ELEC 459 Digital Signal Processing III: Project Report

Chulwoong Jeon(0232515)

cwjeon@ece.uvic.ca

Dept. of Electrical and Computer Engineering

University of Victoria

Contents

Digital Audio Filter

1. Introduction

Motivation

I am looking at a sound waveform, horizontal axis expressing discrete time and vertical axis showing discrete magnitude values. What information does this signal contain, how much information is there? Is there any hidden information behind a certain frequency band? How can I understand why it makes this unique sound? Questions like these have aroused my curiosity and have lead to implementing this GUI – 'Digital Audio Filter'.

'Digital Audio Filter' is a tool that is running under 'Window98/2000/Xp'. It analyzes a sound waveform using FIR filters and FFT analysis. Various signals processing software exists for the digital signal analysis. However, very few applications exist that are tailored for the purpose of digital audio signal analysis. For that reason, I have implemented several algorithms for FIR filtering and FFT analysis into one software application which can be readily used by musicians, engineers or anyone interested in analyzing audio signals from a signal processing point of view.

The 'Digital Audio Filter' is a runtime FIR filter/FFT analysis tool and I did this project in terms of applying all the knowledge that I have learned in class to a real application. If you are not familiar with using the 'Digital Audio Filter', Appendix A will explain how to use 'Digital Audio Filter' so that I strongly recommend you to read Appendix A first before you read this document.

2. Signal Processing Techniques

In this section, I am going to show all the techniques to analyze/filter the audio signals for this GUI.

2.1. DFT/FFT

The process of solving for sine and cosine components of a signal is called 'Fourier analysis', or 'Fourier transform'. If the frequency variable is sampled (in Fourier series, represented as index 'k') and the time variable is sampled, too (in time domain, represented as index 'n'), then the 'Fourier transform is called as 'Discrete Fourier Transform'. Below the Continuous Fourier Transform and Discrete Fourier Transform are shown

Continuous Fourier Transform

$$X(\omega) = \int_{-\infty}^{\infty} x(t) e^{-j\omega t} dt \quad (2.1)$$

Discrete Fourier Transform

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi kn/N} \quad \text{for} \quad 0 \le k \le N-1 \quad (2.2)$$

where N is the length of the signal being analyzed.

The FFT is a faster way to compute the DFT. It is more efficient than trying to calculate DFT directly from the definition. In order to achieve less computation than DFT, FFT can only be computed for signals whose lengths are exactly powers of two. Even though FFT gives us this advantage that is less computation than DFT, the powers of two causes several problems in terms of software implementation. To satisfy the condition the sample length should be powers of two, the signals have to be chopped with smaller size (powers of two), or if the length of signals is shorter than powers of two, the signals have to be padded with zeros to match the next powers of two length on the end of signals.

2.2. Zero Padding

As mentioned, FFT has a problem to implement software. To get the correct performance of FFT, the length of samples must be powers of two. To satisfy that condition, Zero padding was used for this GUI.

Let's assume $F(k) = \sum_{n=0}^{N-1} f(n) e^{-j2\pi kn/N}$ for $0 \le k \le N-1$. New signals $g(n)$ are defined as

$$\begin{aligned} g(n) &= f(n) \quad \text{for } 0 < n < N \\ &= 0 \quad \text{for } N \le n \le PN \end{aligned}$$

P is 'Zero padding factor'.

then

$$\sum_{n=0}^{PN-1} g(n) e^{-j2\pi kn/PN} = \sum_{n=0}^{N-1} f(n) e^{-j2\pi kn/PN} + 0$$

$$= F(k/P) \quad 0 \le k \le PN \qquad (2.3)$$

As you can see, new signals g(n) are composed of the original signals f(n) and (PN – N) of zeros. It keeps the original signals and adds extra zeros. From this method, it can satisfy FFT compute condition for the signals whose length is not 'Power of two' and yields a spectrum that is higher resolution than nonzero padded samples.

2.3. Windowing

To have correct FFT analysis, the Windowing was used.

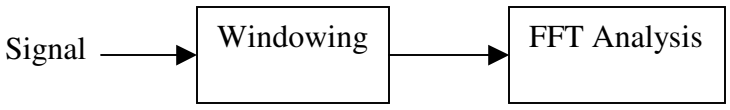

Fig 2.1 Short Time FFT Signal Processing Block

To remove 'Gibbs phenomenon', 'Windowing' is used for Short Time FFT and FIR filtering. Gibbs phenomenon occurs because of rectangular windowing. I will show the spectral characteristics when a signal is

windowed by a simple rectangular window. The truncation of the 'ideal' impulse response $h_d[n]$ to h[n] is form of rectangular windowing.

$$h[n] = h_d[n]w_r[n], \qquad w_r[n]=1 \text{ for } 0\le n\le N$$
$$=0 \quad \text{otherwise}$$

The Fourier transform of $w_r[n]$ is

$$Wr(\omega) = \sum_{n=0}^{N-1} e^{-j\omega n} = \frac{1-e^{-j\omega N}}{1-e^{-j\omega}} = \frac{\sin(0.5\omega N)}{\sin(0.5\omega)} e^{-j0.5\omega(N-1)} \quad (2.4)$$

and Dirichlet kernel is

$$D(\omega, N) = \frac{\sin(0.5\omega N)}{\sin(0.5\omega)} \quad (2.5)$$

The Dirichlet kernel causes distortions that are characterized by the main and side-lobe widths. Each sample in the time domain will render a sinc function in the frequency domain causing side effects in the form of spectral smearing due to the finite main-lobe width and side-lobe interference produced by neighboring samples in the signal.

As I described above, the windowing plays an important role in Short Time Fourier analysis. The main purpose of windowing is to taper off the abrupt end points of the rectangular window achieving gradual transition. This brings minimized side-lobes magnitudes and minimized main-lobe widths. The below is the window characteristics that are used in this GUI.

von Hann window : The von Hann window also known as the Hanning window achieves a side-lobe reduction by superposition. Three Dirichlet kernels are shifted and added together resulting in partial cancellation of the side-lobes. The amount of shift is 2pi/(N-1) from the center. The resulting characteristics of the von Hann window which is sometimes called the cosine window has a side-lobe level of -32 dB and a main lobe width of 8pi/N where N is the number of samples

Hamming window : The Hamming window is similar to the von Hann window with modifications in weighting the Dirichlet kernels. The time and frequency domain windows are as follows. The main-lobe is 8pi/N with

-43dB side-lobes. One characteristic is the non-zero values at both end points and therefore is sometimes referred to as the half-raised cosine window. N is the number of samples.

Blackmann window: The Blackmann window has a -57dB side-lobe and a main-lobe width of 12pi/N. N is the number of samples.

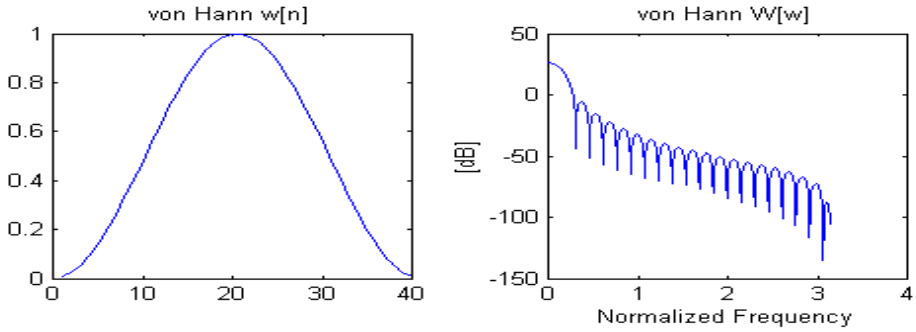

Fig 2.2 ***von Hann Window*** **(window length = 40)**

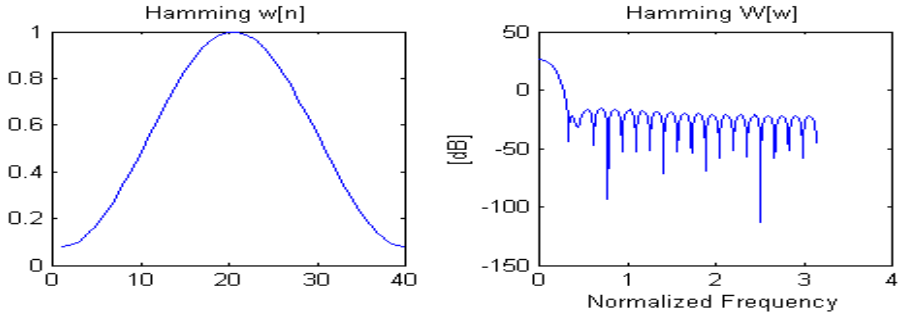

Fig 2.3 ***Hamming Window*** **(window length = 40)**

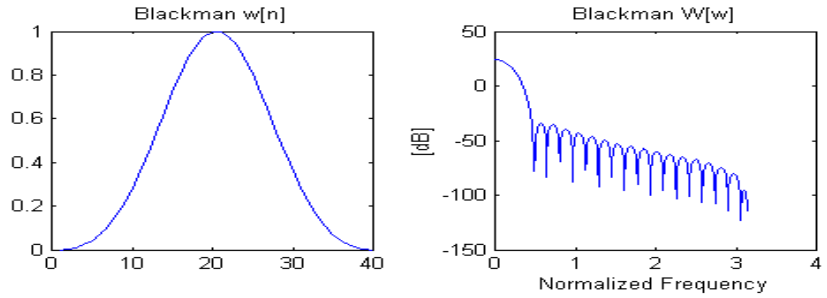

Fig 2.4 ***Blackman Window*** **(window length = 40)**

2.4. FIR Filter

Fig 2.5 shows a signal processing block diagram of the FIR filter that is used for this GUI.

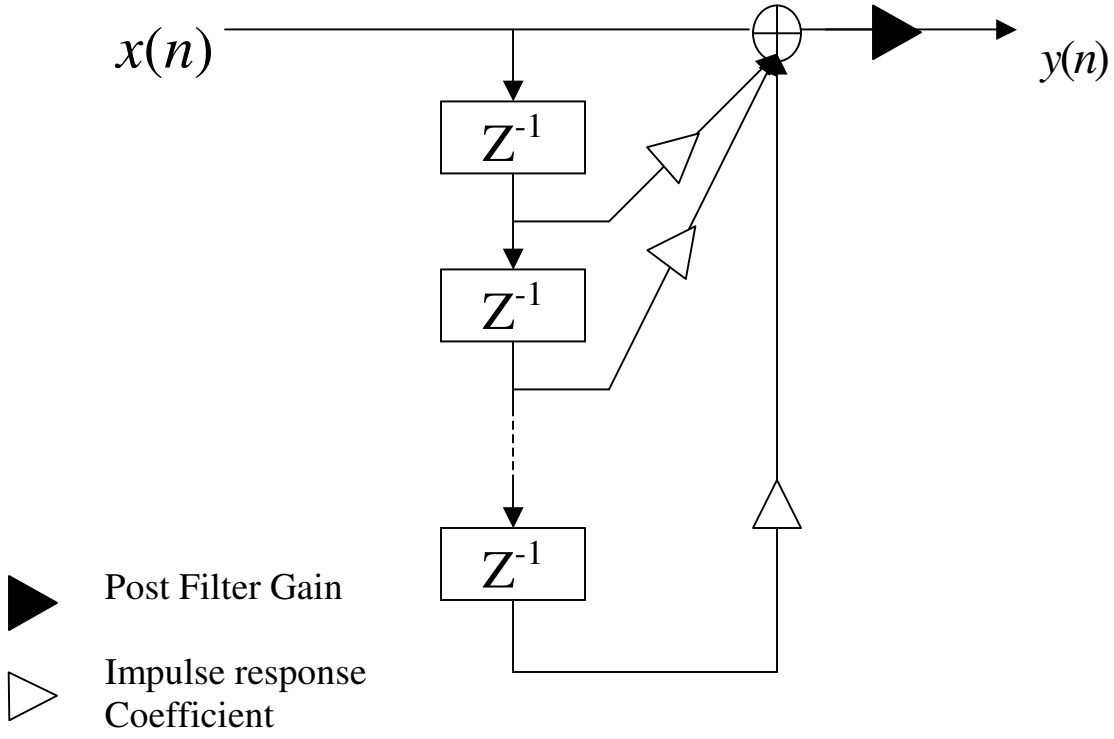

Fig 2.5 ***N^{th} order FIR digital filter***

A common analysis tool for digital filter is the Z transform representation. Z^{-1} is defined as a single sample delay. To transform a filter using the Z transform, capitalize all variables x and y, and replace all time indices with Z^{-1}.

As you can see, the Fig2.5 is nonrecursive filter and shows any impulse response can be stored into the coefficients of a FIR filter and the operation of the filter performs the convolution. Any LTI system with finite length impulse response can be modeled by a FIR filter, provide that impulse response of the LTI system is bandlimited to the Nyquist frequency.

This GUI offers LOWPASS, HIGHPASS, BANDPASS, BANDSTOP and Two Band Pass filters and the ideal amplitude responses of each filter are described at Appendix A.3. 'The Filter Parameter Definition in GUI ' The below are the ideal impulse response of each filter and they are odd number length filters and the middle point of its length is M =(N-1)/2. The variable 'g' means the post filter gain.

***LOWPASS*:**

$$h_d[n] = g\frac{\sin((n-M)\omega_c)}{(n-M)\pi} \quad \text{for } n \neq M$$

$$= g\frac{\omega_c}{\pi} \quad \text{for } n = M$$

***HIGHPASS*:**

$$h_d[n] = -g\frac{\sin((n-M)\omega_c)}{(n-M)\pi} \quad \text{for } n \neq M$$

$$= g(1-\frac{\omega_c}{\pi}) \quad \text{for } n = M$$

***BANDPASS*:**

$$h_d[n] = g\frac{\sin((n-M)\omega_{c2}) - \sin((n-M)\omega_{c1})}{(n-M)\pi} \quad \text{for } n \neq M$$

$$= g\frac{(\omega_{c2} - \omega_{c1})}{\pi} \quad \text{for } n = M$$

***BANDSTOP*:**

$$h_d[n] = g\frac{\sin((n-M)\omega_{c1}) - \sin((n-M)\omega_{c2})}{(n-M)\pi} \quad \text{for } n \neq M$$

$$= g(1 - \frac{(\omega_{c2} - \omega_{c1})}{\pi}) \quad \text{for } n \neq M$$

***Two Band Pass*:** In order to achieve this filter, two of band pass filters are connected in parallel (see fig 2.6). The parallel connection of the filters is used to reduce the harmful effects of the filters. Not only the magnitude frequency response, but also phase responses have to be considered to avoid the harmful effects. For each filter I add in series, its phase response is added to the phase response of the other filter. The phase response also reveals how the filter actually delays the signals. If I add 15 or 31 filters in series, because of the delay of each filter, it will cause distortion.

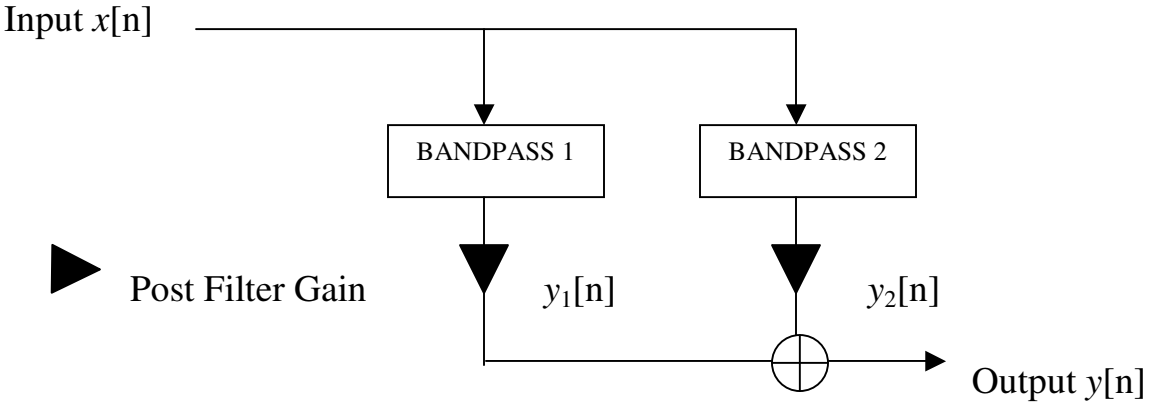

Fig 2.6 Parallel Connection Example

From Fig 2.6, If we put 'BANDPASS 1' as $h_1[n]$ and 'BANDPASS 2' as $h_2[2]$, we can yield the next

$$y_1[n] = x[n] * h_1[n]$$

and

$$y_2[n] = x[n] * h_2[n]$$

The output of Fig 2.6 is

$$y[n] = x[n] * h_1[n] + x[n] * h_2[n]$$

Therefore the output is

$$y[n] = x[n] * (h_1[n] + h_2[n]) \quad (2.6)$$

From equation (2.6), we can get the 'Two Band Filter' function $\boldsymbol{h}[n]$ which is the sum of two of band pass filters.

For $n \neq M$

$$h_d[n] = g_b \frac{\sin((n-M)\omega_{c4}) - \sin((n-M)\omega_{c3})}{(n-M)\pi} + g_a \frac{\sin((n-M)\omega_{c2}) - \sin((n-M)\omega_{c1})}{(n-M)\pi}$$

and for $n = M$

$$h_d[n] = g_b \frac{(\omega_{c4} - \omega_{c3})}{\pi} + g_a \frac{(\omega_{c2} - \omega_{c1})}{\pi}$$

3. GUI Implementation

In this section, I am going to describe how the previously mentioned techniques are implemented.

3.1. Short Time FFT

As I described in 2.1, the 'Block-by-Block' processing was used to satisfy FFT calculation condition.

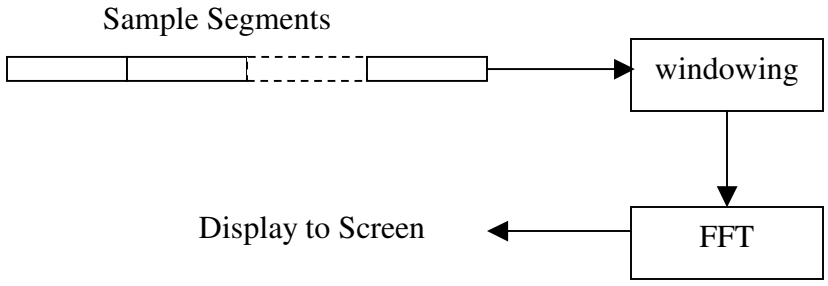

Fig 3.1 Short Time FFT processing

As you can see in Fig 3.1, the sampled signals are segmented with the length of powers of two (i.e.: 256, 512, 1024, 2048) and multiplied with the window. Finally, the FFT processes those windowed samples.

The next figures show the different outputs along with the different windowing applied. All figures uses 'Sine 440Hz (22kHz Sample 16Bit Mono).wav' as a reference, and were measured at 2048th sample with FFT size 512 samples.

Fig 3.2 Short Time FFT with 'von Hann' windowing

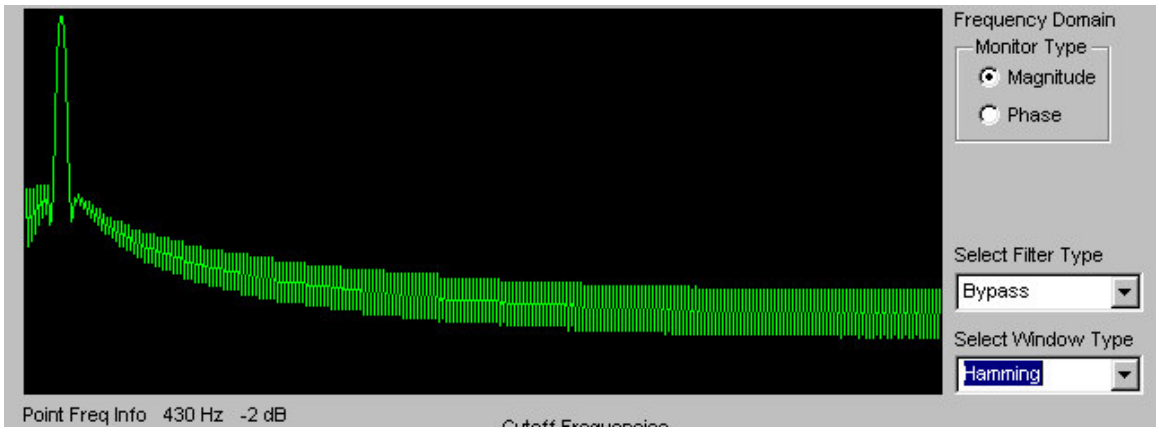

Fig 3.3 Short Time FFT with 'Hamming' windowing

Fig 3.4 Short Time FFT with 'Blackman' windowing

As you can see, because of the different outputs of each windowing, it is important to select the correct window along with the purpose of FFT analysis. The Hamming window has been shown to work particularly well with musical signals ('*Representations of musical signals' - De Poli, 1991*).

3.2. Filtering

The FIR filter performs the convolution. In order to perform the convolution, the N number of past samples has to be stored into memory (N is varied with FIR filter length). The next figures Fig 3.5a and Fig 3.5b explain the convolution in this GUI.

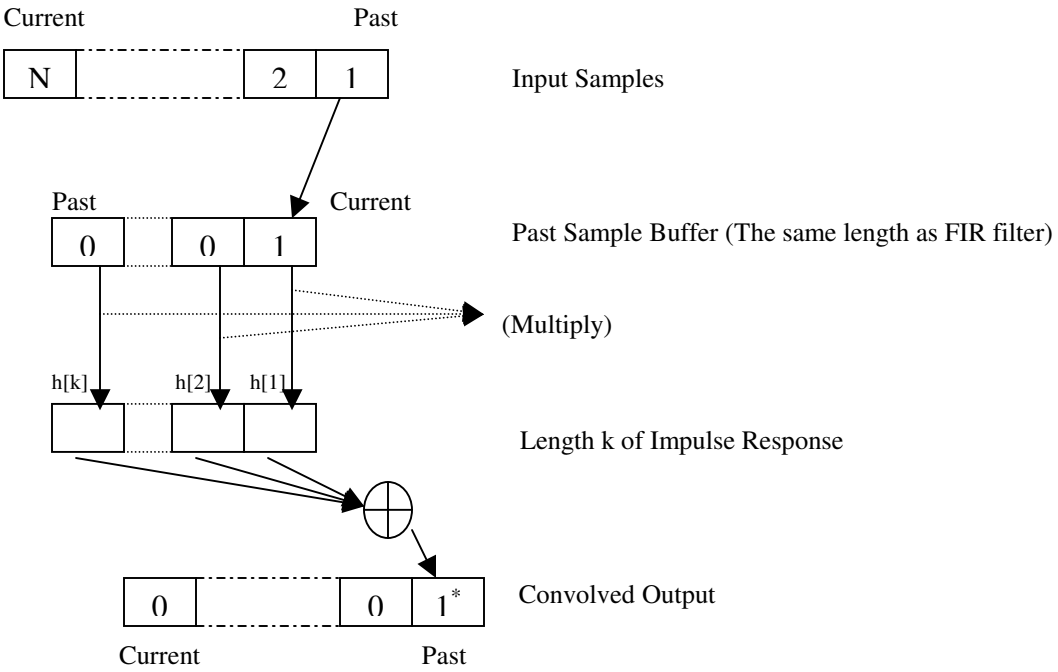

Fig 3.5a Convolution sequence a

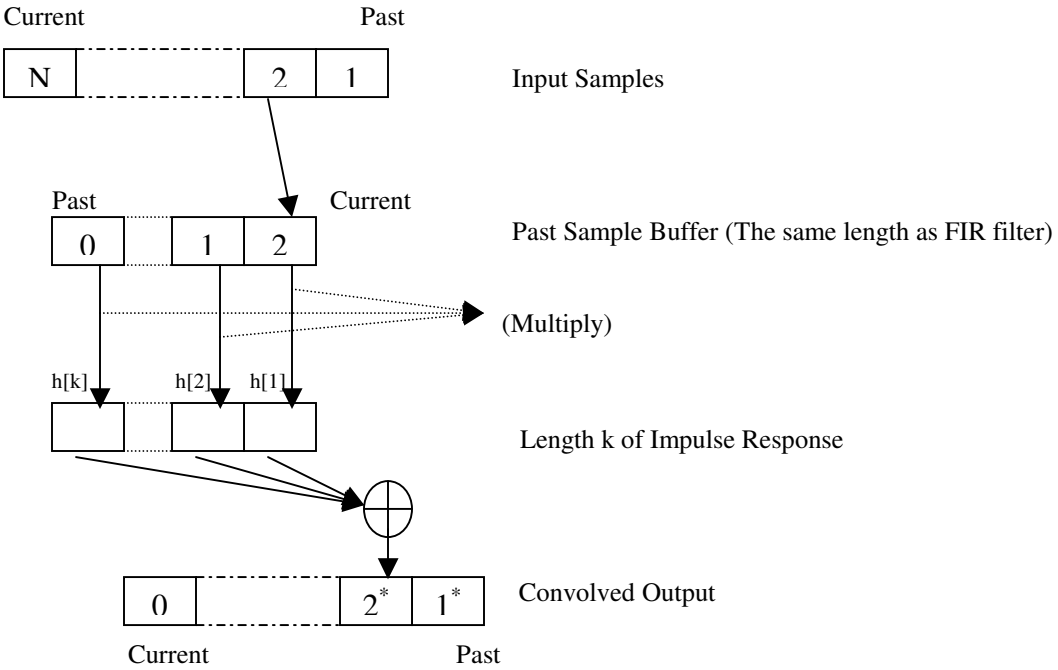

Fig 3.5b Convolution sequence b

The next figures show the different outputs when the different FIR filter length is applied. All figures uses 'Whitenoise (22kHz Sample 16Bit Mono).wav' as a reference, and were measured at 12288th sample with FFT size 512 samples.

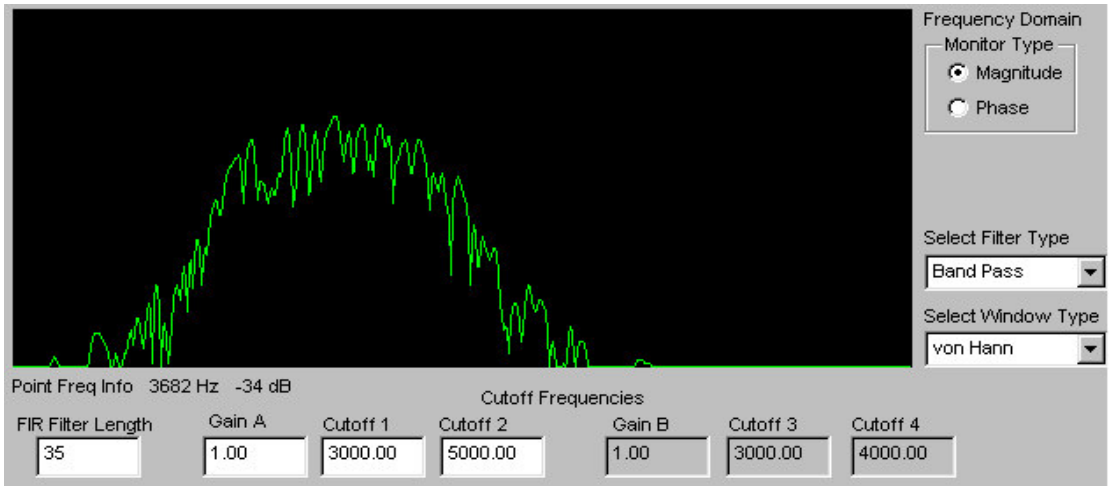

Fig 3.6 BANDPASS filter (cutoff1 = 3khz, cutoff2 = 5kHz, Length 35)

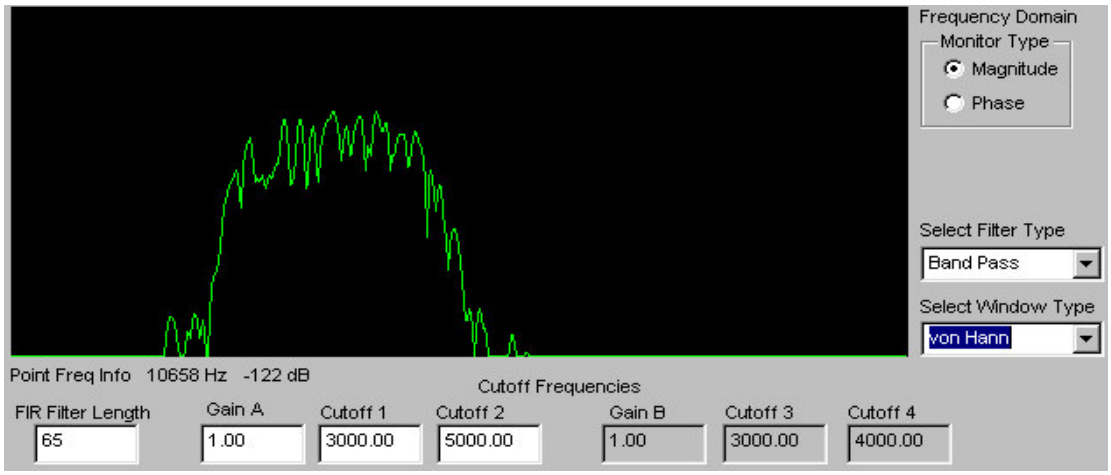

Fig 3.7 BANDPASS filter (cutoff1 = 3khz, cutoff2 = 5kHz, Length 65)

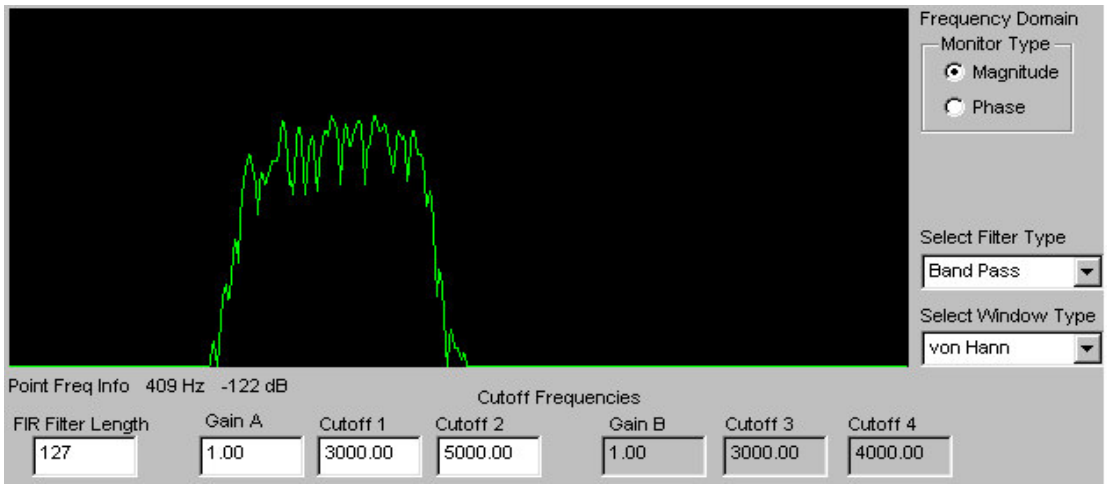

Fig 3.7 BANDPASS filter (cutoff1 = 3khz, cutoff2 = 5kHz, Length 127)

Although the longer filter length has the better output in terms of the amplitude responses, the longer filter length causes the longer system delay, eventually, it is important to choose the proper filter length for the target system.

4. Conclusions

FIR filters operates on sampled signals by performing the convolution between 'Impulse Responses' and samples. For FIR filters have their own delay, if the filters are connected, the filters have to be connected in parallel to avoid the distortion caused by phase.

In real applications, typically, input sequence is long, if FFT is applied to such a long sequence, there is huge delay in the computation. In order to avoid that huge delay, the input sequence has to be segmented with FFT computation condition and each segment should be processed individually.

In order to reduce the distortions cause by the 'Gibbs phenomenon', a proper windowing should be chosen for the FFT analysis. The 'Gibbs phenomenon' is due to rectangular windowing. If a proper windowing is used, the frequency behavior can be enhanced.

References

1. ‘A Course in Digital Signal Processing’ by Boaz Porat
2. ‘Representations of musical signals’ by Giovanni De Poli, Aldo Piccialli and Curtis Roads.
3. ‘Real Sound Synthesis for Interactive Applications’ by Perry R. Cook
4. ‘DIGITAL FILTERS’ by Andreas Antoniou