

Università Politecnica delle Marche

FACCOLTA' DI INGEGNERIA

Corso di Laurea in Ingegneria Elettronica

Insegnamento: CIRCUITI ED ALGORITMI
PER
IL DIGITAL SIGNAL PROCESSING

**STUDIO ED IMPLEMENTAZIONE DI
ALGORITMI DI EFFETTISTICA MUSICALE
IN AMBIENTE VCV RACK**

STUDY AND IMPLEMENTATION OF
MUSICAL EFFECTS ALGORITHMS
IN VCV RACK ENVIRONMENT

Tesi di laurea di:
Henry C. N. Philip

Relatore:
Prof. Stefano Squartini

Correlatore:
Dott. Leonardo Gabrielli

2020/2021

INDICE

1	Introduzione	1
2	Filtro a pettine FIR.....	2
2.1	Definizione, parametri di controllo, eq. alle differenze e FDT.....	2
2.2	Risposta nel tempo, caratteristiche in frequenza e codice Matlab	3
3	Effetti audio basati sul ritardo ed implementazione in Matlab.....	4
3.1	Vibrato.....	4
3.1.1	Definizione, implementazione e valori dei parametri di controllo.....	4
3.1.2	Codice matlab del vibrato.....	5
3.1.3	Commenti	6
3.1.4	Esempio e grafico ampiezza/tempo.....	7
3.2	Chorus	8
3.2.1	Definizione, implementazione e valori dei parametri di controllo.....	8
3.2.2	Codice matlab del chorus.....	9
3.2.3	Commenti	10
3.2.4	Esempio e spettrogramma.....	10
3.3	Flanger.....	12
3.3.1	Definizione, implementazione e valori dei parametri di controllo	12
3.3.2	Esempio e spettrogramma.....	13
4	implementazione in VCV Rack.....	14
4.1	Descrizione della piattaforma e dei moduli utilizzati.....	14
4.2	Implementazione del vibrato.....	26
4.3	Implementazione del chorus.....	27
4.4	Implementazione del flanger.....	28
4.2	Esempi e spettrogramma.....	29
5	Conclusioni.....	33
6	Bibliografia.....	34

Capitolo 1

Introduzione

Negli spazi acustici possono verificarsi dei ritardi. Un'onda sonora riflessa da un muro sarà sovrapposto all'onda sonora alla sorgente.

Se il muro si trova ad una grande distanza (es. una scogliera) sentiremo l'eco, mentre se il muro è abbastanza vicino a noi, noteremo i riflessi attraverso una modifica del colore del suono. Possono apparire riflessi ripetuti tra confini paralleli. In una stanza, tali riflessi saranno chiamati eco piatta.

La distanza tra i confini determina il ritardo che viene imposto ad ogni onda sonora riflessa.

In un cilindro, si svilupperanno riflessioni successive ad entrambe le estremità. Se il cilindro è lungo ascolteremo uno schema iterativo mentre, se il cilindro è corto, sentiremo un tono acuto.

Gli equivalenti di questi fenomeni acustici sono stati implementati come unità di elaborazione del segnale (Udo Zölzer 2002, 63).

I primi delay (ritardi) erano strumenti analogici che usavano un supporto magnetico per esempio un nastro, per registrare il suono in ingresso in modo simile al funzionamento di un normale registratore a nastro; un esempio di un delay di questo genere è il celebre Roland Space Eco. Questo sistema aveva il difetto di comportare la progressiva usura del nastro impiegato. Tuttavia lasciava anche la possibilità a musicisti con competenze tecniche di agire in modo estremamente creativo sulla riproduzione dei suoni.

A partire dagli anni ottanta il mercato ha sostituito rapidamente i delay analogici con i delay digitali che digitalizzano il suono in ingresso e lo memorizzano in un circuito che ha caratteristiche analoghe a quelle della memoria RAM dei computer. Questa tecnica presenta il vantaggio di non comportare virtualmente alcun tipo di usura dei componenti, consente una riproduzione più fedele del suono originale e, attraverso la digitalizzazione, può facilmente essere integrata con altri strumenti di elaborazione del suono, Per esempio, molti delay digitali forniscono anche funzioni di vibrato, flanger, chorus o altri effetti legati alla modifica della forma d'onda del suono riprodotto (Wiki Delay).

Capitolo 2

Filtro a pettine FIR

La rete che simula un singolo ritardo è chiamata filtro a pettine FIR (Fig. 1). L'ingresso viene ritardato di un determinato periodo di tempo. L'effetto sarà udibile solo quando il segnale elaborato viene combinato (sommato) al segnale di ingresso, che in questo caso agisce come riferimento.

Questo effetto ha due parametri di riferimento:

la quantità di ritardo T e l'ampiezza relativa del segnale ritardato a quella del segnale di riferimento .

l'equazione alle differenze (1.1) e la funzione di trasferimento (1.2) sono date da:

$$y(n) = x(n) + g*x(n - M) \quad (1.1)$$

Con $M = T/fs$

$$H(z) = 1 + g*z^{-M} \quad (1.2)$$

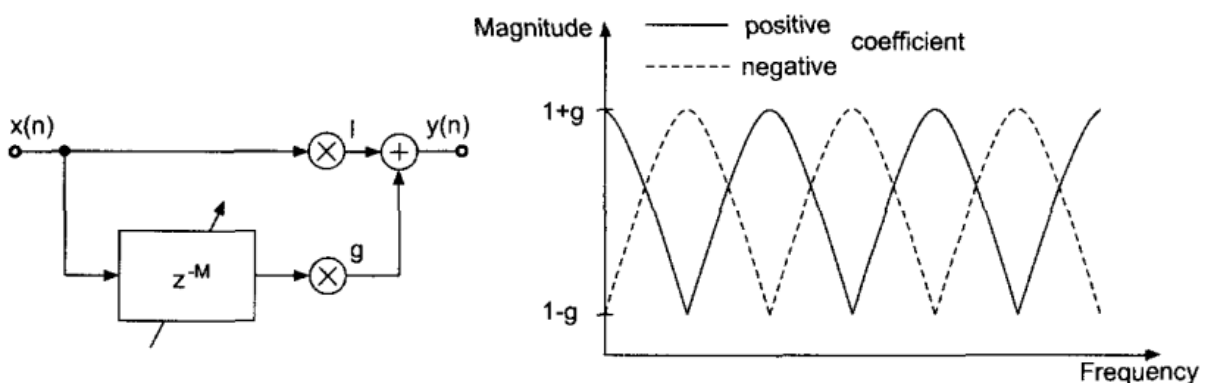


Fig.1. Filtro a pettine FIR e la risposta in ampiezza

La risposta nel tempo di questo filtro è costituito dal segnale diretto e dalla versione ritardata. Questo comportamento apparentemente semplice nel dominio del tempo descrive dei modelli interessanti nel dominio delle frequenze.

Per valori positivi di g , il filtro amplifica tutte le frequenze che sono multipli di $1/T$ e attenua tutte le frequenze intermedie.

La funzione di trasferimento di tale filtro mostra una serie di punte a forma di pettine (Fig.1). Ecco perché questo tipo di filtro è chiamato filtro a pettine. Per valori negativi di g , il filtro attenua le frequenze che sono multipli di $1/T$ e amplifica quelle che giacciono nel mezzo. Il guadagno varia da $1+g$ a $1-g$. Il seguente M-file (file Matlab) mostra l'implementazione di un filtro a pettine FIR campione per campione (Udo Zölzer 2002, 63-64).

M-file 1

```
x=zeros(100,1); x(1)=1; %impulso unitario di durata 100
g=0.5;
Delayline=zeros(10,1); %assegnazione della memoria per la
                        %lunghezza 10
for n=1:length(x);
y(n)=x(n)+g*Delayline(10);
Delayline=[x(n); Delayline(1:10-1)];
end;
```

Come i ritardi acustici, il filtro a pettine FIR ha un effetto sia nel tempo che in frequenza. Il nostro orecchio è più sensibile a un aspetto o all'altro secondo l'intervallo in cui è impostato il ritardo. Per valori maggiori di T , possiamo sentire un'eco distinto dal segnale diretto. Le frequenze amplificate dai pettini sono così vicini l'uno all'altro che a malapena identifichiamo l'effetto spettrale. Per valori più piccoli di T , il nostro orecchio non può più segregare gli eventi temporali ma può notare l'effetto spettrale del pettine (Julius O. Smith III 2010, *Comb Filters*).

Capitolo 3

Effetti audio basati sul ritardo ed l'implementazione in matlab

Vibrato

Quando passa un'auto, sentiamo una variazione di tono è dovuta al fatto che la distanza tra la sorgente e le nostre orecchie viene variato. Variare la distanza nel nostro caso, equivale a variare il ritardo temporale. La variazione continua e periodica del ritardo temporale produrrà una variazione periodica del tono. Questo descrive precisamente l'effetto vibrato.

Per realizzare tale effetto, abbiamo bisogno di una linea di ritardo e di un oscillatore a bassa frequenza (LFO sigla di Low Frequency Oscillator) per variare il parametro tempo del ritardo. I valori tipici del parametro tempo si trova nel range tra 5 a 10 ms e per la frequenza da 5 a 14 Hz.

L'effetto vibrato è udibile quando ascoltiamo il segnale ritardato .

In sintesi:

Vibrato - Variazione periodica del tempo di ritardo.
- Questo equivale a variare la distanza tra l'ascoltatore e la sorgente e produce un cambio di tono.

Implementazione - Una linea di ritardo e un oscillatore a bassa frequenza per variare il ritardo.
- Tempo di ritardo tipico = 5 – 10 Ms.
- Frequenza LFO 5 – 14Hz.
- Ascoltare solo il segnale processato.

L'M-file2 mostra l'implementazione del vibrato in Matlab (Udo Zölzer 2002, 68-69).

M-file 2

%Vibrato

```
function y=vibrato(y,SAMPLERATE,Modfreq,Width)
ya_alt=0;
Delay=Width; % ritardo del campione in ingresso in sec
DELAY=round(Delay*SAMPLERATE); % ritardo in campioni
WIDTH=round(Width*SAMPLERATE); % ampiezza di modulazione
                                % in campioni

if WIDTH>DELAY
    error('delay greater than basic delay !!!');
    return;
end
MODFREQ=Modfreq/SAMPLERATE; % frequenza di modulazione in
                                % campioni
LEN=length(x) ; % numero di campioni nel WAV-file
L=2+DELAY+WIDTH*2; % lunghezza complessiva del ritardo
Delayline=zeros(L,1); %allocazione della memoria per il ritardo
y=zeros(size(x)) ; % allocazione di memoria per il vettore di uscita
for n=1 : (LEN-1)
    M=MODFREQ;
    MOD=sin(M*2*pi*n) ;
    ZEIGER=1+DELAY+WIDTH*MOD;
    i=floor (ZEIGER) ;
    frac=ZEIGER-i;
    Delayline=[x(n) ;Delayline(1:L-1)] ;
    %---Interpolazione Lineare-----
    y(n,1)=Delayline(i+1)*frac+Delayline(i)*(1-frac);
    %---Interpolazione Allpass-----
    %y(n,1)=(Delayline(i+1)+(1-frac)*Delayline(i)-(1-frac)*ya_alt);
    %ya_alt=ya(n, 1) ;
    %---Interpolazione Spline-----
    %y(n,1)=Delayline(i+1)*frac-3/6
    % . . . +Delayline (i) * ( (1+frac) ^3-4*frac^3) /6
    % . . . +Delayline(i-1)*( (2-frac)^3-4*(1-frac)-3)/6
    % . . . +Delayline(i-2)*(1-frac)^3/6;
end
```

Il codice dovrebbe essere relativamente auto-esplicativo, ad eccezione di una parte:

- Calcoliamo il ritardo (modulato da una senoide) ad ogni passo, n:

```
M=MODFREQ;  
MOD=sin(M*2*pi*n);  
ZEIGER=1+DELAY+WIDTH*MOD;
```

- Quindi calcoliamo il campione del passo più vicino:

```
i=floor(ZEIGER);
```

- Il problema è che abbiamo un valore della linea di ritardo frazionario:

```
ZEIGER = 11.2779;  i=11  
ZEIGER = 11.9339;  i=11  
ZEIGER = 12.2829;  i=12
```

- I ritardi di lunghezza variabile del segnale di ingresso vengono utilizzati per simulare diversi effetti sonori. Pertanto, sono necessari ritardi del segnale di ingresso con valori non interi dell'intervallo di campionamento.
- Un ritardo del segnale di ingresso di M campioni più una frazione dell'intervallo di campionamento normalizzato con $0 \leq \text{frac} \leq 1$ è dato da:

$$y(n) = x(n - [M + \text{frac}]) \quad (1.3)$$

- Un algoritmo di interpolazione deve calcolare l'output $y(n)$, che si trova tra due campioni negli istanti temporali M e M+1. Nel nostro caso utilizziamo l'interpolazione lineare (Prof. Dave Marshall, Kirill Sidorov, Chapter 7).

– l'interpolazione lineare

```
y(n,1)=Delayline(i+1)*frac+Delayline(i)*(1-frac);
```

Oppure

$$y(n) = x(n - (M + 1)) * \text{frac} + x(n - M) * (1 - \text{frac}) \quad (1.4)$$

Esempio

L'algoritmo vibrato è stato sperimentato sul file audio seguente (cliccare per l'ascolto):

Keys Output Vibrato

Tramite il comando plot di Matlab si è tracciato una porzione del segnale diretto sovrapposto al segnale processato (Fig. 2). Si può notare il ritardo causato dall'algoritmo vibrato. I valori dei parametri di controllo sono riportati sul grafico.

Modfreq=9Hz Width=0.007s

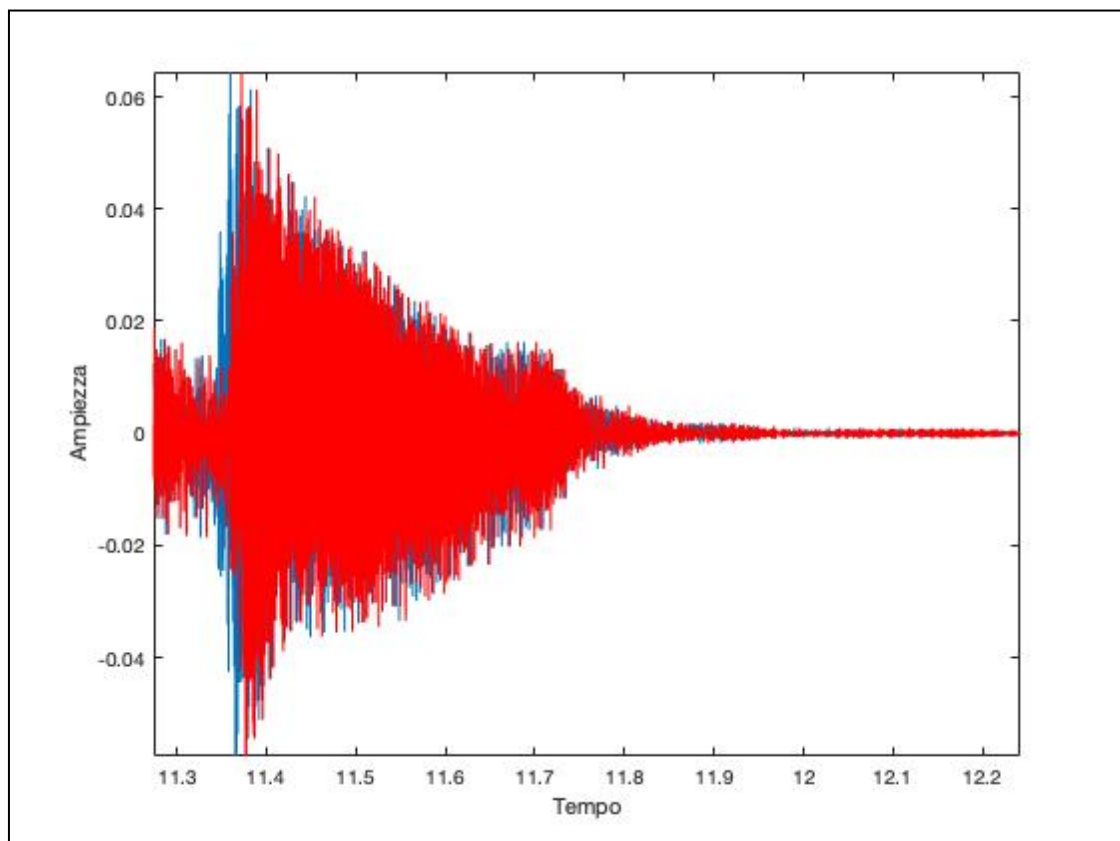


Fig. 2. Il plot blu è il segnale originale

Chorus

Il chorus consiste nell'impiego di una o più linee di ritardo (delay), attraverso le quali viene fatto passare il segnale audio, la somma del segnale diretto e di quello processato dai delay crea un suono che contiene piccole dissonanze (effetto doppler) come avviene in un coro di voci.

L'effetto chorus si ottiene variando il ritardo di pochi (solitamente dai 10 ai 25) millisecondi attraverso un LFO di cui si può modificare la frequenza e a volte anche la forma d'onda.

La variazione di ritardo genera di conseguenza anche una variazione di altezza del suono processato; più l'LFO influisce sul ritardo più evidente sarà l'effetto di dissonanza.

l'effetto chorus, è una combinazione dell'effetto vibrato con il segnale diretto.

In sintesi:

Chorus - Coppie multiple del segnale diretto con variazione periodica del tempo di ritardo.
- Combinazione dell'effetto vibrato con il segnale diretto.

Implementazione - Una o più linee di ritardo e un oscillatore a bassa frequenza per variare i ritardi.
- Tempi di ritardo tipico = 10 – 25 Ms.
- Frequenza LFO= Random.
- Ascoltare solo i segnali processato sommato al segnale originale.

L'M-file3 mostra l'implementazione del chorus in Matlab.

M-file 3

%Chorus

```
function y=chorus(x,SAMPLERATE,Modfreq,Width)
ya_alt=0;
amp=0.5;    %attenuazione sei segnale ritardato
Delay=Width; % ritardo del campione in ingresso in sec
DELAY=round(Delay*SAMPLERATE); % ritardo in campioni
WIDTH=round(Width*SAMPLERATE); % ampiezza di modulazione
                                % in campioni

if WIDTH>DELAY
    error('delay greater than basic delay !!!');
    return;
end
MODFREQ=Modfreq/SAMPLERATE; % frequenza di modulazione in
                                % campioni
LEN=length(x); % numero di campioni nel WAV-file
L=2+DELAY+WIDTH*2; % lunghezza complessiva del ritardo
Delayline=zeros(L,1); %allocazione della memoria per il ritardo
y=zeros(size(x)); % allocazione di memoria per il vettore di uscita
for n=1:(LEN-1)
    M=MODFREQ;
    MOD=sin(M*2*pi*n);
    ZEIGER=1+DELAY+WIDTH*MOD;
    i=floor(ZEIGER);
    frac=ZEIGER-i;
    Delayline=[x(n);Delayline(1:L-1)];
    %---Linear Interpolation-----
    y(n,1)=x(n)+amp*Delayline(i+1)*frac+amp*Delayline(i)*(1-frac);
    %---Allpass Interpolation-----
    %y(n,1)=(Delayline(i+1)+(1-frac)*Delayline(i)-(1-frac)*ya_alt);
    %ya_alt=ya(n,1);
    %---Spline Interpolation-----
    %y(n,1)=Delayline(i+1)*frac^3/6
    %....+Delayline(i)*((1+frac)^3-4*frac^3)/6
    %....+Delayline(i-1)*((2-frac)^3-4*(1-frac)^3)/6
    %....+Delayline(i-2)*(1-frac)^3/6;
end
```

Il codice dovrebbe essere relativamente auto-esplicativo, poiché simile all'algoritmo del vibrato ad eccezione della riga:

$$y(n,1)=x(n)+amp*Delayline(i+1)*frac+amp*Delayline(i)*(1-frac);$$

Si può notare l'aggiunta del segnale diretto e l'attenuazione del segnale ritardato.

Esempio

Keys Output Chorus

In questo caso, il segnale in uscita è composto dal segnale diretto e il segnale processato. Di conseguenza gli effetti dell'algoritmo non sono visibili da un semplice plot, ma sono visibili attraverso lo spettrogramma (Fig.4). I valori dei parametri controllo sono riportati in figura.

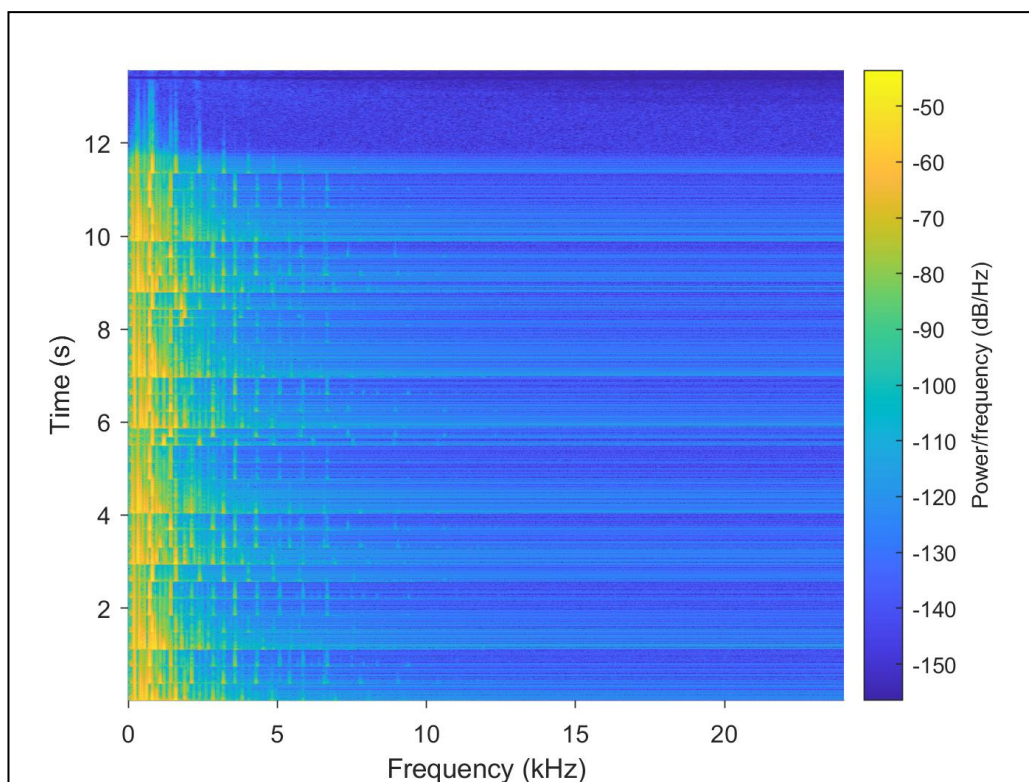


Fig. 3. Spettrogramma di Keys

Modfreq=1Hz Width=0.017s

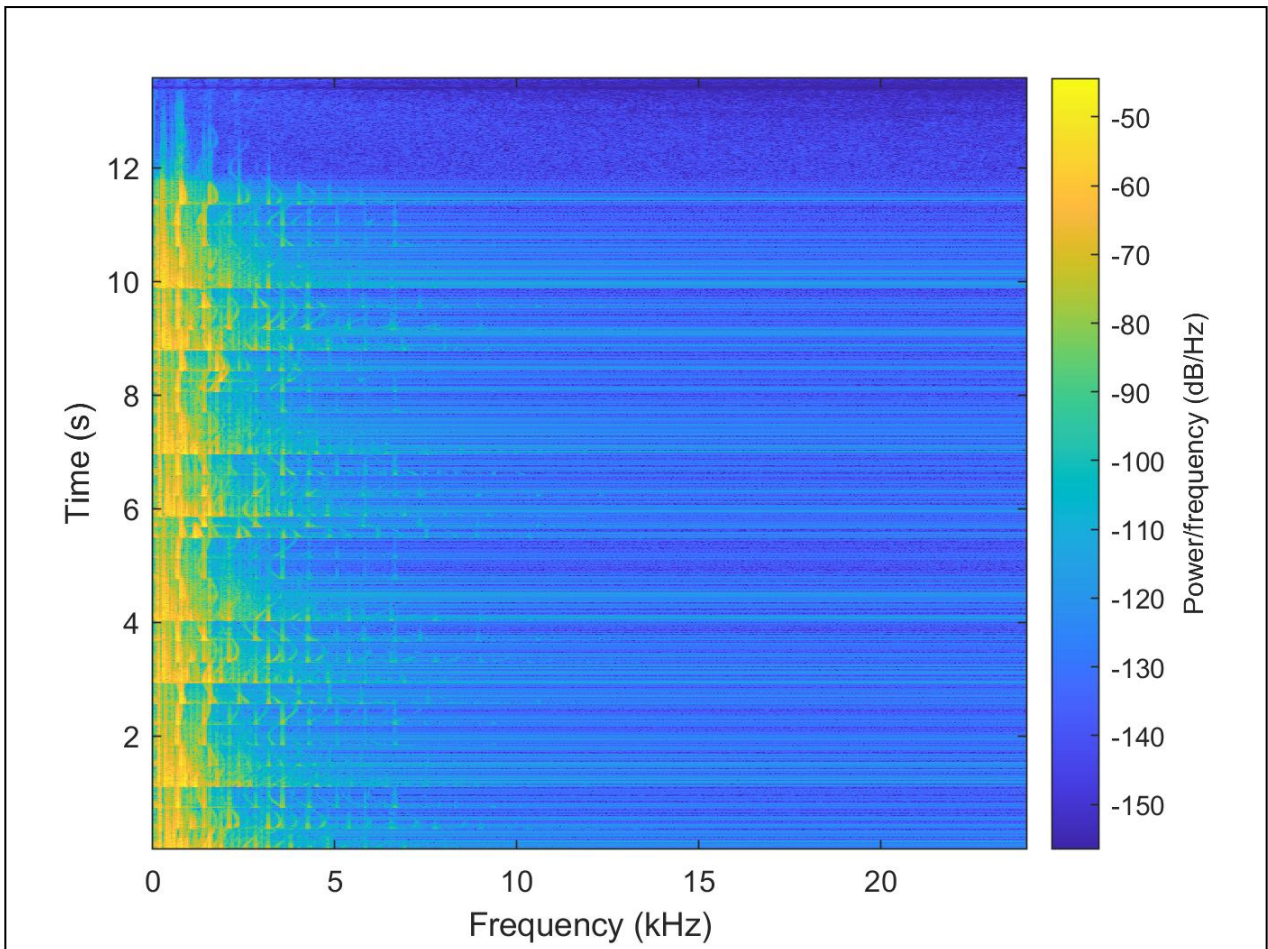


Fig. 4. Spettrogramma di Output Chorus

Flanger

Il flanger si basa sullo stesso funzionamento del chorus cioè aggiunge una linea di ritardo al segnale diretto. La differenza sostanziale è il tempo del ritardo applicato.

Nel flanger il ritardo applicato è così breve da creare degli effetti di controfase. il segnale generato viene poi reintrodotta nella linea di ritardo creando un effetto larsen progressivo e controllabile.

Se il ritardo è breve (meno di 15 ms) e se questo tempo di ritardo viene variato continuamente con una frequenza bassa come 1 Hz, si sentirà l'effetto flanging.

L'adozione del solito LFO applicato al tempo del ritardo crea il particolare suono ciclico. Il Flanger e' conosciuto per la tipica emulazione di un aereo a reazione.

In sintesi:

Flanger - Combinazione di una linea di ritardo breve con il segnale diretto(chorus con un ritardo molto breve).

Implementazione - Una linea di ritardo e un oscillatore a bassa frequenza per variare il ritardo.

- Tempo di ritardo tipico = 0 – 15 Ms.
- Frequenza LFO= circa 1Hz
- Ascoltare il segnale processato e il segnale diretto

L'implementazione del flanger in Matlab è identico a quello del chorus, la differenza è nei valori dei parametri di controllo usati nel richiamo della funzione.

Esempio

Keys Output Flanger

Anche in questo caso occorre uno spettrogramma del segnale processato (Fig.5) per avere un feedback visivo degli effetti dell'algorithm.

Modfreq=1Hz Width=0.017s

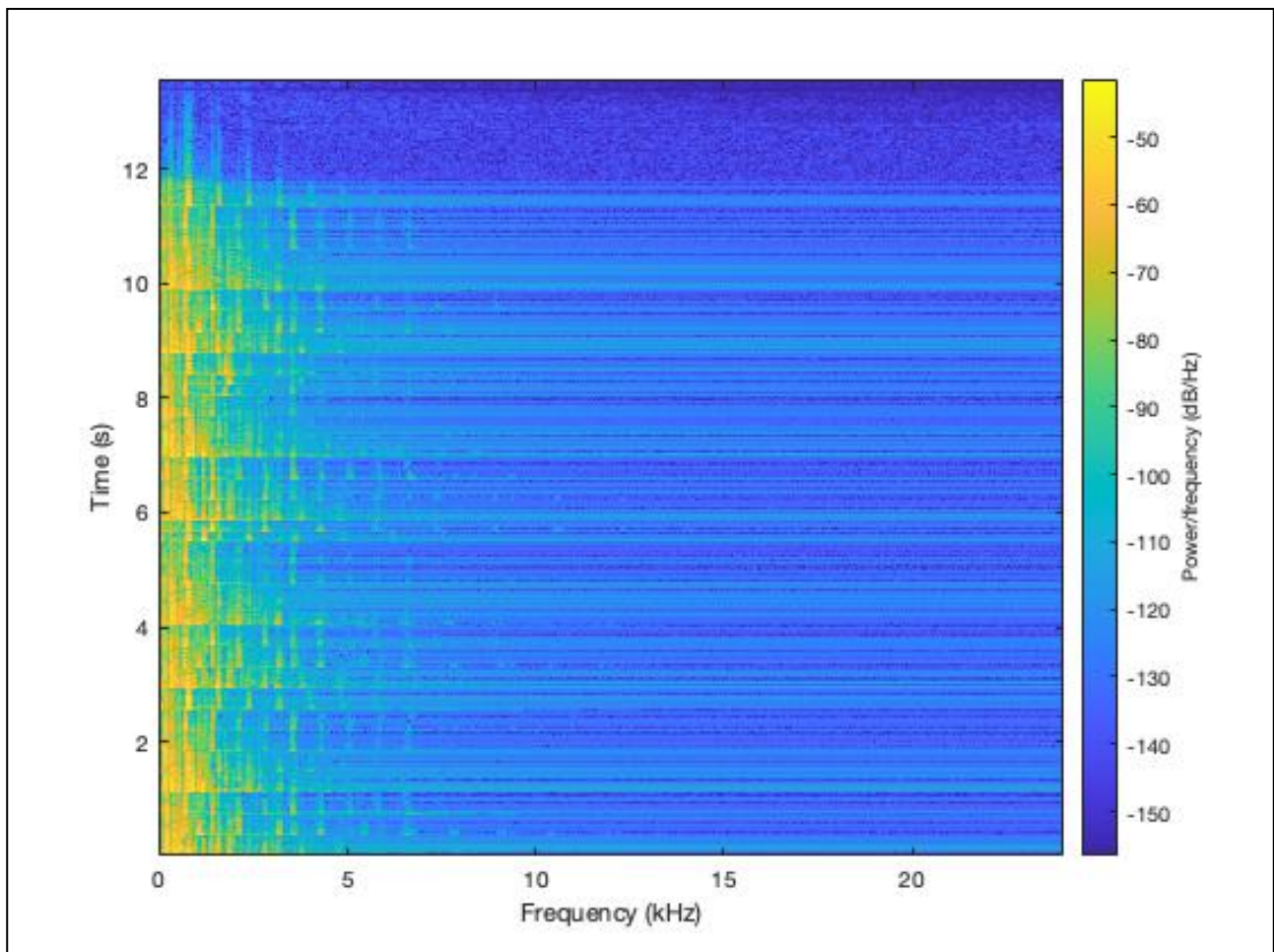


Fig. 5. Spettrogramma di Output Flanger

Capitolo 4

Implementazione in VCV Rack

La piattaforma

VCV Rack è un sintetizzatore modulare virtuale open source gratuito: più moduli possono essere collegati per sintetizzare un suono. Per impostazione predefinita, il software contiene diversi VCO, LFO, MIXER e altri moduli sintetizzatori standard, tuttavia è possibile aggiungerne altri come plug-in tramite il sito Web VCV Rack (VCV Rack, 2021).

Moduli utilizzati per l'implementazione



Fig. 6. Modulo AUDIO

Il modulo AUDIO (Fig. 6) fonde il mondo Rack virtuale con il mondo hardware fisico. La sezione TO DEVICE invia segnali Rack a un dispositivo audio hardware per la riproduzione e la sezione FROM DEVICE riceve segnali hardware in Rack.

Il driver MIDI del gamepad consente di utilizzare controller per videogiochi USB per sorgenti CV e GATE, come alternativa economica ai controller MIDI. I pulsanti del gamepad sono mappati su porte note MIDI che iniziano con C-1, C#-1, D-1, ecc. Gli assi del joystick sono mappati su messaggi MIDI CC che iniziano con CC0, CC1, CC2, ecc. con un'estensione MIDI non standard che consente il negativo Valori CC da utilizzare.

Il driver MIDI della tastiera del computer genera note MIDI quando vengono premuti i tasti mentre la finestra Rack è attiva.

Di seguito è riportato il layout per la tastiera QWERTY (USA).

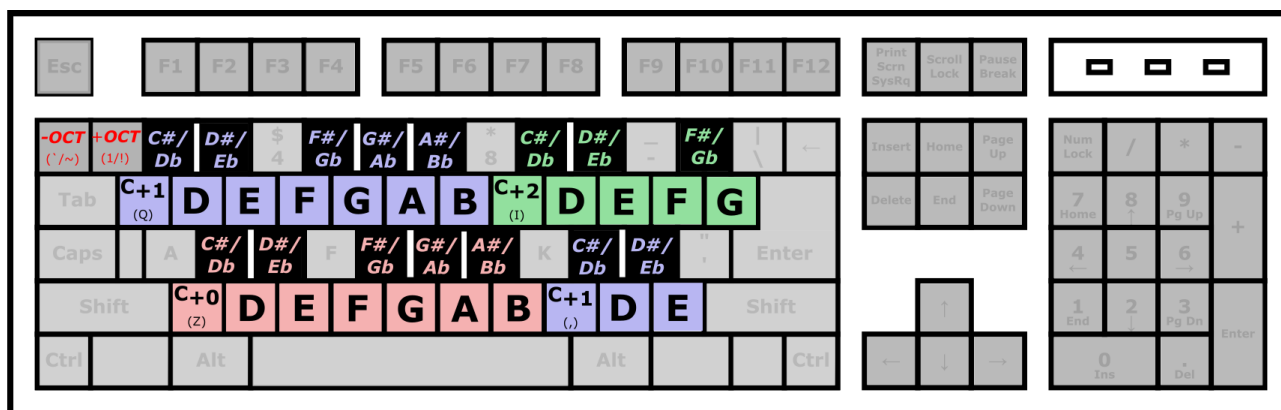


Fig. 7. Layout della tastiera QWERTY

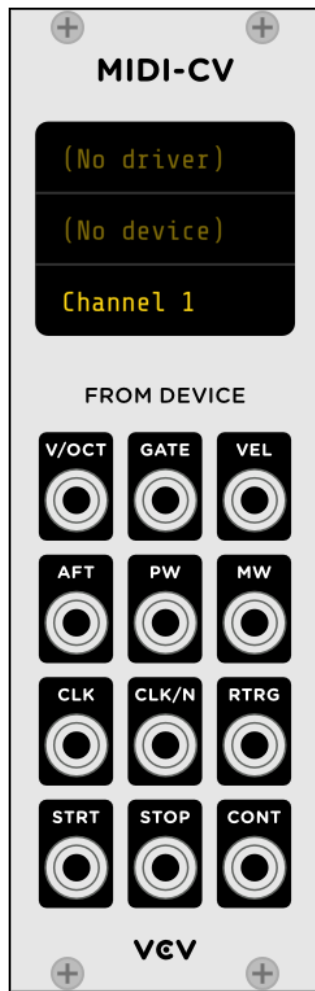


Fig. 8. Modulo MIDI

Permette la comunicazione tra il Rack virtuale ed un dispositivo MIDI (In questo caso tra il Rack e la tastiera QWERTY).

Definiamo i parametri utilizzati

- V/OCT - genera un segnale di pitch di 1V/ottava dell'ultima nota MIDI tenuta.
- GATE - genera 10V quando si tiene premuto un tasto. Non si riattiva quando le note vengono suonate in legato.
- RTRG - genera un trigger quando viene premuta una nuova nota, indipendentemente dal legato.

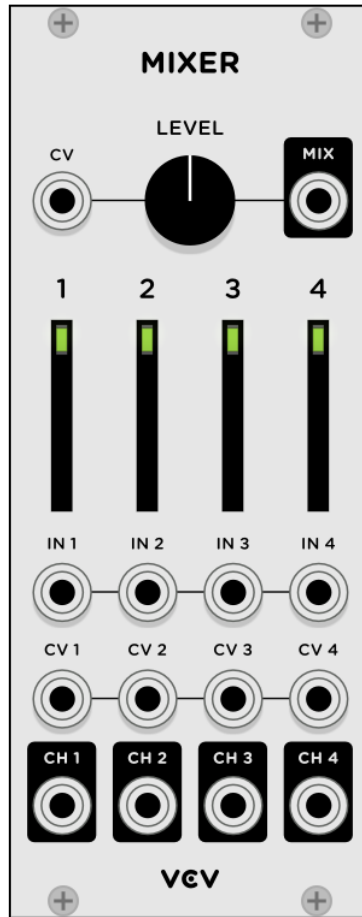


Fig. 9. Modulo MIXER

Mixa 4 segnali con VCA integrati, torna utile per simulare l'attenuazione dei segnali ritardati.

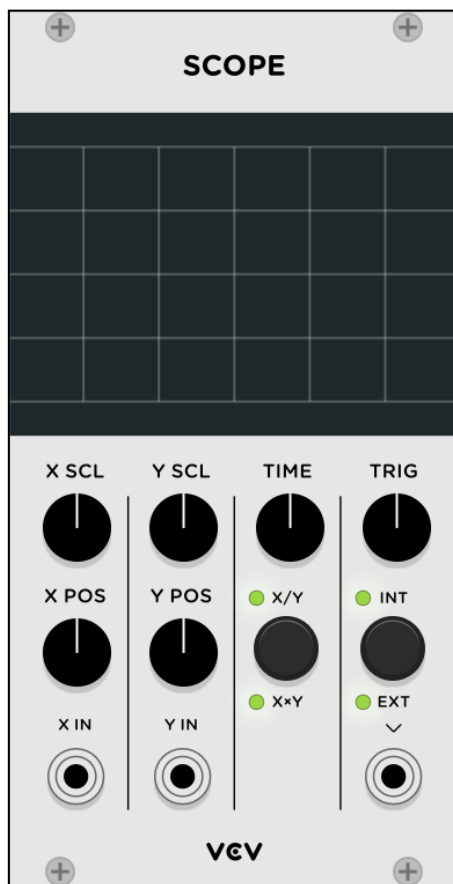


Fig. 10. Modulo SCOPE

Ispeziona le forme d'onda con un oscilloscopio.
Tramite il modulo si può avere un feedback visivo delle onde generate.

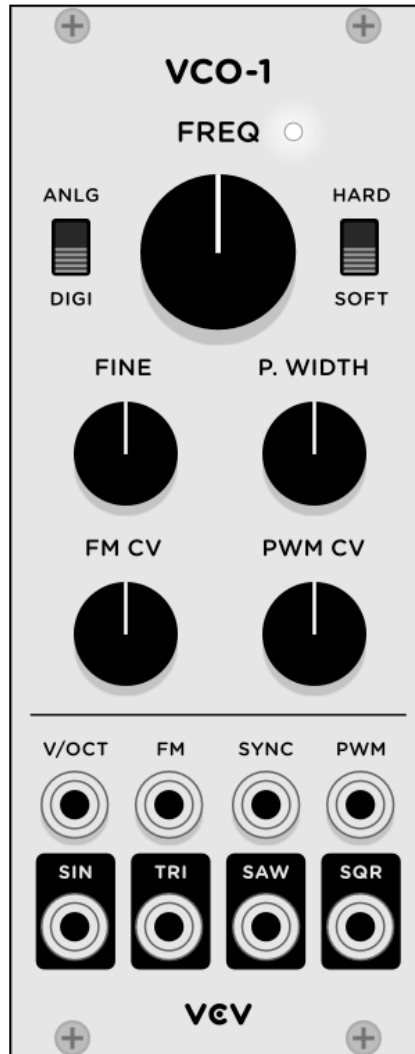


Fig. 11. Modulo VCO-1

Un oscillatore controllato in tensione genera toni grezzi a una particolare frequenza controllata da un segnale di tensione. VCO-1 è un oscillatore analogico/digitale tradizionale con quattro forme d'onda di uscita: SIN (sinusoidale), TRI (triangolo), SAW (a dente di sega) e SQR (quadrato). Il pitch di ogni oscillatore è controllato dalla manopola FREQ, con un range di 9 ottave, e dalla manopola FINE, con un range aggiuntivo di 2 semitoni.

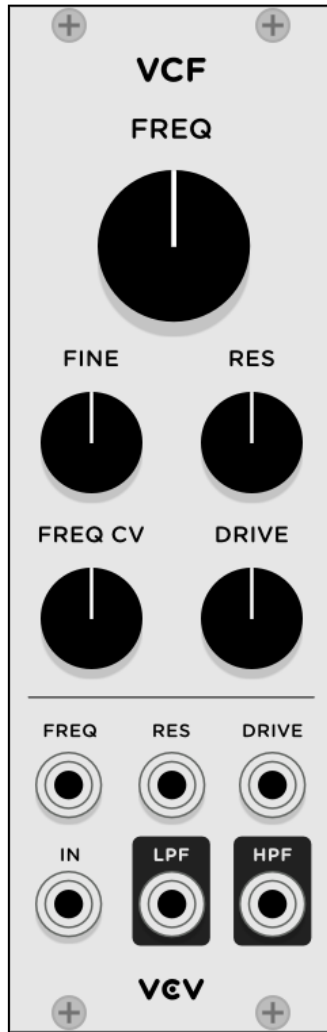


Fig. 12. Modulo VCF

Un filtro controllato in tensione rimuove un range di frequenze da un segnale audio, con la sua frequenza di taglio controllata da una tensione esterna. VCF modella un filtro ladder a transistor a 4 poli con saturazione in ogni fase.

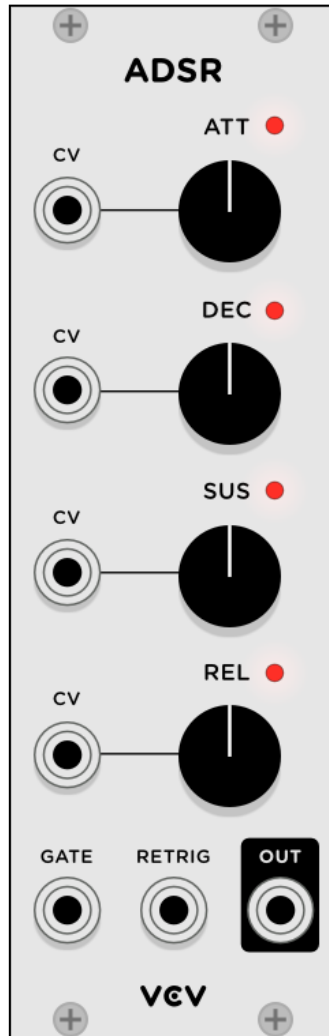


Fig. 13. Modulo ADSR

Genera un inviluppo con Attack/Decay/Sustain/Release, allo scopo di modellare la dinamica del segnale audio.

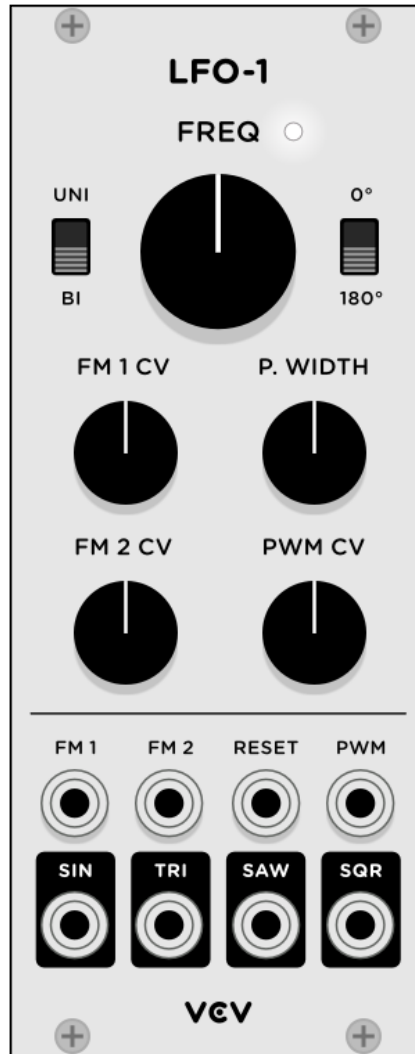


Fig. 14. Modulo LFO-1

Un oscillatore a bassa frequenza è un oscillatore controllato in tensione progettato per oscillare a frequenze inferiori rispetto alle frequenze audio.

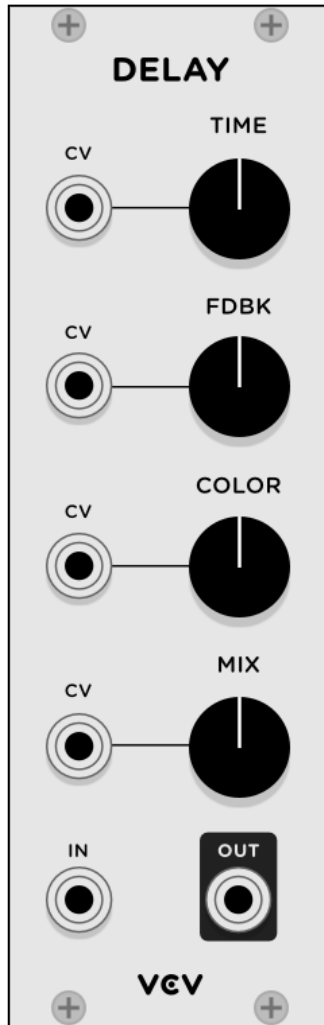


Fig. 15. Modulo DELAY

Riceve in input un segnale audio e genera una versione ritardata in output. Il tempo di ritardo è imposto dalla manopola TIME.

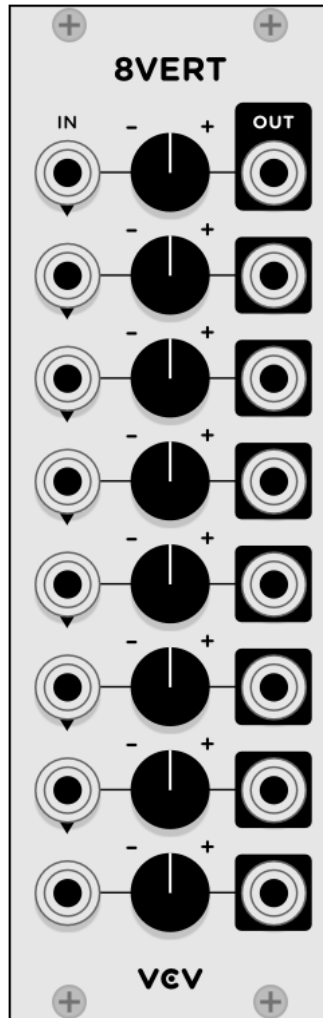


Fig. 16. Modulo 8VERT

Un “attenuverter” è un attenuatore che può raggiungere anche valori negativi. 8vert è un modulo di utilità con 8 ingressi, 8 attenuatori e 8 uscite. Ciascun attenuatore può applicare un guadagno che va da -1 a 1. Nella sua posizione iniziale a ore 12, viene applicato un guadagno di 0, risultando in un segnale di uscita silenzioso. Posizionato tutto a destra, il segnale di ingresso passa senza modifiche, e posizionato tutto a sinistra, l'ingresso è invertito, capovolgendo il suo valore . Servirà a ridurre l'escursione dei LFO.

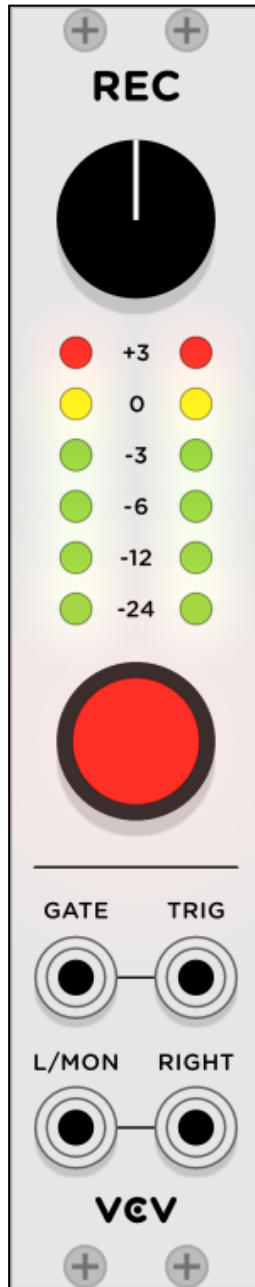


Fig. 17. Modulo RECORDER

Registra audio e video catturati dallo schermo di un file.

Vibrato

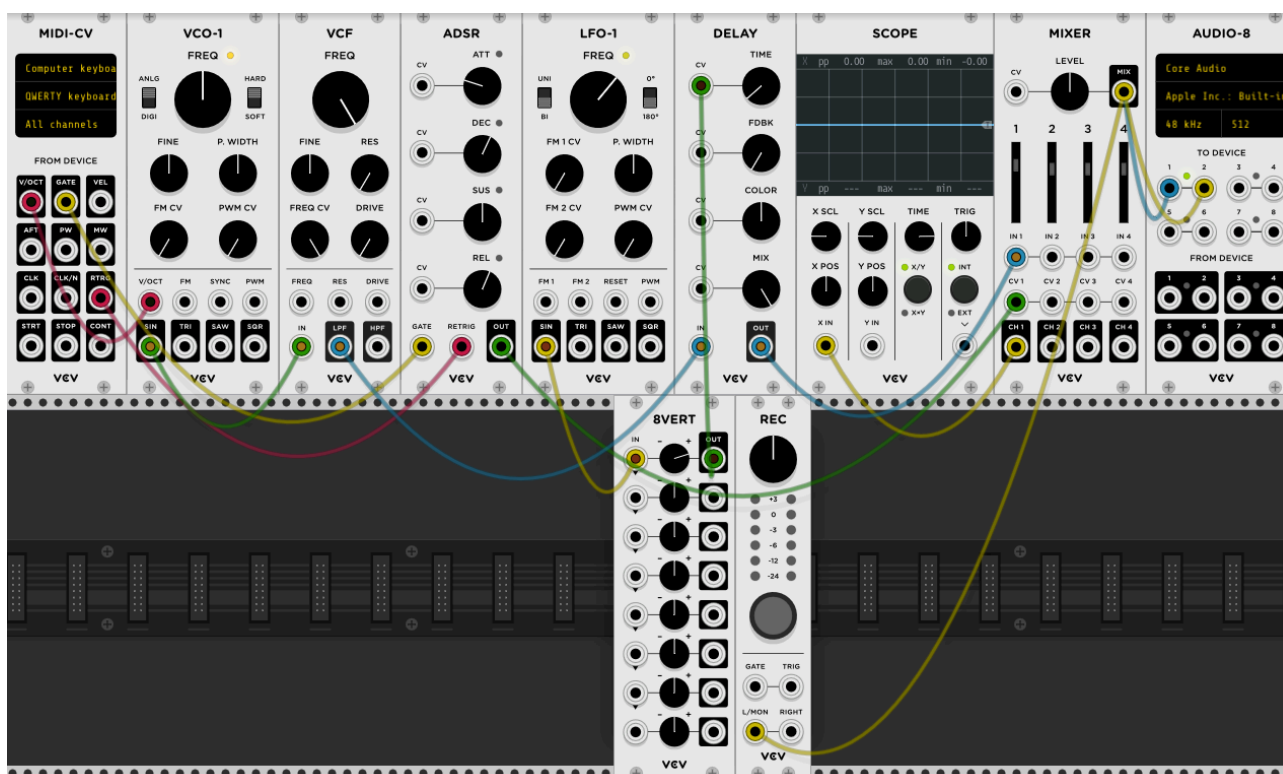


Fig. 18. Layout del Vibrato

Il primo passo è stato quello di generare un segnale audio tramite il modulo VCO-1 (ho optato per un segnale di carattere sinusoidale).

Successivamente, tale segnale è stato diretto verso l'input del modulo filtrante VCF per rimuovere alcune delle frequenze più alte.

Per poter utilizzare la tastiera QWERTY come controller midi, ho collegato la porta V/OCT del VCO-1 alla medesima porta di un modulo MIDI. Le porte GATE e RETRIG di quest'ultimo sono state collegate al loro reciproco del modulo ADSR per avere controllo sull'involuppo del segnale audio.

L'output LPF (Low Pass Filter) di VCF è stato poi diretto verso l'input del modulo DELAY, per applicare il ritardo necessario ad ottenere il vibrato.

Il passo successivo è stato quello di generare un segnale sinusoidale con la frequenza di circa 10Hz, per variare il tempo di ritardo attraverso l'output SIN del modulo LFO-1.

Come si può verificare anche dalla Fig. 18, questa porta non è stata collegata direttamente al CV (Control Voltage) della manopola TIME, ma è stata diretta ad un dei input del modulo

8VERT per un segnale con l'escursione ridotto. L'output è stato diretto verso il CV del modulo DELAY generando a tutti gli effetti il vibrato. Dirigendo l'output del modulo DELAY all'input di uno dei canali del MIXER e facendo un ulteriore routing verso due canali del modulo AUDIO, ha reso possibile l'ascolto del segnale processato in stereo.

Il modulo SCOPE è servito a visualizzare l'andamento del segnale durante l'implementazione.

Chorus

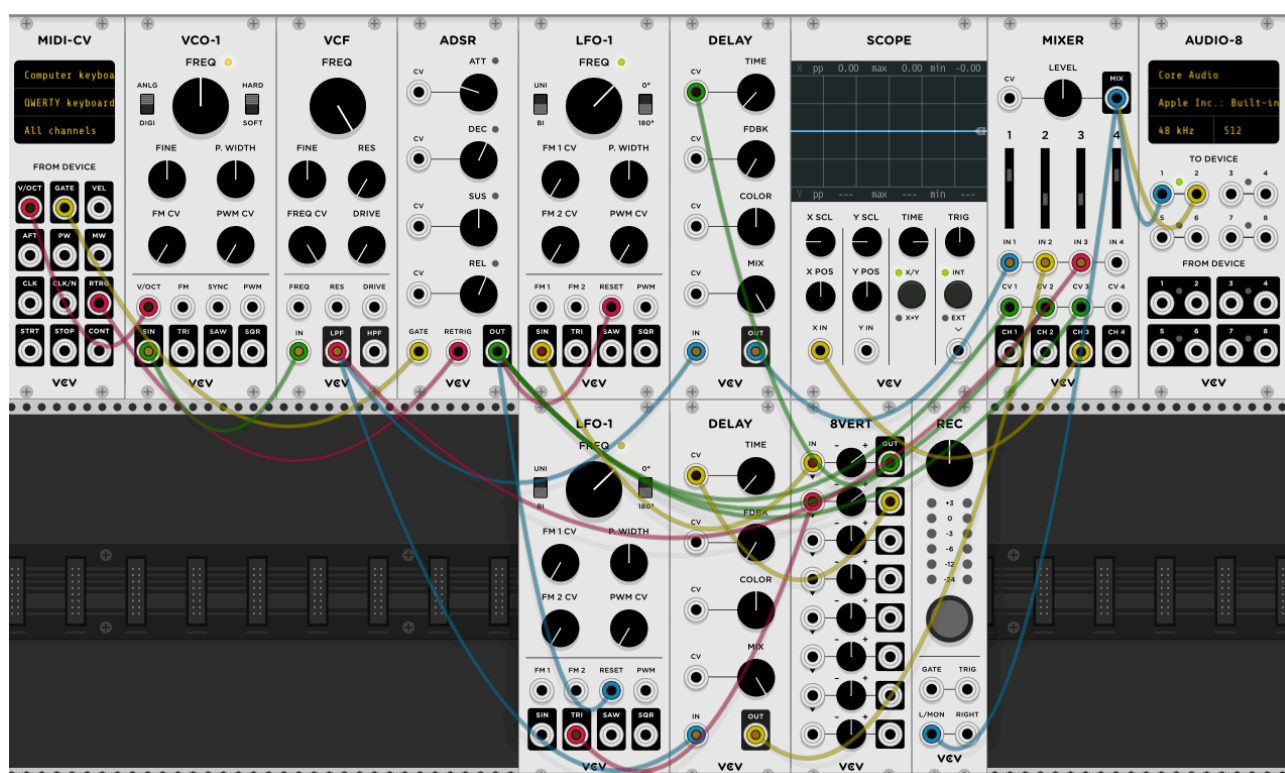


Fig. 19. Layout del Chorus

Per l'implementazione del chorus, ho aggiunto un modulo LFO-1 e un ulteriore modulo DELAY in quanto, per ottenere l'effetto occorre avere due segnali con la variazioni diverse.

Il primo LFO-1 è rimasto uguale al caso considerato precedente, mentre il secondo, genera un segnale triangolare.

Anche in questo caso non ci sono stati collegamenti diretti con i relativi moduli DELAY, ma il segnale, è stato mandato agli ingressi di due canali disponibili sul modulo 8VERT, riducendo così l'escursione.

A differenza del vibrato, al modulo MIXER sono stati mandati in canali separati, il segnale filtrato uscente dalla porta LPF del modulo VCF ed entrambi i segnali processati in uscita dai vari moduli DELAY.

I volumi dei canali MIXER processati sono stati ridotti a circa la metà del volume del canale non processato per ottenere un ascolto più musicale.

Flanger

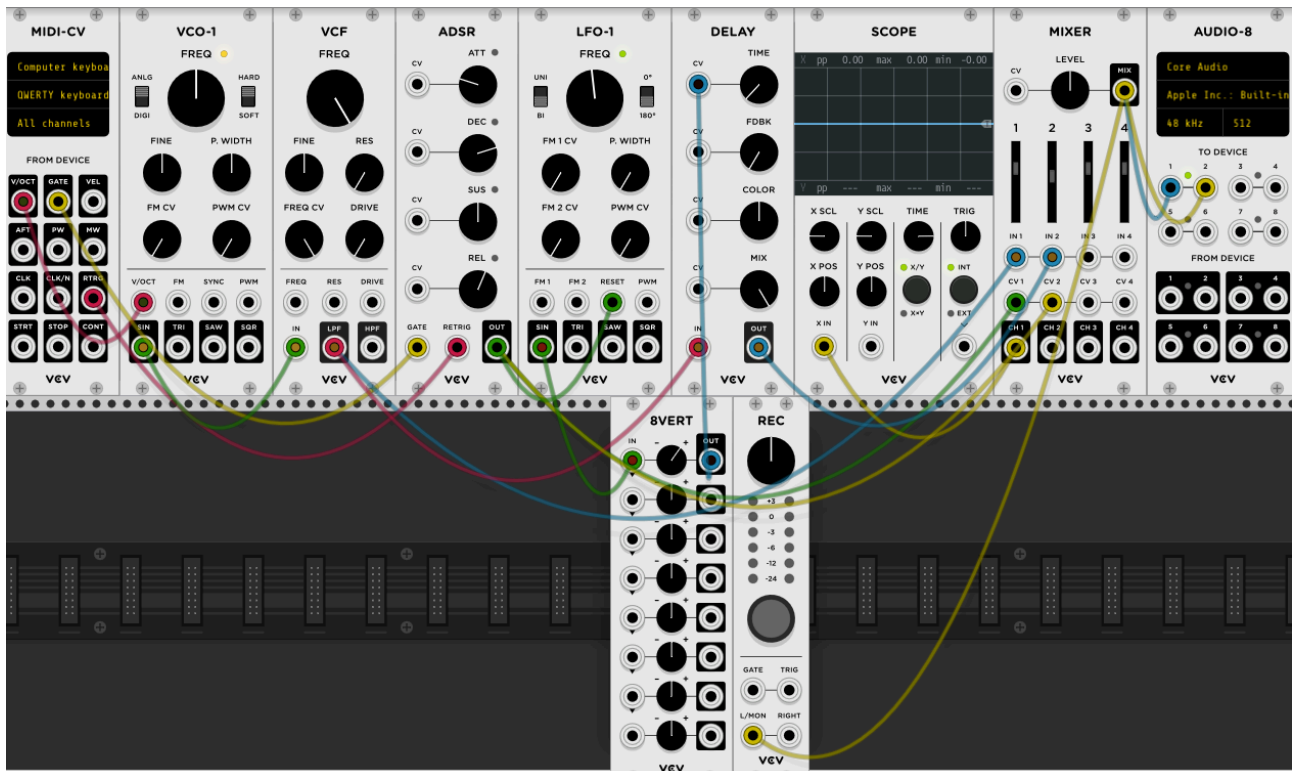


Fig. 20. Layout del Flanger

Il layout (Fig. 20) è simile al vibrato con un paio di differenze.

La prima è che il tempo di ritardo è stato ridotto, e il secondo è il routing nel modulo MIXER. Per emulare l'effetto flanger infatti occorre ascoltare sia il segnale originale che il segnale processato e attenuato.

Il passo successivo è stato quello di generare un file wav dell'output degli effetti implementati attraverso il modulo RECORDER, per avere un confronto con i file generati precedentemente da Matlab.

Output Vibrato VCVRack

Output Chorus VCVRack

Output Flanger VCVRack

Effettuando un'analisi spettrale (Fig. 20, 21, 22) attraverso il comando spectrogram di Matlab, è stato possibile visualizzare l'andamento nel tempo e in frequenza dei segnali processati.

Si può notare dai vari spettrogrammi le modulazioni nel tempo causate dai moduli LFO-1. Dalla Fig. 20, si può facilmente osservare che il vibrato, avendo una frequenza di modulazione più alta, presenta variazioni (oscillazioni) più rapide rispetto al flanger (Fig. 23) che ha una frequenza di variazione minore. Inoltre confrontando i file audio esportati da Matlab con quelli generati dal rack si può verificare la similitudine tra le due implementazioni.

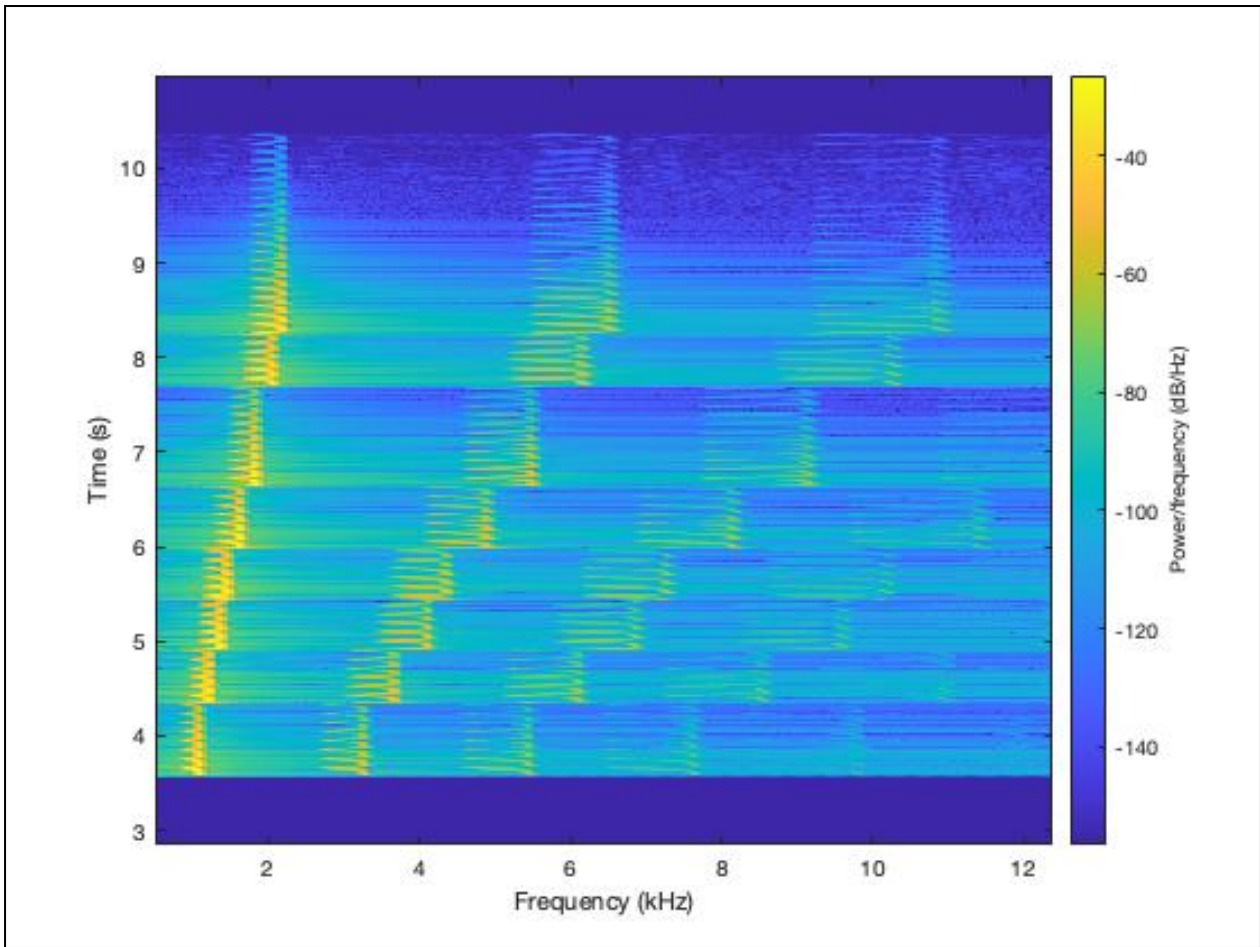


Fig. 21. Spettrogramma di Output VCV Vibrato

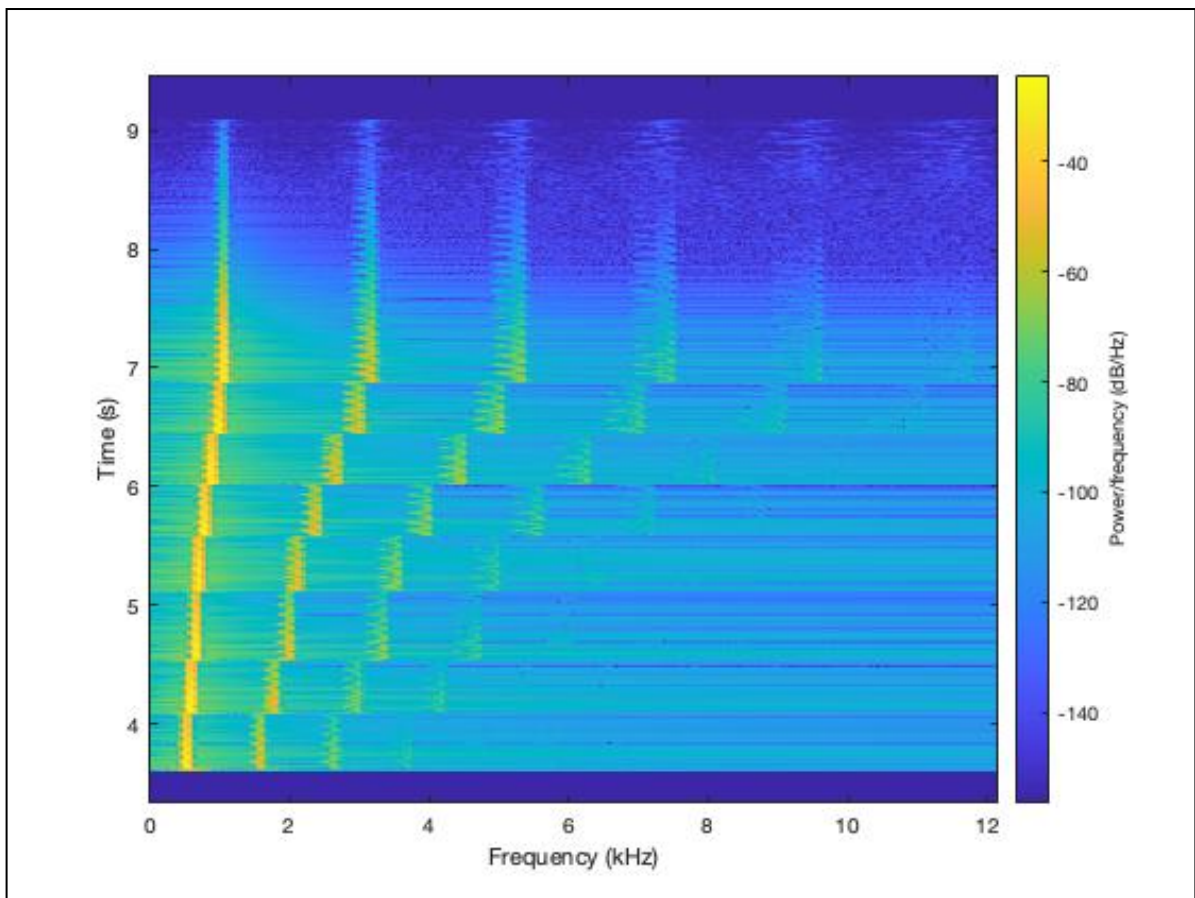


Fig. 22. Spettrogramma di Output VCV Chorus

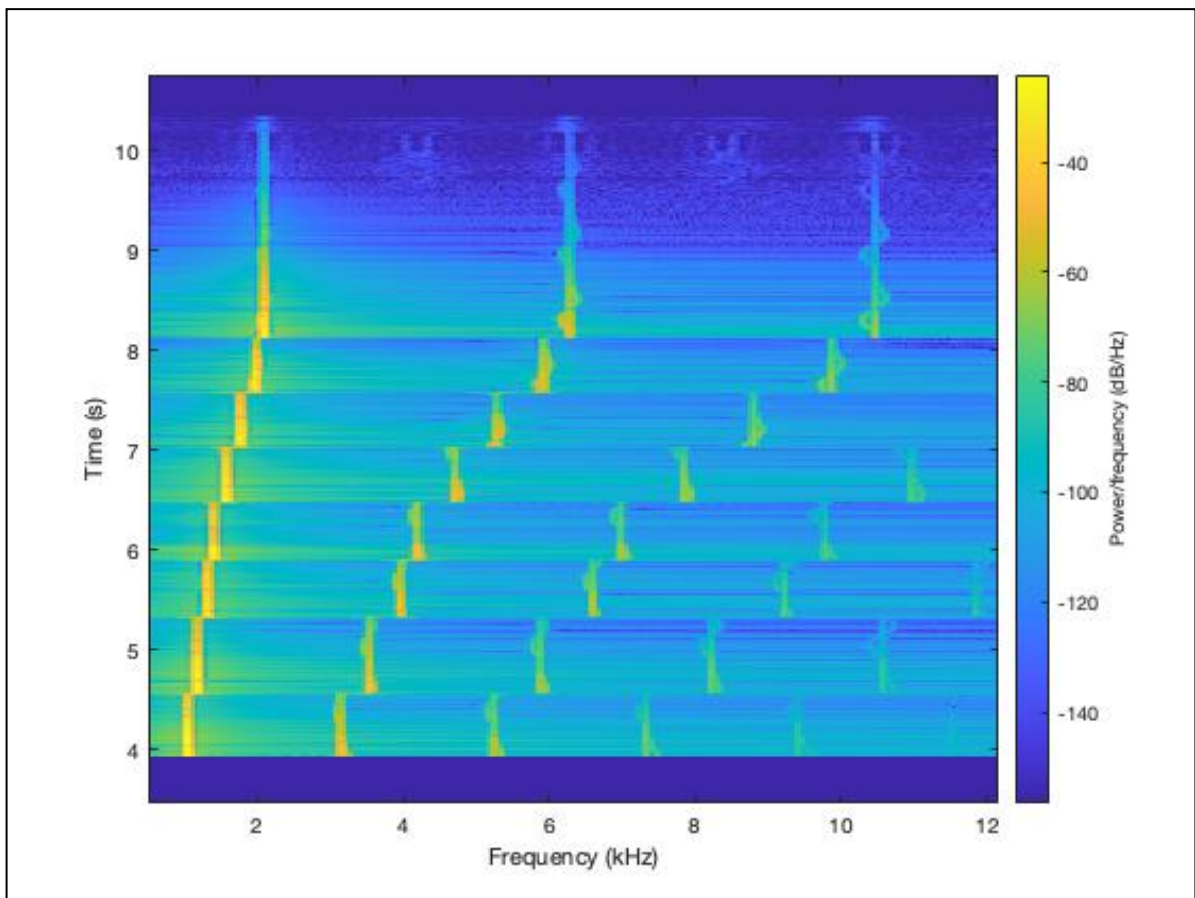


Fig. 23. Spettrogramma di Output VCV Flanger

Conclusioni

In questo lavoro di tesi è stata fornita una panoramica completa riguardo ai ritardi acustici, in particolare gli effetti musicali che ne derivano con relativi approcci d'implementazione nei software Matlab e VCV Rack.

Sono state spiegate le operazioni necessarie per lo sviluppo e il funzionamento degli effetti vibrato, chorus e flanger eseguendo un'analisi del codice Matlab, fornendo esempi uditivi e un'analisi nel tempo o in frequenza all'occorrenza, dei segnali audio processati come mostrato dal Prof. Stefano Squartini durante il corso di CADSP (*Slides del corso CADSP, 2020/2021*).

Questo lavoro di analisi, può essere importante nella comprensione dei ritardi acustici ricreati attraverso i filtri FIR e gli effetti che ne derivano.

Inoltre, gli esempi audio forniti, dimostrano l'importanza degli effetti acustici digitali come il vibrato, chorus e flanger nel mondo della musica. Oltre all'evidente colorazione dei suoni, riescono a dare una dimensione diversa da quello in cui il suono è situato al momento della riproduzione.

La tesi dimostra anche l'importanza delle implementazioni digitali di questi effetti. In particolare, dimostra i vantaggi del digitale rispetto all'analogico. Le tecniche digitali infatti, non comportano nessun usura dei componenti e consentono una riproduzione fedele degli effetti.

Bibliografia

- [1] Udo Zölzer. *DAFX: Digital Audio Effects: DELAYS*. John Wiley, 2002.
- [2] Prof. Dave Marshall and Dr. Kirill Sidorov. *Digital Audio Effects*, Cardiff School of Computer Science.
- [3] Julius O. Smith III. *Physical Audio Signal Processing For Virtual Musical Instruments And Audio Effects* W3K Publishing, 2010, Centre for Computer Research in Music and Acoustics (CCRMA), Stanford University.
- [4] VCV Rack. “VCV Library & Manual.” Consultato il 12 Agosto, 2021.
<https://library.vcvrack.com/Fundamental>
<https://vcvrack.com/manual/Core>
- [5] Prof. Stefano Squartini, *Slides del corso CADSP di laurea in Ingegneria Elettronica*. 2020/2021, Università Politecnica delle Marche (UNIVPM).
- [6] Wikipedia. “Wiki Flanger, Wiki Chorus_(effetto), Wiki Vibrato, Wiki Delay.” Consultato il 7 Maggio, 2021.
<https://it.wikipedia.org/wiki/Flanger>
[https://it.wikipedia.org/wiki/Chorus_\(effetto\)](https://it.wikipedia.org/wiki/Chorus_(effetto))
<https://it.wikipedia.org/wiki/Vibrato>
<https://it.wikipedia.org/wiki/Delay>