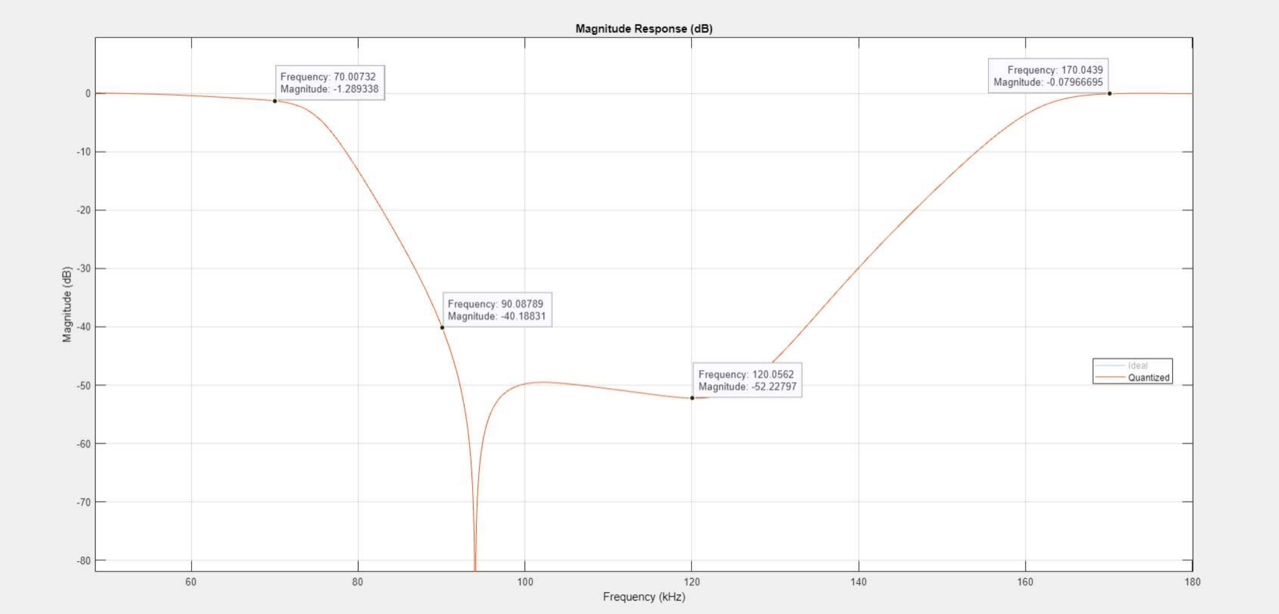
Justin La

ECE 212A Project

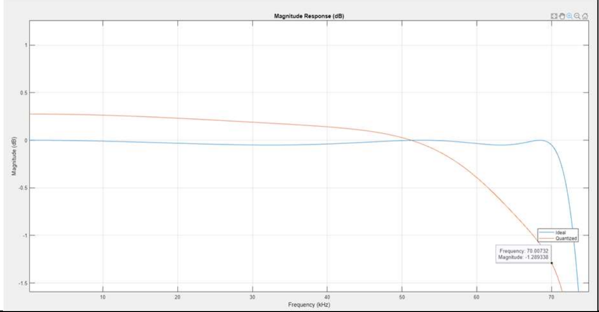
Table of Performance Numbers

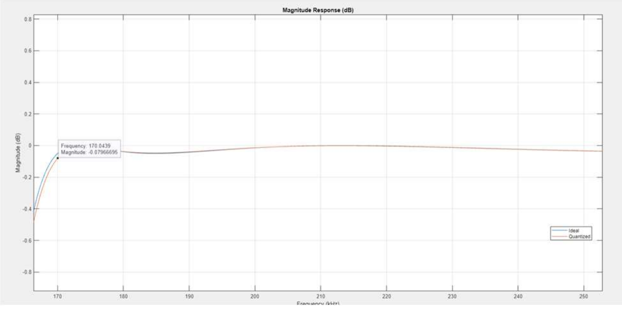
Bits	13
Filter Order	10
SNR (dB)	74.8583
Power Consumption	455P

Frequency Response and Attenuation:



Ripple:





Rationale for Design

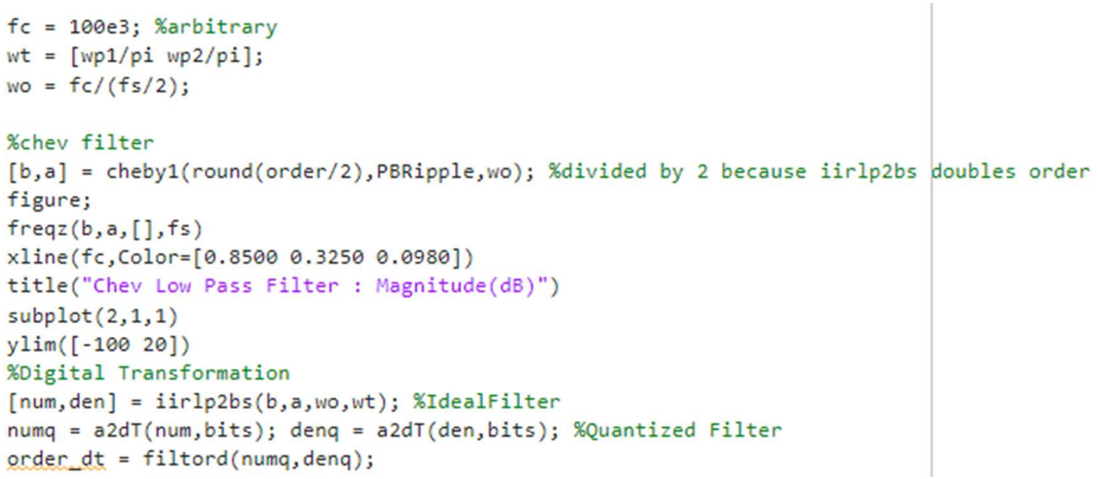
The design uses a Chebyshev type I filter as the baseline for the IIR filter. Since there is a requirement that allows passband ripple the Butterworth or the Chebyshev Type I filter can be used, but not the Chebyshev Type II filter nor the Elliptical Filter as they both have stop band ripples, which when digital transformed to a band stop will persist in the stop bands. The Butterworth filter is known to have a slower roll off frequency compared to the Chebyshev Type I filter, which means that to obtain the same stop band attenuation, a greater order filter would be required. To minimize resource utilization as much as possible the Chebyshev Type I filter is selected to reduce the order of the filter while still achieving the maximum passband ripple requirement.

To achieve the minimum output SNR of 72dB, with as little resources as possible, reducing the number of bits used on the filter was required. By using the input variance (power) of the input sinusoid at a -6dB full scale of the filter and the bits used to quantize the signal into the filter, just like an ADC, we can find the input SNR into the filter. To find the output SNR of the filter, the noise gain of the filter is found by calculating the output variance. The minimum number of bits required to satisfy the minimum output SNR is selected as anymore would increase resource utilization.

Summary of design approach

The general overview of the design approach is to find the analog Chebyshev Type I filter, digital transform it to a band stop filter, apply the quantization of the coefficients, then find the output variance for the SNR of filter.

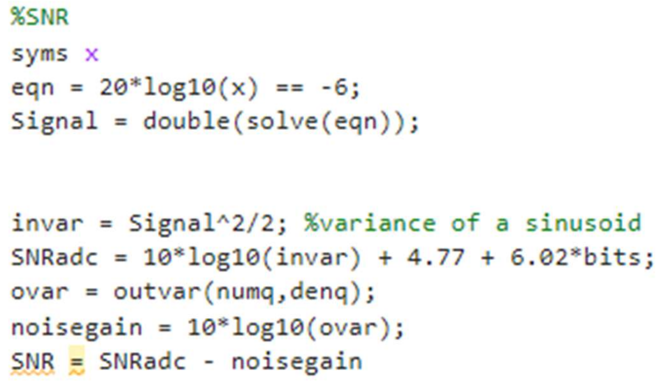
The analog Chebyshev Type I is created by using the cheby1 command in MATLAB. It accepts a passband ripple, but not a stop band attenuation. The attenuation isn’t defined in the function but can be determined in the frequency response and changed by increasing the order, which will create a sharper roll-off. The resulting outputs of the function are passed into the function iirlp2bs, which converts the low pass Chebyshev filter into a digital bandstop filter. It takes in the input of the passband frequency of the bandstop filter and the cutoff frequency of the Chebyshev low pass filter, which is arbitrary selected. After the numerator and denominator coefficients are outputted from the iirlp2bs function, the filter coefficients are quantized by the number of bits determined by the SNR output.



```
fc = 100e3; %arbitrary
wt = [wp1/pi wp2/pi];
wo = fc/(fs/2);

%chev filter
[b,a] = cheby1(round(order/2),PBRipple,wo); %divided by 2 because iirlp2bs doubles order
figure;
freqz(b,a,[],fs)
xline(fc,Color=[0.8500 0.3250 0.0980])
title("Chev Low Pass Filter : Magnitude(dB)")
subplot(2,1,1)
ylim([-100 20])
%Digital Transformation
[num,den] = iirlp2bs(b,a,wo,wt); %IdealFilter
numq = a2dT(num,bits); denq = a2dT(den,bits); %Quantized Filter
order_dt = filtord(numq,denq);
```

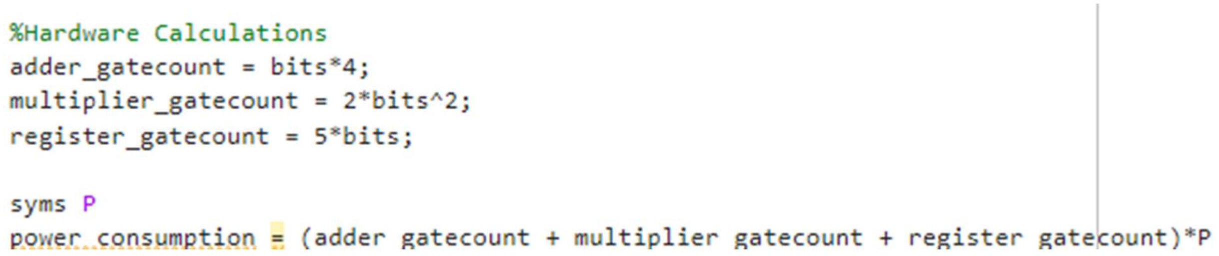
The SNR at the output of the filter is determined by SNR at the input of the filter minus the noise gain generated from the filter. The SNR of the input of the filter is variance of the sinusoid minus the quantization noise which is 4.77 + 6.02*bits. Using the number of bits used in the filter, we can calculate the input SNR. To find noise gain of the filter, a function from the textbook at section 9.5.6 Computation of Output Noise Variance Using MATLAB is used. Despite being labeled as Output Noise Variance, the calculation is actually the noise gain or the normalized output noise variance, as labeled in the textbook. The input SNR minus the noise gain from the textbook function provides the output SNR of the filter.



```
%SNR
syms x
eqn = 20*log10(x) == -6;
Signal = double(solve(eqn));


invar = Signal^2/2; %variance of a sinusoid
SNRadc = 10*log10(invar) + 4.77 + 6.02*bits;
ovar = outvar(numq,denq);
noisegain = 10*log10(ovar);
SNR = SNRadc - noisegain
```

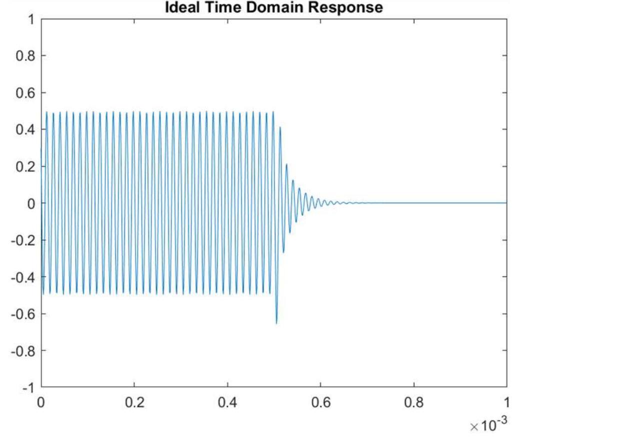
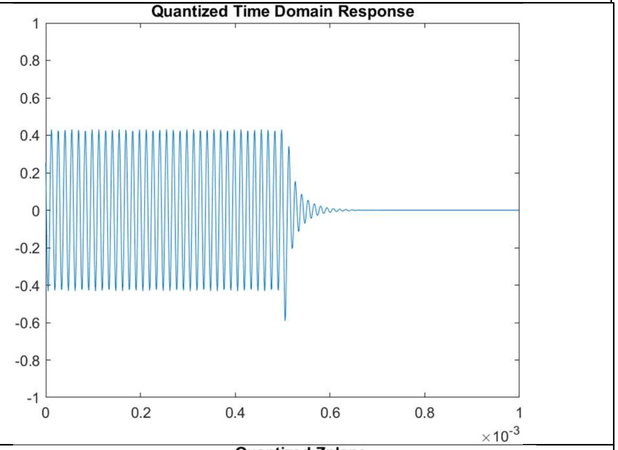
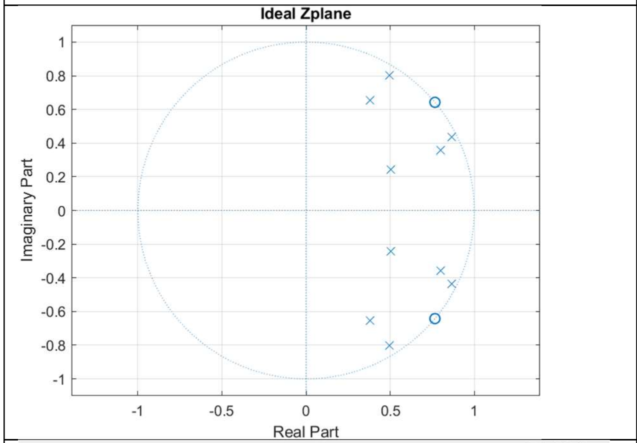
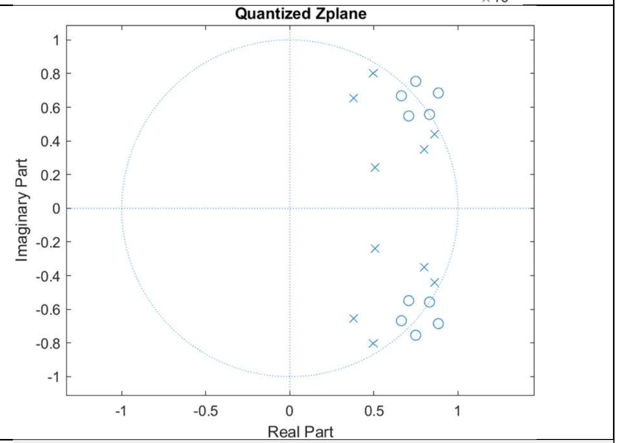
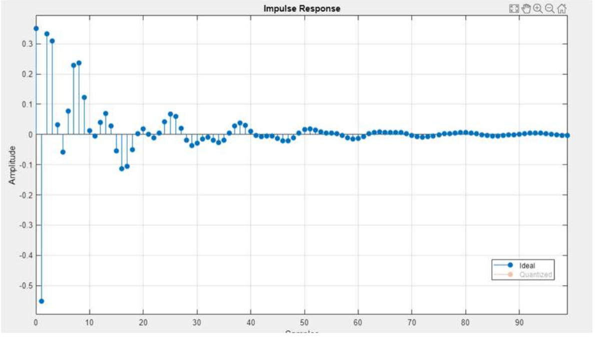
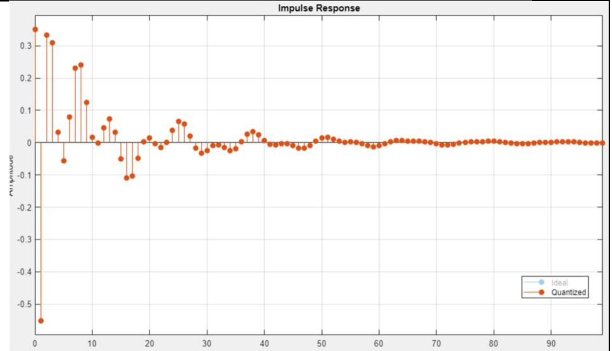
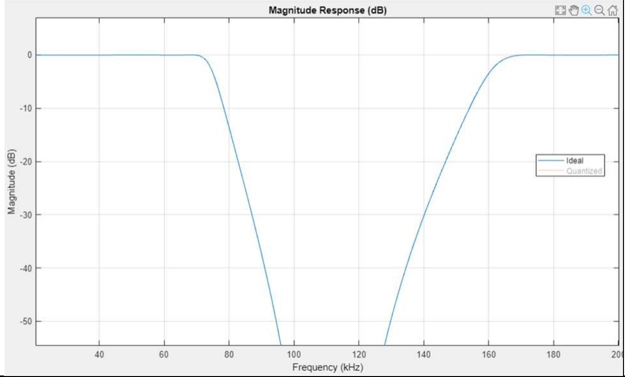
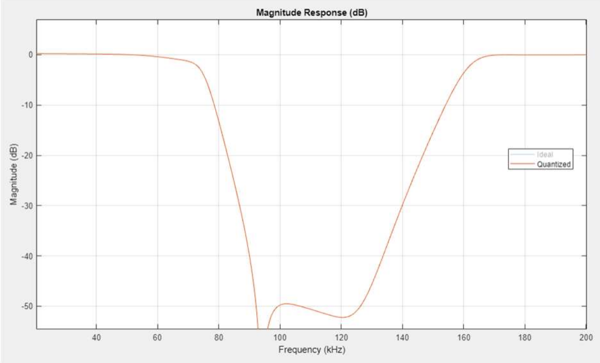
The hardware calculations are simply the formulas provided in the project requirements multiplied with a variable P, for the arbitrary power consumed per gate.



```
%Hardware Calculations
adder_gatecount = bits*4;
multiplier_gatecount = 2*bits^2;
register_gatecount = 5*bits;

syms P
power_consumption = (adder_gatecount + multiplier_gatecount + register_gatecount)*P
```

Other Results:

Ideal	Quantized
 Ideal Time Domain Response 1 0.8 0.6 0.4 0.2 0 -0.2 -0.4 -0.6 -0.8 -1 0 0.2 0.4 0.6 0.8 1 ×10^-3	 Quantized Time Domain Response 1 0.8 0.6 0.4 0.2 0 -0.2 -0.4 -0.6 -0.8 -1 0 0.2 0.4 0.6 0.8 1 ×10^-3
 Ideal Zplane Imaginary Part Real Part 1 0.8 0.6 0.4 0.2 0 -0.2 -0.4 -0.6 -0.8 -1 -1 -0.5 0 0.5 1	 Quantized Zplane Imaginary Part Real Part 1 0.8 0.6 0.4 0.2 0 -0.2 -0.4 -0.6 -0.8 -1 -1 -0.5 0 0.5 1
 Impulse Response Amplitude 0.3 0.2 0.1 0 -0.1 -0.2 -0.3 -0.4 -0.5 0 10 20 30 40 50 60 70 80 90 Ideal Quantized	 Impulse Response Amplitude 0.3 0.2 0.1 0 -0.1 -0.2 -0.3 -0.4 -0.5 0 10 20 30 40 50 60 70 80 90 Ideal Quantized
 Magnitude Response (dB) Magnitude (dB) 0 -10 -20 -30 -40 -50 40 60 80 100 120 140 160 180 200 Frequency (kHz) Ideal Quantized	 Magnitude Response (dB) Magnitude (dB) 0 -10 -20 -30 -40 -50 40 60 80 100 120 140 160 180 200 Frequency (kHz) Ideal Quantized