Introduction to Audio Programming

Meet the Masters, SAE Bochum

Marcus Ficner, 08.12.2021

Senior iOS Developer at Teufel

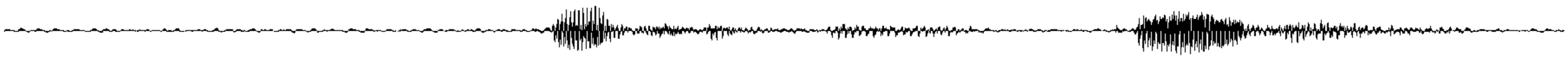
SAE Alumni from 2005
Worked in Studios 301 in Cologne

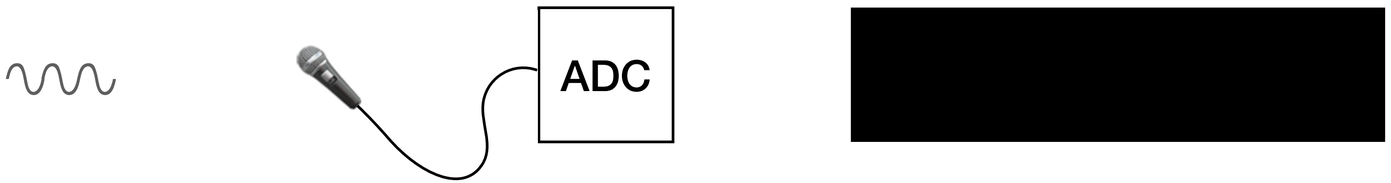
B. Eng. Information and Communication Technology

 @marcusficner

What is audio programming?

What is digital audio?





Sampling Frequency: Fs

Resolution: Q

Human hearing range: ~ 20 Hz - 20 kHz

Fs = 44100 Hz

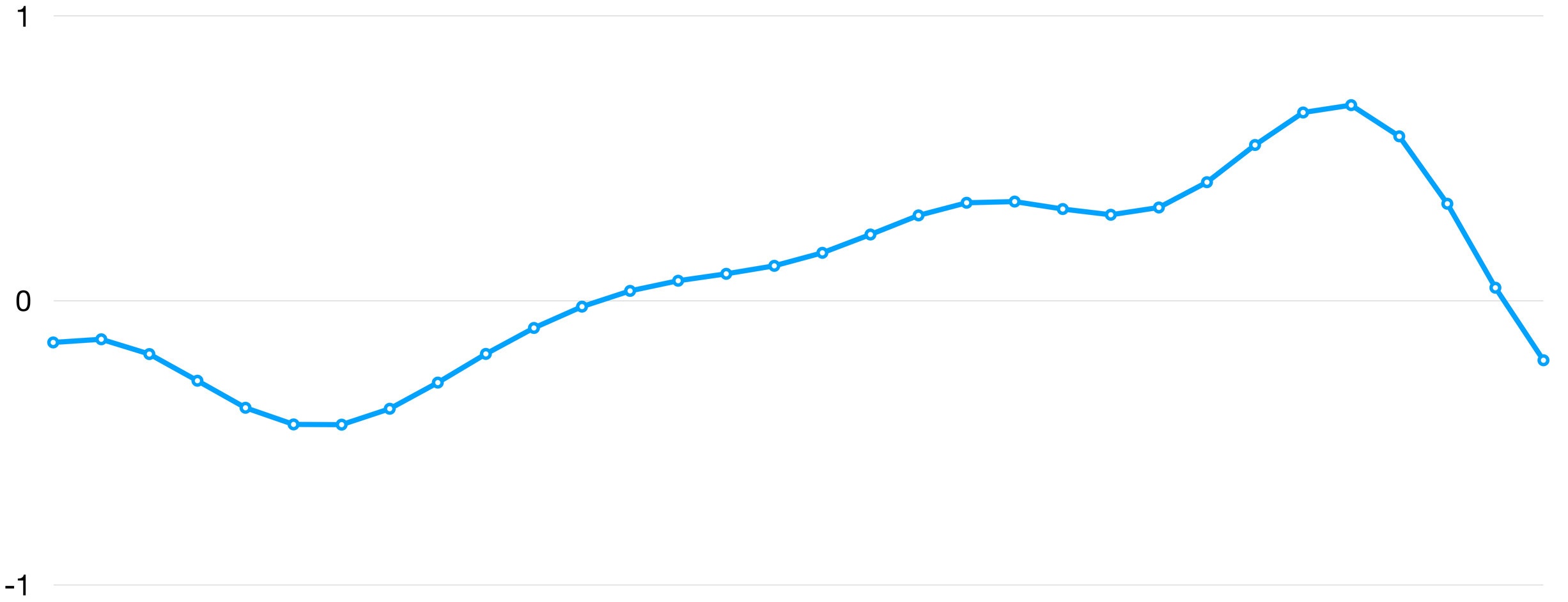
Q = 16 Bit


ADC
DAC

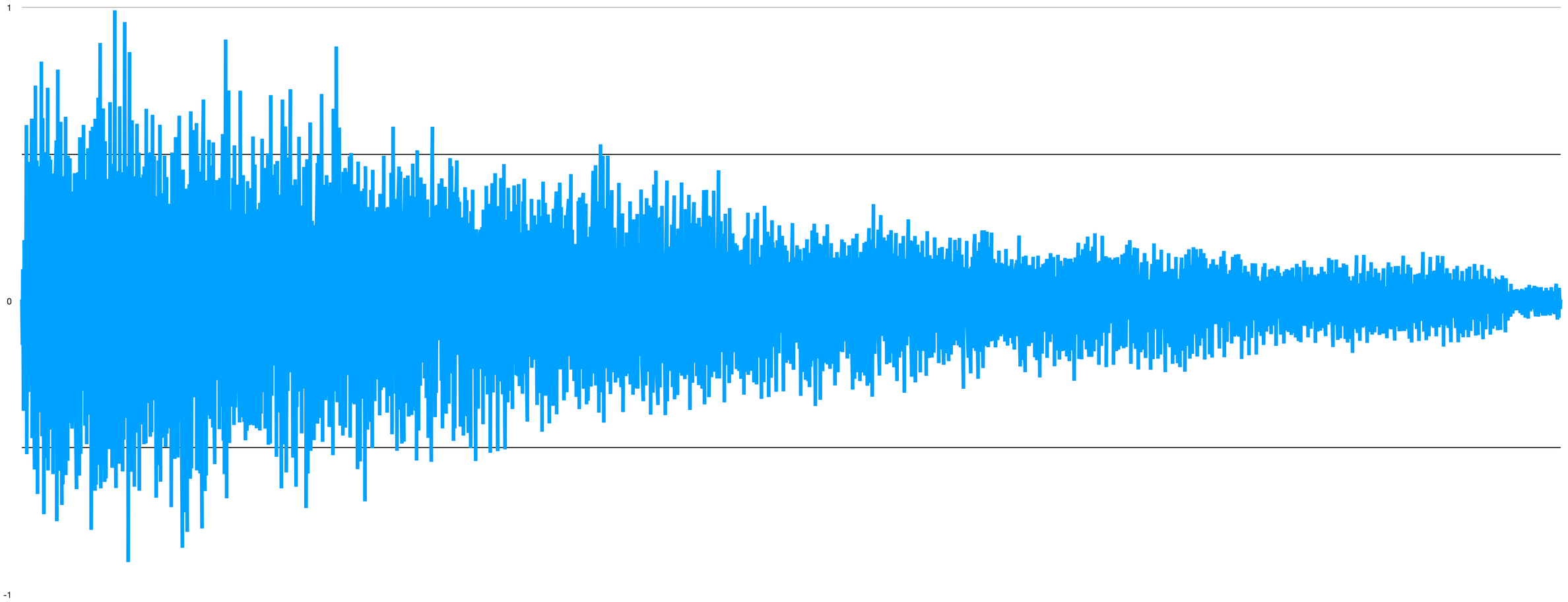

ADC
DAC

Digital Audio

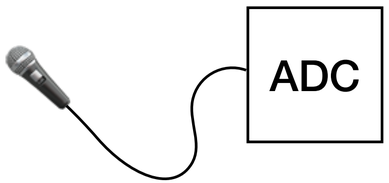
-0.15	-0.14	-0.19	-0.28	-0.38	-0.44	-0.44	-0.38	-0.29	-0.19	-0.10	-0.02	0.03	0.07	0.09	0.12	0.17	0.23	0.30	0.34	0.35	0.32	0.30	0.33	0.42	0.55	0.66	0.69	0.58	0.34	0.04	-0.21



Digital Audio



Visualize



Store

Transmit

Process

Synthesize

DAC

Load

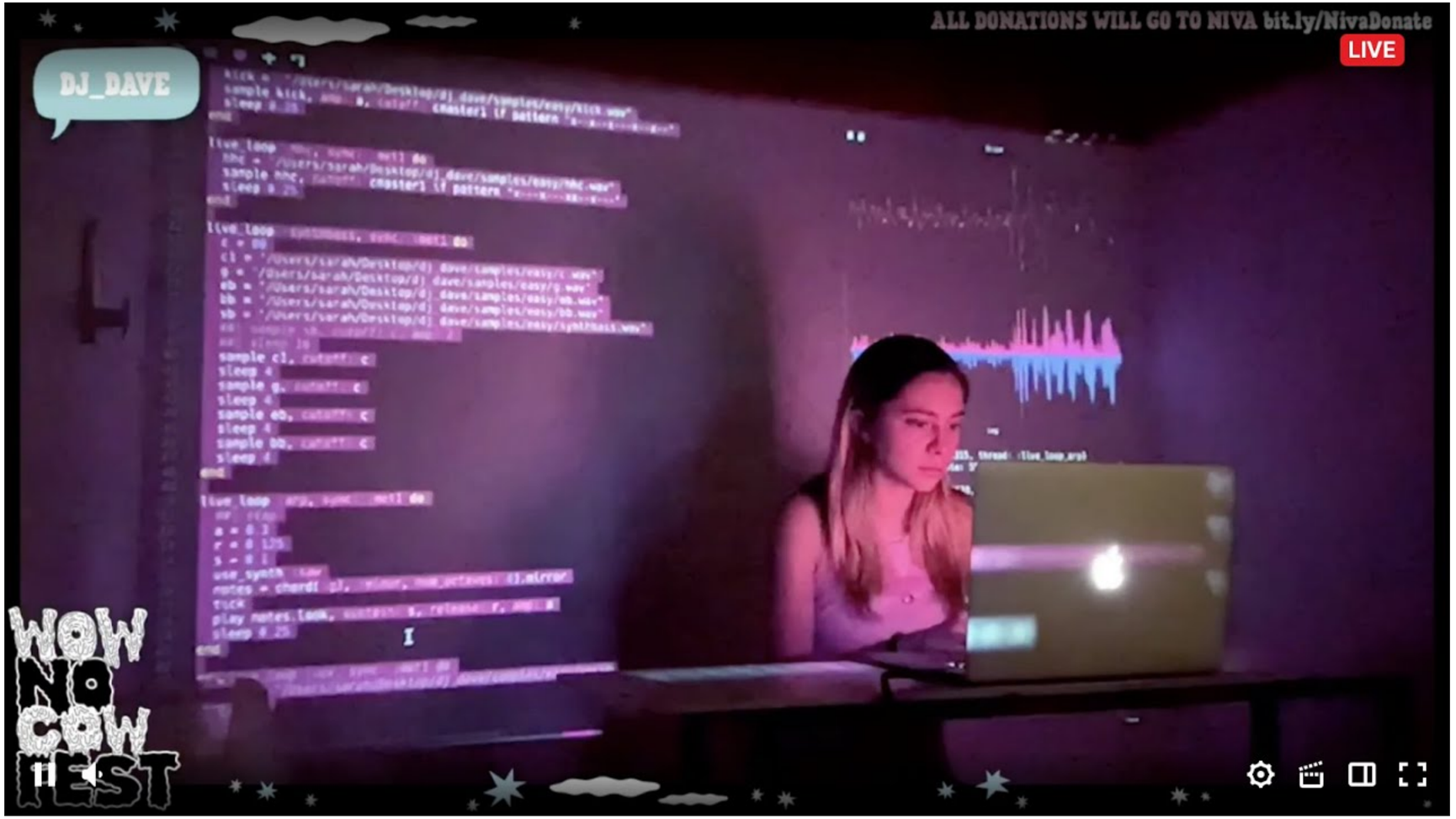
Receive

What is audio programming?

Audio Programming

- Live Coding / Algo raves
- Game Audio
- Audio hardware / embedded programming
- Information extraction
- Machine Learning
- Composing
- Music Production

Live coding / Algo Rave



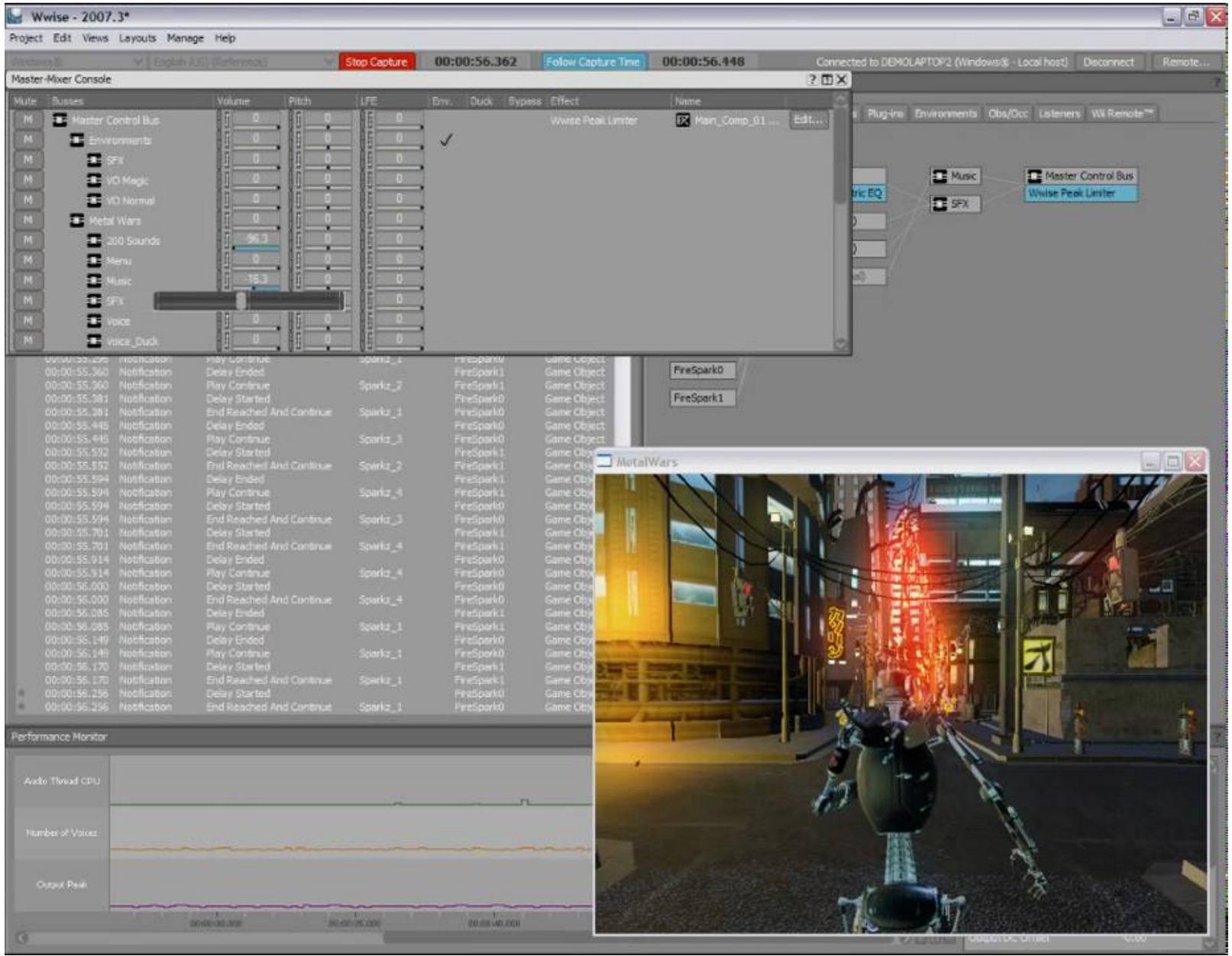
https://www.sarahcdavis.com/dj-dave

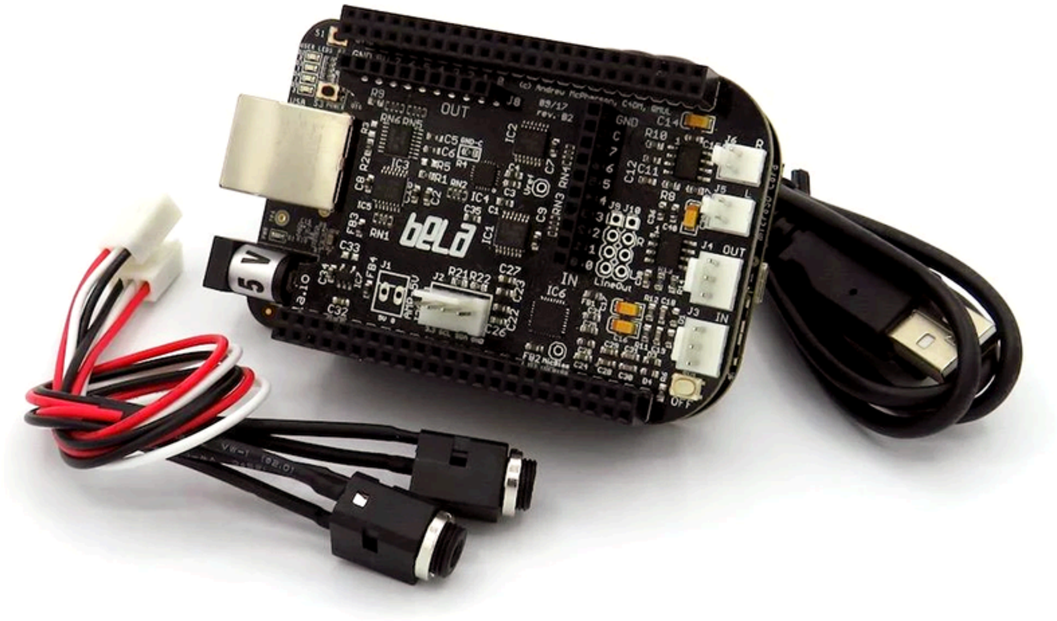
Live coding / Algo Rave

https://sonic-pi.net

Game Audio

Game Audio


Wwise - 2007.3*
Project Edit Views Layouts Manage Help
Stop Capture
00:00:56.362
Follow Capture Time
00:00:56.448
Master-Mixer Console
Mute Busses Volume Pitch LFE Env. Duck Bypass Effect Name
Master Control Bus
Environments
SFX
VO Magic
VO Normal
Metal Wars
200 Sounds
Menu
Music
SFX
Voice
Voice_Duck
Wwise Peak Limiter
Main_Comp_01
Edit...
Plug-ins Environments Obs/Occ Listeners Wii Remote™
Music
SFX
Master Control Bus
Wwise Peak Limiter
FireSpark0
FireSpark1
MetalWars
Performance Monitor
Audio Thread CPU
Number of Voices
Output Peak



Audio Hardware

Voice recognition

Music classification

Acoustic fingerprint

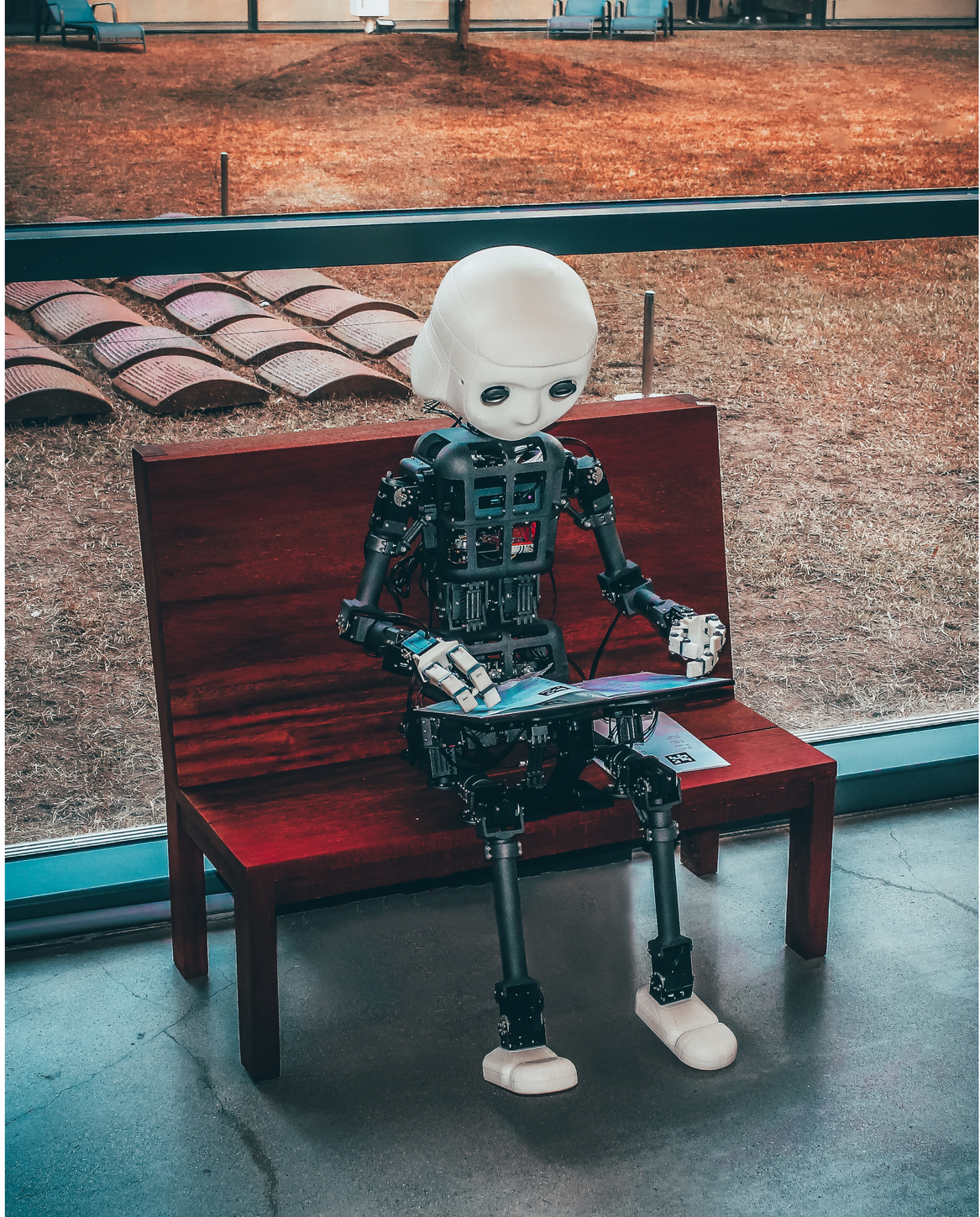
Information extraction

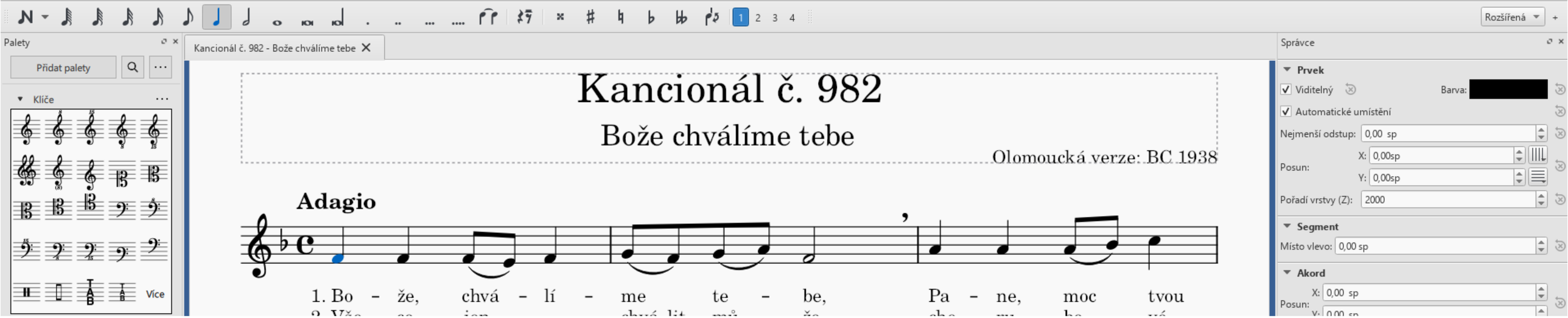
BPM / Beat detection

Automatic music transcription

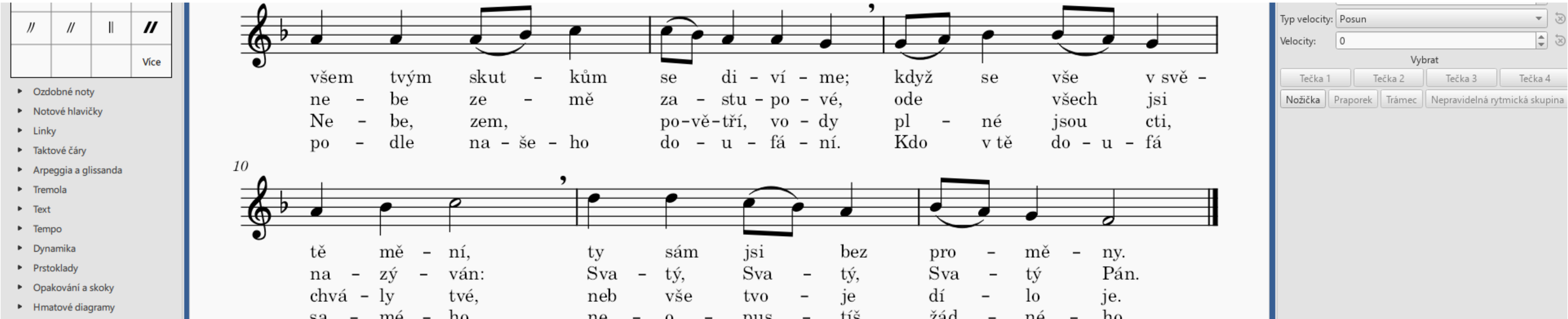
Music source separation and instrument recognition

Machine Learning

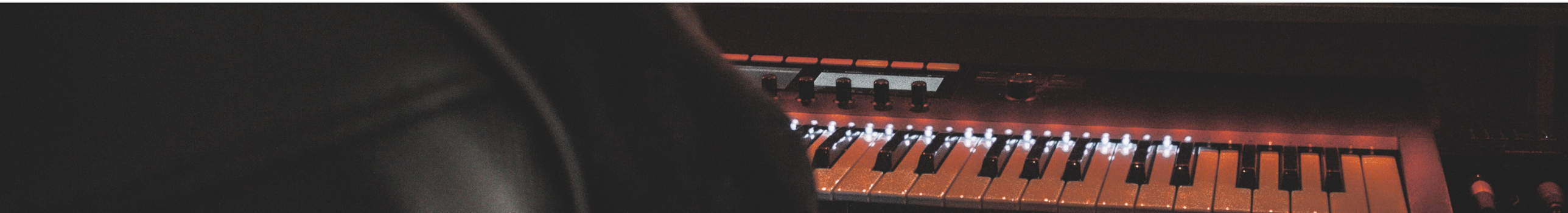



Palety
Přidat palety
Kliče
Více
Kancionál č. 982 - Bože chválíme tebe
Kancionál č. 982
Bože chválíme tebe
Olomoucká verze: BC 1938
Adagio
1. Bo - že, chvá - lí - me te - be, Pa - ne, moc tvou
Správce
Prvek
Viditelný
Barva:
Automatické umístění
Nejmenší odstup: 0,00 sp
Posun:
X: 0,00sp
Y: 0,00sp
Pořadí vrstvy (Z): 2000
Segment
Místo vlevo: 0,00 sp
Akord
X: 0,00 sp
Posun:

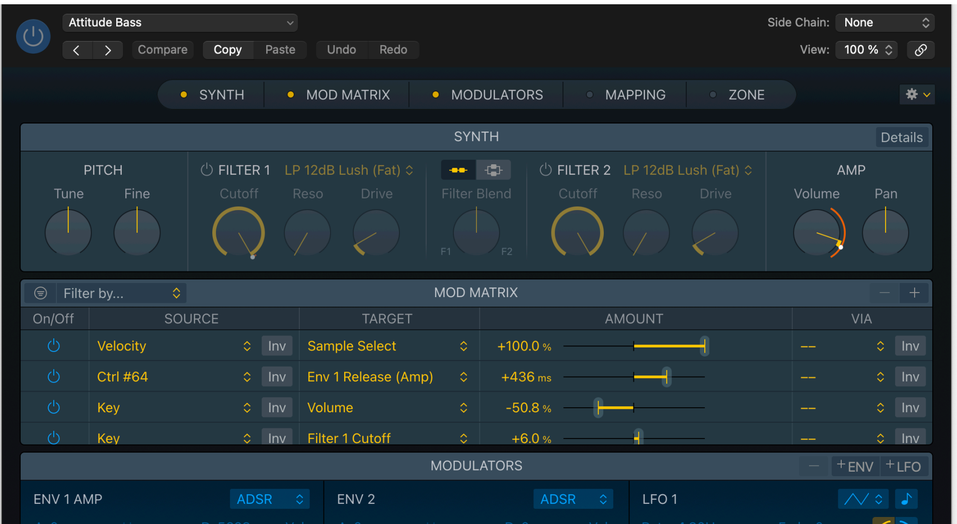
Composing

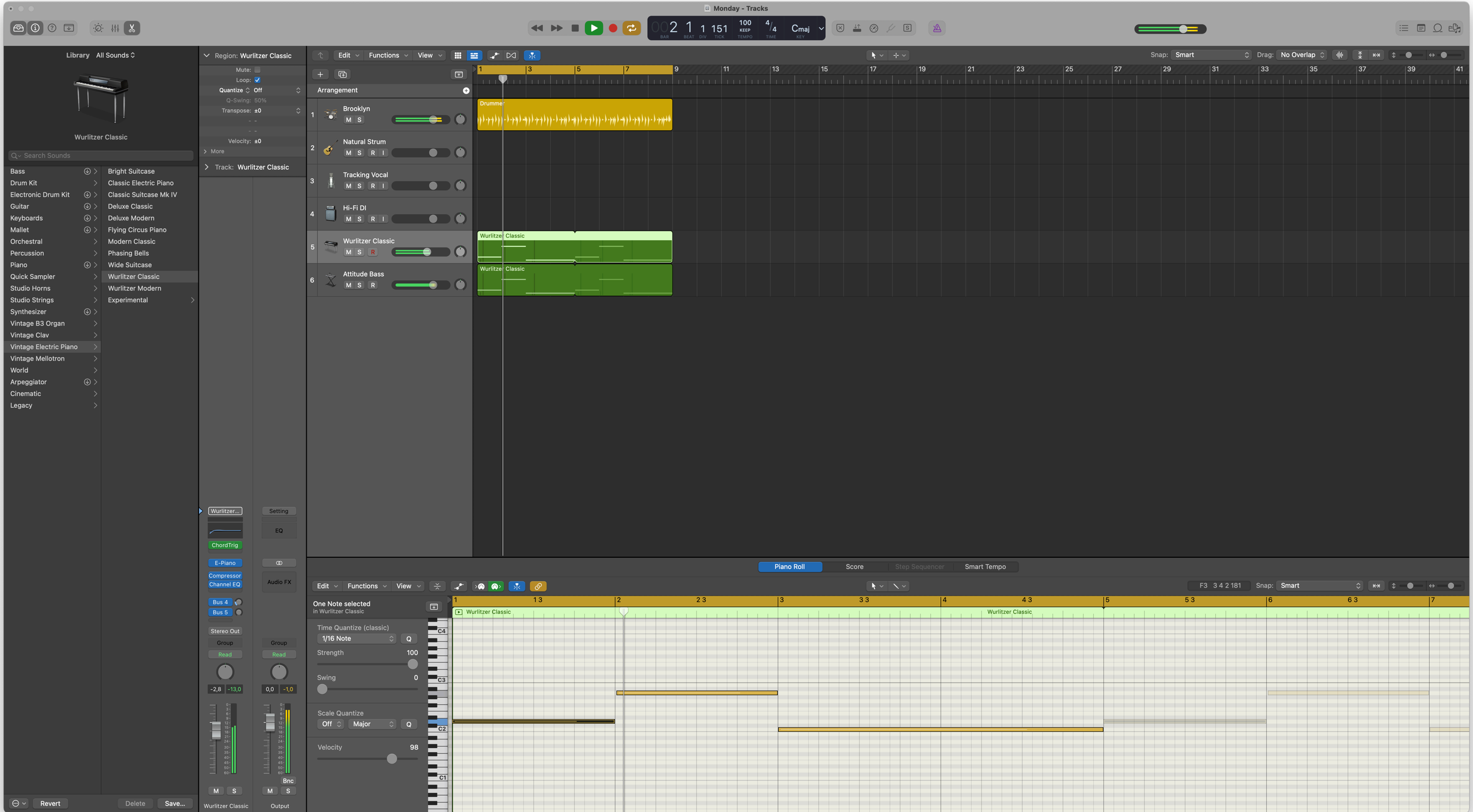

Více
Ozdobné noty
Notové hlavičky
Linky
Taktové čáry
Arpeggia a glissanda
Tremola
Text
Tempo
Dynamika
Prstoklady
Opakování a skoky
Hmatové diagramy
všem tvým skut - kům se di - ví - me; když se vše v své -
ne - be ze - mě za - stu - po - vé, ode všech jsi
Ne - be, zem, po-vě-tří, vo - dy pl - né jsou cti,
po - dle na - še - ho do - u - fá - ní. Kdo v tě do - u - fá
10
tě mě - ní, ty sám jsi bez pro - mě - ny.
na - zý - ván: Sva - tý, Sva - tý, Sva - tý Pán.
chvá - ly tvé, neb vše tvo - je dí - lo je.
Typ velocity: Posun
Velocity: 0
Vybrat
Tečka 1
Tečka 2
Tečka 3
Tečka 4
Nožička
Praporek
Trámec
Nepravidelná rytmická skupina

Music Production

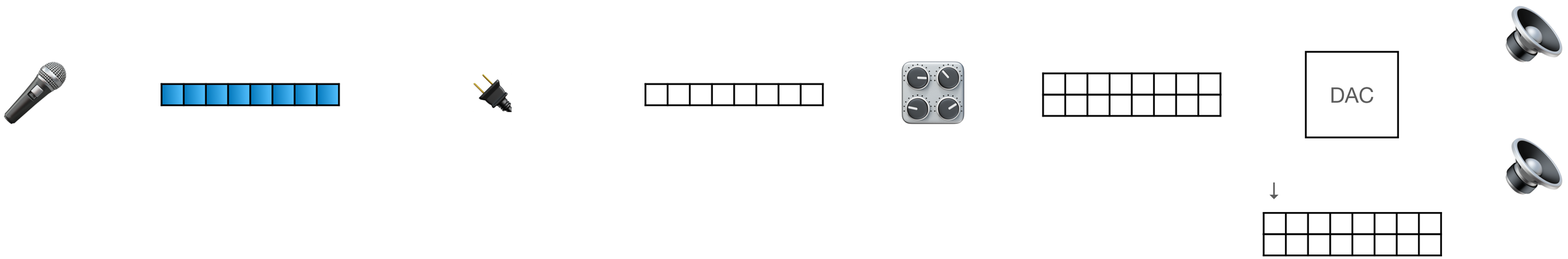


Audio Plug-ins



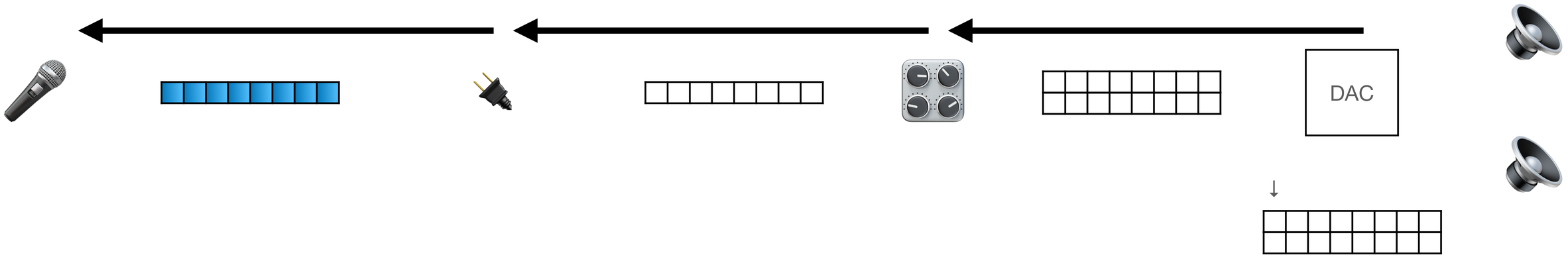

Monday - Tracks
Library All Sounds
Wurlitzer Classic
Search Sounds
Bass
Drum Kit
Electronic Drum Kit
Guitar
Keyboards
Mallet
Orchestral
Percussion
Piano
Quick Sampler
Studio Horns
Studio Strings
Synthesizer
Vintage B3 Organ
Vintage Clav
Vintage Electric Piano
Vintage Mellotron
World
Arpeggiator
Cinematic
Legacy
Bright Suitcase
Classic Electric Piano
Classic Suitcase Mk IV
Deluxe Classic
Deluxe Modern
Flying Circus Piano
Modern Classic
Phasing Bells
Wide Suitcase
Wurlitzer Classic
Wurlitzer Modern
Experimental
Revert
Delete
Save...
Region: Wurlitzer Classic
Mute:
Loop:
Quantize Off
Q-Swing: 50%
Transpose: ±0
Velocity: ±0
More
Track: Wurlitzer Classic
Edit
Functions
View
Snap: Smart
Drag: No Overlap
Arrangement
Brooklyn
Natural Strum
Tracking Vocal
Hi-Fi DI
Wurlitzer Classic
Attitude Bass
Drumme
Wurlitze Classic
Wurlitze Classic
Piano Roll
Score
Step Sequencer
Smart Tempo
One Note selected
in Wurlitzer Classic
Time Quantize (classic)
1/16 Note
Strength 100
Swing 0
Scale Quantize
Off Major
Velocity 98
Wurlitzer Classic
Wurlitzer Classic
Wurlitzer...
Setting
EQ
ChordTrig
E-Piano
Compressor
Channel EQ
Audio FX
Bus 4
Bus 5
Stereo Out
Group
Read
-2,8 -13,0
0,0 -1,0
Bnc
M S
Wurlitzer Classic
Output

Audio Graph



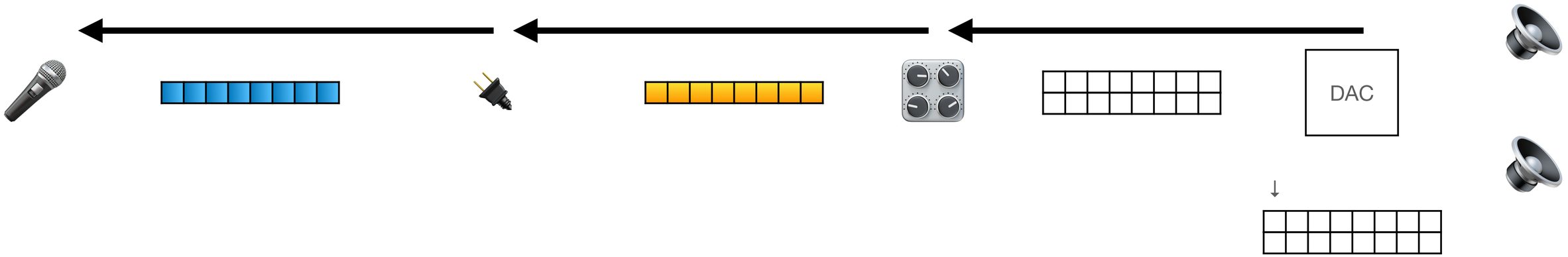
Audio Graph

Pull Model



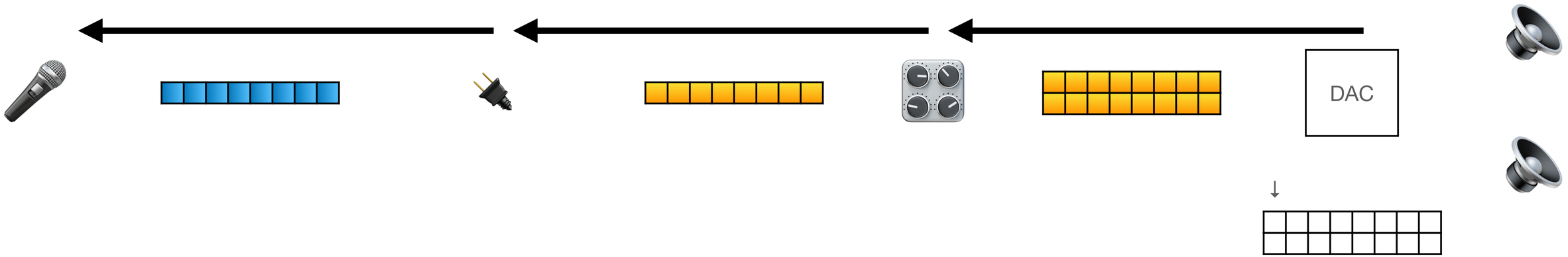
Audio Graph

Pull Model



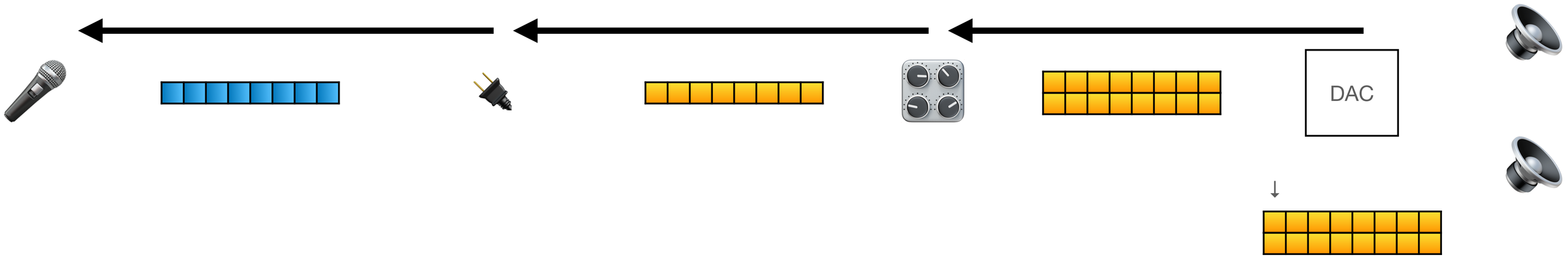
Audio Graph

Pull Model

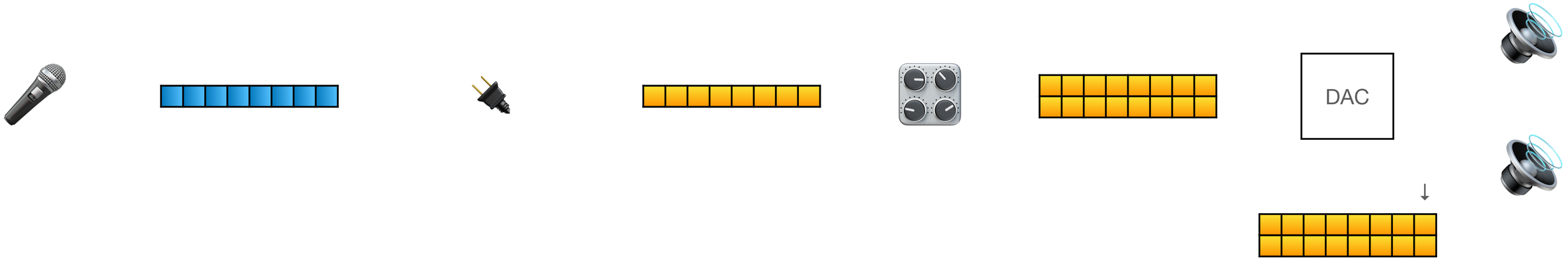


Audio Graph

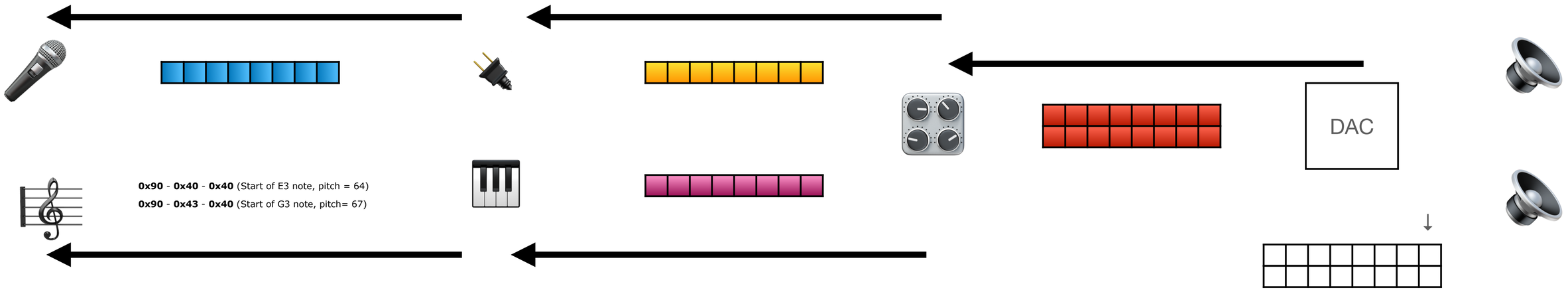
Pull Model



Audio Graph



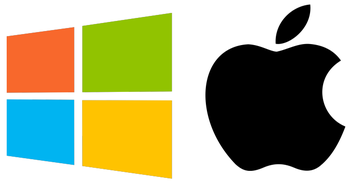
Audio Graph


DAC
0x90 - **0x40** - **0x40** (Start of E3 note, pitch = 64)
0x90 - **0x43** - **0x40** (Start of G3 note, pitch= 67)

Demo

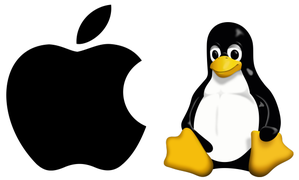

JUCE

Why JUCE?



each framework and operating system
offers a different API and programming model

Cross platform development



iPlug2

https://github.com/iPlug2/iPlug2

in-house frameworks

AudioKit

https://audiokit.io

Audio Callback

```
void SimpleDelayAudioProcessor::processBlock (juce::AudioBuffer<float>& buffer, juce::MidiBuffer& midiMessages)
{
    juce::ScopedNoDenormals noDenormals;
    auto totalNumInputChannels  = getTotalNumInputChannels();
    auto totalNumOutputChannels = getTotalNumOutputChannels();


    // This is the place where you'd normally do the guts of your plugin's
    // audio processing...
    // Make sure to reset the state if your inner loop is processing
    // the samples and the outer loop is handling the channels.
    // Alternatively, you can process the samples with the channels
    // interleaved by keeping the same state.
    for (int channel = 0; channel < totalNumInputChannels; ++channel)
    {
        auto* channelData = buffer.getWritePointer (channel);

        for (int i = 0; i < buffer.getNumSamples(); ++i)
        {
            channelData[i] *= 0.1f;
        }
    }
}
```

Real-time audio programming 101: time waits for nothing

http://www.rossbencina.com/code/real-time-audio-programming-101-time-waits-for-nothing

Anatomy of an audio plug-in

Audio Processor

User Interface

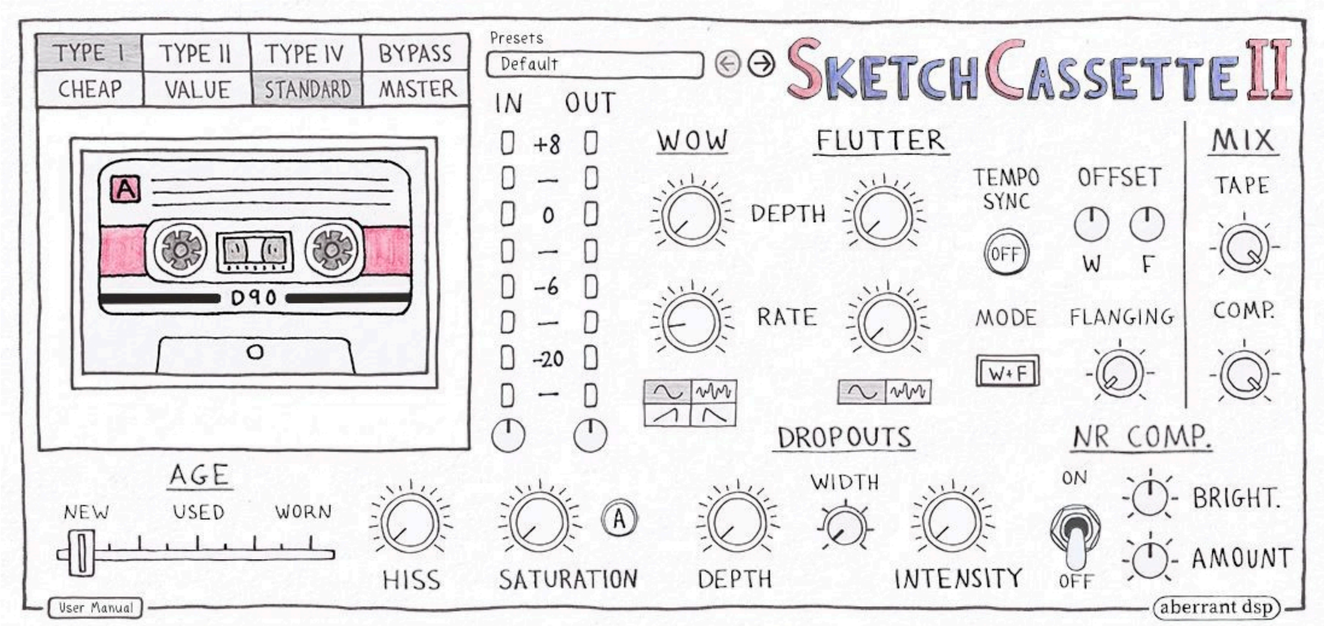
Parameters

Presets

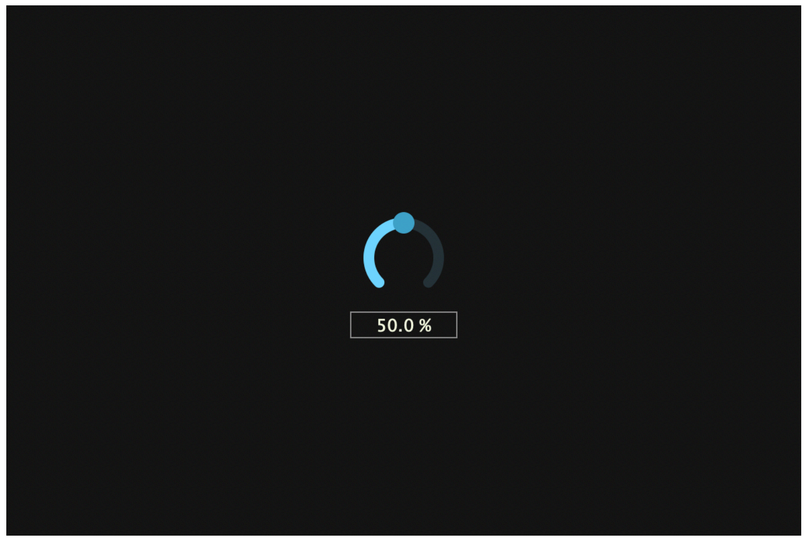
UI Components

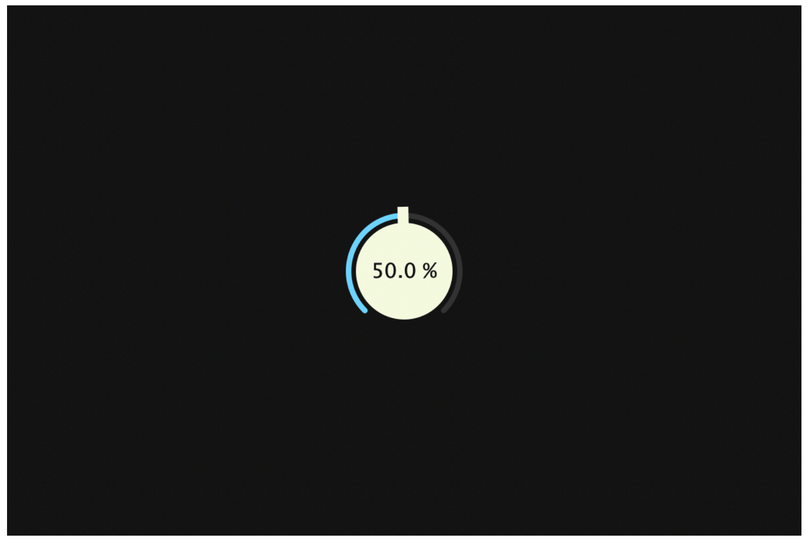
UI Components

UI Components



UI Components

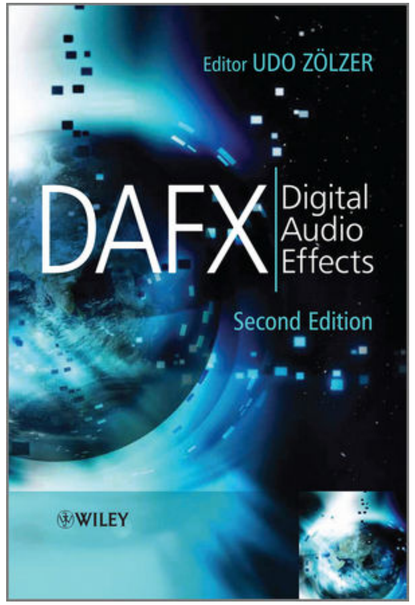




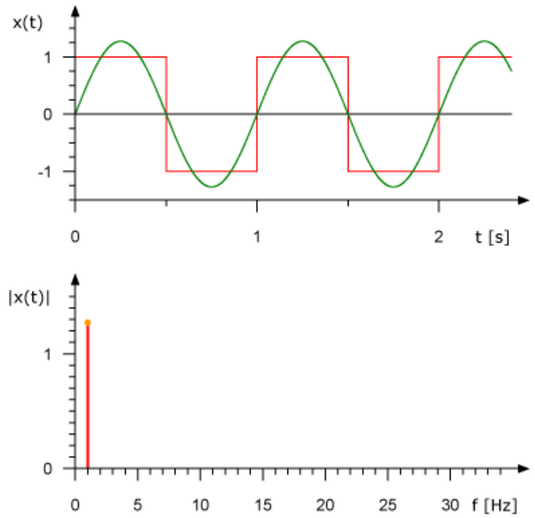
https://suzuki-kengo.dev/

Classification of audio effects

- Filters and delays (resampling)
- Modulators and demodulators
- Non-linear processing
- Spatial effects
- Time-segment processing
- Time-frequency processing
- Source-filter processing
- Adaptive effects processing
- Spectral processing
- Time and frequency warping
- Virtual analog effects
- Automatic mixing
- Source separation



Fourier transform

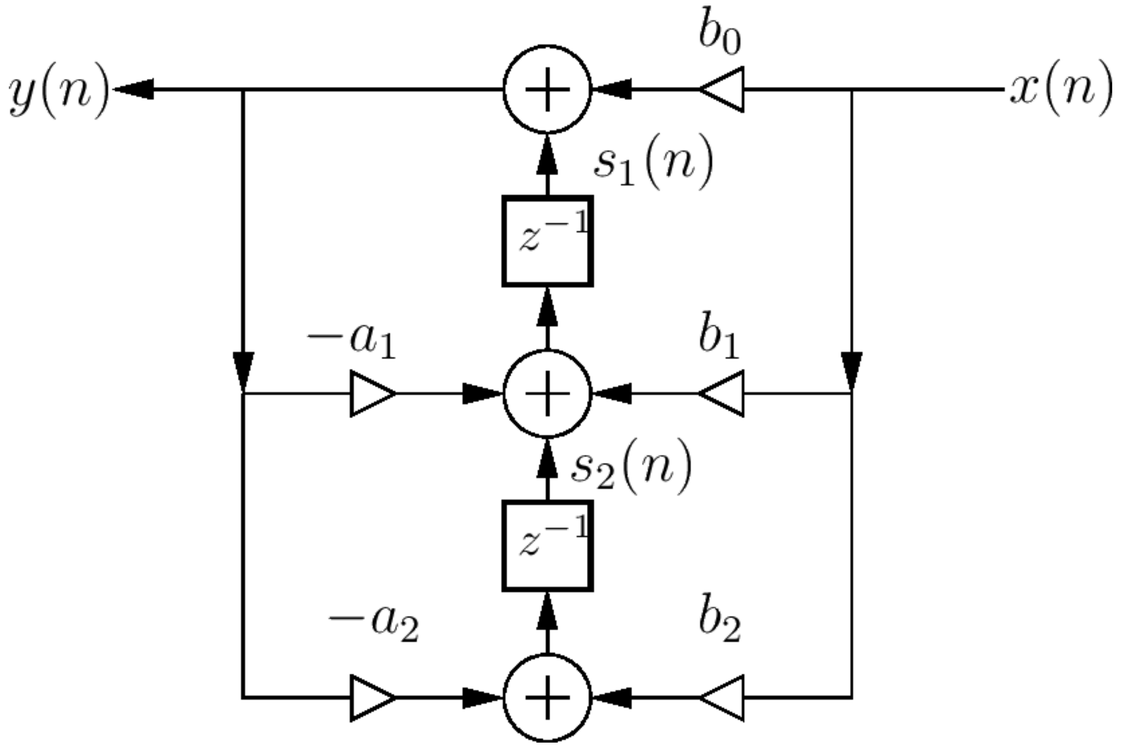


Prototype plug-ins

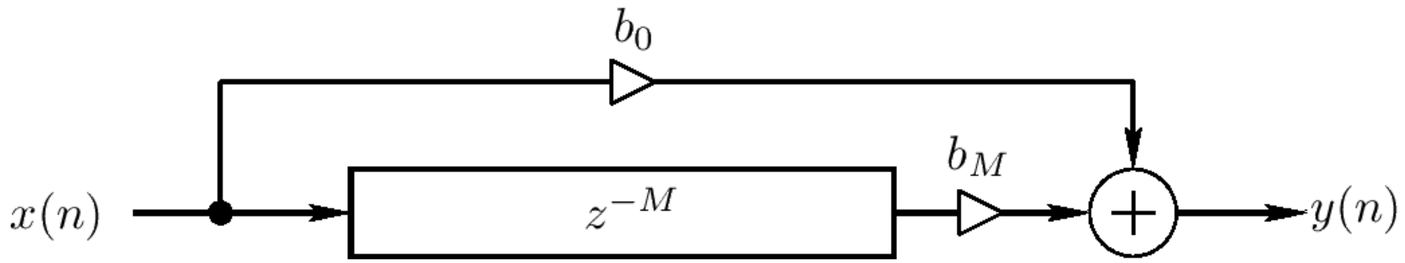
Waveshaping

Waveshaping #2

Block diagrams



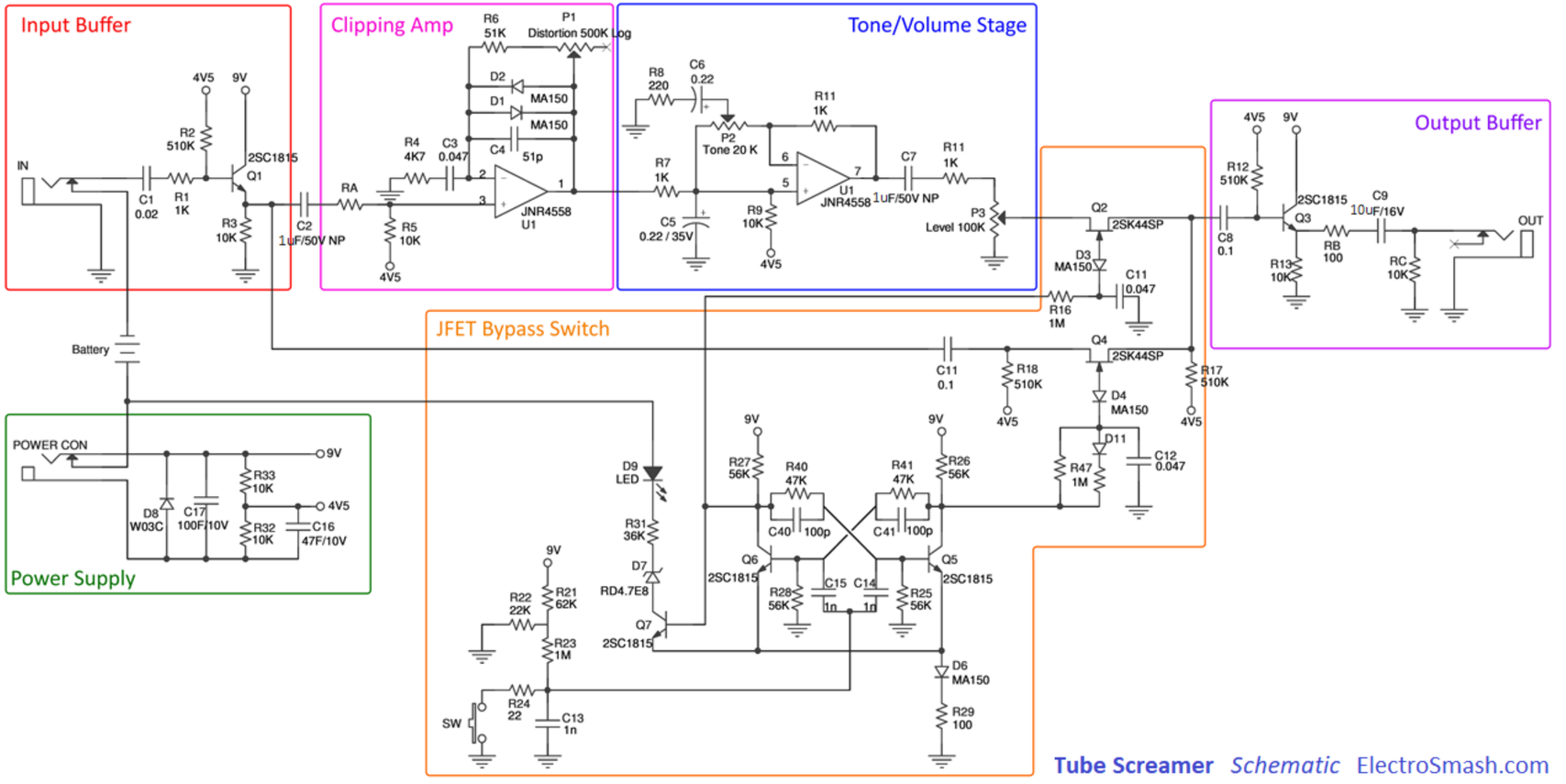
Feedforward Comb Filter



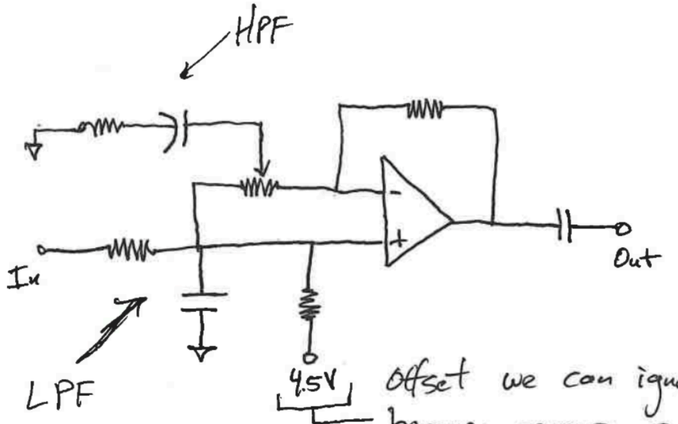
https://ccrma.stanford.edu/~jos/pasp/Feedforward_Comb_Filters.html

gen~ to JUCE

Analog Modelling



https://www.electrosmash.com/images/tech/tube-screamer

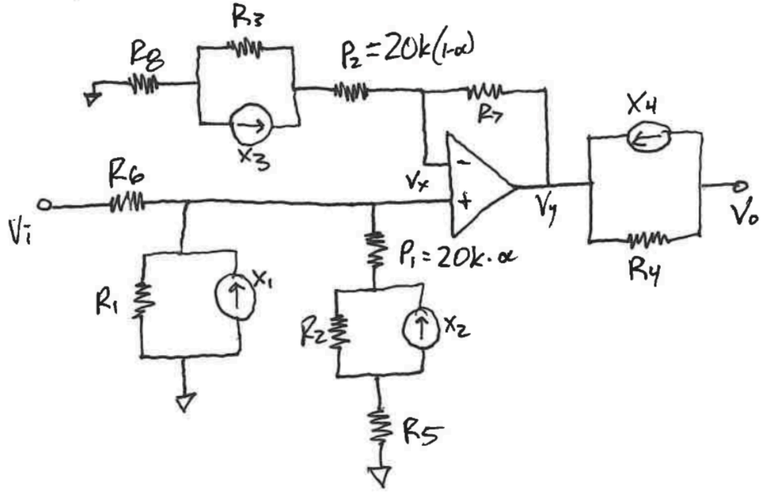


offset we can ignore because op-amp 0-9V

Potentiometer: Tone control single knob

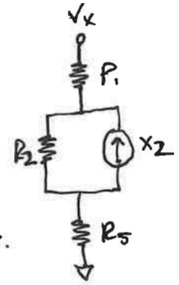
Resistor between "-" & "+" terminals

Assumption no current flow through/between terminals.



(41) (42) Start with sub-circuit

Want current "i" as function of V_x & X_1.



$V_x = V_{P1} + V_{R1} + V_{R5}$

same current "i" through each stage.

$i = \frac{V_{P1}}{P_1}$ $i = \frac{V_{R5}}{R_5}$ $i = \frac{V_{R2}}{R_2} - X_2$

Find each voltage as a function of V_{P1}

$\frac{V_{R5}}{R_5} = \frac{V_{P1}}{P_1}$ $V_{R5} = \frac{R_5}{P_1} \cdot V_{P1}$

$\frac{V_{R2}}{R_2} - X_1 = \frac{V_{P1}}{P_1}$ $\frac{V_{R1}}{R_2} = \frac{V_{P1}}{P_1} + X_1$ $V_{R2} = \frac{R_2}{P_1} V_{P1} + R_2 X_2$

Plug in above

$V_x = V_{P1} + \frac{R_2}{P_1} V_{P1} + R_2 X_2 + \frac{R_5}{P_1} V_{P1}$

$V_x - R_2 X_2 = V_{P1} \underbrace{\left(1 + \frac{P_2}{P_1} + \frac{R_5}{P_1}\right)}_{G_2}$

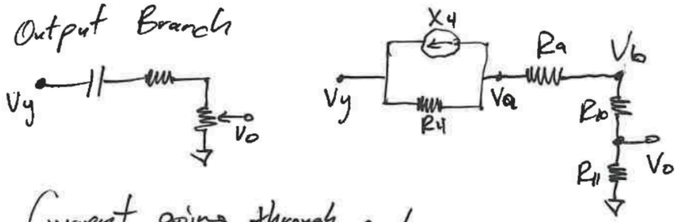
use this for current on this path.

$V_{P1} = \frac{V_x - R_2 X_2}{G_2}$ Therefore $i = \frac{V_{P1}}{P_1} = \frac{V_x - R_2 X_2}{G_2 \cdot P_2}$

Repeat same process

$i = \frac{V_x - R_3 X_3}{G_3 \cdot P_2}$ $G_3 = \left(1 + \frac{P_3}{P_2} + \frac{R_8}{P_2}\right)$

Output Branch



(43)

Current going through each component is equal.
Write everything as a function of V_o for difference equation.

$$\frac{V_b - V_o}{R_{10}} = \frac{V_o}{R_{11}} \qquad \frac{V_b}{R_{10}} = V_o\left(\frac{1}{R_{10}} + \frac{1}{R_{11}}\right) \qquad V_b = V_o\left(1 + \frac{R_{10}}{R_{11}}\right)$$

$$\frac{V_a - V_o}{R_9} = \frac{V_o}{R_{11}} \qquad V_a - V_b = \frac{R_9}{R_{11}} V_o \qquad V_a - \frac{R_9}{R_{11}} V_o = V_b$$

$$V_a - \frac{R_9}{R_{11}} V_o = V_o\left(1 + \frac{R_{10}}{R_{11}}\right) \qquad V_a = V_o\left(1 + \frac{R_{10}}{R_{11}} + \frac{R_9}{R_{11}}\right)$$

$$\frac{V_y - V_a}{R_4} - X_4 = \frac{V_o}{R_{11}} \qquad \frac{V_y - V_a}{R_4} = \frac{V_o}{R_{11}} + X_4$$

$$V_y - V_a = \frac{R_4}{R_{11}} V_o + R_4 X_4 \qquad V_y - \frac{R_4}{R_{11}} V_o - R_4 X_4 = V_a$$

$$V_y - \frac{R_4}{R_{11}} V_o - R_4 X_4 = V_o\left(1 + \frac{R_{10}}{R_{11}} + \frac{R_9}{R_{11}}\right)$$

$$V_y = V_o\underbrace{\left(1 + \frac{R_{10}}{R_{11}} + \frac{R_9}{R_{11}} + \frac{R_4}{R_{11}}\right)}_{G_o} + R_4 X_4$$

↖ save for later.

Now we need to connect branches together.

(44)

Node V_x "−" Terminal current flows $V_y \to V_x$

$$\frac{V_y - V_x}{R_7} = \frac{V_x}{G_3 P_2} - \frac{R_3}{G_3 P_2} X_3 \quad \text{(From previous page)}$$

Isolate V_y to plug in

$$V_y - V_x = \frac{R_7}{G_3 P_2} V_x - \frac{R_7 R_3}{G_3 P_2} X_3$$

$$V_y = V_x + \frac{R_7}{G_3 P_2} V_x - \frac{R_7 R_3}{G_3 P_2} X_3$$

↖ plug in V_y

$$V_o(G_o) + R_4 X_4 = V_x\left(1 + \frac{R_7}{G_3 P_2}\right) - \frac{R_3 R_7}{G_3 P_2} X_3$$

Isolate V_x

$$V_o \cdot G_o + R_4 X_4 + \frac{R_3 R_7}{G_3 P_2} X_3 = V_x\underbrace{\left(1 + \frac{R_7}{G_3 P_2}\right)}_{G_x}$$

$$V_x = \frac{G_o}{G_x} \cdot V_o + \frac{R_3 R_7}{G_x G_3 P_2} + \frac{R_4}{G_x} X_4$$

↖ save

now we need to connect input → output "+" terminal to "−"

Node V_x "+" terminal

$$\frac{V_i - V_x}{R_6} = \frac{V_x}{R_1} - X_1 + \frac{V_x}{G_2 P_1} - \frac{R_2 X_2}{G_2 P_1}$$ (from previous page)

Isolate V_x

$$\frac{V_i}{R_6} + X_1 + \frac{R_2 X_2}{G_2 P_1} = V_x \underbrace{\left(\frac{1}{R_1} + \frac{1}{G_2 P_1} + \frac{1}{R_6}\right)}_{G_2}$$

$$V_x = \frac{V_i}{R_6 G_2} + \frac{X_1}{G_2} + \frac{R_2 X_2}{G_2 G_2 P_1}$$ set equal to other V_x

$$\frac{V_i}{R_6 G_2} + \frac{X_1}{G_2} + \frac{R_2 X_2}{G_2 G_2 P_1} = V_o \frac{G_o}{G_x} + \frac{R_3 R_7}{G_x G_3 P_2} X_3 + \frac{R_4}{G_x} X_4$$

solve for V_o for difference equation

$$\frac{V_i}{R_6 G_2} + \frac{X_1}{G_2} + \frac{R_2}{G_2 G_2 P_1} X_2 - \frac{R_3 R_7}{G_x G_3 P_2} X_3 - \frac{R_4}{G_x} X_4 = V_o \frac{G_o}{G_x}$$

$$V_o = \frac{G_x}{G_o R_6 G_2} V_i + \frac{G_x}{G_o G_2} X_1 + \frac{G_x R_2}{G_o G_2 G_2 P_1} X_2 - \frac{R_3 R_7}{G_o G_3 P_2} X_3 - \frac{R_4}{G_o} X_4$$

$$b_0 = \frac{G_x}{G_o R_6 G_2} \qquad b_1 = \frac{G_x}{G_o G_2}$$

$$b_2 = \frac{G_x R_2}{G_o G_2 G_2 P_1} \qquad b_3 = \frac{-R_3 R_7}{G_o G_3 P_2} \qquad b_4 = \frac{-R_4}{G_o}$$

State update equations

$$X_1[n] = \frac{2}{R_1} \cdot V_x - X_1[n-1]$$

need to calculate.

$$X_2[n] = \frac{2}{R_2} V_{R2} - X_2[n-1]$$ more complicated.

Let's find V_{R2} as function of V_x

Use $V_x = V_{P1} + V_{R2} + V_{R5}$

Write all terms with V_{R2}, use currents equal

$$\frac{V_{P1}}{P_1} = \frac{V_{R2}}{R_2} - X_2 \qquad V_{P1} = \frac{P_1 V_{R2}}{R_2} - P_1 X_2$$

$$\frac{V_{R5}}{R_5} = \frac{V_{R2}}{R_2} - X_2 \qquad V_{R5} = \frac{R_5}{R_2} V_{R2} - X_2 R_5$$

now plug in

$$V_x = \frac{P_1 V_{R2}}{R_2} - P_1 X_2 + V_{R2} + \frac{R_5}{R_2} V_{R2} - X_2 R_5$$

Solve for V_{R2}

$$V_x + P_1 X_2 + R_5 X_2 = V_{R2} \underbrace{\left(\frac{P_1}{R_2} + 1 + \frac{R_5}{R_2}\right)}_{G_R}$$

$$V_{R2} = \frac{V_x}{G_R} + \left(\frac{P_1 + R_5}{G_R}\right) X_2$$

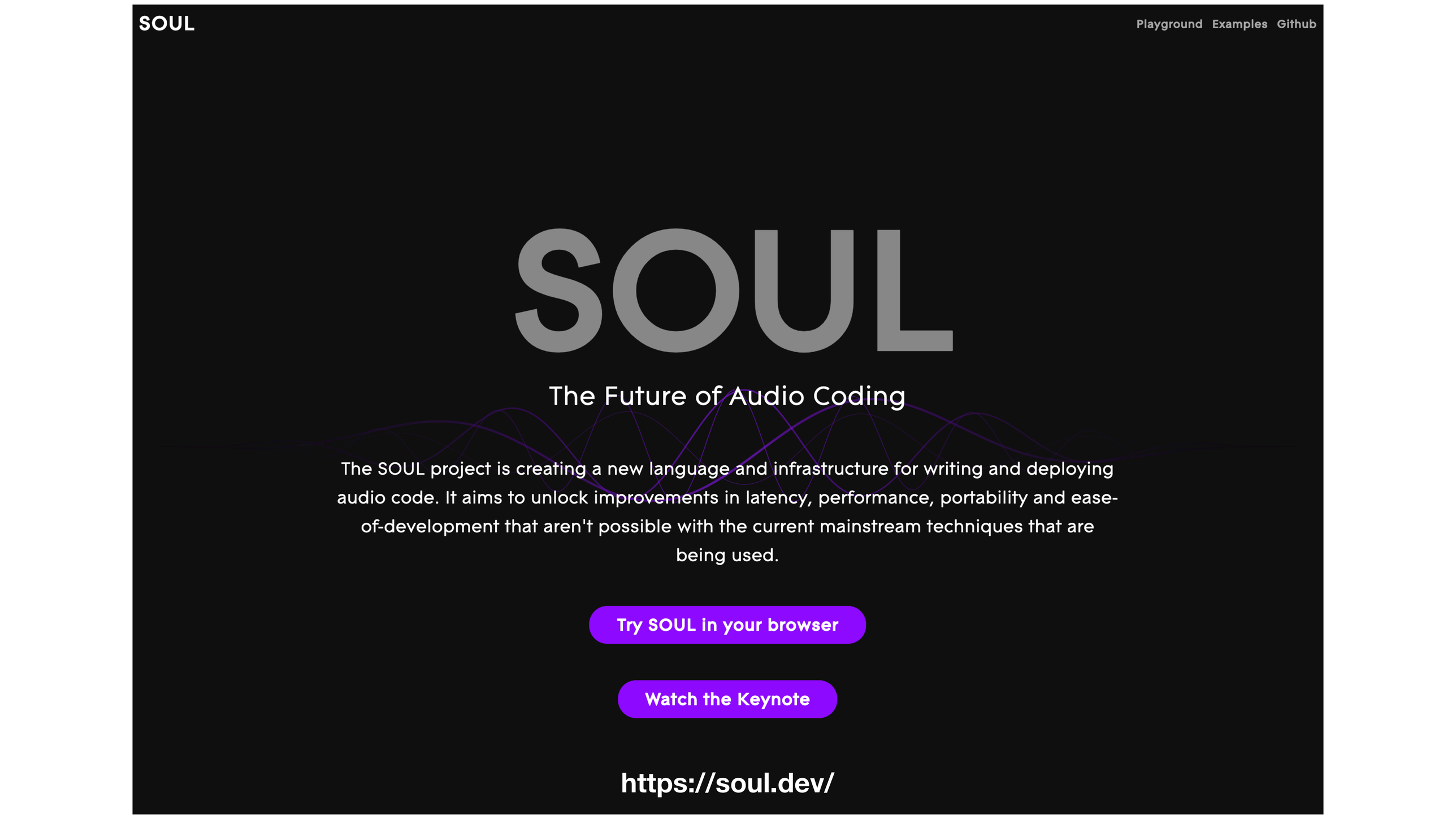
Plug into original state equation

$$X_2[n] = \frac{2}{R_2}\left(\frac{V_x}{G_R} + \left(\frac{P_1 + R_5}{G_R}\right) X_2[n-1]\right) - X_2[n-1]$$


12-8 Electric Arpeggio 04
M S R
12-8 Electric Arpeggio 04
Drag Apple Loops here to
create instrument tracks.
12-8 Electric Arpeggio 04
Manual
Compare
Editor
L R
Couple
Drive
Tone
Output
Tube Screamer
TS-1
TSPedal
12-8 Electric Arpeggio 04
Manual
Compare
Copy
Paste
Undo
Redo
View:
Editor
L R
Couple
Presets
A B A▸B
Routing
MONO
SOLO
Copy to
HIDE METERS AND STATS
SPAN
Spectrum
HOLD
Underlay
Mode
DEFAULT
Out
Statistics
RMS -21.6
Reset
Metering
DBFS
TP
Correlation Meter
L-R 0.0
Max Crest Factor
2.1
True Peak Clippings
0
True Peak
-12.5
SPAN

hack audio

Alternatives?


SOUL
The Future of Audio Coding
The SOUL project is creating a new language and infrastructure for writing and deploying audio code. It aims to unlock improvements in latency, performance, portability and ease-of-development that aren't possible with the current mainstream techniques that are being used.
Try SOUL in your browser
Watch the Keynote
https://soul.dev/

FAUST

Functional Programming Language for Real Time Signal Processing

GitHub | Quick Start | Try It Online!

What is Faust?

Faust (Functional Audio Stream) is a functional programming language for sound synthesis and audio processing with a strong focus on the design of synthesizers, musical instruments, audio effects, etc. Faust targets high-performance signal processing applications and audio plug-ins for a variety of platforms and standards.

The core component of Faust is its compiler. It allows to "translate" any Faust digital signal processing (DSP) specification to a wide range of non-domain specific languages such as C++, C, LLVM bit code, WebAssembly, Rust, etc. In this regard, Faust can be seen as an alternative to C++ but is much simpler and intuitive to learn.

Thanks to a wrapping system called "architectures," codes generated by Faust can be easily compiled into a wide variety of objects ranging from audio plug-ins to standalone applications or smartphone and web apps, etc.

The Faust Web IDE

https://faustide.grame.fr

https://faust.grame.fr/

It's time to simplify the way we write audio software

Elementary is a modern platform for writing high performance audio software that helps you build quickly and ship confidently.

Get Started | Try in your Browser

FEATURES

A fundamentally different way to write audio software

Elementary brings the functional, reactive programming model to a world dominated by imperative, object oriented code. That means faster development, simpler code, and unimpaired creativity.

Declarative

Elementary makes it simple to create

Functional

Build highly reusable components that

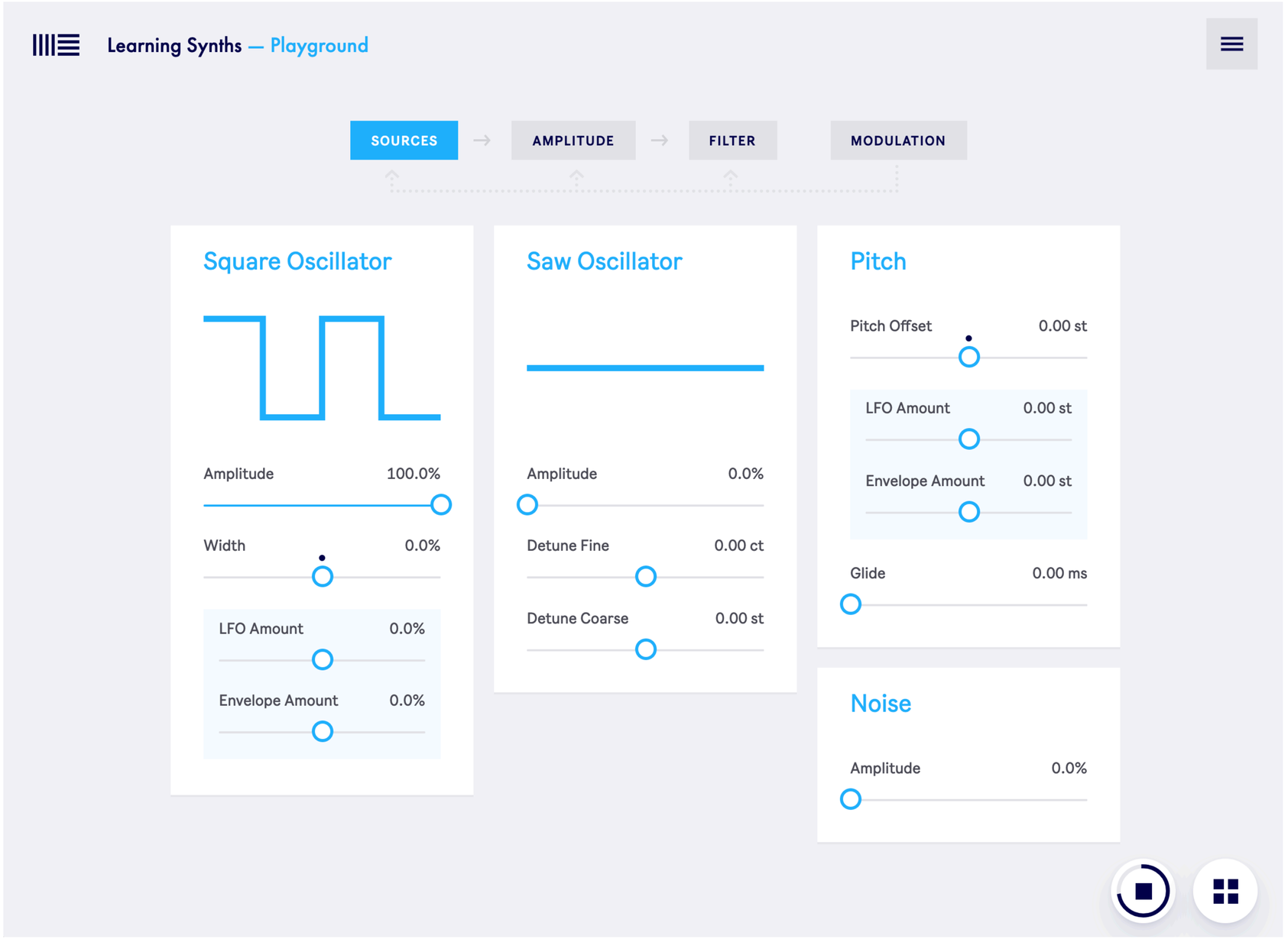
Accessible

One language, one paradigm, no hidden

https://www.elementary.audio/

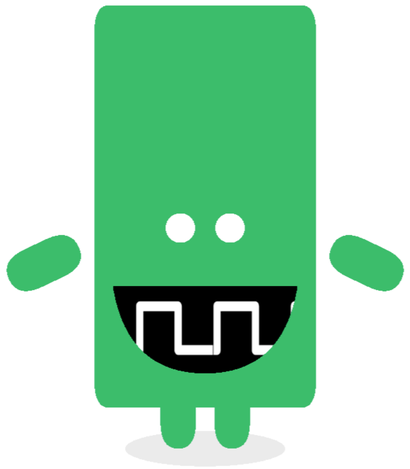
Web Audio API

https://developer.mozilla.org/en-US/docs/Web/API/Web_Audio_API



https://learningsynths.ableton.com/

OSCILLATORS



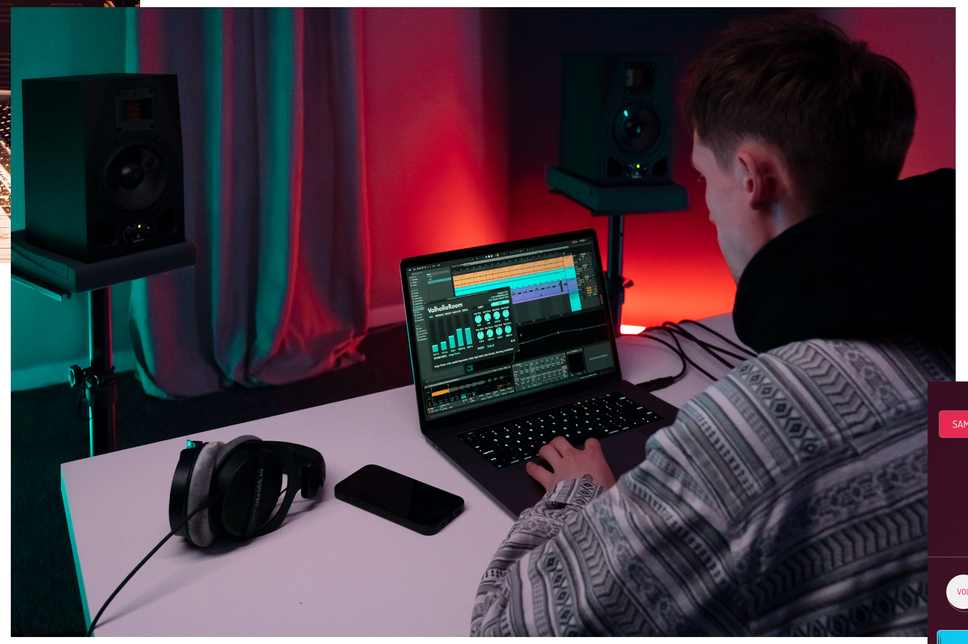
```
oscillator.type = 'square';

oscillator.frequency.value = 1113;
```

https://musiclab.chromeexperiments.com/

Web Assembly

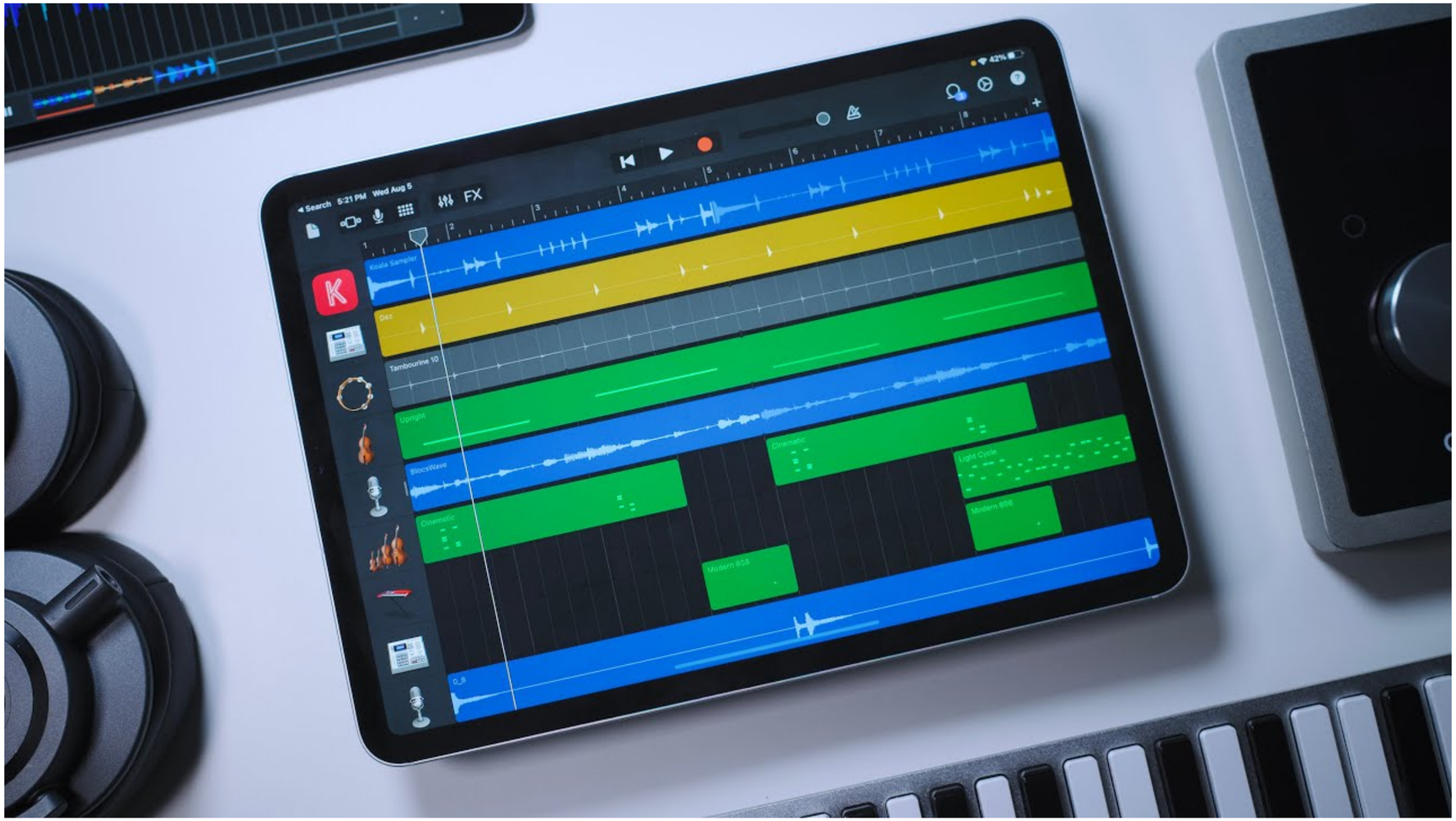
https://virtualcz.io/




SAMPLE
SEQUENCE
PERFORM
FX
MIC
VOL
PITCH
PAN
EDIT
DELETE
A
B
C
D
Guitar
Bass
GarageBand
EQ
REVERB
TREMOLO
GAIN
BASS
MIDS
TREBLE
LEVEL
DEPTH
SPEED
PRESENCE
MASTER
OUTPUT
UNDO
REDO
FX
PLAY
OSCILLATOR BANK
MIXER
MODIFIERS
PITCH
MOD.

Mobile music producers

Henny Tha Buziness



Steve Lacy

Steve Lacy's Demo - EP E

Steve Lacy

R&B/SOUL · 2017

Preview

	Looks	1:31	•••
2	Ryd	2:20	•••
★ 3	Dark Red	2:53	•••
4	Thangs	1:50	•••
5	Haterlovin E	1:56	•••
6	Some	3:00	•••

https://music.apple.com/us/album/steve-lacys-demo-ep/1207820151

Resources

https://www.youtube.com/c/TheAudioProgrammer

https://www.theaudioprogrammer.com/community/discord

https://audio.dev/

https://www.hackaudio.com/

https://www.programmingformusicians.com/

Thank you