

Student Number: 1808651
Module Code: ADAD7005_LON
Lecturer: Ciarán Robinson
MA Commercial Music Producer
Development Assessment, Element 2b

PureData Synthesizer/ Wii Remote MIDI Controller

Kris Erice

Introduction

Because of how modern mixes for music put an emphasis on utilising the full frequency spectrum and as a result, it has standardised the use of saturation, brilliance and sub-bass. Furthermore the modern electronic basses heard in popular musical styles such as Dubstep and Trap bring the baseline to the forefront of the mix, becoming the melody part for the song in some cases. Two famous bass synth sounds include:

- the 'Wobble' bass - 'Cut-off Modulation' that is automated by the Low Frequency Oscillator (LFO) rate
- the Bass Glide - enabling the playing technique of Portmanteau where when a note is transitioning to the next note, the pitches in between are also played. For example, for guitar, the player will play the first note and then slide on the same string to reach the next

It is common to program these functions on a plugin synth like Omnisphere or Massive but for a live performance, electronic bass parts are usually left to the backing tracks or controlled because controlling these functions in real time could prove difficult. According to the drummer of the Electronic Dance band ELEVNS, Aurelián Merz, his bass performer would play simple synth bass parts using a Novation Bass Station or on his bass guitar. If the parts are too complex or the bassist wasn't available, it'd be played in the backing track.

Influenced by live Electronic musicians like SBTRKT and James Blake who will perform all the synth parts using actual synths and trigger samples using sample drum pads, this project could potentially bring a new live performance aspect for basses which are usually triggered.

Firstly starting off as a Raspberry Pi Bass Synth-centred venture, the project evolved into an interactively operated instrument involving a Wii Remote. Due to development difficulties with the initial concept, however, the project moved over to MacOS and Logic Pro for demonstrative purposes and then over to PureData to enact the concept. When discussing the methodology, the documentation will be divided into three parts to clearly illustrate each stage.

Kit List and Software

Raspberry Pi with Wii Remote:

- Raspberry Pi
 - Power supply
 - MicroSD card with Raspbian installed
- USB Keyboard
- USB Mouse
- MIDI Keyboard (Novation Launchkey 49)
- Wii Remote
- USB Audio Interface (Behringer U-Control UCA222)
- Headphones
- Monitor

OSCulator with Logic:

- MacOS computer
- Logic Pro software
- OSCulator software
- Wii Remote

PureData with OSCulator:

- MacOS computer
- PureData software
- OSCulator software
- Wii Remote

Project Overview and Planning

Many project ideas were thought about but creating a sub-bass synth was ultimately chosen because of the author's keenness to explore something interactive within his realm of music making. Ideas considered included Microphone triggering samples via Raspberry Pi and a Fixed Octave Harmoniser for Guitar/Bass/Microphone using an Arduino, the latter option exploring this tutorial <https://www.instructables.com/id/Arduino-Vocal-Effects-Box/> as a foundation but simplifying it to a fixed octave. Helen Leigh, the tutor, explained that due to time and budget restraints the project would not be able to be fully realised but it could possibly be looked at for future projects.

Idea 2: Sub bass generator

- Raspberry Pi
- generating tones from 20Hz to 60Hz
- potentiometers to control LFO rate and Glide
- perhaps the tone is controlled with MIDI keyboard and then you can just map out functions using the existing knobs/sliders
- Just a sine wave
- outputs to USB interface that connects to loudspeakers/headphones

Figure 2 - initial ideas for the Pi Bass Synth

Due to monetary restraints the author decided to use the affordable Raspberry Pi and with that decision, the MIDI instrument software solutions needed to be coded in Python. Furthermore, Helen advised to use the Pi so that there was better demonstrative aspect to the project.

The initial idea was to create a monophonic software synth that specialised in producing low tones allowing the focus to be placed on the interactivity of the project - controlling LFO rates and Glide in real time using a MIDI controller. Following Figure 1, the aim was to create the software synth using a Raspberry Pi 3 that received and played the MIDI information coming from a USB MIDI Keyboard. The sound generated from the Synth program would output to USB Audio Device and not the onboard 3.5mm (it has a low level driver causing latency and audible hiss) and finally out to headphones or speakers.

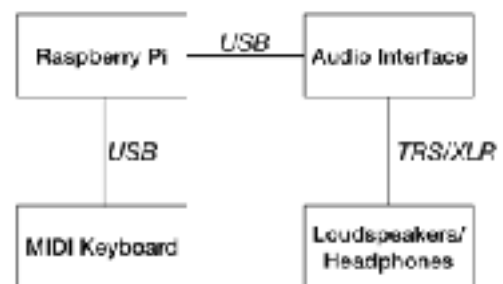


Figure 1 - Original Bass Synth project focusing solely on the creation of the synth written in Python

However as the project progressed, it became apparent to the author that more time should be spent on the interactivity of the project. This was partial to the market research done on existing products that played synth bass and included the MIDI capabilities. Thus, the focus shifted from the creation of the bass synth and onto finding a more engaging way for users to control bass. With the timeframe of the project being over just a month, and with the author trying to fuse two ideas, he felt that getting a stable proof of

concept done and having a demonstration prepared would be a decent result even if the synth wasn't completed.

It was at an early stage when the Wii Remote was decided to be a central focus of the project. Having used the Wii Remote as a wireless mouse before, the author was familiar with the hacking potential of the device. It had a Bluetooth connection which allowed connectivity with computers, 11 buttons that could be assigned to having different functions; but the most unique feature that it had was the accelerometer. This allowed the remote to sense acceleration along three axes and from experience using it as a wireless mouse, it meant that the axes could also be assigned to having a different function.

Another functionality of the remote was Infrared light sensing where in typical use with the Wii Sensor Bar, allowed the console to determine the remote's pointed position. Though interesting, the function wouldn't be explored in this project.

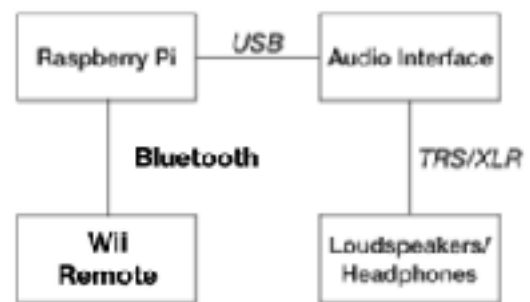


Figure 1 - Signal chain now involving the Wii Remote

Market Research

There are many software synths available on the market. Massive and Omnisphere for example that have many features including multiple filters, effects patch and an in-built arpeggiator, the intent behind the synth created for this project would be something very simple. Inspired by the Waves' 'One Knob' plugins, the synth would as stripped back as possible with limited functionality, just parameters for the ADSR envelope (which affects the Glide), the filter that features the cutoff that is modulated by the LFO rate. Featuring multiple waveforms could be explored but isn't necessary.

Regarding the MIDI controller it needed to be something interactive and engaging to control the LFO rate and the Glide. The band, Pinn Panelle, involves the bass guitarist triggering the wobble rate using a bluetooth ring. For 808 bass kicks (and other instruments), performers will trigger the sample on trigger pads like on the Maschine but an instrument most similar to the proposed project idea is Novation Bass Station.



Figure 1 - Skrillex - Scary Monsters And Nice Sprites (Live Cover by Pinn Panelle) (2011) - <https://www.youtube.com/watch?v=3fVIAGVe9TM>

The Bass Station is a monophonic analogue synth with two filters, two oscillators and an additional sub-filter and flexible modulation with two envelopes and two LFOs, where the software synth for this project aims to be monophonic, one filter, one oscillator and has assignable parameters to control the one envelope and one LFO. As seen in the image/video, the performer plays the bass notes whilst changing the LFO Rate in real time, by turning a



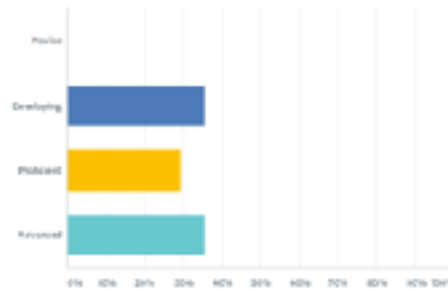
Figure 2 - Novation // Bass Station II Performance (2013) - <https://www.youtube.com/watch?v=3fVIAGVe9TM>

knob.

A survey targeting performing musicians was conducted by the author to gauge the interest for product. A mixture of music Facebook music groups were targeted and the Reddit subgroup ‘/r/musicians’. The questions and results from SurveyMonkey are as follows:

What descriptor most describes your level of Musicianship?

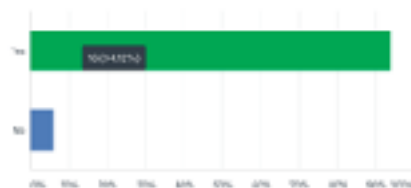
ANSWERS: 17 Skipped: 2



ANSWER CHOICES	PERCENTAGE	COUNT
Novice	3.00%	1
Developing	35.29%	6
Intermediate	58.82%	10
Advanced	3.20%	1
TOTAL		17

Do you think there should be more interesting ways to perform electronic instrument parts live?

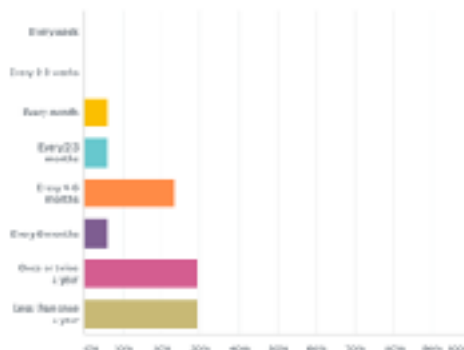
ANSWERS: 17 Skipped: 0



ANSWER CHOICES	PERCENTAGE	COUNT
Yes	94.12%	16
No	5.88%	1
TOTAL		17

If you own musical instruments (MIDI controllers, analog instruments, etc.), on average, how often do you make purchases on new musical equipment?

ANSWERS: 17 Skipped: 0



ANSWER CHOICES	PERCENTAGE	COUNT
Every week	5.88%	1
Every 2-3 weeks	2.94%	0
Every month	1.76%	0
Every 2-3 months	5.88%	1
Every 4-6 months	11.76%	2
Every 6-12 months	11.76%	2
Once or twice a year	11.76%	2
Less than once a year	11.76%	2
TOTAL		17

How much would you spend on a small computer instrument with the following features:- bootable software synth that specialized in making bass/sub-bass tones - connects to any bluetooth input device that allowed you to assign MIDI parameters to it?

Answered 19 · Reported 0



First question about level of Musicianship aims to hit multiple points; to see if there is a correlation between more skilled musicians and people who buy more gear (links to the question regarding frequency of purchasing instruments) and to see if the more skilled musician would want to experiment more with the performance of Electronic bass instrument (links to the 2nd question).

Of the six people who answered Advanced:

- all of them answered 'Yes' to wanting a more interesting way to perform Electronic parts
- two people purchased music equipment every 4-6 months, two people purchased every 6 months, three people purchased only once or twice a year

Though all Advanced musicians wanted a more interesting way to perform, the results about purchasing equipment varied.

The last question asks about how willing a customer would spend on the fully realised product - a small computer synth that specialises in producing bass/sub-bass tones and can connect to any Bluetooth input device with assignable parameters. For Advanced musicians the results varied from £15 to £150 but most answers averaging around £50-£80 mark.

On the other hand, those six who answered Developing:

- Five answered 'Yes' and 1, 'No' to more interesting ways to perform
- Five people purchased music equipment once or twice a year whilst one person purchased every 2/3 months

This data could suggest the Developing musicians need not to make many purchases per year, possibly because of a smaller disposable income and so only the necessities are needed. For the last question about a suggest price point, the result averaged at the £70 mark similar to the Advanced musicians.

Nine respondents who replied to another question concerning their interest in such a product (ranging from 1-10, 1 being the lowest and 10, the highest), only two expressed a 4 or below.

Raspberry Pi with Wii Remote

This Raspberry Pi stage saw the project being split into 3 tasks:

- **Task 1:** Connecting the Wii Remote to the Pi
- **Task 2:** Find a demo synth that could accept MIDI information from USB MIDI Keyboard and get it to play through the USB Audio Device
- **Task 3:** Get the Synth to interact with the Wii Remote

At this stage the Pi was connected to a computer monitor and had a USB keyboard, mouse, MIDI keyboard connected to it.

Task 1

Gus' tutorial from <https://pimylifeup.com/raspberry-pi-wiimote-controllers/> was used initially to connect the remote to the Pi. The process when broken down involved:

- Downloading the packages *python-cwiid* and *wminput*
- allowing non-root users access to input devices
- checking bluetooth status using *systemctl*
- creating a config file that assigns Wii Remote buttons to an action in Linux using the Linux text editor *nano*
- scanning for bluetooth devices and obtaining the remote's MAC address using *hcitool scan*
- creating a *connectwiimote.sh* script that allows the remote to initialise a connection with the Pi by pressing 1+2
- making the script executable by granting it full read/write/execute permissions via *chmod 775*



The package *wmgui* featured a GUI that showed the input messages coming from the Wii Remote.

Task 2

Many demo synths were researched and auditioned. From the table below it was evident that *synthv1* or *amsynth* would be the best solutions but because the third task couldn't be completed, the full capabilities of these synths weren't explored.

Instrument	Raspberry Pi compatible?	Type of Instrument	ADSR Envelope?	Portamento/ Glide?	Cost
Sampler Box	✓	Sampler	?	?	Free for the software

FluidSynth	✓	Polyphonic Synth	X	✓	Free
amsynth	✓	Polyphonic Synth	✓	✓	Free
ZynAddSub FX	✓	Polyphonic Synth	✓	✓	Minimum \$45
synthv1	✓	Polyphonic Synth	✓	✓	Free

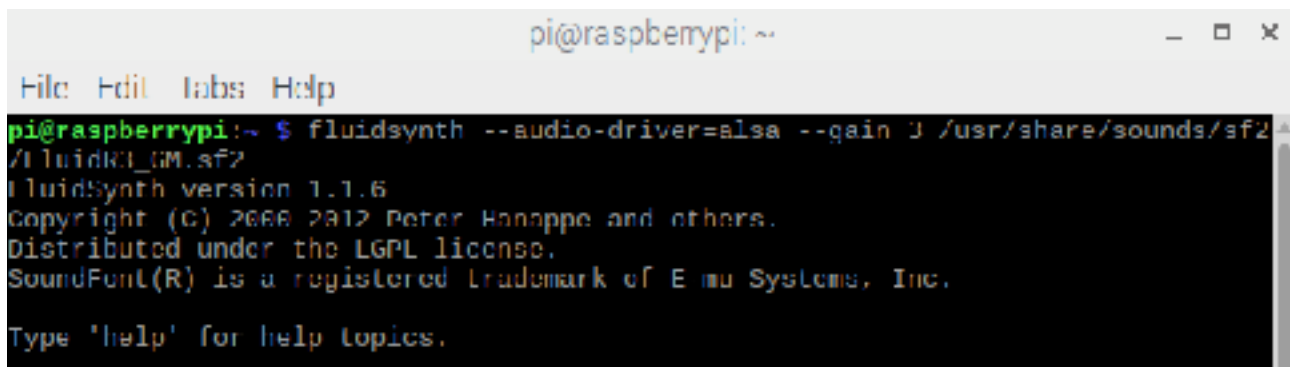
For the initial purpose of getting sound outputting to the USB audio device and connecting a MIDI keyboard to, the author went for FluidSynth because it wasn't so CPU extensive as Synthv1 or amsynth. Also it was quick and easy to obtain and execute in the terminal using:

```
sudo apt-get install fluidsynth
```

Then the following command launched a stereo instance of FluidSynth with the intention to use the built-in ALSA audio driver:

```
fluidsynth --audio-driver=alsa --gain 3 /usr/share/sounds/sf2/FluidR3 GM.sf2
```

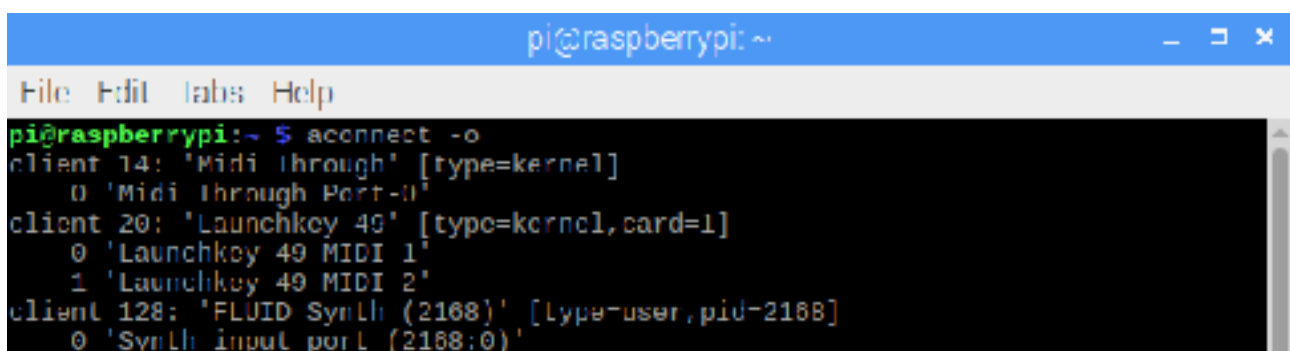
The following screen appeared, confirming that FluidSynth is operational:



```
pi@raspberrypi: ~
File Edit Tabs Help
pi@raspberrypi:~$ fluidsynth --audio-driver=alsa --gain 3 /usr/share/sounds/sf2/FluidR3 GM.sf2
fluidsynth version 1.1.6
Copyright (C) 2000-2012 Peter Hanappe and others.
Distributed under the LGPL license.
SoundFont(R) is a registered trademark of E-mu Systems, Inc.
Type 'help' for help topics.
```

A list of all MIDI devices connected and initialised appears when you execute the following :

```
aconnect -o
```



```
pi@raspberrypi: ~
File Edit Tabs Help
pi@raspberrypi:~$ aconnect -o
client 14: 'Midi through' [type=kernel]
  0 'Midi through Port-0'
client 20: 'Launchkey 49' [type=kernel, card=1]
  0 'Launchkey 49 MIDI 1'
  1 'Launchkey 49 MIDI 2'
client 128: 'FLUID Synth (2168)' [type=user, pid=2168]
  0 'Synth input port (2168:0)'
```

Here we can see that the Pi has detected both the MIDI keyboard and FluidSynth. In order to connect the two together, we'll need to keep in mind their client numbers (20 and 128) and run some code to patch them together:

```
aconnect 20:0 128:0
```

Pressing keys now on the MIDI keyboard plays piano sounds.

Task 3

At this point of development, the author discovered Programmable Input Emulators (PIE) which bridged the gap between different input devices, in our case the input of the Wii Remote and the MIDI of FluidSynth.

Many PIEs were discovered including the outdated WiiMidi that didn't support the latest Raspbian and GlovePIE and FreePIE, both Windows-only. There was also the MacOS input emulator, OSCulator which had better Wii Remote support with MIDI than the other options. With no access to a Windows computer, this meant the author couldn't continue working on the Linux platform.

OSCulator with Logic

'Open Sound Control (OSC) is a protocol for communication among computers, sound synthesizers, and other multimedia devices that is optimized for modern networking technology. Bringing the benefits of modern networking technology to the world of electronic musical instruments, OSC's advantages include interoperability, accuracy, flexibility, and enhanced organization and documentation.' (Anon n.d.)

In OSCulator, the program accepts the Wii Remote's OSC message and convert it into an Event Type of the user's choice. In our case of triggering certain functions of a synth, we will be dealing with the Event Types: MIDI CC, MIDI CC Toggle and MIDI Note.

MIDI Control Change is a message that consists of a value ranging from 0-127 and a controller number. The controller number will be assigned to a unique parameter in the synth such as the LFO Rate of the synth and the '0: pitch' OSC message (the vertical axis) being both assigned the unique number, 1. The value ranging from 0-127 depends on the parameters of the OSC message. For a button that is either on or off, the OSC value is either 0 or 1.0 and thus converted to a MIDI value, either 0 or 127. For a slider that can exist anywhere between the lowest setting to the highest, the OSC value ranges anywhere between 0 and 1.0; and 0 and 127 for MIDI.

On the author's MacBook Pro, Logic Pro and OSCulator is booted up and the synth plugin Omnisphere is being run on a MIDI track in Logic.

Method

1. Make sure the Mac's bluetooth on and then in OSCulator's settings window, click on the option to 'Pair a Wiimote'.

2. On the Wii Remote press buttons 1+2 together to start the pairing indicated by the 4 flashing LEDs. The flashing should stop and be replaced by a solid light on the first LED when they have paired.
3. Pressing all the buttons on the remote will have OSCulator detect all input messages (button presses and movement in any of the three axes). All of this will be shown in the main window. The 'Message' column should detect what the specific message is, for example, the 'A' button is named '/wii/1/button/A'. The input message is indicated by the box column just left of the 'Message' column, green light means an event is being triggered whilst a yellow light means nothing is performed but is detected. Red light means there is an error. The eye icon found at the top right of the main window opens the 'Quick Look' window, an activity monitor illustrating the numerical values of a single message as a graph.
4. To control the wobble rate of the synth using the up and down movement of the remote, find the message '0: pitch' under the message dropdown of '/wii/1/accel/pry'. Choose the event type of 'MIDI CC' and assign both the Channel Value and Value to 1
5. In Omnisphere, select a synth that can play sustained notes (in the author's example, a monophonic Square bass synth) and select the source of Modulation as the 'LFO1 Rate' and the target, 'Filter -> Cut-off'.
6. Right click on the LFO Rate knob under the LFO section and click 'MIDI CC Learn'. The movement of the remote in the Y-axis should now be turning the LFO Rate knob.
7. Back in OSCulator, let's change the Event Type of the A button (/wii/1/button/A) to MIDI Note and assign it to C2. Also change the Channel Number to 1. Now pressing A, the assigned note will play and the rate of the Wobble will be determined by the vertical positioning of the remote
8. Should you ass

The following link is a video tutorial and demonstration of the Wii Remote interfacing with Omnisphere - <https://youtu.be/UUrLZaEtP9U>

User Testing

A first time testing by the author is seen in the following link: <https://youtu.be/0z0F-R6d0l0>

When first trying OSCulator with Logic, the primitive idea was to use a MIDI Keyboard alongside the Wii Remote - the Keyboard to play the note and the remote to control LFO Rate and Glide (this is shown in the video). This is another option that users can explore as the remote is limited to 11 assignable buttons and the Keyboard can compensate for the amount of notes. Other options, though not tested, include the peripherals that work in conjunction with the Wii Remote like the Nunchuk or the Guitar Hero Guitar controller which can add more assignable buttons or even the 360 degree analog stick.

On April 2nd 2019, this iteration of the instrument was presented and demonstrated to Helen and colleagues. Few colleagues tested the product, garnering positive responses all round. Specifically colleagues cited the ease of controlling the Wobble rate and how responsive/real-time it felt. One person from the class commented how difficult it was for them to automate LFO rate at home - how they would have to draw the automation in their DAW, but if they knew a product like this existed then they would definitely purchase it. A friend of the author spoke about how it makes her think of the musical capabilities and how it automatically makes you want to create. However most comments spoke about its use in a recording environment and not for a live performance.

PureData

Because most of the work done so far was interfacing different existing softwares with each other, the author wanted to attempt the creation of a synth that would work with the Wii Remote like he originally intended using the Raspberry Pi. However the time it took trialling and testing the Raspberry Pi section was far too taxing and the author would work up something quick.

Knowing that there isn't another Programmable Input Emulator available other than OSCulator, the author limited his searches for a programming solution to MacOS. PureData was chosen because it was free and had many up-to-date resources available on the internet that would assist with creating a synth with the capabilities of a filter and an ADSR envelope, and the ability to communicate with the Wii Remote.

The author followed these tutorials for the simple synthesiser featuring said capabilities:

- <http://libremusicproduction.com/tutorials/creating-simple-synthesizer-pure-data-%E2%80%93-part-i>
- <http://libremusicproduction.com/tutorials/creating-simple-synthesizer-pure-data-%E2%80%93-part-ii>
- <http://libremusicproduction.com/tutorials/creating-simple-synthesizer-pure-data-%E2%80%93-part-iii>

Most of the time that the author spent in PureData was learning the different objects and their functions and so there was not much individual code unfortunately. Also a big part of it all was getting PureData to accept the MIDI input coming from OSCulator. Instead of writing out instructions that can be found in the tutorial, the author will annotate each patch discussing functionality and explaining key pieces of code.

GUI WINDOW

Attack is the time it takes for the signal to reach full amplitude. The slider scale will be set to Logarithmic which is a better indicator of a perceived change than a Linear scale as is the measure of Decibels (dB). The slider has a range from 10ms to 10,010ms.

'Number' objects, each connected to the outlet of each the Attack, Delay, Sustain and Release sliders.

Decay is the time it takes for the full amplitude level to reach the sustain level. Also set to Logarithmic and has a range from 10ms to 10,010ms.

Sustain is the amplitude maintained whilst a tone is being played. Set to Logarithmic, it has a range of 0.1 to 1.1.

Release is the time it takes for the sustain level to reach an off-condition level. Set to Logarithmic and has a range from 10ms to 10,010ms.

Volume is set to a linear scale and has a range of 0 to 1.

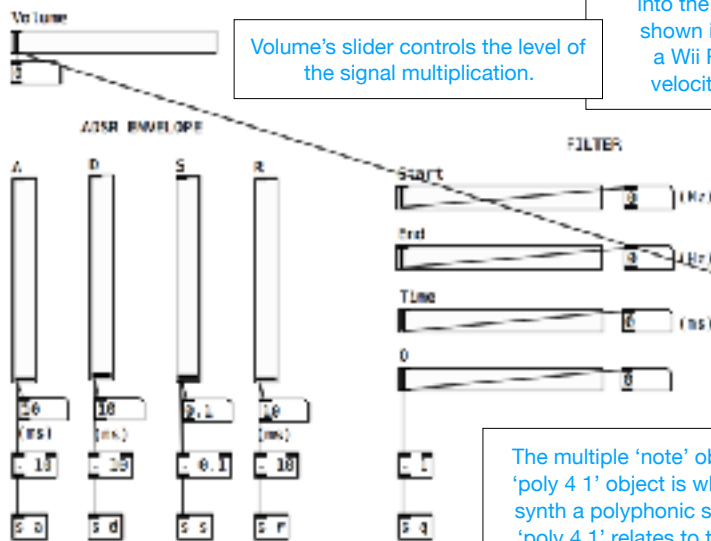
Start relates to the Starting frequency that the bandpass filter begins at.

End relates to the frequency that the bandpass filter will end at. When a note is sustained, the filter will remain at this frequency until release.

Time, in milliseconds, relates to the time taken for the Start frequency to reach the End frequency.

The Q factor is the width of the sweeping Bandpass filter.

MAIN PATCH



Volume's slider controls the level of the signal multiplication.

notein detects the MIDI note being played into the synth. From the demonstration, OSCulator will be the input MIDI device and if the buttons are assigned to the Event Type, MIDI note, then the MIDI note converted into the frequency in Hertz will be shown in the note number. If it is a Wii Remote button pressed, velocity will either be 0 or 127.

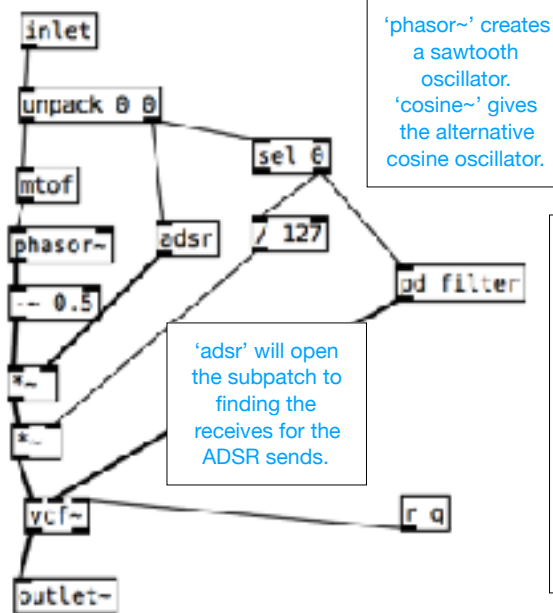


Objects 's a', 's d', 's s' and 's r' are sends for each of the ADSR parameters. The receive objects are found in the 'adrs' subpath and some of sustain's receives are found in the 'pdfilter'.

The multiple 'note' objects and the 'poly 4 1' object is what makes the synth a polyphonic synth. The 4 in 'poly 4 1' relates to the amount of voices in the synth, and the 'route 1 2 3 4' object routes each of the voices to a duplicates of the 'note' subpatch that all contain the parameters for the sound of the note.

'dac~' is the audio output that you have set in the 'Audio Settings' window of PureData. It has two inlets to signify a stereo channel.

'note' SUBPATCH

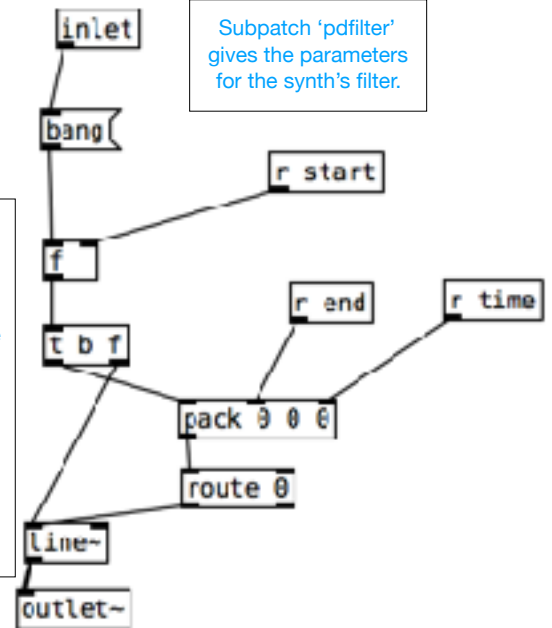


'phasor~' creates a sawtooth oscillator.
'cosine~' gives the alternative cosine oscillator.

'adsr' will open the subpatch to finding the receives for the ADSR sends.

'vcf~' is the voltage-controlled filter with each inlet assigned a different task, the signal you're applying the filter to, signal data to control the cutoff frequency, the control data input for the Q-factor, respectively.

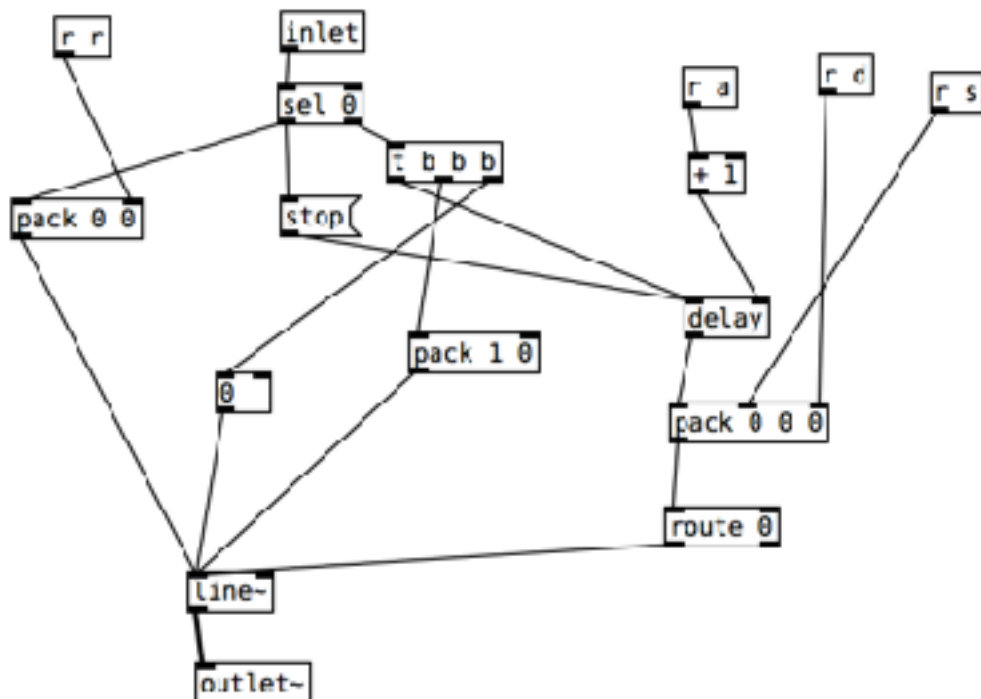
'pdfilter' SUBPATCH



Subpatch 'pdfilter' gives the parameters for the synth's filter.

'bang' triggers when a note is played, triggering the Start slider value, here listed as the object 'f'. This will be used to set the start value for the

'adsr' SUBPATCH



Video Demonstration and Tutorial

The following links to a video by the author to demonstrate the sounds of the instrument by changing the filter and the ADSR parameters: The screen capturing software unfortunately distorted the audio but testing the accompanied PureData code with OSCulator should garner better audio quality from the synthesiser.

1. With OSCulator open, Wii Remote paired and with different MIDI notes assigned to each of the buttons, check that your desired audio listening device is selected in the output found in PureData's 'Audio Settings' found under the Window dropdown menu. In 'MIDI Settings' make sure that 'OSCulator Out' is selected in the Input Devices and 'OSCulator In (PORTNUMBER)' in the Output Devices. The port number is found on the top left of OSCulator's main page and the default number is 8000.
2. Also in the Window dropdown menu is the 'Test Audio and MIDI' which plays a test tone to see if audio is being correctly outputted to your listening device.
3. In the video, we have the main patch opened which features the 'notein' object with the outlets going to 'note number' and 'velocity'. We can also find this in the 'Test Audio and MIDI' window. Press a MIDI note-assigned button on the remote and you'll know if the MIDI information is being recognised by PureData if the '0' objects under 'stripnote' change.
4. Play with the parameters of the Filter and the ADSR envelope with the volume set to a reasonable level. Pressing multiple notes on the remote simultaneously allows chords to be played.

Unfortunately, the author could not get the modulation of the LFO rate to work but definitely sees how one could automate the Q-factor to work with Wii Remote's pitch.

Discussion and Conclusion

The project evolved many times in order to get a result worth demonstrating, unfortunately at the time of writing, the project is left as a proof of concept with plenty to be desired. From the survey, a price point of £70 seems the most expected by consumers. If done on a Raspberry Pi that costs around £50, a profit of £20 per unit doesn't seem like much.

Helen Leigh suggested that the Bela computer (retailing at £60 for the Mini and £120 for the starter kit) would be a better fit for the project instead of the Raspberry Pi though it would need more investigating.

Other suggestions and considerations for the project moving forward which could expand the interest/increase the price point include:

- suitable for more than just bass instruments
- plug and playability, ease of use on stage
- rechargeable battery
- small screen

Should the project continue, the author would want to work on creating an easy to use GUI in PureData and perhaps design his own PIE to not rely on OSCulator.

References:

- <http://andrewdotni.ch/blog/2015/02/28/midi-synth-with-raspberry-p/>
- <https://gizmodo.com/a-beginners-guide-to-the-synth-1736978695>
- <https://www.soundonsound.com/sound-advice/q-whats-best-way-create-sub-bass-synth-sounds>
- <https://flypaper.soundfly.com/produce/5-secrets-making-your-sub-bass-notes-audible/>
- <https://www.rubbermonkey.co.nz/Music-Audio/Musical-Instruments/Keyboards-Synthesizers/Teenage-Engineering-PO-14-Pocket-Operator-Sub-Bass-Synthesizer>
- <https://delightfullylinux.wordpress.com/2016/02/09/usb-audio-and-retropie-3-4-for-better-sound/>