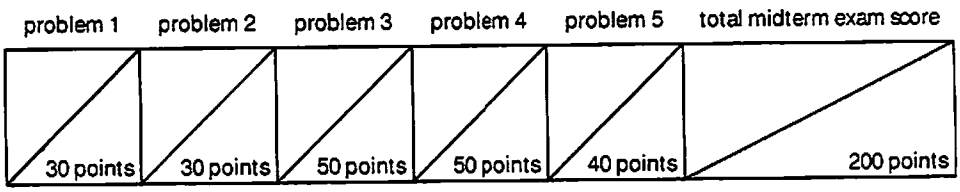
ECE4703 Midterm Exam

Your Name: SOLUTION Your box #: _____

November 19, 2009

Tips:

- Look over all of the questions before starting.
- Budget your time to allow yourself enough time to work on each question.
- Write neatly and show your work!
- This exam is worth a total of 200 points.
- Attach your "cheat sheet" to the exam when you hand it in.



1. 30 points total.

(a) 15 points. Suppose we convert an analog signal $x_1(t)$ from continuous-time to discrete-time by sampling at frequency f_s and then convert this signal back to continuous-time using an ideal reconstruction filter, i.e.,

$$x_1(t) \overset{\text{ideal sample at } f_s}{\rightarrow} x[k] \overset{\text{ideal reconstruction}}{\rightarrow} x_2(t).$$

Is it possible for $x_2(t)$ to be equal to $x_1(t)$? If not, why not? If it is possible, then under what conditions?

Yes. $x_2(t) = x_1(t)$ if the maximum frequency content of $x_1(t) \leq f_s/2$.

In this case

$$x_2(t) = \sum_{k=-\infty}^{\infty} x[k] \frac{\sin(\pi(t - kT_s)/T_s)}{\pi(t - kT_s)/T_s} = x_1(t)$$

(b) 15 points. Suppose we now convert an analog signal from continuous-time to discrete-time by sampling at frequency f_s, then from continuous-valued to discrete-valued using an n-bit quantizer, and then finally back to continuous-time using an ideal reconstruction filter, i.e.,

$$x_1(t) \overset{\text{ideal sample at } f_s}{\rightarrow} x[k] \overset{\text{ideal } n\text{-bit quantize}}{\rightarrow} x_q[k] \overset{\text{ideal reconstruction}}{\rightarrow} x_2(t).$$

Is it possible for $x_2(t)$ to be equal to $x_1(t)$? If not, why not? If it is possible, then under what conditions?

No. Quantization, unlike sampling, produces irreversible distortion in the signal.

$$x_2(t) = \sum_{k=-\infty}^{\infty} x_q[k] \frac{\sin(\,)}{(\,)} \neq \sum_{k=-\infty}^{\infty} x[k] \frac{\sin(\,)}{(\,)} = x_1(t).$$

2. 30 points. After graduating from WPI, your considerable expertise in real-time DSP lands you a job at a company that produces two-way, battery powered, handheld digital radios for police and fire department use. These radios are currently built with all-analog electronics but your company wants to develop new DSP-based design. Do you recommend going with a fixed-point or floating-point DSP? Does your recommendation have any disadvantages? Explain your answer.

Fixed point is probably the better choice here.

+ lower power
+ faster operation (better for radio signals)
+ cheaper

The main disadvantage is that it is going to be more difficult to code.

3. 50 points total. Suppose you have a C program that adds up N 16-bit signed integers and stores the result in a 32-bit signed integer. Your code looks like this:

```
#define N (something)
short a[N];
int b = 0;
int n;
//
// some code in here sets all the values of a
//
for (n=0;n<N;n++)
  b += a[n];
```

(a) 20 points. What is the largest value that N can be such that you can guarantee there will be no overflow?

$\log_2(N) \leq 32-16 = 16$

$\Rightarrow N \leq 2^{16}$ The maximum positive value of b is then $2^{16} \cdot 32767$ which just fits in the int datatype w/o overflow.

(The maximum negative value is $-2^{16} \cdot 32768$ which also fits in an int datatype w/o overflow.

(b) 30 points. Suppose $N = 100000$. Show that overflow could occur in this case and fix the code (without changing N) to avoid overflow while maintaining maximum precision. Explain your answer.

The largest positive value in this case is $100000 * 32767 = 3.2767 \times 10^9 > 2.114748 \times 10^9$ (largest positive value in int datatype).

Hence overflow could occur.

We can avoid this by changing the line

```
b += a[n];
```

with

```
b += a[n] << 1;
```

Now the largest possible positive sum is $100000 * (2^{14}-1) = 1.6383 \times 10^9$

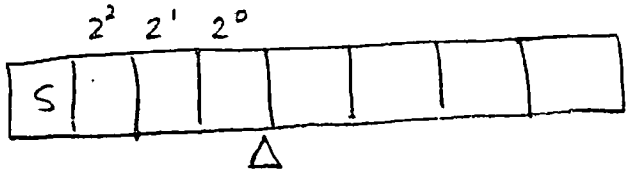
which is smaller than the largest positive int value. The same is true for the largest possible negative sum.

4. 50 points total. You are given the following infinite-precision FIR filter coefficients.

$$b = [1.234 \quad 7.654 \quad -\pi]$$

(a) 20 points. Suppose you are required to store these filter coefficients in a signed 8-bit fixed-point data type. Determine the optimum number of fractional bits to use in your fixed-point representation of the filter coefficients. Show your reasoning.

The second coefficient is the largest and requires 3 non-frac bits. Hence we can have 4 frac bits



If we went with >4 frac bits, we would have overflow in one or more coefficients.

If we went with <4 frac bits, we would not have as much precision.

(b) 30 points. Using your fractional bit specification from part (a), fill out the following table. Use the symbol Δ to show the binary decimal point.

Coefficient Value	Quantized Value (decimal)	Quantized value (binary)	Quantization Error
1.234	1.25	$0001_\Delta 0100$	0.016
7.654	7.625	$0111_\Delta 1010$	0.029
$-\pi$	-3.125	$1100_\Delta 1110$	≈ 0.0166

via 2's complement

$3.125 = 0011_\Delta 0010$

-3.125

$$\begin{array}{r} 1100_\Delta 1101 \\ +\ 0000_\Delta 0001 \\ \hline 1100_\Delta 1110 \end{array}$$

5. 40 points total. In Laboratory Assignment 3, you saw that quantizing the filter coefficients of your DF-II single-section IIR filter led to either an unstable filter or a filter with poor frequency response with respect to your original floating point IIR filter. You also saw that quantizing the filter coefficients of your DF-II *second-order-sections* IIR filter (with the same number of bits for the quantized coefficients as the single-section filter) led to a much more accurate frequency response. Explain why fixed-point DF-II second-order-sections IIR filters tend to give more accurate results than fixed-point DF-II single-section IIR filters. Feel free to use a numerical example to make your arguments clear.

DF-II SOS Filters tend to be less sensitive to coefficient quantization because the dynamic range of the coefficients in each SOS is smaller.

A SOS has the form

$$H_i(z) = \frac{(1 - q_i z^{-1})(1 - q_i^* z^{-1})}{(1 - p_i z^{-1})(1 - p_i^* z^{-1})}$$

and the overall transfer function looks like

$$H(z) = H_1(z) \cdots H_N(z)$$

Suppose $p_i = 0.9$ for all i and $N = 8$ sections

The denominator of $H(z)$ would look like

$$H(z) = \frac{\text{something}}{1 + a_1 z^{-1} + \cdots + a_{16} 2^{-16}}$$

where $a_1 = -7.2$ and $a_{16} = 0.4305$

If we use 8-bit coefficients, we need

1 sign bit, 3 non frac bits, $\Rightarrow$ maximum 4 frac bits

to quantize the denominator coefficients of the single-section filter

For an SOS, however, we have $H_i(z) = \frac{}{1 - 1.8z^{-1} + 0.81z^{-2}}$

Now we need 1 sign bit, 1 non frac bit, $\Rightarrow$ maximum 6 frac bits.

Also note the smallest coefficient is about twice as big as the smallest coefficient for the single-section filter. This will give us a much more accurate response