Technology
Arts Sciences
TH Köln

Faculty 07
Digital Communications Lab

SMP: Signal Processing with Micro Controller and Python

Lab 2: Finite Impulse Response Filters with GNU Radio

Version 1.0

October 18, 2023

1 Introduction

1.1 General rules

Questions must be answered in written form at home in advance.

Measurements will be worked out and documented during the lab.

Extra measurements are not mandatory but can be elaborated to gain a deeper understanding.

Please note:

- Figures are to be labeled before printing. They should have a title and marked axes.
- Play around and modify the schematics to try out your own ideas.

1.2 Bibliography and links

- GNU Radio Tutorials https://wiki.gnuradio.org/index.php/Tutorials, access October 10, 2018
- Ohm, Jens, Lüke, Hans Dieter, 'Signalübertragung', Springer, 2014
- Elders-Boll, 'Vorlesungsskript ASS/DSS TH Köln', 2018
- Schlichthärle, Dietrich, 'Digital Filters', Springer, 2011

2 Overview

In this semester we use GNU Radio Companion to generate and test signal processing algorithms in the audio range in real time.

GNU Radio is a open-source software development toolkit that provides signal processing blocks to implement software radios. It can be used either stand alone for simulation purposes using, e.g., internal or external audio sources connected to the computer or it can be

combined with external RF hardware to create software-defined radios. It is widely used in research, industry, academia, government, and hobbyist environments.

2.1 Learning goals

- designing filters with the GNU Filter Design Tool
- building and testing different types of FIR filters

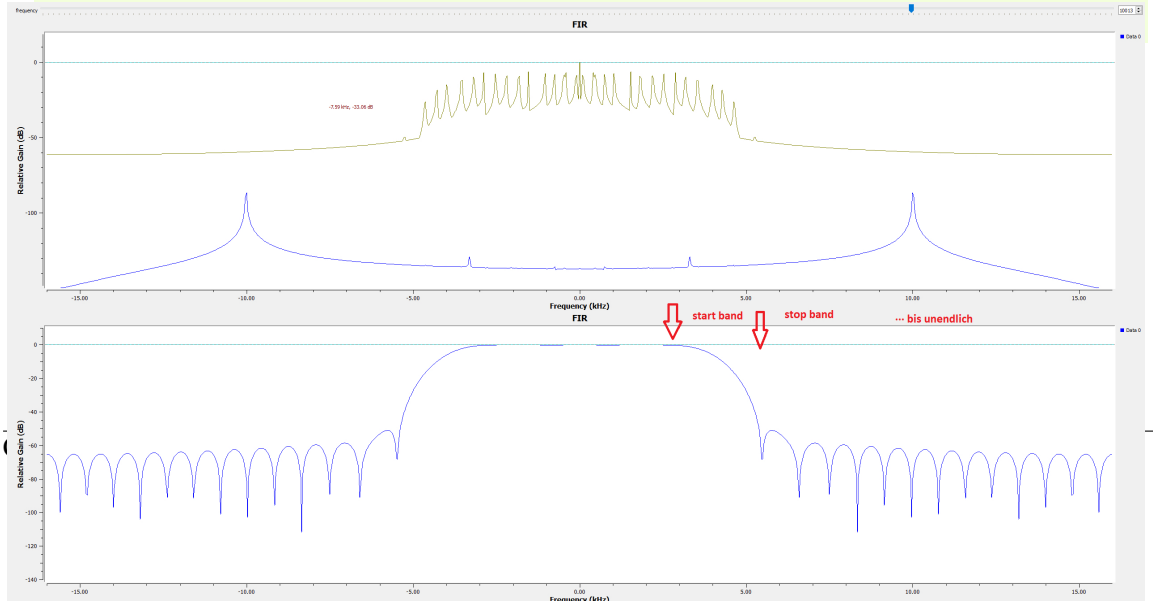
3 Basic schematics for low pass filters

3.1 Low Pass Filter Example

This tutorial explains a basic flowgraph design.

Lab Exercise: Low Pass Filter Example

1. Go to the web page https://wiki.gnuradio.org/index.php?title=Low_Pass_Filter_Example and follow the instructions. Use floating point numbers instead of complex numbers (as in the tutorial) to apply pure cosine signals to the filter input.
2. How can you use the set-up with the cosine input signal to measure the transfer function of the filter? Why this is possible? Plot your result!
3. Can you read out the transition and stop range of the filter? What defines the "stop band attenuation"?
4. What means "cut-off frequency"?
5. Now connect an additional noise source by a switch element and choose it: What do you observe in the frequency domain for the filter transfer function? Can you explain the result?



Your Solution:

2. By using the slider in combination with max hold, the attenuation for each frequency can be measured. The displayed result shows the frequency response of the filter, which resembles the transfer function (shown in the graph below).

3.
transition range: 4-6kHz (where the graph starts dropping)
stop range: 6-inf kHz (maximum signal frequency)
The stop band attenuation is generally defined by the number of taps (filter coefficients), in GnuRadio we did not find a parameter which sets the attenuation directly, but it is influenced by the transition width (higher transition with -> higher attenuation and vice versa).

4. The cut-off-frequency describes the frequency after which the filter starts attenuating the signal.

5. The transfer function can be seen directly, since white noise contains all frequencies equally and therefore no slider for the cosine frequency is required.

Demonstration of the effect of low pass filtering to an audio signal

Lab Exercise: Listen to a low pass filtered audio signal

1. Create a schematic according to figure 1.
2. Chose an appropriate audio file and start the simulation.
3. Switch between the filtered and unfiltered state.
4. Change the filter parameters and watch what happens.

3.2 Moving Average Filter

This tutorial explains how to specify and use filter coefficients for a simple moving average filter.

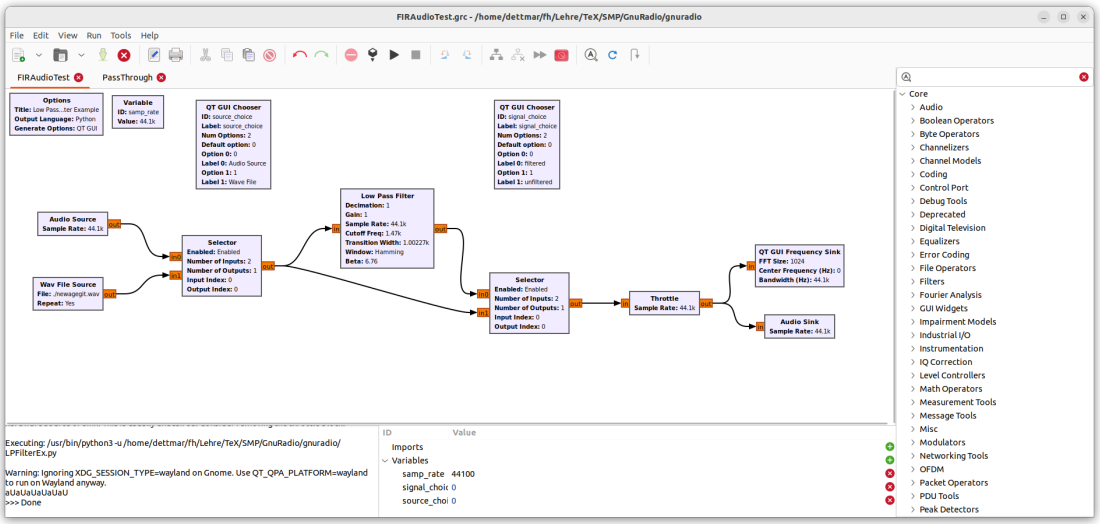


Figure 1: Schematic for audio tests

Lab Exercise: Frequency Xlating FIR Filter

1. Go to the web page https://wiki.gnuradio.org/index.php?title=Designing_Filter_Taps and follow the instructions. Use floating point numbers instead of complex numbers (as in the tutorial) to apply pure cosine signals to the filter input. Note that the Xlating FIR Filter always has a complex output.

2. First use the "Low-Pass Filter Taps block" to reproduce the result from the last experiment but now using the "Frequency Xlating FIR Filter" block.

3. Next follow the instructions to enter the filter taps manually in form of an numpy array. Here, constant values for the filter taps are used. Start the simulation to generate the transfer function of the moving average filter.

4. Which mathematical function describes the transfer function of the filter? **Compute** the first zero in the transfer function for the given filter as a function of the sample rate. Which value f_0 results for a sampling rate of $f_s = 32$ kHz?

5. What do you have to change, so that the first zero is located at $f_0 = 2$ kHz?

6. Now plot the transfer function of the filter for $f_s = 8$ kHz and $np.ones(5)/5$ for later comparison.

Your Solution:

3.3 General Finite Impulse Response filters

Finite Impulse Response (FIR) filters are a family of digital filters with linear phase response. They are non-recursive an thus have a finite impulse response. However, to achieve the same steepness in the transition region of the filter they need in general more coefficients (and complexity) than IIR Filters. They are often called transversal filters, too.

3.3.1 FIR filters with variable coefficient values

We now consider the more general case of FIR filters with coefficients $h[i]$ of different values. Those filters allow a flexible adaptation of the filter properties to the user requirements.

Question: Frequency Response of an FIR Filter
Use the discrete time Fourier transform (DTFT) to derive an algebraic expression for the frequency response of the filter having coefficients $h[n] = (0.0833, 0.2500, 0.3333, 0.2500, 0.0833)$. Write down each step of the derivation in the space provided below and sketch the magnitude of that theoretical frequency response on the axes of figure 2 for a sampling rate of $f_S = 8\,\text{kHz}$

Your Answer:

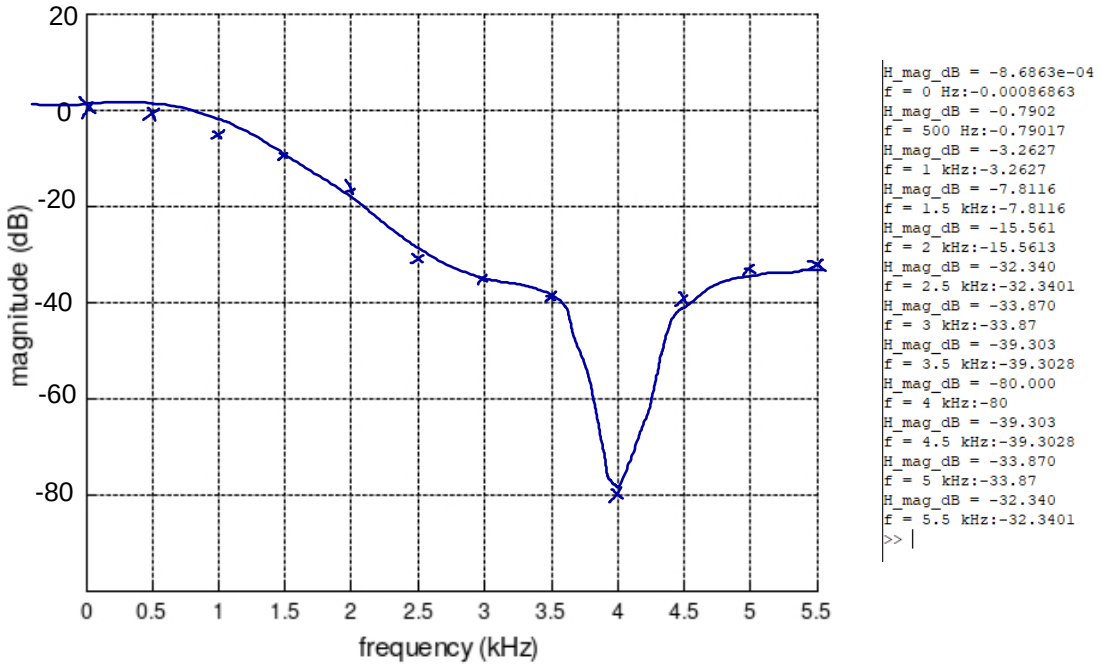


Figure 2: Transfer function of $h[n]$

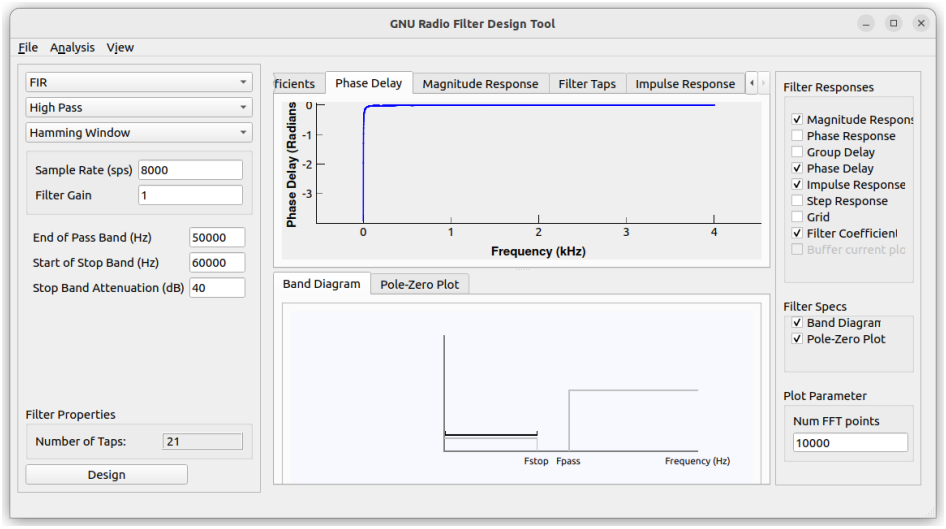
Lab Exercise: FIR low pass filter

- Modify the variable `filter_coeff` of the schematic so that $h(n) = (0.0833, 0.2500, 0.3333, 0.2500, 0.0833)$.
- Observe the frequency response of the filter.
- Now change the sampling rate to $f_s = 8\,\text{kHz}$ and compare the plot with the curve from your calculation.

Your Answer:

3.4 Generating FIR filter coefficients using the GNU Radio Filter Design Tool

GNU Radio Filter Designer allows it easily to design various types of filters, look at the most interesting parameters and read out the filter coefficients to use it in GNU Radio or any programmable HW:



Lab Exercise: FIR high pass filter

- Use the GNU Radio Filter Design tool to compute the filter coefficients for a FIR Filter with a stop band below 2.5 kHz, the passband starting at 3 kHz, an attenuation of 30 dB, a sampling rate $f_s = 8\,\text{kHz}$, and a Hamming window characteristics.
- Look at the magnitude response, the pole-zero plot, the filter coefficients and then copy the coefficients into an appropriate variable `filter_coef` of your GNU Radio schematic. Run this schematic. Does the filter perform as designed?
- Now change the sampling rate to other values, design different other types of filters with high pass, low pass, or band bass characteristics and compare the plots with the ones in the designer.

Your Answer:

4 Conclusions

At the end of this lab you should be familiar with the design of FIR filters using GNU Radio. You should understand how to specify arbitrary FIR filters by using the GNU Radio Filter Design Tool.