Real Time Digital Signal Processing ECE 4703

Lab 3: Implementation of an IIR Filter

Group 10: Marcos Delgado (Github ID: marcosdelgadorosa), Brett Mann (Github ID: BrettMann225)

November 16th, 2025

Introduction

Infinite impulse response (IIR) filters use a combination of feedforward and feedback to isolate a specific passband range while attenuating everything else. These filters can be designed for many applications and are widely used. In this lab we explore the design of an IIR filter to discriminate between DTMF frequencies. The DTMF column frequencies are 1209 Hz, 1336 Hz, 1477 Hz and 1633 Hz and in this lab we will design a filter that will pass only the 1336 Hz tone and attenuate the other three.

Methods

This lab utilized a provided DSP kit consisting of a RPI4 board and a shielded PCB with a high-end DAC and ADC. An Analog Discovery 2 USB scope was used to monitor signals. The appropriate software was installed on all the hardware. An SSH connection was used to access the RPI4 and run necessary programs such as pipewire. Additionally our filter parameters were calculated by MATLAB's filter designer.

Results

For this lab, we want to design an IIR filter that single out 1336 Hz among the frequencies 1209 Hz, 1477 Hz and 1633 Hz. In order to achieve this we decided on a Butterworth bandpass filter with a passband centered on 1336 Hz and ideally we want the length to be at least 10% of the frequency, 5% to either side. Since 5% of 1336 is 66.8, we want our passband to be from about 1269.2 Hz to about 1402.8 Hz. Therefore, we decided that the passband should be from 1260 Hz to 1410 Hz. The stopband must be at the neighboring frequencies since we need them to be fully attenuated, meaning the lower stopband starts at 1209 Hz, and the upper stopband starts at 1477 Hz. Additionally, to ensure that the attenuated signal is at an acceptable noise level, the rejection must be of at least 60 dB. The ripple in the passband should not exceed 0.5 dB. Using MATLAB filter designer to produce coefficients yields a filter of 15 second order sections.

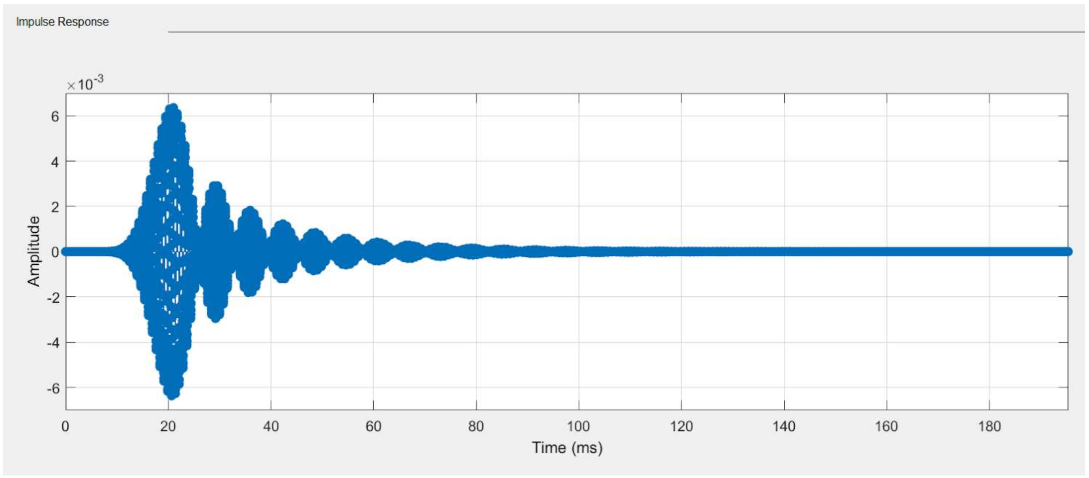


Figure 1. Impulse response of IIR filter. by MATLAB

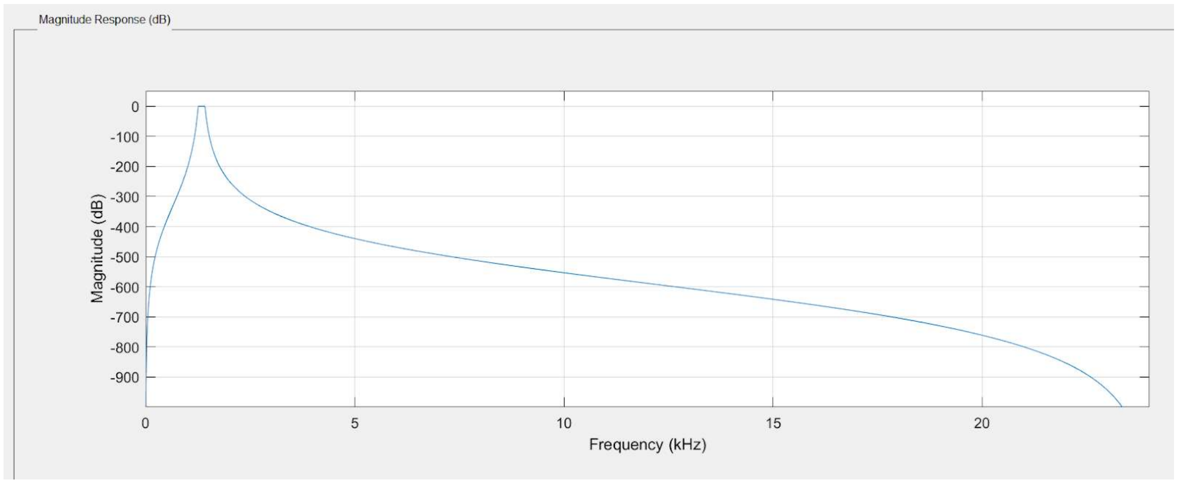


Figure 2. Magnitude response of IIR filter by MATLAB.

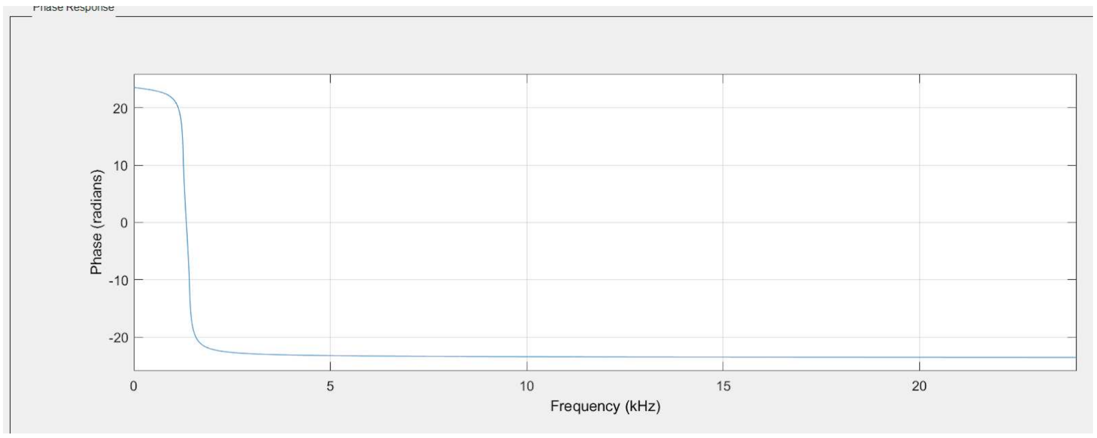


Figure 3. Phase response of IIR filter by MATLAB.

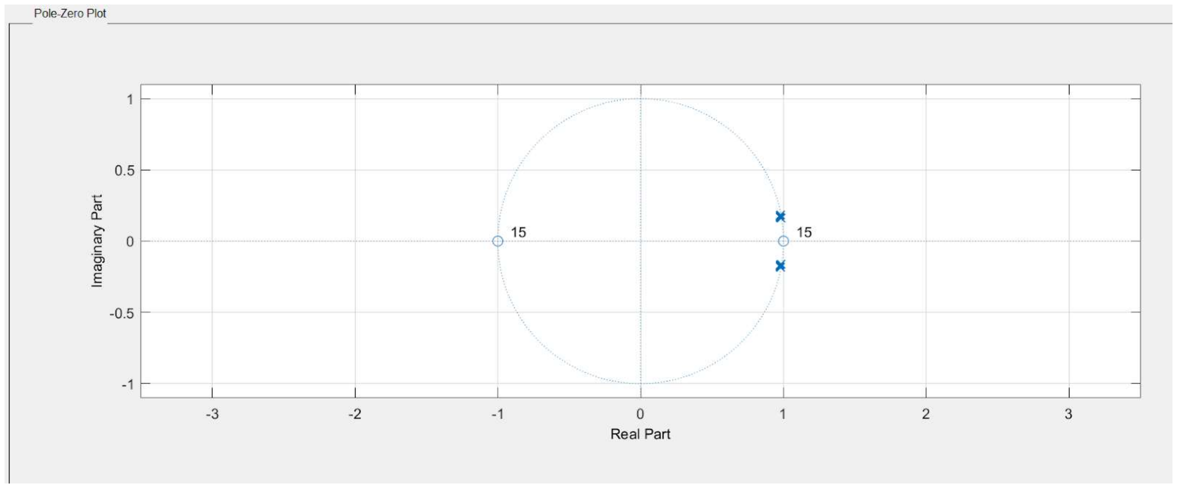
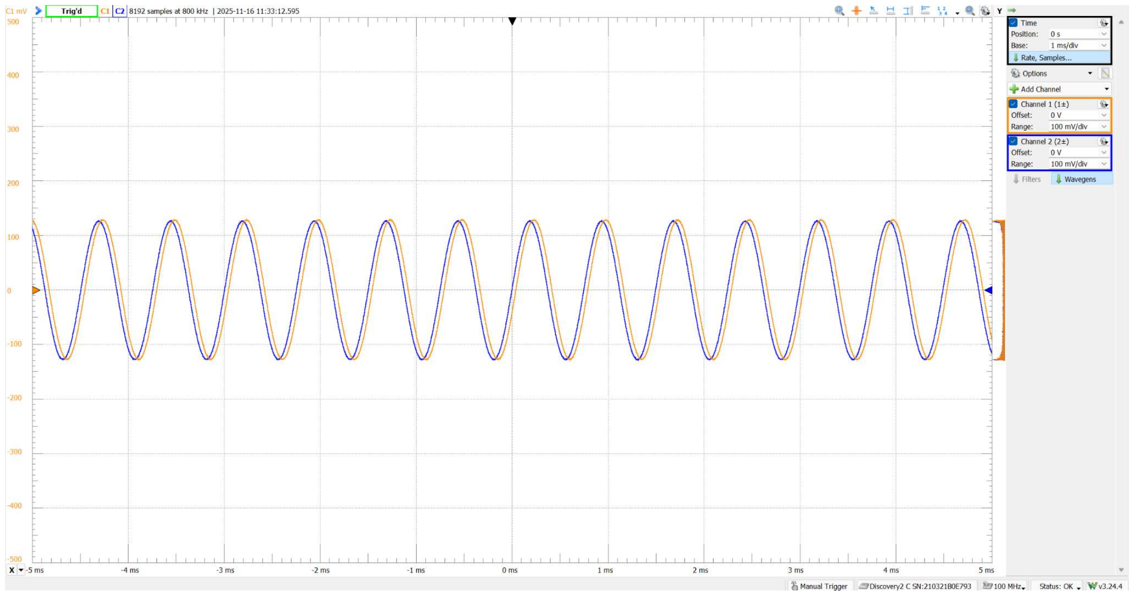


Figure 4. Pole-zero plot of IIR filter generated by MATLAB. There are 30 poles and 30 zeroes which are clustered in tight groups.

Figure 1 below shows the filter's theoretical impulse response. It can be noted that, true to the name, the impulse response is indeed infinite. Figure 2 depicts the magnitude response as calculated by matlab, we can see that the passband is flat only until the designated frequency and then there is sharp attenuation. Figure 3 and figure 4 depict the phase response and pole-zero plot respectively. The pole-plot depicts many poles and zeros bunching closely together.

After implementing a filter using C, the results can be viewed. Figure 5 shows how the center frequency, 1336 Hz, is not only negligibly attenuated, it is also negligibly phase shifted. This is ideal as we want this frequency to be the clearest.


C1 mV | Trig'd | C1 | C2 | 8192 samples at 800 kHz | 2025-11-16 11:33:12.595

Time
Position: 0 s
Base: 1 ms/div

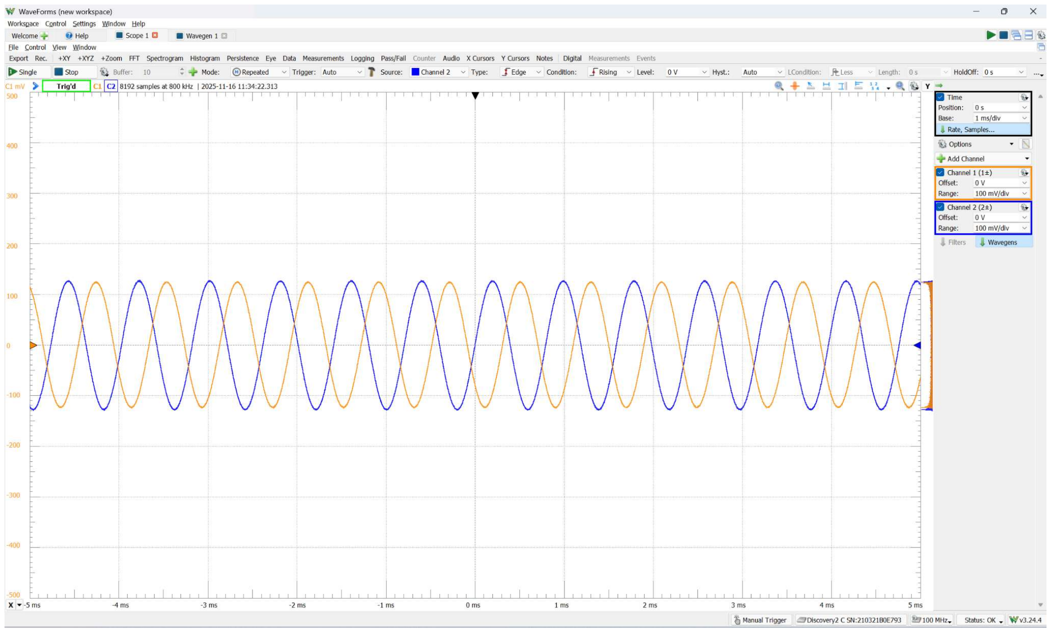
Channel 1 (1±)
Offset: 0 V
Range: 100 mV/div

Channel 2 (2±)
Offset: 0 V
Range: 100 mV/div

Manual Trigger | Discovery2 C SN:210321B0E793 | 100 MHz | Status: OK | v3.24.4

Figure 5. Filtered left channel (orange) and unfiltered right channel (blue) at 1336 Hz.

Figures 6 and 7 below show the borders of the passband, 1260 Hz and 1410 Hz respectively. The filtered singal is only very slightly attenuated but more noticeably shifted. The attenuation is still well within the acceptable range. Figure 8 shows the filter response to a frequency in the stopband, 1477 Hz. It can be clearly seen how the filter attenuates the stopband frequency all the way down to the noise level. These results show that the filter is able to reduce frequencies in the stopband to the noise level while preserving those in the passband, letting us differentiate between the DTMF frequencies.


WaveForms (new workspace)
Workspace Control Settings Window Help
Welcome Help Scope 1 Wavegen 1
File Control View Window
Export Rec. +XY +XYZ +Zoom FFT Spectrogram Histogram Persistence Eye Data Measurements Logging Pass/Fail Counter Audio X Cursors Y Cursors Notes Digital Measurements Events
Single Stop
Trig'd C1 C2
C1 mV
Time
Options
Add Channel
Channel 1 (1±)
Channel 2 (2±)
Filters Wavegens
Manual Trigger
500 400 300 200 100 0 -100 -200 -300 -400 -500
-5 ms -4 ms -3 ms -2 ms -1 ms 0 ms 1 ms 2 ms 3 ms 4 ms 5 ms

Buffer: 10 Mode: Repeated Trigger: Auto Source: Channel 2 Type: Edge Condition: Rising Level: 0 V Hyst.: Auto LCondition: Less Length: 0 s HoldOff: 0 s

8192 samples at 800 kHz | 2025-11-16 11:34:22.313

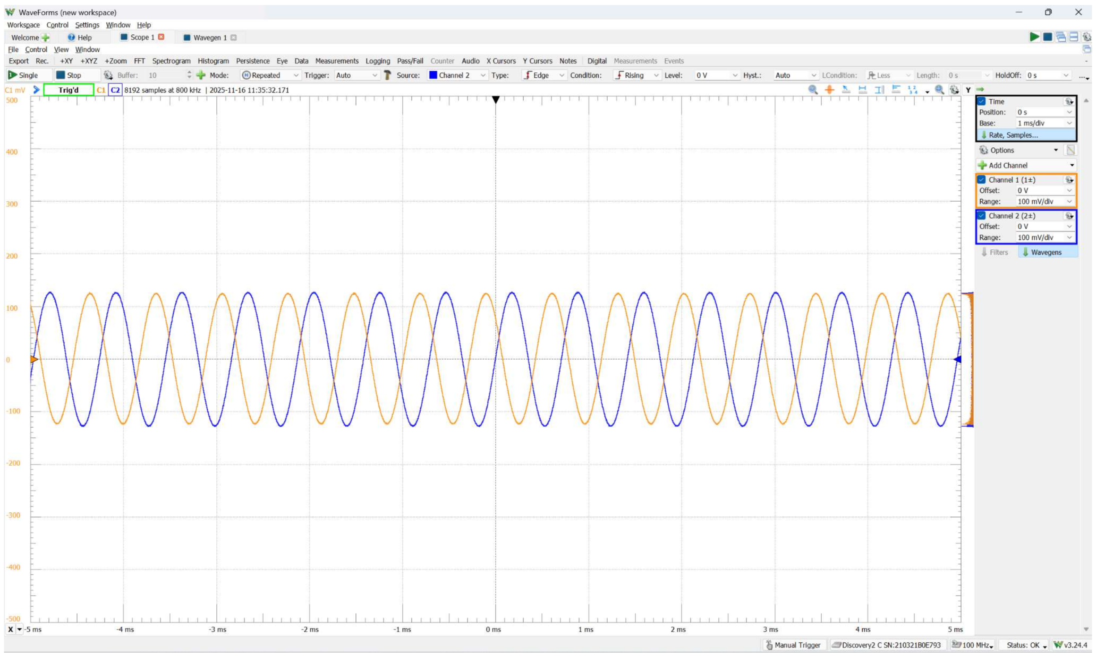
Time — Position: 0 s, Base: 1 ms/div, Rate, Samples...

Channel 1 (1±) — Offset: 0 V, Range: 100 mV/div

Channel 2 (2±) — Offset: 0 V, Range: 100 mV/div

Discovery2 C SN:210321B0E793 100 MHz Status: OK v3.24.4

Figure 6. Filtered left channel (orange) and unfiltered right channel (blue) at the lower limit of the passband, 1260 Hz.


WaveForms (new workspace)
Workspace Control Settings Window Help
Welcome Help Scope 1 Wavegen 1
File Control View Window
Export Rec. +XY +XYZ +Zoom FFT Spectrogram Histogram Persistence Eye Data Measurements Logging Pass/Fail Counter Audio X Cursors Y Cursors Notes Digital Measurements Events
Single Stop
Trig'd C1 C2
C1 mV
Time
Options
Add Channel
Channel 1 (1±)
Channel 2 (2±)
Filters Wavegens
Manual Trigger
500 400 300 200 100 0 -100 -200 -300 -400 -500
-5 ms -4 ms -3 ms -2 ms -1 ms 0 ms 1 ms 2 ms 3 ms 4 ms 5 ms

Buffer: 10 Mode: Repeated Trigger: Auto Source: Channel 2 Type: Edge Condition: Rising Level: 0 V Hyst.: Auto LCondition: Less Length: 0 s HoldOff: 0 s

8192 samples at 800 kHz | 2025-11-16 11:35:32.171

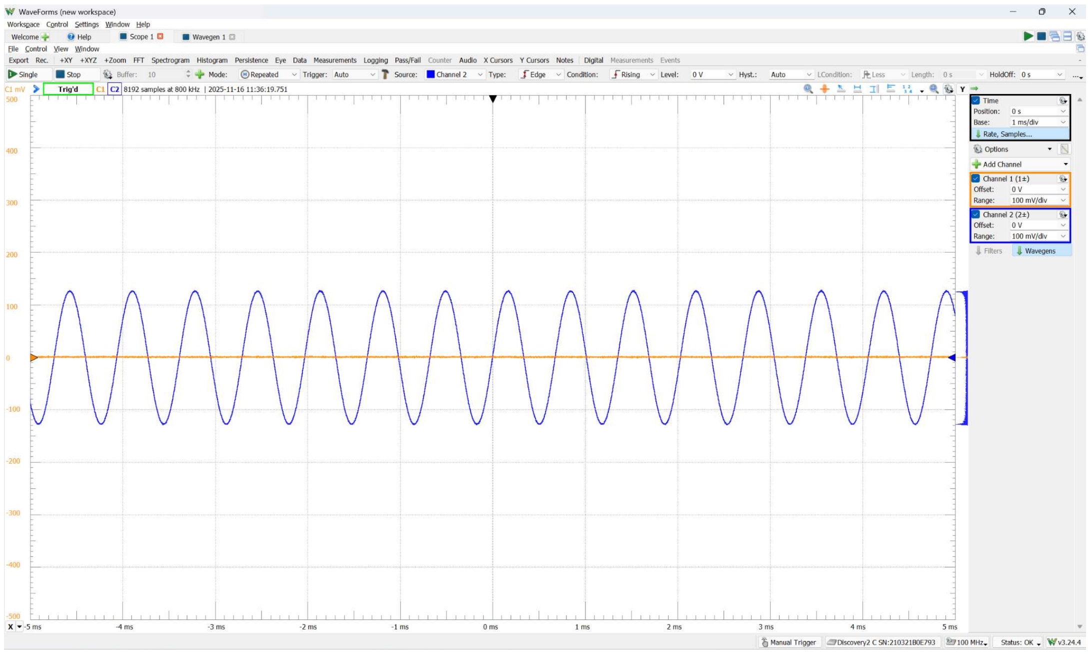
Time — Position: 0 s, Base: 1 ms/div, Rate, Samples...

Channel 1 (1±) — Offset: 0 V, Range: 100 mV/div

Channel 2 (2±) — Offset: 0 V, Range: 100 mV/div

Discovery2 C SN:210321B0E793 100 MHz Status: OK v3.24.4

Figure 7. Filtered left channel (orange) and unfiltered right channel (blue) at the upper limit of the passband, 1410 Hz.


WaveForms (new workspace)
Workspace Control Settings Window Help
Welcome Help Scope 1 Wavegen 1
File Control View Window
Export Rec. +XY +XYZ +Zoom FFT Spectrogram Histogram Persistence Eye Data Measurements Logging Pass/Fail Counter Audio X Cursors Y Cursors Notes Digital Measurements Events
Single Stop
Trig'd C1 C2
C1 mV
500 400 300 200 100 0 -100 -200 -300 -400 -500
-5 ms -4 ms -3 ms -2 ms -1 ms 0 ms 1 ms 2 ms 3 ms 4 ms 5 ms
Options Add Channel Filters Wavegens Rate, Samples...
Manual Trigger

Buffer: 10 Mode: Repeated Trigger: Auto Source: Channel 2 Type: Edge Condition: Rising Level: 0 V Hyst.: Auto LCondition: Less Length: 0 s HoldOff: 0 s

8192 samples at 800 kHz | 2025-11-16 11:36:19.751

Time — Position: 0 s, Base: 1 ms/div

Channel 1 (1±) — Offset: 0 V, Range: 100 mV/div

Channel 2 (2±) — Offset: 0 V, Range: 100 mV/div

Discovery2 C SN:210321B0E793 100 MHz Status: OK v3.24.4

Figure 8. Filtered left channel (orange) and unfiltered right channel (blue) at 1477 Hz.

Having constructed the filter, we want to compare the impulse to response to the theoretical impulse response we attained from MATLAB. However, the impulse response can’t faithfully be recreated in practice as an impulse would need to be infinitely tall and infinitely quick, so it must instead be approximated. One method for such approximations is to use a pulse generator. Figure 9 shows the result of the fourier transform after using a 20Hz pulse as input. We notice that instead of the smooth passband generated by MATLAB, we see specific frequencies being singled out. This makes sense because a pulse is a type of square wave, which is generated as a series of harmonics. The specific spikes in figure 9 are the harmonics that fall in the passband.

Another method for approximation is to use a high voltage broadband noise signal. This simulates the equal presence of all frequencies and provides improved results for the magnitude response, however the phase response will be random and meaningless under this approximation. Figure 10 shows the result of this approximation using a 20 kHz noise. We can immediately notice an improvement from the magnitude response of figure 9 as the pass band is straight and about the same magnitude at all frequencies within it. However it does not demonstrate the 60dB difference in attenuation which was promised by the filter designer meaning that while the overall shape is more similar, the approximation is still incorrect.

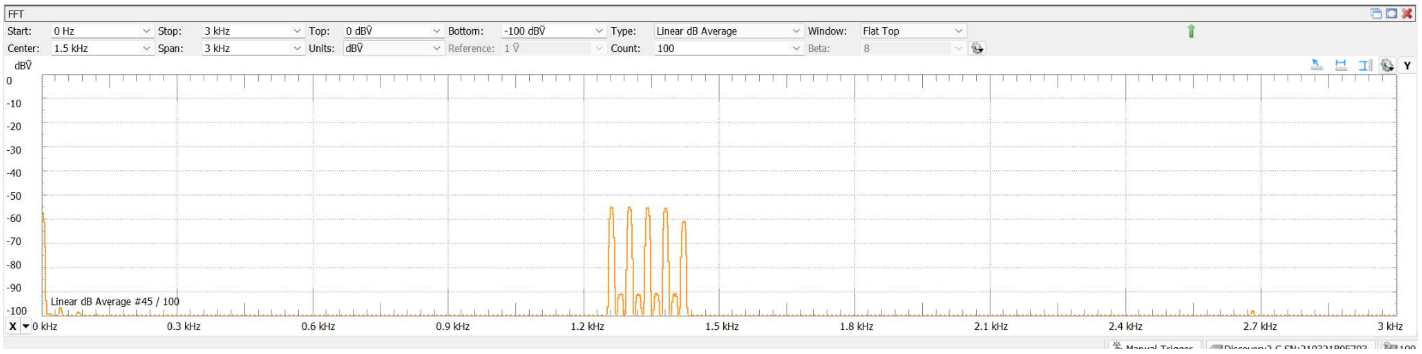


Figure 9. Fourier transform with input signal of 20 Hz, 1 V pulse.

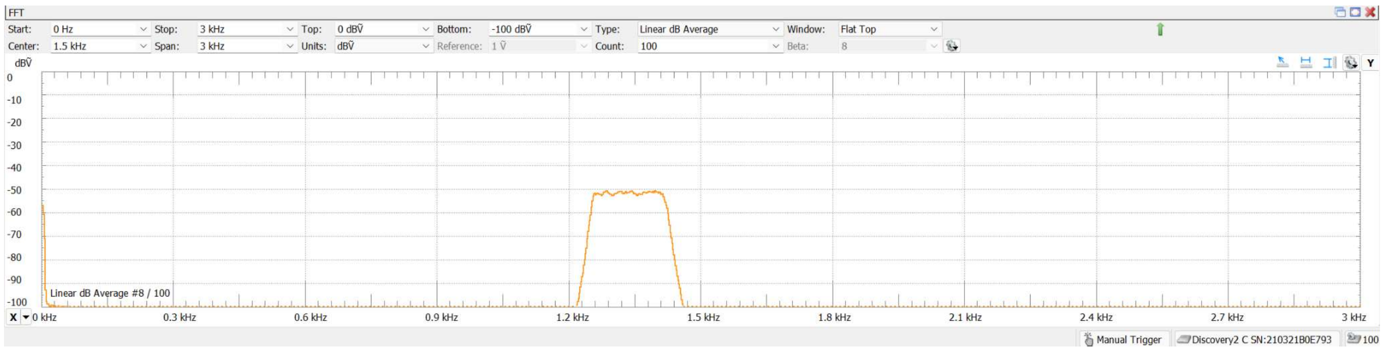


Figure 10. Fourier transform with input signal of 20 kHz, 1 V noise.

Lastly, we tested one final approximation method, that being to generate the random noise internally rather than send it through the ADC. This avoids the filters within the ADC and can therefore give better results. Figure 11 depicts the result of this method. Comparing it to the previous two we can see that like figure 10, it has a flat passband. We can also noticed a much improved rejection, now spanning the full 60 dB. Comparing it to matlab, we notice that although the peak of the signal matches closely, the rejection at all other frequencies doesn't drop below -100 dB whereas MATLAB shows it decreasing rapidly to well below -400 dB. This is likely due to the noise level of the RPI itself, making it impossible to go below that threshold. That being said, the quality of the fourier transform is enough for us to conclude that the filter works as intended and has the desired passband and stopband parameters.

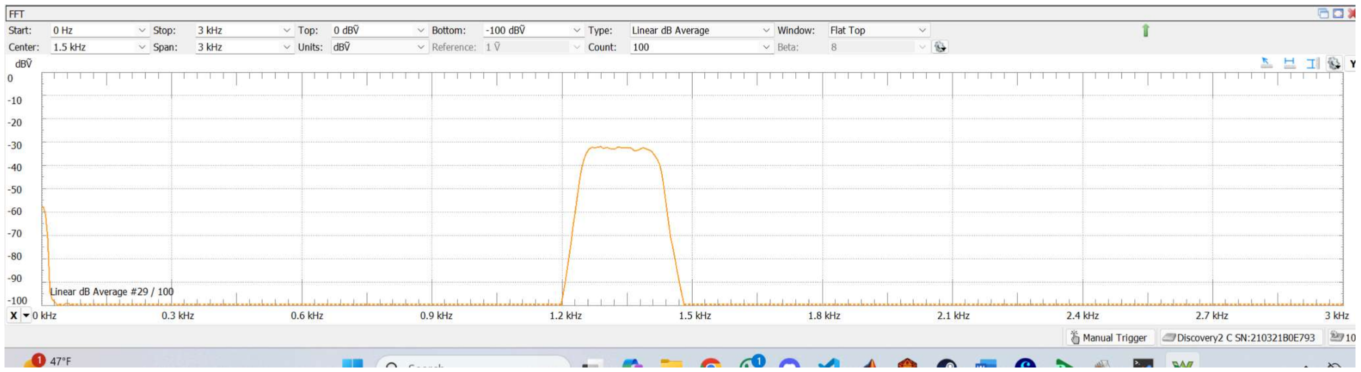


Figure 11. Fourier transform with input signal of program-generated.

Conclusion

Using the MATLAB filter designer, we were able to create a bandpass filter that effectively isolated a single DTMF frequency from the other column frequencies. The filter is able to reject unwanted frequencies down to the noise level while cleanly preserving the amplitude of passband frequencies. Although it is difficult to check the magnitude response of the filter, the best result is achieved when using an internally generated noise in order to bypass the filters of the ADC.

References

No references were used in the completion of this lab.