IMPLEMENTATION AND PERFORMANCE EVALUATION OF SIGMA-DELTA MODULATORS

by
Muharrem Orkun Sağlamdemir
B.S., Electrical and Electronics Engineering, Boğaziçi University, 2005

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Electrical and Electronics Engineering
Boğaziçi University
2007

To Beloved Ones...

ACKNOWLEDGEMENTS

Firstly, I want to express my thanks to my thesis supervisor Prof. Dr. Günhan Dündar for his help and support throughout the thesis and for his advice on graduate study.

I would like to thank Selçuk Talay for his help during thesis.

I would like to thank to my research partner and so called ”soul brother” Ömer Yetik for everything up to now and for everything that will be in the future.

I would like to thank to Ozan Yalçın for being on my life from the start of my Boğaziçi Life, his support, friendship, presence in my future.

I would like to express my gratitude to Yücel Altuğ and Nafiz Polat Ayerden for their support during all my graduate study. I would like to see them by my side in my future.

I would like to express my gratitudes and thanks to Gönenç Seçil Tarakcıoğlu, Işıl Burcu Barla and Selen Özgür for their support and coming to my presentation.

Last but not least, I would like to express my special thanks to my family for their endless support and being by my side in any case. I could do nothing without them.

ABSTRACT

IMPLEMENTATION AND PERFORMANCE EVALUATION OF SIGMA-DELTA MODULATORS

Sigma–Delta A/D converters have recently become popular since they achieve high resolution by means of oversampling without the need of high accuracy matching of elements and complex analog circuitry. They even reach higher performance utilizing feedback and feedforward paths. However finding the paths to be used and their values in an optimal way is an important concern, since there exist a lot of paths.

In search of optimality, this work starts with the realization of the topologies found by an automation tool. Then, some theoretical approach about the performance of the architectures are developed. Following this, using either implementation data or theoretical approach it searches for finding closed–form equations relating performance with path gains. Defining SNR sensitivity, area and power consumption as performance metrics, it reaches some relations which are functions of path gains.

ÖZET

SİGMA-DELTA KİPLEYİCİLERİN GERÇEKLENMESİ ve BAŞARIMLARININ DEĞERLENDİRİLMESİ

Sigma Delta analog sayısal çeviriciler, aşırı örnekleme kullanmaları sayesinde karmaşık analog devrelere ve elemanlarının eşlenmesinde yüksek kesinliğe ihtiyaç olmaksızın yüksek çözünürlüğe ulaşmalarıyla yakın zamanda geniş kullanım alanları bulmuşlardır. Bu çeviriciler, ileri besleme ve geri besleme yolları kullanarak daha yüksek başarımlara ulaşırlar. Fakat çok fazla yol olması sebebiyle kullanılacak yolların tespiti ve değerlerinin en uygun biçimde belirlenmesi dikkat edilmesi gereken bir konudur.

En uygun durumun araştırılması yolunda, bu çalışma otomatik bir yazılım kullanılarak yollarının değerleri tespit edilmiş yapıların gerçeklenmesiyle başlıyor. Daha sonra bu yapıların başarımları hakkında teorik bir yaklaşım getiriliyor. Bunu takiben gerek yapıların gerçeklenmesiyle ilgili bilgileri gerek teorik yaklaşımı kullanarak bu çalışma başarımla yolların değerlerini ilişkilendiren kapalı denklemler bulmaya yöneliyor. Son olarak sinyal gürültü oranına duyarlılığı, alanı ve güç tüketimini başarım değişkenleri olarak tanımlayarak bunlarla yolların değerleri arasında ilişkilere ulaşılıyor.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS/ABBREVIATIONS

C_f	Feedback Capacitor
C_s	Sampling Capacitor
f_d	Nyquist Frequency
f_S	Sampling Frequency
M	Oversampling Ratio
V_{cm}	Common Mode Voltage
V_{in}	Input Voltage
$\sigma^2(e)$	Noise Power
$\Sigma\Delta$	Sigma-Delta
A/D	Analog to Digital
CAD	Computer Aided Design
CMOS	Complementary Metal–Oxide Semiconductor
DAC	Digital to Analog Converter
DC	Direct Current
DSP	Digital Signal Processor
IC	Integrated Circuit
MOS	Metal Oxide Semiconductor
MOSFET	Metal Oxide Semiconductor Field Effective Transistor
NTF	Noise Transfer Function
PSD	Power Spectral Density
RC	Resistor Capacitor
SC	Switched Capacitor
SD	Sigma Delta
SNR	Signal-to-Noise Ratio
STF	Signal Transfer Function
VLSI	Very Large Scale Integration

1. INTRODUCTION

1.1. Background

In nature, every human defined quantity is analog, meaning that its value can take any value at any time. Since this is the case, in it is impossible to store and transmit these signals without error. In that sense, digital representation of signals is logical since those processes are almost error free in digital form. Also, considering processing of signals for special purposes, digital representation is far more powerful with respect to its analog counterpart [1]. All these facts make analog to digital converters an important part of systems dealing with analog world that need the advantages of digital.

According to Nyquist Theorem, a signal has to be sampled at least twice the highest frequency component included in the signal in order to be able to recontruct a signal from the samples taken from it. That frequency is known as Nyquist frequency. There are two types of A/D converters according to the sampling frequency used in the conversion process: Nyquist Rate converters and Oversampling converters.

Nyquist Rate converters sample the signal to be converted at Nyquist frequency. Generally used architectures are: DAC based, Subranging, Integrating Type, Pipelined, Charge Redistribution, Succesive Approximation, Flash. These converters utilize an Anti–Aliasing Filter which filters out the frequencies greater than half the sampling frequency. Following steps are Sampling and Quantization as in Figure 1.1. Most of the Nyquist Rate converters require accurate matching of used elements resulting in complex analog circuitry [2]. For the case of Flash converter number of comparators

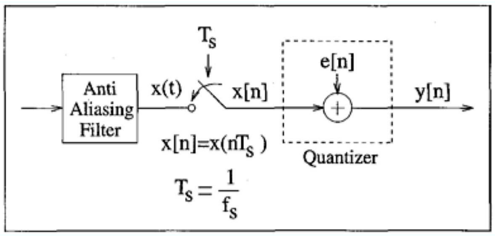


Figure 1.1. Block diagram of a Nyquist rate converter [1].

increases exponentially as a function of the number of bits. An additional bit of resolution doubles the matching and accuracy requirements. Therefore, most Nyquist–rate converters are not capable of achieving 12 bit resolution [3]. High accuracy can be obtained by usage of expensive trimming procedures or complex digital self–calibration algorithms [4]. As a result, in CMOS VLSI processes it is a rare case to achieve 10 to 12 bits of matching without trimming or paying special attention to the fabrication processes[5]. In brief, Nyquist Rate converters need advanced techniques and expensive processes to achieve high resolution, which make them impractical.

The aim of achieving high resolution has made Oversampling converters an attractive research area in last three decades. Oversampling converters sample the signal at a rate much larger than the Nyquist rate. They achieve high resolution at the expense of relatively lower bandwidth. In other words, oversampling converters trade off high resolution for bandwidth which can be seen from Figure 1.2. In that sense, oversampling converters are suitable for usage in audio applications. However instrumentation and communication have become application areas after researches in the last decades [6]. Oversampling converters somewhat loosen the specifications of the analog circuitry needed, but require fast and complex digital processing stages [7]. Antialiasing filter, which has tough requirements to realize in the Nyquist rate converter case, becomes less problematic to deal with since oversampling is utilized. In addition to this, oversampling converters use digital signal processing since VLSI is better

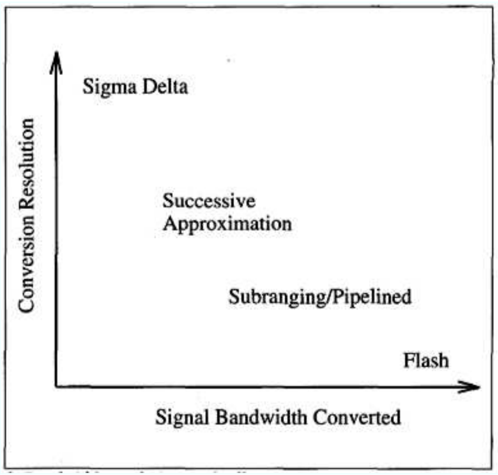


Figure 1.2. Resolution Bandwidth trade off graph of several converters [1].

suited for providing fast digital circuits than for providing precise analog circuits. A general oversampling converter is seen in Figure 1.3. It comprises analog and digital part. Analog part includes the Antialiasing Filter which filters out the frequency components higher than half the sampling frequency. Since oversampling converters sample the signal at a high frequency compared to the Nyquist case, cut–off frequency of this filter is much larger than it is for the Nyquist converter. A sampler and quantizer follows the filter as parts of analog processing. As explained in [1, 6, 7], quantizer generates a white noise uniformly distributed in the frequency band, determined by sampling frequency. For a quantizer in the Nyquist and Oversampling cases, noise power is the same, however since the sampling frequency is much higher for the latter, same power is distributed in larger spectrum. This is illustrated in Figure 1.4. At the end of the conversion process, the important frequency band will be the signal band. So, for the oversampling case much less power will be at the band of interest since noise is distributed in a larger frequency spectrum.

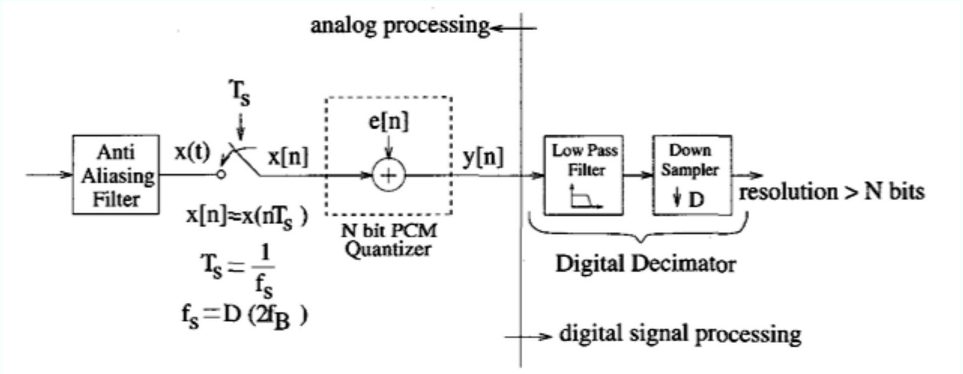


Figure 1.3. Architecture of general oversampling converter [1].

Different from the Nyquist rate converters, oversampling converters include digital processing part as it can be seen in Figure 1.3. This part filters out high frequency components and downsamples the signal to the Nyquist rate, since signal was sampled at a rate much higher than Nyquist rate. In other words, high frequency components in Figure 1.4 are low pass filtered and word length is increased at the digital part.

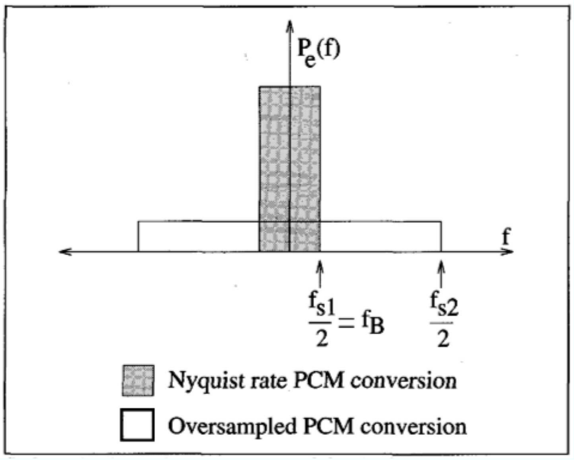


Figure 1.4. Quantization Noise Power Spectral Density for Nyquist and Oversampling converters [1].

1.2. Problem Formulation and Outline of Thesis

As an improvement to the general quantizer used in an oversampling converter, noise can be shaped such a way that even less noise exists at the band of interest. That need gave rise to the proposition of Sigma–Delta converters by Inose et al. in [8]. The general block diagram of a first order sigma–delta converter is shown in Figure 1.5. Input is fed to the quantizer via an integrator and the quantized output feeds back to subtract from the input signal. This feedback forces the average value of the quantized signal to track the average input. Any continual difference between them accumulates in the integrator and eventually corrects itself [7]. In general, 1–bit quantizers are used in this architecture. This modulator shapes the noise such that it is sent to higher frequencies, which is later low pass filtered by the digital filter. Figure 1.6 illustrates this fact by comparing Nyquist, General Oversampling and SD converter.

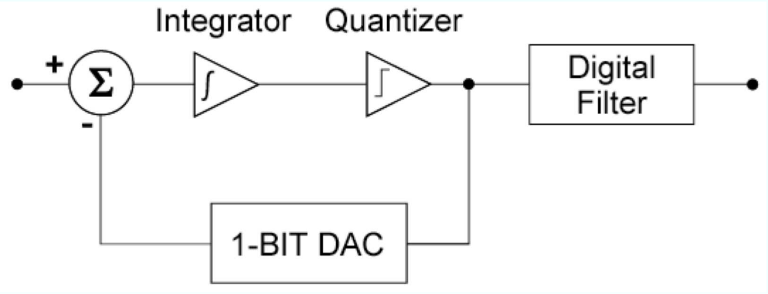


Figure 1.5. First Order Sigma–Delta converter Architecture Block Diagram [9].

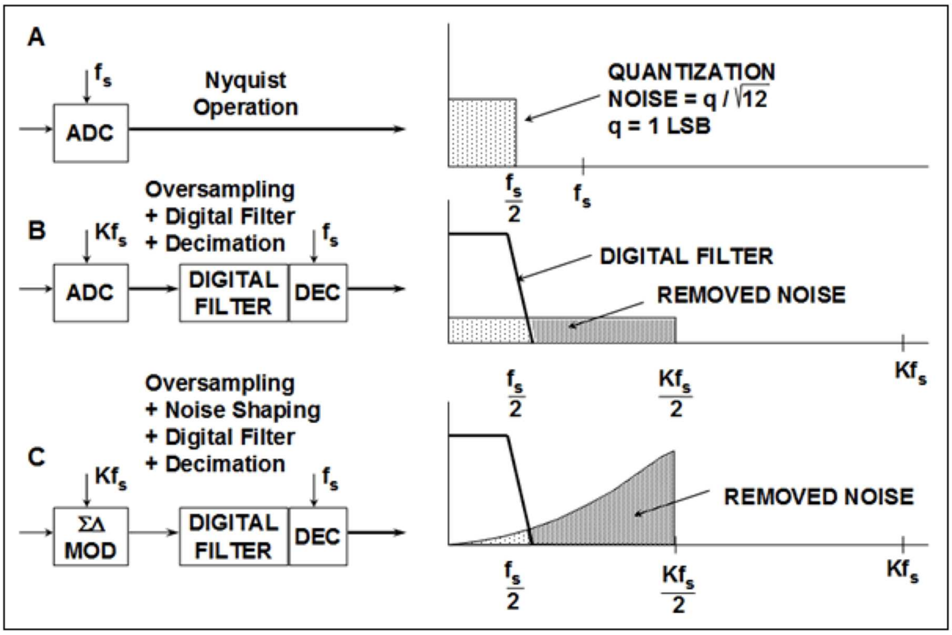


Figure 1.6. Comparison of Nyquist, Oversampling and Sigma–Delta converters [10].

By shaping noise out of the signal band high resolution is attained. However, this poses the penalty of speed since the hardware has to work at a higher frequency, which results in complex digital hardware. In the case of SD modulator architectures, there are different topologies trading off resolution, modulator stability, bandwidth and circuit complexity [1]. Most prominent ones are higher order, multi bit and multistage (*cascaded*) architectures. Details about them will be given in the SD Theory section.

The integrator shown in Figure 1.5 is also called loop filter. In higher order sigma delta structures, which will be the topic of concern in this thesis, more stages of integrators are used in cascade before quantizer. By this way, noise is supressed more in the signal band without disturbing the signal. In higher order modulators, multiples of internal signals are either added to or subtracted from the other signals. As an example, DAC output can be fed back to the other integrators, which were added in cascade after the first one. Even local feedback paths can be added to the architecture. In addition to feedback paths, feedforward paths from input or integrator outputs to integrator and quantizer inputs are seen in the higher order architectures. All these paths can have gains. For a second order SD topology, a completely connected architecture becomes as in Figure 1.7.

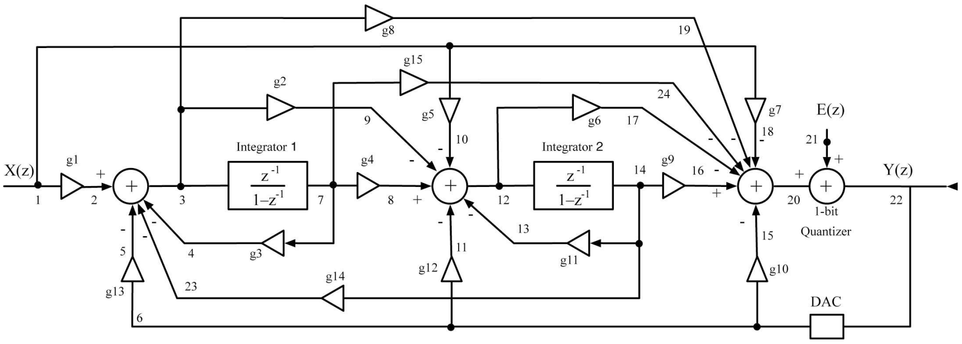


Figure 1.7. Second Order SD Modulator with all possible paths.

As it can be seen from Figure 1.7, the topology really becomes very complex by the inclusion of all possible paths even for the second order case. All those coefficients can be beneficial in a sense that they can be used to achieve a specified transfer function. But determination of needed ones and their values to achieve a specific task and how they will be implemented is the problem. In addition to this, distinguishing between numerous architectures implementing a transfer function is another problem. This thesis starts at the case of having all possible solutions of path gains in hand, which is achieved by the tool proposed in [11, 12].

The tool proposed in [11, 12] aims at finding topologies with the least number of paths in other words, minimization of hardware complexity. The tool can find various architectures with different combinations of feedforward and feedback coefficients. At that point, implementation of the architectures turns out to be an important concern. At first hand, implementability of the architecture is seen as a parameter for the applicability of the optimization. Starting from the path gains at hand, this thesis aims at implementing the found architectures. Although Second Order SD modulator will be the subject of concern, by finding general implementation rules, any high order SD modulator topology implementation will be possible. Then, from implementations of different architectures, some general deductions, which can lead the designer to choose an architecture, are made.

Also, implementation gives some criteria to compare different achitectures, in addition to the ease of implementation and hardware comlexity. As considerations to choose an architecture, sensitivity to coefficients, area and power consumption appear to be good choices [13]. As a result of this, to make performance evaluation and set criteria to compare different architectures, work is focused on power, area and sensitivity considerations. A theoretical approach on each of these considerations is followed, which is supported by the simulations of the implementations. After all, an approach of choosing an architecture for implementation is reached.

The content of thesis report is organized as follows: In Chapter 2, SD modulator theory is visited. Different architectures are discussed and main focus is placed on high order single bit SD modulator since it will be subject of concern. In Chapter 3, SD converter Design Automation Tool [11, 12] is explained, then the implementation of the blocks in SD Architecture is put into words. Following that in Chapter 4, implemented architectures are presented. Explanations, simulation results, data and figures are given at that chapter. Chapter 5 deals with performance issues in order to choose a topology: area, power consumption and sensitivity respectively. Finally report is summarized and further work is presented in Chapter 6.

2. SIGMA DELTA CONVERTERS

As explained in Section 1.1, oversampling converters became very popular in the last few decades because of their advantages compared to Nyquist rate converters. They improve the specifications on the selectivity of the antialiasing filter and sensitivity to the circuit imperfections. Furthermore, they use power of digital signal processing to relax the requirements on analog circuitry which makes them a good choice for implementation of A/D converters. As a further improvement to the oversampling converters, utilization of SD Modulation results in better quantization noise shaping which ends up with higher resolution, robust operation and relative insensitivity to non–ideal effects [6].

2.1. Basic ΣΔ A/D Converter Theory

A ΣΔ A/D comprises three parts as it is shown in Figure 2.1. First one is the Antialiasing filter which filters out the high frequency components that are greater than half the sampling frequency. As it was discussed in Chapter 1, implementation of this block is really simplified by usage of high sampling rate, even a passive first–order filter is sufficient.

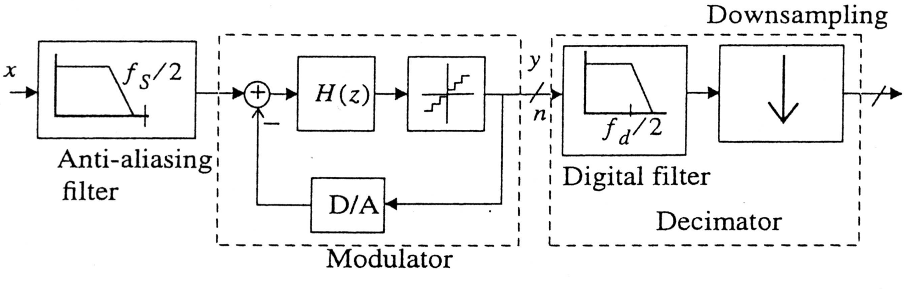


Figure 2.1. Block Diagram of a ΣΔ A/D converter [6].

Second block of the converter is the ΣΔ modulator part which is the main difference of ΣΔ A/D converters from other oversampling converters. Sampling and quantization of the signal is made in this block. Modulator includes loop filter, quantizer and DAC as its parts. Loop filter is the most important part that makes ΣΔ modulator special. By designing the loop filter in an appropriate manner, quantization noise is shaped such that most of its power lies out of signal band, which will be discussed further in Section 2.1.2. Output of the modulator is taken from the output of the quantizer meaning that output will be digitally coded. Since quantizer is generally one–bit, output is a sequence of one bit data at the sampling frequency.

Last block of the converter is the decimator. This block is formed of digital filter and downsampler. Spectral components out of the signal band are filtered by the digital filter, which is a low pass one. Decimation of the output of the modulator is done by the last stage of the decimator, which is the downsampler. When overall design is concerned, modulator block is the most problematic part since antialiasing filtering is eased by oversampling. In addition to this, decimator design is facilitated by highly structured design methodology of DSP blocks and help of CAD tools in hand [6].

2.1.1. Quantization Noise

No matter what type of A/D conversion is discussed, quantization seems to be a part of that conversion process. In quantization, a signal that can have any value is mapped to a finite number of levels. This process inevitably introduces distortion to the conversion process so that our primary objective in designing modulators becomes limiting this distortion [7]. Quantization can be formulated by a non–linear function –i being input and y being output– as follows:

$$y = g_q i + e. \tag{2.1}$$

In equation(2.1) g_q symbolizes the quantizer gain and e symbolizes the quantization error. Error as a function of the input is shown in Figure 2.2. If input is limited by the proper working range of the quantizer, which is the interval $[i_{min}, i_{max}]$, quantization error resides in the range $[-\Delta/2, \Delta/2]$, where Δ is the spacing betwwen consecutive levels of a quantizer. Giving inputs outside of the specified region results in a continuous increase in error, which is named *Overloading.*

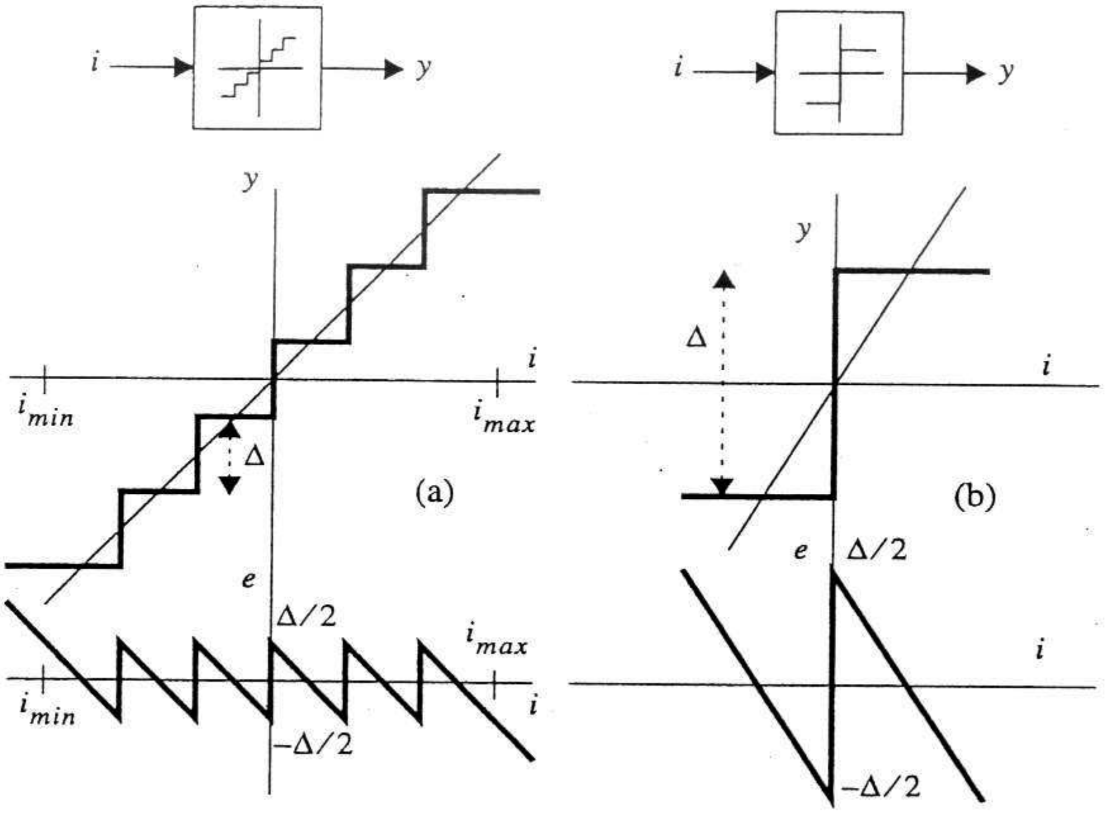


Figure 2.2. Transfer curves and quantization error of (a) a multi–bit quantizer (b) a single bit quantizer (comparator) [6].

Error in quantization is surely dependent on input level. However, it was shown in [14] that if input changes randomly at every sample without overloading and number of quantization levels is large, quantization error distributes uniformly in $[-\Delta/2, \Delta/2]$ with a constant power spectral density like that of white noise. Because of this, quantization error is commonly referred to as *quantization noise.* This quantization noise having a total power of $\sigma^2(e)$ distributes in $[-f_S/2, f_S/2]$ where f_S is the sampling frequency. Then, its power spectral density becomes

$$S_E(f) = \frac{\sigma^2(e)}{f_S} = \frac{1}{f_S}\left[\frac{1}{\Delta}\int_{-\Delta/2}^{\Delta/2} e^2 de\right] = \frac{\Delta^2}{12 f_S}. \tag{2.2}$$

In the case of Nyquist rate converters, sampling frequency is f_S which means that total noise power is $\Delta^2/12$. As discussed in previous sections, in oversampling converters, sampling frequency is much higher than Nyquist Frequency. So, noise power in the signal band of $[-f_d/2, f_d/2]$ becomes

$$P_Q = \int_{-f_d/2}^{f_d/2} S_E(f)df = \frac{\Delta^2}{12}\frac{f_d}{f_S} = \frac{\Delta^2}{12M}, \tag{2.3}$$

where f_d is the Nyquist frequency. M is *Oversampling Ratio (OSR)* in Equation (2.3), which is ratio of sampling frequency to Nyquist frequency. Above mentioned power, P_Q is named as *in–band power* since it is the power in the signal band.

As explained above, one of the assumptions leading to uniform distribution of quantization noise in [14] was that number of levels in the quantizer is large. Thus, for the case of single bit quantization, formulas do not exactly hold. But research on this case shows that white noise approach is approximately valid [15].

2.1.2. ΣΔ Modulator

Quantization block in an oversampling converter is efficiently realized by ΣΔ modulator. A general ΣΔ modulator comprises Loop Filter, $H(z)$, quantizer and Digital to Analog converter in the feedback path as in Figure 2.3. In the architecture, modulator output is subtracted from the input. This difference is then passed through the Loop filter before entering quantizer, which is generally one–bit. Filter transfer function is adjusted in a such way that it attenuates the quantization noise in signal band.

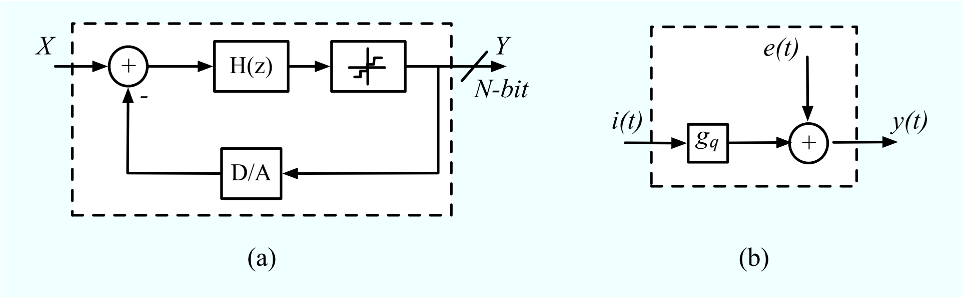


Figure 2.3. (a) Basic structure of the ΣΔ modulator (b) Quantizer model [6].

But at this point, some issues about quantization noise in ΣΔ modulator must be discussed. Low number of levels of the quantizer and determination of the quantization error by the input signal make white noise approximation controversial. However, it is shown in [16] that in the case of time–variant modulator input, previously given results continue to be applicable. So quantizer can be modeled like in Figure 2.3, where $e(t)$, quantization error, is uniformly distributed and its power spectral density is constant with its value in Equation 2.2.

As a result of the above discussion, the whole modulator can be modeled as having two inputs: $x(t)$ and $e(t)$ and one output, $y(t)$. Then, the resulting Z–domain representation of the system becomes

$$Y(z) = STF(z)X(z) + NTF(z)E(z), \tag{2.4}$$

where $X(z)$ and $E(z)$ are Z–domain representation of the input and quantization noise respectively; and $STF(z)$ and $NTF(z)$ are the respective transfer functions of the input and quantization noise [6]. Analysis and simple calculations give STF and NTF as follows:

$$STF(z) = \frac{H(z)}{1 + H(z)} \qquad \text{and} \qquad NTF(z) = \frac{1}{1 + H(z)}. \tag{2.5}$$

Since signal is at the low–frequency region, NTF is preferred to go 0 when $z \rightarrow 1$. These conditions evaluated with equations of NTF and STF impose the following condition on $H(z)$:

$$H(z) \rightarrow \infty \qquad \text{when} \qquad z \rightarrow 1 \tag{2.6}$$

Equation (2.6), which is the main condition on the implementation of $H(z)$, gives a discrete-time integrator with the following transfer function as the simplest way to implement $H(z)$:

$$H(z) = \frac{z^{-1}}{1 - z^{-1}}.$$

Use of discrete-time integrator as loop filter results in $NTF(z) = 1 - z^{-1}$ and $STF(z) = z^{-1}$ forming a *First–Order* $\Sigma\Delta$ *modulator*. There are different modulator architectures which are derived from the basic architecture in Figure 2.3. They will be discussed further in Section 2.2.

2.1.3. ΣΔ Performance Criteria

After introducing First–Order ΣΔ modulator, some performance criteria will be presented:

Signal–to–Noise Ratio(SNR) is the ratio of the output power at the frequency of a sinusoidal input to the in-band noise power which is formulated as

$$SNR(dB) = 10log_{10}\left(\frac{A^2/2}{P_Q}\right), \tag{2.7}$$

where A is the input amplitude. Increase with input amplitude continues until the quantizer overloads. When overloading occurs, a sharp drop is seen in SNR. Noise power is generally the quantization noise power. In addition to that, there are other non-idealities in the ΣΔ architecture which add to the in–band noise power. These non-idealities may include the clock-jitter, kT/C noise, operational amplifier noise, and integrator non-idealities such as the finite direct current(dc) gain (leakage), the switched capacitor mismatches, slew-rate limitation of the operational amplifier, DC offset of the comparator, etc. which is discussed further in [6]. In that case, non–ideality effects are added to noise to give a definition of signal to noise+distortion(S/N+D) ratio.

Dynamic Range(DR) is the ratio of the output power in the case of full–scale range amplitude input to the output power in the case of 0 dB SNR. Full– scale range is determined by the quantizer's full scale which is $\Delta/2$ for single–bit quantization.

Effective resolution(B) is the number of bits that can be obtained from a converter.

Its formula is given as

$$B(bit) = \frac{DR(dB) - 1.76}{6.02}.$$

Another formula that relates oversampling ratio and internal quantization to effective resolution (B) is as follows [6]:

$$B(bit) = \frac{1}{2}log_2\left[\frac{(2^b-1)^2(2L+1)M^{2L+1}}{\Pi^{2L}}\right].$$

2.2. ΣΔ Modulator Architectures

Different ΣΔ modulators exist, all of which mainly aim at reducing in–band quantization noise power. In addition to increasing oversampling ratio, there are two other strategies to achieve that:

- Order of the loop filter $H(z)$ is increased, which causes an increase in the order of NTF, thereby leading to more effective cancellation of noise.
- Increasing the resolution of the internal quantizer, which results in decrease in Δ, thereby causing reduction in power spectral density of noise.

In some cases multi–bit internal quantizers are used, which are called *Multi–Bit* ΣΔ *modulators*. Another topology utilizes cascading modulators of low–order and then using cancellation logic, which is named as *Cascade* or *Multi–Stage Modulator*. In this work, single loop single–bit modulators are dealt with, further details about other architectures can be found in [3, 5, 6, 7].

Single–loop single–bit modulators are also categorized into two according to the order of noise shaping, in other words, order of NTF. Modulators with less than third order noise shaping are called low-order modulators, whereas the others are called high-order modulators. Importance of order is better explained by Figure 2.4. Less power stays in band for higher order topologies but stability of the modulator becomes signal

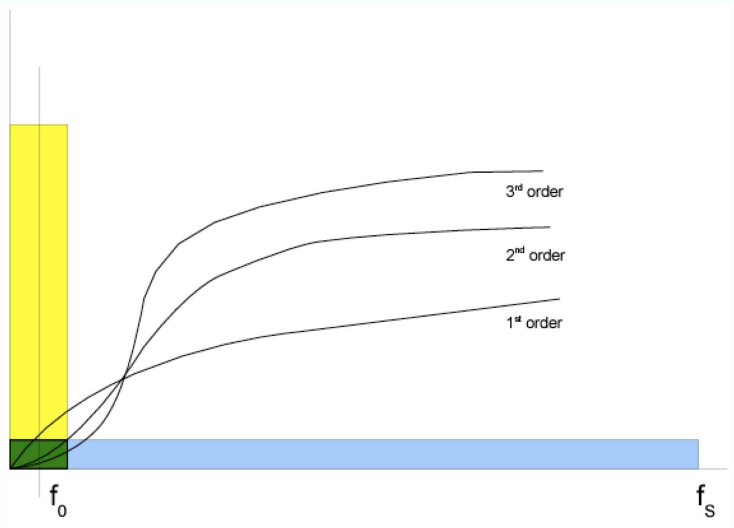


Figure 2.4. Noise shaping curves and noise spectrum in ΣΔ modulator [17].

dependent. In single–loop single-bit modulators, order is determined by the number of cascaded integrators. The block diagram of the second order converter utilizing a second order modulator is given in Figure 2.5. As it is seen in the figure, feedback is given to both integrators. As explained in Chapter 1.2, feedback and feedforward paths with gains are used in order to obtain better noise shaping in modulators. Paths for the case of second order modulator can be seen in Figure 1.7.

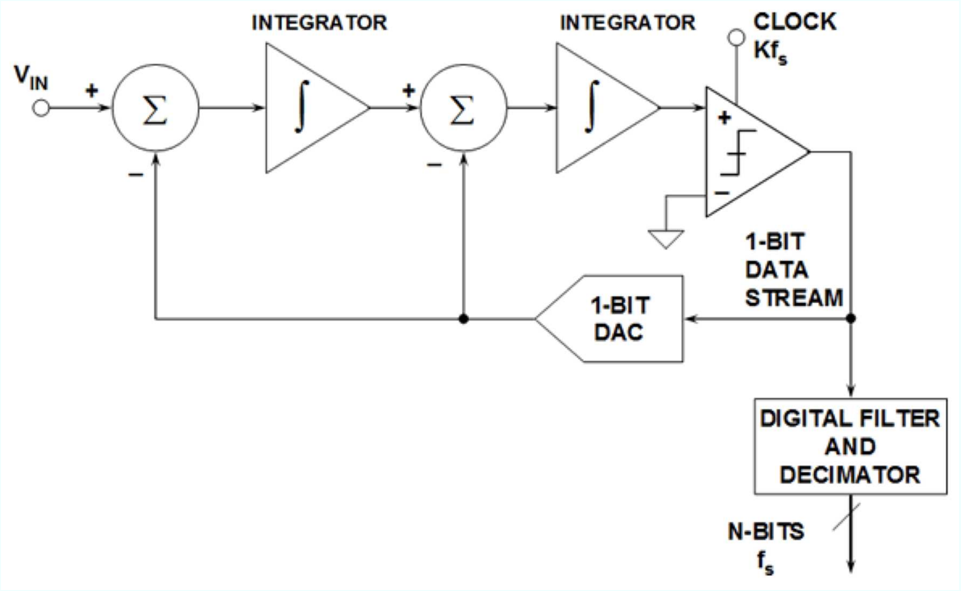
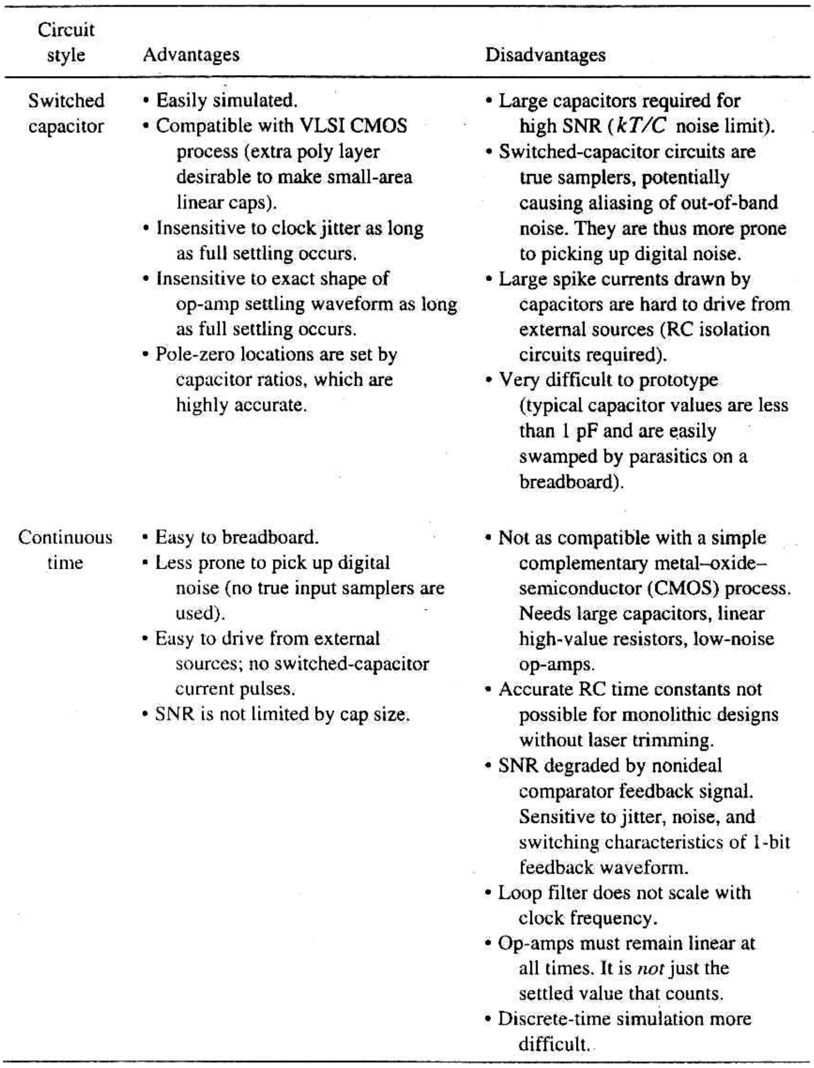


Figure 2.5. Second order ΣΔ A/D converter [10].

3. IMPLEMENTATION

In this work, second order $\Sigma\Delta$ modulators are implemented and then some metrics are discussed in order to make deductions to choose architectures from a bunch of alternatives. In the implementation phase, the first decision to be made is to choose Switched–Capacitor (SC) or a conventional active–RC (continuous–time) design to implement the circuitry. In general, most integrated circuit (IC) implementations of $\Sigma\Delta$ ADCs use SC circuits, whereas most system–level or hybrid implementations use active–RC circuits [7]. There are advantages and disadvantages associated with each method which is tabulated in Figure 3.1.



Circuit style	Advantages	Disadvantages
Switched capacitor	• Easily simulated. • Compatible with VLSI CMOS process (extra poly layer desirable to make small-area linear caps). • Insensitive to clock jitter as long as full settling occurs. • Insensitive to exact shape of op-amp settling waveform as long as full settling occurs. • Pole-zero locations are set by capacitor ratios, which are highly accurate.	• Large capacitors required for high SNR (kT/C noise limit). • Switched-capacitor circuits are true samplers, potentially causing aliasing of out-of-band noise. They are thus more prone to picking up digital noise. • Large spike currents drawn by capacitors are hard to drive from external sources (RC isolation circuits required). • Very difficult to prototype (typical capacitor values are less than 1 pF and are easily swamped by parasitics on a breadboard).
Continuous time	• Easy to breadboard. • Less prone to pick up digital noise (no true input samplers are used). • Easy to drive from external sources; no switched-capacitor current pulses. • SNR is not limited by cap size.	• Not as compatible with a simple complementary metal–oxide–semiconductor (CMOS) process. Needs large capacitors, linear high-value resistors, low-noise op-amps. • Accurate RC time constants not possible for monolithic designs without laser trimming. • SNR degraded by nonideal comparator feedback signal. Sensitive to jitter, noise, and switching characteristics of 1-bit feedback waveform. • Loop filter does not scale with clock frequency. • Op-amps must remain linear at all times. It is *not* just the settled value that counts. • Discrete-time simulation more difficult.

Figure 3.1. Comparison of SC and RC modulator Realizations [7].

Since SC method is compatible with VLSI, they are easily simulated and pole–zero locations are set by capacitor ratios etc. it will be the method of implementation. Therefore, switched capacitor theory and implementation of the $\Sigma\Delta$ converter by using SC circuits will be discussed in following sections.

3.1. Switched Capacitors

Integration of circuits with both analog and digital functionality on the same die has motivated people to develop switched–capacitor techniques. In modern VLSI technology, main elements for implementation of circuits are MOS transistors and capacitors, since resistor implementation is undesirable. Facing the problem of resistor realization in VLSI technology, it was proposed that resistors could be replaced by operating MOS capacitors with MOSFET switches which are rapidly turned on and off periodically [18]. Figure 3.2 illustrates some arrangements and clocking scheme for resistor simulation.

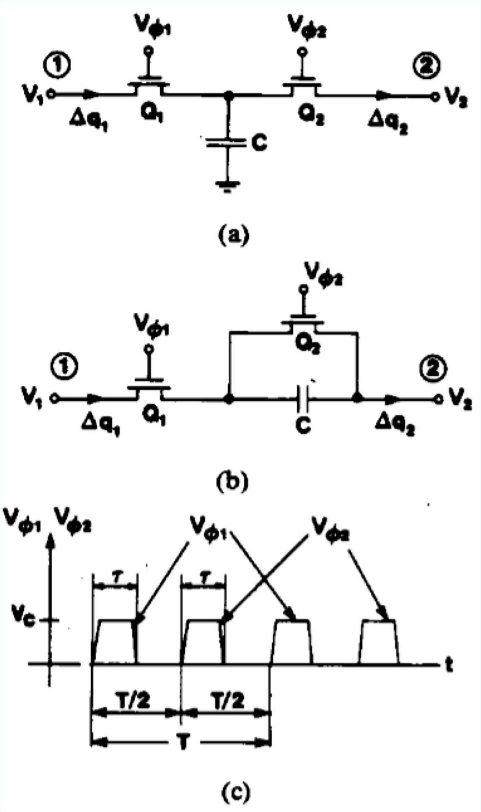


Figure 3.2. Switched capacitor circuits (a) Shunt circuit (b) Series circuit (c) Clock waveforms [19].

In Figure 3.2(a), for the $\phi 1$ cycle, capacitor is charged to V_1 whereas for the $\phi 2$ cycle, it is discharged to V_2 assuming V_1 is larger than V_2. For the case in Figure 3.2(b), capacitor is alternately charged and discharged to a voltage of $V_1 - V_2$. In the total period of T, a charge of $\Delta Q = C(V_1 - V_2)$ flows with the indicated polarities for both cases. This charge transfer occurs at each period as sharp pulses at the outset of the clock pulses. But an average current i_{avg} can be defined as the charge flow, ΔQ, divided by period, T

$$i_{avg} = \frac{\Delta Q}{T} = \frac{V_1 - V_2}{T/C}. \tag{3.1}$$

Charge process occurs in discrete–time rather than continuous–time. However, if clocking frequency is made sufficiently higher than the highest–frequency components of V_1 and V_2, it can be regarded as continuous. Hence, the arrangement having equation (3.1) as operating equation can be modeled as a resistor with the following value:

$$R = \frac{V_1 - V_2}{i_{avg}} = \frac{T}{C} = \frac{1}{Cf}. \tag{3.2}$$

As it is seen from Equation (3.2), resistor value is determined by the clock frequency, which says that the value of the resistor is adjustable. In addition to this advantage, the die area used to implement a resistor is greatly reduced by means of this technique. As an example given in [19], for a resistance of $R = 10^7\Omega$, assuming a clock frequency of 100 kHz, a capacitance of 1 pF is needed, which occupies an area of 3 mil^2 . For that case, direct implementation of the resistance requires an area of 1600 mil^2. Another important advantage is that the frequency response of the SC circuit is controllable by adjusting capacitance ratios. By means of this fact, it is possible to improve the overall accuracy of the circuit at much less physical design cost and effort [3].

Main application area of switched capacitors is filtering, but they are used in numerous applications such as voltage amplifier, programmable capacitor arrays, balanced modulator etc. which are discussed in [19]. In this work, switched capacitor techniques are used for integrator and coefficient implementation, which will be discussed in Section 3.2 and Section 3.5.

3.2. Switched Capacitor Integrator

3.2.1. Parasitic Sensitive Switched Capacitor Integrators

In $\Sigma\Delta$ modulator, integrators are used to implement loop filters. Since SC usage is advantageous as it is presented in Chapter 3 and Section 3.1, SC implementation is used for integrators. A switched capacitor integrator using shunt circuit for resistor simulation, is shown in Figure 3.3.

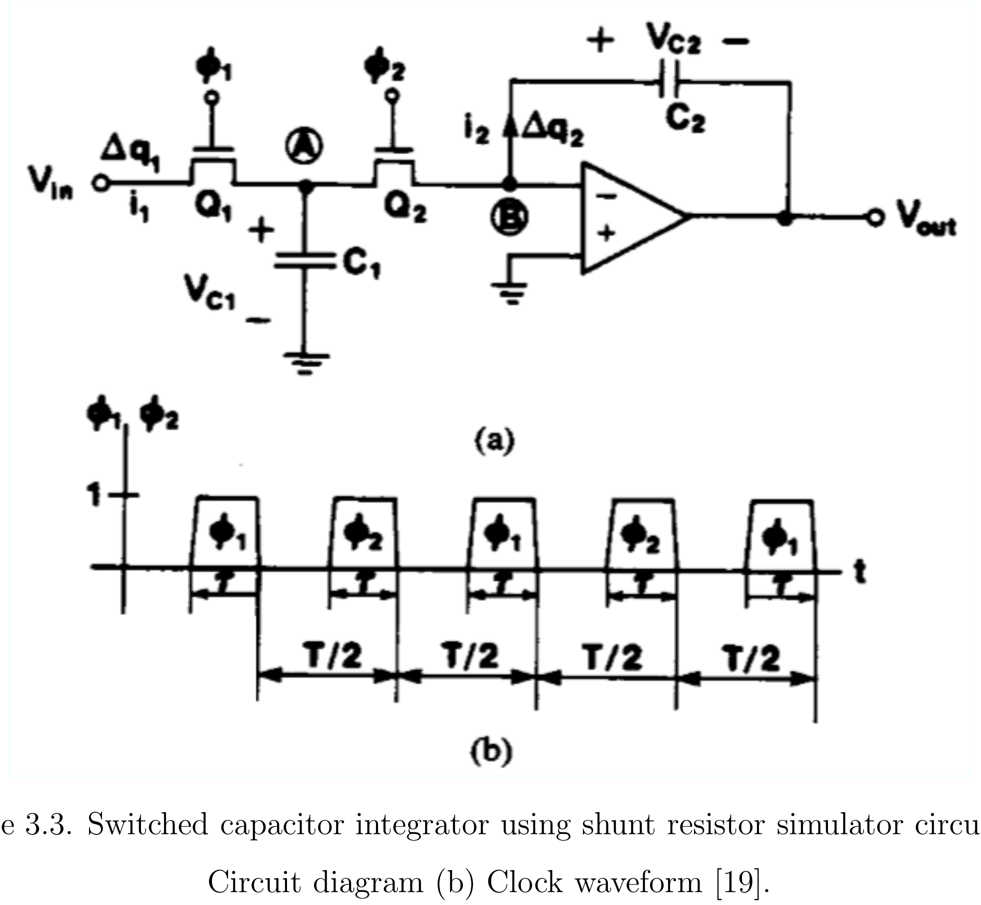


Figure 3.3. Switched capacitor integrator using shunt resistor simulator circuit (a) Circuit diagram (b) Clock waveform [19].

To obtain the governing equation, look at the operation at t_{n-1}, which is the end of first ϕ_1 pulse. C_1 is charged to $V_{in}(t_{n-1})$ with a charge of $C_1 V_{in}(t_{n-1})$. At $t = t_{n-1} + T/2$

this charge is added to C_2 by the ϕ_2 pulse. So at $t = t_n$ output voltage becomes [19]

$$v_{out}(t_n) = -v_{C2}(t_n) = v_{out}(t_{n-1}) - \frac{C_1}{C_2} v_{in}(t_{n-1}). \tag{3.3}$$

Solving this equation using z–transformation gives the following transfer function:

$$H(z) \triangleq \frac{V_{out}(z)}{V_{in}(z)} = -\frac{C_1}{C_2} \frac{z^{-1}}{1 - z^{-1}}. \tag{3.4}$$

This topology is known as *Forward Difference* or *Forward Euler Mapping.* SC Integrator can also be implemented with series resistor simulation which is shown in Figure 3.4 with the same clocking scheme in Figure 3.3.

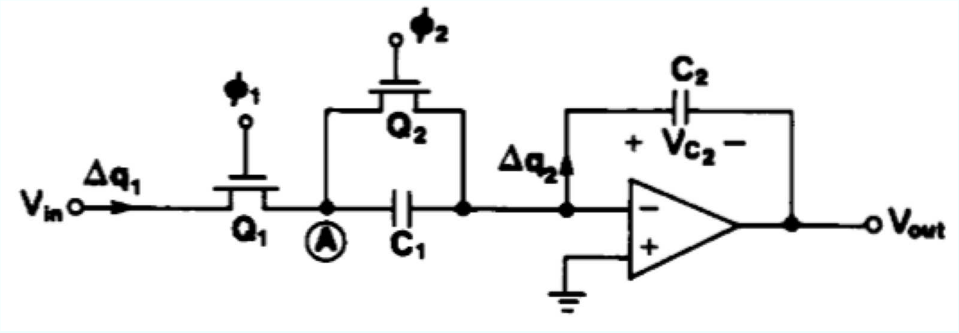


Figure 3.4. Switched capacitor integrator using series resistor simulator circuit [19].

Operation of series simulator is a little bit different than the shunt one. Charge is sampled and sent to C_2 at pulse ϕ_1. Doing the same charge analysis for this circuitry gives the transfer function

$$H(z) \triangleq \frac{V_{out}(z)}{V_{in}(z)} = -\frac{C_1}{C_2} \frac{1}{1 - z^{-1}}. \tag{3.5}$$

As it is seen, delay element of z^{-1} in Equation (3.4) is missing in Equation (3.5). The reason is that charge is sent to C_2 as it is sampled. There is a delay of T between v_{in} and v_{out} for the shunt case. This configuration of series simulation resistor usage is known as *Backward Difference* or *Backward Euler Mapping.*

3.2.2. Parasitic Insensitive Switched Capacitor Integrators

Up to now, everything was assumed to be ideal. However, switches, operational amplifier and MOS Capacitor have parasitic capacitances, which are generically called *Stray Capacitances.* These capacitances are discussed further in [20] and it is proposed in [20] that for shunt case those capacitances can be lumped into a parallel capacitance to C_1. For a sampling capacitance of 1 pF, parasitic capacitance can be as large as 0.05 pF. This can cause problems in operation, so some methods were devised in order to eliminate the effects of this capacitance. The solution is the *Stray Insensitive Integrator* which is shown in Figure 3.5.

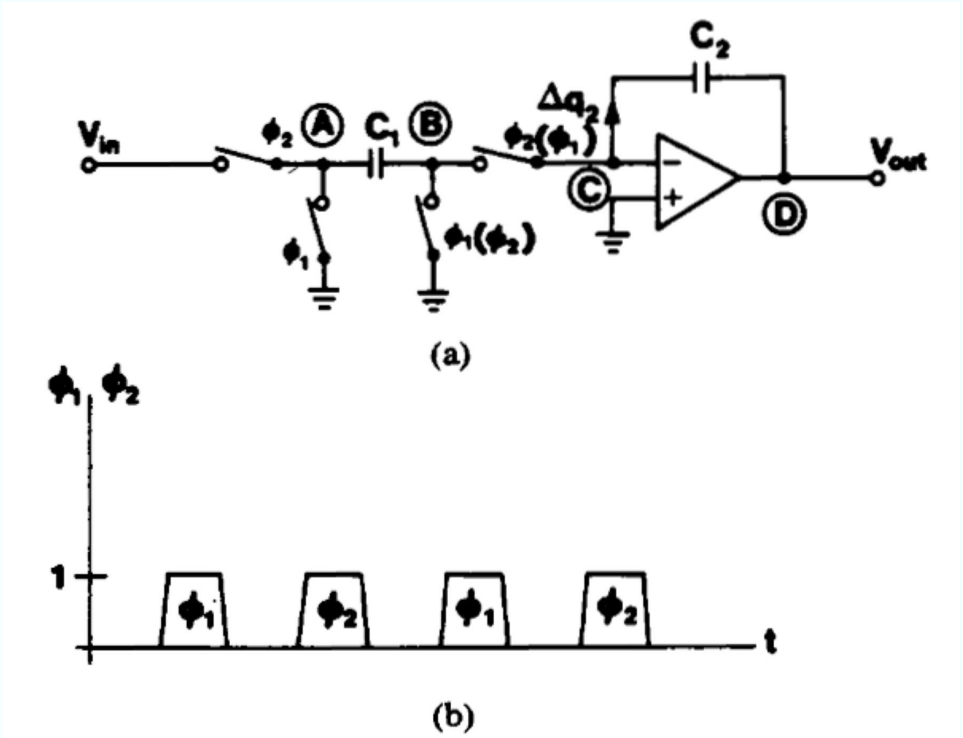


Figure 3.5. Stray Insensitive SC Integrator (a) Circuit Diagram (b) Clock waveforms [19].

In this configuration, again C_1 is charged and discharged with v_{in}, the only change being the additional two switches. By means of those switches, every capacitor terminal is switched either between low-impedance nodes (ground and an op-amp output) or ground and virtual ground. This results in eliminaton of the effect of the stray capacitance. In the case of using clocks without parentheses, operation of the circuit resembles Backward Euler case with ϕ_1 and ϕ_2 interchanged. C_1 discharges during

$\phi_1 = 1$, and recharges through v_{in} and C_2 during $\phi_2 = 1$. Thus, the transfer function in Equation (3.5) holds in the case of output sampling at $\phi_2 = 1$ [19]. In the case of output sampling at $\phi_1 = 1$, an extra factor of z^{-1} is introduced since output is delayed.

If the clocking scheme in parentheses is used, the following occurs. C_1 is charged by v_{in} during $\phi_1 = 1$ and it discharges to C_2 during $\phi_2 = 1$. Output is sampled when $\phi_2 = 1$. In this sense, operation of the circuit is similar to the shunt resistor simulation implementation case. The only change is that the polarity of C_1 is changed since plates connected to ground in phases of $\phi_1 = 1$ and $\phi_2 = 1$ are different. Thus, transfer function of the circuit becomes

$$H(z) = \frac{C_1}{C_2}\frac{z^{-1}}{1 - z^{-1}}. \tag{3.6}$$

As it is observed from Equation (3.6), gain of this integrator is positive meaning that this integrator is non–inverting whereas previous ones were all inverting.

Till now, all the circuits discussed were single–ended. Although single–ended circuits are economical, they suffer from non-ideality effects like amplifier offset voltage and clock feedthrough, which can degrade performance of the topology. The most effective solution to this problem is the usage of differential circuitry. Since this is the choice of implementation, it will be dealt with further in Chapter 4. Another advantage of differential circuitry usage is that interchanging input or output terminals changes the sign of the transfer function [2].

3.2.3. A Switched Capacitor Integrator Design Example

After explanation of SC integrators, a design example of SC integrator will be presented. As discussed in Section 3.2.2, differential implementation of the integrator can result in significant benefits. Non-inverting SC differential integrator was chosen as the example, the block diagram of which is given in Figure 3.6.

In the figure, C_s and C_f symbolize the sampling and feedback capacitors respec-

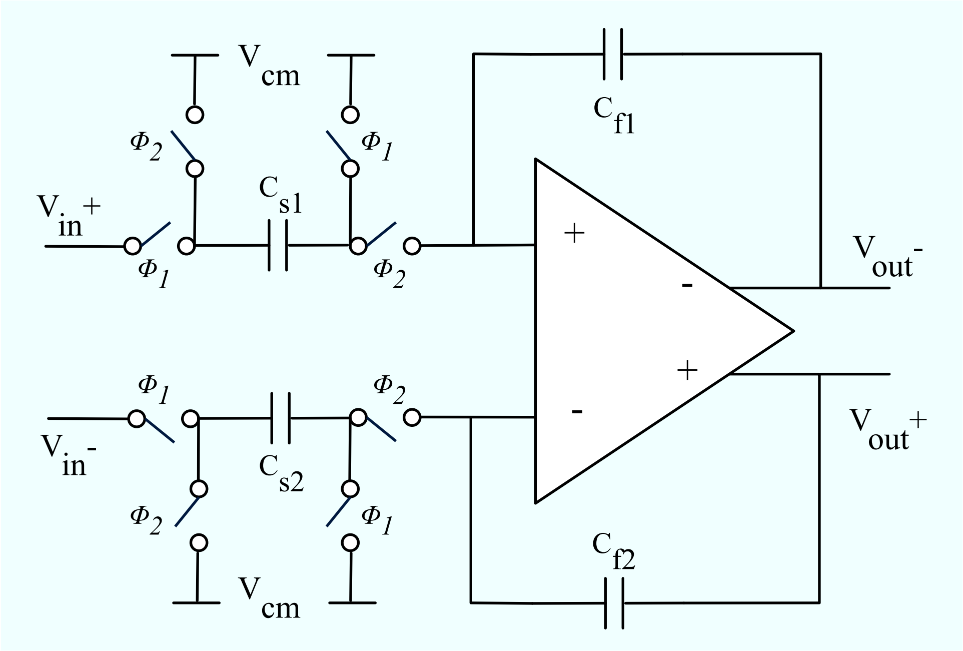


Figure 3.6. Differential non-inverting SC Integrator.

tively. Differences from the previously discussed non-inverting integrator are as follows. At first, topology is differential, so differential operational amplifier is used. Secondly, ground port in the previous topology is replaced by V_{cm}, which is the abbreviation for common mode voltage. Nothing changes about circuit operation, since difference between V_{in} and V_{cm} is sampled on C_s and it is transferred to C_f. Thereby, a factor of the difference, which is C_s/C_f is realized as a gain factor.

Operational amplifier used in this implementation example is a *Fully Differential Folded Cascode Operational Amplifier.* It is the chosen because it has many advantages such as being a single stage amplifier. It does not encounter stability problems. Compensation is achieved by load capacitance, so there is no need for internal capacitance. No need for internal compensation results in high slew rate which can be important in charging capacitances. Theoretical details about the design of operational amplifier can be found in [21]. The designed operational amplifier is depicted in Figure 3.7.

Bias voltages for proper operation are $V_{bias1} = 2.1V$, $V_{bias2} = 1.83V$. The specifications of this op-amp obtained with a load capacitance of 4 pF, are as follows: Gain = 76.5 dB, BW = 200 MHz, 3 dB cut-off = 55 kHz, SR = 182.4 $V/\mu s$. The output

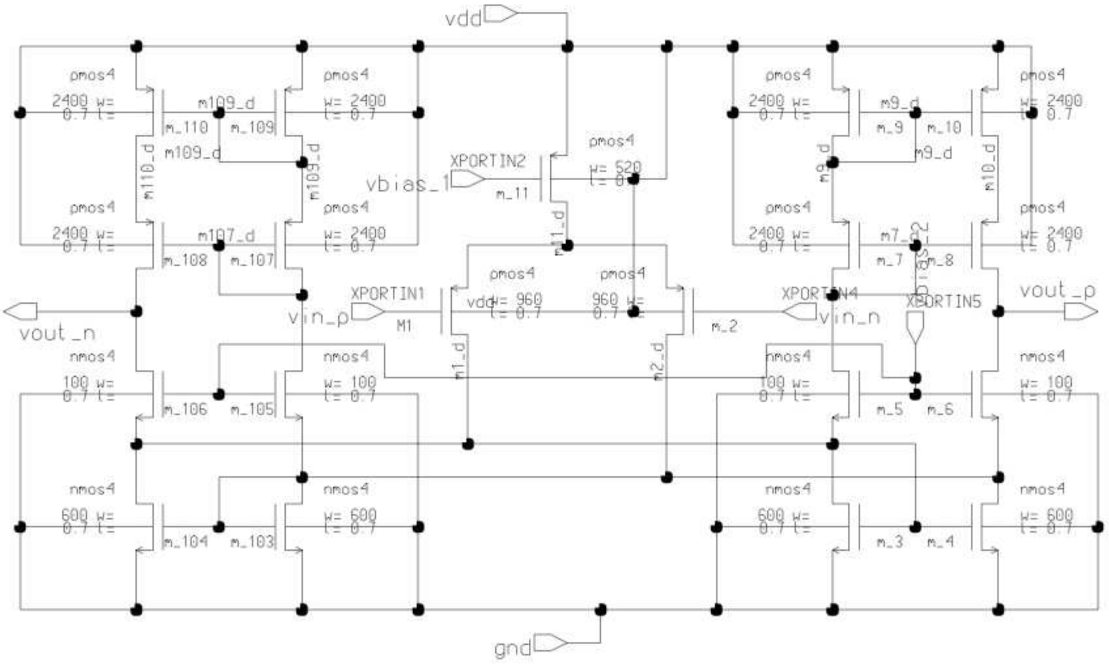


Figure 3.7. Fully differential folded cascode operational amplifier.

DC level of the operational amplifier is 1.472 V.

Switches in the implementation were just gates. Design of the switch is shown in Figure 3.8.

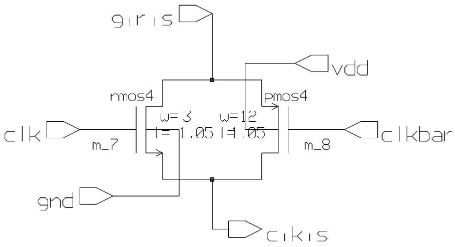


Figure 3.8. Transmission gate switch.

The circuit was simulated using a differential DC input voltage of $V_{in} = 100mV$ and by selecting $C_{s1} = C_{s2} = 1pF$ and $C_{f1} = C_{f2} = 2pF$ in Eldo which is a part of the Mentor Graphics Suite. The simulation result is shown in Figure 3.9. As it is seen from Figure 3.9, output increases with steps of 50 mV. Gain of the integrator is $C_s/C_f = 1pf/2pF = 0.5$ theoretically. This is in accordance with the simulation since with a differential input of 100 mV, output increases with steps of 50 mV. Integrator operates truly up to the output levels of 1.8 V. After this value, operational amplifier saturates and integrator operation becomes problematic.

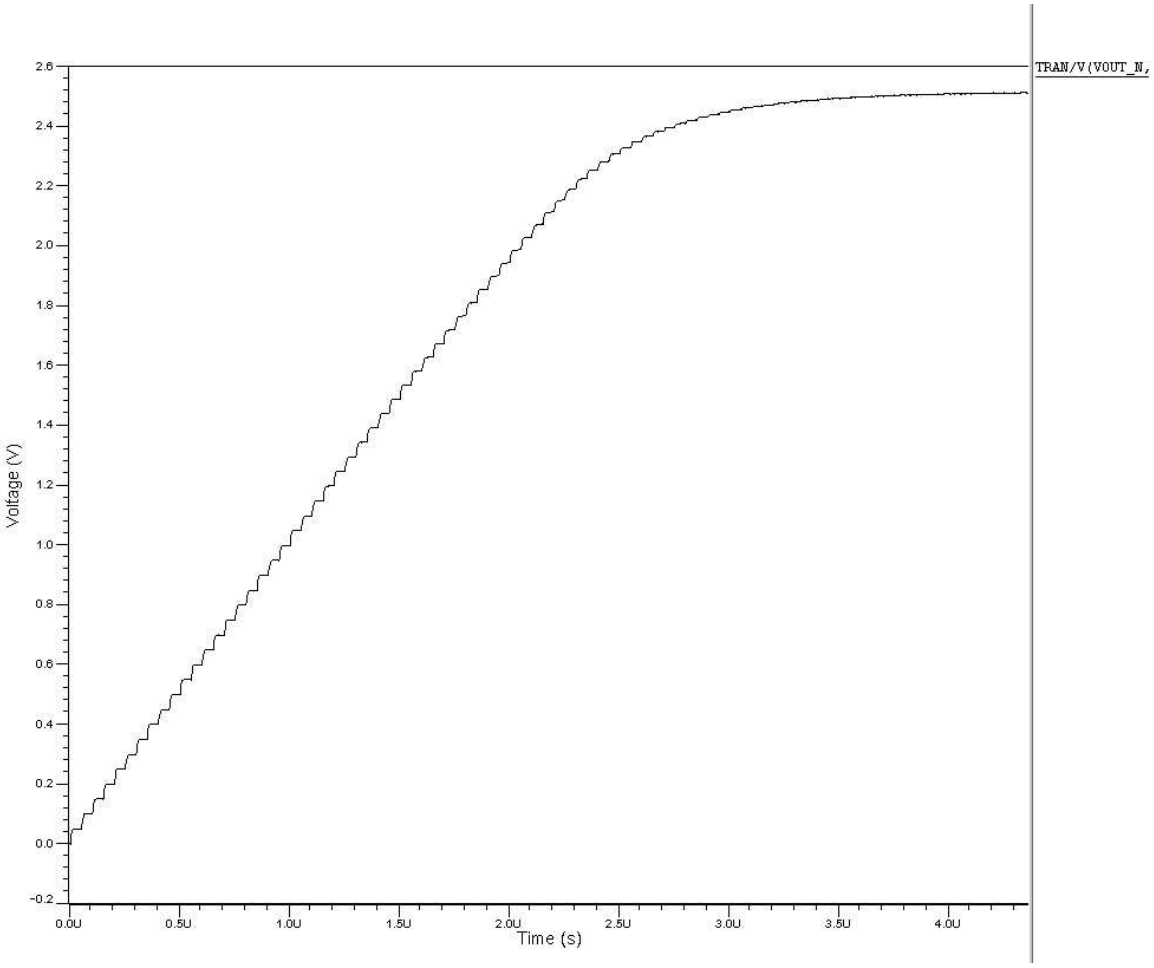


Figure 3.9. Integrator output

3.3. Quantizer

Quantizer block follows the last integrator. Although it can be multi–bit in Multi–Bit ΣΔ Modulators, in this work single–bit modulators are discussed. As a result, single–bit quantizer is used, which is simply a comparator. The comparator used in the implementation of the modulator has been taken from the design database of BETA (Boğaziçi Üniversitesi Elektronik Tasarım Laboratuvarı), which is given in Figure 3.10.As it is seen comparator is clocked. When clock is high, the circuit samples input and changes its output values. When clock is low the circuit is not functioning. When it is functioning, comparator gives 3.3 V and 0 V as outputs. It needs a bias voltage of 0.75 V.

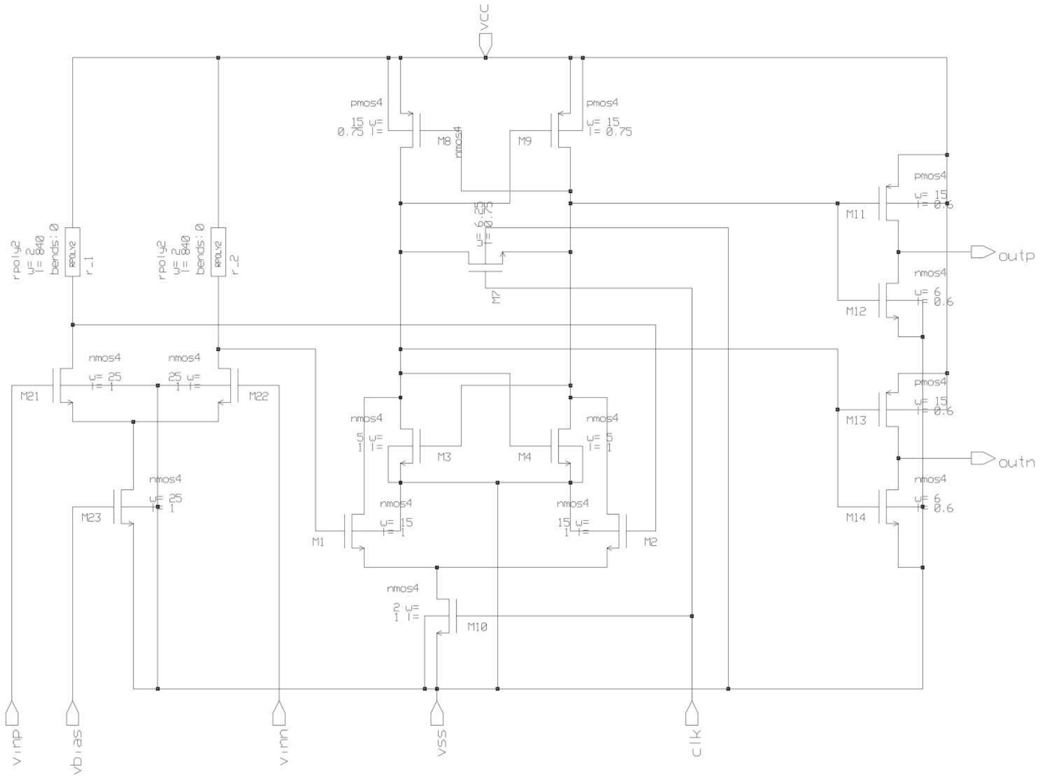


Figure 3.10. Comparator circuit.

3.4. DAC

The other component of the modulator is DAC that is present in feedback path. This DAC is just a multiplexer as it is shown in Figure 3.11. According to the select signals, it selects one of its two inputs, V_{ref+} and V_{ref-}. In our case, $V_{ref+} = 2.15V$ and $V_{ref-} = 1.15V$. Also, the select signals are the two differential outputs of the comparator.

In this work, differential implementation is done, so two DACs are used, which are referred to as DAC1 and DAC2. If $clk1$ and $clk2$ shown in Figure 3.11, are the select signals, DAC1 selects 2.15 V for $clk1 = HIGH$ and $clk2 = LOW$ and DAC2 selects 1.15 V for the same select signals.

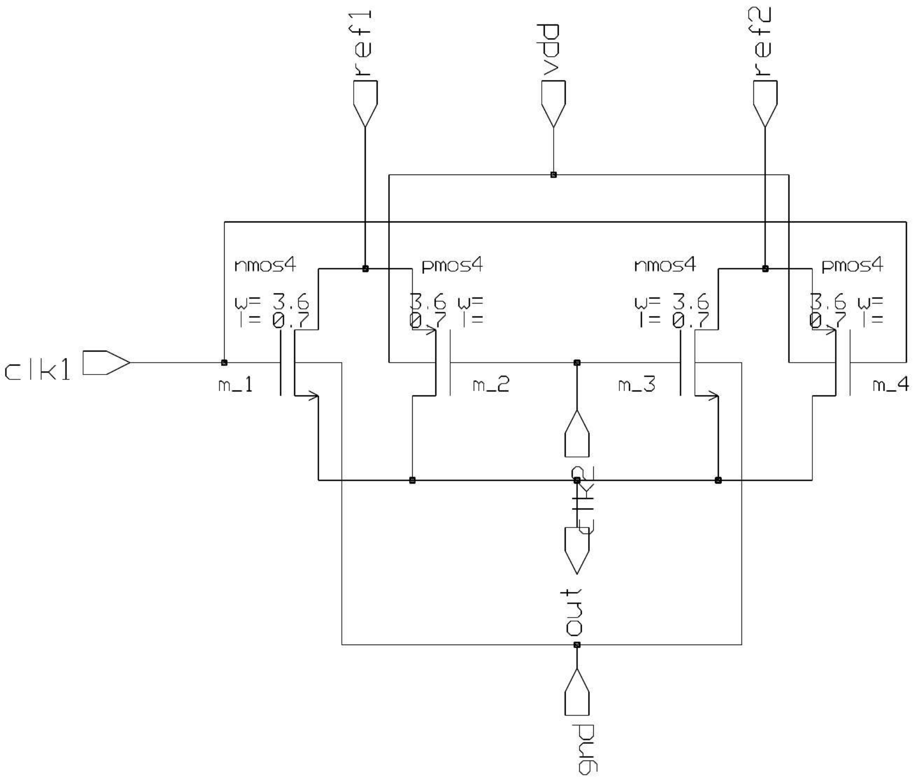


Figure 3.11. Digital to Analog converter

3.5. ΣΔ Modulator Coefficients

3.5.1. Sigma Delta Converter Design Automation Tool

As it was explained in previous section, integrators are implemented using SC circuits. Gains of the integrators are determined by the ratio of sampling capacitor, C_s to the feedback capacitor, C_f. In addition to the gains of the integrators, feedback paths can have gains. Also, new feedback paths from outputs of the integrator, to the inputs of the integrator can be introduced. In addition to this, feedforward paths with gain can be added to the ΣΔ modulator architecture. Those paths are directed from input, inputs of the integrators and outputs of the integrator to the following integrators' inputs, outputs or even to the input of the quantizer. For the Second Order ΣΔ modulator, addition of all possible paths forms the architecture given in Figure 1.7.

As it is observed from Figure 1.7, there are 15 coefficients for the second order case. Choosing among some of those paths and their values gives a great degree of freedom, there exist innumerable number of topologies. There can be a great number of different coefficient sets each satisfying the same STF and NTF. At this point, the need for optimum solution according to a cost function appears as the main motivation for addition of all those paths to the topology. In the process of forming a cost function some metrics can be introduced:

- **Number and value of coefficients:** This can be an important concern since having less coefficients results in simpler architecture. Sum of the values of coefficients may give an idea of the area for implementation. In some cases, coefficients may be large multiples of each other. This difference between the values of coefficients may cause a problem.
- **Power Consumption:** Different architectures may result in different power consumptions, which is an important concern from a design point of view, which will be further discussed in Section 5.3.
- **Sensitivity:** Some architectures may result in topologies which are so sensitive to the value of a path that even small process variations may cause a major performance degradation.

Taking all above given considerations into account, it should be obvious that determination of the paths to be used and their values is an important process. As explained in Section 1.2, this work starts at the point where paths and gains are to be determined. It searches for implementability of the found architecture, makes deductions about implementations such as power consumption, sensitivity and area which will be discussed in Chapter 4 and Chapter 5. At this point, the developed tool, which gives this thesis a start, is discussed. The tool presented in [11, 12] starts with a SPICE like netlist, which includes the connection scheme of the blocks in a $\Sigma\Delta$ modulator. For the case of second order modulator, the netlist is as in Figure 3.12, which also corresponds to modulator in Figure 1.7.

As it is seen, the tool includes main blocks, since it has those blocks modeled.

```
1   IN 1
2   GAIN1 1 2
3   GAIN2 3 9
4   GAIN3 7 4
5   GAIN4 7 8
6   GAIN5 1 10
7   GAIN6 12 17
8   GAIN7 1 18
9   GAIN8 3 19
10  GAIN9 14 16
11  GAIN10 6 15
12  GAIN11 14 13
13  GAIN12 6 11
14  GAIN13 6 5
15  GAIN14 14 23
16  GAIN15 7 24
17  ADDER 2 -5 -4 -23 3
18  ADDER 8 -9 -10 -11 -13 12
19  ADDER 16 -17 -18 -19 -15 -24 20
20  ADDER 20 21 22
21  DAC 22 6
22  INTEGRATORD 3 7
23  INTEGRATORD 12 14
24  NOISE 21
```

Figure 3.12. Netlist for second order modulator.

Both delayless and delayed integrators are modeled by their z–transform counterparts. Other blocks are also modeled in an appropriate way. The tool takes desired NTF and STF from user and finds solutions on the basis of minimizing the number of paths by avoiding delayless loops. It achieves this because it can calculate parametric NTF and STF between any two nodes in the architecture. Then, it matches the desired ones with the parametric ones to solve equations. Finally, it comes with possible solutions.

The developed tool also models the non-idealities in the modulator in the form of non-ideal integrator, which means that non-idealities are taken into account in the determination of the coefficients. It should be noted that coefficients for any order of $\Sigma\Delta$ modulator can be found by using this tool since the only change for the tool is the netlist, which is entered by user.

3.5.2. Implementation of Coefficients

All the blocks used in the design of $\Sigma\Delta$ modulator architecture were introduced in the previous sections. As explained in the previous sections, paths with gains are inserted to the $\Sigma\Delta$ modulator architectures to get a better response as shown in Figure 1.7. Implementation of those coefficients is achieved by circuitry in Figure 3.13. In general, the value of the coefficient is given by

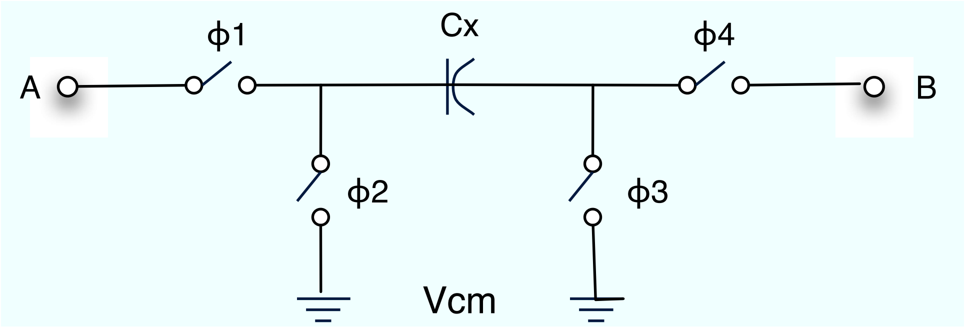


Figure 3.13. Coefficient implementation circuitry.

$$g_x = \frac{C_x}{C_f}, \tag{3.7}$$

where C_x is the value of the capacitance given in Figure 3.13 and C_f is the value of the feedback capacitance of operational amplifier, to the input of which the coefficient is connected. There are some exceptions to this rule, which will be explained in detail later. In the following, implementation of each coefficient given in Figure 1.7 will be explained individually with reference to Figure 3.13. Before going into the details of the implementation of the coefficients, two issues should be noted

i. Since the implemented architectures are differential, two switched capacitor networks will be needed to implement each coefficient.

ii. For the coefficients connected to the input of the quantizer, we should insert the circuit in Figure 3.13 between outputs of the second integrator and inputs of the quantizer. In the rest of the explanations, it will be assumed that the name of the capacitors of these inserted networks is $C_x = C_{out}$.

- g_1 **:** This coefficient is the gain of the first integrator. If $C_x = C_1$, the value of the implemented coefficient is C_1/C_{f1}, where C_{f1} is the feedback capacitance of the first integrator. Clock phases should be adjusted as $\phi_1 = \phi_3 = \phi_a$ and $\phi_2 = \phi_4 = \phi_b$ where ϕ_a and ϕ_b are non-overlapping clocks having the same period. To realize a positive value for g_1, V_{in+} should be connected to port A of one SC network and port B of this network should be connected to the positive input of the first operational amplifier. Similarly V_{in-} should be connected to port A of the other SC network and port B of this network should be connected to the negative input of the first operational amplifier.

- g_2 **:** This coefficient is the feedforward coefficient from the input of the first integrator to the input of the second operational amplifier. Its implementation is rather difficult since its input is the sum of the outputs of a number of other coefficients, which in worst case is 4. These 4 coefficients are namely g_1, g_3, g_{13}, g_{14}. There are two cases for implementation of g_2:

  i. If only g_1 and g_{13} are present at the same time with g_2 and if $g_1 = g_{13}$, g_2 can be realized by taking the difference of the modulator input signal and the signal coming from DAC. This difference can be taken by using the SC network shown in Figure 3.13 by connecting the modulator input signal to port A and replacing V_{cm} on the left of C_x by the signal coming from the DAC. In this case, C_x/C_{f2} should be adjusted in such a way to realize the value $g_2g_1 = g_2g_{13}$, where C_{f2} is the feedback capacitance of the second integrator. Clock phases should be adjusted as $\phi_1 = \phi_3 = \phi_a$ and $\phi_2 = \phi_4 = \phi_b$.

  ii. In other cases, g_2 should be realized by distributing its effect to the coefficients connected to the input of the first integrator, which are $g_1, g_3, g_{13}.g_{14}$. To distribute g_2 over g_1 extra SC networks should be used. Their inputs should be same as the networks realizing g_1 itself and outputs should be going to the input of the second operational amplifier. Capacitance values of these networks should be chosen so as to realize g_1g_2. The situation is the same for the remaining three coefficients over which g_2 should be distributed. Again extra SC networks are needed. In distributing g_2 over g_3, the inputs of the network should be same as those realizing g_3 itself, and outputs should be going to the second operational amplifier again. The procedure and the input output connection scheme is the same for g_{13} and g_{14}.

- g_3 **:** This coefficient is the local feedback of the first integrator. It samples the output of the first integrator and feeds it back to the input of it. If $C_x = C_3$, the value of implemented coefficient is C_3/C_{f1}. Clock phases should be adjusted as $\phi_1 = \phi_3 = \phi_a$ and $\phi_2 = \phi_4 = \phi_b$. To realize a positive value for g_3, positive output of the first integrator should be connected to port A of one SC network and port B of this network should be connected to the positive input of the first operational amplifier. Similarly, negative output of the first integrator should be connected to port A of the other SC network and port B of this network should be connected to the negative input of the first operational amplifier.

- g_4 : This coefficient is the gain of the second integrator. If $C_x = C_4$, the value of the implemented coefficient is C_4/C_{f2}. Clock phases should be adjusted as $\phi_1 = \phi_3 = \phi_a$ and $\phi_2 = \phi_4 = \phi_b$. To realize a positive value for g_4, negative output of the first integrator should be connected to port A of one SC network and port B of this network should be connected to the positive input of the second operational amplifier. Similarly postive output of the first integrator should be connected to port A of the other SC network and port B of this network should be connected to the negative input of the second operational amplifier.
- g_5 : This coefficient is the feedforward coefficient from the modulator input to the input of the second integrator. It samples the modulator input and feeds it to the input of the second operational amplifier. If $C_x = C_5$, the value of the implemented coefficient is C_5/C_{f2}. Clock phases should be adjusted as $\phi_2 = \phi_3 = \phi_a$ and $\phi_1 = \phi_4 = \phi_b$. To realize a positive value for g_5, positive input of the modulator should be connected to port A of one SC network and port B of this network should be connected to the negative input of the second operational amplifier. Similarly, negative input of the modulator should be connected to port A of the other SC network and port B of this network should be connected to the positive input of the second operational amplifier.
- g_6 : Situation is similar to the case of g_2. It can be implemented by taking difference if only g_4 and g_{12} exist at the same time with g_6 and their values are same. In other cases, effect of g_6 should be distributed over $g_2, g_4, g_5, g_{11}, g_{12}$. The implementation procedure and the input output connection schemes for the extra introduced networks is analogous to those for g_2.
- g_7 : This coefficient is the feedforward coefficient from modulator input to the input of the quantizer. It samples the modulator input and feeds it to the input of the quantizer. If $C_x = C_7$, the value of the implemented coefficient is C_7/C_{out},

where C_{out} is the capacitance of the switched capacitor circuit inserted between output of the second integrator and input of the comparator. Clock phases should be adjusted as $\phi_2 = \phi_3 = \phi_a$ and $\phi_1 = \phi_4 = \phi_b$. To realize a positive value for g_7, positive input of the modulator should be connected to port A of one SC network and port B of this network should be connected to the negative input of the comparator. Similarly negative input of the modulator should be connected to port A of the other SC network and port B of this network should be connected to the positive input of comparator.

- g_8 : Implementation is same with g_2, only change is that output is connected to the input of the comparator.
- g_9 : The value of the implemented coefficient is C_{out}/C_{f2}. Clock phases should be adjusted as $\phi_1 = \phi_3 = \phi_a$ and $\phi_2 = \phi_4 = \phi_b$. To realize a positive value for g_9, negative output of the second integrator should be connected to port A of one SC network and port B of this network should be connected to the negative input of the comparator. Similarly negative output of the second integrator should be connected to port A of the other SC network and port B of this network should be connected to the negative input of the comparator.
- g_{10} : If $C_x = C_{10}$, the value of implemented coefficient is C_{10}/C_{out}. But implementation of this coefficient is not useful in practice, since this coefficient introduces delayless loop, which causes problem in operation.
- g_{11} : This coefficient is the local feedback of the second integrator. It samples the output of the second integrator and feeds it back to the input. If $C_x = C_{11}$, the value of the implemented coefficient is C_{11}/C_{f2}. Clock phases should be adjusted as $\phi_1 = \phi_3 = \phi_a$ and $\phi_2 = \phi_4 = \phi_b$. To realize a positive value for g_{11}, positive output of the second integrator should be connected to port A of one SC network and port B of this network should be connected to the positive input of the second operational amplifier. Similarly negative output of the second integrator should be connected to port A of the other SC network and port B of this network should be connected to the negative input of the second operational amplifier.
- g_{12} : This coefficient is the feedback from DAC to the second integrator. It samples the output of DAC and feeds it back to the input of second integrator. If $C_x = C_{12}$, the value of implemented coefficient is C_{12}/C_{f2}. Clock phases should

be adjusted as $\phi_1 = \phi_4 = \phi_b$ and $\phi_2 = \phi_3 = \phi_a$. To realize a positive value for g_{12}, output of DAC1 should be connected to port A of one SC network and port B of this network should be connected to the positive input of the second operational amplifier. Similarly output of DAC2 should be connected to port A of the other SC network and port B of this network should be connected to the negative input of the second operational amplifier, where DAC1 and DAC2 are as explained in Section 3.4.

- g_{13} **:** This coefficient is the feedback from DAC to the first integrator. It samples the output of DAC and feeds it back to the input of first integrator. If $C_x = C_{13}$, the value of the implemented coefficient is C_{13}/C_{f1}. Clock phases should be adjusted as $\phi_1 = \phi_4 = \phi_b$ and $\phi_2 = \phi_3 = \phi_a$. To realize a positive value for g_{13}, output of DAC1 should be connected to port A of one SC network and port B of this network should be connected to the positive input of the first operational amplifier. Similarly output of DAC2 should be connected to port A of the other SC network and port B of this network should be connected to the negative input of the first operational amplifier, where DAC1 and DAC2 are as explained in Section 3.4.
- g_{14} **:** This coefficient is the feedback coefficient from the output of the second integrator to the input of the first integrator. It samples the output of the second integrator and feeds it back to the input of operational amplifier of the first integrator. If $C_x = C_{14}$, the value of the implemented coefficient is C_{14}/C_{f1}. Clock phases should be adjusted as $\phi_1 = \phi_3 = \phi_a$ and $\phi_2 = \phi_4 = \phi_b$. To realize a positive value for g_{14}, positive output of the second integrator should be connected to port A of one SC network and port B of this network should be connected to the positive input of the first operational amplifier. Similarly negative output of the second integrator should be connected to port A of the other SC network and port B of this network should be connected to the negative input of the first operational amplifier.
- g_{15} **:** This coefficient is the feedforward coefficient from output of the first integrator to the input of the quantizer. It samples the output of the first integrator and feeds it to the input of the quantizer. If $C_x = C_{15}$, the value of implemented coefficient is C_{15}/C_{out}. Clock phases should be adjusted as $\phi_2 = \phi_3 = \phi_a$ and

$\phi_1 = \phi_4 = \phi_b$. To realize a positive value for g_{15}, positive output of the first integrator should be connected to port A of one SC network and port B of this network should be connected to the negative input of the comparator. Similarly negative output of the first integrator should be connected to port A of the other SC network and port B of this network should be connected to the positive input of comparator.

4. IMPLEMENTED ARCHITECTURES

All the blocks which compose a $\Sigma\Delta$ modulator were presented in Chapter 3. In addition to the main blocks, implementation of the coefficients, which are inserted to the modulator, was explained in Section 3.5.2. What is to be done finally is to build $\Sigma\Delta$ modulators using all components. $\Sigma\Delta$ modulator to be implemented was chosen as the modulator having standard second order response which is

$$NTF(z) = (1 - z^{-1})^2 \quad \text{and} \quad STF(z) = z^{-2}. \tag{4.1}$$

In Equation (4.1), NTF response is a high–pass response and STF response is an all-pass response. The tool in [11, 12] used for finding solutions for the coefficients. From the solution set, 13 topologies were selected. These topologies were implemented and simulations were done including the power spectral density of the output. In the following pages, block diagrams and PSD graphs will be given. Note that in $\Sigma\Delta$ modulators SNR is found from PSD graph by subtracting the highest noise component in the frequency band of 2.5 times the signal frequency from the signal. In the implementation, sampling frequency was 20 MHz, signal was a sinusoidal with a peak amplitude of 200 mV and frequency of 200 kHz. Implementation of the coefficients were done as it was explained in Section 3.5.2 with feedback capacitances across the operational amplifier being 1 pF.

- **STANDARD ARCHITECTURE**

  Figure 4.1 shows the block diagram of the standard architecture. In the architecture, values of the existing coefficients are

$$g_1 = 1,\ g_4 = 1,\ g_9 = 1,\ g_{12} = 2,\ g_{13} = 1. \tag{4.2}$$

Simulations were done and PSD of the architecture is given in Figure 4.2. Signal is 42.5 dB and noise floor is at -10 dB. SNR is measured as 42.8 dB.

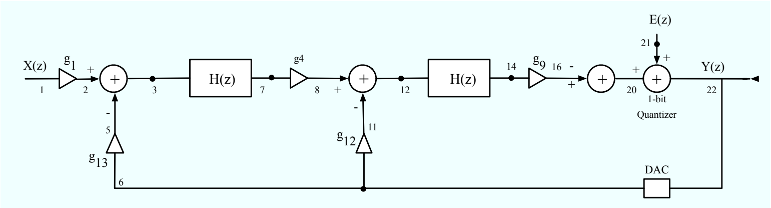


Figure 4.1. Standard architecture.

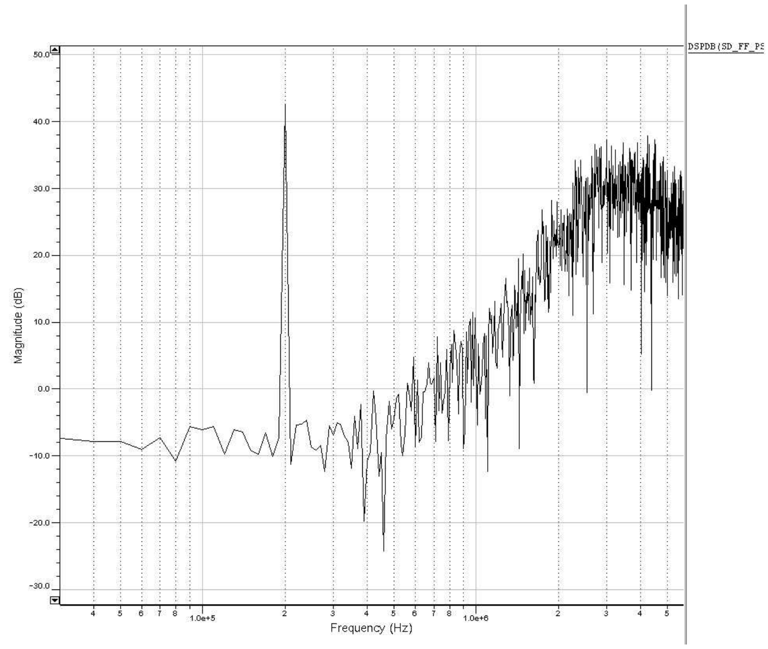


Figure 4.2. Power Spectral Density (PSD) of standard architecture.

- **ARCHITECTURE–1**

Figure 4.3 shows the block diagram of the first architecture.

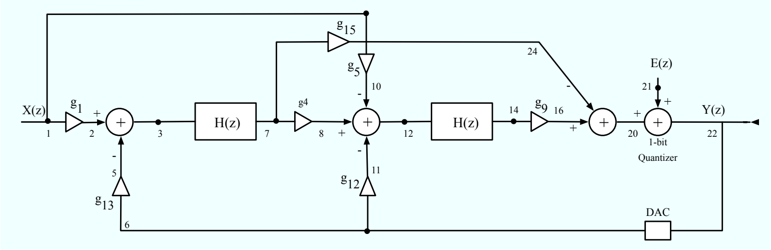


Figure 4.3. First architecture.

In the architecture, values of the existing coefficient are

$$g_1 = 1,\ g_4 = 1,\ g_5 = 1,\ g_9 = 1,\ g_{12} = 1,\ g_{13} = 1,\ g_{15} = -1. \tag{4.3}$$

Simulations were done and PSD of the architecture is given in Figure 4.4. Signal is 42.1 dB and noise floor is at -4 dB. SNR is measured as 39.9 dB.

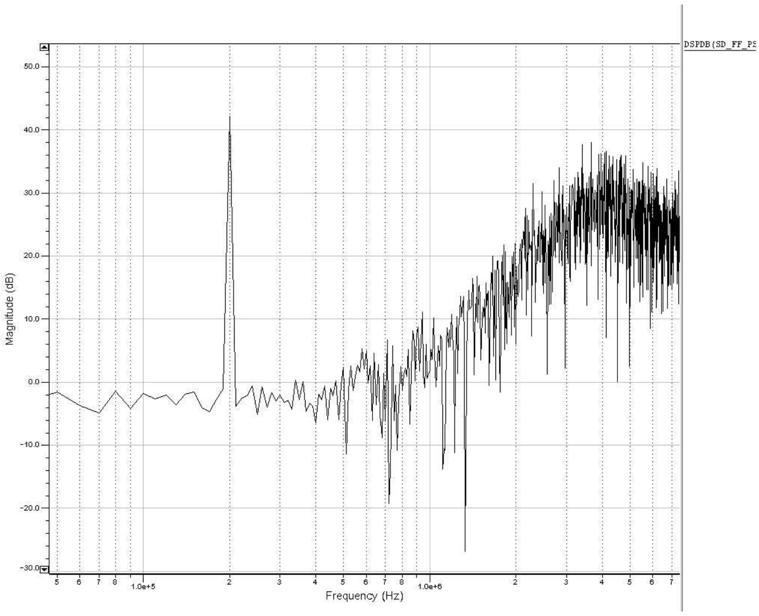


Figure 4.4. PSD of first architecture.

- **ARCHITECTURE–2**

  Figure 4.5 shows the block diagram of the second architecture.

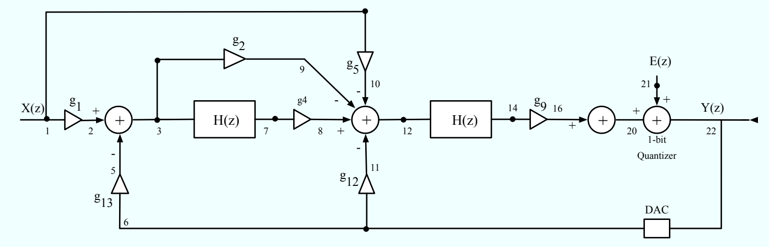


Figure 4.5. Second architecture.

In the architecture, values of the existing coefficient are

$$g_1 = 1,\ g_2 = -1,\ g_4 = 1,\ g_5 = 1,\ g_9 = 1,\ g_{12} = 1,\ g_{13} = 1. \tag{4.4}$$

Implementation of g_2 was done by taking difference since at the input of the first integrator only g_1 and g_13 exist and their values are equal. Simulations were done and PSD of the architecture is given in Figure 4.6. Signal is 42.32 dB and noise floor is at -8.5 dB. SNR is measured as 41.76 dB.

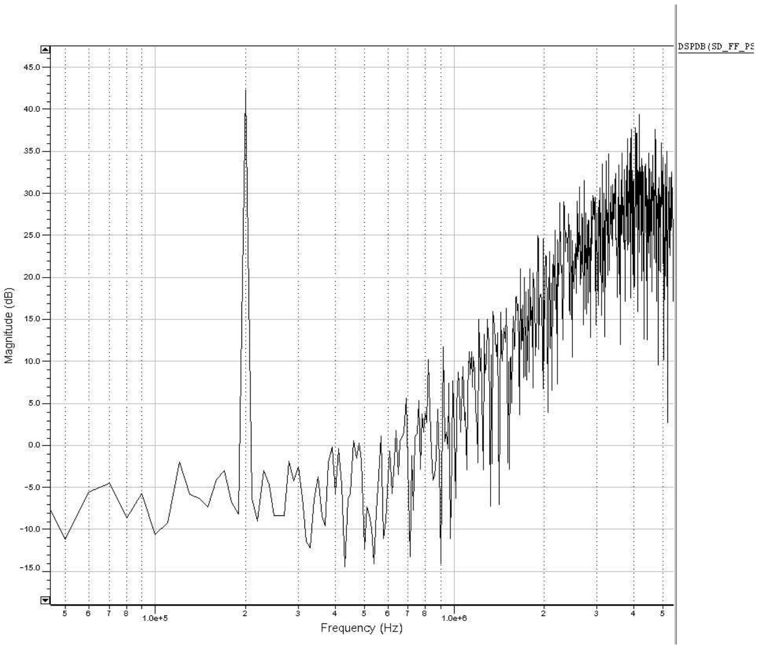


Figure 4.6. PSD of second architecture.

- **ARCHITECTURE–3**

Figure 4.7 shows the block diagram of the third architecture.

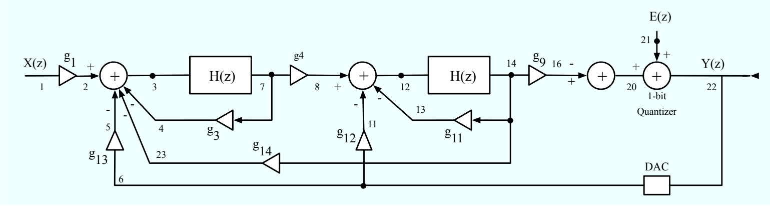


Figure 4.7. Third architecture.

In the architecture, values of the existing coefficient are

$$g_1 = 1,\ g_3 = 1,\ g_4 = 1,\ g_9 = 1,\ g_{11} = -1,\ g_{12} = 2,\ g_{13} = -1,\ g_{14} = 1. \quad (4.5)$$

Simulations were done and PSD of the architecture is given in Figure 4.8. Signal is 44.6 dB and noise floor is at -10 dB. SNR is measured as 44.46 dB.

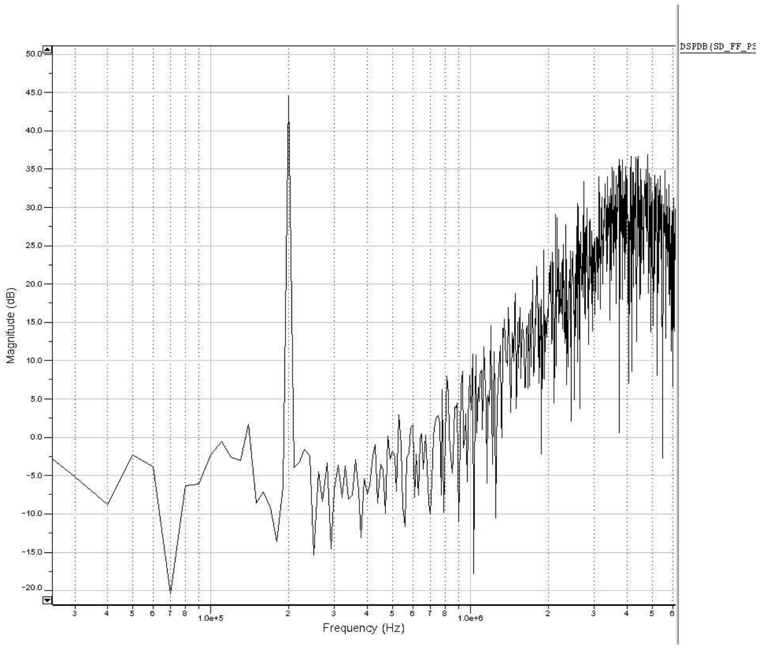


Figure 4.8. PSD of third architecture.

- **ARCHITECTURE–4**

  Figure 4.9 shows the block diagram of the fourth architecture.

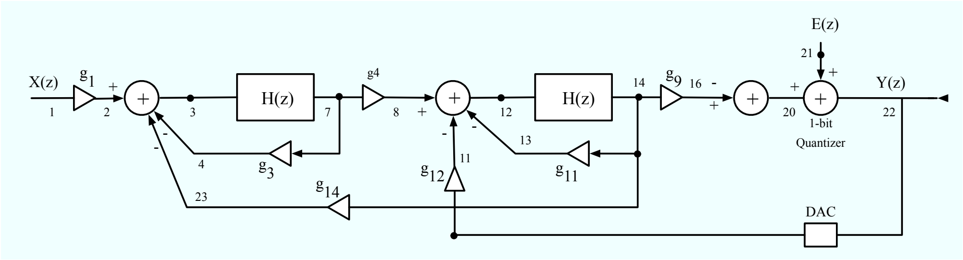


Figure 4.9. Fourth architecture.

In the architecture, values of the existing coefficient are

$$g_1 = 1,\ g_3 = 0.5,\ g_4 = 1,\ g_9 = 1,\ g_{11} = -0.5,\ g_{12} = 2,\ g_{14} = 0.25. \tag{4.6}$$

Simulations were done and PSD of the architecture is given in Figure 4.10. Signal is 43 dB and noise floor is at 5 dB. SNR is measured as 31.4 dB.

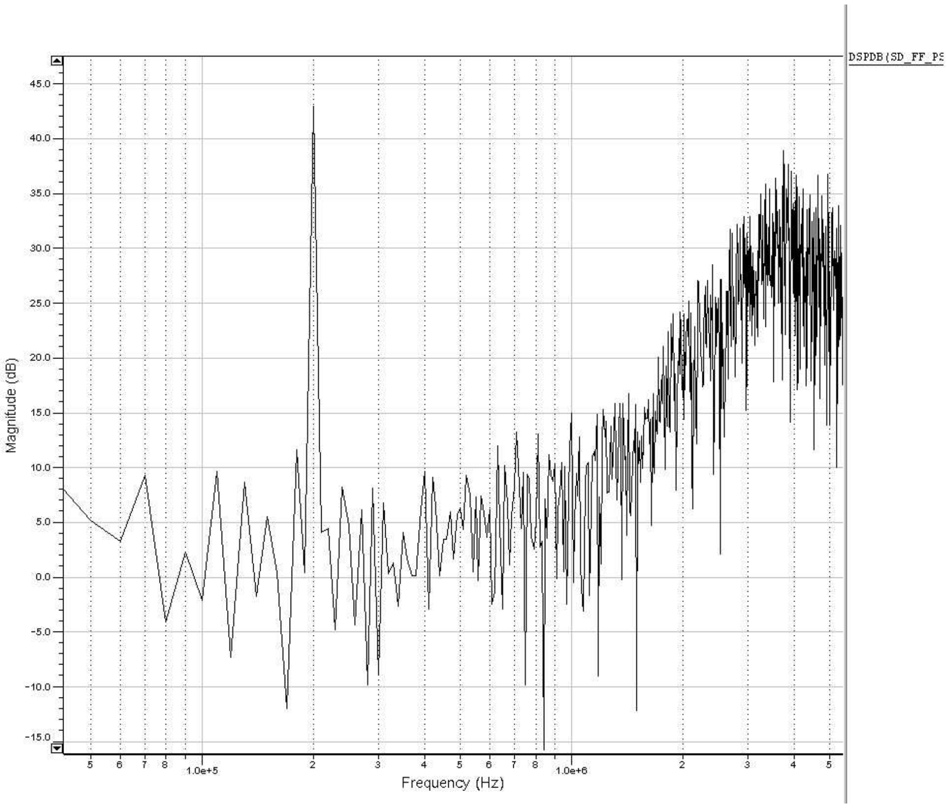


Figure 4.10. PSD of fourth architecture.

- **ARCHITECTURE–5**

  Figure 4.11 shows the block diagram of the fifth architecture.

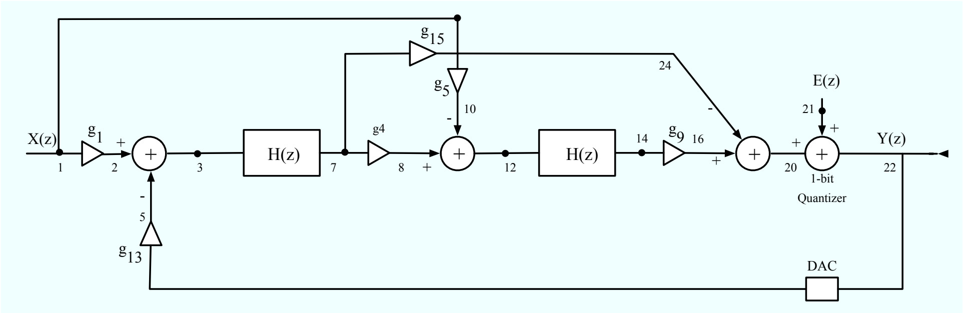


Figure 4.11. Fifth architecture.

In the architecture, values of the existing coefficient are

$$g_1 = 1,\ g_4 = 1,\ g_5 = 2,\ g_9 = 1,\ g_{13} = 1,\ g_{15} = -2. \tag{4.7}$$

Simulations were done and PSD of the architecture is given in Figure 4.12. Signal is 42 dB and noise floor is at -15 dB. SNR is measured as 43.72 dB.

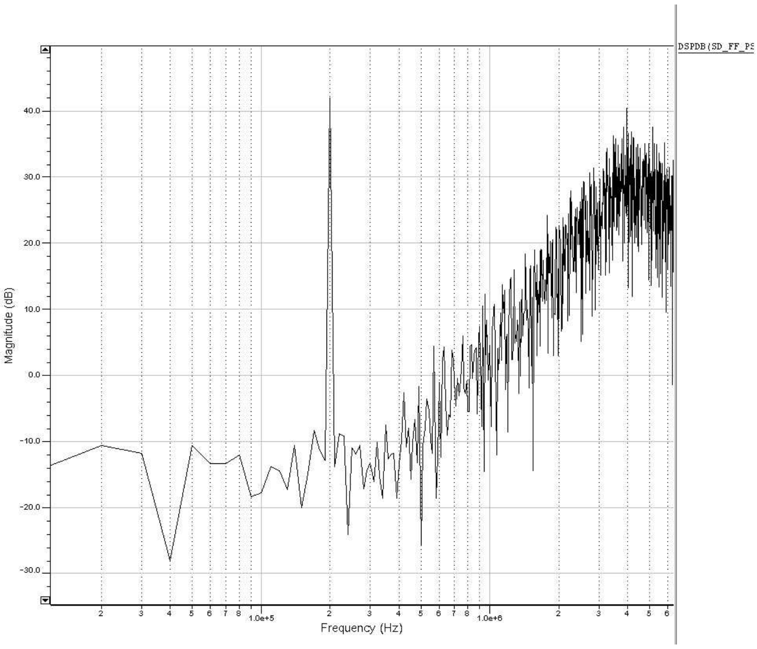


Figure 4.12. PSD of fifth architecture.

- **ARCHITECTURE–6**

Figure 4.13 shows the block diagram of the sixth architecture.

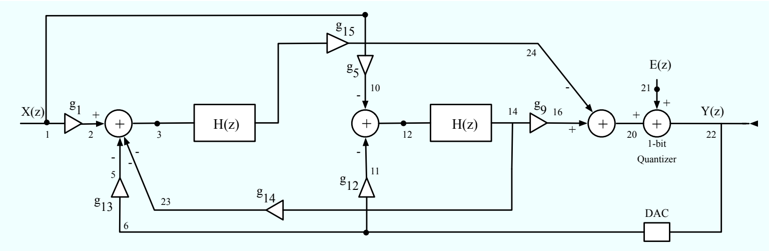


Figure 4.13. Sixth architecture.

In the architecture, values of the existing coefficient are

$$g_1 = 1,\ g_5 = -1,\ g_9 = 1,\ g_{12} = 1,\ g_{13} = -1,\ g_{14} = 1,\ g_{15} = 1. \tag{4.8}$$

In this architecture g_4 is not present, which makes this architecture an interesting topology. But implementation does not change. Simulations were done and PSD of the architecture is given in Figure 4.14. Signal is 42.2 dB and noise floor is at -20 dB. SNR is measured as 45.4 dB.

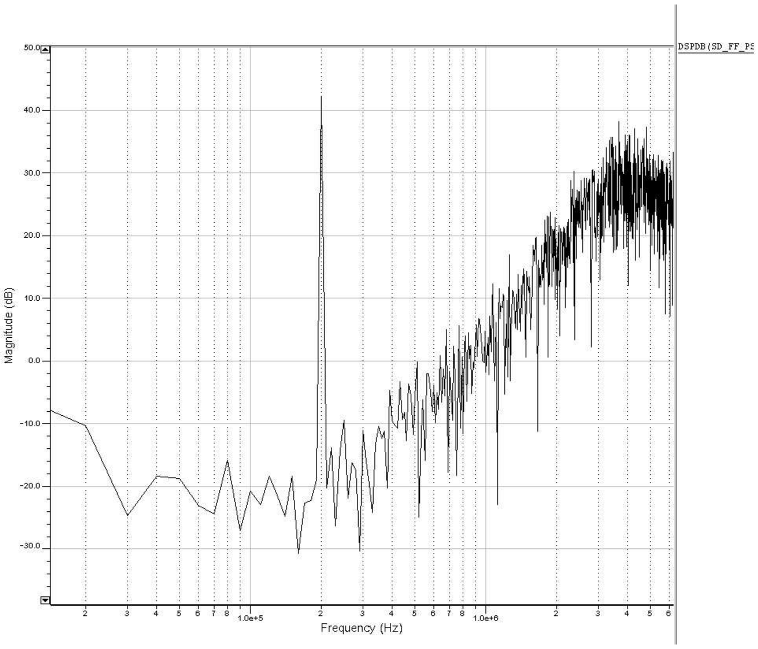


Figure 4.14. PSD of sixth architecture.

- **ARCHITECTURE–7**

Figure 4.15 shows the block diagram of the seventh architecture.

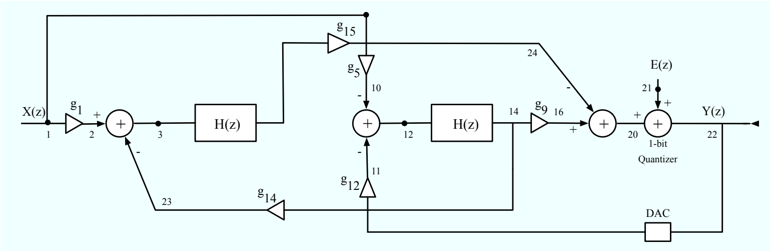


Figure 4.15. Seventh architecture.

In the architecture, values of the existing coefficient are

$$g_1 = 4,\ g_5 = -2,\ g_9 = 1,\ g_{12} = 2,\ g_{14} = 1,\ g_{15} = 0.5. \tag{4.9}$$

Simulations were done and PSD of the architecture is given in Figure 4.16. Signal is 43 dB and noise floor is at 0 dB. SNR is measured as 39.5 dB.

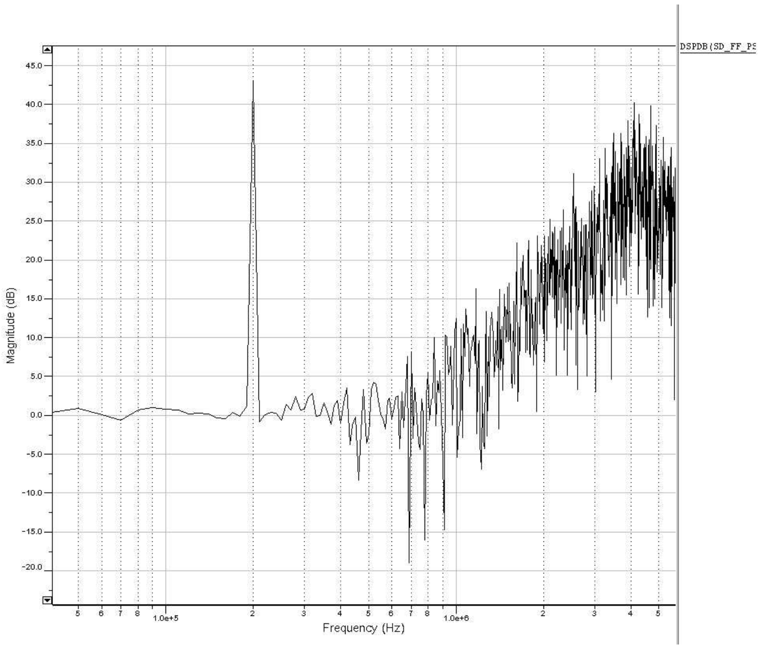


Figure 4.16. PSD of seventh architecture.

- **ARCHITECTURE–8**

Figure 4.17 shows the block diagram of the eighth architecture.

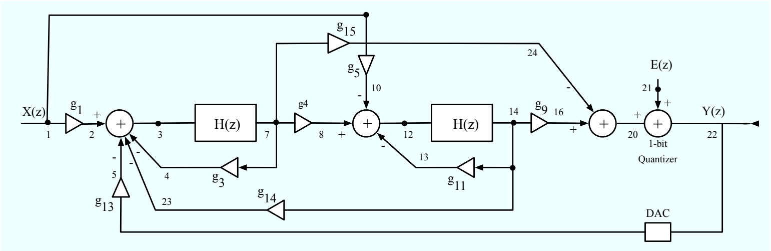


Figure 4.17. Eighth architecture.

In the architecture, values of the existing coefficient are

$$g_1 = 1, \quad g_3 = -1, \quad g_4 = 1, \quad g_5 = -2, \quad g_9 = 1, \quad g_{11} = 1, \quad g_{13} = -1, \\ g_{14} = 1, \quad g_{15} = 2. \tag{4.10}$$

Simulations were done and PSD of the architecture is given in Figure 4.18. Signal is 50.9 dB and noise floor is at -10 dB. SNR is measured as 49.6 dB.

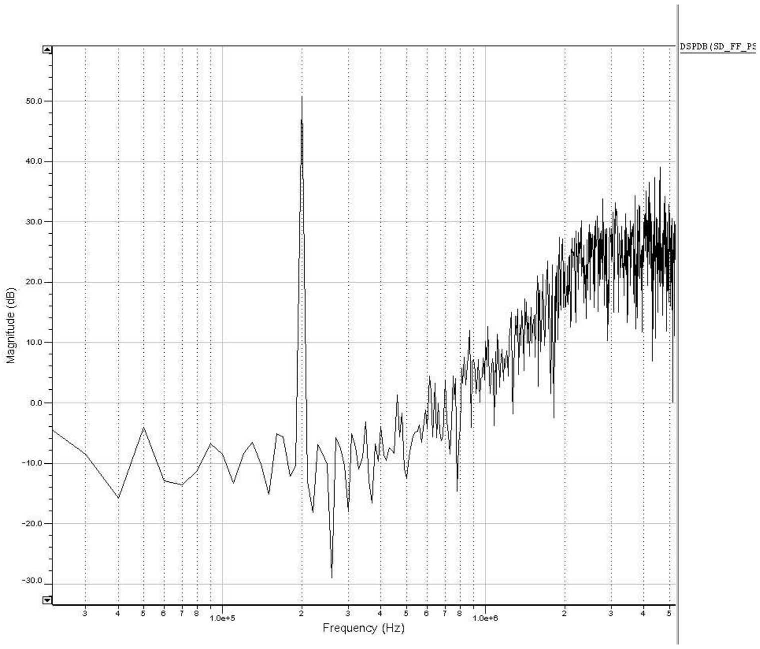


Figure 4.18. PSD of eighth architecture.

- **ARCHITECTURE–9**

Figure 4.19 shows the block diagram of the ninth architecture.

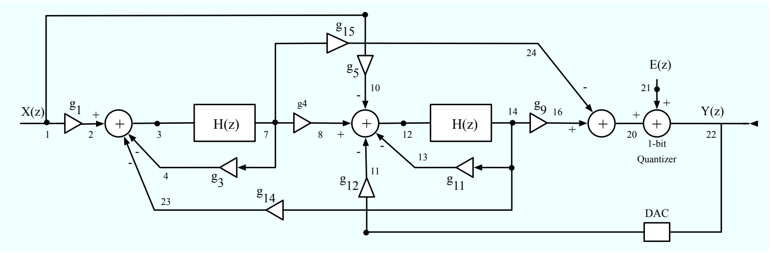


Figure 4.19. Ninth architecture.

In the architecture, values of the existing coefficient are

$$\begin{aligned} &g_1 = 2, \quad g_3 = 1, \quad g_4 = 2, \quad g_5 = 2, \quad g_9 = 1, \quad g_{11} = -1, \quad g_{12} = 2, \\ &g_{14} = 0.5, \quad g_{15} = -1. \end{aligned} \tag{4.11}$$

Simulations were done and PSD of the architecture is given in Figure 4.20. Signal is 52 dB and noise floor is at -5 dB. SNR is measured as 44.9 dB.

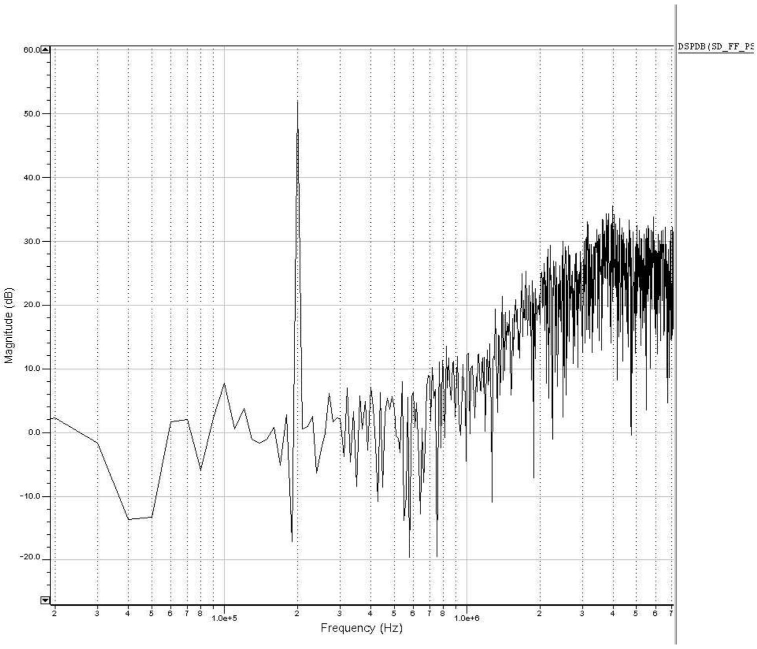


Figure 4.20. PSD of ninth architecture.

- **ARCHITECTURE–10**

  Figure 4.21 shows the block diagram of the 10^{th} architecture.

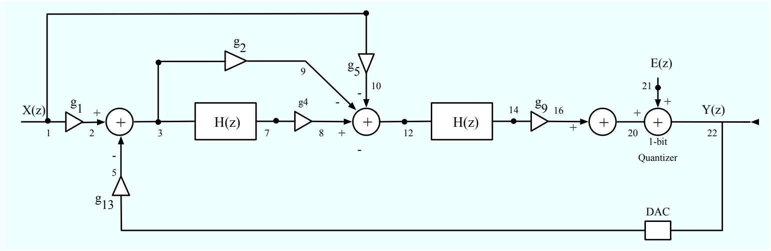


Figure 4.21. 10^{th} architecture.

In the architecture, values of the existing coefficient are

$$g_1 = 1,\ g_2 = -2,\ g_4 = 1,\ g_5 = 2,\ g_9 = 1,\ g_{13} = 1. \tag{4.12}$$

Simulations were done and PSD of the architecture is given in Figure 4.22. Signal is 42 dB and noise floor is at 0 dB. SNR is measured as 37.4 dB.

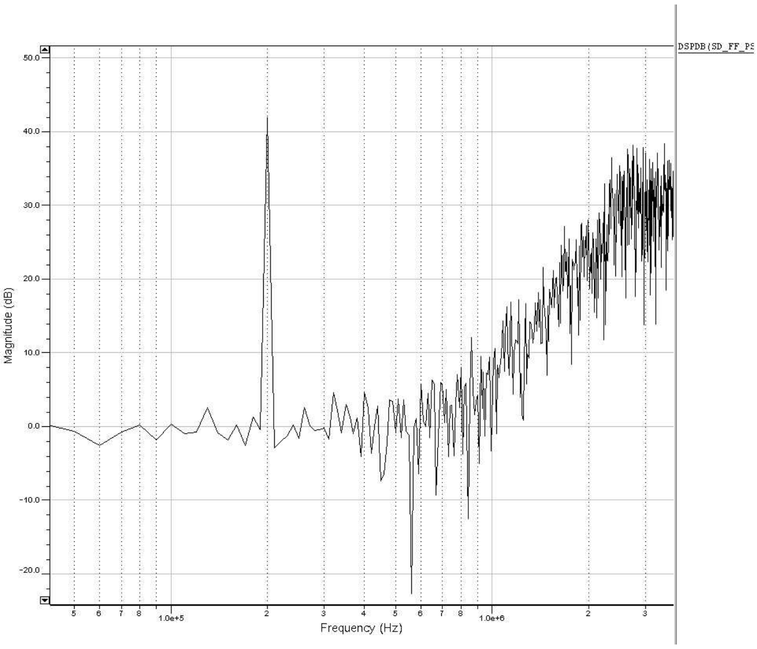


Figure 4.22. PSD of 10^{th} architecture.

- **ARCHITECTURE–11**

Figure 4.23 shows the block diagram of the 11^{th} architecture.

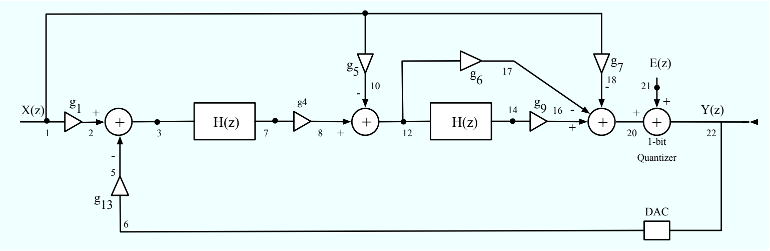


Figure 4.23. 11^{th} architecture.

In the architecture, values of the existing coefficient are

$$g_1 = 1,\ g_4 = 1,\ g_5 = 2,\ g_6 = -2,\ g_7 = -4,\ g_9 = 1,\ g_{13} = 1. \tag{4.13}$$

Implementation of g_6 is done by distributing its effect to g_4 and g_5. Simulations were done and PSD of the architecture is given in Figure 4.24. Signal is 42.15 dB and noise floor is at -10 dB. SNR is measured as 42.15 dB.

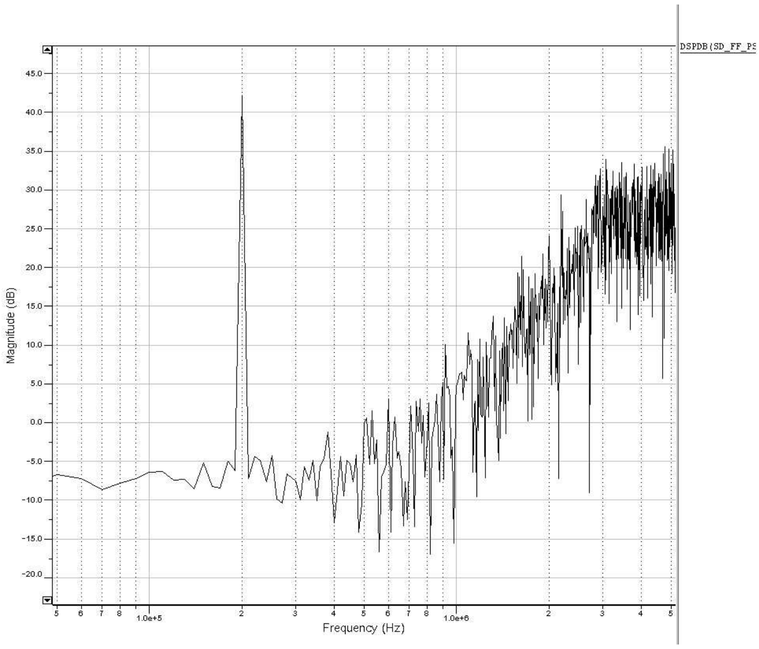


Figure 4.24. PSD of 11th architecture.

- **ARCHITECTURE–12**

Figure 4.25 shows the block diagram of the 12th architecture.

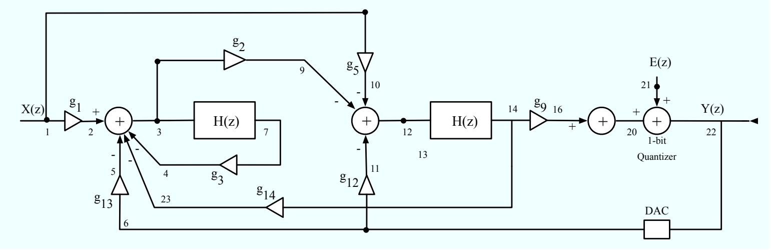


Figure 4.25. 12th architecture.

In the architecture, values of the existing coefficient are

$$g_1 = 1,\ g_2 = 1,\ g_3 = 1,\ g_5 = -1,\ g_9 = 1,\ g_{12} = 1,\ g_{13} = -1,\ g_{14} = 1. \quad (4.14)$$

Simulations were done and PSD of the architecture is given in Figure 4.26. Signal is 44.66 dB and noise floor is at -10 dB. SNR is measured as 44.16 dB.

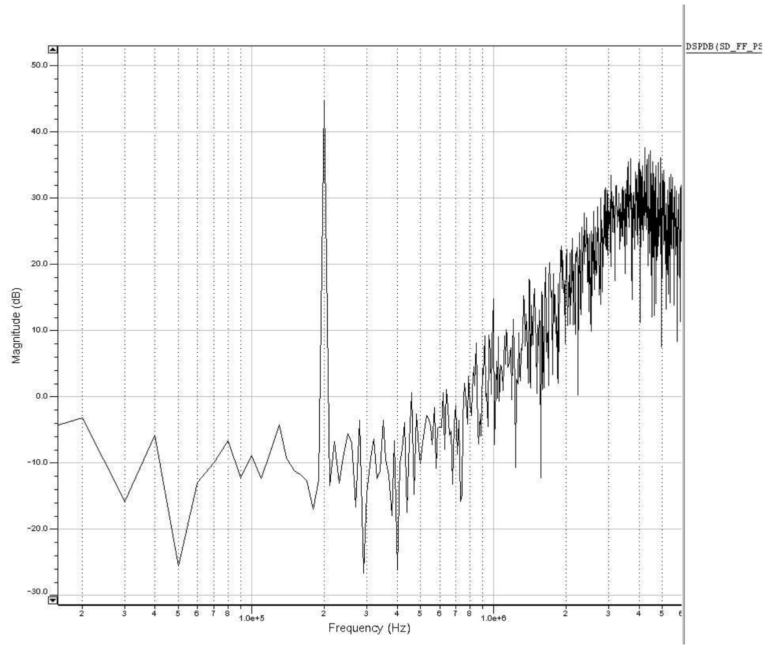


Figure 4.26. PSD of 12^{th} architecture.

Finally, all data related to the architectures are summarized in Table 4.1. As it can be observed from the table, eighth architecture has the highest signal and SNR value. Sixth architecture has the lowest noise floor value. Fourth architecture, which has no feedback path, has the lowest SNR value. Furthermore, it has the highest noise floor value.

Table 4.1. Signal, Noise Floor and SNR values of each architecture.

	Architectures												
	Std.	1	2	3	4	5	6	7	8	9	10	11	12
Signal(dB)	42.5	42.1	42.32	44.6	43	42	42.2	43	50.9	52	42	42.15	44.66
Noise Floor(dB)	-10	-4	-8.5	-10	5	-15	-20	0	-10	-5	0	-10	-10
SNR(dB)	42.8	39.9	41.76	44.46	31.4	43.72	45.4	39.5	49.6	44.9	37.4	42.15	44.16

5. PERFORMANCE CONSIDERATIONS

In Chapter 3, structures used in the design of ΣΔ modulator were introduced and implementation issues about them were discussed. In Chapter 4, some of the topologies, which were generated by an automation tool with the aim of realizing a predetermined STF and NTF were implemented. Their PSD graphs and SNR values were presented. After doing these, those different topologies which realize the same STF and NTF will be compared by means of some criteria; namely area occupied by the circuit, power consumed by the circuit and the sensitivity of the circuit performance to the changes.

5.1. Area

In integrated circuit design, area is an important criterion for comparing different implementations since less die area consumption is anyway a beneficial property. If the case of same order ΣΔ modulators realizing same transfer functions is to be discussed, the parameter to compare becomes the area occupied by the coefficient implementation circuitry. The reason is that, same modulator order implies the same number of integrators meaning the same number of operational amplifiers. In addition to the operational amplifiers, number of quantizers and DACs will be same. Thus, the important parameter is the area occupied by switched capacitor circuitry. Sum of the values of the capacitors used in the implementation gives a variable to compare different topologies. Calculated values of this variable for the implemented topologies are tabulated as follows:

Table 5.1. Capacitor values used in implementation for each architecture

	Architectures												
	Std.	1	2	3	4	5	6	7	8	9	10	11	12
Capacitance (pF)	16	18	18	22	16.5	20	18	25	26	29	20	36	22

5.2. Sensitivity

When performance of a $\Sigma\Delta$ modulator is considered, SNR can be a parameter of interest. In order to describe a modulator as a high performance one, its SNR value must be better than most of the others. The relation governing the SNR of a modulator was given in Equation (2.7). P_Q, which is the in band noise power, has the formula

$$P_Q = \int_{-f_d/2}^{f_d/2} S_Q(f)df = \int_{-f_d/2}^{f_d/2} S_E(f)|NTF(z)|^2 df \cong \frac{\Delta^2}{12}\frac{\pi^2}{3M^3}. \tag{5.1}$$

In this formula, $S_Q(f)$ is the power spectral density of the shaped quantization noise, whereas $S_E(f)$ is the PSD of unshaped quantization noise. Also, Δ is the quantization step and M is the oversampling ratio. Since PSD of the unshaped quantization is generally considered as uniform distributed white noise, the parameter that affects the SNR appears to be $|NTF(z)|^2$ term. The parameters that make the topologies different are the coefficients and their values. All in all, sensitivities of the SNRs of different architectures to the path gains is the parameter to dealt with. Since SNR is directly related to $|NTF(z)|^2$, sensitivity of this term to the path gains is important in comparing the sensitivities of SNRs of the implemented architectures.

There are 15 coefficients in the second order $\Sigma\Delta$ modulator. The tool in [11, 12] gives NTF of the second order modulator in parametric form of those coefficients. So, sensitivity of $|NTF(z)|^2$ to each coefficient can be calculated. The classical sensitivity definition is

$$S_x^y = \frac{x}{y}\frac{\delta y}{\delta x}. \tag{5.2}$$

In our case y would be $|NTF(z)|^2$ and x would be g_i, i changing from 1 to 15. However, the problem is that $|NTF(z)|^2$ can be 0 for some cases which results in infinite sensitivity, As a result there can be no way to look at sensitivity. Hence, a different definition of sensitivity which is called *Semi-relative Sensitivity* is applied, which is discussed in [20] further. The equation for this sensitivity definition is

$$Q_x^{P(x)} = x\frac{dP}{dx}. \tag{5.3}$$

By using Equation (5.3), sensitivity for the cases when $P(x) = 0$ can be accounted for. So $Q_{g_i}^{|NTF(z)|^2}$ will be the parameter of interest, which has the following equation:

$$Q_{g_i}^{|NTF(z)|^2} = g_i\frac{\delta|NTF(z)|^2}{\delta g_i}. \tag{5.4}$$

Having semi-relative sensitivity metric in hand, following steps are followed to reach a value that gives an idea about the sensitivity of architecture's SNR to the path gains:

i. Semi relative sensitivities of $|NTF(z)|^2$ with respect to each coefficient is calculated. End result of this step is same for every architecture since NTF is parametric.

ii. Gain values that are present in a definite topology are substituted to the result of previous step. Resulting relation is only a function of z.

iii. z is substituted with e^{jw}.

iv. L_2 distance approach is applied in order to calculate a comparable quantity for the sensitivities of each architecture. Taking the reference as zero, the following integral is calculated for each sensitivity expression for coefficients g_1 to g_{15}:

$$L_{2g_i} = \left[\int_{-\pi}^{\pi} |Q_{g_i}^{|NTF|^2}(w)|^2 dw\right]^{1/2}. \tag{5.5}$$

v. Since the number of parameters on which the overall architecture sensitivity depends is 15, it can be assumed that calculation is done in a 15–dimensional space. Then each value L_{2g_i} can be assumed as a vector in each dimension and the overall change (gradient) of these values can be used as an overall sensitivity parameter for each architecture which is calculated with formula

$$\sqrt{\sum_{i=1}^{15} {L_{2_{g_i}}}^2}. \tag{5.6}$$

With the above approach a code was written in MATLAB to find overall sensitivity parameter for each of the 13 architectures. The code is given in Appendix A. Just to be used as information, the same procedure was followed for STF. Below NTF and STF sensitivities of each architecture is tabularized:

Table 5.2. STF and NTF sensitivities of each architecture

	Architectures												
	Std.	1	2	3	4	5	6	7	8	9	10	11	12
STF	16.2	17.7	15.8	15.4	22.6	27.7	17.7	27.7	41.8	41.8	23.8	66.2	16.6
NTF	13.7	9.4	11.8	34.2	28	13.7	9.4	13.7	34.2	34.2	15.4	15.4	35.1

As it can be observed from Table 5.2, 12^{th} architecture has the highest sensitivity to the coefficients. It should be noted that in addition to the sensitivity of the overall architecture, sensitivity of the NTF to each gain path is also in hand. So, knowing path with the highest sensitivity, it is possible to see the effect of change in a gain term to SNR value. As an example to see this effect, SIMULINK simulations have been done for 12^{th} architecture using models for each block. For 12^{th} architecture, g_{14} has the highest sensitivity value, so effect of a change in that coefficient was observed. For a 10 percent change in g_{14}, SNR decreased from 90 dB to 74 dB. Also in order to see the success of approach, effect of the change in a gain for an architecture with low sensitivity has been observed. For this aim, sixth architecture has been chosen since it has the lowest sensitivity. g_5 which has the lowest sensitivity value has been selected for the path. When its value is changed 10 percent SNR value decreased from 100 dB to 98.9 dB, which is a change of only 1.1 dB. As it can be observed, change is small which proves that approach is consistent.

5.3. Power Consumption

In design of circuits, power consumption is an important concern. Following this idea, power consumed by a $\Sigma\Delta$ modulator is an important parameter, when comparison is to be done between different topologies realizing same transfer function. Firstly, it should be noted that by simulations with the implemented architectures it has been seen that dynamic power is negligible with respect to the static power consumed by the modulator. This has been observed by comparing the operational amplifier current charging the feedback capacitor with the branch currents of operational amplifier, since the former is much smaller than the latter. So, if power consumption of the different topologies is considered, static power consumption should be the parameter of interest. In a $\Sigma\Delta$ modulator, most of the power is consumed by integrator, which makes the power consumed by integrator important in power consideration. A SC integrator includes operational amplifier, capacitors and switches. High portion of this power is due to operational amplifier. Therefore, $\Sigma\Delta$ modulator power is highly correlated with the operational amplifier power. Operational amplifier used in the design of integrator is shown in Figure 3.7. If the current flowing through the single output stage is I_{branch} and the current supplied to the single input differential stage is I, power consumed by the operational amplifier is

$$P = (4I_{branch} + 2I)V_{DD}. \tag{5.7}$$

For the operational amplifier in hand, $I_{branch} = 0.71I$. So power of the operational amplifier is $P = 6.8V_{DD}I_{branch}$. I has the equation

$$I = \frac{g_{meff}V_{ov}}{2}, \tag{5.8}$$

where g_{meff} is the transconductance of the differential amplifier's single transistor having the equation [22]

$$g_{meff} = Bln(2)C_{L,eff}2f_s, \tag{5.9}$$

where f_s is the sampling frequency, $C_{L,eff}$ is the effective load capacitance, B is the number of bits resolution. $C_{L,eff}$ is defined as a multiple,N_C, of the sampling capacitor, C_s. N_C is called excess capacitance factor and has the equation

$$N_C = 1 + \frac{(C_L)(C_s + C_I)}{C_s C_I}, \tag{5.10}$$

where C_L is the load capacitance, C_I is the feedback capacitance. Only variable left is B which is

$$B = log_2(\sqrt{\frac{2}{3}DR^2} + 1), \tag{5.11}$$

where dynamic range (DR) is $DR^2 = \frac{P_S}{P_N}$, in which P_S is the maximum signal power for a sinusoidal signal having equation ${V_s}^2/2$. P_N is the noise power, which is $\Delta^2/12$, Δ being the step size of the quantizer.

A MATLAB code has been written for finding the power for each architecture. For this aim, load, sampling and input capacitances for each topology have been found. Also it should be noted that $DR^2 = P_S/P_N$ formula is defined for cases where input directly enters the integrator. For cases where input signal is scaled by a transfer function of $H(z)$ and noise signal is scaled by transfer function $G(z)$, following holds:

$$DR^2 = \frac{|H(z)|^2 P_S}{\int |G(z)|^2 dz P_N}. \tag{5.12}$$

This transfer function is defined as the ratio of the signal at integrator input to the original signal which holds for both noise signal and modulator input signal. Since there exists two integrators there exist $H(z)$ and $G(z)$ for each one. So flow for calculation of the power for each architecture is as follows:

i. $H_1(z), H_2(z), G_1(z), G_2(z)$ are calculated and $z = e^{jw}$ is substituted.

ii. In $H_1(w), H_2(w)$, $w = 2\pi f_{signal}$ is substituted since signal is just one tone with frequency f_{signal}.

iii. Since noise exists at every frequency for quantization noise following integral is taken:

$$\int_{-0.04\pi}^{0.04\pi} |G(w)|^2 dw. \tag{5.13}$$

Reason is that our signal is at 200 kHz and sampling frequency is at 20 MHz. 0.04π corresponds to 400 kHz and reason for this choice is that twice the signal frequency has been chosen as the band of interest. This approach is similar to the case of calculating SNR, where 2.5 times signal frequency is used.

iv. DR^2 is calculated by Equation 5.12. Note that $V_s = 200mV$ and $\Delta = 3.3V$ for our case.

v. Then, the code calculates $C_{L,eff}$ value using the capacitor values, which are input,load and sampling capacitances.

vi. Using above calculated values tool calculates I with values $V_{ov} = 0.216, f_s = 20MHz$.

vii. Finally power is calculated.

Capacitance values used in the calculation are tabularized below:

Table 5.3. Load capacitances of each architecture

	Architectures												
	Std.	1	2	3	4	5	6	7	8	9	10	11	12
INT1(pF)	1	2	1	2	1.5	3	1	0.5	4	4	1	3	2
INT2(pF)	1	1	1	3	2.5	1	2	2	3	2.5	1	1	3

Table 5.4. Sampling capacitances of each architecture

	Architectures												
	Std.	1	2	3	4	5	6	7	8	9	10	11	12
INT1(pF)	2	2	2	4	2.5	2	3	5	4	3.5	2	2	4
INT2(pF)	3	3	4	4	3.5	3	2	4	4	7	5	3	4

It should be noted that C_I values are all 1 pF. After all calculated power values have been found, which are tabularized below:

Table 5.5. Power Values

	Architectures												
	Std.	1	2	3	4	5	6	7	8	9	10	11	12
Power(mW)	0.8	1.2	0.83	0.58	0.37	1.79	1.2	2.24	1.26	1.42	0.87	1.79	0.58

It is observed from Table 5.3 that fourth architecture has the lowest power consumption value, whereas seventh one has the highest. To see the effect of the metric devised, these two architectures have been used. Method used to see the effect was to increase all the capacitances in the implementation to their multiple values. Reason for using such a method is that increasing capacitances result in increase in current. Since an operational amplifier, which can source a current up to a predefined level, is used, after a limit value of capacitance increase malfunctioning of the circuit starts. When this method has been applied to those two architectures, it has been observed that seventh started malfunctioning after capacitance values have been doubled. However for the case of fourth architecture, circuit has worked exactly with the same performance when capacitances have been increased to three times their original values. Also it has been observed that somehow it has continued working for five times the original values. This somewhat proves that the approach holds.

6. CONCLUSIONS

Analog to Digital Conversion is an important process in electronic systems, because of the reasons discussed in Chapter 1. There are two conversion methods according to the sampling frequency used in the process. Being one of those methods, oversampling converters achieve high resolution by trading bandwidth for it. $\Sigma\Delta$ converter, which is a special type of oversampling converters use $\Sigma\Delta$ modulator in order to shape noise in a better way.

$\Sigma\Delta$ modulator includes loop filter before quantizer. The order of this filter determines the order of the modulator and increase in order gives better performance in terms of SNR values. Also for better performance, some paths with gains are introduced to the modulator architecture. Those paths are either feedback or feedforward ones. They improve the performance, however they complicates the design process, since for a second order achitecture including all the possible paths results in an architecture with 15 paths. At that point, the need for determining the paths to be used and finding the values of those coefficients arises.

In this work, finding the values of those coefficients has been achieved by using the tool in [11, 12]. Solution set in hand has been realizing a standard second order $\Sigma\Delta$ modulator response. However, the solution set has been so large that extra metrics have been needed in order to choose the optimum topology from the solution set. Metrics to be developed have been defined as area, power consumption and SNR sensitivity of the architecture to the path gains. Approach taken for developing those metrics has been implementation and simulation of somewhat different architectures. Then, theoretical approach has been developed for each metric to support the data in hand. Firstly, implementation has been the issue to deal with. SC implementation has been chosen since VLSI is compatible with SC circuit technique. Methods to implement both circuitry and coefficients have been devised and 13 different architectures including the standard architecture have been implemented.

Following the implementation, performance considerations have been the issue of concern. The property that differs in the second order topologies implemented is the paths employed in the topology and their values. So difference between the areas of the topologies is due to the paths and their values. As a second metric SNR sensitivity to path gains has been focused on. Theoretical approach has shown that SNR sensitivity to gains went in parallel with the sensitivity of $|NTF(z)|^2$ to path gains. Sensitivities of each architecture have been calculated using semi–relative sensitivity approach. Applicability of the approach has been verified by doing simulations on SIMULINK using models for each block in $\Sigma\Delta$ modulator. Finally power consumption of the modulator has been investgated. It has been observed that dynamic power consumption of the modulator is negligible with respect to the static power consumption. For the static power consumption, power consumed by the operational amplifiers has been focus point since in literature integrator power is claimed to be the dominant power in a $\Sigma\Delta$ modulator. A theoretical approach from [22] has been applied to this case. Following the calculation of the power using MATLAB code, architectures with minimum and maximum power consumption have been compared by using the approach of increasing all the capacitances to their multiples. Approach proved to be useful.

Having all the implementations and metrics in hand, it has been interesting to observe that all the metrics can be formulated in terms of path gains. This fact can be used as a starting point for a future work. An optimization problem can be formulated in order to maximize a cost function satisfying certain specifications. The tool in [11, 12] can be further upgraded to solve that optimization problem resulting in the best topology considering performance issues.

APPENDIX A: MATLAB Code written for sensitivity analysis

This is the code for calculation of NTF sensitivity.

```
modify_14;
stf = x22/x1;
ntf = x22/x21;

clear s ss l
l_total = 0;
l_total_ntf = 0;

gains = top

syms E IN w real
syms z
syms g1 g2 g3 g4 g5 g6 g7 g8 g9 g10 g11 g12 g13 g14 g15 real;

j = sqrt(-1);

stf2 = subs(stf, E, 0);
ntf2 = subs(ntf, IN, 0);
stf2 = collect(simplify(stf2));
ntf2 = collect(simplify(ntf2));

[numstf denstf] = numden(stf2);
[numntf denntf] = numden(ntf2);

n = subs(numntf, z, exp(j*w));
d = subs(denntf, z, exp(j*w));
```

```
n2 = n*conj(n);
d2 = d*conj(d);

for k = 1 : 15
    eval([’s.g’ num2str(k) ’= (diff(n2, g’ num2str(k) ’)*
g’ num2str(k) ’) - (diff(d2, g’ num2str(k) ’)*
g’ num2str(k) ’);’])
    eval([’ssub.g’ num2str(k) ’= subs(s.g’ num2str(k) ’,
{g1, g2, g3, g4, g5, g6, g7, g8, g9, g10, g11,
 g12, g13, g14, g15}, gains);’])
    eval([’ss.g’ num2str(k) ’= ssub.g’ num2str(k) ’*
conj(ssub.g’ num2str(k) ’);’])
    eval([’l.g’ num2str(k) ’ = double(sqrt(int(ss.g’ num2str(k) ’
, w, -pi, pi)))’])
end

 for k = 1 : 15
     eval([’l_total = l_total + l.g’ num2str(k) ’^2;’])
 end
 l_total_ntf = sqrt(l_total)
```

This is the code for calculation of STF sensitivity.

```
modify_14
stf = x22/x1;
ntf = x22/x21;

clear s ss l

l_total = 0;
l_total_stf = 0;
gains = top
```

```
syms E IN w real
syms z
syms g1 g2 g3 g4 g5 g6 g7 g8 g9 g10 g11 g12 g13 g14 g15 real;

j = sqrt(-1);

stf2 = subs(stf, E, 0)
ntf2 = subs(ntf, IN, 0)
stf2 = collect(simplify(stf2))
ntf2 = collect(simplify(ntf2))

[numstf denstf] = numden(stf2);
[numntf denntf] = numden(ntf2);

n = subs(numstf, z, exp(j*w));
d = subs(denstf, z, exp(j*w));

n2 = n*conj(n);
d2 = d*conj(d);

for k = 1 : 15
    eval(['s.g' num2str(k) '= (diff(n2, g' num2str(k) ')
*g' num2str(k) ') - (diff(d2, g' num2str(k) ')
*g' num2str(k) ');'])
    eval(['ssub.g' num2str(k) '= subs(s.g' num2str(k) ',
{g1, g2, g3, g4, g5, g6, g7, g8, g9, g10, g11, g12, g13,
g14, g15}, gains);'])
    eval(['ss.g' num2str(k) '= ssub.g' num2str(k) '
*conj(ssub.g' num2str(k) ');'])
    eval(['l.g' num2str(k) ' = double(
sqrt(int(ss.g' num2str(k) ', w, -pi, pi)))'])
```

```
end


for k = 1 : 15
    eval(['l_total = l_total + l.g' num2str(k) '^2;'])
end
l_total_stf = sqrt(l_total)
```

APPENDIX B: MATLAB Code written for power analysis

```
syms w real

h1 = x3/x1;
h2 = x12/x1;

n1 = x3/x21;
n2 = x12/x21;

h1 = subs(h1, E, 0);
h1 = simplify(collect(h1));

h2 = subs(h2, E,0);
h2 = simplify(collect(h2));

n1 = subs(n1, IN, 0);
n1 = simplify(collect(n1));

n2 = subs(n2, IN, 0);
n2 = simplify(collect(n2));

for i = 1 : 13
    h1_top = subs(h1, ’g1, g2, g3, g4, g5, g6, g7, g8, g9, g10,
g11, g12, g13, g14, g15’, topo{i});
    h1_top = simplify(collect(h1_top));
    h1_z{i} = h1_top;
    h1_top = subs(h1_top, z, exp(j*w));
    h1_abs{i} = abs(h1_top);
    h1_abs{i} = h1_abs{i}^2;
    h1_abs{i} = subs(h1_abs{i}, w, 0.02*pi);
    h2_top = subs(h2, ’g1, g2, g3, g4, g5, g6, g7, g8, g9, g10,
g11, g12, g13, g14, g15’, topo{i});
    h2_top = simplify(collect(h2_top));
    h2_z{i} = h2_top;
    h2_top = subs(h2_top, z, exp(j*w));
    h2_abs{i} = abs(h2_top);
    h2_abs{i} = h2_abs{i}^2;
    h2_abs{i} = subs(h2_abs{i}, w, 0.02*pi);

    n1_top = subs(n1, ’g1, g2, g3, g4, g5, g6, g7, g8, g9, g10,
g11, g12, g13, g14, g15’, topo{i});
    n1_top = simplify(collect(n1_top));
    n1_z{i} = n1_top;
    n1_top = subs(n1_top, z, exp(j*w));
    n1_abs{i} = abs(n1_top);
    n1_abs{i} = n1_abs{i}^2;
    n1_abs{i} = int(n1_abs{i}, w, 0, 0.04*pi);

    n2_top = subs(n2, ’g1, g2, g3, g4, g5, g6, g7, g8, g9, g10,
g11, g12, g13, g14, g15’, topo{i});
    n2_top = simplify(collect(n2_top));
    n2_z{i} = n2_top
    n2_top = subs(n2_top, z, exp(j*w));
    n2_abs{i} = abs(n2_top);
    n2_abs{i} = n2_abs{i}^2;
    n2_abs{i} = int(n2_abs{i}, w, 0, 0.04*pi);

    p_topo(i) = double(p_disp(vov, h1_abs{i}, n1_abs{i}, amp,
delta, cl_1(i), ci_1(i), cs_1(i), fs) + p_disp(vov, h2_abs{i},
n2_abs{i}, amp, delta, cl_2(i), ci_2(i), cs_2(i), fs));
```

```
end
```

This code invokes following code piece for calculation of power:

```
function pow = p_disp(vov, h_abs, n_abs, amp, delta, cl, ci, cs, fs)


       ps = amp^2/2;
       pn = delta^2/12;


       dr_sq = (h_abs*ps)/(n_abs*pn);


       nc = 1 + (cl*(cs + ci)/(cs*ci));
       cl_eff = nc*cs;


       current = 0.71*vov*log2(sqrt(2/3*dr_sq) + 1)*log(2)*cl_eff*fs;


       pow = 3.3*6.8*current;

end
```

REFERENCES

1. Aziz, P., H. Sorensen and J. Spiegel, ”An Overview of Sigma–Delta Converters”, *IEEE Signal Processing Magazine,* pp. 61–84, Jan. 2006.

2. Ismail, M. and T. Fiez, *Analog VLSI Signal and Information Processing*, McGraw–Hill Series in Electrical and Computer Engineering, 1994.

3. Liu, M., *Demystifying Switched–Capacitor Circuits*, Newnes, 2006.

4. Unbehauen, R. and A. Cichocki, *MOS Switched–Capacitor and Continuous–Time Integrated Circuits and Systems*, Springer–Verlag, 1989.

5. Rabii, S. and B. Wooley, *The Design of Low–Voltage, Low–Power Sigma–Delta Modulators*, Kluwer Academic Publishers, 1999.

6. Medeiro, F., A. Perez and A. Vazquez, *Top–Down Design of High–Performance Sigma–Delta Modulators*, Kluwer Academic Publishers, 1999.

7. Norsworthy, S., R. Schreier and G. Temes, *Delta–Sigma Data Converters : Theory, Design, and Simulation* , IEEE Circuit & Systems Society IEEE Press, 1997.

8. Inose, H., Y. Yasuda and J. Murakami, ”A Telemetering System by Code Manipulation – Modulation”, *IRE Trans on Space Electronics and Telemetry,* Sep. 1962, pp. 204-209.

9. $http : //en.wikipedia.org/wiki/Delta - sigma\_modulation$

10. *http://www.analog.com/en/content/0,2886,760%255F%255F92393,00.html*

11. Yetik, O., O. Saglamdemir, S. Talay and G. Dundar, ”A Coefficient Optimization and Architecture Selection Tool for $\Sigma - \Delta$ Modulators in MATLAB”, Design, Automation & Test in Europe Conference & Exhibition, 2007. DATE ’07 , vol.,

no., pp.1-6, 16-20 April 2007.

12. Saglamdemir, O., O. Yetik, S. Talay and G. Dundar, "A coefficient optimization and architecture selection tool for SD modulators considering component nonidealities", *In Proceedings of the 17th Great Lakes Symposium on Great Lakes Symposium on VLSI (Stresa-Lago Maggiore, Italy, March 11 - 13, 2007). GLSVLSI '07.* ACM Press, New York, NY, 423-428.

13. Tang, H. and A. Doboli, "High-level synthesis of $\Delta\Sigma$ Modulator topologies optimized for complexity, sensitivity, and power consumption", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol.25, no.3, pp. 597-607, March 2006.

14. Bennett, W. R., "Spectra of quantized signals," *Bell Syst. Tech. J.,* vol. 27, pp. 446–472, July 1948.

15. Candy, J. C. and G. C. Temes, *Oversampling Delta–Sigma Converters*, IEEE Press, 1992.

16. Gray, R. M., "Quantization noise spectra," *IEEE Transactions on Information Theory*, vol.36, no.6, pp.1220-1244, Nov 1990.

17. *http://en.wikipedia.org/wiki/Image:DeltaSigmaNoise.svg*

18. Franco, S., *Design with Operational Amplifiers and Analog Integrated Circuits*, McGraw Hill, 2002.

19. Gregorian, R., K. W. Martin and G. C. Temes, "Switched-capacitor circuit design", *Proceedings of the IEEE,* vol.71, no.8, pp. 941-966, Aug. 1983.

20. Schaumann, R., M. Ghausi and K. Laker, *Design of Analog Filters*, Prentice Hall Series in Electrical and Computer Engineering, 1990.

21. Sedra, A. S. and K. C. Smith: *"Microelectronic Circuits* 4_{th} *Edition"*, Oxford

University Press, New York, 1998.

22. Peluso, V., M. Steyaert and W. Sansen, *Design of Low–Voltage Low–Power CMOS Delta–Sigma A/D Converters*, Kluwer Academic Publishers, 1999.