

Creating a Software Instrument: Exploring the Process of Building the Monophonic Quadruple Oscillator Synth Patch in Max MSP 8

Sarah Carton

1914093

Development Module

Element 2b – Instrument Essay

Creating a Software Instrument: Exploring the Process of Building the Monophonic Quadruple Oscillator Synth Patch in Max MSP 8

In this essay I will be discussing the process of building my Monophonic Quadruple Oscillator Synth Patch in Max MSP and Max for Live, for use in Ableton Live. I aim to discuss the key differences my research has suggested between other popular patching software environments such as Reaktor and Pure Data, as well as exploring other software instruments, plugins and synthesizers on the market today, that have both influenced my choices and inspired my ideas for the future development of my patch. Additionally, over the course of this essay, I wish to reflect on the potential for growth for my patch, and how Max MSP's wealth of readily available resources can aid me in progressing my skills, before building new features for the Monophonic Quadruple Oscillator Synth Patch.

The first patching software I looked into using for the creation of my instrument was Reaktor. It seemed to have some fantastic support channels and tutorials available, and the fact that it *'comes with over 60 instruments and plug-ins'* as well as having the ability to *'reverse engineer any module'* (Truesdale, C. 2007) seemed very impressive. Though, as mentioned both in online forums and in Cliff Truesdell's *'Mastering Digital Audio Production...'*, Reaktor is definitely *'on the expensive side'*, and while being on a bedroom producer's budget, Pure Data sounded like the more accessible option for me. The 30 blocks that come with Reaktor that are said to mimic hardware modular systems, as well as the factory content synths, looked like fantastic tools for starting out building a software instrument (macProVideoDotCom. 2015). Though, when comparing Reaktor's patching environment to PD and Max, it seemed to be much less of a DIY programming environment, with less user control over capabilities and build than these other two programming platforms. For me, I felt that as it was my first time working with building software instruments and programming, I wanted to work with a software that would give me a better understanding of the technical build of these plugins from scratch and a more in-depth knowledge around the processing behind some of my favourite software instruments and plugins.

Pure Data (PD), similarly to Reaktor, had a wealth of resources available for learning online. PD seemed to have everything I needed to create the kind of oscillator synth patches I was after. Before I'd learnt more about the user experience of Max MSP, PD was the software I was keen to begin using. One thing that had struck me as a key drawback to PD in

comparison to Max MSP though, was that it didn't have Jitter, which I could see as being a useful tool for creating visuals for future live shows and DJ sets. These worries were counteracted however, by an article on PD, that reassured me that *'as well as audio and MIDI, Pure Data includes functions (called objects, or externals) for generating 3D graphics and video, and can also control external hardware like stage lighting or robotics'* (Armfield, A. 2006). However, it was when I began watching beginner YouTube tutorial videos for PD, I found the basic nature of the patching environment and layout seemed a bit too stripped back and inaccessible for a beginner (Vreahli, 2007). It felt like a very separate world from the DAW environments I'm used to working in, and the only guides and PDF's I could source were filled with math and complex instructions.

When first comparing PD with Max MSP, PD was described as a very similar program; both were *'comprised of patches, starting with the main patch'*, that begin with an *'empty patch'* where the design is created by *'placing objects in this patch and connecting the inputs and outputs of the objects'* (Christensen, M. p.194). I couldn't see at first, other than the improved design, why people would pay for Max MSP when PD is so similar. From a technical capabilities perspective, there's not much I was able to find that differentiated one from the other, but as I started reading discussions around the two in online forums, it became clear that for someone who had never used these types of programs before, Max MSP's friendlier patch environment, programming tools, online community, tutorials and documentation would make for easier and less taxing learning.

While researching the different software options for my patch, I didn't come across many negatives for Max, though the information below came up in several books, forums and articles;

'Max/ MSP is very good for designing complex DSP systems but does not have very good tools for sequencing. By design, Max/MSP does not impose any particular organizational model on the user; the starting model for a Max/MSP patch is a blank page. But this means that the user interface must be built from scratch each time a new piece is created'. (Lyon, E. p.305)

Both its shortcomings as a tool for sequencing and Max's 'start from scratch' intimidating blank page seemed daunting to me at first glance, though as I continued getting to grips with

the patch tools through Max tutorials, I learnt to create more realistic goals for myself and my software instrument. I wanted to keep it simple, focusing on a synthesizer I could use going forward in elements of my production, rather than a multi-purpose instrument with a host of effects. I also chose Max MSP 8 for this project, not only due to these comparisons with Reaktor and PD, but as I wanted a versatile, user friendly software that also worked well with Ableton Live. I was still running Ableton Live Lite at first and had not yet upgraded to Ableton Live Suite, as I was only just getting to grips with this DAW, but I knew this was a DAW I'd like to move forward with in the future with more electronic beat-based tracks, and for future live shows.

I had originally aimed to build a simple polyphonic soft synthesizer pad, as soft pads are a staple component in my production style. I was after a synth that I could incorporate into chill electronic and lo-fi music, that would be suitable to add light atmospheric effects and filters to in the future. I wanted a lo-fi sounding piano synth pad that could work as a layer in my production. While looking for the best lo-fi plugins available and searching for similar sounds on the market, I came across the SQ8ROM plugin, which is a *'sampled version of SQ8L, a VST plugin based on the sound and synthesizer architecture of the Ensoniq SQ80'* (Sample Science, 2020). The SQ8ROM had a great vintage piano pad sound, so I used this as the basis of the development of my sound I wanted for my patch. When tackling this, I first looked at a YouTube tutorial named *'Max 7 Tutorial 7 - Building a Basic Synthesizer'* (WestSideElectronicMusic, 2016). Using the steps from this and finding some more in depth information on how Max MPS Polyphony works from cycling74's own tutorials, such as how *'the poly~ object takes as its argument the name of a patcher file, followed by a number that specifies the number of copies (or instances) of the patch to be created'* (Cycling 74'), I was able to learn the basics and create a simple polyphonic synth, where I used a cycle oscillator to create my desired pad sound. My main issue with this patch was the level of sustain, as I had not built in filters to this yet, so the notes seemed to stay sustained continuously, which sounded messy in chord progressions. This meant I had to keep switching off the audio on the Max patch to create new chords and sounds. As I began researching ways to overcome this issue, such as adding live dials, envelopes and filters, I came across the series of YouTube tutorials that ended up becoming my main guide for creating the Monophonic Quadruple Oscillator Synthesizer.

The tutorial series had been put together by Eric Ameres (Ameres, E. 2011) and was originally created for Max for Live users, but I found this simple enough to replicate in Max MSP. These tutorials quickly became the basis of my final patch. The first few videos discussed building multiple oscillators to the patch. I then went in a different direction from my original single oscillator pad synth goal, deciding to expand my sound to create a patch that reflected my favourite Logic synthesizer leads and that worked in a similar way to the hardware Ensoniq SQ80 synthesizer I had researched previously when brainstorming sounds for the patch (Sample Science 2020). I chose to create a plugin with 4 oscillators, as most I found on the market had three, so I felt this would be quite unique. Though Ameres' tutorials didn't include cycle, I wanted to keep cycle on there for my soft pad sounds. One plugin on the market I found with 4 oscillators was the Sylenth by Lennar Digital; *'at its core Sylenth1 houses 4 alias-free unison oscillators, which generate analog shaped waveforms'* Lennar Digital (2020). The Sylenth is an impressive plugin, including a range of filters, modulation and master FX. As I continue to develop my plugin, I'd be looking to mimic some of Sylenth's impressive range of effects, such as its arpeggiator, delay, reverb and distortion. Following on from this, I looked at which oscillator I'd want to use for each sound, inspired by the oscillators in the Ensoniq SQ80 and my favourite synth leads on Logic (my preferred DAW). The first Logic stock instrument I found to be closest to one of my oscillators was the 'Analog Mono Lead' synthesizer. This sounded similar to the sawtooth oscillator I'd created. The sawtooth was the first oscillator I wanted to add to my patch, following some research around the different oscillator waveforms available in Cousins & Sawyer's *'Logic Pro 8: Audio and Music Production'*, as they suggested *'the sawtooth produces the richest amount of harmonics and is therefore the most popular choice for creating many of the classic synthesizers sounds'* (Cousins, M. Hepworth-Sawyer, R. 2013). I found that the triangle oscillator I created had clear similarities with Logic's 'Analog Fizz Lead', the square oscillator fit with Logic's 'Soft Square Lead', and my cycle oscillator's closest partner was the 'Ambient Lead' synthesizer. The cycle oscillator was the closest wave form sound I could find that worked as a soft pad; the sound I had initially set out to create, so I was glad to find a sound I could use in this way within the more developed patch.

I was glad the tutorials by Ameres included adding velocity into the patch, as I had recently purchased the Novation Launchpad Pro with velocity sensitive pads. I had to do some further research into this for Max, as at first, I had some trouble getting velocity to work with my Launchpad Pro. Luckily after some tweaking, I worked out that I had connected up my pitch

and velocity combinations the wrong way, so this issue was easy enough to get back on track. This was a helpful mistake, as it prevented me from connecting objects through the wrong route again for the rest of the project. Another issue I had in developing my patch, was during an attempted to build a second oscillator patch to the synth by encapsulation. The encapsulation process seemed to get rid of all audio. I tried a few troubleshooting steps noted on the YouTube comments of the Ameres video teaching encapsulation, from users with the same issue, though I found none of these worked for me. I have begun to continue my research on encapsulation, mainly through cycling 74's tutorial information, to better understand the process, so I have a better chance of troubleshooting this issue and succeeding in encapsulation in the future. I wished to build multiple oscillator patches into my software instrument so that I could use live dials to build unique sounds that blended two or more of these, in order to create a more unique and detailed sound.

Around halfway through my process of creating my patch, I decided to do some further research into Max for Live. As a software, it seemed Max for Live worked '*as a bridge between the two programs*' (Lyon, M. 1962) and would be the best tool for me to easily transfer my patch into live sets and for use in production on Ableton in the future. Luckily, I was able to get the free 90 day trial of Ableton Live Suite that comes with Max for Live for this project. A massive benefit of Max for live, is that '*externals designed for Max/MSP now have a larger potential audience, since the user base for Live is significantly larger than that of Max/ MSP*' (Lyon, M. 1962). This is an important factor to take into consideration, as the end goal for many Max developers would be to eventually monetise or share their patches with other producers or Max users and Max for Live has a larger audience for this. I felt both the potential for a larger audience for future Max patches on this platform, and the additional ease of compatibility with Ableton, were good reasons to move to this and continue developing my patch on this software. The ability to just transfer the Max MSP patch over to Max for Live by copy and pasting, made the workflow between these two fantastically simple. I'm looking forward to continuing to develop my use of both platforms together in the future, beginning the patch on Max MSP then transferring it to Max for Live before use in Ableton.

While recording and creating Ableton Live sets with the patch, I found that the sawtooth really drove the tracks, but that in the queues in which I took the sawtooth out, the square and cycle alone worked together to create a very warm, delicate, '*distinctive hollow and open*'

sound (Cousins, M. Hepworth-Sawyer, R. 2013). Similarly to Cousins & Sawyer, I found that *'overuse of the saw-tooth can definitely make a track appear rather stoggy and one-dimensional'*, and also drown out some of the more delicate sound waves that built uniqueness into the production in other areas; with heavy use of sawtooth my live set didn't have as much originality to it. With this in mind, I'd be interested in maybe creating a version of this patch with no sawtooth, but with more effects and filters, so that the attention could be focused on experimenting with the more open sounds; still sounding full but without *'grabbing large parts of the sound spectrum (and eating up your mix, therefore) like the sawtooth'* (Cousins, M. Hepworth-Sawyer, R. 2013). I would perhaps make two patches with a similar design and filters, but one based on just the sawtooth and the other focused on cycle, square and triangle.

In regard to what I would change as I continue to learn Max and become more independent in my approach to creating patches, without relying as heavily on reading, forums and YouTube videos, I'd like to find a more experimental approach to developing new sounds with the software, such as creating unique reverb, delay, glitching and crackling effects. I created a test reverb unit on Max MSP which worked well, so I'm looking to add this to my patch. I started by taking the signal, passing it through a series connection of several allpass filters, then splitting it into a parallel bank of feedback comb filters before mixing together in a mixing matrix. Reverb is one of the most popular plugins and effects on the market today and this continues to develop, *'with the ever-accelerating development of audio technology, reverb plug-ins have become powerful and sophisticated enough to rival even the results of natural reverb'* (Griffin, B. 2020). It's therefore important to build effects such as reverb and delay into a patch, as most plugins out there have these as standard, so to compete with other software plugins and patches this is an essential.

I'd also look to build a vintage crackle effects filter into my patch. The Waves Abbey Road Vinyl Plugin is one of my most used plugin effects for both lo-fi experimentation and when building in ambient sounds to my tracks. The plugin adds an *'authentic vintage vinyl sound to your tracks'* as well as *'noise and crackle'*. (Arblaster, S. 2016). I have seen a cycling 74' forum discussion speaking about this, with one user suggesting to *'multiply noise~ with pink or two times with pink'* (Thilenius, R. 2010) to create this effect. I look to trial this as an optional filter for my software instrument in addition to the reverb function as it develops.

A final and most important element I would add to my patcher would be Polyphony; *'If we wanted to make a polyphonic synthesizer, we would need some way to divide up the midi notes as they come in and route them to different instances of this patch'* (Manzo, V.J. 2011 p.228), this research in J. Manzo's *'Practical Guide to Developing Interactive Music'* was my first insight into developing polyphony for my monophonic quadruple synth patch. As I continue development I shall be using Manzo's advice to learn how to divide up and re-route notes in Max in this way. I attempted this for my current patch, though I ran into some difficulty with this. It couldn't get audio out when trying to add the polyphonic inputs. After some troubleshooting, it seemed the envelope wasn't getting to the correct instance. From online forums, I found some advice on adding a trigger so that the [midinote] message only goes to [poly~] before the envelope (Cycling 74' 2020), though this didn't seem to do the trick. I now plan to re-start my patch, building in polyphony from the beginning and not trying to simply add it in later, as I've learnt from working through a range of tutorials this is an essential element to get right from the start. As a composer and singer/ songwriter, a polyphonic synthesizer is important for me, as it's essential for chord progressions and composition. Even though, and as I learnt, *'there are a few technical reasons that make a polyphonic unit rather difficult to design at best'* (Electronic Music Synthesizers, Delton T. Horn p.20), the popularity of both hardware and software polyphonic synthesizers continues to grow in the market and in the homes of musicians. This can be seen by a quick search of the top hardware and software synthesizers on the market today. If you look at Music Radar's list of the *'Best High-End Hardware Synthesizers In the World Today'* (MusicRadar. 2020), almost all synthesizers on that list have at least 3 note polyphony, with some in the hundreds. It would therefore be important for my synthesizer to have these capabilities in order to compete with both the demand and other industry standard synthesizers on the market today.

To conclude, the Monophonic Quadruple Oscillator Synth Patch is still very much in its early days of programming, thus would not be a competitor to the mass number of highly functional and varying plugins and patches available both on Max for Live and in other markets today, with multiple oscillator functionality. That said, it's four oscillators do give the user both some choice and an ability to create versatile combinations with the sawtooth, triangle, square and cycle wave sounds. It's ability to alter frequency and filter capabilities mean that there is scope for both experimentation and individuality in its use, plus Max MSP provides ample opportunity for further development and growth of simple patches like these through its fantastic learning and tutorial recourses. Max MSP & Max for Live seem to be the

best programs for me for creating patches in comparison to PD and Reaktor, as Max works as a happy medium between the two; enough math, DIY creation and resources to counteract the temptation to drag and drop ready built functions as could be the case in Reaktor, but a more user friendly, DAW compatible and developed programming resource than PD. As I continue to trial building in effects and filters into my patch, I look forward to both the possibilities Max MSP and Max for Live bring for my Monophonic Quadruple Oscillator Synth Patch and the new Max community I now feel ready to become a vocal member of.

Bibliography

- Ameres, E (2011) *M4L: Build a MIDI Synth in 1 minute Pt 2: ADSR~ envelope : Learning Max for Ableton live tutorial*
Distribution. [online video] Available at:
<https://www.youtube.com/watch?v=-3seRXXNR9o>
[Accessed 2nd July 2020]
- Ameres, E (2011) *M4L: Build a MIDI Synth in 1 minute Pt 3: SVF~ Filter : Learning Max for Ableton live tutorial*
Distribution. [online video] Available at:
<https://www.youtube.com/watch?v=gBAMW5WTitI>
[Accessed 4th July 2020]
- Ameres, E (2011) *M4L Tutorial #4 add a filter envelope to 1 minute synth: Max for Ableton Live*
Distribution. [online video] Available at:
<https://www.youtube.com/watch?v=XDiciEiFhHc>
[Accessed 4th July 2020]
- Ameres, E (2011) *M4L Tutorial #5 Adding New Waveforms to our synth in Max for Ableton Live*
Distribution. [online video] Available at:
<https://www.youtube.com/watch?v=3whll-BnT-w>
[Accessed 5th July 2020]
- Ameres, E (2011) *M4L Tutorial #6 : add a 2nd oscillator to synth by encapsulation : Max for Ableton Live*
Distribution. [online video] Available at:
https://www.youtube.com/watch?v=dI_0v_5lzM4
[Accessed 6th July 2020]

- Arblaster, S. (2016) *Cut some serious grooves with Waves Abbey Road Vinyl* Distribution. [online article] Available at:
<https://www.musicradar.com/news/tech/cut-some-serious-grooves-with-waves-abbey-road-vinyl-642889>
[Accessed 25th July 2020]
- Armfield, A. (2006) *Pure Data: An Introduction* Distribution. [online article] Available at:
<https://www.soundonsound.com/techniques/pure-data-introduction>
[Accessed July 1st 2020]
- Christensen, M. (2019) *Introduction to Audio Processing*. Denmark. Springer
- Cousins, M. Hepworth-Sawyer, R. (2013) *Logic Pro 8: Audio and Music Production*. Burlington, MA. Taylor & Francis.
- Cycling 74' (2020) *Polyphony Tutorial 1: Using the poly~ Object* Distribution. [online article] Available at:
https://docs.cycling74.com/max8/tutorials/11_polychapter01
[Accessed 17th July 2020]
- Cycling 74' (2020) *Tutorial 14: Encapsulation* Distribution. [online article] Available at:
<https://docs.cycling74.com/max5/tutorials/max-tut/basicchapter14.html>
[Accessed 6th July]
- Griffin, B. (2020) *A History of Reverb in Music Production* Distribution. [online article] Available at:
<https://www.izotope.com/en/learn/a-history-of-reverb-in-music-production.html>
[Accessed July 20th 2020]

- Lennar Digital (2020) *Sylenth 1*
Distribution. [online article] Available at:
<https://www.lennardigital.com/sylenth1/>
[Accessed July 6th 2020]
- Lyon, M. (1962) *Designing Audio Objects for Max/ MSP and PD*. Middleton, Wisconsin. A-R Editions, Inc.
- macProVideoDotCom. (2015) *Reaktor 6: What is Reaktor?*
Distribution. [online article] Available at:
<https://www.youtube.com/watch?v=oqYi74CIJGc>
[Accessed July 20th 2020]
- Manzo, V.J. (2011) *Max/MSP/Jitter for Music: A Practical Guide to Developing Interactive Music Systems For Education and More*. Oxford. Oxford University Press
- Music Radar (2020) *The 23 best high-end hardware synthesizers 2020: keyboards and modules for \$/£600 and above*
Distribution. [online article] Available at:
<https://www.musicradar.com/news/the-best-high-end-hardware-synthesizers-in-the-world-today>
[Accessed 17th July 2020]
- Sample Science (2020) *SQ8ROM*
Distribution. [online article] Available at:
<https://www.samplescience.ca/2019/09/sq8rom.html>
[Accessed July 2nd 2020]
- Thilenius, R. (2010) *Crackle Synthesis*
Distribution. [online article - forum] Available at:
<https://cycling74.com/forums/crackle-synthesis>
[Accessed July 20th 2020]

- Truesdell, C. (2007) *Mastering Digital Audio Production The Professional Music Workflow with Mac OS X* Indianapolis, Indiana. Wiley Publishing, Inc.
- User 909955847736 (2009) *Ensoniq SQ-80 demo 1 - Kirk Slinkard* Distribution. [online video] Available at:
https://www.youtube.com/watch?v=Dw_sd0RHlZY
[Accessed July 2nd 2020]
- Vreahli (2007) *Beginner's Puredata Tutorial Part 1 of 2* Distribution. [online video] Available at:
<https://www.youtube.com/watch?v=FkmMzqohsDo>
[Accessed 1st July 2020]
- WestSideElectronicMusic (2016) *Max 7 Tutorial 7 - Building a Basic Synthesizer* Distribution. [online video] Available at:
<https://www.youtube.com/watch?v=TOtzaJBKhfM>
[Accessed July 9th 2020]