

ENGG*3390- Signal Processing

Lab 3

Bilal Ayyache- 0988616

Palak Sood - 0986802

Kathlene Titus – 0954584

November 8th, 2018

Introduction

The purpose of this lab is to implement and evaluate the performance of 2 digital filters, the FIR and IIR filters. Digital filters are comprised of a system that performs mathematical operations on a discrete time signal. The main objective of a filter is to perform a computation on an input and create a new output sequence. The general form of a constant-coefficient difference equation for a discrete time linear time invariant system can be described as:

$$a_0y[n] + a_1y[n - 1] + \cdots + a_Ny[n - N] = b_0x[n] + b_1x[n - 1] + \cdots + b_Mx[n - M]$$

Equation 1: Constant-Coefficient difference equation

Note: $x[n]$ is the input, and $y[n]$ is the output.

The following system can enhance or reduce aspects of a signal. A digital filter consists of an analog to digital converter, digital to analog converter, and a microprocessor. In lab 3, a Texas instrument TMS320C5505 eZdsp USB Stick was used as a signal processor in which it was programmed to perform digital filtering through discrete time system. This discrete time system includes Finite Impulse Response (FIR), Infinite impulse response (IIR), and second order IIR. In the equation below, a and b represents the systems coefficients. In FIR filters the “A” coefficient is zero while in IIR filters, the “A” coefficient is a nonzero value.

$$y[n] = \frac{1}{a_0} (-a_1y[n - 1] - \cdots - a_Ny[n - N] + b_0x[n] + b_1x[n - 1] + \cdots + b_Mx[n - M])$$

Equation 2: Constant-Coefficient difference equation in terms of $Y[n]$

Materials

During the lab, the following tools and equipment we used:

1. Function generator and oscilloscope
2. Input audio source, e.g. PC workstation, MP3 player, headphones, etc.
3. TI TMS320C5505 eZdsp USB Stick with cable, audio patch cords
1. Lab 3 project files as posted on CourseLink
4. TI Code Composer Studio (CCS) SDK

Procedure

In signal processing, a FIR filter is a filter whose impulse response is of finite duration as it settles to zero in finite time. A FIR filter has several useful properties such as the fact that it does not require any feedback, is inherently stable since the output is a sum of a finite number, can easily be designed to be linear phase, uses a fractional arithmetic and very simple to implement. One disadvantage is that it requires a lot of memory due to a lot of calculations performed during the filtering process. Referring to Equation 2, it is noted that the “A” coefficient equals to 0 since the impulse response has a finite duration. The Basic block diagram of a FIR filter is described below:

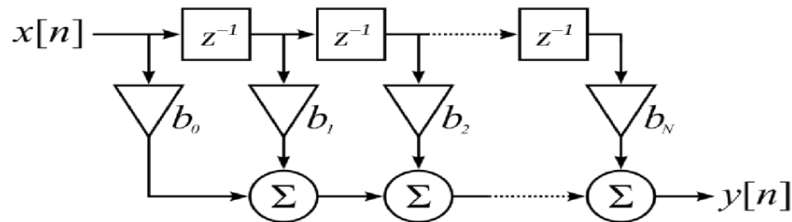


Figure 3: Basic Block Diagram of FIR Filter of Order N

IIR filters are digital filters with an infinite impulse response due to recursive calculations. Unlike the FIR filters, IIR filters have feedback. IIR filters are the most efficient type of filter as they are very easy to implement in digital signal processing due to the low amount of processing required to perform the filtering. In summary, IIR filters require less memory. Cascade of second order IIR filters are used to reduce the quantization error of higher order due to lack of precision with smaller numbers. The following block diagram describes a basic block diagram of an IIR Filter:

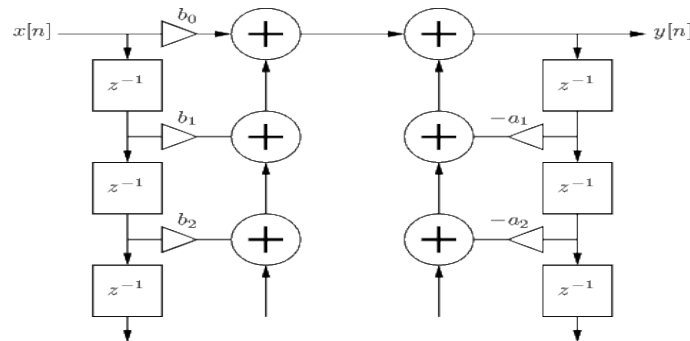


Figure 4: Basic Block Diagram of an IIR Filter

Lab 3 consists of two parts, IIR and FIR filtering. Filtering was completed using a switch statement in C that performed functions which was accessed using the header files IIR_Filter.h and FIR_filter_asm.h.

```

129     switch (Step)
130     {
131         case 1:
132             left_output = mono_input;           // Directly connect inputs to outputs for reference.
133             right_output = mono_input;
134             break;
135
136         case 2:
137             left_output = FIR_filter_asm(FIR_Order5, mono_input);
138             right_output = left_output;
139             break;
140
141         case 3:
142             left_output = FIR_filter_asm(FIR_Order11, mono_input);
143             right_output = left_output;
144             break;
145
146         case 4:
147             left_output = FIR_filter_asm(FIR_Order21, mono_input);
148             right_output = left_output;
149             break;
150
151         case 5:
152             left_output = second_order_IIR_direct_form_I (IIR_2nd_Order, mono_input);
153             right_output = left_output;
154             break;
155
156         case 6:
157             left_output = fourth_order_IIR_direct_form_I(IIR_4th_Order, mono_input);
158             right_output = left_output;
159             break;
160     }
161
162     aic3204_codec_write(left_output, right_output);
163 }
164
165 /* Disable I2S and put codec into reset */
166 aic3204_disable();
167
168 printf( "\n***Program has Terminated***\n" );
169 SW_BREAKPOINT;
170 }
171 }
172

```

Figure 5: Program used to apply filtering

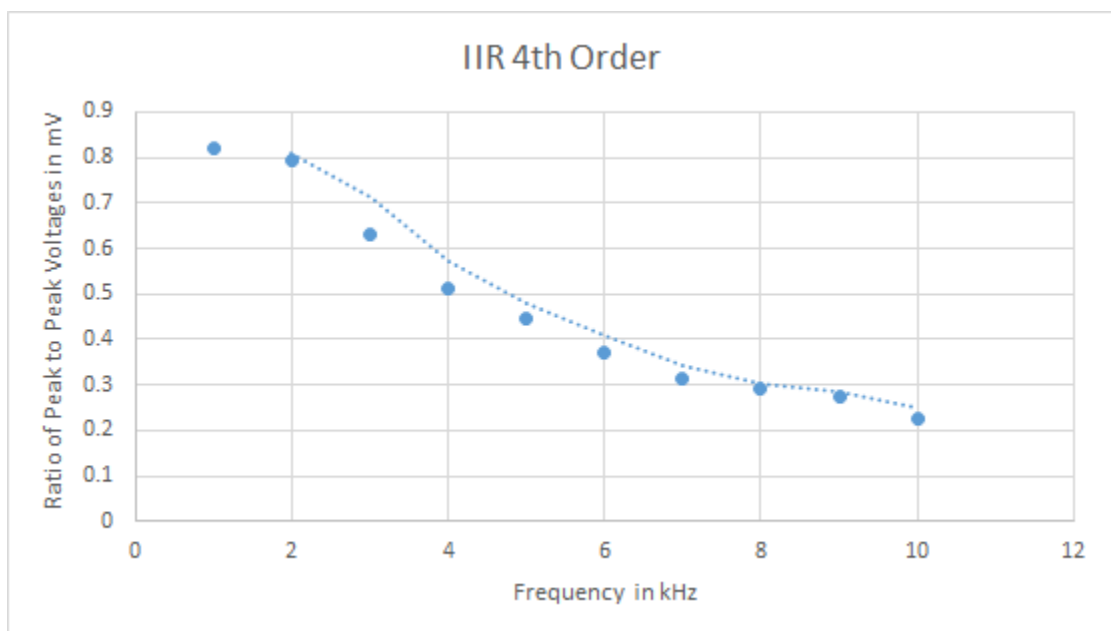
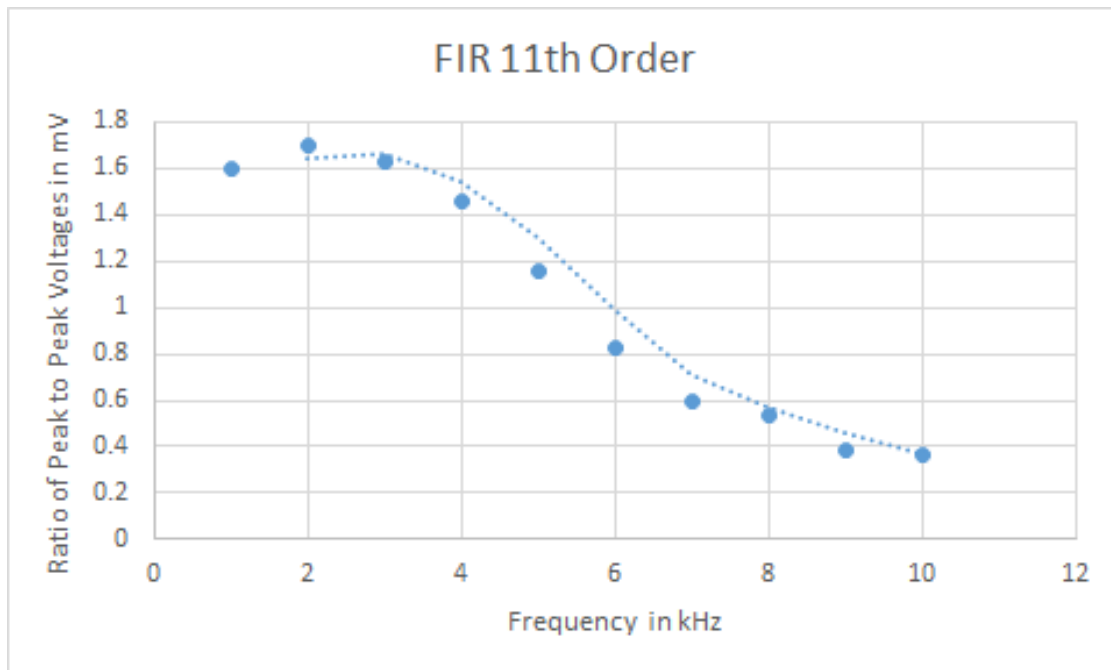
The code in Figure A1 included 3 different FIR Filters with a varying filter length N (Frequency is kept constant at a value of 1 KHz). The code was first executed as an audio source was connected (input). The frequency response plot for FIR filter order 11 is shown in Figure 3. The quality of the audio goes from fuzzy to muffled, to a deeper/ more muffled/ lower sound as

the 3 filters were applied. The function generator and the oscilloscope were later connected to characterize the frequency response of only the FIR filter with an M (N in code) value of 11. The ratio of the output to input peak to peak voltages was compared for the following frequencies: 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000, 10000. These values were obtained using the freqz plot in MATLAB from the Prelab part 1.

The code in figure A2 included 2 different IIR Filters with an N order of 2 and an N order of 4 (Frequency is kept constant at a value of 1KHz). The frequency response plot for IIR filter order 4 is shown in Figure A4. The code was first executed as an audio source was connected (input). The quality of the audio goes from clear high sounds to muffled high and low sounds as the 2 filters were applied. The function generator and the oscilloscope were later connected to characterize the frequency response of only the IIR filter with an N value of 4. The ratio of the output to input peak to peak voltages was compared for the following frequencies: 1000, 1100, 1200, 1300, 1400, 1500, 1600, 1700, 1800, 1900, 2000. These values were obtained using the freqz plot in MATLAB from the Prelab part 2.

Results/ Discussion

Frequency	IIR 4 (4 th order)- Peak to Peak	Ratio of Peak - Peak in mV	Frequency	FIR 11(11 th order)-Peak to Peak	Ratio of Peak - Peak in mV
1100	780mV- 640mV	0.820513	1000	800mV- 1.28mV	1.6
1200	780mV- 620mV	0.794872	2000	660mV- 1.12V	1.696969
1300	760mV -480mV	0.631579	3000	540mV- 880mV	1.629629
1400	740mV -380mV	0.513514	4000	440mV- 640mV	1.454545
1500	720mV -320mV	0.444444	5000	380mV- 440mV	1.157894
1600	700mV -260mV	0.371429	6000	340mV- 280mV	0.823529
1700	700mV- 220mV	0.314286	7000	300mV- 180mV	0.6
1800	680mV- 200mV	0.294118	8000	260mV- 140mV	0.538461
1900	660mV -180mV	0.272727	9000	260mV- 100mV	0.384615
2000	620mV- 140mV	0.225806	10000	220mV- 80mV	0.363636



- The higher the voltage, the clearer and audible the sound is
- The lower the voltage, the choppier and more disturbed the sound is
- As we apply the IIR filter, the peak to peak gets lower
- The peak to peak voltage is basically the quality after filter is applied

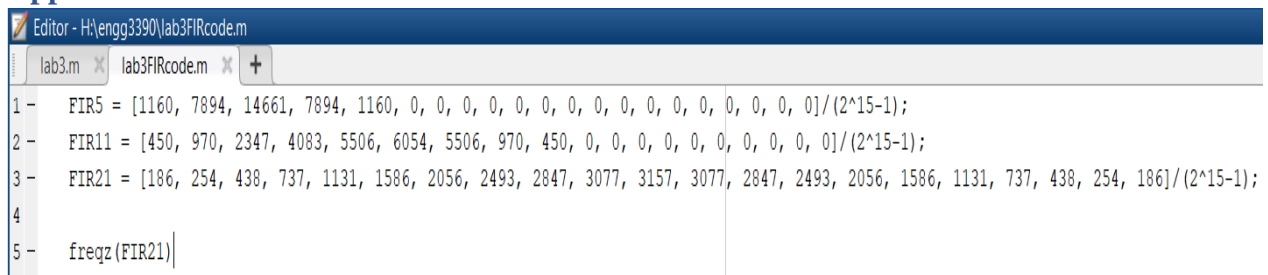
Conclusion

Though this lab, the FIR and IIR filters were successfully used to create a low-pass filter. The observations recorded verify that the low-pass filter was created successfully. Both systems presented in the results section had a similar response to the changes in frequency of the input response. In summary, we learn that IIR Filters seem to have a sharper cut-off frequency, fast computational speed, non-constant phase/delay, and instable compared to FIR filters. The results and the discussion of the experiment proves that FIR filters are more powerful than IIR filters, but also require more processing power and more work to set up the filters. However, their greater power means more flexibility and ability to finely adjust the response of your active loudspeaker. These observations are crucial to understanding the system behind different types of digital filters. Understanding the system is very important in design for applications such as sound filtering, communications, and digital conversions of audio files. Some sources of error that could have happened were mainly due to human error. This includes the song selection and instrumental error.

References:

1. ENGG*3390 Signal Processing Lab 3 Manual (F18)
2. ENGG*3390 Signal Processing Lab-guide (F18)

Appendices

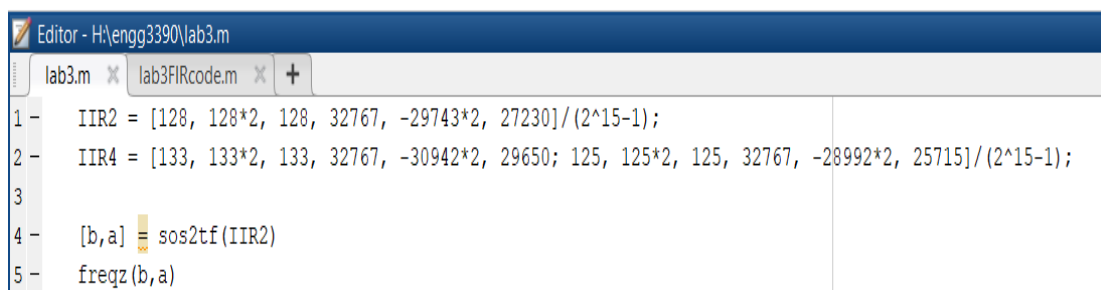


Editor - H:\engg3390\lab3FIRcode.m

lab3.m x lab3FIRcode.m x +

```
1 - FIR5 = [1160, 7894, 14661, 7894, 1160, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]/(2^15-1);
2 - FIR11 = [450, 970, 2347, 4083, 5506, 6054, 5506, 970, 450, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]/(2^15-1);
3 - FIR21 = [186, 254, 438, 737, 1131, 1586, 2056, 2493, 2847, 3077, 3157, 3077, 2847, 2493, 2056, 1586, 1131, 737, 438, 254, 186]/(2^15-1);
4
5 - freqz(FIR21)
```

Figure A1: FIR code



Editor - H:\engg3390\lab3.m

lab3.m x lab3FIRcode.m x +

```
1 - IIR2 = [128, 128*2, 128, 32767, -29743*2, 27230]/(2^15-1);
2 - IIR4 = [133, 133*2, 133, 32767, -30942*2, 29650; 125, 125*2, 125, 32767, -28992*2, 25715]/(2^15-1);
3
4 - [b,a] = sos2tf(IIR2)
5 - freqz(b,a)
```

Figure A2: IIR code

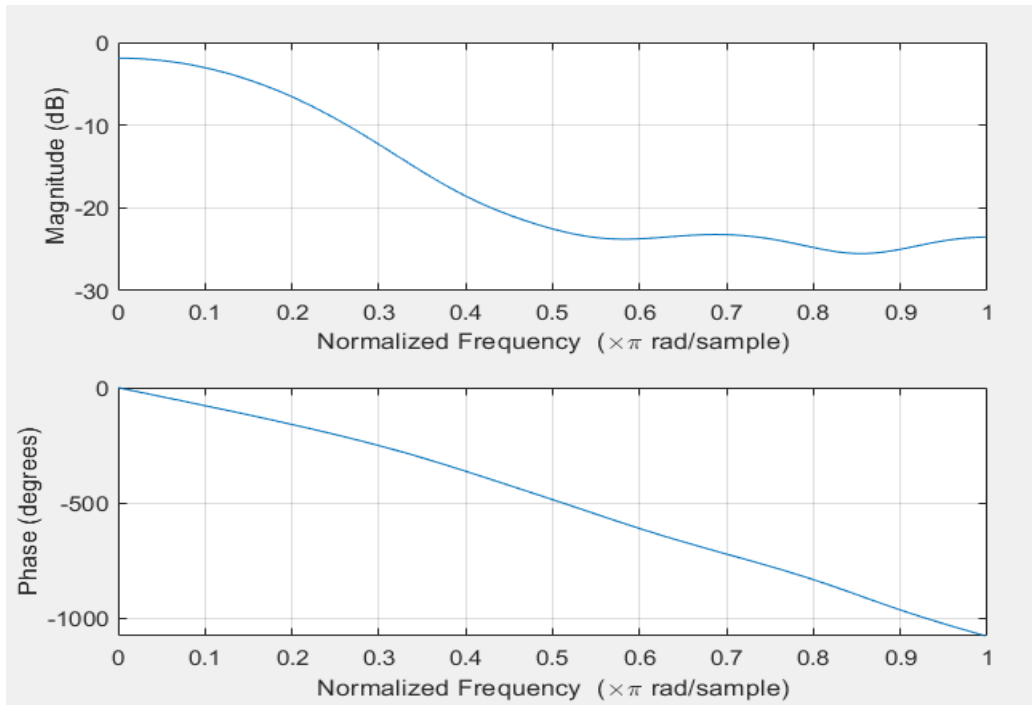


Figure A3: FIR order 11 frequency response plot

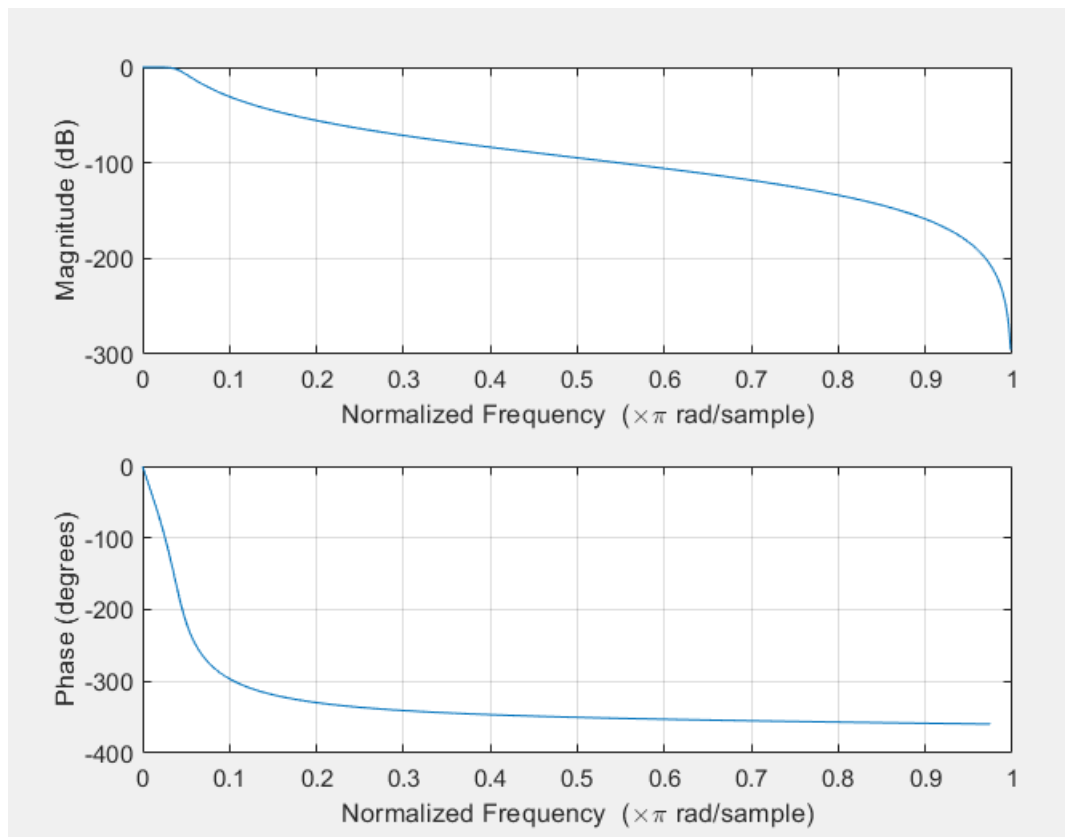


Figure A4: IIR order 4 frequency response plot

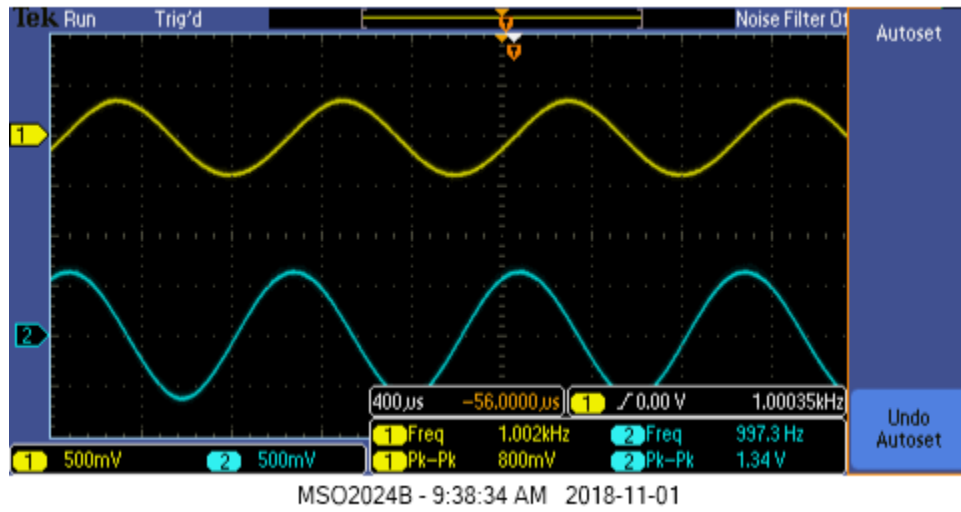


Figure A5: FIR order 5

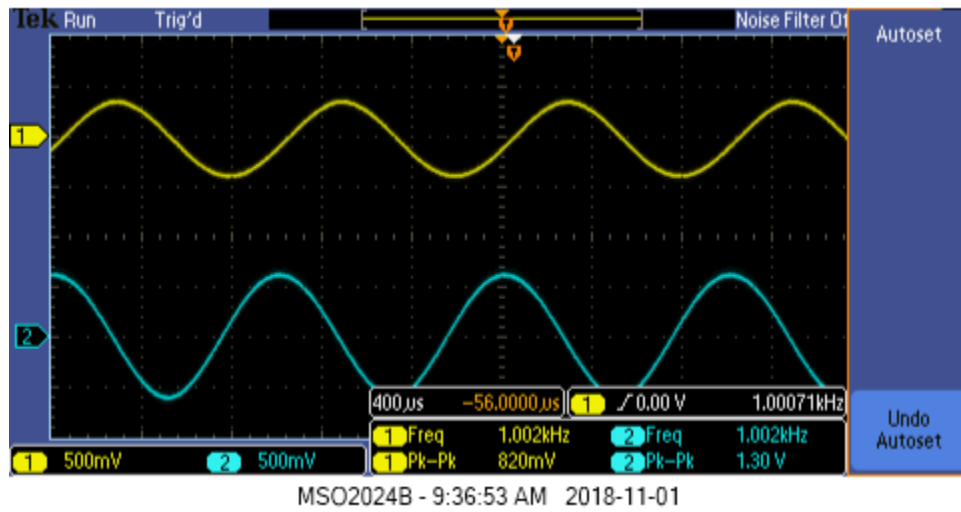


Figure A6: FIR Order 11

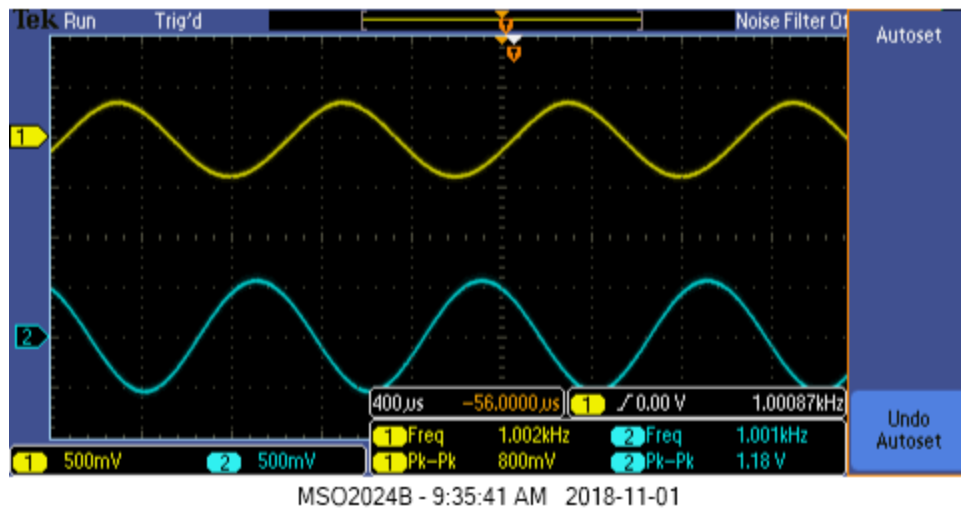


Figure A7: FIR order 21

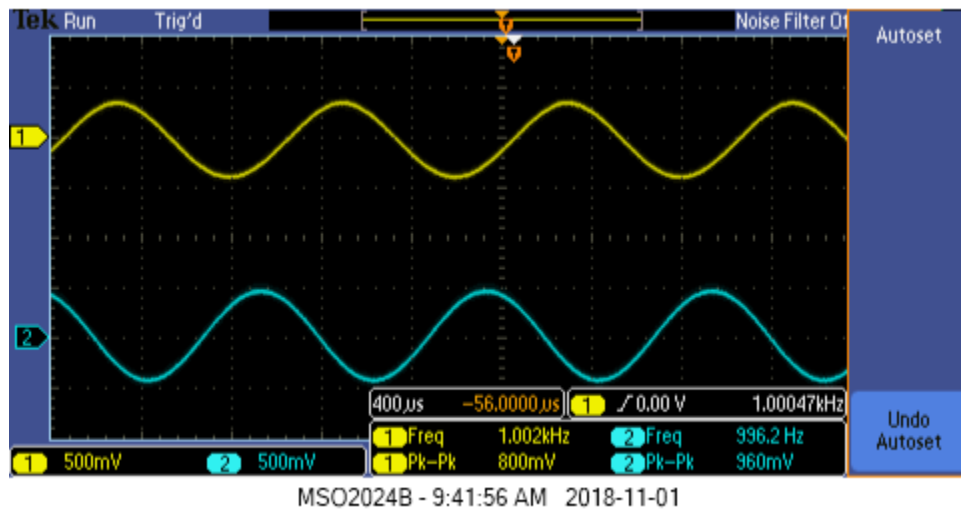
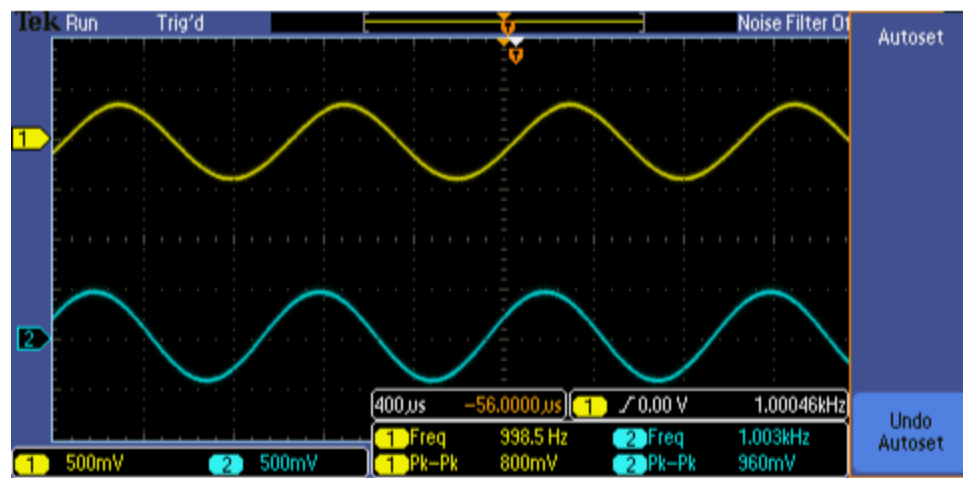


Figure A8: IIR order 2



MSO2024B - 9:43:34 AM 2018-11-01

Figure A9: IIR Order 4