MAP Final Project: Digital Subharmonicon

Sara Adkins, 210077786

April 2022

1 Abstract

A digital replica of the Moog Subharmonicon synthesizer is implemented on the Bela platform. The synthesizer features two voltage controlled oscillators with controllable subharmonics that share a resonant low pass filter. Two step sequencers control the frequency of the oscillators, and the four rhythm generators allow the creation of polyrhythms by generating integer divisions of the main sequencer tempo. Aliasing effects in the digital system are reduced using the differentiated parabolic waveform method, though not eliminated completely. Despite this, the digital subharmonicon has the potential to be a low cost, open source alternative to the expensive Moog original.

2 Introduction and Background

The Subharmonicon is an analog synthesizer that uses integer ratios to control the oscillators and rhythm generation. The basis of the sound generation is two voltage controlled oscillators, VCO1 and VCO2, that can each be set between 262 and 4182 Hz(C4 to C8). Each VCO also has two subharmonic oscillators, that can be set to produce an integer division(between 1 and 16) of the base VCO frequency. This results in a total of 6 oscillators, each with its own amplitude control knob. The oscillators are passed through a shared 4-pole resonant low-pass filter, and two envelope generators that control the amplitude and filter cutoff. The full control interface is shown in figure 1.

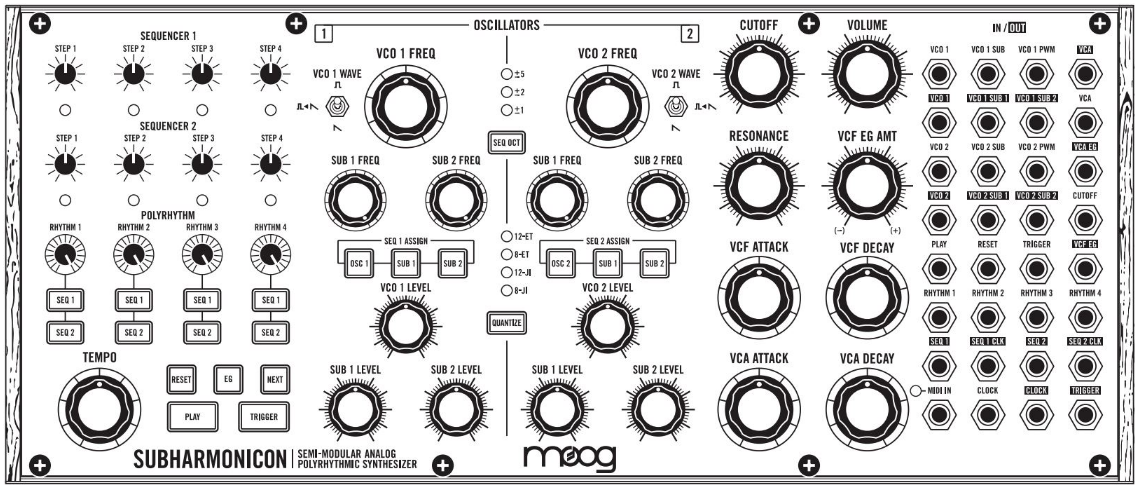

Figure 1: Subharmonicon interface, taken from Moog manual

The defining feature of the Subharmonicon is it's rhythm generators, which make it possible to play polyrhythms. Two step sequencers allow the user to create 4 note sequences by modulating the frequency of either VCO 1 or 2. The speed at which the oscillator is triggered to begin a new note is controlled by the the global tempo knob as a default. However, by turning up one of the rhythm knobs and assigning it to a sequence, a note increment can be triggered by an integer division of the tempo between 1 and 16. Assigning multiple rhythm generators to the same sequence creates a polyrhythm, where the different tempo divisions combine to form a longer and more complex rhythm that will eventually repeat due to the shared source tempo. An example is shown in figure 2, where a T/2 and T/3 rhythm combination repeats every 6 beats(the least common multiple of the divisions). Combining more rhythm generators and using prime divisions will create longer loops.

T	x	x	x	x	x	x	x
T/2	x	-	x	-	x	-	x
T/3	x	-	-	x	-	-	x

Figure 2: Example polyrhythm cycle that repeats every 6 beats

A digital implementation of the Subharmonicon presents several challenges. First, the sawtooth and square waveforms used by the synthesizer cause aliasing that is especially audible at higher frequencies. This is because an infinite number of harmonics are needed to generate the sharp cutoffs in these waveforms, but in a digital system we can only accurately render up the half the sampling rate, the Nyquist frequency. Second, the analog resonant low-pass filter that is a staple of any Moog synthesizer is difficult to realize accurately in the digital domain, because the delay line added to produce the resonance couples the cutoff and resonance controls. Finally, the sheer amount of controls on the synthesizer and the complicated routing of rhythms, sequences, envelopes and oscillators make it a substantial programming effort. Section 3 will address these challenges and outline the details of the digital Subharmonicon design.

3 Design

The main features of the Subharmonicon can be broken down into oscillators, envelopes, step sequencers and rhythm generators. The design of each of these features, how they work together, and how they are controlled is covered in this section.

3.1 Subharmonic Oscillators

The Oscillator class represents a base oscillator and its associated subharmonic oscillators. Through this class the wave type, base frequency, subharmonic divisions, and relative amplitudes are controlled. When the base frequency is updated through the setFrequency() method, the subharmonic frequencies are automatically updated as well to maintain their integer harmonic relationship. There is also an option to quantize the oscillator frequency to a chromatic, major, minor or pentatonic scale. This quantization makes it much easier to tune the two VCOs to each other and make melodic sequences.

The aliasing effects of the sawtooth and square oscillators are mitigated using the Differentiated Parabolic Waveform(DPW) method introduced in [1]. The SawAntiAlias class generates a sawtooth waveform by ramping from -1 to 1, then squares the signal to create a parabolic waveform. Finally, a 1st order difference filter is applied and the result is scaled to bring it back to the original amplitude. The transfer function of the difference filter is $H(z) = 1 - z^{-1}$, so the equation $y[n] = x[n] - x[n-1]$ is applied to each sample. The resulting waveform shown in figure 3 is very close to the original sawtooth but with slight smoothing at the jump points from 1.0 to -1.0. This results in reduced aliasing effects especially at low frequencies. The DPW technique is extended to produce square waves in the SquareAntiAlias class by subtracting one DPW sawtooth from another with a 50% phase shift. The anti-aliasing performance is evaluated in section 4.

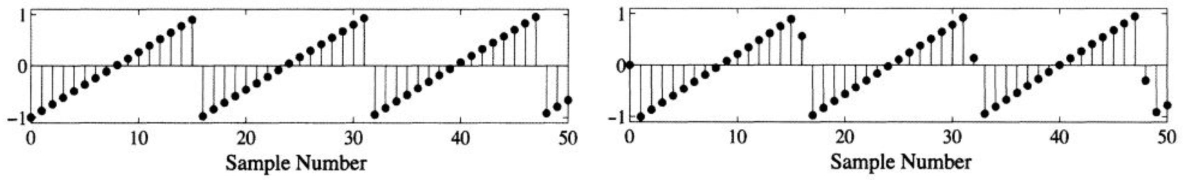

Figure 3: original sawtooth waveform(left) and output transformed by the DPW algorithm(right). Graphic from [1].

3.2 Resonant Filter and Envelopes

The two VCOs share an amplitude envelope generator, resonant low-pass filter, and filter cut-off envelope generator. The resonant filter is the same implementation as assignment 1, and was moved to the ResFilter class. The two envelope generators are instances of the ASR class, which is an altered version of the ADSR example class that removes the decay state since it is not present in the Subharmonicon. The sustain state in the ASR envelope can also be toggled on and off. This is needed because the Subharmonicon does not have a sustain when playing a sequence, but sustain is needed when holding down the trigger button. This allows the user to hear a sustained version of the sound without any sequences or rhythms, which is useful for finetuning oscillator parameters.

A noteable feature of the Subharmonicon is an envelope can not be retriggered while in the attack phase. This means that triggers from the sequence generators are skipped if received when the envelope is still ramping up, and allows the attack parameter to act as another control for the overall rhythm of the system. The digital version implements this feature by ignoring all calls to the trigger() function while in the ATTACK state.

The filter envelope has a VCF EG AMT knob, which allows the user to control the depth of the cutoff frequency change. The Subharmonicon manual does not specify the exact range of this knob, only that positive values should open the filter on the attack stage and close it on the decay, while negative values behave the opposite. The digital implementation allows for a depth ranging from -10,000Hz to 10,000Hz, with the final cutoff value clipped between 20Hz and 20,000Hz.

3.3 Sequences and Rhythm Generation

The polyrhythms produced by the Subharmonicon are crafted from the tempo knob, two step sequencers, and four rhythm generators. The tempo knob defines a base tempo T, which ranges from 0.333Hz to 3,000Hz in the analog version, but is limited to 0.333Hz to 300Hz in the digital implementation. In the synth's default setting, it cycles through the four steps of SEQ 1 at a rate of T steps per second and triggers VCO 1 only.

The two step sequencers on the Subharmonicon, SEQ 1 and SEQ 2, are tied to VCO 1 and VCO 2 respectively. Each step knob alters the frequency of it's associated VCO by +/-1 octave. This range can be adjusted to +/-2 or +/-5 octaves using the RANGE button, and is shared by both VCOs. Each sequence can also be set up to control the base VCO frequency or either of the subharmonic ratios. As demonstrated in figure 4, changing the base frequency updates the subharmonic frequencies automatically, while pointing the sequence at a subharmonic alters only that subharmonic ratio. Each sequence is represented as an instance of the Sequence class, which keep track of it's target, range, step values and current step. A sequence is progressed to the next step by calling the beat() method.

Each of the 4 rhythm generator knobs divides the base tempo T by an integer value between 1 and 16. For instance, a knob value of 3 at $T = 12Hz$ would result in a new tempo $T_1 = 4Hz$. A rhythm generator can be targeted at SEQ 1, SEQ 2 or both. At the start of each beat, the target sequence is incremented to the next step. So at tempo $T_1 = 4Hz$, this results in the 4-step target sequence repeating once every second. Since the rhythm generators are always triggering on multiples of the base beat, we can think of

Osc	Osc Val	Orig F	Range	Step	Target	Final F
VCO	600	600	+/-1	1.0	VCO	1200
SUB1	2	300	+/-1	1.0	VCO	600
SUB2	3	200	+/-1	1.0	VCO	400
VCO	600	600	+/-1	-1.0	SUB1	600
SUB1	2	300	+/-1	-1.0	SUB1	300
SUB2	3	200	+/-1	-1.0	SUB1	300

Figure 4: Expected oscillator frequencies from different sequencer targets. In the top 3 lines, a change of +1 octave targeted at VCO alters the frequencies of all 3 oscillators to maintain the subharmonic ratios. In the bottom 3 lines, a change of -1 octave targeted at SUB1 alters only that subharmonic ratio.

each generator as a measure of X beats and represent their states using two global arrays. gRhythmDivs[] stores the number of beats each generator is set to, while gRhythmCounters[] stores the current beat. The transition diagram for tracking rhythms and triggering sequences is shown in figure 5. By targeting one sequence with multiple rhythm generators set to different divisions, a polyrhythm is created.

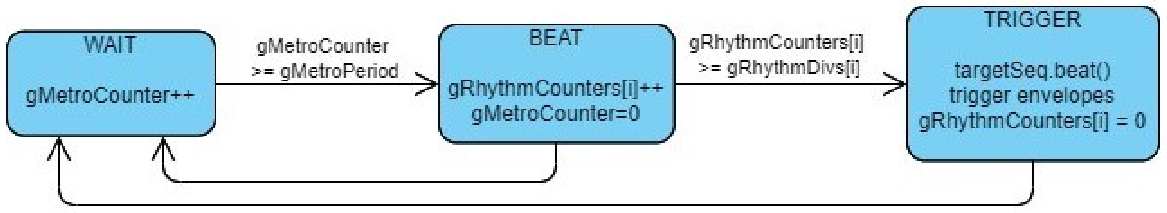

Figure 5: STM for tracking rhythm generators and triggering sequences. Cycles once per audio sample

3.4 Parameter Control

The Subharmonicon has 33 control knobs, 21 buttons and 2 switches(not to mention the patch bay, which is not implemented in this digital version). This is way more controls than the available analog and digital inputs on Bela, so many of the controls were relegated to the Javascript GUI. When deciding which knobs would be controlled with the 8 analog inputs on Bela, priority was given to parameters that had immediate effect and were continuous. These parameters were: VCO 1 FREQ, VCO 2 FREQ, VCO 1 LEVEL, VCO 2 LEVEL, CUTOFF, RESONANCE, VOLUME and TEMPO. Each was controlled with a pontentiometer and sampled at 22050Hz, aligned with every other audio sample. Two buttons, connected to Bela as digital inputs with $10k\Omega$ resistors, acted as controls for playback. The state transition diagram for the different playback modes is shown in figure 6.

The remaining parameters for the envelopes, sequencers, rhythm generators and subharmonics are controllable from the Bela GUI. All of these parameters are passed as a buffer from Javascript to the audio application, and read once per audio block. They update much more slowly than the parameters with hardware control, since the GUI refresh rate is much slower than the 22050Hz of the analog inputs.

4 Evaluation

This section discusses where the digital replica differs from the analog original, and evaluates the performance of the DPW method for reducing aliasing effects. The functionality of the subharmonic waveform generator and the shared hardware/software interface are also discussed.

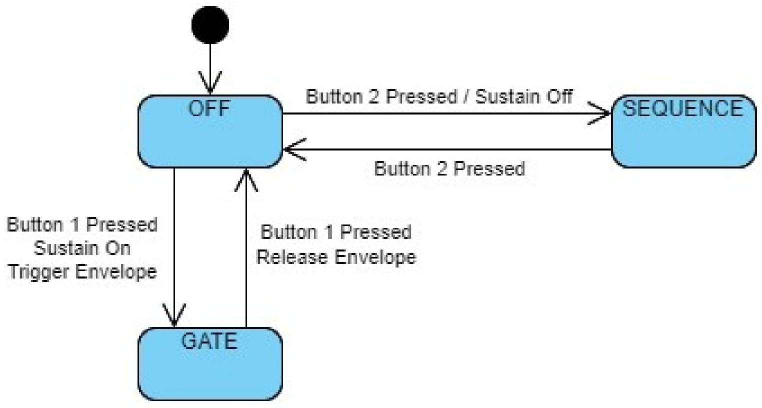

Figure 6: STM for Subharmonicon playback states. In the GATE state button presses open and close the envelope. In the SEQUENCE state button presses start and stop the metronome counter, and the envelopes are triggered by the rhythm generators

4.1 Feature Comparison

The digital Subharmonicon implements the majority of the features on the original synthesizer, with a few notable exceptions. First is the patchbay, which allows the user to route signals from one area of the synth to control the parameter of another. The analog Subharmonicon also has a 3rd waveform setting for the oscillators, where the VCO outputs a square wave and the subharmonics output sawtooth waves that apply pulse width modulation to the square wave. Finally, the analog Subharmonicon has a few additional controls for manually stepping through the sequencer steps that are not implemented.

Other than those 3 exceptions, all of the other knobs and buttons on the Subharmonicon are fully implemented in the digital implementation as specified in the user manual. All of the knob ranges specified in the manual are upheld, with the exception of the tempo knob. In the analog synth tempos of up to 3000Hz are supported, but this wide range was very difficult to precisely control with the potentiometer so the maximum was reduced to 300Hz. The ranges for the RES and VCF EG AMT knobs were not specified in the manual and were set to [0.0, 1.0] and [-10000, 10000] respectively. Finally, the digital Subharmonicon extends the quantization options for the oscillators, adding minor and pentatonic scale options on top of the existing chromatic and major ones.

4.2 Aliasing Analysis

The trivial digital implementations of square and sawtooth waves cause aliasing, because an infinite number of sine waves are needed to represent the sharp corners in these signals. Due to sampling, all of these sines above the Nyquist frequency are aliased into range, resulting in artifacts. One way around this is to build up a wavetable that approximates the target waveform by only including harmonics below the Nyquist frequency. However this solution does not work well when there is a wide range of target frequencies as is the case with the Subharmonicon. At low frequencies the harmonics will be cut off before Nyquist, and at high frequencies the upper harmonics will still alias.

The alternate approach taken in this digital implementation was to suppress aliasing using the DPW method detailed in section 3.1. The results of applying this method to a 1.1kHz sawtooth with a fully open filter are shown in figure 7. The DPW algorithm was highly effective at removing most of the aliasing and the audio sounds much cleaner. However, at the highest frequencies there is still some visible aliasing that approaches the level of the actual signal.

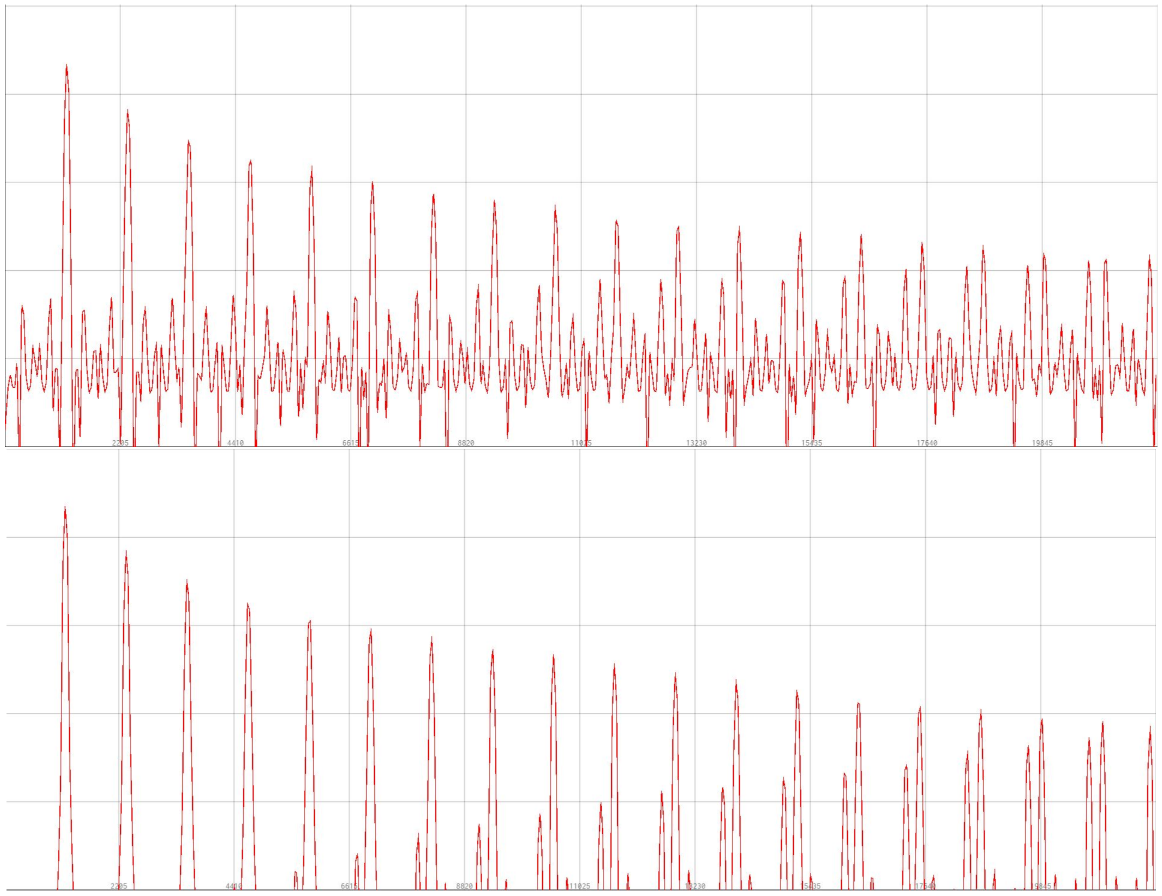

Figure 7: FFT of 1.1kHz sawtooth with 20kHz cutoff. Trivial sawtooth(top) and DPW sawtooth(bottom)

Even with the suppression from the DPW method, aliasing is also a problem when adjusting the resonance parameter at high cutoff frequencies. As shown in figure 8, maxing out the resonance when the cutoff is close to the maximum of 20kHz amplifies the suppressed aliasing and causes loud buzzing artifacts. High resonance at lower cutoffs does not produce such artifacts. For most parameter settings the DPW anti-aliasing method does a good job at preserving signal quality, but when approaching the upper limits of oscillator frequency, cutoff frequency and resonance the quality degradation compared to the original analog synthesizer is very apparent.

4.3 Subharmonic Waveform

The two subharmonic oscillators for each VCO are restricted to integer divisions of the VCO frequency, and match the VCO wave type(saw or square). The functionality of the subharmonics is demonstrated in figure 9. With SUB1 tuned to 2, the FFT shows a new waveform starting at $VCO/2$ with harmonics halfway between each VCO harmonic. When SUB2 is tuned to 3, the FFT then shows a waveform starting at $VCO/3$ with 2 harmonics between each VCO harmonic. Finally, the VCO and two subharmonics combined produce the expected 3 harmonics between each VCO harmonic.

4.4 Usability of Interface

A big draw of analog synthesizers is the satisfaction of interacting with physical knobs, and the immediate auditory feedback when you make an adjustment. These features are preserved in the Bela implementation by using potentiometers and buttons for the most time-sensitive parameters, and polling the hardware ev-

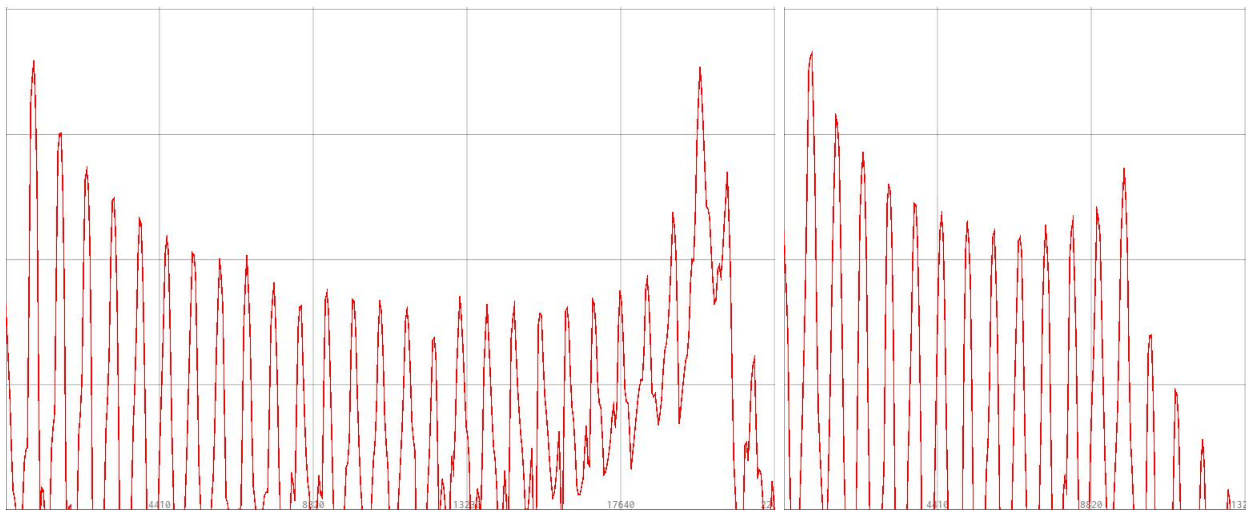

Figure 8: Maximum resonance at 19kHz cutoff(left) and at 10kHz cutoff(right)

ery 2 audio samples. However, since the hardware is laid out on breadboards and connected to Bela with breadboard wires, the physical interface is quite cramped and not nearly as clean as the original synth.

The Javascript GUI is not as satisfying to use for parameter adjustment, though it is better layed out than the hardware interface and the controls are actually labeled. It is difficult to make small adjustments to the sliders, and the auditory response is not as immediate. Furthermore, having to alternate between the hardware and software interfaces is clunky. While its possible to rotate two of the physical controls at the same time, its very difficult to rotate a physical knob and drag a slider simultaneously. The split hardware/software interface is not ideal, but given the limited number of analog inputs on Bela it was necessary to move some of the controls to software.

5 Conclusion

The digital Subharmonicon replica preserves the core oscillator and polyrhythm functionality of its analog counterpart, and Bela's 22050Hz sampling rate of the analog inputs allow the potentiometers to acheive the same responsiveness analog synthesizers are known for. However, moving to the digital domain causes a few problems. Aliasing causes signal degradation at extreme parameter settings, and many of the controls had to be moved to software due to a shortage of analog input pins.

Future work could focus on improving the hardware interface by designing a circuit board layout for the analog controls, and adding LEDs to indicate step sequencer position. Additionally, more of the parameters could be moved to analog control by adding a button to cycle between different sets of parameters to be controlled by the potentiometers. With a few refinements, the digital subharmonicon has the potential to be a low cost, open source alternative to the expensive Moog original.

References

[1] V. Välimäki and A. Huovilainen, "Oscillator and filter algorithms for virtual analog synthesis," *Computer Music Journal*, vol. 30, no. 2, pp. 19–31, 2006.

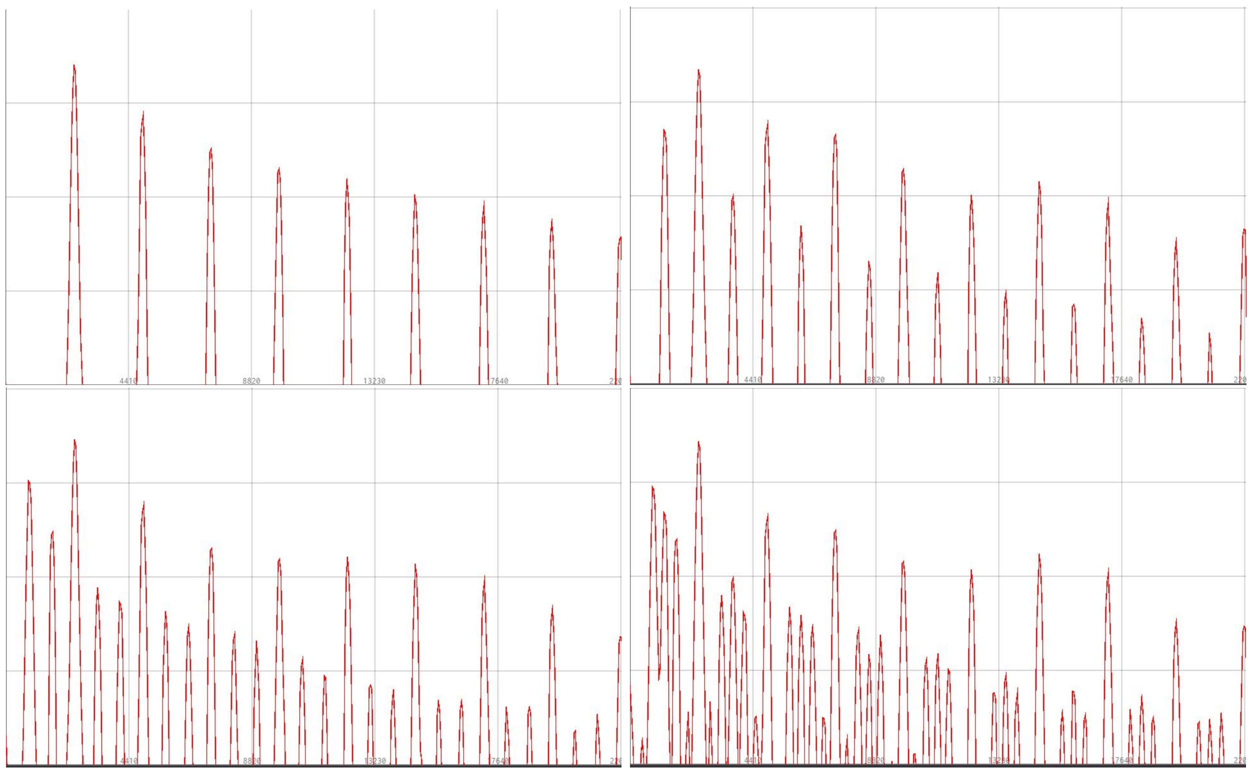

Figure 9: FFT of sustained oscillator 1 from top left to bottom right: VCO only, VCO with SUB1=2, VCO with SUB2=3, VCO with SUB1=2 and SUB2=3