Politecnico di Milano
School of Industrial and Information Engineering

Computer Music – Languages and Systems

Homework #1

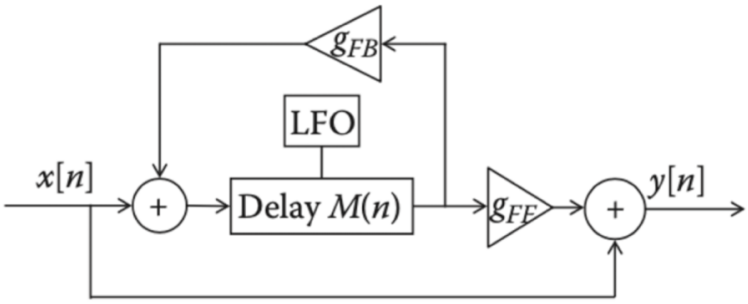
Multi effect: Flanger – Wah Wah – Phaser

Group composition:

Brusca Alfredo	10936149
Marazzi Alice	10625416
Pomarico Riccardo	10661306

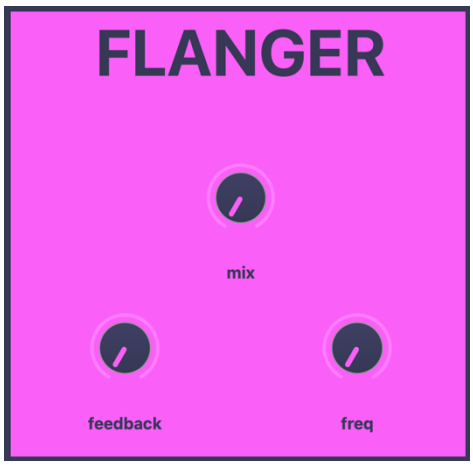
The goal of this project is to implement a multi effect able to manage Flanger, Wah Wah and Phaser effects and an interface to control it. We used a SynthDef for each effect, let's analyze them one by one.

Flanger

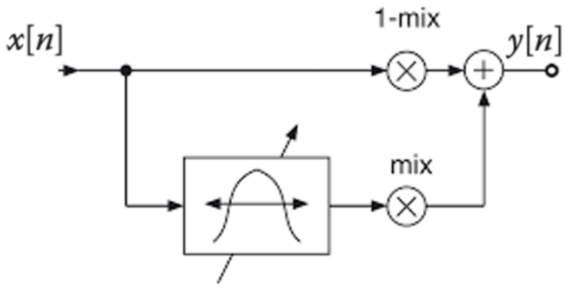


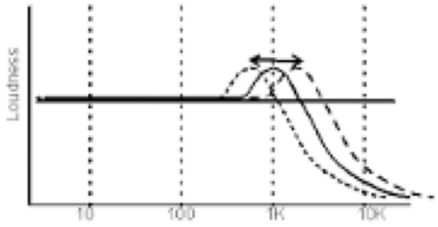
The user has the possibility to control some parameters of this effect, such as its frequency and its feedback gain. What the Flanger does is create an oscillator with a frequency equal to *mfreqF* which modulates a delay. The resulting signal is fed back through a loop and summed to the input signal. We then output this signal by mixing it with the base signal.

```
(
SynthDef("flange", {arg inBus, outBus, feedback = 0.1, mfreqF = 0.1, mixF
= 0;
      sigF=In.ar(inBus, 2);
      outF = sigF + LocalIn.ar(2);
      outF= DelayN.ar(outF,0.02,SinOsc.kr(mfreqF,0,0.005,0.005));
      LocalOut.ar(feedback*outF);
      outF = Mix.ar([outF*mixF, sigF*(1-mixF)]);
      Out.ar(outBus, Pan2.ar(outF));
}).send(s);
);
```



Wah Wah

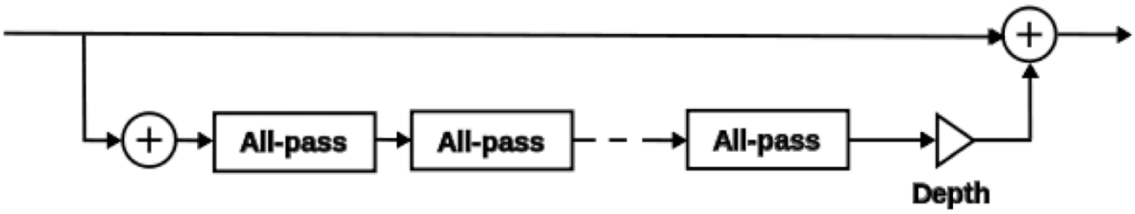




In this case, the user can modify the frequency *freqW* of the oscillator that modulates the central frequency of the bandpass filter that is applied to the input signal. The user can also modify the bandwidth of the filter. We output the processed signal mixing it with the original signal.

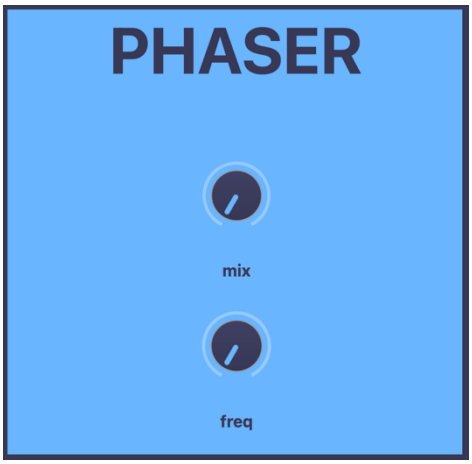
```
(
SynthDef("wah",{
      arg mixW=0, freqW=0.5, bwW=0, inBus, outBus;
      sigW = In.ar(inBus, 2);
      freqBand = SinOsc.kr(freqW).range(0,1);
      outW = BBandPass.ar(sigW, 2100*freqBand+400, 3*bwW+0.05, mixW);
      outW =Mix.ar([outW*mixW, sigW*(1-mixW)]);
      Out.ar(outBus, Pan2.ar(outW));
}).send(s);
);
```

Phaser



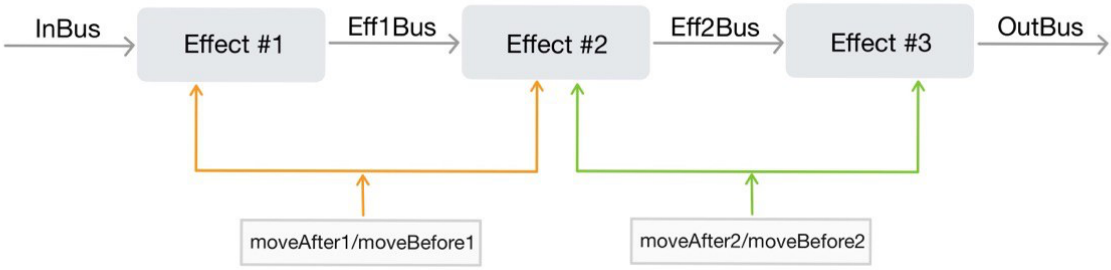
As far as this effect is concerned an allpass filter is applied to the signal, modulated by a sinusoid with user-controlled frequency *mfreqP*. The output of the Phaser effect is the modified waveform after consecutive filters.

```
(
SynthDef("phaser", {arg inBus, outBus, mfreqP = 0.2, mixP = 0;
      sigP = In.ar(inBus, 2);
      outP= BAllPass.ar(sigP,0.02,SinOsc.kr(mfreqP,0,0.01,0.01));
      outP = Mix.ar([outP*mixP, sigP*(1-mixP)]);
      Out.ar(outBus, Pan2.ar(outP));
}).send(s);
);
```



The management of the different buses is as follows:

- when the input signal is received, it is inserted into the *input bus*;
- secondly, we apply the first effect to the signal present on the input bus and we insert it into the *first effects bus*;
- then we apply the second effect and insert it into the *second effects bus*;
- finally, we apply the third and last effect taking the signal from the second effects bus and sending it to the *output bus*.



The GUI creates a rectangular window, and for each parameter that can be changed for the various effects there is a dedicated knob. There are also three *mix* knobs that change the amount of dry and wet signal that goes to the output buffer.
The chain of the effects can be changed by the user through "*Move Before*" and "*Move After*" buttons. These buttons change the layout of the GUI, the group order of the synths and the bus routing.
The effects can also be bypassed by selecting the "*Bypass"* buttons below each effect.
On the right side of the GUI there is a slider that controls the volume and a mute button.
Lastly, on the bottom of the window there is a signal visualization scope where the user can see both the input and the output signals.