IMPLEMENTATION PERFORMANCE ANALYSIS COMPARISON FOR TWO METHODS OF IMPLEMENTING A FOURTH-ORDER CHEBYSHEV TYPE II CASCODE NTF DELTA SIGMA CONVERTER ARCHITECTURE

A Thesis by

Prakash BhojeGowda

Bachelor of Engineering, Bangalore University, 2001

Submitted to the College of Engineering
and the faculty of the Graduate School of
Wichita State University in partial fulfillment of
the requirements for the degree of
Master of Science

July 2005

IMPLEMENTATION PERFORMANCE ANALYSIS COMPARISON FOR TWO METHODS OF IMPLEMENTING A FOURTH-ORDER CHEBYSHEV TYPE II CASCODE NTF DELTA SIGMA CONVERTER ARCHITECTURE

I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science with a major in Electrical Engineering.

____

Larry D. Paarmann, Ph.D., Committee Chair.

We have read this thesis and recommend its acceptance:

____

John M. Watkins, Ph.D., Committee Member

____

Zhiren Jin, Ph.D., Committee Member

DEDICATION

To my parents
Mr. D. T. Bhoje Gowda and Mrs. S. N. Meenakshi
for their unfathomable love

ಪೂಜ್ಯ ತಂದೆ ಮತ್ತು ತಾಯಿಯವರಿಗೆ ಒಂದು ಸಮರ್ಪಣೆ.

ACKNOWLEDGEMENTS

First and foremost, I would like to sincerely thank my advisor, Dr Larry D. Paarmann, for his constant support of my thesis research. I feel extremely honored and privileged to have worked with him. His advice was valuable at every step of my thesis work and without it; I would not have completed this thesis. I would like to thank my parents for making my dream come true in pursuing my Master's degree. Only through their constant support and love was I able to accomplish this work. Also, I would like to thank my sisters Veda, Banu and my brother Lokesh for their support and encouragement during this endeavor. I also thank all the members of the Delta-Sigma research group for giving me valuable suggestions on my thesis work. I thank my committee members Dr John M Watkins and Dr Zhiren Jin for spending their valuable time in attending my thesis defense. Lastly, I would like to thank God for making my dream come true.

ABSTRACT

There is increasing interest in continuous-time implementations of sigma-delta modulators, especially for high-speed and low-power applications. This is primarily due to some important advantages continuous-time realizations have compared to their switched-capacitor counterparts: can operate at higher sampling frequencies. Chebyshev cascode architecture is one of the promising architectures to implement in continuous-time domain.

In this thesis, two circuit topologies are proposed to implement a fourth order Chebyshev type II cascode filter in continuous-time for use as a noise transfer function (NTF) in a delta-sigma data converter. These circuits were validated and performance analysis was done.

The first circuit topology involves splitting a fourth order filter into two second order filters, and then cascoding them together. The over all filter was implemented using ideal components. Each second order filter was implemented using a biquad circuit with four op-amps. In the second circuit topology the fourth order filter was split into two second order filters and then cascoded similar to the first circuit topology. In this circuit topology, every second order filter was implemented using only one op-amp, as compared to four op-amps in the first circuit topology.

The performance analysis of the circuit was done using SPICE. The obtained result was exported to workspace of Matlab, where the spectral analysis was obtained using Welch algorithm. This method shows that the proposed cascode architecture is robust and gives valuable information that is the best way to implement the circuit hardware.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER I

INTRODUCTION

1.1 Motivation

The traditional approach to simulate the behavior of a sigma delta modulator is by creating z-domain models and using a discrete time simulator, such as Matlab. This approach is logical since sigma-delta modulators are sampled systems by nature. As a result, most of the sigma-delta implementations were done by using techniques such as switched capacitor and switched current. However, there is an increased interest in the high speed continuous time sigma delta modulators. The principle reason is that the discrete domain simulators are not well connected to the back end IC design tools.

Continuous-time delta-sigma modulators offer less noisy virtual ground nodes at the input, inherent protection against signal aliasing, and the potential to use a physical rather than an electrical integrator in the first stage for novel applications like accelerometers and magnetic flux sensors. More significantly, they relax settling time restrictions so that modulator clock rates can be raised. This opens the possibility of wideband (1 MHz or more) converters, possibly for use in radio applications at an intermediate frequency.

The errors of the sample and- hold circuit are suppressed, and the modulator provides implicit anti-alias filtering, and are less prone to pick-up digital noise. Most of the reported continuous-time implementations are band-pass ones and some of these are not using CMOS technology, but silicon bipolar or other processes. There are only few reported realizations of low-pass continuous-time sigma-delta modulators, either for low-frequency applications or having modest SNR values, below 40dB

Sigma-delta data converters use time-redundant data to exchange operation speed for resolution and thus increase robustness against unavoidable parasitic and components tolerances. Consequently, they are very well suited for on-chip design of high-resolution analog interfaces in mixed signal application specific integrated circuits (ASICS) fabricated in general-purpose CMOS technologies. Although initial activities on sigma-delta converters ICs focused mainly on audio applications, their current application scope is considerably wider: from instrumentation to telecom. The exploration of these application fields are driven by industry in the direction of increased operation speed and resolution, decreased power consumption, and the convenience to migrate to modern low-voltage submicron technologies.

It is very important to evaluate the sigma-delta modulators at the device level. The SPICE simulation to evaluate the proposed circuit topologies was a challenge. A new approach of transferring the SPICE results to Matlab workspace was adopted. Spectral analysis was done on the processed data by using Welch algorithm.

1.2 Thesis Description

A data converter is a key component in any electronic system. Real-world signals are inherently analog; however, the digital form of an analog signal can be processed using robust, flexible and reliable digital signal processing. Therefore, analog-to-digital conversion becomes critical.

Various architectures for implementing delta-sigma data converters have been discussed in prior research [3] one of these, Chebyshev cascode architecture has been considered an efficient way of implementing it. So far, Chebyshev cascode has been evaluated in MATLAB and Simulink only. To obtain the best information about how to implement the circuit in hardware, realization of the Chebyshev cascode architecture also must be done using continuous time components.

In this thesis, Chebyshev cascode architecture was implemented with ideal continuous time components, with and without integrators. Simulations were to be carried out using MULTISIM 7.0.

1.3 Thesis Organization

Types of analog to digital converters are explained in detail and compared in CHAPTER I. This is followed by types of modulation techniques in analog to digital conversion. Sigma-delta data converters are introduced and explained in CHAPTER II. A brief over-view of sigma-delta data converters relative to how they work is discussed first. Types of sigma-delta data converters are briefly explained, followed by the effects of noise and the elimination of noise at the end of the chapter.

The beginning of CHAPTER III the design algorithm for implementing a cheyshev cascode filter in time domain using all ideal continuous time components with integrators is explained in detail.

In later part of CHAPTER III the design algorithm for implementing Chebyshev cascode filter in time domain with out using integrators and also using only a single op-amp per a second order section is discussed in detail.

CHAPTER IV the simulation results for the proposed architectures are documented.

CHAPTER V presents the conclusions.

Chapter II

Background

2.1 Literature Review

It is difficult to implement conventional filter in very large scale integration (VLSI) technology. This is due to the property of conventional filters being vulnerable to noise and interference. The popularity of delta-sigma data converters in low-speed and high-resolution applications has increased. One of the reasons being it can be easily implemented in VLSI technology. It is also, due to the requirement of minimal analog circuitry replacing complex digital circuitry. This being the major advantage in the low-voltage modern-submicron technologies and the preference for systems on chips where the analog circuits have to contend with noise generated by digital cores.

2.2 Analog to Digital Converters

The basic principle of operation is to use the comparator principle to determine whether or not to turn on a particular bit of the binary number output. It is typical for an ADC to use a digital-to-analog (DAC) converter to determine one of the inputs to the comparator. ADC can be implemented in three ways, as follows.

2.2.1 Digital Ramp ADC

This is a simple and efficient way of implementing an ADC. It involves a comparator, where by the value of the input analog voltage is compared with some standard that is realized by applying the analog voltage to one terminal of a comparator and triggering a binary counter that drives the digital-to-analog converter (DAC). The output of the DAC is applied to the other terminal of the comparator. As the value of the counter increases, the output of the DAC increases and at some point will trigger the comparator when the voltage at the DAC is greater than the input analog voltage. The binary counter is stopped when the output of the comparator is a digital value corresponding to the input analog voltage.

2.2.2 Successive Approximation ADC

The successive approximation ADC is much faster than the digital ramp ADC since it uses digital logic to converge on the value closest to the input voltage. The circuit consists of sample and hold, a successive approximation register consisting of clocked flip-flops and gates, an internal reference DAC, and a voltage comparator that compares the analog input voltage and the analog output voltage of the DAC.

As the successive approximation ADC is clocked, it generates a series of digital codes, which are fed into the reference DAC one at a time. The digital codes are generated in binary search fashion. If the bit toggled to 1 causes the DAC to output an analog voltage that exceeds the input voltage, then it is returned to O; otherwise, it is kept at logic 1.

2.2.3 Flash ADC

Flash ADCs are the fastest way to convert an analog signal to a digital signal and are ideal for applications requiring very large band width; however, they consume more power than the other ADCs. Flash ADCs are made by cascading high-speed comparators. Each comparator represents a one LSB, and the output code can be determined in one complete cycle.

For an N-bit converter, the circuit employs 2^n-1 comparators. A resistive divider with 2^n resistors provides the reference voltage .The reference voltage for each comparator is one least significant bit greater than the reference voltage for the comparator immediately below it. Each comparator produces a 1 when its analog input voltage is higher than the reference voltage applied to it; otherwise, the comparator output is 0.

2.3 Types of Modulation

2.3.1 Pulse Code Modulation

In pulse code modulation, signals have only two states, represented by either logic 1 or logic 0. It is one of the most efficient ways of digitizing any form of analog data, such as motion video, telemetry, music, etc.

The basic operation consists of sampling the input analog signal at regular intervals. The sample rate usually will be more than the maximum signal frequency. The instantaneous amplitude of the analog signal at each sampling is rounded off to the nearest of several specific predetermined levels (called quantization).The number of levels is always a power of 2. These numbers can be represented by three or four binary digits, respectively. The output of a pulse code modulator is a series of binary numbers represented by some power of 2 bits.

At the receiver or the destination end, the binary numbers are converted back into pulses having the same quantum levels as those in the modulator. Also, these pulses are processed further to get back the analog signal. The stream of pulses and non-pulse streams of 1s and 0s are not easily affected by interference and noise. In figure 2.1 a) shows the determination of the instantaneous amplitude at uniform intervals, figure 2.1 b) shows the instantaneous amplitudes of the input analog signal and figure 2.1 c) shows the reconstructed signal at the output.

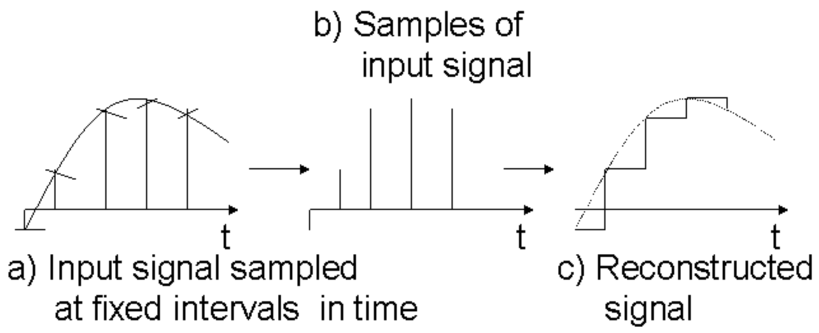
b) Samples of input signal
t
t
t
a) Input signal sampled at fixed intervals in time
c) Reconstructed signal

Figure 2.1 Pulse code modulation (taken from reference [6]).

2.3.2 Delta Modulation

The basic block diagram of the delta modulator is shown in figure 2.2.

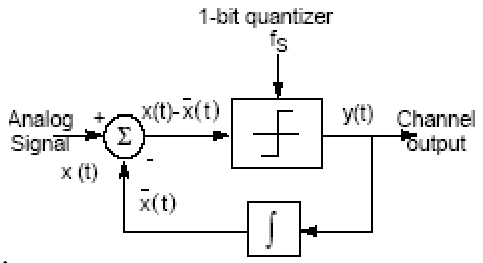

Figure 2.2 Delta modulation (taken from reference [11]).

Delta modulation is based on quantizing the change in the signal from sample to sample rather than the absolute value of the signal at each sample. The output of the integrator in the feedback loop tries to predict the input signal x (t) the integrator works as a predictor. The prediction error term $x(t) - \bar{x}(t)$ in the current prediction is quantized and used to make the next prediction. The quantized delta-modulated output is integrated in the receiver just as it is in the feedback loop.

Delta modulators have slope overload characteristics for increasing input signals. Therefore, the performance of delta modulators is dependent on the input signal frequency.

The delta-modulated output is shown in figure 2.3. The spectrum of quantization noise of the prediction error is flat, and the noise level s set by one-bit comparator. The signal-to-noise can be enhanced by the decimation process.

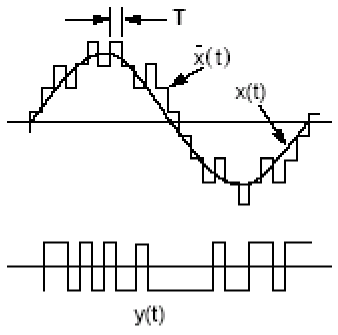

Figure 2.3 Delta modulated input (taken from reference [6]).

The comparison between different types of converters with respect to their resolution and bandwidth is shown in figure 2.4.

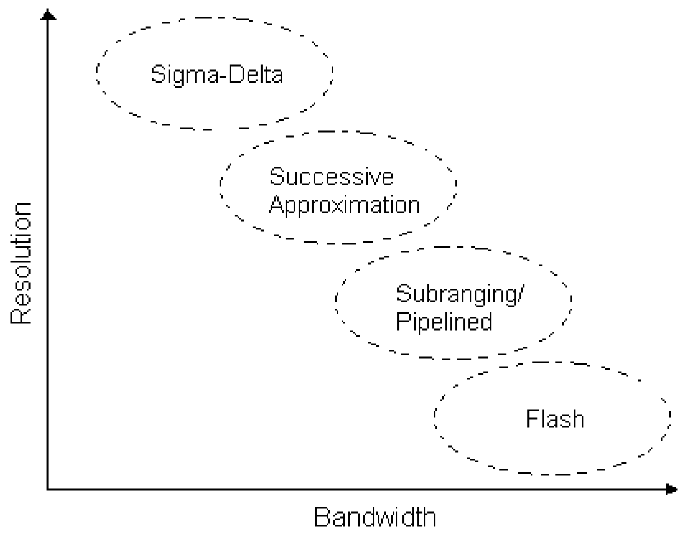

Figure 2.4 Comparison of resolution versus bandwidth for different ADCs

(taken from reference [8]).

2.4 Introduction to Sigma-Delta Converter

Sigma-Delta converters as shown in figure 2.5 are used to achieve high resolution with imprecise analog components through the use of over sampling and noise shaping by moving the quantization errors to high frequencies where it is filtered using digital techniques. With existing integrated technologies, higher nyquist rates can be easily achieved. Higher-order delta sigma converters suffer from inherent instability.

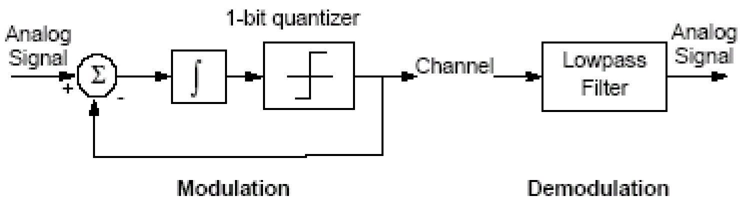

Figure 2.5 Block diagram of a sigma-delta converter (taken from reference [11]).

The name sigma delta modulator comes from putting the integrator in front of a delta modulator. These systems encode the integral of the signal itself, and thus making their performance insensitive to the rate of change of the signal.

The fundamental concepts of delta-sigma converters like, over sampling, quantization, noise shaping, digital filtering and decimation are explained in detail:

2.4.1 Over sampling

The frequency domain transfer function of a traditional multi-bit ADC with a sine wave input signal is considered. The input is sampled at a frequency F_s. According to the nyquist theory, F_s must be at least twice the bandwidth of the input signal.

When observing the results of FFT analysis on the digital output a single tone and large amount of random noise extending from the DC to the Fs/2 is observed, this is known as quantization noise. The input to ADC is continuous, but the output is digital and a discrete function, whose number of different states is determined by the converter’s resolution. Therefore, during the process of conversion from analog to digital, some information is lost and some distortion is introduced into the signal.

The signal-to-noise ratio, SNR is the ratio of fundamental amplitude to the root mean square sum of all the frequencies representing noise. Usually for an N-bit ADC, SNR =6.02 N + 1.76db. Number of bits must be increased to improve the SNR in a conventional ADC. The SNR can be easily improved by a technique called over sampling. The SNR for a one-bit ADC is 7.78 dB (6.02 +1.76). Each factor of 4 over sampling increases the SNR by 6 dB, and each 6 dB increase is equivalent to gaining one-bit. Sigma delta enables a gain of more then 6 db for each factor of 4.times over sampling.

2.4.2 Quantization

Quantization in amplitude and sampling in time are the two main functions of all digital modulators. After sampling the signal samples must also be quantized in amplitude to a finite set of output values.

Typical characteristics of the quantizer for the input signal plotted against the x-axis and output signal plotted against the y-axis is shown in figure 2.6.

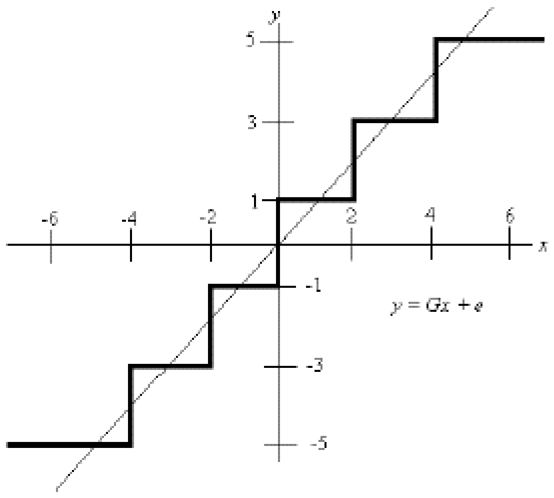

Figure 2.6 Typical characteristics of a quantizer (taken from reference[6]).

The quantized signal y is represented by a linear function Gx with an error e.

$$y = Gx + e \tag{2.1}$$

In this example, the level spacing delta = 2. The slope of the straight line is the gain G, which passes through the center of quantization characteristics so that when the quantizer does not saturate, the error is bounded by $\pm$ delta/2.

2.4.3 Noise Shaping

Noise shaping can be better understood with the figure 2.7 shown below.

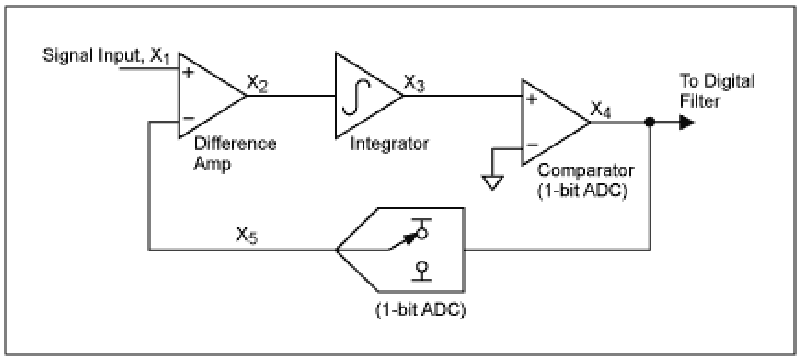

Figure 2.7. Noise shaping (taken from reference [11]).

This figure shows a difference amplifier, an integrator, and a one-bit comparator with a feedback loop with a one-bit ADC. The purpose of the feedback DAC is to maintain the average output of the integrator near the comparator's reference level.

The density of "ones" at the modulator output is proportional to the input signal. For an increasing input, the comparator generates a greater number of "ones." By summing the error voltage the integrator acts as a low pass filter to the input signal and a high-pass filter to the quantization noise. Thus, the noise is pushed into higher frequencies. Figure 2.8 shows the noise shaped spectrum.

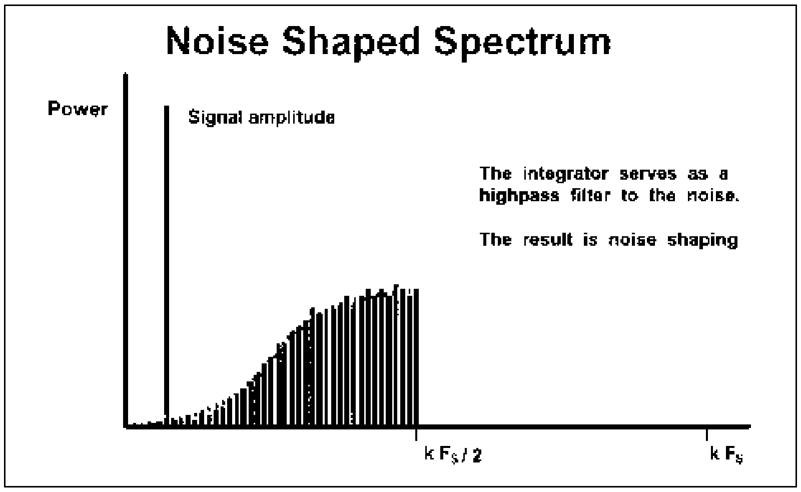

Figure 2.8 Result of noise shaping (taken from reference [7]).

Therefore in a lower frequency band, the desired input signals are passed and changed very little while the noise is suppressed to a great extent. With higher-order modulators, the noise is even more suppressed. This is called noise shaping. Due to noise shaping, higher-order modulators produce less output noise.

2.4.4 Digital Filtering and Decimation

The output of the sigma delta modulator is a one-bit data stream at the sampling rate, which can be in the mega hertz range. The main function of the digital and decimation filter is to extract information from this data stream and reduce the data rate to a more useful value. Figure 2.9 below shows the digital filter and decimation.

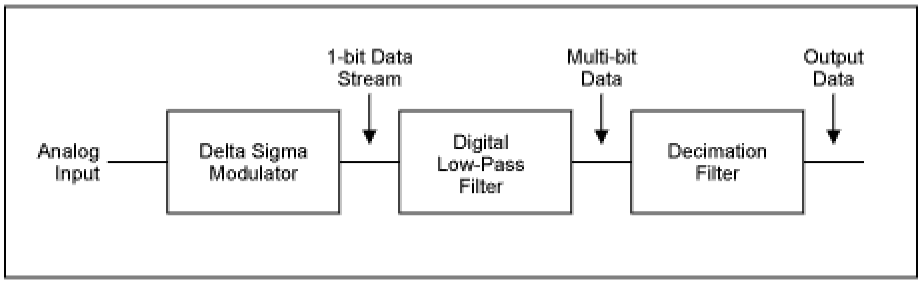

Figure 2.9 Digital filter and decimation (taken from reference [7]).

Decimation takes place in delta-sigma converters at the output of the low-pass filter. Since the bit stream is clocked with the sampling rate multiplied the over sampling rate, the output of the low-pass filter also is clocked with the over sampling rate. However, the sample rate clock (twice the input bandwidth) is required at the digital output only. For an over sampling rate of 64, for example, every 64th sample is taken and all others are discarded. This is possible because the signal is bandwidth limited by the low pass filter.

The digital filter improves the ADC resolution by averaging the one-bit data stream and removes quantization noise that is outside the band of interest. It determines the signal bandwidth, settling time, and stop-band rejection.

2.5 First-order Sigma-Delta Data Converter

A first order sigma delta converter is shown in the above figure. 2.10

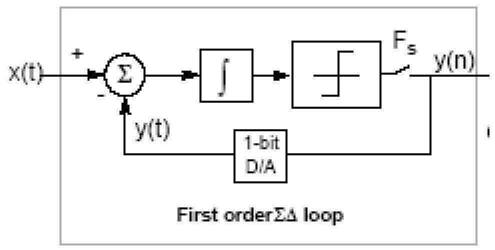

Figure 2.10. First-order sigma-delta data converter (taken from reference [11]).

It consists of an integrator and a comparator with a one-bit DAC in a feedback loop. The modulator output has only one bit of information either +1 or -1. The modulator output y(n) is converted to $y(t)$ by a one-bit DAC. The input to the integrator of the modulator is the difference between the input signal x(t) and the quantized output value of y(n) converted back to the predicted analog signal $y(t)$. Considering the DAC is perfect then the difference between the input signal x(t) and the predicted analog signal $y(t)$ is the quantization error. This error is summed up by the integrator and once again quantized by a one-bit ADC. Even though the quantization error at every sampling instance is large due to the nature of the two-level quantizer, the action of the sigma-delta modulator loop is to generate a ± 1, which can be averaged over several input sample periods to produce a very precise result.

The averaging is usually performed by a decimation filter at the output of the modulator. The input and output waveforms of the first-order modulator are shown in figure 2.11 below.

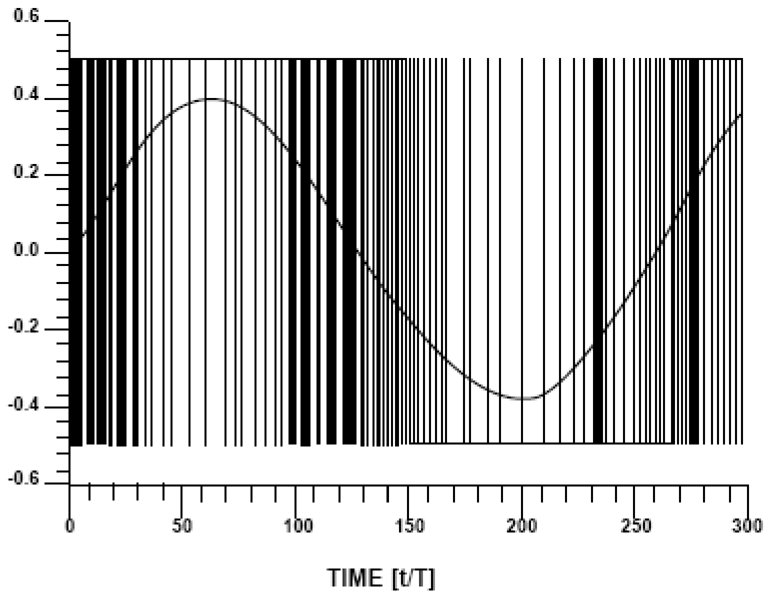

Figure 2.11. Input and output of first order sigma-delta converter (taken from reference [11]).

In each clock cycle, the value of the output of the modulator is either ± 1.These modulators can achieve a very high SNR. If the over-sampling ratio is doubled, then the SNR will improve approximately by 9dB.

CHAPTER III

CASCODE ARCHITECTURE IN S DOMAIN

3.1 Analysis of Sigma-Delta Modulator in S Domain

A linearized model of a first-order sigma-delta modulator in S domain is shown in figure 3.1. It can be observed that the feedback loop does not have a delay element.

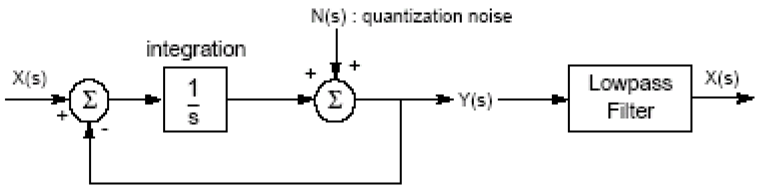

Figure 3.1 Linearized model. (taken from reference [11]).

The signal-transfer function STF is given by

$$Y(s) = [X(s) - Y(s)]\frac{1}{s} \tag{3.1}$$

$$STF = \frac{Y(s)}{X(s)} = \frac{\frac{1}{s}}{1+\frac{1}{s}} = \frac{1}{s+1} \tag{3.2}$$

When the quantization noise

$$N(s) = 0 \tag{3.3}$$

The system transfer function is given by

$$H(s) = \frac{1}{s} = \frac{Y(s)}{X(s) - Y(s)} \tag{3.4}$$

The noise-transfer function NTF when the input signal X(s) is zero is given by

$$Y(s) = -Y(s)\left[\frac{1}{s}\right] + N(s) \tag{3.5}$$

$$NTF = \frac{Y(s)}{N(s)} = \frac{s}{s+1} \tag{3.6}$$

Therefore, the STF and NTF can be expressed in terms of H(s) as ,

$$STF = \frac{Y(s)}{X(S)} = \frac{H(s)}{1+H(s)} \tag{3.7}$$

$$NTF = \frac{Y(s)}{N(s)} = \frac{1}{1+H(s)} \tag{3.8}$$

where x (t) is the input signal, y(t) is the output signal, and n(t) is the quantization noise. It follows from equations (3.7) and (3.8) that

$$STF = H(s)NTF = 1 - NTF \tag{3.9}$$

Since one of the properties of the sigma-delta modulator is to push the quantization noise to higher frequencies beyond the signal band, the signal-transfer function will be constant across the signal band, and the noise transfer function will be a high-pass transfer function.

Cascode architecture is considered since the conventional delta-sigma data converter will be unstable in higher orders. In the case of cascade architectures, the use of integrators to represent H(s) will lead the system to be unstable.

3.2 Implementation of Fourth order Chebyshev cascode type II filter in S domain with the use of integrators

Since the proposed architecture will be cascode, the system transfer function will be implemented in the form as given in figure 3.2.

$$H(s) = H_1(s) + H_2(s) \tag{3.10}$$

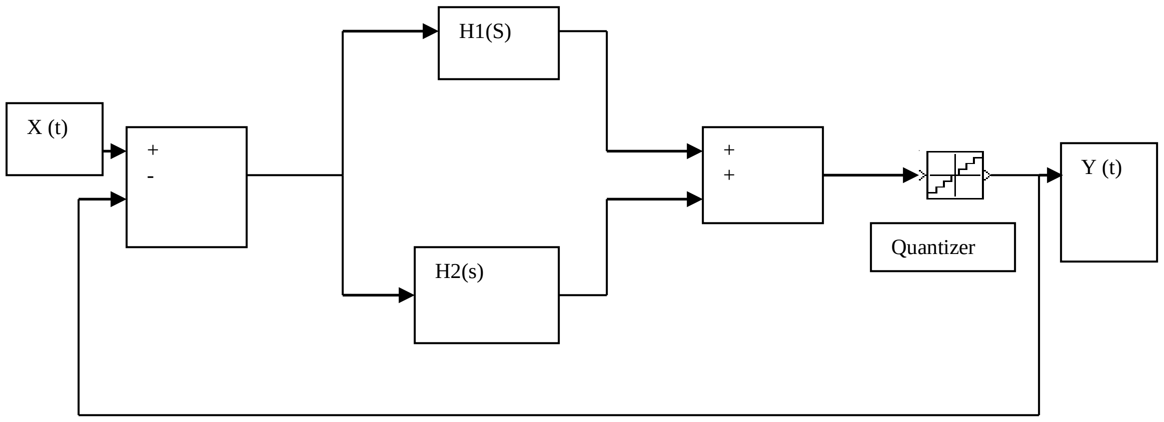

Figure 3.2. Basic cascode architecture.

3.2.1 Obtaining H(s) from Noise-Transfer Function NTF

In the following section, a fourth order Chebyshev type II high-pass NTF is specified, and the transfer function H(s) is determined. NTF will be of the form

$$NTF = \frac{K(s^4 + n_1 s^3 + n_2 s^2 + n_3 s + n_4)}{s^4 + d_1 s^3 + d_2 s^2 + d_3 s + d_4} \tag{3.11}$$

The value of K will be unity assuming the high frequency gain to be unity. Therefore, K=1. From equation (3.8)

$$NTF = \frac{1}{1 + H(s)} \tag{3.12}$$

$$H(s) = \frac{1 - NTF}{NTF} \tag{3.13}$$

For K=1 equation (3.11) becomes

$$NTF = \frac{N(s)}{D(s)} \tag{3.14}$$

where

$$N(s) = s^4 + n_1 s^3 + n_2 s^2 + n_3 s + n_4. \tag{3.15}$$

and

$$D(s) = s^4 + d_1 s^3 + d_2 s^2 + d_3 s + d_4 \tag{3.16}$$

Equation (3.13) may be expressed as

$$H(s) = \frac{D(s) - N(s)}{N(s)} \tag{3.17}$$

$$H(s) = \frac{(d_1 - n_1)s^3 + (d_2 - n_2)s^2 + (d_3 - n_3)s + d_4 - n_4}{s^4 + n_1 s^3 + n_2 s^4 + n_3 s + n_4} \tag{3.18}$$

By comparing equations (3.11) and (3.18), the process of obtaining H(s) from NTF seems to be easy. The above form of H(s), where the order of the numerator is less than the order of the denominator, is suitable to do partial-fraction expansion.

Since H(s) must to be expressed as the sum of two second-order functions, partial fraction expansion must be done.

3.2.2 Partial-Fraction Expansion

H(s) in equation (3.18) can be expressed as the sum of two lower-order functions as shown in equation (3.12) and may be accomplished by simple partial fraction expansion as

$$H(s) = \sum_{k=1}^{4} \frac{A_k}{s - p_k} \tag{3.19}$$

where
$$A_k = (s - p_k)H(s)\big|_{s=p_K} \tag{3.20}$$

The poles of H(s) occur in complex-conjugate pairs, the expansion of coefficients. is done Therefore, it follows that H(s) may be expressed as in equation (3.10) where

$$H_1(s) = \frac{\beta_1 s + \beta_2}{s^2 + \gamma_1 + \alpha_2} \tag{3.21}$$

$$H_2(s) = \frac{\xi_1 s + \xi_2}{s^2 + \gamma_1 s + \gamma_2} \tag{3.22}$$

$$\beta_1 = 2\operatorname{Re}\{A_1\}$$
$$\beta_2 = -2\operatorname{Re}\{A_1 p_1^*\}$$

$$\alpha_1 = -2\operatorname{Re}\{p_1\}$$
$$\alpha_2 = |p_1|^2$$

$$\xi_1 = 2\operatorname{Re}\{A_3\}$$
$$\xi_2 = -2\operatorname{Re}\{A_3 p_3^*\}$$

$$\gamma_1 = -2\operatorname{Re}\{p_3\}$$
$$\gamma_2 = |p_3|^2$$

Where it is assumed that the poles in equation (3.19) are arranged such that $\begin{aligned} p_2 &= p_1^* \\ p_4 &= p_3^* \end{aligned}$

The partial fraction expansion is summarized above may be readily programmed to be accomplished algorithmically in appendix A. This allows for ease in conducting experiments while varying NTF parameters, such as the cutoff frequency and the minimum attenuation in the stop band.

3.2.3 Design Example

This section details an example resulting in the full determination of $H_1(s)$ and $H_2(s)$. Let the NTF be a fourth-order Chebyshev type II high-pass filter with $A_s = 60dB$, and $f_s = 20kHz$.The NTF, with K in equation (3.11) set to unity, is

$$NTF = \frac{s^4 + n_1 s^3 + n_2 s^2 + n_3 s + n_4}{s^4 + d_1 s^3 + d_2 s^2 + d_3 s + d_4} \qquad (3.23)$$

where

$n_1 = 0, n_2 = 1.58 \times 10^{10}, n_3 = 0, n_4 = 3.12 \times 10^{19}$

and

$d_1 = 1.07 \times 10^6, d_2 = 5.92 \times 10^{11}, d_3 = 1.92 \times 10^{17}, d_4 = 3.12 \times 10^{22}$

It is noted that the zeros are on the jw axis. The transfer function H(s) may be obtained by direct application of equation (3.18) as

$$H(s) = \frac{c_1 s^3 + c_2 s^2 + c_3 s + c_4}{s^4 + n_1 s^3 + n_2 s^2 + n_3 s + n_4} \qquad (3.24)$$

where the denominator coefficients are given as,

$c_1 = 1.07 \times 10^6, c_2 = 5.76 \times 10^{11}, c_3 = 1.92 \times 10^{17}, c_4 = 3.11 \times 10^{22}$

Then, $H(s)$ may be expanded as the sum of $H_1(s)$ and $H_2(s)$ using partial-fraction expansion resulting in

$$H_1(s) = \frac{\beta_1 s + \beta_2}{s^2 + \alpha_1 s + \alpha_2} \tag{3.25}$$

and

$$H_2(s) = \frac{\xi_1 s + \xi_2}{s^2 + \gamma_1 s + \gamma_2} \tag{3.26}$$

where

$\beta_1 = -1.59 \times 10^7$, $\beta_2 = -2.09 \times 10^{12}$, $\alpha_1 = 0$, $\alpha_2 = 1.35 \times 10^{10}$,

and

$\xi_1 = 1.70 \times 10^7$, $\xi_2 = 2.67 \times 10^{12}$, $\gamma_1 = 0$, $\gamma_2 = 2.31 \times 10^9$

The magnitude responses for STF, NTF, H(s), H_1(s), and H_2(s) can be determined by running the MATLAB file Expand Cheby.m in Appendix A.

3.2.4 Design for Implementing H1(s)

The transfer functions $H_1(s)$ and $H_2(s)$ are implemented as biquad circuits as shown in figure 3.3.

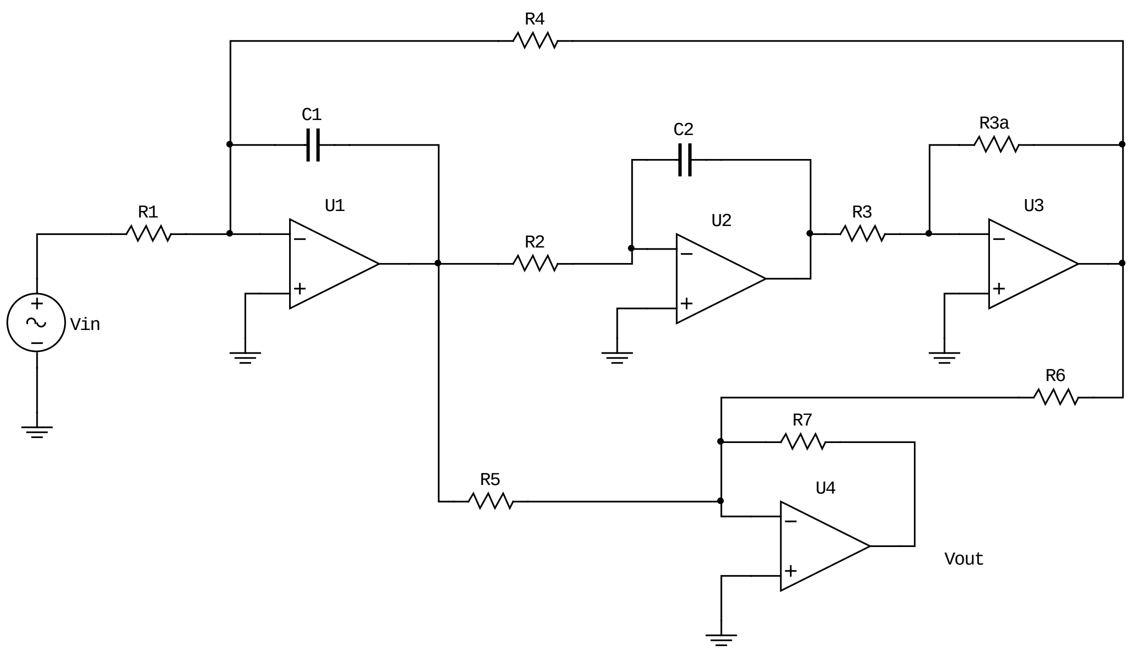

Figure 3.3 Architecture of a single second order stage in continuous time

The necessary design equations are also specified Implementing $H_1(s)$ as a biquad circuit where

$$H_1(s) = \frac{\beta_1 s + \beta_2}{s^2 + \alpha_1 s + \alpha_2} \qquad \text{where } \alpha_1 = 0$$

$$\frac{V_{out}}{V_{in}} = \frac{\beta_1 s + \beta_2}{s^2 + \alpha_2} \tag{3.27}$$

where
$$\alpha_2 = \frac{1}{R_2 R_4 C_1 C_2} \tag{3.28}$$

$$\beta_1 = \frac{R_7}{R_1 R_5 C_1} \tag{3.29}$$

$$\beta_2 = \frac{R_7}{R_2 R_4 R_6 C_1 C_2} = \frac{R_7}{R_6}\alpha_2 \tag{3.30}$$

choose the values for R$_2$, R$_4$, C$_{1,}$ and C$_2$ such that

$$\frac{1}{R_2 R_4 C_1 C_2} = \alpha_2 \tag{3.31}$$

choose the values of R$_7$ and R$_6$ such that

$$\frac{R_7}{R_6}\alpha_2 = \beta_2 \tag{3.32}$$

and

$$R_1 R_5 = \frac{R_7}{\beta_1 C_1} \tag{3.33}$$

3.2.5 Component values for H$_1$(s):

The obtained values of the Matlab file Expand_cheb .m. are

$$\beta_1 = -1.59\times 10^7 ,\ \beta_2 = -2.09\times 10^{12} ,\ \alpha_1 = 0,\ \alpha_2 = 1.35\times 10^{10}$$

Let

$$R_1 = R_2 = R_4 = R_6 = 1K\Omega$$

then

$$C_1 = C_2 = 8.6nF$$

and

$$R_7 = 155.3K\Omega \quad R_5 = 1135\Omega$$

By substituting all the component values and using only ideal op-amp's the H$_1$(s) is shown in figure 3.4.

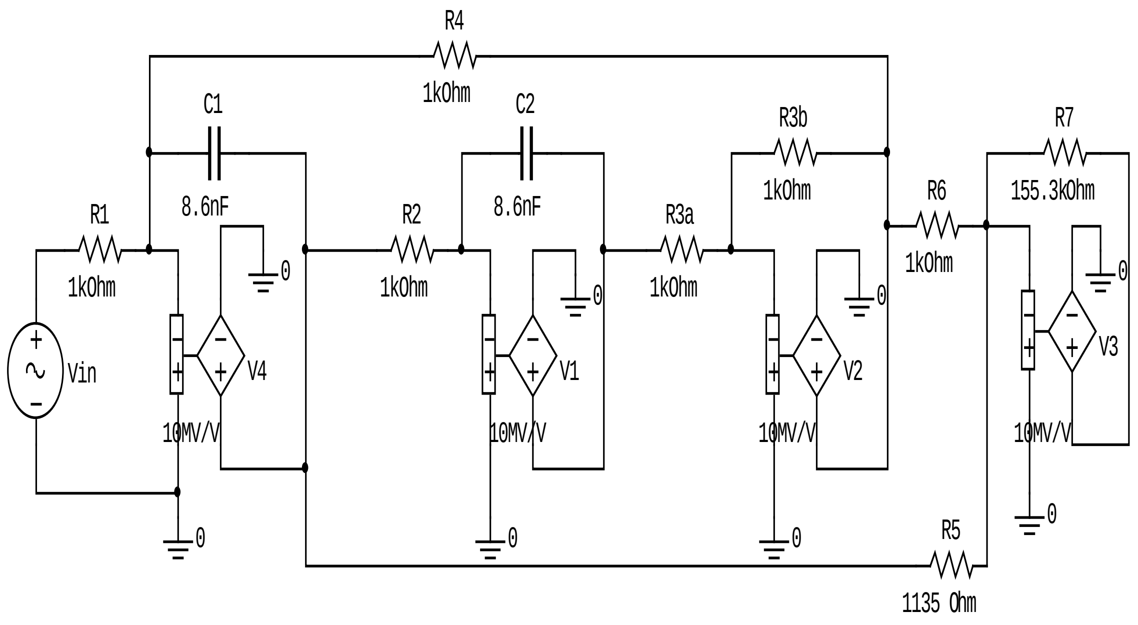

Figure 3.4. Structure of a second order stage using ideal integrators.

3.2.6. Design Example for Implementing H2(s)

$$H_2(s) = \frac{\xi_1 s + \xi_2}{s^2 + \gamma_1 + \gamma_2} \qquad (3.34)$$

where

$$\xi_1 = 1.70 * 10^7,\ \xi_2 = 2.67 * 10^{12},\ \gamma_1 = 0 \text{ and } \gamma_2 = 2.31 * 10^9$$

$$\frac{V_{out}}{V_{in}} = \frac{\xi_1 s + \xi_2}{s^2 + \gamma_2} \qquad (3.35)$$

where

$$\gamma_2 = \frac{1}{R_2 R_4 C_1 C_2} \qquad (3.36)$$

$$\xi_1 = \frac{R_7}{R_1 R_5 C_1} \tag{3.37}$$

$$\xi_2 = \frac{R_7}{R_2 R_4 R_6 C_1 C_2} = \frac{R_7}{R_6}\gamma_2 \tag{3.38}$$

Let R$_3$=1KΩ, and choose the values for R$_2$, R$_4$, C$_1$, and C$_2$ such that

$$\frac{1}{R_2 R_4 C_1 C_2} = \gamma_2 \tag{3.39}$$

Choose the values of R$_7$ and R$_6$ such that

$$\frac{R_7}{R_6}\gamma_2 = \xi_2 \tag{3.40}$$

$$R_1 R_5 = \frac{R_7}{\xi_1 C_1} \tag{3.41}$$

3.2.7 Component Values for H$_2$(s)

Let

$$R_1 = R_2 = R_4 = R_6 = 1K\Omega$$

$$C_1 = C_2 = 20.8nF$$

then

$$R_7 = 1155.8K\Omega \text{ and } R_5 = 3268.66\Omega$$

Refer to appendix A for the MATLAB code, which is used to find the cascode implementation of a fourth-order delta-sigma converter H(s) that has a Chebyshev type II high-pass NTF and an analysis of the design.

The circuit is implemented using an ideal op-amp with a large gain bandwidth product, and all the components used are ideal in nature.

3.3 Fourth-order Chebyshev cascode type II filter in S domain without using integrators

A design for implementing the Chebyshev type II filter in S domain using the minimum number of op-amps is explained here.

In the previous section, each second-order stage was implemented using four op-amps. The active filter realization is based on building block concepts using single amplifier biquad (SAB) circuits developed by Friend [4] and [5]. Friend's SAB circuit below realizes an arbitrary biquadratic function given by

$$H(s) = \frac{\beta_2 s^2 + \beta_1 s + \beta_0}{\alpha_2 s^2 + \alpha_1 s + \omega_0^2} \tag{3.42}$$

By making $\beta_2 = \alpha_1 = 0$ and $\alpha_2 = 1$,

the above transfer function becomes

$$H(s) = \frac{\beta_1 s + \beta_0}{s^2 + \omega_0^2} \tag{3.43}$$

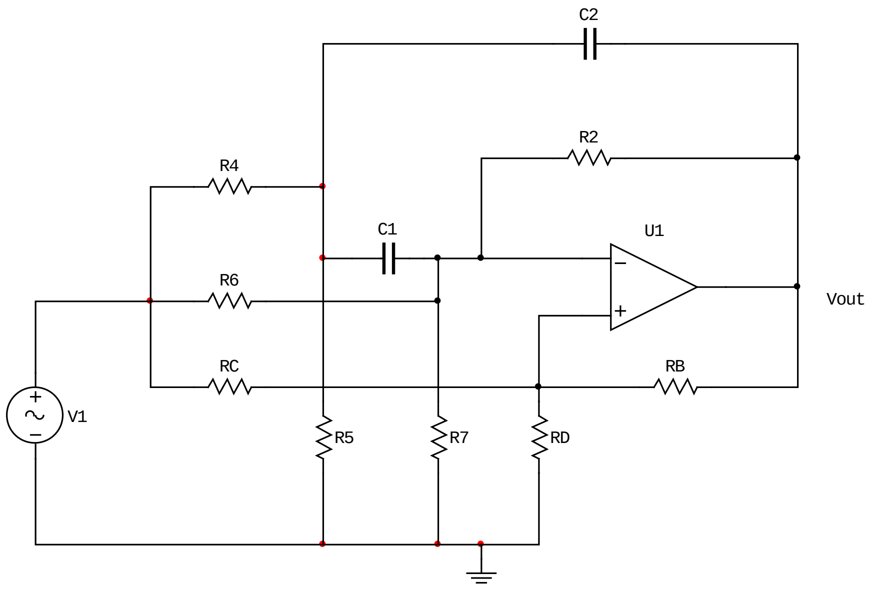

Figure 3.5. Friend's SAB circuit.

3.3.1 Design Algorithm

The design equations for the previous mentioned circuit are:

$$G_1 = \frac{C_2 G_A}{2G_B}\left[-\alpha_1 + \left[\alpha_1^2 + 4(1+\frac{C_1}{C_2})\omega_0^2 \frac{G_B}{G_A}\right]^{1/2}\right] \tag{3.44}$$

$$G_2 = \frac{C_1 C_2 \omega_0^2}{G_1} + \frac{G_B G_3}{G_A} \tag{3.45}$$

$$G_3 = \frac{C_1 C_2 G_A\left(\beta_0 - \omega_0^2 \beta_2\right)}{G_1\left(G_A + G_B\right)\left(\beta_2 - K_2\right)} \tag{3.46}$$

$$G_4 = \frac{G_1 n_2 + \left(1 + \frac{C_1}{C_2}\right)\beta_0 \frac{C_2^2}{G_1} - \beta_1 C_2}{1 + \frac{G_B}{G_A}} \tag{3.47}$$

$$G_5 = G_1 - G_4 \tag{3.48}$$

$$G_A \equiv G_C + G_D \tag{3.49}$$

$$G_C = \beta_2 G_A \tag{3.50}$$

$$G_D = G_A - G_C \tag{3.51}$$

$$K_2 \equiv \frac{G_6}{G_6 + G_7} \tag{3.52}$$

where G is called the conductance

$$G = \frac{1}{R} \tag{3.53}$$

3.3.2 Component values for H1(s)

The desired form for the transfer function for a single stage will be

$$H_1(s) = \frac{\beta_1 s + \beta_2}{s^2 + \omega_0^2}$$

where

$$\beta_1 = -1.59\times10^7,\ \beta_0 = -2.09\times10^{12} \text{ and } \omega_0^2 = 1.35\times10^{10}$$

Let

$$R_4 = R_C = \infty$$

By letting equation (3.47) be zero, solving it for the resultant required G_1, equating the result to equation (3.44), and letting C_1=C_2, it follows that

$$\frac{G_A}{G_B} = \frac{2\left(\frac{\beta_0}{\beta 1}\right)^2}{\omega_0^2} = 2.559 \qquad (3.54)$$

Also, for simplicity, let G_6 = G_7, which makes K_2= 0.5.

Let $C_1 = C_2 = 10\,pF$ and $G_B = 10^{-5}$. From equation (3.54) $G_A = G_D = 2.559\times10^{-4}$

$$G_1 = G_5 = 2.628\times10^{-6}$$

$$G_3 = 1.143\times10^{-4}$$

$$G_2 = 4.517\times10^{-5}$$

$$G_6 = G_7 = \frac{G_3}{2} = 5.705\times10^{-5}$$

$$R_2 = 22.137K\Omega,\ R_5 = 380.437K\Omega,\ R_6 = R_7 = 17.487K\Omega,\ R_B = 100K\Omega,\ R_D = 39K\Omega$$

Substituting all the component values and using an ideal op-amp the circuit for $H_1(s)$ is as shown in figure 3.6.

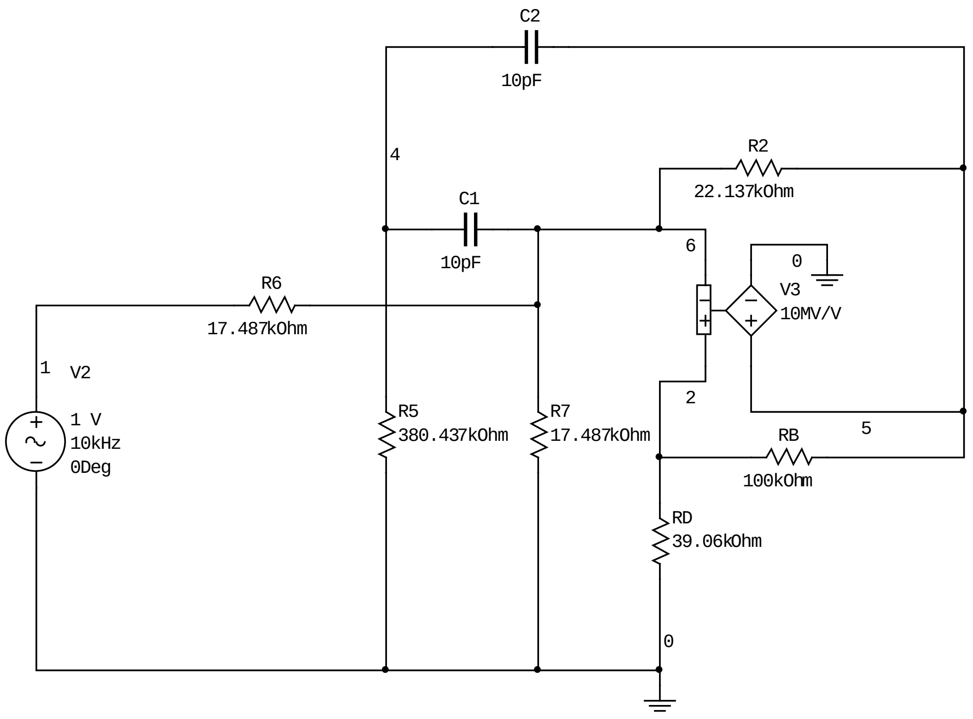

Figure 3.6 Structure of a second-order stage using one integrator

3.3.3 Component Values for $H_2(s)$

$$H_2(s) = \frac{\xi_1 s + \xi_0}{s^2 + \omega_0^2}$$

$\xi_1 = 1.70\times10^7$, $\xi_0 = 2.67\times10^{12}$ and $\omega_0^2 = 2.31\times10^9$. Let $R_4=R_C=\infty$

$$\frac{G_A}{G_B} = \frac{2\left(\frac{\xi_0}{\xi_1}\right)^2}{\omega_0^2} = 21.357$$

Let $G_B = 10^{-5}$, then $G_A = G_D = 21.357\times10^{-5}$. Let $C_1 = C_2 = 10pF,\ K_2 = 0.5$, then

$G_1=G_5=3.141\times10^{-6}$, $G_3=1.624\times10^{-4}$ and $G_2=7.59\times10^{-6}$ and

$$G_6 = G_7 = \frac{G_3}{2} = 8.1\times10^{-5}$$

$R_B = 100K\Omega,\ R_D = 4.682K\Omega,\ R_2 = 130K\Omega,\ R_5 = 318.36K\Omega,\ R_6 = R_7 = 12.345K\Omega$

CHAPTER IV

RESULTS AND SIMULATIONS

4.1 SPICE Simulation

The challenging aspect of designing the sigma-delta modulators is the simulation of performance at the device level. Multisim 7.0 is an efficient simulated program with integrated circuit emphasis (SPICE) tool to deal with different signal techniques. Device-level simulation is the ultimate to produce accurate estimation that is more dependable before fabrication. It is, therefore, desirable that the simulations and optimization of sigma-delta data converters be realized at the device level.

The input sinusoidal signal of the simulation was chosen to be 1v, 10 kHz, which was very much lower than the desired bandwidth of the modulator. Because the input signal frequency is very much smaller than the sampling frequency which is in the megahertz range, the simulation time was very long. The sampling clock pulse had a period of 1 micro second and a pulse width of 50 Nano second.

AC steady state analysis was done on the circuit and the magnitude response of H(s) and STF was obtained. Obtained response was similar to the desired magnitude response from the Mat lab Expand_cheb .m. A 10 kHz sine wave was used for the input of the modulator and the transient analysis was done. The output was a bit stream of +1 and -1.The sampled data was collected within adequate transient analysis time.

The collected data was saved as a text file which had both the time base and voltage base values at all time instants. Only the voltage base values were retained in the text file. The text file was imported to the workspace of Matlab as a data file.

For every one time period of the sampling clock pulse, the SPICE output had 134 data points. Therefore one out of every 134 data points was chosen and all the other data points were discarded. On this decimated file, windowed fast fourier transform was performed to obtain the signal-to-noise ratio and the power spectral spectrum.

Accurate results were obtained by setting the parameters in SPICE by trial an error. The parameters varied were the input signal voltage, input signal amplitude, settings of the one bit comparator and the circuit feedback resistances. Careful observations were made to determine the accurate output.

4.2 Simulation Results of Chebyshev Cascode Type II Filter

The design specifications are

$$A_S = 60dB$$

$$\text{fs\_design} = 20\text{kHz}$$

The quantizer used is considered to be binary, and the sampling frequency is equal to one mega hertz.

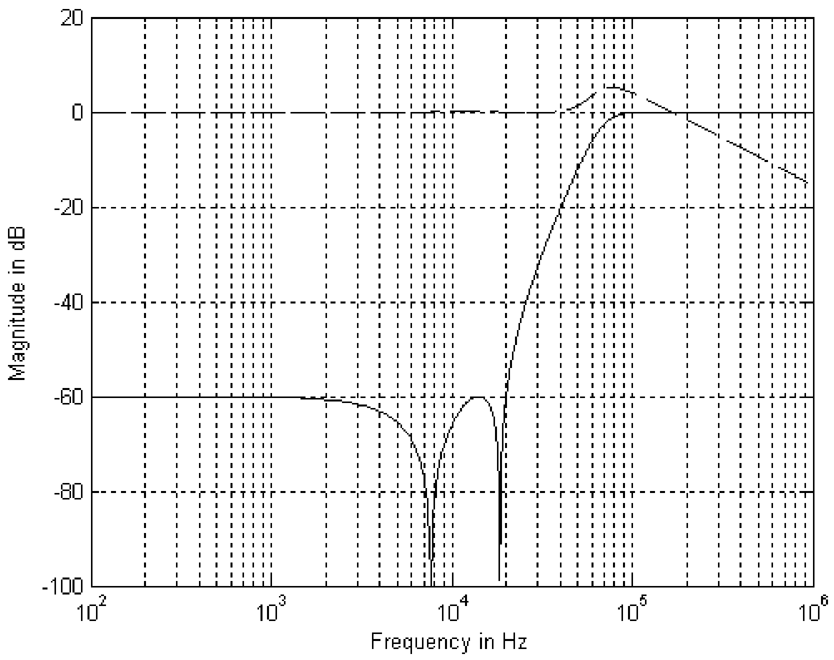

Figure 4.1 Magnitude response of NTF (solid line) and STF (dashed line)

Figure 4.1 shows the noise transfer function and signal transfer function which is Chebyshev cascode in nature.

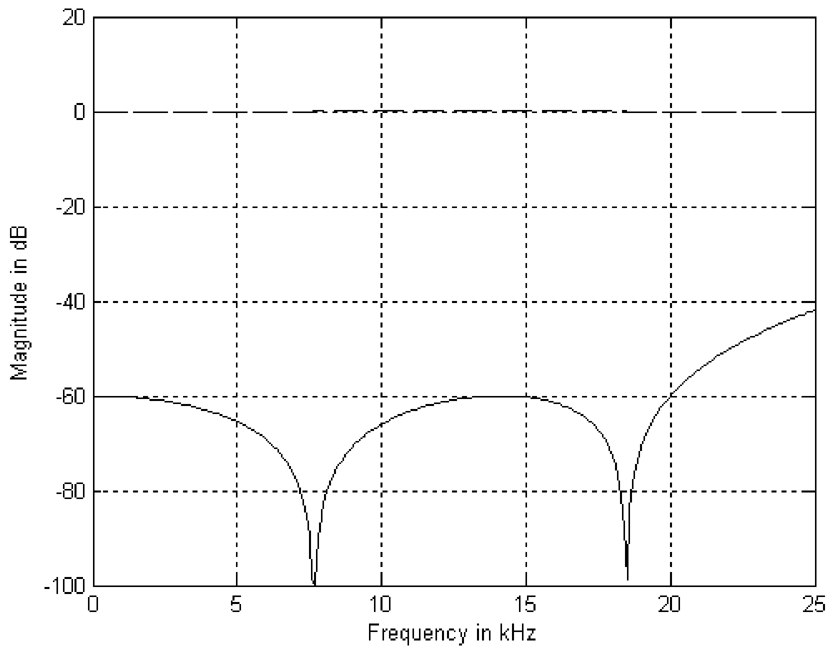

Figure 4.2 Signal Band Zoom of NTF (solid line) and STF (dashed line)

From the figure 4.2 we can see that the signal transfer function STF is constant across the signal band which is required for the efficient functioning of a delta-sigma converter. It can be seen that from figure 4.2 that the signal transfer function is constant across the signal band from 0 to 25 kHz.

Figure 4.3 shows the magnitude response of H(s) and figure 4.4 shows the signal band zoom of H(s).

Figure 4.5 shows the phase response of H(s).It can be observed from the figure that the phase undergoes a shift of 360 degrees.

Figure 4.6 shows the magnitude response for each second order stage .The solid line shows the response of $H_1(s)$ and the dashed line shows the response of $H_2(s)$.

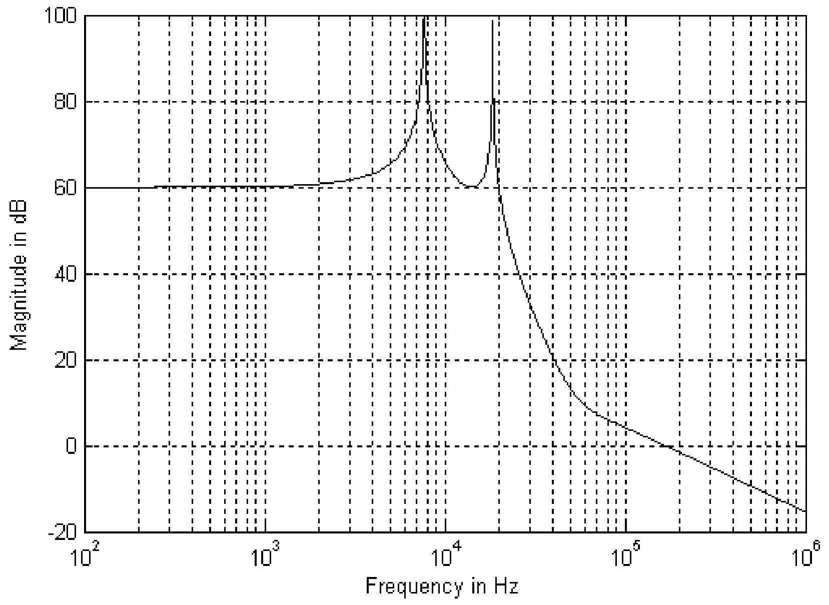

Figure 4.3 Magnitude Response of H(s)

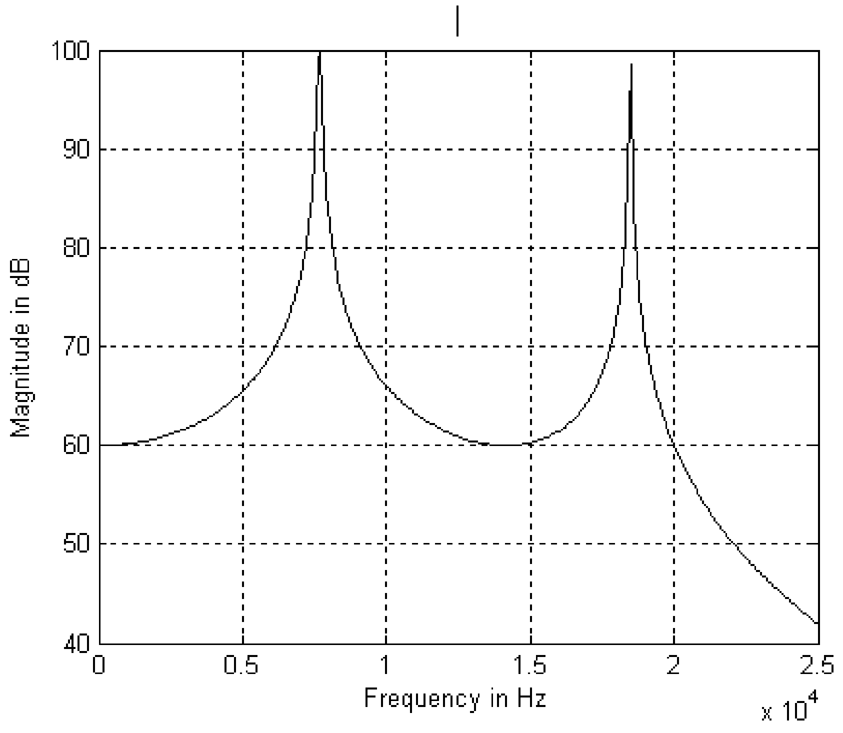

Figure 4.4 Signal Band Zoom of H(s)

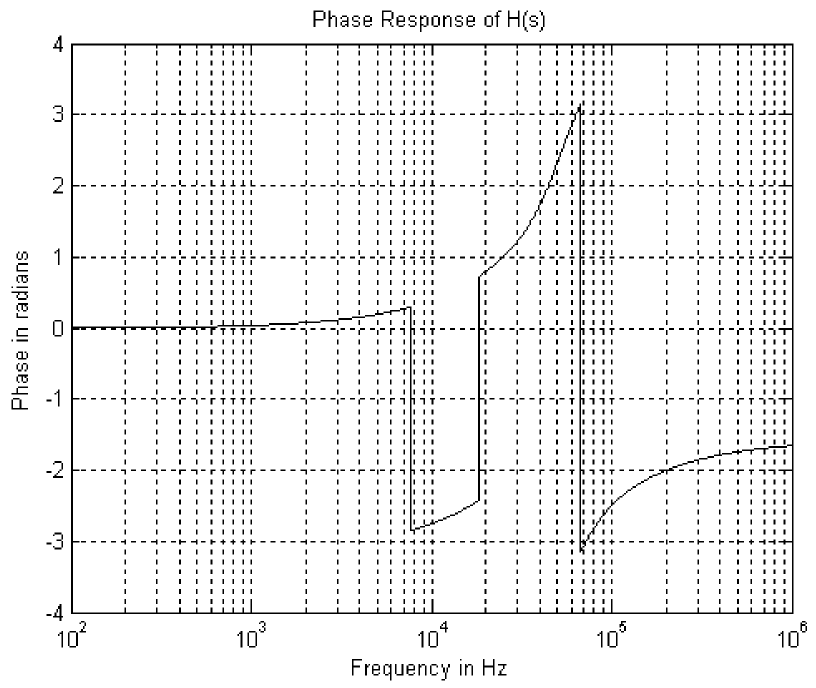

Figure 4.5 Phase Response of H(s)

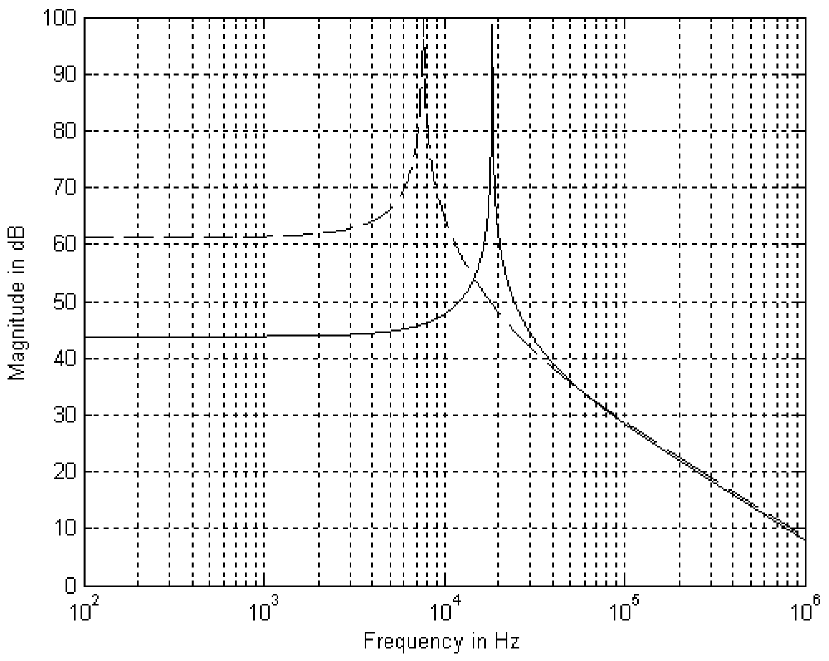

Figure 4.6 Responses of $H_1(s)$ (solid line) and $H_2(s)$ (dashed line)

Figures 4.7 and 4.8 shows the plot of poles and zeros of NTF and STF. It can be seen that the zeros are on the jw axis, and the poles are on the left half of the s plane.

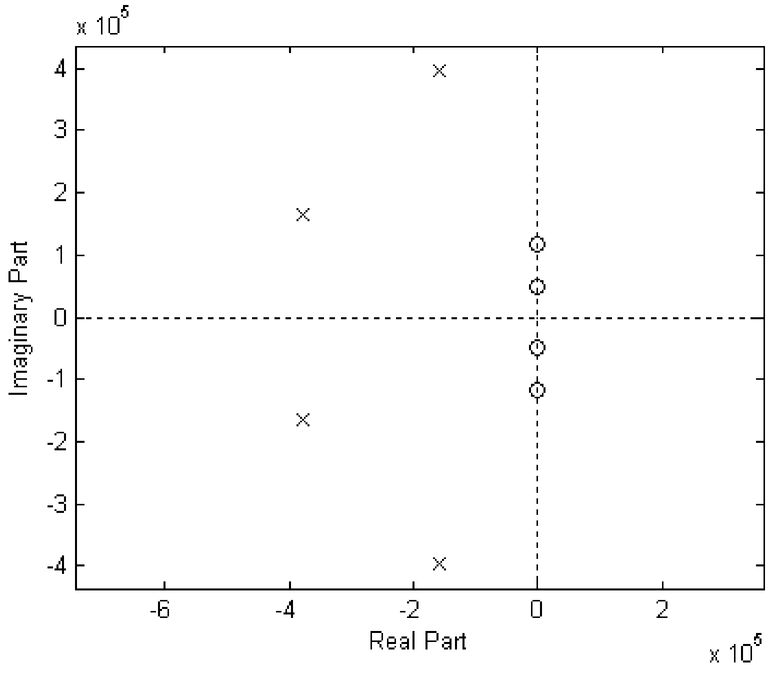

Figure 4.7 Poles and zeros of NTF

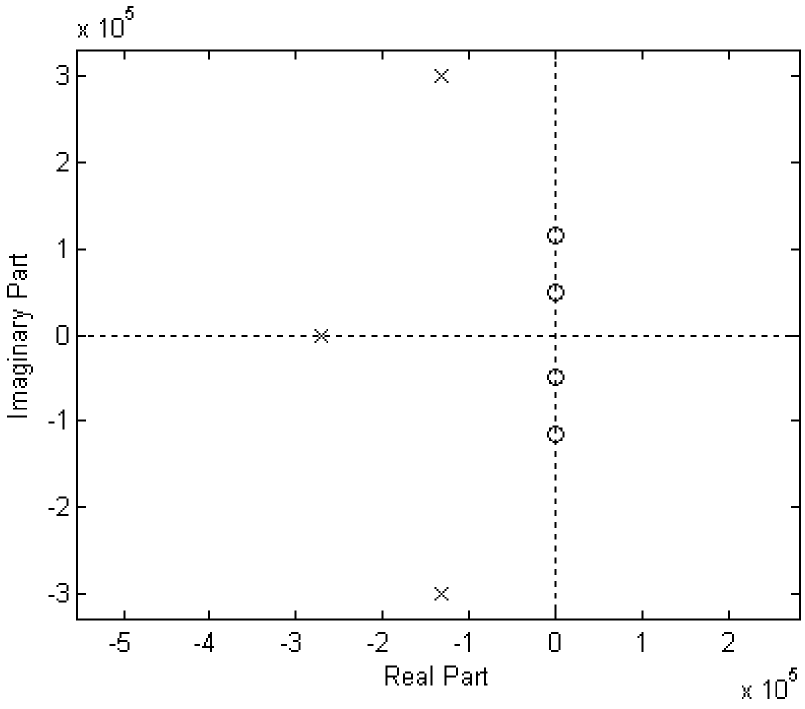

Figure 4.8 Poles and zeros of H(s)

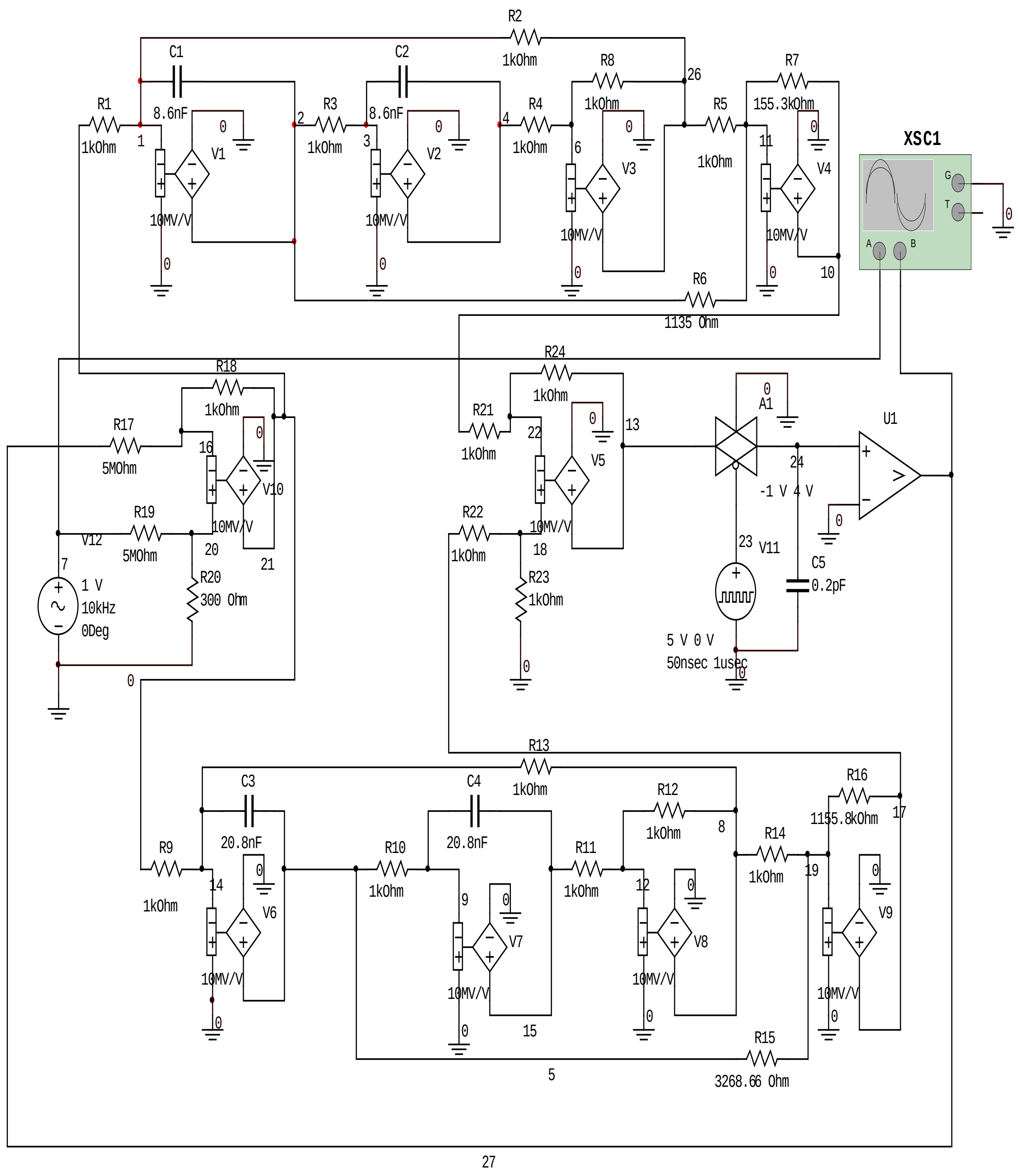

Figure 4.9 Cascode implementation of Chebyshev cascode type II filter for

Sine wave input- Topology I

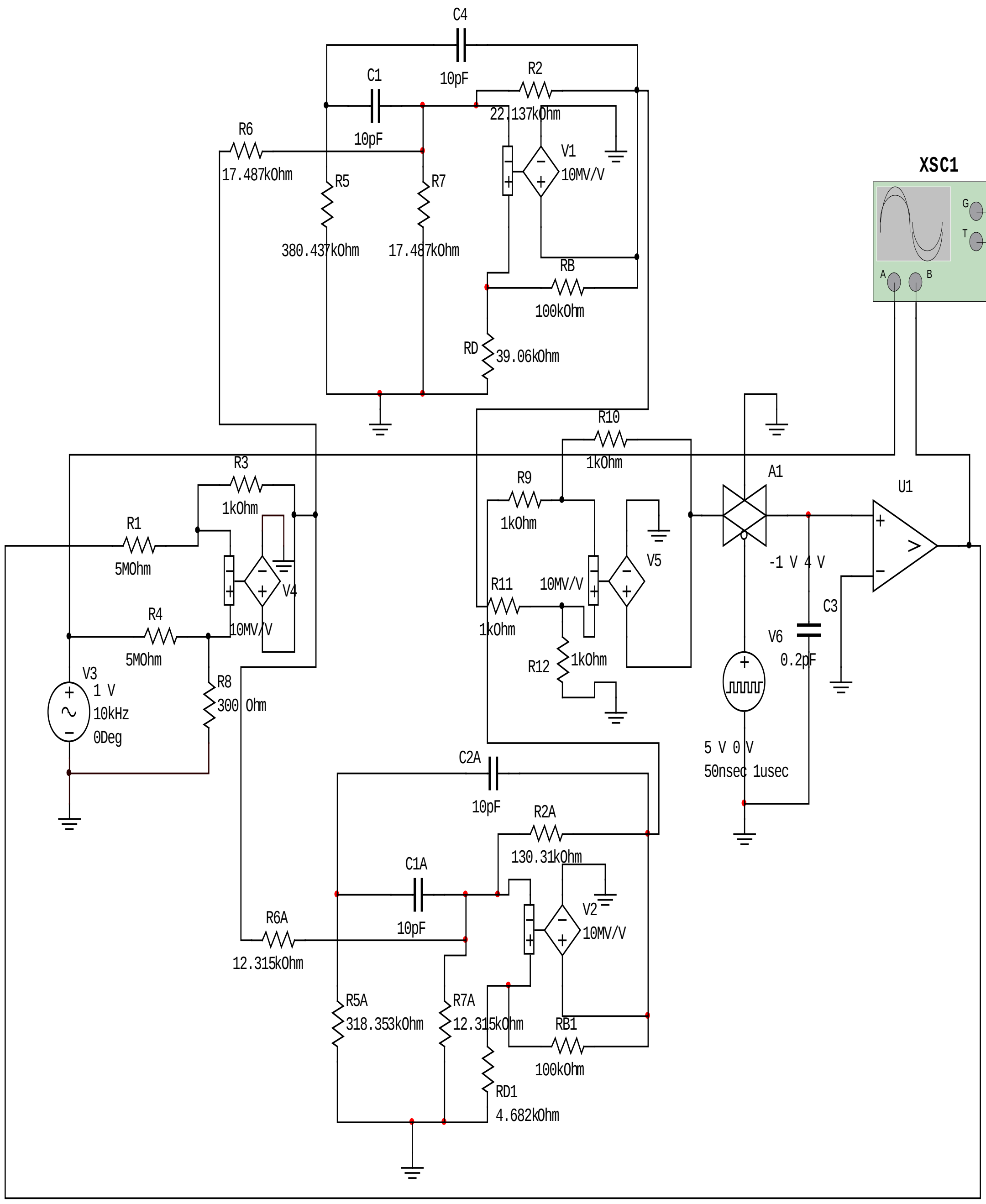

Figure 4.10 Cascode implementation of Chebyshev cascode type II filter for a sine wave input without using integrators- Topology II.

4.3 Welch Algorithm

The Welch algorithm is used to obtain the magnitude spectrum of the output. The time domain output was exported to MATLAB work space as a data file, and the data had to be decimated. In MULTISIM 7.0, the system clock was running 134 times faster than the specified period so the file was decimated and one out of 134 samples was chosen. The code for Welch algorithm is given in the appendix B.

The spectrum is obtained by dividing the data into non overlapping windowed segments. Welch method performs fast fourier transform to calculate the power spectrum.

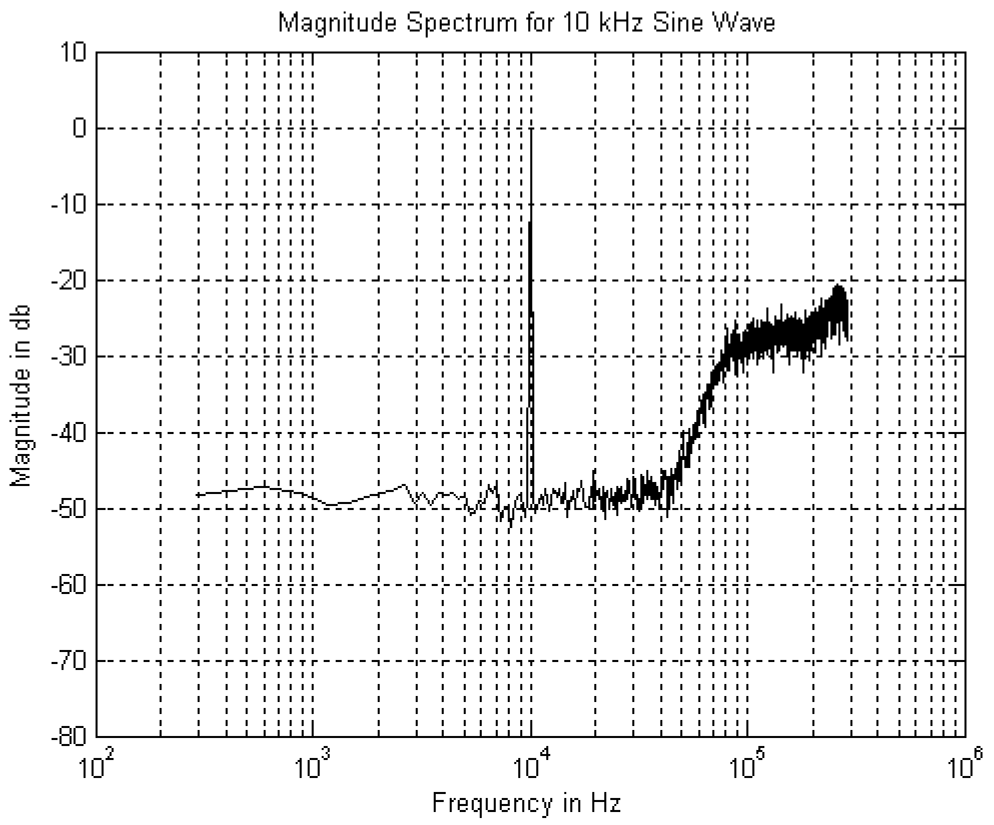

Figure 4.11 Magnitude spectrum for Topology I

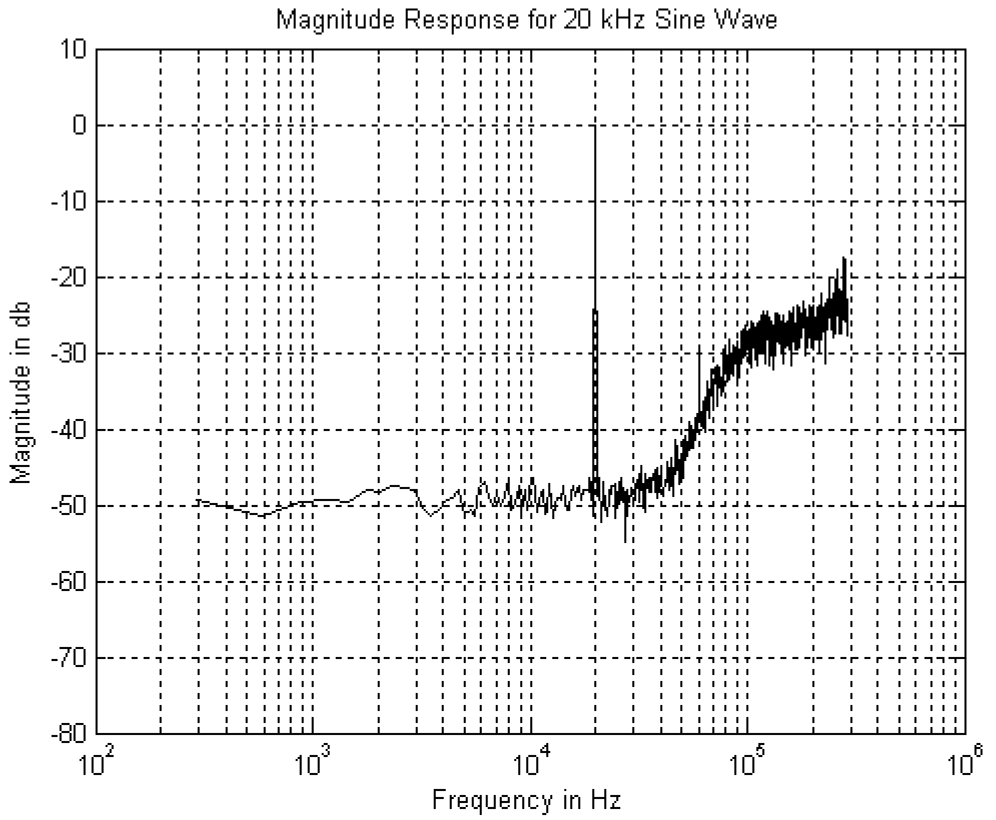

Fig 4.12. Magnitude spectrum for Topology I.

The magnitude spectrum for 10 kHz sine wave and 20 kHz sine wave was obtained for topology I as shown in figures 4.11 and 4.12. The spike at 10 kHz and 20 kHz represents the sine wave.

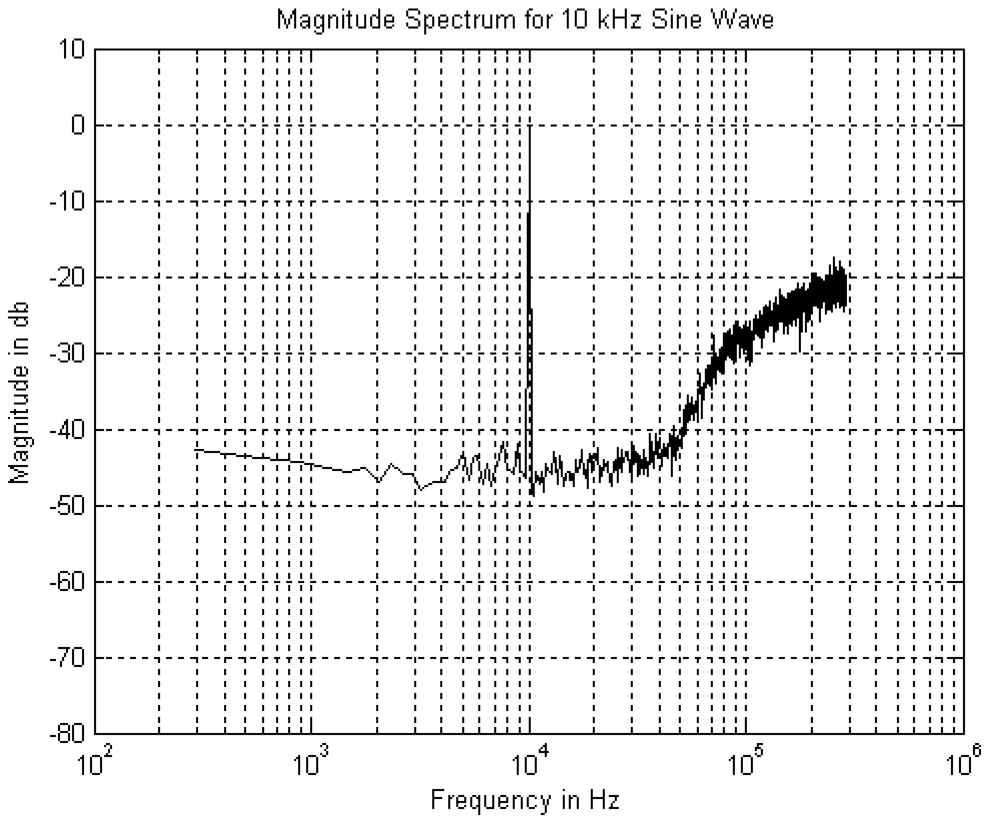

Figure 4.13 Magnitude spectrum for Topology II

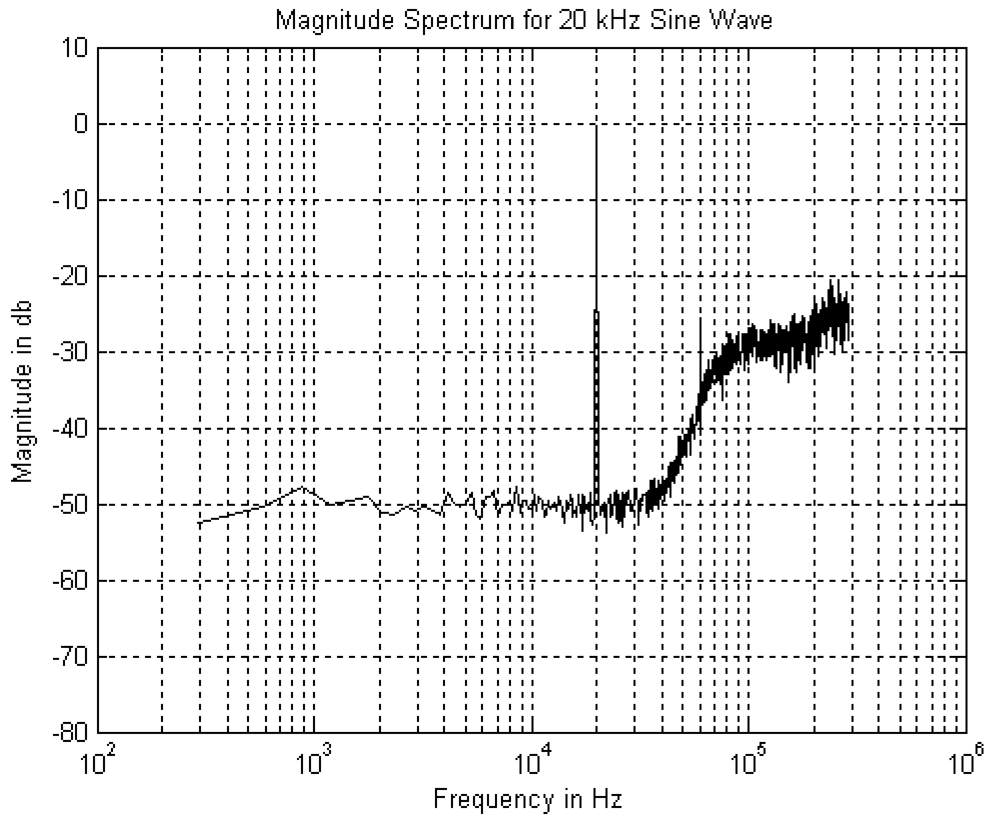

Figure 4.14 Magnitude spectrum for Topology II

Figure 4.13 and figure 4.14 shows the magnitude spectrum obtained by the topology II .The 10 kHz and 20 kHz spike represents the sine wave applied at the input. As shown, the signal to noise ratio (SNR) for architecture I is approxiamtely 50 dB and for topology II is approximately 48 dB, for a 10 kHz sine wave input and 50 dB for a 20 kHz sine wave input. However the numbers of op-amps used in topology II was one per each stage making the circuit less bulky than the topology I, which used 4 op-amps per stage.

CHAPTER V

CONCLUSIONS

Continuous-time delta-sigma architectures have become popular for achieving high resolution and high signal-to-noise ratio. The design for both proposed circuit topologies, with and with out using integrators for implementing a fourth-order Chebyshev cascode type II filter in continuous time was discussed in detail. The performance of both the topologies was presented and compared. Simulations were carried out in MULTISIM 7.0.

The results for topology I had a signal to noise ratio SNR of 51 dB. The topology II was novel method since each stage used only one op-amp, thus making the circuit less bulky. A new approach of representing delta-sigma architecture in the S domain was found promising. This approach yielded valuable information how to best realize a data converter in the hardware.

REFERENCES

[1] Candy, J.C., and G.C.Temes (1992). *Oversampling Delta-Sigma Data Converters*, IEEE Press, New York.

[2] Chowdhury, R., and S. Jain (1999). *Linear Integrated Circuits,* Wiley, New York.

[3] Darsi, S., and L.D. Paarmann (2003). "A Cascode Architecture Which Implements a Fourth-Order Chebyshev Type II Highpass NTF in Delta Sigma Converters," *Int. Signal Processing Conf.*, March 31-April 3, Dallas, TX.

[4] Friend, J.J., C.A. Harris, and D. Hilberman (1975). "STAR: An Active Biquadratic Filter Section," *IEEE Transactions on Circuits and. Systems,* vol. CAS-22, no. 2, pp. 115-121, February.

[5] Friend, J.J., (1970). "A Single Operational-Amplifier Biquadratic Filter Section," *Proc. IEEE Int. Symp. Circuit Theory*, pp. 189-190, Atlanta, GA.

[6] Haykin, S (1997). *Communication Systems*. Second Edition John Wiley and Sons, New York.

[7] Maxim/Dallas Semiconductor Co., (2003). "Demystifying Sigma-Delta ADCs," Application Note 1870, http://www.maxim-ic.com/appnotes.cfm/appnote_number/1870

[8] Norsworthy, S.R., R. Shafer, and G.C. Temes (1997). *Sigma-Delta Data Converters*, IEEE Press, New York.

[9] Oppenheim, A.V., and R.W. Schafer (1975). *Digital Signal Proces*sing, Prentice-Hall, Englewood Cliffs, NJ.

[10] Paarmann, L.D., (2001). *Design and Analysis of Analog Filters: A Signal Processing Perspective*, Kluwer Academic Publishers, Norwell, MA.

[11] Park, S. (1999) *Principles Of Sigma-Delta Modulation For Analog To Digital Converters*. Motorola Inc.

[12] Van Valkenburg, M.E., (1982). *Analog Filter Design*, Oxford University Press, New York.

APPENDICES

APPENDIX A

MATLAB CODE FOR DESIGN OF CASCODE FILTER-EXPAND CHEBY.m

% Expansion to find the cascode representation of a fourth-order

% delta-sigma data converter H(s) that has a Chebyshev type II highpass NTF,

% and analysis of the design

%

% The multisim model implemented using the design parameters generated by this

% matlab file is in "IV_ChebyBlock.msm"

%

% This matlab code will give the co-efficients for

% the desired transfer function H(s).

% H(s) has to be represented as: H(s) = H_1(s) + H_2(s)

%where

% $H(s) = \dfrac{c_1 s^3 + c_2 s^2 + c_3 s + c_4}{s^4 + n_1 s^3 + n_2 s + n_3 s + n_4}$

%

%and

% $H_1(s) = \dfrac{\beta_1 s + \beta_2}{s^2 + \gamma_1 + \alpha_2}$

%

%and

% $H_2(s) = \dfrac{\xi_1 s + \xi_2}{s^2 + \gamma_1 s + \gamma_2}$

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Enter the desired design specifications for the NTF:
Order = 4;        % The order is fixed at 4 for this m-file.
fs_design = 20E3; % The design specification for fs can be changed
                  % and this m-file will still work.
Min_Atten = 60;    % The design specification for As can be changed
                   % and this m-file will still work.
%
% Design the Chebyshev type II high pass filter:
ws = 2*pi*fs_design;
[BB, AA] = cheby2 (order, Min_Atten, ws,'high','s');
%
% Obtain the coefficients of H(s) from the modified transfer function
% for the Chebyshev Type II high pass filter:
n1 = BB (2);
n2 = BB (3);
n3 = BB (4);
n4 = BB (5);
%



%
```

```
k1 = AA (2);

k2 = AA (3);

k3 = AA (4);

k4 = AA (5);

%

c1 = k1 - n1;

c2 = k2 - n2;

c3 = k3 - n3;

c4 = k4 - n4;

%

HNUM = [c1 c2 c3 c4];

HDEN = [1 n1 n2 n3 n4];

%

% Compute and plot the magnitude response of H(s):

ff = (1:10000)*100;

ww = ff*2*pi;

HH = freqs (HNUM, HDEN, ww);

semilogx (ff, 20*log10 (abs (HH)))

xlabel ('Frequency in Hz')

ylabel ('Magnitude in dB')

title ('Magnitude Response of H(s)')

grid

pause
```

```
close
%
% Compute and plot the frequency response of H(s):
ff = (1:10000)*100;
ww = ff*2*pi;
HH = freqs (HNUM, HDEN, ww);
semilogx (ff, angle (HH))
xlabel ('Frequency in Hz')
ylabel ('Magnitude in dB')
title ('Frequency Response of H(s)')
grid
pause
close
%

% Zoom in on the signal band:
plot (ff (1:250), 20*log10 (abs (HH (1:250))))
xlabel ('Frequency in Hz')
ylabel ('Magnitude in dB')
title ('Signal Band Zoom of H(s)')
grid
pause
close
```

```
% Compute the magnitude frequency response of the NTF and the STF:
NTF = freqs (BB, AA, ww);
STF = 1 - NTF;
semilogx (ff, 20*log10 (abs (NTF)),'b', ff, 20*log10 (abs (STF)),'r')
xlabel ('Frequency in Hz')
ylabel ('Magnitude in dB')
title ('Magnitude Response of the NTF (blue) and the STF (red)')
grid
pause
close
%
% Zoom in on the signal band:
plot(ff(1:250)/1000,20*log10(abs(NTF(1:250))),'b',ff(1:250)/1000,20*log10(abs(STF(1:2
50))),'r')
xlabel ('Frequency in kHz')
ylabel ('Magnitude in dB')
title ('Signal Band Zoom of the NTF (blue) and the STF (red)')
grid
pause

% Perform partial fraction expansion of H(s):
poles = roots (HDEN);
%
```

```
stest = poles (1)';

ptest = [poles (2) poles (3) poles (4)];

temp = poly (ptest);

DENtest = [temp];

seval = [stest^3 stest^2 stest 1];

A1 = (HNUM*seval') / (DENtest*seval');

%

stest = poles(2)';

ptest = [poles (1) poles (3) poles (4)];

temp = poly (ptest);

DENtest = [temp];

seval = [stest^3 stest^2 stest 1];

A2 = (HNUM*seval') / (DENtest*seval');

%

stest = poles(3)';

ptest = [poles (1) poles (2) poles (4)];

temp = poly (ptest);

DENtest = [temp];

seval = [stest^3 stest^2 stest 1];

A3 = (HNUM*seval') / (DENtest*seval');

%

stest = poles (4)';

ptest = [poles (1) poles (2) poles (3)];
```

```
temp = poly (ptest);

DENtest = [temp];

seval = [stest^3 stest^2 stest 1];

A4 = (HNUM*seval') / (DENtest*seval');

%

% Compute second-order expansion parameters:

poles1 = [poles (1) poles(2)];

beta1 = 2*real(A1);

beta2 = - 2*real (A1*poles (1)');

DEN1 = poly (poles1);

alpha1 = DEN1 (2);

alpha2 = DEN1 (3);

NUM1 = [beta1 beta2];

%

poles2 = [poles (3) poles (4)];

xi1 = 2*real (A3);

xi2 = - 2*real (A3*poles (3)');

DEN2 = poly (poles2);

gamma1 = DEN2 (2);

gamma2 = DEN2 (3);

NUM2 = [xi1 xi2];

%

% Compute and plot the magnitude frequency response of H1(s) and H2(s):
```

```
HH1 = freqs (NUM1, DEN1, ww);

HH2 = freqs (NUM2, DEN2, ww);

semilogx (ff, 20*log10 (abs (HH1)),'b', ff, 20*log10 (abs (HH2)),'r')

xlabel ('Frequency in Hz')

ylabel ('Magnitude in dB')

title ('Response of H1(s) (blue) and H2(s) (red)')

grid

pause

close

%

% Reconstruct H(s) from H1(s) and H2(s) and compare frequency responses:

semilogx (ff, 20*log10 (abs (HH)),'b', ff, 20*log10 (abs(HH1+HH2)),'r')

xlabel ('Frequency in Hz')

ylabel('Magnitude in dB')

title ('Response of H(s) (blue), and H1(s) plus H2(s) (red)')

grid

pause

close

%

%

% Plot the poles and zeros of the NTF:

Npoles = roots (AA);

Nzeros = roots (BB);
```

```
zplane (Nzeros, Npoles)

title ('Poles and Zeros of the NTF')

pause

close

%

% Plot the poles and zeros of H (z):

Hpoles=roots (HNUM);

Hzeros=roots (HDEN);

zplane (Hzeros, Hpoles)

title ('Poles and Zeros of H(s)')

pause

close

%

% Check coefficients:

error1 = alpha1 + gamma1 - n1;

error2 = alpha1*gamma1 + alpha2 + gamma2 - n2;

error3 = alpha1*gamma2 + alpha2*gamma1 - n3;

error4 = alpha2*gamma2 - n4;

error5 = beta1 + xi1 - c1;

error6 = beta1*gamma1 + beta2 + xi1*alpha1 + xi2 - c2;

error7 = beta1*gamma2 + beta2*gamma1 + xi1*alpha2 + xi2*alpha1 - c3;

error8 = beta2*gamma2 + xi2*alpha2 - c4;

%
```

APPENDIX B

WELCH ALGORITHM USED TO PERFORM THE STATISTICAL ANALYSIS WITH CASCODE IMPLEMENTATION WITH SINE WAVE INPUT-WELCH.m

```
% It is assumed that a data file, named "SimOut Sine", produced by the
% MultiSim file "IV_ChebyBlock.msm", is available in the workspace.
%
%
% performing the decimation of the produced SimOut Sine file
to choose every 134th data point%
a = 134;
b_max = 4288000%   length of the file produced by the transient analysis of the circuit
%
p = zeros (b _max /a, 1);
q = zeros (b _max/a. 1);
i = 1:a:b_max
j = 2:
p = SimOut Sine (i , j);
q = (p);
out = q
qqq = (b _max/a) / 10;
FFtem = zeros ( qqq, 1);              % Creates an array of length qqq of zeros%
% The next 6 lines performs the Welch Method for 10 segments:
for k = 0: 9
```

```
temp = out ((k* qqq +1): (k* qqq +qqq));

FF = (abs (fft (temp, qqq)). ^2) / qqq;

FFtem = FFtem + FF;

end

FFM = FFtem / 10;

FFD = 10 *Log 10 (FFM);

FFD = FFD – 24;

% Scale so that the sine wave is normalized to 0 dB:

Scale = max (FFD (1:500)) ;

FFP = FFD (1: 990);

ww = (1:989) * (pi /qqq);

ff = ww* (10^6) /pi:

semi log x (ff, FFP)

grid

axis ([100 10^6 -80 10])

xlabel (' Frequency in Hz')

ylabel ('Magnitude in dB')

title (‘Magnitude Spectrum for 10 kHz Sine Wave’)

grid

pause

%

%
```