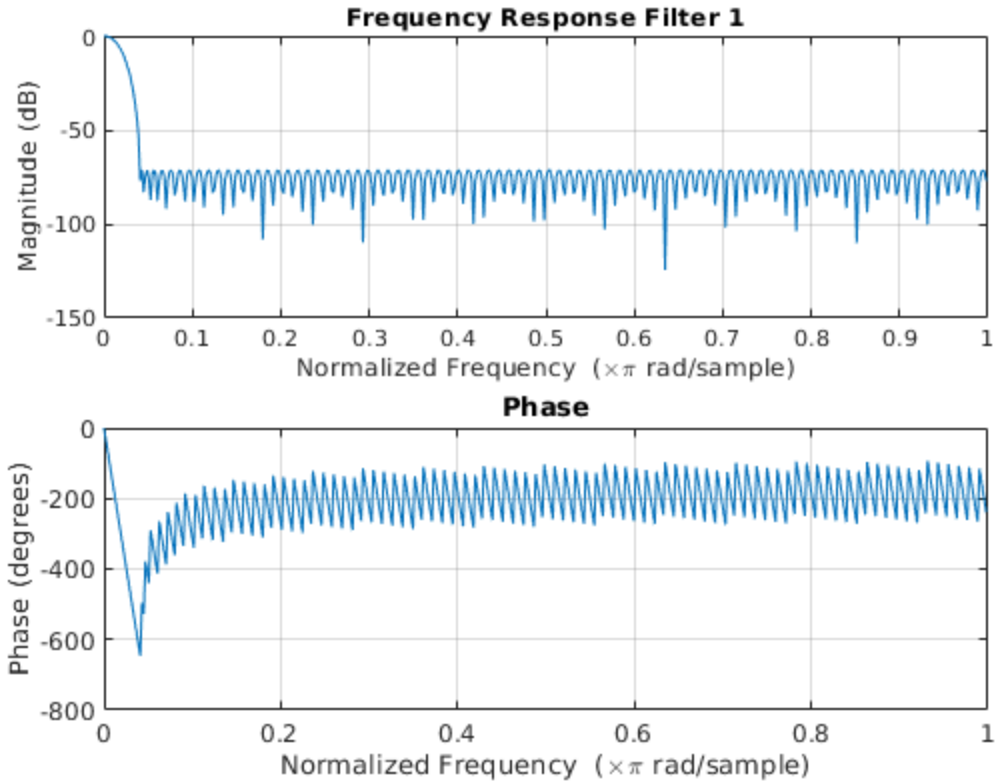
```
%! ------------------------------------------------------------------------
%! SOC Desgin Lab 4
%!  - Create channel filter using a filter chain and model against Xilinx
%!  FIR compiler bit accurate model
%!      - Filter 1: Sample rate = 125Mhz, Fpass = 576kHz, Fstop = 2.549MHz,
%!          Decimation rate: 40, Apass <= .5dB, Astop >= 80dB
%!      - Filter 2: Sample rate = 3.125Mhz, Fpass = 18kHz, Fstop = 30kHz,
%!          Decimation rate: 64, Apass <= .5dB, Astop >= 80dB
%! ------------------------------------------------------------------------

%! Enviorment
clear; close all;
dds_scale       = 2^14;
dds_bit_width   = 16;
scale_val       = 18;
fp_scale        = 2^scale_val;

%! Read filter coefficents from mat files
mat_filter1 = load("filter1.mat");
mat_filter2 = load("filter2.mat");
num_coeffs1 = length(mat_filter1.Num);
num_coeffs2 = length(mat_filter2.Num);

%! Plot Freq response
figure()
freqz(mat_filter1.Num)
title('Frequency Response Filter 1')
figure()
freqz(mat_filter2.Num)
title('Frequency Response Filter 2')

%! Write .coe files for the filter coefficents
% Cast to S16.15 format
filter1_fixed = int32(mat_filter1.Num*fp_scale);
filter2_fixed = int32(mat_filter2.Num*fp_scale);

% Write to file
fid = fopen('filter1.coe', 'w');
fprintf(fid, 'radix=10;\ncoefdata=\n');
fprintf(fid, '%d\n', filter1_fixed);
fid = fopen('filter2.coe', 'w');
fprintf(fid, 'radix=10;\ncoefdata=\n');
fprintf(fid, '%d\n', filter2_fixed);

%! Create FIR bit accurate models
% Filter 1
filter_bit_acc_1 = fir_compiler_v7_2_bitacc();
config_1 = get_configuration(filter_bit_acc_1);
config_1.coeff = round(mat_filter1.Num * fp_scale);
config_1.data_fract_width = 0;
config_1.output_width = 35;
config_1.output_rounding_mode = 0;
```

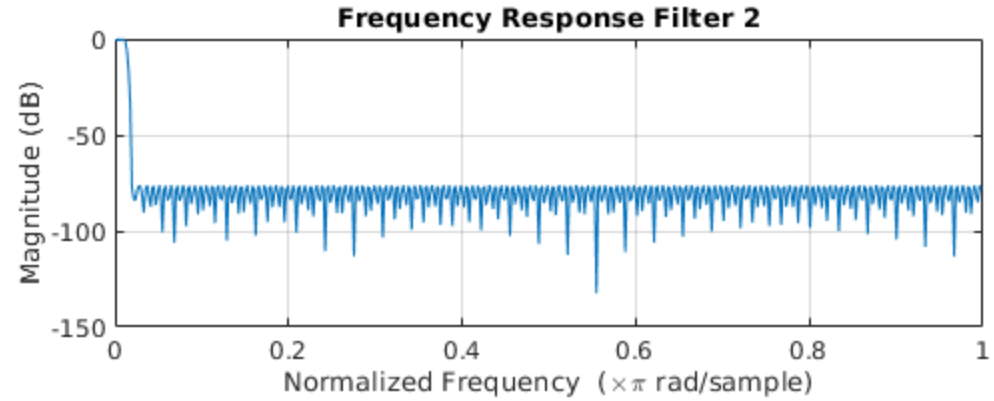
```
config_1.filter_type = 2;
config_1.decim_rate = 40;
config_1.coeff_fract_width = 0;
config_1.output_fract_width = 15;
filter_bit_acc_1 = fir_compiler_v7_2_bitacc(config_1);

% Filter 2
filter_bit_acc_2 = fir_compiler_v7_2_bitacc();
config_2 = get_configuration(filter_bit_acc_2);
config_2.coeff = round(mat_filter2.Num * fp_scale);
config_2.data_fract_width = 0;
config_2.output_width = 35;
config_2.output_rounding_mode = 0;
config_2.filter_type = 2;
config_2.decim_rate = 64;
config_2.coeff_fract_width = 0;
config_2.output_fract_width = 15;
filter_bit_acc_2 = fir_compiler_v7_2_bitacc(config_2);

%! Create test tones
fs = 125000000;
num_end_samples = 1024;
N = num_end_samples * 40 * 64;
f1 = 2000;
f2 = 25000;
f3 = 50000;

signals = zeros(3, N);
signals(1, :) = round(dds_scale*cos(2*f1*pi*(0:N-1)/fs));
signals(2, :) = round(dds_scale*cos(2*f2*pi*(0:N-1)/fs));
signals(3, :) = round(dds_scale*cos(2*f3*pi*(0:N-1)/fs));

%! Loop over the test signals, filter and plot
for i=1:3
    % Filter the signal
    filt_stage_1 = filter(filter_bit_acc_1, signals(i,:));
    filt_stage_1 = round(filt_stage_1 / 2^(scale_val-1));
    filt_stage_2 = filter(filter_bit_acc_2, double(filt_stage_1));
    filt_stage_2 = round(filt_stage_2 /  2^(scale_val-1));

    % Plot the signals
    figure()
    subplot(5,1,1)
    plot(signals(i,:))
    title("Input signal " + i)
    subplot(5,1,2)
    plot(filt_stage_1)
    title("Signal after stage 1 of Cascaded Filters")
    subplot(5,1,3)
    plot((-32768:32767)*fs/40/2^16, abs(fftshift(fft(filt_stage_1))))
    title("FFT of output from Filters 1")
    xlabel("Freq")
    subplot(5,1,4)
    plot(filt_stage_2)
```

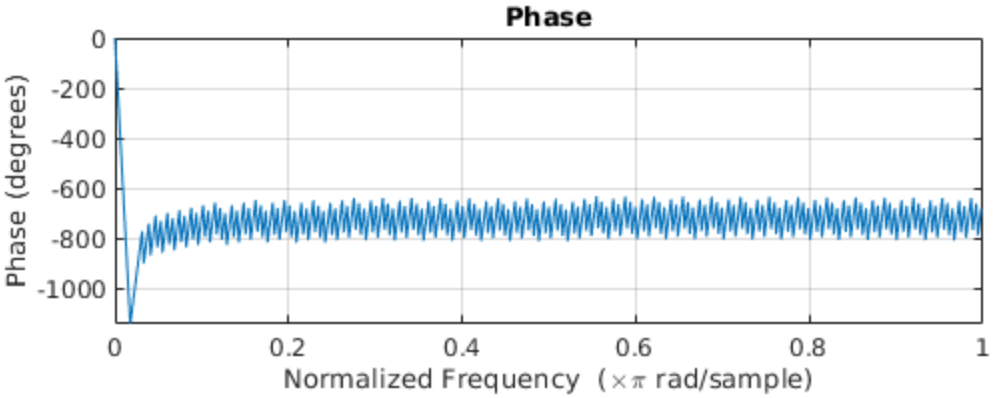
```
    title("Signal after stage 2 of Cascaded Filters")
    subplot(5,1,5)
    plot((-512:511)*fs/64/40/1023, abs(fftshift(fft(filt_stage_2))))
    title("FFT of output from Cascaded Filters")
    xlabel("Freq")
end

% Measure PSD
figure()
periodogram(double(filt_stage_2),[],'onesided',512, 48.828e3)

% Compare ILA and Model Data
ila_filenames = ["ila_25kHz.csv", "ila_50kHz.csv"];
titles = ["25kHz ILA", "25kHz ILA vs Model", "50kHz ILA", "50kHz ILA vs
 Model"];
for i=1:length(ila_filenames)
    fid = fopen(ila_filenames(i));
    % the textscan is looking for the following fields, which matches the
    % format of my file
    C = textscan(fid, '%d%d%d%d%d%d%d%d%d%d%d', 'headerlines', 2, ...
        'delimiter', ',');
    fclose(fid);
    sig = C;
    sample_buff = C{1};
    sample_win  = C{2};
    trigger = C{3};
    ila_data = double(C{4});
    %recorded_sig_gen = circshift(recorded_sig_gen, -9);
    figure()
    subplot(2,1,1)
    plot(ila_data)
    title(titles((i-1)*2+1))
    subplot(2,1,2)
    plot(ila_data(1:8:end) - signals(2))
    title(titles((i-1)*2+2))
end

%! Lab question
disp(['Question 1: The attenuation at 22kHz is about -10dB in the ' ...
    'freqz plot and seems to be about 13dB smaller than the passband' ...
    'signal in the PSD plot.'])

% Quantative Analysis
disp(['The HW ILA data is getting very quantized when being reported' ...
    'from vivado. But looking at the signal size from the ILA graph' ...
    'and the signal from the filter model they matched fairly close.' ...
    'It is also like that since I needed to downsample the ILA to get' ...
    '1024 samples.'])
```

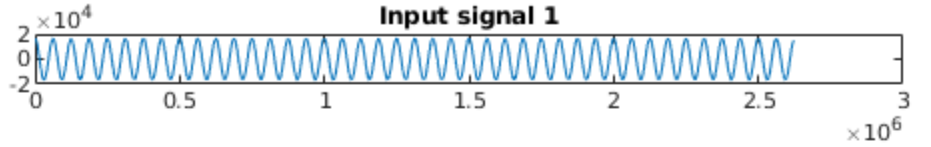
```
Warning: WARNING: Setting num_coeffs to: 176
Warning: WARNING: Setting num_coeffs to: 724
Question 1: The attenuation at 22kHz is about -10dB in the freqz plot and
 seems to be about 13dB smaller than the passbandsignal in the PSD plot.
```

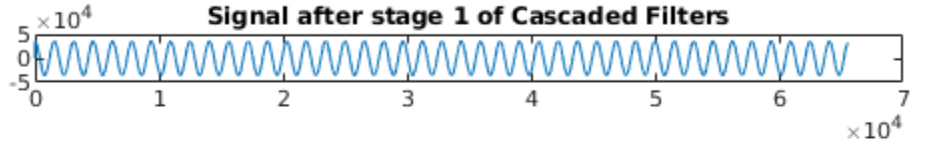
The HW ILA data is getting very quantized when being reportedfrom vivado. But looking at the signal size from the ILA graphand the signal from the filter model they matched fairly close.It is also like that since I needed to downsample the ILA to get1024 samples.

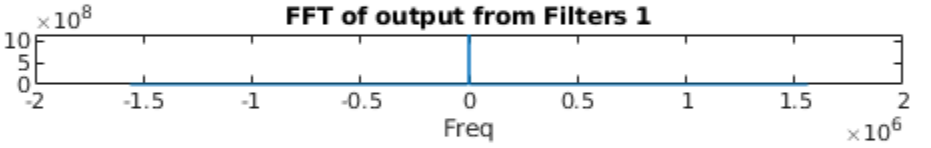


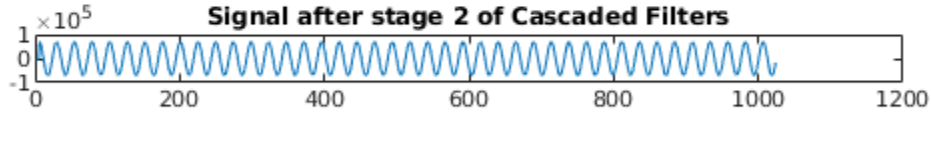

Frequency Response Filter 2
Magnitude (dB)
0
-50
-100
-150
0
0.2
0.4
0.6
0.8
1
Normalized Frequency (×π rad/sample)

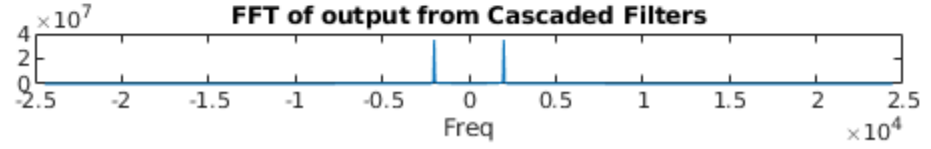

Phase
Phase (degrees)
0
-200
-400
-600
-800
-1000
0
0.2
0.4
0.6
0.8
1
Normalized Frequency (×π rad/sample)

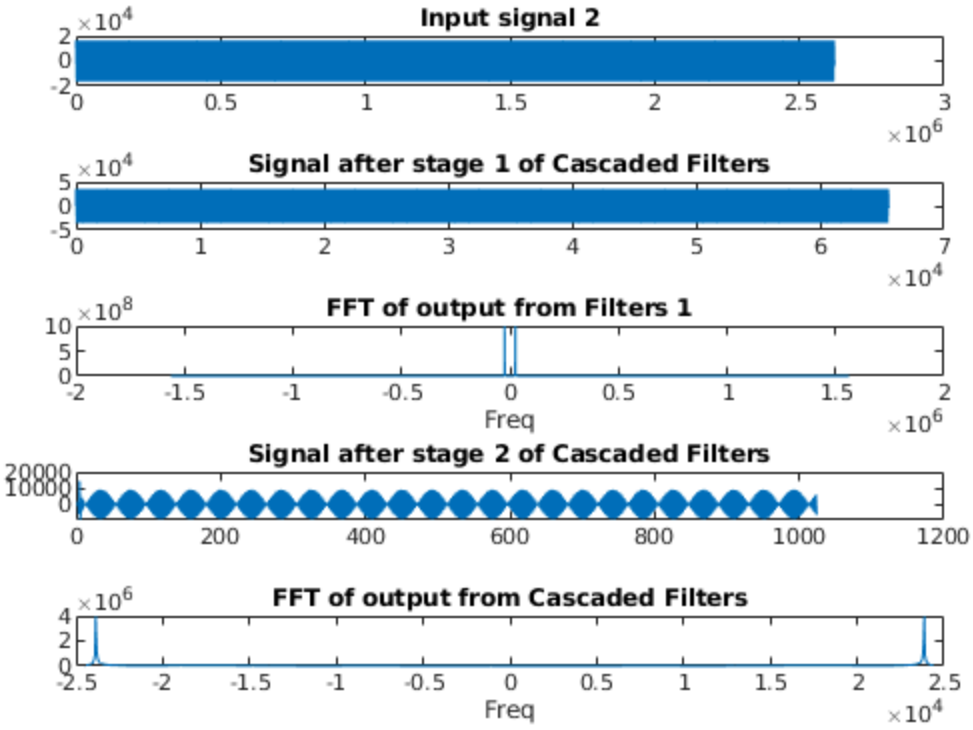

Input signal 1
×10^4
2
0
-2
0
0.5
1
1.5
2
2.5
3
×10^6

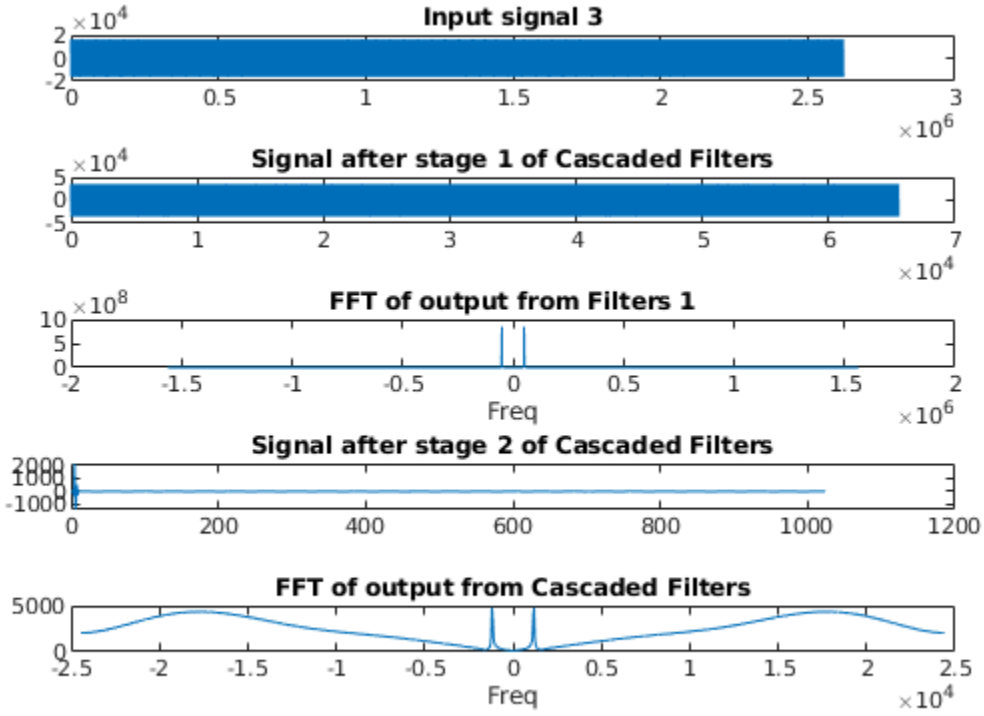

Signal after stage 1 of Cascaded Filters
×10^4
5
0
-5
0
1
2
3
4
5
6
7
×10^4

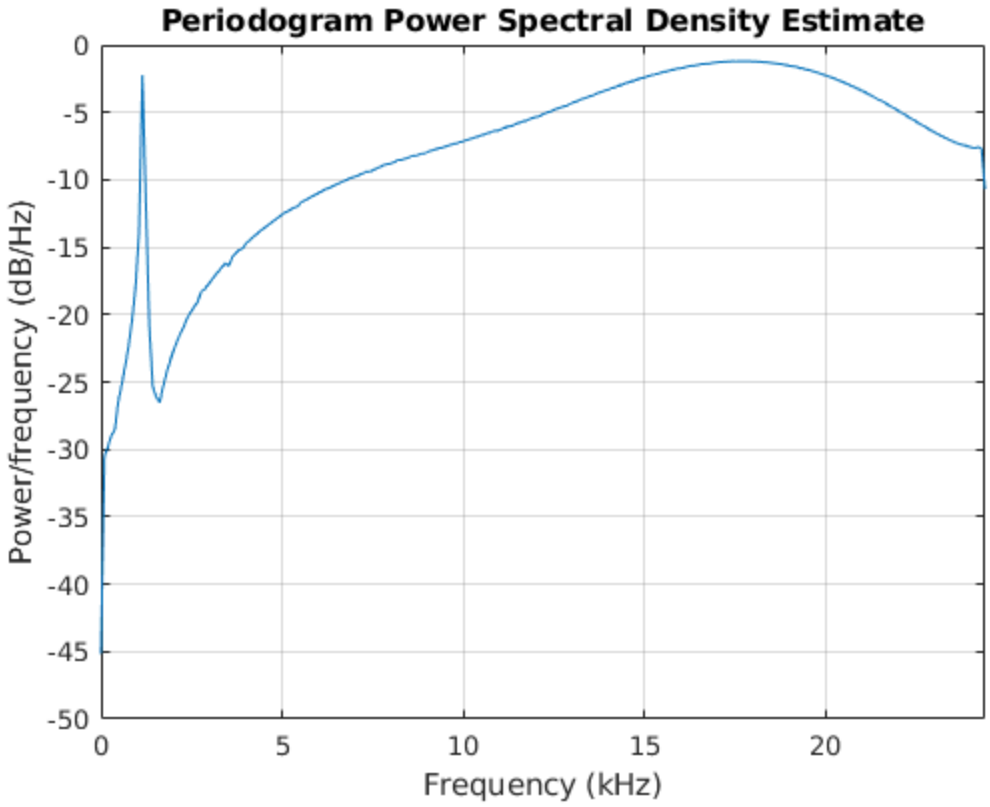

FFT of output from Filters 1
×10^8
10
5
0
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
Freq
×10^6

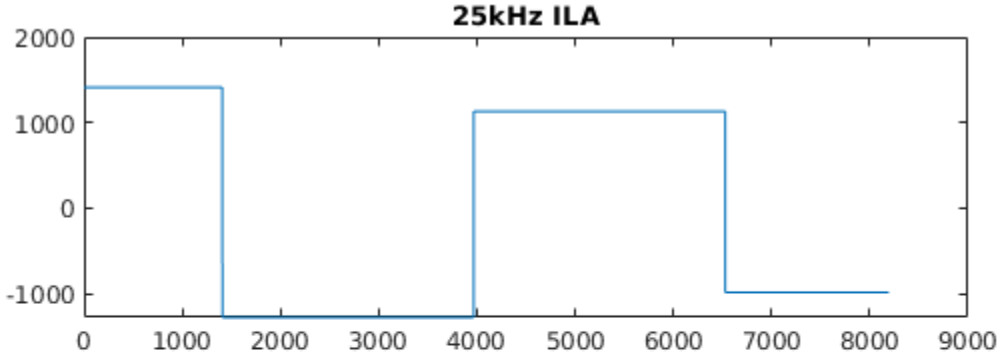

Signal after stage 2 of Cascaded Filters
×10^5
1
0
-1
0
200
400
600
800
1000
1200

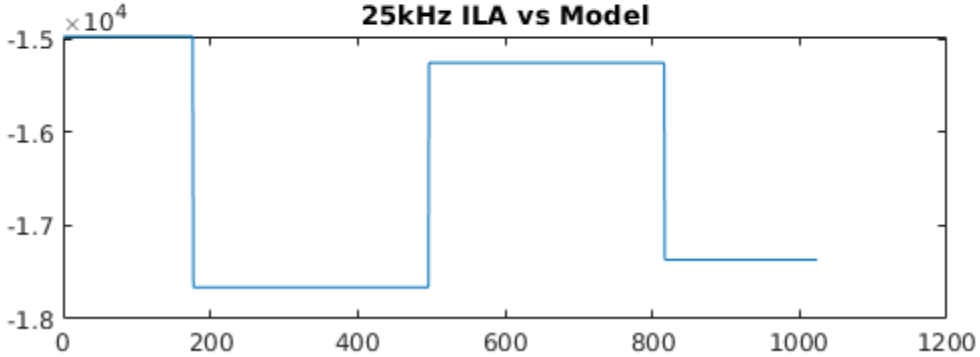

FFT of output from Cascaded Filters
×10^7
4
2
0
-2.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
Freq
×10^4

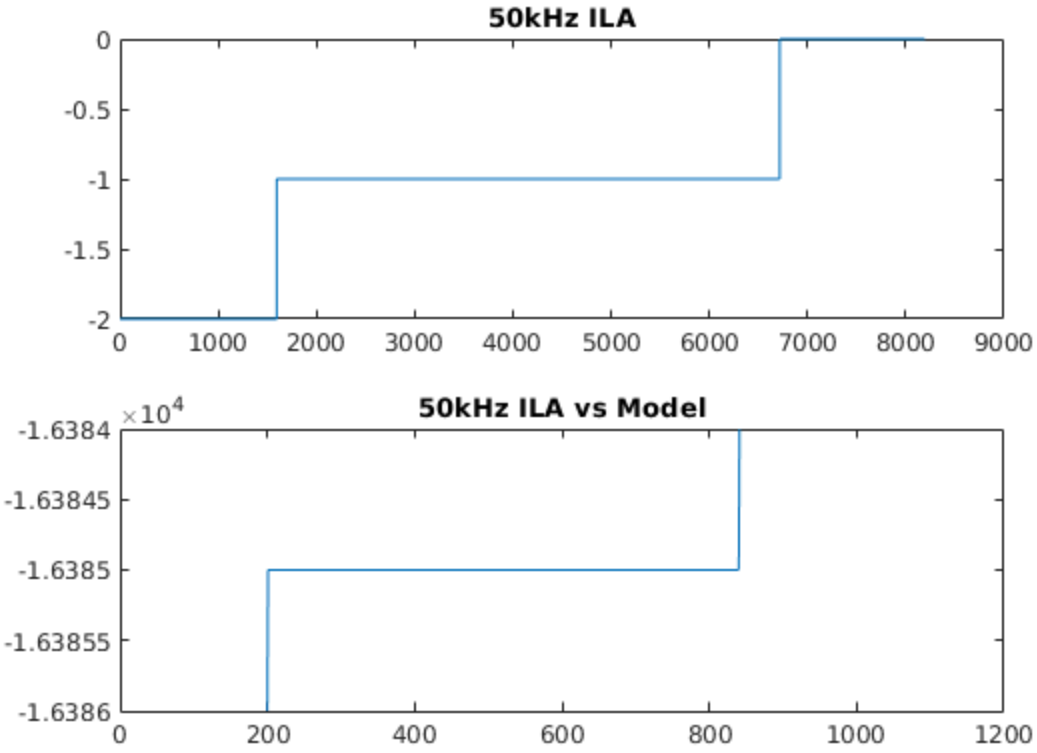

Input signal 2
Signal after stage 1 of Cascaded Filters
FFT of output from Filters 1
Freq
Signal after stage 2 of Cascaded Filters
FFT of output from Cascaded Filters
Freq


Input signal 3
Signal after stage 1 of Cascaded Filters
FFT of output from Filters 1
Freq
Signal after stage 2 of Cascaded Filters
FFT of output from Cascaded Filters
Freq


Periodogram Power Spectral Density Estimate
Power/frequency (dB/Hz)
Frequency (kHz)
0
-5
-10
-15
-20
-25
-30
-35
-40
-45
-50
0
5
10
15
20


25kHz ILA
2000
1000
0
-1000
0
1000
2000
3000
4000
5000
6000
7000
8000
9000


25kHz ILA vs Model
×10^4
-1.5
-1.6
-1.7
-1.8
0
200
400
600
800
1000
1200



Published with MATLAB® R2023a