Error Mechanisms In Sigma-Delta Analog-To-Digital Converters

By Deron L. Gerow

Thesis Advisor: Donald M. Hummels, Ph.D.

An Abstract of the Thesis Presented in
Partial Fulfillment of the Requirements for the
Degree of Master of Science (in Electrical Engineering)
August, 1996

This thesis explores in detail the structure of the Sigma-Delta analog-to-digital converter (ADC). The Sigma-Delta ADC is of particular interest due to its ability to shape the quantization noise floor of low resolution quantizers, moving a majority of the quantization noise to a high frequency range, increasing the low frequency dynamic range of the converter. This noise shaping is achieved through feedback loops, and multiple loops can be combined to improve the noise shaping effect. The result after filtering and decimation is a much higher low frequency dynamic range than can be achieved by the quantizer alone. Possible sources of distortion in a three stage, third order noise shaping Sigma-Delta ADC are examined and simulated, so that various error parameters may be modeled independently of others. Using these simulation results, the distortion is identified and traced back to its origin, and a modified architecture is designed that attempts to compensate for these distortion terms.

ERROR MECHANISMS IN SIGMA-DELTA ANALOG-TO-DIGITAL CONVERTERS

By

Deron L. Gerow

B.S. University of Maine, 1994

M.S. University of Maine, 1996

A THESIS

Submitted in Partial Fulfillment of the

Requirement for the Degree of

Master of Science

(in Electrical Engineering)

The Graduate School

University of Maine

August, 1996

Advisory Committee:

Donald M. Hummels, Associate Professor of Electrical and Computer Engineering, Thesis Advisor

Fred H. Irons, Castle Professor of Electrical and Computer Engineering,

Allison I. Whitney, Lecturer in Electrical and Computer Engineering.

This work has been supported in part by the ARPA HBT/ADC program under a contract administered by the Office of Naval Research Grant N000149311007, and the Army Research Office Grant DAAH04-94-G-0387.

ACKNOWLEDGMENTS

Above all, I would like to thank my parents, Nancy and Carlton Gerow, for their encouragement and support throughout my educational and professional career. They were always supportive regardless of the newest endeaver that caught my interest, whether it be drumming, computers, or golf, allowing me to explore the world for myself to discover who I really was. This is an ongoing process, of course, and I can take comfort in knowing that they will always be supportive in the future as well.

Next, I would like to thank my thesis committee: Professors Don Hummels, Fred Irons, and Al Whitney. Without their expert guidance, this thesis would still only be a collection of notes in a binder. My thanks also go out to the Electrical Engineering department and alumni for providing such a professional work environment filled with state-of-the-art equipment.

Many thanks go to the members of the Communications Devices and Applications Laboratory, where I spent a large portion of my waking hours for the past two years. I could always count on them to stop me when I was being too serious by beating me up on the basketball or walleyball court, or humiliating me on the golf course. I will never forget the good times we had here.

Finally, a heartfelt thanks to my past roommate of five years, Joshua Caron. Through the good times and bad, we managed to survive each other and have a little fun. I am proud of your accomplishments, as I am sure you are of mine. I will always think of you as a brother, and no, you aren't keeping me awake.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Introduction

The recent need for high resolution Analog-to-Digital Converters (ADCs) has forced the communications industry to research new radical designs to overcome the physical limitations of classic flash and other Nyquist rate converters. One of the new designs emerging as a candidate for low frequency quantization, such as in digital audio or Integrated Services Digital Network (ISDN) applications [4], is the Sigma-Delta converter. Flash converters typically compare the input signal to many separate reference voltages via comparators, requiring $2^n - 1$ comparators and $2^n - 1$ reference voltages for an n-bit converter. For a high resolution converter ($\geq$ 16 bits), this translates to reference voltages with unreachable tolerances. For example, a 16 bit converter with an input range of 2 volts peak-to-peak requires reference voltages at increments of less than 62 microvolts. With current fabrication processes, it is extremely difficult to satisfy these tolerances. The Sigma-Delta converter utilizes oversampling and feedback loops to produce high resolution output samples using multiple quantizers, each with a low number of bits. This allows relaxed tolerances for each individual quantizer, while still producing the desired high resolution output samples. The first objective of this thesis is to introduce and examine the operation of various Sigma-Delta structures.

As is usual in the design of ADCs, the applications in which Sigma-Delta ADCs are implemented demand extremely high performance, beyond that which even high order Sigma-Delta converters can provide, due to limitations in current fabrication processes. Examination of primary sources of error in the Sigma-Delta converter is the second objective of this thesis. Once these error mechanisms have been identified, ways in which these flaws can be overcome will be investigated.

1.1 Background

The Sigma-Delta converter was first introduced in 1962 by Inose, Yasuda, and Marakami [10]. The primary motivation was to improve the existing Delta modulation scheme, increasing the passband resolution while requiring only a coarse quantizer inside the feedback loop. This first order loop provides a first order noise shaping function, effectively moving a large portion of the quantization noise to a high frequency range, beyond the passband of the digital filters that follow the loop. This structure was then extended to incorporate a second feedback loop, providing a second order noise shaping function. Until 1987, it was believed that noise shaping beyond second order was not feasible due to instability in the feedback loops, when Lee and Sodini presented a manner for creating higher order Sigma-Delta converters [12]. However, it was not until several higher order converters were commercially available that their stability was proven beyond a doubt [1]. This led to the design of multiple stage converters, typically referred to as MASH converters [15]. This latter type of Sigma-Delta converter is widely utilized due to it's modular construction using stable first order loops.

Another Sigma-Delta architecture that is beginning to appear in research laboratories is the current steering analog Sigma-Delta modulator. This converter does not use discrete-time switched capacitor circuits, as do the converters described in this thesis. Instead, they use current sources and analog integrators to increase the speed of the converter much beyond the speed of switched capacitor implemented converters. Since this converter is not commercially avaliable at this time, it will not be investigated in this thesis.

While the number of various Sigma-Delta loop structures has increased steadily in recent years, the digital filters that follow the loops have changed very little. The purpose of these filters is twofold: to remove high frequency quantization noise that the loops produce, and to reduce the output sampling rate, taking advantage of the oversampled nature of the Sigma-Delta converter. The most popular digital filter is the comb filter [5]. However, this filter has several drawbacks, and is usually followed by an additional Finite Impulse Response (FIR) filter. Since these filters are implemented digitally, there are no component tolerance limits that are typically seen in the corresponding analog filters, given sufficient word size for the filter coefficients.

1.2 Thesis Organization

This thesis begins with an introduction to Sigma-Delta Analog-to-Digital Converters, and a brief history of the converters' development. In chapter 2, a detailed description of the more popular topologies is presented, including the design of the first, second, and third order noise shaping loops, and the corresponding comb and FIR filters necessary to provide the desired output resolution. A simulation of a three stage, third order noise shaping Sigma-Delta architecture is then given. Portions of this simulation are described in detail in chapter 3, where specific error mechanisms commonly found in all switched capacitor implementations of Sigma-Delta converters are examined. Chapter 4 focuses on one dominant error mechanism, and simulation results lead to a procedure in which the architecture of a three stage, third order noise shaping Sigma-Delta converter can be modified to compensate for this source of distortion. Finally, conclusions from this work are drawn in chapter 5, and possible directions for future work are described.

CHAPTER 2

Sigma-Delta ADC Design Fundamentals

This chapter introduces concepts needed to understand the operation of the Sigma-Delta analog-to-digital converter (ADC). Ideal generic oversampled converters are discussed first, and the results are applied to develop concepts for ideal oversampled converters with noise shaping feedback paths. One specific oversampled converter with noise shaping is then focused upon, namely the ideal Sigma-Delta converter. Several Sigma-Delta topologies are briefly introduced, with the discussion culminating in a concentrated analysis of a three stage, third order noise shaping Sigma-Delta converter; this topology will be revisited in succeeding chapters. The corresponding digital filtering required to obtain the desired output resolution is also developed. Various advantages and disadvantages of each structure are presented throughout this discussion, and general guidelines are provided for the selection and design of a Sigma-Delta converter, given a set of output specifications.

2.1 Oversampled Analog-to-Digital Converters

An ideal oversampled analog-to-digital converter is composed of three distinct stages: an m-bit quantizer, operating at a high sampling rate, a digital filter, operating at the same sampling rate, and a downsampler, used to reduce the output sampling rate. A block diagram of this structure is shown in Fig. 2.1. To ensure proper operation of the converter, the analog input $x(t)$ is bandlimited to the effective Nyquist frequency band, $|f| < \frac{f_{clock}}{2D}$. The quantity D is defined as the oversampling ratio, which is the ratio of the quantizer sampling rate to the desired downsampled output sampling rate:

$$D = \frac{f_{quantizer}}{f_{decimator}}. \tag{2.1}$$

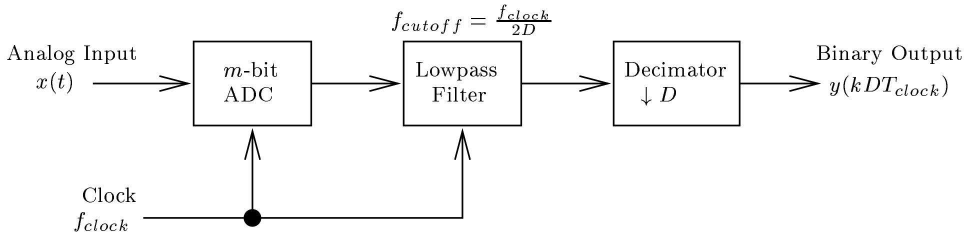

Figure 2.1: Generic Oversampled ADC Block Diagram

The m-bit ADC introduces quantization error across the entire oversampled frequency band $|f| < \frac{f_{clock}}{2}$. The digital lowpass filter following the ADC attenuates all frequencies beyond the frequency band of interest, $|f| > \frac{f_s}{2D}$, and the decimator reduces the sample rate of the converter by a factor D. Since the quantization error has been attenuated outside the frequency range of the filter passband, it will not contribute to the output noise floor due to aliasing limits of the decimator. In effect, the quantization error due to the m-bit ADC has been spread evenly across a frequency band that is D times larger than the frequency band of interest, thus increasing the output resolution of the converter beyond that which can be provided by the m-bit quantizer alone.

A quantitative approach will now characterize the operation of an oversampled converter in more detail. The first step in this analysis is to represent the m-bit ADC of Fig. 2.1 with an additive white noise source model [2]. This substitution is accurate if the quantization noise from the ADC is independent from the input signal and uncorrelated from sample to sample [6]. Although this statement is not accurate for quantizers with low numbers of bits, the model does lead to useful design guidelines for Sigma-Delta converters [6]. Exact analysis for low resolution quantizers is difficult, and usually requires simulation. The revised block diagram is shown in Fig. 2.2, where the clock sampling frequency is denoted f_s. It will be assumed that the input signal $x(kT_s)$ is bounded, $|x(kT_s)| < V_{max}$, and band limited, $S_{X_{IN}}(f) = 0$ for $|f| > \frac{f_s}{2D}$, where $S_{X_{IN}}(f)$ is the normalized power

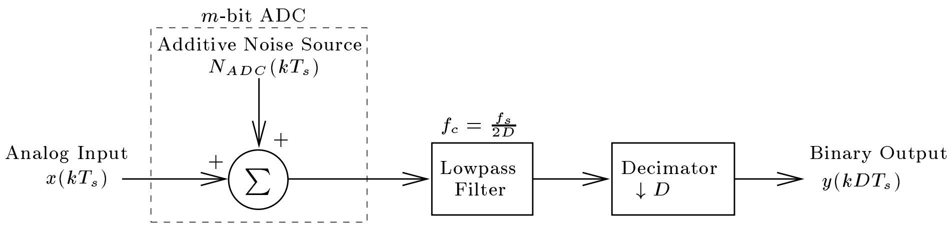

Figure 2.2: Revised Generic Oversampled ADC Block Diagram

spectral density of the input signal. Since the additive noise source is a white noise source, it's power spectral density is constant over the sampling frequency range, $|f| < \frac{f_s}{2}$, and has a variance of

$$\sigma^2_{N_{ADC}} = \frac{1}{12}\left(\frac{2V_{max}}{2^m}\right)^2, \tag{2.2}$$

where m is the number of bits utilized by the quantizer being approximated. This procedure produces a noise power spectral density at the lowpass filter input

$$S_{N_M}(f) = \frac{1}{12 f_s}\left(\frac{2V_{max}}{2^m}\right)^2 \quad \frac{V^2}{Hz}, \qquad |f| < \frac{f_s}{2}. \tag{2.3}$$

The output of the lowpass filter is simply the sum of the band limited input signal, $x(kT_s)$, and the filtered quantization error, which is represented as $N_{out}(kT_s)$. For an ideal lowpass filter, the power spectral density of the filtered noise is

$$S_{N_{out}}(f) = \begin{cases} \frac{1}{12 f_s}\left(\frac{2V_{max}}{2^m}\right)^2 & \frac{V^2}{Hz}, \quad |f| < \frac{f_s}{2D} \\ 0 & \frac{V^2}{Hz}, \quad |f| > \frac{f_s}{2D}. \end{cases} \tag{2.4}$$

Integrating $S_{N_{out}}(f)$ over the passband of the filter yields the total noise power out of the converter,

$$\sigma^2_{N_{out}} = \frac{1}{12D}\left(\frac{2V_{max}}{2^m}\right)^2$$

$$= \frac{1}{12}\left(\frac{2V_{max}}{2^{m+\frac{1}{2}\log_2(D)}}\right)^2. \tag{2.5}$$

From (2.5), it can be seen that the lowpass filter output of an oversampled converter operating at a sampling frequency f_s and utilizing an m-bit quantizer has the same accuracy as an $m + \frac{1}{2}\log_2(D)$ bit converter operating at a sampling frequency of $\frac{f_s}{D}$. For example, an oversampled converter with a 3 bit quantizer, $m = 3$, and an oversampling ratio of 64, $D = 64$, provides $3 + \frac{1}{2}\log_2(64) = 6$ bits of output resolution.

It should be noted that the effective resolution of the converter may also be found directly from the noise power spectral density function, rather than integrating the density and finding the total noise power at the filter output. Rewriting (2.4),

$$S_{N_{out}}(f) = \frac{1}{12\frac{f_s}{D}}\left(\frac{2V_{max}}{2^{m+\frac{1}{2}\log_2(D)}}\right)^2 \quad \frac{V^2}{Hz}, \qquad |f| < \frac{f_s}{2D}, \tag{2.6}$$

and comparing this result to the power spectral density of a non-oversampled ADC shows that oversampling produces $\frac{1}{2}\log_2(D)$ more bits of resolution.

As can be seen in the preceding discussion, oversampling allows one to use low resolution quantizers to obtain higher resolution. There is also an added benefit to this structure, since the typical analog circuitry required prior to the quantizer in non-oversampled converters is replaced by digital postprocessing circuits [4, 16]. This digital processing is much easier to implement in hardware, and does not require the tolerance limits imposed by the corresponding analog circuitry, given sufficient word size for the filter coefficients. However, an oversampled converter does require a compromise between conversion speed and output resolution.

2.2 Utilizing Feedback for Noise Shaping

The preceding section described a converter structure that provides increased output resolution using a low resolution ADC and oversampling. However, in order to

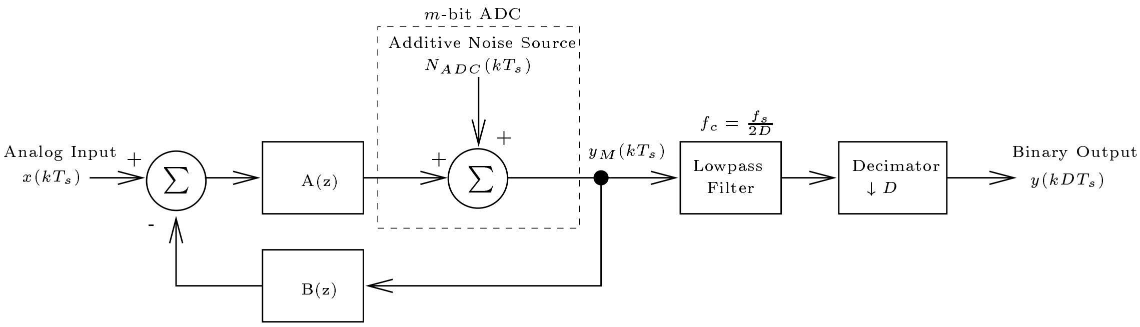

Figure 2.3: Block Diagram of an Oversampled ADC Utilizing Feedback

achieve extremely high output resolution, either the oversampling ratio, D, or the number of bits inside the ADC, m, or both, must be increased. In most cases, it is more economical to increase the oversampling ratio, which ideally requires little additional die space during fabrication of the device. However, high oversampling ratios may create undesirable bandwidth limitations on the input signal, since extremely high clock frequencies are difficult to achieve in MOS technology, which is commonly used to fabricate Sigma-Delta converters. One manner in which high output resolution can be achieved while avoiding these problems is through the use of noise shaping feedback inside the oversampled converter. The effect of this feedback is to lower noise power in the desired passband without having to increase either the number of bits, m, or the oversampling ratio, D.

A typical oversampled converter utilizing noise shaping feedback is shown in Fig. 2.3. The purpose of the feedback loop is to create a transfer function, $H_X(z)$, between the input $x(kT_s)$ and the output $y_M(kT_s)$ that does not affect the magnitude of $x(kT_s)$, while creating a transfer function, $H_N(z)$, between $N_{ADC}(kT_s)$ and $y_M(kT_s)$ that reduces the power from $N_{ADC}(kT_s)$ located in the frequency band of interest, $|f| < \frac{f_s}{2D}$. The transfer function between $x(kT_s)$ and $y_M(kT_s)$ is

$$H_X(z) = \frac{Y_M(z)}{X(z)} = \frac{A(z)}{1 + B(z)A(z)}, \tag{2.7}$$

while the transfer function between $N_{ADC}(kT_s)$ and $y_M(kT_s)$ is

$$H_N(z) = \frac{Y_M(z)}{N_{ADC}(z)} = \frac{1}{1 + B(z)A(z)}, \tag{2.8}$$

where $z = e^{j2\pi f/f_s}$ is a unit sample delay variable. In order to lower the noise from quantization error in the input frequency band, $|H_N(z)|$ must be small for frequencies below $\frac{f_s}{2D}$, while $H_X(z)$ is required to be a function in the form $H_X(z) = z^{-\ell}$ at the same time. This constraint allows the input sequence to the converter to be delayed by an integer number of samples, ℓ, during the quantization process, thus not affecting the magnitude of the sequence inside the converter. Equating (2.7) to $z^{-\ell}$ and solving for $B(z)$ yields

$$B(z) = z^{\ell} - \frac{1}{A(z)}, \tag{2.9}$$

and substituting this result into (2.8) yields

$$H_N(z) = \frac{1}{z^{\ell} A(z)}. \tag{2.10}$$

Note that for $z = e^{j2\pi f/f_s}$, $|z| = 1$. Since it is required that $|H_N(z)|$ be lowpass, $|A(z)|^{-1}$ must also be lowpass, implying large $|A(z)|$ for low frequencies and small $|A(z)|$ for high frequencies. Therefore, $A(z)$ must be a lowpass function. Converters with lowpass $A(z)$ are described as oversampled noise shaping ADCs, and one particular form of this converter is the focus of this thesis.

2.3 The First Order Sigma-Delta ADC

The results derived in the previous section will now be extended to the special case where $A(z)$ is a discrete time integrator:

$$A(z) = \frac{z^{-1}}{1 - z^{-1}}. \tag{2.11}$$

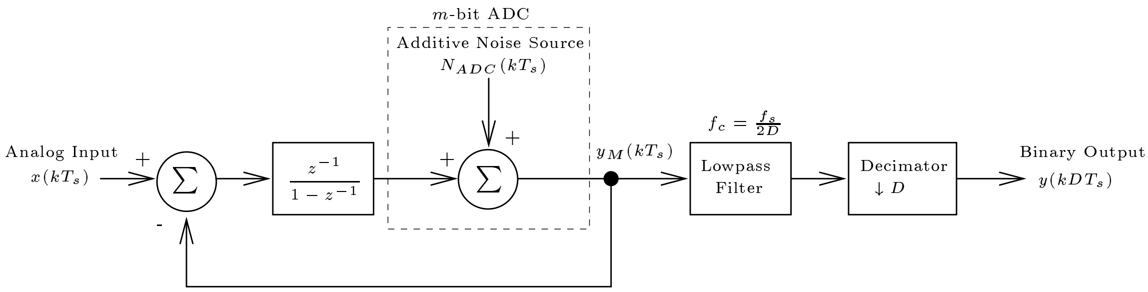

Figure 2.4: First Order Sigma-Delta Converter Block Diagram

The corresponding feedback loop filter is then written

$$B(z) = z^{\ell} - z + 1. \tag{2.12}$$

The value of ℓ is typically chosen so that the amount of delay between the input $x(kT_s)$ and the output $y_M(kT_s)$ is as small as possible, which implies that $\ell = 1$, resulting in $B(z) = 1$. A block diagram of this structure is shown in Fig. 2.4, which is referred to as a first order Sigma-Delta loop.

The benefits of noise shaping on the converter's output resolution will now be examined. Since the transfer function from the input $x(kT_s)$ to the output $y_M(kT_s)$ is simply a delay, it is unchanged from the previous oversampled converter's structure. The advantage of this topology is observed in the change in the power spectral density of the quantization error, which is now processed through the transfer function of the discrete-time integrator. This may be written

$$S_{N_M}(f) = S_N(f)\left[H_N(z)H_N\left(\frac{1}{z}\right)\right]\Big|_{z=e^{j2\pi f/f_s}}, \tag{2.13}$$

where the transfer function from noise source to loop output is

$$H_N(z) = \frac{1}{zA(z)} = 1 - z^{-1}. \tag{2.14}$$

This results in a new noise power spectral density:

$$\begin{aligned} S_{N_M}(f) &= S_N(f)\left[\left(1 - e^{-j2\pi f/f_s}\right)\left(1 - e^{j2\pi f/f_s}\right)\right] \\ &= S_N(f)\left[4\sin^2\left(\pi\frac{f}{f_s}\right)\right]. \end{aligned} \tag{2.15}$$

The new noise power density is simply the old noise power density that is *shaped* by the function $4\sin^2(\pi f/f_s)$, which is a nondecreasing function for $0 < f < \frac{f_s}{2}$. The maximum noise power density in the frequency band of interest, $|f| < \frac{f_s}{2D}$, is then found at $f = \frac{f_s}{2D}$, so that the resulting maximum output noise power spectral density of the converter is

$$\begin{aligned} S_{N_M}(f) &\leq S_N\left(\frac{f_s}{2D}\right)\left[4\sin^2\left(\frac{\pi}{2D}\right)\right] \\ &\leq \frac{1}{12\frac{f_s}{D}}\left(\frac{2V_{max}}{2^{m+\frac{1}{2}\log_2(D)}}\right)^2\left[4\sin^2\left(\frac{\pi}{2D}\right)\right] \quad \frac{V^2}{Hz}, \quad |f| < \frac{f_s}{2D}. \end{aligned} \tag{2.16}$$

This result shows that a first order m-bit Sigma-Delta loop passband output has the same amount of noise power as a non-oversampled ADC utilizing b bits and operating at a sampling frequency of $\frac{f_s}{D}$, where

$$b = m + \frac{1}{2}\log_2(D) - \frac{1}{2}\log_2\left[4\sin^2\left(\frac{\pi}{2D}\right)\right]. \tag{2.17}$$

For large D, $4\sin^2(\frac{\pi}{2D}) \approx (\frac{\pi}{D})^2$, and (2.17) can be approximated

$$b \approx m + \frac{3}{2}\log_2(D) - \log_2(\pi), \quad D \gg 1. \tag{2.18}$$

Extending the example of the previous section, a 3-bit, first order Sigma-Delta with an oversampling ratio $D = 64$ can realize over 10 bits of resolution, where an oversampled converter without noise shaping provides only 6 bits. The graph of figure 2.5 depicts available resolution from a first order Sigma-Delta loop using m-bits and an oversampling ratio D.

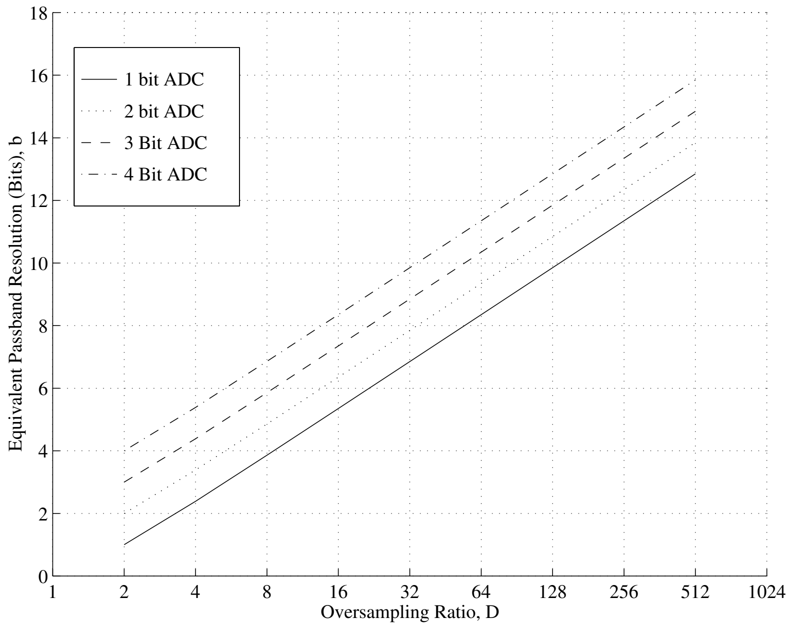

Figure 2.5: First Order Sigma-Delta Passband Resolution

At this point, the oversampled ADC architecture has been modified via feedback to obtain noise shaping, while the band-limited input signal is merely delayed by an integer amount of samples. The result is an ADC structure that increases output resolution using a minimal amount of additional hardware. However, one may ponder how additional feedback loops may be utilized to obtain even greater output resolution. The next several sections are devoted to investigating different Sigma-Delta topologies that attempt to accomplish this task.

2.4 Second Order Loop

In order to improve the performance of the first order Sigma-Delta loop, many designers have attempted to implement higher order noise shaping transfer functions through the use of multiple feedback loops. One of the more popular configurations is the second order Sigma-Delta loop, which is shown in Fig. 2.6 [6, 16, 13, 8, 3]. This structure effectively squares the noise shaping effect of the first order loop by using two integrators, as will be shown below, while still using only one quantizer.

Analyzing this structure, the transfer function from the input, $x(kT_s)$, to

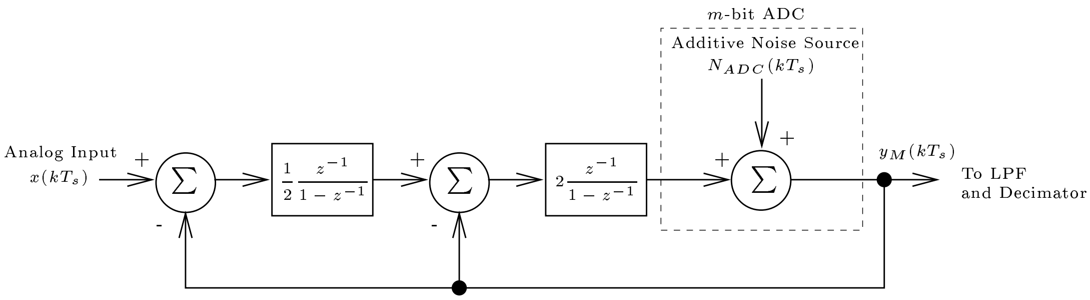

Figure 2.6: Second Order Sigma-Delta Block Diagram

the loop output, $y_M(kT_s)$, is found to be

$$H_X(z) = z^{-2}, \tag{2.19}$$

while the noise transfer function from $N_{ADC}(kT_s)$ to $y_M(kT_s)$ is

$$H_N(z) = \left(1 - z^{-1}\right)^2. \tag{2.20}$$

The noise transfer function is once again a highpass function, attenuating the noise power at low frequencies, while the signal transfer function is now two time delays, not affecting the magnitude of the input signal inside the converter. The noise power spectral density in the frequency band of interest, $|f| < \frac{f_s}{2D}$, may be written as in (2.13) and simplified, yielding the following maximum noise power spectral density:

$$S_{N_M}(f) \leq \frac{1}{12\frac{f_s}{D}}\left(\frac{2V_{max}}{2^{m+\frac{1}{2}\log_2(D)}}\right)^2\left[4\sin^2\left(\frac{\pi}{2D}\right)\right]^2 \quad \frac{V^2}{Hz}, \quad |f| < \frac{f_s}{2D}. \tag{2.21}$$

The end result is a Sigma-Delta converter that provides second order noise shaping, providing the same effective output resolution as a non-oversampled converter utilizing b bits, where

$$\begin{aligned} b &= m + \frac{1}{2}\log_2(D) - \frac{1}{2}\log_2\left(\left[4\sin^2\left(\frac{\pi}{2D}\right)\right]^2\right) \\ &\approx m + \frac{5}{2}\log_2(D) - 2\log_2(\pi), \quad D \gg 1. \end{aligned} \tag{2.22}$$

For example, a 3-bit, second order Sigma-Delta converter can provide over 14 bits of resolution, while the first order converter provided less than 11 bits, and an oversampled converter without noise shaping provided only 6 bits.

While this second order Sigma-Delta loop structure provides the desired second order noise shaping function, it is only conditionally stable, dependent upon the input signal amplitude, total delay in the feedback loops, and the gain of each integrator [13]. This topology also cannot be extended to provide third order or higher noise shaping due to the instability observed in loops that utilize three or more integrators [13]. The following section details structures that are commonly used to accomplish higher order noise shaping without introducing stability problems.

2.5 Higher Order Sigma Delta Converters

The previous sections of this chapter detailed how noise shaping can be used to greatly increase effective converter output resolution. The second order noise shaping discussion showed that the filter transfer function that shaped the noise power due to the quantizer was the square of the function that shaped the noise power in the first order Sigma-Delta loop. A Sigma-Delta loop of order n is designed in such a way that the quantization noise power spectrum is as shown in the following equation:

$$S_{N_M}(f) = \frac{1}{12f_s}\left(\frac{2V_{max}}{2^m}\right)^2\left[4\sin^2\left(\pi\frac{f}{f_s}\right)\right]^n \quad \frac{V^2}{Hz}, \quad |f| < \frac{f_s}{2}. \tag{2.23}$$

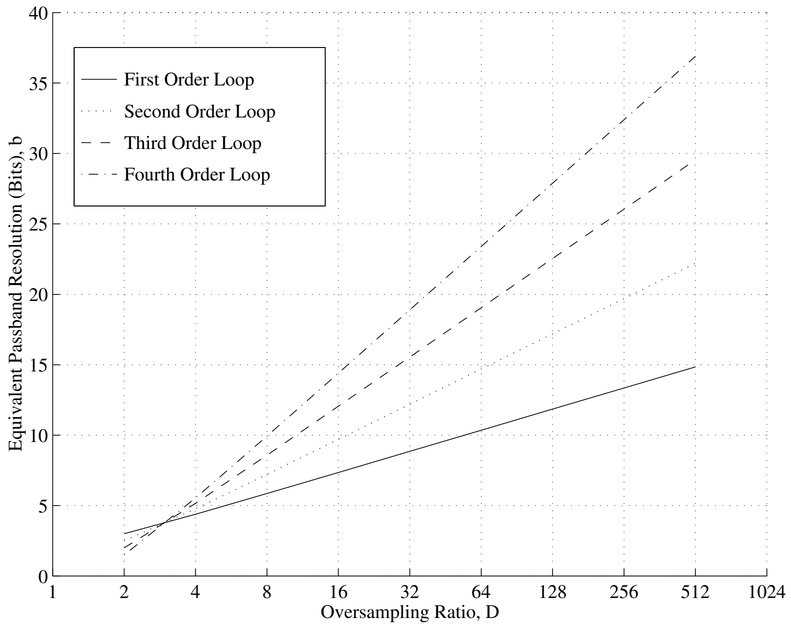

Figure 2.7: Output Resolution of Various 3-Bit Sigma-Delta Loop Orders

The resulting output resolution in bits is then given by

$$\begin{aligned} b &= m + \frac{1}{2}\log_2(D) - \frac{1}{2}\log_2\left(\left[4\sin^2\left(\frac{\pi}{2D}\right)\right]^n\right) \\ &\approx m + (n+\frac{1}{2})\log_2(D) - n\log_2(\pi), \quad D \gg 1, \end{aligned} \tag{2.24}$$

assuming ideal lowpass filters at the converter output. A plot of the effective output resolution for a Sigma-Delta converter of loop order n at $m = 3$ is shown in Fig. 2.7 versus oversampling ratio, D.

In order to avoid stability problems described in the previous section, cascading multiple integrators is avoided in the design of Sigma-Delta converters. Since the single loop Sigma-Delta has proven to be unconditionally stable [13], it would be wise to construct a high order Sigma-Delta by combining multiple first order loops. This architecture is typically referred to as a *MASH*, or mulitstage noise shaping, architecture [4, 15]. In this architecture, successive first order loops are driven by a value proportional to the quantization noise from the previous loop, and each loop's output is passed through a digital filter before being summed to

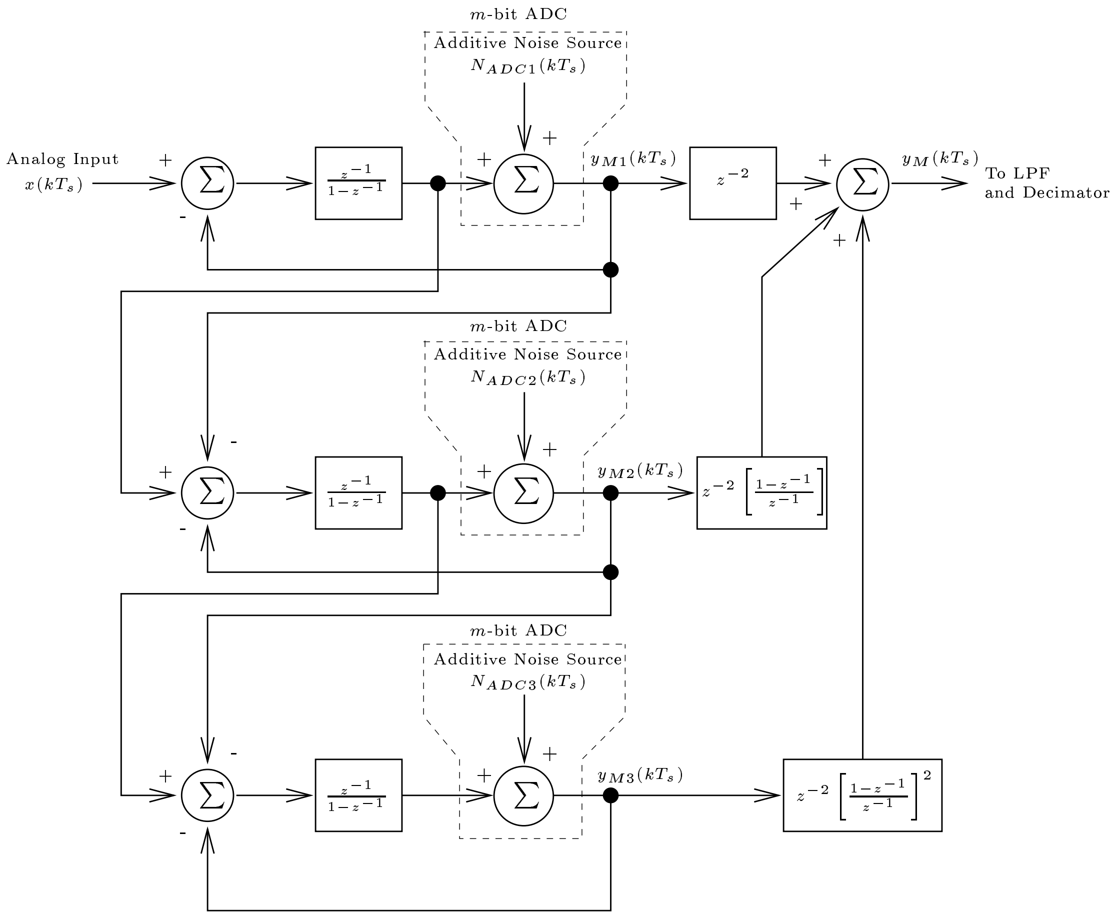

Figure 2.8: Third Order Multistage Sigma-Delta Block Diagram

form the converter output. A block diagram for a three stage, third order noise shaping Sigma-Delta converter is shown in Fig. 2.8.

In the quantitative analysis of this architecture, it is first noted that the input to each loop, with the exception of the first, is the negative of the uniform noise introduced by the quantizer of the previous stage. The input signal, $x(kT_s)$, is only present in the first loop, and thus the transfer function from the input signal to the converter output, $y_M(kT_s)$, is the same as derived for the single loop

Sigma-Delta, with two additional delays due to the first loop filter:

$$H_X(z) = z^{-3}. \tag{2.25}$$

Deriving the noise transfer function for this structure is more difficult, due to the three individual noise sources representing the three quantizers. In order to simplify the analysis, the three loop output equations can be written as

$$\begin{aligned} y_{M1}(z) &= z^{-1}X(z) + (1 - z^{-1})N_{ADC1}(z) && (2.26) \\ y_{M2}(z) &= z^{-1}(-N_{ADC1}(z)) + (1 - z^{-1})N_{ADC2}(z) && (2.27) \\ y_{M3}(z) &= z^{-1}(-N_{ADC2}(z)) + (1 - z^{-1})N_{ADC3}(z). && (2.28) \end{aligned}$$

By summing each of these outputs after they are passed through their respective loop filters, the final converter output can be written as

$$\begin{aligned} y_M(z) &= z^{-3}X(z) + z^{-2}(1 - z^{-1})N_{ADC1}(z) - z^{-2}(1 - z^{-1})N_{ADC1}(z) \ldots \\ &\quad + z^{-1}(1 - z^{-1})^2 N_{ADC2}(z) - z^{-1}(1 - z^{-1})^2 N_{ADC2}(z) \ldots \\ &\quad + (1 - z^{-1})^3 N_{ADC3}(z) \\ &= z^{-3}X(z) + (1 - z^{-1})^3 N_{ADC3}(z). && (2.29) \end{aligned}$$

From this result, it can be seen that the additional noise shaping of this structure is derived from the loop filters following each individual first order loop, and the subsequent summation of these individually filtered loop outputs.

The noise power density at the lowpass filter output may now be written as was done in previous sections to obtain

$$S_{N_M}(f) \leq \frac{1}{12\frac{f_s}{D}} \left(\frac{2V_{max}}{2^{m+\frac{1}{2}\log_2(D)}} \right)^2 \left[4\sin^2\left(\frac{\pi}{2D}\right) \right]^3 \quad \frac{V^2}{Hz}, \quad |f| < \frac{f_s}{2D}. \tag{2.30}$$

Note that the output noise power is only a function of the quantization noise introduced in the third loop, assuming that the cancellation of terms seen in (2.29) is exact. For non-ideal loop filters or mismatched first order loops, this cancellation will not be exact, resulting in distortion in the final converter output. This phenomenon will be examined in detail in chapters 3 and 4.

A graphical interpretation of the resulting output resolution of this third order noise shaping structure is shown in Fig. 2.7 for the case $n = 3$. For the specific example used in the preceding sections (oversampling ratio $D = 64$ and $m = 3$ in each loop), this structure provides over 19 bits of output resolution.

Up to this point, a variety of oversampled and noise shaping converter architectures have been discussed and analyzed from the converter input to output using ideal lowpass filters and decimators. However, this discussion is not complete without an introduction to the design and resulting limitations of the more popular non-ideal digital postprocessing techniques, as they pertain to a given Sigma-Delta noise shaping architecture. The following sections describe the design and analysis of the filters and decimators that are typically used in Sigma-Delta converters.

2.6 Digital Filter and Decimation Stages

Once a Sigma-Delta converter has been designed to provide the desired noise shaping, lowpass filter and decimation stages are then incorporated in the ADC architecture. These stages attenuate the high frequency quantization noise that was produced by the Sigma-Delta converter, while reducing the output sample rate to a desired level. The high frequency noise must be attenuated prior to decimation to avoid aliasing noise back into the desired low frequency passband. The most common method of implementing these stages is to first design a comb filter utilizing decimation by N, followed by an FIR filter utilizing decimation by $\frac{D}{N}$. The comb filter is used as the first stage filter due to it's simplicity in construction, since it is merely an FIR filter with unity coefficients. This requires no multiplication or filter coefficient storage in memory, which is attractive due to the high input rate to the filter from the Sigma-Delta converter. If a filter with non-unity coefficients were used, it would have to perform computationally intensive algorithms at the high sampling rate, f_s [18]. However, there are several drawbacks to using a comb filter: it is not extremely effective at removing large quantities of high frequency noise, it has a nonflat passband characteristic, and it has a poor passband to stopband transition. For these reasons, a second FIR filter with non-unity coefficients is designed to follow the comb filter. This filter is designed to compensate the passband of the comb filter and to provide a sharp passband to stopband cutoff, and is able to use non-unity coefficients due to the reduced sample rate from the preceding comb filter. The following sections detail the design of the comb and FIR filter stages.

2.6.1 Comb Filter

The comb filter is an FIR filter with all unity coefficients described by the difference equation

$$y_C(kT_s) = \sum_{i=0}^{N-1} y_M((k-i)T_s), \tag{2.31}$$

where $y_M(nT_s)$ is the input sample to the filter at time nT_s, and $y_C(nT_s)$ is the output sample from the filter at time nT_s. This results in the discrete time model

$$\begin{aligned} Y_C(z) &= \sum_{i=0}^{N-1} z^{-i} Y_M(z) \\ &= \left(\frac{1-z^{-N}}{1-z^{-1}}\right) Y_M(z), \end{aligned} \tag{2.32}$$

where $Y_M(z)$ is the z-transform of the input sequence to the comb filter from the Sigma-Delta converter output. In order to provide additional filtering, multiple comb filter stages are cascaded, since a single comb filter stage typically does not provide adequate attenuation for high frequencies. This transfer function may be written

$$C^k(z)\Big|_{z=e^{j2\pi f/f_s}} = \left(\frac{1-z^{-N}}{1-z^{-1}}\right)^k\Bigg|_{z=e^{j2\pi f/f_s}}$$
$$= \left[\frac{\sin(N\pi\frac{f}{f_s})}{\sin(\pi\frac{f}{f_s})}e^{-j(N-1)\pi f/f_s}\right]^k. \tag{2.33}$$

The output spectral density from this filter, given the input spectral density from a Sigma-Delta converter of order n, shown in (2.23), is then

$$S_{N_C}(f) = S_{N_M}(f)\left|C^k(e^{j2\pi f/f_s})\right|^2$$
$$= \frac{1}{12f_s}\left(\frac{2V_{max}}{2^m}\right)^2\left[4\sin^2\left(\pi\frac{f}{f_s}\right)\right]^n\left[\frac{4\sin^2(N\pi f/f_s)}{4\sin^2(\pi f/f_s)}\right]^k$$
$$\frac{V^2}{Hz}, \quad |f| < \frac{f_s}{2}. \tag{2.34}$$

In order for this function to provide adequate attenuation at high frequencies, k must be greater than n, resulting in

$$S_{N_C}(f) = \frac{1}{12f_s}\left(\frac{2V_{max}}{2^m}\right)^2\frac{\left[4\sin^2(N\pi f/f_s)\right]^k}{\left[4\sin^2(\pi f/f_s)\right]^{k-n}}\ \frac{V^2}{Hz}, \quad k>n, \quad |f|<\frac{f_s}{2}. \tag{2.35}$$

A plot of this power spectral density is given in Fig. 2.9, where the constant $\frac{1}{12f_s}\left(\frac{2V_{max}}{2^m}\right)^2$ is unity for simplicity, and $k=4, n=3$, and $N=16$. By examining this figure, it can be seen that a comb filter with only one stage in excess of the order of noise shaping in the Sigma-Delta converter, i.e. $k=n+1$, does not remove all of the high frequency quantization noise. However, if the frequency band of interest, $|f|<\frac{f_s}{2D}$, lies below the first null of the comb filter, $f=\frac{f_s}{N}$, and the ratio $\frac{D}{N}$ is chosen correctly, then the amount of quantization noise in the passband after the final decimation by D should be sufficiently small to produce the desired output resolution. This is due to the fact that the aliased noise bands at the final converter output after decimation by D will result from the small

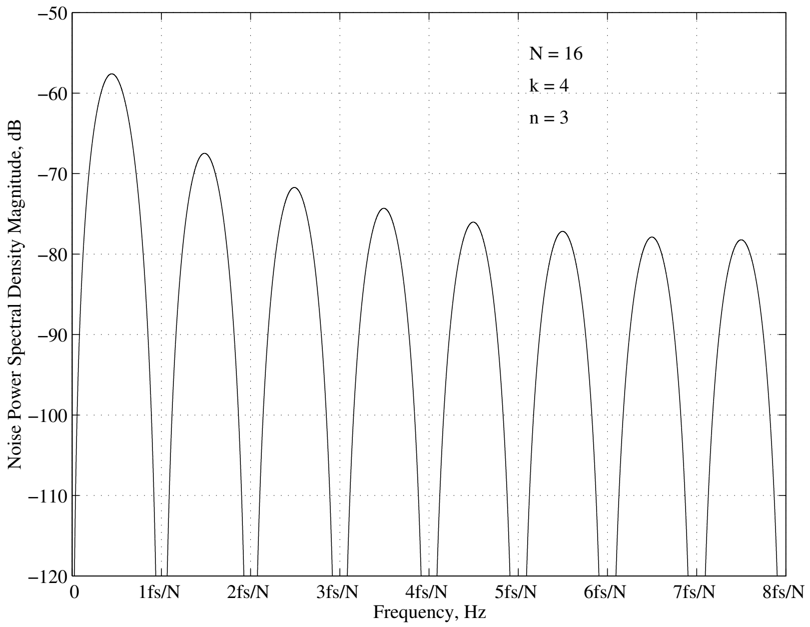

Figure 2.9: Four Stage Comb Filter Power Spectral Density

frequency bands centered around each null of the comb filter transfer function. In Fig. 2.10, the darkened areas are the portions of the noise power spectral density function that are aliased to the baseband frequency range after decimation by D. The in-band noise power density after decimation by D is calculated by summing the noise power density from each aliased band:

$$\begin{aligned} S_{N_C}(f) &= \sum_{i=0}^{N-1} S_{N_M}\left(f + \frac{if_s}{N}\right) |C^k(e^{j2\pi(f+if_s/N)/f_s})|^2 \\ &= \sum_{i=0}^{N-1} \frac{1}{12f_s}\left(\frac{2V_{max}}{2^m}\right)^2 \frac{\left[4\sin^2(N\pi(f+\frac{if_s}{N})/f_s)\right]^k}{\left[4\sin^2(\pi(f+\frac{if_s}{N})/f_s)\right]^{k-n}} \quad \frac{V^2}{Hz}, \\ & \quad k > n, \quad |f| < \frac{f_s}{2N}. \end{aligned} \tag{2.36}$$

The maximum in-band noise power spectral density may then be calculated using the above equation evaluated at $f = \frac{f_s}{2D}$. Therefore, given an equivalent output noise power spectral density requirement, the corresponding ratio $\frac{D}{N}$ may be chosen, while minimizing the number of comb filter stages.

At this point, little discussion has been focused on choosing the number of

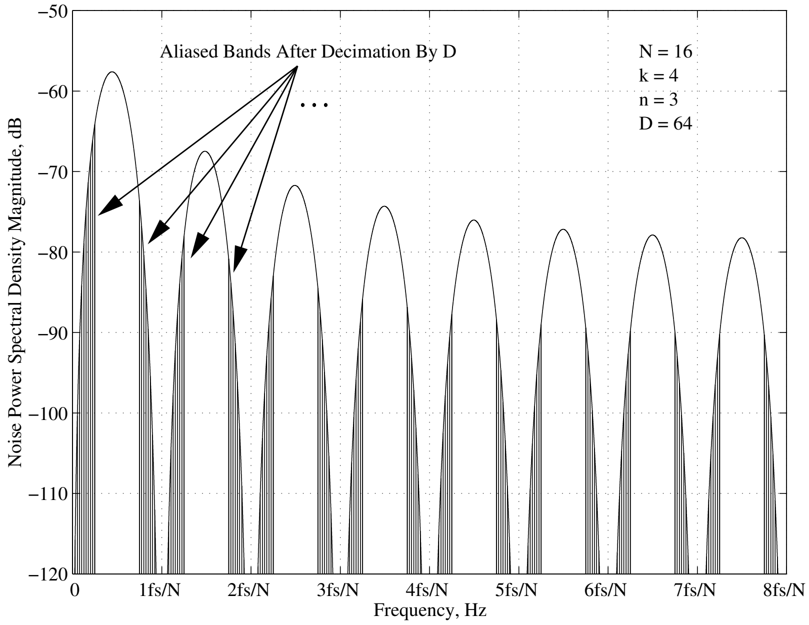

Figure 2.10: Aliased Noise Power Regions From A Four Stage Comb Filter

comb filter stages, k, for a given Sigma-Delta converter topology employing noise shaping of order n. Typically, k is chosen to be just large enough to provide a lowpass function, implying $k = n + 1$, since it is required to use an additional digital lowpass filter with non-unity coefficients to compensate the desired filter passband and transition region. Additional comb filter stages decrease inband noise power, while at the same time they worsen passband response and cutoff transition, thus requiring more coefficients in the compensating output FIR filter. The typical solution is to use a $k = n+1$ stage comb filter followed by an FIR filter with the minumum number of coefficients required to obtain the desired output resolution.

Although the comb filter presented above removes a majority of the shaped high frequency quantization noise, it has several drawbacks, including a non-flat passband loss and a poor transition between the stop and pass bands. Also, from the preceding discussion, the comb filter is only effective when the total decimation rate, D, is greater than the decimation rate implemented in the comb filter, N. This leaves the transition frequency range $f_s/2D < f < f_s/N$ unattenuated after the comb filter decimation by N; this band must be further shaped by the output FIR lowpass filter, and then decimated by $\frac{D}{N}$. The following section explores this important output FIR lowpass filter that is typically implemented to complete the Sigma-Delta architecture.

2.6.2 FIR Filter

At this point, the converter design has included one or more Sigma-Delta loops providing noise shaping of order n, and the corresponding first lowpass filter and decimator, which is typically a $n+1$ stage comb filter. The decimation by N achieved by this comb filter is required to be less than the desired oversampling ratio, D, in order for the comb filter to provide the desired output resolution. The fact that N is less than D, coupled with the fact that the comb filter has a poor transition region and passband response, results in a need for a second lowpass filter and final decimation by $\frac{D}{N}$. The following section will now detail how these requirements are satisfied.

The final stage of a Sigma-Delta converter is typically a digital finite impulse response (FIR) filter with all coefficients not identically equal to unity, so that multipliers and coefficient memory is required. For this reason, the size of this filter is required to be as small as possible while providing the desired output resolution. However, since the final output resolution of the converter is directly related to the effectiveness of this filter, the number of coefficients cannot be reduced arbitrarily.

The first task that this filter must accomplish is to compensate for the non-flat passband of the comb filter. Since the signal component of $y_M(kT_s)$, the input sequence to the comb filter, is band limited, $(S_{X_M}(f) = 0, |f| > \frac{f_s}{2D})$, the signal

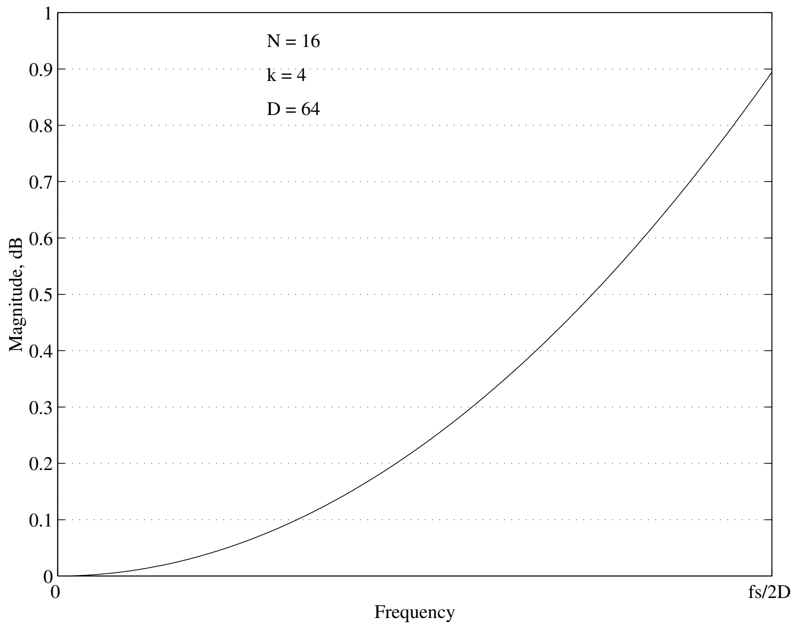

Figure 2.11: Ideal FIR Filter Passband Requirement

power density at the output of the comb filter can be written as

$$S_{X_C}(f) = \begin{cases} S_{X_M}(f)\left[\frac{4\sin^2(N\pi f/2f_s)}{4\sin^2(\pi f/2f_s)}\right]^k & \frac{V^2}{Hz}, \quad |f| < \frac{f_s}{2D} \\ 0 & \frac{V^2}{Hz}, \quad |f| > \frac{f_s}{2D}. \end{cases} \tag{2.37}$$

The result is that the input power spectrum has been altered by the comb filter term in the square brackets. In order to compensate for the resulting comb filter passband shape, the passband of the FIR filter must have a magnitude squared response that is the inverse of the comb filter magnitude squared response in (2.37), as shown in Figure 2.11.

The second task of the FIR filter is to provide sharp cutoff between the passband and stopband. This transition is directly related to the number of coefficients used in designing the filter, as is the maximum stopband attenutation, which is the determining factor in the final output resolution of the converter. Figure 2.12 shows the desired ideal magnitude response of this filter for the case $\frac{D}{N} = 4$ and $k = 4$, and when the desired stopband attenuation is 96 dB, as in the Motorola DSP56ADC16.

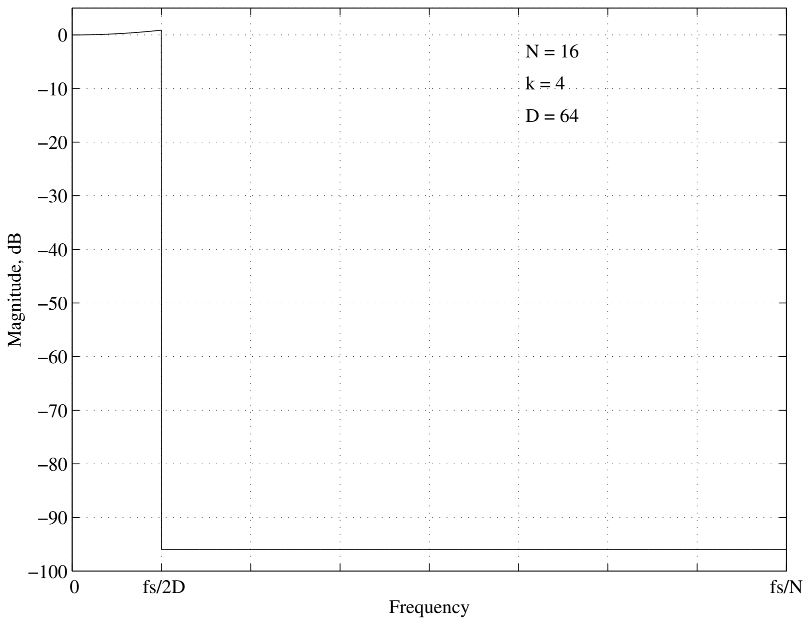

Figure 2.12: Ideal FIR Filter Frequency Response

Designing an FIR filter from these ideal specifications is easily accomplished using windowing functions [17]. The next section will present a Sigma-Delta design example that utilizes a 255 coefficient Kaiser window to produce an example final stage FIR filter.

2.7 Design Example: A Three Stage, Third Order Noise Shaping Sigma-Delta Converter Simulation

In order to better understand the internal operation of a Sigma-Delta converter, a simulation for a three stage, third order noise shaping converter was written. The code was written in *MATLAB* [14], and proved to be an invaluable tool, allowing many of the internal functions of the converter to be simulated at the component level. The primary reason for simulating the converter was to accurately observe the distortion effects that a real quantizer introduces in the output samples. While the additive white noise source model is convenient and provides useful insight into the design of the converter, it cannot reflect the nonlinear behavior of the device.

Table 2.1: Motorola DSP56ADC16 Specifications

Parameter	Value
Output Resolution	16-bits (96 dB typ.) at 100 kHz from FIR filter
	12-bits (72 dB typ.) at 400 kHz from comb filter
SNR	90dB
In-Band Ripple	< 0.001 dB
Output Sampling Rate	8 kHz to 100 kHz (FIR Filter)
	32 kHz to 400 kHz (Comb Filter)
Max. Input Sample Rate	6.4 MHz
Max. Internal Clock Rate	12.8 MHz
Input Signal Range	3.5 V_{pp} differential
Power Consumption	300 mW at 100 kHz sampling rate

The simulation produces true quantization, thus allowing the nonlinearity of the quantizer to affect the output as it does for a real converter. The simulation also allows sample sequences internal to the converter to be examined, which is typically not avaliable when physically constructing a Sigma-Delta converter. The following section details the simulation.

The Sigma-Delta simulation is based on the Motorola DSP56ADC16, a 16-bit, three stage, third order noise shaping Sigma-Delta analog-to-digital converter [18, 9]. The technical specifications of this converter are given in Table 2.1. This converter uses three first order noise shaping loops, each with a 1-bit ADC internal to the loop, as shown in Fig. 2.8. These loops operate at an oversampling ratio of $64 : 1$, and are followed by a four stage comb filter utilizing $16 : 1$ decimation. The final stage of the converter is a 255 coefficient FIR filter, utilizing $4 : 1$ decimation. The output of the converter can be obtained directly from the output of the comb filter, after decimation by 16, or from the FIR filter output, after a total decimation of 64. This converter was chosen to be simulated due to it's commercial availability and ample documentation on internal construction.

The only portion of the simulation which necessitates additional discussion is the design of the 255 coefficient FIR filter. The ideal FIR filter must compensate

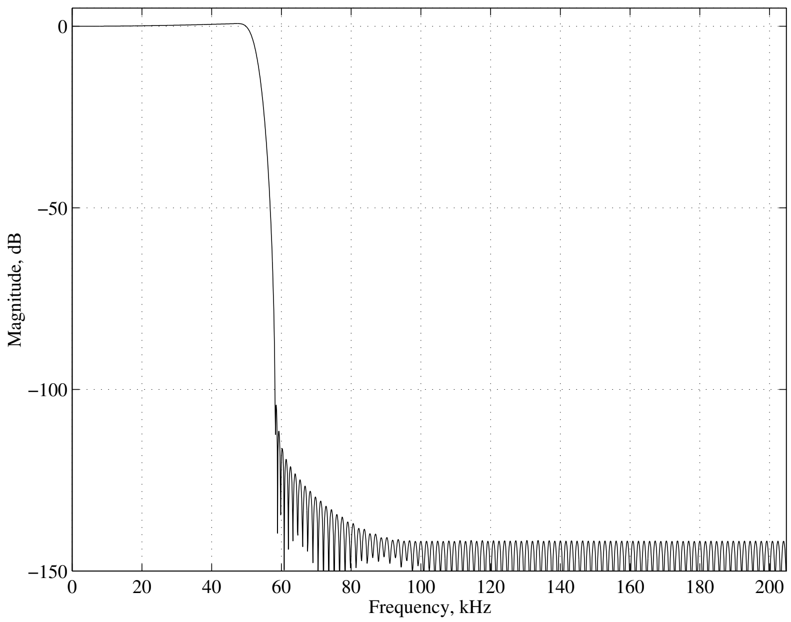

Figure 2.13: 255 Coefficient FIR Filter

for the non-flat passband of the comb filter while providing a sharp transition from passband to stopband, as discussed in sections 2.6.1 and 2.6.2. The filter used in the simulation was designed by windowing the ideal lowpass filter function with a 255 point Kaiser window, and modifying the side lobe attenuation factor β until the desired minimum stopband attenuation of 96dB was attained. The resulting FIR filter response is shown in Fig. 2.13. An expanded view of the filter passband is shown in Fig. 2.14.

At this point, many forms of the Sigma-Delta converter have been discussed, along with the corresponding filtering necessary to produce the desired output. The focus has also now been placed on the three stage, third order noise shaping converter, specifically the Motorola DSP56ADC16, and a component level simulation has been developed. In the next chapter, the physical limitations of several of these components will be examined, and the simulation will be modified to include these limitations. The results from this modified simulation will then provide insight into the effects of component nonidealities, resulting in physical modifications that can be made to this converter to reduce such effects.

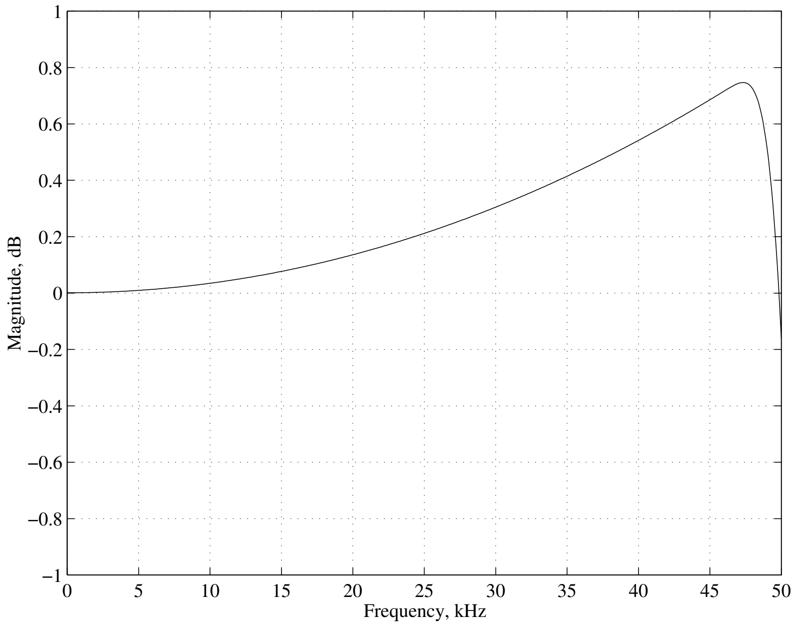

Figure 2.14: Passband of FIR Filter Utilizing 255 Coefficients

CHAPTER 3

Error Mechanisms in Sigma-Delta Converters

The focus of this chapter is to study, in detail, the possible sources of error in the output samples of a Sigma-Delta converter. The converter is examined from the component level, where error mechanisms are related to specific component limits or tolerances. These component level error mechanisms are then incorporated into the Sigma-Delta converter simulation for study. A three stage, third order noise shaping Sigma-Delta converter model is used to illustrate various distortion mechansisms. The simulation of this converter was briefly introduced in the previous chapter.

Although there are many possible error mechanisms in the Sigma-Delta architecture, concentration is on the discrete-time integrator inside each individual first order loop. This portion of the converter is the only area where component values have a significant effect on signal distortion, and also the major region in which analog signals are present in the converter. The remainder of the converter, including the filters and decimators, are all implemented digitally. Therefore, mismatched component values in these devices do not significantly contribute to the output distortion, assuming the digital filter coefficients are quantized with sufficient resolution.

The analysis begins with an introduction to switched capacitor circuits, which are well suited to implementation in Sigma-Delta converters. This analysis provides background for the study of switched capacitor discrete-time integrators, followed by description of four possible sources of error from the integrators: voltage dependent capacitors, finite amplifier output slew rate, and mismatched amplifier open loop gains and capacitor values. Further study of the errors due to amplifier open loop gain and capacitor mismatches, along with a compensating topology for these error sources, is the focus of chapter 4.

3.1 Switched Capacitors

Since the Sigma-Delta converter is an oversampled converter, the design of the discrete-time integrator is ideally suited for a switched capacitor implementation [4, 16]. The fact that the converter is oversampled implies the clock rate is much higher than the maximum input signal frequency. Switched capacitor filters are also implemented in this manner, so that the average charge passed through the switched capacitor over time is equal to a fixed value. This results in the switched capacitor circuit acting as a fixed resistance, assuming that the input signal does not change significantly over one cycle time T_s. This allows designers to implement circuits with large resistor values on small die sizes while dissipating little power. This section gives a brief analysis of a generic switched capacitor circuit, followed by a more detailed analysis of the switched capacitor integrator in the next section.

Figure 3.1 shows a switched capacitor circuit with the corresponding clock cycles for the switches. The switches are typically metal oxide semiconductor (MOS) switches, which are switched very quickly. The clock phases are designed to be non-overlapping, so that the input and output signals are never connected to the capacitor at the same time. During the time that ϕ_1 is high, the input voltage is across the capacitor C, and the amount of charge stored in the capacitor, assuming ϕ_1 is active long enough for C to fully charge, is

$$q_i = CV_i. \tag{3.1}$$

When ϕ_2 is active, the capacitor is charged to the voltage V_o. The resulting charge in the capacitor is

$$q_o = CV_o. \tag{3.2}$$

This results in an average current into the load in one clock period of

$$i_{ave} \cong \frac{\Delta q}{\Delta t} = \frac{C(V_i - V_o)}{T_s} = \frac{V_i - V_o}{R_{SC}}, \tag{3.3}$$

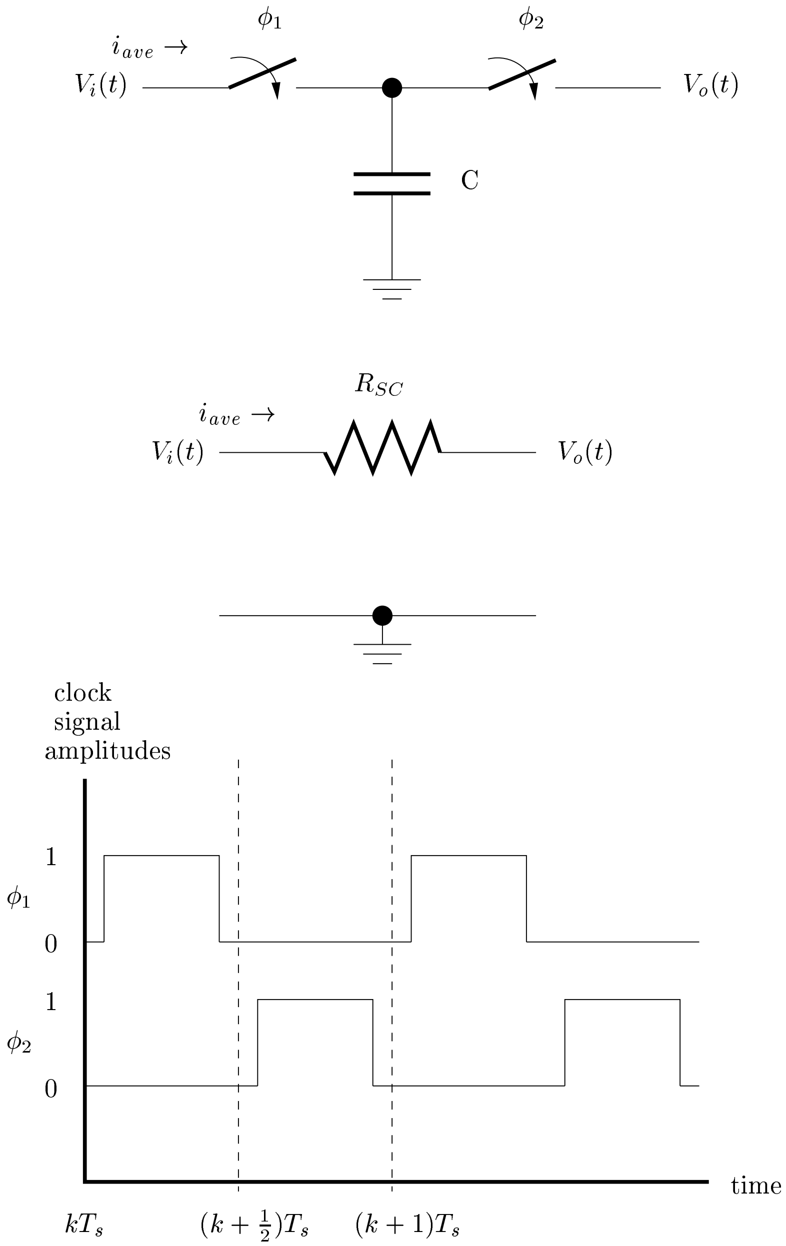

Figure 3.1: Switched Capacitor Circuit, Equivalent Circuit, and Timing Diagram

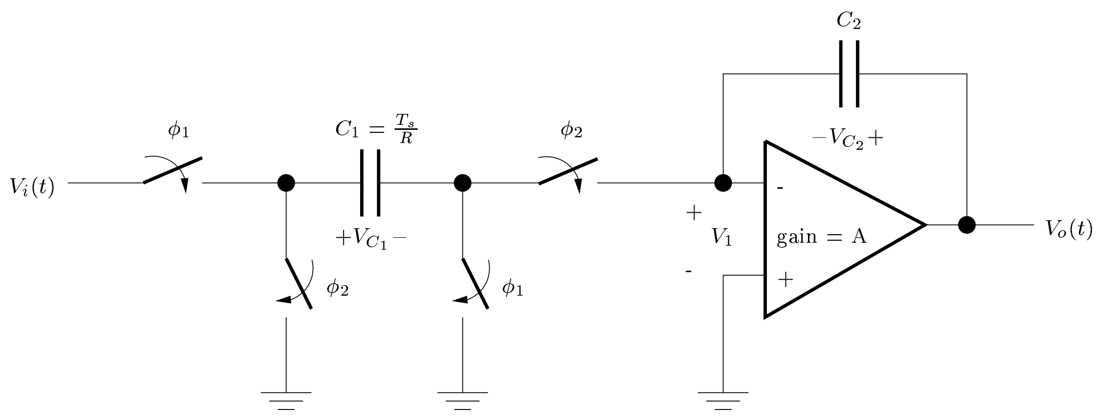

Figure 3.2: Non-Inverting Switched Capacitor Discrete-Time Integrator

and the resulting resistance of this circuit is

$$R_{SC} = \frac{T_s}{C} = \frac{1}{f_s C}. \tag{3.4}$$

Using this circuit, nearly any analog filter can be implemented in MOS technology using switched capacitor circuits for resistors and MOS capacitors and operational amplifiers, given that the switching speed is much higher than the maximum input frequency. In particular, the discrete-time integrator in the first order Sigma-Delta loop is an ideal application for switched capacitor components.

3.2 Switched Capacitor Integrators

Sigma-Delta converter architectures employs multiple discrete-time integrators with the ideal transfer function given in (2.11). The component level model of this device is the focus of this section.

A typical switched capacitor non-inverting discrete-time integrator are shown in Fig. 3.2. To begin the analysis of the switched capacitor integrator, assume ϕ_1 is active, so that the input voltage V_i is across the input capacitor C_1. Assuming that the input has not changed during the time that ϕ_1 is active

($V_i((k+\frac{1}{2})T_s) = V_i(kT_s)$), the charge stored in C_1 when ϕ_1 switches low is

$$Q_{C_1}((k+\tfrac{1}{2})T_s) = C_1 V_i(kT_s). \tag{3.5}$$

The charge stored in C_2 at this time is also written, assuming that the output has not changed while ϕ_1 is active ($V_o((k+\frac{1}{2})T_s) = V_o(kT_s)$):

$$Q_{C_2}((k+\tfrac{1}{2})T_s) = C_2 V_o(kT_s) - C_2 V_1(kT_s). \tag{3.6}$$

Given that the transfer function of the operational amplifier is

$$V_o = -AV_1, \tag{3.7}$$

where A is the open loop gain of the amplifier, (3.6) is rewritten

$$Q_{C_2}((k+\tfrac{1}{2})T_s) = C_2 V_o(kT_s) + \frac{C_2}{A} V_o(kT_s). \tag{3.8}$$

When ϕ_2 is active, the charge stored in C_1 is discharged until the voltage across C_1 is equal to the voltage at the input terminals of the amplifier, assuming that ϕ_2 is active long enough. The charge stored in C_1 at time $(k+1)T_s$ is now written:

$$Q_{C_1}((k+1)T_s) = -C_1 V_1((k+1)T_s) = \frac{C_1}{A} V_o((k+1)T_s). \tag{3.9}$$

The charge stored in C_2 is also written:

$$\begin{aligned} Q_{C_2}((k+1)T_s) &= C_2 V_o((k+1)T_s) - C_2 V_1((k+1)T_s) \\ &= C_2 V_o((k+1)T_s) + \frac{C_2}{A} V_o((k+1)T_s). \end{aligned} \tag{3.10}$$

The amount of charge moved through C_1 during one cycle (with period T_s)

is now written by subracting (3.5) from (3.9):

$$\Delta Q_{C_1} = \frac{C_1}{A} V_o((k+1)T_s) - C_1 V_i(kT_s). \tag{3.11}$$

This same procedure is used for C_2:

$$\Delta Q_{C_2} = C_2 V_o((k+1)T_s) + \frac{C_2}{A} V_o((k+1)T_s) - C_2 V_o(kT_s) - \frac{C_2}{A} V_o(kT_s). \tag{3.12}$$

By the law of conservation of charge at a node, we equate the charges entering the node V_1, with the result that

$$-\Delta Q_{C_1} = \Delta Q_{C_2}, \tag{3.13}$$

and substitution yields

$$\begin{aligned} -C_1 V_o((k+1)T_s) + AC_1 V_i(kT_s) &= AC_2 V_o((k+1)T_s) + C_2 V_o((k+1)T_s) \\ &- AC_2 V_o(kT_s) - C_2 V_o(kT_s). \end{aligned} \tag{3.14}$$

By solving (3.14) for V_o/V_i and taking the z-transform, the discrete-time model for the switched capacitor integrator is written

$$\frac{V_o(z)}{V_i(z)} = \frac{\left[\frac{AC_1}{AC_2+C_2+C_1}\right] z^{-1}}{1 - \left[\frac{AC_2+C_2}{AC_2+C_2+C_1}\right] z^{-1}}. \tag{3.15}$$

This equation allows a simulation to model switched capacitor discrete-time integrators, such as those found in Sigma-Delta loops, given specific component values and amplifier parameters. It should be noted that when $C_1 = C_2$, and as the open loop amplifier gain approaches infinity, this model approaches the ideal discrete-time model for an integrator given in (2.11).

3.3 Limitations of the Switched Capacitor Integrator

Although using switched capacitors in a Sigma-Delta converter conserves die space and makes efficient use of the converter's oversampled structure, it is not without faults. There are physical limitations that create nonlinearities in the converter's output due to the speed of operation and small component sizes. This section explores four of the most common limitations of switched capacitor integrators and how these limitations are modeled in this computer simulation of the Sigma-Delta converter.

3.3.1 Voltage Dependent Capacitors

Upon close examination of (3.15), one can see that as the open loop gain of the amplifier approaches infinity, a factor C_1/C_2 remains as a gain on the input sample $V_i((k-1)T_s)$. Any small nonlinear deviations in capacitor values has the potential to cause nonlinear gain in the integrator, regardless of the corresponding amplifier's open loop gain, resulting in distortion in the converter output. The nonlinearity is a result of the capacitance being dependent upon the voltage across the capacitor terminals [3, 11, 7].

Simulating voltage dependent capacitors in switched capacitor circuits has been widely documented [3, 11, 7], with the most common simulation method being the substitution of an equation that is quadratic in the capacitor voltage for the capacitor value, i.e.

$$C = \alpha_0 + \alpha_1 V_c + \alpha_2 V_c^2. \tag{3.16}$$

This substitution is made in (3.15) for C_1 and C_2 for accurate Sigma-Delta converter simulation. The quantity α_0 represents the nominal capacitance, and is typically on the order of 10 pF for switched capacitor circuits.

Figure 3.3 shows the output spectrum of the three stage, third order noise shaping Sigma-Delta simulation (section 2.7) when the capacitors for each of the

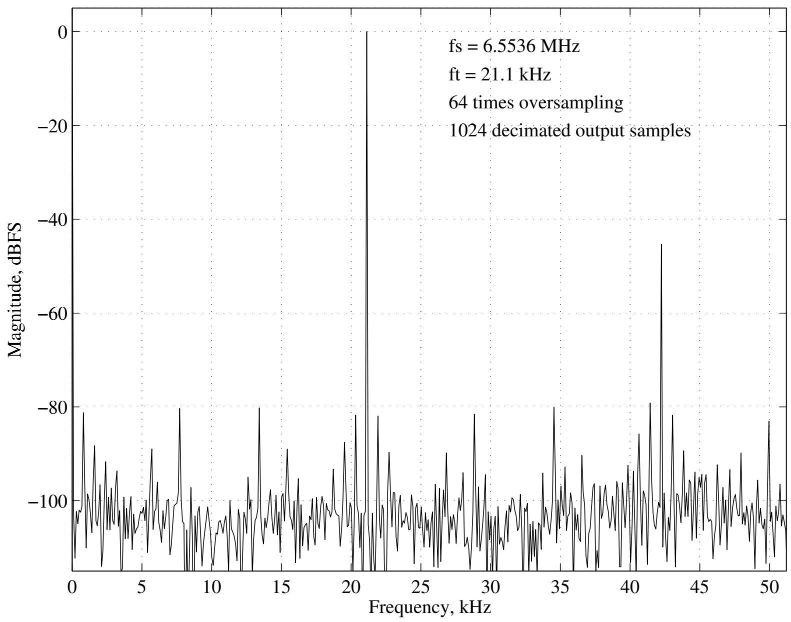

Figure 3.3: Simulated Output Spectrum Using Nonlinear Capacitors

three switched capacitor integrators are modeled as in (3.16). The output spectrum is shown for a sinusoidal input with no harmonics. The nonlinearity parameters α_1, α_2, and α_3 were chosen to produce visible output distortion, but were not based on measured data. This is due to the fact that although this nonlinearity produces large amounts of harmonic and intermodulation distortion in the output spectrum of the Sigma-Delta converter, it has been proven that measuring the nonlinear capacitor coefficient values is quite difficult [7]. Therefore, although the Sigma-Delta converter simulation may be modified to include these parameters, additional analysis of this distortion mechanism is not covered in this thesis.

3.3.2 Finite Amplifier Output Slew Rate

Limitations of the switched capacitor integrator are extremely dependent on the limitations of the operational amplifier used in the circuit. Any nonlinearities in the amplifier produces nonlinearities in the output of the integrator, and in this case, the Sigma-Delta converter output. One of the most common nonlinearities of an amplifier is due to a finite output slew rate. The amplifier has internal parasitic capacitances throughout it's architecture that must be charged and discharged as the output of the amplifier changes state. Therefore, when there is a sudden change at the input to the amplifier, the output of the amplifier does not change instantaneously; it has a finite amount of time during which it changes state. This phenomenon is known as slew limiting, and is simulated in switched capacitor integrator computer models.

Due to the fact that the switched capacitor integrator produces discrete output samples, simulation of amplifier slew limiting is achieved by examining the change in output voltage during the period of time that the output is allowed to change; specifically, the time that ϕ_2 is high in Fig. 3.1. If the difference between the previous and current output voltages of the amplifier is greater than the largest voltage that the amplifier can slew through during that time, the integrator output is not ideal; it is the largest voltage that the amplifier provides given it's specific slew rate. Otherwise, the amplifier output is the ideal output. Figure 3.4 is a graph of this description. Amplifiers used in switched capacitor circuits for Sigma-Delta converters typically have slew rates on the order of 100 to 350 volts per microsecond [4, 8, 3].

The error mechanism described in this section is overcome by designing the operational amplifier with adequate slew rate for the specific integrator implementation. By simulating the ideal switched capacitor integrator in the desired Sigma-Delta converter topology, the amplifier output slew rate necessary to avoid distortion due to slew limiting is predicted. Figure 3.5 shows the voltage that integrator amplifier must slew through for an ideal switched capacitor integrator. From this figure, the maximum difference in voltage between adjacent samples is ± 2 Volts. This number, along with the known time that clock ϕ_2 is high (in this case, 40% of T_s, or 61 nanoseconds), shows that this amplifier must have a minimum slew rate of 33 Volts per microsecond. However, since this slew rate is calculated using ideal components, it is common to design the amplifier to have a

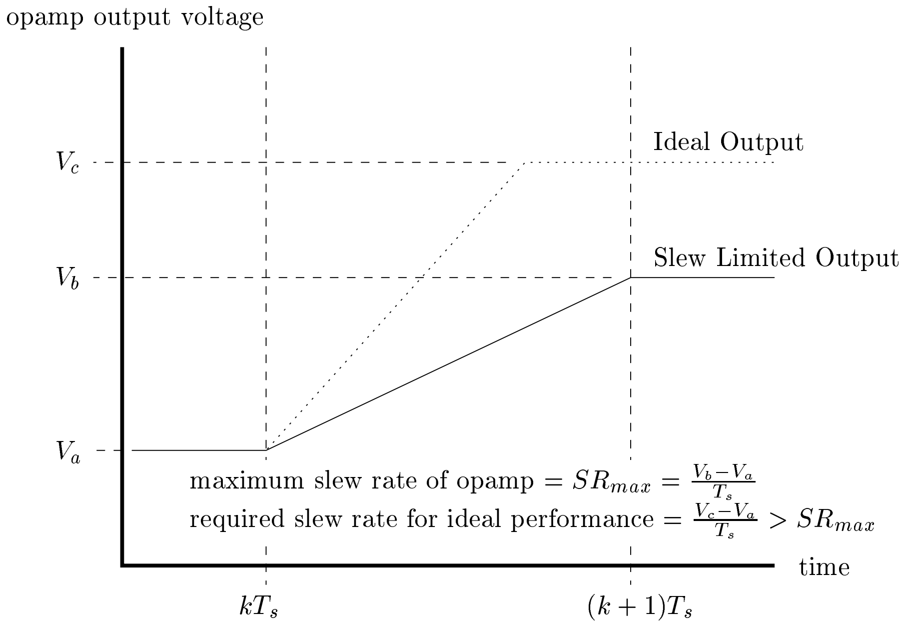

Figure 3.4: Slew Limiting In A Switched Capacitor Integrator

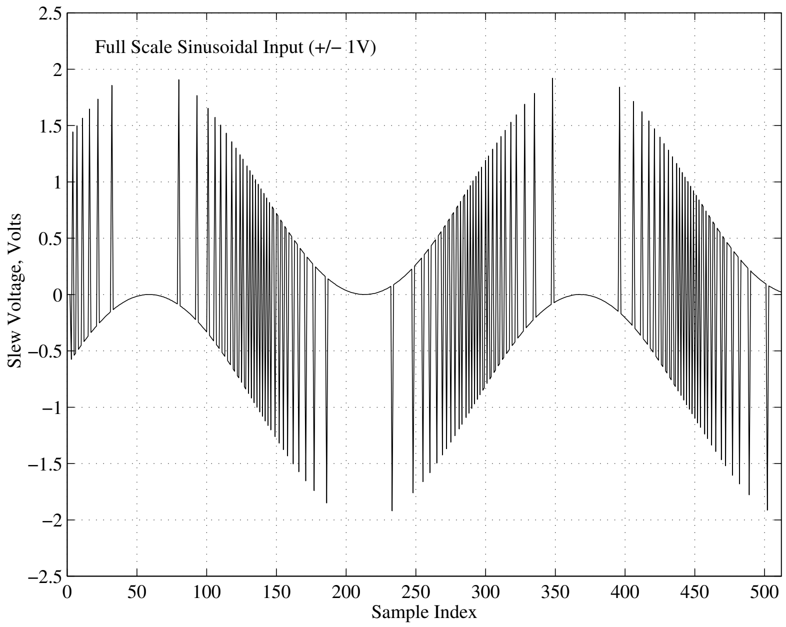

Figure 3.5: Simulated Difference in Voltage Between Adjacent Output Samples of the Integrator Amplifier

slew rate as much as triple this value to ensure no distortion due to slew limiting. In this case, a slew rate of 100 Volts per microsecond is sufficient.

3.3.3 Open Loop Amplifier Gain and Capacitor Mismatches

The error mechanism studied last is the effect of mismatched open loop operational amplifier gain and capacitor values. These parameters are grouped together, since they are simulated in the same manner, and result in the same distortion. Simulating these parameters is accomplished by substituting finite amplifier open loop gain values or finite capacitor values, or both, into the switched capacitor integrator equation given in (3.15). By examining this equation, one can see that the same results are obtained by either equating the capacitor values and assigning a finite amplifier gain, or by allowing the amplifier gain to approach infinity and assigning unequal capacitor values. The end result from either scenario is a non-unity scalar coefficient on each delay term. Although this error mechanism does not result in a nonlinear function, it does result in nonlinear distortion in the output spectrum for each Sigma-Delta loop due to the nonlinearity introduced by the quantizer. The distortion in each loop is removed from the output spectrum of multistage converters to some extent through the cancellation of quantization noise, as seen in (2.29). However, some distortion remains, as seen in Fig. 3.6, where a sinusoid with no harmonics is passed through the three stage, third order noise shaping Sigma-Delta converter simulation, with the three amplifier open loop gains set to unequal, finite values, and with all equal capacitor values.

Capacitor mismatches result from process tolerances when fabricating the Sigma-Delta converter, while open loop amplifier gain differences are obtained from utilizing MOS amplifiers that conserve die area and power consumption by reducing the number of gain stages, and therefore the open loop gain, of the amplifier. The ideal open loop gain is $A \to \infty$, yet typical values used in MOS technology for switched capacitor circuits is on the order of 60 to 80 dB (1000 to 10000

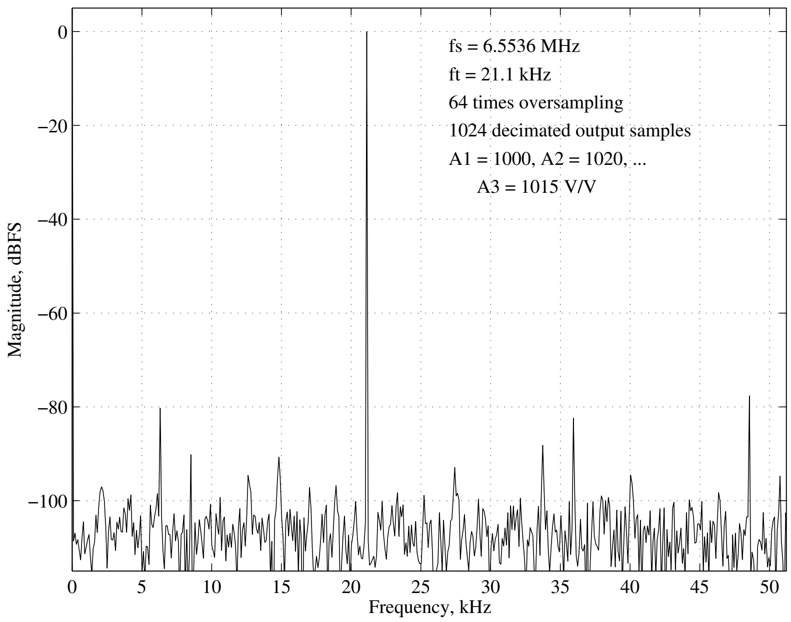

Figure 3.6: Simulated Ouput Spectrum With Mismatched Amplifier Gains and Equal Capacitor Values

V/V) [4, 15, 16]. However, if this gain could be reduced even further, die size could be further reduced, allowing more area for filtering, while reducing power consumption.

At this point, several potential error mechanisms found in the switched capacitor integrators of Sigma-Delta converters have been analyzed, and the three stage, third order noise shaping Sigma-Delta converter simulation was modified to model these mechanisms for study. The next chapter closely examines the mismatched open loop amplifier gains and capacitor ratio errors, and derives a procedure in which these errors are overcome.

CHAPTER 4

Characterization and Compensation of Mismatched Amplifier Open Loop Gains and Capacitance Values

This chapter investigates the output distortion in Sigma-Delta converters due to open loop amplifier gain and capacitor mismatches. The previous chapter briefly introduced this topic, as well as several other distortion mechanisms commonly found in Sigma-Delta architectures. The analysis of the distortion begins by first showing simulation results when these error mechanisms are enabled, followed by a detailed derivation showing how the mismatched parameters produce the nonlinear error terms. This derivation leads to a modified Sigma-Delta architecture that corrects for these mismatched parameters.

4.1 Simulation Results With Nominal Loop Filter Coefficients

As was detailed in chapter 3, mismatched amplifier gains or capacitor values produce distortion in the output spectrum of a Sigma-Delta converter. Fig. 3.6 was created using the three stage converter simulation with the three amplifier gains set to $A_1 = 1000$ V/V, $A_2 = 1020$ V/V, and $A_3 = 1015$ V/V. This distortion is also present when the amplifier gains are equal and very small, as shown in Fig. 4.1. In this figure, the amplifier gains are all set to 500 V/V, and again result in non-unity coefficients in the discrete-time integrator model. This gain is smaller than is typically used in Sigma-Delta converter amplifiers, but reduces the power consumption of the device. Use of low-gain amplifiers could be made more practical by developing procedures to remove the resulting distortion. The process of removing this distortion is referred to as compensation. Fig. 4.2 shows the three stage Sigma-Delta simulation when all of the amplifier gains are equal and suffi-

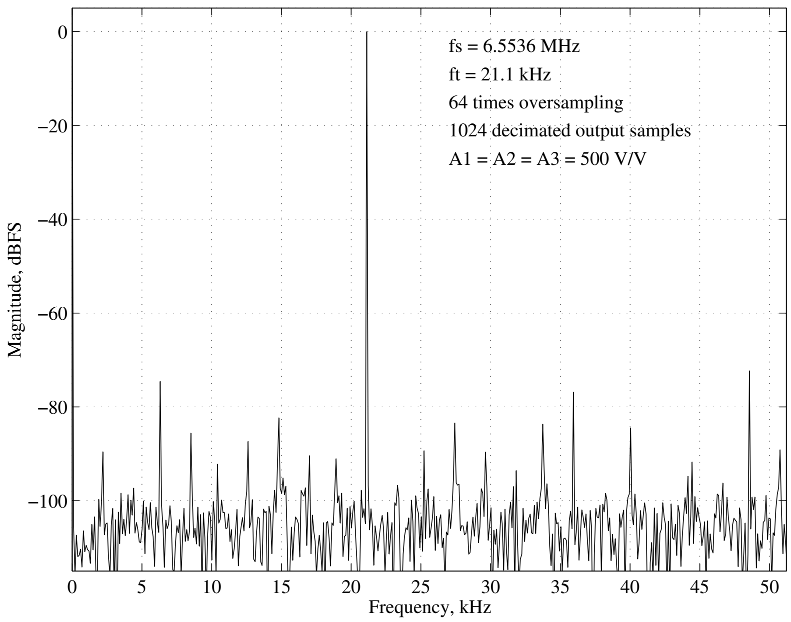

Figure 4.1: Simulated Output Spectrum Using Low Amplifier Gains

ciently large to produce little output distortion; in this case, all gains are set to 10000 V/V. This is the desired output spectrum that the compensation attempts to achieve.

In chapter 2, the significance of the loop filters in the three stage Sigma-Delta converter was mentioned. The ideal converter uses loop filters with "nominal" coefficients as seen in Fig. 2.8, i.e. all filter coefficients are zero or one. The next section of this chapter will examine how modification of these loop filter coefficients affects the distortion in the output spectrum of the converter.

4.2 Derivation of Ideal Loop Filter Coefficients

This section derives necessary changes in the loop filter coefficients to compensate for non-ideal amplifier gains or capacitor mismatches in a Sigma-Delta converter. The derivation begins by defining the Sigma-Delta modulator as the portion of the converter that digitizes the analog signal. This portion includes the three first order loops inside the three stage converter, as shown in Fig. 4.3. The loop integrator transfer functions are denoted $H_1(z)$, $H_2(z)$, and $H_3(z)$, and the corresponding

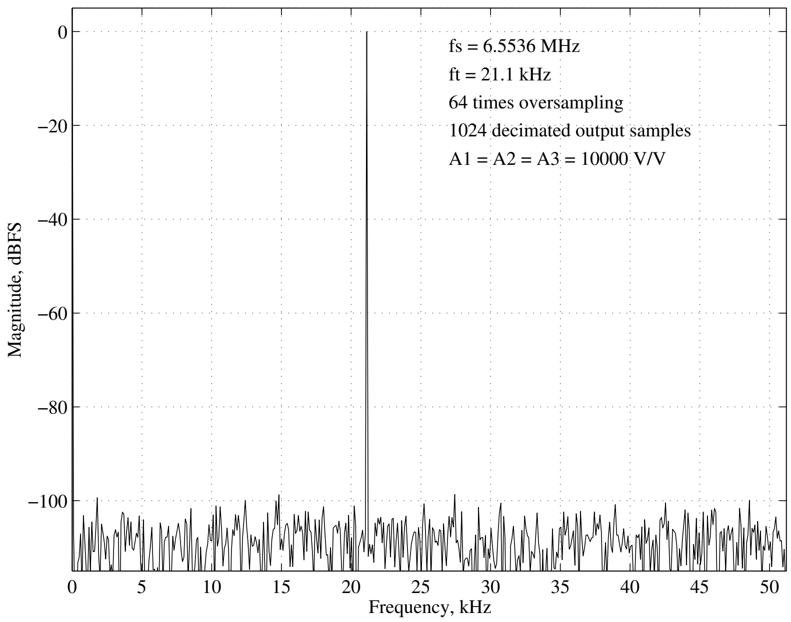

Figure 4.2: Simulated Output Spectrum Using Large Amplifier Gains

loop filters are denoted $G_1(z)$, $G_2(z)$, and $G_3(z)$.

We begin by recalling the discrete-time switched capacitor integrator model, given in (3.15),

$$H_i(z) = \frac{\left[\frac{A_i C_{i1}}{AC_{i2}+C_{i2}+C_{i1}}\right] z^{-1}}{1 - \left[\frac{A_i C_{i2}+C_{i2}}{A_i C_{i2}+C_{i2}+C_{i1}}\right] z^{-1}} = \frac{P_i z^{-1}}{1 - Q_i z^{-1}}, \quad i = 1, 2, 3, \tag{4.1}$$

where, for the i^{th} loop, A_i is the open loop amplifier gain, C_{i1} is the input capacitance, and C_{i2} the integrating capacitance in the switched capacitor integrator, as shown in Fig. 3.2. The output of the first loop is written

$$\begin{aligned} Y_{M1}(z) &= \left[\frac{H_1(z)}{1 + H_1(z)}\right] X(z) + \left[\frac{1}{1 + H_1(z)}\right] N_{ADC1}(z) \\ &= \left[\frac{P_1 z^{-1}}{1 - (Q_1 - P_1) z^{-1}}\right] X(z) + \left[\frac{1 - Q_1 z^{-1}}{1 - (Q_1 - P_1) z^{-1}}\right] N_{ADC1}(z). \end{aligned} \tag{4.2}$$

Similarly, the second and third loop outputs are written

$$Y_{M2}(z) = -\left[\frac{P_2 z^{-1}}{1 - (Q_2 - P_2) z^{-1}}\right] N_{ADC1}(z) + \left[\frac{1 - Q_2 z^{-1}}{1 - (Q_2 - P_2) z^{-1}}\right] N_{ADC2}(z) \tag{4.3}$$

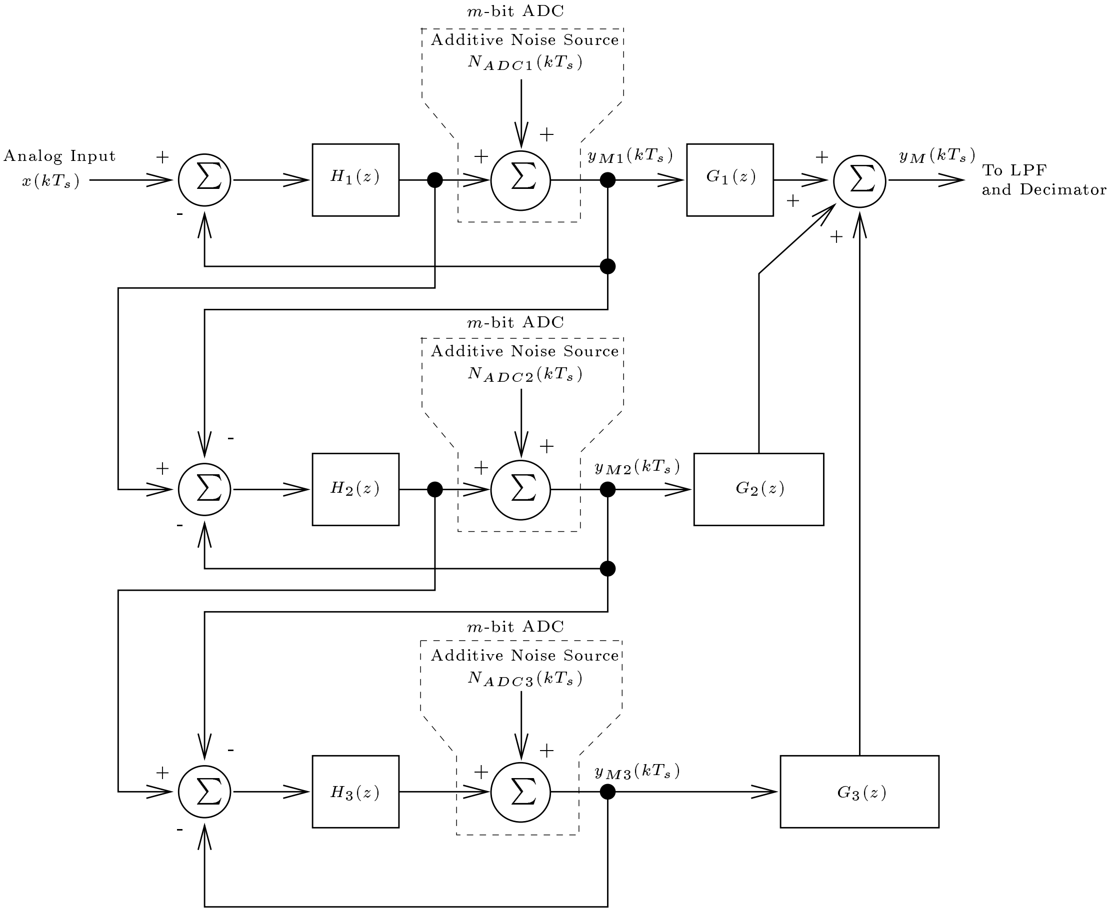

Figure 4.3: Three Stage Sigma-Delta Modulator

and

$$Y_{M3}(z) = -\left[\frac{P_3 z^{-1}}{1-(Q_3-P_3)z^{-1}}\right] N_{ADC2}(z) + \left[\frac{1-Q_3 z^{-1}}{1-(Q_3-P_3)z^{-1}}\right] N_{ADC3}(z). \quad (4.4)$$

The modulator output is then written

$$\begin{aligned} Y_M(z) &= \left[\frac{G_1(z)P_1 z^{-1}}{1-(Q_1-P_1)z^{-1}}\right] X(z)\ldots \\ &+ \left[\frac{G_1(z)(1-Q_1 z^{-1})}{1-(Q_1-P_1)z^{-1}} - \frac{G_2(z)P_2 z^{-1}}{1-(Q_2-P_2)z^{-1}}\right] N_{ADC1}(z)\ldots \\ &+ \left[\frac{G_2(z)(1-Q_2 z^{-1})}{1-(Q_2-P_2)z^{-1}} - \frac{G_3(z)P_3 z^{-1}}{1-(Q_3-P_3)z^{-1}}\right] N_{ADC2}(z)\ldots \\ &+ \left[\frac{G_3(z)(1-Q_3 z^{-1}}{1-(Q_3-P_3)z^{-1}}\right] N_{ADC3}(z). \end{aligned} \quad (4.5)$$

For large amplifier gains and perfectly matched capacitors, $P_i = Q_i = 1$. In this case, use of the nominal loop filter coefficients given in chapter 3 ($G_1(z) = z^{-2}, G_2(z) = z^{-2}\left(\frac{1-z^{-1}}{z^{-1}}\right), G_3(z) = z^{-2}\left(\frac{1-z^{-1}}{z^{-1}}\right)^2$) causes the second and third lines of (4.5) to drop out. In practice, these loop filters are not optimal, as seen in Fig. 4.4, and the performance of the device may be improved by changing $G_1(z)$, $G_2(z)$, and $G_3(z)$.

In order to provide third order noise shaping, the $N_{ADC1}(z)$ and $N_{ADC2}(z)$ terms must cancel, leaving the $N_{ADC3}(z)$ term as the only noise source. It is also desired to have the factor on the input signal term $X(z)$ be of the form $z^{-\ell}$, as discussed in chapter 2. Specifically, for a third order noise shaping converter, $\ell = 3$, and the ideal $G_1(z)$ is calculated to be

$$G_1(z) = z^{-2}\left[\left(\frac{1}{P_1}\right) - \left(\frac{Q_1-P_1}{P_1}\right) z^{-1}\right]. \quad (4.6)$$

Similarly, the remaining ideal loop filters are solved via forward substitution, yielding

$$G_2(z) = z^{-1}\left[\left(\frac{1}{P_1P_2}\right) - \left(\frac{Q_1+Q_2-P_2}{P_1P_2}\right)z^{-1} - \left(\frac{Q_1P_2-Q_1Q_2}{P_1P_2}\right)z^{-2}\right] \qquad (4.7)$$

and

$$\begin{aligned} G_3(z) \quad = \quad & \left(\frac{1}{P_1P_2P_3}\right) - \left(\frac{Q_1+Q_2+Q_3-P_3}{P_1P_2P_3}\right)z^{-1}\ldots \\ & - \left(\frac{Q_1P_3+Q_2P_3-Q_1Q_3-Q_2Q_3-Q_1Q_2}{P_1P_2P_3}\right)z^{-2}\ldots \\ & - \left(\frac{Q_1Q_2Q_3-Q_1Q_2P_3}{P_1P_2P_3}\right)z^{-3}. \end{aligned} \qquad (4.8)$$

Using these loop filters, the resulting ideal modulator output is

$$Y_M(z) = z^{-3}X(z) + \frac{(1-Q_1z^{-1})(1-Q_2z^{-1})(1-Q_3z^{-1})}{P_1P_2P_3}N_{ADC3}(z). \qquad (4.9)$$

The loop filter coefficients given in (4.6), (4.7), and (4.8) will be referred to as the "ideal" loop filter coefficients. It should be noted that as the open loop gains approach infinity and $C_{i1} = C_{i2}$ for $i = 1, 2, 3$, $P_i = Q_i = 1$, and the loop filters and modulator output approach the nominal forms.

Using these ideal loop filter coefficients, the loop filters are once again able to cancel the quantization error from the first and second loop quantizers for non-ideal integrators. Fig. 4.5 shows the Sigma-Delta converter output spectrum when ideal loop filter coefficients are used in the scenario developed for Fig. 3.6. This figure clearly demonstrates the effectiveness of the ideal loop filter coefficients; no distortion terms are present in the output spectrum. The Sigma-Delta converter simulation also indicates that in practice the third loop filter coefficients need not be calculated to remove a majority of the distortion from the output spectrum. Fig. 4.6 shows the simulated output spectrum for the same scenario as Fig. 4.5 when the third loop coefficients remain at their nominal values, i.e. $G_3(z) = z^{-2}(\frac{1-z^{-1}}{z^{-1}})^2$,

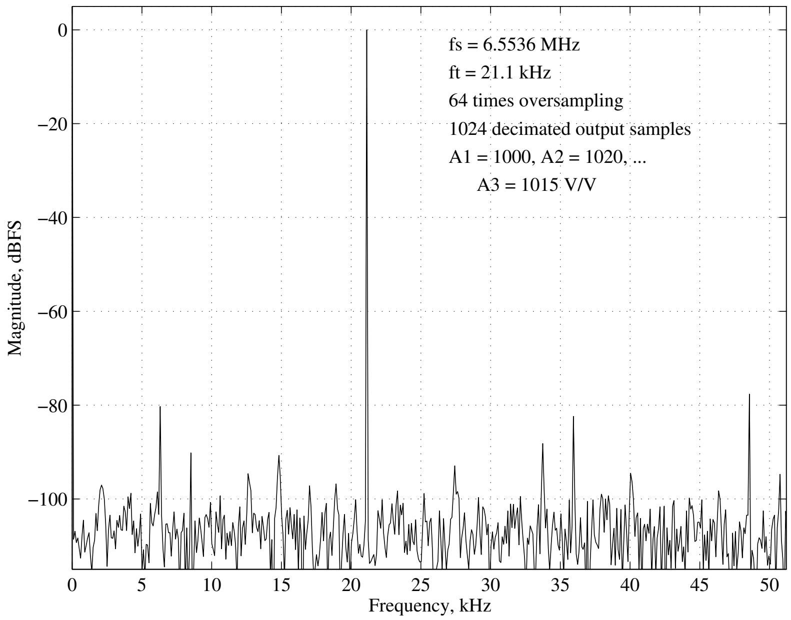

Figure 4.4: Simulated Output Spectrum Using Nominal Loop Filter Coefficients

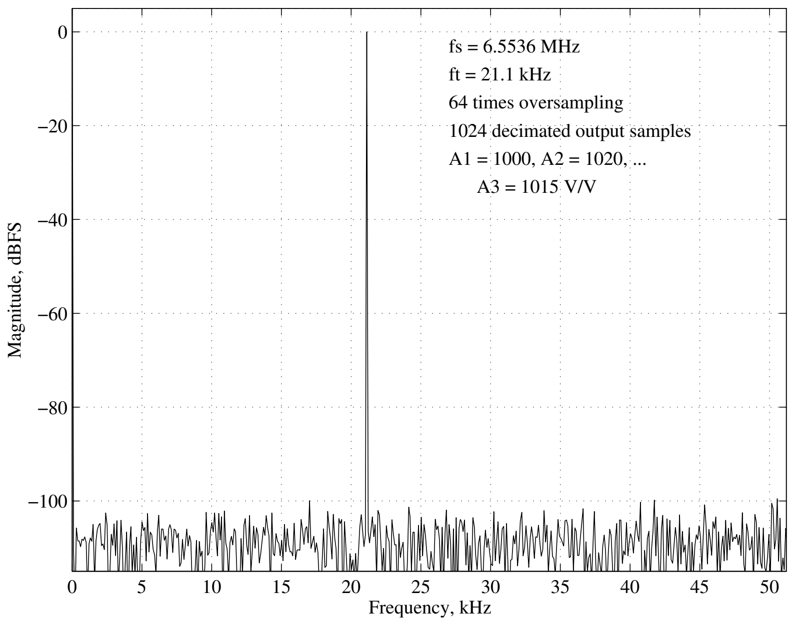

Figure 4.5: Simulated Output Spectrum Using Ideal Loop Filter Coefficients

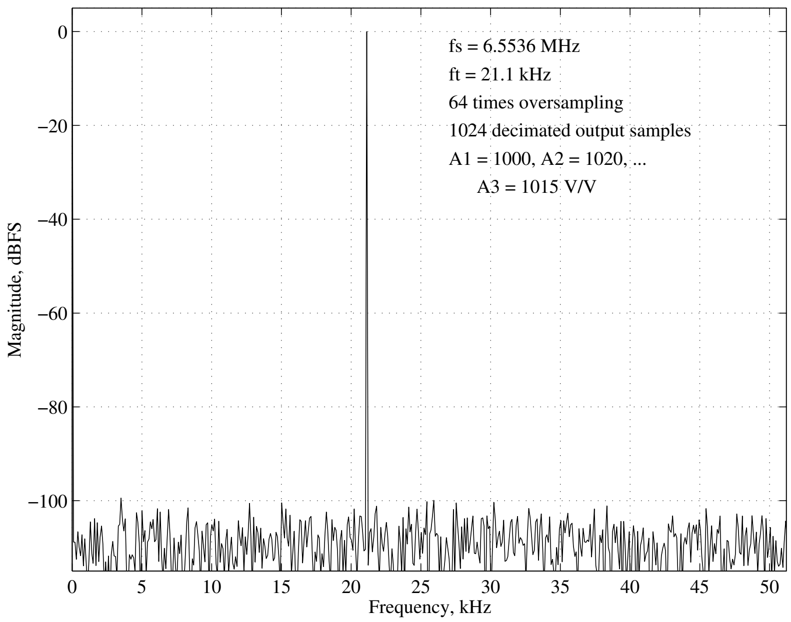

Figure 4.6: Simulated Output Spectrum Using Ideal First and Second Loop Filter Coefficients and Nominal Third Loop Filter Coefficients

and the remaining two loop filters are configured to their ideal values as in (4.6) and (4.7). This figure shows that only the first and second loops' coefficients are mandatory in correcting for the amplifier gain and capacitor mismatch errors.

At this point, it has been shown that distortion caused by amplifier gain and capacitor mismatch errors can be corrected, provided the appropriate loop filter coefficients are used. To exploit this technique, procedures must be developed to estimate effective filter coefficients, since a prior knowledge of integrator component values is not usually possible. The next section explores this topic.

4.3 Estimating Loop Filter Coefficients

Given perfect knowledge of amplifier gains and capacitor values, the distortion introduced by non-ideal values may be compensated as shown in section 4.2. Since these error mechanisms result from fabrication tolerances, small deviations in the capacitances or amplifier gains are not known prior to device construction, and cannot be measured after the device is fully packaged. Therefore, it is neccessary to estimate the filter coefficients using data that is obtained from a fully fabricated device. This section introduces a manner in which effective filter coefficients are calculated using only one set of output samples from the first two loops in the three stage Sigma-Delta structure.

To begin the process of estimating the loop filter coefficients, note that the loop filter coefficients are independent from the input signal and clock sampling frequency. This results in one set of coefficients that will compensate for all input frequencies and sampling rates, provided the amplifier open loop gains and capacitor values do not change as a function of these frequencies. This fact also allows the coefficients to be calculated only one time, and they will function properly, provided the amplifier gains and capacitor values do not change as a function of time. Using these facts and assumptions, only one set of calibration data need be taken to calculate the coefficients, using a least squares fit to minimize the distortion terms introduced into the output spectrum.

To solve for the filter coefficients using a least squares approach, the ideal loop filter equation must be rewritten in terms of the nominal loop filter coefficients:

$$z^2 G_1(z) = (1+\epsilon_1) - (\epsilon_2) z^{-1} \tag{4.10}$$

and

$$zG_2(z) = (1+\epsilon_3) - (1+\epsilon_4) z^{-1} - (\epsilon_5) z^{-2}. \tag{4.11}$$

Here, $\epsilon_1, \epsilon_2, \ldots, \epsilon_5$ represent the unknown adjustments to the nominal coefficients which are required to remove the distortion introduced by the converter. The third loop filter equation need not be rewritten since the third loop filter coefficients do not contribute to output distortion and therefore are not calculated. The Sigma-Delta modulator output equation is now written

$$Y_M(z) \;\; = \;\; z^{-2}\Big[(1+\epsilon_1)Y_{M1}(z) - (\epsilon_2)z^{-1}Y_{M1}(z)\Big]\ldots$$

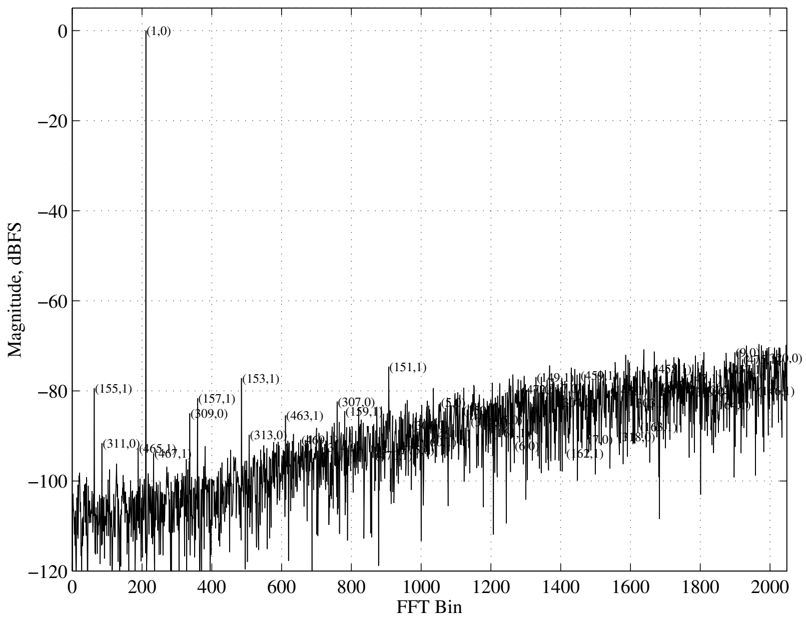

Figure 4.7: Spectrum Used to Identify Distortion Terms For Coefficient Estimation

$$+z^{-1}\Big[(1+\epsilon_3)Y_{M2}(z) - (1+\epsilon_4)z^{-1}Y_{M1}(z) \ldots \\ - (\epsilon_5)z^{-2}Y_{M2}(z)\Big], \quad (4.12)$$

again realizing that the third loop need not be included in the least squares solution. Using data collected from the first two loop outputs, $y_{M1}(kT_s)$ and $y_{M2}(kT_s)$, and creating the output sequence as defined in (4.12) for the case $\epsilon_i = 0$, $i = 1, 2, \ldots, 5$, an output spectrum is observed that contains all the distortion terms that are seen in the final converter output spectrum when the nominal loop filter coefficients are used. An example of this spectrum is shown in Fig. 4.7 for the case $A_1 = 1000$ V/V, $A_2 = 1020$ V/V, $A_3 = 1015$ V/V, and $C_{i1} = C_{i2}$ for $i = 1, 2, 3$. The distortion terms in this figure are identified and labeled as intermodulation distortion between the input signal frequency and one-half the sampling frequency. For example, the label $(155, 1)$ indicates that the distortion term is aliased from the location $f = 155f_t + \frac{f_s}{2}$, where f_t is the pure sinusoidal input signal frequency.

To identify effective filter coefficients, the power in the labeled distortion terms is minimized by adjusting $\vec{\epsilon} = [\epsilon_1 \ \epsilon_2 \ \epsilon_3 \ \epsilon_4 \ \epsilon_5]^T$. A least squares method of curve fit is used to find the best set of coefficients that minimize the total distortion power. The least squares equation is

$$\vec{0} = \vec{x} + \mathbf{B}\vec{\epsilon} + \vec{\mathbf{n}}, \tag{4.13}$$

where $\vec{x}$ is a vector of the magnitudes of the distortion components in the spectrum generated from (4.12) with $\vec{\epsilon} = \vec{0}$, $\vec{n}$ contains the magnitudes of the distortion components after the minimization ($\vec{n}$ is the term that is minimized), and $\mathbf{B}\vec{\epsilon}$ is a vector of the magnitudes of the distortion components in the spectrum generated from (4.12) using the estimated values of $\vec{\epsilon}$:

$$\mathbf{B}\vec{\epsilon} = G(\vec{q}), \tag{4.14}$$

where

$$\vec{q} = \begin{bmatrix} y_{M_1}(n) & y_{M_1}(n-1) & y_{M_2}(n) & y_{M_2}(n-1) & y_{M_2}(n-2) \\ y_{M_1}(n+1) & y_{M_1}(n) & y_{M_2}(n+1) & y_{M_2}(n) & y_{M_2}(n-1) \\ y_{M_1}(n+2) & y_{M_1}(n+1) & y_{M_2}(n+2) & y_{M_2}(n+1) & y_{M_2}(n) \\ y_{M_1}(n+3) & y_{M_1}(n+2) & y_{M_2}(n+3) & y_{M_2}(n+2) & y_{M_2}(n+1) \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \cdot \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \epsilon_4 \\ \epsilon_5 \end{bmatrix}. \tag{4.15}$$

$G(\cdot)$ is a function that finds the magnitudes of the distortion components of the Fast Fourier Transform (FFT) of the arguments. However, a constraint is needed in the least squares solution to avoid the trivial solution $\vec{\epsilon} = [0\ 0\ 0\ 0\ 0]^T$. This is done by forcing the first loop filter to have a filter gain of 0 dB at the frequency $f = 0$ Hz. This constraint allows the magnitude of the input signal $x(kT_s)$ to be effected as little as possible, since the input signal is bandlimited to low frequencies. At $f = 0$ Hz, $z^{-1} = 1$, and the first loop filter equation is rewritten from (4.10) and equated to unity:

$$1 = 1 + \epsilon_1 - \epsilon_2. \tag{4.16}$$

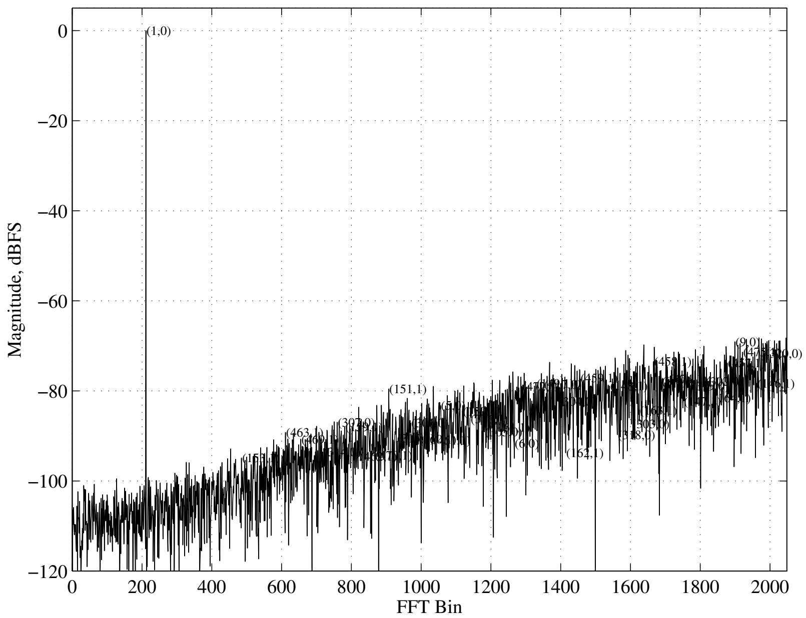

Figure 4.8: Least Squares Equation Spectrum Using Estimated Coefficients

The constraint equation is then written

$$0 = \vec{s}^T \vec{\epsilon}, \tag{4.17}$$

where $\vec{s}^T = [1 \ -1 \ 0 \ 0 \ 0]$. To solve the least squares equation with a constraint, a LaGrange mulitplier is used [19], resulting in the coefficients

$$\vec{\epsilon} = \left[\mathbf{B^T B}\right]^{-1}\left[\mathbf{B^T}(-\vec{\mathbf{x}}) + \vec{\mathbf{s}}\left[\vec{\mathbf{s}}^{\mathbf{T}}\left(\mathbf{B^T B}\right)^{-\mathbf{1}}\vec{\mathbf{s}}\right]^{-\mathbf{1}}\left[-\vec{\mathbf{s}}^{\mathbf{T}}\left(\mathbf{B^T B}\right)^{-\mathbf{1}}\mathbf{B^T}(-\vec{\mathbf{x}})\right]\right]. \tag{4.18}$$

The resulting spectrum from (4.12), using the filter coefficients estimated from the spectrum shown in Fig. 4.7, is shown in Fig. 4.8. As predicted, all distortion terms disappear.

Once the filter coefficients have been estimated, they are substituted into the Sigma-Delta simulation to examine their performance. Fig. 4.9 shows the output spectrum of the converter using the estimated coefficients. This spectrum is a large improvement over the uncompensated spectrum shown previously in Fig. 3.6. These coefficients also prove to be adequate over all input frequencies, as

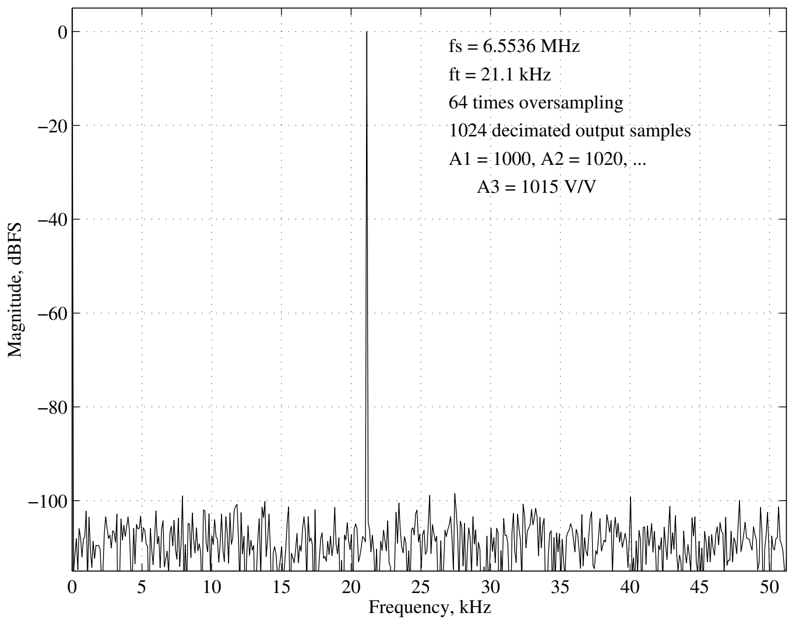

Figure 4.9: Simulated Output Spectrum Using Unequal Amplifier Gains and Equal Capacitor Values Using Estimated Coefficients

shown in Fig. 4.10. In this figure, the Spurious Free Dynamic Range (SFDR) is plotted for frequencies covering the input frequency range from 0 Hz to 51.2 kHz. The SFDR is a measure of the difference in magnitude between the input signal and the largest distortion term in the band of interest. For a band of frequencies, this is a measurement of the dependence of the output distortion on the input signal frequency, and how well the estimated coefficients compensate the ouput spectrum over the input signal frequency range. As seen in this figure, the SFDR of the Sigma-Delta converter is improved by at least 10 dB over the entire input frequency range, and there is little correlation between the input frequency and the effectiveness of the estimated coefficients.

In order to better understand how the estimated coefficients compare to the ideal coefficients, plots of the magnitude responses of the loop filters with both the ideal and estimated coefficients are shown in Figs. 4.11 and 4.12. In these figures, the responses appear to be vastly different. However, the only region of interest is the frequency band $|f| < \frac{f_s}{2D}$, which is $\frac{1}{64}$ of the frequency band shown in these

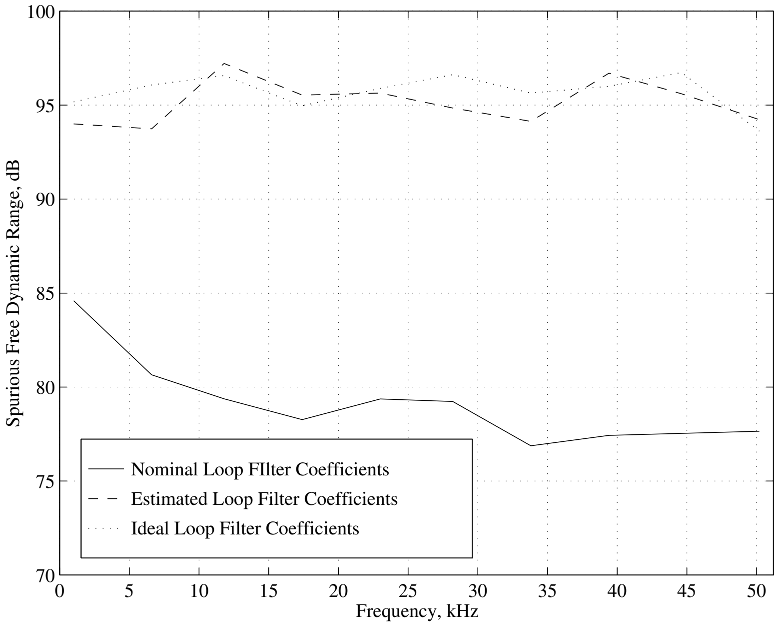

Figure 4.10: Simulated Spurious Free Dynamic Range Using Unequal Amplifier Gains and Equal Capacitor Values

figures. In these regions, both the ideal and estimated frequency responses are nearly identical, and therefore both minimize the error in this band. Because of the unimportance of the frequency range $|f| > \frac{f_s}{2D}$, an infinite number of solutions are available for the coefficient values. This results in estimated coefficients that may be significantly different than the ideal coefficient values, yet still be equally effective.

Given a procedure for compensating disortion produced by amplifier gain or capacitor mismatch errors, the Sigma-Delta architecture must be modified to use this new technique. The last section of this chapter explores several architectures which may be constructed to use estimated filter coefficients.

4.4 New Three Stage Sigma-Delta Architectures

Section 4.3 introduced a manner in which loop filter coefficients are estimated using one set of calibration data. The Sigma-Delta converter architecture must now be modified to make use of these filter coefficients. These coefficients are

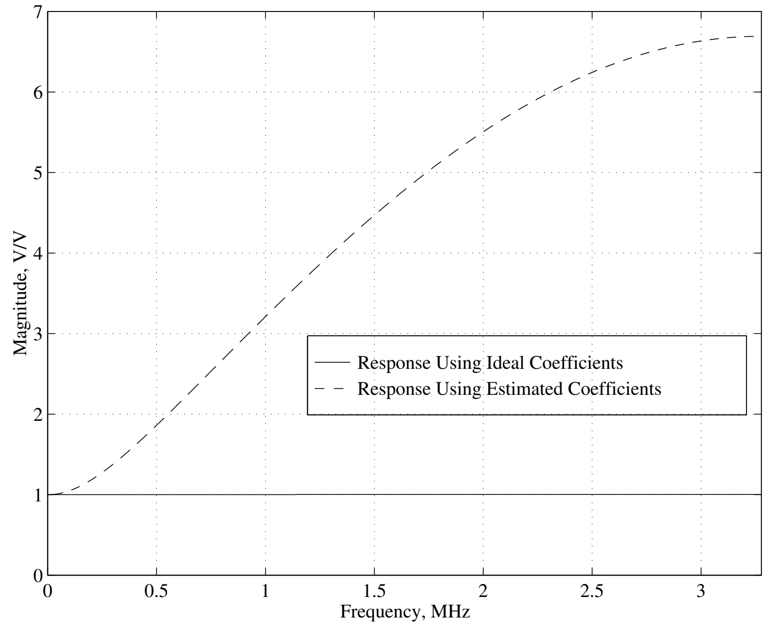

Figure 4.11: Magnitude Response of the First Loop Filter

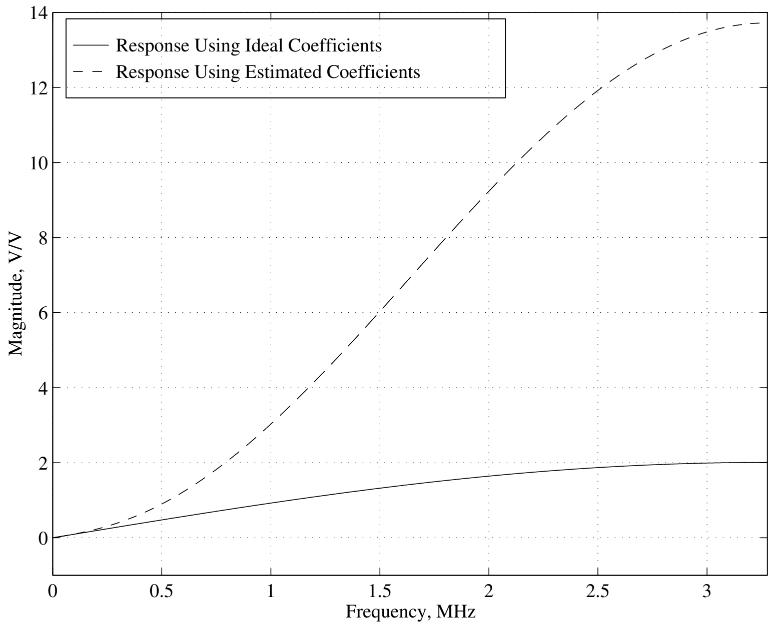

Figure 4.12: Magnitude Response of the Second Loop Filter

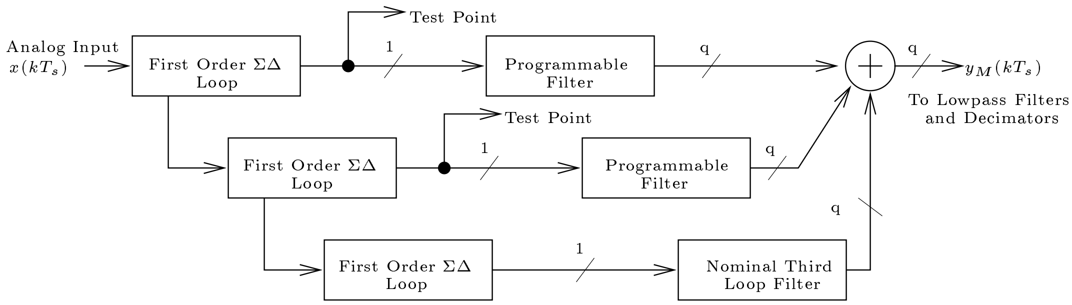

Figure 4.13: Compensating Three Stage Sigma-Delta Architecture: Architecture No. 1

only calculated one time, since they are frequency independent, so the loop filters can use Programmable Read-Only Memories (PROMs) that are programmed one time, or erasable PROMS, depending upon the availability of these memories and the economics involved with fabricating the device. In either case, including programmable filters into the architecture requires a redesign of the converter.

One architecture that can be used to include programmable filters is shown in Fig. 4.13. In this figure, the three stage Sigma-Delta modulator is shown with the loop filters replaced by programmable filters, with the architecture otherwise unchanged. The drawback of this topology is that the number of output bits of the loop filters must be much higher than the number of output bits leaving the filters in the nominal architecture shown in Fig. 2.8. This is due to the fact that the loop filter coefficients require a high precision to be effective. However, the comb filter that typically follows the Sigma-Delta modulator increases the number of bits itself, leading to the second architecture shown in Fig. 4.14.

In Fig. 4.14, the Sigma-Delta modulator had been modularized, using three first order branches. Each branch includes a first order Sigma-Delta loop, a four stage comb filter without a decimator, and the programmable filter. The comb filter is now inside each loop, and therefore three complete comb filters are required, as opposed to one comb filter for the nominal architecture of Fig. 2.8. However, the

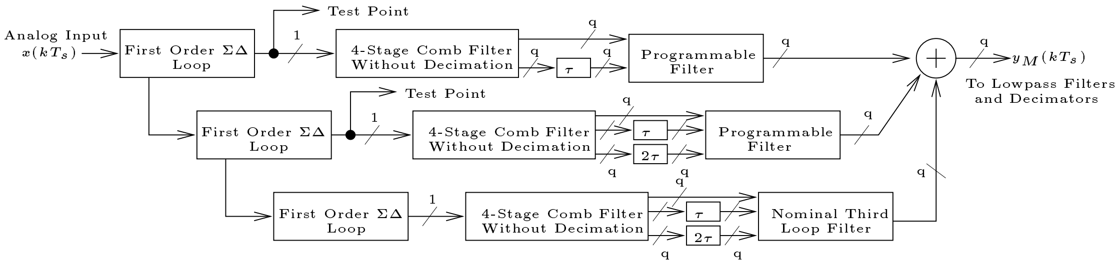

Figure 4.14: Compensating Three Stage Sigma-Delta Architecture: Architecture No. 2

comb filter is not difficult to construct, and the modularity of the entire device is an attractive feature for mass production.

In both modified Sigma-Delta structures shown above, the loop filters operate at the high sampling rate f_s using non-unity coefficients, whereas in the nominal architecture they operated at the high sampling rate using coefficients which where either zero or one. Operating the programmable filters at this high sampling rate is not extremely attractive, and avoiding this scenario was the primary purpose behind using a comb filter as the first lowpass filter in the Sigma-Delta converter. In order to use programmable filters, the sampling rate may have to be reduced. However, the advantage of removing all distortion due to reduced or mismatched amplifier gains or mismatched capacitor values may offset the drawback of operating at a lower sampling frequency; this design decision must be made for each unique application of the Sigma-Delta converter.

CHAPTER 5

Conclusions

This thesis had three primary goals: to detail the operation of the Sigma-Delta analog-to-digital converter, to explore dominant error mechanisms that introduce distortion into the converter output, and to modify a Sigma-Delta architecture to compensate for error mechanisms. This chapter presents the results that were achieved, and expands on these goals by discussing several options for future research.

5.1 Achieved Goals

There were several major accomplishments of this thesis, the first being the justification of using a Sigma-Delta converter in place of a non-oversampled converter, such as a flash converter, in low frequency applications. The Sigma-Delta converter, through the analysis provided in chapter 2, was proven to yield output resolution much higher than the resolution provided by the quantizer alone. The Sigma-Delta is also a much simpler design, requiring fewer components, and therefore using less die space and power than a comparable flash converter. This type of converter lends itself well to low frequency applications, such as ISDN, digital audio, sonar, and DAT systems.

The second major accomplishment was the identification and justification, in chapter 3, of several dominant error mechanisms commonly found in Sigma-Delta converters. The primary focus was on the analog portion of the converter, where nonlinear capacitors, finite amplifier slew rate, and mismatched amplifier open loop gains and capacitor values all proved to be potential sources of output distortion. The nonlinear capacitor distortion was shown to be a difficult mechanism to accurately simulate from real-world data, while the finite slew rate problem was solved by requiring the converter to provide ample slew rate prior to device fabrication. The last two error mechanisms produced the same type of distortion, and were examined in detail in chapter 4. It was discovered that the distortion was the result of intermodulation between the input signal frequency and one-half the clock frequency, and could be compensated by modifying each loop filter of the Sigma-Delta converter.

The last accomplishment of this thesis was the design of proposed three stage, third order noise shaping Sigma-Delta converter architectures that compensate the nonlinear distortion introduced by mismatched amplifier open loop gain and capacitor values. By collecting one set of output data, loop filter coefficients can be calculated that dramatically reduce the distortion from these error sources. The end result is a one-time calibration that can increase the output resolution of the converter by more than 10 dB over the full output frequency range.

5.2 Future Work

Although much was accomplished in the research associated with this thesis, much more work needs to be done in order to fully exploit the Sigma-Delta analog-to-digital converter architecture. There were only a handful of error mechanisms exposed in this thesis, and many more are avaliable for study, including quantizer threshold and misplaced feedback level errors. Errors from the digital portions of the converter, including the digital lowpass filters and decimators, are also areas for possible examination.

This thesis also focused only on the three stage, third order noise shaping Sigma-Delta structure, with compensation based upon this architecture. Future work could extend this compensation technique to other Sigma-Delta topologies, such as the analog current steering modulators described in chapter 1. The coefficients calculated in this thesis could also be iteratively adjusted in real time, avoiding the collection of loop data for the calibration step, and the resulting structure could compensate for temperature variations.

One final area for future work could be to either obtain a commercially available three stage Sigma-Delta converter, or construct one of these converters in MOS technology, to verify the validity of the new compensating structure detailed in this thesis. Verifying the simulation results derived in this thesis in hardware would be a worthwhile extension of this work.

REFERENCES

[1] R. Adams, "Design Aspects of High-Order Delta-Sigma A/D Converters," *Circuits and Systems Tutorials, 1994 IEEE International Symposium on Circuits and Systems*, Oxford, UK.: IEEE Press, pp. 235-260, 1994.

[2] W.R. Bennett, "Spectra of Quantized Signals," *Bell Systems Technical Journal*, vol. 27, pp. 446-472, July 1948.

[3] B.E. Boser and B.A. Wooley, "The Design of Sigma-Delta Modulation Analog-to-Digital Converters," *IEEE Journal of Solid-State Circuits*, vol. 23, no. 6, pp. 1298-1307, December, 1988.

[4] B.P. Brandt and B.A. Wooley, "A 50-MHz Multibit Sigma-Delta Modulator for 12-b 2-MHz A/D Conversion," *IEEE Journal of Solid-State Circuits*, vol. 26, no. 12, pp. 1746-1755, December 1991.

[5] J.C. Candy, "Decimation for Sigma Delta Modulation," *IEEE Transactions on Communications*, vol. COM-34, no. 1, pp. 72-76, January 1986.

[6] J.C. Candy and G.C. Temes, *Oversampling Methods for A/D and D/A Conversion*, New York: IEEE Press, 1992.

[7] V.F. Dias, G. Palmisano, and F. Maloberti, "Harmonic Distortion in SC Sigma-Delta Modulators," *IEEE Transactions on Circuits and System I*, vol. 41, no. 4, pp. 326-329, April, 1994.

[8] V.F. Dias, G. Palmisano, P. O'Leary, and F. Maloberti, "Fundamental Limitations of Switched Capacitor Sigma-Delta Modulators," *IEE Proceedings-G*, vol. 139, no. 1, pp. 27-32, February, 1992.

[9] *DSP56ADC16 16-bit Sigma-Delta Analog-to-Digital Converter*, Arizona: Motorola Literature Distribution, 1992.

[10] H. Inose, Y. Yasuda, and J. Marakami, "A Telemetering System By Code Modulation, Delta-Sigma Modulation," *IRE Trans. on Space, Electronics and Telemetry*, SET-8, pp. 204-209, Sept. 1962.

[11] K.L. Lee and R.G. Meyer, "Low-Distortion Switched-Capacitor Filter Design Techniques," *IEEE Journal of Solid-State Circuits*, vol. SC-20, no. 6, pp. 1103-1113, December, 1985.

[12] W.L. Lee and C.G. Sodini, "A Topology For Higher Order Interpolative Coders," *Proceedings of the 1987 IEEE International Symposium on Circuits and Systems*, vol. 4, pp. 459-462, May 1987.

[13] B. Leung, "Theory of Σ-Δ Analog to Digital Converter," *Circuits and Systems Tutorials, 1994 IEEE International Symposium on Circuits and Systems*, Oxford, UK.: IEEE Press, pp. 195-223, 1994.

[14] *MATLAB Reference Guide*, Massachusetts: The MathWorks, Inc. , 1992.

[15] Y. Matsuya, K. Uchimura, A. Iwata, T. Kobayashi, M. Ishikawa, and T. Yoshitome, "A 16-bit Oversampling A-to-D Conversion Technology Using Triple- Integration Noise Shaping," *IEEE Journal of Solid-State Circuits*, vol. 22, no. 6, pp. 237-244, December, 1987.

[16] S.R. Norsworthy, I.G. Post, and H.S. Fetterman, "A 14-bit 80-kHz Sigma-Delta A/D Converter: Modeling, Design, and Performance Evaluation," *IEEE Journal of Solid-State Circuits*, vol. 24, no. 2, pp. 256-265, April, 1989.

[17] A.V. Oppenheim and R.W. Schafer, *Discrete-Time Signal Processing*, New Jersey: Prentice Hall, 1990.

[18] S. Park, *Principles of Sigma-Delta Modulation for Analog-to-Digital Converters*, Arizona: Motorola Literature Distribution, 1993.

[19] L.L. Scharf, *Statistical Signal Processing: Detection, Estimation, and Time Series Analysis*, Massachusetts: Addison-Wesley, 1991.

APPENDIX A
Three Stage Sigma-Delta ADC MATLAB Simulation Module

```
function [vi1,vl1,vi2,vl2,vi3,vl3,vloop] = ...
  sd_3c_sim(vin,ns,nw,m,A1,A2,A3,a0,a1,a2,b0,b1,b2,SR,T,epsilon,flag)

%*********************************************************************
%*
%* Deron L. Gerow           sd_3c_sim.m                  5/30/96
%*
%*  Three stage Sigma-Delta.
%*
%*  function [vi1,vl1,vi2,vl2,vi3,vl3,vloop] = sd_3c_sim(vin,ns,nw, ...
%*           m,A1,A2,A3,a0,a1,a2,b0,b1,b2,SR,T,epsilon,flag)
%*
%*    This function implements a third order Sigma-Delta converter
%*    using bit-by-bit reaction. This code implements the loop in
%*    the Motorola DSP56ADC16.
%*
%*    INPUT VARIABLES
%*      vin     = input sequence
%*      ns      = number of samples
%*      nw      = number of samples to waste
%*      m       = oversampling ratio
%*      A1      = open loop gain of op-amp inside integrator #1 (V/V)
%*      A2      = open loop gain of op-amp inside integrator #2 (V/V)
%*      A3      = open loop gain of op-amp inside integrator #3 (V/V)
%*      a1      = nominal summing capacitor value (F)
%*      a2      = summing capacitor linear coefficient (F/V)
%*      a3      = summing capacitor quadratic coefficient (F/V^2)
%*      b1      = nominal integrating capacitor value (F)
%*      b2      = integrating capacitor linear coefficient (F/V)
%*      b3      = integrating capacitor quadratic coefficient (F/V^2)
%*      SR      = slew rate of op-amp inside integrators (V/sec)
%*      T       = integration time (time phi2 is high) (seconds)
%*      epsilon = coefficient matrix for matching differentiators
%*      flag    = 0 for differentiators with coefficients in epsilon
%*              = 1 for differentiators with correct coefficients in
%*                  loops 1 and 2
%*              = 2 for differentiators with correct coefficients in
%*                  loops 1, 2, and 3
%*
%*    OUTPUT VARIABLES
```

```
%*      vi1     = output of first integrator
%*      vl1     = output of first loop
%*      vi2     = output of second integrator
%*      vl2     = output of second loop
%*      vi3     = output of third integrator
%*      vl3     = output of third loop
%*      vloop   = three stage converter output
%*
%*********************************************************************

%
%% Create vectors to be used in simulation
%
vi1 = zeros(1,m*ns+3+nw);        %% first integrator output samples
vl1 = ones(1,m*ns+3+nw);         %% first loop output samples
vf1 = ones(1,m*ns+3+nw);         %% first loop feedback samples
vs1 = zeros(1,m*ns+3+nw);        %% first loop output after differ.
vi2 = zeros(1,m*ns+3+nw);        %% second integrator output samples
vl2 = ones(1,m*ns+3+nw);         %% second loop output samples
vf2 = ones(1,m*ns+3+nw);         %% second loop feedback samples
vs2 = zeros(1,m*ns+3+nw);        %% second loop output sfater differ.
vi3 = zeros(1,m*ns+3+nw);        %% third integrator output samples
vl3 = ones(1,m*ns+3+nw);         %% third loop output samples
vf3 = ones(1,m*ns+3+nw);         %% third loop feedback samples
vs3 = zeros(1,m*ns+3+nw);        %% third loop output after differ.
vloop = zeros(1,m*ns+3+nw);      %% final converter output

%
%% Start a new display line and keep track of elapsed execution time
%
percent_done = 0;
tic;
fprintf('  Percent done = %3d%%, ??:?? remaining...',percent_done);

for i = 4:m*ns+3+nw,             %% loop through one sample at a time

  %
  %% Display some text so user doesn't think program is hung up
  %
  percent_done_new = floor(100*i/(m*ns+nw));
  if (percent_done_new ~= percent_done),
    percent_done = percent_done_new;
    time_now = toc;
    time_rem = (100/percent_done)*time_now-time_now;
    time_min = floor(time_rem/60);
```

```
  time_sec = ceil(rem(time_rem,60));
  fprintf('\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b');
  fprintf('%3d%%, %02d:%02d remaining...',percent_done,time_min, ...
          time_sec);
end;

%
%% Find first integrator output
%
in = vin(i-1)-vf1(i-1);         %% current input to the integrator
old = vi1(i-1);                 %% last integrator output
out = (in*(A1*(a0+a1*in+a2*in^2)) + ...
      old*((A1+1)*(b0+b1*old+b2*old^2)))/ ...
      (a0+b0+A1*b0);
vi1(i) = out - (a1+b1+A1*b1)*(out^2) - (a2+b2+A1*b2)*(out^3);
if (abs((vi1(i)-vi1(i-1))) > (SR*T)),     %% Check if slew limited
  sign_vi1 = sign(vi1(i)-vi1(i-1));
  vi1(i) = vi1(i-1) + (SR*T)*sign_vi1;    %% adjust for slewing
end;

%
%% Find the output of the first 1-bit ADC
%
if vi1(i) >= 0,                 %% find feedback value
  vf1(i) = 1;                   %%   and binary loop output
  vl1(i) = 1;
else
  vf1(i) = -1;
  vl1(i) = 0;
end;

%
%% Find second integrator output
%
in = vi1(i-1)-vf1(i-1)-vf2(i-1);          %% current input
old = vi2(i-1);                           %% last integrator output
out = (in*(A2*(a0+a1*in+a2*in^2)) + ...
      old*((A2+1)*(b0+b1*old+b2*old^2)))/...
      (a0+b0+A2*b0);
vi2(i) = out - (a1+b1+A2*b1)*(out^2) - (a2+b2+A2*b2)*(out^3);
if (abs((vi2(i)-vi2(i-1))) > (SR*T)),     %% Check if slew limited
  sign_vi2 = sign(vi2(i)-vi2(i-1));
  vi2(i) = vi2(i-1) + (SR*T)*sign_vi2;    %% adjust for slewing
end;
```

```
  %
  %% Find the output of the second 1-bit ADC
  %
  if vi2(i) >= 0,                %% find feedback value
    vf2(i) = 1;                  %%   and binary loop output
    vl2(i) = 1;
  else
    vf2(i) = -1;
    vl2(i) = 0;
  end;

  %
  %% Find third integrator output
  %
  in = vi2(i-1)-vf2(i-1)-vf3(i-1);          %% current input
  old = vi3(i-1);                           %% last integrator output
  out = (in*(A3*(a0+a1*in+a2*in^2)) + ...
        old*((A3+1)*(b0+b1*old+b2*old^2)))/...
        (a0+b0+A3*b0);
  vi3(i) = out - (a1+b1+A3*b1)*(out^2) - (a2+b2+A3*b2)*(out^3);
  if (abs((vi3(i)-vi3(i-1))) > (SR*T)),     %% Check if slew limited
    sign_vi3 = sign(vi3(i)-vi3(i-1));
    vi3(i) = vi3(i-1) + (SR*T)*sign_vi3;    %% adjust for slewing
  end;

  %
  %% Find the output of the third 1-bit ADC
  %
  if vi3(i) >= 0,                %% find feedback value
    vf3(i) = 1;                  %%   and binary loop output
    vl3(i) = 1;
  else
    vf3(i) = -1;
    vl3(i) = 0;
  end;

end;

%
%% Find converter output after passing loop outputs through
%% differentiators
%
min_length = min([length(vl1) length(vl2) length(vl3)])-5;

if flag == 0,                    %% diff. coefficients are from epsilon
```

```
  %
  %% Pass each loop output through appropriate differentiator
  %
  vs1 = (1+epsilon(1))*vl1(4:min_length+3) - ...
        epsilon(2)*vl1(3:min_length+2);
  vs2 = (1+epsilon(3))*vl2(4:min_length+3) - ...
        (1+epsilon(4))*vl2(3:min_length+2) - ...
        epsilon(5)*vl2(2:min_length+1);
  vs3 = vl3(4:min_length+3) - 2*vl3(3:min_length+2) + ...
        vl3(2:min_length+1);

elseif flag == 1,                %% diff. coefficients are ideal

  %
  %% Define ideal differentiator coefficients for loops 1 and 2
  %
  P1 = (A1*a0)/(A1*b0+b0+a0);
  Q1 = (A1*b0+b0)/(A1*b0+b0+a0);
  P2 = (A2*a0)/(A2*b0+b0+a0);
  Q2 = (A2*b0+b0)/(A2*b0+b0+a0);
  P3 = (A3*a0)/(A3*b0+b0+a0);
  Q3 = (A3*b0+b0)/(A3*b0+b0+a0);
  C1_1 = 1/P1;
  C1_2 = (Q1-P1)/P1;
  C2_1 = 1/(P1*P2);
  C2_2 = (Q1+Q2-P2)/(P1*P2);
  C2_3 = (Q1*P2-Q1*Q2)/(P1*P2);

  %
  %% Pass each loop output through appropriate differentiator
  %
  vs1 = C1_1*vl1(4:min_length+3) - C1_2*vl1(3:min_length+2);
  vs2 = C2_1*vl2(4:min_length+3) - C2_2*vl2(3:min_length+2) - ...
        C2_3*vl2(2:min_length+1);
  vs3 = vl3(4:min_length+3) - 2*vl3(3:min_length+2) + ...
        vl3(2:min_length+1);

elseif flag == 2,                %% diff. coefficients are ideal

  %
  %% Define ideal differentiator coefficients for loops 1,2, and 3
  %
  P1 = (A1*a0)/(A1*b0+b0+a0);
  Q1 = (A1*b0+b0)/(A1*b0+b0+a0);
```

```
  P2 = (A2*a0)/(A2*b0+b0+a0);
  Q2 = (A2*b0+b0)/(A2*b0+b0+a0);
  P3 = (A3*a0)/(A3*b0+b0+a0);
  Q3 = (A3*b0+b0)/(A3*b0+b0+a0);
  C1_1 = 1/P1;
  C1_2 = (Q1-P1)/P1;
  C2_1 = 1/(P1*P2);
  C2_2 = (Q1+Q2-P2)/(P1*P2);
  C2_3 = (Q1*P2-Q1*Q2)/(P1*P2);
  C3_1 = 1/(P1*P2*P3);
  C3_2 = (Q1+Q2+Q3-P3)/(P1*P2*P3);
  C3_3 = (Q1*P3+Q2*P3-Q1*Q3-Q2*Q3-Q1*Q2)/(P1*P2*P3);
  C3_4 = (Q1*Q2*Q3-Q1*Q2*P3)/(P1*P2*P3);

  %
  %% Pass each loop output through appropriate differentiator
  %
  vs1 = C1_1*vl1(4:min_length+3) - C1_2*vl1(3:min_length+2);
  vs2 = C2_1*vl2(4:min_length+3) - C2_2*vl2(3:min_length+2) - ...
        C2_3*vl2(2:min_length+1);
  vs3 = C3_1*vl3(4:min_length+3) - C3_2*vl3(3:min_length+2) - ...
        C3_3*vl3(2:min_length+1) - C3_4*vl3(1:min_length);

end;

%
%% Sum the loop outputs to get the converter output
%
vloop = vs1(2:min_length-2) + vs2(3:min_length-1) + vs3(4:min_length);

%
%% Start a new display line
%
fprintf('Done.\n');
```

APPENDIX B
Comb Filter MATLAB Simulation Module

```
function [vcomb] = sd_comb(vout,m)

%*****************************************************************
%*
%* Deron L. Gerow           sd_comb.m                    5/30/95
%*
%*  1-stage comb filter.
%*
%*  function [vcomb] = sd_comb(vout,m)
%*
%*    This function implements a single stage comb filter, which
%*    averages the input vector over m elements.
%*
%*    INPUT VARIABLES
%*      vout    = vector of samples to be filtered
%*      m       = length of comb filter
%*
%*    OUTPUT VARIABLES
%*      vcomb   = vector of filtered samples
%*
%*****************************************************************

%
%% Make initial condition on output
%
vcomb = zeros(size(vout));             %% Initialize output vector
vcomb(m) = sum(vout(1:m));             %% Initialize first output

%
%% Loop through input vector a sample at a time
%
for i = (m+1):length(vout),

  %
  %% Implement the filter
  %
  vcomb(i) = vcomb(i-1) + (vout(i) - vout(i-m));

end;

%
```

```
%% Format the output vector
%
vcomb = vcomb(m+1:length(vout));
```

APPENDIX C
FIR Filter MATLAB Simulation Module

```
function vfir = sd_use_fir(vin,fir_coeff);

%*****************************************************************
%*
%* Deron L. Gerow           sd_use_fir.m                  5/30/95
%*
%*  N-coefficient FIR filter.
%*
%*  function vfir = sd_use_fir(vin,fir_coeff)
%*
%*    This function implements the FIR filter as described by the
%*    coefficients in the mat-file specified.
%*
%*    INPUT VARIABLES
%*      vin             = input vector of samples to FIR filter
%*      fir_file        = vector containing FIR filter coefficients
%*
%*    OUTPUT VARIABLES
%*      vfir            = output vector of samples from FIR filter
%*
%*****************************************************************

%
%% Assign values to some constants
%
n_taps = length(fir_coeff);                 %% Length of FIR filter
ns = length(vin);                           %% Number of input samples
max_tap = floor(n_taps/2);                  %% Center coefficient

%
%% Now loop through and implement filter
%
for i = (max_tap+1):(ns-max_tap),
  vfir(i-max_tap) = vin(i-max_tap:i+max_tap)*fir_coeff';
end;
```

BIOGRAPHY OF THE AUTHOR

Deron L. Gerow was born in Bangor, Maine, on August 17, 1971. He obtained his high school degree from Brewer High School, Brewer, Maine, in June, 1990.

Deron enrolled in the University of Maine at Orono in the fall of 1990, where he obtained his Bachelor of Science degree in Electrical Engineering with highest distinction in May, 1994. As an undergraduate, he worked as a intern student at the James River Corportation, Old Town Mill, for three years. He continued to work there throughout his graduate career.

In the fall of 1994, Deron entered the graduate program at the University of Maine at Orono. He has served as a research assistant in the Department of Electrical and Computer Engineering, and his current research interests include analog-to-digital conversion, communications and signal processing, and modern filters. He is a member and past vice-president of HKN, and is also a member of $\mathrm{T}\mathrm{B}\Pi$, $\Phi\mathrm{K}\Phi$, and the IEEE. Deron is a candidate for the Master of Science Degree in Electrical Engineering from the University of Maine in August, 1996.

LIBRARY RIGHTS STATEMENT