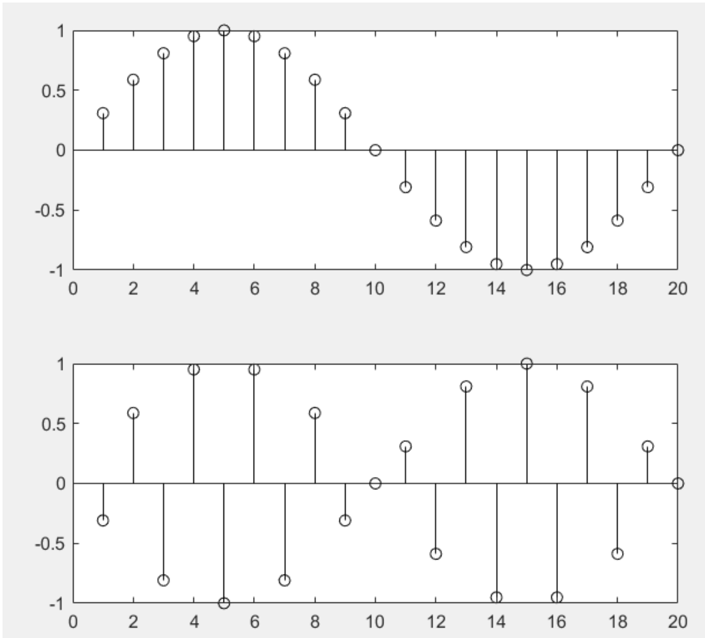
Thomas Smallarz ECE444 Homework 2

1) Spectral Inverter

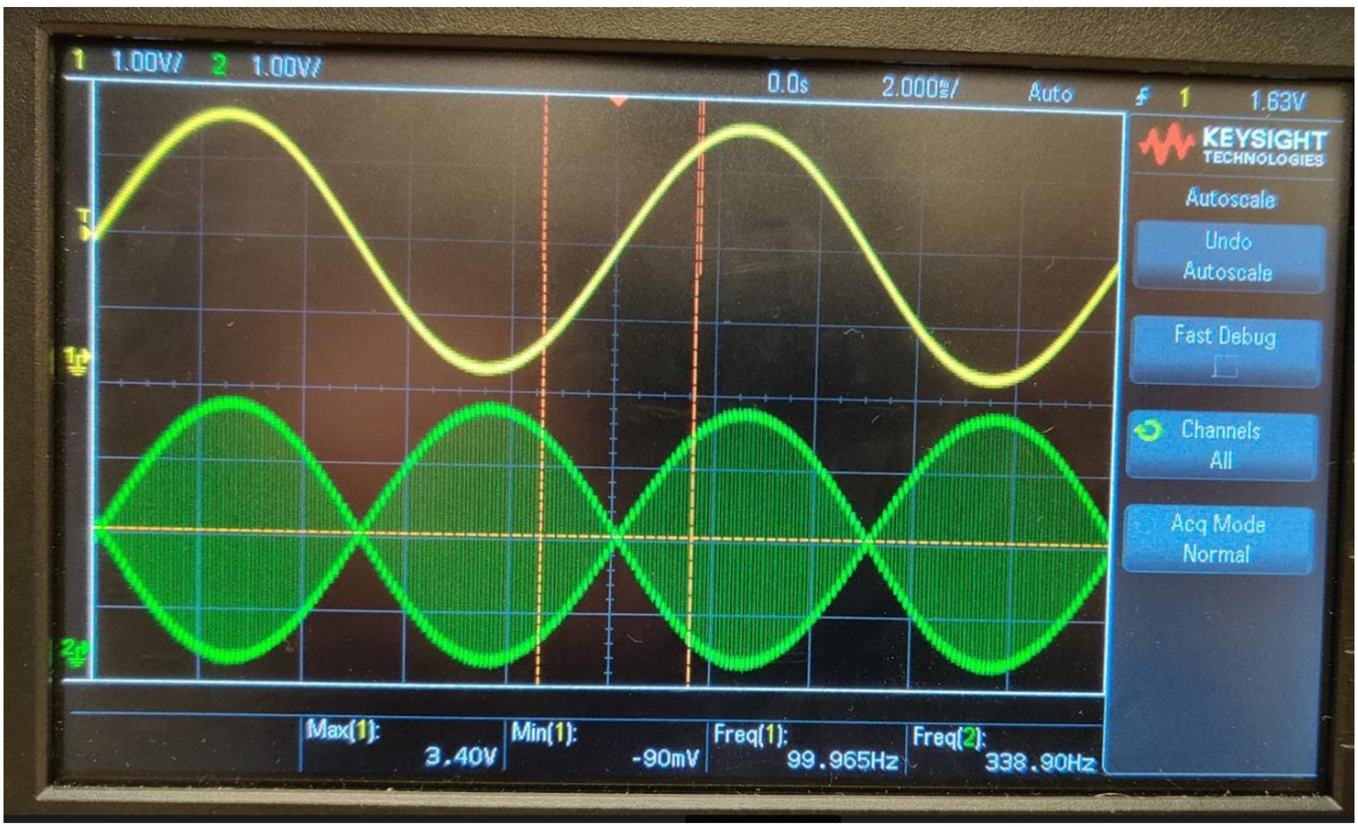
Whatever is inputted is mirrored over the zero axis (1.65V axis in this case). We can see this using MATLAB:

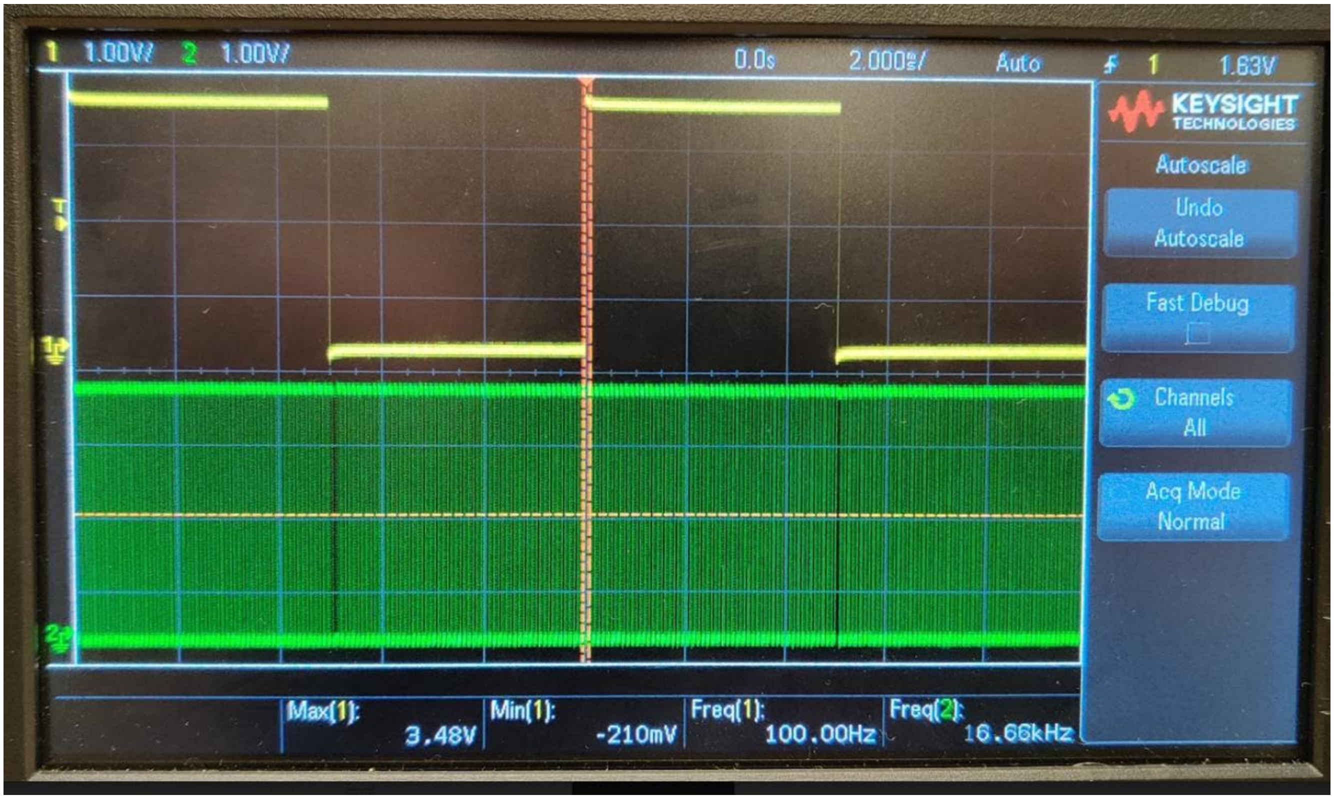


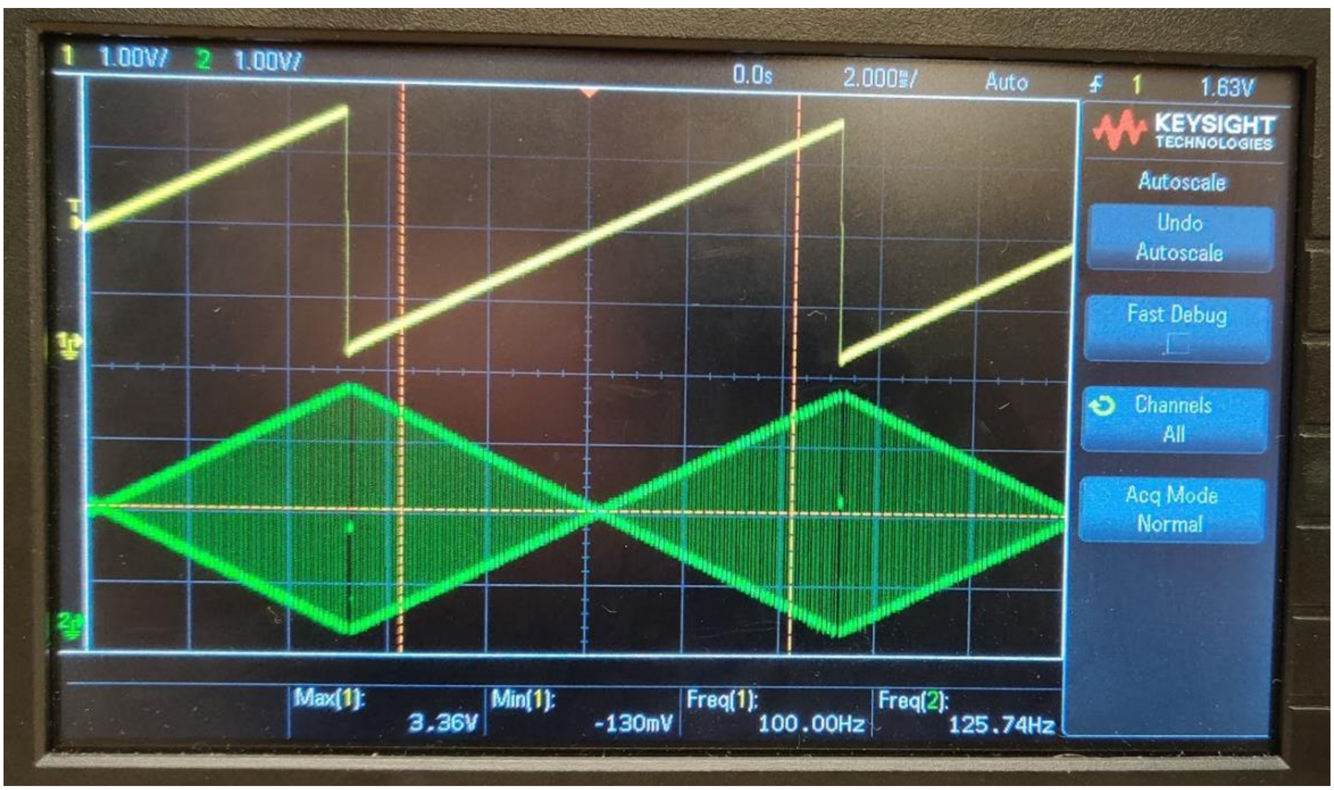

Editor - C:\Users\NI98JNA\OneDrive-Deere&Co\OneDrive - Deere & C

SpectralInverter.m | Q1.m

```
n = 1:20;
x = sin(pi*n./10);
y = zeros(length(x));
for k = 1:length(x)
    y(k) = ((-1).^k) .* x(k);
end

subplot(2,1,1); stem(n,x,'k');
subplot(2,1,2); stem(n,y(:,1),'k');
```




1 1.00V/ 2 1.00V/
0.0s
2.000ms/
Auto
1 1.63V
KEYSIGHT TECHNOLOGIES
Autoscale
Undo Autoscale
Fast Debug
Channels All
Acq Mode Normal
Max(1): 3.48V
Min(1): -210mV
Freq(1): 100.00Hz
Freq(2): 16.66kHz


1 1.00V/ 2 1.00V/
0.0s
2.000ms/
Auto
1 1.63V
KEYSIGHT TECHNOLOGIES
Autoscale
Undo Autoscale
Fast Debug
Channels All
Acq Mode Normal
Max(1): 3.36V
Min(1): -130mV
Freq(1): 100.00Hz
Freq(2): 125.74Hz

Thomas Smallarz ECE444 Homework 2

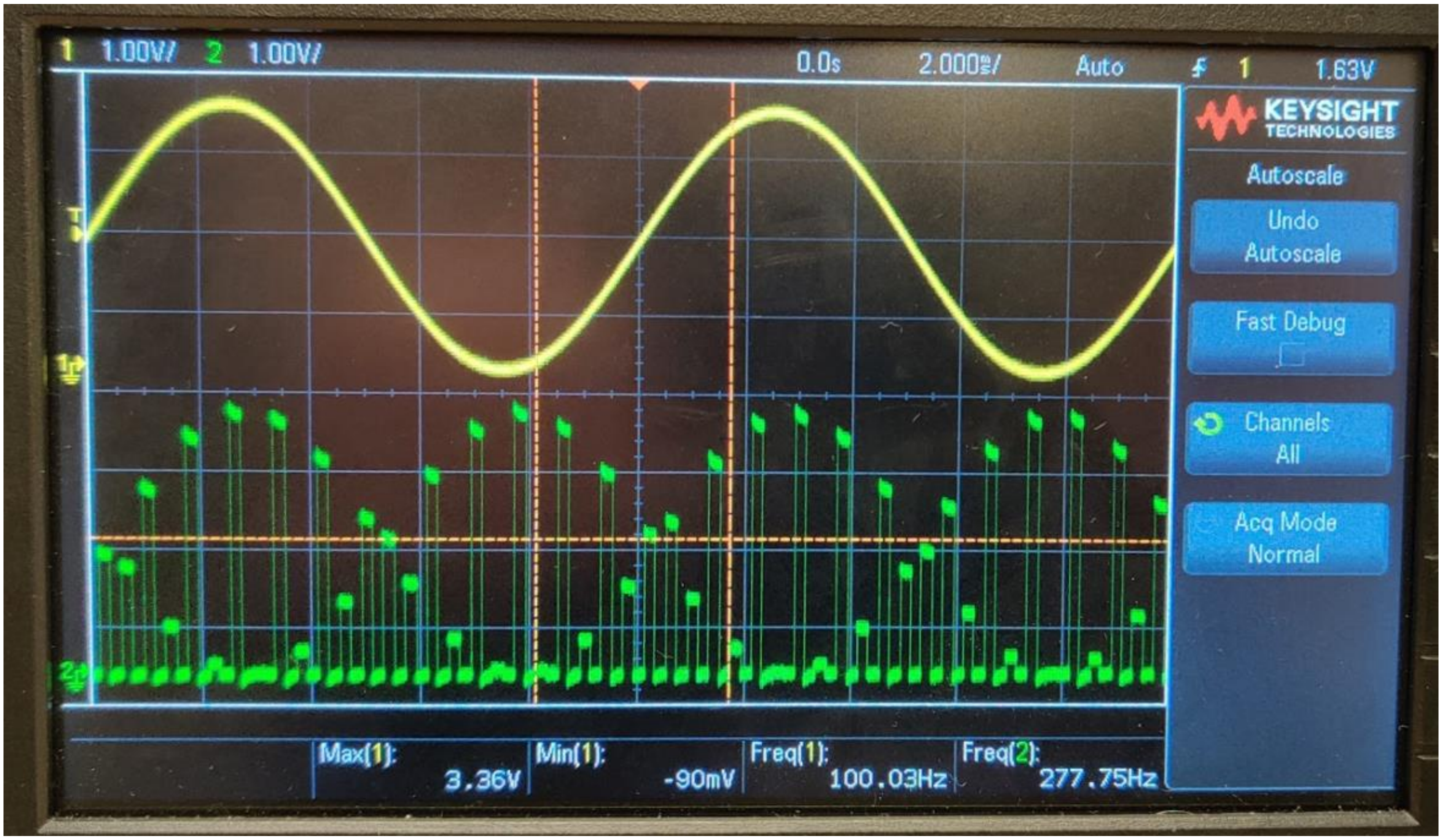
Code:

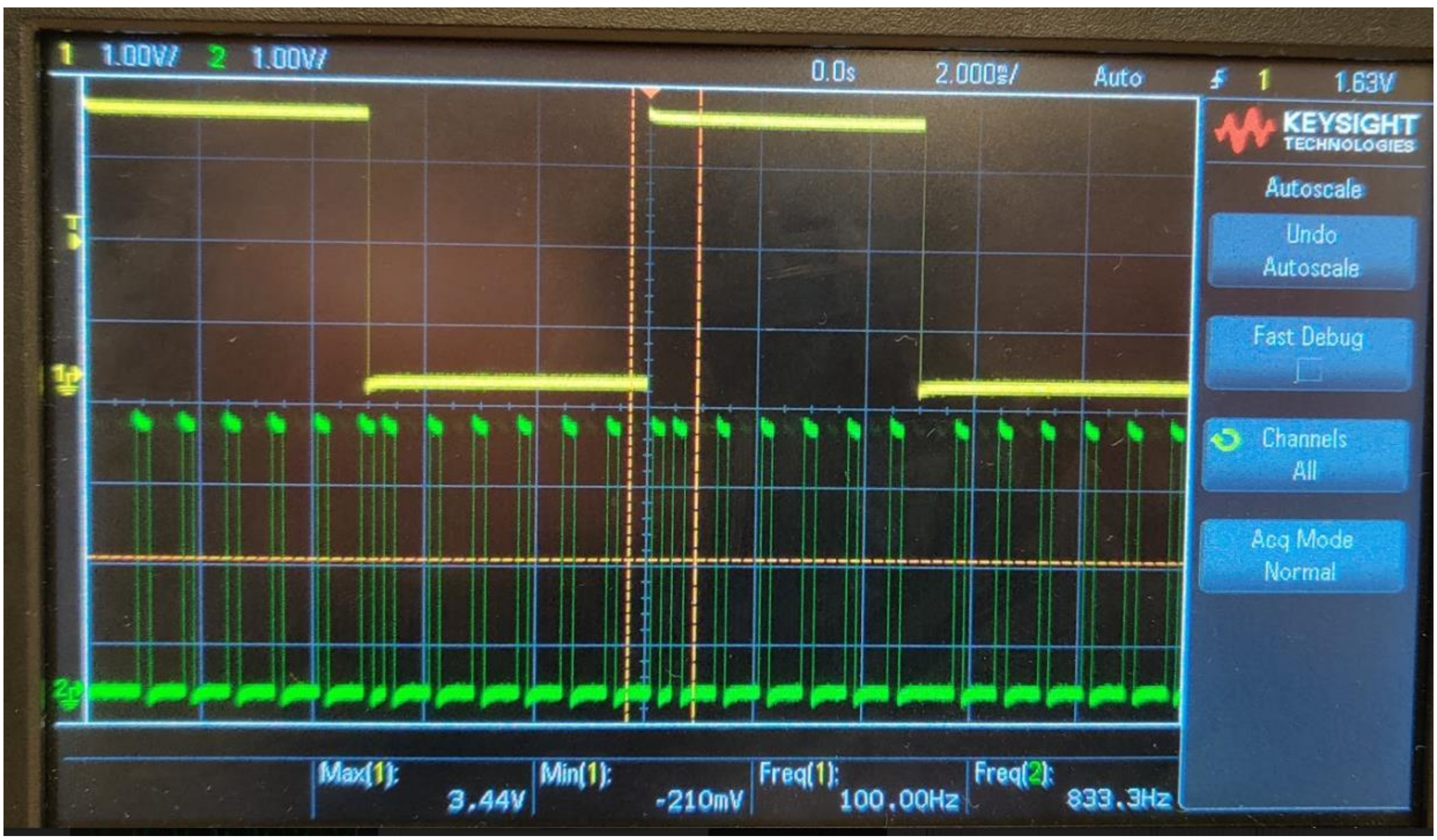
main.c | TimerInt.c | MK22F51212.h | ADC.c | MCG.c | DAC.c | startup_MK22F51212.s | stdint.h

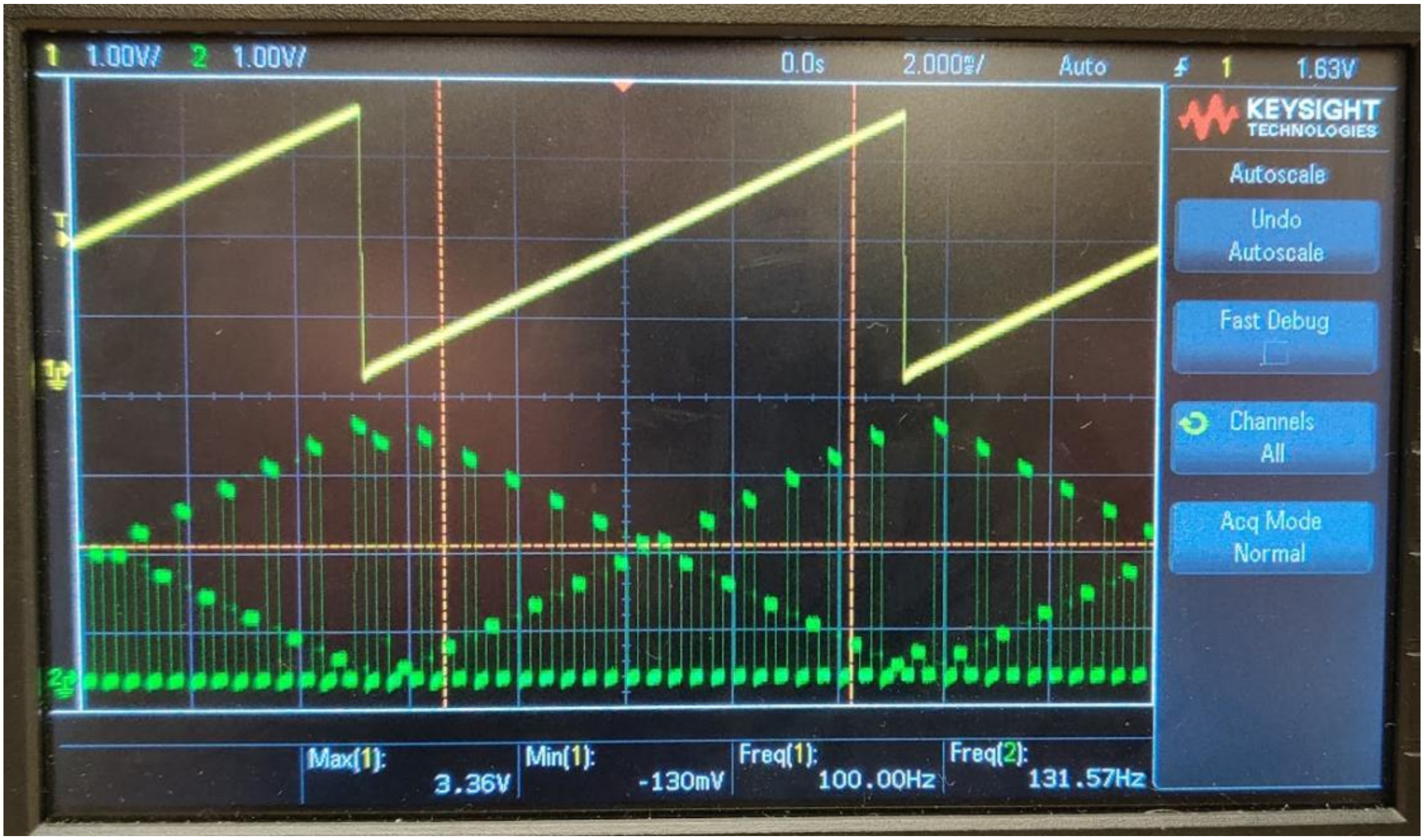
```c
#include "MCG.h"                                        //Clock header
#include "TimerInt.h"                                   //Timer Interrupt Header
#include "ADC.h"                                        //ADC Header
#include "DAC.h"                                        //DAC Header

uint16_t adc_measurement;                     //ADC is setup for 12-bit conversion (because DAC can only
uint8_t n=0;

void PIT0_IRQHandler(void){ //This function is called when the timer interrupt expires
  //Place Interrupt Service Routine Here
  ADC0->SC1[0] &= 0xE0;    //Start conversion of channel 1
  //while(ADC_SC1_COCO(ADC0->SC1[0]));  //Wait until conversion is done
  adc_measurement = ADC0->R[0];         //Read results of conversion

  // flips value
  if(n) adc_measurement = 0x0FFF - adc_measurement;

  DAC0->DAT[0].DATL = DAC_DATL_DATA0(adc_measurement & 0xFF);    //Set Lower 8 bits of Output
  DAC0->DAT[0].DATH = DAC_DATH_DATA1(adc_measurement >> 0x8);    //Set Higher 8 bits of Output

  NVIC_ClearPendingIRQ(PIT0_IRQn);              //Clears interrupt flag in NVIC Register
  PIT->CHANNEL[0].TFLG  = PIT_TFLG_TIF_MASK;    //Clears interrupt flag in PIT Register

  if(n) n = 0;
  else n = 1;
}

int main(void){
  MCG_Clock120_Init();
  ADC_Init();
  ADC_Calibrate();
  DAC_Init();
  TimerInt_Init();
  while(1){
    //Main loop goes here
  }
}
```

2) Spectral Inverter with a twist

Thomas Smallarz ECE444 Homework 2



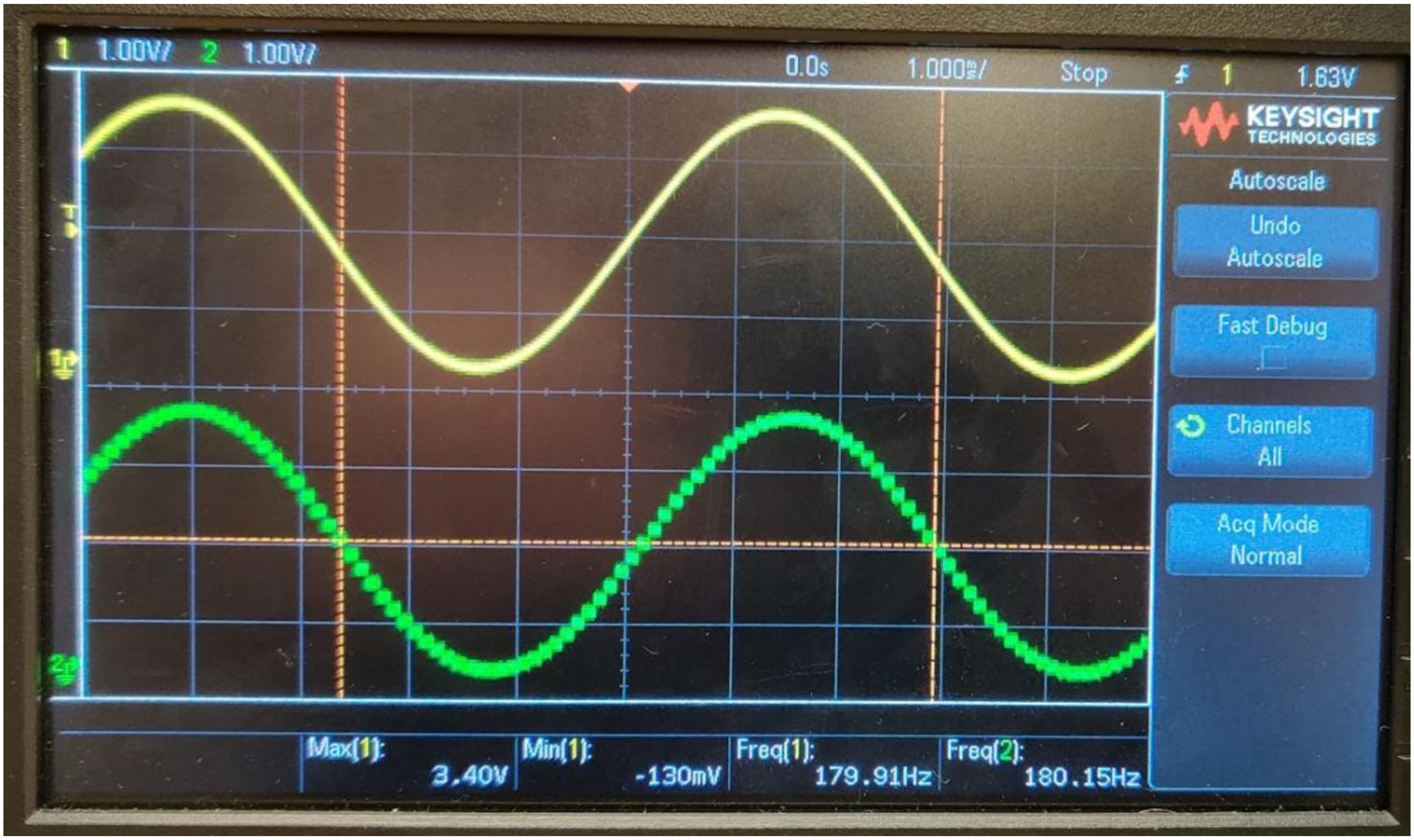


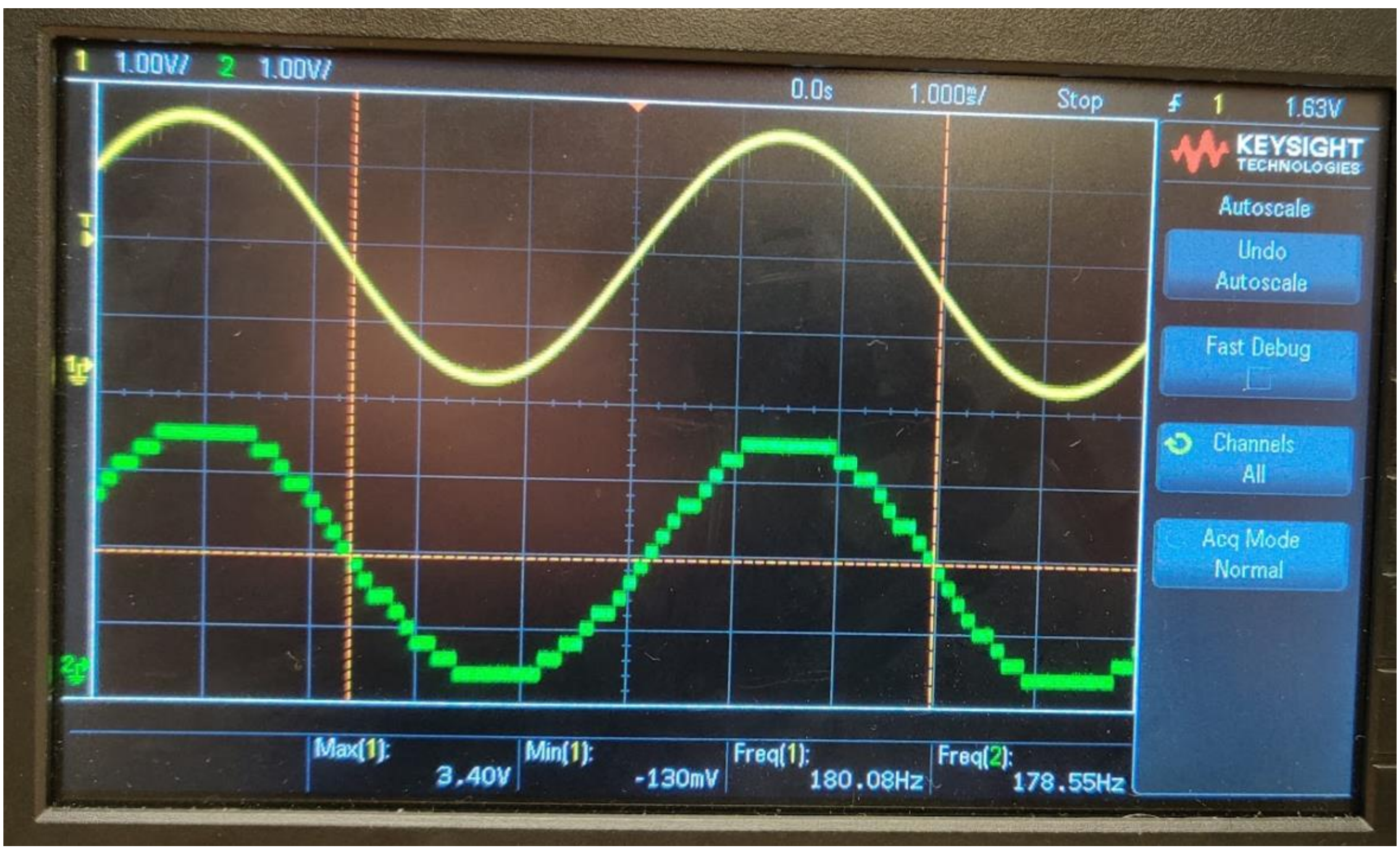

Code:

```c
#include "MK22F51212.h"                         //Device header
#include "MCG.h"                                //Clock header
#include "TimerInt.h"                           //Timer Interrupt Header
#include "ADC.h"                                //ADC Header
#include "DAC.h"                                //DAC Header

uint16_t adc_measurement;             //ADC is setup for 12-bit conversion (because DAC can only output 12-bit) but using 16-bit variable
uint8_t n = 0;

void PIT0_IRQHandler(void){ //This function is called when the timer interrupt expires
  //Place Interrupt Service Routine Here
  ADC0->SC1[0] &= 0xE0;   //Start conversion of channel 1
  //while(ADC_SC1_COCO(ADC0->SC1[0])); //Wait until conversion is done
  adc_measurement = ADC0->R[0];       //Read results of conversion

  // flips value
  if((n%4)==0 || (n%4)==2) adc_measurement = 0x0;
  if((n%4)==3) adc_measurement = 0x0FFF - adc_measurement;

  DAC0->DAT[0].DATL = DAC_DATL_DATA0(adc_measurement & 0xFF);   //Set Lower 8 bits of Output
  DAC0->DAT[0].DATH = DAC_DATH_DATA1(adc_measurement >> 0x8);   //Set Higher 8 bits of Output

  NVIC_ClearPendingIRQ(PIT0_IRQn);            //Clears interrupt flag in NVIC Register
  PIT->CHANNEL[0].TFLG  = PIT_TFLG_TIF_MASK;  //Clears interrupt flag in PIT Register

  n++;
}

int main(void){
  MCG_Clock120_Init();
  ADC_Init();
  ADC_Calibrate();
  DAC_Init();
  TimerInt_Init();
  while(1){
    //Main loop goes here
  }
}
```

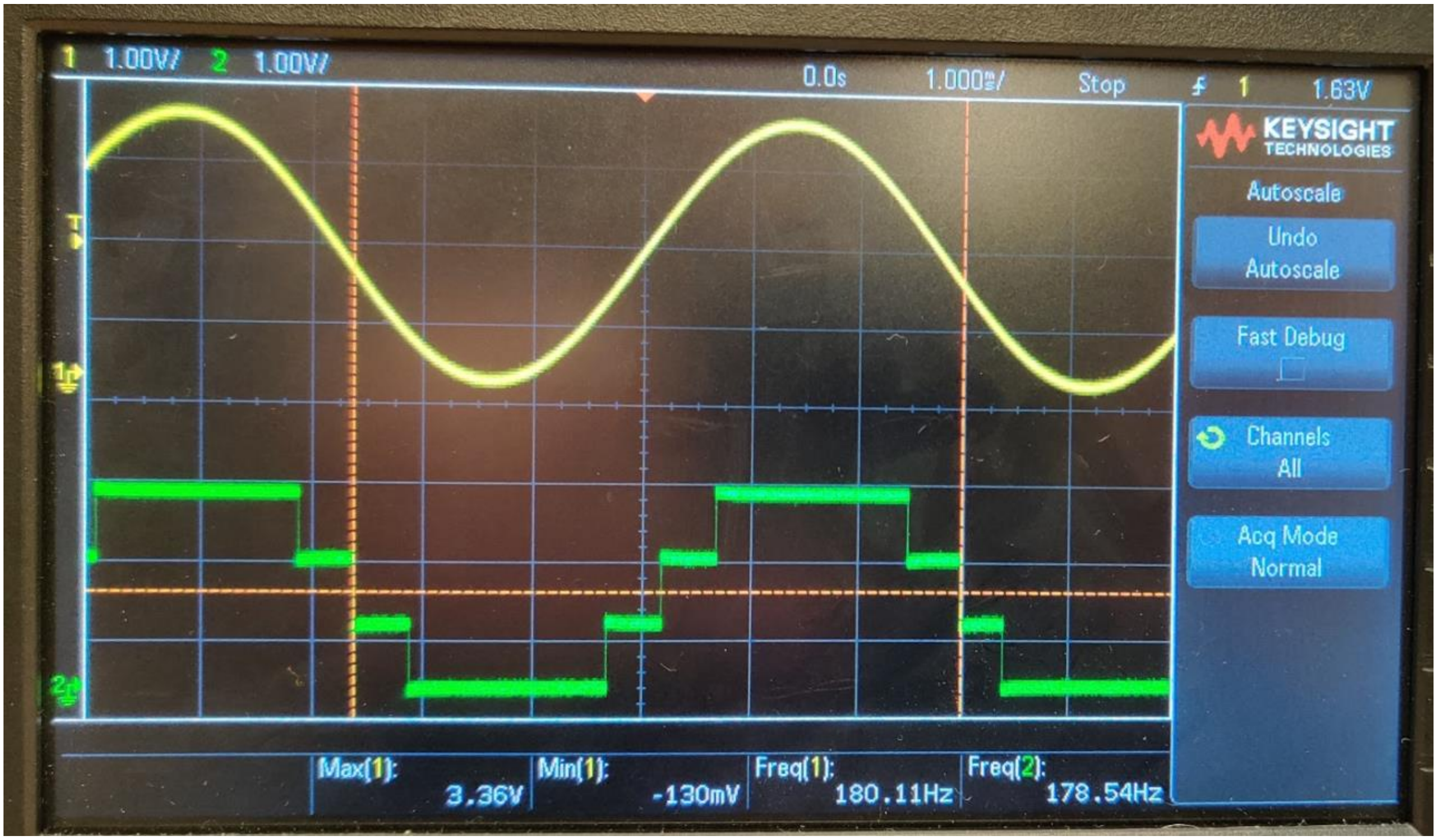
3. Adjustable Resolution

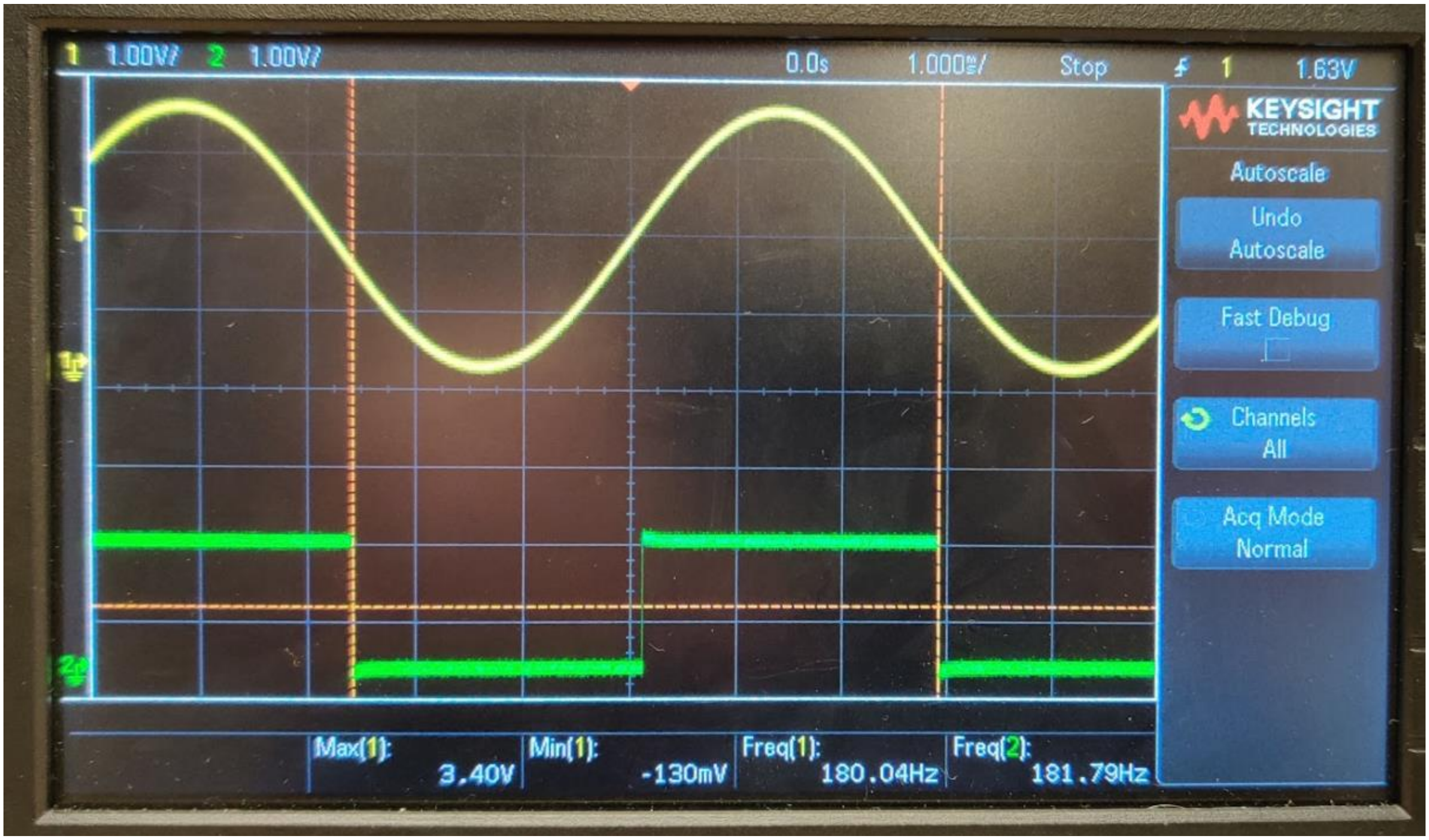
Thomas Smallarz ECE444 Homework 2



Thomas Smallarz ECE444 Homework 2



Code:

main.c | TimerInt.c | MK22F51212.h | ADC.c | MCG.c | DAC.c | startup_MK22F51212.s | stdint.h

```c
/**************************************************************************/

#include "MK22F51212.h"                                 //Device header
#include "MCG.h"                                        //Clock header
#include "TimerInt.h"                                   //Timer Interrupt Header
#include "ADC.h"                                        //ADC Header
#include "DAC.h"                                        //DAC Header

uint16_t adc_measurement;             //ADC is setup for 12-bit conversion (because DAC can only output 12-bit) but using 16-bit variable
uint8_t n = 11; // range of 0-11. Masks off (12-n) bits from LSB to MSB

void PIT0_IRQHandler(void){ //This function is called when the timer interrupt expires
  //Place Interrupt Service Routine Here
  ADC0->SC1[0] &= 0xE0;   //Start conversion of channel 1
  //while(ADC_SC1_COCO(ADC0->SC1[0]));  //Wait until conversion is done
  adc_measurement = ADC0->R[0];        //Read results of conversion

  // flips value
  adc_measurement = adc_measurement >> n;
  adc_measurement = adc_measurement << n;

  DAC0->DAT[0].DATL = DAC_DATL_DATA0(adc_measurement & 0xFF);   //Set Lower 8 bits of Output
  DAC0->DAT[0].DATH = DAC_DATH_DATA1(adc_measurement >> 0x8);   //Set Higher 8 bits of Output

  NVIC_ClearPendingIRQ(PIT0_IRQn);              //Clears interrupt flag in NVIC Register
  PIT->CHANNEL[0].TFLG  = PIT_TFLG_TIF_MASK;    //Clears interrupt flag in PIT Register

}

int main(void){
  MCG_Clock120_Init();
  ADC_Init();
  ADC_Calibrate();
  DAC_Init();
  TimerInt_Init();
  while(1){
    //Main loop goes here
  }
}
```

**2.1-5** For each of the following transfer functions, determine and plot the poles and zeros of $H(s)$, and use the pole and zero information to predict overall system behavior. Confirm your predictions by graphing the system's frequency response (magnitude and phase).

**(c)** $H(s) = \frac{0.03s^4+1.76s^2+15.22}{s^4+2.32s^3+9.79s^2+13.11s+17.08}$

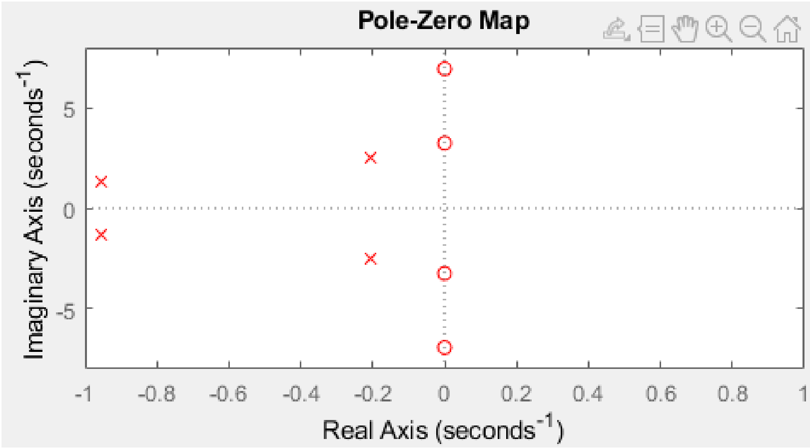
Using MATLAB,

```
num = [0.03 0 1.76 0 15.22];
den = [1 2.32 9.79 13.11 17.08];

H = tf(num,den);
Hp = pole(H)
Hz = zero(H)

subplot(3,1,1);
pzplot(H); axis([-1 1 -8 8]);
```

Command Window

```
>> C2_1_5c

Hp =

  -0.2063 + 2.4986i
  -0.2063 - 2.4986i
  -0.9537 + 1.3446i
  -0.9537 - 1.3446i


Hz =

   0.0000 + 6.9372i
   0.0000 - 6.9372i
   0.0000 + 3.2469i
   0.0000 - 3.2469i
```

Thomas Smallarz ECE444 Homework 2




From looking at the P/Z Map we can see that

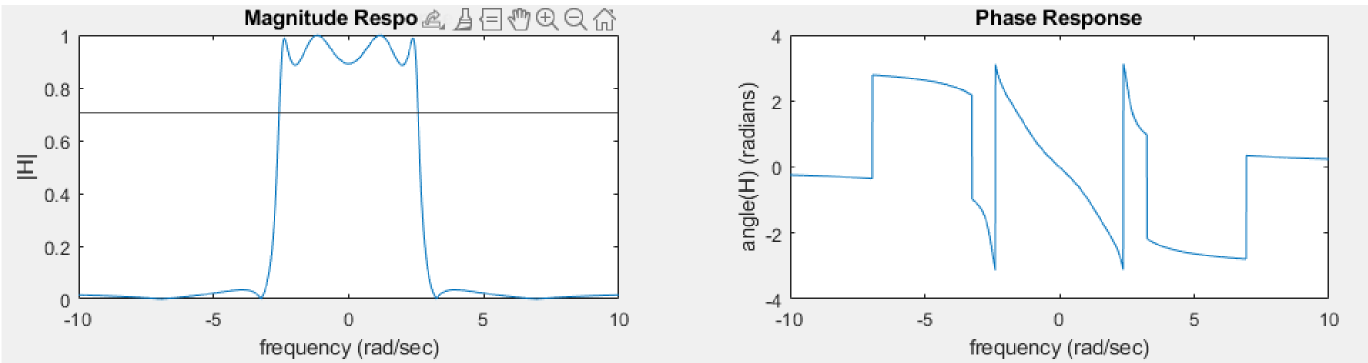
- As omega goes to infinity |H(s)| = 0
- There are some poles close to omega = 0 rad/sec (DC)

From this I would assume

- This transfer function has low pass-ish characteristics. Looking at the graph I would make a guess that it passes frequencies in the 0 – 2 rad/sec band
- Difficult to guess what type of phase response this T.F. just by looking at the P/Z Map

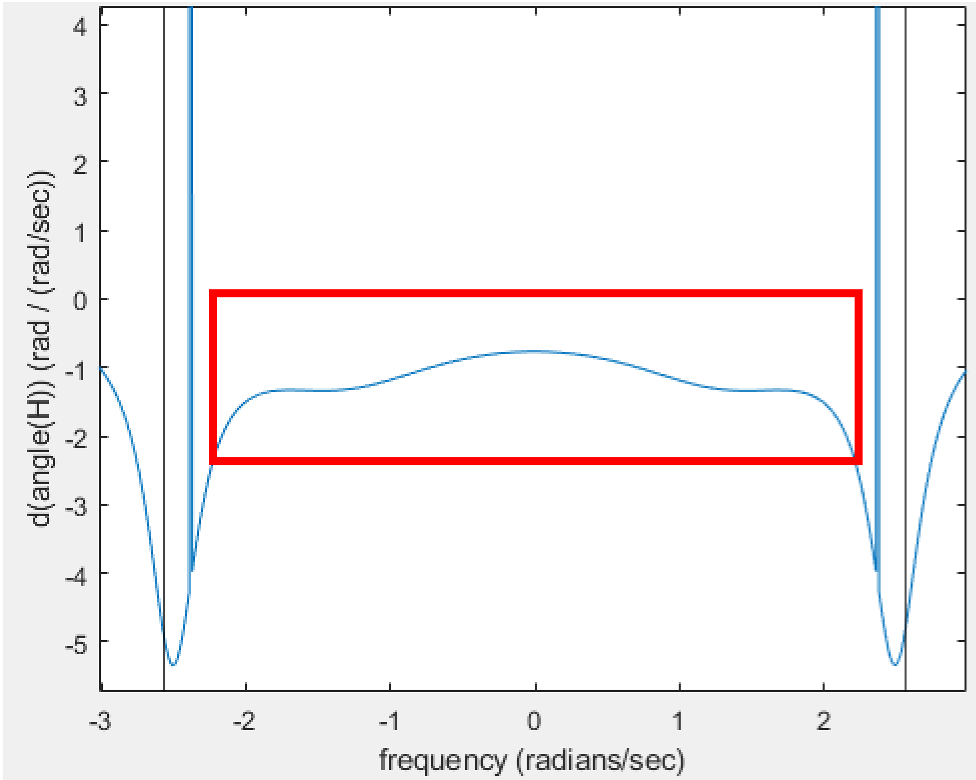
In MATLAB,

```
w = -10:0.1:10;
s = j*w;
H = (0.03.*s.^4 + 1.76.*s.^2 + 15.22) ./ (s.^4 + 2.32.*s.^3 + 9.79.*s.^2 + 13.11.*s + 17.08);
subplot(3,1,2);
plot(w,abs(H)); title("Magnitude Response"); xlabel("frequency (rad/sec)"); ylabel("|H|");
subplot(3,1,3);
plot(w,angle(H)); title("Phase Response"); xlabel("frequency (rad/sec)"); ylabel("angle(H) (degrees)");
```




From these plots we can see that:

- T.F. is a low-pass filter. From its shape it looks epileptic
- Has a ½ power point at ±2.57 rad/sec. So, our guess of the pass band being from 0 → 2 rad/sec was a somewhat good estimate
- It appears the phase response in the pass band (-2.57 rad/sec → 2.57 rad/sec) is somewhat close to linear. Plotting the first derivative of the phase response (seen on next page), we see the slope of phase is sticks close to -1 rad / (rad/sec) . This can help keep distortion less transmission in the pass band


d(angle(H)) (rad / (rad/sec))
frequency (radians/sec)
4
3
2
1
0
-1
-2
-3
-4
-5
-3
-2
-1
0
1
2

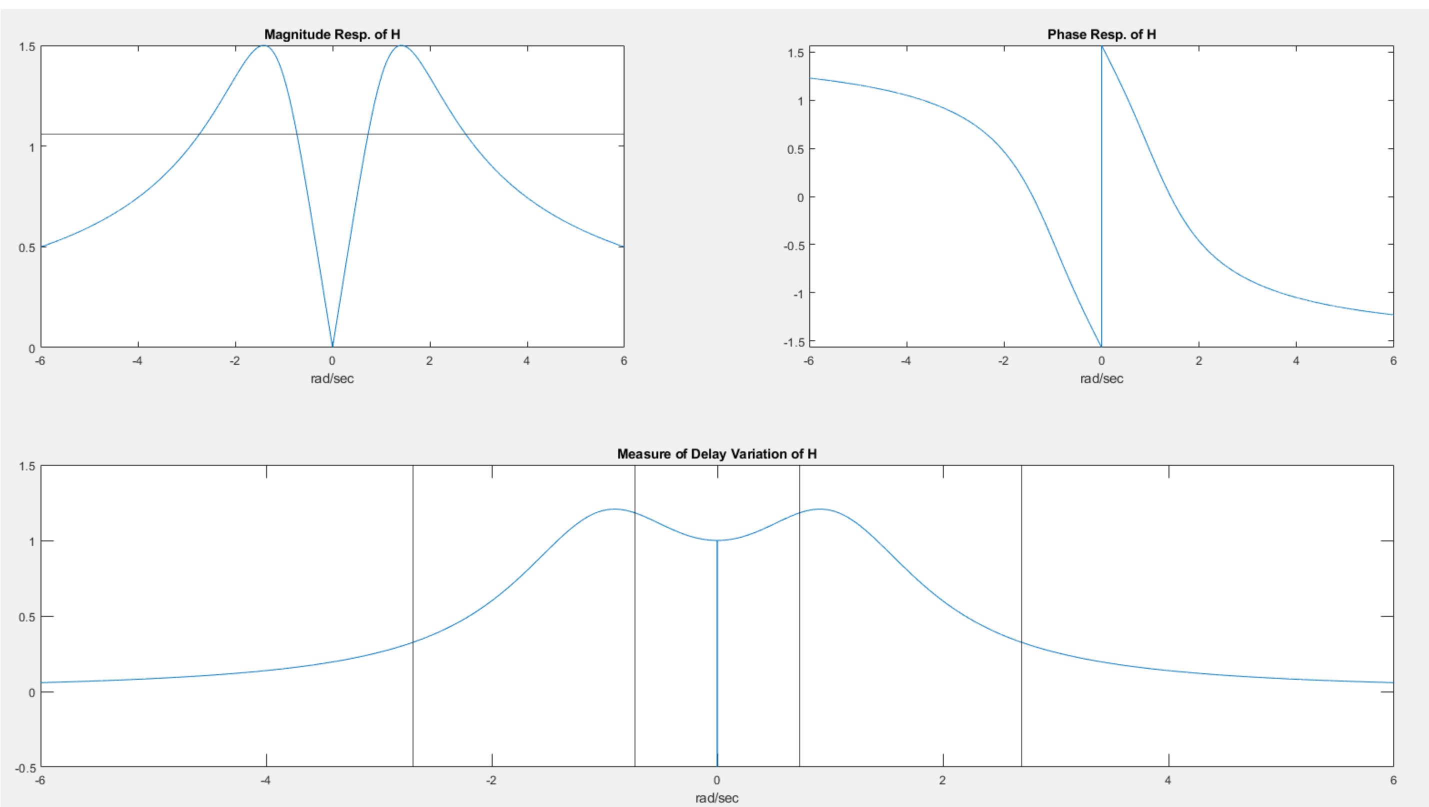
**2.2-1** Consider the LTIC system with transfer function $H(s) = \frac{3s}{s^2+2s+2}$. Using Eq. (2.15), plot the delay response of this system. Is the delay response constant for signals within the system's passband?

$$t_g(\omega) = -\frac{d}{d\omega}\angle H(\omega). \tag{2.15}$$

In MATLAB,

```
C2_1_5c.m   C2_2_1.m   +
step = 0.001;
w = -10:step:10; s = j.*w;
H = (3.*s) ./ (s.^2 + 2.*s + 2);

tg = (-diff(angle(H)))./step;

subplot(2,2,1); plot(w,abs(H)); xlabel("rad/sec"); title("Magnitude Resp. of H"); axis([-6 6
subplot(2,2,2); plot(w,angle(H)); xlabel("rad/sec"); title("Phase Resp. of H"); axis([-6 6 -p

w(end) = [];
subplot(2,2,[3 4]); plot(w,tg); xlabel("rad/sec"); title("Measure of Delay Variation of H");
axis([-6 6 -0.5 1.5]); xline(-2.7,'k'); xline(2.7,'k'); xline(-0.73,'k'); xline(0.73,'k');
```

In the bottom plot we can see that this systems delay varies from a max of 1.2 $\frac{rad}{rad/sec}$ to 0.33 $\frac{rad}{rad/sec}$ over the pass band $\left(0.73\ \frac{rad}{sec}\ to\ 2.7\frac{rad}{sec}\right)$. Therefore, this system **does not** have a constant delay response in the passband.



**2.3-3** Consider the simple $RC$ circuit shown in Fig. P2.3-3. Let $R = 1$ kΩ and $C = 1$ nF.

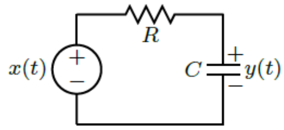



Figure P2.3-3

**(a)** Find the system transfer function.

**(b)** Plot the magnitude and phase responses.

**(c)** Show that a lowpass signal $x(t)$ with bandwidth $W \ll 10^6$ will be transmitted practically without distortion. Determine the output.

**(d)** Determine the approximate output if a bandpass signal $x(t) = g(t)\cos(\omega_c t)$ is passed through this filter. Assume that $\omega_c = 3 \times 10^6$ and that the envelope $g(t)$ has a very narrow band, on the order of 50 Hz.

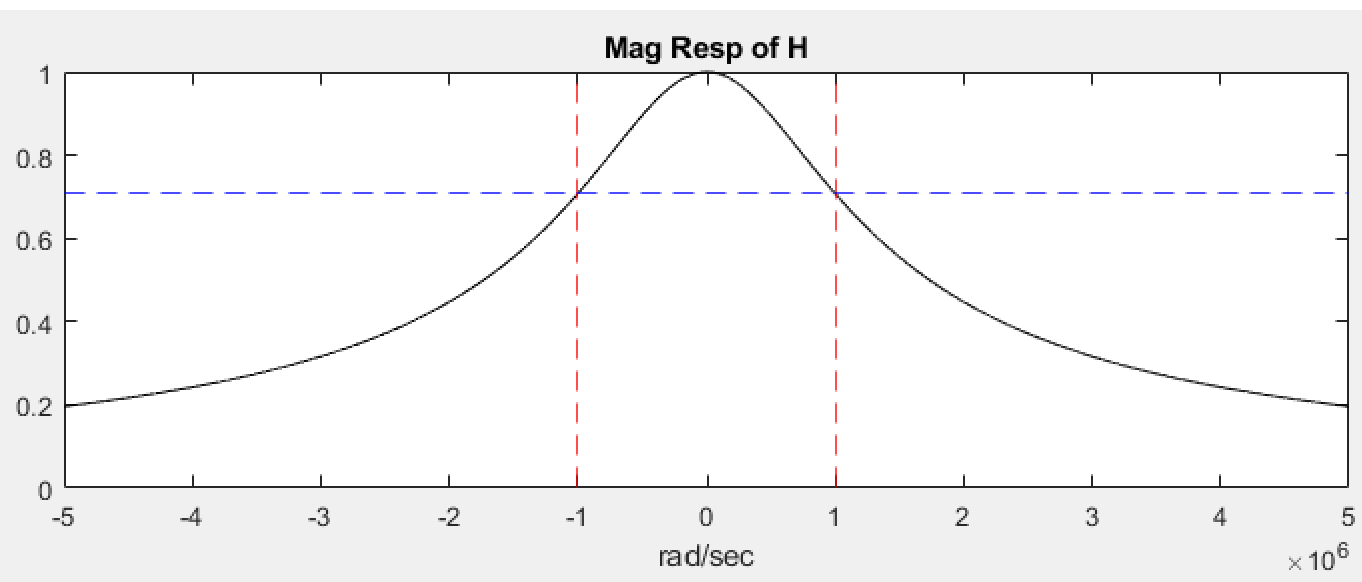
a) Use voltage division

$R \to R \qquad C \to \frac{1}{Cs}$

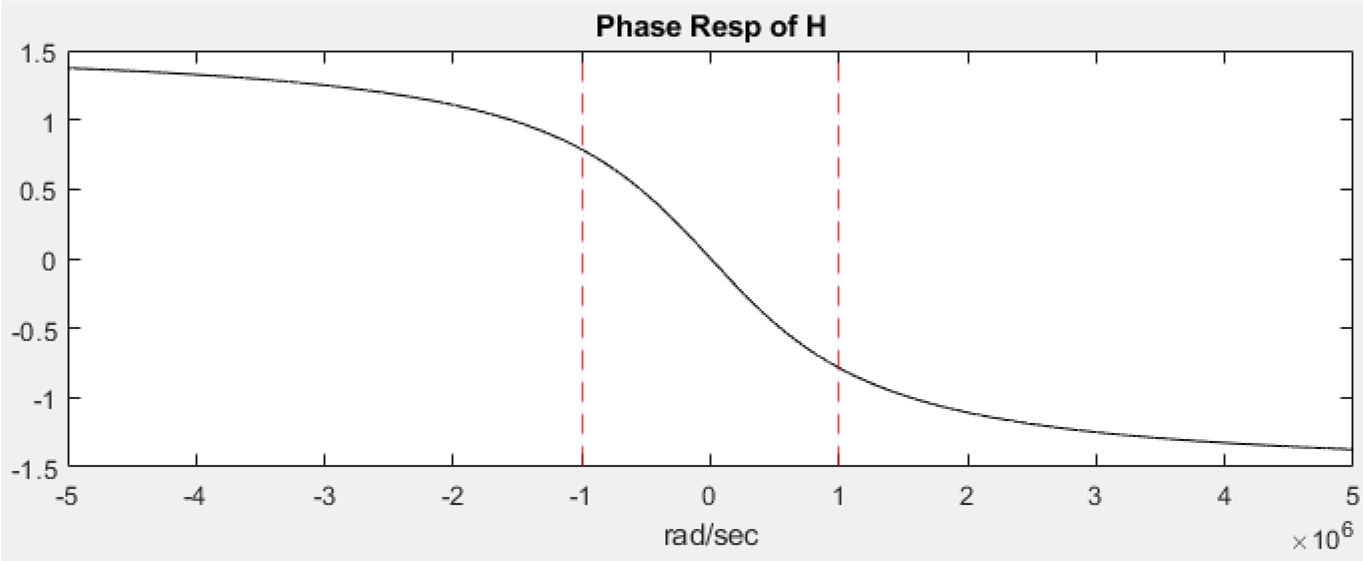
Then, $Y(s) = X(s)\left(\frac{1/_{Cs}}{R+1/_{Cs}}\right)$

Or, $H(s) = (1 + CRs)^{-1} = (1 + 0.000001s)^{-1}$

b) Using MATLAB, where cutoff frequency is shown with dashed lines



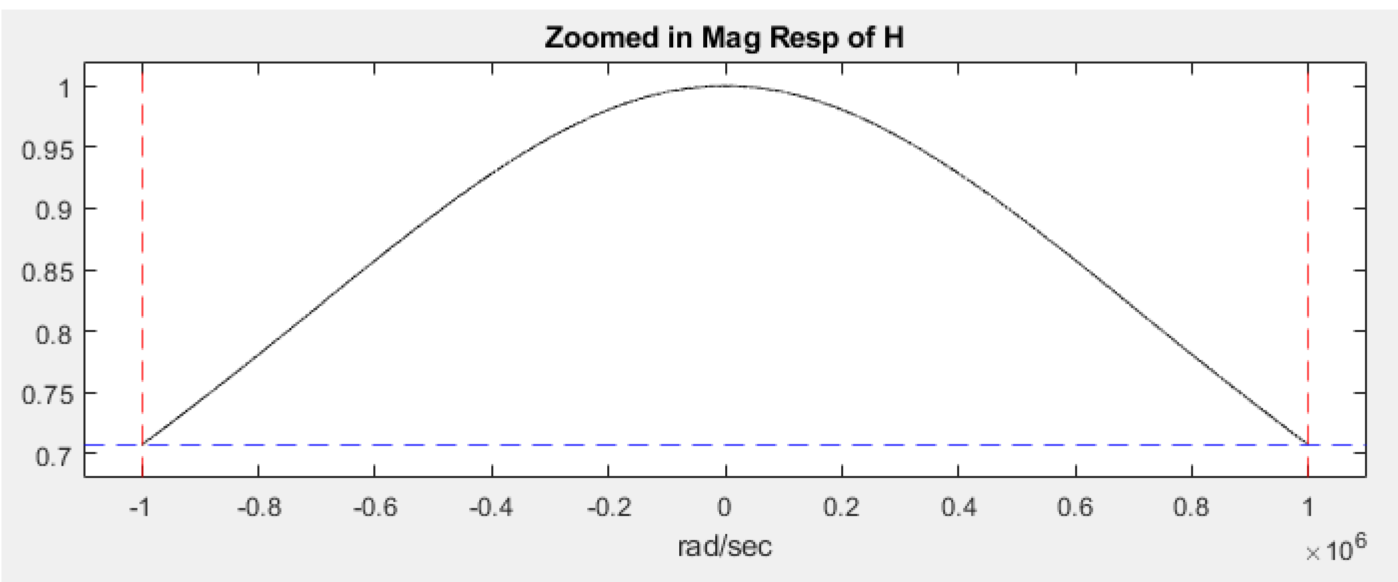

.



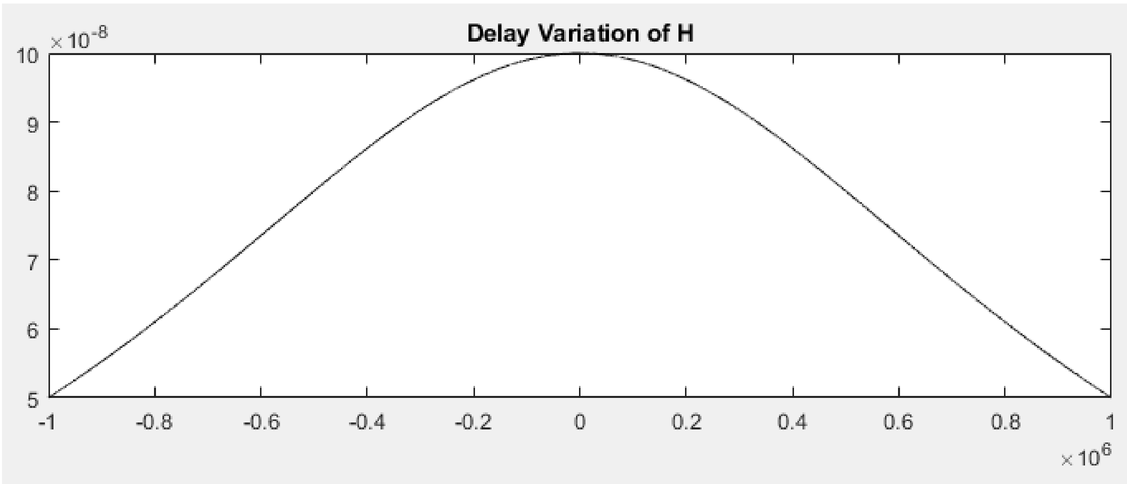

c) To transmit a signal without distortion two things are necessary

1. constant scale factor (magnitude response = constant)
2. constant delay variation (slope of phase response = constant)

Below is the Magnitude response. The scale factor changes from a minimum of 0.707 to a maximum of 1. This isn't ideal but it is somewhat constant-ish over the pass region.



Below is the delay variation (slope of the phase delay). Over the pass band it varies from $10 \times 10^{-8} \frac{rad}{rad/sec}$ $5 \times 10^{-8} \frac{rad}{rad/sec}$




If signal $x(t)$ is inputted to this system, you could expect an output of $y(t) = Ax(t - t_d)$

Where:

$A = 0.7071 \rightarrow 1$

$t_d = -0.7854\, rad \rightarrow 0.7854\, rad$

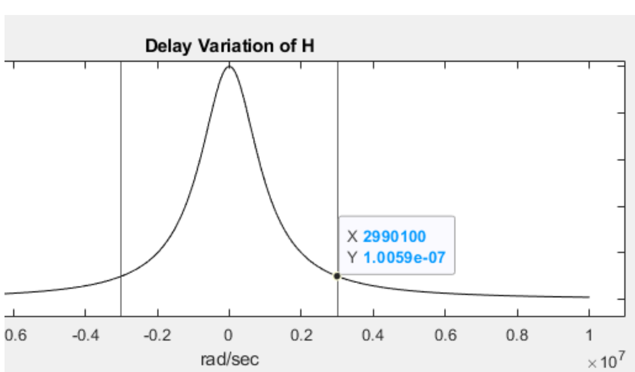
d) From page 98 in book, output should have form

$$y_{\mathrm{bp}}(t) = |a| x(t - t_{\mathrm{g}}) \cos\left[\omega_{\mathrm{c}}(t - t_{\mathrm{g}}) + \phi_0\right]$$

$$|a| = |H(j\omega_c)| = 0.3162$$

```
>> abs(H(j*3*10^6))

ans =

    0.3162
```

$$t_g = d\big(\angle H(j\omega_c)\big) = 1 * 10^{-7}$$



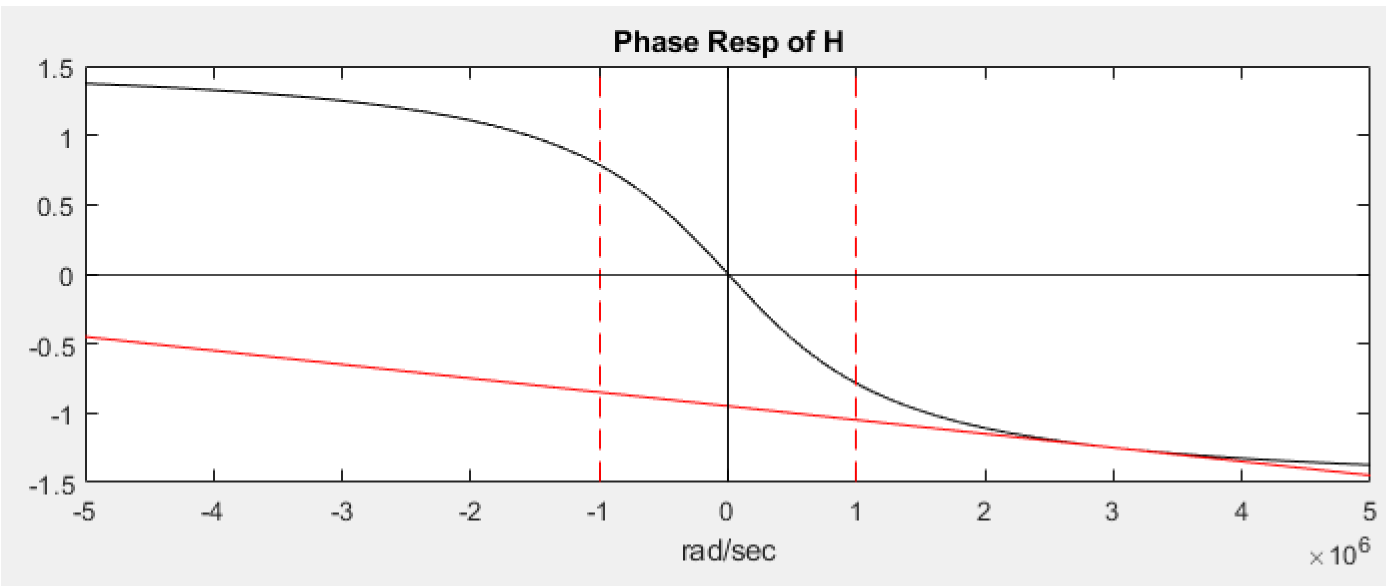
Using point slope we can find $\phi_0$

$$y - y_1 = m(x - x_1)$$

$$y - (-1.249) = 0.0000001(x - 3{,}000{,}000)$$

```
y = @(x) -0.0000001.*(x-3000000) - 1.249;
```

```
>> y(0)

ans =

   -0.9490
```



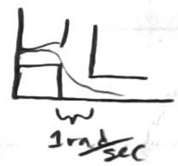

Therefore, $y_{bp}(t) = 0.3162x(t - 1 \times 10^{-7})\cos[3 \times 10^6 (t - 1 \times 10^{-7}) - 0.949]$

Editor - C:\Users\thomas.smallarz\Documents\MATLAB\HW2\C2_3_3.m

C2_3_3.m

```matlab

% Essentials of Digital Signal Proscessing
% Problem 2.3-3
step = 100;
w = -1e7/2:step:1e7/2; s = j.*w;
H = @(s) (s.*1e-6 + 1).^(-1);

subplot(3,2,1); plot(w,abs(H(s)),'k'); title("Mag Resp of H"); xlabel("rad/sec");
xline(1e6,'--r'); xline(-1e6,'--r'); yline(1/sqrt(2),'--b');
subplot(3,2,2); plot(w,angle(H(s)),'k'); title("Phase Resp of H"); xlabel("rad/sec");
xline(1e6,'--r'); xline(-1e6,'--r'); hold on; plot(w,y(w),'r'); xline(0,'k'); yline(0,'k');

w2 = -1e6:step:1e6; s2 = j.*w2;

subplot(3,2,3); plot(w2,abs(H(s2)),'k'); title("Zoomed in Mag Resp of H"); xlabel("rad/sec");
xline(1e6,'--r'); xline(-1e6,'--r'); yline(1/sqrt(2),'--b'); axis([(-1e6-100000) (1e6+100000) 0.680 1.02]);


subplot(3,2,4); plot(w2,angle(H(s2)),'k'); title("Zoomed in Phase Resp of H"); xlabel("rad/sec");
xline(1e6,'--r'); xline(-1e6,'--r'); axis([(-1e6-100000) (1e6+100000) -1 1]);


tg = -diff(angle(H(s2)))./step;
w2_new = w2; w2_new(end) = [];
subplot(3,2,5); plot(w2_new,tg,'k'); title("Delay Variation of H"); xline(1e6); xline(-1e6); xlabel("rad/sec");

w3 = -1e7:step:1e7; s3 = j.*w3;
tg = -diff(angle(H(s3)))./step;
w3_new = w3; w3_new(end)=[];
subplot(3,2,6); plot(w3_new,tg,'k'); title("Delay Variation of H"); xline(3e6); xline(-3e6); xlabel("rad/sec");

y = @(x) -0.0000001.*(x-3000000) - 1.249;

```

2.4-3 Find suitable width T so when applied to ideal LPF Impulse resp.
$h(t) = \frac{10}{\pi} \operatorname{sinc}\left(\frac{10t}{\pi}\right)$ the transition band is approx. $1 \frac{rad}{sec}$




we know that the width of the transition band of a windowed filter is approx. half the width of the main lobe (pg. 107)

ex: rect window $\rightarrow \frac{2\pi}{T} \frac{rad}{sec}$ transition band

triangle window $\rightarrow \frac{4\pi}{T} \frac{rad}{sec}$ transition band

using Table 2.1 on pg 110

a) Rectangular

Main lobe width $= \frac{4\pi}{T}$

$$\frac{2\pi}{T} = 1 \frac{rad}{sec} \Rightarrow \boxed{T = 2\pi}$$

b) Triangular

$$MLW = \frac{8\pi}{T}$$

$$\frac{4\pi}{T} = 1 \Rightarrow \boxed{T = 4\pi}$$

c) Hann window

$$MLW = \frac{8\pi}{T}$$

$$\frac{4\pi}{T} = 1 \Rightarrow \boxed{T = 4\pi}$$

d) Hamming window

$$MLW = \frac{8\pi}{T}$$

$$\frac{4\pi}{T} = 1 \Rightarrow \boxed{T = 4\pi}$$

e) Blackman

$$MLW = \frac{12\pi}{T}$$

$$\frac{6\pi}{T} = 1 \Rightarrow \boxed{T = 6\pi}$$

2.5-1) Consider a microphone intended for use in a music recording studio. Determine a suitable frequency response for the microphone, and provide suitable values to specify the response.

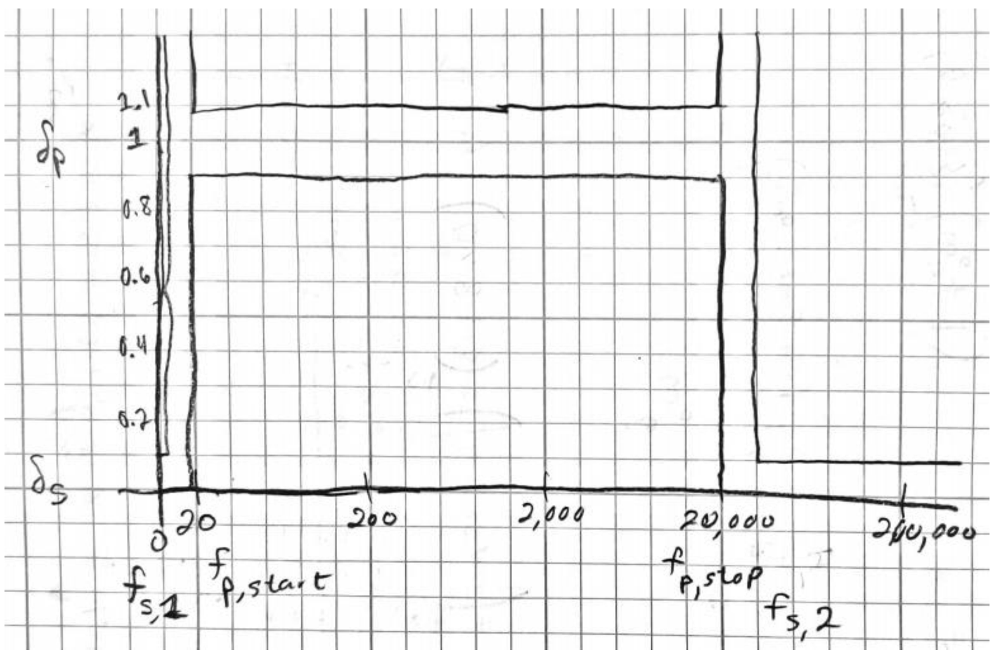
First, some info

- Human hearing is in the range of 20Hz → 20kHz
- Humans "readily perceive amplitude distortion, but are relatively insensitive to phase distortion"

What this means

- We should definetly pass 20Hz → 20kHz.
- The pass-band ripple should be very low

An ideal frequency response for a microphone would be a "pulse" with a pass-band from 20Hz to 20kHz with a gain of 1. If we are talking a non-ideal world, then I would think something that has little pass-band ripple and covers the whole frequency range would be good.




$$\delta_p = 0.9 \to 1.0\ , \quad \delta_s = 0 \to 0.1$$

$$f_{s,1} = 10Hz \quad f_{s,2} = 22{,}000Hz$$

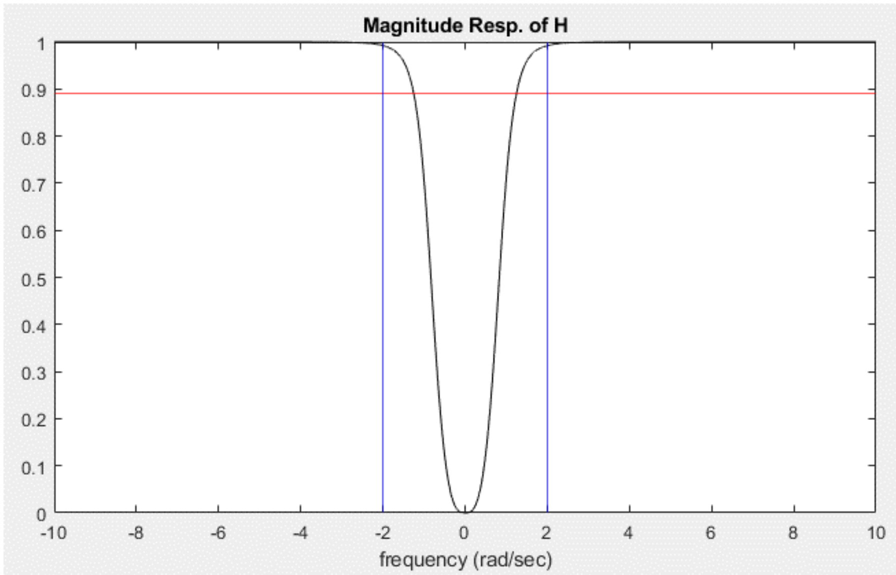
$$f_{p,start} = 20Hz \quad f_{p,stop} = 20{,}000Hz$$

Thomas Smallarz ECE444 Homework 2

2.6-2 (b,c))

$$H(s) = \frac{s^3}{s^3 + 2s^2 + 2s + 1}$$

First, we need a $w_0$ value. Let's plot the prototype high-pass transfer function.




It looks like $w_0$ is around 2 rad/sec. Let's use a cost function in MATLAB to figure out the exact frequency

Editor - C:\Users\thomas.smallarz\Documents\MATLAB\HW2\cost.m

cost.m | C2_6_2b.m

```
function [J] = cost(z)
    ideal = 0.8913;
    s = j*z;
    guess = abs(s^3 / (s^3 + 2*s^2 + 2*s + 1));

    e = abs(ideal) - abs(guess);
    J = e^2;
end
```

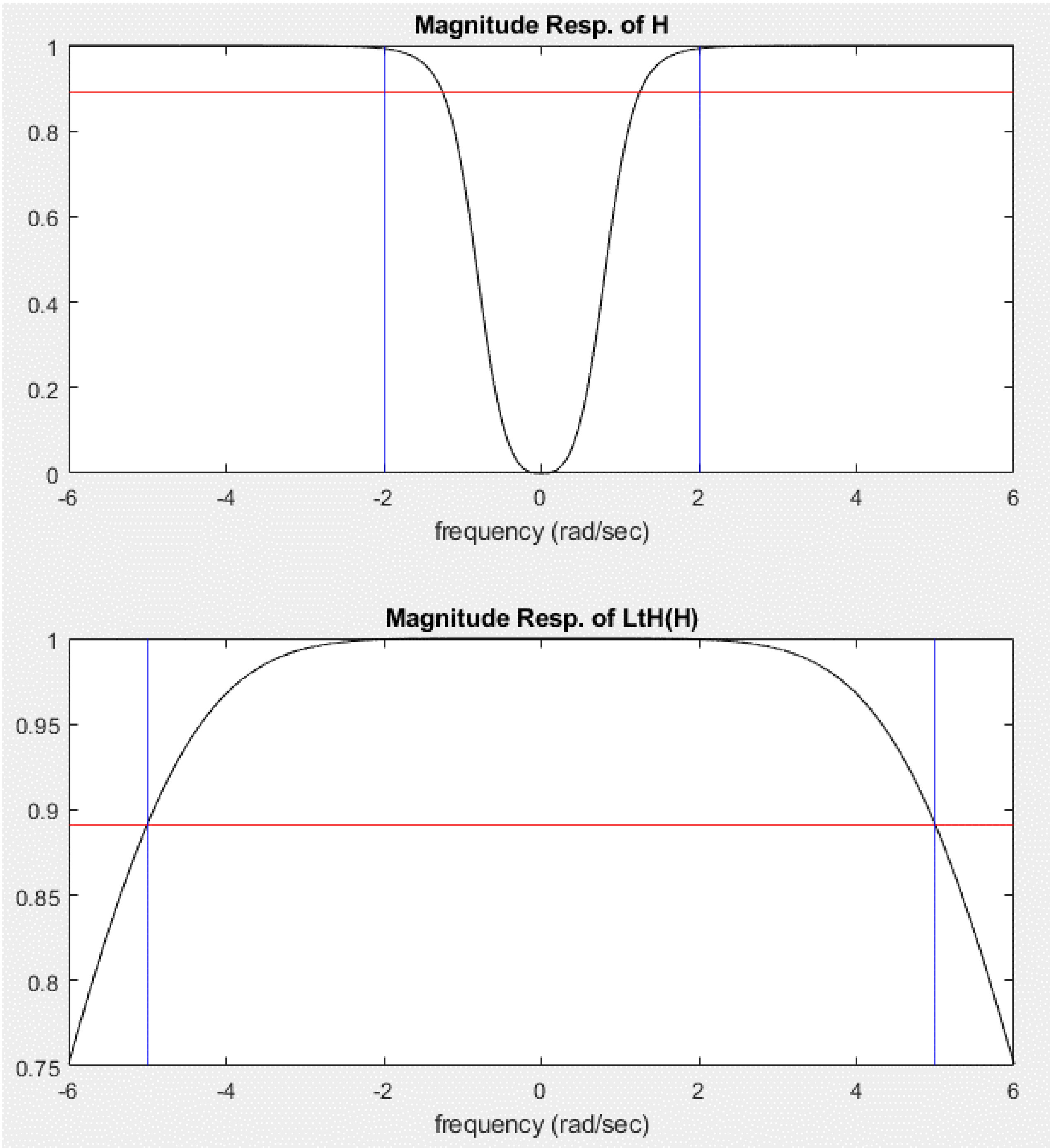
Command Window

```
>> [w,error] = fminsearch('cost',2)

w =

    1.2527


error =

   4.1112e-10
```

b) Low-pass to High-pass transformation

$$s \to \frac{\omega_0\omega_1}{s} \qquad \omega \to \frac{\omega_0\omega_1}{-\omega} \ where\ \omega_0\omega_1 = 1.2527 * 5 = 6.2637$$

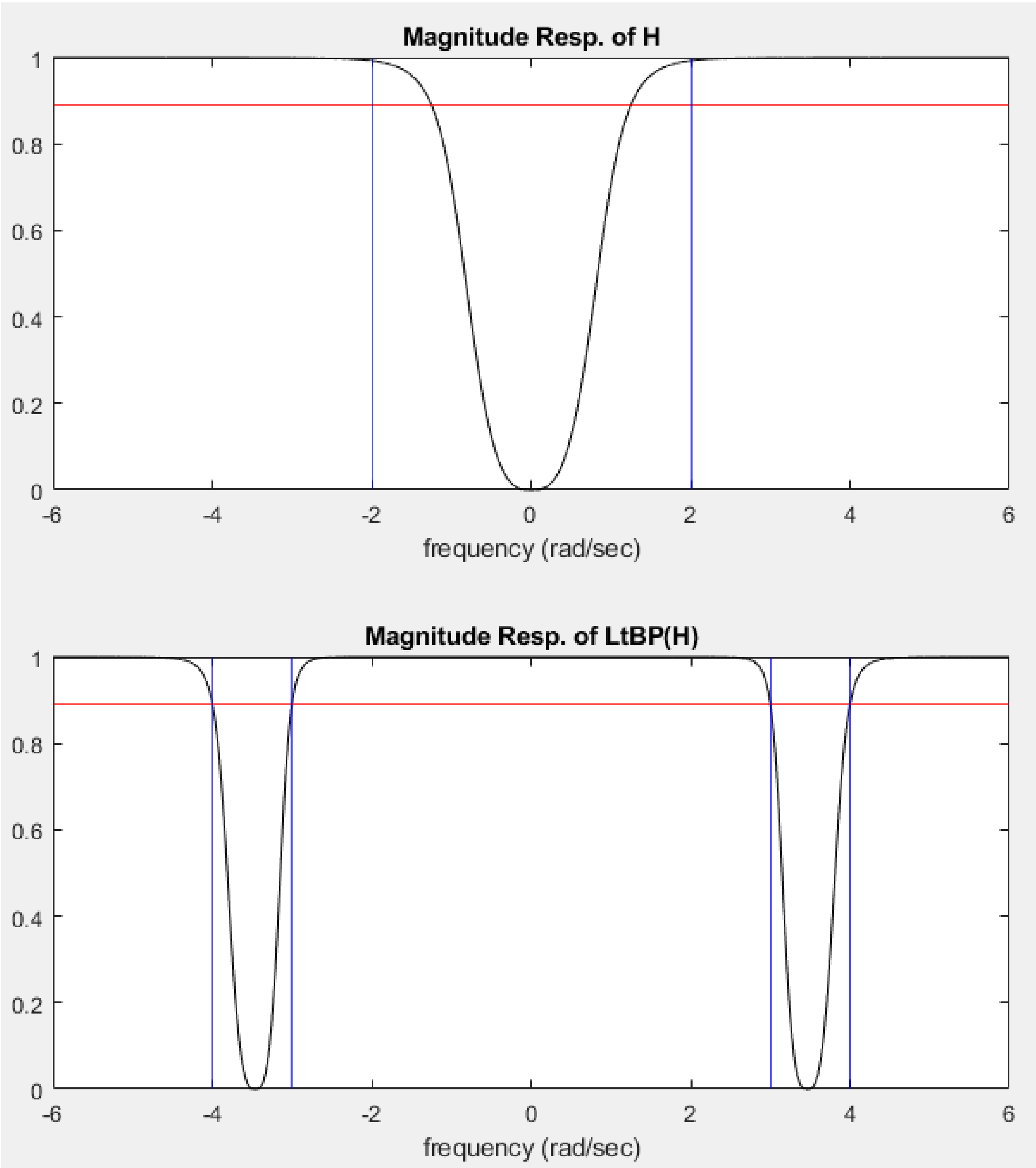
In MATLAB,



c) Lowpass-to-bandpass transformation with $\omega_1 = 3\frac{rad}{sec}$ $\quad and \quad$ $\omega_2 = 4\frac{rad}{sec}$

$$s \to \omega_0 \frac{s^2 + \omega_1\omega_2}{s(\omega_2 - \omega_1)}$$

In MATLAB,



MATLAB FOR 2.6-2 (b,c)

Editor - C:\Users\thomas.smallarz\Documents\MATLAB\HW2\C2_6_2b.m

C2_6_2b.m

```
H = @(s) s.^3 ./ (s.^3 + 2.*s.^2 + 2.*s + 1);

w = -6:0.001:6;
s_a = j.*w;
s_b = 6.2637 ./ (s_a);
s_c = 1.2527 .* (s_a.^2 + 12) ./ (s_a);

subplot(221); plot(w,abs(H(s_a)),'k'); yline(0.8913,'r'); xline(2,'b'); xline(-2,'b');
title("Magnitude Resp. of H"); xlabel("frequency (rad/sec)");

subplot(223); plot(w,abs(H(s_b)),'k'); yline(0.8913,'r'); xline(5,'b'); xline(-5,'b');
title("Magnitude Resp. of LtH(H)"); xlabel("frequency (rad/sec)");

subplot(222); plot(w,abs(H(s_c)),'k'); yline(0.8913,'r');
xline(3,'b'); xline(4,'b'); xline(-3,'b'); xline(-4,'b');
title("Magnitude Resp. of LtBP(H)"); xlabel("frequency (rad/sec)");
```

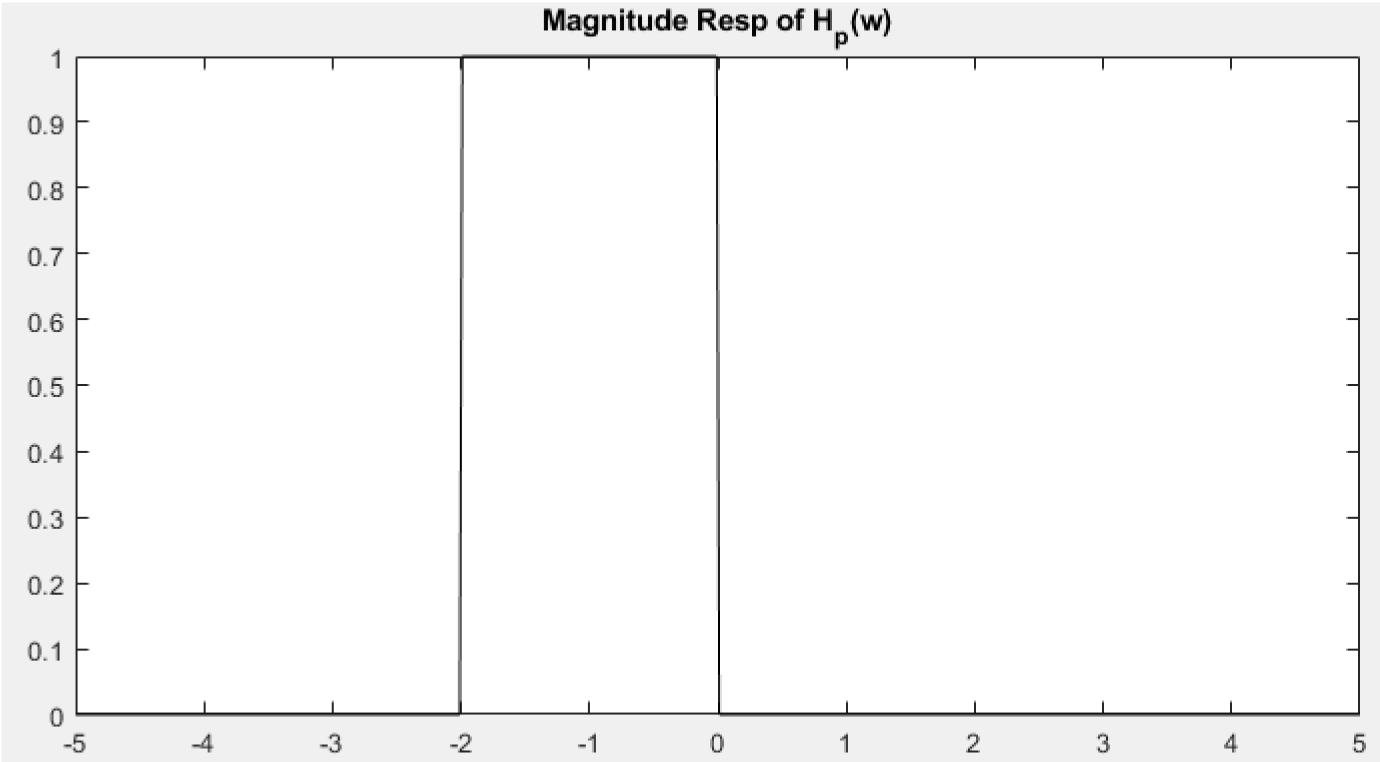
Editor - C:\Users\thomas.smallarz\Documents\MATLAB\HW2\cost.m

C2_6_2b.m cost.m

```
function [J] = cost(z)
    ideal = 0.8913;
    s = j*z;
    guess = abs(s^3 / (s^3 + 2*s^2 + 2*s + 1));

    e = abs(ideal) - abs(guess);
    J = e^2;
end
```

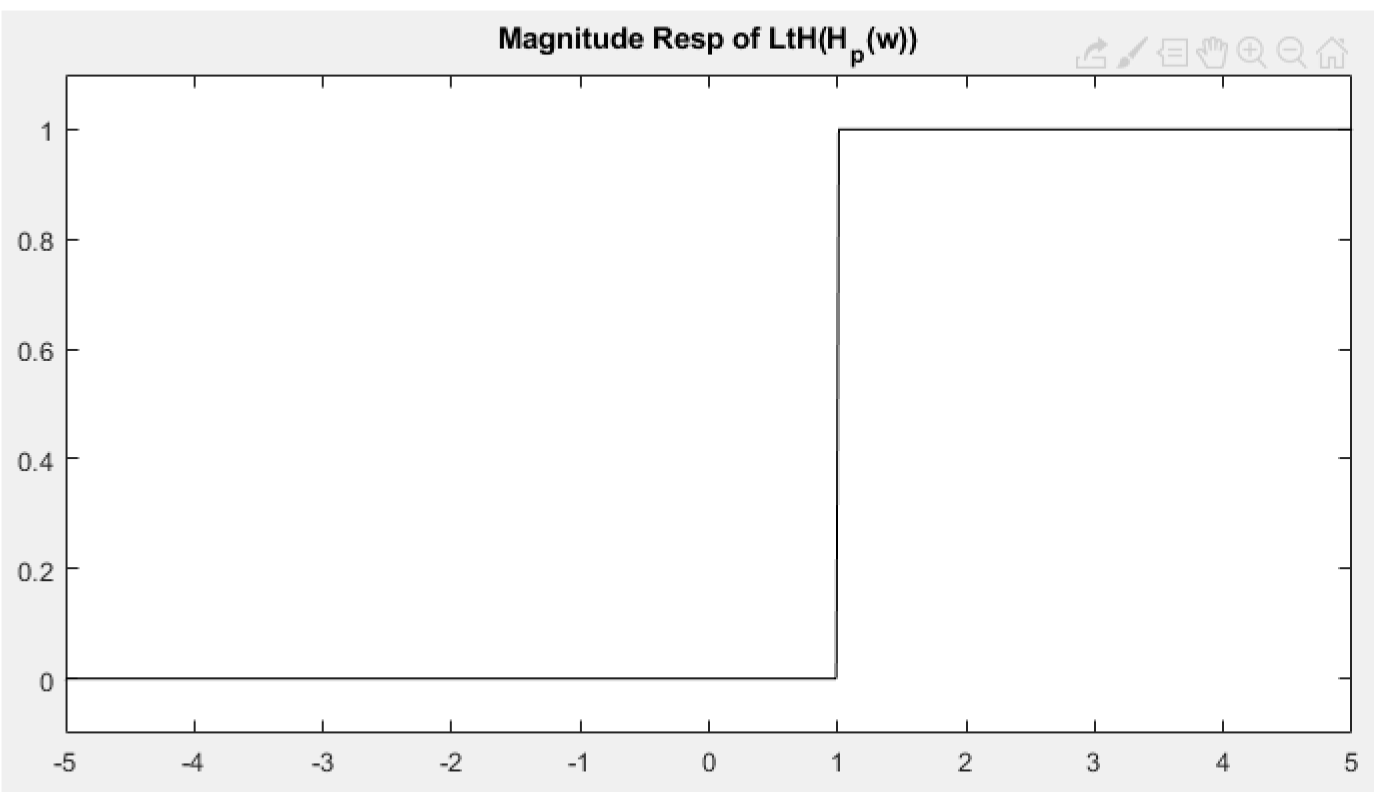
2.6-4)

a) Using MATLAB,



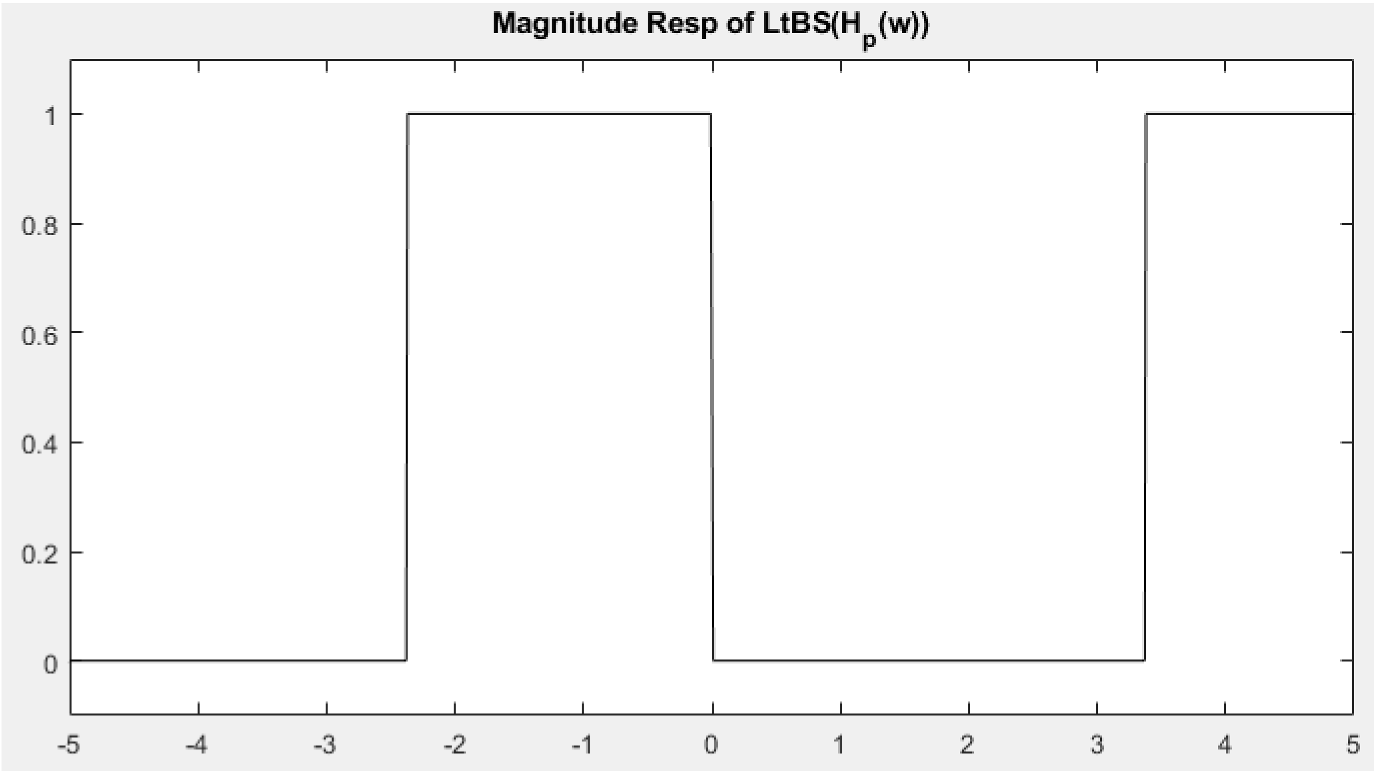

c) Lowpass-to-highpass with $\omega_0 = 1\ and\ w_1 = 2$

$$\omega \rightarrow \frac{\omega_0 \omega_1}{-\omega} = \frac{2}{-\omega}$$



e) lowpass-to-bandstop with $\omega_0 = 1$ *and* $w_1 = 2$ *and* $w_2 = 4$

$$\omega \to \omega_0 \left( \frac{\omega(\omega_2 - \omega_1)}{-\omega^2 + \omega_1 \omega_2} \right) = \frac{2\omega}{-\omega^2 + 8}$$




Editor - C:\Users\thomas.smallarz\Documents\MATLAB\HW2\C2_6_4.m

C2_6_4.m

```
gate = @(w) (abs(w) < 0.5) + (0.5).*(abs(w) == 0.5);

H_p = @(w) gate((w+1)./2);
w = -6:0.01:6;
w_c = 2./-w;
w_e = (w.*2)./(-w.^2+8);

subplot(221); plot(w,abs(H_p(w)),'k'); title("Magnitude Resp of H_p(w)");
subplot(222); plot(w,abs(H_p(w_c)),'k'); title("Magnitude Resp of LtH(H_p(w))"); axis([-5 5 -0.1 1.1]
subplot(223); plot(w,abs(H_p(w_e)),'k'); title("Magnitude Resp of LtBS(H_p(w))"); axis([-5 5 -0.1 1.1
```