

Tileyard Education
University of Wales Trinity Saint David

Student ID: 2203868
Ruddy Edson Justiniano Garcia

MA Commercial Music Production

Module: Development
Module Code: ADAD7005

Word Count: 3758

SES-SynthBoy-01

Super Brass Bros. Edition

INTRODUCTION

SES-Synthboy01 is a virtual hybrid instrument consisting of a sample instrument and synthesizer with the aim to create 8-bit music otherwise known as Chiptune. The inspiration behind this instrument was the Nintendo Entertainment System (NES) and the Gameboy DMG-01, which were both video game consoles released in the 1980's having sold a combine total of 182 million units worldwide (The Telegraph, 2017). This brought 8-bit music into the ears of millions of consumers through the games they played such as Super Mario Bros., The Legend of Zelda, and Tetris. In creating this instrument, I researched on how music was created for these games consoles as well as others in the market such as the Sega Genesis (Sega Mega Drive in the UK) so I could have an understanding on how I would approach my instrument regarding sound and functionality. The main goal of my project was to make it as easy as possible to open up the SES-Synthboy-01 in a DAW and start creating nostalgic music. The 'SES' in the name stands for Synthesizer Entertainment System.

VIDEO GAMES IN THE 80'S, SOUND CHIPS AND WAVEFORMS

In the 1980s video game consoles and computers used microprocessors (computer chips) for dedicated functions such as audio, CPU, and graphics (Hopkins, 2022). The chips used for music in these consoles were known as sound chips which generated different wave forms to create sound – IBM in 1987 used an Adlib card which had a Yamaha YM3812 chip that allowed 9 voices (waveforms) (Purcaru, 2014). The waveforms most of these sound chips generated was a pulse (square), triangle, sawtooth, noise waveforms and a low quality sampler as will be explained with the following consoles.

The Nintendo Entertainment System (NTSC region) used the Ricoh 2A03 sound chip and in the UK/PAL region, they used the Ricoh 2A07 chip – both were identical in their function of offering 5 voices: two independent square waves, a triangle wave, noise generator and a PCM sample (McAlpine, 2019). All these 5 voices could have been used as the composer pleased but in general the square waves were used for melody and harmony, the triangle wave for bass and drum, the noise generator used for percussive instrument and the PCM samples were used for low quality samples such as bongos.

The Gameboy DMG was a handheld system, so it used a sound chip called PSG that allowed 2 square waves channel for melody and harmony, 1 programmable (variable) wave channel which was commonly used for the bass line and a white noise generator used for percussive sounds (Collins, 2008).

The Sega Genesis used a Yamaha YM2612 FM chip which had more capabilities than the chips used by Nintendo including a stereo output which was revolutionary at the time (Audeze, 2020). This was also the same chip used in the Yamaha DX7 Synthesizer and with the Sega Genesis it allowed for 6 different voices – essentially 6 oscillators to be modulated to any waveform which allowed more creativity.

CONCEPT

I wanted to emulate the look and feel of the Nintendo Entertainment System and make it as simple as possible so that even a school student was able to open this virtual instrument and understand it. I also wanted to provide the functionality the Ricoh 2A03 chip gave:

4 Voices:

1. A sampled instrument which the end user will not be able to manipulate instead it will be a predetermined sample.
2. Square Wave for melody and harmony
3. Triangle Wave for bass, melody, and harmony
4. White Noise for the drums or percussive sounds

The reason I opted to use a predetermined sample rather than allowing the end user to upload their own samples was because before the NES concept, I had already planned to create some sort of brass instrument or an instrument that had a brass element to it. In this case I decided to have a trumpet sample which I edited it to fit alongside the waveforms. I also decided for 4 voices unlike the 5 in the NES because unlike the limitations of the 1980's sound chips, the end user will have many channels available in their DAW.

I wanted a basic user interface with the least number of buttons so it can look clean and friendly where the end user is more focused on creating music rather than creating individual sounds. I wanted to achieve this by have a button for each voice, a simple drum machine to manipulate the white noise generator and a keyboard which shows which notes are being played.

The design aspect of SES-Synthboy-01 was inspired by both the NES and the GameBoy DMG-01 consoles using the colour scheme of grey, dark grey, black, and red. I also created a logo that consist of an 8-bit cartoon keyboardist with the instrument name and edition name (Super Brass Bros. Edition).

HISE STANDALONE APPLICATION

The choice of software I decided to use to build my instrument was HISE which is an open-source application for building virtual instruments (<http://hise.audio>). The reasoning behind this decision was the ability to do everything I needed for my project:

- A sampler for my brass sample
- Scripting which allows me to build and design a custom user interface.
- A waveform generator (square wave, triangle wave, and a white noise generator)
- Audio FX such as a filter, degrader, etc

The installation process of HISE Standalone application was also simple with the only problem I encountered was that my Apple computer was refusing to install the application which was resolved in the System Preference > Security & Privacy tab and clicking allow.

BUILDING THE VIRTUAL INSTRUMENT

Voice 1 – The sampled instrument

The sampled instrument I decided to use was a trumpet because I felt it has the right timbre to be manipulated into an 8-bit instrument. I started looking for trumpet samples online as I did not have the facilities to record a live trumpet. I did find a sample library pack on Splice called Jazz Cuts feat Alita Moses which had 3 single shot trumpet samples. The notes were E, A, and C, so I imported these into my DAW (Logic Pro) and pitch shifted each note accordingly to get an octave of notes starting from Eb to D. I did try to create an octave higher and lower but the more pitch shifting I was doing to the notes, the more robotic they were sounding, so I stayed with the 12 notes.

Velocity was not going to play a part in my instrument however I did try and create fake round robins to experiment how it would sound. One way of creating fake round robins as explained by David Hilowitz Music (2018) is by using the two-semi tones next to the note and pitching it to the round robin note however this option is only available in Kontakt and I was not able to find a script for this in HISE. Another way of creating fake velocity is by using a transient shaper and playing with the attack and release as explained by Pro Audio Files (2017). I used the same Neutron Transient Shaper by izotope to create a slow and fast velocity for each note however the outcome was not as desirable.

Once I had my trumpet samples exported into wav. files and ready to be imported into HISE, I followed a walkthrough video by the creator of HISE - Christoph Hart (Juce, 2018) which showed me how to build a sampler. After I dragged the trumpet samples into the sampler editor, I realised the volume levels were extremely loud, so I normalised the samples in the sampler editor and lowered the volume down further by -6.0db. This helped save time as otherwise I would have had to go back into Logic Pro and fix the volume problem. Once I was happy with the trumpet instrument, I saved the sample map and exported the sample map into a HLAC monolith file as Christoph Hart (Juce, 2018) explains that this makes it easy for distribution.

I wanted my trumpet sampler to sound 8-bit rather than real and clean, so I added an FX chain in the editor and experimented with a few units such as the Shape FX unit which gave a distorted sound, but it did not sound 8-bit. Harmonia (2018) explains

that most likely if you do try to reduce a musical audio file to 8 bits you will likely get a horrible, distorted audio file rather than a desired video game sound. I decided to give this a try and inserted the Degradar FX unit and reduced the sample rate and bit rate which worked well getting a low-quality trumpet sample which I could envision coming out of an NES.

Voice 2 & 3 – Square Wave & Triangle

HISE has a sound generator built in so getting voice 2 and 3 was very simple by adding 2 waveform generators on the master container and then changing the synthesizer waveform on each to a square wave and the second waveform generator to a triangle wave.

Voice 4 – White Noise

White noise was also simple to create because HISE offers two options via a waveform generator, or its own unit called Noise Generator. As this is meant to be modulated to be a drum or a percussive voice, I added a AHDSR envelope under the gain modulation and lowered the attack time, hold time, sustain level, and release time for a snappy sound. The decay time slider will be added in the user interface so the end user can manipulate this as they please for example if they would like to create a longer lasting snare then you raise the decay slider. Lastly for manipulating the noise generator I added a Filter FX chain as this is utilised to get the snare, hi hat or kick drum sound which the end user will have access to with a frequency slider and filter knob for example for a hi hat you will need a high pass filter and for a snare you will need a low pass filter. When using a high pass filter to create a hi hat, the loudness drops so to solve this issue I added a simple gain knob for the end user.

The User Interface

Once I had all my 4 voices sorted, I clicked the Interface Designer in HISE to find that I was not able to right click to create a component and start building. I was using an Apple mouse so to right click I had to press control on the keyboard and click the mouse which worked normally outside HISE but not in the application. I tried to resolve this by deleting the default interface and starting a new interface under the master container with script editor and writing *Content.makeFrontInterface(650, 650)* but this did not work. I tried looking in the HISE forum but could not find any help but I did see a previous Apple user had the same problem in the past, so I went and bought a new mouse with both a left and right click and this solved the issue.

The first component I added was a keyboard which I placed at the bottom of the interface as I wanted my virtual instrument to look like a synthesizer in the shape of a Yamaha DX7 or Roland JX-3P. I also edited the size of the interface from 650 width, 650 height to 850 width and 300 height to look like the synthesizers above. As the design inspiration behind the SES-Synthboy-01 is the NES and the Gameboy consoles, the colour scheme will be light grey, dark grey, black, and red, which I started by using a light grey background.

The panel of my instrument which will hold the buttons and knobs consists of a black background with the logo on the left, buttons for each voicing in the middle and the parameters to edit the white noise on the right. I created the logo as 3 separate components in HISE:

1. The name “SES-SYNTHBOY-01” was made on a website called The Word Finder (<https://www.thewordfinder.com/nintendo-font-generator/>)
2. The 8-bit cartoon of a man playing a keyboard was found on <https://www.clipartmax.com>
3. The edition name “Super Brass Bros. Edition” was a label component in HISE which I edited the font and colour.

I wanted there to be a contrast between the 4 voicings and the parameters for the white noise, so I added a dark grey panel through the middle with a title “>PRESS ANY MODE:” which indicates to the end user to click on any voicing to get a sound. I named each voice:

1. BRASS (the sample)
2. MELODY + CHORDS (square wave)
3. BASS + MELODY (triangle wave)
4. DRUM MACHINE > (Noise)

The default component button in HISE was very dull and not engaging, so I explored the KnobGallery website (<https://www.g200kg.com/en/webknobman/gallery.php>) and I found a button called “Behringer_Switch_Red_LED_Button”. I edited this knob adding an image of an 8-bit Super Mario character. I tried then to upload the .knob file into the filmstripimage under the component specific properties but the button did not change into the one I designed. I then found a video by David Healey (2023) that you must upload your buttons as a .png file for HISE to read it. So, I went back to the website to redownload my button and it worked, so when the end user clicks a voice, the Super Mario character will go dark. I opted for light to dark rather than dark to light because the dark Super Mario is not as clear as when it’s lit up.

I wanted each button to be linked to a specific voice for example MELODY + CHORDS to turn on the square waveform. I found a way to do this in the HISE forum by musictop (2018) which instructed to change the parameter properties and point to the waveform in ‘precessorid’ menu and clicking -bybass in ‘parameterid’. This bypasses each voice so when an instrument is on it will play however the other voices must be off otherwise, they will play at the same time.

On the right-hand side of the black panel, it’s where I placed the white noise control parameters. It contains a decay knob which is linked directly to the AHDSR for a longer snare sound. A gain knob for when the volume levels drop when using a high pass filter. A filter graph which shows visuals of when the frequency knob and frequency mode are being manipulated.

HOW IT WORKS

The SES-Synthboy-01 once loaded in your DAW you should be able to open it up and have a choice of 5 different voices. To create a full production ideally, you’ll need 4 tracks opened in your DAW all having my virtual instrument loaded. A basic example would be for each track:

1. Click on MELODY + CHORDS and arpeggiate an A minor 7 chord (A, G, C) and on a new track in your DAW create a melody line.
2. Click again to turn off and click on BASS + MELODY and create a bassline.
3. Click again to turn off and click on BRASS to create a counter melody.
4. Click again to turn off and click on DRUM MACHINE, then set the decay for a hi hat sound, click 7=SVF HP (high pass filter) and record in your hi hat.

This process can continue with more tracks in your DAW to create a snare, kick, more melodies, and riffs. There are no limits in your DAW however unless you would like to experiment it is important to turn off the voices, so they won't play at the same time.

REFLECTIONS AND IMPROVEMENTS

The virtual instrument turned out well and it functions as it looks, from the moment you open it, it is very clear what each voice means and does. Even the white noise generator is self-explanatory, although the name "Drum Machine" may sound deceiving for a professional producer as there isn't any programming of drums, the end user will understand what it is. In the wider context of using this in a project I also believe the qualities are there to help compose music and there isn't anything that can hold you back when writing a demo. There are many areas of improvements which can be achieved via a SES-SYNTHBOY-02 by allowing the end user to upload their own samples for examples as there are only so many editions you can have before the end user becomes tired of downloading a new VST. I also believe implementing more parameters to give the end user more control will improve it drastically, but this will also mean a completely new user interface design to try and fit a lot more knobs in.

The waveforms is also another area of improvement in trying to replicate the right timbre and colours of the NES Ricoh 2A03 chip as the one in my instrument sounds too clean and modern. I would need to do more research if it is an area I would need to work on outside of HISE. This also leads to having preset of sounds from iconic video games to offer something more to those who want to load a sound rather than build or manipulate one.

BETA Testing

I did have a BETA testing session in my house as I work with a Mac mini, so I didn't have the portability of a laptop to do this elsewhere. The testing was done directly via the HISE Standalone application using the Interface preview. In total 2 people took part:

Person A: Musician (keyboard player) that has experience with Logic Pro and VSTs.

Person B: Musician (drummer) and sound engineer that has experience with Logic Pro and VSTs.

Verbal feedback:

Person A: The waveform sounded too clean which works if you want to make a Chiptune sound however try and recreate the NES sound even if it means more work but overall, it looks good and works. The filter also doesn't feel smooth when using it.

Person B: The product looks good; it has a range of sounds and it's easy to use.

I do agree with the BETA testing feedback but trying to implement these improvements will have to be a completely new design and concept because there is not much space to add more functions in the interface to try and get the NES sound. The end user will need to have parameters to be able to manipulate the sound to achieve the exact sound from the 1980's. It also runs away from the idea that anybody can load the instrument up and understand it which was one of my main aims at the start of this project.

Competitors

In the market there is a wide range of real and virtual synthesizers that can generate the 4 voices in my virtual instrument so I will be narrowing it down to 3 vsts that is designed for 8-Bit/Chiptune and has a retro gaming user interface design.

8-Bit Synth by UVI is a synthesizer in the market at the price point of €99 which includes a free sample library called Toy Suite. This VST tries to emulate the sound chips of the 1980s rather than a single sound chip. It contains 375 patches of sound which its users can manipulate using ADSR envelope, pitch, portamento, and more. The design is very unique, and arcade inspired, the user interface is also very easy to understand and with different tabs/layers the screen doesn't feel crowded. The feel of this VST feels very much like a game rather than an instrument compared to mine however it does offer the end user a lot more capabilities.

<https://www.uvi.net/8bit-synth>

NES by Matt Montag is a free synthesizer that like my instrument tries to emulate the Ricoh 2A03 chip used by the NES console. It does give a more accurate wave sound to that in the NES and allows the user to upload their own samples in a DPCM (delta pulse code modulation) unlike the predetermined trumpet sample in mine. The NES by Matt Montag also provides more modulation control including volume, pitch, fine pitch and duty cycle giving the end user more control of the waveforms. The design of this instrument follows a similar colour scheme of grey with the logo being a sound chip, music notes and a keyboard with the name "NES VST", which looks distinctly different to my design.

<https://www.mattmontag.com/projects-page/nintendo-vst>

Basic 65 by De la mancha is a free synthesizer that tries to emulate the Commodore 64 in design and sound. It has 3 oscillators that can generate pulse, saw, triangle, and noise however it does have large amount of modulation parameters, effects, and a built in arpeggiator that makes it more powerful. Because it is powerful, the end user would need to have knowledge on how to use these compared to my simple instrument which is quick and easy to use.

<https://delamanchavst.wordpress.com/2013/12/12/basic65-retro-chiptune-synth-is-released-for-free/>

Conclusion

At the start of this project my main objective was to create a playable virtual instrument that I would enjoy and learn via trial and error. HISE is a very powerful application that I felt I did not use to its full capacity due to my lack of scripting (coding) but I did create a final product in the end. If I had more time to learn how to script, I would have improved the small things such as the voice buttons turning off automatically once another is pressed or fixing the filter mode list as number 11 was unassigned. I also realised how small but active the HISE forum community is especially to David Healey and his YouTube channel which helped me on a few occasions.

Researching at the start of this project I learned a lot about a generation of sound chips and the limitations composers had in the 1980's creating music with low quality waveforms and making the most out of a few voices. Although it may sound irrelevant to this project, 8-bit music theory on YouTube did help in my decision to keep my design and functions to a minimal. There is also a thriving community in 8-bit music and a lot of VSTs available in the market with complex design so creating something simple was different.

Finally, the whole process of learning how to build a virtual instrument was enjoyable and I have picked up skills along the way, I will continue to learn more about sound chips and synthesizers of the 1980's.

Bibliography

Audeze. (2020) *Behind The Bit: How Sega and Nintendo Used Hardware Synthesis To Create New Worlds*. Available at: <https://www.audeze.com/blogs/audeze-journal/behind-the-bit> (Accessed: 16 March 2023)

Collins, K. (2008) *Game Sound: An Introduction to the History, Theory, and Practice of Video Game Music and Sound Design*. Cambridge MA: The MIT Press.

David Healey (2023) *HISE: How to set an image for a button | button filmstrips*. Available at: <https://www.youtube.com/watch?v=d77QJfEMGms> (Accessed: 16 March 2023)

David Hilowitz Music (2018) *Kontakt Tutorial: How to Fake Round Robin Samples and More*. Available at: <https://www.youtube.com/watch?v=GaTxCq9avf8&t=2s> (Accessed: 7 March 2023)

Harmonia. (2018) *Digital Audio: the real meaning of 8-bit music*. Available at: <https://medium.com/@harmonia.global/digital-audio-the-real-meaning-of-8-bit-music-1be5fc8ab2b1> (Accessed: 8 March 2023)

Hopkins, C. (2022) *Video Game Audio: A History, 1972 – 2020*. North Carolina: McFarland & Company Inc.

Juce (2018) *Cristoph Hart – Build virtual instrument with HISE*. Available at: <https://www.youtube.com/watch?v=how73v1PNng&t=2161s> (Accessed: 7 March 2023)

McAlpine, K, B. (2019) *Bits and Pieces: A History of Chiptune*. New York: Oxford University Press.

Musictop (2018) 'How to make a bypass button ?', *HISE Forum*. Available at: <https://forum.hise.audio/topic/783/how-to-make-a-bypass-button/7> (Accessed: 16 March 2023)

Pro Audio Files (2017) *How to Mimic Velocity for Audio Loops*. Available at: <https://www.youtube.com/watch?v=wpstvLzEdAY> (Accessed: 7 March 2023)

Purcaru, B, I. *Games vs. Hardware. The History of PC video games*. September 22, 2014.
https://www.amazon.co.uk/gp/product/B00NUIJSIY/ref=dbs_a_def_rwt_bibl_vppi_i0

The Telegraph. (2017) *Nintendo Consoles: A history in pictures from NES to the SNES Mini*. Available at: <https://www.telegraph.co.uk/technology/0/nintendo-switch-history-nintendo-consoles-pictures/> (Accessed: 17th March 2023)