SYSTEM DESIGN OF A WIDE BANDWIDTH CONTINUOUS-TIME SIGMA-DELTA MODULATOR

A Thesis

by

VIJAYARAMALINGAM PERIASAMY

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

May 2010

Major Subject: Electrical Engineering

SYSTEM DESIGN OF A WIDE BANDWIDTH CONTINUOUS-TIME SIGMA-DELTA MODULATOR

A Thesis

by

VIJAYARAMALINGAM PERIASAMY

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Aydin I. Karsilayan
Committee Members,	Jose Silva-Martinez
	Peng Li
	Duncan M. Walker
Head of Department,	Costas Georghiades

May 2010

Major Subject: Electrical Engineering

ABSTRACT

System Design of a Wide Bandwidth Continuous-Time Sigma-Delta Modulator.

(May 2010)

Vijayaramalingam Periasamy, B.Tech, National Institute of Technology,

Tiruchirappalli, India

Chair of Advisory Committee: Dr. Aydin I. Karsilayan

Sigma-delta analog-to-digital converters are gaining in popularity in recent times because of their ability to trade-off resolutions in the time and voltage domains. In particular, continuous-time modulators are finding more acceptance at higher bandwidths due to the additional advantages they provide, such as better power efficiency and inherent anti-aliasing filtering, compared to their discrete-time counterparts.

This thesis work presents the system level design of a continuous-time low-pass sigma-delta modulator targeting 11 bits of resolution over 100MHz signal bandwidth. The design considerations and tradeoffs involved at the system level are presented. The individual building blocks in the modulators are modeled with non-idealities and specifications for the various blocks are obtained in detail. Simulation results obtained from behavioral models of the system in MATLAB and Cadence environment show that a signal-to-noise-and-distortion-ratio (SNDR) of 69.6dB is achieved.

A loop filter composed of passive LC sections is utilized in place of integrators or resonators used in traditional modulator implementations. Gain in the forward signal path is realized using active circuits based on simple transconductance stages. A novel method to compensate for excess delay in the loop without using an extra summing amplifier is proposed.

ACKNOWLEDGEMENTS

Graduate school at the AMSC group has been a wonderful experience and I would like to thank the people who made it possible.

I would like to thank my advisor, Dr Aydin Karsilayan, for giving the opportunity to pursue my MS thesis under his guidance. I'm greatly indebted to him for his constant support throughout my graduate studies in both academic and personal matters. He set a perfect example for me to look up to with his methodical approach to solving problems and his attention to detail. I would like to thank both Dr. Karsilayan and Dr. Jose Silva-Martinez for having faith in me and giving me the chance to work on the sigma-delta modulator project. Discussions with both professors have been a great learning experience giving me the chance to appreciate the intricacies of circuit design. I would also like to thank Dr. Peng Li and Dr. Duncan Walker for being part of my thesis committee.

I would like to express my gratitude to all the professors who handled courses that I took during graduate studies for sharing their expertise with us. Thanks to Ella Gallagher, Tammy Carda and Jeanie Marshall for helping and advising with all manner of administrative work.

I would like to thank my team members, Aravind Kumar Padyana, Cheng-Ming Chien and Seokmin Hwang, for making work on the project a good experience. Thanks are due to graduate students at the group who had worked on sigma-delta projects earlier and were kind enough to share their knowledge with us – Manisha Gambhir, Fabian Silva-Rivas, Cho-Ying Lu, Vijayakumar Dhanasekaran. Being part of the sigma-delta project with Cho-Ying Lu, Marvin Onabajo, Venkata Gadde and Barry Chen was a good learning exercise and thanks for the same. Thanks to all my friends in the AMSC group for making the stay here an enjoyable one.

Financial support towards my graduate studies from the Analog & Mixed Signal Center by means of the Texas Instruments Fellowship is gratefully acknowledged.

I’m really thankful for the love and support that I have received from my family. My parents and brothers have stood by me in every endeavor of mine and I’m very grateful for the same.

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

FIGURE Page

LIST OF TABLES

1. INTRODUCTION

Technological evolution in the semiconductor industry in the past couple of decades has been following Moore's law, which aims to pack higher digital functionality into a smaller area coupled with smaller power consumption continuously over time. This has made available vast amount of computing power in the digital realm that has resulted in many applications going 'digital', such as storage and communications. This shift is especially apparent in the field of communication where there is a push to perform as much of the processing in the digital domain as possible, with little RF and analog pre-processing. This fact is highlighted in Fig. 1 that compares the traditional super-heterodyne receiver architecture with that of software defined radio.

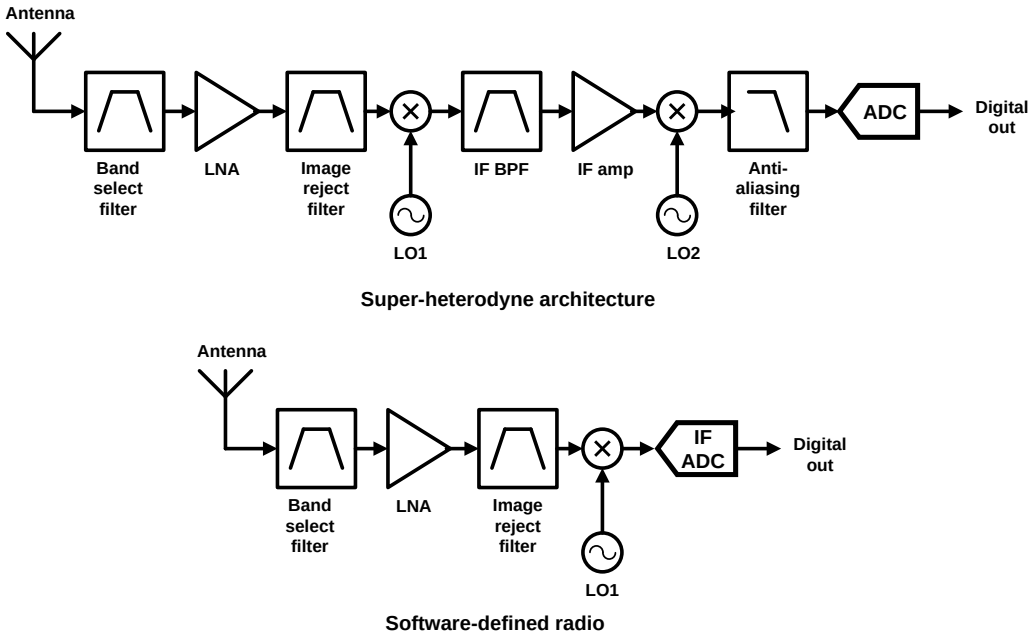


Figure 1 Comparison of radio architectures

As can be seen in the Fig. 1, in the traditional architecture, digitization is preceded by significant analog processing (down-conversion, filtering, and amplification). In the more digital intensive software-defined radio architecture, the signal is digitized much

---

This thesis follows the style of *IEEE Journal of Solid State Circuits.*

earlier enabling the performing of filtering and amplification operations in digital with more flexibility. This however places much higher requirements of bandwidth and dynamic range on the analog-to-digital converter. Also, the quest for higher data rates is leading to the proliferation of standards with larger signal bandwidths. This again reinforces the need for wide bandwidth analog-to-digital converters.

Traditionally, ADCs with sampling rates in the range of 100MSPS and above have been Nyquist-sampling based ones. However, the use of oversampling ADCs in their place can provide power and cost improvements at the system level because of their inherent advantages like simpler anti-aliasing filtering. As mentioned above, there has been tremendous advancement in silicon technologies in the past decade. This results in the availability of very fast devices that gives additional motivation to use oversampling converters that make use of resolution in the time domain while trading off with resolution in the voltage domain.

Oversampling ADCs can be divided into two categories based on the point in the signal chain where sampling takes place: discrete-time (DT), which are built using switched-capacitor filters and continuous-time (CT), which, as their name suggests, are implemented using continuous-time filters. CT sigma-delta ADCs are becoming more popular recently than DT ADCs primarily because of their reduced settling time requirements that results in better power efficiency. Also, they present a constant load to the previous driver stage in place of the switching capacitive load in case of discrete-time sigma-delta modulators. This results in additional power savings in the driver stage. In addition, they have some other advantages like inherent anti-aliasing and reduced sample-and-hold requirements. However, they do have some drawbacks like increased sensitivity to clock jitter, susceptibility to time constant variations and excess loop delay. In spite of these shortcomings, there has been a tremendous interest in continuous-time sigma-delta ADCs as seen by papers published in the recent literature [1-10].

In this work, the design of a wide-bandwidth (100MHz) continuous-time sigma-delta modulator providing 11 bits of resolution is presented. The loop filter is designed using

passive LC sections instead of the conventional implementation using active integrators or resonators. The system level design is done in MATLAB using the ∑Δ toolbox. Excess loop delay is a critical issue in continuous-time modulators, especially when the loop is being operated at a high sampling frequency. In this work, a novel scheme to compensate for excess loop delay is presented.

1.1 Thesis organization

The organization of the thesis is highlighted next.

Section 2 presents an overview of analog-to-digital conversion. The concepts of oversampling and noise shaping are introduced. A literature survey of recent wide bandwidth sigma-delta modulators is presented as well.

Section 3 presents the system design of a continuous-time sigma-delta modulator. The use of tools such as MATLAB and Verilog-A in the design process is highlighted.

Section 4 presents the incorporation of non-idealities into the ideal model built in the previous section. Specifications for the individual building blocks are derived in this section.

Section 5 focuses on the problem of excess loop delay in continuous-time sigma-delta modulators. Various compensation methods found in literature and the method proposed in this work are detailed.

Section 6 summarizes the work by presenting the entire system along with the specifications for the individual blocks.

Section 7 presents the conclusions and also some directions for future work.

2. OVERSAMPLING ANALOG-TO-DIGITAL CONVERTERS

2.1 Introduction

Signals in the real world like sound, temperature and pressure are analog in nature. However, with the progress in semiconductor technology and the availability of vast processing power in the digital domain, more and more signal processing operations are performed on the digital side. Digital signals, by their very nature, are defined only at discrete instances of time and can only take discrete values of amplitude as opposed to the continuous-time, continuous-amplitude nature of analog signals. Analog-to-digital converters serve as the interface between these two domains. A typical signal processing chain is shown in Fig. 2.

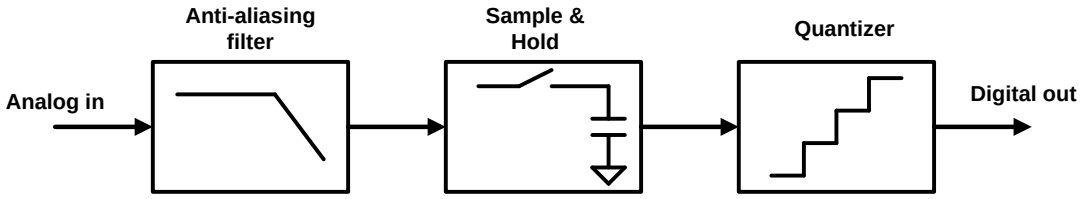


Figure 2 Signal processing chain

Assume that the input analog signal has useful content up to a frequency of f_b. The sample and hold block in Fig. 2 performs the operation of converting the continuous-time signal into discrete-time. To perform this operation without a loss of information, the sampling frequency has to satisfy the following relationship as specified by the Nyquist criterion.

$$f_{sample} \geq 2.f_b \tag{2.1}$$

Note that the above criterion makes sure that the required signal content less than the frequency f_b is not lost due to sampling. However, if the input signal has other frequency content higher than f_b, the sampled signal can be corrupted due to a process known as aliasing. The anti-aliasing filter (AAF) in the signal chain prior to the sample and hold

block prevents this from happening by removing any frequencies higher than f_b before it is sampled.

The output of the sample and hold block, while being discrete in time, still spans a continuous range of values in amplitude. The conversion from continuous to discrete amplitudes is done by the quantizer. While the discretization process in time is lossless, the operation of the quantizer is inherently lossy. The error introduced by this quantization process is dependent on the number of levels in the quantizer.

Assume that the quantizer has a full-scale value of ±Vref and outputs N digital bits. The step size, Δ, of such a quantizer is given by,

$$\Delta = \frac{2V_{ref}}{2^N} \tag{2.2}$$

In such a quantizer, the error, ε, introduced in the quantization operation at every sampling instant will lie in the range (-Δ/2, Δ/2). Even though we know the value of the quantization error at sampling instant, if the input signal is sufficiently 'busy', then the behavior of the quantization error can be considered a white noise process with the probability density function as shown in Fig. 3 [11].

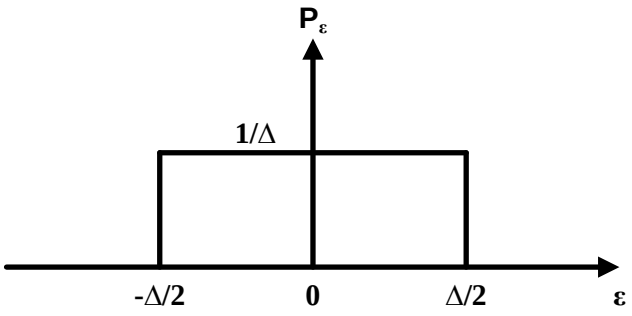


Figure 3 Quantization noise probability density function

The quantization noise introduced can hence be computed as,

$$\sigma_{\varepsilon}^{2} = \int_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} \epsilon^{2}.\frac{1}{\Delta}.d\varepsilon = \frac{\Delta^{2}}{12} \tag{2.3}$$

Assuming that the input to the quantizer is a full-scale sine wave, the power of such a signal is given by $(V_{ref}^{2}/2)$. Knowing the power of the signal and the quantization noise introduced, the signal to quantization noise can be obtained as,

$$SQNR = \frac{\frac{(2^{N-1}\Delta)^{2}}{2}}{\frac{\Delta^{2}}{12}} = \frac{3}{2}.2^{2N} \tag{2.4}$$

Expressed in dB, the above equation turns into the more familiar expression relating SNR with an equivalent number of bits.

$$SQNR\ (dB) = 6.02N + 1.76 \tag{2.5}$$

The above principle is used in a number of ADC architectures to perform analog-to-digital conversion. Such ADCs with sampling rate equal to twice the signal bandwidth are called Nyquist-rate ADCs.

2.2 Oversampling

In the previous section, the expression for quantization noise power was obtained. Since this noise is considered as a white noise process, the power is distributed equally across all frequencies from DC to half the sampling rate, i.e. the entire signal bandwidth. If the signal were to be sampled at a rate higher than the Nyquist rate (2 * signal bandwidth), the same quantization noise power is spread over a higher frequency range. Hence the noise within the frequency of interest gets reduced and the signal-to-quantization noise ratio (SQNR) can be improved a shown in Fig. 4 [11].

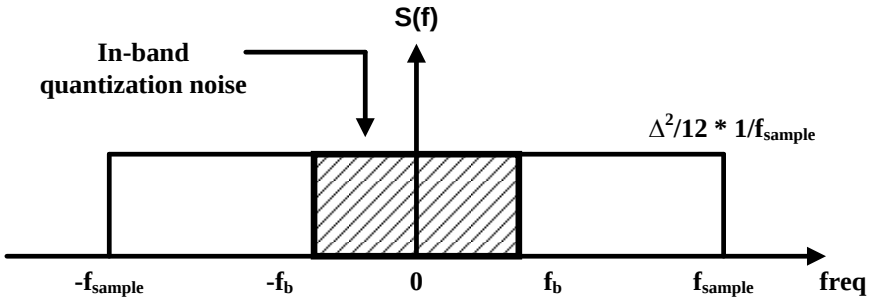


Figure 4 Oversampled quantization noise power spectral density profile

The improvement in SQNR due to oversampling is quantified by,

$$SQNR\ (dB) = 6.02N + 1.76 + 10\ log_{10} OSR \tag{2.6}$$

where,
$$OSR = \frac{f_{sample}}{2f_b} \tag{2.7}$$

For example, oversampling improves SQNR at the rate of 3dB/octave or 0.5bit/octave.

2.3 Noise shaping

The section above shows the advantages that can be obtained through oversampling. As an example, say we want to increase the resolution of a 6 bit converter by 6 bits over a bandwidth of 2MHz. From the relations shown above, we need 12 octaves of OSR, i.e. a sampling frequency of 16.3Gsamples/s. This kind of implementation will be very expensive in terms of power.

A more effective way of improving resolution is by using some method that would shape the quantization noise out of the signal band. This is the principle of sigma-delta data converters. A simple block diagram of a first order sigma-delta converter is shown in Fig. 5.

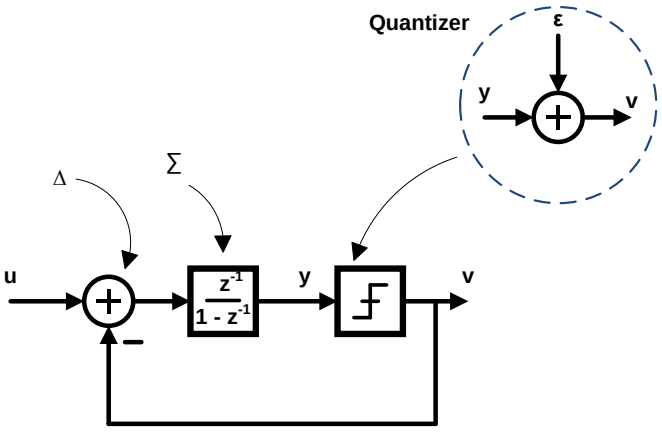


Figure 5 First order sigma-delta modulator

In Fig. 5, the quantization noise is considered as an additive noise. The model in Fig. 5 is that of a linear system with 2 inputs and 1 output. The output can hence be expressed as,

$$v = STF.u + NTF.\varepsilon \tag{2.8}$$

where, STF and NTF refer to the signal transfer function and noise transfer function respectively and are given by,

$$STF = \frac{V(z)}{U(z)} = \frac{H(z)}{1 + H(z)} = z^{-1} \tag{2.9}$$

$$NTF = \frac{V(z)}{E(z)} = \frac{1}{1 + H(z)} = 1 - z^{-1} \tag{2.10}$$

As seen in equations (2.9) and (2.10), the signal and quantization noise have different transfer functions to the output. While the signal appears unchanged at the output with just a delay, the noise is shaped as shown in Fig. 6.

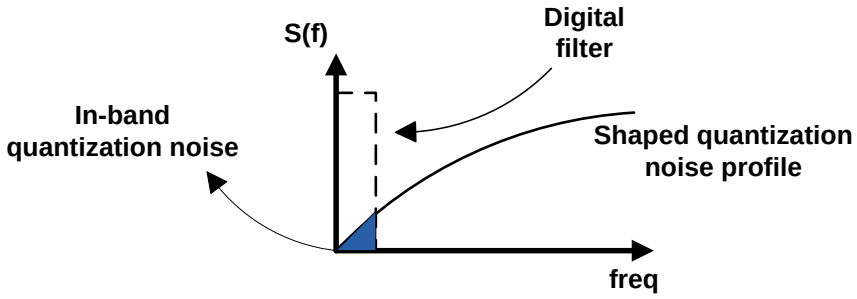


Figure 6 First order modulator quantization noise profile

Upon integrating the noise over the frequency of interest, the total noise is obtained as,

$$\sigma_{\epsilon}^2 = \frac{\Delta^2}{12} \cdot \frac{1}{OSR^3} \cdot \frac{\pi^2}{3} \tag{2.11}$$

Expressing as SQNR, we can see that we get an improvement in SQNR of 9dB/octave (1.5bit/octave) of oversampling.

In the above example, the quantization noise is shaped out of band by a first order transfer function. We can obtain further improvements in resolution by using higher orders of noise shaping. For example, by using an L^{th} order loop filter, we can obtain an NTF given by,

$$NTF = (1 - z^{-1})^L \tag{2.12}$$

The SQNR in such a case is given by [12],

$$SQNR_{max} = 6.02N + 1.76 + (20L + 10)log_{10}OSR - 10log_{10}\frac{\pi^{2L}}{2L + 1} \tag{2.13}$$

Hence, the SQNR will improve at the rate of (L+0.5) bits/octave of oversampling in the case of an L^{th} order sigma-delta modulator.

2.4 Types of sigma-delta modulators

In the previous section, the modulators considered were implemented using loop filters that were composed of discrete-time integrators. Hence it is a sampled data system and the sampling for such a system happens at the input of the modulator. These modulators are known as discrete-time (DT) sigma-delta modulators. They are implemented using switched-capacitor techniques. They are very popular for use at low frequencies for applications such as audio signal processing. At such low signal frequencies (20kHz), they make use of a high oversampling ratio (of the order of 128 or higher) and an inherently linear 1-bit quantizer/DAC to obtain resolutions of the order of 20 bits or higher.

Because of their implementation using switched capacitor circuits, the amplifiers used in the integrator structures need to have a unity gain frequency (UGF) of the order of 10 times the sampling frequency to obtain sufficient settling to the desired accuracy. As a result, when the same architecture is extended to higher frequencies, the UGF requirements become very high resulting in very high power consumption. Hence an approach that avoids the switching action and the settling requirements is necessary.

Continuous-time sigma-delta modulators process data using continuous-time filters and the sampling operation is performed after the filter within the loop. Since the input now processes continuous data instead of one that is switching, the UGF requirements are relaxed. This makes them very suitable for use in high bandwidth applications. Also, since the sampling action occurs within the loop, any errors due to sampling are introduced at the same point as quantization noise. Hence these errors undergo the same transfer function and are hence noise shaped outside the frequency band of interest. Sigma-delta converters, by their very oversampling nature, relax the anti-aliasing requirements significantly in comparison to Nyquist-rate converters. Continuous-time sigma-delta converters have the additional advantage that they provide inherent anti-aliasing as well.

For all the advantages compared to their discrete-time counterparts, continuous-time modulators have their own distinct drawbacks as well. The time-constants defining the pole locations are defined by capacitor ratios in DT modulators. These can be accurate to the order of 1% in modern technologies. However, the same pole locations are defined by RC products in CT modulators and these can have variations of the order of ±20%. Hence tuning for setting the proper pole locations is often necessary. CT modulators are also susceptible to error introduced due to clock jitter while DT modulators are not.

2.5 Literature survey

Table 1 Brief summary of wide bandwidth sigma-delta modulators

Year	Technology	Sampling Freq (Hz)	Bandwidth (Hz)	SNR (dB)	SNDR (dB)	Power (mW)
1998[1]	InGaAs HEMT	5G	100M	43	39	400
2001[2]	InGaAs HBT	18G	500M	-	42 *	1500
2003[3]	InP HBT	8G	250M	-	40	1800
2006[4]	SiGe HBT	20G	312.5M	30.5	-	490
2009[5]	SiGe HBT	35G	100M	58.9	53.1	350
2006[6]	130nm CMOS	640M	20M	76	74	20
2007[7]	180nm CMOS	400M	25M	53	52	18
2008[8]	90nm CMOS	420M	20M	72	70	27.9
2009[9]	65nm CMOS	250M	20M	62	60	10.5
2009[10]	130nm CMOS	900M	20M	81.2	78.1	87

* Two-tone SNR

As shown in Table 1, there has been tremendous amount of work going on in the field of continuous-time sigma-delta modulators trying to achieve high bandwidths at reasonable power consumption.

Because of the widespread use of CMOS technology for digital applications, it has emerged as the technology of choice for cost-conscious designs as it enables easy integration with further downstream digital processing. This is shown by publications such as [6-10] showing modulator implementations in the most advanced process technology nodes. Note that, recently there have been novel ideas [9-10] that make use of the high speed capability of the latest CMOS technologies to harness resolution in the time domain instead of being limited to just that in amplitude. Although these CMOS implementations achieve high resolutions with low power consumption, the bandwidths achieved are limited to the 20-25MHz range.

Works reported in [1-3] have achieved medium resolutions over very wide bandwidths (>100MHz). They make use of exotic III-V process technologies for their ability to provide very high speed operation. However, as of now, these technologies are limited in their use to niche applications and hence tend to be very expensive. Also, the power consumptions shown are quite high (>1W) [2-3].

3. SYSTEM LEVEL DESIGN OF MODULATOR

In the last decade, sigma-delta modulators were primarily discrete-time in nature and hence were implemented using switched-capacitor techniques. Only in the recent past, with the push towards higher bandwidths together with higher power efficiency, have continuous-time modulators become more prominent. Because of the wide usage of switched-capacitor modulators, design methodologies for DT modulators have been very well studied and many tools geared for the same have also been developed, e.g. the ΔΣ MATLAB toolbox by Richard Schreier [13], DAISY [14] etc. Hence, during the design of loop filters for continuous-time modulators as well, it is more common to first obtain a DT loop filter and then use the impulse-invariant transformation to convert the same into a CT loop filter. This approach is followed in this work as well. However, it should be noted that this is not the only way to design continuous-time modulators and the design of the same can be done entirely in the CT domain as shown by [15], [16].

3.1 Design considerations

The target in this work is to realize a continuous-time sigma-delta modulator with the specifications shown in Table 2 in 180nm BiCMOS technology.

Table 2 Specifications

Performance parameter	Targeted specification
Bandwidth	100 MHz
Resolution	11 bits
Power consumption	< 500mW

For this purpose, the first step in the design process is to obtain an optimum noise transfer function (NTF) with the major system level parameters as the variables. At the system level, the variables to be considered are over-sampling ratio (OSR), number of bits in the internal quantizer (N), order of the loop (L) and the aggressiveness of the

noise shaping (determined by the out-of-band gain of the NTF) [12]. This optimization is performed by using the MATLAB toolbox by Richard Schreier to obtain the best NTF.

The roles played by the variables mentioned above will be highlighted in the next few paragraphs. The signal-to-quantization ratio (SQNR) of an L^{th} order modulator incorporating an N-bit internal quantizer operating with an oversampling ratio of OSR is shown as:

$$SQNR_{max} = 6.02N + 1.76 + (20L + 10)log_{10}OSR - 10log_{10}\frac{\pi^{2L}}{2L+1} \quad (3.1)$$

From equation (3.1), we can see that increase in the order of the modulator leads to a higher SQNR. However, higher order modulators are more difficult to stabilize and usually the order of the system is limited to 5.

Similarly, increasing the OSR of the design provides a better SQNR. However, the maximum speed of operation is usually limited by technology to some finite value. Also, operation at higher speeds leads to higher power dissipation.

Along the same lines, we would like to have more number of bits in the internal quantizer to obtain small quantizaton noise. However the power and area of the implementation of quantizers rise exponentially with increase in the number of bits. Also, higher number of bits in the quantizer places more stringent requirements on the DAC elements used in the feedback path.

The aggressiveness of the noise shaping is determined by the maximum gain of the NTF outside the signal band (NTF_{max}). A higher value of NTF_{max} tends to push more of the quantization noise from the signal band to higher frequencies. The downside is that higher values of NTF_{max} tend to degrade the performance of the modulator when jitter in the system clock is taken into account. This will be discussed in more detail in section 4.

Based on extensive simulations in MATLAB using the ΣΔ toolbox, the values in Table 3 were arrived at to obtain the best SQNR performance.

Table 3 Noise transfer function parameters

OSR	10
Order (L)	5
No of levels in quantizer	9
NTF_{max}	3.36

The NTF used in the design is shown in Fig. 7.

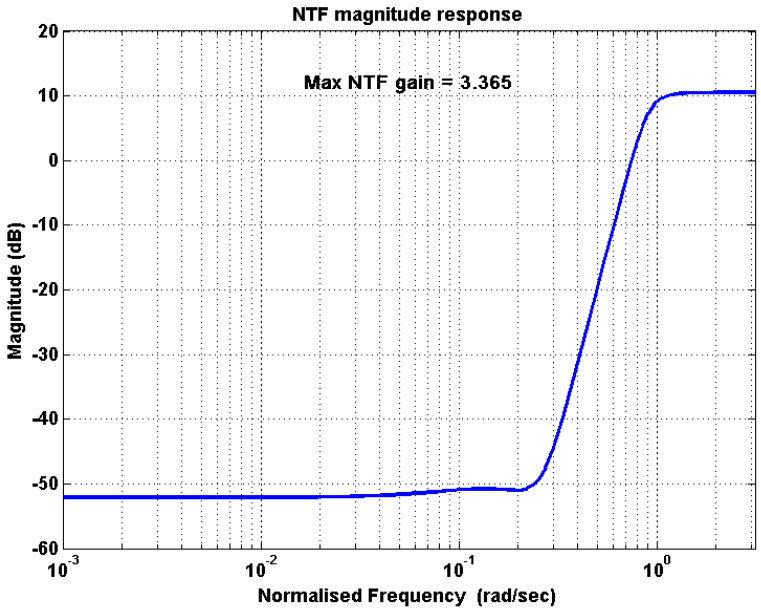


Figure 7 Noise transfer function

A plot of the SQNR of the discrete-time system as a function of the input amplitude is shown in Fig. 8. From the figure, we can see that the modulator can provide a maximum SQNR of 76.1dB at input amplitude of -3dBFS.

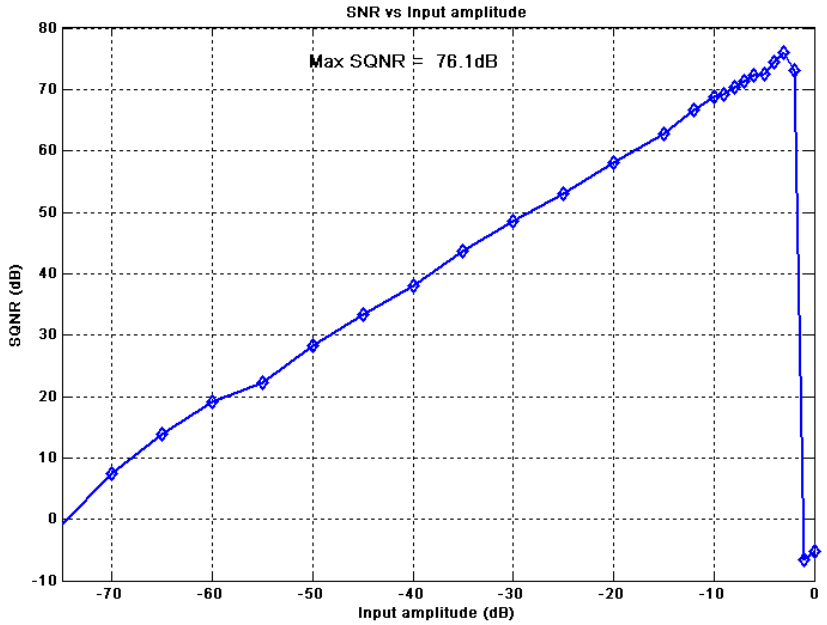


Figure 8 SNR vs input amplitude of discrete-time system

Once the NTF is known, the loop transfer function (LTF) of the modulator can be easily obtained using the following relation [17]:

$$LTF = \frac{1}{NTF} - 1 \tag{3.2}$$

The corresponding loop transfer function is obtained as:

$$LTF = \frac{2.253z^4 - 5.539z^3 + 5.727z^2 - 2.8z + 0.5389}{z^5 - 4.387z^4 + 7.797z^3 - 7.016z^2 + 3.196z - 0.5899} \tag{3.3}$$

Excess loop delay (discussed in section 5) is a real concern in CT sigma-delta modulators and it is good practice to incorporate compensation for the loop delay during the system design phase itself. Hence, in this work, an excess delay of 1 cycle is taken into account and the discrete time loop transfer function is modified by factoring out the delay (z^{-1}) term from the loop transfer function in equation (3.3) [6]. The modified loop transfer function to be implemented in the modulator is obtained as,

$$LTF = \frac{2.253z^5 - 5.539z^4 + 5.727z^3 - 2.8z^2 + 0.5389z}{z^5 - 4.387z^4 + 7.797z^3 - 7.016z^2 + 3.196z - 0.5899} \quad (3.4)$$

3.2 Impulse-Invariant transformation

The input to a discrete-time modulator is sampled in nature and data transfer through the modulator happens at fixed instants in time controlled by a clock. In contrast, a continuous-time modulator, as its name suggests, processes data that is continuous in nature. However, the internal quantizer in the loop is clocked and there is a sampling action happening within the continuous-time modulator loop. This provides a means to obtain equivalence between a DT and CT modulators as described below [18].

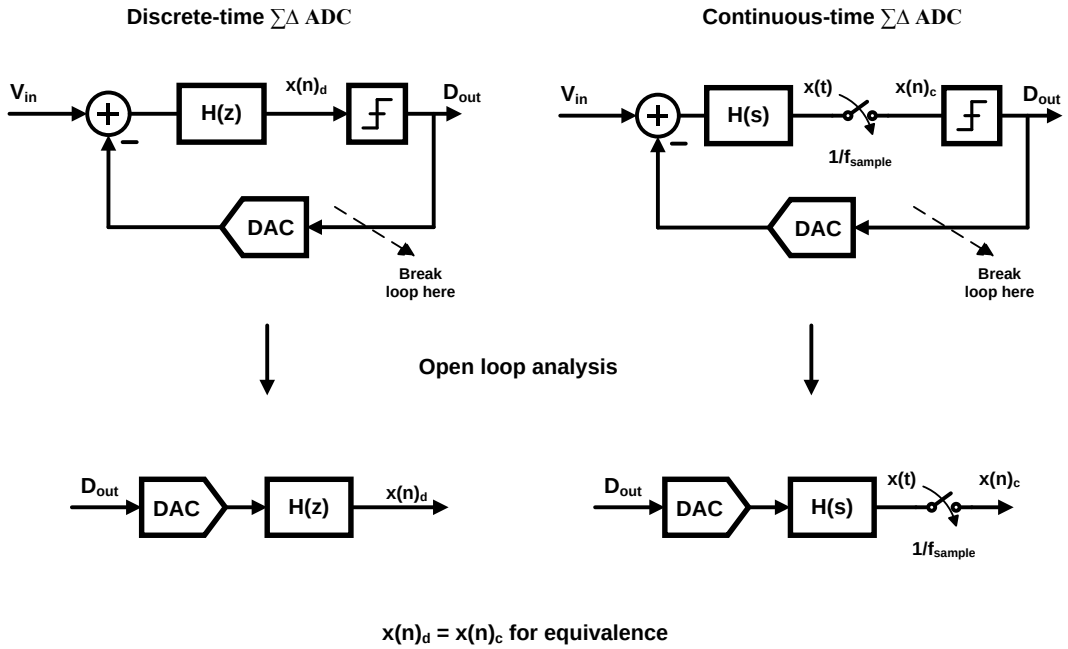


Figure 9 Equivalence of discrete and continuous-time modulators

Fig. 9 shows discrete-time and continuous-time modulator loops and the corresponding open loop structures obtained by breaking the loops at the DAC inputs. For the two loops to be identical, for the same input, the outputs of the two loops at the sampling instants should be the same.

$$x(n)_d = x(t)|_{t=nT_s} \tag{3.5}$$

The condition in equation (3.5) is satisfied if the impulse responses of the two loops in question are the same. Expressed in the frequency domain, this takes the form given by,

$$Z^{-1}[H(z)] = L^{-1}[H_d(s) * H(s)] \tag{3.6}$$

where, Z^{-1} and L^{-1} stand for inverse Z and Laplace transforms respectively and $H_d(s)$ represents the frequency response of the DAC in the feedback loop.

In the time domain, equation (3.6) can be expressed as,

$$h(n) = \{h_d(t) * h(t)\}|_{t=nT_s} \tag{3.7}$$

where, h(n), $h_d(t)$ and h(t) represent the impulse responses of the discrete-time loop filter, the DAC and the continuous-time loop filter respectively.

Since the above principle is based on the impulse response being the same in the discrete-time and continuous-time structures, it is known as the impulse-invariant transformation.

As can be seen from equations (3.6) and (3.7), the continuous-time loop filter to be used in the modulator is dependent on the impulse response of the DAC pulse used in the feedback loop. The most commonly used DAC pulses in continuous-time modulators are Non-Return to Zero (NRZ), Return to Zero (RZ) and Half-Return to Zero (HRZ). The impulse responses of the mentioned DAC pulses are shown in Fig. 10 [12].

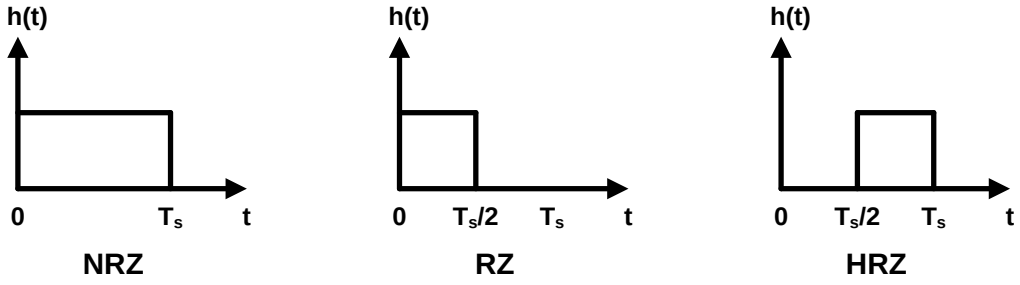


Figure 10 NRZ, RZ and HRZ DAC pulses

Because of their inherent nature, the RZ and HRZ DAC pulses have more number of transitions on average in each clock cycle. As a result they are more susceptible to clock jitter than the NRZ DAC pulse shape. Hence, in this work, the NRZ DAC pulse is used for its better jitter robustness.

It should be noted that these are not the only DAC pulse shapes possible. Pulse shapes such as exponential [19] and sinusoidal [20] have been reported in the literature.

For the case of the NRZ DAC pulse, the impulse invariant transformation is available in MATLAB using the built-in function 'd2c' [6]. Upon usage of this function, the corresponding continuous-time loop filter for the discrete-time transfer function in equation (3.4) is obtained as,

$$H(s) = \frac{2.253s^5 + 8.716e9s^4 + 1.719e19s^3 + 2.212e28s^2 + 1.811e37s + 7.57e45}{s^5 + 1.056e9s^4 + 9.471e17s^3 + 4.115e26s^2 + 1.418e35s + 1.88e43} \tag{3.8}$$

The procedure to obtain the loop transfer function, starting from the design of the NTF, is summarized below.

1. Choose a high-pass filter transfer function for the NTF. The common filter types chosen are Butterworth, Chebyshev and Inverse-Chebyshev. When done in MATLAB, for a filter of order n, the resulting transfer function takes the form,

$$H(z) = \frac{b_1 + b_2 z^{-1} + \cdots + b_{n+1} z^{-n}}{1 + a_2 z^{-1} + \cdots + a_{n+1} z^{-n}} \tag{3.9}$$

   For a noise transfer function to be realizable, $H(\infty) = 1$. To satisfy this condition, divide the transfer function $H(z)$ in equation (3.9) by b_1. A good starting point for the stop-band of the NTF is the desired signal bandwidth. The zeros of the transfer function can be spread across the signal bandwidth to obtain optimum SQNR [21].
2. Choose values for the oversampling ratio (OSR) and quantizer resolution.

3. Simulate the modulator obtained using 'simulateDSM' function of the ∑Δ toolbox. Use 'ds_hann' and 'calculateSNR' functions to obtain the SQNR.
4. In case the SQNR obtained is not sufficient, go back to steps 1 & 2 and pick a higher stop-band frequency, OSR or resolution in the quantizer, and vice-versa if the SQNR is too high.
5. Once the desired SQNR is obtained, use equation (3.2) to obtain the discrete-time loop transfer function from the NTF.
6. Modify the loop transfer function to account for excess loop delay by factoring out z^{-k} term, where k is the amount of delay in the loop.
7. Use the impulse-invariant transformation to convert the discrete-time loop transfer function to continuous-time, taking into account the shape of the DAC feedback pulse.

3.3 Loop filter implementation

Once the loop transfer function is obtained as shown in the previous section, the next step is the implementation of the transfer function using individual building blocks. In continuous-time sigma-delta modulators, the loop filters are generally implemented as a cascade of integrators or resonators comprised of biquads. The individual integrator or biquad sections can be connected together in two different configurations called feedback and feed-forward implementations. These two configurations are shown in Fig. 11 and 12 for the case of a 3rd order modulator with all the poles at DC (implemented with integrators).

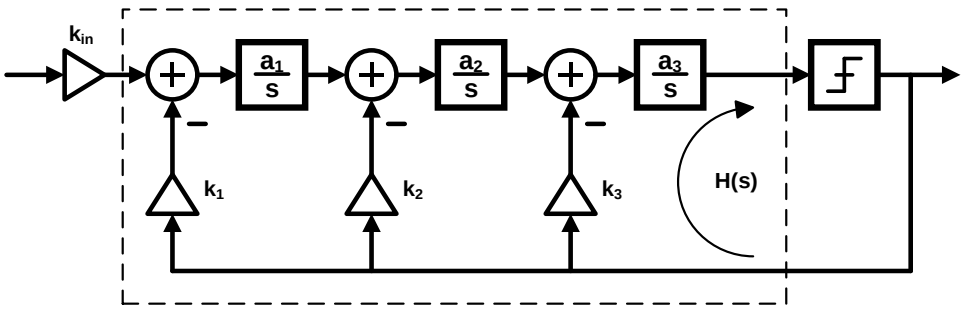


Figure 11 Feedback configuration

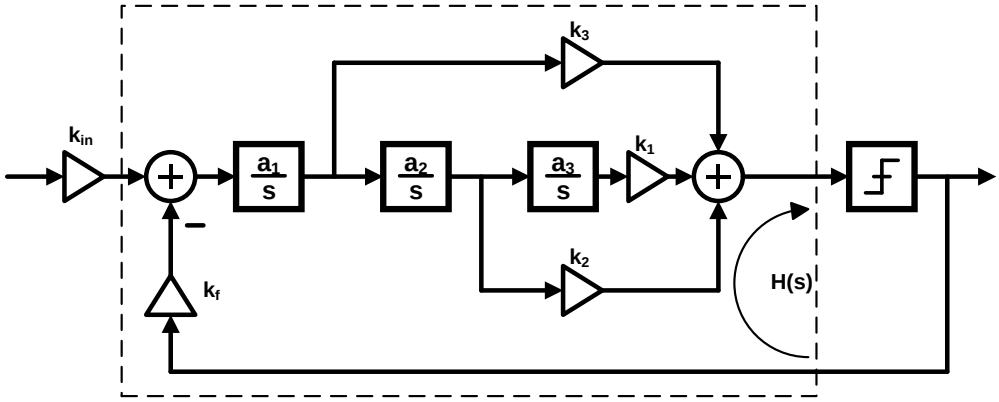


Figure 12 Feed-forward configuration

As shown in Fig. 11, in a feedback implementation of the modulator, the zeros are implemented by the feedback to each node of the filter. The loop filter, with transfer function H(s), is shown in the figure within the dotted box. The advantage of the feedback topology is that it provides anti-aliasing of an order equal to the order of the loop filter. However, it needs multiple DACs in the feedback path and also tends to be power hungry.

The feed-forward configuration, as shown in Fig. 12, uses feed-forward paths from the individual integrators to the quantizer to implement the zeros. The advantage with this configuration is that only one DAC is needed in the feedback path. Also, this architecture is more power-efficient when compared to the feedback based implementation. However, the feed-forward implementation only provides first order anti-aliasing filtering. Also, the signal transfer function shows peaking outside the signal band, which can lead to saturation in the modulator in the presence of blockers. The summing amplifier in front of the quantizer is present in the high frequency path and is very critical for stability.

In both the feedback and feed-forward representations shown in Fig. 11 and 12 the poles are realized by integrators. Non-DC poles can be realized using resonators formed by biquad sections. In all these cases, as the bandwidth of interest in a modulator implementation increases, the power consumption in the amplifier within the integrators

or biquads rises to accommodate the higher frequency signals. Hence, in this work, the use of passive filters comprised of LC sections is explored.

Also, the feedback architecture is used as compared to a feed-forward one for the following two reasons:

1. The proposed LC filter has convenient feeding points to obtain low-pass and band-pass transfer functions.
2. It avoids the summing amplifier that would be required in a feed-forward architecture case. With a sampling frequency of 2GHz, the summing amplifier can be a potential speed bottle-neck.

A single-ended representation of the proposed loop filter with the feedback paths included is shown in Fig. 13.

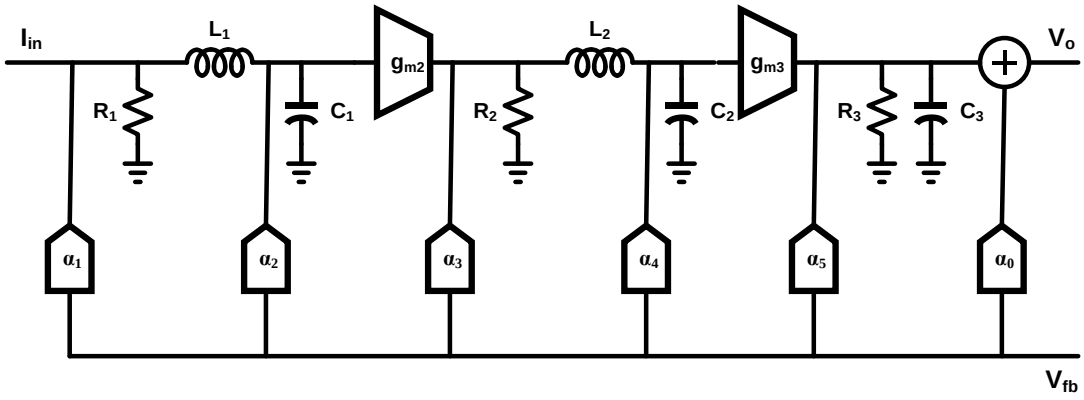


Figure 13 Proposed loop filter

The loop transfer function of the filter in Fig. 13 is given by,

$$\frac{V_o(s)}{V_{fb}(s)} = \alpha_0 + \frac{\alpha_5/C_3}{s+\omega_3} + \frac{g_{m3}/C_3}{s+\omega_3}.\frac{1}{s^2 + s\left(\frac{\omega_2}{Q_2}\right) + \omega_2^2}.\left[s\left(\frac{\alpha_4}{C_2}\right) + R_2\omega_2^2(\alpha_3+\alpha_4)\right] + \frac{g_{m3}/C_3}{s+\omega_3}.\frac{g_{m2}R_2\omega_2^2}{s^2 + s\left(\frac{\omega_2}{Q_2}\right) + \omega_2^2}.\frac{1}{s^2 + s\left(\frac{\omega_1}{Q_1}\right) + \omega_1^2}.\left[s\left(\frac{\alpha_2}{C_1}\right) + R_1\omega_1^2(\alpha_1+\alpha_2)\right] \quad (3.10)$$

where,

$$\omega_1 = \frac{1}{\sqrt{L_1C_1}} \quad (3.11)$$

$$\omega_2 = \frac{1}{\sqrt{L_2C_2}} \quad (3.12)$$

$$\omega_3 = \frac{1}{R_3C_3} \quad (3.13)$$

$$Q_1 = \frac{\omega_1 L_1}{R_1} \quad (3.14)$$

$$Q_2 = \frac{\omega_2 L_2}{R_2} \quad (3.15)$$

By comparing the transfer function of the loop filter shown above and the desired filter response obtained earlier in equation (3.8) the coefficients can be obtained.

The parameters of the modulator obtained after performing the comparison mentioned above are shown in Table 4.

Table 4 Modulator component values

α_0	0.45	L_1	100nH	L_2	100nH
α_1	20μ	C_1	35.6pF	C_2	31.4pF
α_2	2.58m	R_1	33Ω	R_2	51Ω
α_3	6.20m	g_{m2}	100mS	g_{m3}	40mS
α_4	3.84m	R_3	2kΩ	C_3	2.4pF
α_5	3.16m				

A representation of the complete modulator with the loop filter, quantizer and the feedback DACs is as shown in Fig. 14.

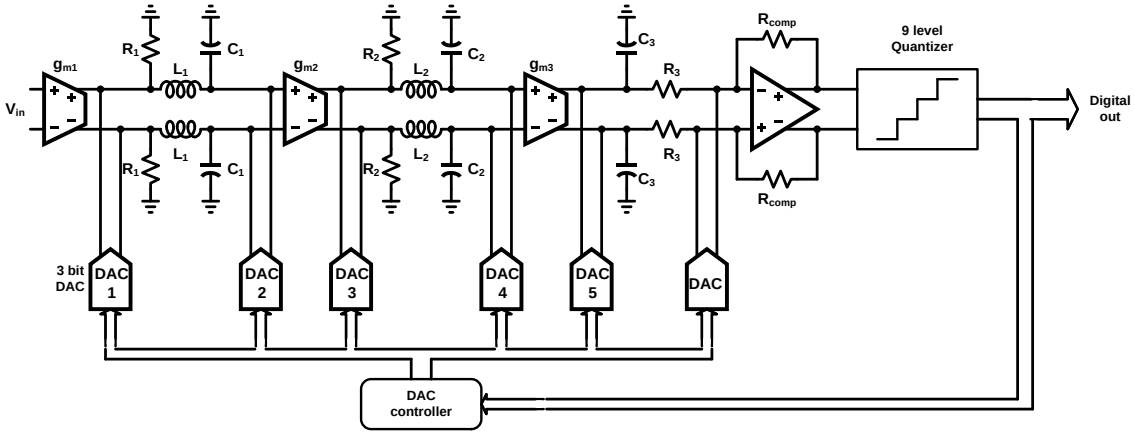


Figure 14 Modulator implementation

The model of the system built in the simulink environment of MATLAB is shown in Fig 15.

3.4 Simulation-based synthesis

In the previous section, it was shown how to obtain the parameters of the sigma-delta modulator by using the desired and theoretical loop transfer functions and comparing the corresponding coefficients. This process was made simple by the existence of a built-in function in MATLAB for the special case of the NRZ pulse shape chosen. For other pulse shapes, the process is not as straight-forward. In such cases, the simulation-based synthesis method proposed in [12] can be used. A brief explanation of the method is given below.

The method is based on the fact that the loop in a sigma-delta modulator is a linear time-invariant system. Hence the output response of the loop is composed of a natural response and a forced response [12].

The natural response is determined primarily by the denominator (poles) of the transfer function. These poles and their quality factors can be obtained from the corresponding DT poles using the impulse-invariant transformation using [12],

$$z_p = e^{s_p} \tag{3.16}$$

where, z_p and s_p denote the DT and CT poles, respectively.

Once the pole locations are obtained, these can be mapped into component values for the 2^{nd} order sections using,

$$\omega = \frac{1}{\sqrt{LC}} \tag{3.17}$$

$$Q = \frac{\omega L}{R} \tag{3.18}$$

Similarly, for the first order section, we get,

$$\omega = \frac{1}{RC} \tag{3.19}$$

Once the individual 2nd order and 1st order sections of the loop are determined, what remains to be found are the coefficients that form the different feedback paths to give the entire loop transfer function. This is what determines the forced response of the system.

The loop filter is composed of six independent paths from the DAC input to the input of the quantizer (5 feedback paths and the fast path for stability). Since this is a linear system, the impulse response of the entire system will be the linear sum of the impulse responses of the individual paths mentioned above. The impulse response of each path can be found by exciting the loop with an impulse by enabling only the required path with a coefficient of 1 and setting the other coefficients to 0. A simulation setup in MATLAB to obtain the coefficients is shown in Fig. 16 for the 1st feedback path.

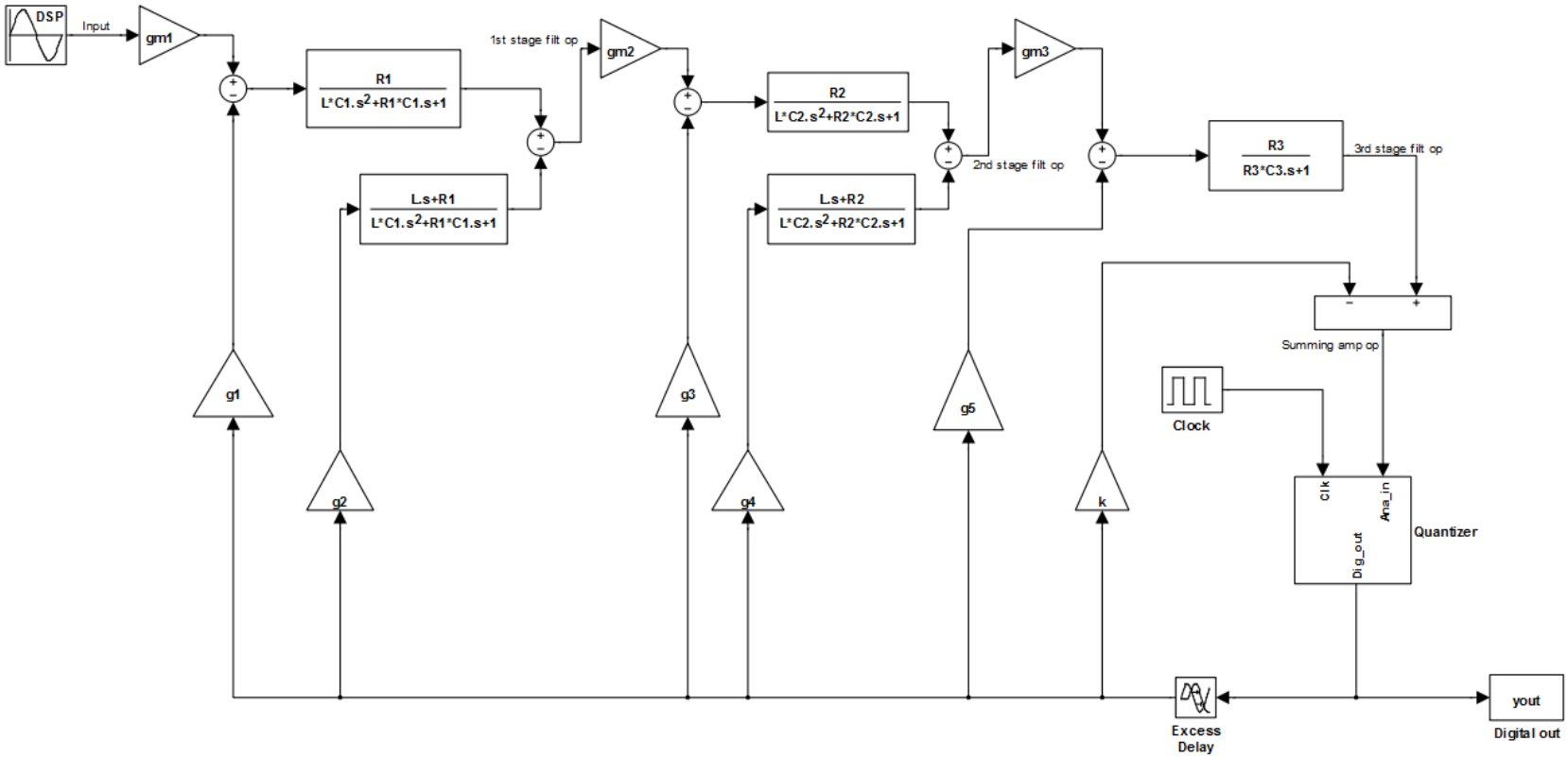


Figure 15 MATLAB model of system

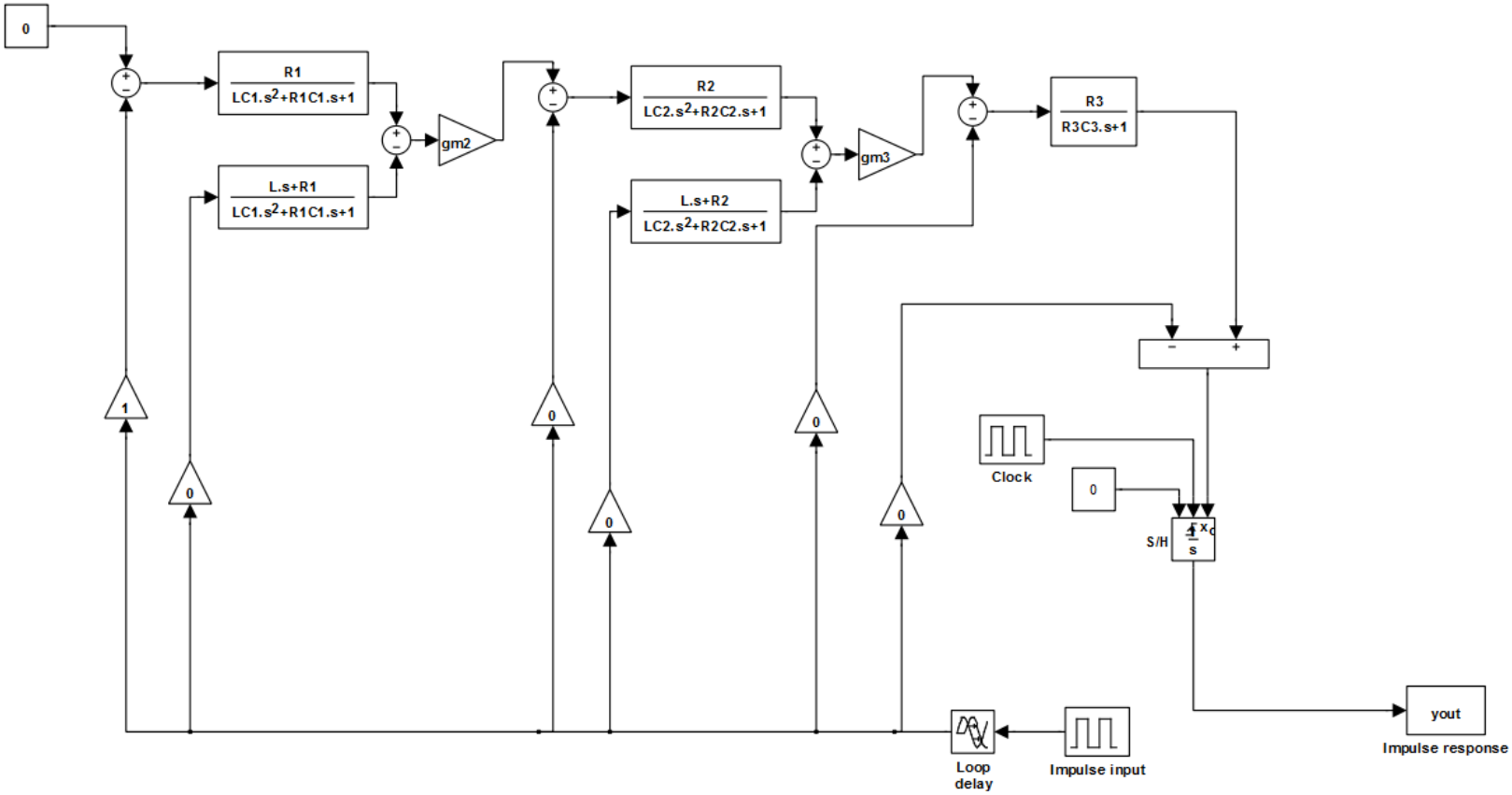


Figure 16 MATLAB coefficient extraction setup

Let the response obtained by this step for the 1st path is denoted by $h_1(m), m$ denoting the sampling instants. Similar responses can be obtained for each of the individual paths and let they be denoted by $h_n(m)$. Let the required coefficients in the feedback path be denoted by $[\alpha_0, \alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5]$. As mentioned above, the net impulse response of the system, $h(m)$, is the linear sum of the above obtained responses and is given by,

$$h(m) = \sum_{n=0}^{5} \alpha_n h_n(m) \tag{3.20}$$

By the impulse invariant transformation, this response should be equal to the response of the desired equivalent discrete-time system. The discrete-time impulse response can be easily obtained from the z-domain loop transfer function $H(z)$ in equation (3.4). Let this impulse response be denoted by $h_d(m)$. By equating the two sets of values, we obtain a system of linear equations given by,

$$\begin{aligned} \alpha_0 h_0(m) + \alpha_1 h_1(m) + \alpha_2 h_2(m) + \alpha_3 h_3(m) + \alpha_4 h_4(m) \\ + \alpha_5 h_5(m) = h_d(m), \qquad m \to 1-6 \end{aligned} \tag{3.21}$$

By rearranging the equations (3.21) and using matrix algebra, we can solve for the coefficients $[\alpha_0, \alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5]$.

$$\begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \\ \alpha_5 \end{bmatrix} = \begin{bmatrix} h_0(1) & h_1(1) & h_2(1) & h_3(1) & h_4(1) & h_5(1) \\ h_0(2) & h_1(2) & h_2(2) & h_3(2) & h_4(2) & h_5(2) \\ h_0(3) & h_1(3) & h_2(3) & h_3(3) & h_4(3) & h_5(3) \\ h_0(4) & h_1(4) & h_2(4) & h_3(4) & h_4(4) & h_5(4) \\ h_0(5) & h_1(5) & h_2(5) & h_3(5) & h_4(5) & h_5(5) \\ h_0(6) & h_1(6) & h_2(6) & h_3(6) & h_4(6) & h_5(6) \end{bmatrix}^{-1} \begin{bmatrix} h_d(1) \\ h_d(2) \\ h_d(3) \\ h_d(4) \\ h_d(5) \\ h_d(6) \end{bmatrix} \tag{3.22}$$

3.5 Modeling in Verilog-A

In the previous section, design of the sigma-delta modulator at the system level in MATLAB was shown. The next step in the implementation process is the design of the individual blocks at the circuit level and verification of the modulator at circuit level. This is usually done using spice simulations in Cadence. However, the simulations at

this stage tend to be much time-consuming and debugging of issues, if any, becomes difficult. An intermediate step here can be the building of a behavioral model for the system using ideal elements in the Cadence environment. Now system simulations can be performed more easily with circuits of the individual blocks replacing the ideal ones in the behavioral model.

In building the ideal model in Cadence, while elements of the passive filter and transconductance elements can be easily obtained from the basic devices available in the component library, more complex blocks like quantizers are not readily available. These components were modeled in this work using hardware description language Verilog-A [22]. The schematic of the system built in Cadence with Verilog-A models included is shown in Fig. 17.

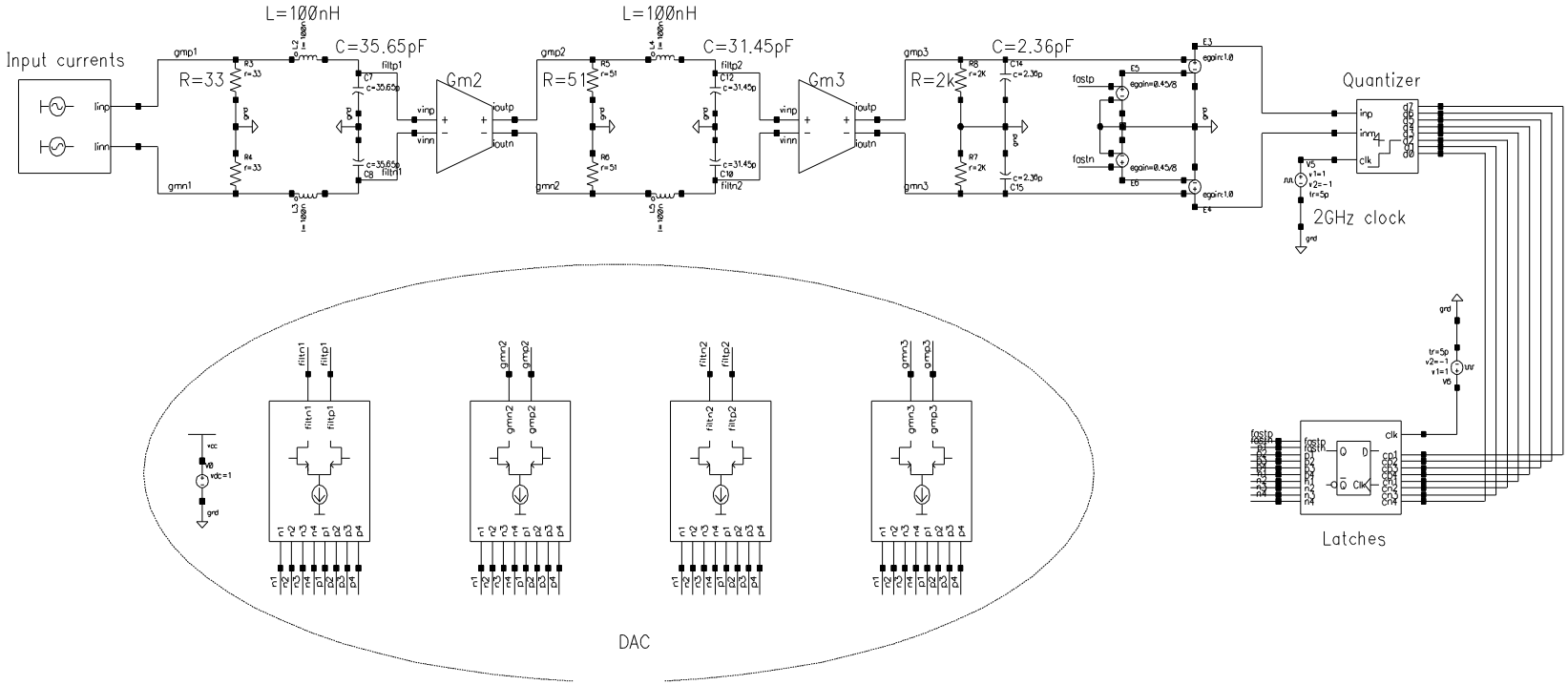


Figure 17 Cadence schematic of the system

3.6 Results

A behavioral model of the system was built in MATLAB using simulink. The output obtained from a simulation of the system is shown in Fig. 18. The input is a -3dBFS sine wave (the input amplitude for maximum SQNR as shown in Fig. 8) at 19.53MHz and the SNR is computed over a bandwidth of 100MHz.

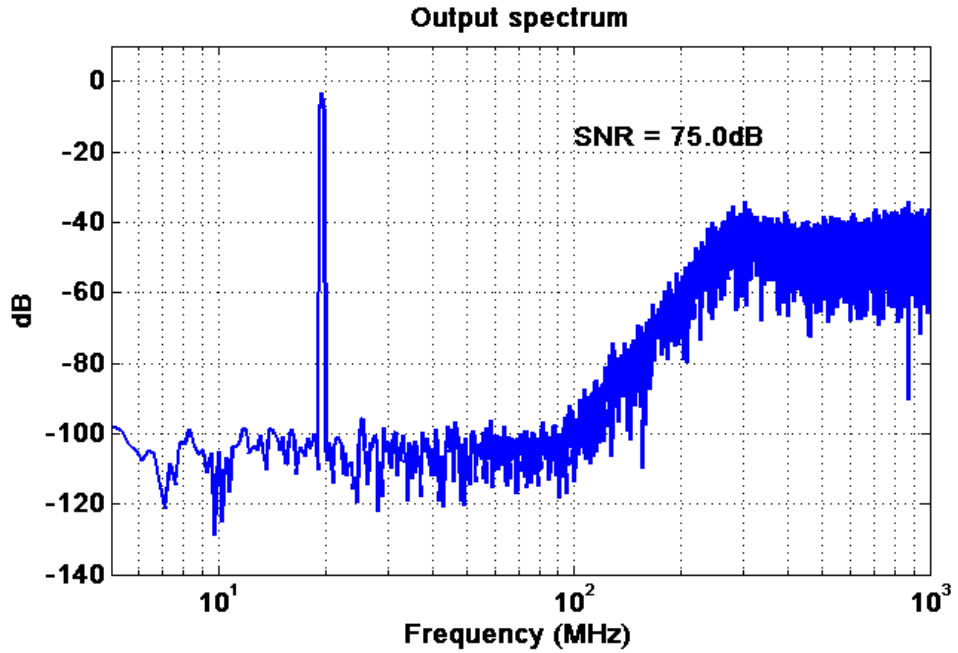


Figure 18 Output spectrum from ideal behavioral model

The closed loop transfer functions from the input to different points in the loop filter were obtained using simulink. An illustration of the loop filter with the different points annotated and the closed loop transfer function results are shown in Fig 19 and 20 below respectively.

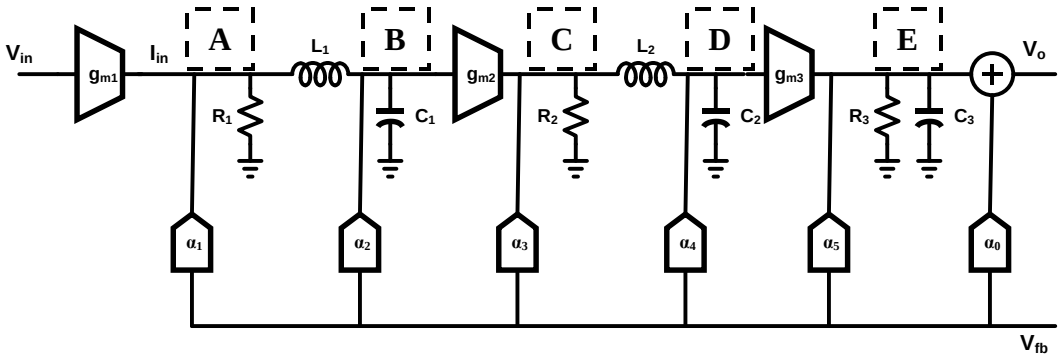


Figure 19 Loop filter

It can be seen from Fig 20 that out-of-band signals can have a gain of up to 10dB at node A in the loop filter. Care needs to be taken that this does not result in saturation in the filter stage.

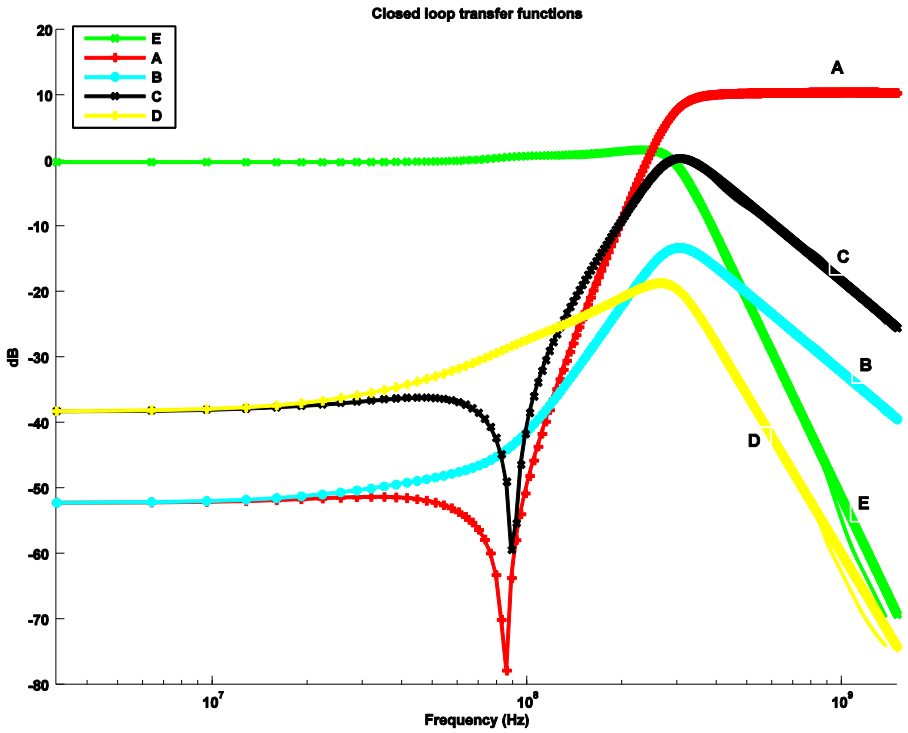


Figure 20 Closed loop transfer functions

4. MODELING OF NON-IDEALITIES

In section 3, the design for the sigma-delta modulator was performed at the system level using SIMULINK and Verilog-A in Cadence. The design process then assumed that all the components used in the loop are ideal and will behave exactly as assumed in the model. However, in the real world, this is not the case and various non-idealities are associated with the different elements in the loop. We will have a look at some of the major deviations from the ideal model in this section.

4.1 Non-idealities in the transconductors

4.1.1 Non-linearity in the transconductors

The proposed architecture in this work utilizes passive elements in the signal path to obtain the filtering action. However, a sufficient amount of gain is still needed in the forward path and active circuits are used for this purpose. In contrast to passive elements that are inherently linear, active circuits exhibit non-linearity and in the case of a transconductor, this non-linear relationship between input voltage and output current can be expressed as,

$$i = g_m . v + g_{m,3} . v^3 \qquad (4.1)$$

where, i is the output current of the transconductor, v is the input voltage of the transconductor, g_m is the linear transconductance and $g_{m,3}$ is the third order non-linear transconductance.

Only odd-order non-linearities are considered here, since the circuit will be implemented in a differential manner eliminating even-order disturbances.

Since the transconductor element g_{m2} in Fig. 19 does not have any gain in the signal path before it, any non-linearity introduced by the transconductor, when reflected back to the input, will appear without any attenuation. This non-linearity will then directly appear at the output due to the feedback action. Hence, the transconductor g_{m2} needs to have

stringent linearity requirements. In contrast, the signal received by the transconductor g_{m3} has already been amplified in the loop filter (with gain $g_{m2}R_2$) and hence its non-linearity is not as critical. In this work, an inter-modulation distortion (IM3), in a two-tone test, of 70dB is targeted.

Using equation (4.1) to model the non-linearity in the two transconductors, system simulations were performed in MATLAB. IM3 specifications for each of the blocks were derived to obtain an overall system IM3 of 70dB. The specifications derived are as shown in Table 5:

Table 5 Transconductor non-linearity specifications

Transconductor	**IM3 specification (dB)**
g_{m2}	68.5
g_{m3}	54.5

4.1.2 Excess phase in the transconductors

In the model of the transconductors considered so far, it is assumed that they have the same value of transconductance over all frequencies. In reality, there are finite impedances and capacitances within the transconductance circuit implementations and these tend to introduce one or more poles. In this work, the transconductors are modeled as single-pole systems and the output current is hence given by,

$$i = \frac{g_m}{1 + \frac{s}{\omega_p}} . v \tag{4.2}$$

where, ω_p represents the dominant pole of the transconductance block.

The primary effect of poles within the transconductance block is due to their phase characteristic that tends to introduce additional delay in the signal path. Since the sigma-delta modulator is a closed loop system, excess delay due to these blocks can cause the

loop to become unstable. Hence, we need to make sure that the poles are at high enough frequencies so that this is not an issue.

The effect of excess phase was modeled by using an RC network ahead of the ideal transconductance block. The transconductance model incorporating both non-linearity and excess phase is as shown in Fig. 21.

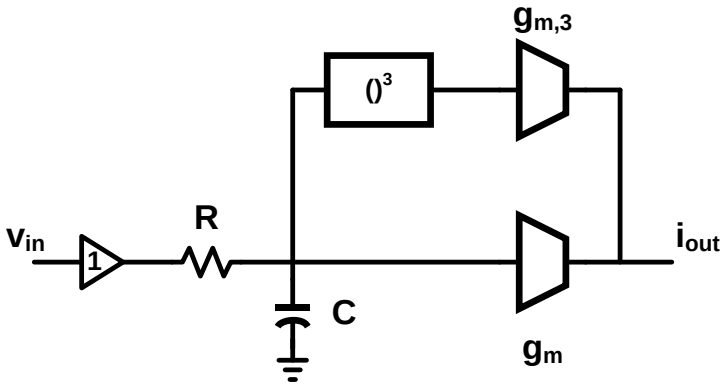


Figure 21 Transconductor model

After several time-domain simulations of checking the system for stability, the dominant poles of the transconductors were obtained as shown in Table 6.

Table 6 Transconductor excess phase specifications

Transconductor	Dominant pole (MHz)
g_{m2}	300
g_{m3}	500

4.1.3 Output capacitance

The transconductors when implemented at the transistor level finally will have finite values of parasitic capacitance at the output nodes. The capacitance of transconductor g_{m3} will cause a shift in the pole frequency of the 3rd stage of the filter. The external

capacitor needs to be tuned to compensate for this output capacitance. In the case of transconductor g_{m2}, the output capacitance appears in parallel with resistance R_2. It was observed that high value of this capacitance produces peaking in the output spectrum of the modulator. This is illustrated in Fig. 22 where a capacitance of 3.5pF was used on the output nodes.

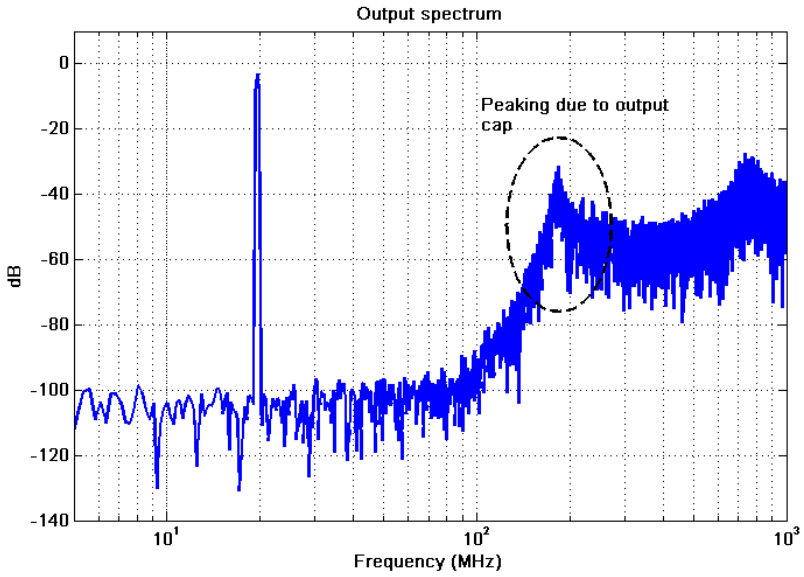


Figure 22 Output spectrum with peaking

In the presence of a blocker signal in the input around the peaking frequency, the modulator can become unstable. Even in the absence of a blocker, higher values of capacitance can cause the quantization noise to rise significantly due to the peaking to cause instability. To avoid this phenomenon, a value of 2pF for the output capacitance was arrived at to have some margin in the design.

4.2 Non-idealities in the DAC

4.2.1 Element mismatch

As discussed in section 3, high performance sigma-delta modulators use multi-bit internal quantizers to reduce the quantization noise power. However, the use of a multi-

bit quantizer necessitates the use of a multi-bit DAC in the feedback path. These multi-bit DACs are generally implemented as a summation of many unit elements and mismatch between these unit elements results in non-linearity being introduced at the output of the DAC. The output of the DAC enters the modulator loop at the same point as the input. Hence any non-linear components introduced at the output of the DAC appear directly along with the signal at the output and degrade the performance of the modulator.

There have been various methods reported in the literature, collectively called Dynamic Element Matching (DEM) techniques, to compensate for the non-linearity of the DACs. The essential idea in all these methods is to pre-process the digital data from the quantizer before applying it to the input of the DAC. This digital data processing has two effects [12]:

1. It makes sure that any error introduced by the DAC is no longer correlated to the input signal. Hence, the error that would have manifested itself as non-linear tones at the output gets converted to white noise.
2. Some of the techniques also tend to shape the mismatch-error introduced by the DAC out of the signal band thus reducing their damaging effect.

Some of the popular digital pre-processing techniques are DWA (Data Weighted Averaging) [23], ILA (Individual Level Averaging) [24].

Another way to compensate for the non-linearity of the DACs is to use analog calibration on the individual unit elements [25]. In this method, each of the unit elements is compared to an ideal reference and any existing error in the unit element is corrected. When calibration is used, the number of unit elements in the DAC is more than that actually required. This ensures that unused unit elements of the DAC can be calibrated while ensuring continuous operation with the other elements of the DAC.

In this work, calibration is used to combat non-linearity in the feedback DACs.

4.2.2 Output resistance of the DAC

The feedback DACs are implemented in current-mode and they connect to different points in the LC filter-section as shown in Fig 14. Since the output resistance of the DAC elements appears in parallel with the resistance element in the filter section, a low value of output resistance would degrade the gain in the forward path (at low frequencies) and raise the noise floor of the quantization noise in the output spectrum. Hence, a minimum value needs to be maintained for the output resistance in the DACs. Using time-domain simulations in MATLAB, the value shown in Table 7 was obtained for the output resistance.

Table 7 DAC output resistance specification

Output resistance	25kΩ

4.3 Quantizer non-idealities

4.3.1 Excess loop delay

Excess loop delay is a unique problem in continuous-time modulators because of the fact that the feedback waveform is continuous in nature and any delay in the feedback path changes the feedback into the loop filter. In this modulator, NRZ pulse shape is used for the feedback DAC and hence delay in the quantizer will appear as excess loop delay. Hence, a half clock period (250ps) delay each is assigned to the quantizer and feedback DAC and the 1 clock cycle delay is later compensated. In this work, a novel method is used to overcome the effect of excess loop delay. This and a couple of other methods reported in the literature will be discussed in detail in section 5.

4.3.2 Offset voltage

In sigma-delta modulators, errors associated with the quantizer are noise-shaped the same way as quantization noise and in general, are not critical. However, in high-order,

high-resolution implementation, care needs to be taken to make sure that the quantizer non-idealities do not limit performance.

In a quantizer, presence of an offset can be considered as a change in the threshold level and hence additional quantization error is introduced. Also, in a multi-bit quantizer, the individual comparators will each have a random offset voltage. Hence, the transfer function of the quantizer is no longer a straight line and the quantizer adds distortion to the output. These effects tend to reduce the signal-to-noise-and-distortion-ratio (SNDR) of the modulator.

A Monte-Carlo simulation of the modulator was performed with the offset voltage of the comparators being considered as random variables. The input signal in this simulation is a -9dBFS signal (half of the maximum stable amplitude). The variation of the SNR with offset voltage is shown in Fig. 23. At each value of offset voltage, 10 different simulations were performed. In Fig. 23, all the obtained SNR values are plotted and the minimum SNR values are shown connected by a solid line.

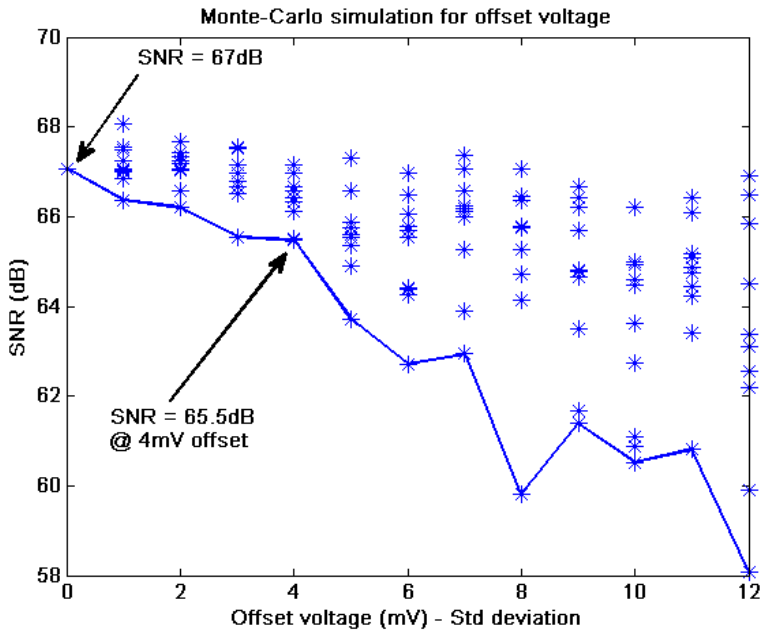


Figure 23 Monte-Carlo simulations for offset voltage of comparators

From Fig. 23, it can be concluded that, to keep SNR degradation due to offset voltage less than 2dB, the standard deviation of the offset voltage needs to be 4mV or smaller.

4.4 Clock jitter

Nyquist-rate and discrete-time sigma-delta ADCs are sampled-data systems. Hence any error in the sampling process appears together with the input to the ADC and degrades the SNR directly. However, in continuous-time sigma-delta modulators, the sampling occurs in the quantizer which is present after the loop filter. Hence, any errors introduced at this point due to sampling have the same transfer function to the output as the quantization noise and are hence noise-shaped outside the signal band.

Since oversampled converters are closed-loop systems, there is feedback to the input of the system and this feedback action is also controlled by the same clock used in the quantizer. Hence the effect of clock jitter needs to be considered at the input of the loop as well. Discrete-time modulators are insensitive to jitter at the input to the system because of their sampled nature. As long as the signal settles to the required accuracy within the clock period, any jitter in the feedback clock does not introduce any errors. However, in the case of continuous-time modulators, any timing error in the feedback signal is equivalent to an error in the feedback signal [12]. Since the transfer function from the feedback signal to the output is the same as the input, this error appears at the output without any attenuation.

The jitter noise power at the output of a continuous-time modulator is dependent on the shape of the DAC feedback pulse. This can be seen in Fig. 24 where the feedback waveform shapes in the case of NRZ, RZ and HRZ type feedback pulses [18] for the same modulator digital output are shown.

In Fig. 24, the digital pattern being considered is shown at the top of the figure. The effect of clock jitter is to change the point in time at which the feedback signal is applied at the input. The feedback applied between the ideal and jittered time instant manifests

itself as error applied at the input. This is shown by the shaded regions in each of the feedback pulses in Fig. 24.

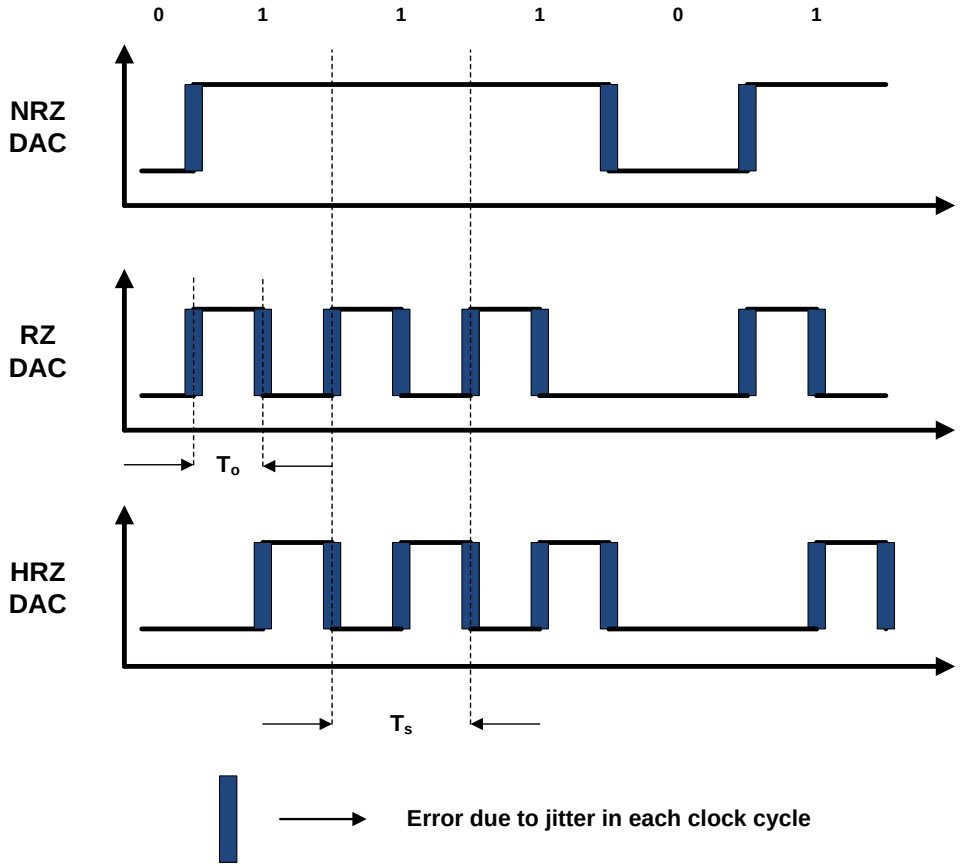


Figure 24 Effect of clock jitter on different feedback DAC pulses

Since the error due to clock jitter is injected into the modulator only during the feedback signal edges and the NRZ pulse shape has lower number of transitions compared to the RZ and HRZ shapes, the NRZ feedback shape is more tolerant to clock jitter. This is further quantified by the relations for the in-band error powers given by [12],

$$\sigma_{NRZ}^2 = \frac{1}{OSR}\left(\frac{\sigma_{jitter}}{T_s}\right)^2 \sigma_{dy}^2 \tag{4.3}$$

$$\sigma_{RZ}^2 = \frac{2}{OSR}\left(\frac{\sigma_{jitter}}{T_o}\right)^2 \sigma_y^2 \tag{4.4}$$

where, OSR is the oversampling ratio, σ_{jitter}^2 is the variance of clock jitter, T_s is the clock period, T_o is amount of time for which feedback is activated in RZ case, σ_y^2 is the variance of the digital output signal, y(n), of the modulator and σ_{dy}^2 is the variance of the signal (y(n) – y(n-1)).

The improvement in signal-to-jitter noise ratio when NRZ feedback is used in place of RZ can be obtained by taking the ratio of equations (4.3) and (4.4) [12] and is given by,

$$SJNR_{improvement}\ (dB) = 10log\left(\frac{\sigma_{RZ}^2}{\sigma_{NRZ}^2}\right) = 10log\left(2.\left(\frac{T_s}{T_o}\right)^2.\frac{\sigma_y^2}{\sigma_{dy}^2}\right) \tag{4.5}$$

Hence, in this work NRZ feedback is used to avoid degradation of SNR due to clock jitter.

4.4.1 Modeling jitter

To study the effect of clock jitter at the system level, the simplest method is to replace the ideal clock by a jittered clock with the required rms jitter power. However, this can increase the simulation time significantly because the simulation now has to account for clock edges that vary randomly around the ideal value. Hence, a different approach is necessary in this case [26].

When jitter is present in the feedback signal, the error introduced is in reality an error in the amount of charge fed into the loop filter. This error charge is given by the area in the jitter period in the current-time waveform. The same effect can be obtained by changing the signal amplitude by constant amount over the entire clock period to produce the same error charge. The required change in the signal amplitude is given by,

$$e_{j,NRZ}(n) = \big(y(n) - y(n-1)\big).\frac{\Delta t(n)}{T_s} \tag{4.6}$$

This is shown pictorially in Fig. 25 [12].

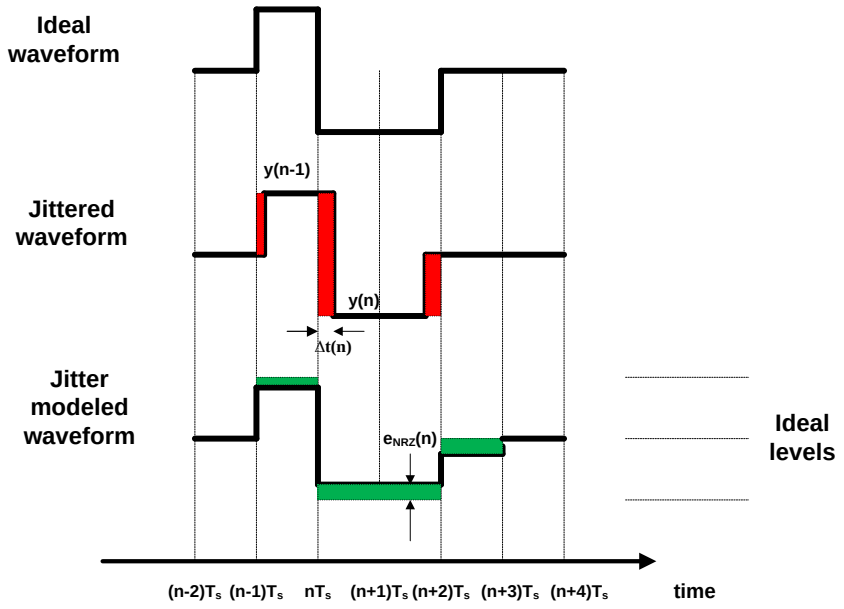


Figure 25 Modeling effect of clock jitter

The approach shown in equation (4.6) can be used to model the effect of jitter in a continuous-time sigma-delta modulator. The fact that the simulation now uses ideal clock edges once again helps to reduce the simulation time. The MATLAB model used to study the jitter phenomenon is shown in Fig. 26.

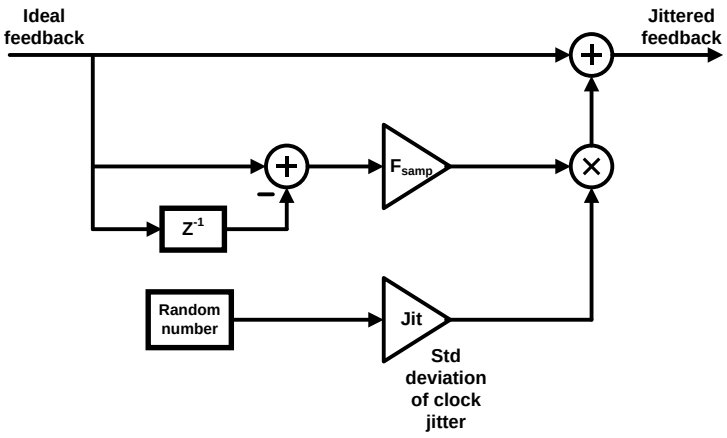


Figure 26 Clock jitter model in MATLAB

The model shown in Fig. 26 was used to study the effect of jitter on the proposed modulator by adding different amounts of jitter to the clock and noting the change in the SNR at the output. The result obtained is shown in Fig. 27. From the figure, we can see that to maintain the SNR in the presence of jitter above 70dB, we need a clock source with an RMS jitter less than 0.25ps.

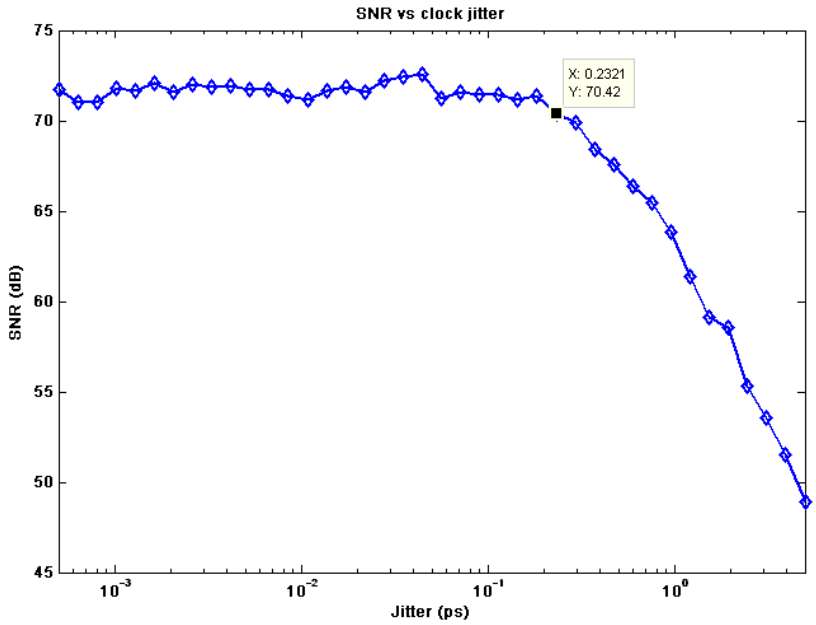


Figure 27 SNR as a function of clock jitter

4.5 Non-ideal inductor

The inductors used in this work would be fabricated outside the chip but within the package itself. This inductor would have non-idealities of its own and these were taken into account by using a quality factor of 20 and a self-resonant frequency of 1.6GHz. The model of the real inductor is shown in Fig. 28.

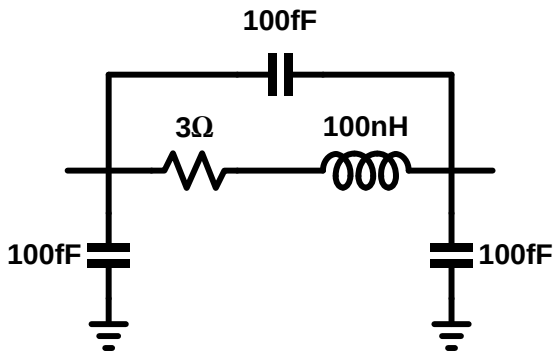


Figure 28 Non-ideal inductor

4.6 Thermal noise

Apart from the non-idealities described earlier, thermal noise in the circuits also needs to be taken into consideration. In this section, thermal noise specifications for the blocks will be obtained.

The resolution of the modulator in this work is targeted at 11 bits. This implies a signal-to-noise ratio (SNR) of 68dB. Assigning equal weights to thermal noise and the sum of other noise sources, we obtain a signal-to-thermal noise ratio of 71dB. To give some margin in the design, a target of 72dB is used. Since the differential input signal is $250mV_{peak}$, the above SNR value leads to an rms noise value of 44μV.

The loop filter with the different noise sources from the first stage is shown in Fig. 29. Noise sources of the stages ahead are not considered because they are attenuated by the gain factor $g_{m2}R_2$.

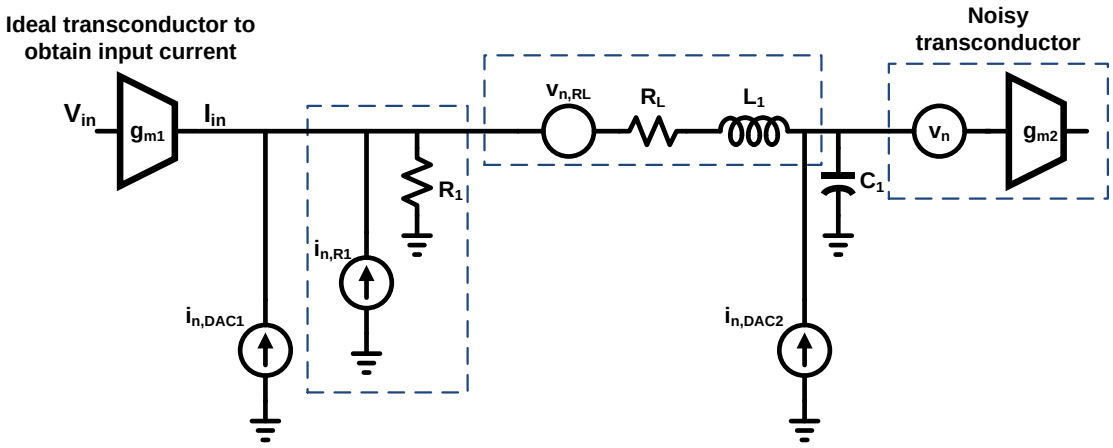


Figure 29 Noise sources in the loop filter

The input referred noise power spectral density is given by,

$$v_{noise}^2 = \frac{1}{g_{m1}^2}\left[i_{n,R1}^2 + i_{n,DAC1}^2 + \frac{v_{n,RL}^2}{R_1^2} + i_{n,DAC2}^2\left|1 + \frac{sL_1}{R_1}\right|^2 + \frac{v_n^2}{R_1^2}|s^2L_1C_1 + sR_1C_1 + 1|^2\right] \tag{4.7}$$

where,

$$i_{n,R1}^2 = \frac{4kT}{R_1} \tag{4.8}$$

$$i_{n,DAC1}^2 = 4kT\left(\frac{2}{3}g_m\right) \tag{4.9}$$

$$i_{n,DAC2}^2 = 4kT\left(\frac{2}{3}g_m\right) \tag{4.10}$$

$$v_{n,RL}^2 = 4kTR_L \tag{4.11}$$

The values of the various circuit elements in equation (4.7) and Fig. 29 are given in Table 3. Using the given values and picking a value of 200mV for the $v_{d,sat}$ of the current sources implementing the DACs, the input-referred noise contributions from the various blocks for a differential implementation are obtained as shown in Table 8.

Table 8 Input referred RMS noise values

Block	RMS noise
Resistor R_1	10.5μV
Inductor resistor R_L	3.3μV
DACs	22.8μV
Transconductor g_{m2}	36μV

4.7 Variation of pole locations

The loop filter pole locations in discrete-time modulators are set by ratios of capacitors and hence can be defined with good accuracy. However in continuous-time modulators they are either set by RC products or g_m/C ratios and can vary by as much as ±30%. In this work, the pole locations are set by LC products and are subject to the same variations. To assess the behavior of the modulator in presence of component variations, simulations of the modulator were performed where the values of L and C were varied from about -10% to 30%. The SNR of the modulator obtained under these conditions is shown in Fig. 30.

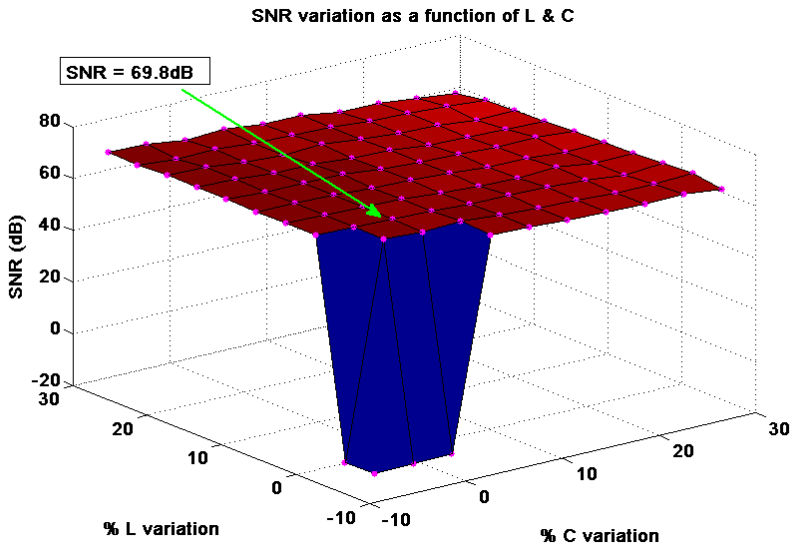


Figure 30 SNR variations with change in values of L and C

From Fig. 30, we can see that the SNR at the default values of L and C (as shown by the arrow) is 69.8dB. As the values of L and C increase, the poles move to lower frequencies and the noise shaping of the loop becomes less optimum and hence the SNR reduces as seen in Fig. 30. On the other hand, reductions in the L and C values move the poles to high frequencies and provide better noise shaping for a while. But soon the modulator becomes unstable and SNR drops drastically. For the current work, we can see that this happens when the L and C values reduce by about 5%.

Similarly, the pole in the 3rd stage of the filter is defined by the RC product in that stage. Variation of SNR of the modulator with variation in the R and C is shown in Fig. 31.

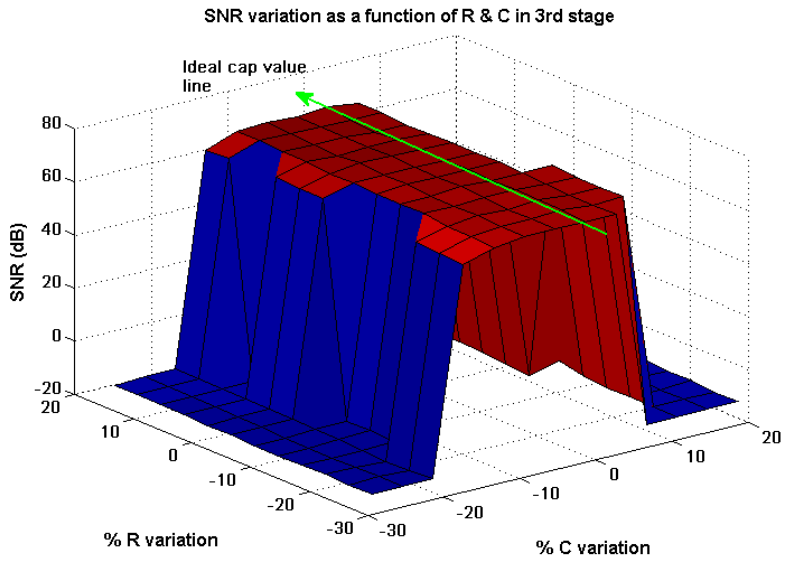


Figure 31 Variation of SNR with change in the R & C values of 3rd stage

We can see from Fig. 31 that when the value of the capacitor increases by more than 4%, the modulator can become unstable.

To counter the effects of change in pole locations due to variations in component values, we need to be able to tune the values of the components to obtain the desired poles. In this work, this is accomplished by implementing the capacitor as a capacitor bank controllable by a digital code.

4.8 Results

The non-idealities discussed in this section were incorporated into the ideal behavioral model of the system in MATLAB. The output obtained for the same input conditions used earlier in section 3 (-3dBFS sine wave at 19.53MHz) is shown in Fig. 32.

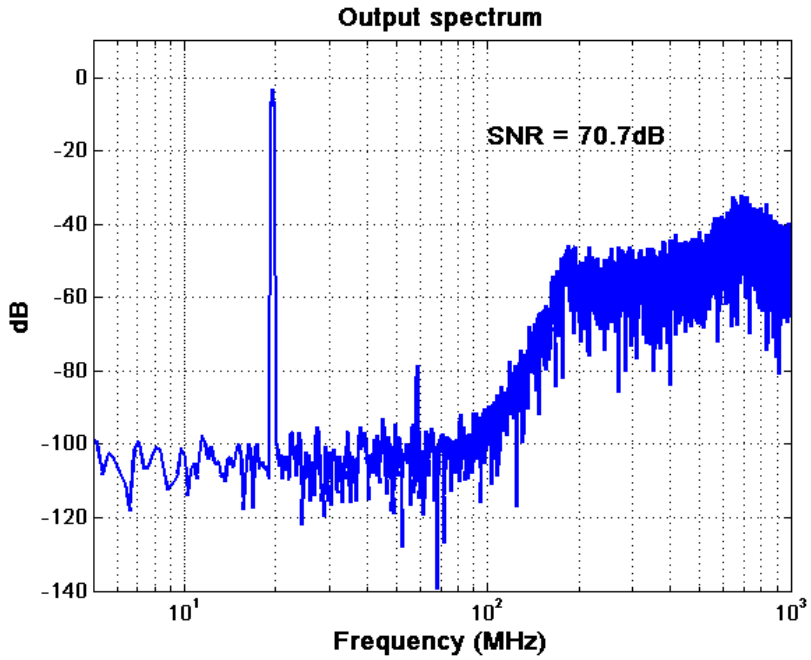


Figure 32 Output spectrum of modulator with non-idealities

The output of a two-tone simulation to observe non-linearity is shown in Fig. 33. The inputs in this case are 2 -9dBFS sine waves at 87.9MHz and 92.8MHz, close to the signal bandwidth of 100MHz.

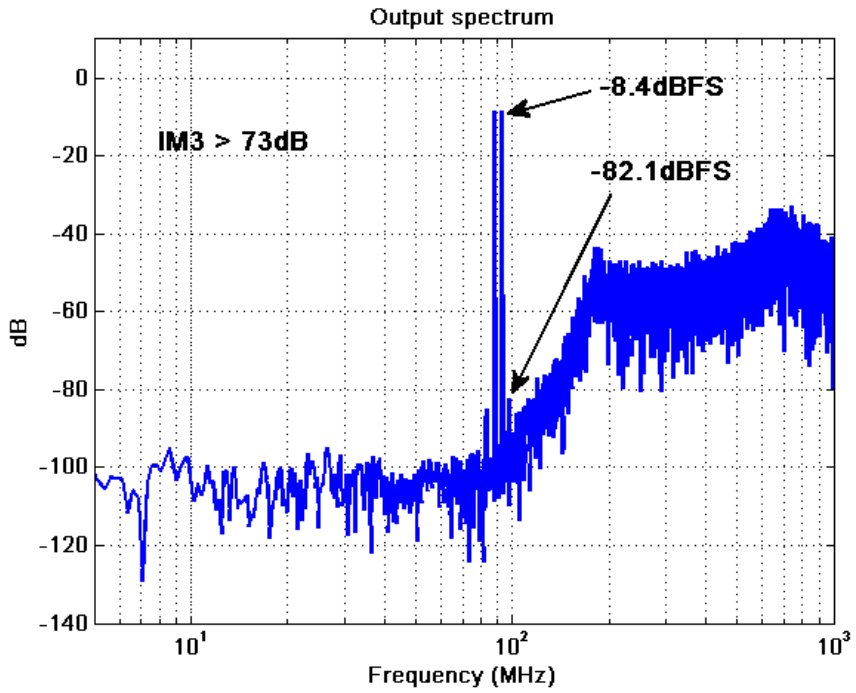


Figure 33 Two-tone simulation of modulator with non-idealities

5. EXCESS LOOP DELAY

Excess loop delay (ELD) is a non-ideality that is unique to continuous-time (CT) sigma-delta modulators. ELD is caused due to the fact that the feedback pulse cannot be fed back instantaneously after the sampling instant due to delays in the quantizer and feedback DACs. This is not an issue in discrete-time sigma-delta modulators since they are sampled-data systems and only the final value of the settled waveforms are of interest. In multi-bit continuous-time modulators, if any linearization techniques like Dynamic Element Matching (DEM) are used to compensate for mismatches in the DACs, the additional delay aggravates the problem of ELD.

5.1 Effect on stability

Fig. 34 shows a typical continuous-time sigma-delta modulator loop, where τ_d represents the excess delay in the loop.

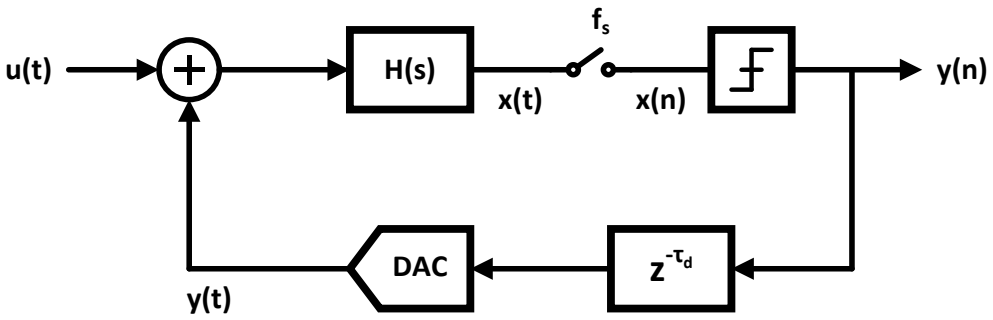


Figure 34 Continuous-time sigma-delta modulator with excess loop delay

Fig. 35 shows the effect of such a delay on the feedback pulse in the system.

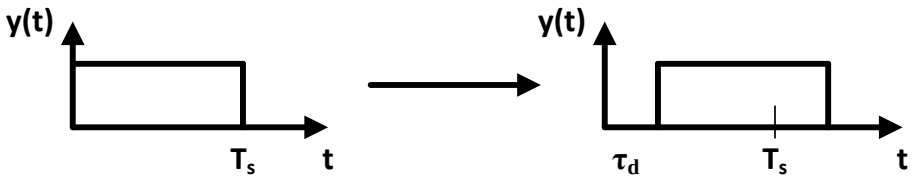


Figure 35 Feedback pulse in a modulator with excess loop delay

It was shown in section 3 that every continuous-time sigma-delta modulator can be transformed into an equivalent discrete-time one which has the same impulse response at the sampling instants by means of the impulse invariant transformation. From Fig. 35, it can be seen that the effect of the delay is that the feedback pulse is extended beyond the current sampling period. It has been shown that this alters the equivalence between the designed continuous-time modulator and corresponding discrete-time system [27]. In particular, this increases the order of the loop and can lead to instability.

5.2 Compensation Techniques

Various techniques have been reported in the literature to compensate for the effect of excess loop delay in continuous-time sigma-delta modulators and guarantee stability of the modulator. A few of them are noted below.

5.2.1 Compensation using a direct path around the quantizer

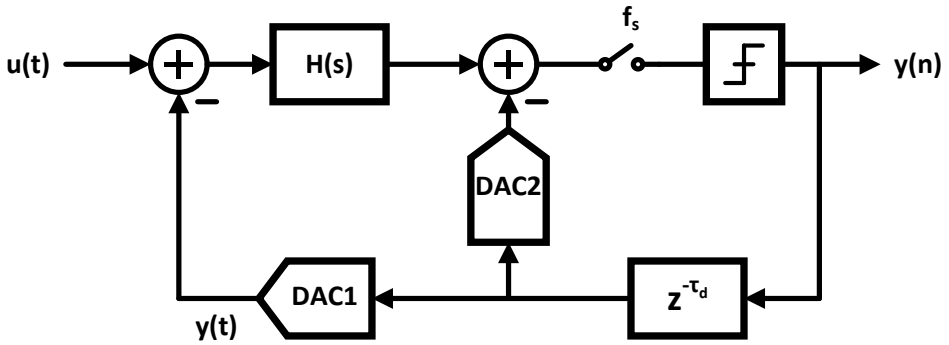


Figure 36 Loop delay compensation with direct path around quantizer

In the previous page, it was mentioned that the excess delay in the loop disturbs the equivalence between the continuous-loop and the corresponding discrete-time loop. Put another way, the impulse response of the continuous-time and discrete-time loops are no longer the same. By inclusion of the extra DAC as shown in Fig. 36, we are striving to once again make the impulse response of the continuous time loop, the same as that of the desired discrete-time loop.

The compensating action of the extra DAC is illustrated in the Fig. 37 for the special case where the excess delay is one complete clock cycle [28].

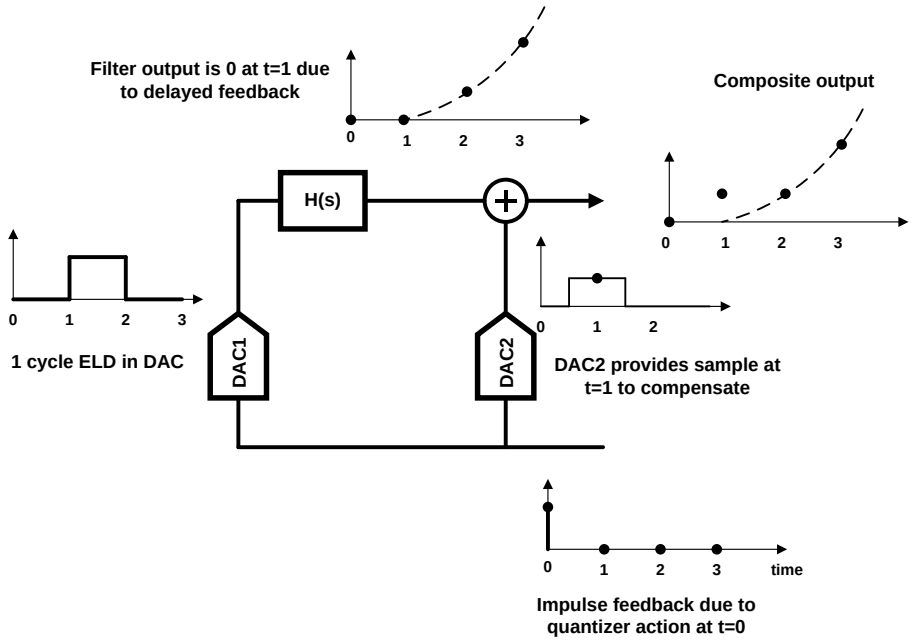


Figure 37 Principle of direct path compensation scheme

5.2.2 Compensation using a digital differentiator

The disadvantage of the compensation scheme discussed in the last section is that it adds a new summing amplifier in the feedback path. If this summing block has a significant delay by itself, it can become a bottleneck for loop stability and hence significant power must be burnt in the summer for fast operation.

A typical way to implement the loop filter in continuous-time sigma-delta modulators is using a cascade of integrators. In this case, if the quantizer is preceded by an integrator, the compensation for ELD can be simplified from the previous scheme by moving the additional feedback input from the output of the last integrator to its input. In this case, since the feedback signal sees an integration operation before being sampled, it must first be differentiated. Since the feedback is implemented by means of switched currents,

differentiation can be easily obtained by simple addition of 2 currents. The sequence of obtaining this type of compensation is shown in Fig. 38 [6].

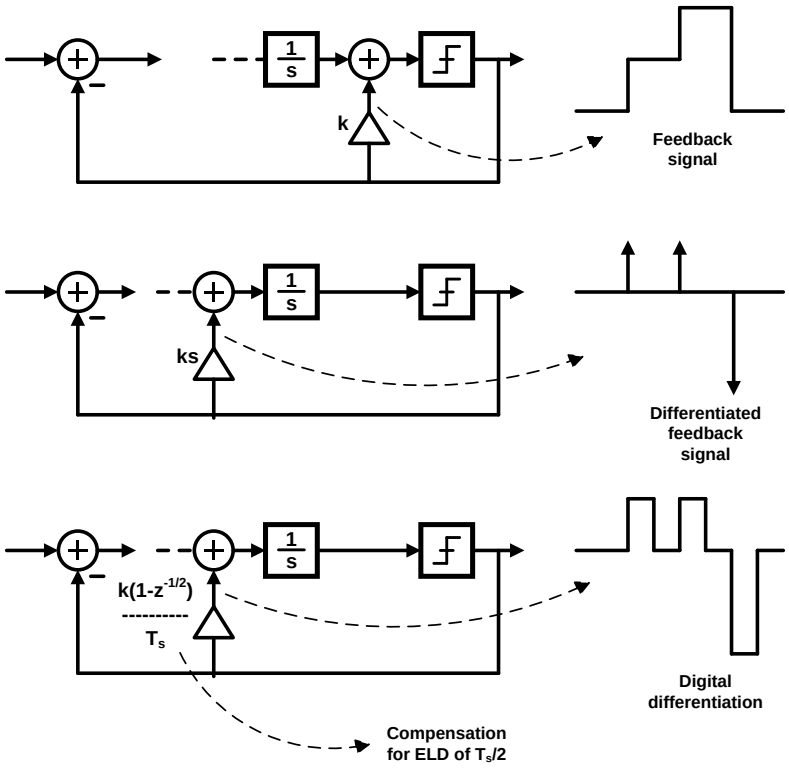


Figure 38 Principle of digital differentiator compensation scheme

The continuous-time modulator with the above compensation scheme incorporated in the loop is shown in Fig. 39. Note that for stability, the delays introduced must meet the condition [29],

$$\tau_d + \tau_d^{'} \leq {}^{1}/_{f_s} \qquad (5.1)$$

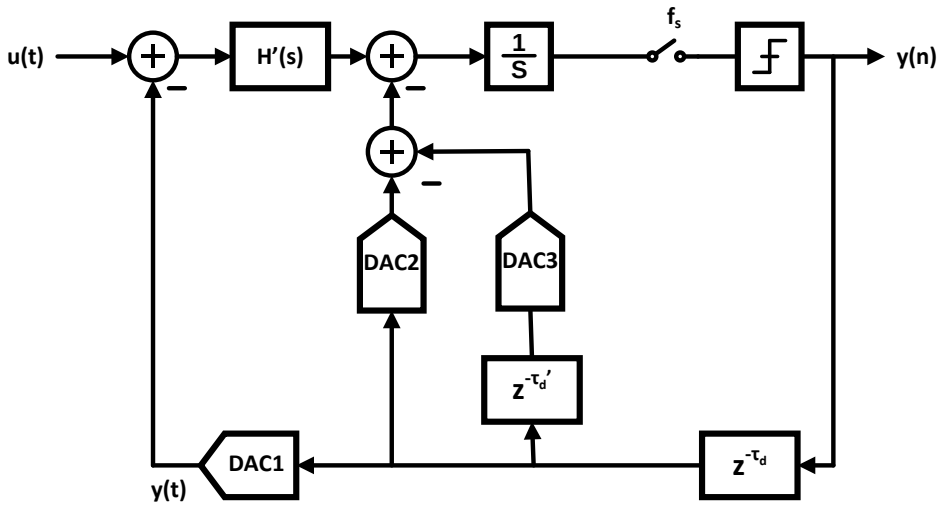


Figure 39 Modulator with digital differentiator based loop compensation

It must be noted that the methods described above are just a couple of the schemes found in literature to compensate for ELD. A good description of the effects and compensation techniques for ELD can be found in [27]. Also, [29] gives a good comparison of the above described and some more additional techniques to compensate for ELD in continuous-time modulators.

5.3 Proposed compensation technique

In the current modulator design, because of the high signal bandwidth targeted, the sampling frequency is a high value of 2GHz. Hence, excess delay in the loop is a real concern and compensation for the loop delay must be designed carefully. As mentioned in section 3.1, a loop delay of 1 clock cycle (500ps) is accounted for in the design and is equally divided between the quantizer and the DAC. Compensation on the lines of scheme described in the section 5.2.1 would need a very high GBW summing amplifier leading to significant power dissipation. Hence, every effort was made to use a compensation scheme that would not need any additional summing amplifier in the feedback loop.

From section 3, it is seen that the last stage of the loop filter is realized by a transconductance stage feeding current into an RC filter. The fact that the input of the

quantizer has an RC section is used to realize the direct feedback path in voltage mode using a resistive-capacitive voltage divider. The crux of the scheme for a single bit of the multi-bit feedback is shown in Fig. 40.

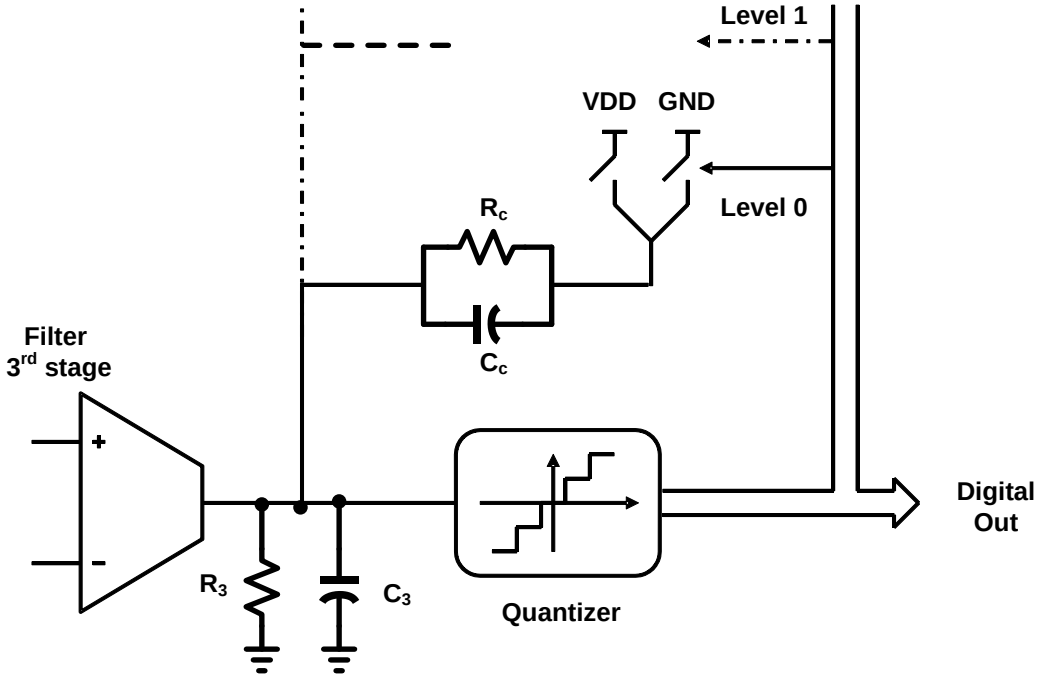


Figure 40 Proposed loop delay compensation scheme

The derivation of the values of the resistors and capacitors in the above architecture to satisfy system requirements is shown below.

Let ‘k’ be the feedback coefficient from each bit back to the input of the quantizer. Also, let

$$Z_c = R_c||C_c = \frac{R_c}{1 + sR_cC_c} \tag{5.2}$$

$$Z_3 = R_3||C_3 = \frac{R_3}{1 + sR_3C_3} \tag{5.3}$$

Since this work uses a 9-level quantizer, we have

$$k^{'} = \frac{k}{V_{dd}} = \frac{\frac{Z_c}{7} || Z_3}{\frac{Z_c}{7} || Z_3 + Z_c} \tag{5.4}$$

Substituting the definitions above for Z_c and Z_3 and simplifying, we get

$$k' = \frac{R_3}{R_c + 8R_3}\left[\frac{1 + sC_cR_c}{1 + s\frac{R_cR_3}{R_c + 8R_3}(C_3 + 8C_c)}\right] \tag{5.5}$$

At low frequencies near DC where the resistances dominate the impedances over the capacitors, we can simplify the above equation to

$$k' = \frac{R_3}{R_c + 8R_3} \tag{5.6}$$

Simplifying the above, we get

$$\frac{R_3}{R_c} = \frac{k'}{1 - 8k'} \tag{5.7}$$

Similarly, at high frequencies where the capacitors dominate, performing similar calculations, we get

$$\frac{C_c}{C_3} = \frac{k'}{1 - 8k'} \tag{5.8}$$

We can show that when the elements R_c, R_3, C_c and C_3 satisfy the relations given above, the pole and zero in equation (5.5) are at the same frequency and hence the feedback is constant over all frequencies.

Let the 3-dB frequency of the 3rd stage of the filter be f_{3dB}, and the equivalent resistor and capacitor be R_{eq} and C_{eq} so that

$$f_{3dB} = \frac{1}{2\pi R_{eq} C_{eq}} \tag{5.9}$$

The equivalent impedance seen by the transconductance element in the 3rd stage is given by

$$R_{eq} = \frac{R_c}{8} || R_3 \tag{5.10}$$

$$C_{eq} = 8C_c + C_3 \tag{5.11}$$

Using the relations derived between R_c, R_3, C_c and C_3, we can simplify the equations (5.10) and (5.11) to,

$$R_{eq} = k' R_c \tag{5.12}$$

$$C_{eq} = \frac{C_c}{k'} \tag{5.13}$$

In summary, knowing the desired 3-dB frequency of the filter and the feedback coefficient from the quantizer, we can obtain the values of the elements forming the feedback network as shown,

$$R_c = \frac{R_{eq}}{k'} \tag{5.14}$$

$$C_c = k' C_{eq} \tag{5.15}$$

$$R_3 = \frac{R_{eq}}{1 - 8k'} \tag{5.16}$$

$$C_3 = (1 - 8k') C_{eq} \tag{5.17}$$

In the current design, the value of the feedback coefficient, 3-dB frequency of the filter and hence the values of the passive elements are given in Table 9.

Table 9 Loop delay compensation circuit parameters

k	0.9/8	**R_c**	32kΩ
f_{3dB}	33.15MHz	**R_3**	4kΩ
R_{eq}	2kΩ	**C_c**	0.15pF
C_{eq}	2.4pF	**C_3**	1.2pf

The sizing of the transistors implementing the switches was done as a compromise between having small resistance for good settling and presenting a small load to the previous stage for easier driving capability. The buffers for the switches were implemented with CML logic for high-speed operation.

The compensation scheme discussed in this section was included in the sigma-delta modulator designed in section 3. The output spectrum of the modulator for identical input conditions used earlier (-3dBFS at 19.53MHz) is shown in Fig. 41.

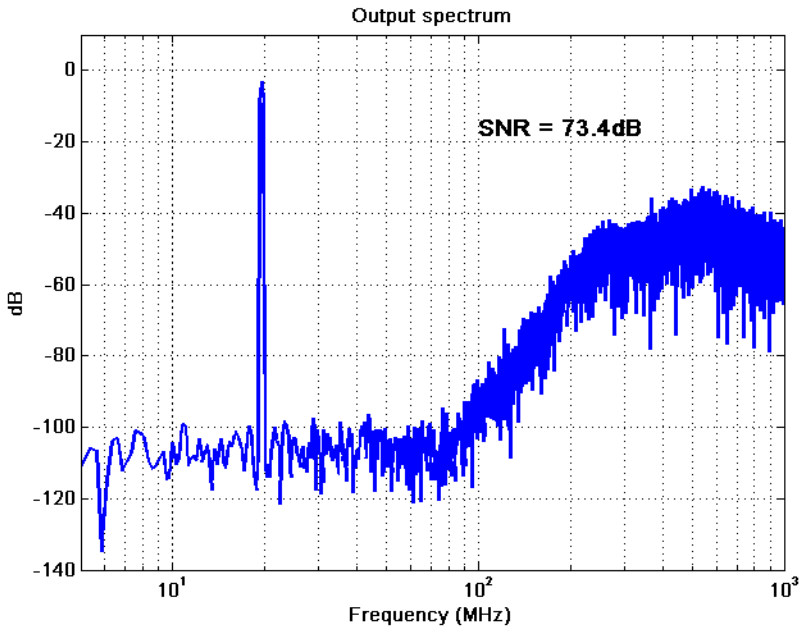


Figure 41 Output spectrum with proposed loop delay compensation

6. SUMMARY

In sections 3-5, a sigma-delta modulator able to digitize the required signal bandwidth of 100MHz was designed and the building blocks modeled with their corresponding non-idealities. In this section, the specifications obtained for the individual blocks will be summarized and the entire system will be shown.

6.1 Specifications

The block diagram of the entire system is shown once again for reference in Fig. 42. This representation also includes the loop delay compensation as proposed in section 5.

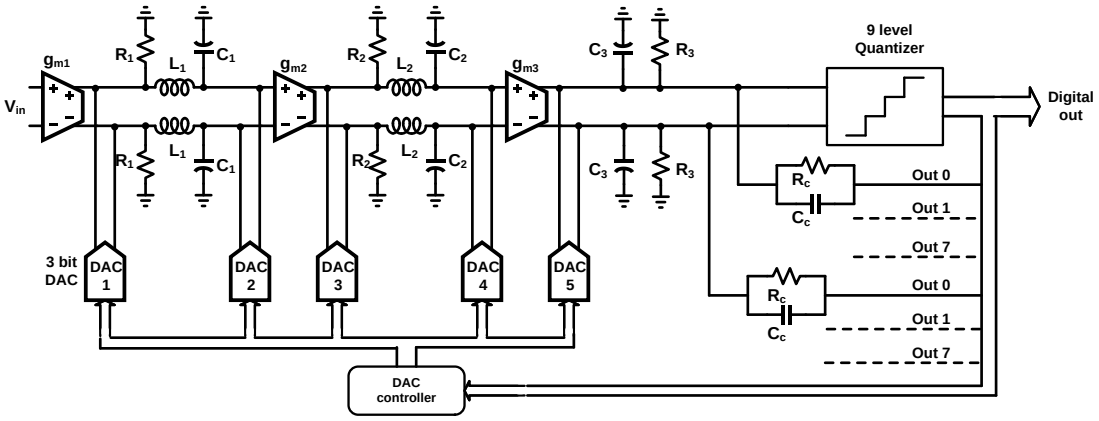


Figure 42 Modulator implementation with proposed loop compensation

The values of the components in the system are given in Table 10.

Table 10 Component values

DAC1	0	**L_1**	100nH	**L_2**	100nH
DAC2	5.16m	**C_1**	35.6pF	**C_2**	31.4pF
DAC3	12.4m	**R_1**	33Ω	**R_2**	51Ω
DAC4	7.68m	**g_{m2}**	100mS	**g_{m3}**	40mS
DAC5	6.32m	**R_3**	4kΩ	**C_3**	1.2pF
g_{m1}	30mS	**R_c**	32kΩ	**C_c**	0.15pF

The specifications for the individual blocks are listed in Tables 11 through 14.

Table 11 Transconductor specifications

Parameter	**g_{m2}**	**g_{m3}**
Value	100mS	40mS
IM3	68.5dB	54.5dB
Dominant pole	300MHz	500MHz
Input-referred noise density	4.3nV/√Hz	21.5nV/√Hz

Table 12 DAC specifications

Parameter	**Value**
Output resistance	25kΩ
Delay	250ps
Calibration accuracy	70dB

Table 13 Quantizer specifications

Parameter	Value
No of levels	9
Delay	250ps
Offset voltage (standard deviation)	4mV

Table 14 Clock specifications

Parameter	Value
Frequency	2GHz
Jitter	0.25ps

The complete model of the system in MATLAB including the effect of clock jitter is shown in Fig. 43. Similarly, the complete system implemented in Cadence using the proposed loop delay compensation method is shown in Fig. 44. The schematics of the individual blocks, namely DAC implementation, loop delay compensation circuit and a CML-to-CMOS converter [30], are shown in figures 45, 46 and 47 respectively.

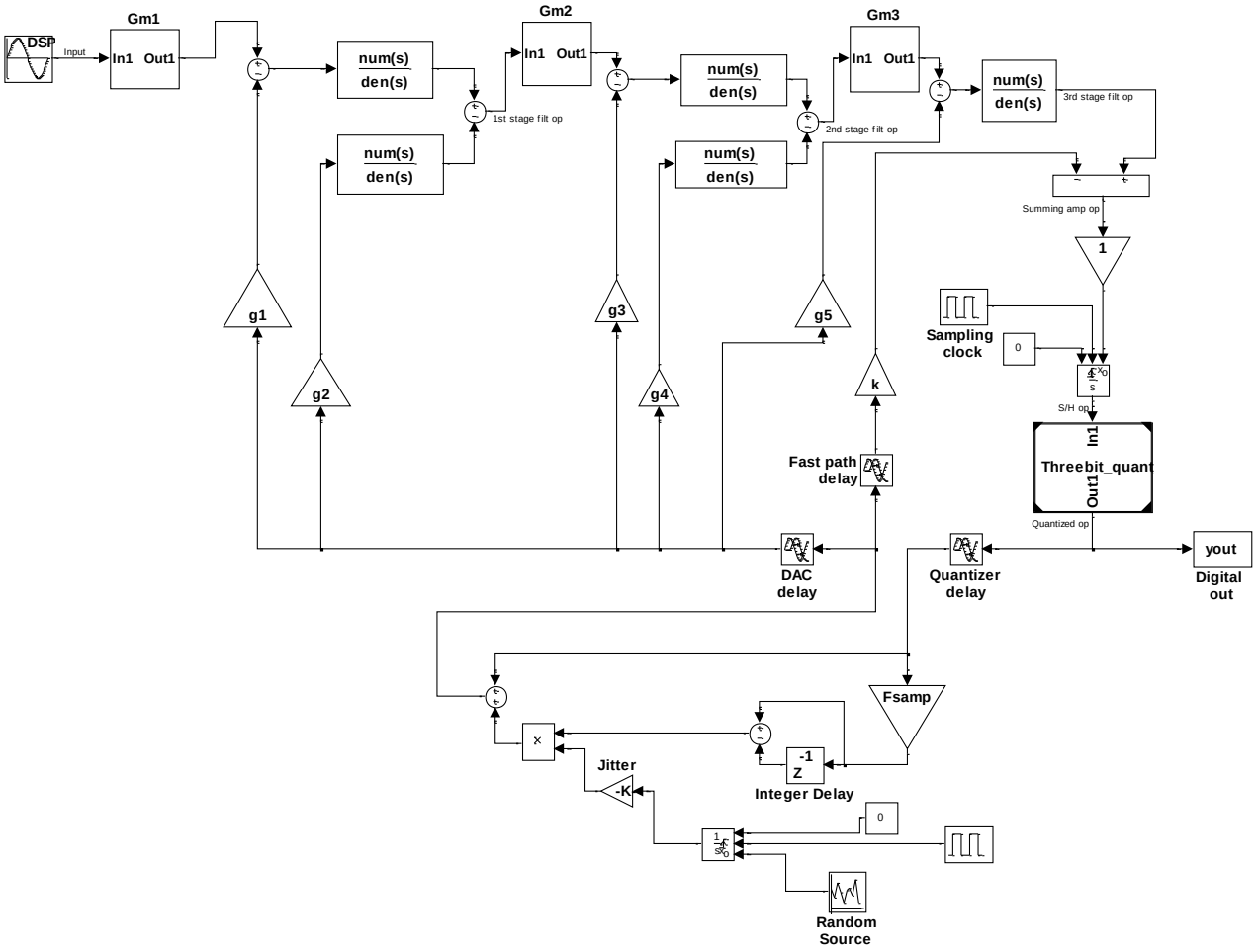


Figure 43 MATLAB model of complete system with clock jitter

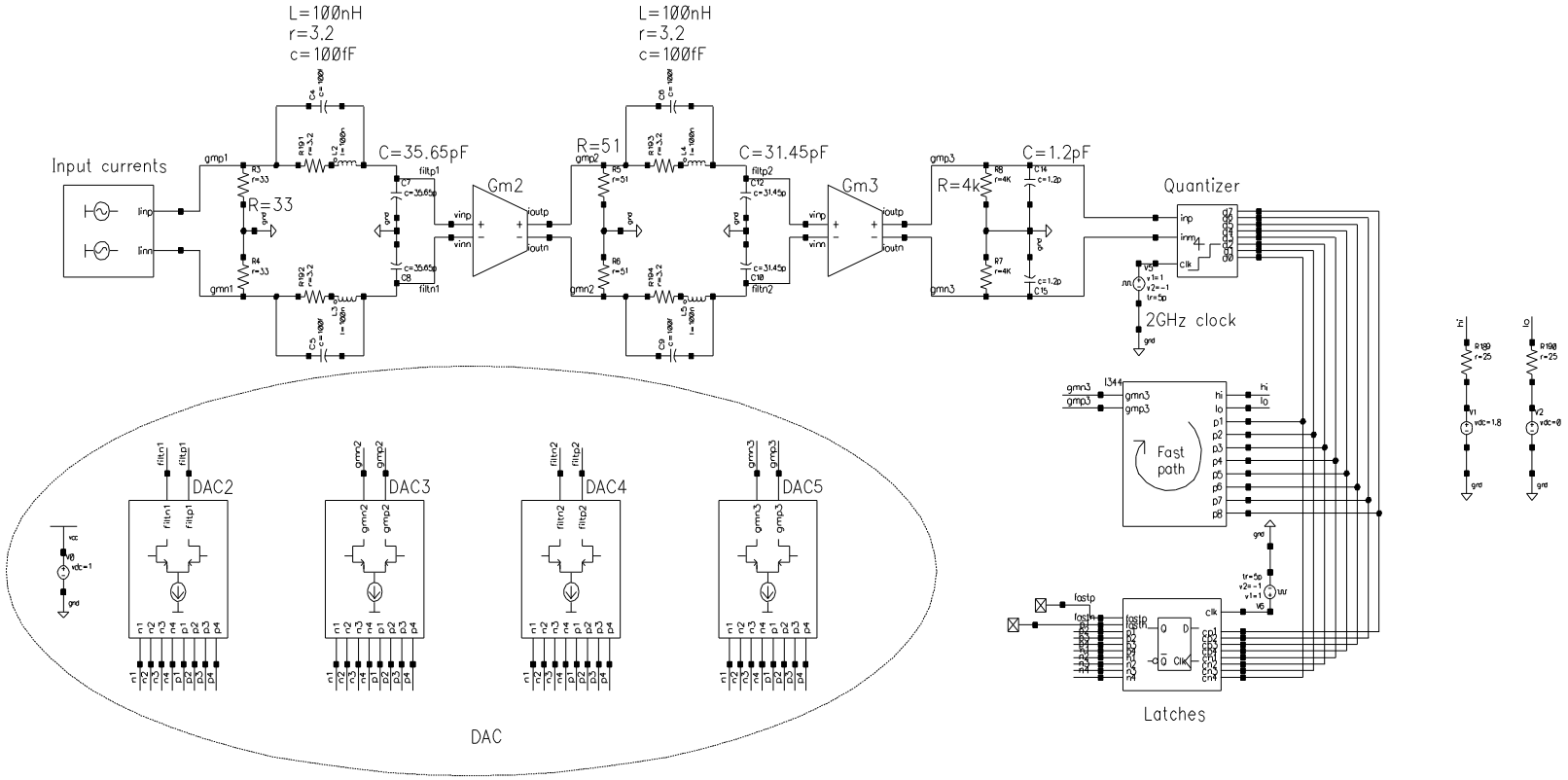
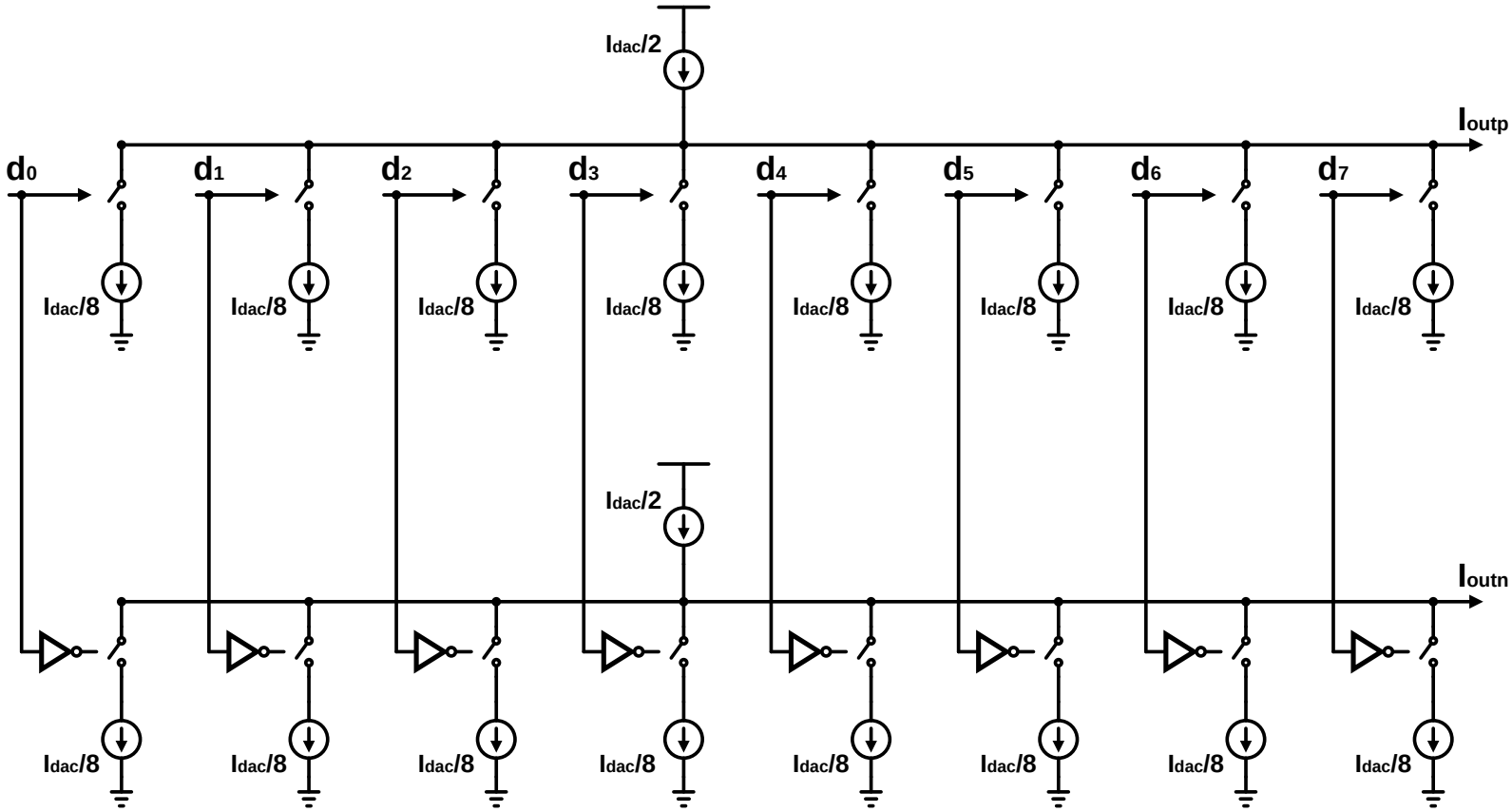


Figure 44 Cadence schematic of the complete system with proposed loop delay compensation



For the feedback current, use the corresponding value for each DAC from Table 10

Figure 45 DAC implementation

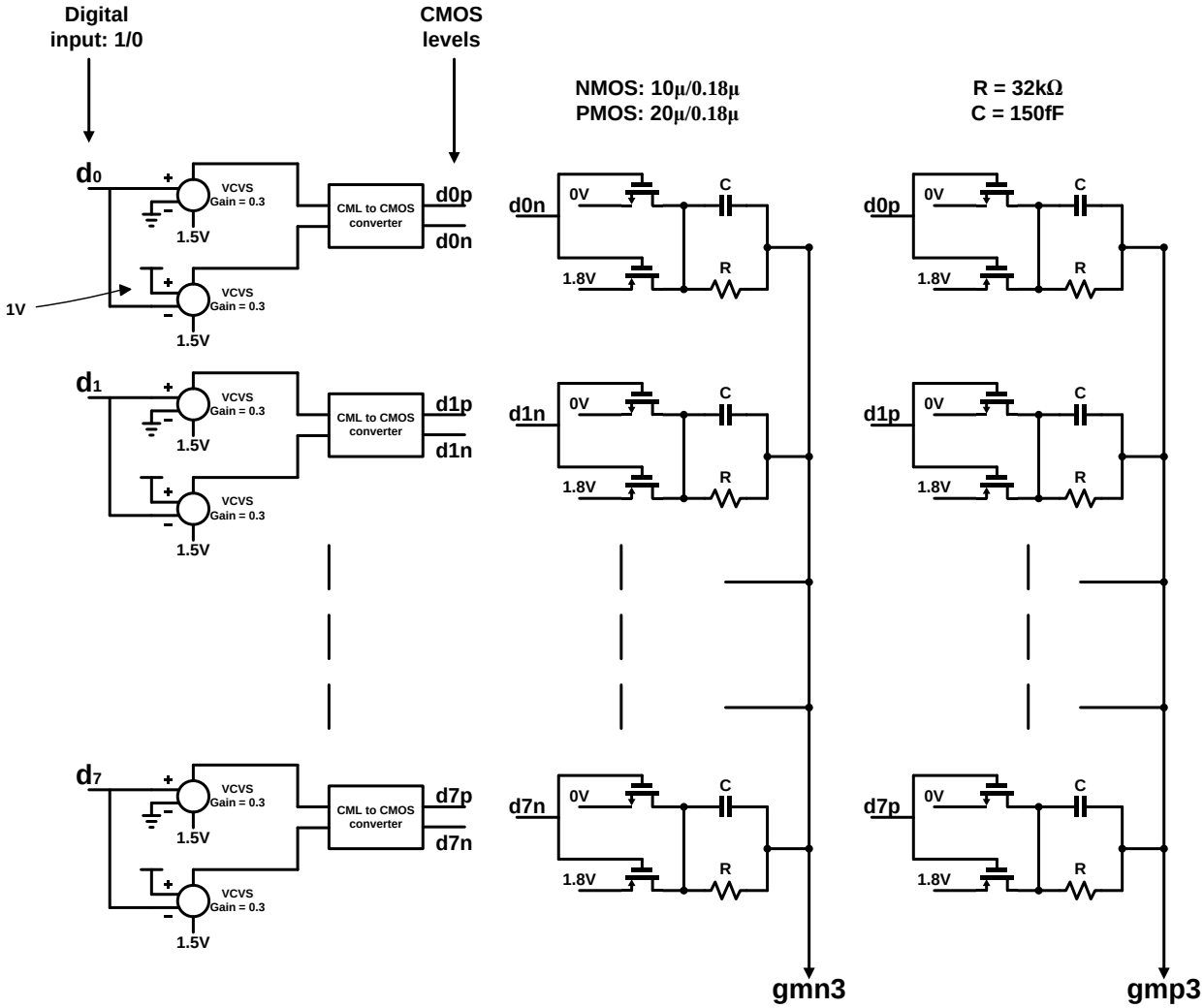


Figure 46 Loop delay compensation circuit

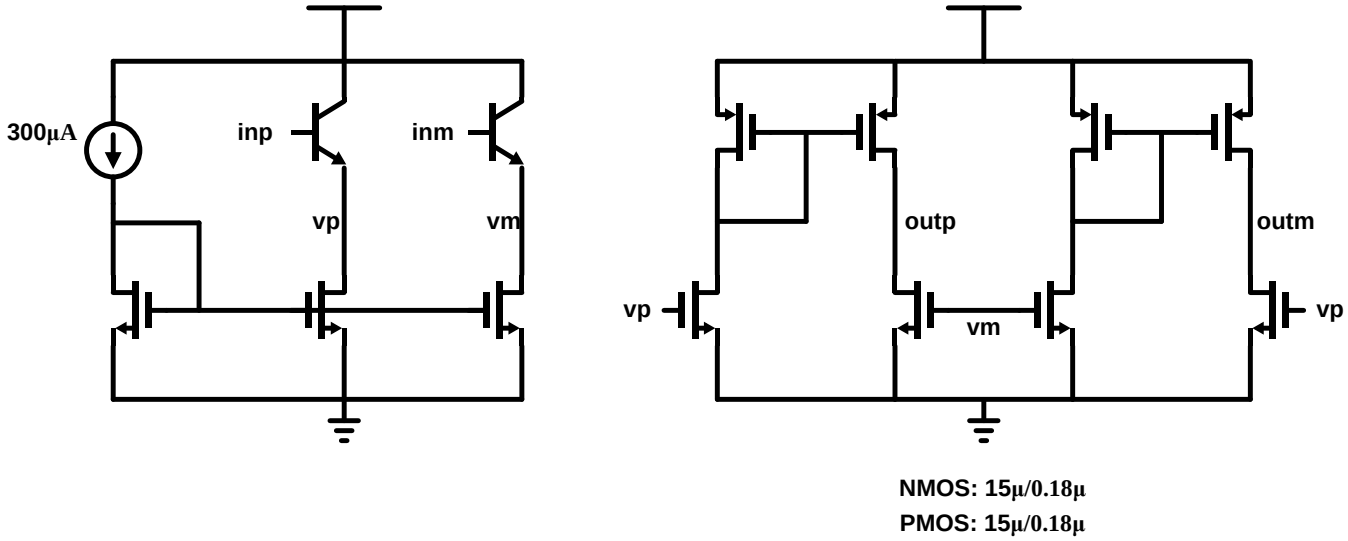


Figure 47 CML-to-CMOS converter

6.2 Results

The output spectrum obtained from the simulation of the complete system for an input of -3dBFS at 19.53MHz is shown in Fig. 48.

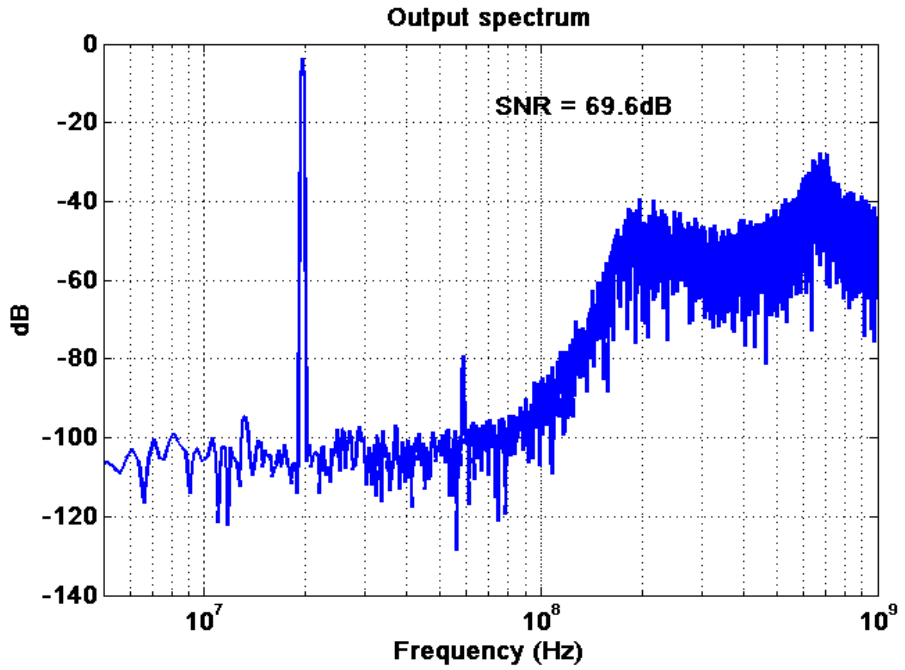


Figure 48 Output spectrum from complete system simulation

The output for a two-tome simulation to test linearity is shown in Fig. 49.

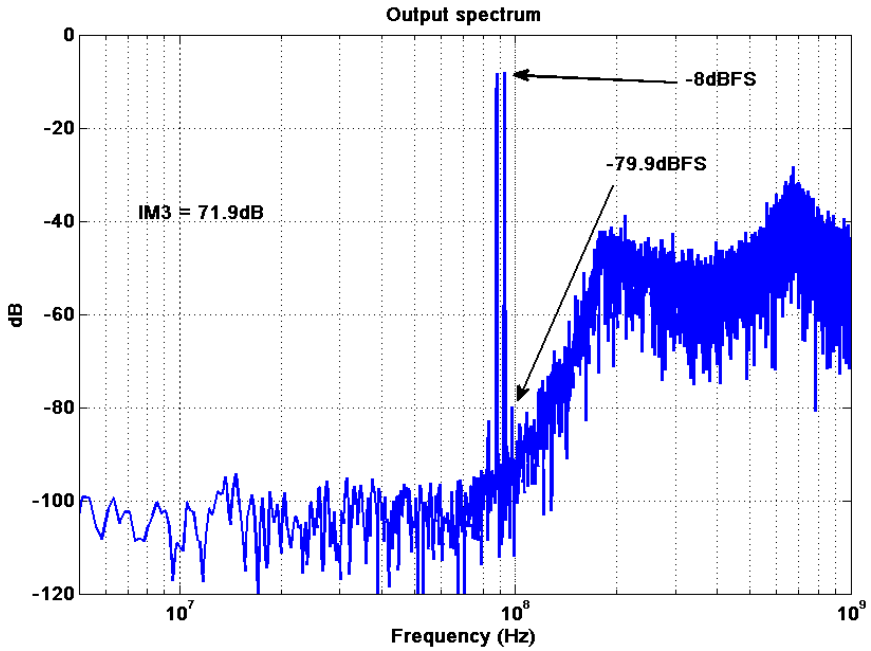


Figure 49 Two-tone simulation at complete system level

The performance achieved by the designed system is summarized in Table 15.

Table 15 System performance

	Specification	**Result**
Bandwidth	100MHz	100MHz
SNDR	68dB	69.6dB
IM3	70dB	71.9dB

7. CONCLUSIONS

In this thesis work, a continuous-time sigma-delta modulator capable of digitizing signals over a bandwidth of 100MHz with a resolution of 11bits was presented. The various design variables available at the system level for optimization were highlighted and the $\Delta\sum$ toolbox from MATLAB was used to perform the noise transfer computation. A novel loop filter composed of passive LC elements was proposed and the filter components designed based on the noise transfer function. Non-idealities of the various components in the modulator were then built into the ideal models to obtain specifications for the various building blocks. The problem of excess loop delay in continuous time sigma-delta modulators was highlighted and various compensation techniques available in literature highlighted. A method of compensating loop delay in the proposed loop filter was shown.

7.1 Future work

A couple of approaches to improve on the work presented in this thesis are noted below.

The poles in the proposed loop filter are realized using inductors and capacitors. High frequency poles can be more readily realized using these components in an area efficient manner. Hence, they are suitable for wider bandwidth modulators with poles of the loop filter spread across the bandwidth of interest. This can be further explored.

In the current work, the resolution of the modulator is limited primarily by the jitter of the clock signal in the feedback path. Approaches to mitigate the effect of jitter in the modulator need to be explored.

REFERENCES

[1] A. Olmos, T. Miyashita, M. Nihei, E. Charry, and Y. Watanabe, "A 5 GHz continuous time sigma-delta modulator implemented in 0.4µm InGaP/InGaAs HEMT technology," in *Proc. IEEE Int. Symp. Circuits and Systems (ISCAS 1998)*, vol. 1, Jun. 1998, pp. 575–578.

[2] S. Jaganathan, S. Krishnan, D. Mensa, T. Mathew, Y. Betser, W. Yun, D. Scott, R. Urteaga, and M. Rodwell, "An 18-GHz continuous-time ΣΔ analog-digital converter implemented in InP-transferred substrate HBT technology," *IEEE Journal of Solid-State Circuits*, vol. 36, no. 9, pp. 1343–1350, Sep. 2001.

[3] S. Krishnan, D. Scott, Z. Griffith, M. Urteaga, Y.Wei, N. Parthasarathy, and M. Rodwell, "An 8-GHz continuous-time delta-sigma analog-digital converter in an InP-based HBT technology," *IEEE Trans. Microw. Theory Tech.*, vol. 51, no. 12, pp. 2555–2561, Dec. 2003.

[4] X. Li, W. M. L. Kuo, Y. Lu, and J. D. Cressler, "A 20 GS/sec analog-to-sigital sigma-delta modulator in SiGe HBT technology," in *Proc. Custom Integrated Circuits Conf. (CICC)*, Sept 2006, pp. 221–224.

[5] A. Hart and S.P. Voinigescu, "1 GHz bandwidth low-pass ADC with 20-50 GHz adjustable sampling rate," *IEEE Journal of Solid State Circuits*, vol.44, no.5, pp.1401-1414. May 2009.

[6] G. Mitteregger, C. Ebner, S. Mechnig, T. Blon, C. Holuigue, and E. Romani, "A 20 mW 640-MHz CMOS continuous-time sigma-delta ADC with 20-MHz signal bandwidth, 80-dB dynamic range, and 12-bit ENOB," *IEEE Journal of Solid-State Circuits*, vol. 41, no. 12, pp. 2641–2649, Dec. 2006.

[7] X. Chen, Y.Wang, Y. Fujimoto, P. Lore, Y. Kanazawa, J. Steensgaard, and G. C. Temes, "A 18 mW CT ΔΣ modulator with 25 MHz bandwidth for next generation wireless applications," in *Proc. IEEE Custom Integrated Circuits Conf. (CICC)*, Sep. 2007, pp. 73–76.

[8] P. Malla, H. Lakdawala, K. Kornegay, and K. Soumyanath, "A 28mW Spectrum-Sensing Reconfigurable 20MHz 72dB-SNR 70dBSNDR DT ΔΣ ADC for

802.11n/WiMAX Receivers," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2008, pp. 496-497.

[9] V. Dhanasekaran, M. Gambhir, M.M. Elsayed, E. Sánchez-Sinencio, J. Silva-Martinez, C. Mishra, L. Chen, and E. Pankratz, "A 20MHz BW 68dB DR CT ΔΣ ADC Based on a Multi-Bit Time-Domain Quantizer and Feedback Element," in *IEEE Int. Solid-State Circuits Conf. (ISSCC)*, Feb. 2009, pp. 174-175.

[10] M. Park and M.H. Perrott, "A 78 dB SNDR 87 mW 20 MHz Bandwidth Continuous-Time Delta-Sigma ADC With VCO-Based Integrator and Quantizer Implemented in 0.13 um CMOS," *IEEE Journal of Solid-State Circuits*, vol. 44, no. 12, pp. 3344-3358, Dec. 2009.

[11] ELEN 610: Data Converters, Department of Electrical & Computer Engineering, Texas A&M University, http://www.ece.tamu.edu/~hoyos/

[12] X. Chen, "A wideband low-power continuous-time delta-sigma modulator for next generation wireless applications," Ph.D. Thesis, Oregon State University, Corvallis, Oregon, Mar. 2007.

[13] R. Schreier, ΔΣ Toolbox. [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/loadFile.do?objectId=19&objectType=file, accessed on Apr 1, 2008.

[14] K. Francken, P. Vancorenland, and G. Gielen, "DAISY: A Simulation- Based High-Level Synthesis Tool for ΔΣ Modulators," in *Proc. IEEE Int. Conf. Computer-Aided Design (ICCAD)*, Nov 2000, pp. 188-192.

[15] L. Breems and J.H. Huising, *Continuous-Time Sigma-Delta Modulation for A/D Conversion in Radio Receivers*, Boston, MA: Kluwer, 2001.

[16] S. Paton, A. Di Giandomenico, L. Hernandez, A. Wiesbauer, T. Pötscher, and M. Clara, "A 70mW 300MHz CMOS continuous-time ΣΔ ADC with 15MHz bandwidth and 11 bits of resolution," *IEEE Journal of Solid-State Circuits*, vol. 39, no. 7, pp 1056-1063, Jul 2004.

[17] R. Schreier and G.C. Temes, *Understanding Delta-Sigma Data Converters*, Piscataway NJ: IEEE Press, 2005.

[18] B.K. Thandri, "Design of RF/IF analog to digital converters for software radio communication receivers," Ph.D. Thesis, Texas A&M University, College Station, May 2006.

[19] M. Ortmanns, F. Gerfers, and Y. Manoli, "Clock Jitter Insensitive Continuous-Time ∑Δ Modulators," in *Proc. IEEE Int. Conf. Electronics, Circuits, and Systems*, vol. 2, Sept. 2001, pp. 1049-1052.

[20] B. Zhang, "Delta-sigma modulators employing continuous-time circuits and mismatch-shaped DACs," Ph.D. Thesis, Oregon State University, Apr 1996.

[21] R. Schreier, "An empirical study of high-order single-bit delta-sigma modulators," *IEEE Trans. Circuits & Syst. II*, vol. 40, pp. 461–466, Aug. 1993.

[22] R. Sobot, S. Stapleton, and M. Syrzycki, "Behavioral Modeling of Continuous Time Delta-Sigma Modulators," in *Proc International Workshop on Behavioral Modeling and Simulation (BMAS)*, Oct. 2003, pp.88-91.

[23] R.T. Baird and T.S. Fiez, "Linearity enhancement of multibit ∑Δ A/D and D/A converters using data weighted averaging," *IEEE Trans. Circuits & Syst. II*, vol. 42, pp.753-762, Dec 1995.

[24] F. Chen and B. H. Leung, "A High Resolution Multibit Sigma-Delta Modulator with Individual Level Averaging," *IEEE Journal of Solid-State Circuits*, vol. 30, no. 4, pp. 453-460, Apr. 1995.

[25] S. Yan and E. Sánchez-Sinencio, "A Continuous-Time ∑ΔModulator With 88-dB Dynamic Range and 1.1-MHz Signal Bandwidth," *IEEE Journal of Solid-State Circuits*, vol. 39, no. 1, pp 75-86, Jan 2004.

[26] L. Hernandez, A. Wiesbauer, S. Paton, and A. Di Giandomenico, "Modelling and optimization of low pass continuous-time sigma-delta modulators for clock jitter noise reduction," in *Proc. Int. Symp. Circ. Syst.*, May 2004, pp 1072-1075.

[27] J. A. Cherry and W. M. Snelgrove, "Excess Loop Delay in Continuous-Time Delta-Sigma Modulator," *IEEE Trans. Circuits & Systems II*, vol. 46, pp. 376-389, Apr. 1999.

[28] R. Schreier, N. Abaskharoun, H. Shibata, D. Paterson, S. Rose, I. Mehr, and Q. Luu, "A 375-mW Quadrature Bandpass $\Delta\sum$ ADC with 8.5-MHz BW and 90-dB DR at 44 MHz," *IEEE Journal of Solid-State Circuits*, vol. 41, no. 12, pp 2632-2640, Dec 2006.

[29] M. Keller, A. Buhmann, J. Sauerbrey, M. Ortmanns, and Y. Manoli, "A Comparative Study on Excess-Loop-Delay Compensation Techniques for Continuous-Time Sigma–Delta Modulators," *IEEE Transactions of Circuits and Systems I*, vol. 55, no. 11, pp. 3480-3487, Dec. 2008.

[30] A. Emami-Neyestanak, A. Varzaghani, J. F. Bulzacchelli, A. Rylyakov, C.-K. K. Yang, and D. J. Friedman, "A 6.0-mW 10.0-Gb/s receiver with switched-capacitor summation DFE," *IEEE Journal of Solid-State Circuits*, vol. 42, no. 4, pp. 889–896, Apr. 2007.

APPENDIX A

VERILOG-A CODES

A.1 Verilog-A code for quantizer

```
// VerilogA for Proj, 9level_quant_norm, veriloga

`include "constants.vams"
`include "disciplines.vams"

module Ninelevel_quant_norm_diff(inp,inm,clk,d7,d6,d5,d4,d3,d2,d1,d0);

input inp,inm,clk;
output d7,d6,d5,d4,d3,d2,d1,d0;
electrical inp,inm,clk,d7,d6,d5,d4,d3,d2,d1,d0;

real dp1,dp2,dp3,dp4,dn1,dn2,dn3,dn4;
real signal;
parameter real Delay = 0 from [0:inf);
parameter real Fullscale = 0.4 from (0:1];

analog begin
// Initiate output to mid-scale value
@(initial_step)
begin
        dp1 = 1;
        dp2 = 1;
        dp3 = 1;
        dp4 = 1;
        dn1 = 0;
        dn2 = 0;
        dn3 = 0;
        dn4 = 0;
```

```
end

// Make decision on rising edge of clk
@(cross(V(clk),+1))
begin
        signal = V(inp) - V(inm);
        if (signal>(7.0*Fullscale/8))
                begin
                        dp1 = 1;
                        dp2 = 1;
                        dp3 = 1;
                        dp4 = 1;
                        dn1 = 1;
                        dn2 = 1;
                        dn3 = 1;
                        dn4 = 1;
                end
        else if ((signal<=(7.0*Fullscale/8)) && (signal>(5.0*Fullscale/8)))
                begin
                        dp1 = 0;
                        dp2 = 1;
                        dp3 = 1;
                        dp4 = 1;
                        dn1 = 1;
                        dn2 = 1;
                        dn3 = 1;
                        dn4 = 1;
                end
        else if ((signal<=(5.0*Fullscale/8)) && (signal>(3.0*Fullscale/8)))
                begin
```

```
                dp1 = 0;
                dp2 = 0;
                dp3 = 1;
                dp4 = 1;
                dn1 = 1;
                dn2 = 1;
                dn3 = 1;
                dn4 = 1;
        end
else if ((signal<=(3.0*Fullscale/8)) && (signal>(1.0*Fullscale/8)))
        begin
                dp1 = 0;
                dp2 = 0;
                dp3 = 0;
                dp4 = 1;
                dn1 = 1;
                dn2 = 1;
                dn3 = 1;
                dn4 = 1;
        end
else if ((signal<=(1.0*Fullscale/8)) && (signal> (-1.0*Fullscale/8)))
        begin
                dp1 = 0;
                dp2 = 0;
                dp3 = 0;
                dp4 = 0;
                dn1 = 1;
                dn2 = 1;
                dn3 = 1;
```

```
                dn4 = 1;
        end
else if ((signal<= (-1.0*Fullscale/8)) && (signal> (-3.0*Fullscale/8)))
        begin
                dp1 = 0;
                dp2 = 0;
                dp3 = 0;
                dp4 = 0;
                dn1 = 0;
                dn2 = 1;
                dn3 = 1;
                dn4 = 1;
        end
else if ((signal<= (-3.0*Fullscale/8)) && (signal> (-5.0*Fullscale/8)))
        begin
                dp1 = 0;
                dp2 = 0;
                dp3 = 0;
                dp4 = 0;
                dn1 = 0;
                dn2 = 0;
                dn3 = 1;
                dn4 = 1;
        end
else if ((signal<= (-5.0*Fullscale/8)) && (signal> (-7.0*Fullscale/8)))
        begin
                dp1 = 0;
                dp2 = 0;
                dp3 = 0;
```

```
                dp4 = 0;
                dn1 = 0;
                dn2 = 0;
                dn3 = 0;
                dn4 = 1;
            end
        else if (signal<= (-7.0*Fullscale/8))
            begin
                dp1 = 0;
                dp2 = 0;
                dp3 = 0;
                dp4 = 0;
                dn1 = 0;
                dn2 = 0;
                dn3 = 0;
                dn4 = 0;
            end
end

// Assign values to output nodes
V(d7) <+ transition(dp1,Delay,5p,5p);
V(d6) <+ transition(dp2,Delay,5p,5p);
V(d5) <+ transition(dp3,Delay,5p,5p);
V(d4) <+ transition(dp4,Delay,5p,5p);
V(d3) <+ transition(dn1,Delay,5p,5p);
V(d2) <+ transition(dn2,Delay,5p,5p);
V(d1) <+ transition(dn3,Delay,5p,5p);
V(d0) <+ transition(dn4,Delay,5p,5p);
end
endmodule
```

A.2 Verilog-A code for DAC driver

```
// VerilogA for Proj, DAC_3bit, veriloga
`include "constants.vams"
`include "disciplines.vams"

module
DAC_3bit_diff(cp1,cp2,cp3,cp4,cn1,cn2,cn3,cn4,clk,p1,p2,p3,p4,n1,n2,n3,n4,fastp,fastn
);
input cp1,cp2,cp3,cp4,cn1,cn2,cn3,cn4,clk;
output p1,p2,p3,p4,n1,n2,n3,n4,fastp,fastn;
electrical cp1,cp2,cp3,cp4,cn1,cn2,cn3,cn4,clk,p1,p2,p3,p4,n1,n2,n3,n4,fastp,fastn;

parameter real Delay = 0 from [0:inf);
parameter real DACHi = 1 from (0:inf);
parameter real DACLo = -1 from (-inf:0];
real dp1,dp2,dp3,dp4,dn1,dn2,dn3,dn4,fp,fn;

analog begin
@(initial_step)
begin
        dp1 = 0;
        dp2 = 0;
        dp3 = 0;
        dp4 = 0;
        dn1 = 1;
        dn2 = 1;
        dn3 = 1;
        dn4 = 1;
end
```

```
// No delay between DAC and ADC
@(cross(V(clk),1))
begin
        dp1 = V(cp1);
        dp2 = V(cp2);
        dp3 = V(cp3);
        dp4 = V(cp4);
        dn1 = V(cn1);
        dn2 = V(cn2);
        dn3 = V(cn3);
        dn4 = V(cn4);
        fp = (V(cp1) + V(cn1) + V(cp2) + V(cp3) + V(cp4) + V(cn2) + V(cn3) +
V(cn4))*2.0;    // Generate fast path signals
        fn = (8 - (V(cp1) + V(cn1) + V(cp2) + V(cp3) + V(cp4) + V(cn2) + V(cn3) +
V(cn4)))*2.0;  // Generate fast path signals
end

V(p1) <+ transition(dp1,Delay,5p,5p);
V(p2) <+ transition(dp2,Delay,5p,5p);
V(p3) <+ transition(dp3,Delay,5p,5p);
V(p4) <+ transition(dp4,Delay,5p,5p);
V(n1) <+ transition(dn1,Delay,5p,5p);
V(n2) <+ transition(dn2,Delay,5p,5p);
V(n3) <+ transition(dn3,Delay,5p,5p);
V(n4) <+ transition(dn4,Delay,5p,5p);
V(fastp) <+ transition(fp,Delay,5p,5p);
V(fastn) <+ transition(fn,Delay,5p,5p);
end
endmodule
```

APPENDIX B

MATLAB CODE

B.1 MATLAB code to synthesize loop filter

```
% Variables
OSR = 10;
order = 5;
nlev = 8;

% Highpass filter for NTF
Rstop = 90;
[b1,a1] = cheby2(order,Rstop,1/(1.5*OSR),'high');

% Ideally place zeros spread across signal bandwidth.
% To keep values of passive components reasonable, choosing
% high frequency zeros here.

b1 = conv([1 -.9],conv([1 -2*0.88*cos(2*pi*80e6/2e9) 0.88^2],...
   [1 -2*0.92*cos(2*pi*80e6/2e9) 0.92^2]));

ntf1 = filt(b1,a1,1);
ntf1 = zpk(ntf1);

% Plot NTF

ntf_mag = bode(ntf1,pi);
bodemag(ntf1)
grid on;
s = sprintf('Max NTF gain = %4.3f \n',ntf_mag);
text(0.15,12,s);

%%%%% To obtain SNR at various amplitudes %%%%%

N = 8192; % No of points in the FFT
fB = ceil(N/(2*OSR)); % Signal bandwidth
f = 200; % Input tone

amp1 = [-90:5:-15 -12 -10:0];

npoints = length(amp1);
snr1 = zeros(1,npoints);
```

```
maxsnr = snr1(1);
inp_maxsnr = amp1(1);

for i=1:npoints
   ampl = 10^(amp1(i)/20);
   u = ampl*(nlev-1)*sin(2*pi*f/N*[0:N-1]);
   v = simulateDSM(u,ntf1,nlev);
   spec = fft(v.*ds_hann(N))/(N/4);
   snr1(i) = calculateSNR(spec(1:fB),f);
   if snr1(i)>maxsnr
      maxsnr = snr1(i);
      inp_maxsnr = amp1(i);
   end
end

figure
plot(amp1,snr1,'b-d')
grid on;
s = sprintf('Max SNR = %4.1fdB @ %5.1fdB input\n',maxsnr,inp_maxsnr);
text(-80,60,s)

%%%%% Discrete to continuous transformation %%%%%

ntf1 = filt(b1,a1,1/2e9);
ntf1 = zpk(ntf1);

%%%%% Loop filter computation from NTF %%%%%

L1 = filt(1,1,1/2e9) - inv(ntf1);
[b1,a1]=tfdata(-L1,'v');

%%%%% Compensate for 1 cycle loop delay %%%%%

for i=1:5
b2(i)=b1(i+1);
end

b2(6)=0;
L2 = tf(b2,a1,1/2e9);

d2c(L2); % Continuous-time loop filter
```

VITA

Vijayaramalingam Periasamy received the B.Tech degree in Electronics and Communication Engineering from National Institute of Technology, Tiruchirappalli, India in 2004. He was with Analog Devices India Pvt Ltd, Bangalore, India between 2004 and 2007 working on successive approximation analog-to-digital converters. He interned with Broadcom Corporation, Irvine during Fall 2009 in the RF Tuner group. He received the M.S degree in Electrical Engineering from the Analog & Mixed Signal Center, Electrical & Computer Engineering department, Texas A&M University, College Station in May 2010. His main research interests are in the field of data converters. He can reached through the Department of Electrical Engineering, 3128 TAMU, College Station, TX 77843.