
with_transpose 0 { zero chd[1],1,sus:12 }
with_transpose -12 { zero chd[2],1,sus:12 }
with_transpose -24 { zero chd[0],1,sus:12 }
else
with_transpose 0 { zero chd[0],1,sus:12 }
end
sleep 1/2.0
sleep 1/4.0
end
end
@repl_electric
http://www.repl-electric.com

@repl_electric

http://www.repl-electric.com


}
with_transpose 12.{cpu2 score[2], 60, sus: 2}
whitespace score[-1],1, sus: 2
pf= 1.8
i=(method :operator)
at{
i.call(score[0], 60*pf, sus: 4)
sleep 0.5
i.call(score[1], 30*pf, sus: 4)
sleep 0.25
i.call(score[2], 55*pf, sus: 4)
sleep 0.25
i.call(score[3], 70*pf, sus: 8)
sleep 0.25
i.call(score[4], 30*pf, sus: 8)
sleep 0.25
i.call(score[5], 50*pf, sus: 8)
sleep 0.25
cp
electric
alivecol 0.2
kick_machine K1, accent: 1.5,
if score.length > 1
at{
excavation root(score),100, sus: 2, atk: 0.01
@repl_electric
http://www.repl-electric.com

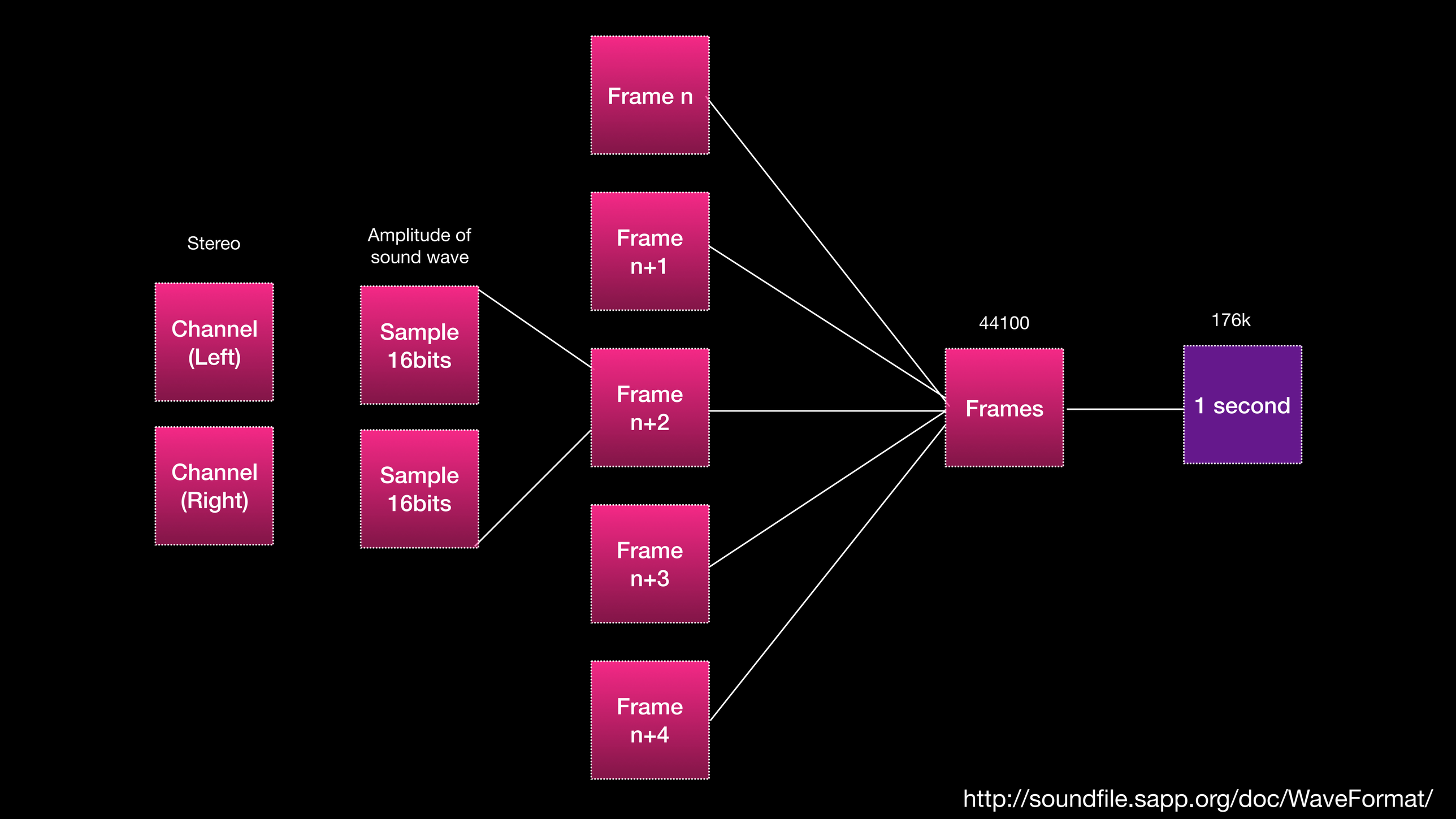

Noise
Silver Apples

SAMPLES

“Real-world noise of any sort has a very wide frequency range, far wider and more complex than any electronically generated sound.”

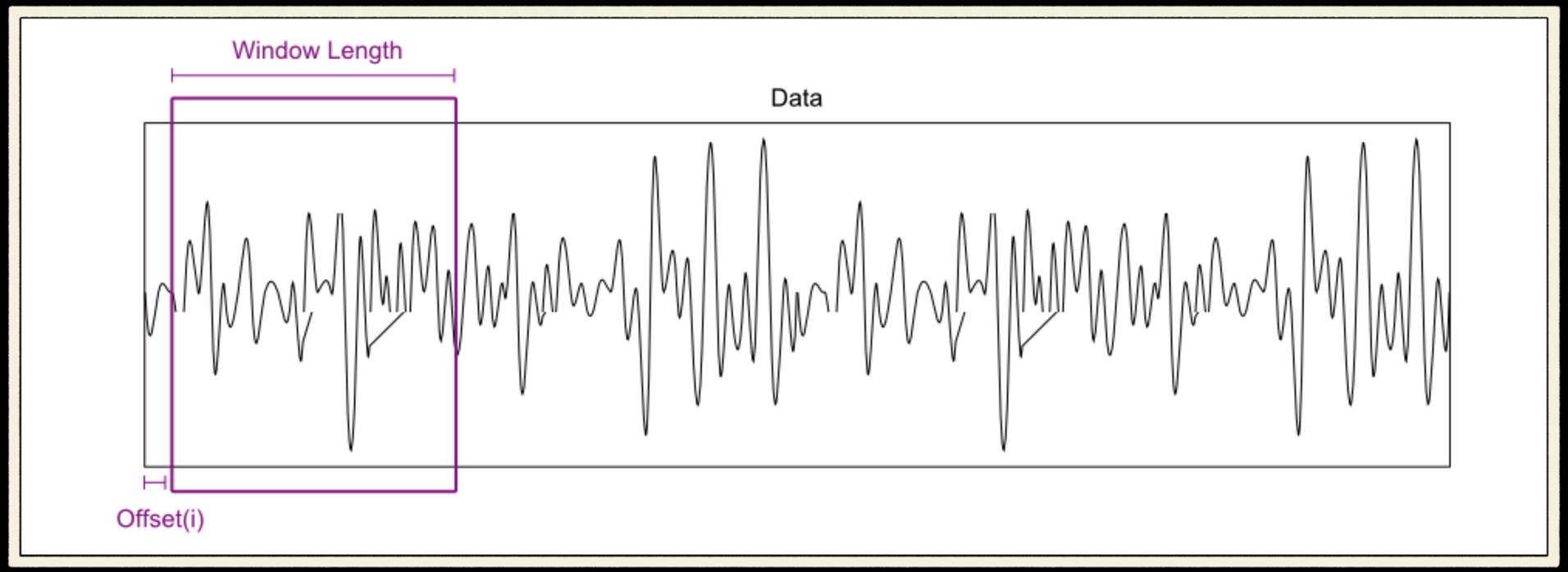
Jon Hopkins

https://en.wikipedia.org/wiki/Jon_Hopkins



http://soundfile.sapp.org/doc/WaveFormat/


SAMPLES AS DATA



Fundamental frequency

Midi note

Root mean squared amplitude

Spectral-centroid

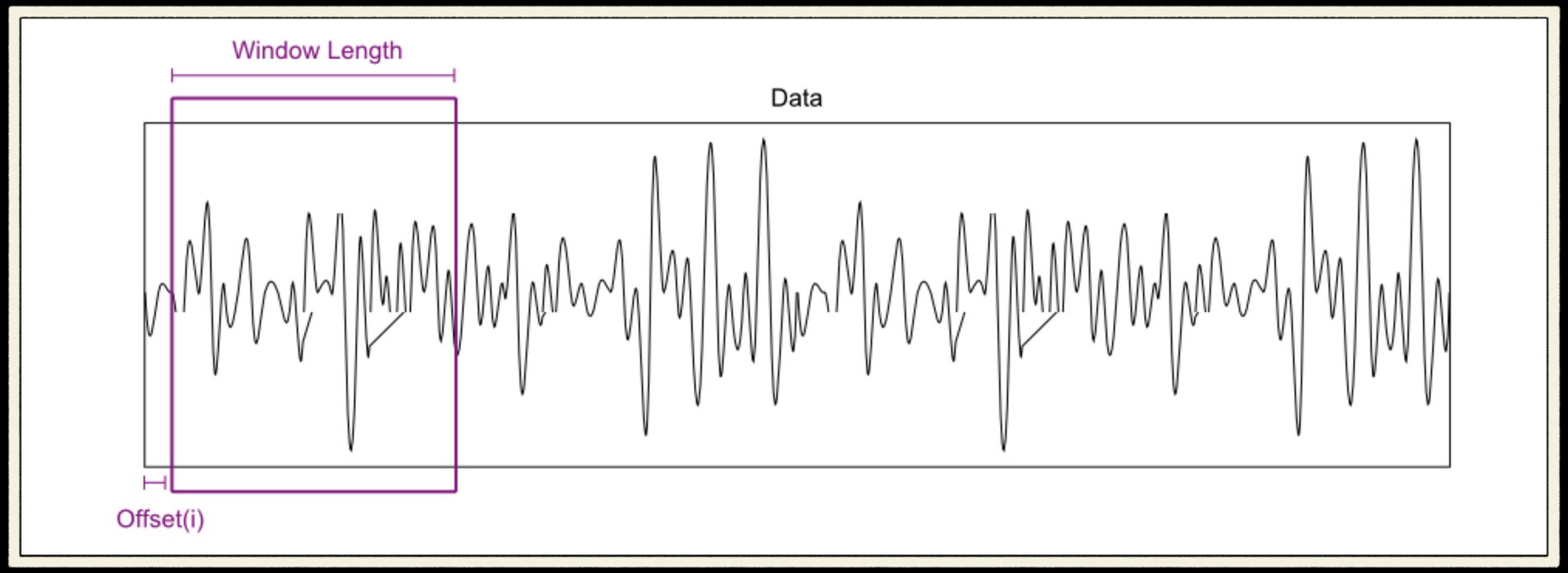
Spectral-irregularity

Spectral-inharmonicity

Spectral skewness

Spectral kurtosis

http://jamiebullock.github.io/LibXtract/documentation/



Fundamental frequency

Midi note

Volume

Brightness of Sound

Noisiness of a sound

How harmonic

Spectrum skew

Pitchiness

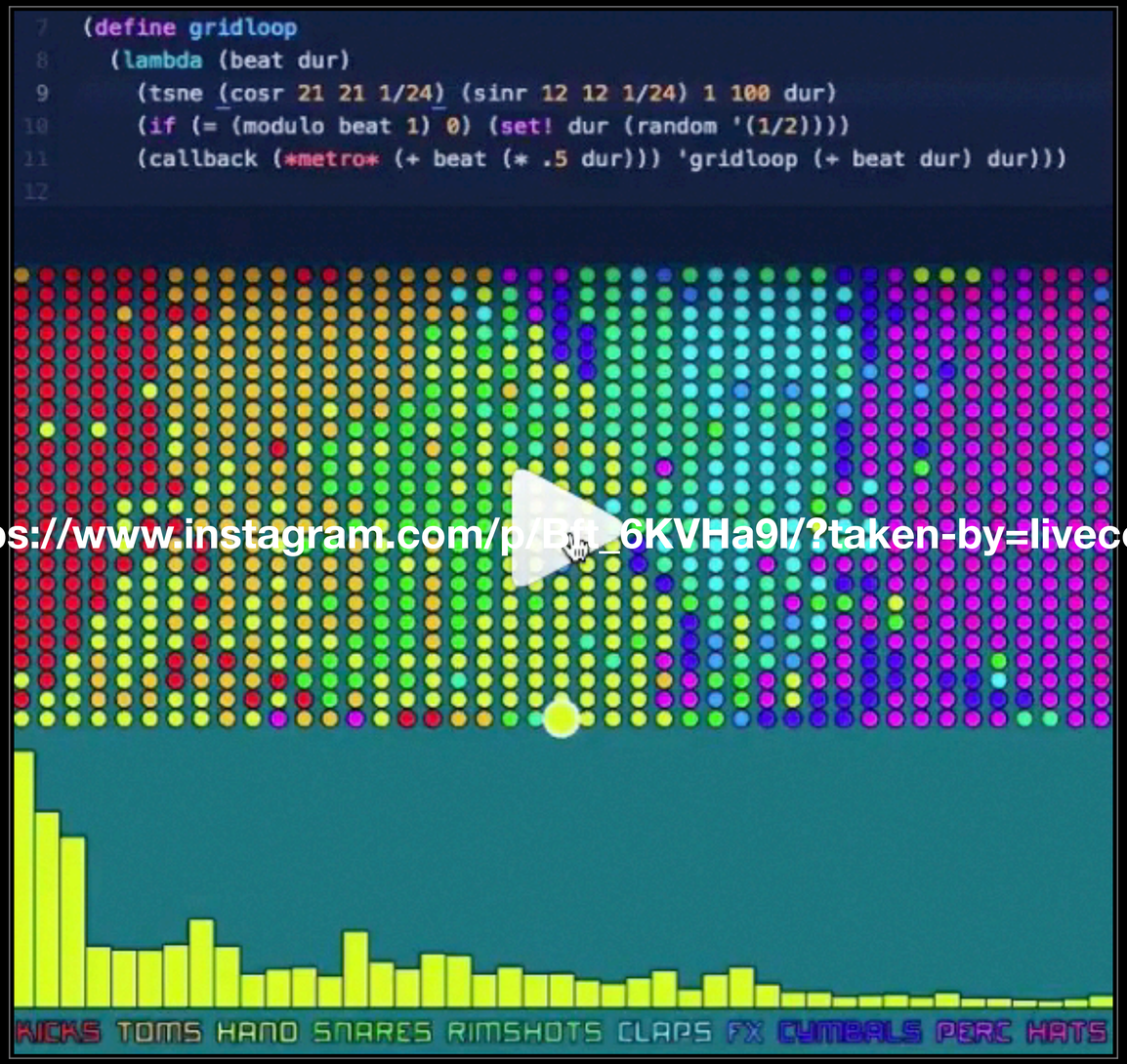
http://jamiebullock.github.io/LibXtract/documentation/

T-SNE

“magical ability to create compelling two-dimensional “maps”
from data with hundreds or even thousands of dimensions”

https://lvdmaaten.github.io/tsne/

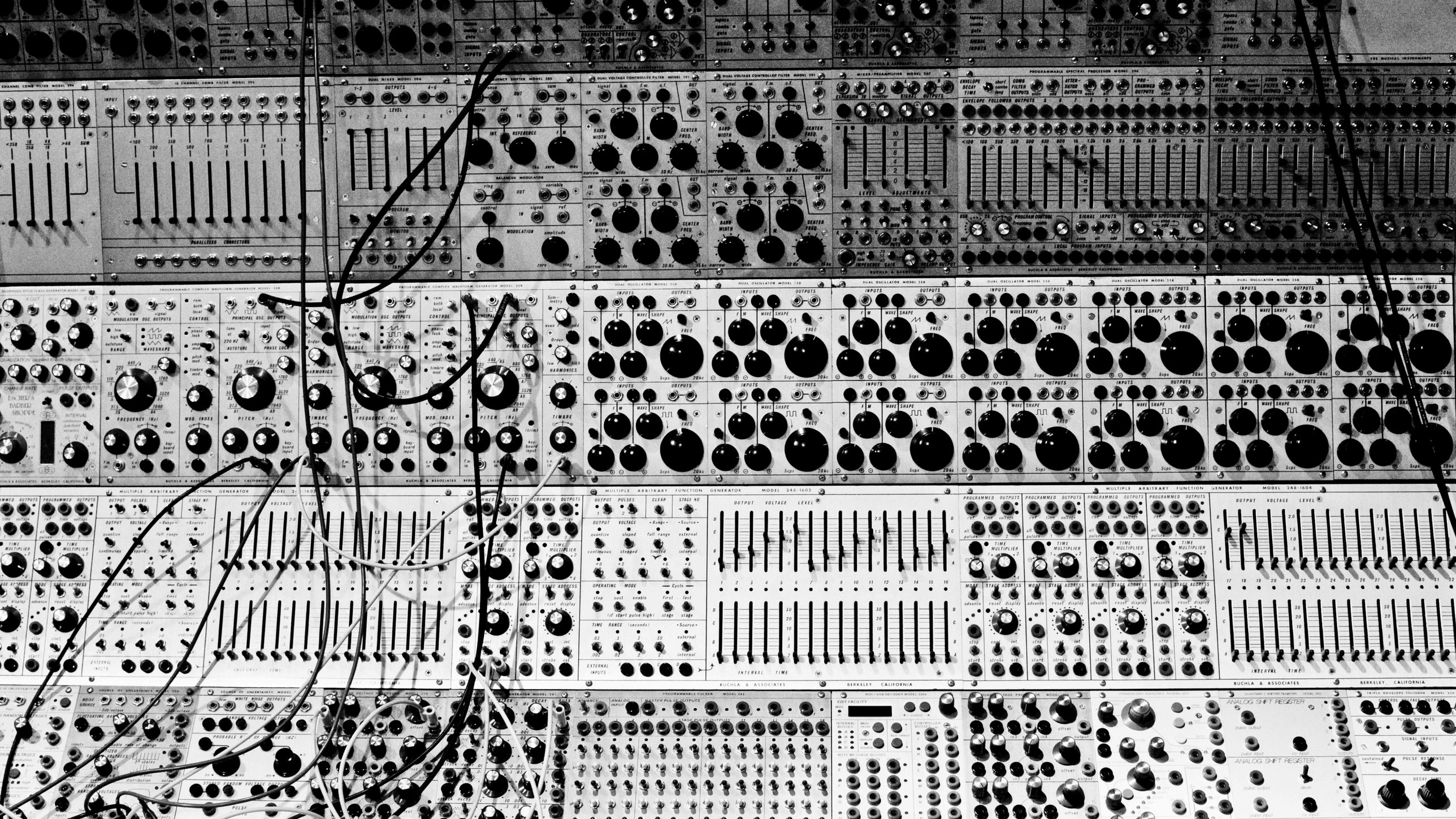

Gene Kogan

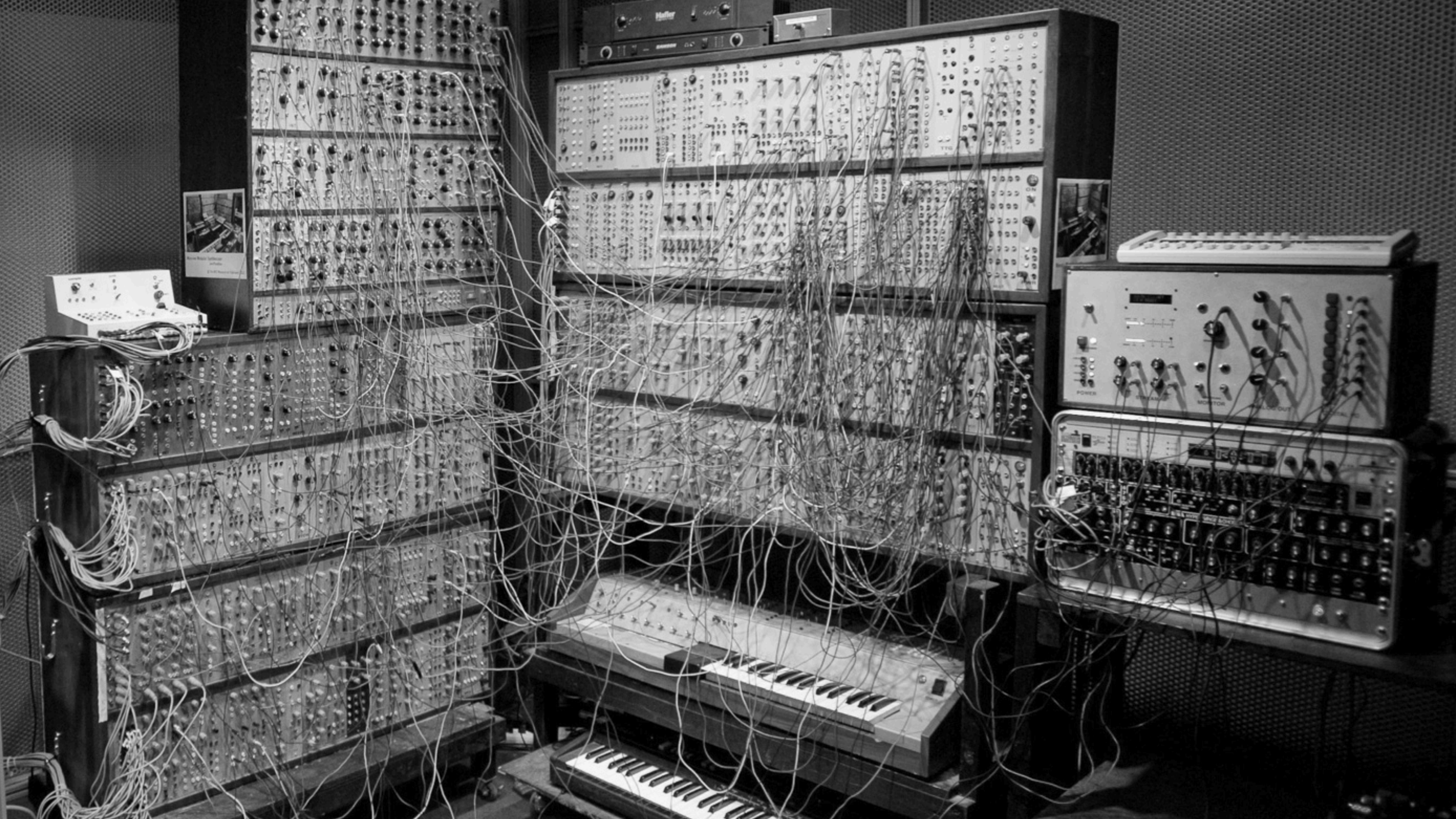


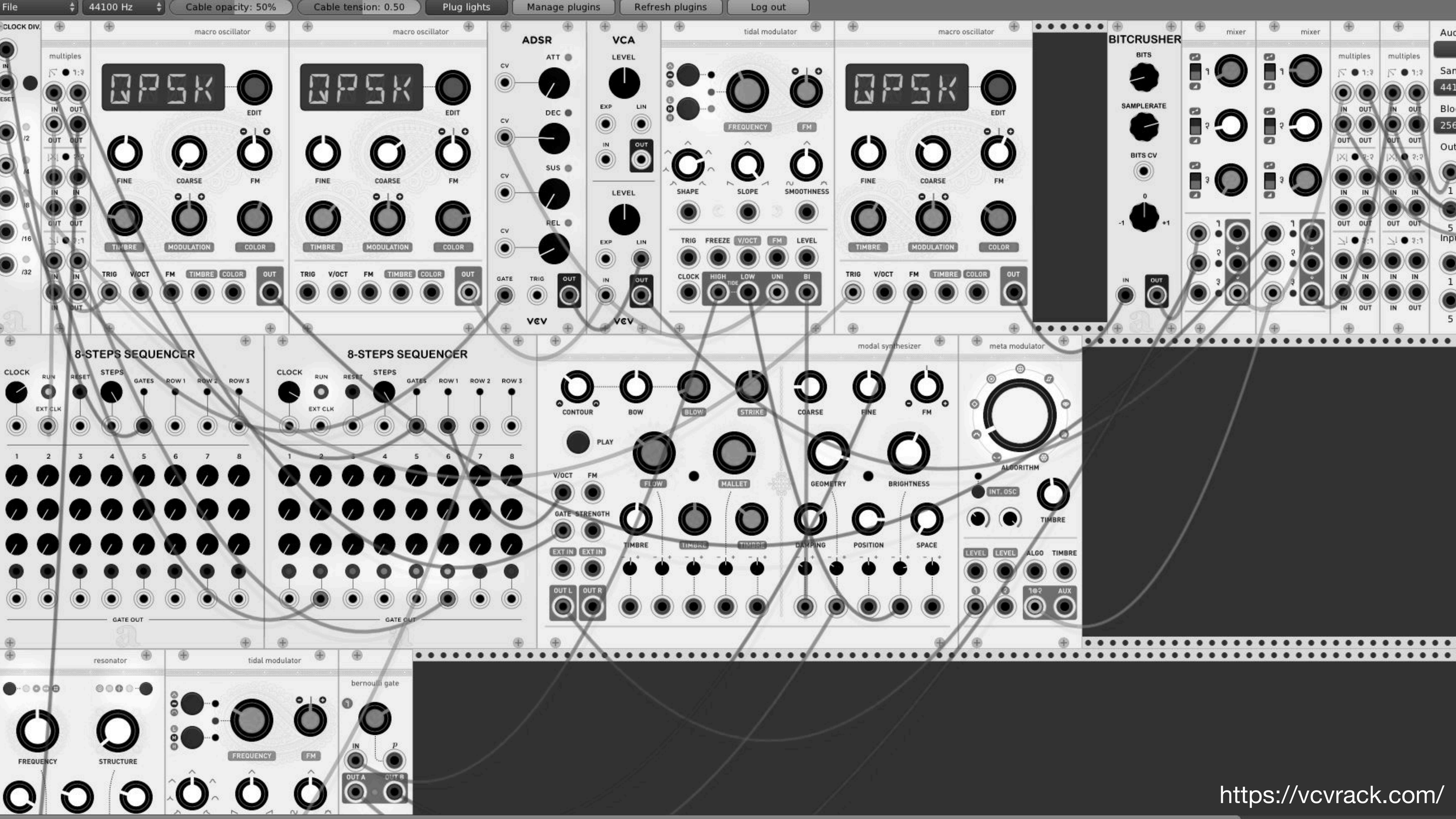
https://twitter.com/xululululuuum

PATCHING

https://en.wikipedia.org/wiki/Wendy_Carlos


DUAL VOLTAGE CONTROLLED FILTER MODEL 291
DUAL OSCILLATOR MODEL 258
MULTIPLE ARBITRARY FUNCTION GENERATOR MODEL 248-1602
MULTIPLE ARBITRARY FUNCTION GENERATOR MODEL 248-1604
BUCHLA & ASSOCIATES
BERKELEY, CALIFORNIA


Hafler
POWER


File
44100 Hz
Cable opacity: 50%
Cable tension: 0.50
Plug lights
Manage plugins
Refresh plugins
Log out
CLOCK DIV.
multiples
macro oscillator
QPSK
EDIT
FINE
COARSE
FM
TIMBRE
MODULATION
COLOR
TRIG
V/OCT
FM
TIMBRE
COLOR
OUT
ADSR
ATT
DEC
SUS
REL
CV
GATE
TRIG
OUT
VCA
LEVEL
EXP
LIN
IN
OUT
tidal modulator
FREQUENCY
FM
SHAPE
SLOPE
SMOOTHNESS
TRIG
FREEZE
V/OCT
FM
LEVEL
CLOCK
HIGH
LOW
UNI
BI
TIDE
BITCRUSHER
BITS
SAMPLERATE
BITS CV
mixer
8-STEPS SEQUENCER
CLOCK
RUN
RESET
STEPS
GATES
ROW 1
ROW 2
ROW 3
EXT CLK
GATE OUT
modal synthesizer
CONTOUR
BOW
BLOW
STRIKE
COARSE
FINE
FM
PLAY
V/OCT
FM
FLOW
MALLET
GEOMETRY
BRIGHTNESS
GATE
STRENGTH
TIMBRE
DAMPING
POSITION
SPACE
EXT IN
OUT L
OUT R
meta modulator
ALGORITHM
INT. OSC
TIMBRE
LEVEL
ALGO
AUX
resonator
FREQUENCY
STRUCTURE
bernoulli gate
OUT A
OUT B
https://vcvrack.com/

CODE

```
// 60Hz Gabber Rave 1995
Server.default.boot;

(
SynthDef(\gabberkick, {
    var snd, freq, high, lfo;
    freq = \freq.kr(440) * (Env.perc(0.001, 0.08, curve: -1).ar * 48 *
\bend.kr(1)).midiratio;
    snd = Saw.ar(freq);
    snd = (snd * 100).tanh + ((snd.sign - snd) * -8.dbamp);
    high = HPF.ar(snd, 300);
    lfo = SinOsc.ar(8, [0, 0.5pi]).range(0, 0.01);
    high = high.dup(2) + (DelayC.ar(high, 0.01, lfo) * -2.dbamp);
    snd = LPF.ar(snd, 100).dup(2) + high;
    snd = RLPF.ar(snd, 7000, 2);
    snd = BPeakEQ.ar(snd, \ffreq.kr(3000) * XLine.kr(1, 0.8, 0.3), 0.5,
15);
    snd = snd * Env.asr(0.001, 1, 0.05).ar(2, \gate.kr(1));
    Out.ar(\out.kr(0), snd * \amp.kr(0.1));
}).add;
```

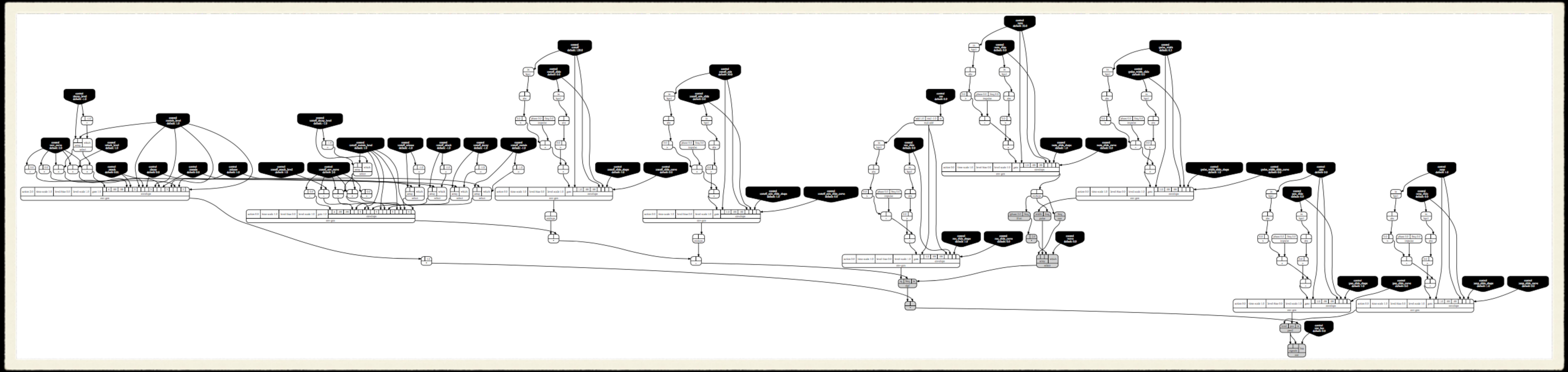
https://supercollider.github.io/

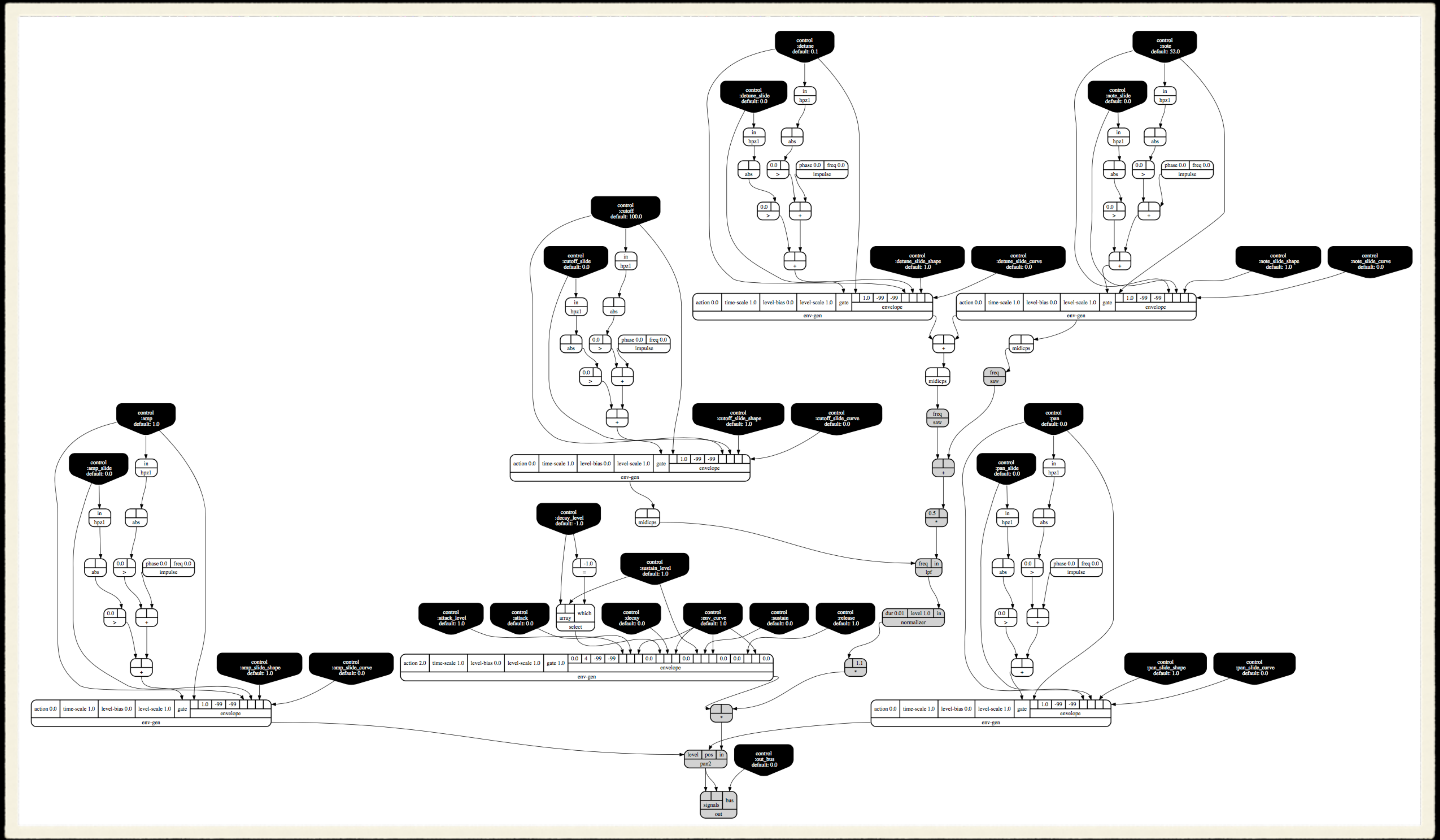
CODE

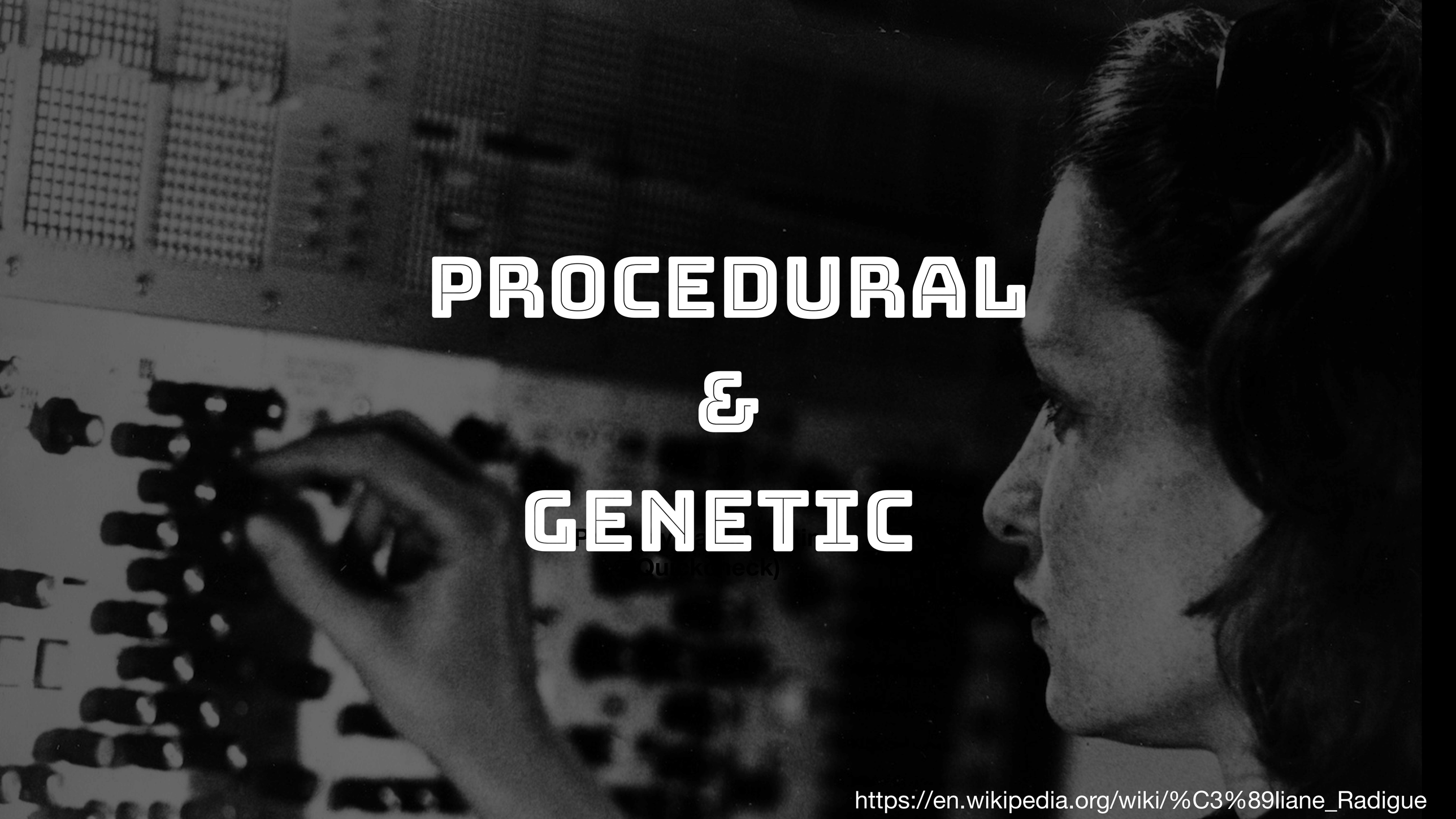
```
// 60Hz Gabber Rave 1995
Server.default.boot;


(
SynthDef(\gabberkick, {
    var snd, freq, high, lfo;
    freq = \freq.kr(440) * (Env.perc(0.001, 0.08, curve: -1).ar * 48 *
\bend.kr(1)).midiratio;
    snd = Saw.ar(freq);
    snd = (snd * 100).tanh + ((snd.sign - snd) * -8.dbamp);
    high = HPF.ar(snd, 300);
    lfo = SinOsc.ar(8, [0, 0.5pi]).range(0, 0.01);
    high = high.dup(2) + (DelayC.ar(high, 0.01, lfo) * -2.dbamp);
    snd = LPF.ar(snd, 100).dup(2) + high;
    snd = RLPF.ar(snd, 7000, 2);
    snd = BPeakEQ.ar(snd, \ffreq.kr(3000) * XLine.kr(1, 0.8, 0.3), 0.5,
15);
    snd = snd * Env.asr(0.001, 1, 0.05).ar(2, \gate.kr(1));
    Out.ar(\out.kr(0), snd * \amp.kr(0.1));
}).add;
```

CODE

```
// 60Hz Gabber Rave 1995
Server.default.boot;


(
SynthDef(\gabberkick, {
    var snd, freq, high, lfo;
    freq = \freq.kr(440) * (Env.perc(0.001, 0.08, curve: -1).ar * 48 *
\bend.kr(1)).midiratio;
    snd = Saw.ar(freq);
    snd = (snd * 100).tanh + ((snd.sign - snd) * -8.dbamp);
    high = HPF.ar(snd, 300);
    lfo = SinOsc.ar(8, [0, 0.5pi]).range(0, 0.01);
    high = high.dup(2) + (DelayC.ar(high, 0.01, lfo) * -2.dbamp);
    snd = LPF.ar(snd, 100).dup(2) + high;
    snd = RLPF.ar(snd, 7000, 2);
    snd = BPeakEQ.ar(snd, \ffreq.kr(3000) * XLine.kr(1, 0.8, 0.3), 0.5,
15);
    snd = snd * Env.asr(0.001, 1, 0.05).ar(2, \gate.kr(1));
    Out.ar(\out.kr(0), snd * \amp.kr(0.1));
}).add;
```




control :detune default: 0.1
control :note default: 52.0
control :cutoff default: 100.0
control :amp default: 1.0
control :pan default: 0.0
control :decay_level default: -1.0
control :sustain_level default: 1.0
control :attack_level default: 1.0
control :attack default: 0.0
control :decay default: 0.0
control :env_curve default: 1.0
control :sustain default: 0.0
control :release default: 1.0
control :out_bus default: 0.0
env-gen
envelope
impulse
midicps
saw
lpf
normalizer
select
pan2
out



https://en.wikipedia.org/wiki/%C3%89liane_Radigue

PROPERTY BASED TESTING

Generators

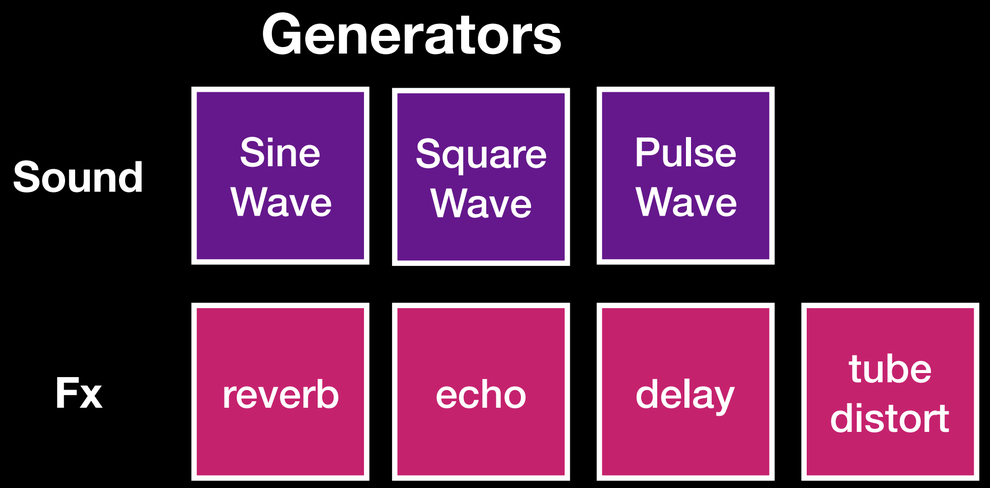
```
(fn []
  (let [color (gen/rgb-color)
        size  (gen/int)]
    (build-a-square color size)))
```

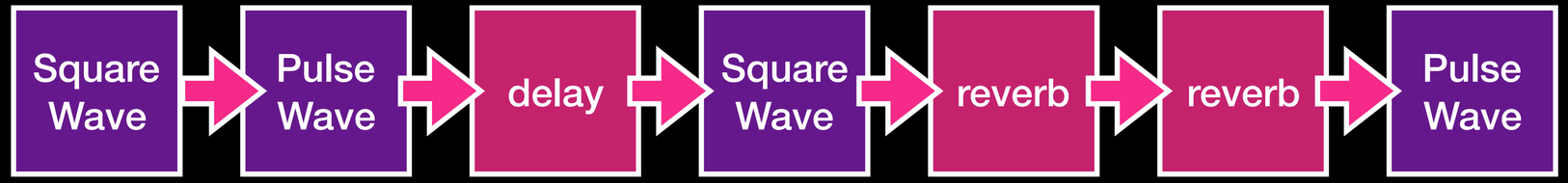
Properties

All things that are a square
should have 4 sides of equal length

https://en.wikipedia.org/wiki/QuickCheck

PROPERTY BASED TESTING






Square Wave
Pulse Wave
delay
reverb
Tube

FITNESS ?

Tweets
6

Follow

Fingerprint Smudger

@Finger_smudger

I'm a robot creating generative music that tricks audio identification services into thinking its another music track.
soundcloud.com/stream By @josephwilk

Joined June 2016

Tweets | Tweets & replies

Pinned Tweet

Fingerprint Smudger @Finger_smudger · 26 Jul 2016
My generated track soundcloud.com/fingerprint-sm...
Identified as the musicians:
* George Michael
* Duran Duran
* Axiuw
* Dezt

Generation #1.1 - 1469483772764
Generative electronic music using Shazam to guide generation maximising the number of incorrectly identify

New to Twitter?

Sign up now to get your own personalised timeline!

Sign up

Worldwide trends

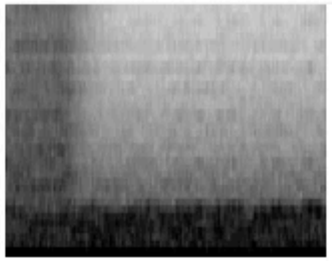
TAMAM
647K Tweets

#LOVE_YOURSELF
1.37M Tweets

Fingerprint Smudger

Station | Following | Share

Language: English (US)



2:01

Write a comment

2 | Repost | Share | More | 63

Fingerprint Smudger — 1 year

Generation #1.1 - 1469483772764

#MusicForMachines

0:13 — 2:01

Write a comment

1 | Repost | Share | More | 124

Fingerprint Smudger — 1 year

Generation #1 - 1467636802259

#MusicForMachines

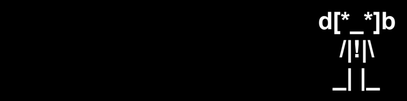
0:24 — 5:30

Write a comment

0:00 — 2:01

https://twitter.com/Finger_smudger

https://en.wikipedia.org/wiki/Daphne_Oram

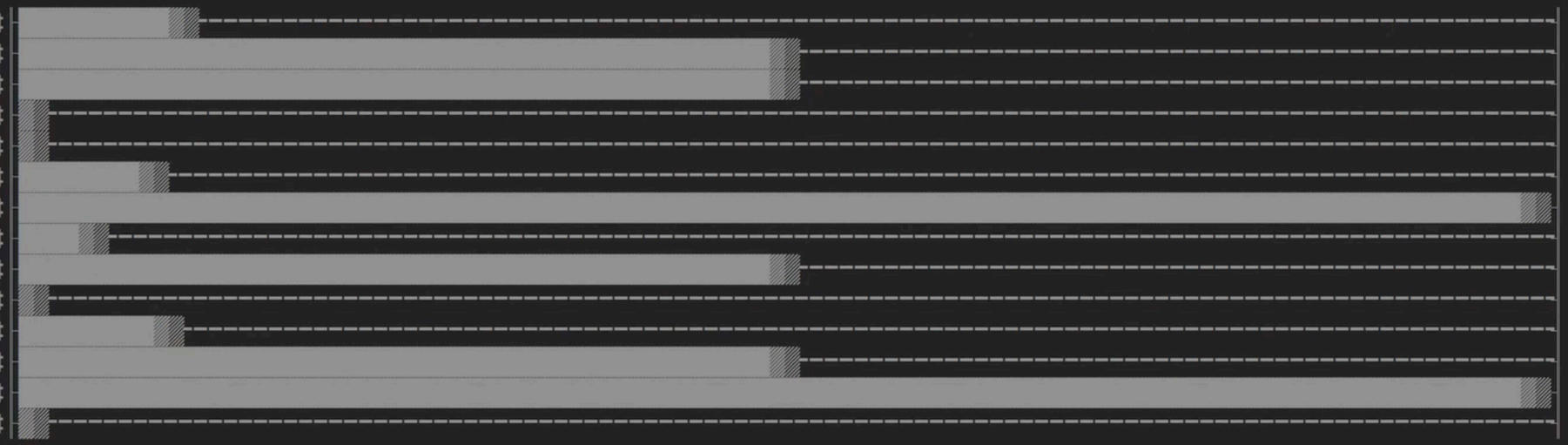


EMACS IS A OPERATING SYSTEM

IS A MUSICAL INSTRUMENT

YES REALLY

```
#sop2_on :A3

sop2_cc oct: 0.10      #
sop2_cc drive: 0.50    #
sop2_cc charge: 0.50   #
sop2_cc fm: 0.00       #
sop2_cc wav: 0.00      #
sop2_cc atk: 0.08      #
sop2_cc mod: 1.00      #
sop2_cc motion: 0.036  #
sop2_cc formant: 0.50  #
sop2_cc mul: 0.00      #
sop2_cc bass: 0.09     #
sop2_cc width: 0.50    #
sop2_cc center: 1.00   #
sop2_cc head: 0.00     #

eq lo: 0.85
eq mi: 0.85
eq hi: 0.85
```



https://vimeo.com/265188142

```
-UU-:**--F1  break.spi      All (3,15)     (n AC G-+ Undo-Tree yas VHl Outl -1-) ----------
```

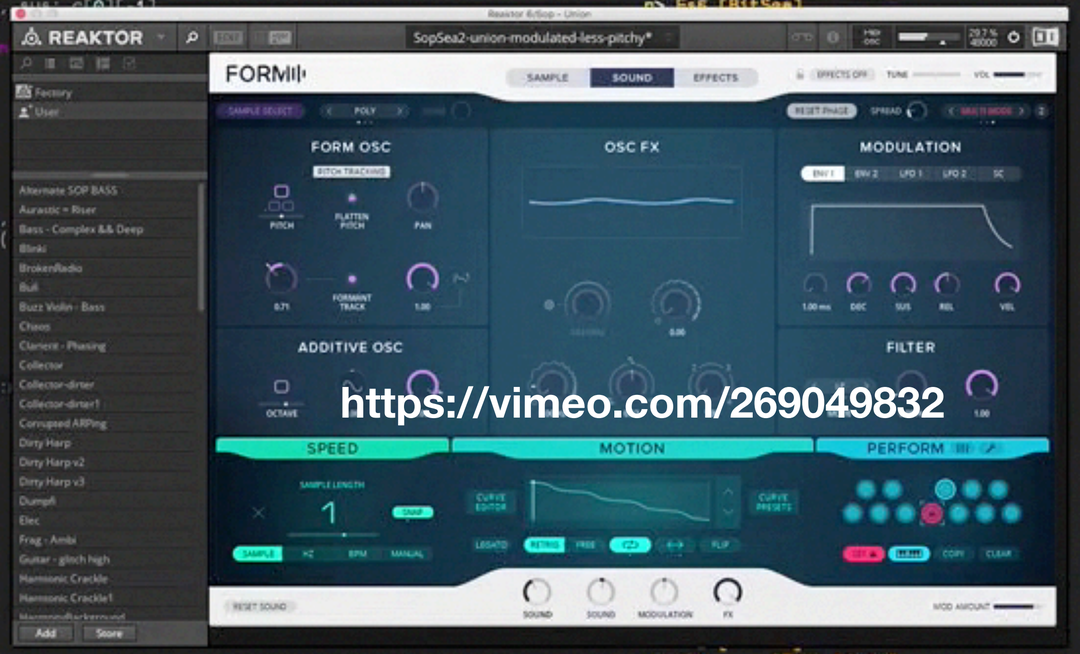
SCRATCHING SAMPLES WITH EMACS

d[x_x]b
/|!|\
_/ _


PRESETS
BANK
PATCH
ACTIVATE PANEL
PITCH
MIDI
OCTAVE
MODULATION
SOURCE
PITCH AMT
OSC 2 ONLY
OSCILLATORS
1
2
HARD SYNC
OSC 2
MIXER
FILTER
ENVELOPES
FILTER
AMPLIFIER
ATTACK
DECAY
SUSTAIN
RELEASE
OUTPUT
SUB PHATTY
SUBTRACTIVE ANALOG SYNTHESIZER
Designed & Handcrafted By Moog Music In Asheville, NC
moog
PITCH
MOD

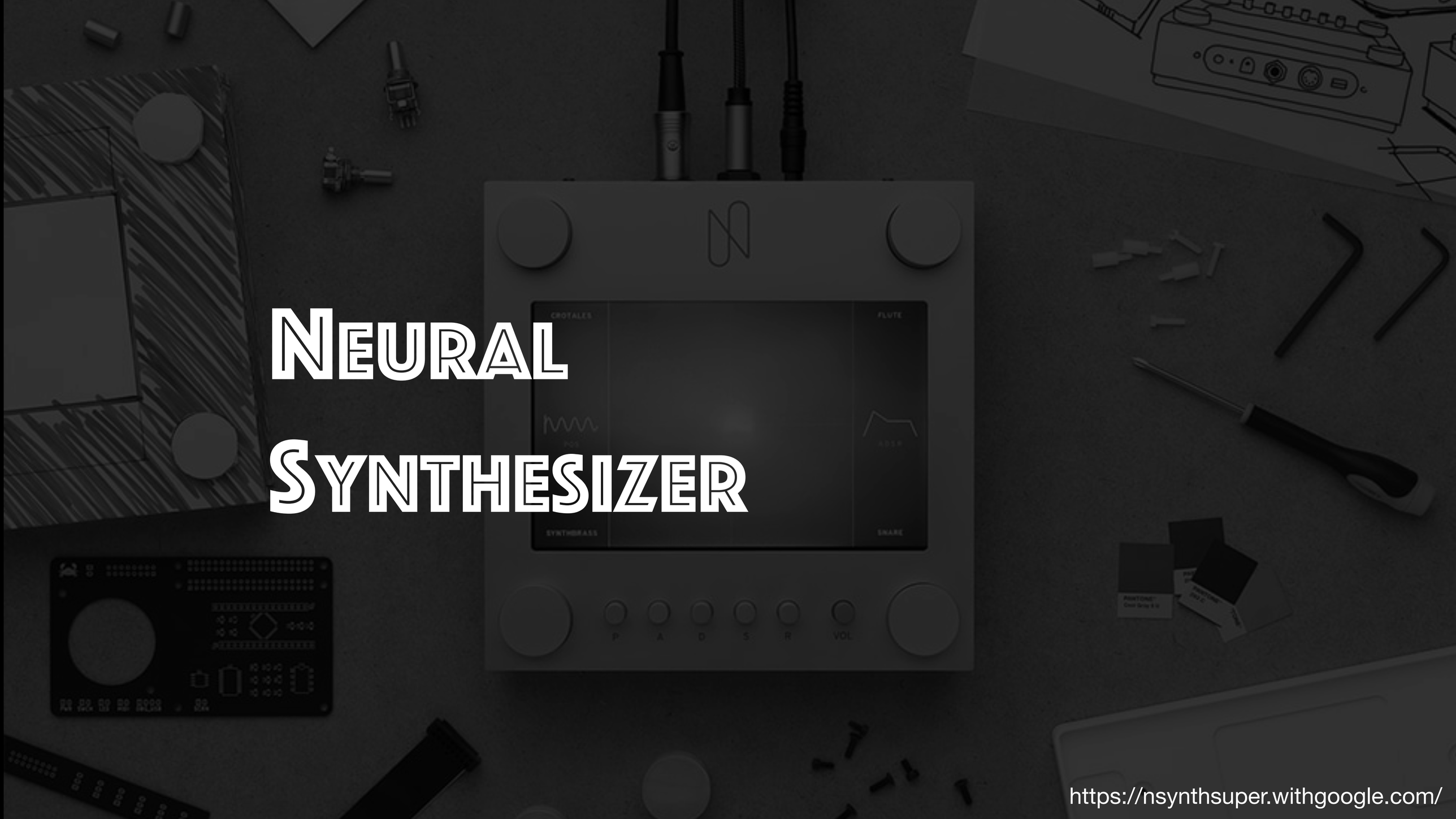
```
  score = (ing
    [[[ a3 2]  [ cs4 2] [ a3 2]] 2]          [[[ fs5 0.125]] 0.5]
    [[[ cs3 2] [ e4 2] [ fs3 2]] 2]          [[[ gs5 0.125]] 0.5]
  )
  s=score.look;c=s[0];zzz=s[-1]
  #eek :a1,  sus: 12
  #eek :cs2, sus: 12
  if spread(1,6).look
    #rev :b5, 1
    #eek :Gs3, 10
  end
  #with_transpose 0 { cpu3 root(c),(ing 127 120).look, sus: 2 }
  #rev :a6, 15, sus: 12
  #callstack root(c), (ring 35,40).look*1.7, sus: c[0][-1]
  c.each_tick(){|n|
    bitsea n, 1.0*(ing 80 70 85 70).look( in
(:oo)#, cutoff: 1.0
    bitsea_cc cutoff: 0.01
  }
  #null root(c), 127*1.0, mode: 1, sus: 1
  exception root(c), 127, sus: 2
  strings c[0][-2], 35
  sleep zzz
  #corrupt root(c)[0]+12*0, 10, motion: 0.1,
  #verb_slice Drip[/effect/].look if spread(
  sleep zzz
end

#zero_on
zero_cc pulse: 0.00
zero_cc more:  0.00
-UU-:----F1  live.spi        2% (22,56)    Git:
          qbitsea_cc args_h
        end
      end
    end

    def exception(n,*args)
      if n
        if n.is_a?(Array)
          args =  args  << {sus: n[1]}
          n = n[0]
        end
        if args && args[0].is_a?(Numeric)
          velocity = args[0]
          args = args[1..-1]
        else
          velocity = 15
        end
        args_h = resolve_synth_opts_hash_or_array(args)
        if n && ((n != "_") && n != :_)
          dshader :decay,  :Sharp, (note(n)/69.0)
          puts "%s%s" %[SonicPi::Note.new(n).midi_string.ljust(9, " "), "[Exception!]"]  unless note(n\
) < MODE_NOTE
          midi n, velocity, *(args << {port: :iac_bus_1} << {channel: 8})
        end
        exception_cc(args_h)
      end
    end

    def exception_cc(cc)
      cc.keys.each do |k|
        n = case k
-UU-:----F1  midi.rb        25% (241,11)   (Ruby AC G-+ Undo-Tree yas VHL Outl -2-) ---------- -UUU:**--F1
Wrote /Users/josephwilk/Workspace/repl-electric/adventures-in-algorhythm/you_fall_into_your_screen/live.spi
```

```
n> Cs4 [BitSea]
n> Fs5 [BitSea]
n> Fs5      [Exception!]
n> Fs3 [BitSea]
n> Cs2      [Exception!]
n> Cs2 [BitSea]
n> E4  [BitSea]
n> Gs5 [BitSea]
n> Ab5      [Exception!]
n> A2  [BitSea]
n> Cs4 [BitSea]
n> E3  [BitSea]
n> A2       [Exception!]
n> Ab5      [Exception!]
n> Cs4 [BitSea]
n> E3  [BitSea]
n> A2       [Exception!]
n> A2  [BitSea]
```



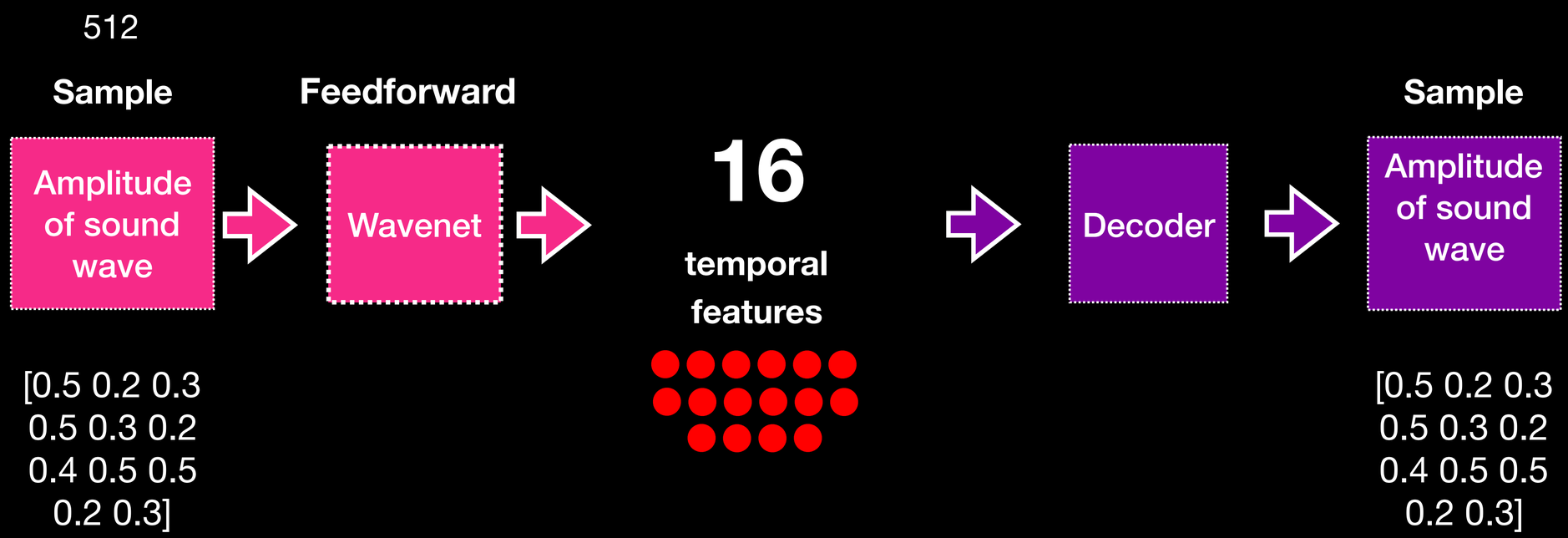
MACHINE LEARNING

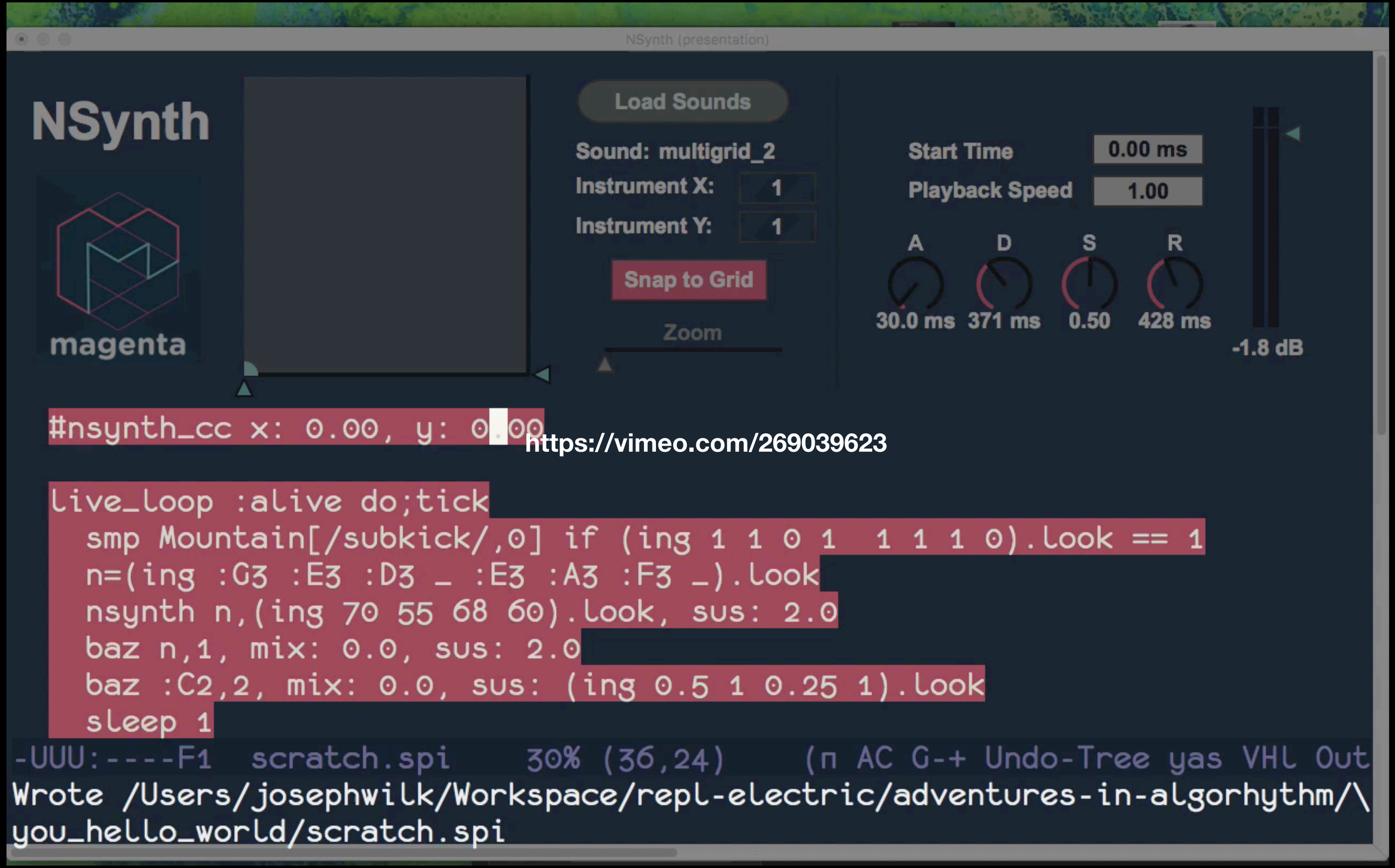
https://en.wikipedia.org/wiki/Else_Marie_Pade



Neural Synthesizer

https://nsynthsuper.withgoogle.com/


512
Sample
Amplitude of sound wave
[0.5 0.2 0.3
0.5 0.3 0.2
0.4 0.5 0.5
0.2 0.3]
Feedforward
Wavenet
16
temporal features
Decoder
Sample
Amplitude of sound wave
[0.5 0.2 0.3
0.5 0.3 0.2
0.4 0.5 0.5
0.2 0.3]


NSynth (presentation)
NSynth
magenta
Load Sounds
Sound: multigrid_2
Instrument X: 1
Instrument Y: 1
Snap to Grid
Zoom
Start Time 0.00 ms
Playback Speed 1.00
A D S R
30.0 ms 371 ms 0.50 428 ms
-1.8 dB
#nsynth_cc x: 0.00, y: 0.00
https://vimeo.com/269039623
live_loop :alive do;tick
smp Mountain[/subkick/,0] if (ing 1 1 0 1 1 1 1 0).look == 1
n=(ing :G3 :E3 :D3 _ :E3 :A3 :F3 _).look
nsynth n,(ing 70 55 68 60).look, sus: 2.0
baz n,1, mix: 0.0, sus: 2.0
baz :C2,2, mix: 0.0, sus: (ing 0.5 1 0.25 1).look
sleep 1
-UUU:----F1 scratch.spi 30% (36,24) (⊓ AC G-+ Undo-Tree yas VHl Out
Wrote /Users/josephwilk/Workspace/repl-electric/adventures-in-algorhythm/\
you_hello_world/scratch.spi

WaveGAN Demo

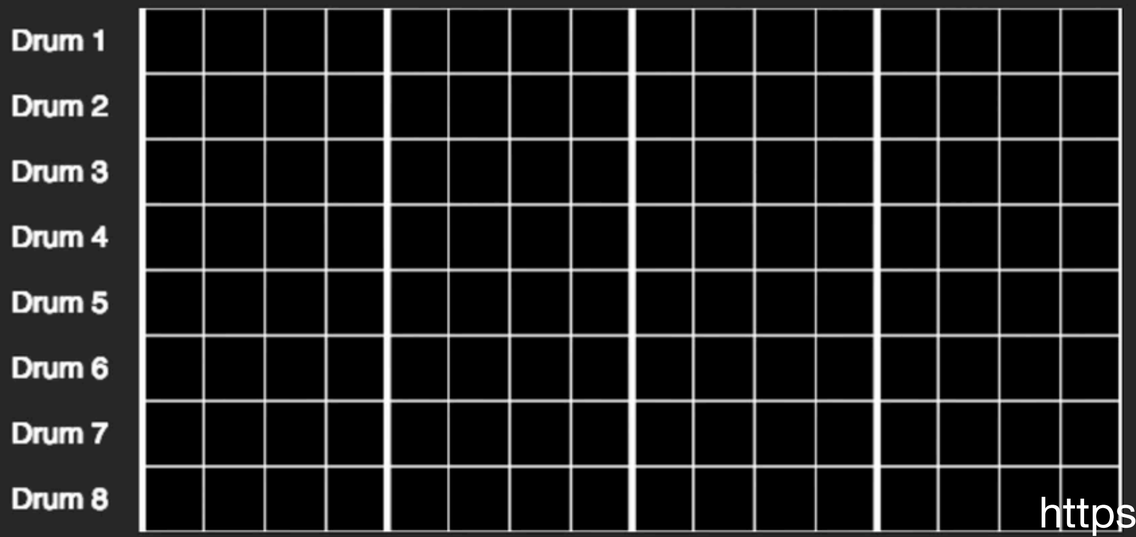
Chris Donahue, Julian McAuley, Miller Puckette

This is a demo of our WaveGAN method trained on drum sound effects (paper, code). All drum sounds are synthesized in browser by a neural network.

Shortcuts: Keys **1-8** play sounds. **Shift+[1-8]** changes sounds. **Space** starts/stops sequencer.

Volume Reverb

Play Stop Clear Tempo Swing



https://chrisdonahue.github.io/wavegan/

https://en.wikipedia.org/wiki/Suzanne_Ciani

Joseph Wilk