Computer Music – Languages and Systems

Homework #1
Multi Effect: Flanger – Wah Wah - Phaser

Group 4: DRUM TEAM
Brusca Alfredo 10936149
Marazzi Alice 10625416
Pomarico Riccardo 10661306

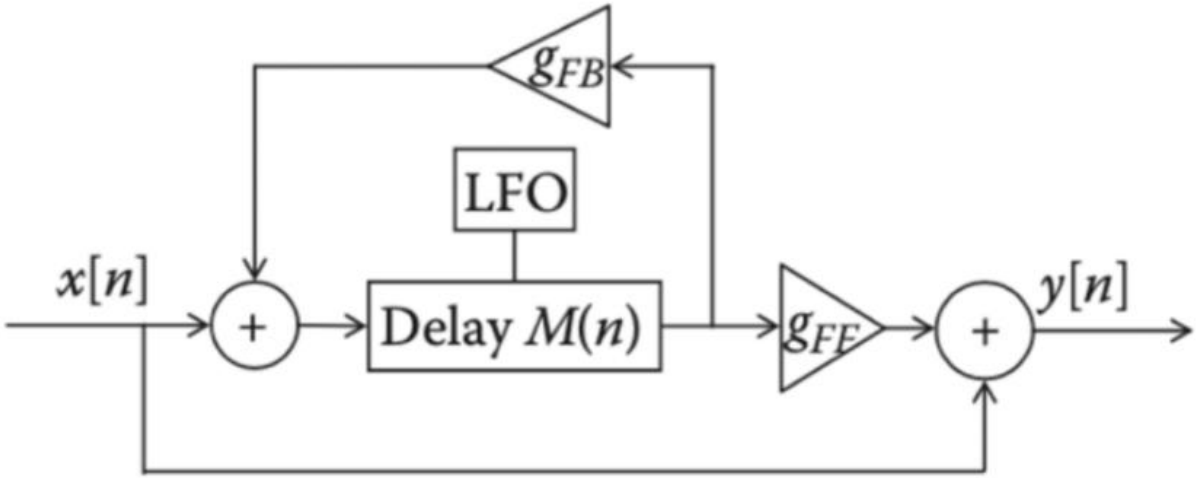
Project Goal

The goal of this project is to create a multi effect that can apply Flanger, Wah Wah and Phaser effects to an input signal. The effects have parameters that can be modified through an interface.

Flanger

The Flanger effect creates an oscillator with frequency *mFreqF* (set by the user) and applies a delay to it.

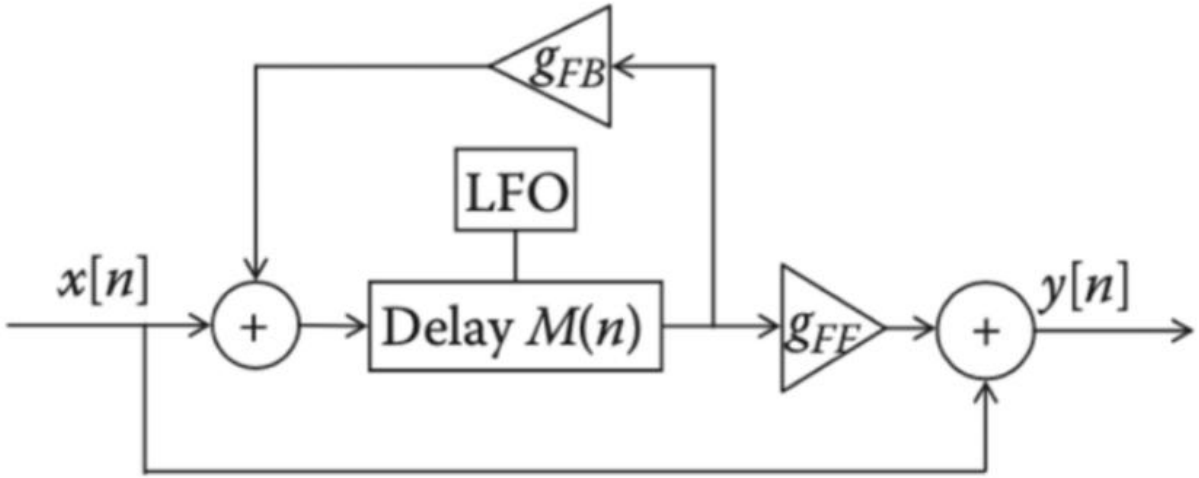
The user can control the frequency and the feedback of the flanger.



Flanger

The Flanger effect creates an oscillator with frequency *mFreqF* (set by the user) and applies a delay to it.

The user can control the frequency and the feedback of the flanger.

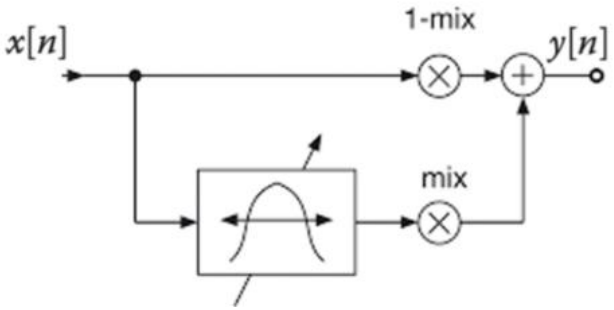


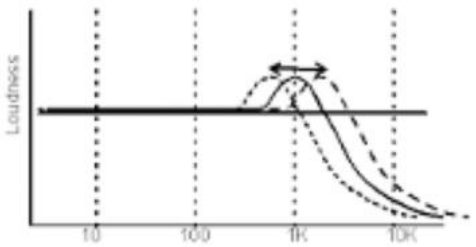

```
(
SynthDef("flange", {
    arg inBus, outBus, feedback = 0.1, mfreqF = 0.1, mixF = 0;
    sigF=In.ar(inBus, 2);
    outF = sigF + LocalIn.ar(2);
    outF= DelayN.ar(outF,0.02,SinOsc.kr(mfreqF,0,0.005,0.005));
    LocalOut.ar(feedback*outF);
    outF = Mix.ar([outF*mixF, sigF*(1-mixF)]);
    Out.ar(outBus,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

Wah Wah

The Wah Wah effect creates a oscillator that operates as a passband filter, whose central frequency varies over time in a prescribed range and according to a modulating function.

The user can control the central frequency of the oscillator and its bandwidth.



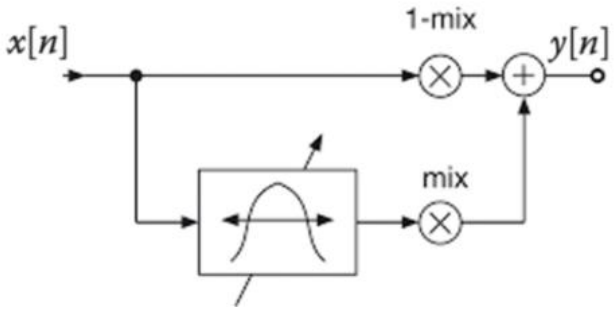


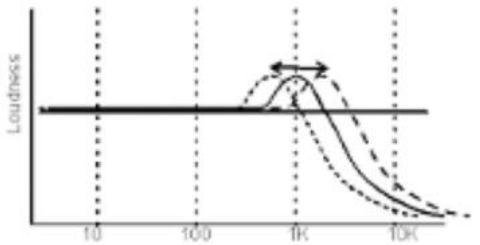
Wah Wah

The Wah Wah effect creates a oscillator that operates as a passband filter, whose central frequency varies over time in a prescribed range and according to a modulating function.

The user can control the central frequency of the oscillator and its bandwidth.

```
(
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);
);
```

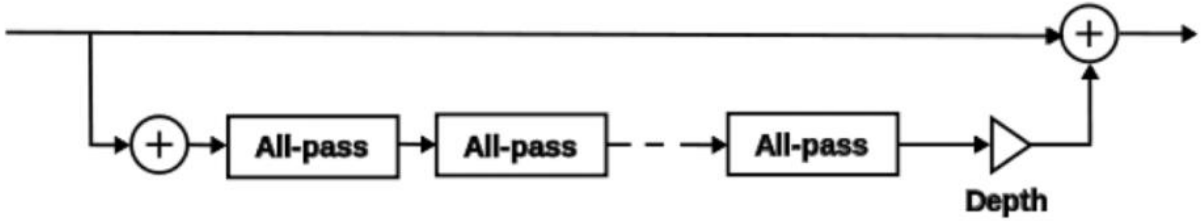


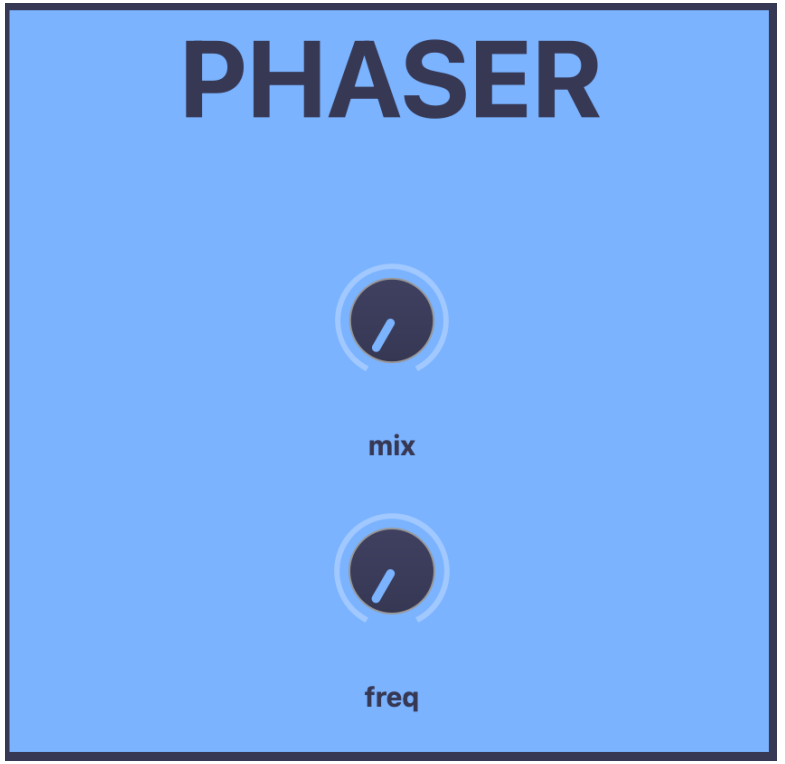


Phaser

The Phaser effect applies an all-pass filter using a delayed signal created by a pulse train. This filter is mixed with the input signal, so the output of the effect is a modified waveform after consecutive filters.

The only parameter that the user can control is the frequency of the modulation signal.



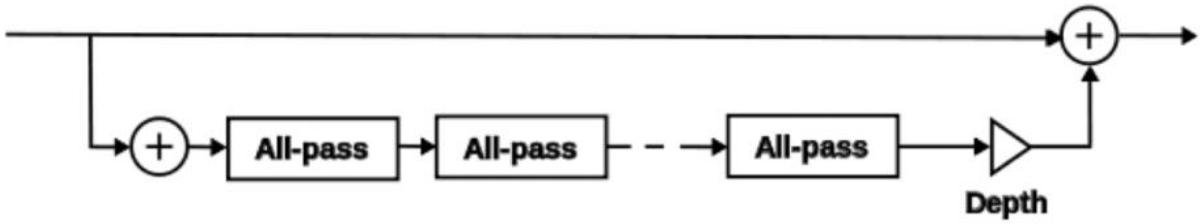


Phaser

The Phaser effect applies an all-pass filter using a delayed signal created by a pulse train. This filter is mixed with the input signal, so the output of the effect is a modified waveform after consecutive filters.

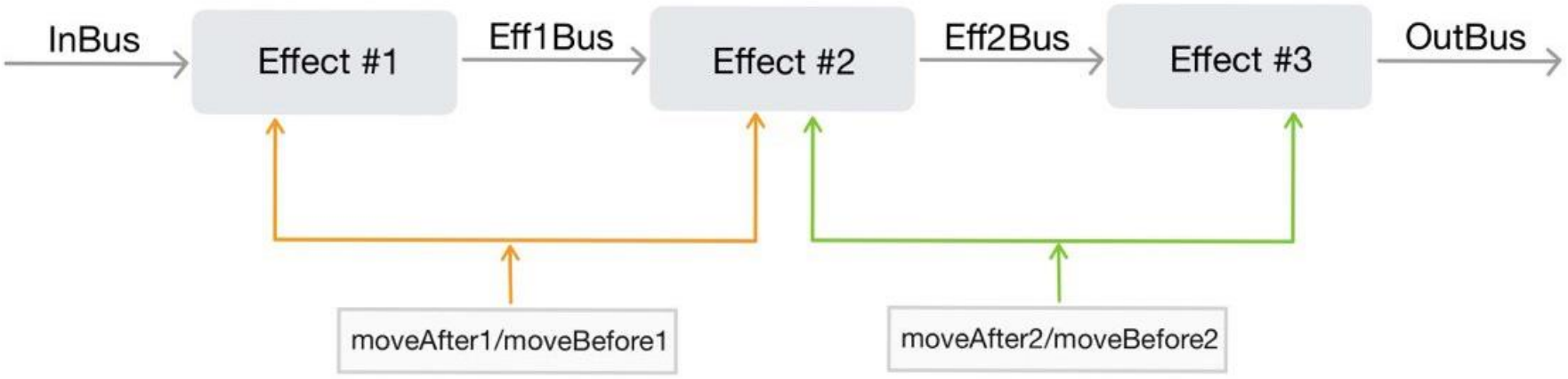
The only parameter that the user can control is the frequency of the modulation signal.

```
(
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);
);
```



Bus Routing

The effects are routed through serial bus connection.
The application order of the effects can be controlled by the *moveBefore* and *moveAfter* buttons that will swap the position of the effects in the GUI and their relative input and output busses.



GUI

The GUI shows each effect in the order in which they are applied to the input signal. Each effect can be bypassed by selecting the relative BYPASS button.
At the bottom of the GUI the waveforms of the input and the output signals are shown as the effects are applied to the signal.

Each effect has one or more knobs that can be used to control its parameters. In addition to those, there is a mix knob for each effect that can modify the mixing value of the input and the effect-modified signal.
On the right side of the GUI there is a volume slider and a mute button that mutes or unmutes the signals.