

# LAB 2: PRELAB (PART 2)\*

Thomas Shen  
Douglas L. Jones

Based on *Multirate Filtering: Filter-Design Exercise in MATLAB*<sup>†</sup> by

Douglas L. Jones  
Swaroop Appadwedula  
Matthew Berry  
Mark Haun  
Jake Janovetz  
Michael Kramer  
Dima Moussa  
Daniel Sachs  
Brian Wade

This work is produced by The Connexions Project and licensed under the  
Creative Commons Attribution License <sup>‡</sup>

## Abstract

You will design a low-pass finite impulse-response filter using the zero-placement method in MATLAB. The filter can be used as an anti-aliasing and anti-imaging filter in a multirate system.

## 1 Filter-Design Exercise

Using the zero-placement method, design the FIR filters for the multirate system in *Multirate Filtering: Introduction*<sup>1</sup>. Recall that the  $z$ -transform of a length-  $N$  FIR filter is a polynomial in  $z^{-1}$ , and that this polynomial can be factored into  $N - 1$  roots.

$$\begin{aligned} H(z) &= h_0 + h_1 z^{-1} + h_2 z^{-2} + \dots \\ &= (z_1 - z^{-1})(z_2 - z^{-1})(z_3 - z^{-1}) \dots \end{aligned} \tag{1}$$

---

\*Version 1.1: Aug 20, 2006 9:38 pm GMT-5

<sup>†</sup><http://cnx.org/content/m10815/2.6/>

<sup>‡</sup><http://creativecommons.org/licenses/by/2.0/>

<sup>1</sup>"Multirate Filtering: Introduction", Figure 1 <<http://cnx.org/content/m10024/latest/#fig1>>

Use this relation to design a low-pass filter (for the anti-aliasing and anti-imaging filters of the multirate system) by placing twelve complex zeros on the unit circle at  $\pm(\frac{3\pi}{8})$ ,  $\pm(\frac{\pi}{2})$ ,  $\pm(\frac{5\pi}{8})$ ,  $\pm(\frac{3\pi}{4})$ ,  $\pm(\frac{7\pi}{8})$ , and  $\pm\pi$ . This filter that you have just designed will serve for both FIR 1 and FIR 3. For filter FIR 2 (operating at the decimated rate), use four equally-spaced zeros on the unit circle located at  $\pm(\frac{\pi}{4})$  and  $\pm(\frac{3\pi}{4})$ . Be sure to adjust the resulting filter coefficients to ensure that the gain does not exceed one at any frequency.

Design your filters by writing a MATLAB script to compute the filter coefficients from the given zero locations. The MATLAB function `poly` is very useful for this; type `help poly` in MATLAB for details.

Once you have determined the coefficients of the filters, use MATLAB function `freqz` to plot the frequency responses. You will find that the frequency response of these filters has a large gain. Adjust the resulting filter coefficients to ensure that the largest frequency gain is less than or equal to one by dividing the coefficients by an appropriate value. Do the frequency responses match your expectations based on the locations of the zeros in the z-plane?