# ECE 3210 Midterm 2

*Week of: November 3, 2020*

Instructor:
Eric Gibbons
ericgibbons@weber.edu
801-626-6861

You have 15 hours for 5 problems.

- Show enough (neat) work in the clear spaces on this exam to convince us that you derived, not guessed, your answers.
- Put your final answers in the boxes at the bottom of the page.

This test is open notes, open book, and open calculator/Python. Consulting with any third party is considered cheating.

| Problem | Score | Possible Points |
|---|---|---|
| 1 | | 20 |
| 2 | | 20 |
| 3 | | 20 |
| 4 | | 20 |
| 5 | | 20 |
| **Total score** | | 100 |

## 1 Short answer

(a) Suppose we have two discrete signals $a_k$ and $b_k$ which have sequence lengths of 20 and 34, respectively. If we are interested in computing linear convolution using the radix-2 decimation-in-time FFTs (like we did in lab), how much zero padding do we need to do to $a_k$ and $b_k$? (I.e., how many zeros do we need to add to $a_k$ and how many zeros do we need to add to $b_k$?) f

$n_{\text{zeros}, a_k} =$ $n_{\text{zeros}, b_k} =$

(b) If the bandwidth of $f_1(t)$ is $B_1 = 250\,\text{Hz}$, and the bandwidth of $f_2(t)$ is $B_2 = 750\,\text{Hz}$. Determine the Nyquist sampling rates for signals $f_1(t)f_2(t)$ and $f_1(t) * f_1(t)$.

$f_1(t)f_2(t)$: $f_1(t) * f_1(t)$:

(c) Consider the second order transfer function $T(s) = \frac{10000}{s^2+100s+10000}$. Find $t_p$, $t_r$, $t_s$, and the percent overshoot.

| | | | |
|---|---|---|---|
| $t_p =$ | $t_r =$ | $t_s =$ | P.O. = |

(d) Do you prefer the Laplace transform or the Fourier transform? (Circle one.)

| | |
|---|---|
| Laplace | Fourier |

## 2 Differential equations

Given the differential equation

$$(D^2 + 6D + 25)y(t) = (D + 1)f(t)$$

where $y(0^-) = 1$, $y'(0^-) = 2$, and $f(t) = u(t)$, answer the following questions.

(a) Find the zero-input response.

$y_{\text{zi}}(t) =$

(b) Find the zero-state response.

$y_{\text{zs}}(t) =$

(c) Find the total response.

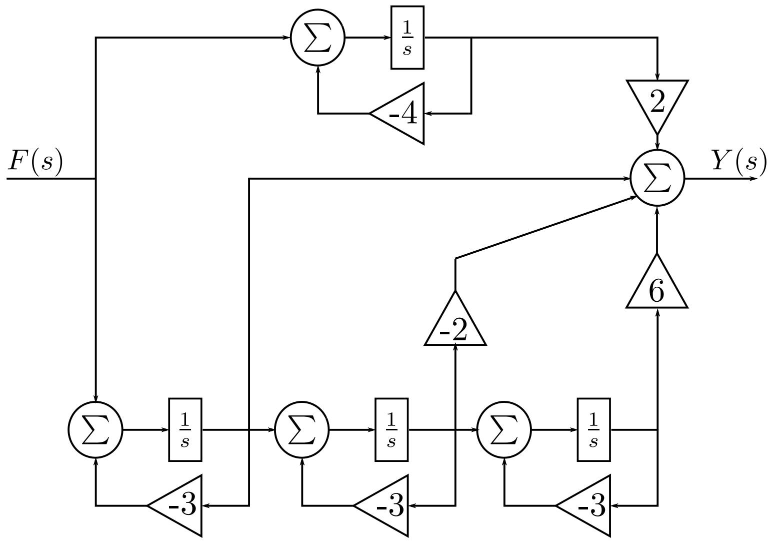
$y_{\text{tot}}(t) =$

## 3 System realization

(a) Draw the canonical block diagram realization for the transfer function

$$H(s) = \frac{2s^3 + 2}{(s+2)^3}$$

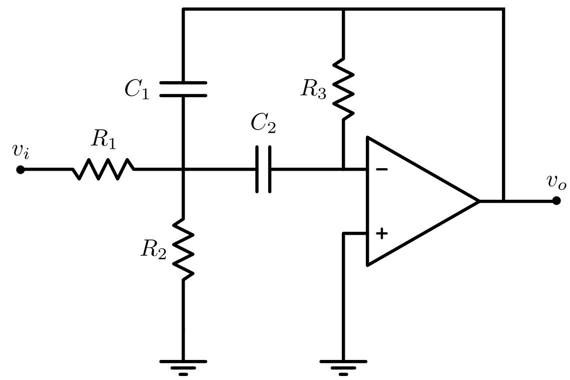
(b) Given the following block diagram for a LTIC system find the impulse response $h(t)$.




$h(t) =$

## 4 Filtering

(a) One common active filter topology is given below.




Like the filter we saw in Lab 7, this comes with a generic transfer function

$$H(s) = \frac{2\xi\omega_0 K s}{s^2 + 2\xi\omega_0 s + \omega_0^2}.$$

If we are given component values $R_1 = 5.1\,\mathrm{k\Omega}$, $R_2 = 5.1\,\mathrm{k\Omega}$, $R_3 = 15\,\mathrm{k\Omega}$, $C_1 = 0.015\,\mu\mathrm{F}$, and $C_2 = 0.047\,\mu\mathrm{F}$, find the values of $K$, $\xi$, and $f_0$. (Note that $\omega_0 = 2\pi f_0$.)

Your work, continued...

| | | |
|---|---|---|
| $K =$ | $\xi =$ | $f_0 =$ |

(b) Similar to what we did in Lab 7, plot the frequency response of this filter. In Python, plot $|H|$ as a function of frequency $f$ and also plot $\angle H$ as a function of frequency $f$ on separate plots. In these plots, let $f \in [10\,\mathrm{Hz},\, 100\,\mathrm{kHz}]$. Write your code in Python script called `frequency_response.py` and submit it to Canvas.

(c) Lastly, circle what type of filter this is: high-pass, low-pass, band-pass, or notch.

| | | | |
|---|---|---|---|
| high-pass | low-pass | band-pass | notch |

## 5 Circular convolution

In lab wrote our own *linear* convolution function using a direct time-domain approach as well as a FFT approach. Here, we will take a look at the other flavor of discrete convolution: circular convolution.

(a) Given discrete sequences $f = \{1, 1, 1, 1, 0, 0, 0\}$ and $g = \{1, 2, 3, 4, 3, 2, 1\}$ find the circular convolution

$$y_k = f_k \circledast g_k.$$

$y_k =$

(b) In a Python file called `myconv.py` (it can be part of your `myconv.py` from Lab 6 or an entirely new file), find write a function `circ_conv(f, g)` where $f$ and $g$ are both 1D, real-valued `numpy` arrays with the same length. The length of `f` and `g` can be arbitrary but must be the same for both arrays. The output of `circ_conv(f, g)` is also a 1D array that is real-valued. You can check your work here by using the FFT approach. You are welcome to use the `np.fft` functions to check your work. However, you may not use any of the functions in the `scipy` library or the `np.convolve()` function in your `myconv.py` function. Please submit your `myconv.py` file to Canvas.