EECS 452 Midterm *Exam*

Winter 2012

Name: _____ unique name: _____

Sign the honor code:

I have neither given nor received aid on this exam nor observed anyone else doing so.

_____

Scores:

#	Points
Section I	/40
Section II	/30
Section III	/30
Total	**/100**

NOTES:

- Open book, open notes.
- There are **8** pages including this one.
- Calculators are allowed, but no smart phones, laptops, or other wireless devices are allowed.
- You have 120 minutes for the exam.
- **Be sure to show work and explain what you've done when asked to do so.** You will not receive partial credit without showing work.
- In the context of the exam you are to assume all signed numbers are two's complement numbers.

Section I -- Short answer 40 points

1) A 8 bit binary adder adds two values 0xA5 and 0x64 allowing overflow without saturation. What is the result (in Hexadecimal and Decimal) of the addition in the accumulator of this adder if the two input values are **[4]**

	Hex	Decimal
a) Unsigned binary numbers:	0x09	9
b) 2's complement binary numbers:	0x09	9
c) Unsigned Q4 binary numbers:	0x09	0.5625
d) Signed (2's complement) binary Q4 numbers	0x09	0.5625

2) Express the signed decimal fraction -0.625 in the following (Hex or binary) format **[2]**

a) 8 bit two's complement Q7: 0xB0

b) 8 bit two's complement Q3: 0xFB

3) The result of multiplying two 16 bit two's complement Q15 numbers together is a (put a check on the line next to the correct answer) **[4]**

a) 16 bit Q15 number: _____

b) 32 bit Q30 number: X

c) 32 bit Q15 number: _____

d) None of the above: _____

4) A 16 bit quantizer achieves a post-quantization SNR of 60dB. If you only needed attain a specification (spec) of 30dB post-quantization SNR could you reduce the number of bits in the quantizer to12 bits and still statisfy the spec? Why or why not? **[6]**

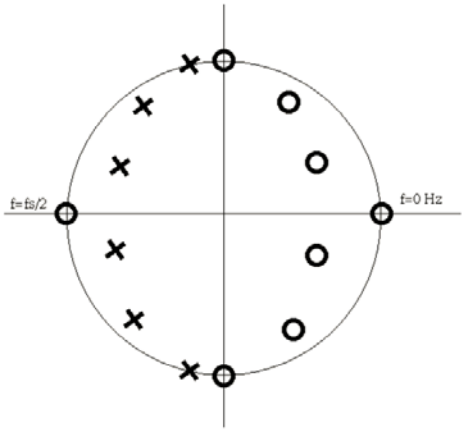
Yes, you can achieve the spec. From Lecture notes (Lec 6, p 26) you will recall that adding one additional bit to a quantizer increases the signal to quantization noise ratio (SQNR) by 6dB. Therefore the effect of reducing a quantizer by 4 bits is to decrease the

SQNR by 6*4 =24dB. Thus with 12 bits we would achieve a SQNR of 36dB which is above the spec of 30dB.

5) A digital spectrum analyzer of a continuous time signal uses discrete sampling and a 512 point FFT. If the frequency spacing in the FFT spectrum is to be at most 100Hz what is the maximum sampling frequency that the analyzer can handle? **[6]**

The frequency spacing in Hz corresponding to the indices of an N-point FFT is Fs/N. Therefore to attain a spacing of less than 100Hz with a 512 point FFT we require Fs be less than N*100 =51.2kHz.

6) A certain filter has the pole zero constellation shown below. Is this an FIR or an IIR filter? What is the filter order? Is is lowpass, highpass, bandpass, or none of the above? **[6]**



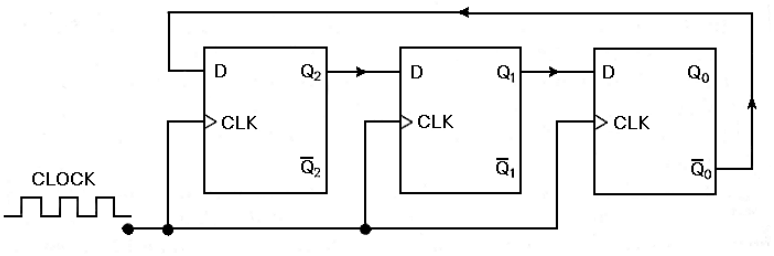
This pole constellation has 6 poles and 8 zeros so it must correspond to an IIR filter. The order of the filter is the number of poles and the number of zeros (either or both answers would be acceptable), i.e. order 6 or order 8. The filter is not lowpass nor is it highpass due to the zeros at 0degrees. 90degrees and 180degrees, corresponding to digital frequencies f=0,1/4,1/2 respectively. Either of the answers “bandpass” or “none of the above” would be acceptable here.

7) The ADC of a DSP system samples an input signal at 48,000 samples/sec sampling rate. The signal is processed by dropping every even sample and sending only the odd samples to the DAC. Assume that the DAC inserts 1/24 msecs between each sample. If the input were a sinusoid at frequency 8kHz what would be the output of the DAC assuming no quantization error and an ideal reconstruction filter? **[6]**

This question confused a many students. The key to solving this correctly is to remember that once the signal is digitized by the ADC the original sampling frequency is irrelevant, you just have an integer indexed sequence. The DAC synthesizes its own output period without accounting for the sampling rate at the ADC – ADC and DAC operate independently in a CODEC. So here’s the solution. As the 8kHz sinusoid is sampled at 48kHz by ADC there are

1/48 msecs between each sample of the original sinusoid. Therefore there are 1/24 msecs between each even sample of the original sinusoid. Therefore, since the DAC inserts 1/24 msecs between each of the samples, which correspond to the even samples since the odd ones have been dropped, the output is exactly the input, except sampled at half the sample rate, i.e. 24kHz instead of 48kHz. Therfore, as there is no aliasing of the input sinusoid at this sample rate, the output is a sinusoid at 8kHz – identical to the input except perhaps for some delay due to processing time.

8) The counter below uses 3 D flip flops. Draw the timing table indicating the states of the counter over the first 8 clock cycles (assume initial state is 000). What are the signal activity levels (α_1, α_2 and α_3) at the outputs Q2, Q1, Q0 of the counter? What is the total power dissipation of the counter assuming that the logic voltage is V_d=5V, the clock rate is f_{clk}=1MHz and the outputs of each flip flop see the same capacitance C=10pF? **[6]**



The timing table is drawn below (assuming it started from initial state 000). The cycle time of the counter, which is a 3 register Johnson counter, is 6 clock cycles so all that we need to do is count the number of transitions per counter cycle in order to find the signal activity levels. There are 2 transitions per counter cycle so therefore $\alpha_1 = \alpha_2 = \alpha_3 = 2/6 = 1/3$. The total power dissipation is therefore 3Pff where Pff is the power dissipated by a single flip flop. Using the formula we developed in class $Pff = C(V_d)^2(1+\alpha 3/2)f_{clk}$ which is $10*10^{-12}*(25)*(1+1/2)*10^6 = 375\mu W$. Therefore the total power dissipated is 3Pff=1.125mW.

Clk	Q2	Q1	Q0
1	1	0	0
2	1	1	0
3	1	1	1
4	0	1	1
5	0	0	1
6	0	0	0
7	1	0	0
8	1	1	0

Section II – longer answers 30 points

1) You wish to implement a filter on a 16 bit DSP chip that runs at 250MHz to do audio signal filtering. On this DSP chip the number of clock cycles required for a complex vector multiply of two 128-element vectors is 1500 cycles. The number of cycles for a 128 point complex FFT or IFFT (with scaling) is 2500 cycles. The number of cycles required for computing the output of a 16^{th} order IIR filter is 1000 cycles [10].

This question also seemed confuse many students. The question is similar to Q8 of Prelab 6 that asked you about throughput of FFT computations by considering the number of CPU cycles needed for FFT processing of audio stream. It then related the number of cycles to the maximum sampling rate that the CPU could handle if it were operating at a given clock frequency. This problem is a variation on Q8 of Prelab 6 that also asks you about memory requirements and IIR filters. There are many reasonable answers to the memory requirements part of a) and b), depending on how the IIR and FFT were implemented, and many students got the answer correct but with quantitatively different answers.

a) One implementation of the filter is to use a 16^{th} order IIR filter whose transfer function approximates the desired transfer function in both phase and magnitude. How many words of memory (16 bit words) are required to store the coefficients and the filter states in order implement this filter? What is the maximum sampling rate F_s such that the filter computation can be performed in real time, i.e. that the computation for the previous sample is completed before the next sample comes in? [4]

A general 16^{th} order IIR filter can be implemented as cascade of 8 biquad filters. Each biquad has up to 4 coefficients and 4 associated internal states. Without any additional structure, e.g. specification that certain coefficients are zero, and assuming that an internal state (data) and a coefficient each take up one 16 bit word of memory, this means that the number of words of memory per biquad is 4. Therefore, a 16^{th} order IIR filter requires 2*4*8=64 words of memory. In the case that the filter is all pole this drops to 32 words. The answer can change if there is specialized buffer for internal states or if there is pipelining or other architectural variants. The maximum sample rate is calculated from the given information that 1000 cycles are required for computation of each output and that the CPU clock rate is Fclk=250MHz. The sampling period Ts has to be greater than the time required to compute one IIR output: Ts>1000/Fclk or Fs=1/Ts<250kHz.

b) Another implementation of the filter is to use a DFT filter bank that works as follows. Perform a 128 point FFT on the 128 previous input samples. Then multiply these complex values by the desired transfer function. Finally, perform the IFFT on the result. How much memory is required to store the coefficients and the filter states in order implement this filter? What is the maximum sampling frequency that can be used so that the filter computation can be performed in real time, i.e., so that the filter output can be computed on the previous 128 block of samples before the next sample comes in? [4]

The 128 point DFT filterbank might need memory for the FFT and IFFT complex twiddle factors, the 128 complex transfer function values, and the 128 input and output data

points. If we do not use the fact that IFFT and FFT coefficients are complex conjugates of each other, we would need to assign 2*128 complex words for these coefficients, 128 complex words for the transfer function, and 2*128 words for the input and output data. This works out to 1024 16 bit words (16 bits for real and imaginary parts) assuming that the input and output data are real valued. As for the maximum sampling rate, the sampling period Ts has to be greater than the time required to compute one output which, using the given information, will be 2500 CPU cycles for computing the FFT, 1500 CPU cycles for the complex multiply of the FFT output and the transfer function, and 2500 CPU cycles for computing the IFFT, or a total of 6500 CPU cycles. Hence Ts>6500/Fclk or Fs=1/Ts<38.4kHz.

c) If you had a fast enough clock rate to handle all the computations in real time and enough memory storage capacity, are there any advantages to using the DFT filterbank as compared to the IIR filter? [2]
There are several advantages to the DFT filterbank solution, despite the increase memory and computation time requirements. The principal advantage is that you can implement virtually any transfer function, not just the IIR ratio of polynomials in Z. For example you can implement a TF with magnitude $|H(f)|= \exp(-f^2/a)$, where a is positive – this is called a Gabor filter, which is not a ratio of polynomials. Thus you can directly synthesize any phase and magnitude characteristic you like, and in particular linear phase.

2) Consider the discrete time signal $x[n]=0.5*\cos(2\pi f_1 n)+\sin(2\pi f_2 n)$, n=1,…,8. [10]

a) What is the DFT of x[n] if $f_1=1/4$ and $f_2=3/8$? Carefully plot the magnitude and phase being sure to label and quantify the ranges of the axes. [5]
We compute the N-point DFT for N=8 as follows. Rewrite x[n] as

$$x[n]=0.5*\cos(2\pi f_1 n)+\cos(2\pi f_2 n-\pi/2),$$

As f_1 and f_2 are 2/8 and 3/8, respectively, they are integral multiples of 1/N and we know that the DFT will consist of only four non-zero components located at DFT indices 2,3,N-3,N-2. and with magnitude and phases listed below (N=8)

DFT Frequency	Magnitude	Phase
2/N	N/4	0
3/N	N/2	$-\pi/2$
5/N	N/2	$\pi/2$
6/N	N/4	0

b) If x[n] corresponded to a continuous-time signal x(t) sampled at 48kHz what frequencies (Hz) are present in the original signal x(t)? [3]

Here you need to use the simple relation that the Hz frequency F associated with a digital frequency f is of the form: F=f*Fs, where Fs is the sampling frequency. Therefore F1=1/4*48kHz=12kHz and F2=3/8*48kHz=18kHz.

.

c) Is there spectral leakage in this example? Why or why not? [2]

There is no leakage since the frequencies F1 and F2 are integral multiples of Fs/N

3) The input output relation of a particular linear time invariant digital filter is: [10]

$$y[n] = \sum_{k=0}^{N} (-1)^k x[n-k]$$

a) What is the transfer function H(z) associated with the filter? Plot poles-zeros constellation of H(z) in the complex plane. [3]
The impulse response of the filter is $h[k] = (-1)^k$ for k=0,1,…N and zero otherwise. The transfer function is the Z-transform $H(z) = 1 - z^{-1} + z^{-2} - \cdots + (-z^{-1})^N$.
Let's specialize to the case N=1 for simplicity. Then $H(z) = 1 - z^{-1} = z^{-1}(z-1)$ which has a single root at z=1. Thus, on the complex Z plane there is a zero at z=1 and a pole at z=0. In the general case of N>1, there are N zeros on the unit circle at angles k/(N+1) for k=0,1, ...,N. Note that if you did this problem by using the geometric series to obtain $H(z) = (1 - (-z^{-1})^{N+1})/(1 + z^{-1})$ (which is correct), you can fool yourself into thinking that there is a pole at z = -1. However, since $(1 - (-z^{-1})^{N+1}) = (1 + z^{-1})(1 - z^{-1} + z^{-2} - \cdots + (-z^{-1})^N$ there is actually a pole-zero cancelation, so there is actually no pole at z=-1.

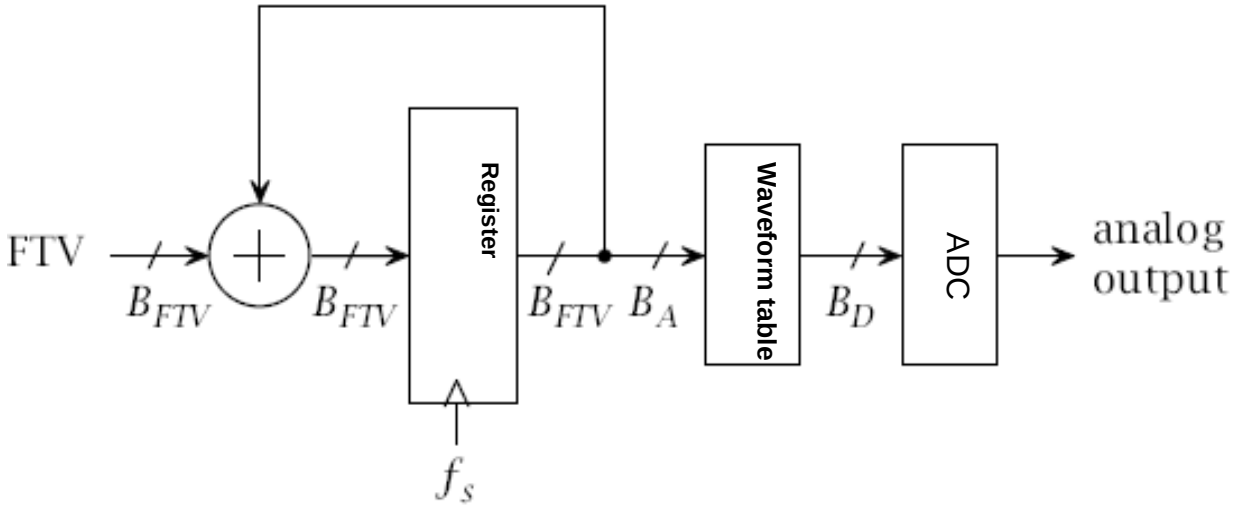
b) Compute and plot the magnitude and phase frequency response as a function of the digital frequency variable f of the DTFT. [3]
Use the fact that the impulse response of the filter can be written as $h[k] = (-1)^k = \exp(-j\pi k)$. Then the DTFT of h is $\mathrm{H}(e^{j2\pi f}) = \sum_{\{n=0\}}^{N}(-1)^n e^{j2\pi fn} = \sum_{\{n=0\}}^{N} e^{j2\pi(f-\frac{1}{2})n}$.
Now, if you recall Lecture 8 p. 7 where we introduced the moving average filter, the DTFT above is identical to the DTFT of the moving average filter except that it is shifted by frequency ½. Therefore, from the formula in the lecture notes, we immediately obtain the form for the transfer function (as usual H(f) is shorthand for $\mathrm{H}(e^{j2\pi f})$)

$$\mathrm{H(f)} = (N+1)^{-1}\left\{\exp(^{-}j\pi N\left(f-\frac{1}{2}\right))\right\}\sin\left(\pi(N+1)\left(f-\frac{1}{2}\right)\right)/\sin(\pi(f-\frac{1}{2}))$$

c) Is this filter linear phase? What is its group delay? [2]
This filter is linear phase and its group delay is N/2.

d) Is this filter LP, BP, HP or none of the above? [2]
The passband of the magnitude transfer function depends on N. If N>1 then it is a highpass filter centered at frequency f=1/2. If N=0 it is an all-pass filter and if N=1 it has a broad passband that extends from f=1/2 to near f=0.

Section III Longer answers 30 points

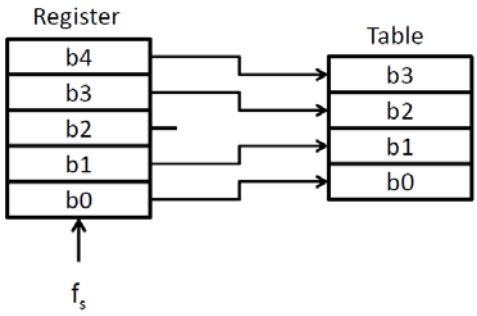


1) A waveform synthesizer uses the DDS system above to generate a periodic waveform using a 4 bit waveform table containing 16 samples of a single period of the linear ramp waveform s[n]=n, n=0,1,2 … 15. **[15]**

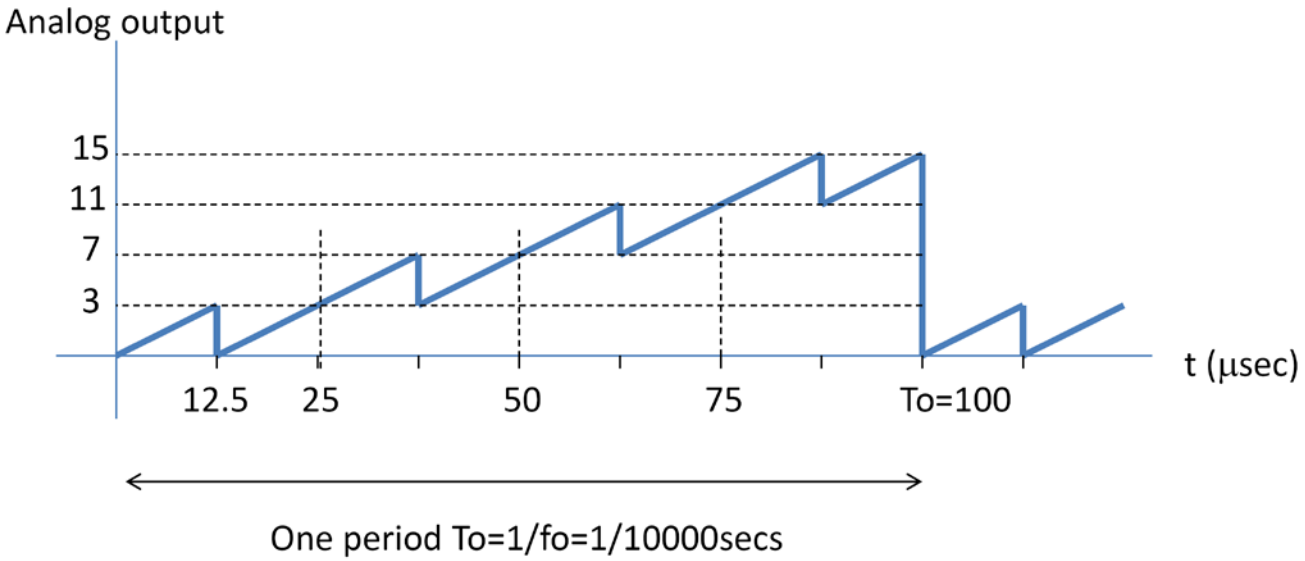
   a) Assume FTV=1 and $B_A = B_{FTV}$. What should the clock frequency f_s (Hz) be in order for the output waveform frequency to be equal to 10kHz? Plot three or four periods of the analog output over time. Be careful to label your axes and indicate the amplitude and period of the waveform on your plot.**[7]**
   The output frequency is equal to f_0=FTV $f_s/2^{BFTV}$. As FTV=1 and BFTV=4, f_s must be $f_0 2^{BFTV}$=10*16kHz=160kHz. The analog output will be a sawtooth waveform with period equal to 1/10,000 or 0.1msec.

   b) Now we modify the DDS by increasing B_{FTV} and connected the table and to the register in the following non-standard manner. We set $B_{FTV}=B_A+1$ and make the upper 2 MSB's of the register address the 2 MSB's of the waveform table and the lower 2 LSB's of the register address the 2 LSB's of the waveform table (see diagram below). What is now the output waveform frequency in terms of the clock rate f_s (Hz), B_{FTV} and B_A? What should the clock frequency be for the waveform frequency to be 10kHz? Plot a single period of the output assuming the linear ramp waveform you assumed in part a). **[8]**
   This part of the question was problematic for many students. The key to seeing the solution is to draw a timing diagram, which many of the successful students did. You should note that because of the fact that the middle bit is not addressing the table, there will be a single repetition of the first 4 values in the table (the LSB's b0 and b1) before each increment of the MSB's b2 and b3 in the table. As there are four increments of the MSB's before the counter cycles back to 0, this single repeated set of values will occur 4 times in a cycle. The output waveform frequency is given by the same formula as used in part a) except that now BFTV=5 bits instead of 4 (the time to cycle the counter is now 2^5). Therefore to attain an

output frequency of 10kHz we require 2 times the frequency we had before, that is fs=320kHz.



The plot of the waveform using this value of fs is drawn below.



2) Consider the digital filter below that produces an output y[n] given input x[n]. **[15]**

a) Is this filter FIR or IIR? What is the filter order?

It is IIR since it is composed of recursive filters arranged in parallel: a recursive first order IIR filter at top, a biquad IIR filter in the middle, and a biquad IIR filter at the bottom. The filter is a Direct Form II implemention of the transfer function (http://www.dsprelated.com/dspbooks/filters/Parallel_Second_Order_Signal_Flow.html) y[n]=x[n]+0.5^3x[n-3]-0.9^5y[n-5] as a parallel bank of real first and second order digital filter sections.

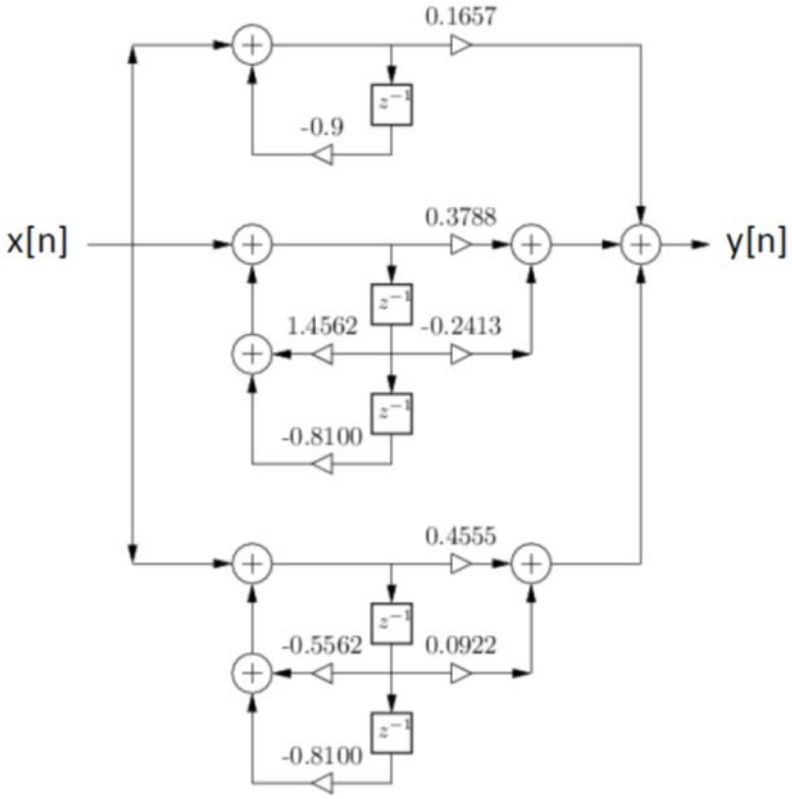
.

b) Assuming that that the input is scaled appropriately is this filter implementable in 16 bit Q(14) arithmetic? Could it be implemented in 16 bit Q(15) with a small modification? If so how would you modify it?

The filter can be implemented in Q(14) since the maximum magnitude coefficient value is less than 2. It could be modified by dividing by the coefficients 1.4562 and -0.8100 by 2 prior to summation and then multiplying them by 2 after summation (assuming that the signal was scaled so that it never caused overflow after multiplication by 2 at this internal state.

c) Can this filter be implemented with biquads? How many biquads would be necessary?

The filter as given is implemented with 2 biquads and one single stage, as shown in the diagram. Unlike in the filters you constructed in lab 5, this filter is in parallel form. To implement the finite precision analysis and overflow protection procedure you would need to convert to series cascade form.


0.1657
-0.9
x[n]
0.3788
y[n]
1.4562
-0.2413
-0.8100
0.4555
-0.5562
0.0922
-0.8100