**Creating a perfectly reconstructive and linear-phase 1/3 octave filterbank**

**By Angelo Farina, 24/02/2023**

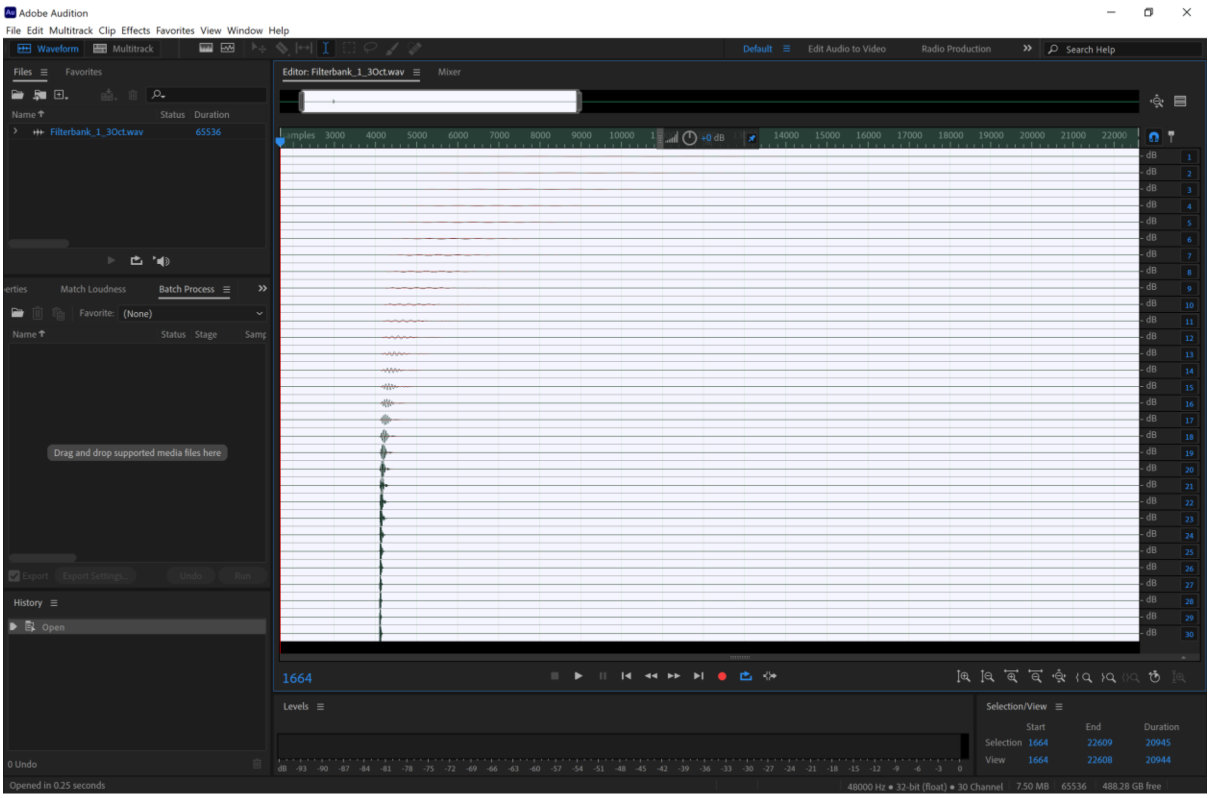
Most filterbanks available in Matlab or elsewhere are NOT perfectly reconstructive, nor linear-phase. This means that summing all the filters you do NOT get a perfectly flat spectrum, nor a perfect Dirac Delta function in time domain.

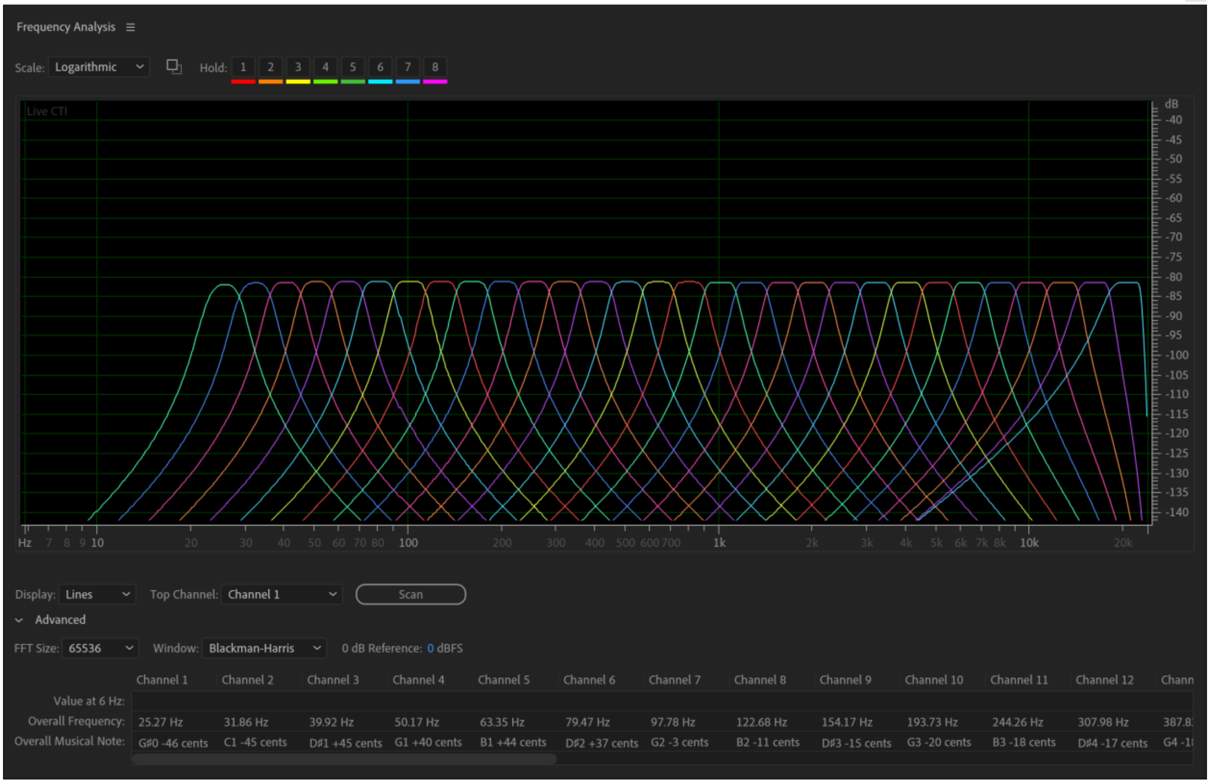
Hence I built a perfectly reconstructive filterbank making use of Matlab, Plogue Bidule. X-MCFX VST plugin, Adobe Audition (both CC 2023 and 1.5) and my Aurora Invert Kirkeby plugin.

We start with a simple Matlab script generating a 30-bandes 1/3 octave FIR filterbank, starting from a Dirac pulse at sample 4096, with a length f 65536 samples:

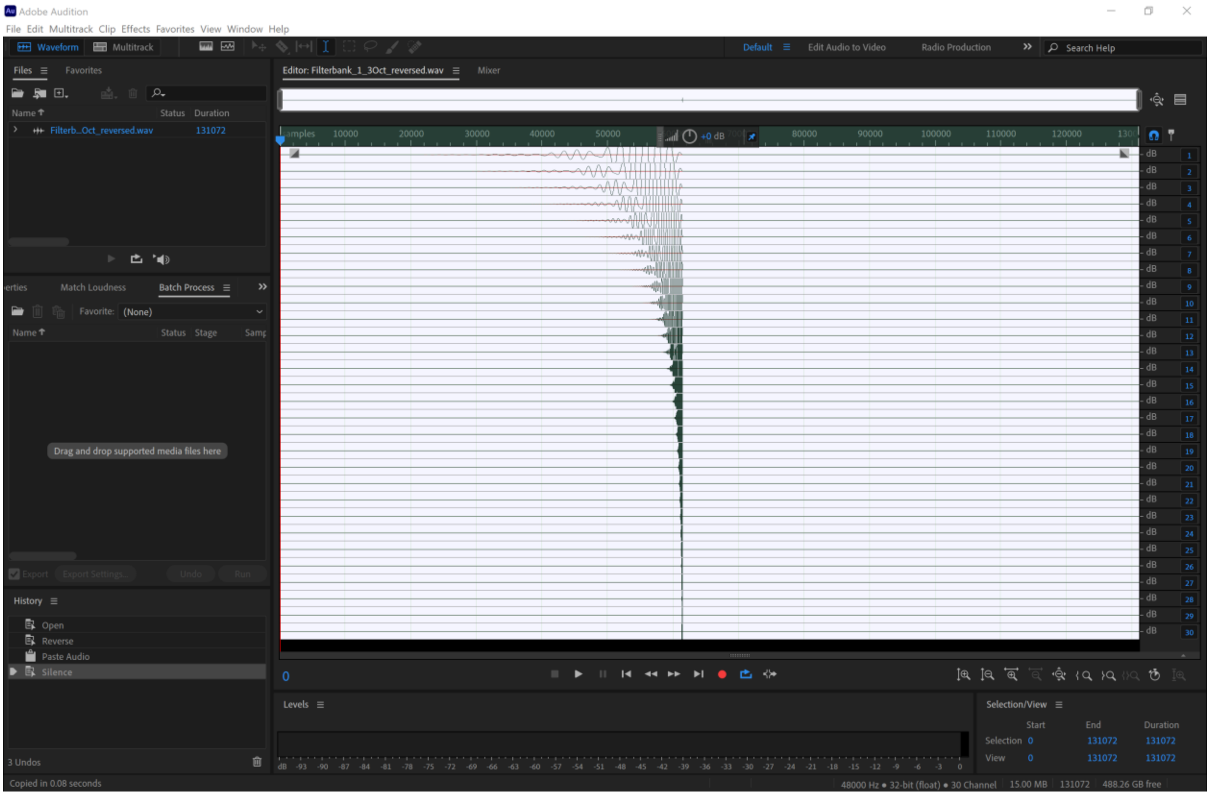
```
% generate a perfectly reconstructive filterbank
fs=48000;
octFiltBank = octaveFilterBank('1/3
octave','FrequencyRange',[25,20000],'FilterOrder',6,'SampleRate',fs)
x=zeros(65536,1);
x(4096,1)=1;
y = octFiltBank(x);
audiowrite('Filterbank_1_3Oct.wav',y,fs,'BitsPerSample',32);
```

The resulting 30-bands WAV file contains a set of minimum-phase, not reconstructive filters, as shown here:

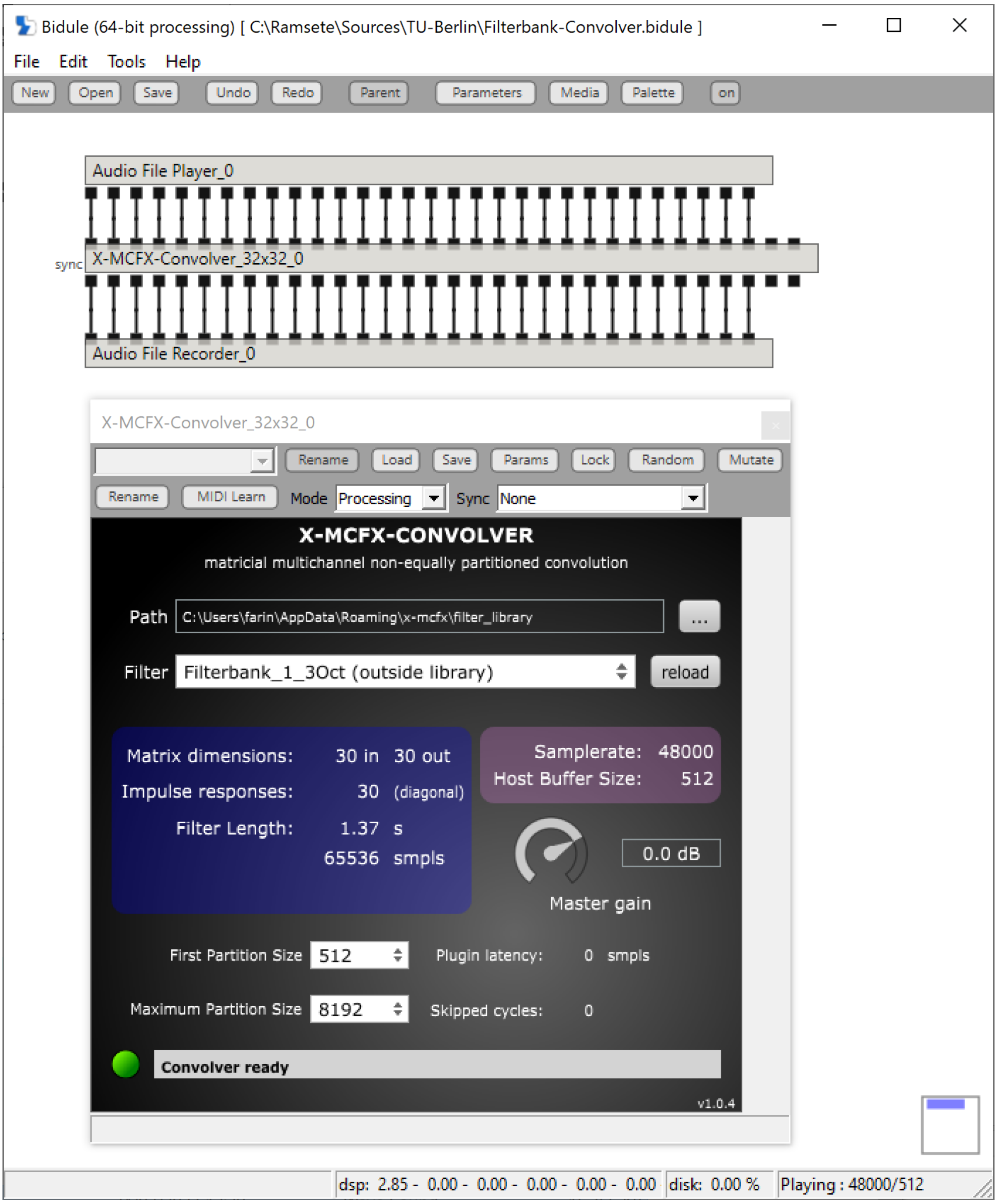




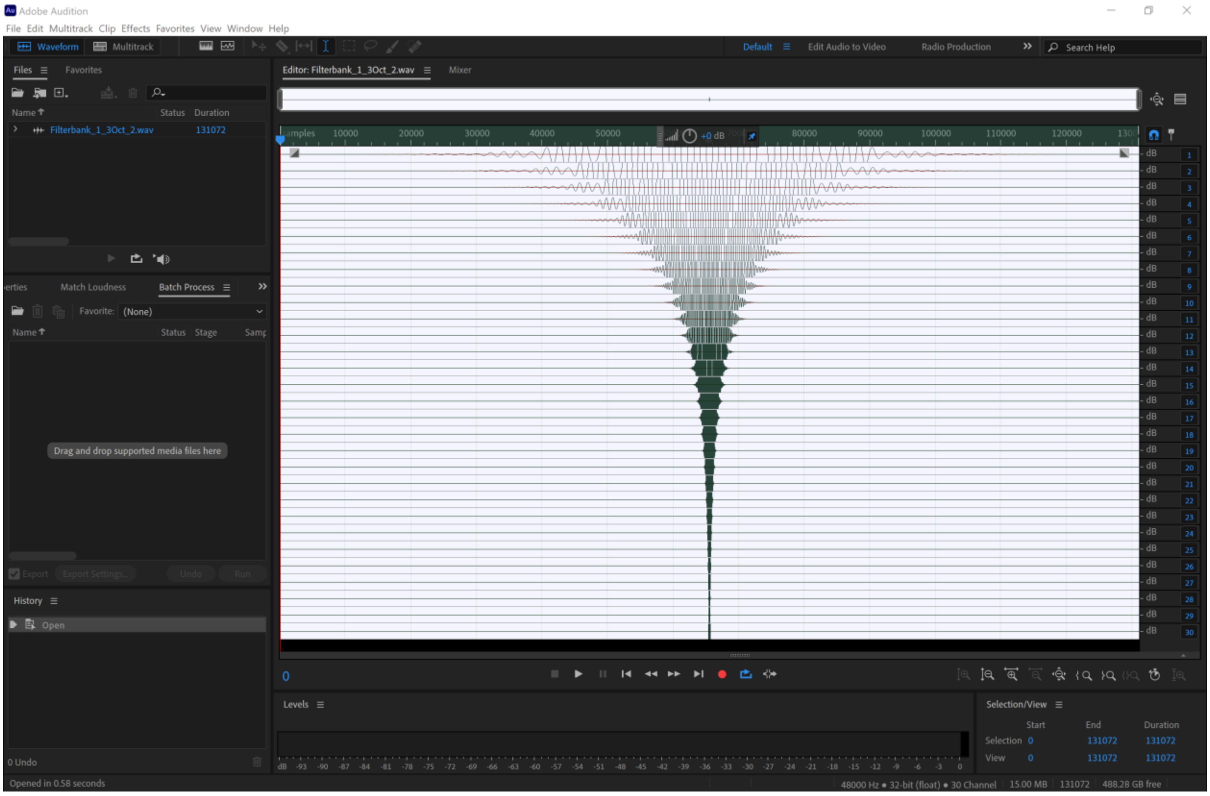

For making the filterbank linear-phase, the simplest approach is to time reverse it, zero pad with other 65536 samples, and then convolve it with the original filterbank.
The Time reversal and zero padding are done in Audition CC, and the resulting reversed and elongated WAV file is saved as Filterbank_1_3Oct_reversed.wav:

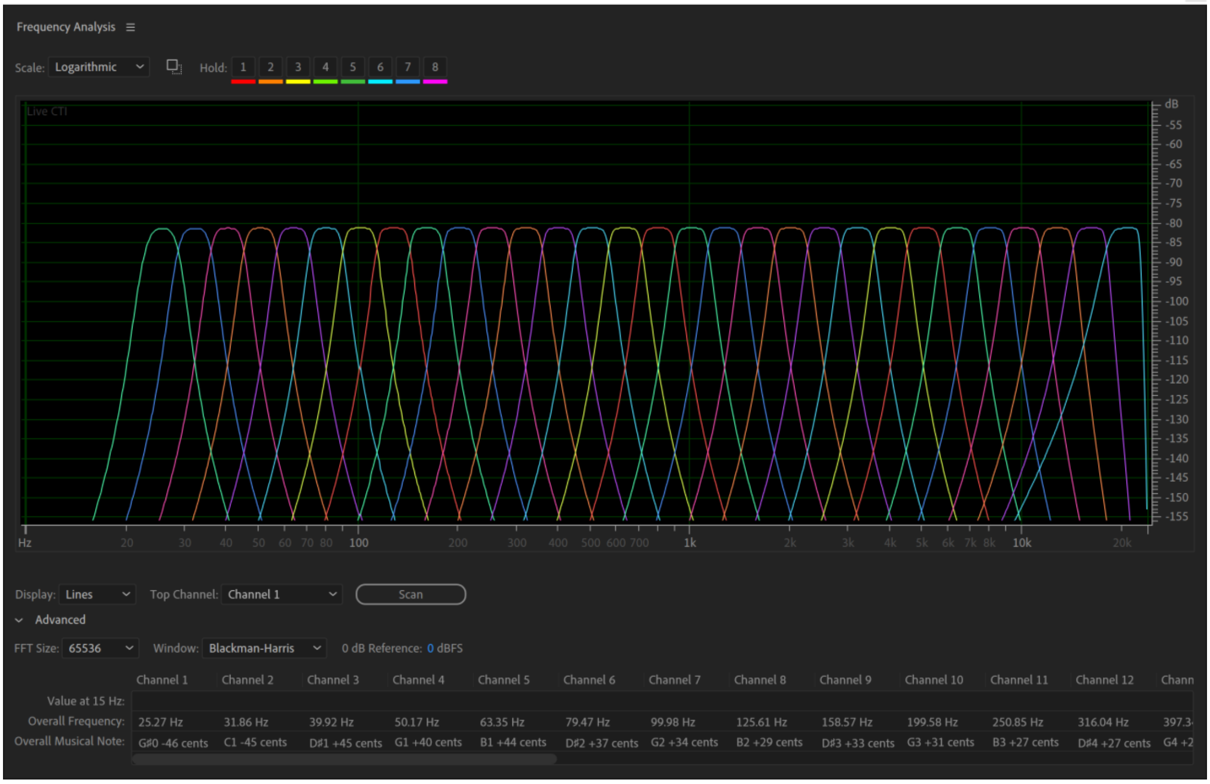


We now use Plogue Bidule and my fast convolution plugin X-MCFX for performing the convolution between this time-reversed filterbank and the original one, saving the result in a third filterbank, which is now linear-phase and doubled-order (order 123 instead of order 6). Her the patch in Bidule:



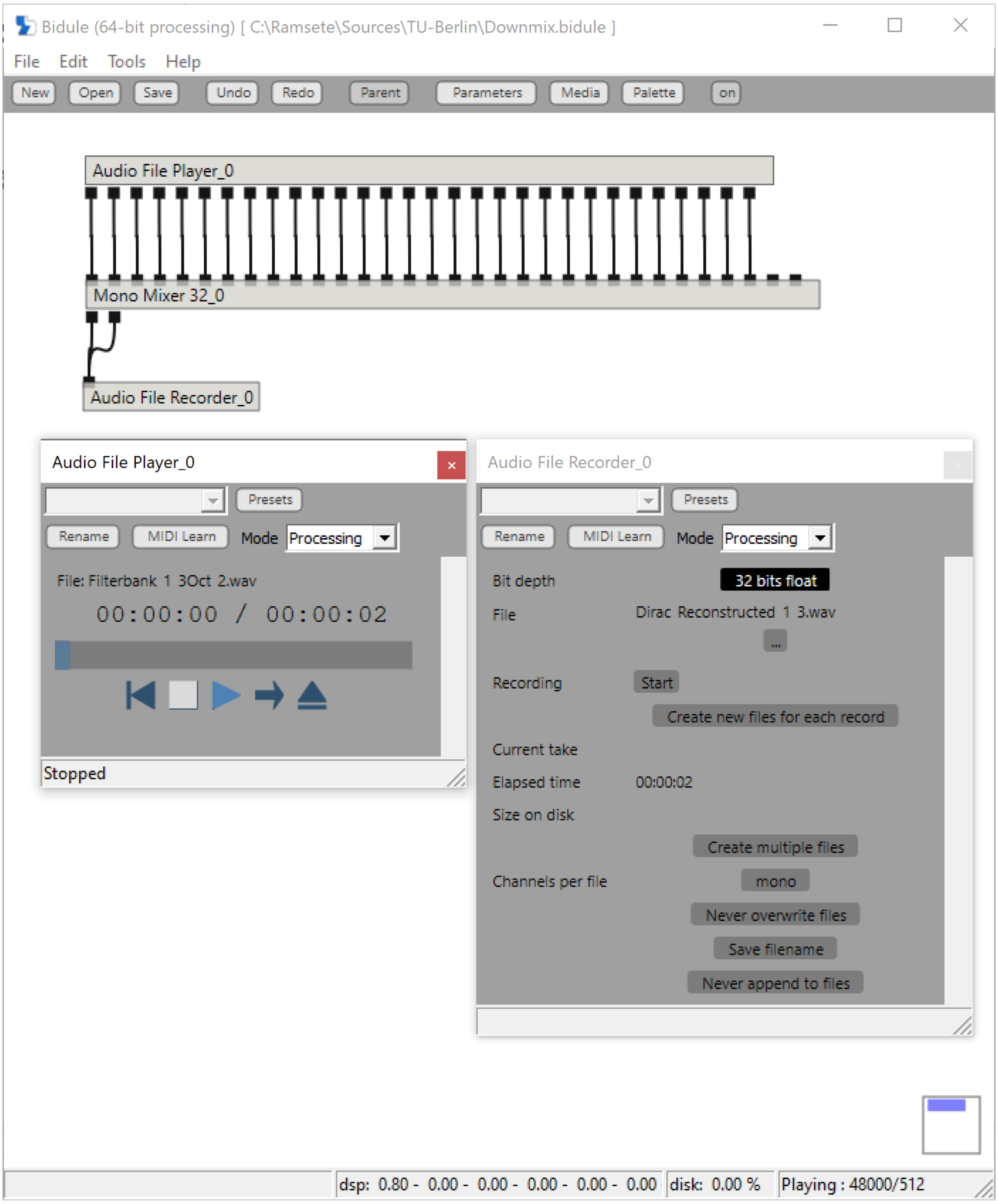

The result is a new filterbank with linear phase:



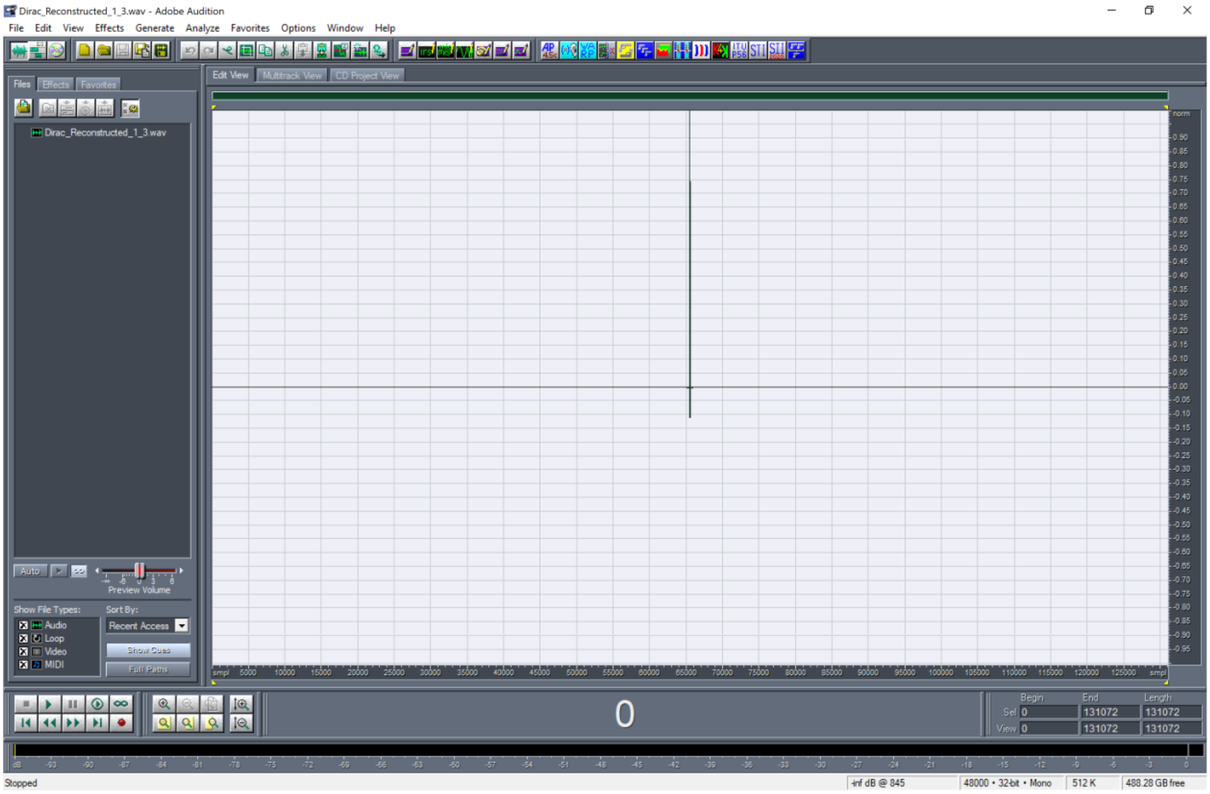



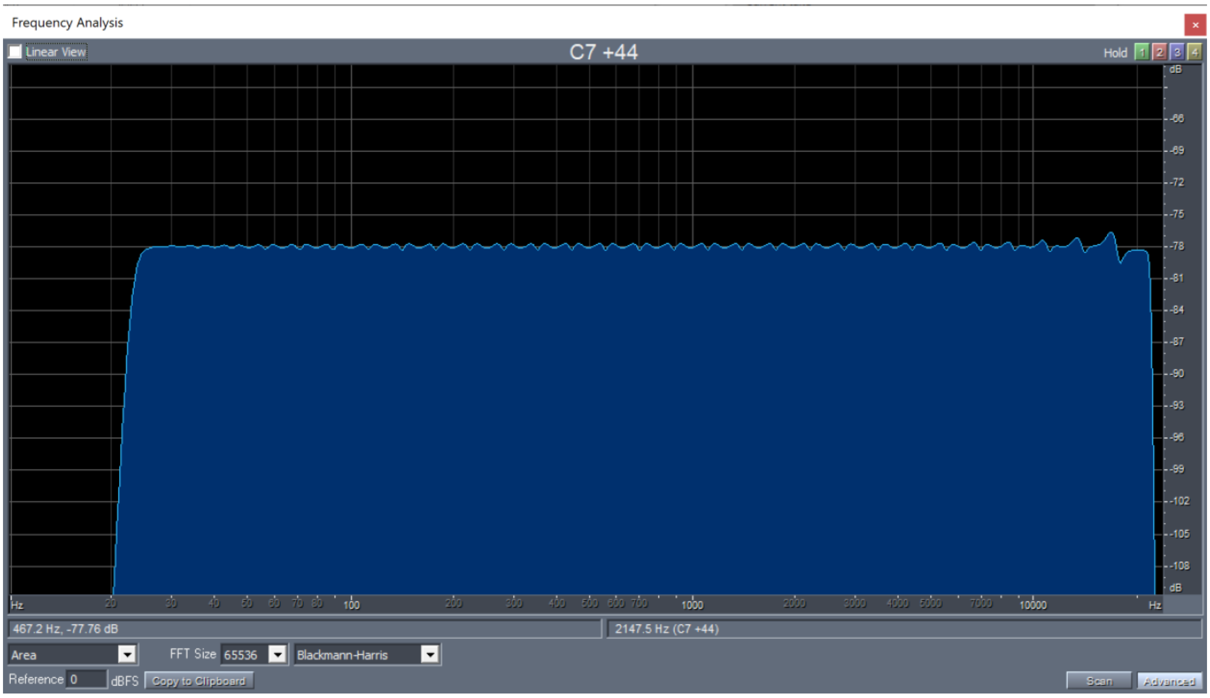

Thanks to this double filtering, the FIR filters are now perfectly symmetrical in time domain (linear phase) and their amplitude spectra cross at -6 dB instead of -3 dB, which is a prerequisite for reconstructive filterbanks.

We can now check the reconstructivity by summing all the filters, creating the reconstructed Dirac function, as shown here:

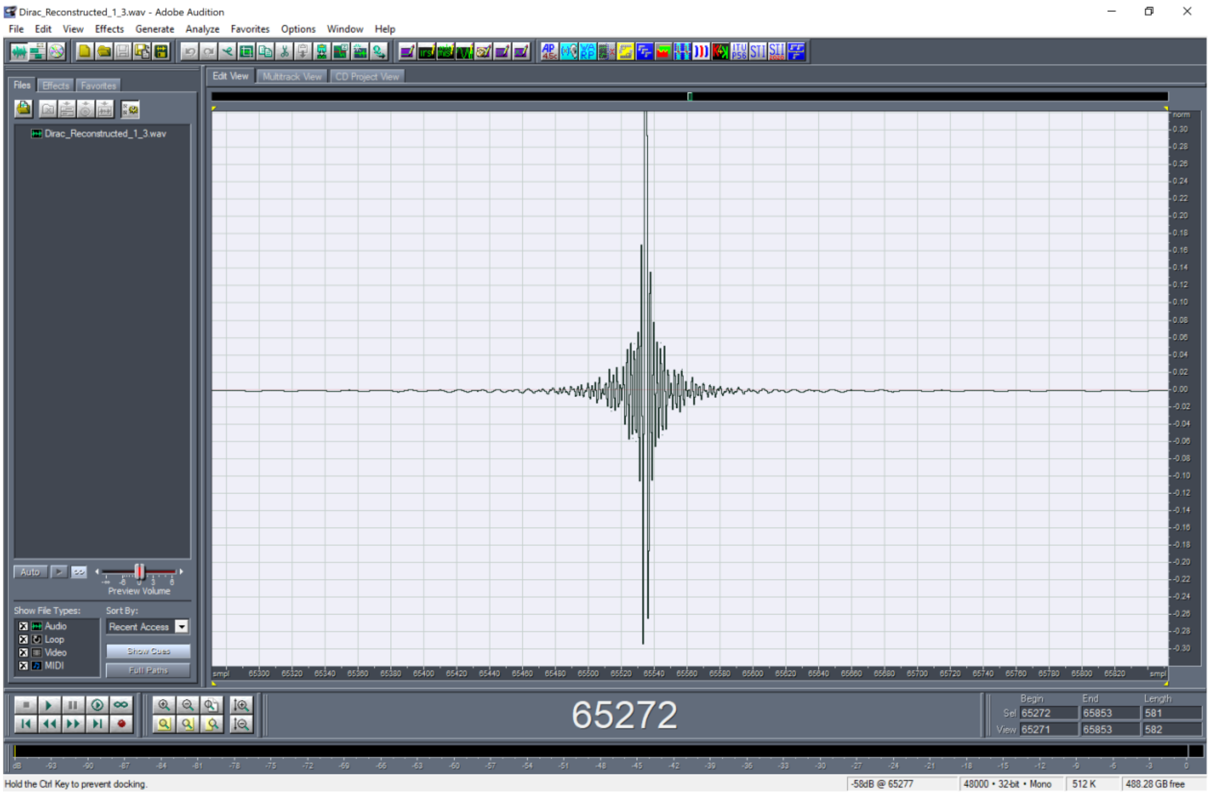



Here the resulting mono signal in the good old Audition 1.5:



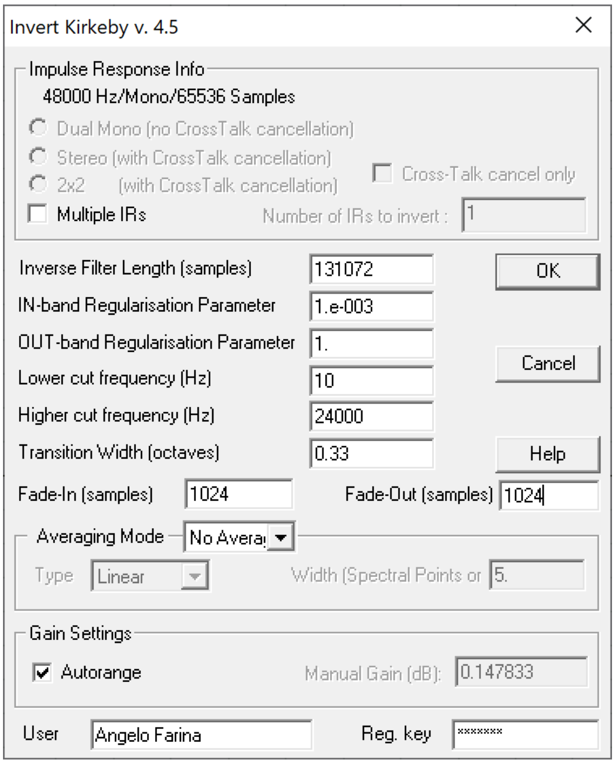




The result is good, but NOT perfect, as the frequency response has some ripple, and the time domain waveform is NOT a perfect Dirac's Delta, as you can see zooming:

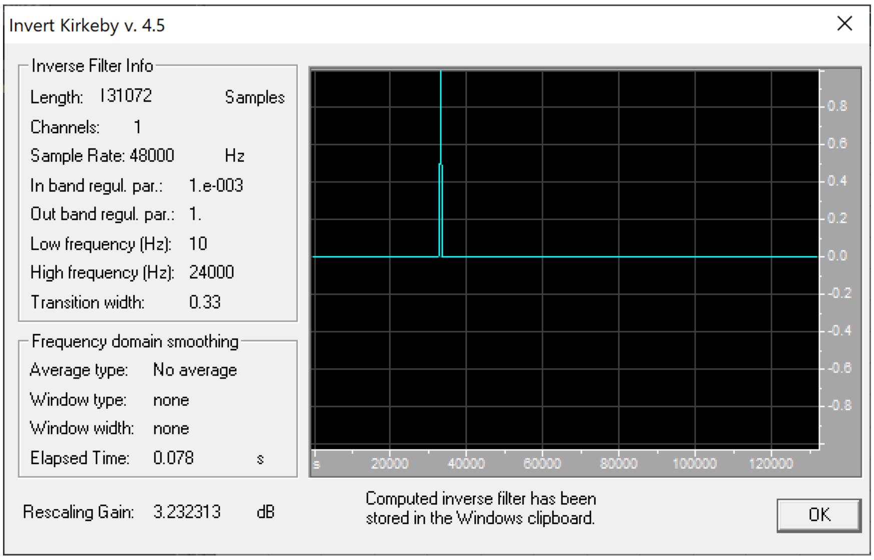


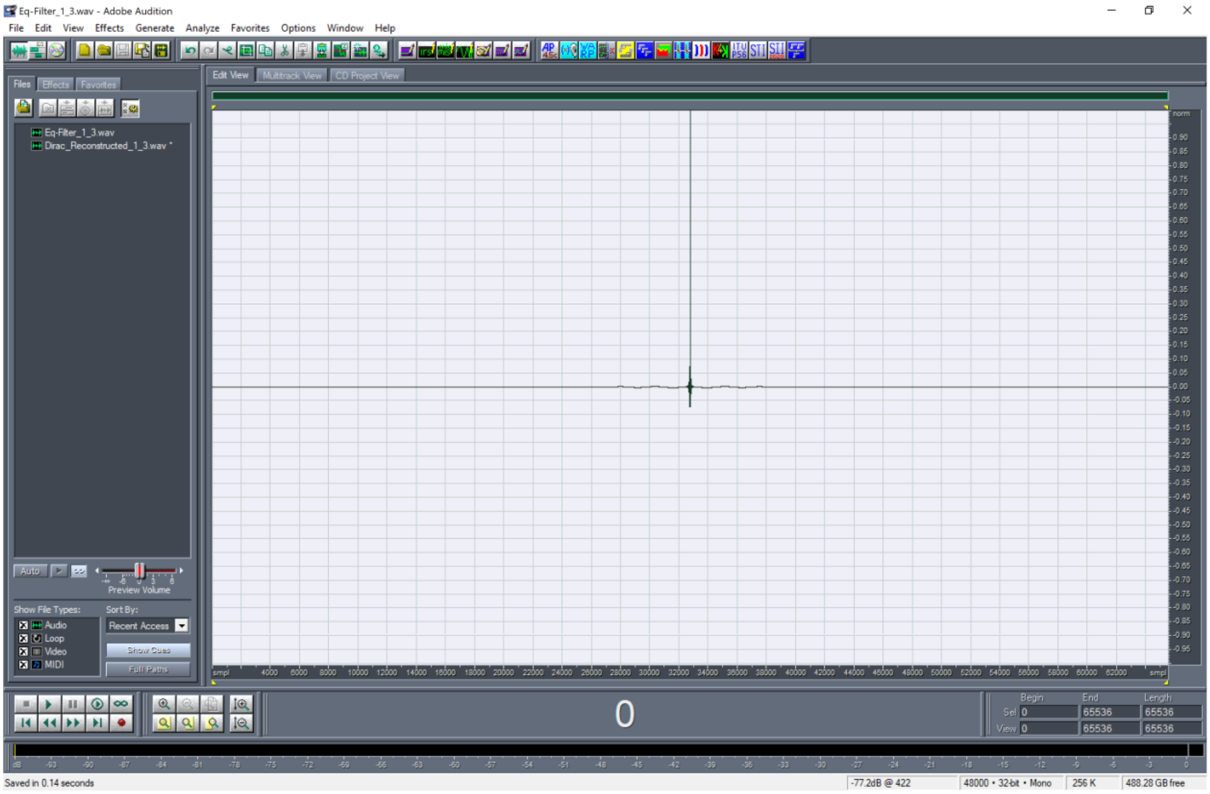

For improving the reconstructivity, we need to create a Kirkeby inverse filter of this "imperfect" Dirac's Delta, and use it for feeding from start the whole generation process.

Here the settings for the Invert Kirkeby plugin applied to the reconstructed Dirac's Delta:




The result is an inverse filter, performing the required equalisation, which is saved as Eq_Filter_1_3.wav:

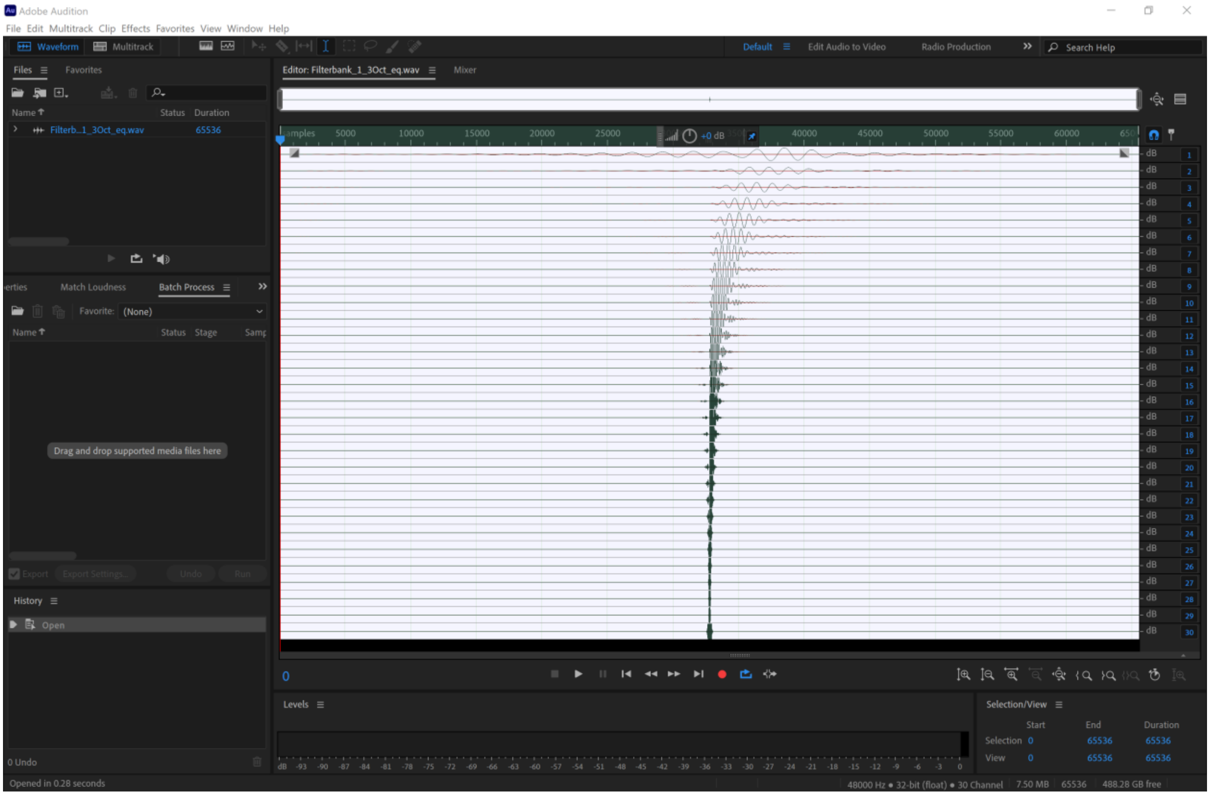


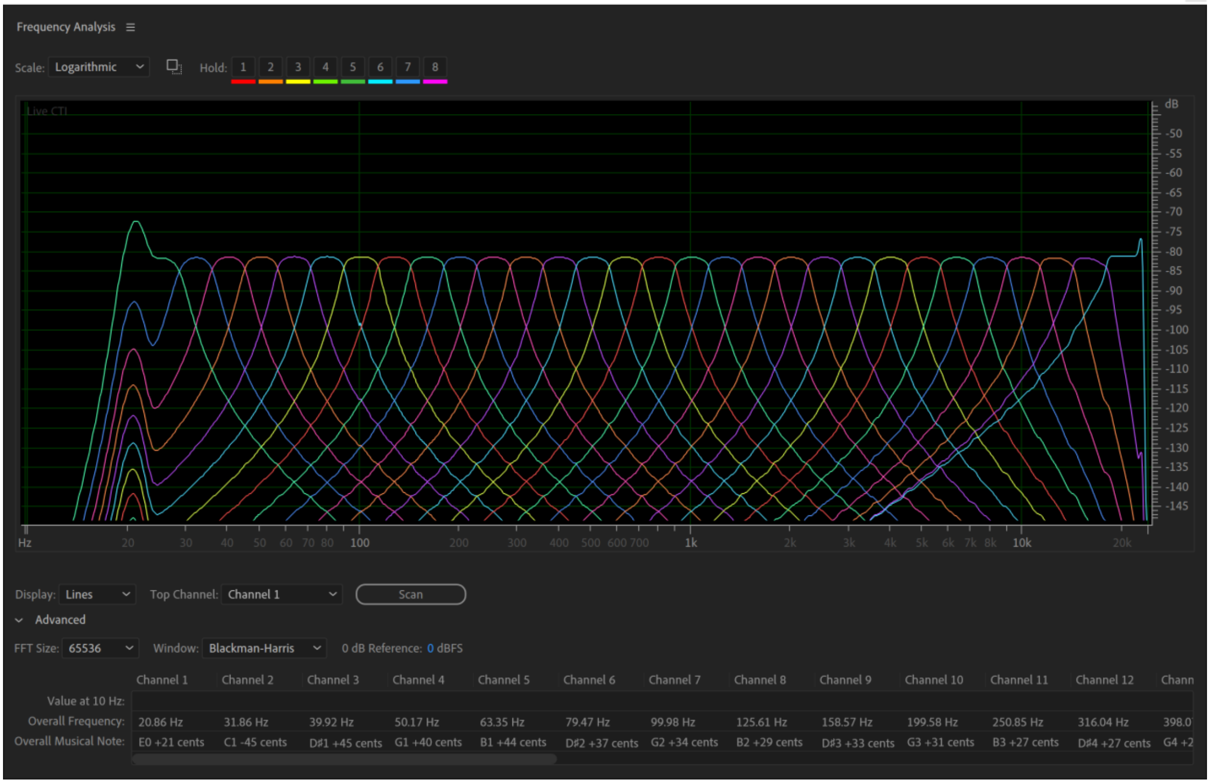




Now we come back to Matlab, modifying our script so that we now start the generation of the filterbank from this pre-equalised Dirac's Delta, instead of from a "perfect" Delta function:

```
% generate a perfectly reconstructive filterbank
fs=48000;
octFiltBank = octaveFilterBank('1/3
octave','FrequencyRange',[25,20000],'FilterOrder',6,'SampleRate',fs)
[x fs]=audioread('Eq-Filter_1_3.wav');
y = octFiltBank(x);
audiowrite('Filterbank_1_3Oct_eq.wav',y,fs,'BitsPerSample',32);
```

This outputs a new pre-equalised filter bank, which indeed is still minimum-phase and not fully reconstructive, as shown here:



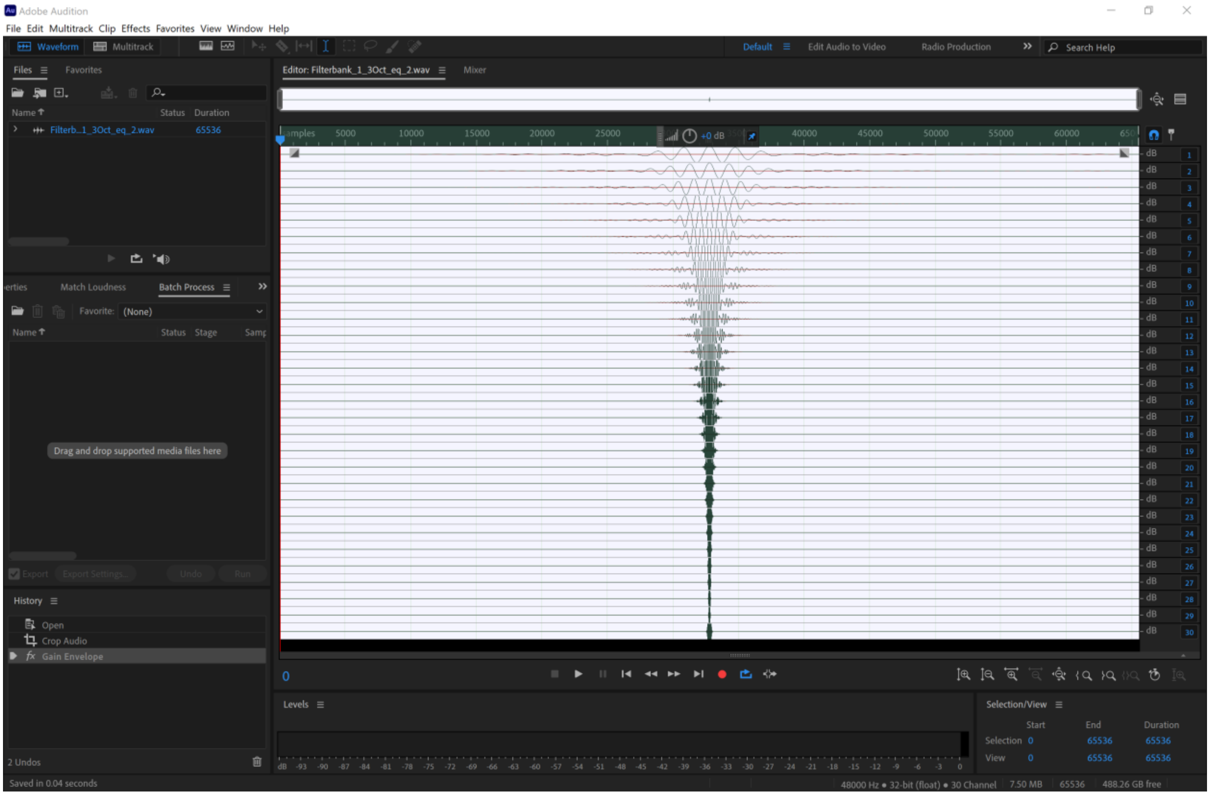



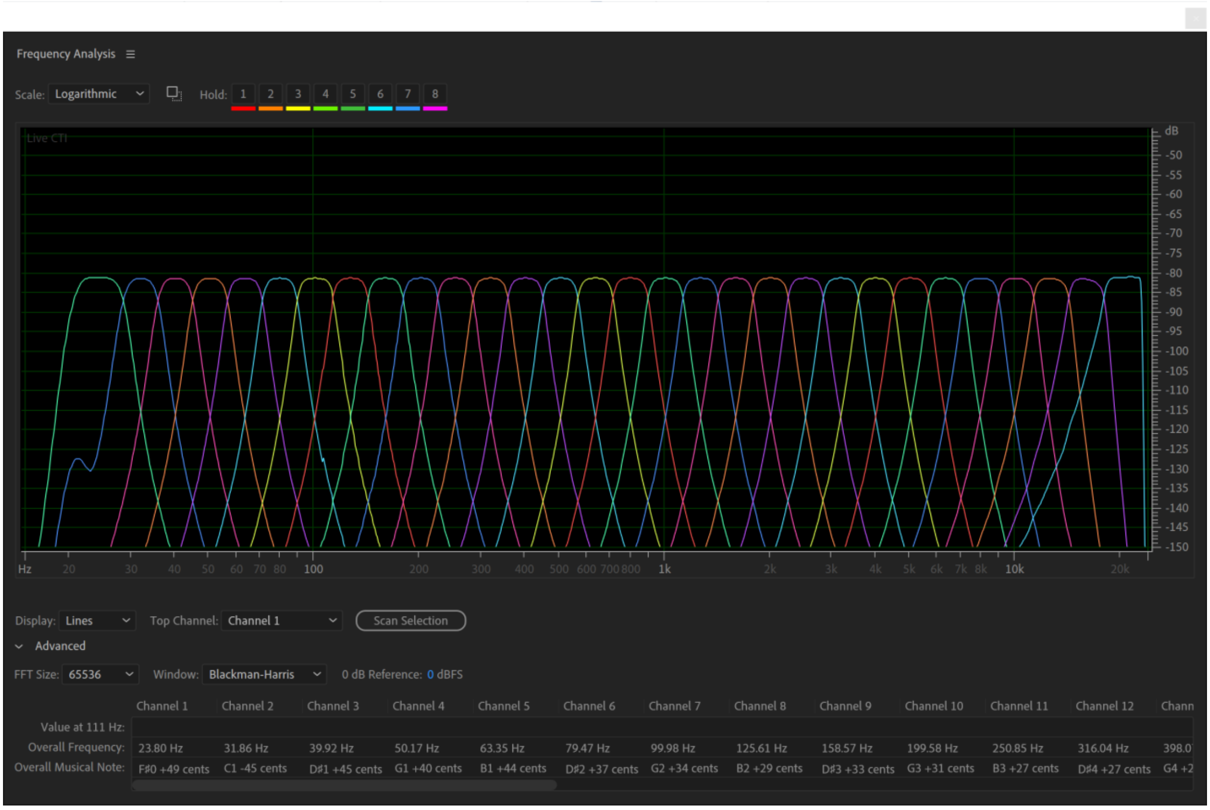

Please note how the pre-equalization now affects the frequency responses of the filters, particularly at very low and very high frequencies.

We have now to repeat the same procedure already done for the not-pre-equalised filterbank, that is making a time reversal of this new pre-equalised filterbank, zero-padding, and convolving with the not-pre-equalised filterbank.

It is important to use the previous not-pre-equalised filterbank. If you convolve the time-reversed pre-equalised filterbank with the pre-equalised filterbank itself, you end up applying the pre-equalization TWICE, which of course would be wrong.

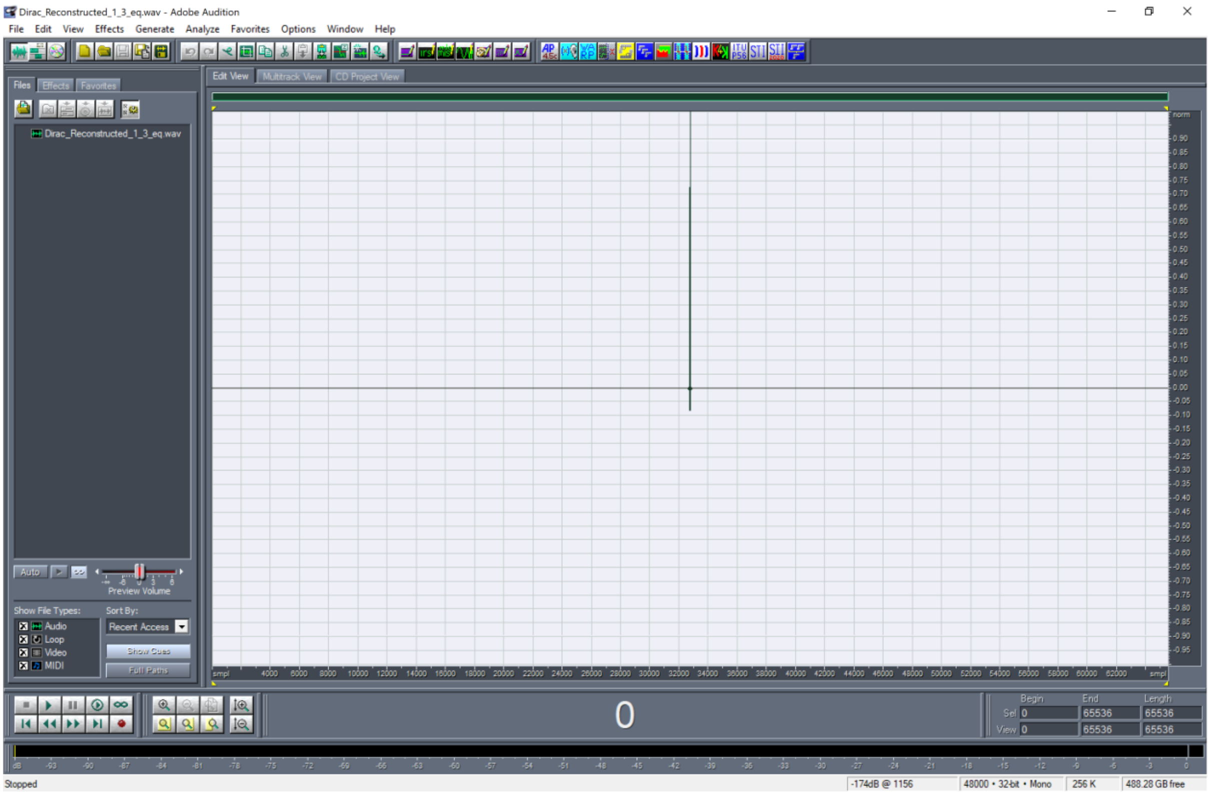
Here the result, after cutting the result to 65536 samples, centered on the filter's central spike:

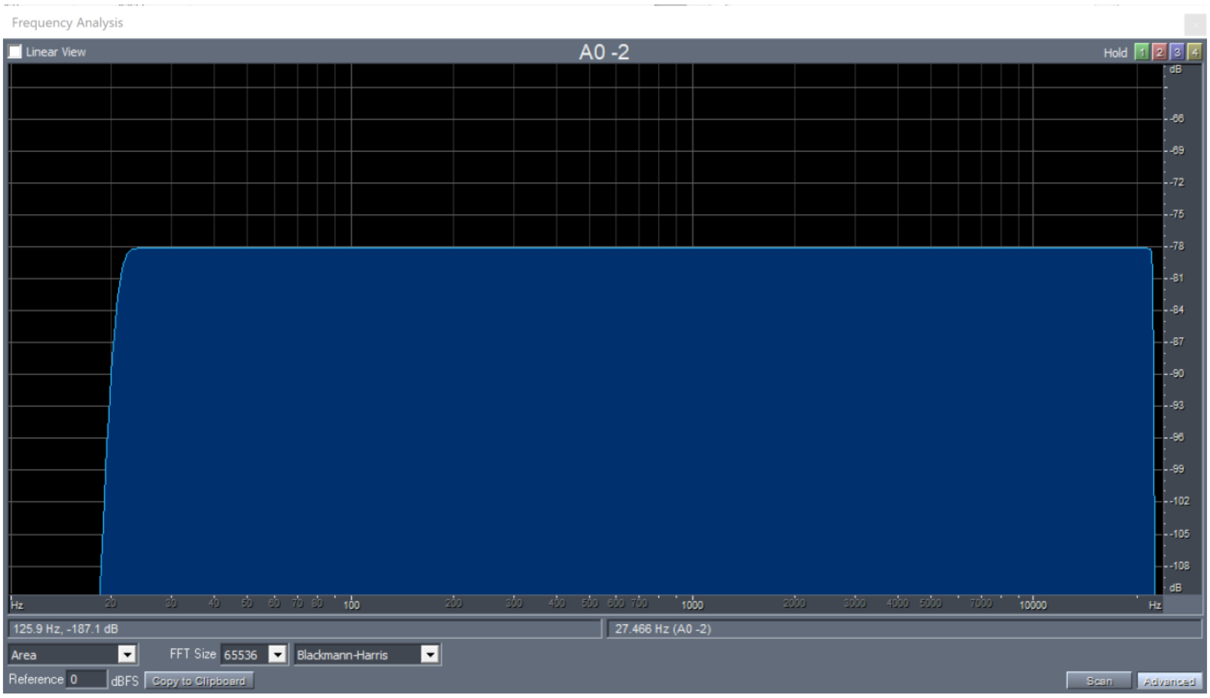



You can see how the filters are now perfectly symmetrical in time domain.

Finally we check the reconstructivity of this filter bank, downmixing it to a mono signal. Here the result:




The result is now PERFECT, both in time domain and in frequency domain. So we have now a perfectly reconstructive, linear-phase 1/3 octave filter bank.
I produced also an 1/1 octave filter bank with exactly the same procedure.

Both filter banks are available either in a “long” format (65536 samples) and in a shorter format (16384 samples). They can be further shortened if accuracy at very low frequency can be compromised.