Politecnico di Milano
School of Industrial and Information Engineering

Computer Music – Languages and Systems

Homework #1

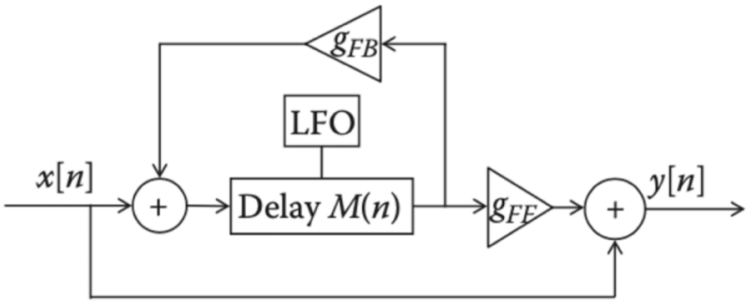
Multi effect: Flanger – Wah Wah – Phaser

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.

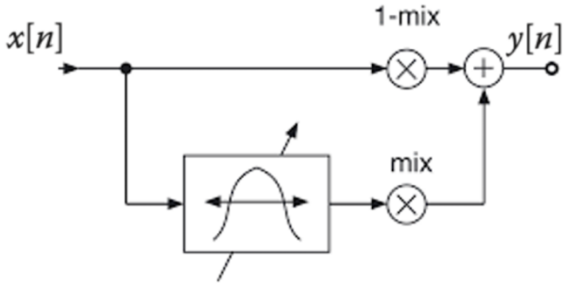
Flanger

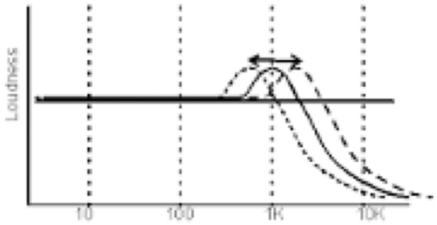


The user has the possibility to control some parameters of this effect, such as its frequency, its intensity (through the parameter called *depth*) and its delay. What the Flanger does is to create an oscillator with a frequency equal to *mfreq* and, once the input signal is received, we apply a delay according to these parameters. We then output this signal by mixing it with the base signal.

```
(
SynthDef("flange", {arg inBus, outBus;
      var x, sig,out, mod1, depth, mfreq,maxfreq, maxdepth, maxdelay,
basedelay;
      maxfreq=10;
      maxdelay=0.5;
      basedelay=maxdelay/2;
      maxdepth=1;
      depth = maxdepth/2;
      mfreq = maxfreq/2;
      mod1 = SinOsc.ar(mfreq,0, depth);
      sig =  In.ar(inBus, 1);
      out = DelayC.ar(sig, maxdelay,basedelay + mod1);
      out = Mix.ar([out,sig]);
      if (~flangerActive == 1, {
            Out.ar (outBus, out);
      }, {
            Out.ar (outBus, sig);
      });
}).send(s);
)
```

Wah Wah

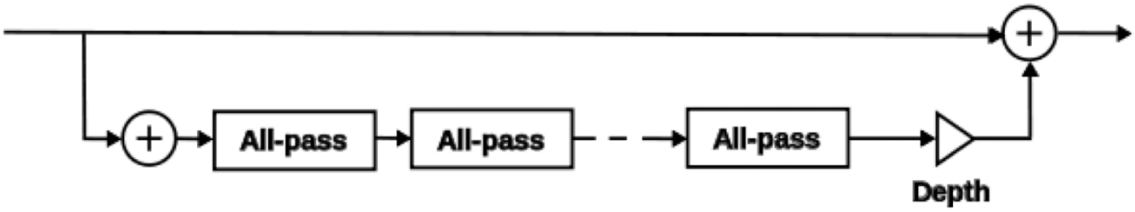




In this case, the user can modify the frequency *mfreq* of the oscillator and its central frequency. The *mfreq* ranges between 0 and 1, and its cutoff frequency follows the central one, which base is 800 Hz. It is then shifted by LFO * 400, and so its range goes from 800 to 1200.
We use *Lag.kr* to make the cutoff smoother and then we make a cut around the cutoff frequency.
So, this effect consists in a passband filter, whose central frequency varies over time in a prescribed range and according to a modulating function. We output the processed signal mixing it with the original signal.

```
(
SynthDef("wah", {arg inBus, outBus;
      var sig, out, lfo, cFreq, cutoff, resonance, freq, mfreq, mix;
      mix = 0.5;
      sig = In.ar(inBus, 1);
      mfreq = 0.25;
      cFreq = 800;
      lfo = SinOsc.kr(mfreq).range(0, 1);
      cutoff = cFreq + (lfo * 400);
      resonance = 0.5;
      freq = Lag.kr(cutoff, 0.1);
      out = BPF.ar(sig, freq, resonance);
      out = Mix.ar([out*mix, sig*(1-mix)]);
      if (~wahActive == 1, {
            Out.ar (outBus, out);
      }, {
            Out.ar (outBus, sig);
      });
}).send(s);
);
```

Phaser



As far as this effect is concerned there is a modulation signal with *mfreq* and a delay is applied to a certain frequency, which is the sum of the base frequency (*cfreq*) and a modulating sine wave. The delay is defined by *CombL*, which creates a pulse train. Then we apply an all-pass filter using the delayed signal and we mix it with the received signal. So, the output of the Phaser effect is the modified waveform after consecutive filters.

```
(
SynthDef("phaser", {arg inBus, outBus;
    var sig, out, delay, lfo, freq, cfreq, depth, feedback, mfreq;
    sig = In.ar(inBus, 1);
    mfreq = 0.5;
    lfo = SinOsc.kr(mfreq).range(0, 1);
    cfreq = 0.1;
    freq = cfreq + (lfo * 0.1);
    depth = 0.5;
    delay = CombL.ar(sig, 0.05, depth, freq);
    feedback = 0.5;
    delay = AllpassN.ar(delay, 0.05, { cfreq + (lfo * 0.1) }, 1);
    out = Mix.ar([sig, delay]) * feedback;
     if (~phaserActive == 1, {
          Out.ar (outBus, out);
     }, {
          Out.ar (outBus, sig);
     });
}).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. The chain of the effects can be changed by the user through "*Move Before*" and "*Move After*" buttons. These two change the order of the layout and the order of the variables based on the position of each effect.