

# Development Module

## Assignment 2b - Instrument Essay

Richard Buck - 1917533

### **Introduction:**

For the instrument creation task this module I chose to build a software version of a vocoder called 'voDDPcoder'. This was completed in Native instruments Reaktor software using both primary and core. This Essay will take you through the process of why I chose this particular instrument and software, the challenges I faced, relevant research and reflection on my finished product.

### **Why a Vocoder in Reaktor?:**

"The human voice can be thought of as the ultimate melodic instrument, because it is capable of instant expression with no instrument required to translate thoughts and feelings into sound" - ([www.learntoplaymusic.com](http://www.learntoplaymusic.com)). From the first song ever performed by a human no instrument can replicate its timbre, ability to pronounce words and the uniqueness of each individual. Vocoder can merge the human voice with computerised sounds or other instruments to give endless new sounds and effects that can be used in many different environments. These are the reasons why I chose to create my instrument, as it has always fascinated me how they work, and by understanding every element to these instruments I could use them very effectively in my productions or future works in video games and film.

You can build a vocoder within a software modular editing environment like Reaktor, Max or synth edit. Alternatively you can go the hardware route. Due to costs and equipment limitations software is an obvious choice for this project. When deciding on which software environment to create the vocoder I researched each one and decided that Reaktor would be best suited to me because:

- The layout of the environment and blocks used to create seemed logical to me
- As I was new to each software I felt learning code like in pure data would be a longer learning curve.
- The cost of £80 would also give me multiple instruments and effects to use in productions and it's easy to open them up to see how they work.

Once I had decided on the instrument and software I needed to have a plan on how I will go about creating the vocoder. Below shows the stages I went through to get to the finished product:

### Understanding the fundamentals:

Vocoders were first invented back in the 1930's by Bell laboratories with the idea of helping telephone lines accept more data. This technology was quickly snapped up by the military to send coded messages for Churchill and Roosevelt during WW2. Other uses stemming from the initial invention was synthetic voices that allows people like Stephen Hawkins to communicate - ([www.izotope.com](http://www.izotope.com)).

Musically the first use of this technology was on the track "A Clockwork Orange" back in 1971. Then through to the 90's where electronic music developed massively it was used on some big electronic tracks, namely "The Robots" by Kraftworks, "O Superman" by Laurie Anderson, "pony" by Ginuwine (produced by Timbaland) and featuring on the hit movie Magic Mike XXL 20 years later and of course Daft punks hits "Around the World" and "Harder, Better, Faster, Stronger".

The vocoder is not obviously used in most modern music like a few decades ago however when used effectively it can be used to give tracks an originality and shine in the mix. Also vocals are swapped out for drum loops or other instruments to make unique sounding beats and melodies which can help make a track stand out from the rest.

So how do they work? In general two audio inputs (modulator and Carrier) are split into multiple frequency bands. Using an envelope follower these are then combined back together so that the modulator shapes the sound of the Carrier. Normally the Modulator would be a vocal sample or directly hooked to a microphone and the carrier would be a synthesiser. However these can both be any kind of audio. Figure (a) shows a basic diagram of how this works - ([www.splice.com](http://www.splice.com)):

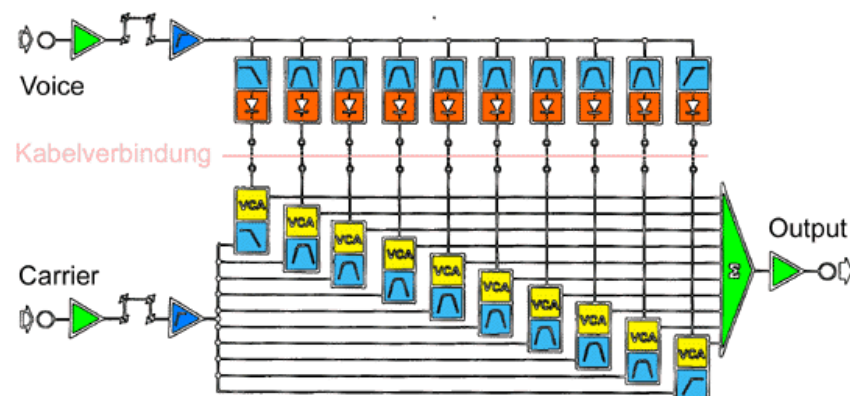


Figure (a)

Multiple bandpass filters are used to separate the audio and depending on how many are used affects the quality of the audio output signal. A majority of the original hardware vocoders would have anywhere between 8 and 12 bands. The famous Roland VP-330 was a 10 band vocoder and this amount of bands gives a retro type sound. Modern Vocoder can now have nearly unlimited bands depending on computer performance. The DAW Reason comes with a 512 band vocoder! Most modern vocoders now allow you to change the amount of bands with a simple knob or switch. The higher number of bands mean a very good quality output, so vocals would be easily identifiable (if that is what was being used as the modulator).

It is important to consider what sound is being used as the carrier. The frequency content must be a wide range, with a lot of mid/high. This ensures that there is enough information for the vocoder to give a decent output. Also, should the vocoder have a synth or sound library already included or would this need to be connected externally? Whichever is used, generally if a synthesiser is used for the carrier Saw and noise oscillators are generally the best to use due to the high frequency content - ([www.splice.com](http://www.splice.com)).



Figure (b)



Figure (c)

Figure (b) shows the Roland VP-03, this has preset synth sounds, namely strings and choir patches. A midi keyboard can be connected to control these sounds as the carrier, alternatively a completely different synth can also be connected. What surprises me is the limitations of the sound design parameters on a lot of the hardware vocoders. Figure (c) shows Sonivox's Vocaliser Pro. This has 4 spectral synthesis modules with 16 different synthesis types, also 16 different filter configurations. Coupled with the LFO modulation and other variable parameters the opportunities to create unique carrier sounds are massive.

## **Sound Manipulation:**

To get a good sounding vocoder, it is not just the carrier that needs to have a wide frequency range and good harmonic content. The modulator is also important. The quality and style of the audio sample or drum loop etc needs to be high however this is more down to the user. When creating a vocoder it is essential that the modulator is processed in certain ways to help the final output sound strong.

Many good vocoders allow an un-vocoded part of the modulator to pass - ([www.izotope.com](http://www.izotope.com)). This is high pass filtered and helps distinguish specific words when vocals are used or specific transients in drum loops. Also a formant or pitch shifter can be found on a lot of modern software or hardware Vocoder, this is used to shift the modulators pitch prior to being processed in the circuit. Generally a lower pitch or formant yields greater results due to a similar robotic sound when pitching down vocals.

Other than pre processing before the vocoder circuit, most hardware Vocoder will have an ensemble option which in essence is a chorus effect added to the vocoded signal, this helps fatten up the sound and increase the stereo image. Also careful EQ helps sculpt the output sound.

## **My Aims:**

- Create a basic working vocoder
- Bands are to be adjustable
- A basic synth to be integrated into design
- Options to modulate parameters
- A Pitch or formant shifting control
- Post processing effects/EQ
- User friendly interface

## **Basic Vocoder and band adjustment:**

Firstly I had to create a working Vocoder instrument inside Reaktor. My plan is to create this and then add the other elements around this. So I needed a way of splitting 2 audio signals into multiple bandpass filters. The modulator filters need to be followed by an envelope filter with the output of this multiplied to the corresponding carriers bandpass filter. Once these are added back together we should hear a vocoder effect. My research found 2 different methods for this using either primary or core. The primary method used a voice

information module followed by some maths to split a designated frequency spectrum into a specific number of bands, this could also be adjusted with a simple knob control.

Figure (d) shows my attempt at this design:

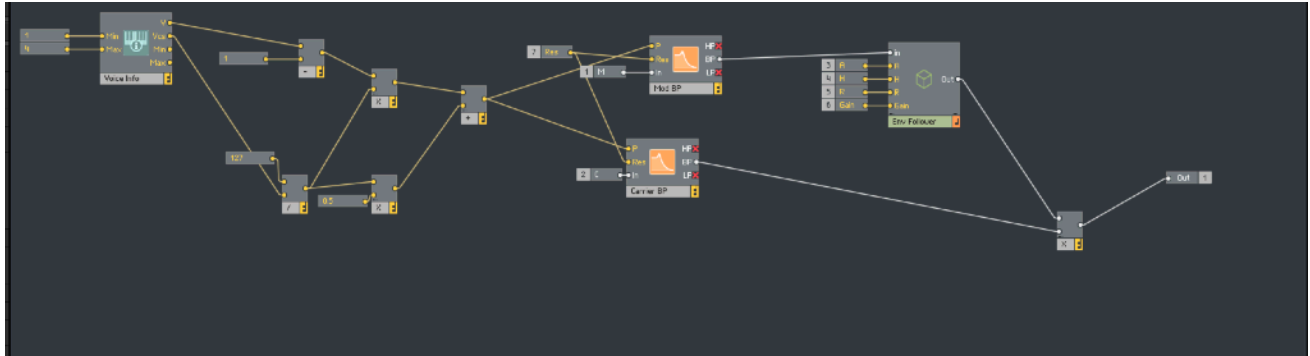


Figure (d)

This was based on a 4 bands to begin with. The math breaks down the 128 midi notes into the relevant number of frequency bands. By changing the max input on the voice info module the bands are instantly split into equal bands over the frequency range. I liked this idea as it would make the aim of selecting bands very simple for the user. The bandpass filters for the modulator has been connected to an envelope filter macro to shape the carrier waves once multiplied before output. This whole section was then duplicated so I had one for the left and right channels.

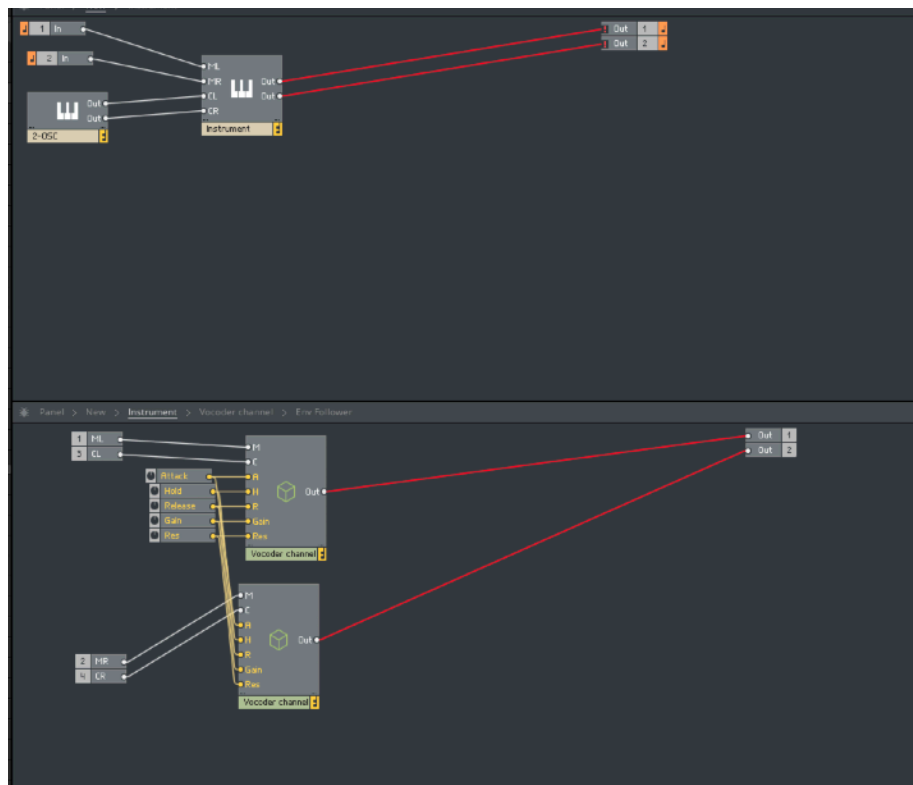


Figure (e)

Unfortunately there was an error with this method and I struggled to get the outputs to accept the audio from the vocoder instruments. It turns out this was to do with the monophonic and Polyphonic settings for each module. Figure (e) shows the error that was being shown. For this reason I attempted the other technique which required the filters to be created in a core cell.

For this method no voice info module is used and each band pass filter has to be copied and inserted in the cell. As I wanted to have at least a 32 band vocoder I began to create the core cell. A 32 band vocoder actually requires 64 band pass filters, 32 for the modulator and 32 for the carrier. This was a large circuit and had some trouble with the wiring, however managed to get the vocoder to work with an audio sample and a built in Reaktor synth.

The first issue to deal with was the drain on CPU. Reaktor was struggling and giving a lot of crackles and pops. To fix this I downgraded the core cell to half the filters, giving me a 16 band vocoder. This worked much better and the sound quality was very similar just without the pops and crackles.

The next issue was to try and add a method of changing the band pass filter locations and width, this in essence would allow the user to change the number of bands affecting different frequency ranges. This was achieved by adding a control to the base frequency and interval length. This allows you to choose where the first band pass filter starts from in the frequency spectrum and then how wide each band pass filter is. So by increasing the interval gives the affect of reducing the number of bands. Figure (f) shows the circuitry used for this on each filter:

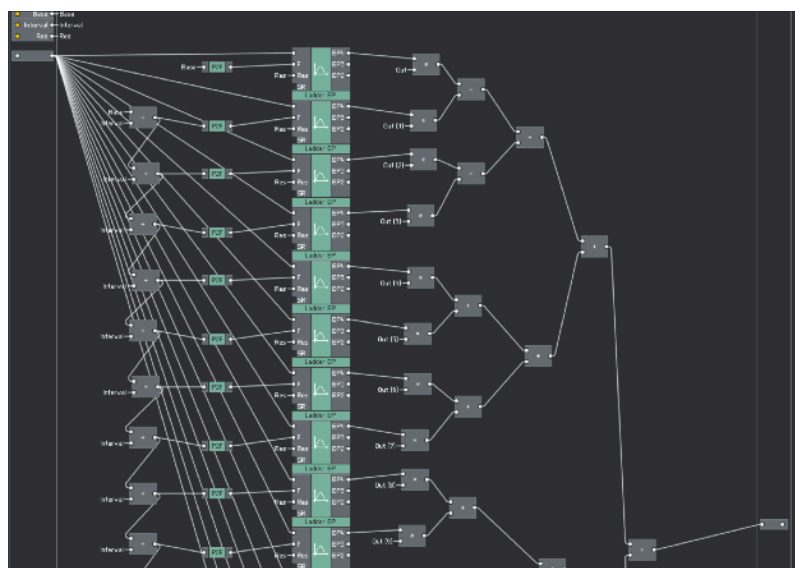


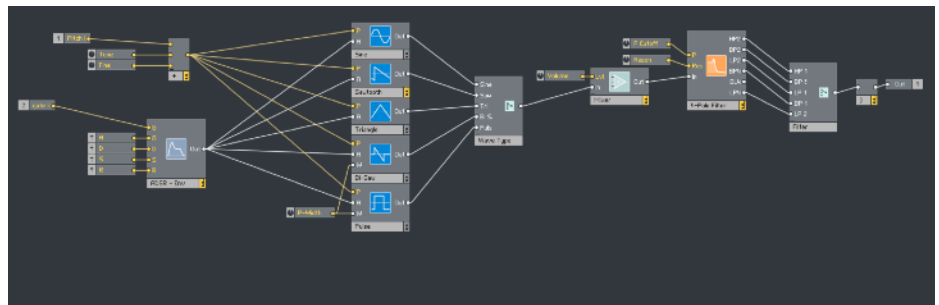
Figure (f)

### Synth Integration:

Now the vocoder was working I needed to create a synthesiser within the ensemble to allow the user to create sounds ‘straight out the box’. I had much more experience with subtractive synthesis so chose to implement this method into the synthesiser. A subtractive synth is quite simple, using filters, envelopes, modulation and effects to create sounds. After watching a few tutorials on this for Reaktor I decided I wanted a 3 oscillator synth. One of these oscillators needed to have a noise option. Vocoder can use a filtered noise oscillator to help improve the high end sounds of the human voice and fills in gaps in the frequency range that the standard oscillators may miss. This can give the vocoder output a much more detailed image.

When adding the amplitude envelope and filters I made an error and ended up with an envelope and filter on each individual oscillator. This could have been modified to work on all 3 together like most standard synths however I felt this made it slightly more original and also allowed for more detailed sounds to be created with the amplitude. An example is having each oscillators attack set at different rates which could help make a string patch sound more like an ensemble with multiple instruments playing different attacks.

Oscillator (filters  
and envelopes) ———  
Figure (g)



I ensured the user would be able to select an appropriate filter for the oscillators so used a multi 4-pole filter with a selector module. Generally sounds for a vocoder want to be quite unfiltered to ensure maximum effect from the modulator however with multiple oscillators different sounds can be used in different ranges.

Next I wanted the user to have a portamento option so that there can be a glide between notes played. This can help give the vocoder a ‘singing’ sound. This was created

using a 1-pole filter attached to the note pitch with a control knob to adjust the amount of glide. Figure (h) shows the completed Synth Instrument.

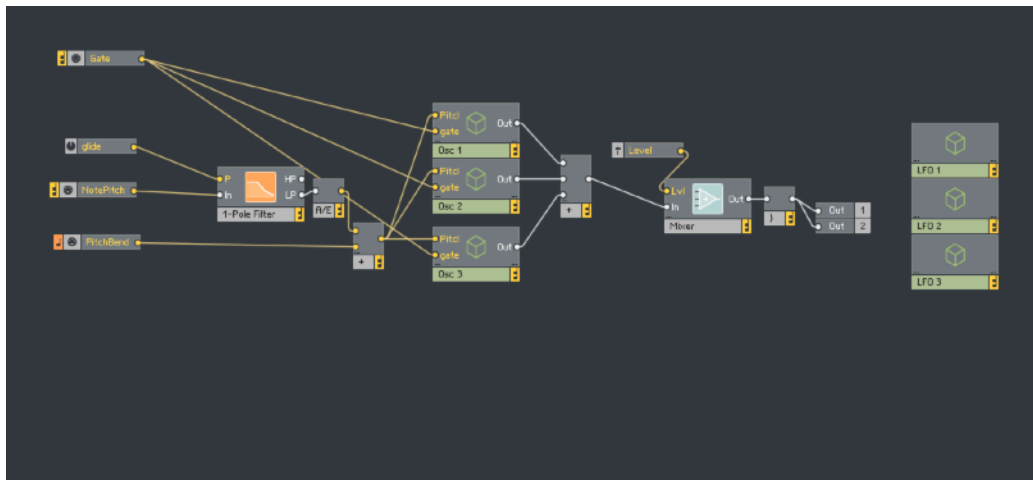


Figure (h)

### Modulation:

As you can see from figure (h) there are 3 LFO macros incorporated into the synth. I chose to have 3 as it made sense to have the controls next to each oscillator. In theory the more LFO's the better as possibilities to the sound design just increase. The 3 LFO macros were designed so they can be connected to any parameter in the ensemble and also multiple parameters can be selected for each oscillator. This was simple to achieve with an IC send module.

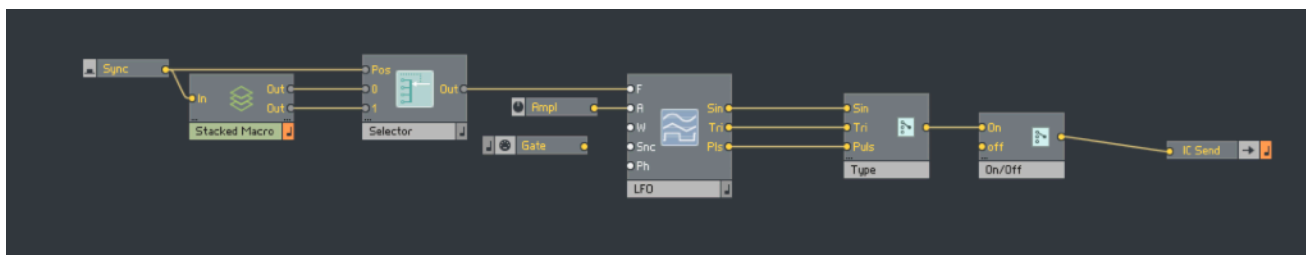


Figure (i)

By using the tempo info module which sends data at a 1/4 beat I was able to set up the sync option which is inside the stacked macro in figure (i). So by using a selector module and multiplying the tempo info by different amounts I was able to control the speed of the LFO in

relation to the track tempo. This includes a simple switch which then connects the speed to a standard frequency control for further control if required. The LFO module is followed by a further 2 switches to select the LFO wave type and to turn the option on or off.

### Pitch Shifter:

One method of improving the outputted sound is to pitch shift the modulator audio after it has passed through the vocoders filters. Unfortunately as I have created my filters in Core, this was slightly out of my technical ability. To compensate I created a pitch shifter in primary by using ramp oscillators and an amplitude quantise oscillator. 2 single delay modules then allow the pitch to be changed without affecting tempo. The x-y power module was important to ensure that the pitch went up or down in semitones by ensuring a ratio of 1.05946 is applied to the pitch modulators as this is the exponential growth of frequency in relation to pitch. (See figure (j) for the completed Pitch shifter macro).

On completion of the pitch shifter CPU was again a big issue with the sound quality as pops and crackles were very loud and taking away from the effect. The CPU reading on Reaktor was only hitting 30%. I increased buffer size in Reaktor to 2048 and this solved the problem. However sound quality was now a worrying factor if I was to add many further parts to the ensemble

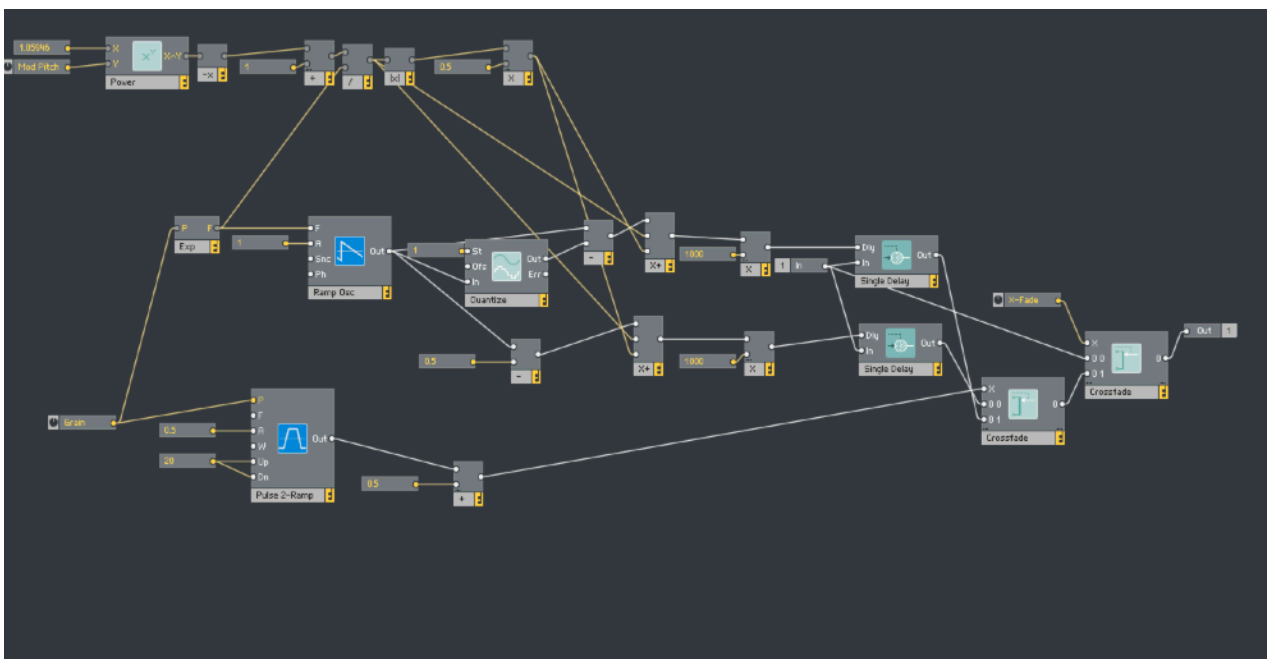


Figure (j)

## Additional Effects and EQ:

My initial plan was to add built in chorus and EQ to the Vocoder instrument. However due to the issues with CPU and audio quality I decided to include just the EQ. I wanted to create a 5 band Equaliser to sculpt the output signal as this is critical in finalising a polished sound. This means that any factory presets can have a bigger impact when loaded than just leaving it to the engineer using it in a production.

To create the 5 band Equaliser I imagined the old style set up on Hi-Fi's and boom boxes. Essentially 5 faders set to individual frequency ranges allowing the user to boost or dip where needed. This was very simple to create and consists of 3 bandpass filters with a Low shelf and high shelf on each end. Each filter I set controls for amplitude, frequency and Q values. See figure (k).

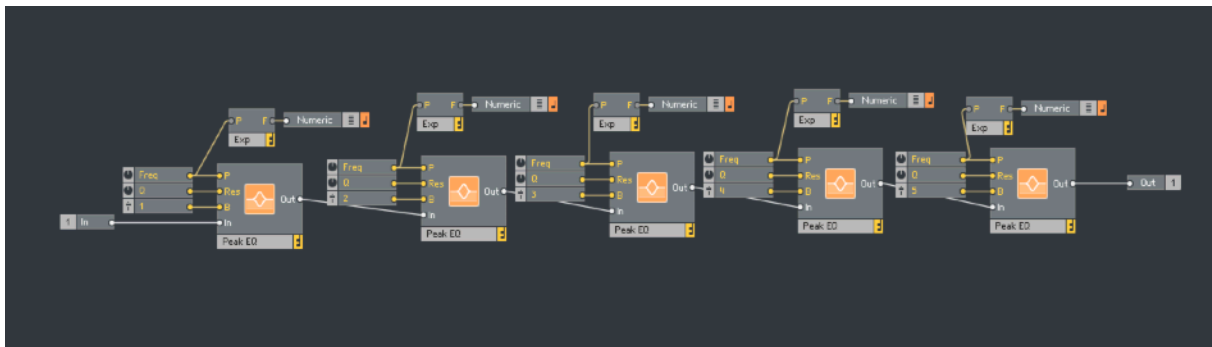
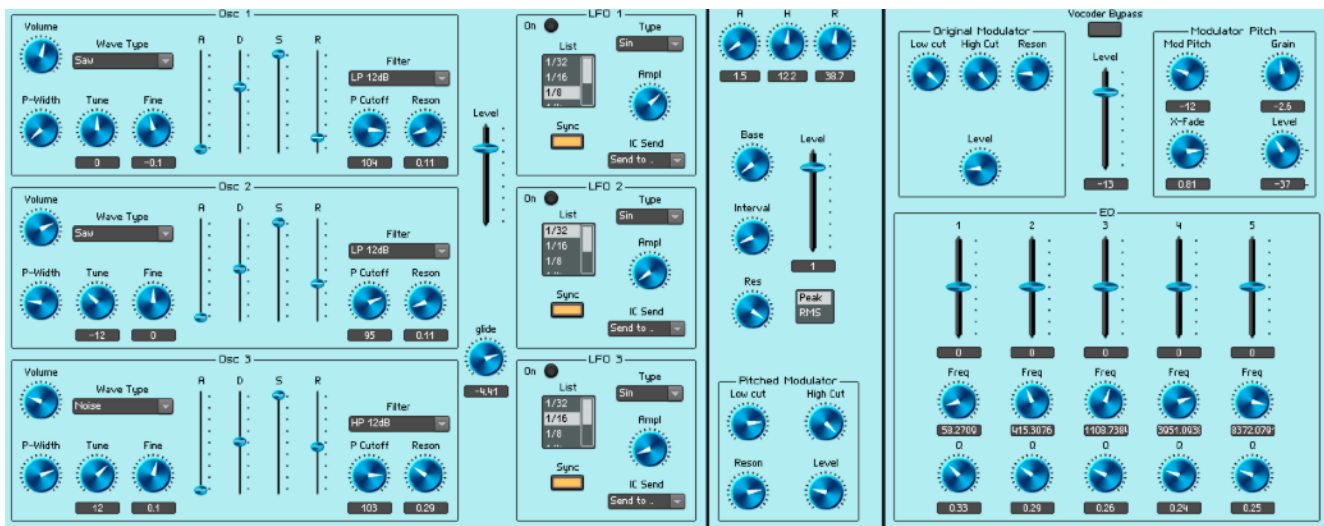


Figure (k)

## Graphical User Interface:

Now I had all the components in place I needed to create a nice looking, user friendly interface. This is very important as if it does not make sense or is messy this could distract the user from wanting to use the instrument. I decided to arrange the instrument left to right:

*Synth - LFO's - Vocoder - EQ/Effects*



I have coloured the panel so each instrument in the ensemble matched and was pleasing to the eye. To create the Knobs and sliders I used a programme called ‘Knobman’. This allowed me to create a unique style which is important to give the instrument its own identity. In whole I feel the instrument looks professional, organised and works how it should.

Improvements I would like to add is customising the drop down menus and switches, also the parameter readouts, as that way the whole GUI is unique. Other issues is the frequency readouts on the EQ are to 3 decimal places which I was unfortunately unable to change. The drop down menus for the IC send also seems messy so I would like to be able to change this to a more user friendly ‘touch parameter’ option to select the destination.

I would also like to have voDDPcoder stretched over instruments panel however I had issues trying to import images onto the interface. They were not movable and looked extremely messy so decided to leave it off.

## Beta Testing:

I sent the instrument to fellow students and friends in the industry to get feedback on what they thought of the instrument and got the following feedback:

- The vocoder is clearly laid out and easy to follow even without instructions
- The (A,H,R) controls top centre could be labeled as plenty of room
- Would be useful for the IC menu to display what is selected without re selecting
- Having the centre section labeled as not clear on what base/interval means
- The base and interval parameters help the tone and colour greatly
- Unsure how to deselect LFO sends
- Unsure on glide parameters

In general a majority of the feedback was on the user interface itself. Which I think is really good as it means the sound of the vocoder must have lived up to the users expectations. For my first instrument created in Reaktor I am happy with the end result however there are many things I would like to reflect on:

- **Continue to complete tutorials on basic and advanced processes:**

The tutorials are what allowed me to create the voDDPcoder and are very impressive, however there are just too many. By the time I had finished the project I had discovered why my initial vocoder method of using a voice info module did not work. This was due to using polyphonic modules and macros and trying to connect the to a monophonic output. This is something that can only be accomplished with an audio voice combiner module. Very simple but this took a while for me to figure out, and that was after discussing with fellow students.

- **Optimisation:**

I struggled with CPU issues during this build, and luckily I found a way round this with buffer sizes. This is something that can be gained with experience however some basics were missed in my original instrument. With the synth I had set a value of 16 voices. This was affecting the sound quality massively and once I had reduced to 8, a noticeable improvement was heard and is the setting I have used for the presentation demonstrating its performance. If I can reduce the total load on CPU I can add the chorus effect which is seen on so many vocoders as ensemble and give a great wider sound.

- **Synthesiser:**

As my main focus of the instrument was the vocoder sound itself, I did not fully dive into the world of synthesis in Reaktor. Moving forward I would spend a lot of time building a very advanced synthesiser. This way the user has so many options that the vocoder could be just a side function. With additional time, this is what would make it a far more superior product. Options would be different kinds of synthesis (wavetable, FM, Vector, additive, spectral etc), more ways to modulate parameters (envelopes assignable).

- **EQ:**

The 5-band equaliser I have incorporated into the design is extremely basic. I feel this could be so much better as EQ software today is very advanced. Ideally a graphical display showing the frequency curve would be a start. EQ's are a very specific art these days with many companies designing specialised units specifically for this reason. So in theory a majority of professional engineers would probably use their own EQ anyway. Even so I was aiming to make the voDDPcoder playable 'straight out the box' so I still feel some attention to this would benefit some users.

- **Bypasses and Faders:**

Whilst I was building the instrument I made a few bypasses and connections to allow for unmodulated sounds to pass through to the output un-vocoded. This I feel has made the interface slightly confusing and messy as it is slightly unclear on what each fader does and some of them can affect similar parts of the audio. I would definitely go back and re-wire this as I feel these options definitely help define the output but would focus more on how the user can control them.

Also the un-vocoded modulator signals need a switch of some sort so that they only play when the vocoder plays as currently this means more work for the recording artist as they would manually need to remove sections in-between takes and also would make live use very difficult.

- **Pre vocoder signals:**

I feel I could have spent some more time and effort into processing the carrier and modulator signals before they are run through the vocoder. These could have been compressed, saturated and EQ'd which would have made a much more stronger output audio signal. The next version of this instrument will definitely have this integrated into the circuitry but without the user controls. Maybe with just a switch which turns them on and off to the users preference.

## **Conclusion:**

All in all I am happy with the final outcome of my VoDDPcoder. I accomplished all that I set out to do except the integration on an effect (chorus). However I have a clear vision of how this idea can be developed into a much more efficient, detailed, professional instrument. I faced many challenges from the start however thanks to the The sounds I have been able to create with it have been inspirational and definitely usable in some sort of music production, be it a musical track, film or games. I have learnt a great amount of sound design fundamentals and circuitry required to build instruments and feel with some more dedicated research and learning I would be ready to create a product that is worthy of a price tag.

## **Bibliography:**

- Anagram, S. 2017. 'Building an analog modelled vocoder in Reaktor' [online video] - available from [www.reaktortutorials.com](http://www.reaktortutorials.com), viewed June 2020
- Anagram, S. 2017. 'Subtractive synthesis in primary' [online video] - available from [www.reaktortutorials.com](http://www.reaktortutorials.com), viewed June 2020
- Anagram, S. 2017. 'LFO's and sequencers' [online video] - available from [www.reaktortutorials.com](http://www.reaktortutorials.com), viewed July 2020
- Anagram, S. 2015. 'Add glide to any Reaktor project' [online video] - available from [www.reaktortutorials.com](http://www.reaktortutorials.com), viewed July 2020
- Anagram, S. 2016. 'Pitch shift and time stretch effect' [online video] - available from [www.reaktortutorials.com](http://www.reaktortutorials.com), viewed July 2020
- Anagram, S. 2015. 'Oscillators part 1 and 2' [online video] - available from [www.reaktortutorials.com](http://www.reaktortutorials.com), viewed June 2020
- Flipmu. 2013. 'Reaktor Tutorial:Building a vocoder' [online video] - available from [www.vimeo.com](http://www.vimeo.com), viewed June 2020
- Rewark, M. 2018. A Brief history of the Vocoder - available from [www.izotope.com](http://www.izotope.com), viewed June 2020
- LTP. 2013. Why the Human Voice can be considered the Ultimate Melodic Instrument - available from [www.learntoplaymusc.com](http://www.learntoplaymusc.com), viewed June 2020

- Rework, M.2018. How does a Vocoder work? - New York - available at [www.splice.com](http://www.splice.com), viewed June 2020