Isacco Arnaldi

Design of Sigma-Delta Converters in MATLAB®/ Simulink®

Springer

Design of Sigma-Delta Converters in MATLAB®/Simulink®

Isacco Arnaldi

Design of Sigma-Delta Converters in MATLAB®/ Simulink®

Springer

Isacco Arnaldi
Caldogno, Italy

Additional material to this book can be downloaded from http://extras.springer.com.

ISBN 978-3-319-91538-8 ISBN 978-3-319-91539-5 (eBook)
https://doi.org/10.1007/978-3-319-91539-5

Library of Congress Control Number: 2018943124

© Springer International Publishing AG, part of Springer Nature 2019
This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.
The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
The publisher, the authors, and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by Springer Nature, under the registered company
Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

To Soardi Giovanna and Arnaldi Eugenio

Preface

The design of Analog-to-Digital Sigma-Delta Data Converters involves engineers who possess the knowledge, intuition, creativity, and technical expertise to turn their ideas into real-life devices. Schreier and Temes' excellent 2005 IEEE. Press book *Understanding Delta-Sigma Data Converters* deals with a comprehensive collection of important subjects relative to Sigma-Delta modulator design. I have attempted to make this monograph complementary to their book: using graphical behavioral simulation tools, the essential theoretical concepts are demonstrated while underlying a systematic approach to the design process. Although this book should provide enough information to be understood on its own, I do not hesitate to recommend *Understanding Delta-Sigma Data Converters* as a source of additional information; I will make frequent reference to it. It is my hope that this monograph will provide you with an intuitive understanding of the theoretical and practical aspects of Sigma-Delta modulators in a way that fuels your creativity.

What Makes This Book Unique

A number of Sigma-Delta modulators books seem to throw plenty of technical formulas and theory at the reader from the start before ever giving an idea of what the design process involves, the trade-offs to consider, how a modulator is actually simulated, and how to consider a specific design successful. If practical information is present, it is often toward the conclusion of the chapter, and by this time, the reader may have totally lost interest in the subject or may have missed the "big picture," confused by details and formulas. The *Design of Sigma-Delta Converters in MATLAB®/Simulink®* does not have this effect on the reader. It aims at answering many of the often misconceived or rarely mentioned concepts in Sigma-Delta converter design, which much of the current literature tends to miss. Each chapter is broken up into sections, with the essential, practical information listed first, while the theoretical concepts are presented through results evaluation of the suggested simulation exercises of the modulators supplied in the MATLAB®/Simulink® Toolbox software accompanying this book.

Worked Out Example Problems

There seems to be a lack of tools in academic and in industry to allow entry-level engineers to familiarize with the concepts governing the Sigma-Delta conversion technique, especially for continuous time, high orders, multi-bit implementations, and modern architectures of modulators. However, nothing helps to demystify and accelerate the understanding and appreciation of the principles, applications, and limitations of Sigma-Delta converters as much as actually experimenting with some simple modulators to get a real feel for their operation and behavior. Many Sigma-Delta converters books list a number of design problems that tend to be overly simplistic or impractical. Some books provide interesting problems, but often they do not explain how to solve them, especially for real-world situations. Such problems tend to be like exam or homework exercises, and unfortunately, the readers have to learn the hard way to solve them themselves. Even when the readers finish solving such problems, there may not be the possibility to check whether the solution is correct or not because no answers are provided. It is quite frustrating. The *Design of Sigma-Delta Converters in MATLAB®/Simulink®* will not leave the readers guessing. It provides the answers, along with a detailed description showing how the problem was solved. Thus, the first chapters focus on specific system-level simulations, instead of complete designs, in order to help beginners make the important leap from theory to actually understanding the significance of the subtle concepts that are essential to becoming confident in designing a successful modulator.

How to Use This Book: The Sigma-Delta Toolbox

Given the importance of simulations in Sigma-Delta converters, the book is accompanied with a dedicated Simulink® Toolbox which allows the readers to perform all the common simulations required to evaluate a complete design, individually investigate the most important non-idealities affecting single blocks, and explore some of the most famous Sigma-Delta architectures. Although a variety of software to design and simulate ΣΔ-modulators already exist, for example:

- *Delsig Toolbox* (Schreier R. and Temes G.C., *Understanding Delta-Sigma Data Converters*. New Jersey: IEEE Press, John Wiley & Sons Inc., 2005).
- *SIMSIDE* (De la Rosa J.M. and Del Rio R.F., *CMOS Sigma-Delta Converters: Practical Design Guide*. London: IEEE Press, John Wiley & Sons, Ltd., 2013).

these often seem to be too complicated to allow entry-level engineers to appreciate how the Sigma-Delta technology works. This is because such software packages are usually developed in a coded language and make use of complex mathematical functions which do not allow for an intuitive, visual representation of the signal flow. Furthermore, most software programs do not allow for a direct comparison between the different design solutions available, and often the choice of the appropriate implementation to be used must rely solely on strict manufacture specifications

and/or the designer knowledge. The proposed book presents a toolbox developed to overcome these problems by offering a graphical platform that allows an intuitive understanding of the Sigma-Delta conversion process as well as the possibility to compare some of the most popular design arrangements. Since the Toolbox is aimed at educational purposes, Simulink® became the best candidate to develop a graphical platform that can be appreciated even by "less technical" individuals. Therefore, by exploiting the Toolbox graphical approach, not a single line of code has to be written! However, the book also aims to instruct the readers on developing their own designs since the models provided and the simulation suggested inherently allow for an understanding on how a simulation model is set up. In this regard, the Toolbox's user manual included in the appendix of the book will provide specific instructions on the MATLAB® programming feature of the Toolbox.
The software is freely available on the *MATLAB File Exchange* website:

https://www.mathworks.com/matlabcentral/fileexchange/64429-sigma-delta-simulink-toolbox

Who Would Find This Book Useful

The book is intended to help inexperienced students and entry-level engineers design. It assumes little or no prior knowledge of Sigma-Delta converters and/or MATLAB®/Simulink®. Therefore, educators, students, and aspiring designers will find the book a good initial text. At the same time, experienced engineers and scholars may find the book a useful reference source.

Acknowledgments

I wish to express my deep gratitude to Manuele De Pretto who supported me throughout this project and helped me realize the illustrations in this book, to Dr. Robert Henderson of the University of Edinburgh for the inspiring course in Sigma-Delta modulators and Yonghao Wang of Birmingham City University for having introduced me to the subject in my undergraduate years, to my wonderful family whose help and support made this book possible to begin with, and to all my friends for their encouragement and understanding.

Caldogno, Italy Isacco Arnaldi

Contents

1 Introduction 1
- 1.1 The Need for Sigma-Delta Converters 1
- 1.2 Sigma-Delta Converters Fundamentals 2
  - 1.2.1 The Concept of Noise Shaping 2
  - 1.2.2 The Sigma-Delta Modulator 4
  - 1.2.3 The Sigma-Delta Analog-to-Digital Converter 6
- 1.3 Essential Performance Metrics 8
- 1.4 The Role of Simulation in ΣΔ-Converter Design 11
- 1.5 Conclusions and Essential Takeaways 13

Exercises 13
References 18

2 The First-Order Sigma-Delta Modulator 21
- 2.1 Simulink® Model of MOD1 21
- 2.2 Operation for DC Inputs 23
  - 2.2.1 DC Conversion Accuracy 24
  - 2.2.2 Idle Tone Generation 27
  - 2.2.3 Tonal Behavior and DC Input Sweep 29
  - 2.2.4 Tone Prevention 31
- 2.3 Stability of MOD1 33
- 2.4 The Effects of Finite Integrator Gain 34
- 2.5 Operation for Sinewave Inputs 38
  - 2.5.1 Sinewave Amplitude Sweep 40
- 2.6 Theoretical Analysis 43
- 2.7 Conclusions and Essential Takeaways 46

Exercises 47
References 50

3 The Second-Order Sigma-Delta Modulator 51
- 3.1 Simulink® Model of MOD2 51
- 3.2 Operation for DC Inputs 53
  - 3.2.1 Idle Tone Generation 54
  - 3.2.2 Tonal Behavior and DC Input Sweep 55

3.3 Stability of MOD2 . . . 58
3.4 The Effects of Finite Integrator Gain . . . 59
3.5 Operation for Sinewave Inputs . . . 61
3.5.1 Integrator Saturation . . . 63
3.5.2 Sinewave Amplitude Sweep . . . 64
3.6 Theoretical Analysis . . . 68
3.7 Alternative Architectures . . . 70
3.7.1 The Boser-Wooley Modulator . . . 71
3.7.2 The Silva-Steensgaard Modulator . . . 73
3.7.3 Architecture Selection . . . 74
3.8 Conclusions and Essential Takeaways . . . 75
Exercises . . . 76
References . . . 78

4 High-Order Sigma-Delta Modulators . . . 79
4.1 Simulink® Model of MOD3 . . . 79
4.2 Introduction to Stability of High-Order Modulators . . . 80
4.3 An Improved Third-Order Modulator . . . 82
4.3.1 Operation for DC Inputs . . . 83
4.3.2 The Effects of Finite Integrator Gain . . . 84
4.3.3 Operation for Sinewave Inputs . . . 86
4.4 CIFF ΣΔ-Modulator with Multi-bit Quantization . . . 89
4.5 Theoretical Analysis . . . 94
4.5.1 High-Order Modulators Realizability . . . 94
4.5.2 Loop Filter Architectures . . . 96
4.5.3 Optimization of the NTF Zeros . . . 100
4.5.4 Quantization Noise Coupling . . . 101
4.5.5 Conclusions . . . 103
4.6 Design Procedure . . . 104
4.7 Schreier's Delta-Sigma Toolbox . . . 106
4.8 Conclusions and Essential Takeaways . . . 108
Exercises . . . 109
References . . . 110

5 Multi-bit Quantizers . . . 113
5.1 Single-Bit Versus Multi-bit Quantizer . . . 113
5.2 The Problem with Multi-bit Implementations . . . 114
5.3 The Effects of DAC Mismatches in a Modulator . . . 116
5.3.1 Quantizer and DAC Simulink® Models . . . 117
5.3.2 Introducing DAC Mismatches . . . 118
5.4 Random Element Selection . . . 121
5.5 Alternative DEMs Theory: DWA and ILA . . . 124
5.5.1 Data Weighting Averaging . . . 125
5.5.2 Individual Level Averaging . . . 126

5.6 Alternative DEMs Simulations: DWA and ILA . . . 127
5.7 DEM-DAC Within a Sigma-Delta Modulator . . . 132
5.8 Conclusions and Essential Takeaways . . . 134
Exercises . . . 134
References . . . 135

6 MASH Sigma-Delta Modulators . . . 137
6.1 MASH ΣΔ-Modulators Fundamentals . . . 137
6.2 A Third-Order 1+ 1 + 1 MASH ΣΔ-Modulator . . . 140
6.2.1 Operation for DC Inputs . . . 141
6.2.2 Operation for Sinewave Inputs . . . 141
6.2.3 The Effects of Finite Integrator Gain . . . 144
6.2.4 A Third-Order 2 + 1 MASH ΣΔ-Modulator . . . 147
6.3 Additional Simulations for MASH Modulators . . . 148
6.3.1 Quantizer Gain . . . 148
6.3.2 Signal Path Gain . . . 149
6.4 Conclusion and Essential Takeaways . . . 150
Exercises . . . 151
References . . . 152

7 Continuous Time Sigma-Delta Modulators . . . 153
7.1 DT-CT Loop Filter Equivalence . . . 153
7.2 DT-CT Modulators Overview . . . 155
7.3 DT-CT Modulators Non-idealities . . . 157
7.4 Integrators . . . 159
7.4.1 Active RC Integrator: Amplifier's Finite DC Gain and GBW . . . 161
7.4.2 Active RC Integrator: Coefficients Variation . . . 162
7.5 Feedback DAC . . . 163
7.5.1 Excess Loop Delay . . . 164
7.5.2 Inter-symbol Interference . . . 166
7.6 Clock Jitter . . . 168
7.7 Continuous Time NTF Design . . . 171
7.7.1 Additional Simulink® Simulation Information . . . 173
7.8 Conclusions and Essential Takeaways . . . 174
Exercises . . . 176
References . . . 179

8 DT ΣΔ-Modulator Design Example . . . 181
8.1 Modulator Specifications . . . 181
8.2 Theoretical Design . . . 182
8.2.1 Modulator Order and Oversampling Ratio . . . 182
8.2.2 Quantizer Bit . . . 183
8.2.3 Loop Filter Typology . . . 184
8.2.4 Architecture Selection . . . 185
8.2.5 Summary of Design Parameters . . . 186

8.3 High-Level Design . . . 187
8.3.1 NTF Synthesis . . . 187
8.3.2 DC Behavior . . . 191
8.3.3 Stability and Tonal Behavior . . . 192
8.3.4 Integrator Finite Gain and Saturation . . . 194
8.3.5 Feedback DAC and Element Mismatch . . . 194
8.3.6 Coefficients Mismatch . . . 195
8.3.7 Total Harmonic Distortion . . . 198
8.4 Conclusions . . . 198
Exercises . . . 199
References . . . 200

9 CT ΣΔ-Modulator Design Example . . . 201
9.1 Modulator Specifications . . . 201
9.2 Theoretical Design . . . 202
9.2.1 Modulator Order, Oversampling Ratio, and Loop Filter . . . 202
9.2.2 Feedback DAC and Quantizer Considerations . . . 203
9.2.3 Architecture Selection . . . 204
9.2.4 Summary of Design Parameters . . . 205
9.3 High-Level Design . . . 206
9.3.1 NTF Synthesis . . . 209
9.3.2 DC Behavior . . . 213
9.3.3 Stability and Tonal Behavior . . . 214
9.3.4 Integrator Finite Gain and Saturation . . . 216
9.3.5 Feedback DAC and Element Mismatch . . . 216
9.3.6 Coefficients Mismatch . . . 217
9.3.7 Inter-symbol Interference . . . 217
9.3.8 Excess Loop Delay . . . 218
9.3.9 Jitter Behavior . . . 219
9.3.10 Total Harmonic Distortion . . . 221
9.4 Conclusions . . . 221
Exercises . . . 222
References . . . 223

10 Frontiers of Sigma-Delta Modulators . . . 225
10.1 SMASH ΣΔ-Modulators . . . 225
10.2 Incremental ΣΔ-Modulators . . . 227
10.2.1 Digital Filter Design . . . 229
10.2.2 Theoretical Resolution . . . 229
10.2.3 Incremental ΣΔ-Modulators Conclusions . . . 230
10.3 Time Domain ADCs . . . 231
10.3.1 Introduction to VCO Operation . . . 232
10.3.2 VCO Voltage-to-Frequency ΣΔ-Modulators . . . 234

10.3.3 VCO Voltage-to-Phase ΣΔ-Modulators 238
10.3.4 Dual-VCO ΣΔ-Modulators 240
10.4 Conclusion and Essential Takeaways 242
References 243

Appendix A: Sigma-Delta Modulators Design Tools 245

Appendix B: The Sigma-Delta Simulink® Toolbox 249

Index 259



Introduction

1

Despite the practical approach that this book hopes to provide, a brief theoretical introduction to the most basic concepts to understand Sigma-Delta Analog-to-Digital Converters (ΣΔ-ADCs) behavior and design practices is, somewhat, mandatory. Therefore, after a brief outline to set the topic of ΣΔ-ADCs in the context of modern electronics, Sect. 1.2 introduces the fundamentals of this technology. In addition, Sect. 1.3 presents the performance metrics that are necessary to evaluate a converter, while Sect. 1.4 discusses the importance and major issues of simulations in the context of the design process.

1.1 The Need for Sigma-Delta Converters

At the advent of emerging conversion standards in electronic applications that seek ever-increasing high fidelity with stringent linearity requirements and the system on chip design trend that continuously improves the level of integration while reducing the power budget of electronic devices, new conversion techniques which allow for high resolution and low costs and that are power conscious need to be developed. In recent years, a popular choice for realizing such converters is Sigma-Delta modulators, and they can be found in a variety of modern applications (e.g., audio, voice-band, communication systems, bio-medics/bioengineering equipment, industrial measurements, etc.). Their architecture design is conveniently suitable for CMOS processes making integration with highly digital functions, such as digital signal processors (DSPs), practical. Further, as ΣΔ-modulators are closed loop systems, their tolerance to accurate matching or calibration of the analog components in low voltage environments results in being greatly relaxed allowing the use of simple and low-cost analog building blocks [1]. Different ΣΔ-converter's architectures and arrangements exist in order to achieve high resolution, and the latest design trends [2] show a shift from cascaded low-order, discrete-time single-bit solutions to high-order, continuous time, multi-bit implementations. From a design

© Springer International Publishing AG, part of Springer Nature 2019

I. Arnaldi, *Design of Sigma-Delta Converters in MATLAB®/Simulink®*,
https://doi.org/10.1007/978-3-319-91539-5_1

perspective, switched capacitor has been the common realization technique of discrete-time (DT) modulators, and in order to achieve a linear settling, the sampling frequency used in these converters needs to be significantly lower than the gain-bandwidth product (GBW) of the amplifiers. Therefore, for a defined power budget, the highest conversion speed which can be achieved by DT modulators is limited. On the other hand, less stringent amplifier's speed specifications are required in continuous time (CT) ΣΔ-modulators, allowing to achieve a higher speed of operation and lower power consumption. Further, thanks to the inherent anti-aliasing properties of CT modulators, the requirements of the converter's anti-aliasing filter (AAF) are greatly relaxed and in some cases even eliminate the need for such filter at the input, resulting in significant power and cost reduction. This is extremely valuable in applications such as ones involving portable devices where power consumption is of great concern as battery life has to be maximized. Therefore, it is not surprising to find such a trend in the literature.

1.2 Sigma-Delta Converters Fundamentals

Although an impressive amount of resources dealing with the description of the architecture and theory of ΣΔ-ADCs have been published since the first appearance of such technology in the 1960s, most of the literature commence with a maze of integrals and deteriorate from there. Studying from a typical published article or book usually leads less specialized individuals, who are unclear of the theory of operation of ΣΔ-ADCs, to believe that the topic is too difficult to comprehend easily, if comprehensible at all. There is nothing particularly difficult to understand about ΣΔ-ADCs, as long as the detailed mathematics is avoided, and this section has been written in an attempt to clarify the subject.

1.2.1 The Concept of Noise Shaping

Sigma-Delta Analog-to-Digital Converters exploit oversampling, noise shaping, and digital signal filtering to generate a high-resolution digitized output [3]. To familiarize with these concepts, such techniques are first to be analyzed in the frequency domain.

According to basic ADC theory [4, 5], the quantizer is a strongly deterministic and nonlinear system which introduces a *quantization error* in the converter. Such error, in approximating the input signal to the quantizer (typically ½LSB – least significant bit), appears as noise in the system and limits the theoretical resolution that could be achieved. Since the quantizer's behavior in the converter is very complicated to investigate analytically, the quantizer is usually approximated as an *additive white noise source* [6] making possible to estimate the quantization error and hence the ADC performance. Given this approximation, in a typical Nyquist-rate N-bit ADC, the root-mean-square (rms) quantization noise is considered to be uniformly distributed within the Nyquist band of DC to $f_s/2$, where f_s is the sampling

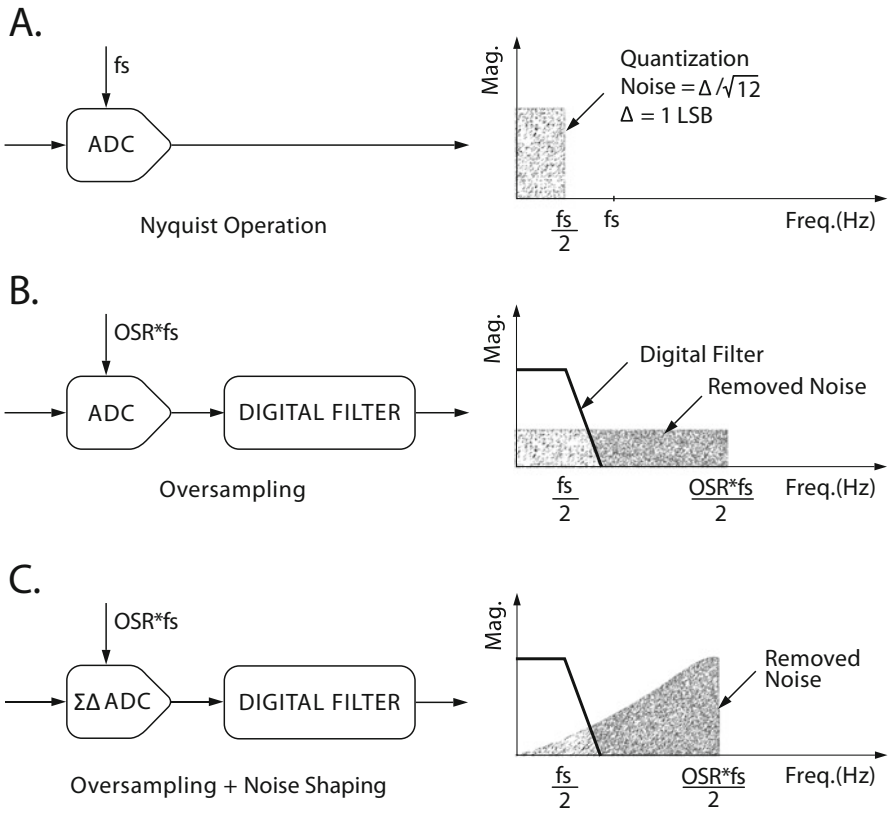


Fig. 1.1 (**a**) Nyquist, (**b**) oversampling, and (**c**) ΣΔ ADCs noise spectrum

rate (Fig. 1.1a). For a full-scale sinewave input, the signal-to-noise ratio (SNR) of such system, and hence the converter's resolution (i.e., ENOB – effective number of bits), corresponds to SNR = 6.02ENOB + 1.76 (dB).

Applying the technique of oversampling to the same N-bit ADC, which is sampling at a higher rate by a factor of OSR $*$ f_s, the same amount of rms quantization noise is found in the system, but the noise is now distributed over the wider bandwidth of dc to OSR $*$ f_s (Fig. 1.1b). Applying a digital low-pass filter (LPF) to the output, much of the quantization noise can be removed without affecting the wanted signal – so the ENOB results improved and SNR = 6.02ENOB + 1.76 + 10 $\log_{10}$(OSR). Therefore, a high-resolution AD conversion can be achieved by using an otherwise low-resolution ADC.

The main drawback of oversampling is that in order to lower the in-band quantization noise such that an N-bit increase in resolution is achieved, the system must be oversampled by a factor of 2^{2N}. In other words, oversampling achieves a 0.5-bit increase per doubling of OSR. This is impractical to achieve high resolutions as high-speed systems are difficult to design and lead to high power consumption. To

keep the oversampling factor at a reasonable value while achieving very high resolution, the technique of noise shaping, which is shaping the quantization noise such that most of it resides outside the signal passband of interest, comes in handy. The technique is illustrated in Fig. 1.1c, and it is the main concept behind all ΣΔ-converters since it is the ΣΔ-modulator in such converters that allow achieving the noise-shaping characteristic. To give some perspective, the simplest first-order ΣΔ-modulator provides a 1.5-bit increase per doubling of OSR, resulting in it being much more efficient than using only oversampling.

1.2.2 The Sigma-Delta Modulator

To understand how noise shaping is achieved, Fig. 1.2a illustrates the basic block diagram of a low-pass (LP) ΣΔ-modulator, which consists of the following [1]:

- Loop filter
- Quantizer (ADC)
- Feedback digital-to-analog converter (DAC)

The basic idea of ΣΔ-modulation is easy enough to state [4]: to modulate the analog input signal into a digital word sequence whose spectrum approximates that of the analog input well in a narrow frequency range but which is otherwise noisy. This noise arises from the quantization of the analog signal, and the loop filter *shapes* the quantization noise such that most of its power lies away from the narrow, desired frequency range. To accomplish this, filtered negative feedback compensation is used to obtain the noise-shaping characteristic of the quantization error.

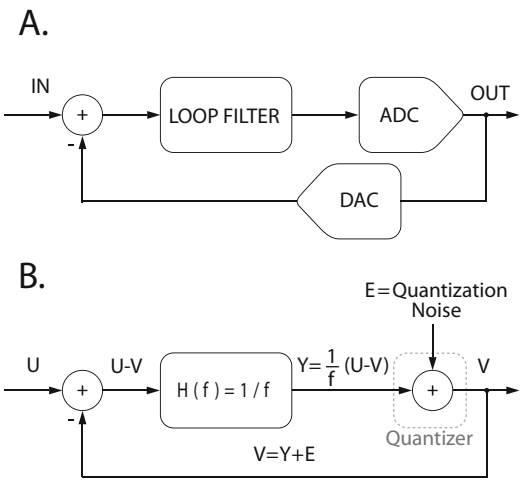


Fig. 1.2 (**a**) Low-pass ΣΔ-modulator block diagram and (**b**) respective linear model

Following the frequency domain analysis in Fig. 1.2b [6], the loop filter transfer function corresponds to $H(f) = 1/f$. The amplitude response of $H(f)$ is inversely proportional to the input frequency, hence resembling a low-pass filter. The quantizer is treated as a white noise source, where the quantization noise E is injected directly into the output signal. Naming the input signal U and the output signal V, the signal coming out of the input summer block must equal to $U - V$. This signal is then multiplied by the loop filter transfer function, $1/f$, and the result summed with E at the output. By inspection, the expression for the output signal V can then be written as:

$$V = \frac{1}{f}(U - V) + E \tag{1.1}$$

Rearranging and solving for V in terms of U, f, and E:

$$V = \frac{U}{f+1} + \frac{Ef}{f+1} \tag{1.2}$$

where the first term is the signal transfer function (STF) and the second one is the noise transfer function (NTF). According to Eq. 1.2, as the frequency f approaches zero, the output signal V approaches U with no noise component. At high frequencies, the amplitude of the signal component U approaches zero, while the noise component approaches E; hence the output V consists primarily of quantization noise. Clearly, the loop filter determines the spectral properties of the ΣΔ-modulator and has a low-pass effect on the signal and a high-pass effect on the quantization noise, resulting in the desired noise-shaping characteristic [3].

The loop filter is formed by one or many cascaded integrators connected by a series of feedback and/or feedforward paths. The number of integrators defines the loop filter's order, which determines the aggressiveness of the noise-shaping characteristic in pushing the noise out of the band of interest. Intuitively, the higher the order of the filter, the lower the noise left in the passband (Fig. 1.3). Unfortunately, high-order loop filters are difficult to design for reasons that will be discussed to a great extent later in this book, and so, for now, the important concept to grasp is that when designing ΣΔ-converters, the main goal is to obtain a loop filter which provides high stability and few artifacts, such that over the passband [7]:

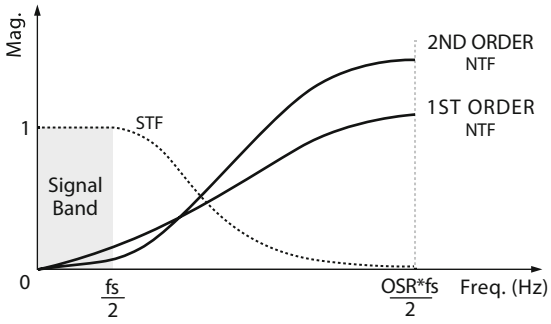


Fig. 1.3 STF and various order NTFs

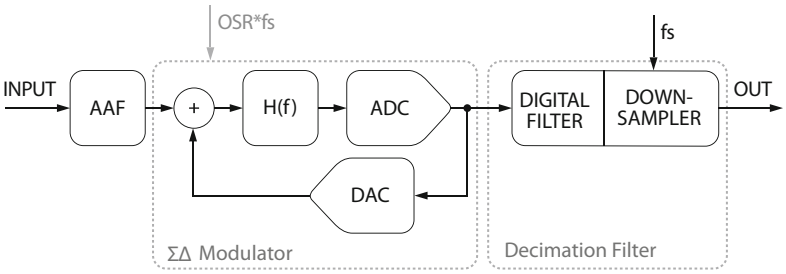


Fig. 1.4 ΣΔ-modulator block diagram

$$\text{STF} \cong 1 \qquad \text{and} \qquad \text{NTF} \cong 0 \tag{1.3}$$

This ensures that the signal passes unchanged, while the noise is being shaped away from the band of interest.

1.2.3 The Sigma-Delta Analog-to-Digital Converter

Having understood the concept of noise shaping and how this is achieved, a complete ΣΔ-ADC can now be analyzed. As illustrated in Fig. 1.4, it is formed by [1]:

- Anti-aliasing filter
- Sigma-Delta modulator
- Decimation filter

The anti-aliasing filter [6] is used to band-limit the analog input signal in order to avoid aliasing during its subsequent sampling. It should be noted at this point that since ΣΔ-ADCs operate at an oversampling ratio of $\text{OSR} * f_s$, the attenuation requirements of the analog AAF are greatly relaxed so that smooth transition bands are usually sufficient compared to Nyquist-rate converters (Fig. 1.5).

Having introduced the frequency domain analysis of the modulator, it is now fundamental to investigate the ΣΔ-ADC behavior from a time domain perspective [3]. Since the integrators forming the loop filter are *memory* [6] elements, each output of the modulator is generated utilizing all preceding input values. Such property fundamentally differs from the distinctive one-to-one relation of Nyquist-rate converters between input and output samples; therefore only a comparison of the complete input and output waveform allows to evaluate the ΣΔ-converter's accuracy. This concept can be better appreciated by looking at Fig. 1.6. As it can be seen, in each clock cycle, the value of the output of the modulator is either plus or minus the full scale of the assumed single-bit quantizer (e.g., ± 1). When the sinusoidal input to the modulator is close to a plus full scale, the output of the modulator is

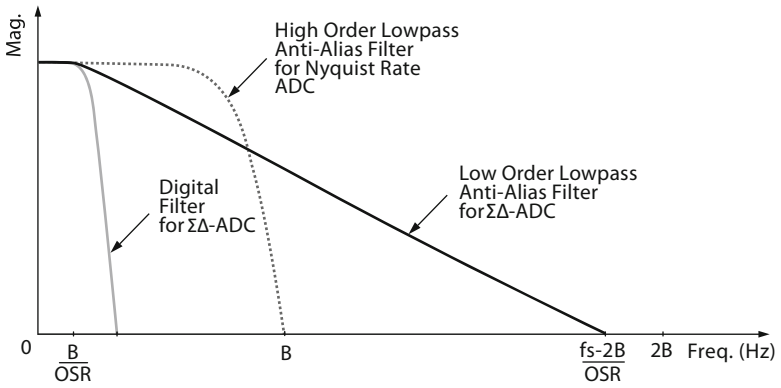


Fig. 1.5 Nyquist vs. ΣΔ-ADCs anti-aliasing filter requirements

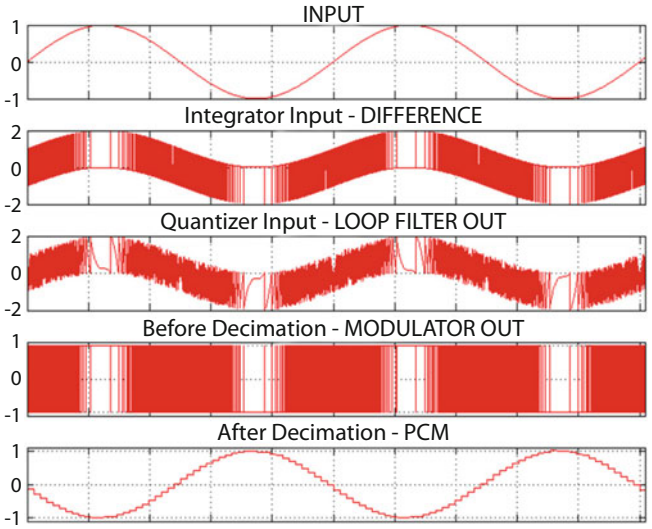


Fig. 1.6 First-order, single-bit ΣΔ-modulator output at each conversion stage

positive during most clock cycles and vice versa. Thus, it is the local average of the modulator output that tracks the analog input. When the input is near zero, the output value of the modulator varies rapidly between a plus and a minus full scale with approximately zero mean.

The last component of a ΣΔ-ADC is the decimation filter [7]. Since the output of a ΣΔ-modulator is a stream of bits and since the final bandwidth of the output signal is to be reduced by the digital filter, the output data rate may also be lowered (downsampling) from the oversampling ratio of $OSR*f_s$ to the Nyquist rate of f_s. In the process, redundant data caused by oversampling are eliminated while preserving an accurate AD conversion, since the *Nyquist criterion* [6] is not to be violated.

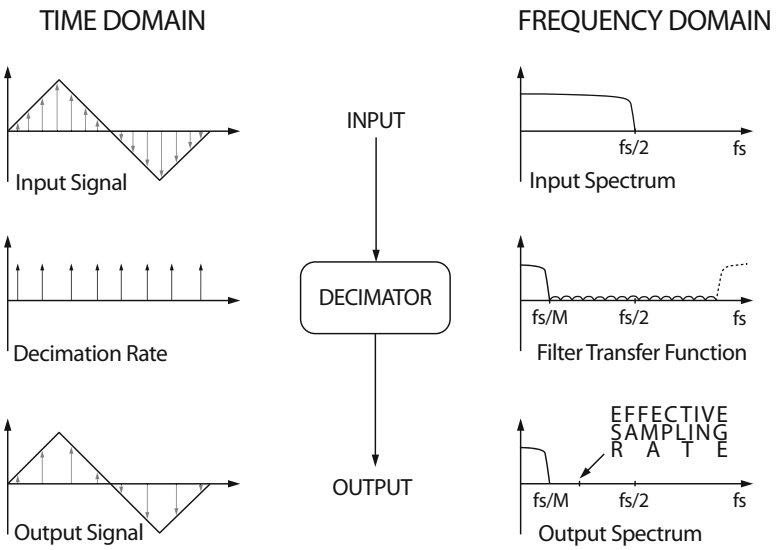


Fig. 1.7 Decimation

Downsampling is usually desirable in order to minimize the amount of information for subsequent transmission, storage, or digital signal processing as well as to meet the working Nyquist frequency of most systems [5]. Downsampling of the output data rate is achieved by passing every *M-th* result to the output and discarding the remainder. The process is better known as *decimation* by a factor of M, where M can have any integer value provided that the output data rate is more than twice the input signal bandwidth. The decimation filtering process in the time and frequency domains is illustrated in Fig. 1.7. As it can be seen, decimation simply reduces the output sample rate and does not cause loss of information.

Another important aspect to be aware of is that in ΣΔ-converters the decimation filter can also be used to provide increased resolution [1, 7]. An example of this is presented in Fig. 1.8, which shows a 16:1 decimation process with 1-bit input samples. Although the input data resolution to the decimation filter is only 1 bit, the averaging method (decimation/downsampling) yields more resolution (4 bits) through reducing the sampling rate by 16:1. Note that the signal full scale in the example corresponds to 0 or 1, instead of the ± 1 of the modulator, due to the digital nature of the filtering process.

1.3 Essential Performance Metrics

Performance metrics [1, 2] are used to evaluate the behavior of a converter. Contrary to Nyquist-rate ADCs whose performance is mainly characterized by static performance metrics such as gain and offset errors, monotonicity, differential nonlinearity (DNL), integral nonlinearity (INL), etc. [6] – ΣΔ-ADCs are evaluated by dynamic

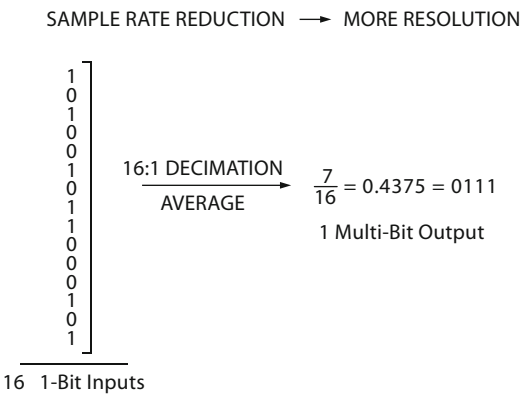


Fig. 1.8 Increase of resolution through decimation

performance metrics. Dynamic metrics are time-dependent and obtained from the frequency domain representation of the time domain digital output sequence. This requires the computation of the fast Fourier transform (FFT) of a finite-length output sequence with a specific *windowing function* [6], as it will be explained later. From that power spectrum representation of a ΣΔ-modulator's output sequence, some spectral metrics are directly measured, and other noise and power metrics are derived. The most common dynamic performance metrics of a ΣΔ-ADC are illustrated in Fig. 1.9, and include:

- *Signal-to-Noise Ratio (SNR)*: Ratio between the power of the sinusoidal signal and the total noise power including quantization and circuit noise but excluding the DC component and distortion, expressed in dB.

$$\text{SNR (dB)} = 10\log_{10}\left(\frac{P_{\text{Signal}}}{P_{\text{Noise}}}\right) \tag{1.4}$$

- *Signal-to-Noise and Distortion Ratio (SNDR)*: Ratio between the signal power of the fundamental frequency to the power of all other spectrum components, including noise and distortions, but excluding the DC component, expressed in dB.

$$\text{SNDR (dB)} = 10\log_{10}\left(\frac{P_{\text{Signal}}}{P_{\text{Noise}} + P_{\text{Distortions}}}\right) \tag{1.5}$$

- *Signal to Quantization Noise Ratio (SQNR)*: Ratio between the fundamental frequency signal power to the power of the quantization noise, expressed in dB. The maximum SQNR is achieved with a full-scale input signal because, according to the linear model [6], the quantization noise power is constant and not

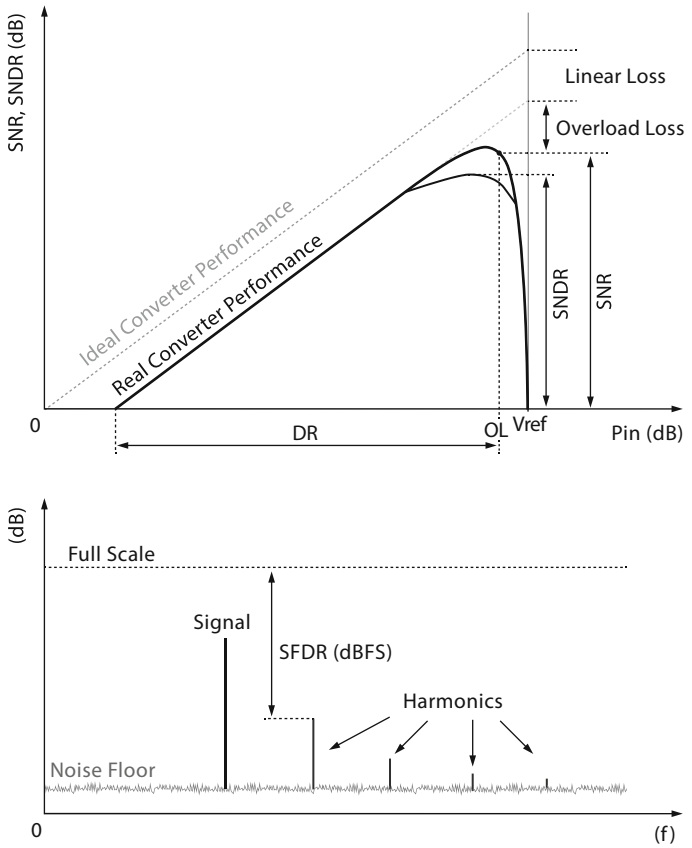


Fig. 1.9 Graphical representation of the dynamic performance metrics

signal dependent. Typically used to estimate the upper limit for the theoretical dynamic performance of the ΣΔ-ADC.

- *Spurious-Free Dynamic Range (SFDR)*: Ratio of the signal power of the fundamental frequency to the power of the strongest spectral tone.
- *Dynamic Range (DR):* Ratio between the power of the full-scale input signal and the power of the smallest detectable input signal, expressed in dB.
- *Total Harmonic Distortion (THD)*: Ratio of the total power of the harmonics to the signal power, expressed in dB.

$$\mathrm{THD}\ (\mathrm{dB}) = 10\log_{10}\left(\frac{{H_{\mathrm{D},2}}^2 + {H_{\mathrm{D},3}}^2 + \ldots + {H_{\mathrm{D},n}}^2}{{H_{\mathrm{D},1}}^2}\right) \tag{1.6}$$

- *Signal to Total Harmonic Distortion + Noise (S/(THD + N))*: The ratio of the rms value of the fundamental input signal to the rms sum of all other spectral components in the passband, expressed in dB.

- *Overload Level (OL)*: The SNR of a ΣΔ-modulator increases monotonously with the input signal amplitude but sharply drops for input amplitudes close to half of the full-scale input range of the embedded quantizer due to its overload and the associated in-band noise power increase. The overload level is considered to define the maximum input amplitude for which the ΣΔ-modulator still operates correctly and can almost be arbitrarily defined, but it is typically chosen as the amplitude for which the SNR drops 6 dB below the peak SNR [8].
- *Effective Number of Bit (ENOB)*: Effective ADC resolution, expressed in bits. Assuming a sinusoidal input, it results to be:

$$\text{ENOB (Bit)} = \frac{\text{SNR (dB)} - 1.76}{6.02} \tag{1.7}$$

1.4 The Role of Simulation in ΣΔ-Converter Design

Sigma-Delta modulators – irrespectively, from being ADCs, DACs, or even purely digital designs such as a divider in a fractional-N phase-locked-loop (PLL) – are typically designed following the "top-down" [8] design approach, which involves the realization of one or multiple mathematical models in order to predict and investigate system behaviors, optimize aspects of its performance, and/or compare competing designs. Only once a satisfactory system-level design is achieved, detailed implementation and/or circuital issues are considered.

However, it should be noted that Sigma-Delta modulators are nonlinear systems since a quantizer is implemented in the ΣΔ-loop. Therefore, differently from linear systems, an exact mathematical analysis is often very difficult or even impossible, and to date there is no complete theoretical analysis for arbitrary ΣΔ-modulators that can be used to accurately and unquestionably predict if a proposed modulator design is going to fully meet its performance targets and do so with no undesirable side effects [6]. Nevertheless, some useful analytical results that can be used in the design process of a ΣΔ-modulator have been achieved over the years, including [1]:

- Few specific cases in which an exact mathematical analysis may be possible.
- Some general mathematical analyses become possible when making simplifying assumptions, but violating these assumptions can result in the real modulator behaving very differently than predicted (e.g., reduced stability, input signal range, etc.).
- Some analyses yield only upper or lower performance bounds which may be considered more like warning guidelines when constructing a new design rather than as absolute design limits. Therefore, it may be possible to push some designs to achieve higher performance without incurring in catastrophic results.
- Some commonly used analytical results are just accepted rules of thumb based more on extensive experience than theoretical consideration! The consequence of the various approximations and limitations invoked, either to get mathematically

tractable results or simply because experience suggests they are good rules of thumb, is that extensive system-level simulation is essential to verify whether a ΣΔ-modulator will indeed perform to specifications when designed using existing theory and without unwelcome surprises (i.e., instability, saturation, etc.).

Due to the difficulties in developing an accurate mathematical model and the consequent importance that simulations play in the analysis of the initial design, this book introduces MATLAB® and Simulink® as very useful tools for system-level modeling and predicting the performance of ΣΔ-modulators. Important to note is that, to prevent unwanted behaviors, it is common when designing and simulating ΣΔ-modulators at system level to include some real circuit implementation effects [2, 7] since these can have a strong influence on the modulator's performance. As an example, these may include:

- Finite signal-swing headroom effects (i.e., soft saturation, hard clipping, etc.)
- Practical circuital limitations (i.e., finite gain and bandwidth, etc.)
- Nonlinearities of the modulator circuits (i.e., quantizer, multi-bit DAC, etc.)
- Effects of systematic or random component deviations (i.e., mismatch)
- Timing issues (i.e., clock jitter in CT modulators, finite settling time, etc.)

These effects, even if only in approximate form, are often included at the system-level simulation stage due to the fact that circuit-level simulation is typically not an effective tool for investigating their effects. This is because system-level simulations usually run at least thousands of times faster than circuit-level simulation using tools such as HSPICE™ or Spectre™ (seconds/minutes versus hours/days). Considering that many hundreds or thousands of simulations may be required to convince a designer that the ΣΔ-modulator designed will behave as expected for likely inputs, it is unreasonable in terms of simulation time requirements to use only circuit simulators for examining the consequences of real circuit effects. Including approximations of circuit effects in the system-level model can, however, allow their likely effects to be studied under a wide range of modulator operating conditions and hence hugely increases the likelihood that any problems they cause will be found, and found early, in the design process.

Given the importance of simulations in ΣΔ-modulators and in order to keep the focus on the design process, instead of system-level modeling techniques, this book is accompanied with a dedicated Simulink® Toolbox which allows the reader to perform all the common simulations required to evaluate a complete design, individually investigate the non-idealities affecting single blocks, and even explore some of the most famous ΣΔ-architectures. Moreover, the Toolbox facilitates the creation of individual, personalized designs that the reader may want to implement. To further facilitate the reader in these tasks, the Toolbox exploits a graphical approach, and so not a single line of code has to be written! Therefore, particular knowledge of either MATLAB® or Simulink® is not required, and everyone should hopefully be able to enjoy designing modulators.

The software is freely available on the *MATLAB File Exchange* website:

https://www.mathworks.com/matlabcentral/fileexchange/64429-sigma-delta-simulink-toolbox

1.5 Conclusions and Essential Takeaways

In this chapter, an introduction to the topic of Sigma-Delta Analog-to-Digital Converters has been provided. The benefits of employing oversampling and quantization noise shaping in the digitization of signals have been analyzed and compared to the performance of Nyquist-rate ADCs. Further, the process by which noise shaping is achieved, along with the role of the various blocks forming a complete ΣΔ-ADC, has been presented. Finally, essential performance metrics and the importance of simulations and system-level modeling in the design of ΣΔ-ADCs have been discussed. Although the rather descriptive tone used in providing initial information, some very important concepts have been covered, including [1]:

- Oversampling an ADC can increase resolution beyond matching limit of the analog components – speed is traded for accuracy.
- Simple oversampling (averaging) is very inefficient (0.5 bits increase per doubling of OSR).
- Noise shaping moves quantization noise to high frequencies where it can be separated from the signal by a low-pass filter (e.g., decimation filter).
- Sigma-Delta modulation is much more efficient (i.e., 1.5 bits increase per doubling of OSR in first-order modulators).
- The spectral properties of noise shaping (aggressiveness) are determined by the loop filter (i.e., number of integrators) in the ΣΔ-modulator.
- Since the integrators forming the loop filter are memory elements, it is the local average of the modulator's output that tracks the analog input.

These concepts have been discussed mainly considering the quantization noise as the only source of error limiting the resolution of a ΣΔ-modulator. Therefore, in the following chapters, the effects of non-idealities associated with the practical implementation of the modulator blocks are to be analyzed.

Exercises

The set of exercises proposed in this first chapter aims not only to test the concepts covered but also to provide a brief review of the general ADC theoretical knowledge expected to successfully understand the topic of the book. Although all the essential information to complete the exercises have been provided, it is highly recommended to follow the references in the text as a source of additional information in the case some of the concepts covered are unclear or difficult to understand for the less experienced reader.

Q.1
Briefly describe the *Sampling* process and the *Nyquist theorem* underlying basic ADCs theory [5, 6]. What is oversampling? What are the main advantages and disadvantages of oversampling a signal in regard to ADCs?

Q.2
Quantization is a well-established concept in ADC theory (i.e., see [3, 6]), where the quantization levels are typically referred as the allowed values in the output signal after quantization, whereas the quantization step (i.e., Δ) refers to the distance between two successive quantization levels (Fig. 1.10).

The quantization error is simply the difference between the input and output to the quantizer. It can easily be seen that the quantization error, $e_q(n)$, is always bounded by:

$$-\Delta/2 \leq e_q(n) \leq \Delta/2$$

Since quantization is a highly nonlinear process, the exact effect of quantization on the signal content and the nature of quantization noise is difficult to measure. For this reason, several assumptions are often made:

- The error sequence, $e_q(n)$, is a stationary, random process.
- The error sequence is uncorrelated with itself and with the input sequence.
- The probability-density function (PDF) of the error is uniform over the range of quantization error.

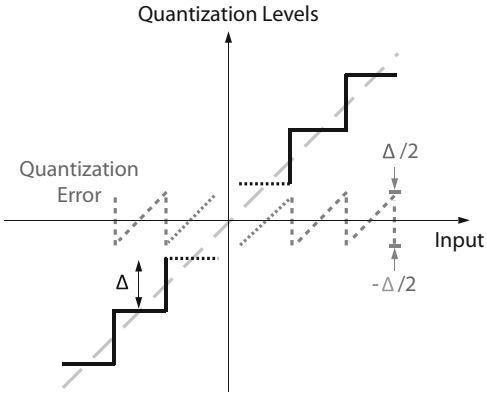


Fig. 1.10 Input vs. output characteristic of a quantizer and quantization error

Such assumptions are known to be, in general, untrue. However, they are a reasonable approximation for large-amplitude, time-varying input signals when N is large and successive quantization error values are not highly correlated. These assumptions allow to represent the quantization process as the introduction of an additive white noise source, as illustrated in Fig. 1.11.

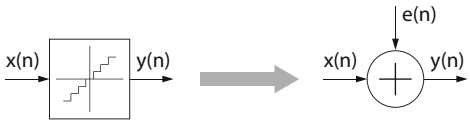


Fig. 1.11 Linear white noise model of a quantizer

Since the error is treated as white noise, the power spectral density of the noise will also have a uniform distribution within the limits of the Nyquist band (Fig. 1.12) and, assuming that the sampling theorem is satisfied (i.e., the signal is sampled at least twice the highest frequency in the input signal, $f_s > 2f_b$), then the quantization error can be regarded as the only error in the AD conversion process.

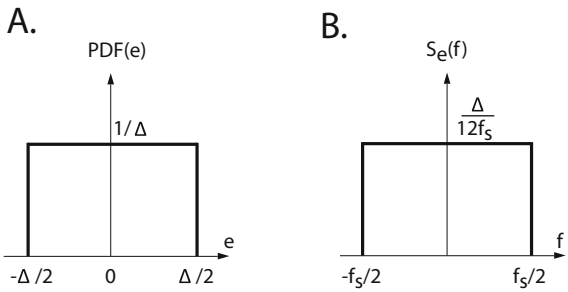


Fig. 1.12 Quantization noise (**a**) probability-density function and (**b**) power spectral density

Using the assumption of uniform distribution, the quantization noise power, P^2_{Noise}, is given by:

$$\int_{-\Delta/2}^{\Delta/2} \frac{1}{\Delta} e_q^2 de = \frac{1}{\Delta}\frac{e^3}{3}\Big|_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} = \frac{1}{\Delta}\left[\frac{\left(\frac{\Delta}{2}\right)^3}{3} - \frac{\left(-\frac{\Delta}{2}\right)^3}{3}\right] = \frac{1}{\Delta}\left[\frac{\Delta^3}{24} - \frac{-\Delta^3}{24}\right] = \frac{\Delta^2}{12}$$

With the root mean square (rms) of the error corresponding to:

$$\sqrt{P^2_{\text{Noise}}} = \sqrt{\frac{\Delta^2}{12}} = \frac{\Delta}{2\sqrt{3}}$$

The rms value for a full-scale sinusoidal input can be derived as [4]:

$$\sqrt{P^2_{\text{Signal}}} = \frac{\Delta 2^N}{2\sqrt{2}}$$

Therefore, starting from the general formula

$$\text{SNR}_{\max} = 20\log_{10}\left(\frac{P_{\text{Signal}}}{P_{\text{Noise}}}\right),$$

find the typical form used in the analysis of Nyquist converters (i.e., $\text{SNR} = 6.02N + 1.76$ (dB)) .

Q.3

For a Nyquist ADC, in which $f_s = 2f_b$, all the quantization noise power falls inside the signal band and passes to the ADC output as a part of the input signal itself. Conversely, if an oversampled signal is quantized, because $f_s \gg 2f_b$, only a fraction of the total quantization noise power lies within the signal band, so that the larger the OSR, the smaller the *in-band noise power* caused by the quantization process. Starting from the general formula

$$\mathrm{SNR}_{\max} = 10\log_{10}\left(\frac{P_{\mathrm{Signal}}}{P_{\mathrm{Noise}}}\right)$$

and considering that [6]

$$P_{\mathrm{Signal}} = \left(\frac{\Delta^2 2^{2N}}{8}\right) \quad \text{and} \quad P_{\mathrm{Noise}} = \frac{\Delta^2}{12}\frac{1}{\mathrm{OSR}},$$

find the typical form used in the analysis of Nyquist converters (i.e., SNR = 6.02*N* + 1.76 + 10 log (OSR) (dB)) .

Q.4

Complete the table below using the SNR formula for oversampling converters. For the noise-shaping cases use the formula:

$$\mathrm{SNR} = 6.02N + 1.76 + (20L + 10)\log_{10}(\mathrm{OSR}) - 10\log_{10}\left(\frac{\pi^{2L}}{2L+1}\right)$$

where N is the number of bits (i.e., always 1 in this exercise) and L is the ΣΔ-modulator order. Analyzing the table, what relation can be found in terms of SNR and increase of OSR as well as of ΣΔ-modulator order?

SNR (dB)					
Oversampling rate	OSR-ADC	First ΣΔ order	Second ΣΔ order	Third ΣΔ order	Fourth ΣΔ order
1					
2					
4					
8					
16					
32					
64					
128					
256					
512					
1024					

Q.5
What is the effective resolution of an ADC having an SNR of 96 dB? Given that a 1-bit ADC has a 6 dB SNR, what sample rate is required using oversampling (no noise shaping) to obtain a 96 dB SNR if $f_{b_{max}} = 25$ KHz? What considerations should be drawn from the result about the implementation of such converter?

Q.6
Calculate the OSR required for a SNR of 70 dB for a pure oversampling and oversampling in combination with noise shaping. Discuss the results.

Q.7
A sampled signal is band-limited to $f_b = 22$ KHz. What is the sampling frequency, f_s, for an oversampling ratio of 128? Since a 1-bit ADC has an inherent 6 dB SNR, which maximum SNR is acquired by combining it with strict oversampling of 128 if no noise shaping is used? What is the maximum SNR in the similar case exploiting a second-order noise-shaping modulator? If a 1-bit ADC using third-order noise shaping has a maximum SNR of 125 dB for an OSR of 128, what is the expected maximum SNR if the OSR is reduced to 32?

Q.8
Figures 1.13 and 1.14 show the test results for a wideband ΣΔ-modulator. From these two plots, find the approximate values of the peak SNR, peak SNDR, SFDR, dynamic range (DR), and input signal level for the peak SNR.

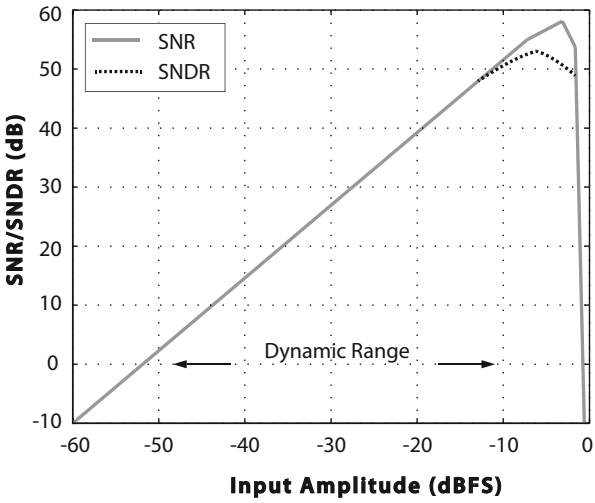


Fig. 1.13 Measured SNR, SNDR, and DR for a ΣΔ-modulator

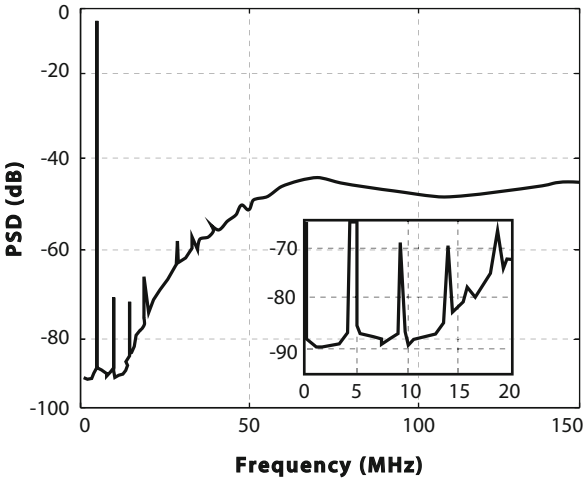


Fig. 1.14 PSD of the tested ΣΔ-modulator for

Q.9
Sketch the block diagram of a first-order Sigma-Delta modulator. Use a time discrete delaying integrator with transfer function $H(z) = \frac{1}{z-1}$. Use an additive white noise source to model the quantizer noise. Sketch the block diagram again for this case. What considerations are important in regard to the input signal of the modulator?

Q.10
Similarly to the analysis of Fig. 1.2 in Sect. 1.2.2, determine the signal transfer function (STF) as well as the noise transfer function (NTF) of the first-order Sigma-Delta modulator sketched.

Q.11
In a low-pass Sigma-Delta modulator, the quantization noise is frequency shaped (select one):

$$\text{Lowpass-Bandpass-Highpass-Notch (Band-reject)}.$$

Q.12
Explain the different relation between input and output samples of ΣΔ-modulators compared to Nyquist ADC. What is the main reason of this difference?

Q.13
Briefly explain the role of the decimator in a ΣΔ-ADC.

References

1. Schreier R, Temes GC. Understanding delta-sigma data converters. Hoboken: Wiley; 2005.
2. De la Rosa JM, Del Rio RF. CMOS sigma-delta converters: practical design guide. London: Wiley; 2013.

3. Candy JC, Temes GC. Oversampling delta-sigma data converters: theory, design and simulation. New York: Wiley-IEEE Press; 1991.
4. Kester W. The data conversion handbook. Norwood: Analog Devices Inc.; 2004.
5. Pohlmann KC. Principles of digital audio. London: McGraw-Hill; 2008.
6. Oppenheim AV, Schafer RW. Discrete-time signal processing. Upper Saddle River: Prentice-Hall Inc.; 1989.
7. Norsworthy SR, Schreier R, Temes GC. Delta-sigma data converters: theory, design and simulation. New York: Wiley-IEEE Press; 1996.
8. Medeiro F, Verdù BP, Vazquez AR. Top down design of high performance sigma delta modulators. Boston: Kluwer; 1999.



The First-Order Sigma-Delta Modulator

2

This chapter focuses on specific system-level simulations of the simplest Sigma-Delta modulator, to be referred as MOD1, formed by a first-order loop filter and a single-bit quantizer. The aim of this chapter is to help beginners make the important leap from theory to actually understand the significance of the subtle concepts that are essential to becoming confident in designing a successful modulator. As mentioned, MATLAB® and Simulink® are the tools used to explore the properties of MOD1. Therefore, after a brief description of the Simulink® model provided in the Toolbox accompanying this book, a step-by-step set of practical exercises is proposed. Specifically, the reader will be able to observe the time domain waveforms of the modulator in response to both DC and sinewave inputs as well as noise shaping in the frequency domain via fast Fourier transform (FFT). A number of non-ideal effects are studied such as tones, dead zones, and saturation. Further, the prevention of the non-idealities discussed is investigated, mainly through the use of dither. Some simple mathematics is used as an aid in presenting the theoretical concepts, and for results evaluation, however, formula proofs are generally avoided in order to keep the focus on the results themselves and a rather practical approach to the subject. The interested reader is encouraged to consult the references for complete mathematical derivations.

2.1 Simulink® Model of MOD1

To investigate the properties of ΣΔ-modulators, it is perhaps easier to introduce them as discrete-time implementations when working at system level. Opening the Simulink® Toolbox file *"mod1.mdl,"* the schematic illustrated in Fig. 2.1 should appear. Conceptually the schematic represents the model discussed in Chap. 1 (Fig. 1.2) but with few added elements to allow system behavior investigation. The elements forming the model include:

© Springer International Publishing AG, part of Springer Nature 2019

I. Arnaldi, *Design of Sigma-Delta Converters in MATLAB®/Simulink®*,
https://doi.org/10.1007/978-3-319-91539-5_2

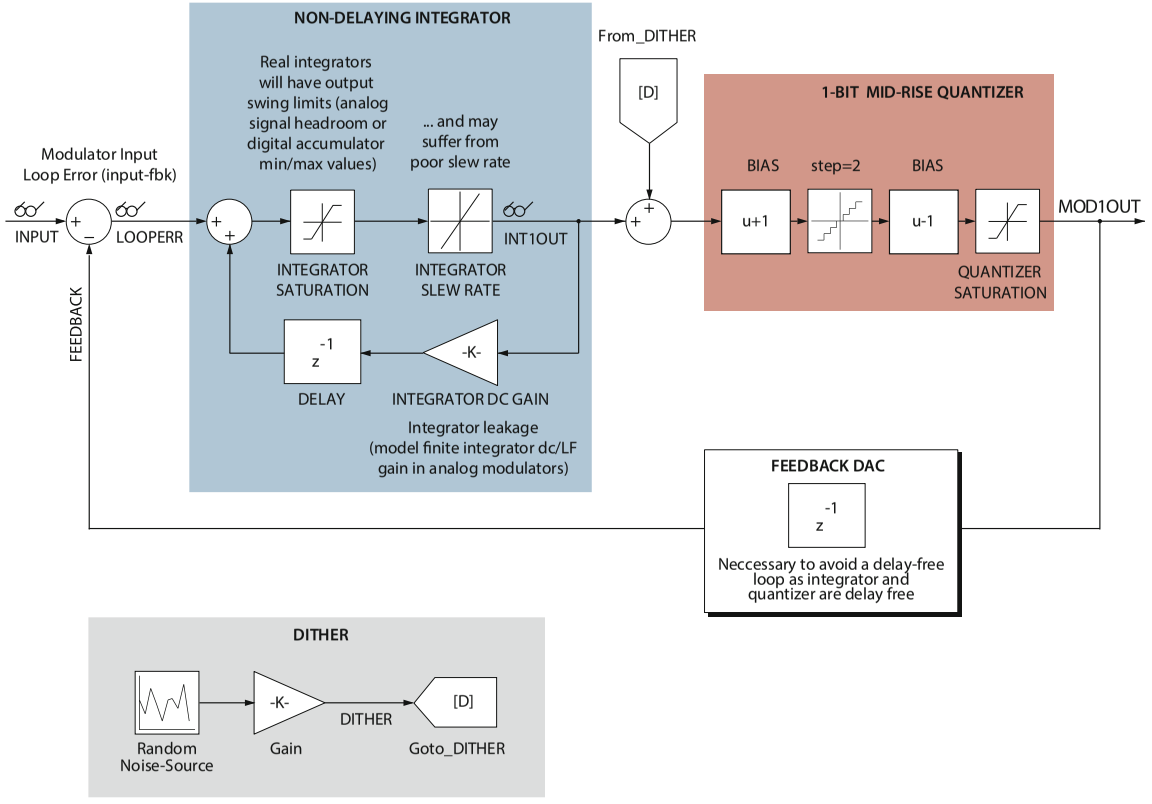


Fig. 2.1 SIMULINK® model of MOD1

- *Non-delaying integrator* [2] – The core block responsible for noise shaping in the modulator. Formed by a (discrete) $1/z$ unit delay element in feedback to an adder. To represent any finite integrator gain at DC, caused by the finite gain of an amplifier used to implement a real analog integrator, a gain block preceding the delay element has been added. The saturation block following the adder is used to represent the clipping effect of, for example, the finite power supply voltages or digital word lengths in real modulators. The rate limiter block is used to simulate the effect of limited slew rate. By default, all blocks are set to $\pm$inf (i.e., infinity), giving an ideal integrator.
- *Mid-rise quantizer* [2] – Single-bit, ± 1 output states and a step size of two. Since the default MATLAB® quantizer is *mid-tread* [2], the signal has been shifted up by +1 before and down by -1 after the quantizer element in order to implement the mid-rise behavior.
- *Delaying feedback loop* – From the quantizer output to the input, where the two signals are subtracted, and the error signal is applied to the integrator. The delay in the feedback represents the time to quantize, sample, and feed the integrator output back – usually a single clock cycle.
- *Dither source* – Generating white noise between ± 1. This is scaled by a gain block and summed at the input of the quantizer. By default, the dither is off.

In addition to these main functional elements shown in the picture, there are a number of additional inputs and output blocks in the model, including:

- A *DC* and *signal generator* inputs followed by an ideal sampler.
- A *time scope*, used for visualizing the various internal signals in the modulator.
- An *analog filter*, implemented to aid visualization of how the modulator digital output encodes the analog input signal. Note that a real ΣΔ-ADC would use a decimation filter (e.g., see [3]).

The default sampling frequency is chosen as $f_s = 3.072$ MHz with an oversampling ratio of OSR $= 64$. The signal band of interest, $f_b = f_s/(2 \cdot \text{OSR})$, corresponds to that of the audio band (i.e., from 0 Hz to 24 KHz). Further, note that since MOD1 uses a 1-bit quantizer, its DAC [1] is capable of perfectly linear operation (i.e., more on this in Chap. 5), although in practice this may not always be true (i.e., memory effects in the DAC, variations in voltage references, etc.). In the model, an ideal 1-bit DAC is assumed, and due to its inherent linearity and Simulink® operation process, no specific DAC block is required.

2.2 Operation for DC Inputs

The first task proposes to examine how MOD1 behaves in the case of DC input signals.

In the Toolbox open the model *"mod1.mdl"* found in the folder *2_MOD1*.

Type – *load_par* – in the MATLAB Command Window to initialize the variables required by the model.

In the model, set the DC input level to zero by typing – $dc = 0$ – into the MATLAB Command Window, and run the simulation.

Once the simulation is finished, open the time scope by double-clicking on any of the small blue glasses found in the signal connections to observe the waveform at the nodes of the modulator. The reader may wish to use the horizontal magnifying glass to zoom into a few cycles of the LOOPERR, INT1OUT, and MOD1OUT waveforms.

Type – *mod_SNDR* – into the MATLAB Command Window to obtain the output spectrum.

Save the result and repeat for the following DC input values: 0.5, 0.75, 0.875, and 0.9375.

As an example, Fig. 2.2 reports the time domain and FFT results for a DC input signal of 0.5. Similar results should be obtained for all the DC input values proposed.

From the simulations conducted, few aspects should be apparent:

- The value of the filtered bit sequence of the modulator output corresponds to that of the DC input signal, as it can be seen from the time domain results. Therefore, the conversion can be considered accurate.
- The modulator output is periodic.
- Examining the FFT results, noise shaping is not apparent, and strong tones are present in the spectrum.

2.2.1 DC Conversion Accuracy

The ability of MOD1 to accurately convert DC input signals is easily confirmed by a simple mathematical analysis [1]. Considering the basic MOD1 block diagram in Fig. 2.3 and assuming the quantizer Q have simplified characteristics such as:

$$v(n) = \mathrm{sgn}[y(n)] \tag{2.1}$$

A simple first-order difference equation can be written by following the signal flow in the diagram:

$$y(n) = y(n-1) + u(n) - v(n-1) \tag{2.2}$$

Such as, for $n = 1, 2, \ldots, N$:

$$y(N) - y(0) = \sum_{n=0}^{N} [u(n) - v(n-1)] \tag{2.3}$$

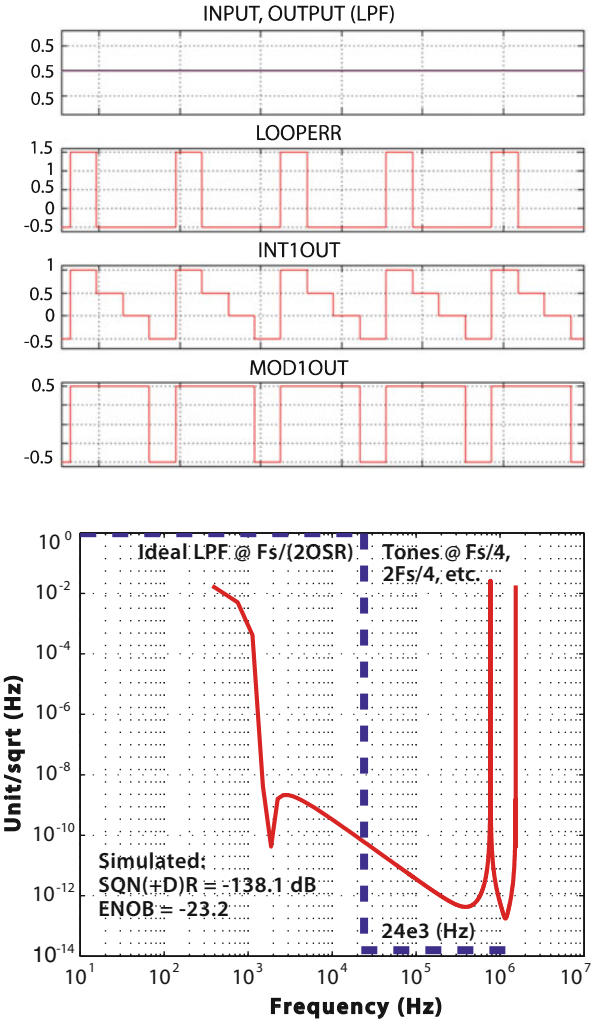


Fig. 2.2 (Top) Time domain and (bottom) FFT results for DC input signal 0.5

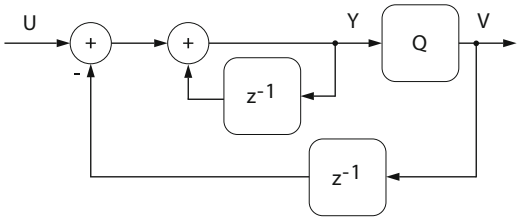


Fig. 2.3 MOD1 block diagram in z-domain

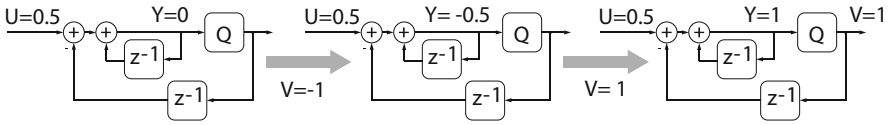


Fig. 2.4 Graphical representation of Eq. 2.4

Providing the loop is stable, so that $y(n)$ is bounded (i.e., finite), it follows that the average of the input samples equals that of the digital output v.

The validity of the equations presented is readily verified for the example of a 0.5 DC input considered. Assuming the initial conditions at $t = 0$ of $y = 0$ and $v = 1$, it follows that for subsequent clock cycles (e.g., Fig. 2.4):

$$y(n+1) = y(n) + [u(n+1) - v(n)] \text{ and } v(n+1) = v(n) \tag{2.4}$$

$$\begin{aligned}
&\text{Initial Conditions:} \quad u(t) = 0.5, y(0) = 0, v(0) = +\mathbf{1} \\
&y(1) = y(0) + [u(1) - v(0)] = 0 + (0.5 - 1) = -0.5, \qquad \text{so } v(1) = \mathbf{-1} \\
&y(2) = y(1) + [u(2) - v(1)] = -0.5 + [0.5 - (-1)] = 1, \quad \text{so } v(2) = +\mathbf{1} \\
&y(3) = y(2) + [u(3) - v(2)] = 1 + (0.5 - 1) = 0.5, \qquad \text{so } v(3) = +\mathbf{1} \\
&y(4) = y(3) + [u(4) - v(3)] = 0.5 + (0.5 - 1) = 0, \qquad \text{so } v(4) = +1
\end{aligned}$$

Continuing the analysis would show that the pattern of $v(n) = \pm 1$ repeats with a period of 4 as long as the DC input does not change. Note that such cyclical behavior is consequently also valid for the internal states at the integrator output $y(n)$. Comparing the mathematical results with those from the Toolbox (Fig. 2.2), it can be seen that the pattern is identical, hence verifying the validity of the equations presented. As discussed in Chap. 1, for a ΣΔ-ADC it is the average of the output sequence that provides the converted input, thus:

$$\frac{[v(0) + v(1) + v(2) + v(3)]}{4} \tag{2.5}$$

$$\frac{[3 \cdot (+1)] + [1 \cdot (-1)]}{4} = 0.5$$

As it can be seen there is no error, and the important conclusion of the analysis presented is that MOD1 is capable of converting a DC input signal with arbitrarily good accuracy, provided it is allowed to operate for a sufficiently long time, and it is followed by a perfect low-pass filter.

Example

Assuming the initial conditions at $t = 0$ of $y = 1$ for the DC input $u = 0.5$ studied, confirm that the same periodicity of the output signal is to be found and that the conversion is still accurate. What can be noticed regarding the output sequence $v(n)$ compared to the results obtained in the main text?

Solution
Since $y(0)$ is a positive value, the quantizer output must be positive. Therefore:

$$\begin{aligned}
&\text{Initial Conditions:} \quad u(t) = 0.5, \qquad y(0) = 1, \;\; v(0) = +\mathbf{1} \\
&y(1) = y(0) + [u(1) - v(0)] = 0,5 + (1 - 1) = 0.5, \qquad \text{so } v(1) = +\mathbf{1} \\
&y(2) = y(1) + [u(2) - v(1)] = 0.5 + [0.5 - 1] = 0, \qquad \text{so } v(2) = +\mathbf{1} \\
&y(3) = y(2) + [u(3) - v(2)] = 0 + (0.5 - 1) = -0.5, \qquad \text{so } v(3) = \mathbf{-1} \\
&y(4) = y(3) + [u(4) - v(3)] = -0.5 + (0.5 - (-1)) = 1, \quad \text{so } v(4) = +1
\end{aligned}$$

The same periodicity of the output signal is found (e.g., 4) but with the sequence of $v(n) = \pm 1$ differing, since it is dependent on the initial conditions of $y(0)$. Therefore, the ability to accurately convert DC signals is further confirmed, and theoretically it is independent from the initial conditions of the converter (i.e., note that in real circuits the initial conditions of, especially, the integrators may be of importance [1]).

2.2.2 Idle Tone Generation

Regarding the FFT result of the conversions proposed, noticeable tones are present in the spectrum. Considering the periodic behavior of the modulator output generated by DC inputs, a relation between tones and cyclic patterns results, such that, for the example of DC input = 0.5, tones appear at $f_s/4$ and its harmonics, where the denominator corresponds to the periodicity of the output sequence. Further theoretical investigation [1] shows that it is possible to predict the cyclic patterns of MOD1 and relative tones. Considering the case of a general rational input $u = a/b$, where $0 < a < b$, the output will be a periodic sequence of length b containing:

$$\frac{b+a}{2} \rightarrow +1's \tag{2.6}$$

$$\frac{b-a}{2} \rightarrow -1's \tag{2.7}$$

where the average output value corresponds to:

$$\frac{\left(\frac{b+a}{2}\right) - \left(\frac{b-a}{2}\right)}{b} = \frac{a}{b} \tag{2.8}$$

In the case of a or b being even, the period corresponds to $2b$; therefore:

$$a + b \rightarrow +1's \tag{2.9}$$

$$b - a \rightarrow -1's \tag{2.10}$$

with the average output value of:

$$\frac{(a+b)-(b+a)}{2b} = \frac{a}{b} \tag{2.11}$$

The validity of the equations presented can be verified by repeating the process of Eq. 2.2 or by inputting such values in the Toolbox and inspecting the time domain waveforms. From the outcomes of this mathematical analysis, note again the ability of MOD1 to accurately convert DC signals, with u corresponding to the average output value. Further, it results that a periodic output with a constant input u ($|u| < 1$) implies that u is rational. It follows that, if u ($|u| < 1$) is constant but irrational, the output cannot be periodic [1]. Although these last statements are true, note that even with irrational DC inputs some repetition would still be apparent due to the complex nonlinear behavior of MOD1. This can be verified in the Toolbox by trying irrational DC inputs of, for example, *pi*/10, 0.4367, etc. However, it should result that MOD1 behaves much more similarly to what expected when compared to rational DC inputs (e.g., noise shaping becomes apparent in the FFT, etc.). Since tones are eventually to be counteracted in any good design, further investigations on the complex nonlinear MOD1 behavior for DC inputs are not presented in this beginner book, but more information can be found in [4, 5] for the interested reader.

The periodic sequences investigated are called *pattern noise*, *idle tones*, or *limit cycles* (i.e., if the signal is present). They do not represent a non-ideal operation of the loop but are a consequence of the complex MOD1's dynamics. Further, their amplitude does not change with time but is a complicated function of u [1]. Their frequency also depends on the input, as already discussed. Further, note that in all the proposed examples tones appeared outside the signal band due to the default Toolbox settings provided, hence being of low interest since these are eliminated during output filtering (i.e., decimation). However, it should be noted that longer cyclic periods may generate low-frequency quantization noise to fall within the passband, hence severely degrading the converter performance. Long periodic sequences generate in the case of rational DC inputs with large denominators, and the reader is encouraged to try the DC value of *1/100* in the Toolbox for verification, noting how tones fall in the passband.

Example

Considering a DC input $u(t) = 5/7$, find the length of the repeating sequence $v(t)$ as well as the number of $\pm$ 1's expected. At what frequencies tones will appear in the spectrum? What is the minimum oversampling ratio that may be used to avoid baseband tones for appearing for these inputs? Repeat the exercise for the input $u(t) = 2/7$.

Solution

Since a and b are odd, the length of the repeating sequence would correspond to $b = 7$, while the numbers of $\pm$ 1's contained in it would be $(b + a)/2 = 6 \rightarrow +1'$ s and $(b - a)/2 = 2 \rightarrow -1'$s. Thus, the average output value would result to $(6 - 1)/7 = 5/7$. Considering the relation between tones and the cyclic period, tones will appear at $f_s/7$ and its harmonics. The minimum oversampling ratio that has to be used to avoid tones falling into the passband would correspond to $\mathrm{OSR}_{\min} = f_s/(2f_b) = f_s/(2f_s/7) = 3.5$.

In the case of a DC input $u(t) = 2/7$, note that the numerator, a, is even. Therefore, the length of the repeating sequence would correspond to $2 * b = 14$, while the numbers of $\pm$ 1's contained in it would be $(a + b) = 9 \rightarrow +1'$s and $(b - a) = 5 \rightarrow -1'$s.

Thus, the average output value would result to $(9 - 5)/14 = 2/7$. Tones appear at $f_s/14$ and its harmonics, while the minimum oversampling ratio to avoid tones in the passband would correspond to $\mathrm{OSR}_{\min} = f_s/(2f_b) = f_s/(2f_s/14) = 7$.

2.2.3 Tonal Behavior and DC Input Sweep

The previous analysis highlighted the highly tonal behavior of MOD1 for rational DC inputs and the simple relationship between the input level and resulting tone frequencies. To further comprehend the tonal behavior of MOD1, it is useful to investigate the effect of limit cycles on the signal band (i.e., from DC to $f_s/(2 \cdot \mathrm{OSR})$) across a swept range of input DC levels which creates long cyclic patterns.

Close *"mod1.mdl"*.

Type – *target_mod = 'sweep_mod1'* – in the MATLAB Command Window.

Open "*sweep_testbench.mdl*" found in the main Toolbox folder, right click on the *Modulator* block, and enter in the *Model Reference Parameters*. Change the *Model Name* to – *sweep_mod1* -, and press *OK*. Save and close the model.

Type – *sweep_dc_bbpwr* – in the MATLAB Command Window and run the routine → *Warning: the simulation analyzes the signal-band noise power for 1000 DC levels, hence it may take a long time to complete!*

Type – *OSR = 32* – in the MATLAB Command Window and repeat the exercise.

A plot of total baseband noise power in dBW is to be generated for input amplitudes from 0 to 1, the maximum allowed input, as in Fig. 2.5.

As discussed earlier, both the frequency and power of the tones are functions of the DC input u. Hence, so is the in-band noise which they introduce. Figure 2.5 illustrates that high tonal noise power (e.g., large peaks) occurs near simple rational values (e.g., 1/2, 1/3, etc.) and in particular around $u = 0$ and 1. Important to note is

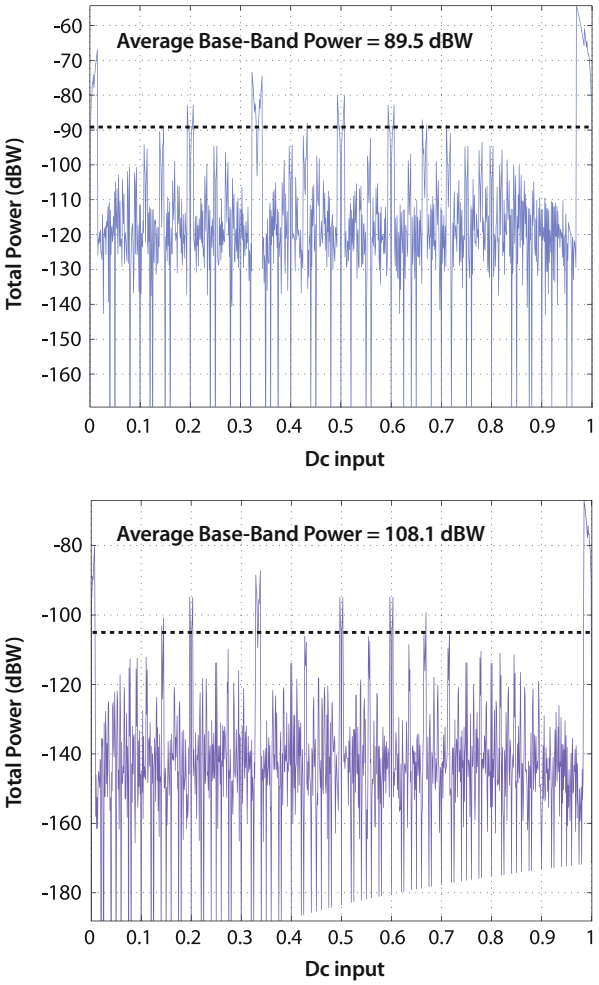


Fig. 2.5 Baseband noise power vs. DC input and average power over all inputs for MOD1 with (Top) OSR = 32 and (Bottom) OSR = 64

that the signal-band power is small at exactly simple rational values (i.e., note the notches in the noise features at these levels) since, as shown before, tones are to appear outside the passband. In other words [1]:

- DC input values with large rational denominator generate long tonal sequences which do not "average out" to the correct value within the digital filter bandwidth – Tone frequencies can be seen in the baseband of the FFT and correspond to maxima in the DC error plot of Fig. 2.5.
- DC input values with small rational denominators generate short tonal sequences which are averaged to the correct value by the digital filter – No tone frequencies in the baseband of the FFT and correspond to minima in the DC error plot of Fig. 2.5.

Thus, the analysis further highlights that long cyclic patterns are more problematic than the short sequences generated by simple rational DC inputs.

Another aspect shown in Fig. 2.5 is the mean-square levels of the in-band noise, averaged over all input values. Comparing the results note that, for higher OSR, the widths and absolute heights of the anomalous region decrease, but relative to the mean noise level, the noise peaks are higher. In [4] it has been demonstrated that the central noise peaks have a height of $-20\log\left(\sqrt{2}\mathrm{OSR}\right)$ dB and a width of $(\mathrm{OSR})^{-1}$. Further, [5] showed that the dominant pattern consists of two large spikes surrounding a mound of smaller spikes, and that this pattern is duplicated between adjacent pairs of the smaller spikes in an endless recursion. These studies are in accordance with the "*M*" shape, width, and height of the noise peaks in Fig. 2.5 around rational values.

At this point, it should be clear that the tonal behavior is a direct consequence of the nonlinear difference equations which govern MOD1. Consequently, it results that tones are a manifestation of violation of the assumption presented in Chap. 1, in which the quantizer and the relative quantization error can be approximated as an *additive white noise source* [6]. In fact, the quantizer's white noise model which allows to obtain a linearized model of MOD1, hence enabling a simpler mathematical analysis, can only be valid under specific conditions, including [2]:

- The quantization noise and input signal must be uncorrelated, which implies that large and fast random variation on the input y to the quantizer must occur. This can only be satisfied if the input u is a relatively large signal with fast random variations.
- The quantizer does not overload.
- The quantization steps are uniform.

According to the first point, it is clear that DC input signals cannot meet such specification since these do not vary over time. Further, since the quantization levels can be equiprobable only for signals with uniformly distributed amplitude, it results that idle tones may generate also in the presence of slowly varying input signals which stay near a critical level (e.g., a simple rational value) long enough for a limit cycle to become apparent [6]. This is particularly likely to occur for low-order modulators, such as MOD1, when sinusoidal inputs are used [1].

2.2.4 Tone Prevention

The prevention of tone generation is an important aspect of ΣΔ-modulator design since these often cannot be tolerated (e.g., in audio applications tones even 20 dB below the level of any white noise present can be detected by the human ear [7]). To counteract tones two of the most popular techniques include to use high-order modulators, as discussed in later chapters, and to use *dither* [2], which is an added pseudorandom signal. Although dither randomizes the input of the quantizer, the

technique requires an additional signal source and also typically reduces the dynamic range of the ADC [1, 8]. To appreciate the effect of dither, in the Toolbox:

> Type – *mod.dither = 0.2* – in the MATLAB Command Window to add some dither.
>
> Type – *sweep_dc_bbpwr* – in the MATLAB Command Window and run the routine.

In the case of OSR = 64, a plot of the baseband noise power as the one illustrated in Fig. 2.6 should result.

As it can be seen, dither reduces maxima in DC error sweep (i.e., smaller peaks) but also increases minima (i.e., higher noise level) since noise has been added to the system. Therefore, the amplitude of dither to be used in the system is chosen as a compromise between reducing the SQNR (i.e., caused by the increase of minima) and reduced tone amplitudes. To further prove the efficiency of dither, the reader is encouraged to repeat the first simulation proposed in Sect. 2.2 of the model – *mod1.mdl* – with a DC input of 1/100 and dither of 0.2. Similar results of the ones in Fig. 2.7 should be obtained.

As it can be seen some disruption of the long periodic sequence to the input of the quantizer are introduced, and, since the modulator behaves more closely to its linearized model, less and smaller tones are seen in the FFT graph. Further note that the noise-shaping characteristic becomes apparent and in accordance with a first-order loop filter (e.g., 20 dB/decade slope). Thus, it is clear that adding dither is an efficient technique to reduce the tonal behavior of MOD1. Although it is out of the scope of this book to discuss all the various dithering techniques, such as whether the dither must be white noise or not (i.e., Gaussian, triangular, etc.), note that [9, 10] may be considered good introductory texts for investigations into the topic.

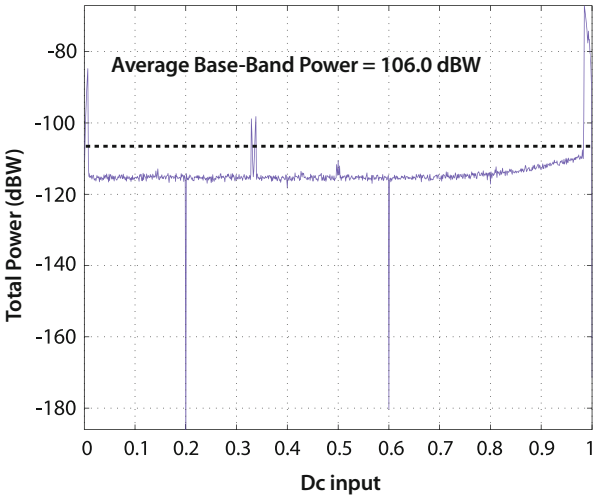


Fig. 2.6 Baseband noise power vs. DC input and average power over all inputs for MOD1 with OSR = 64 and dither = 0.2

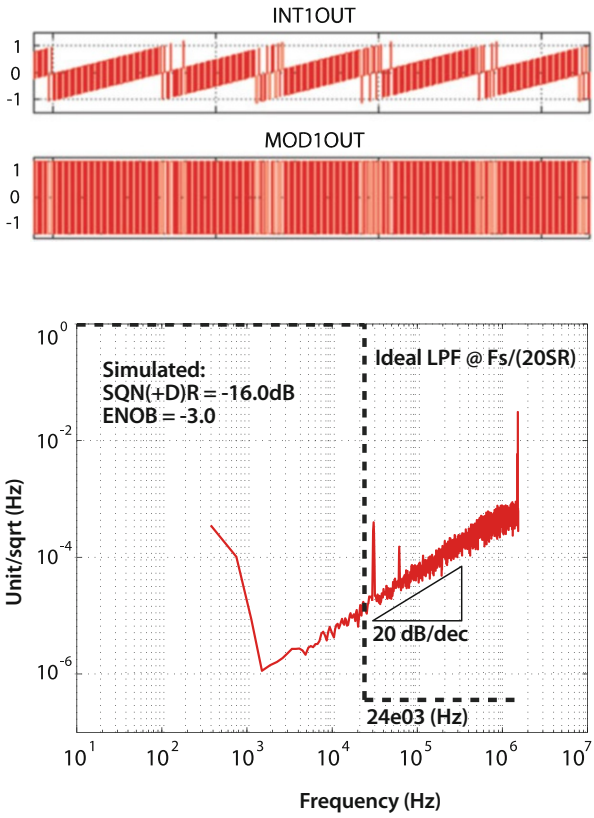


Fig. 2.7 (Top) Time domain and (bottom) FFT for MOD1 with DC input = 1/100 and dither = 0.2

2.3 Stability of MOD1

An important aspect of ΣΔ-modulators is the stability of the loop.

In the Toolbox open the model *"mod1.mdl"* found in the folder 2_MOD1.

Set the DC input level to – *dc = 0.99* – and run the simulation.

Once the simulation is finished, open the time scope by double-clicking on any of the small blue glasses found in the signal connections, and save the results.

Repeat the simulation for a DC input of 1.02.

Comparing the results it should be apparent that as soon as the amplitude of the input signal exceeds 1, the modulator becomes unstable, with the integrator output (i.e., INT1OUT) growing monotonically (i.e., or decreasing monotonically in the case of a negative input signal).

Despite first-order modulators being inherently stable, as linear analysis based on Bode plots would predict since the loop gain decreases by −6 dB/octave and the loop phase is −90° at all frequencies [3], the prediction does not take into account the actual signal processing performed by the quantizer. Hence, time domain considerations taking into account nonlinearities are required. Intuitively, in the example of $u = 1.02$, the DAC tries to balance u by feeding back a signal of −1 at every clock cycle. Even so, a net input of 0.02 enters in the integrator until y grows so large that the circuit becomes dysfunctional. Vice versa, if $|u| < 1$ and the initial value of y satisfy $|y(0)| \leq 2$, then the loop remain stable, with $|y|$ bounded by 2. This stability condition is sufficient for time-varying input signals as well [1, 8]. Important to note is that first-order modulators can recover from instability, provided the input is restored to $|u(t)| < 1$. This is not the case for high-order modulators, as demonstrated in later chapters.

2.4 The Effects of Finite Integrator Gain

The simulations conducted so far assumed an ideal integrator with infinite gain at DC. This is relatively easy to achieve in digital modulators, but, in analog modulators, the integrator gain

at DC will be finite owing to things like finite amplifier gains, leakage currents discharging integration capacitors, etc. [8]. Therefore, the next task aims to introduce the effects of finite integrator gain on the time domain waveforms and DC transfer function of MOD1.

In the Toolbox open the model *"mod1.mdl"* found in the folder *2_MOD1*, and type – *load_par* – in the MATLAB Command Window for initialization of model parameters.

Introduce a finite DC gain error in the modulator by typing – *mod.igain1 = 50* – in the MATLAB Command Window.

Set – *psdset.win = 'Hann'* – by typing it into the MATLAB Command Window, and run the simulation.

Once the simulation is finished, open the time scope by double-clicking on any of the small blue glasses found in the signal connections.

Type – *mod_SNDR* – into the MATLAB Command Window to obtain the output spectrum and save the results.

Try few input DC values, such as 0.4367, 1/10, 0.011, etc.

Considering a DC input of 1/10, important results similar to those in Fig. 2.8 should be obtained.

The first thing to be noticed is that the modulator is no longer an accurate ADC since the input and output DC levels do not match. Recalling that an ideal integrator gain [2] tends toward infinity for input frequencies tending toward DC, it results that

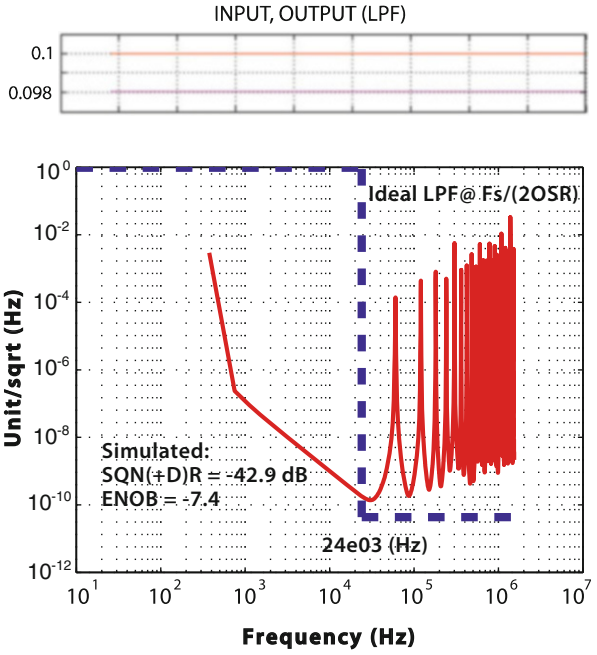


Fig. 2.8 (Top) Time domain and (bottom) FFT results for MOD1 with DC input = 1/10

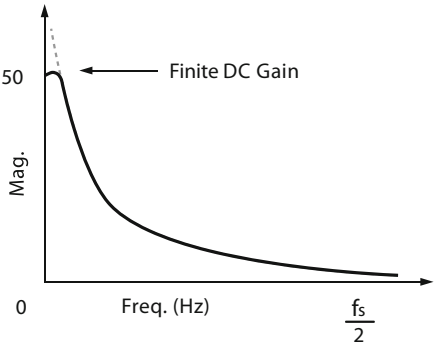


Fig. 2.9 Effect of finite integrator gain on the integrator transfer function

for a finite gain the integrator behaves similarly to a low-pass filter with a defined DC gain, as illustrated in Fig. 2.9. Therefore, MOD1 loses its ability to achieve infinite precision with DC signals, due to the limited gain.

Another consequence of finite integrator gain is reflected in the higher tonal behavior that can be seen in the FFT results when compared to the ideal case of infinite integrator gain. To appreciate this [1], note that the limit cycles of an ideal MOD1 with a DC input are unstable or *non-attracting* because arbitrarily small shifts in the input eventually lead to large changes in the integrator state and hence a different output pattern. However, in the case of finite gain, the resulting limit cycles are stable or *attracting* because sufficiently small shifts in the input lead to small

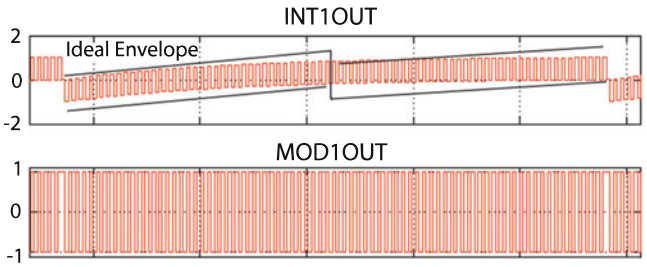


Fig. 2.10 Time domain results for MOD1 with DC input = 0.011

changes in the integrator state and no change in the output pattern. This is a detrimental effect since limit cycles are undesirable. To better appreciate this concept, it is easier to investigate the DC input of 0.011 in the Toolbox and inspect the time domain results of the integrator and MOD1 output sequence (Fig. 2.10).

Comparing the results with an ideal envelope (i.e., infinite integrator gain), it is seen that if the shifts in the input are too small relative to the finite integrator gain, too small changes in the integrator state occurs, and consequently there is no change in the output pattern. Therefore, the integrator output "leaks away," and longer periodic patterns are formed, explaining the higher tonal behavior.

To further comprehend the consequences of finite integrator gain, it is also useful to investigate a DC sweep across small DC levels.

Close the model *"mod1.mdl"*.

Type – *target_mod = 'sweep_mod1'* – in the MATLAB Command Window.

Open "*sweep_testbench.mdl*" found in the main Toolbox folder, right click on the *Modulator* block, and enter in the *Model Reference Parameters*. Change the *Model Name* to – *sweep_mod1* -, and press *OK*. Save and close the model.

Type – *sweep_dc_dc* – in the MATLAB Command Window and run the routine. A DC sweep for a subset of the input range (i.e., ±0.1) will be performed.

Figure 2.11 illustrates the results to be obtained, where the bright line represents the ideal DC transfer function of MOD1 and the darker line the one resulting in the case of finite integrator gain. The flat regions where the modulator output is the same for all modulator inputs, in the case of non-ideal conditions, are usually referred as *dead zones* [1, 8].

The strongest dead zone is seen around a DC input of 0, but small additional dead zones can be noted around other rational inputs. Such results further highlight the inability to achieve an infinite precision of DC conversions in the case of finite integrator gain.

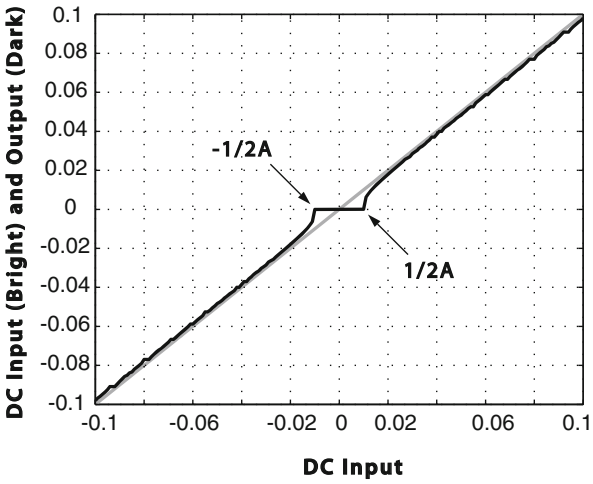


Fig. 2.11 MOD1 DC transfer function around DC input = 0, $A = 50$ with integrator gain of 50

Naming the finite integrator gain value as A, and zooming on the dead zone around DC input of 0, it can be verified that inputs smaller than $1/(2A)$ in normalized value have no effect on the modulator output. The derivation of such simple formula can be found in [1]. Further, it can be noted that for inputs close to the dead zones, the error quickly decays and gets closer to its ideal value. The inputs contained in this "decaying" zones are the main enablers of long cyclic patterns and relative increase in tones, as seen in the example of DC = 0.011.

From the results observed, it appears that a finite integrator gain is detrimental to the system. However, the essence of ΣΔ-converters is their ability to utilize low-precision elements, and one of the possible solutions already seen to counteract long cyclic patterns is to use dither. Following the increased tonal behavior caused by finite integrator gain, the implementation of dither into the system may be intuitive to investigate if the modulator behavior can be, somehow, improved.

Type – *mod.dither = 0.3* – into the MATLAB Command Window.
Type – *sweep_dc_dc* – in the MATLAB Command Window and run the routine.

Performing the simulation, similar results to the one in Fig. 2.12 should be obtained:

According to the results of Fig. 2.12, dither can, indeed, recover dead zones by preventing the modulator from locking into limit cycles. It results that the modulator behaves much more like the ideal case of infinite integrator gain. However, note that in order to recover the strongest dead zones around the DC input of 0, a large amount of dither would be required, which is unpractical. This is because, as mentioned, dither also reduces the achievable SQNR of the system as well as the dynamic range of the ADC. Therefore, as demonstrated later, increasing the loop-filter order usually results in a more convenient solution when high DC precision is required.

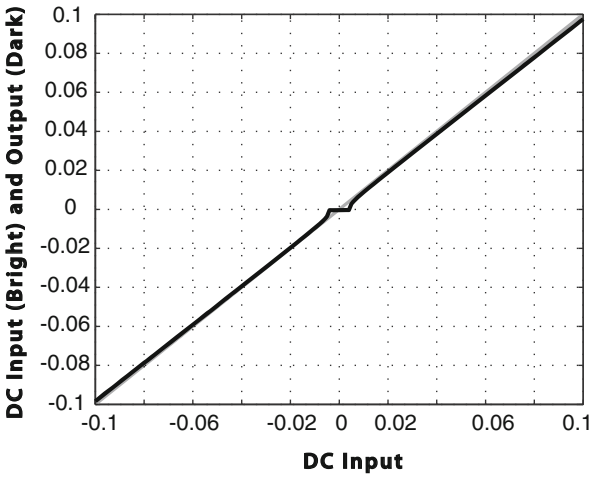


Fig. 2.12 MOD1 DC transfer function around DC input = 0 with integrator gain of 50 and dither of 0.3

Example

Calculate the amplifier gain required to reduce the dead zone range from −50μV to +50μV, without the use of dither, of a first-order sigma-delta modulator with feedback levels + 1 V/−1 V.

Solution

The dead zone of a first-order modulator has width $1/A$ so $A = 1/100e{-}6 = 10000$.

2.5 Operation for Sinewave Inputs

The next task proposed is to examine the behavior of MOD1 in response to sinewave inputs. In the Toolbox the variables "*sinfreq*" and "*sinamp*" have a default value of 12 KHz and 0.707, respectively, setting the frequency and amplitude of the input sinewave. Important to note is that the 12 KHz frequency has been chosen such that an integer multiple of complete periods of the sinusoid are present within the 8192 samples acquired by the FFT routine, and the sinewave frequency falls exactly into one of the PSD bins [11]. Such measures are generally necessary to avoid effects like spectral leakage (i.e., even if *windowing* is being used) or waveform truncation from corrupting the simulation results. Therefore, when conducting sinewave simulations, the number of frequencies which can be successfully simulated may often be limited due to the dual need of satisfying a complete number of waveform cycles in a power-of-2 sample window length and the signal being in an exact FFT bin. *Consequently, note that careless frequency choices can easily generate plots dominated by FFT artifacts instead of the real modulator behavior, especially at the high SNR values which are usually of interest.*

Reset initial conditions by typing – *load_par* – into the MATLAB Command Window.

Open the model "*mod1.mdl*" found in the folder 2_MOD1.

In the model double-click on the *manual switch* to select the *signal generator* input source, and run the simulations.

Once the simulation is finished, open the time scope by double-clicking on any of the small blue glasses found in the signal connections to observe the waveform at the nodes of the modulator. The reader may wish to use the horizontal magnifying glass to zoom into a few cycles of the LOOPERR, INT1OUT, and MOD1OUT waveforms.

Type – *mod_SNDR* – into the MATLAB Command Window to obtain the FFT graph.

Similar results to those illustrated in Fig. 2.13 should be obtained.

The first thing to note in the time domain graphs is that, as already seen in Chap. 1, the varying density of ± 1's at the modulator output resembles the converted sinewave since it is the average of the entire output code that defines the waveform. Ignoring any phase shift between the input and output waveforms, caused predominantly by the low-pass filter, it can be seen that the modulator is able to accurately convert the input. Further, according to the FFT plot, it is seen that the noise-shaping characteristic of a first-order filter of 20 dB/decade slope is now recognizable, hence confirming the correct operation of the modulator. However, the highly tonal behavior of MOD1 is evident even in the case of varying signals, as already predicted in Sect. 2.2.3.

Repeating the exercise for lower frequencies (e.g., by typing – $sinfreq = 6000$ – in the MATLAB Command Window), it can be seen that such strong tones fall into the passband, which is unacceptable. It is clear that MOD1 is hardly implementable without recurring to tone prevention techniques, such as dither.

It is also useful to repeat the simulation for amplitudes larger than the maximum input range allowable of ± 1. Typing – $sinamp = 1.1$ – into the MATLAB Command Window for $sinfreq = 12000$ (i.e., to avoid tones in the passband) and running the simulation, similar results to that of Fig. 2.14 are to be obtained.

This simulation demonstrates that as long as the input signal stays bounded to $|u(t)| < 1$, and hence the integrator output to $|\, y(n)\,| \leq 2$, the modulator is stable, and the correct code is obtained, as discussed in Sect. 2.3. Vice versa, as soon as the peak amplitude of the input sinewave exceeds the bound of ± 1, the modulator becomes unstable, and the output code locks itself to either plus or minus 1 according to the polarity of its input signal. Further, the ability of MOD1 to recover from instability is clearly noticeable since as soon as the amplitude of the input sinewave is restored to its ideal bounded value, the modulator is capable of keeping providing the correct output code sequence. As mentioned, note that this is not the case for high-order modulators.

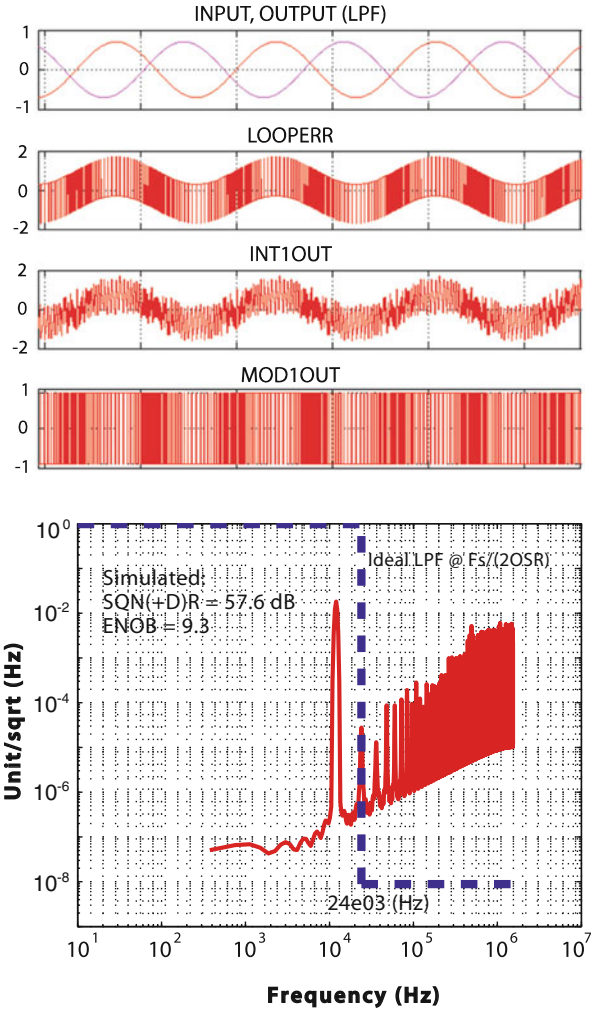


Fig. 2.13 (Top) Time domain and (bottom) FFT results for MOD1 with sinewave input

2.5.1 Sinewave Amplitude Sweep

This paragraph investigates the achievable signal-to-quantization noise ratio (SQNR) of MOD1 in response to a range of sinewave amplitudes while providing further insights into the non-ideal behaviors discussed so far.

Reset initial conditions by typing – *load_par* – into the MATLAB Command Window.

Type – $Fs = 1e6$ – and – $sinfreq = 4882.8$ – into the MATLAB Command Window. The sampling frequency and input signal frequency are here changed in

(continued)

order to obtain graphs that are nicer to visualize and less affected by the complex dynamics of MOD1. However, the principles underlying the results here described are obviously valid for any sampling frequency and/or input signal frequency the reader may choose.

Type – *sweep_sinamp* – into the MATLAB Command Window and run the simulation. Note that it may take some time to complete.

Once the simulation is completed, a plot of signal-to-quantization noise ratio versus input amplitude in decibels (dBFS), where 0 dBFS corresponds to an amplitude of 1, should appear. The graph is expected to be similar to the one reported in Fig. 2.15.

To verify the peak value of SQNR achieved, it is useful to compare its value with the expected theoretical SQNR from the formula [2.1]:

$$\text{SQNR} = 6.02N + 1.76 + (20L + 10)\log_{10}(\text{OSR}) - 10\log_{10}\left(\frac{\pi^{2L}}{2L+1}\right) \cong 56.8\ (\text{dB}) \tag{2.12}$$

where the modulator order is $L = 1$, the oversampling ratio is $\text{OSR} = 64$, and the number of bits corresponds to $N = 1$.

As it can be seen the peak SQNR found from the simulation is consistent with the predicted theoretical value, confirming the correct operation of MOD1. The uneven SQNR slope is due to the presence of limit cycles, while for a maximum signal amplitude of 0 (dBFS), or 1, the SQNR is much lower than expected because the

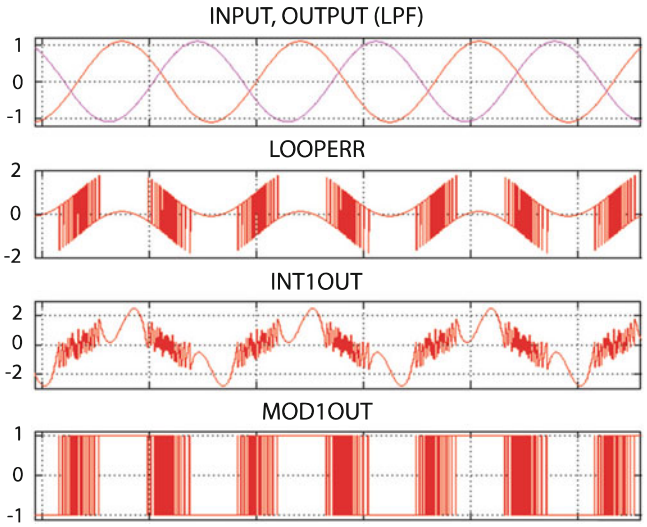


Fig. 2.14 Time domain results for MOD1 with sinewave input of amplitude 1.1

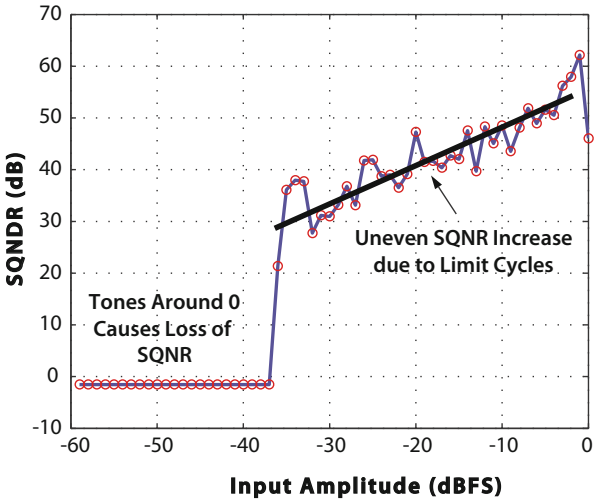


Fig. 2.15 SQNR vs. swept sine input amplitude for MOD1

bound of $|u(t)| < 1$ is violated at the peak value itself. Further, note the SQNR below the input level of -35 (dBFS). This is caused by the fact that there is no signal at those input levels, so the SNR $= 0$. This is not a dead zone but an artifact of MOD1's tendency to tonal behavior [1]. This behavior is highly input dependent and, for example, running the simulation again by using $sinfreq = 610.35$(Hz) would provide a much "better behaved" result. The reader may wish to try for verification. Such complex behavior is the reason why in this simulation a different sampling frequency has been chosen, in order to achieve results that are nicer to examine since less affected by such non-ideal artifacts.

Since dither is likely to be used in any MOD1 design in order to prevent limit cycles, it is interesting to study its effect to the SQNR for the different amplitudes tested in the sweep simulation.

Type – *mod.dither = 0.2* – into the MATLAB Command Window and then – *sweep_sinamp* – to run the simulation.

Figure 2.16 illustrates the result.

It is clear that since dither prevents tones from appearing in the spectrum, the modulator is able to behave much more like its ideal model. However, as already examined during DC input simulations, dither increases the minima of the noise in the system, and so the maximum SQNR achievable is reduced (e.g., approximately from 57 dB to 53 dB in the example considered). The interested reader may wish to rerun the simulation with greater values of dither to see if the modulator behavior would improve even further. Provided correct simulation, however, it should result that this is not the case and that the peak SQNR rapidly degrades as more dither is introduced in the system.

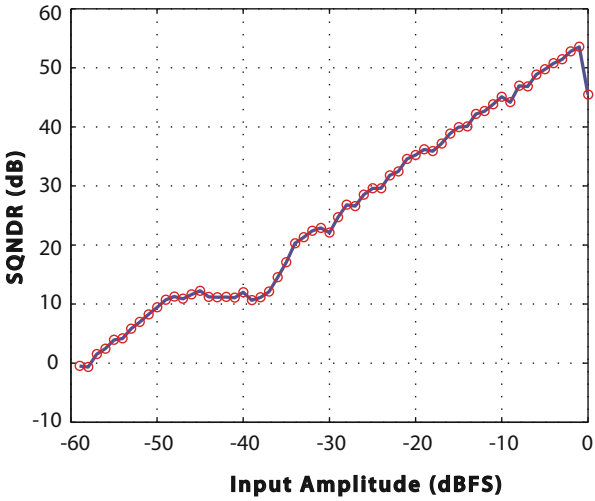


Fig. 2.16 SQNR vs. swept sine input amplitude and dither = 0.2 for MOD1

The last suggested task involves running the sweep in the conditions of finite integrator gain.

Reset the dither to zero – *mod.dither = 0.*
Type – *mod.igain1 = 50* – into the MATLAB Command Window.
Type – *sweep_sinamp* – into the MATLAB Command Window and run the simulation.

According to the results presented in Fig. 2.17 and from previous analysis with DC inputs, it is further confirmed that finite integrator gain degrades the modulator efficiency. Comparing the results with Fig. 2.15, it can be seen that for the strongest dead zone around small inputs, the SQNR further decreases to approximately −10 dB. Moreover, note how the signal range is also reduced, from −35 dBFS of Fig. 2.15 to approximately −30 dBFS. Applying dither would, to some extent, improve the modulator performance; however, note that for small inputs such degraded SQNR would still occur due to strong dead zones.

2.6 Theoretical Analysis

The mathematical analysis of the discrete MOD1, as found in [1], is here reported in order to gain a broader understanding of its properties. Note that a delaying integrator is used in order to demonstrate that it does not affect the operation of the modulator. Analyzing the model in the z-domain (Fig. 2.18), it results to be:

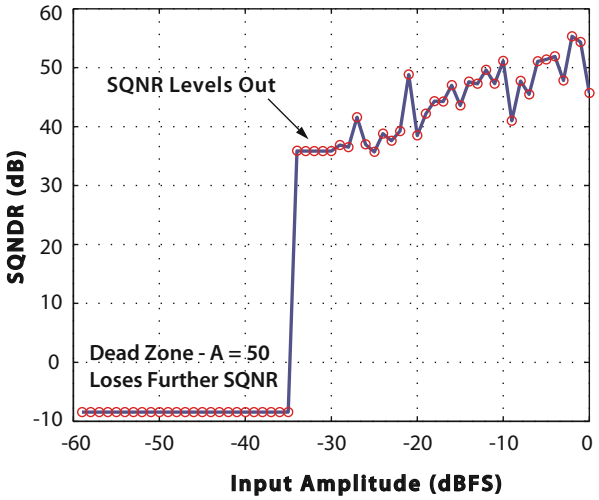
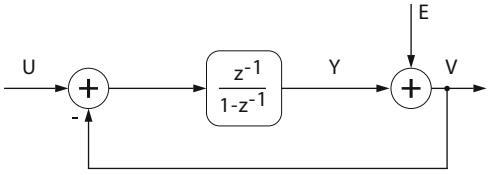


Fig. 2.17 SQNR vs. 4882.8 Hz sine input amplitude and finite integrator gain of 50 for MOD1

Fig. 2.18 Simplified MOD1 linearized block diagram



$$
\begin{aligned}
V(z) &= Y(z) + E(z) = z^{-1}Y(z) + U(z) - z^{-1}V(z) + E(z) \\
V(z) &= U(z) + E(z) - z^{-1}(V(z) - Y(z)) = U(z) + E(z) - z^{-1}E(z) \\
V(z) &= z^{-1}U(z) + (1 - z^{-1})E(z)
\end{aligned}
\tag{2.13}
$$

where:

$$
\begin{aligned}
\mathrm{STF}(z) &= z^{-1} \\
\mathrm{NTF}(z) &= 1 - z^{-1}
\end{aligned}
\tag{2.14}
$$

Therefore, the output is a delayed replica of the input while the error is differentiated. Continuing the analysis to investigate the noise-shaping properties of the modulator, it results to be:

$$
\begin{aligned}
|\mathrm{STF}|^2 &= \left|z^{-1}\right|^2 = 1 \\
|\mathrm{NTF}|^2 &= \left|1 - z^{-1}\right|^2 = |1 - \mathrm{e}^{-j\omega T}|^2 = |1 - \cos(\omega T) + j\sin(\omega T)|^2 \\
|\mathrm{NTF}|^2 &= (1 - \cos(\omega T))^2 + 2(\sin(\omega T)) = \left(2\sin\left(\tfrac{\omega T}{2}\right)\right)^2
\end{aligned}
\tag{2.15}
$$

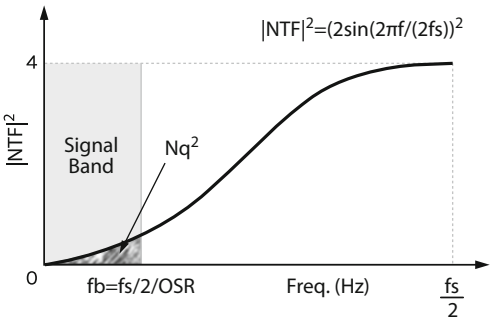


Fig. 2.19 MOD1's noise-shaping property

where $\omega = 2\pi f$ and $T = 1/f_s$. Plotting the $|\text{NTF}|^2$ against the frequency, it is seen that the NTF has a high-pass characteristic; therefore, the noise is suppressed at low frequencies and moved to f_s (Fig. 2.19).

The noise left in the baseband, N_q^2, can be estimated from:

$$N_q^2 = \frac{1}{\pi}\int_0^{\frac{\pi}{\text{OSR}}} \sigma_q^2 |\text{NTF}|^2 d\omega \tag{2.16}$$

$$N_q^2 = \frac{\sigma_q^2}{\pi}\int_0^{\frac{\pi}{\text{OSR}}} \left(2\sin\frac{\omega}{2}\right)^2 d\omega$$

$$N_q^2 \cong \frac{\sigma_q^2}{\pi}\int_0^{\frac{\pi}{\text{OSR}}} \omega^2 d\omega \quad \text{assuming OSR} \gg 1$$

$$N_q^2 = \frac{\sigma_q^2 \pi^2}{3\text{OSR}^3}$$

where σ_q^2 is the power spectral density of the quantization noise. The fundamental conclusion to be drawn from Eq. 2.16 is that doubling the OSR reduces the noise power by a factor of 8, which corresponds to 9 dB or 1.5 bit, thus confirming what discussed in Chap. 1.

Assuming a mid-rise quantizer with LSB step of 2 (as in the simulations conducted), the quantization noise power is $\sigma_q^2 = 1/3$. Since the SQNR = *10*log *(signal power/quantization noise power)* in dB and assuming a sinewave input of amplitude A (rms = $0.707A$ and power $0.5A^2$), an alternative SQNR formula to that of Eq. 2.12 can be found [1]:

$$\text{SQNR} = 10\log_{10}\left(\frac{9A^2\text{OSR}^3}{2\pi^2}\right) \tag{2.17}$$

Equation 2.17 is important because it highlights the fact that in order to achieve a high SQNR (e.g., $\geq$ 100 dB) the first-order modulator would require high OSR, which is impractical.

Example

A first-order modulator achieves 100 dB in SNR while operating at a sampling frequency of 10 MHz with feedback levels ± 1 V. Calculate the signal bandwidth that can be obtained.

Solution

$$\mathrm{SNR} = 10\log\left(\frac{9\mathrm{OSR}^3}{2\pi^2}\right) \rightarrow 10^{\left(\frac{48}{10}\right)} = \frac{9\mathrm{OSR}^3}{2\pi^2}$$

$$\mathrm{OSR} = \sqrt[3]{\frac{\left(10^{\left(\frac{100}{10}\right)} \cdot 2\pi^2\right)}{9}} \cong 2799$$

This implies a baseband of 10 MHz/(2 × OSR) = 1.8 KHz.

2.7 Conclusions and Essential Takeaways

This chapter has examined some key, fundamental properties of ΣΔ-modulators by studying specifically the behavior of the basic first-order modulator. Hopefully being able to experiment with a modulator in simulations and seeing effects like noise shaping, DC tone generation and the effects of dither in action, among others, will aid the understanding of the various significant basic principles underlying all ΣΔ-modulator operations covered in this book.

The reader may also have noted that despite being the obvious starting point for modulator study, MOD1 is actually quite a dreadful modulator, being highly prone to:

- DC limit cycles (tonal spectrum with no noise shaping)
- Idle tones with sinewave or irrational DC input (noise shaping evident, but with strong tonal behavior also present)
- Wide dead zones unless the integrator DC gain is very high

Consequently, MOD1 is not widely used on its own. Applying dither helps greatly, but in practice with real modulators having limited analog performance that tends to overload single-bit MOD1s (e.g., limited integrator output swing, etc.), their implementation is often unsatisfactory in terms of performance. Further, offering only first-order noise shaping, MOD1 is also limited in the achievable SQNR for moderate oversampling ratios. For these reasons, MOD1, especially with a 1-bit quantizer, is not commonly seen, even for the least demanding applications. However, as presented in later chapters, MOD1 does often appear as part of certain cascaded modulators called MASH and SMASH converters.

The next chapter, Chap. 3, introduces second-order modulators that behave much more like ideal ΣΔ-modulators (and so will hopefully be a much better aid than

MOD1 in gaining a better insight into and understanding of modulator behavior), that give far better SQNR at moderate oversampling ratios, and that are thus much more commonly used.

Exercises

Q.1

The modulator in Fig. 2.20 is formed by a non-delaying integrator. How the STF and NTF differ from the case where a delaying integrator is used? Is Fig. 2.11 entirely correct? What considerations should be taken into account?

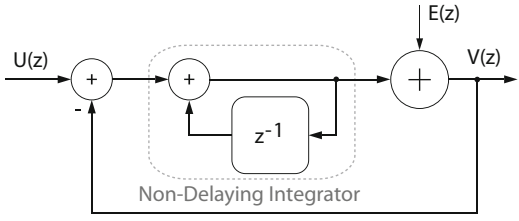


Fig. 2.20 First-order modulator with a 1-bit quantizer

Q.2

A first-order modulator has a 1-bit quantizer with full-scale range + 1/−1 and takes a DC input $u(t) = 3/7$. What length of repeating sequence $v(t)$ do you expect? How many +1's and how many −1's will this contain?

Q.3

The same modulator in Q.2 now takes a DC input $u(t) = 2/7$. What length of repeating sequence $v(t)$ do you expect? How many +1's and how many −1's will this contain?

Q.4

Using the time-difference equations and assuming the integrator output a $t = 0$ $y(0) = 0$t and feedback state is $v(0) = 1$, compute a sequence of quantizer outputs $v(nT)$ until the sequence repeats for a dc input $u(t) = 3/7$.

Q.5

Using the time-difference equations and assuming the integrator output at $t = 0$ $y(0) = 0$ and feedback state is $v(0) = 1$, compute a sequence of quantizer outputs $v(nT)$ until the sequence repeats for a DC input $u(t) = 2/7$.

Q.6

Sketch an approximate magnitude FFT of the waveforms obtained in *Q4* and *Q5*. What is the minimum oversampling ratio that may be used to avoid baseband tones for appearing for these inputs?

Q.7
Calculate the baseband noise power of a first-order modulator with oversampling ratio 256 and feedback levels ±1 V.

Q.8
Calculate the signal-to-quantization noise ratio of a first-order modulator with input amplitude −3 dB FS (full scale), oversampling ratio 256, and feedback levels ±1 V.

Q.9
A first-order modulator is to achieve a 16-bit resolution and operate at a sampling frequency of 10 MHz with feedback levels ±1 V. Calculate the signal bandwidth that can be obtained.

Q.10
Calculate the amplifier gain required to reduce the dead zone width of a first-order sigma-delta modulator with feedback levels ±1 V to below 100 μV.

Q.11
Starting from the NTF's high-pass characteristic previously demonstrated in Eq. 2.15:

$$|\mathrm{NTF}(f)| = 2\sin\left(\frac{\pi f}{f_\mathrm{s}}\right)$$

and considering that the quantization noise power over the frequency band 0 to f_b is now given by:

$$P_\mathrm{noise} = \int_{-f_\mathrm{b}}^{f_\mathrm{b}} S_\mathrm{e}^2(f)\cdot|\mathrm{NTF}|^2 df = \int_{-f_\mathrm{b}}^{f_\mathrm{b}} \left(\frac{\Delta^2}{12}\right)\frac{1}{f_\mathrm{s}}\left[2\sin\left(\frac{\pi f}{f_\mathrm{s}}\right)\right]^2 df$$

it is convenient to make the assumption $f_\mathrm{b} \ll f_\mathrm{s}$ (i.e., OSR $\gg$ 1) so that $sin\left(\frac{\pi f}{f_\mathrm{s}}\right)$ can be approximated to be $\frac{\pi f}{f_\mathrm{s}}$:

$$P_\mathrm{noise} = \int_{-f_\mathrm{b}}^{f_\mathrm{b}} \left(\frac{\Delta^2}{12}\right)\frac{1}{f_\mathrm{s}}\left[2\left(\frac{\pi f}{f_\mathrm{s}}\right)\right]^2 df = \int_{-f_\mathrm{b}}^{f_\mathrm{b}} \left(\frac{\Delta^2}{12}\right)\left(\frac{1}{f_\mathrm{s}}\right)\left(\frac{4\pi^2}{f_\mathrm{s}^2}\right) f^2 df$$

Letting $K = \left(\frac{\Delta^2}{12}\right)\left(\frac{1}{f_\mathrm{s}}\right)\left(\frac{4\pi^2}{f_\mathrm{s}^2}\right)$:

$$P_{\text{noise}} = K \int_{-f_b}^{f_b} f^2 df = \frac{K}{3}\left(f_b^3 - (-f_b)^3\right) = \frac{K}{3} \cdot 2f_b^3$$

$$P_{\text{noise}} = \frac{\Delta^2}{12} \cdot \frac{1}{f_s} \cdot \frac{4\pi^2}{3f_s^2} \cdot 2f_b^3 = \frac{\Delta^2}{12} \cdot \frac{\pi^2}{3} \cdot \left(\frac{2f_b}{f_s}\right)^3$$

and using $\text{OSR} = \frac{f_s}{2f_b} \leftrightarrow \frac{2f_b}{f_s} = \frac{1}{\text{OSR}}$, the quantization noise power results to be:

$$P_{\text{noise}} \cong \frac{\Delta^2\pi^2}{36}\left(\frac{1}{\text{OSR}}\right)^3$$

Assuming that the maximum signal power is the same as seen in Chap. 1, such as:

$$P_{\text{signal}} \frac{\Delta^2 2^{2N}}{8}$$

demonstrate that $\text{SNR}_{\text{max}} = 6.02N + 1.76 + 30\log_{10}(\text{OSR}) - 10\log_{10}\left(\frac{\pi^2}{3}\right)$. How much is the increase of SNR for doubling of OSR in a first-order modulator?

Q.12
What are the factors that limit the resolution of a sigma-delta ADC?

Q.13
Find the STF and NTF of the modulator in Fig. 2.21. What is its order?

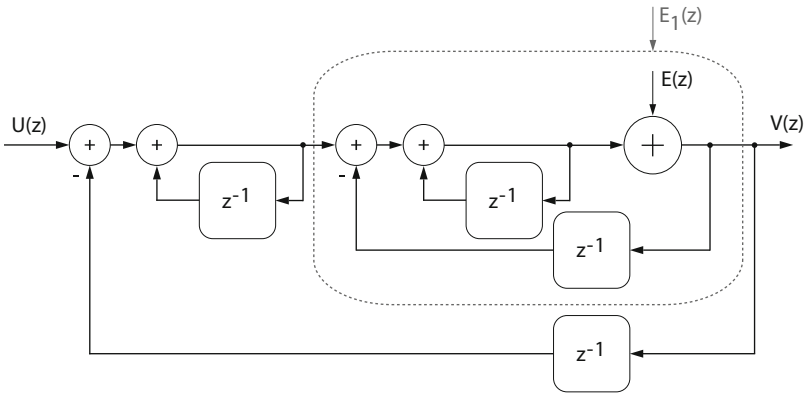


Fig. 2.21 Block schematic of a ΣΔ-modulator with a 1-bit quantizer

References

1. Schreier R, Temes GC. Understanding delta-sigma data converters. Hoboken: Wiley; 2005.
2. Oppenheim AV, Schafer RW. Discrete-time signal processing. Upper Saddle River: Prentice-Hall Inc.; 1989.
3. MathWorks. Sigma-delta A/D conversion. [Online] https://it.mathworks.com/help/dsp/examples/sigma-delta-a-d-conversion.html?requestedDomain=www.mathworks.com. 2017.
4. Candy JC, Benjamin OJ. The structure of quantization noise from sigma-delta modulation. IEEE Trans Commun. 1981;29(9):1316–23.
5. Friedman V. The structure of the limit cycles in sigma delta modulation. IEEE Trans Commun. 1988;36(8):972–9.
6. Candy JC, Temes GC. Oversampling delta-sigma data converters: theory, design and simulation. New York: Wiley-IEEE Press; 1991.
7. Pohlmann KC. Principles of digital audio. London: McGraw-Hill; 2008.
8. Norsworthy SR, Schreier R, Temes GC. Delta-sigma data converters: theory, design and simulation. New York: Wiley-IEEE Press; 1996.
9. Widrow B, Kollar I. Quantization noise: roundoff error in digital computation, signal processing, control and communications. Cambridge, UK: Cambridge University Press; 2008.
10. Watkinson J. The art of digital audio. Oxford, UK: Focal Press; 2013.
11. Medeiro F, Verdù BP, Vazquez AR. Top down design of high performance sigma delta modulators. Boston: Kluwer; 1999.



The Second-Order Sigma-Delta Modulator 3

This chapter focuses on specific system-level simulations of the second-order Sigma-Delta modulator, to be referred as MOD2, formed by a second-order loop filter and a single-bit quantizer. Similarly to Chap. 2, the aim is to gain a significant appreciation of the practical aspects needed during the design process. Therefore, the operation for DC and sinewave inputs as well as the non-idealities affecting the modulator behavior will be studied by performing the practical exercises suggested in the MATLAB® and Simulink® environment. Since the chapter follows almost exactly the sequence of simulations conducted for MOD1, the reader is encouraged to make frequent comparisons between the performance of MOD1 and MOD2 in order to gain a better understanding of the concepts covered. Moreover, in Sect. 3.7 a comprehensive investigation on the theory underlying alternative architectures is presented, as well as suggestions on the simulations to conduct on those modulator structures.

3.1 Simulink® Model of MOD2

Opening the Toolbox file *"mod2.mdl,"* the schematic illustrated in Fig. 3.1 should appear. The elements forming the model include the following:

- *Two Non-delaying Integrators* [1] – Each composed of a (discrete) $1/z$ unit-delay element in feedback to an adder. To represent any finite integrator gain at DC, caused by the finite gain of an amplifier used to implement a real analog integrator, a gain block preceding the delay element has been added. The saturation block following the adder is used to represent the clipping effect of, for example, the finite power supply voltages or digital word lengths in real modulators. The rate limiter block is used to simulate the effect of limited slew rate. By default, all blocks are set to $\pm$inf (i.e., infinity), giving an ideal integrator.

© Springer International Publishing AG, part of Springer Nature 2019

I. Arnaldi, *Design of Sigma-Delta Converters in MATLAB®/Simulink®*,
https://doi.org/10.1007/978-3-319-91539-5_3

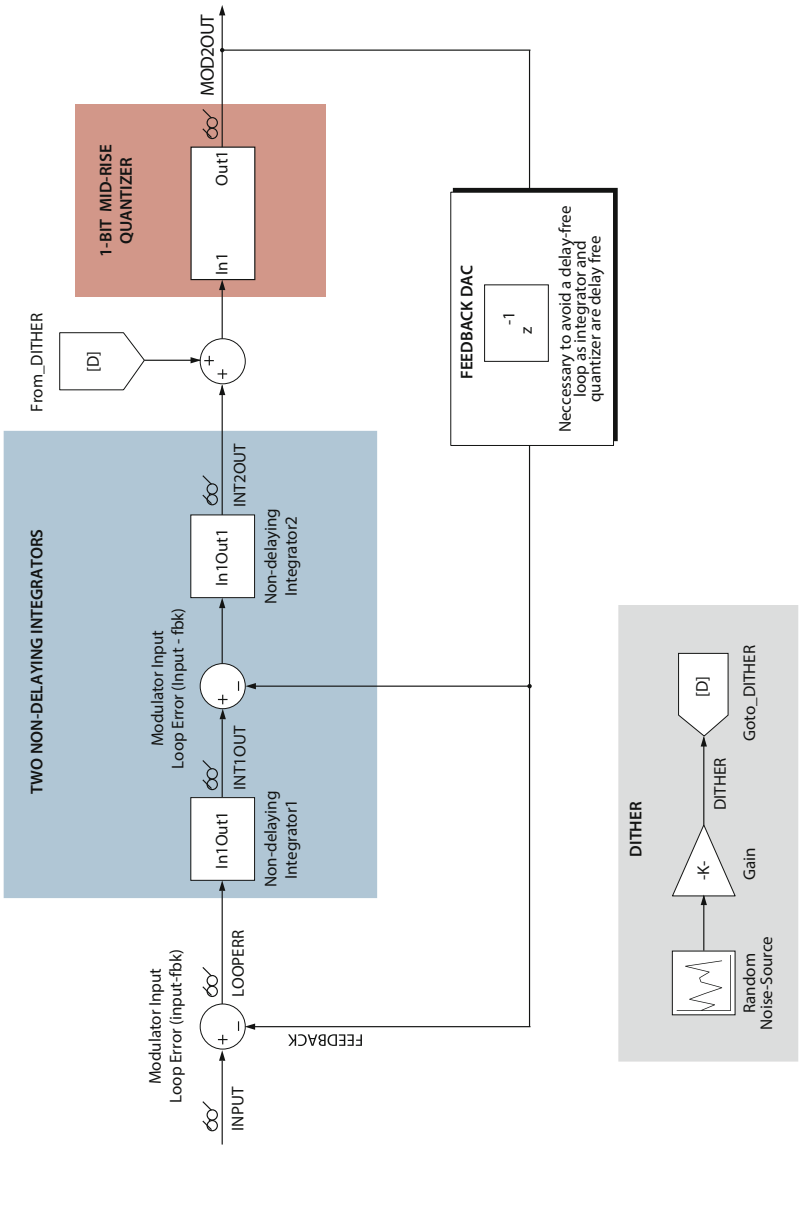


Fig. 3.1 Simulink® Model of MOD2

- *Mid-rise Quantizer* [1] – Single-bit, ±1 output states and a step size of two. Since the default MATLAB quantizer is *mid-tread* [2], the signal has been shifted up by +1 before and down by −1 after the quantizer element in order to implement the mid-rise behavior.
- *Two Delaying Feedback Loop* – From the quantizer output to the input as well as to the output of the first integrator, where the signals are subtracted and the error signals are applied to the first and second integrator, respectively. This is equivalent to replacing the quantizer in MOD1 with another MOD1. The delay in the feedback represents the time to quantize, sample, and feed-back the integrators output – usually a single clock cycle.
- *Dither Source* – Generating white noise between ±1. This is scaled by a gain block and summed at the input of the quantizer. By default, the dither is off.

In addition to these main functional elements there are a number of input and output
blocks:

- A *DC* and *signal generator* inputs followed by an ideal sampler.
- A *time scope*, used for visualizing the various internal signals in the modulator.
- An *analog filter*, implemented to aid visualization of how the modulator digital output encodes the analog input signal. Note that a real ΣΔ-ADC would use a decimation filter (e.g., see [3]).

The default sampling frequency is chosen as $f_s = 3.072\,MHz$ with an oversampling ratio of OSR = 64. The signal band of interest, $f_b = f_s/(2OSR)$, corresponds to that of the audio band (i.e., from 0 Hz to 24 KHz). Further, note that since MOD1 uses a 1-bit quantizer, its DAC [1] is capable of perfectly linear operation (i.e., more on this in Chap. 5), although in practice this may not always be true (i.e., memory effects in the DAC, variations in voltage references, etc.). In the model, an ideal 1-bit DAC is assumed, and due to its inherent linearity and Simulink® operation process, no specific DAC block is required.

3.2 Operation for DC Inputs

In this paragraph, the second-order ΣΔ-modulator behavior to DC inputs is examined.

In the Toolbox open the model *"mod2.mdl"* found in the folder *3_MOD2*.
Type – *load_par* – in the MATLAB Command Window to initialize the variables required by the model.

(continued)

In the model, set the DC input level to zero by typing – *dc = 0* – into the MATLAB Command Window and run the simulation.

Once the simulation is finished, open the time scope by double-clicking on any of the small blue glasses found in the signal connections to observe the waveform at the nodes of the modulator. The reader may wish to use the horizontal magnifying glass to zoom into a few cycles of the LOOPERR, INT1OUT, INT2OUT, and MOD2OUT waveforms.

Type – *mod_SNDR* – in the MATLAB Command Window to obtain the output spectrum.

Save the result and repeat for the following DC input values: 0.5, 0.75, 0.875, and 0.9375.

As an example, Fig. 3.2 reports the time domain and FFT results for a DC input signal of 0.5. Similar results for all the DC input values proposed should be obtained.

From the results, it should be apparent that:

- The value of the filtered bit sequence of the modulator output corresponds to that of the DC input signal, as it can be seen from the time domain results. Therefore, the conversion can be considered accurate.
- The modulator output is periodic.
- Due to the periodic behavior, noise shaping is not apparent and strong tones are present in the spectrum.

3.2.1 Idle Tone Generation

Regarding the generation of idle tones, it is important to mention that limit cycles in MOD2 are more complicated than in MOD1 [3–5].

Consequently, MOD2 is not to be analyzed in detail with a view to predicting limit cycle frequencies for given DC inputs, as previously done for MOD1. However, although not entirely true for MOD1, a significant conclusion presented in the previous chapter was that if the DC input u ($|u| < 1$) is constant but irrational, the output cannot be periodic. This property should be investigated for MOD2 by typing, for example, *dc = pi/10* in the MATLAB Command Window and running the simulation. Comparing the results between MOD1 and MOD2 (Fig. 3.3), it can be seen that such property is valid since fewer tones are present in the MOD2 spectrum. Further, it is apparent that MOD2 behaves much more closely to an ideal modulator, having a clear noise shaping characteristic of 40 dB/dec which is typical of a second-order filter.

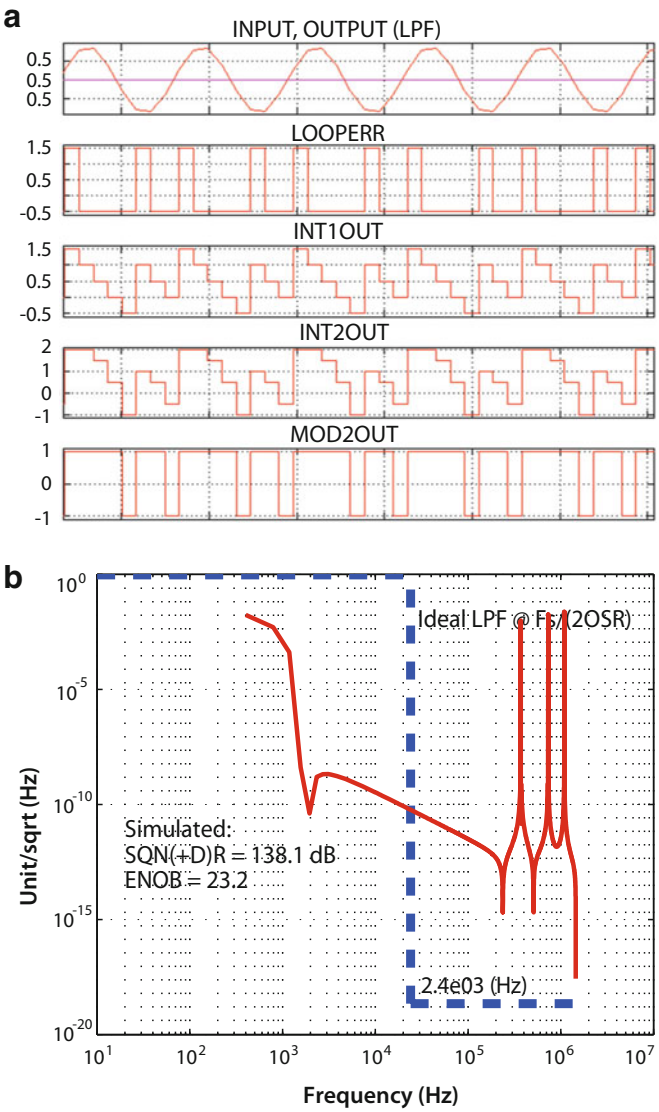


Fig. 3.2 (Top) time domain and (bottom) FFT results for DC input signal 0.5

3.2.2 Tonal Behavior and DC Input Sweep

The previous exercise briefly highlighted that the basic MOD2, like MOD1, still gives strong output tones instead of a shaped noise spectrum in the case of rational DC inputs. To better comprehend the tonal behavior of MOD2, it is useful to investigate the effect of limit cycles on the signal-band (i.e., from DC to $f_s/(2 * \mathrm{OSR})$) across a swept range of input DC levels which creates long cyclic patterns.

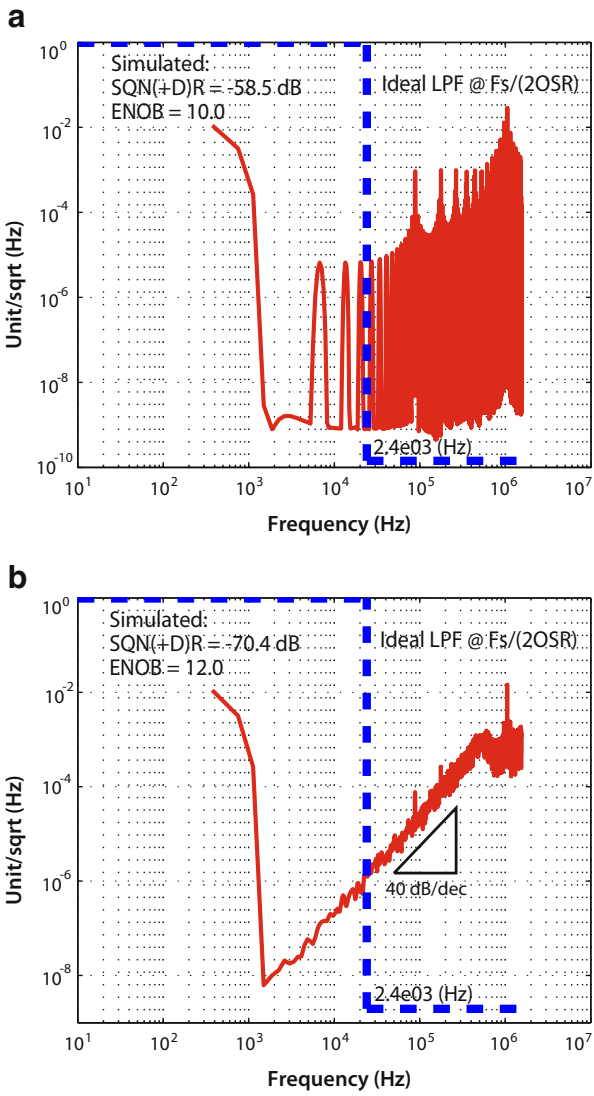


Fig. 3.3 (Top) MOD1 and (bottom) MOD2 FFT for irrational DC input signal *pi*/10

Close *"mod2.mdl."*

Type – *target_mod* = *'sweep_mod2'* – into the MATLAB Command Window.

Open "*sweep_testbench.mdl*," right click on the *Modulator* block and enter in the *Model Reference Parameters*. Change the *Model Name* to – *sweep_mod2* – and press *ok*. Save and close the model.

(continued)

Type – *sweep_dc_bbpwr* – in the MATLAB Command Window and run the routine → *Warning: the simulation analyzes the signal-band noise power for 1000 DC levels, hence may take a long time to complete!*

A plot of total baseband noise power in dBW is to be generated for input amplitudes from 0 to 1, the maximum allowed input (Fig. 3.4).

Comparing the results with those of MOD1 (Fig. 2.5), it is apparent that much of the spikes associated with a periodic behavior are now absent. Therefore, MOD2 results more resistant to tones than MOD1 [3, 4]. However, MOD2 still exhibits vestiges of the periodic behavior as it can be seen, for example, around the rational DC value of 0. Therefore, the interested reader may find instructive to investigate the effect of dither on the modulator. As seen for MOD1, dither should help in reducing the tonal behavior of the modulator although the baseband noise would result slightly increased.

A second feature apparent in the plot of Fig. 3.4 is the gradual increase of the in-band noise power, which culminates in a large noise peak as the DC input tends to $1 - |u| \rightarrow 1$. This behavior is caused by the signal-dependent quantizer gain, and hence signal-dependent NTF, of MOD2 [1, 3, 6]. As the input increases in magnitude, the quantizer gain decreases, reducing the loop gain and hence the effectiveness of the noise shaping. It results that the noise at the output of MOD2 increases with the increase of the input signal level. In the terminology of stochastic processes [3, 7], this property makes the quantization noise of MOD2 with a large deterministic input *nonstationary*. Specifically, when the absolute value of the signal is large, the quantization noise power is also large, and thus the statistics of the quantization noise are time-varying. The topic of effective quantizer gain is not to be treated mathematically nor in greater detail in this book in order to keep a practical approach

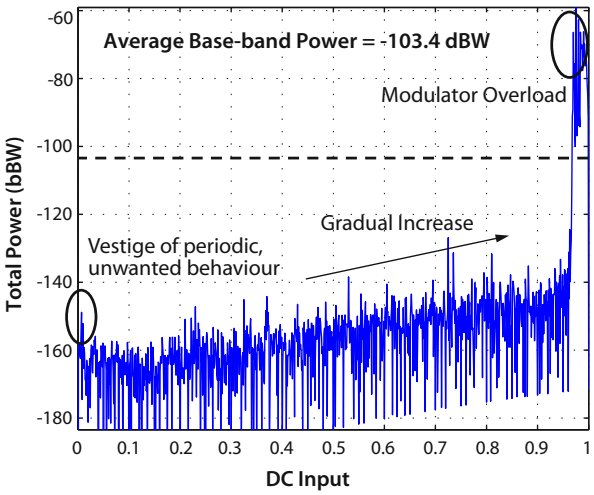


Fig. 3.4 Baseband noise power vs. DC input and average power overall inputs for MOD2 with OSR = 64

to the subject. However, due to its relevance, the more interested reader is advised to refer to [1, 3, 8] for further studies. The important conclusion to be understood is that, due to signal-dependent quantizer gain, the mathematical predictions of the linear model would not match the performance of a real modulator, especially for inputs close to full scale. This further highlights the importance of simulations in Sigma-Delta modulators.

3.3 Stability of MOD2

Contrary to MOD1, which is always stable as long as $|u| < 1$, MOD2 stability is guaranteed only for arbitrary inputs less than 0.1 in magnitude. Note that the upper limit on the input amplitude for which stable operation is guaranteed is not ensured by mathematical predictions [3]. In order to understand the stability properties of MOD2, it is useful to investigate the behavior of the output of the integrators.

In the Toolbox, open the model *"mod2.mdl"* found in the folder *3_MOD2*.

Type – *load_par* – in the MATLAB Command Window to initialize the variables required by the model.

In the model, set the DC input level to zero by typing – $dc = 0$ – into the MATLAB Command Window and *run* the simulation.

Once the simulation is finished, type – *integ_stats(INT1OUT); integ_stats (INT2OUT);* – in the MATLAB Command Window to obtain histograms and statistical data of the integrators' output.

Record the maximum integrators output values (i.e., MAX) and repeat for the following DC inputs: 0.5, 0.75, 0.875, and 0.9375.

Figure 3.5 illustrates the maximum output values of the integrators relative to the modulator input, and, as it can be seen, the amplitude of the output of the second integrator grows rapidly as the modulator input approaches full scale (e.g., $|u| \rightarrow 1$). However, according to the mathematical analysis reported in [3], the second-order modulator is guaranteed to be stable in the case of DC inputs less than 1 in magnitude. Hence, since MOD2 tracks the low-frequency content of its input, it seems fair to assume that arbitrary time-varying inputs satisfying $|u(n)| < 1$ for all n would ensure stability. Unfortunately, this is not always the case [3, 6, 7], although the conditions for instability to occur, especially for relatively small inputs, are unlikely to be encountered in practice. Another problem associated with the rapid growth of the output state in the second integrator is that, in real modulators, this may lead the integrator's amplifiers to clip, hence introducing other sources of distortion. Therefore, it results wise to limit the input of MOD2 to $|u| < 0.8$ *or* 0.9 so that the state of the second integrator is not unduly large. However, even though such an input limit will keep the modulator state reasonable for DC and slowly varying

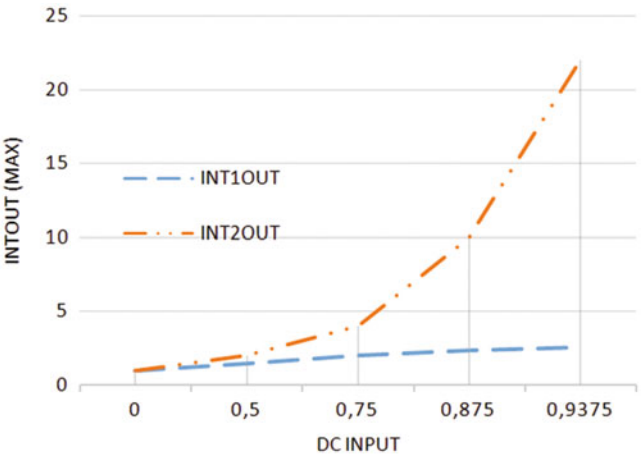


Fig. 3.5 Maximum integrators output vs. DC input

inputs, it is still possible for the modulator state to become much larger than intended. Therefore, it is important to include means for detecting overly large states and for placing the modulator in a "good" state. This would ensure stability since second-order modulators recover from a state where the integrators are saturated as soon as the input is restored to a lower amplitude.

3.4 The Effects of Finite Integrator Gain

This paragraph investigates the effects of finite integrator gain on the time domain waveforms and DC transfer function of MOD2.

In the Toolbox open the model *"mod2.mdl"* found in the folder *3_MOD2* and type – *load_par* – in the MATLAB Command Window for initialization of model parameters.

Introduce a finite DC gain in the modulator by typing – *mod.igain1 = 10; mod.igain2 = 10;* – into the MATLAB Command Window.

Once the simulation is finished, open the time scope by double-clicking on any the small blue glasses found in the signal connections.

Type – *mod_SNDR* – into the MATLAB Command Window to obtain the output spectrum and save the results.

Try few input DC values, such as 0.5, 1/10, 0.011, etc.

As for MOD1, the reader should note that MOD2 loses its ability to achieve infinite precision with DC signals due to the limited gain of the integrators. Further investigations into the dead zone behavior of MOD2, through a DC sweep across small DC levels, allow a better appreciation of the finite integrator gain effects.

Close the model *"mod2.mdl."*

Type – *target_mod = 'sweep_mod2'* – into the MATLAB Command Window.

Open "*sweep_testbench.mdl*" found in the main Toolbox folder, right click on the *Modulator* block and enter in the *Model Reference Parameters*. Change the *Model Name* to – *sweep_mod2* – and press *ok*. Save and close the model.

Type – *sweep_dc_dc* – in the MATLAB Command Window and run the routine. A DC sweep for a subset of the input range (i.e., ±0.1) is to be performed.

Figure 3.6 illustrates the results, where the bright-gray line represents the ideal DC transfer function and the dark-gray line the one resulting in the case of finite integrator gain.

Naming the finite integrator gain value as A, and zooming on the dead zone around the DC input of 0, it can be verified that inputs smaller than $1.5/A^2$ in normalized value have no effect on the modulator output. The derivation of such simple formula can be found in [3]. Therefore, due to the squared gain obtained by cascading two integrators, the dead zones result much smaller than the one found for MOD1. Further, note that the gain of $A = 10$ used in this example is unlikely to be encountered in practice since it results very small. Hence, it is easily appreciable that MOD2 is much more tolerant to finite op-amp gain and, consequently, less susceptible to idle tones. Moreover, using dither would aid the recovery of dead zones by

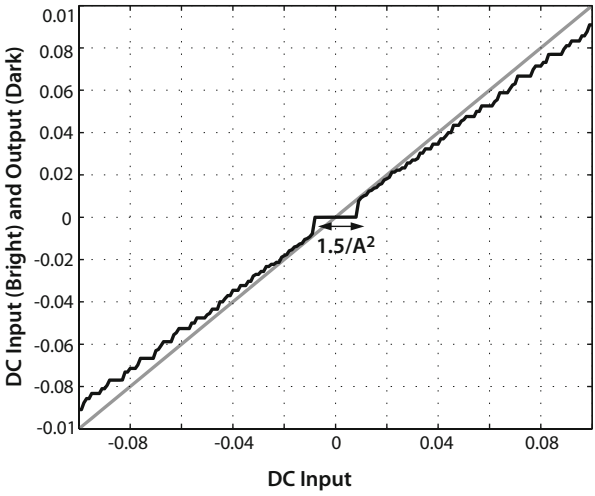


Fig. 3.6 MOD2 DC transfer function around DC input = 0

preventing the modulator from locking into limit cycles. It would result that the modulator will behave much more like the ideal case of infinite integrator gain. Therefore, for modulators of order higher than the first, the issue of finite integrator gain should not result too problematic to achieve a satisfactory modulator performance.

Example

A second-order modulator with feedback levels ± 1 V has an amplifier gain of 140. What is the expected width of the dead zone around DC?

Solution
Dead zone $= 1.5/A^2 = 1.5/140^2 = 76.5e{-}6$ V

3.5 Operation for Sinewave Inputs

The next task proposes to simulate the behavior of MOD2 in response to a sinewave input while comparing the results with the behavior of MOD1.

Set the initial conditions by typing – *load_par* – into the MATLAB Command Window.

Open the model "*mod2.mdl*" found in the folder *3_MOD2*.

In the model double-click on the *manual switch* to select the *signal generator* input source. *Run* the simulations.

Set the sinewave frequency to 6 KHz by typing – *sinfreq = 6e3* – into the MATLAB Command Window.

Once the simulation is finished, open the time scope by double-clicking on any the small blue glasses found in the signal connections to observe the waveforms at the nodes of the modulator. The reader may wish to use the horizontal magnifying glass to zoom into a few cycles of the LOOPERR, INT1OUT, INT2OUT, and MOD2OUT waveforms.

Type – *mod_SNDR* – into the MATLAB Command Window to obtain the FFT graph.

Examining the time domain waveforms, it should be evident that the input sinewave is captured in the modulator output bit sequence, where the only difference with MOD1 is the

increased tendency for the output to be +1 when the input is positive and −1 when the input is negative. The filtered output sequence of ±1 s also confirms the ability of the modulator to successfully convert varying signals. Analyzing the output spectrum reported in Fig. 3.7 clearly confirms the second-order nature of the noise shaping exhibited of 40 dB per decade.

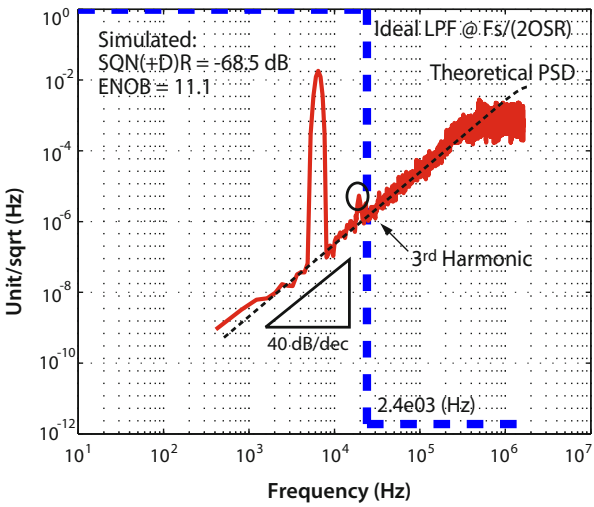


Fig. 3.7 FFT results for MOD2 with sinewave input

Important to note is the presence of the third harmonic tone, which falls into the passband. Further, depending on the input, note that other harmonics may result clearly visible in the spectrum. Since white quantization noise cannot produce harmonics of the signal, this is another proof that the white noise model assumption does not accurately represent Sigma-Delta modulators behavior. To account for the harmonics, a relatively complex mathematical analysis of the signal-dependent quantizer gain as reported in [3] is needed. In order to keep a rather descriptive approach to the discussion, mathematical formulas are not to be reported here; however, the important conclusion of the analysis in [3] is that the coefficients derived from the quantizer transfer curve (QTC) can be used to estimate the distortion induced by the nonlinear QTC itself. Then, it is shown that since the distortion term is added at the same place in the loop as the quantization error, the spectrum of the distortion is to be shaped by the modified – due to the signal-dependent quantizer gain – NTF. Thus, the distortion would result greatest for frequencies where the NTF gives the least protection – when the distortion term lies at the edge of the passband. For a small low-frequency sinewave input of amplitude A, the local average of the output follows the input and thus, according to the linear model, the local average of the quantizer input is also a low-frequency sinewave, but of amplitude A divided by a factor dependent on the QTC. Thus, the amplitude of the third harmonic induced by the QTC would be less affected by the NTF, and, as a consequence, its suppression would be less significant, explaining why harmonics appear in a spectrum of a second-order modulator. Further, since the greater the signal the smaller the quantizer gain, the NTF would result less efficient when the input signal is large and the distortion is severe. To better understand this concept, the interested reader may wish to repeat the simulation with a larger input signal amplitude – e.g., $sinamp = 0.97$ – and verify how quickly the output signal of the second integrator escalates to very high amplitude values. Since the integrators used are ideal (i.e., no distortion induced by clipping effect, etc.), it should be noted

from the FFT how the SQNR diminishes while the distortion increases when compared to the previous example where the input signal amplitude was −3 dBFS. This is only due to the signal-dependent quantizer gain and nonlinear QTC.

Another consequence of the signal-dependent quantizer gain is the discrepancy seen between theoretical and actual PSD shape (Fig. 3.7). The theoretical curve is similar to the results obtained but has a corner frequency that is higher than that of the observed PSD and as such predicts a greater suppression of the in-band quantization noise. As described in [3, 8], it is possible to account for the shift in NTF shape by evaluating the effective quantizer gain for a determined input amplitude from simulation, in order to obtain a more accurate estimate of the actual performance of the NTF of MOD2. However, since such performance is dependent on the input used, it is common in a design process to avoid such precise calculations and instead perform extensive simulations in order to ensure satisfactory performance over the entire range of the input. Therefore, the important conclusion from this analysis is that the predictions of the linear model on the achievable SQNR would be too optimistic, and, in reality, the modulator performs worse than it is expected. Therefore, it results wise in any design to include some tolerance from the minimum required specifications in order to account for this discrepancy as well as to other non-ideal sources which affect the theoretical performance of the modulator (e.g., thermal noise, analog matching, etc.).

3.5.1 Integrator Saturation

In a real MOD2, the large amplitude of the output signal of the integrators may lead to amplifier saturation, due to the limited headroom set by the power supplies. Therefore, it is useful to simulate the effect of clipping in order to understand how it affects the modulator performance.

Reset initial conditions by typing – *load_par* – into the MATLAB Command Window.

Open the model "*mod2.mdl*" found in the folder *3_MOD2*.

Set the sinewave frequency to 6 KHz by typing – *sinfreq = 6e3* – into the MATLAB Command Window.

Set the saturation limit of the integrators to 2 by typing – *mod.isat1 = 2; mod. isat2 = 2;* – into the MATLAB Command Window and run the simulation.

Once the simulation is finished, open the time scope by double-clicking on any of the small blue glasses found in the signal connections to observe the waveforms at the nodes of the modulator. The reader may wish to use the horizontal magnifying glass to zoom into a few cycles of the LOOPERR, INT1OUT, INT2OUT, and MOD2OUT waveforms.

Type – *mod_SNDR* – into the MATLAB Command Window to obtain the FFT graph.

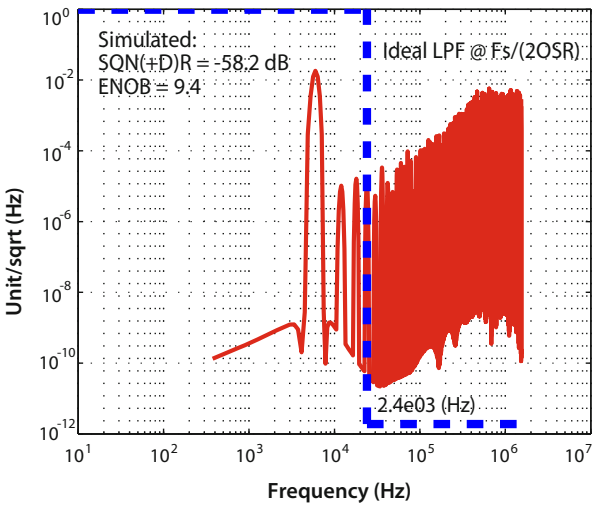


Fig. 3.8 FFT results for MOD2 with saturated integrators

Analyzing the time domain results, it should be apparent that the output of the integrators is now limited to ± 2 in amplitude. Figure 3.8 illustrates the FFT of the simulation and, as it can be seen, the large amplitude of the output of (especially) the second integrator constitutes a real problem in the case of amplifier saturation.

If clipping occurs, large distortion is introduced, as proven by the results obtained. Therefore, due to the very low tolerance of MOD2 to the even slightest integrator overload, it is apparent that such issue should be carefully considered during the design, either by reducing the input range of the modulator or by scaling the internal signals of the modulator appropriately.

3.5.2 Sinewave Amplitude Sweep

This paragraph investigates the actual achievable signal-to-quantization-noise ratio (SQNR) of MOD2 in response to a range of sinewave amplitudes while providing further insights into the non-ideal behaviors discussed so far.

Reset initial conditions by typing – *load_par* – into the MATLAB Command Window.

Set the sinewave frequency to 6 KHz by typing – *sinfreq = 6e3* – into the MATLAB Command Window.

Type – *target_mod = 'sweep_mod2'* – in the MATLAB Command Window.

Type – *sweep_sinamp* – into the MATLAB Command Window and run the simulation. Note it may take some time to complete.

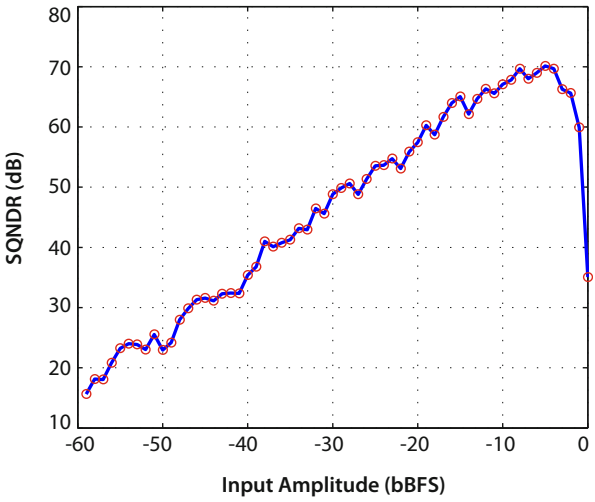


Fig. 3.9 SQNR vs. swept amplitude of a 6 KHz sine input for MOD2

Once the simulation is completed, a plot of the SQNR versus input amplitude in decibels (dBFS), where 0 dBFS corresponds to the amplitude of 1, will appear. The graph should resemble the one reported in Fig. 3.9.

To verify the peak value of SQNR achieved, it is useful to compare its value with the theoretical SQNR from the formula [3]:

$$\text{SQNR} = 6.02N + 1.76 + (20L + 10)\log_{10}(\text{OSR}) - 10\log_{10}\left(\frac{\pi^{2L}}{2L+1}\right) \cong 85\ (\text{dB}) \tag{3.1}$$

where the modulator order $L = 2$, the oversampling ratio OSR $= 64$, and the number of bits corresponds to $N = 1$.

As it can be seen, the peak SQNR found from the simulation is not consistent with the predicted theoretical value, confirming the increase in noise power as the input signal amplitude increases caused by the signal-dependent quantizer gain. However, besides the poorer performance than the one predicted by theory, it can be seen that when compared to the results of MOD1, the SQNDR graph is fairly smooth over the signal range due to the higher tolerance of MOD2 to tones and dead zone as a consequence of the higher loop gain. Therefore MOD2 behaves much more like an ideal modulator. The interested reader may wish to try to add some dither – e.g, *mod. dither = 0.2* – in order to verify that dither does smooth the SQNDR slope since it prevents the generation of tones and dead zone.

Example

For the modulator simulated in this paragraph, calculate the ENOB predicted by theory and the ENOB achieved using the simulation results of Fig. 3.9. How much is the difference in terms of resolution bits?

Solution

$$\text{ENOB (Bit)} = \frac{\text{SNR (dB)} - 1.76}{6.02}$$

The ENOB predicted using the peak SNR of 85 dB obtained from theoretical calculations is approximately 14 bits. The ENOB predicted using the peak SNR of 70 dB obtained from simulation is approximately 11.5 bits. Almost 2.5 bits of difference in resolution.

It is also useful to repeat the simulation in the case of saturated integrators, in order to have a further insight on the degradation of the effectiveness of the NTF.

Set the saturation limit of the integrators to 2 by typing – *mod.isat1 = 2*; *mod.isat2 = 2;* – into the MATLAB Command Window and run the simulation.

Type – *sweep_sinamp* – into the MATLAB Command Window and run the simulation.

Comparing Fig. 3.10 with Fig. 3.9, it can be seen that even for small inputs, a decrease in SQNR of approximately 10 dB occurs in the case of saturation. Further, it is apparent that distortion is introduced, as it can be noticed from the less linear slope obtained. Therefore, it results clear that signal scaling is important in the design of a second-order modulator [3, 6, 7], and in the next paragraphs, techniques to accomplish this are presented.

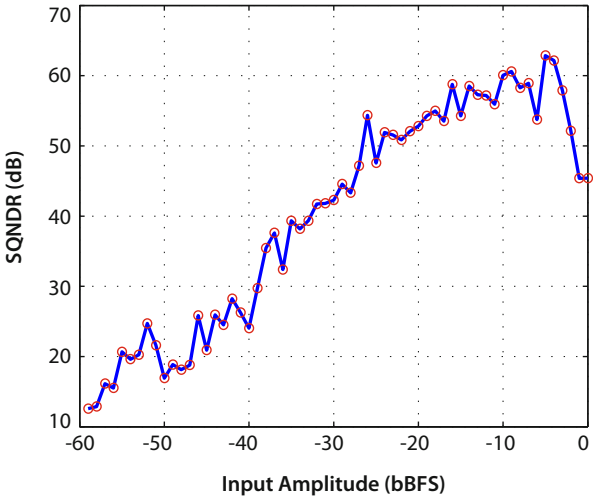


Fig. 3.10 SQNR vs. swept amplitude of a 6 KHz sine input for MOD2 with saturated integrators at ±2

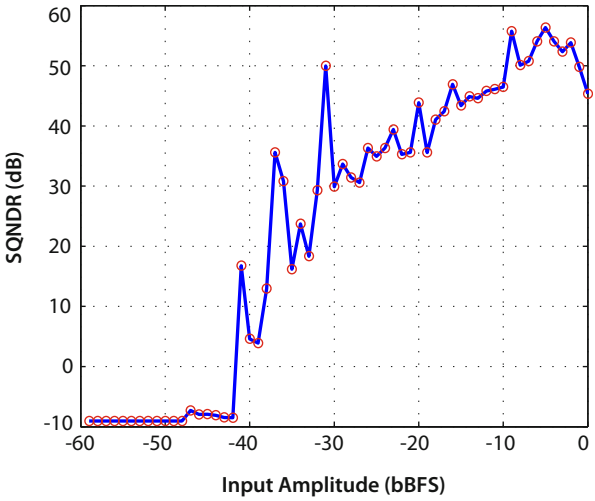


Fig. 3.11 SQNR vs. swept amplitude of a 6 KHz sine input for MOD2 with limited integrator gain of 10

The last suggested task involves running the sweep in the conditions of finite integrator gain.

Reset the saturation conditions to the default state by typing – *mod.isat1 = inf; mod. isat2 = inf;* – into the MATLAB Command Window.

Type – *mod.igain1 = 10; mod.igain2 = 10;* – into the MATLAB Command Window.

Type – *sweep_sinamp* – into the MATLAB Command Window and run the simulation.

According to the results presented in Fig. 3.11, it can be seen that the presence of dead zone causes loss of SQNDR, especially at small signal amplitudes. Applying dither would, to some extent, help to improve the modulator performance; however, for very small inputs (e.g., $\leq$40 dBFS), such degraded SQNR would still occur due to the strong dead zone around DC. Besides the expected performance degradation, it should be noted that this is not too severe compared to the results of Fig. 3.9 and the MOD1 case. This should further highlight that MOD2 is more tolerant to finite integrator gain, and, considering that a gain of 10 is unlikely to be encountered in practice, it should not be too problematic to achieve a good performance when translating the high-level design into a circuital solution.

Example

According to the simulations conducted so far, underline the building block with the most stringent performance requirements in a MOD2:

First Integrator – Second Integrator – Comparator

Which simulation could support your statement?

Solution
First Integrator. A simple method to investigate that the first integrator has the most stringent performance requirements is to repeat the simulation with finite integrator gain so that – mod.igain1 = 10; mod.igain2 = inf; – and – mod. igain1 = inf; mod.igain2 = 10. It is seen that the worst modulator performance is obtained when it is the first integrator that has a limited integrator gain. On the other hand, when the first integrator is ideal and the second one has a finite gain, it is seen that the modulator performance is better than the case of first integrator finite gain.

3.6 Theoretical Analysis

In this paragraph, the mathematical analysis as found in [3] of the discrete MOD2 is reported in order to gain a broader understanding of its properties. Analyzing the model in the z-domain (Fig. 3.12) gives:

$$\begin{aligned} V(z) &= W(z) + \left(1 - z^{-1}\right)E(z) \\ \left(1 - z^{-1}\right)W(z) &= U(z) - z^{-1}V(z) \\ V(z) &= U(z) + \left(1 - z^{-1}\right)^2 E(z) \end{aligned} \tag{3.2}$$

where:

$$\mathrm{STF}(z) = 1 \tag{3.3}$$

$$\mathrm{NTF}(z) = \left(1 - z^{-1}\right)^2$$

Therefore, the output corresponds to the input while the error is the squared NTF of MOD1. Continuing the analysis to investigate the noise shaping properties of the modulator, it results:

$$\left|\mathrm{NTF}\left(e^{j2\pi f}\right)\right|^2 = 2(\sin \pi f)^4 \cong (2\pi f)^4, \text{for } f \ll 1 \tag{3.4}$$

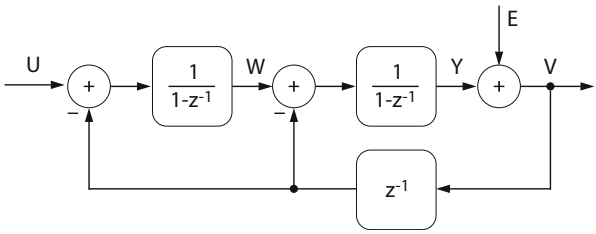


Fig. 3.12 Simplified MOD2 linearized block diagram

Plotting the $|\text{NTF}|^2$ against the frequency would show that the noise is suppressed at low frequencies and moved to f_s. The noise left in the baseband, N_q^2, can be estimated from:

$$N_q^2 = \frac{1}{\pi}\int_0^{\frac{\pi}{\text{OSR}}} \sigma_q^2 |\text{NTF}|^2 d\omega \tag{3.5}$$

$$N_q^2 = \frac{\sigma_q^2}{\pi}\int_0^{\frac{\pi}{\text{OSR}}} \left(2\sin\frac{\omega}{2}\right)^4 d\omega$$

$$N_q^2 \cong \frac{\sigma_q^2}{\pi}\int_0^{\frac{\pi}{\text{OSR}}} \omega^4 d\omega \text{ assuming OSR} \gg 1$$

$$N_q^2 = \frac{\sigma_q^2 \pi^4}{5\text{OSR}^5}$$

where σ_q^2 is the power spectral density of the quantization noise.

Comparing Eq. 3.5 with Eq. 2.16 of MOD1 shows that more noise is to be suppressed in the baseband, due to the square nature of the NTF. The fundamental conclusion of Eq. 3.5 is that doubling the OSR reduces the noise power by a factor of 32, which corresponds to 15 dB or 2.5 bit.

Assuming a mid-rise quantizer with an LSB step of 2 (as in the simulations), the quantization noise power is $\sigma_q^2 = 1/3$. Since the *SQNR = 10*log*(Signal Power/ Quantization Noise Power)* in dB, and assuming a sinewave input of amplitude A (e.g., rms = 0.707 A and power 0.5 A^2), an alternative SQNR formula to that of Eq. 3.1 is found:

$$\text{SQNR} = 10\log_{10}\left(\frac{15A^2\text{OSR}^5}{2\pi^4}\right) \tag{3.6}$$

Equation 3.6 clearly exhibits the ability of MOD2 to achieve higher SQNR for the same OSR when compared to MOD1. To give some perspective, by considering audio applications where the maximum bandwidth is 20 KHz, let the OSR be 128 and the maximum input signal amplitude be 1. Then Eq. 3.6 gives SQNR = 94.2 dB, which corresponds to almost 16-bit resolution, and a sampling frequency of f_s = 2*OSR = 5.12 MHz, which is a highly practical value. To achieve the same resolution with MOD1, theoretical calculations predict that an OSR of 1800 would be needed, resulting in a f_s of 72 MHz. This would make the implementation of the modulator unnecessarily difficult.

Example

Calculate the baseband noise power of a second-order modulator with OSR = 32 and feedback levels ± 1 V.

Solution
The noise power is:

$$10\log\left(\frac{\pi^4}{15\mathrm{OSR}^5}\right) = -67.13\ \mathrm{dBW}$$

3.7 Alternative Architectures

In order to achieve a more optimal noise shaping and to enhance the stability and dynamic range of MOD2, it is often desirable to include arbitrary feedforward and/or feedback coefficients which allow controlling the placement of the poles and zeros in the NTF (Fig. 3.13).

The signal and noise transfer function of this general structure can be described as [3, 7, 8]:

$$\mathrm{STF}(z) = \frac{B(z)}{A(z)} \tag{3.7}$$

$$\mathrm{NTF}(z) = \frac{(1-z^{-1})^2}{A(z)}$$

Where $A(z)$ and $B(z)$ are second-order polynomials:

$$B(z) = b_1 + b_2(1-z^{-1}) + b_3(1-z^{-1})^2 \tag{3.8}$$

$$A(z) = 1 + (a_1 + a_2 + a_3 - 2)z^{-1} + (1 - a_2 - 2a_3)z^{-2} + a_3 z^{-3}$$

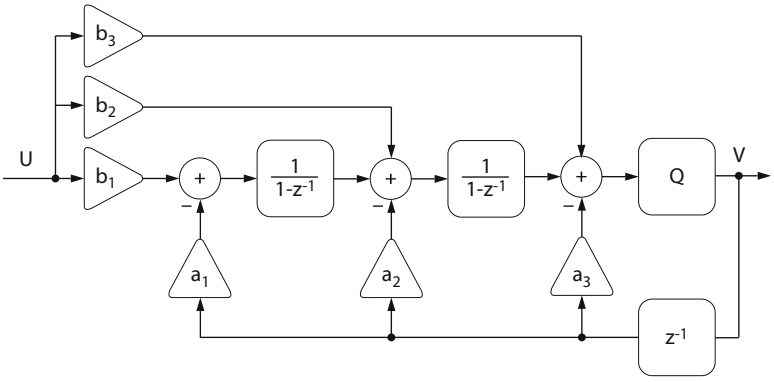


Fig. 3.13 MOD2 block diagram with arbitrary feed-in and feedback coefficients

Important to note is that $A(z)$ has to be chosen appropriately in order to ensure the stability of the system. Further, since the a_3 feedback term increases the NTF order to 3 but does not increase the number of in-band NTF zeros, the term is rarely used, although it may be useful if the modulator is continuous time, rather than discrete time as assumed here.

In the next paragraphs, two of the most common second-order architectures implemented to develop Analog-to-Digital Converters which use feedforward and feedback coefficients are presented.

3.7.1 The Boser-Wooley Modulator

The Boser-Wooley modulator [3, 9] is a second-order modulator where only delaying integrators are used, as shown in Fig. 3.14. Therefore, no quantizer delay is needed in the feedback path to make the architecture stable. Further, since converting the delay-free integrators of the architecture studied in previous paragraphs into delaying integrators and removing the feedback delay alters the NTF and yields a marginally stable system, coefficients need to be implemented in order to control the placement of poles and zeros of the NTF (i.e., $A(z)$ polynomial).

The STF and NTF of this architecture correspond to:

$$\mathrm{STF}(z) = \frac{a_1 a_2 z^{-2}}{(1-z^{-1})^2 + a_2 b z^{-1}(1-z^{-1}) + a_1 a_2 z^{-2}} \tag{3.9}$$

$$\mathrm{NTF}(z) = \frac{(1-z^{-1})^2}{(1-z^{-1})^2 + a_2 b z^{-1}(1-z^{-1}) + a_1 a_2 z^{-2}}$$

To achieve $\mathrm{STF}(z) = z^{-2}$ and $\mathrm{NTF}(z) = (1-z^{-1})^2$, various solutions are possible; however, $a_1 = a_2 = 1$ and $b = 2$ are commonly chosen since they ensure the best dynamic range scaling [6, 8, 10]. The interested reader is encouraged to experiment with the Boser-Wooley model provided in the Simulink® Toolbox, paying particular attention on how changing the coefficients affects the integrators' output ranges versus the signal input range and hence the dynamic range of the modulator (i.e., using the function – *integ_stats(INT'N'OUT)* – facilitates the investigation).

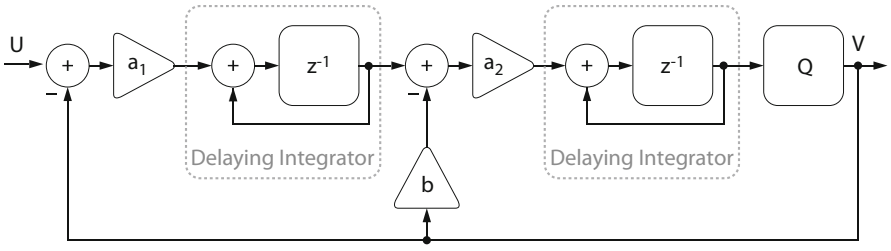


Fig. 3.14 Boser-Wooley MOD2 block diagram

The main advantage of the Boser-Wooley architecture is that, by using delaying integrators, the settling characteristics of the circuital implementation of the modulator are improved since each stage settles independently and not as a long chain. Therefore, the speed at which the integrators op-amps have to operate can be relaxed, allowing for an easier design.

Example

Analyze the modulator shown in Fig. 3.15.

Determine expressions for the noise transfer function NTF(z) and signal transfer function STF(z).

Solution

$$V = E + Y$$

$$W = \frac{1}{(1 - z^{-1})}(U - V) - V$$

$$Y = (Y + W) \text{ so } \; Y = \frac{Wz^{-1}}{(1 - z^{-1})}$$

$$V = E + \frac{Wz^{-1}}{(1 - z^{-1})}$$

$$V = E + \frac{z^{-1}}{(1 - z^{-1})^2}(U - V) - \frac{Vz^{-1}}{(1 - z^{-1})}$$

$$V\left(1 - 2z^{-1} + z^{-2}\right) = E\left(1 - z^{-1}\right)^2 + z^{-1}U - z^{-1}V - z^{-1}V + z^{-2}V$$

$$V = z^{-1}U + E\left(1 - z^{-1}\right)^2$$

$$\text{STF}(z) = z^{-1} \quad \text{and} \quad \text{NTF} = \left(1 - z^{-1}\right)^2$$

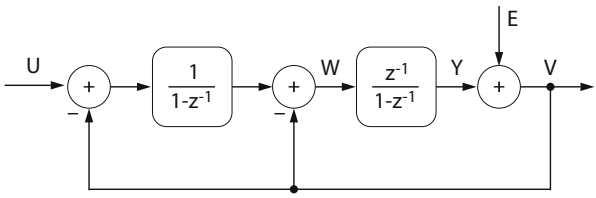


Fig. 3.15 MOD2 block diagram with one delaying integrator

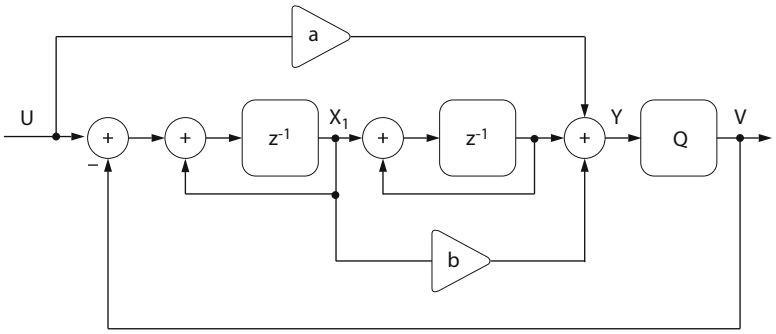


Fig. 3.16 Silva-Steensgaard MOD2 block diagram

3.7.2 The Silva-Steensgaard Modulator

The Silva-Steensgaard modulator [3, 9] is a second-order modulator where feedforward paths are used (Fig. 3.16). Moreover, delaying integrators are generally implemented for the same advantages seen in the Boser-Wooley architecture.

Important to note in this architecture is that thanks to the direct feedforward path from the input signal to the input of the quantizer, the input signal to the loop filter contains only the shaped quantization noise:

$$U(z) - V(z) = -\left(1 - z^{-1}\right)^2 E(z) \tag{3.10}$$

assuming the a and b coefficients are chosen properly. Since no signal content is present at the input of the integrators, the linearity requirements of the op-amps of the integrators are greatly relaxed, which is a significant practical advantage. Again, the interested reader is encouraged to experiment with the model of this architecture provided in the Simulink® Toolbox, paying particular attention at the integrators' output ranges. Another advantage of this architecture is that the output of the second integrator is $-z^{-2}E(z)$. This is advantageous if the modulator is the input stage of a MASH structure, as discussed in Chap. 6.

Example

Analyze the Silva-Steensgaard modulator of Fig. 3.16. Determine expressions for the NTF(z), the STF(z), and the coefficient values such that NTF(z) = $(1 - z^{-1})^2$ and STF(z) = 1.

Solution

Analyzing the Silva-Steensgaard model yields:

$$V = Y + E$$

$$Y = aU + bX_1 + X_1\frac{z^{-1}}{(1 - z^{-1})}$$

$$X_1 = \frac{z^{-1}}{(1 - z^{-1})}(U - V)$$

$$V - E = \left[\frac{a(1 - z^{-1})^2 + bz^{-1} + (1 - b)z^{-2}}{(1 - z^{-1})^2}\right]U - \left[\frac{bz^{-1} + (1 - b)z^{-2}}{(1 - z^{-1})^2}\right]V$$

$$\begin{aligned} &V(1 - z^{-1})^2 + V[bz^{-1} + (1 - b)z^{-2}] \\ &\quad = E(1 - z^{-1})^2 + U[a + (b - 2a)z^{-1} + (1 - b + a)z^{-2}] \end{aligned}$$

$$V(z) = \left[\frac{a + (b - 2a)z^{-1} + (1 - b + a)z^{-2}}{1 + (b - 2)z^{-1} + (2 - b)z^{-2}}\right]U(z) + \left[\frac{(1 - z^{-1})^2}{1 + (b - 2)z^{-1} + (2 - b)z^{-2}}\right]E(z)$$

The coefficients values are typically chosen in accordance to:

$$\text{NTF}(z) = \frac{(1 - z^{-1})^2}{1 + (b - 2)z^{-1} + (2 - b)z^{-2}} \rightarrow \left(1 - z^{-1}\right)^2 \text{ if } b = 2$$

$$\text{STF}(z) = \frac{a + (b - 2a)z^{-1} + (1 - b + a)z^{-2}}{1 + (b - 2)z^{-1} + (2 - b)z^{-2}} \rightarrow 1 \text{ if } a = 1 \text{ and } b = 2,$$

therefore achieving the typical MOD2 transfer function.

3.7.3 Architecture Selection

It results clear that a variety of modulator architectures can be developed and a question that might arise in the reader might well be: "Which architecture is the best?" In terms of typology, practical considerations such as the type of application of the converter, required dynamic range, etc., are often more decisive, rather than fundamental mathematical limits. On the other hand, when aiming to the optimal NTF, a precise mathematical investigation as demonstrated in [3] can be conducted in order to achieve an SQNR curve that is more linear and supports signals closer to full scale without saturating. Generally, a performance of peak SQNR 6 dB higher than that of a basic MOD2 is achievable. Note that, in any case, the STF results in a secondary consideration since it merely filters the signal and plays no role in determining the SQNR performance of the modulator.

3.8 Conclusions and Essential Takeaways

This chapter expanded on the previous MOD1 study by examining the properties of the basic second-order modulator – MOD2 – and some other implementations of second-order modulators. Compared to MOD1, the reader should have noticed that MOD2 behaves much more closely to an ideal modulator than MOD1 [3, 10]:

- Tonal limit cycles are still seen for DC inputs, but the strong signal-band tonal behavior seen with MOD1 is less apparent.
- Related to the above, the modulator output spectrum for sinewave inputs (and with no quantizer dither) generally follows the shape of the NTF better, lacking the many strong tones superimposed on the NTF that are usually seen with MOD1.
- Considerably higher SQNR can be achieved at a given OSR by MOD2.
- Since there are two integrators providing DC gain before the quantizer, dead zone behavior when using real integrators with finite DC gain is much improved over MOD1.
- Care must be taken to avoid integrators clipping in real modulators, especially when high SQNR is to be achieved.

Consequently, unlike MOD1 which is plagued by non-ideal behaviors, basic MOD2 and its variants, such as those examined in Sect. 3.7, are commonly seen in practice.

Despite the relatively better performance of MOD2 seen in this chapter, achieving an extremely high SQNR (e.g., >>100 dB) may still require unacceptably high OSRs [3, 10]. Recall that, for a given signal bandwidth of interest (e.g., a 24 kHz audio band), increasing the OSR increases the speed that any analog circuits in the modulator must operate on (e.g., for an OSR of 64 an audio modulator will operate at $f_s = 24 \times 2 \times 64 = 3072$ kHz). If the OSR requirement becomes too high to achieve the target SQNR, then the modulator design may become unacceptable in other regards, such as:

- Excessively high power consumption to achieve very fast integrator settling in analog modulators or small adder's delays in digital modulators.
- Excessively high power consumption to achieve very small quantizer's delays.
- Often a sufficiently fast clock may not be available in the system and so a dedicated modulator clock must then be generated, e.g., by adding a PLL with its associated circuits costs and power requirements (i.e., the modulator's clock requirements can be further complicated by the fact that the modulator data must then synchronize to the rest of the system, ideally without the further complication of needing sample rate conversion if the modulator and system clocks do not have simple ratios such as 1:2, 1:1.5, etc.).

- In extreme cases, the circuit speeds required to achieve the necessary OSR may even exceed the various speed limits of the medium in which the modulator is to be realized (e.g., available MOS transistor fT and/or channel transit times in ICs).

In very high SQNR applications (e.g., >100 dB), a higher-order modulator such as MOD3 operating at moderate OSR can thus be a better solution than MOD2 operating at very high OSR. However, increasing the modulator order above two introduces other problems, such as possible loop instability due to each integrator adding further loop phase, which degrades phase margins. Therefore, the next chapter examines some of these effects and different approaches to successfully implement high-order modulators.

Exercises

Q.1
Calculate the baseband noise power of a second-order modulator with OSR = 256 and feedback levels ±1 V.

Q.2
Calculate the signal-to-quantization-noise ratio of a second-order modulator with input amplitude -3 dBFS (full scale), OSR = 256, and feedback levels ±1 V.

Q.3
A second-order modulator is to achieve a 16-bit resolution and operate at a sampling frequency of 10 MHz and ± 1 V feedback levels. Calculate the maximum baseband frequency that can be obtained.

Q.4
Calculate the amplifier gain required to reduce the dead zone width of a second-order Sigma-Delta modulator with feedback levels ±1 V to below 100 μV.

Q.5
Analyze the Boser-Wooley modulator of Fig. 3.15 in Sect. 3.7.2. Determine expressions for the noise transfer function NTF(z) and signal transfer function STF (z).

Q.6
Analyze the error-feedback MOD2 illustrated in Fig. 3.17. Determine expressions for the noise transfer function NTF(z) and signal transfer function STF(z).

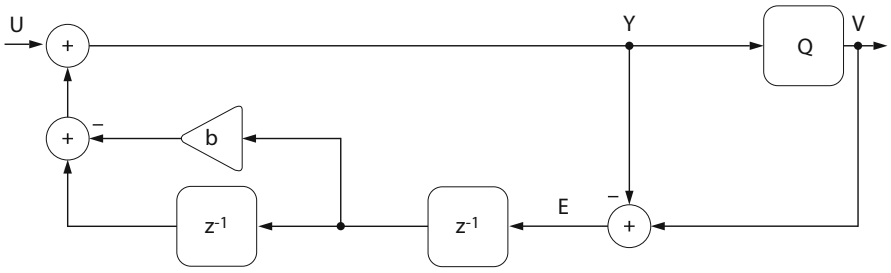


Fig. 3.17 Error-feedback MOD2 block diagram

Q.7
Determine the optimum value of the coefficient b in Fig. 3.17 for second-order noise shaping. Assuming b is mismatched by 1%, perform a Simulink® simulation to determine the maximum SQNR obtainable from the modulator at OSR = 64 and full scale ± 1. Compare this to the value obtainable without mismatch. Discuss your results.

Q.8
What effects does finite integrator op-amp gain have on the performance of a Sigma-Delta ADC? Are second-order modulators sensitive to this non-ideality?

Q.9
Briefly explain the issue of integrator saturation in a MOD2. How can it be avoided?

Q.10
Discuss the advantages and disadvantages of adding dither to Sigma-Delta modulators.

Q.11
For a second-order low-pass Sigma-Delta modulator, briefly explain the requirements for the integrators forming the loop filter.

Q.12
Create a Simulink® model of the second-order Silva-Steensgaard modulator and simulate the SQNR for a sinewave input of 12 KHz at -3 dBFS. Use the default settings of the – *load_par* – routine. Further plot a graph of the SQNR for different amplitudes of the input sinewave.

Q.13
Briefly explain the advantages and disadvantages of using a second-order modulator instead of the simple MOD1.

References

1. Oppenheim AV, Schafer RW. Discrete-time signal processing. New Jersey: Prentice-Hall Inc.; 1989.
2. MathWorks. Sigma-Delta A/D conversion. [Online] https://it.mathworks.com/help/dsp/examples/sigma-delta-a-d- conversion.html?requestedDomain=www.mathworks.com, 2017.
3. Schreier R, Temes GC. Understanding Delta-Sigma data converters. New Jersey: Wiley; 2005.
4. Candy JC, Benjamin OJ. The structure of quantization noise from Sigma-Delta modulation. IEEE Trans Commun. 1981;29(9):1316–23.
5. Friedman V. The structure of the limit cycles in Sigma Delta modulation. IEEE Trans Commun. 1988;36(8):972–9.
6. De la Rosa JM, Del Rio RF. CMOS Sigma-Delta converters: practical design guide. London: Wiley; 2013.
7. Norsworthy SR, Schreier R, Temes GC. Delta-Sigma data converters: theory, design and simulation: Wiley-IEEE Press; 1996.
8. Candy JC, Temes GC. Oversampling Delta-Sigma data converters: theory, design and simulation. USA: Wiley-IEEE Press; 1991.
9. Casier H, Steyaert M, Van Roermund AHM. Analog circuit design: robust design, Sigma Delta converters, RFID. London: Springer; 2011.
10. Reiss JD. Understanding Sigma-Delta modulation: the solved and unsolved issues. JAES. 2008;56(1/2):49–64.



High-Order Sigma-Delta Modulators

4

This chapter comprises two sets of exercises. In the first set, the main properties of a third-order modulator (MOD3) are investigated, following the procedure of previous chapters. Based on the simulations proposed, it is demonstrated that modulators with noise-shaping order higher than two can be unstable for some or even all possible inputs, usually displaying chaotic limit cycle behavior unrelated to the input signal when unstable. However, careful architecture selection for single-loop modulators – which entails controlling the form of NTF to limit its maximum gain – and the implementation of multi-bit quantization will be investigated, with the aim to demonstrate how high-order modulators stability and stable input range can be increased. Following the results obtained from the simulations proposed, an investigation of the theory required to design efficient high-order modulators will be presented. The architectures analyzed comprise Cascaded Integrators Feedback (CIFB), Cascaded Integrators Feedforward (CIFF), Cascaded Resonators Feedback (CRFB), and Cascaded Resonators Feedforward (CRFF). Regarding the second section of this chapter, the widely used MATLAB®-based Schreier's Delta-Sigma Toolbox is introduced as an effective tool to generate coefficients for generic high-order modulator structures. Moreover, a simple design example to demonstrate the procedure to generate the coefficients required by the Simulink® models provided will be presented.

4.1 Simulink® Model of MOD3

The basic idea for increasing the order L of modulators, as already discussed, is to connect additional integrator stages in series in order to achieve [1, 2]:

$$V(z) = U(z) + \left(1 - z^{-1}\right)^{L} E(z) \tag{4.1}$$

© Springer International Publishing AG, part of Springer Nature 2019

I. Arnaldi, *Design of Sigma-Delta Converters in MATLAB®/Simulink®*,
https://doi.org/10.1007/978-3-319-91539-5_4

Therefore, similarly to the low-order modulators analyzed so far, Fig. 4.1 illustrates the respective Simulink® model of MOD3 available in the Toolbox under the filename *"mod3.mdl."* Following the same analysis procedure that has been seen so far, the modulator behavior for DC inputs is first examined.

In the Toolbox open the model *"mod3.mdl"* found in the folder *4_MOD3*.

Type – *load_par* – in the Matlab Command Window to initialize the variables required by the model and run the simulation.

Once the simulation is finished, open the time scope by double-clicking on any the small blue glasses found in the signal connections to observe the waveforms at the nodes of the modulator. The reader may wish to use the horizontal magnifying glass to zoom into a few cycles of the INT1OUT, INT2OUT, INT3OUT, and MOD3OUT waveforms.

Try various rational and irrational DC input values: 0.5, $-pi/10$, etc.

Regardless of the input used, the reader should come to the conclusion that the system is always unstable. Particular attention should be paid to the magnitude of the integrators outputs, realizing that these are completely unbounded. This is because each integrator provides a 90° phase shift, resulting in a total of 270° in a third-order modulator, implying loop instability [3].

4.2 Introduction to Stability of High-Order Modulators

The previous exercise highlighted that basic MOD3, which is obtained from MOD2 by replacing the quantizer of MOD2 with another MOD1, is completely unstable with DC inputs. The reader should not be surprised to learn it is unstable with sinewave inputs too and that the system does not recover from instability even if the input signal decreases to very small values. This does not mean, however, that all third- and higher-order modulators are completely unstable. Considering that the STF acts essentially like a prefilter, the stable input range of a ΣΔ-modulator is primarily determined by the NTF and the number of bits in the quantizer, although the STF does have some incidence too, as discussed later in the text. Having considered only single-bit implementations so far, what's left is to focus on the NTF properties required to achieve stable operations. Unfortunately, simple and exact parameters are not known! Despite extensive research on the topic is available [1, 4, 5, 6], the proven results are generally either too restrictive (i.e., that is, too conservative) or apply only to specific modulators with constant inputs. The most widely used approximate criterion [1, 6, 7, 8, 9] to design stable, high-order NTFs is the (modified) *Lee's criterion* (Fig. 4.2):

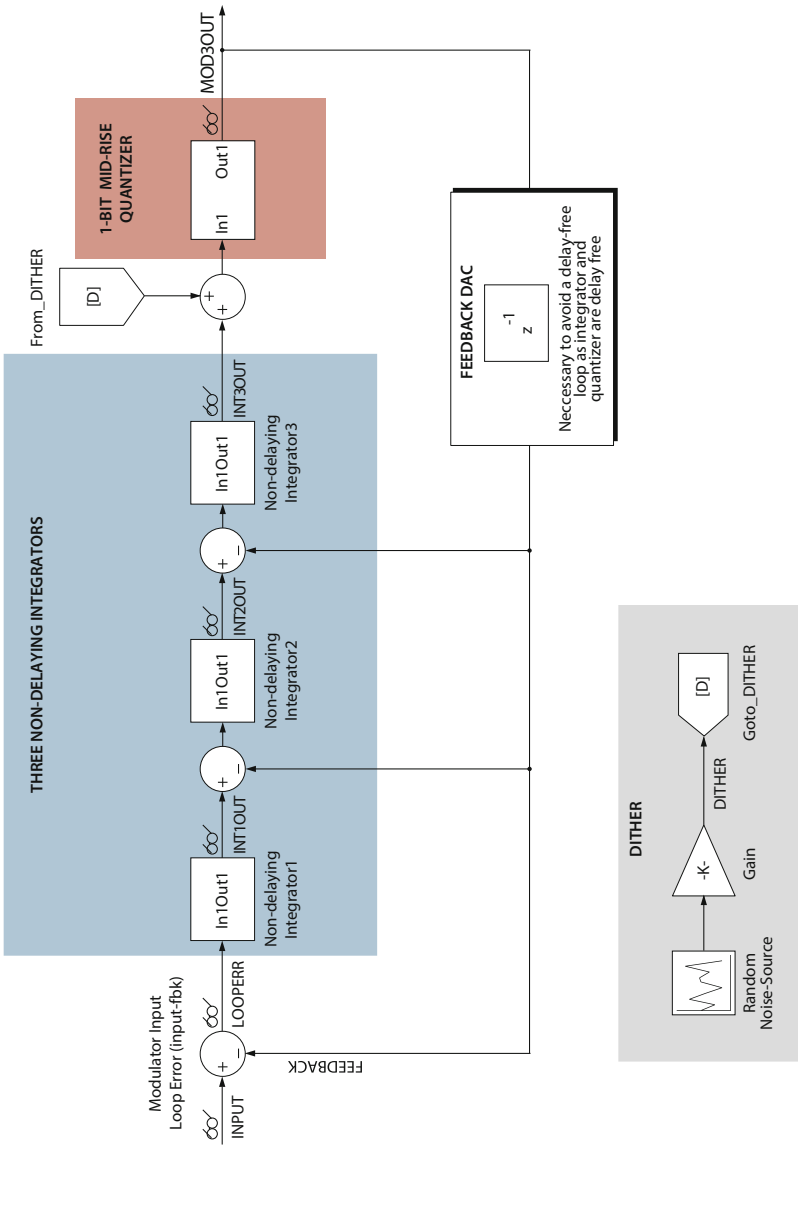


Fig. 4.1 SIMULINK® Model of MOD3

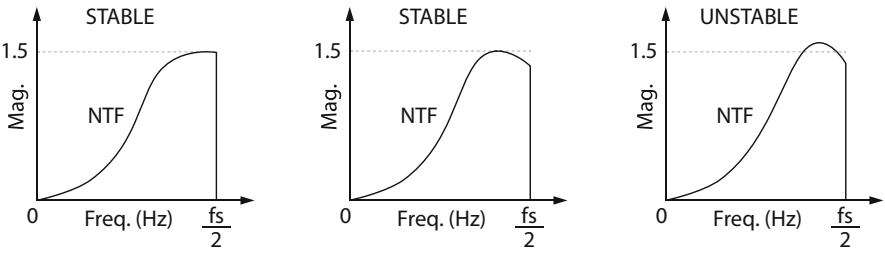


Fig. 4.2 Lee's criterion representation

A ΣΔ-modulator with binary quantizer is likely to be stable if the maximum NTF gain results $\max|\mathrm{NTF}(e^{j2\pi f})| < 1.5$.

This is an approximate "rule of thumb" which is neither necessary nor sufficient (i.e., the criterion says nothing about a limit on the input signal) to ensure stability and has obvious exceptions, e.g., the basic second-order modulator studied in the previous chapter had a $\max \mid \mathrm{NTF}(e^{j2\pi f}) \mid = 4$. Therefore, although the Lee's criterion is a helpful guideline for predicting a priori instabilities in single-bit modulators, the designer of high-order single-bit modulators must resort to extensive simulations to confirm correct operation of high-order modulators.

4.3 An Improved Third-Order Modulator

Before entering too much into theoretical details, an example of a functional MOD3 is here introduced in order to gain an appreciation of the workings of high-order modulators. Similarly to the MOD2 Silva-Steensgaard architecture, the "improved" MOD3 uses Cascaded Integrators with Feedforward paths (i.e., also known as CIFF architecture) to control the placement of the NTF's poles and zeros and thus the NTF gain (i.e., how to design such architectures will be shown later in the text). The proposed "*mod3_ciff_1bit.mdl*" model available in the Simulink® Toolbox has thus a reduced (third-order) maximum NTF gain from 8 to 1.71 (i.e., at $z = -1$, which corresponds to Fs/2) to achieve a moderately stable system. The penalty for achieving this is that the new NTF is now approximately 20 dB higher than the ideal MOD3 NTF $(1 - z^{-1})^3$ in the signal passband and hence about 20 dB more in-band quantization noise (i.e., the SQNR for a given signal is about 20 dB less than a true $(1 - z^{-1})^3$ NTF of which the modulator could achieve). It should also be mentioned that in order to demonstrate that the Lee's criterion is indeed an approximate assumption, a max|NTF| gain of 1.71 has been chosen instead of 1.5. Generally [1, 7, 9], slightly higher values of max|NTF| gains are tolerable for moderate-order modulators (e.g., order 3 or 4), while very high-order loop filters (e.g., 5 or more) usually require a more conservative max|NTF| (e.g., <1.4). Further, in the

"mod3_ciff_1bit.mdl" model, note the alternating delaying and non-delaying integrators are used, which helps to reduce the noise peaking and hence improve the stability of the modulator [1, 4]. Again, this highlights the importance of simulations in the design of ΣΔ-modulators since not precise theoretical mathematical tools are yet available to ensure, with no doubt, correct operation of the system.

4.3.1 Operation for DC Inputs

In the Toolbox open the model *"mod3_ciff_1bit.mdl"* found in the folder *4_MOD3*.

Type – *load_par* – in the Matlab Command Window to initialize the variables required by the model and run the simulation.

Once the simulation is finished, open the time scope by double-clicking on any the small blue glasses found in the signal connections to observe the waveforms at the nodes of the modulator. The reader may wish to use the horizontal magnifying glass to zoom into a few cycles of the INT1OUT, INT2OUT, INT3OUT, and MOD3OUT waveforms.

Type – *mod_SNDR* – in the Matlab Command Window to obtain the output spectrum.

Save the result and repeat for the following DC input values: 0.25, 0.5, 0.75, 0.8, and 0.9.

As an example, Fig. 4.3 reports the time domain and FFT results for a DC input signal of 0.5. Similar results for all the DC input values $\leq$0.75 proposed should be obtained.

From the results, it should be apparent that:

- The *mean* value of the filtered output bit sequence of the modulator output accurately represents the analog DC input, hence confirming the ability of the CIFF-MOD3 to convert signals successfully.
- The modulator output is periodic.
- Due to the periodic behavior, noise shaping is not apparent and strong tones are present in the spectrum.
- The modulator is stable only for a limited range of DC inputs (e.g., $\leq$0.75).
- As seen in previous chapters, dither could be used to recover the tonal behavior of the modulator. The main drawback of this practice in high-order modulators is a further reduction of the stable input range and achievable SQNR [3].

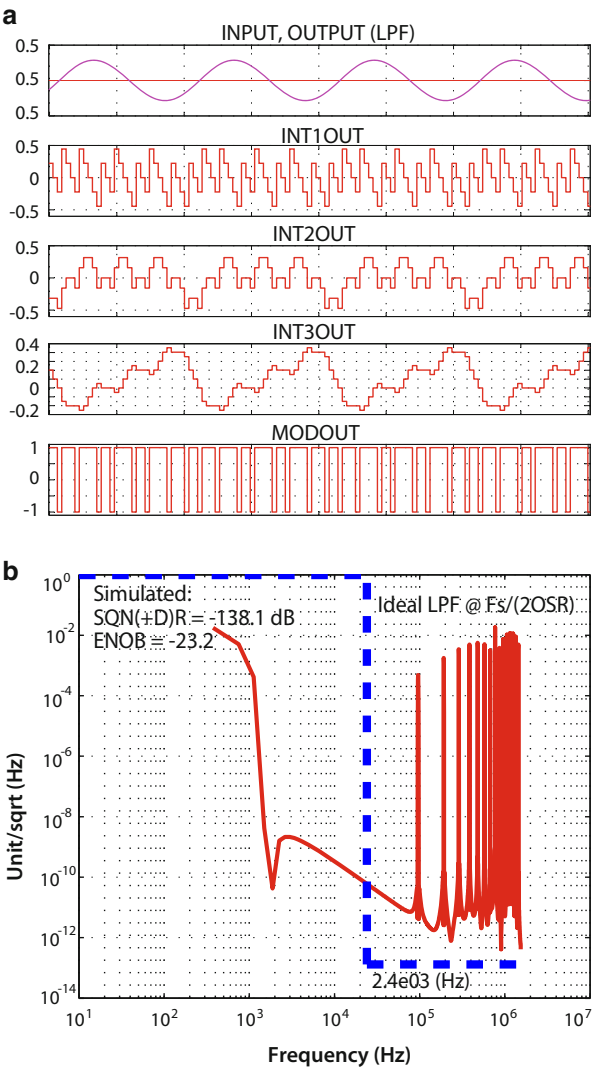


Fig. 4.3 Time domain (top) and FFT (bottom) results for DC input signal 0.5

4.3.2 The Effects of Finite Integrator Gain

Recalling from previous chapters that for analog modulators where the integrators may have a finite DC gain, A MOD1 required very high gain to minimize dead zones (i.e., its dead zone width is proportional to $1/A$) and MOD2 was much more tolerant of low gains, as the two integrator gains are multiplied before the input reaches the quantizer (dead zone width proportional to $\approx 1/A^2$), it is useful to investigate if this pattern applies to the stable MOD3 as well.

Close the model *"mod_ciff_1bit.mdl"* and type – *load_par* – into the Matlab Command Window for resetting initial model's parameters.

Type – *target_mod = 'sweep_mod3_ciff_1bit'* – into the Matlab Command Window.

Open "*sweep_testbench.mdl,*" right click on the *Modulator* block and enter in the *Model Reference Parameters*. Change the *Model Name* to *sweep_mod3_ciff_1bit* and press *ok*. Save and close the model.

Set the DC gain of all integrators to 10 by typing – *mod.igain1 = 10; mod.igain2 = 10; mod.igain3 = 10; –* into the Matlab Command Window.

Type – *sweep_dc_dc* – into the Matlab Command Window and run the routine. A DC sweep for a subset of the input range (i.e., ±0.1) will be performed.

Similar results of Fig. 4.4 should be obtained. Comparing the DC transfer function with finite integrator gains of MOD3 to the ones of MOD1 and MOD2, it should be clear that the gain of which the width of the dead zone around DC = 0 depends is proportional to the gain power of 3 (i.e., steps of 0.001), hence resulting very small. The interested reader may wish to repeat the simulation with smaller values of integrator gains in order to investigate the greater tolerance of MOD3 to dead zones when compared to low-order modulators. From this investigation, it should result clear that achieving a high enough DC integrator gains to suppress dead zones is likely to be a minor issue for real implementations of high-order modulators [1, 5, 7].

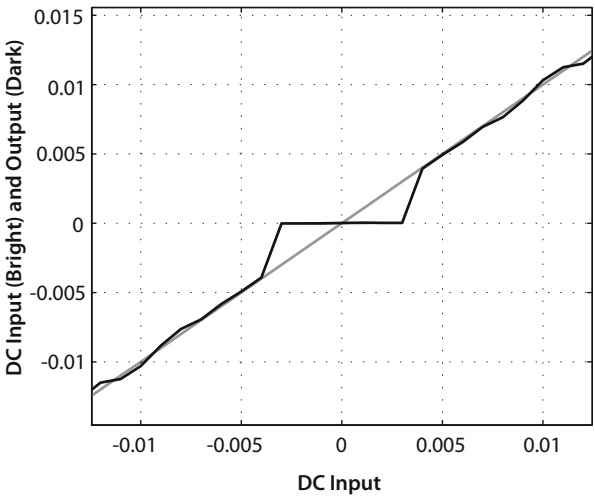


Fig. 4.4 CIFF-MOD3 DC transfer function around DC input = 0 with finite integrator gain of 10

4.3.3 Operation for Sinewave Inputs

In the Toolbox open the model *"mod3_ciff_1bit.mdl"* found in the folder *4_MOD3*.

Type – *load_par* – in the Matlab Command Window to initialize the variables required by the model and run the simulation.

Set the sinewave amplitude by typing – *sinamp = 0.1* – into the Matlab Command Window (i.e., the default value of 0.707 would make the CIFF-MOD3 unstable!).

In the model double-click on the *manual switch* to select the *signal generator* input source, and run the simulation.

Once the simulation is finished, open the time scope by double-clicking on any the small blue glasses found in the signal connections to observe the waveforms at the nodes of the modulator. The reader may wish to use the horizontal magnifying glass to zoom into a few cycles of the INT1OUT, INT2OUT, INT3OUT, and MOD3OUT waveforms.

Type – *mod_SNDR* – into the Matlab Command Window to obtain the FFT graph.

The resulting CIFF-MOD3 FFT output spectrum of Fig. 4.5 clearly illustrates the noise-shaping characteristic, where the 60 dB per decade slope confirms the third-order nature of the loop filter.

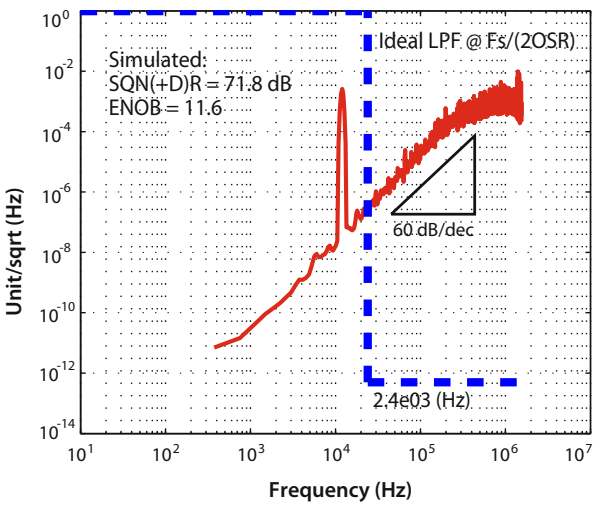


Fig. 4.5 FFT Results for CIFF-MOD3 with sinewave input

Example

Verify that the SQNR found in Fig. 4.5 of approximately 72 dB corresponds to what found from theoretical calculations.

Solution
To verify the value of SQNR achieved, it is useful to compare its value with the theoretical SQNR from the formula [1]:

$$\begin{aligned} \mathrm{SQNR} &= 6.02N + 1.76 + (20L + 10)\log_{10}(\mathrm{OSR}) - 10\log_{10}\left(\frac{\pi^{2L}}{2L+1}\right) \\ &\cong 112.8\ (\mathrm{dB}) \end{aligned} \tag{4.2}$$

where the modulator order is $L = 3$, the oversampling ratio is $\mathrm{OSR} = 64$, and the number of bits corresponds to $N = 1$.

Note that Eq. 4.2 corresponds to the theoretical SQNR for a full-scale input – *sinamp = 1* (i.e., 0 dBFS). Having used an amplitude of 0.1 for the simulation, corresponding to − 20 dB, and recalling also the discussion at the beginning of this section about the passband penalty of − 20 dB incurred by modifying the NTF in order to achieve stability, Eq. 4.2 needs to be modified appropriately in order to be able to compare the results obtained with the theory. Therefore, by subtracting the value of − 40 dB to the results obtained in Eq. 4.2 (i.e., 112.8 – 40 = 72.8 dB), it is easily seen that the modulator matches relatively well the expected theoretical value. Therefore, the modulator can be considered to work correctly.

In order to appreciate the achievable SQNR within the stable input range for this typology of MOD3 and to compare it with the one of MOD1 and MOD2, it is useful to perform a sweep of sinewave amplitudes.

Reset initial conditions by typing – *load_par* – into the Matlab Command Window.

Set the sinewave frequency to 6 KHz by typing – *sinfreq = 6e3* – into the Matlab Command Window.

Type – *target_mod = 'mod3_ciff_1bit'* – into the Matlab Command Window.

Type – *sweep_sinamp* – into the Matlab Command Window and run the simulation. Note it may take some time to complete.

Similar results to that of Fig. 4.6 should be obtained.

The first thing to notice is the largely improved linearity of the SQNR curve when compared to the ones of MOD2 and MOD1. This is the consequence of the higher loop gain obtained from using a larger number of integrators [1, 5]. Therefore, it results that CIFF-MOD3 behaves much more like an ideal modulator, being more tolerant to tones and dead zone.

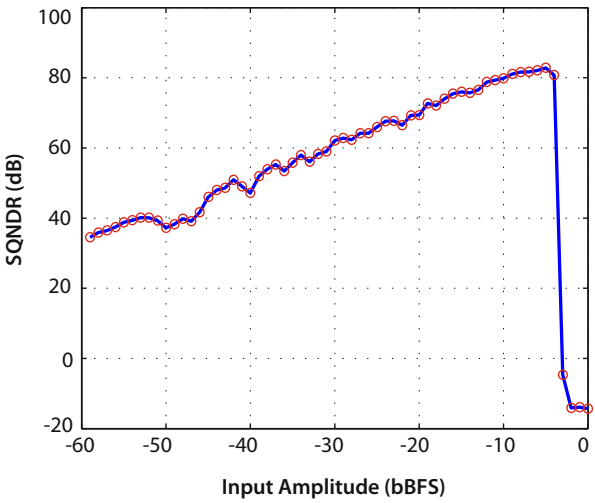


Fig. 4.6 SQNR vs. swept amplitude of a 6 KHz Sine Input for CIFF-MOD3

The peak SQNR achieved is approximately 82 dB for a maximum sinewave input amplitude of about 0.65. Comparing it with the respective peak SQNR of MOD2, it can be seen that this is only about 12 dB higher, which is not much when considering the difference in peak SQNR between MOD1 and MOD2. This is due to both the reduction of the ideal third-order NTF gain to make MOD3 stable and the limited input range available [3].

Regarding the stable input range, it is seen that for input amplitudes larger than 0.7, the system becomes unstable, and the SQNR curve rapidly falls to negative values. This is due to the quantizer gain being signal dependent, as already investigated in previous chapters. Therefore, since the magnitude of the input to the quantizer is of relevance in determining the stability of the loop, important to note at this point is that the STF must also be similarly relevant in that it effectively provides a (frequency-dependent) scaling of the input magnitude [1, 4, 7, 8].

Recalling that proper modulator operation is assured if the loop filter remains linear (i.e., internal signals do not grow so large as to saturate the op-amps) and if the internal quantizer is not severely overloaded, it follows that the stable input range must be less than or equal to the full scale of the feedback DAC [1, 6]. In a high-order modulator, especially in one employing single-bit quantization, the stable input range is usually a few dB below the full-scale range of the feedback DAC, as seen from the simulation. This loss in range usually results from the nonlinear effects of quantizer overload, rather than from an insufficient linear range in the loop filter [4, 7]. Since the input to the quantizer is given by:

$$Y(z) = \mathrm{STF}(z)U(z) + (\mathrm{NTF}(z) - 1)E(z) \tag{4.3}$$

it is clear that when the modulator input $u(n)$ – as amplified by the STF – approaches the edge of the no-overload range of the quantizer, the addition of filtered quantization error may push $y(n)$ into the overload range. This overload will increase

$e(n)$, which will aggravate the original overload, and this vicious circle eventually leads in a severe quantizer overload and hence the saturation of the active blocks. Correction of this runaway behavior often requires some sort of intervention, such as modulator reset, since the return of the input to the stable input range of the modulator may not be sufficient to restore stable operation [5, 9]. Further, it should also be noted that some rapidly varying input signals may drive an apparently stable modulator loop into instability, even if their amplitudes are relatively small [7]. This further highlights the importance of the frequency-dependent amplification of the STF. To better understand the concept [3], note that the STF is generally designed to have a magnitude of 1 for signals in the passband, at the cost of some amplification of high-frequency signals that are outside the band of interest. If such amplification should result too, severe, quantizer overload may occur. Since rapidly varying input signals have larger high-frequency content than slow varying ones, these are more prone to negatively affect the modulator stability due to the STF amplification at those frequencies. Therefore, when designing high-order modulators, particular attention should be paid to the STF characteristic too, in order to limit stability issues. Unfortunately, since ΣΔ-modulators are nonlinear systems, the dependency of the quantizer gain to the input signal cannot be captured by any linear model, and, thus, accurate and straightforward mathematical tools to predict the stable input range are not available. Therefore, the designer of high-order modulators must resort to simulations [1, 4, 9] in order to accurately define the stable input range, as well as to ensure correct operation under worst-case input signals (e.g., maximum amplitude square wave at the dominant pole frequency of the NTF).

It should be clear that implementing high-order modulators successfully, as well as achieving high SQNR values (e.g., >100 dB), is not an easy or intuitive task as it may have appeared in the initial studying of MOD1 and MOD2. Therefore, the next paragraphs will study techniques and theoretical considerations that can be used to maximize the potential of high-order modulators in order to achieve highly efficient converters.

4.4 CIFF ΣΔ-Modulator with Multi-bit Quantization

The previous exercises examined stability issues in high-order ΣΔ-loops and demonstrated how a stable third-order modulator can be constructed by using different loop architectures (e.g., CIFF-MOD3) than the basic MOD3. However, the CIFF modulator considered is only stable for inputs in the range of, approximately, $-0.7 \leq \text{input} \leq 0.7$. One common way to further improve loop stability and extend the stable input range is to progress from using a single-bit quantizer and feedback path to multi-bit quantization and feedback. In [1, 6, 7], a detailed theoretical analysis for predicting (to some approximation) the stable input range achievable relative to the increase in the number of bits is presented; however, since extensive behavioral simulations are still advisable before implementing the modulator, such analysis is not to be reported here. Generally, the lower the resolution of the

quantizer used, the more suspicious the designer should be about unforeseen instability [9]; however, the practical approach undertaken in this book should result sufficient for the inexperienced designer to confidently use multi-bit techniques in ΣΔ-modulators.

The next exercise proposes to utilize the same CIFF-MOD3, but now with a 2-bit quantizer. Thus, the quantizer step size is 2/3 (e.g., switching thresholds, −2/3, 0, +2/3, and output levels, −1, −1/3, +1/3, +1), instead of an output step size of 2 (e.g., −1 to +1) as seen for the 1-bit quantizer.

In the Toolbox open the model *"mod3_ciff_2bit.mdl"* found into the folder *4_MOD3*.

Type – *load_par* – into the Matlab Command Window to reset the default variables needed for the model to function, and run the simulation.

In the model double-click on the *manual switch* to select the *signal generator* input source, and run the simulations.

Once the simulation is finished, open the time scope by double-clicking on any the small blue glasses found in the signal connections to observe the waveforms at the nodes of the modulator. The reader may wish to use the horizontal magnifying glass to zoom into a few cycles of the INT1OUT, INT2OUT, INT3OUT, and MOD3OUT waveforms.

Type – *mod_SNDR* – into the Matlab Command Window to obtain the FFT graph.

Figure 4.7 illustrates the time domain and output spectrum results. It is apparent that:

- The value of the filtered modulator output accurately represents the analog DC input, confirming the ability of CIFF-MOD3 with a 2-bit quantizer to convert signals successfully.
- The modulator output is not periodic.
- The 2-bit modulator output is a less coarse representation of the sinewave analog input, and the 2-bit operation is clearly recognizable.
- The loop error signal amplitude is smaller compared to the 1-bit case (i.e., lower noise) and has reduced signal content.
- The output spectrum clearly exhibits a third-order noise-shaping characteristic.

Example

Verify that the SQNR found in Fig. 4.7 of approximately 96 dB corresponds to what found from theoretical calculations.

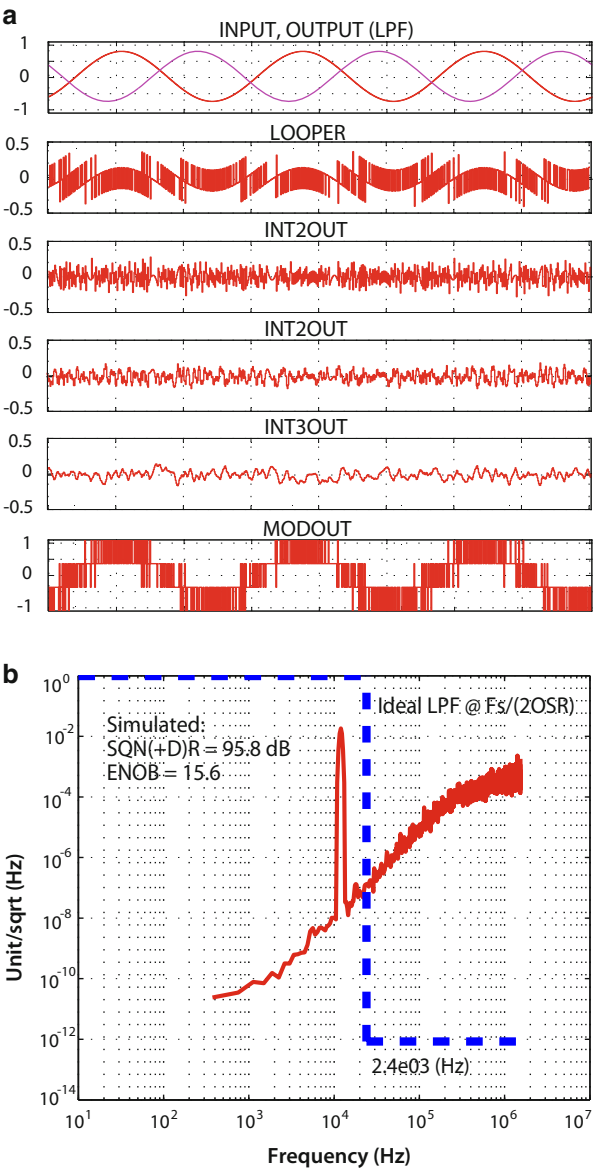


Fig. 4.7 FFT results for CIFF-MOD3 with sinewave input

Solution

To verify the value of SQNR achieved, it is useful to compare its value with the theoretical SQNR from Eq. 4.2 (i.e., ≈113 dB). As already mentioned, note that Eq. 4.2 corresponds to the theoretical SQNR for a full-scale input – sinamp = 1 (i.e., 0 dBFS). Having used an amplitude of 0.707 in the simulation,

corresponding to − 3 dB, and recalling the passband penalty of − 20 dB incurred by modifying the NTF in order to achieve stability, Eq. 4.2 needs to be modified appropriately to compare the results obtained with the theory. Further, since each additional bit in the quantizer corresponds to a 6 dB increase in resolution, then Eq. 4.2 can be rearranged as:

$$\text{SQNR} = 112.8 - 20 - 3 + 6 = 95.8 \text{ dB} \tag{4.4}$$

It is easily seen that the modulator matches the expected theoretical value relatively well; therefore, it can be considered to work correctly.

In order to verify the maximum SQNR achievable, as well as the stable input range, it is useful to perform a sweep of the sinewave input amplitude.

Set the initial conditions by typing – *load_par* – into the Matlab Command Window.

Set the sinewave frequency to 6 KHz by typing – *sinfreq = 6e3* – into the Matlab Command Window.

Open "*sweep_testbench.mdl,*" right click on the *Modulator* block, and enter in the *Model Reference Parameters*. Change the *Model Name* to *sweep_mod3_ciff_2bit* and press *ok*. Save and close the model.

Type – *target_mod = 'mod3_ciff_2bit'* – into the Matlab Command Window.

Type – *sweep_sinamp* – into the Matlab Command Window and run the simulation. Note it may take some time to complete.

Comparing Fig. 4.6 with Fig. 4.8, it can be seen that the achievable peak SQNR is highly improved, increasing from approximately 82 dB (Fig. 4.6) to 94 dB. Note that the increase in peak SQNR is greater than the 6 dB expected by the addition of 1-bit in the quantizer because the maximum stable input range also improved from about $\leq$0.7 to roughly $\leq$0.9. This extended input range is an important advantage since the stability of the modulator result much more robust (i.e., recall that instability in high-order modulators is often not recoverable).

Example

Investigate if further improvements can be achieved by using the *CIFF-MOD3* with a 3-bit quantizer. Which are the quantizer's levels? Compare the results with the 1- and 2-bit cases.

Solution

The 3-bit (i.e., 8 levels) quantizer has equally spaced input switching thresholds at (e.g., −6/7, −4/7, −2/7, 0, 2/7, 4/7, 6/7) and outputs levels (e.g., −1, −5/7, −3/7, −1/7, 1/7, 3/7, 5/7, 1).

Open "sweep_testbench.mdl," right click on the Modulator block and enter in the Model Reference Parameters. Change the Model Name to *sweep_mod3_ciff_3bit* and press ok. Save and close the model.

Type – *target_mod = 'mod3_ciff_3bit'* – into the Matlab Command Window.

Type – *sweep_sinamp* – into the Matlab Command Window and run the simulation. Note it may take some time to complete.

Comparing the results of Fig. 4.9 with that of the 2-bit case (Fig. 4.8), it results that the additional bit in the quantizer allows for a further 6 dB increase in SQNR across all frequencies, as expected. The peak SQNR is now found at roughly 99 dB. Regarding the maximum stable input range, this is only slightly improved, with the modulator being able to provide stable operation, as well as high SQNR values, up to an amplitude of approximately 0.95 (i.e., note that worst-case signals have not been considered yet, so the results may be too optimistic!).

At this point, it should be mentioned that a trade-off exists when utilizing multi-bit quantization as a technique to improve the operation of high-order modulators [1, 4, 6, 7]. If increasing the number of bits improves the overall SQNR and stable input range, the modulator performance is still heavily affected by the architectural limitations (i.e., max|NTF|, order, etc.) that are intrinsic to the architecture design. Further, increasing the number of bits increases the complexity, area, and power consumption of the system, which is undesirable. In addition, as discussed in the next chapter, real multi-bit implementations are highly affected by a series of non-idealities that greatly degrade the modulator performance, rather than improving it, and as such are not easily implementable at a circuital level. Therefore, in order to improve the SQNR performance of high-order modulators while saving area and power consumption, it may be often advisable to maximize the design of a given architecture at first, rather than implementing a higher number of quantizer bits. In

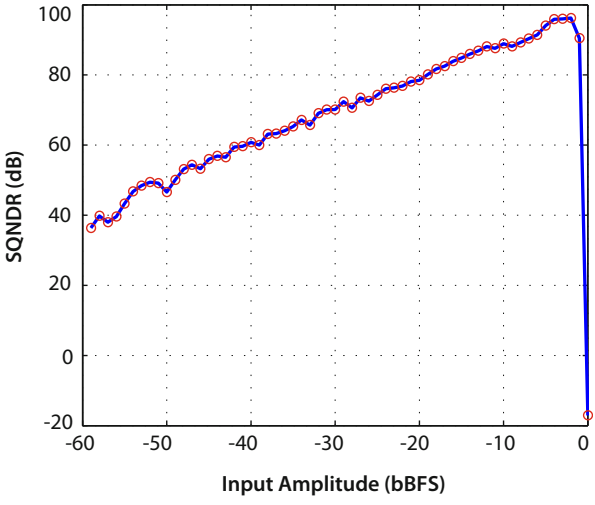


Fig. 4.8 SQNR vs. swept amplitude of a 6 KHz Sine Input for CIFF-MOD3 with 2-bit quantizer

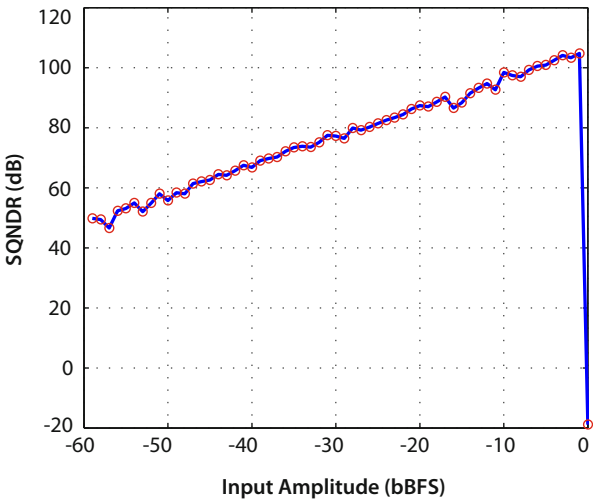


Fig. 4.9 SQNR vs. swept amplitude of a 6 KHz Sine Input for CIFF-MOD3 with 3-bit quantizer

the next paragraph, a theoretical investigation of the architectures which allow achieving high-performance and high-order modulators is analyzed in order to provide the knowledge needed to design efficient ΣΔ-converters.

4.5 Theoretical Analysis

This paragraph briefly reports the most fundamentals theoretical concepts needed for designing high-order modulators. The most interested reader is encouraged to consult [1] for a more comprehensive background and detailed mathematical information.

4.5.1 High-Order Modulators Realizability

Figure 4.10 shows the general structure of a single-bit ΣΔ-modulator [1], divided into two parts: a *linear part* (e.g., the loop filter) containing *memory* elements (e.g., the integrators) and a *memoryless nonlinear part* (e.g., the quantizer). The loop filter is a two-input system whose single output Y can be expressed as a linear combination of its inputs U and V:

$$Y(z) = L_0(z)U(z) + L_1(z)V(z) \tag{4.5}$$

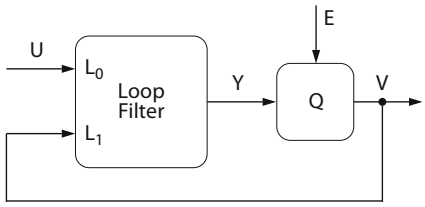


Fig. 4.10 General structure of a ΣΔ-modulator (single-bit)

The quantizer operation is, as usual, described as the addition of an error signal:

$$V(z) = Y(z) + E(z) \tag{4.6}$$

Using these two equations to express the output V as a linear combination of the modulator input U and the quantization error E yields:

$$V(z) = \mathrm{STF}(z)U(z) + \mathrm{NTF}(z)E(z) \tag{4.7}$$

where:

$$\mathrm{NTF}(z) = \frac{1}{1 - L_1(z)} \quad \text{and} \quad \mathrm{STF}(z) = \frac{L_0(z)}{1 - L_1(z)} \tag{4.8}$$

Conversely, the loop filter transfer functions which are required to implement the desired NTF and STF can be computed by rearranging Eq. 4.8:

$$L_0(z) = \frac{\mathrm{STF}(z)}{\mathrm{NTF}(z)} \quad \text{and} \quad L_1(z) = 1 - \frac{1}{\mathrm{NTF}(z)} \tag{4.9}$$

Since these relations apply regardless of the structure of the loop filter, the input-output characteristics of the loop filter (and hence of the modulator) are determined solely by the STF, NTF, and the properties of the quantizer. Structural details of the loop filter are thus irrelevant in determining the (ideal) input-output behavior of the modulator [3, 4].

Recalling that the quantizer Q in Fig. 4.10 is, in practice, an analog-to-digital converter, $y(n)$ has to be an analog signal (i.e., typically voltage), while $v(n)$ has to be a digital data stream. Therefore, Eq. 4.6 implies that the reference voltage *VREF* of the ADC is assumed to be unity or, equivalently, that all analog quantities are in fact normalized to *VREF*. This also means that an ideal DAC using the same *VREF* is a unity gain block [1]. Hence, the DAC was omitted from the feedback path of the structure of Fig. 4.10. Note that this would not be the case for multi-bit architectures, where the DAC would have to be included [6].

According to Eq. 4.8 and to the theory discussed in Chap. 1, L_1 must be large in the signal passband (e.g., 0 to f_b) to reduce the NTF there, while L_0 must be large in the same range to allow the STF to remain close to unity. Therefore, both L_1 and L_0

should have their poles in this range. In fact, as can be foreseen from the structure of Fig. 4.10 and its input-output relations (Eqs. 4.5, 4.6, 4.7, 4.8, and 4.9), due to the shared circuitry used to realize both L_0 and L_1, these two functions indeed usually have the same poles (i.e., which are also the zeros of NTF). However, L_0 and L_1 generally have different zeros.

Important to note is also that there must be at least one clock period of delay in the loop containing Q and L_1, since a delay-free loop would be physically unrealizable (i.e., $y(n)$ cannot depend on $v(n)$). Hence, an important realizability condition results [1, 2], which is that the impulse response – $h(n)$ – corresponding to the NTF(z) must satisfy the condition $h(0) = 1$. From the equation $\mathrm{NTF}(z) = h(0) + z^{-1}h(1) + z^{-2h(2)} + \cdots$, it can also easily be seen that this is equivalent to the condition $H(\infty) = 1$. Letting NTF(z) be a rational function of the form:

$$\mathrm{NTF}(z) = \frac{b_m z^m + b_{m-1} z^{m-1} + \ldots + b_1 z + b_0}{a_n z^n + b_{n-1} z^{n-1} + \ldots + a_1 z + a_0} \tag{4.10}$$

Then it follows from $H(\infty) = 1$ that $m = n$ and $b_m = b_n = a_n$ must hold.

In conclusion, if a physically unrealizable delay-free loop is to be avoided in the modulator, the equivalent conditions [5, 8]:

$$h(0) = H(\infty) = \frac{b_n}{a_n} = 1 \tag{4.11}$$

on the NTF must hold. This seriously restricts the choice of NTF(z) available to the designer.

4.5.2 Loop Filter Architectures

The general model introduced in Fig. 4.10 is commonly used to design stable high-order modulators. If a chain of integrators is used to form the loop filter, two main modulators typologies exist [1, 5, 8]:

- *Chain of Integrators with Feedback paths (CIFB)*: a cascade of M-delaying integrators is used to form the filter, where the feedback and the input signals are being fed to each integrator input with different weight factors a_i and b_i (Fig. 4.11).

The transfer function of the signal filter, L_0, is:

$$L_0(z) = \sum_{i=1}^{N+1} \frac{b_i}{(z-1)^{N+1-i}} = \frac{b_1 + b_2(z-1) \ldots + b_{N+1}(z-1)^N}{(z-1)^N} \tag{4.12}$$

while the feedback filter L_1 can be derived as:

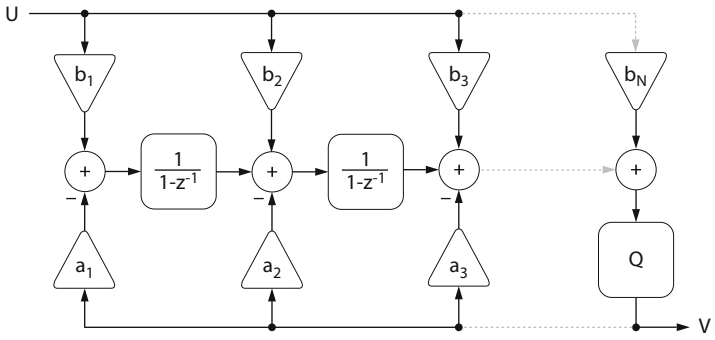


Fig. 4.11 General structure of a CIFB ΣΔ-modulator

$$L_1(z) = \sum_{i=1}^{N} \frac{-a_i}{(z-1)^{N+1-i}} = \frac{a_1 + a_2(z-1) \ldots + a_{N+1}(z-1)^{N-1}}{(z-1)^N} \tag{4.13}$$

where a_i and $b_i > 0$. The NTF results:

$$\mathrm{NTF}(z) = \frac{1}{1 - L_1(z)} = \frac{(z-1)^N}{D(z)} \tag{4.14}$$

with:

$$D(z) = a_1 + a_2(z-1) + \ldots + a_N(z-1)^{N-1} + (z-1)^N \tag{4.15}$$

Note that the NTF zeros must lie at $z = 1$ (i.e., DC). The weight factors a_i can be used to introduce finite non-zero poles into the NTF and to determine the zeros of $L_1(z)$. The a_i factors can be found by comparing (z) to the denominator of the desired NTF and equating the coefficients of like powers of $(z-1)$.

The STF results:

$$\mathrm{STF}(z) = \frac{L_0(z)}{1 - L_1(z)} = \frac{b_1 + b_2(z-1) + \ldots + b_{N+1}(z-1)^N}{D(z)} \tag{4.16}$$

Equation 4.16 shows that the b_i factors determine the zeros of the STF and the a_i its poles. The b_i values can be found by coefficient matching with the numerator of the specified STF. Again, note that the poles of the STF and the NTF are shared.

Important to note is that the a_i values must be non-zero to realize poles suitable for stable operation. To simplify the circuit, the zeros of the STF, b_i, can be set to zero except for b_1. In such case, all the STF zeros lie at infinity in the z-plane, and the STF is determined by $b_1/(z)$. Note that $|D(e^{j\omega})|$ should be flat in the signal band in order to make |STF| constant there.

The advantage of this architecture includes the fact that it results easy to implement as it provides low sensitivity to components variations (i.e., analog tolerances). Moreover, in the case of CT implementations, it exhibits a superior anti-aliasing characteristic compared to CIFF modulators [7]. The main disadvantage is that a scaled replica of the input signal is present at each integrator output, resulting in a large signal swing. This may facilitate saturation as both input and output signals are present. In DT implementations, large capacitors are needed to accommodate such swing and avoid saturation, resulting in higher power consumption. In CT implementations the output swing increases the op-amp requirements and, to avoid overload, small gain coefficients are needed for scaling the integrator output down. Small gain coefficients in CT modulators lead to integrators with low transconductance values, which contribute to increase the noise in the circuit [7]. Additionally, op-amp nonlinearities cause harmonic distortion that is a function of the input signal, limiting the achievable SQNR. Further, note that each feedback path requires a DAC, increasing costs, circuit complexity, area, and power consumption. For these reasons, CIFB modulators are generally discouraged for low-power applications [3, 4, 7].

- *Chain of Integrators with Feedforward paths (CIFF)*: one feedback and $L - 1$ feedforward paths which converge in an adder block placed before the quantizer (Fig. 4.12)

  The transfer function of the feedback filter is:

$$L_1(z) = -a_1 I(z) - a_2 I(z)^2 - \ldots - a_N I(z)^N \tag{4.17}$$

where $I(z)$ is the delaying integrator, $1/(z-1)$.

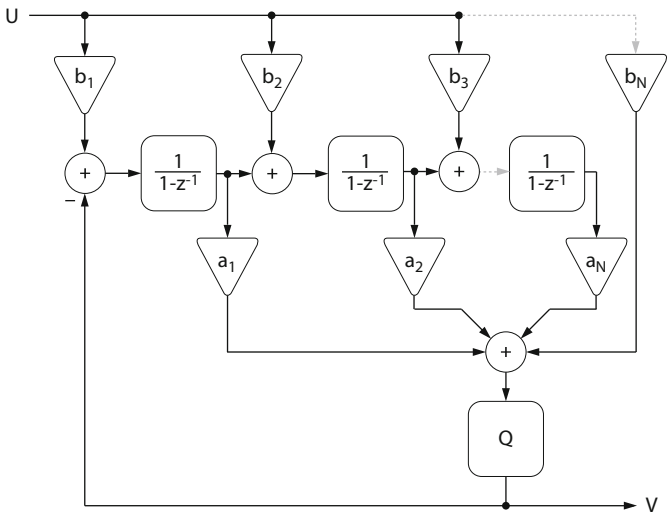


Fig. 4.12 General structure of a CIFF ΣΔ-modulator

The signal filter function is derived as:

$$L_0 = b_1 \cdot \left(a_1 I(z) + a_2 I(z)^2 + \ldots + a_N I(z)^N\right) + b_2 \cdot \left(a_2 I(z) + \ldots a_N I(z)^{N-1}\right) + \ldots + b_{N-1} \tag{4.18}$$

Note that all N-poles of $L_1(z)$ lie at $z = 1$ (DC) as well as all the zeros of $H(z)$.

A common design practice [10] is to set all b factors to zero while $b_1 = b_{N+1} = 1$, so that $L_0(z) = 1 - L_1(z)$, ensuring a unity STF(z). The input to the loop filter becomes:

$$U(z) - V(z) = U(z) - [U(z) + \text{NTF}(z)E(z)] = -\text{NTF}(z)E(z) \tag{4.19}$$

According to Eq. 4.19, by setting the architecture to provide STF(z) = 1,the loop filter does not process the input signal, and, as the quantization noise power is typically much smaller than the signal power, larger integrator coefficients are allowed without causing saturation of the integrator outputs. This allows the implementation of smaller, power-saving capacitors in the integrators. Additionally, not having to process the input signal results in reduced op-amp output swing (i.e., especially if a multi-bit quantizer is used) as well as avoiding the introduction of harmonic distortion into the output signal caused by integrators nonlinearities [7]. Another advantage that lowers the power dissipation is the fact that only one DAC is used. Therefore, CIFF filters are usually preferred [3, 4, 7] in applications where low-power consumption and low-signal distortion are required (e.g., audio, etc.). A drawback of CIFF modulators is the closed loop frequency response having a peak at high frequencies due to the zeros in the signal path. The amplification of the out-of-band frequencies caused by the high-frequency boost can overload the quantizer and drive the modulator into instability, especially in the case of input signals close to that frequency. In some cases, prefiltering of the input signal may be necessary [7]. Further, the NTF and STF are not independent, and the selection of the NTF sets the magnitude of the high-frequency boost in the STF. Another disadvantage is that a summing circuit is needed in front of the quantizer in order to add the loop filter output and the relative feedforward branches, increasing power consumption, area, and circuit complexity [7]. The operational speed of the op-amp used in the adder needs to be relatively high for an accurate summation, resulting in a challenging design. Two of the most common op-amp architectures that can be used to form the adder are the differential folded cascode and the differential two-stage amplifier. Note that, in the case of CT modulator designs, the latter is usually regarded as more efficient when considering resistive loads in terms of DC gain [7].

4.5.3 Optimization of the NTF Zeros

Purely differentiating NTFs (i.e., $(1 - z^{-1})^N$ which have all their zeros at $z = 1$ and all their poles at $z = 0$) obtained with a loop filter formed by a chain of integrators do not ensure optimal noise shaping [3], and significant improvement in the SQNR can be achieved by using more general functions. These are NTFs with separated zeros on the unit circle, spread over the signal range, and with complex poles located within the unit circle surrounding the signal band. Spreading the zeros reduces the total noise power in the signal band, while moving the poles nearer to the zeros reduces the out-of-band NTF gain, resulting in improved stability.

For a given ΣΔ-modulator of order L, the optimal NTF's zeros can be found by minimizing the normalized quantization noise power in the signal band [1], that is, by finding the minimum of the integral:

$$P_{\mathrm{N}} = \int_{-f_{\mathrm{b}}}^{f_{\mathrm{b}}} \frac{\Delta^2}{12 f_{\mathrm{s}} |\mathrm{NTF}(f)|^2}\, df \tag{4.20}$$

Providing the poles are chosen accordingly to stability constraints, the solution of the problem consists in finding:

$$\min \left[\int_{-f_{\mathrm{b}}}^{f_{\mathrm{b}}} |N(z)|^2_{\mathrm{e}^{j\omega}} df \right] \tag{4.21}$$

where $N(z)$ is the NTF numerator in the form of a polynomial of z of order L.

In a more descriptive argumentation, the principle of optimization is that the normalized noise power, given by the integral of the squared magnitude of the NTF over the signal band, is minimized with respect to the values of all its zeros. The optimal zeros are found by equating the partial derivatives of the integral to zero. Therefore, by solving Eq. 4.21 numerically, the frequency of the zeros which minimize the in-band noise and the SQNR improvement can be calculated [5].

Table 4.1 illustrates the resulting values for the zeros (i.e., normalized to the signal band limit ω_{b}) for NTFs with degrees from 1 to 5 as well as the respective improvement in SQNR. Note that the optimal NTF zeros are always located on the

Table 4.1 Location of the NTF zeros for optimal suppression of the quantization noise [1]

Order L of NTF	Freq. of the zeros normalized to f_{b}	Location of the zeros on the z-plane	SQNR improvement (dB)
1	0	1	0
2	±0.577	$0.994 \pm 0.113j$	3.5
3	0, ±0.775	$1, 0.988 \pm 0.152j$	8
4	±0.34, ±0.861	$0.998 \pm 0.067j$, $0.986 \pm 0.168j$	13
5	0, ±0.538, ±0.906	$1, 0.994 \pm 0.106j$, $0.987 \pm 0.177j$	18

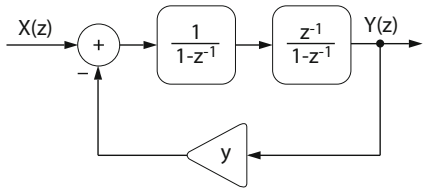


Fig. 4.13 Resonator for generation of complex-conjugate NTF zeros

unit circle (e.g., $|z_i = 1|$), so the damping factor is zero. However, as it can be seen, they are either at DC or complex-conjugate. This cause the NTF frequency response to have a notch at the frequencies of the zeros. Further, note that the optimization process giving these zeros assumed that the quantization noise is white and that the poles of the NTF have no significant effect on the in-band noise. If these conditions do not hold, or if the noise at different frequencies should be weighted differently as is the case, for example, of A-weighting of audio signals, then the optimization may still be performed by incorporating these factors into the optimization process in the form of a weight factor under the integral [10, 11]. Additionally, consider that for even-order NTFs, the optimized zeros do not provide perfect DC suppression [2] since there are no zeros at $z = 1$. Often, such DC zeros are desirable in order to ensure exact reproduction of the DC input level and also to reduce the probability of low-frequency tones. In such cases, it is possible to place a double zero at $z = 1$ and optimize the remaining zeros as before, so to minimize the mean square in-band gain of the NTF.

To realize the complex-conjugate NTF zeros in a ΣΔ-modulator a *weighted* local feedback path across two consecutive integrators of the loop filter must be implemented. This circuit arrangement is known as resonator (Fig. 4.13), with general transfer function [8]:

$$H_{\text{res}}(z) = \frac{Y(z)}{X(z)} = \frac{z^{-1}}{1 - (2 - \gamma)z^{-1} + z^{-1}} \tag{4.22}$$

where γ is the feedback coefficient.

Note that the resonators by themselves are unstable, as can be inferred by analyzing their pole locations [1]. However, when embedded in a stable feedback system (e.g., the modulator), local oscillations are prevented. The CIFB and CIFF architectures embedding resonators are typically referred as CRFB and CRFF architectures (i.e., Cascaded Resonators Feedback and Cascaded Resonators Feedforward, respectively).

4.5.4 Quantization Noise Coupling

The coupling of quantization noise, known also as noise-shaping enhancement, was introduced in a split architecture ΣΔ-ADC in [12]. It was shown there that a higher-order noise shaping is achieved with the stability condition of one orderless

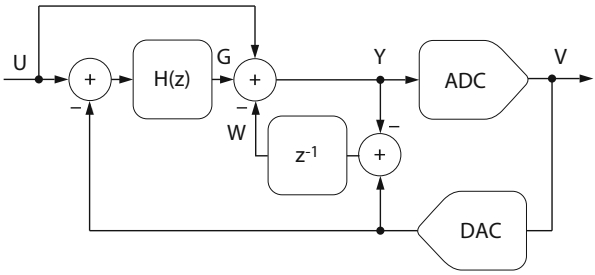


Fig. 4.14 Introducing shape quantization noise in the modulator using noise coupling

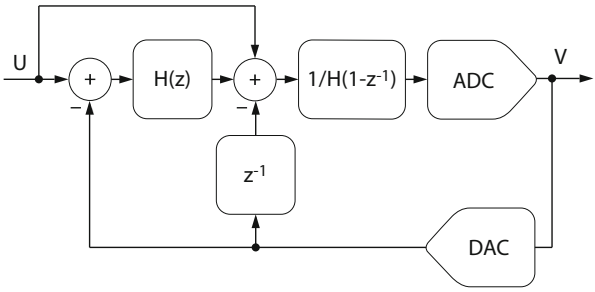


Fig. 4.15 Equivalent block level representation of shape quantization noise

structure. The idea of mutual noise coupling in split architecture ΣΔ-ADCs was extended to single stage ΣΔ-loops and split architecture with self-enhancement in [13]. The scheme proposed in [13] replaces the NTF to $(1 - z^{-1})$ times NTF, so to increase the effective order by one. According to [14], the shaped quantization noise $(1 - z^{-1}) \cdot Q(z)$ is obtained by the introduction of an extra branch, where the quantization noise $Q(z)$ is delayed by a clock cycle and subtracted by the summing node before the quantizer, as shown in Fig. 4.14.

Looking at the ΣΔ-modulator block diagram of Fig. 4.15, it can be seen how actually quantization noise coupling is achieved at a system level. It introduces a delaying integrating function in the signal path with an extra delayed output branch added to the summing node. The enhancement does not change the STF of the modulator, however, is equivalent to adding one more integrator to the loop filter. The technique is effective, but the use of an additional active adder at the quantizer input often limits the power reduction benefit, especially for low-order modulators.

Example

Analyze the block schematic of Fig. 4.14 relative to the quantization noise coupling technique. Determine expressions for the noise transfer function NTF (z) and signal transfer function STF(z) in order to verify that the modulator order is increased by one.

Solution

$$V = Y + E$$

So:

$$Y = U + G - W$$

Having a delaying integrating function, such as:

$$Y = U + \left[\frac{z^{-1}}{1 - z^{-1}}(U - V)\right] - \left[(V - Y)z^{-1}\right]$$

and noting that:

$$E = V - Y$$

Then:

$$Y = U + \left[\frac{z^{-1}}{1 - z^{-1}}(U - V)\right] - Ez^{-1}$$

$$Y = \frac{U(1 - z^{-1})}{(1 - z^{-1})} + \frac{Uz^{-1}}{(1 - z^{-1})} - \frac{Vz^{-1}}{(1 - z^{-1})} - \frac{Ez^{-1}(1 - z^{-1})}{(1 - z^{-1})}$$

Therefore:

$$V\left(1 - z^{-1}\right) = U\left(1 - z^{-1}\right) + Uz^{-1} - Vz^{-1} - Ez^{-1}\left(1 - z^{-1}\right) + E\left(1 - z^{-1}\right)$$

$$V - Vz^{-1} + Vz^{-1} = Uz^{-1} + U - Uz^{-1} - Ez^{-1} + Ez^{-2} + E - Ez^{-1}$$

$$V(z) = U + E\left(1 - z^{-1}\right)^2$$

$$\text{STF} = 1 \text{ and NTF} = \left(1 - z^{-1}\right)^2$$

The order is increased by one.

4.5.5 Conclusions

Figure 4.16 compares the three different cases presented for a generic third-order modulator: a purely differentiating NTF, a stabilized NTF, and a stabilized NTF with zero spreading [1, 5, 8].

As it can be seen, architectures with stabilized NTF (e.g., CIFB/CIFF) allow less quantization noise to be pushed out of baseband than purely differentiating NTF in

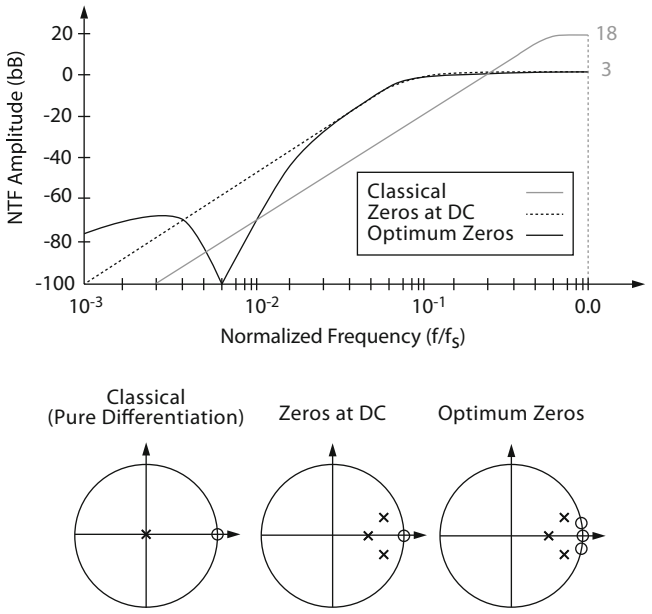


Fig. 4.16 Third-order NTF design comparison showing the effect of zero/poles mapping

order to meet the realizability conditions discussed, with the downsize to have higher noise in the passband. Architectures with stabilized NTF and zero spreading (e.g., CRFB/CRFF) allow stronger noise suppression in the higher signal band at the expense of lower suppression at very low frequencies (i.e., note the notch introduced by the complex-conjugate zeros on the unit circle). However, the total in-band noise power results smaller compared to that of a purely differential NTF, so the performance of the modulator results improved. Thus, zero spreading helps to alleviate the reduction of the overall performance of the converter caused by the stability arrangements which have to be taken to implement the modulator. Important to note is that the technique of quantization noise coupling can be applied to all the three cases presented.

All the architectures discussed are provided in the Simulink® Toolbox accompanying this book. The reader is highly encouraged to experiment with those by following the same simulation procedure seen in previous modulators.

4.6 Design Procedure

When determining the poles of the NTF, a number of constraints must be observed [1, 5, 8]:

- NTF(z) must satisfy the realizability condition $H(\infty) = 1$ (i.e., no delay-free loops).

- Being the out-of-band NTF gain and hence the stability of the whole modulator, largely determined by the choice of the poles (i.e., the zeros, as already explained, are constrained for efficient noise shaping to the signal band region), the location of those should be, at least, partially based on stability considerations.
- In optimizing the zeros, it is generally assumed that the influence of the poles on the magnitude of the NTF in the signal band is minimal. Also, note that for many modulator structures, the STF and NTF have the same poles. Therefore, the denominator of the NTF(z) should be flat over the signal band to avoid attenuation of the STF(z) in the baseband.

These constraints make the selection of poles a trade-off between conflicting design considerations, with no unique general solution. Fortunately, there are software tools available to perform this task for the designer (e.g., Schreier's MATLAB® Delta-Sigma Toolbox, discussed in the next paragraph). The typical "cookbook" design methodology used to design NTFs, and hence ΣΔ-modulators, consist of the following steps [1, 4, 11]:

1. Choose the modulator order accordingly to SQNR and OSR specifications. This task can be easily accomplished by referencing to Fig. A1 in appendix A. The curves in Fig. A1 illustrate the achievable peak SQNR for modulators of orders L from 1 to 8 employing optimal zero placement, with 1–3-bit quantization. The curves include the effects of the reduction of the input u necessary to satisfy the stability conditions. Hence, they accurately predict the actual performance of the real, nonlinear modulator.
2. Choose the NTF approximation type. Typical choices are Butterworth, inverse Chebyshev, and Elliptic (i.e., in this book only Butterworth approximation is presented).
3. Place the 3 dB cutoff frequency of the NTF, ω_{3dB}, slightly above the baseband edge.
4. Find the poles and zeros using an approximation software program (e.g., Schreier's MATLAB® Delta-Sigma Toolbox).
5. Predict stability (e.g., Lee's criterion).
6. Confirm stability with extensive simulations under all practical conditions (e.g., components mismatches, etc.) and worst-case inputs (e.g., Simulink® Toolbox presented in this book).
7. If the predicted stability is unsatisfactory, the poles need to be shifted away from the $z = -1$ point while maintaining the flat gain in the signal band. This can be achieved by reducing the cutoff frequency ω_{3dB}. As shown in [15], this reduces the peak NTF gain and hence enhances the stability of the modulator.
8. If the stability is robust, but the SQNR does not reach the limit values predictable from Fig. A1, it may be beneficial to make the design more aggressive by increasing ω_{3dB} and then repeating the stability tests. Steps 6 to 8 can be iterated until the optimum condition (e.g., maximum SQNR with adequate stability) is approached.

4.7 Schreier's Delta-Sigma Toolbox

The reader should download the freely available Schreier's Delta-Sigma Toolbox [1] from the link http://www.mathworks.co.uk/matlabcentral/fileexchange/19-delta-sigma-toolbox, and save it in the same folder of the Simulink® Sigma-Delta Toolbox which accompanies this book.

It is now possible to begin the second part of this chapter, which focuses on the actual design of high-order modulators.

It has been seen in the first part of this chapter that stable, high-order modulators can be implemented with a judicious choice of loop filter architecture. A natural question that thus arises is "*how can I choose/design a loop architecture for my high-order modulator?*" Traditionally, this has been something of a black art, requiring extensive experience in analyzing and trying many possible architectures to find the ones that work.

Luckily, much of the hard work can now be performed using the very popular and widely used Schreier's Toolbox, which allows users to implement a useful range of high-order loop types and report key performance parameters such as the stable input range, SNR, etc.

The Schreier's Toolbox is a utility which can be used to synthesize complex *L*th-order modulator structures using MATLAB® without the need of doing any mathematical analysis by hand! The toolbox has many capabilities, such as [1]:

- Discrete time modulator design
- Continuous time modulator design
- Quadrature modulator design
- Low-pass and band-pass modulators
- Supports six different loop filter types
- Synthesize coefficients for all these loop filters
- Simulate and analyze the modulator NTF synthesized

Although such Toolbox can be used to develop a complete high-level modulator design, often it results difficult for entry-level engineers or designers who are not familiar with the MATLAB® code language to easily and time-effectively complete a design task. Therefore, in this book, the Schreier's Toolbox is presented only as the tool to generate coefficients for low-pass modulators similar to the ones simulated in previous chapters. The functions needed to do a complete design and/or analyze other types of modulators will not be covered (e.g., band-pass modulators, Chebyshev Loop Filters, etc.), in order to keep the required MATLAB® coding language knowledge to the minimum. However, the interested reader can consult Chap. 8 and Appendix B, page 391 of [1], for a complete list of all Schreier's Toolbox functions.

Example

The proposed exercise aims to design a third-order CRFB modulator with a single-bit quantizer. According to Fig. A1 in Appendix A, such modulators would achieve a maximum SQNR of approximately 95 dB for an OSR of 64. Although the code to synthesize the NTF has been provided in the Simulink® Toolbox (e.g., MakeModulator.m), it is still necessary to enter the variables desired for a specific design. These variables will be used by the Schreier's Toolbox functions to synthesize the NTF coefficients.

Solution

Type – *load_par* – in the Matlab Command Window to set the default variables.
Type – *mex simulateDSM.c* – into the Matlab Command Window. This routine is required by the Schreier's Toolbox to function more efficiently.
Open – MakeModulator.m – and enter the following variables in the – Step [1] – portion of the code:

```
order = 3;
OSR = 64;
form = 'CRFB';
nLev = 2;
OBG = NaN;
opt = 1;
```

Save the code and type – MakeModulator – into the Matlab Command Window.

Once the code has been computed, the *a*, *g*, *b*, and c coefficients are displayed in the Matlab command window. These coefficients (three in each of the *a*, *b*, and *c* matrices and one in *g)* correspond directly to the multiplier gains of the *'mod3tb_CIFB_CRFB_1bit'* model found in the folder – *4_MOD3-> Schreier's Models – folder of the Simulink® Toolbox*. Further, the poles and zeros of the NTF designed as well as the STF and NTF transfer curves should be displayed. These plots are mainly used to understand if the NTF synthesized corresponds to what the designer aimed for.

The reader should now perform the simulations presented in previous chapters on the *'mod3tb_CIFB_CRFB_1bit' Simulink® model* in order to investigate the behavior of the system. As a brief example, Fig. 4.17 illustrates the output spectrum results for a − 3 dBFS sinewave input, and, as it can be seen, the SNDR corresponds to what predicted by the theory (e.g., ≈ 95 dB − 3 dB ≈ 92 dB), indicating that the modulator may function satisfactorily. Obviously, further simulation would be needed to ensure the correct operation of the system; however, should the modulator not meet some of the desired specifications (i.e., such as stability, input range, etc., which have not been defined for this introductory example), the designer should repeat the procedure while changing the variables entered in – MakeModulator – accordingly (e.g., by increasing the out-of-band gain, higher SQNR can be achieved, but the stability of the modulator would be compromised as a smaller input range would be allowed, etc.).

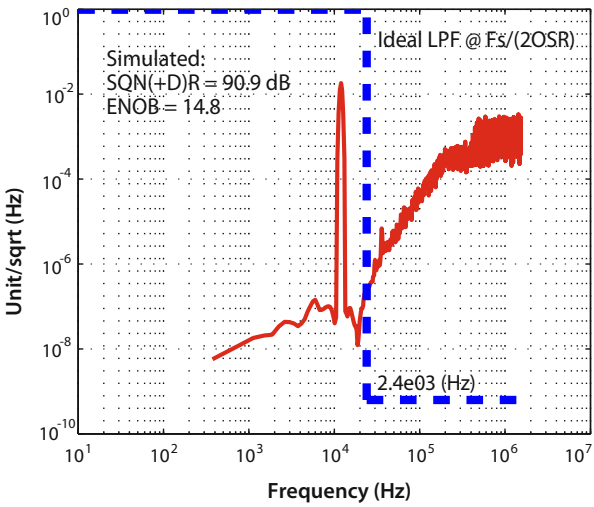


Fig. 4.17 Third-order CRFB output spectrum for a 12 KHz sinewave of amplitude −3 dBFS

4.8 Conclusions and Essential Takeaways

This chapter discussed the issue of stability in creating high-order modulators and introduced two approaches to synthesize stable designs with wide input range:

- Single-loop architectures with signal paths added to limit the maximum gain of the NTF
- Multi-bit quantizer to help stabilize the NTF and hence, the loop filter

Although single-loop designs [3] present good tolerance to errors (e.g., finite DC integrator gain, components tolerances, etc.), they can be difficult to stabilize due to having long chains of integrators in the NTF path. To obtain stability, various heuristic rules exist (e.g., Lee's criterion) for modifying the NTF via changing the loop typology/architecture, but extensive simulations are still required to ensure that, for example, exceeding the stable input range does not result in the loop locking up indefinitely (i.e., there are ways of detecting this and resetting the integrators, but these are beyond the scope of this book). Further, such stabilizing techniques often do not allow for an aggressive noise-shaping in the signal passband as the penalty for reducing the maximum NTF gain, hence, simulations to verify that the desired SQNR is achieved are still required.

Implementing multi-bit quantization has several advantages, including [6]:

- The reduction of the in-band noise due to the use of a smaller quantization step.
- Improved quantizer gain which allows more aggressive NTF designs with a lower passband noise and a wider stable input range

However, the problem of using multi-bit quantizers is that even the slight nonlinearity in the multi-bit DAC feedback paths can have disastrous consequences for the modulator performance [1], as investigated in the next chapter.

To conclude, a wide range of trade-offs exist when designing high-order modulators and given the empirical/heuristic nature of much ΣΔ-modulator theory, experience, and experiments still play a large role in choosing an appropriate architecture [4, 7, 9, 11]. However, with the advent of tools such as the Schreier Delta-Sigma Toolbox [1] to aid architecture selection, this is now a considerably easier task than in the past.

Exercises

Q.1
Discuss the stability criteria for first-, second-, and higher-order single-bit Sigma-Delta modulators.

Q.2
Calculate the achievable SQNR for high-order modulators up to the order of eight assuming the Lee's criterion is applied. Consider an OSR between 32 and 256 and 1, 2, 3, and 4 number of bits.

Q.3
List the most significant DT ΣΔ-modulator non-idealities and their effect on the system. If possible, group them by the circuit portion which they affect (i.e., a switched-capacitor implementation is assumed).

Q.4
Describe the advantages and disadvantages of CIFF and CIFB architectures.

Q.5
What are the advantages of using a multi-bit quantizer in a high-order Sigma-Delta modulator? Does it help improving the modulator stability? Is it always worth to increase the number of bits in the quantizer?

Q.6
Briefly explain the concept of NTF's zeros optimization, and, starting from the results of Q.2 (i.e., Lee's criterion applied), calculate the increase in gain achieved by using resonators in modulators of order between 2 and 5. Consider an OSR between 32 and 256 and 1, 2, 3, and 4 number of bits.

Q.7
Create a Simulink® model of the block schematic of Fig. 4.14 relative to the quantization noise coupling technique (e.g., MOD1 with noise coupling). Verify, through simulation, that the modulator order is increased by one.

Q.8

Determine expressions for the noise transfer function NTF(z) and signal transfer function STF(z) of the modulator in Fig. 4.18. What is its order? How can you categorize its architecture and what advantage does it have relative to the technique of quantization noise coupling?

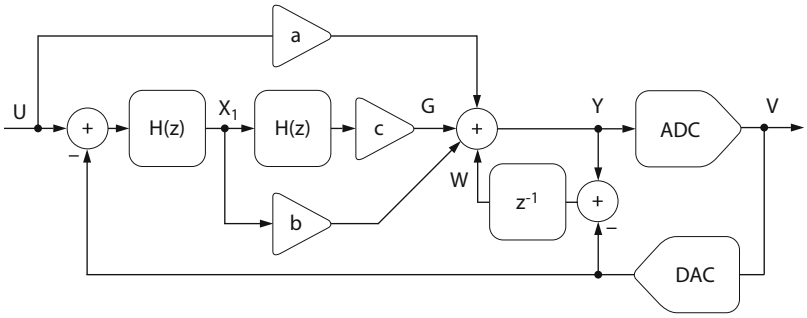


Fig. 4.18 ΣΔ-modulator with noise coupled injection technique

Q.9

Create a Simulink® model of the modulator analyzed in Q.7, and verify its SQNR/spectrum characteristics in order to prove what found in Q.7.

(Hint: depending on the simulation settings, note that the Simulink® model may need a gain block between the adder at the end of the loop filter and the quantizer – for example, 1/2 – to scale the input signal to the quantizer to a detectable level!)

Q.10

Synthesize the coefficients for an audio, third-order, 2-bit CRFF modulator with OSR = 64. Lee's criterion applied. Create a Simulink® model of the modulator and simulate the SQNR with an input sinewave at −3 dBFS. Is the SQNR found similar to what predicted by theoretical calculations?

References

1. Schreier R, Temes GC. Understanding Delta-Sigma data converters. Hoboken: Wiley; 2005.
2. Oppenheim AV, Schafer RW. Discrete-time signal processing. New Jersey: Prentice-Hall Inc.; 1989.
3. Reiss JD. Understanding Sigma-Delta modulation: the solved and unsolved Isses. JAES. 2008;56(1/2):49–64.
4. De la Rosa JM, Del Rio RF. CMOS Sigma-Delta converters: practical design guide. London: Wiley; 2013.
5. Candy JC, Temes GC. Oversampling Delta-Sigma data converters: theory, design and simulation. New York: Wiley-IEEE Press; 1991.
6. Geerts Y, Steyaert M, Sansen W. Design of multibit Delta Sigma A/D converters. New York: Kluwer Academic Publisher; 2002.

7. Casier H, Steyaert M, Van Roermund AHM. Analog circuit design: Robust design, Sigma Delta converters, RFID. London: Springer; 2011.
8. Norsworthy SR, Schreier R, Temes GC. Delta-Sigma data converters: theory, design and simulation. New York: Wiley-IEEE Press; 1996.
9. Medeiro F, Verdù BP, Vazquez AR. Top down design of high performance Sigma Delta modulators. New York: Kluwer Academic Publisher; 1999.
10. Pohlmann KC. Principles of digital audio. London: McGraw-Hill; 2008.
11. Bourdopoulos G, et al. Delta-Sigma modulators: modelling, design and applications. London: Imperial College Press; 2003.
12. Lee K, Temes GC. Enhanced split-architecture Delta-Sigma ADC. Electron Lett. 2006;42 (13):737–8.
13. Lee K, Bonu M, Temes GC. Noise-coupled Delta-Sigma ADCs. Electron Lett. 2006;42 (24):1381–2.
14. Lee K, Miller MR, Temes GC. An 8.1 mW, 82 dB Delta-Sigma ADC with 1.9 MHz BW and 98 dB THD. IEEE J Solid State Circuits. 2009;44(8):2202–11.
15. Schreier R. An empirical study of high-order single-bit Delta-Sigma modulators. IEEE Trans Circuits Syst II. 1993;40(8):461–6.



Multi-bit Quantizers

5

This chapter investigates the properties of multi-bit feedback DACs through a set of Simulink® simulations, both within the feedback loops of Sigma-Delta modulators and as stand-alone blocks.

The topics covered include:

- Comparison between the properties of the single-bit and multi-bit quantizers
- Demonstration of the need for excellent feedback DAC linearity through examination, via simulation, of the effects of nonlinearities in high-SNR multi-bit modulators caused by DAC element mismatches
- Demonstration of how DAC linearity in oversampled multi-bit systems can be achieved, in a time-average sense, by implementing dynamic element matching (DEM) techniques
- Examination of some basic properties of the DEM techniques presented

By the end of this chapter, the reader should have a practical feel regarding the significant problems caused by feedback DAC nonlinearity in analog multi-bit Sigma-Delta modulators and have an understanding of common methods used to address this key issue.

5.1 Single-Bit Versus Multi-bit Quantizer

The design implications of choosing a single-bit or multi-bit quantizer are very similar for both discrete- and continuous-time implementations of ΣΔ-modulators, and the major differences are summarized in Table 5.1 [1–3]. While single-bit quantizers allow for a simpler modulator design where cost, area, and power consumption are much smaller than multi-bit solutions, multi-bit quantizers offer many advantages, including [4, 5]:

© Springer International Publishing AG, part of Springer Nature 2019

I. Arnaldi, *Design of Sigma-Delta Converters in MATLAB®/Simulink®*,
https://doi.org/10.1007/978-3-319-91539-5_5

Table 5.1 Single-bit vs. multi-bit quantizer in ΣΔ-ADCs

	Single bit	Multi-bit
Achievable SNR	Low	High
Stability	Low	High
Power consumption	Low	High
Chip area	Low	High
Jitter sensitivity	High	Low
Slew rate required	High	Low
Linearity requirements	Low	High
Dither tolerance	Low	High
Circuit complexity	Low	High

- Improved modulator stability as the feedback loop linearity results increased since the effective quantizer gain variation with the signal is reduced, allowing to implement a more aggressive NTF. Hence, higher SQNR can be obtained and/or larger input signals can be allowed.
- For a fixed full scale, the SQNR is increased by 6 dB for every bit added to the resolution of the quantizer. This also relaxes the stop-band performance of the decimation filter used to suppress the out-of-band noise required.
- Increased tolerance to the use of dither as a greater input range is typically allowed.
- Relaxed slew rate requirements of the op-amps as the width of a single DAC step halves for each additional bit. This may also allow for a lower op-amp power dissipation.
- In CT modulators multi-bit implementations are more robust toward jitter, since smaller transitional changes and fewer transitions result in less amount of charge variation due to jitter (i.e., more on this in Chap. 7).

5.2 The Problem with Multi-bit Implementations

Chapter 4 briefly investigated how multi-bit quantization can improve the modulator stability and SQNR. Figure 5.1a illustrates the inherent linearity and undefined gain of single-bit quantizers versus the well-defined quantizer gain (i.e., since more than two points define linear approximation), and hence NTF, of the multi-bit case [1, 4]. As already discussed, for a fixed full scale the quantization error is reduced by 6 dB for every bit added to the resolution of the quantizer. This is because recalling that for a two-level mid-rise quantizer with $V_{LSB} = 2$ (i.e., to obtain ± 1 quantization thresholds) the quantization noise power is $\sigma_q^2 = 1/3$, it results that for an N-bit quantizer (i.e., $2^N - 1$ intervals between ± 1) the noise power corresponds to $\sigma_q^2 = \dfrac{1}{3 * 2^{2N}}$.

If multi-bit quantization is used in a modulator, then clearly the feedback has also to be multi-bit [6], and, in the case of an analog modulator, a DAC is needed to

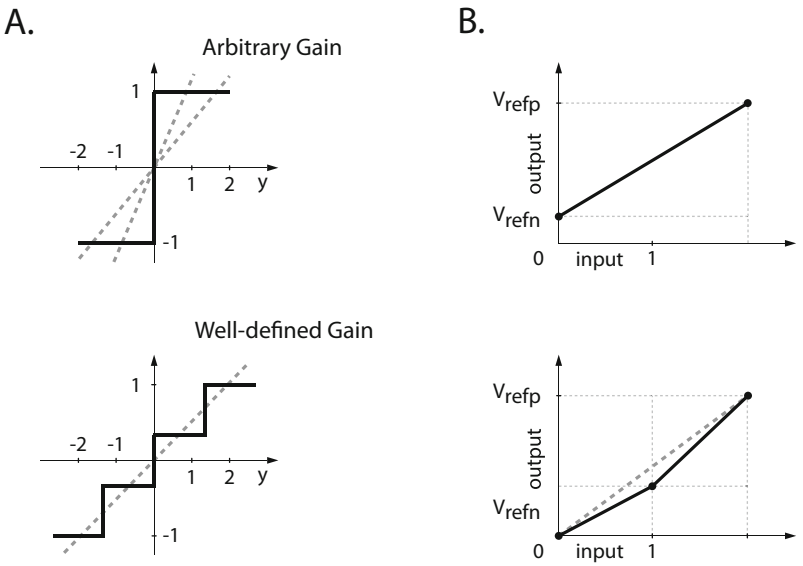


Fig. 5.1 (**a**) Single-bit vs. multi-bit quantizer and (**b**) respective DAC transfer functions

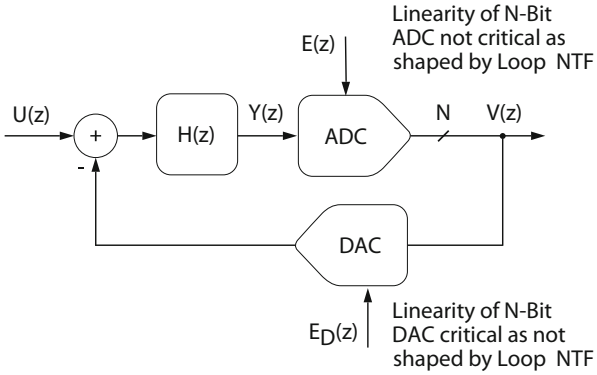


Fig. 5.2 Multi-bit Sigma-Delta block schematic

convert the digital quantizer output back to analog feedback for the integrators [7]. However, while single-bit DACs are inherently linear because they have only two output states and a straight line can always be fitted perfectly to two points, a linear multi-bit DAC circuit needs perfectly matched components, and real DACs are thus nonlinear (Fig. 5.1).

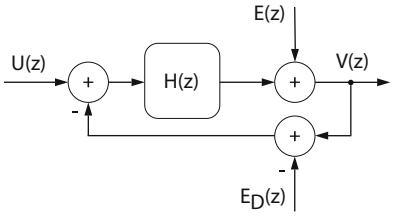
One major problem [1, 4, 7] with using a multi-bit feedback DAC is that the DAC nonlinearity is not shaped by the loop NTF (Fig. 5.2).

Thus, any error is added directly to the input signal, severely degrading the modulator SQNR performance and introducing harmonic distortion (i.e., quantizer nonlinearity, $E(z)$, is shaped by the NTF and is less critical).

Example

Analyzing the modulator in Fig. 5.2 demonstrates that the linearity of the N-bit DAC is not shaped by the loop filter.

Solution
Starting by drawing the linear model of the block schematic in Fig. 5.2:



Find the z-domain transfer function of the linear model, such as:

$$V(z) = \{U(z) - [V(z) - E_{\mathrm{D}}(z)]\} \cdot H(z) + E(z)$$

$$V(z) = U(z)H(z) - V(z)H(z) + E_{\mathrm{D}}(z)H(z) + E(z)$$

$$V(z)[1 + H(z)] = H(z) \cdot [U(z) + E_{\mathrm{D}}(z)] + E(z)$$

Expressing the equation found in terms of STF and NTF results:

$$\mathrm{STF} = \left.\frac{V(z)}{U(z)}\right|_{E(z)=0} = \frac{H(z)}{1 + H(z)}$$

$$\mathrm{NTF} = \left.\frac{V(z)}{E(z)}\right|_{U(z)=0} = \frac{1}{1 + H(z)}$$

$$V(z) = \mathrm{STF}(z)U(z) + \mathrm{STF}(z)E_{\mathrm{D}}(z) + \mathrm{NTF}(z)E(z)$$

As it can be seen from the equation, any ADC nonlinearity (i.e., $E(z)$) is affected by the loop filter (i.e., NTF(z)), while the DAC nonlinearity (i.e., $E_{\mathrm{D}}(z)$) is not shaped and directly affects the signal (i.e., STF(z)).

5.3 The Effects of DAC Mismatches in a Modulator

In this section, the consequences of employing a nonlinear DAC in an analog modulator *without* making any attempt to address the nonlinearity are investigated. The aim of the exercise is to demonstrate how the feedback DAC linearity is critical to achieving high SNR in a ΣΔ-modulator.

5.3.1 Quantizer and DAC Simulink® Models

In the Toolbox open the model *mod2_2bit_NO_DEM.mdl* found in the folder *5_MB_DAC*.

Make sure the *Manual Switch* in the feedback loop is in the *bottom* position (Real DAC).

The Simulink® model includes a standard MOD2 implemented with a two-bit quantizer. Important to note is that although in previous chapters the default Simulink® quantizer block has been used to achieve multi-bit quantization, in this model the quantizer and feedback DAC are implemented more like real circuits in order to allow direct examination of the effects of circuit-component errors on linearity and hence overall modulator behavior. On this regard, by double-clicking on the quantizer block to view the schematic details, it can be seen that a *flash-ADC-like* implementation of the quantizer compares the input values and generates a digital thermometer-coded output as summarized in Table 5.2:

The DAC elements to be used to generate a quantized analog output (e.g., voltage, current, charge, etc.) for the integrators are then selected according to the thermometer code produced by the quantizer, as reported in Table 5.3. Note that, by entering the DAC block in the Simulink® model (e.g., double-click on the block), it can be seen how the DAC converts the quantizer output back to a real value *DACout* using the – *dac* – MATLAB® structure (i.e., also inspect the – *load_par.m* – MATLAB® code in the folder *Routines*), which holds a representation of the element values (i.e., circuit components) as *dac.elements* (i.e., "ELE" in Table 5.3) and *dac.elecount* values (i.e., number of elements).

By default, all DAC elements have matched unit values (e.g., *dac.elements* = [1]), and so the DAC outputs are exactly 0, 1, 2, and 3. To verify the perfectly linear operation of the model:

Table 5.2 Multi-bit quantizer model: thermometer-code operation

Quantizer input	Thermometer output code
Input $\leq -2/3$	000 ("0")
$-2/3 <$ input ≤ 0	001 ("1")
$0 <$ input $\leq 2/3$	011 ("2")
$2/3 <$ input	111 ("3")

Table 5.3 Multi-bit DAC model: thermometer-code operation

Thermometer code	DAC output
000	0
001	ELE(0)
011	ELE(0) + ELE(1)
111	ELE(0) + ELE(1) + ELE(2)

Type – *load_par* – into the MATLAB Command Window to set initial conditions.

Type – *psdset.win = 'hann'* – into the MATLAB Command Window to set the window function for this exercise and run the simulation.

Once the simulation is complete, type – *mod_SNDR_ac* – into the MATLAB Command Window. Note that – *mod_SNDR_ac* – is just a version of – *mod_SNDR* – that first removes any DC signal component. This is needed because the output of the modulator has a range between 0 and 3 instead of ± 1, as a consequence of the thermometer-code operation.

Double-click on the *blue glasses* at the output of the quantizer and on the *scope* block to open the two time scopes of the model.

The reader should examine the time scopes output waveform to familiarize with the quantizer and DAC arrangements, noting how correct operation is achieved (e.g., confirm that if the ideal quantizer is cascaded into the ideal DAC and then the DAC output is multiplied by 2/3 and subtracted by 1, then the overall transfer function of 1 is obtained, that is, a line drawn through the midpoints of the steps and treads after scaling and offsetting which gives a transfer function – *output level = input level*). Regarding the spectral analysis, the second-order operation (i.e., 40 dB/dec slope) with an SNDR of about 78 dB should result, confirming the correct operation of the model.

5.3.2 Introducing DAC Mismatches

Having familiarized with the quantizer and DAC models introduced, the next task proposed is then to simulate the small random mismatches due to manufacturing errors (e.g., *dac.elements = [1.0112, 0.9973, 0.9934]*) that unavoidably would occur in a real DAC.

Add some elements mismatch to the DAC by typing – *dac.elements = [1.05, 1, 0.95]* – into the MATLAB Command Window. This represent DAC elements with 5%, 0%, and − 5% mismatch. Note that, in order to investigate the effects of DAC mismatches, an error much worse than that of typical tolerances achievable in modern integrated circuits has been introduced, so to emphasize the effects.

Run the simulation.

Type – *mod_SNDR_ac* – into the MATLAB Command Window and open the time scopes.

According to the time scope results, the reader should verify that the modulator and feedback DAC signals output range do not fit exactly between the ideal 0–3

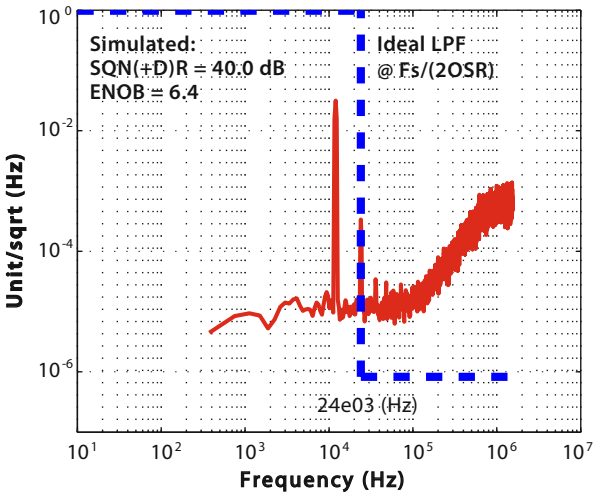


Fig. 5.3 MOD2 FFT with nonlinear DAC

range (e.g., level 2 is always at 2.05 instead of exactly 2, etc.), thus, confirming the presence of the inaccuracies introduced. Consequently, as it can be seen in the FFT of Fig. 5.3, a dramatic degradation in SNDR from the 78 dB of the ideal feedback DAC case to the 40 dB of the non-ideal case results (i.e., approximately 40 dB difference!). Important to note is that each input thermometer code to the DAC always selects the same DAC elements to form the DAC output, hence always generating the same error for a given thermometer code input to the DAC.

The poor performance achieved is the result of the increased noise floor at approximately 10^{-5} instead of the 10^{-9} of the ideal case, caused by the out-of-band quantization noise inter-modulating via the DAC nonlinearity [1, 3, 6]. Moreover, due to signals intermodulation [4, 8], out-of-band signals generate much more components/tones in the signal band when compared to the ideal case (Fig. 5.4), clearly highlighting the need for excellent feedback DAC linearity in high-SNR converters to achieve satisfactory performance.

Having experienced the mismatch effects of the feedback DAC in the modulator, it is instructional at this point to simulate a more realistic situation where mismatches account only for $\pm 0.25\%$. This may be considered a fairly representative matching level for modern integrated circuit components (i.e., some processes may be a bit better, some a bit worse).

Add some elements mismatch to the DAC by typing – *dac.elements = [1.0025, 1, 0.9975]* – into the MATLAB Command Window and run the simulation.

Type – *mod_SNDR_ac* – into the MATLAB Command Window and open the time scopes.

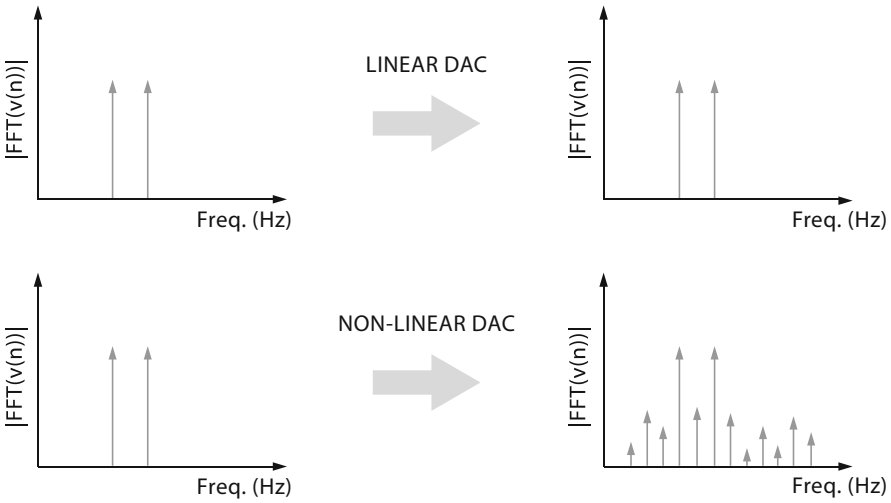


Fig. 5.4 Effects of DAC nonlinearity

Fig. 5.5 Effects of DAC nonlinearity with ±0.25% mismatch

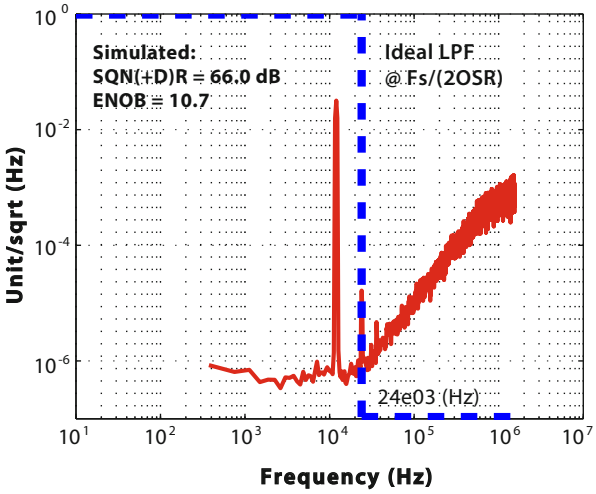


Figure 5.5 illustrates the FFT. As it can be seen, despite the relatively small mismatch values used, the presence of tones and the increase in noise floor level caused by the DAC nonlinearity due to signals intermodulation still dramatically affects the modulator performance. Therefore, achieving the theoretical expected SNDR of ≈80 dB (i.e., Table A.1, Appendix A) is clearly very difficult in practice, and methods to counteract the DAC nonlinearity should be employed to achieve satisfactory performance of multi-bit modulators.

5.4 Random Element Selection

In the previous section, it has been shown how the feedback DAC nonlinearity can distort not only the input signal when seen at the modulator output (i.e., causing signal harmonics to appear) but also the quantization noise itself, leading to the high out-of-band quantization noise inter-modulating with itself and appearing in the signal band as a white noise. As discussed, even for typical matching tolerances achievable in modern IC devices, it is difficult, if not impossible, to design high-performance converters (i.e., assuming very expensive device trimming methods are not to be implemented or additional calibration techniques and circuits devised). Thus, methods for dealing with the DAC nonlinearity are essential in order to take advantage of multi-bit quantization in ΣΔ-modulator.

Recalling that for the quantizer and DAC models introduced in Sect. 5.3 a given quantizer output code always selected the same DAC elements to create the DAC outputs (e.g., for an input code 011 ("2"), the analog output would always be generated by using the first two DAC elements with values 1.05 and 1, giving 1.05 + 1 = 2.05, etc.), it results that the error is directly correlated to the DAC code.

In order to de-correlate the mismatch error from the DAC code, hence avoiding the introduction of harmonics due to signal intermodulation, it is sufficient to select the elements which have to be used to represent each code at random – e.g., for the input code (*2*) the element combinations (*1.05 and 1*), (*1.05 and 0.95*), *and* (*1 and 0.95*) are used at random with equal probability to create the *DACoutput*. Therefore, all elements are used equally frequently in generating the *DACoutput* for code (*2*), resulting in the average of the equally used code pairs to be 2. In other words, the mismatch error is set to zero every time all elements are used once.

In the Toolbox open the model *mod2_2bit_RES.mdl* found in the folder *5_MB_DAC*. Set the *Manual Switch* in the feedback loop in the *bottom* position (real DAC), and type – *load_par* – into the MATLAB Command Window.

Type – *psdset.win = 'hann'* – into the MATLAB Command Window to set the window function for this exercise and run the simulation.

Add some elements mismatch to the DAC by typing – *dac.elements = [1.05, 1, 0.95]* – into the MATLAB Command Window.

Run the simulation and confirm that similar results to that of Sect. 5.3 are achieved since no randomization has been applied yet.

Type – *dac.dem_enable = 1* – into the MATLAB Command Window to enable the random element selection DEM.

Rerun the simulation, type – *mod_SNDR_ac* – into the MATLAB Command Window, and open the time scopes.

Figure 5.6 reports the time scope plot of the DAC output when the random element selection (RES) DEM is enabled, and, as it can be seen, the output does

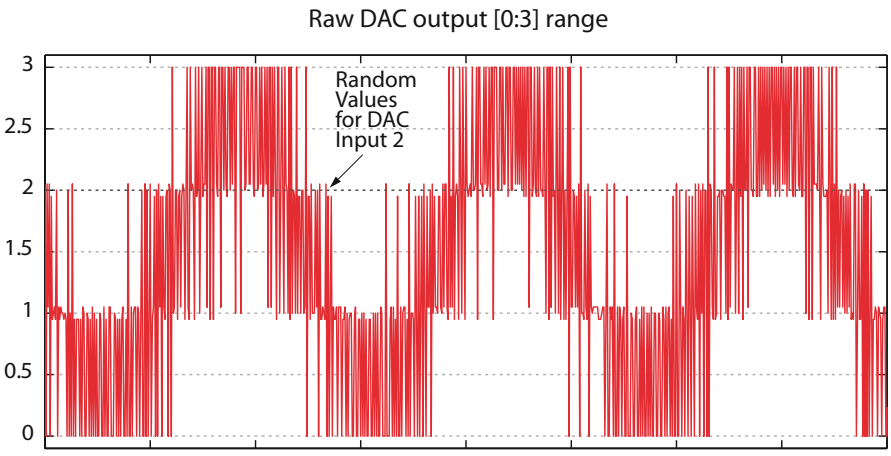


Fig. 5.6 Time scope result illustrating the implementation of the RES-DEM

not always have the same value for a given DAC input code (e.g., 011 ("2")), confirming the expected operation of the RES algorithm.

Comparing the FFT results of Fig. 5.7 (top) to the case with no-DEM enabled (Fig. 5.3), it is seen that the RES algorithm does indeed linearize the feedback DAC, as evidenced by the fact that the signal harmonics vanish. Consequently, the SNDR slightly increases, although the improvement is quite unsatisfactory considering the aim to achieve a high-performance second-order modulator with a theoretical SNR of roughly 80 dB. The main reason for the poor SNDR performance achieved is that the passband white noise previously generated by the out-of-band quantization noise inter-modulating via the DAC nonlinearity is now replaced with a different white noise caused by the randomized DAC errors having a white noise spectrum. Therefore, the randomization of the elements selection of the DAC alone is typically not sufficient to address the DAC nonlinearity; it does eliminate nonlinearity but adds a broadband white noise too [1, 3–5].

Example

Consider a feedback DAC in a ΣΔ-modulator formed by unit elements with values $E[0:7] = [0.93,\ 1.05,\ 1.01,\ 0.94,\ 0.99,\ 1.06,\ 1.07,\ 0.95]$. The DAC output is to be generated from these elements according to the formula $\mathrm{DACout}(n) = \sum_{i=0}^{V(n)-1} E(i)$. Assuming the DAC receives an input sequence $V(n) = [2, 7, 1, 0, 8, 2, 4]$, calculate the DAC output sequence DACout(n) in the case a RES scheme is employed.

Solution

Many solutions as selection of elements is at random!

Input $V(n)$	Elements selected	DACout(n)
2	DACout(0) = 1.01 + 1.07	= 2.08
7	DACout(1) = 0.93 + 1.05 + 1.01 + 0.99 + 1.06 + 1.07 + 0.95	= 7.06
1	DACout(2) = 1.07	= 1.07
0	DACout(3) = 0	= 0
8	DACout (4) = 0.93 + 1.05 + 1.01 + 0.94 + 0.99 + 1.06 + 1.07 + 0.95	= 8.0
2	DACout(5) = 0.93 + 1.06	= 1.99
4	DACout(6) = 0.94 + 1.06 + 1.07 + 1.05	= 4.12

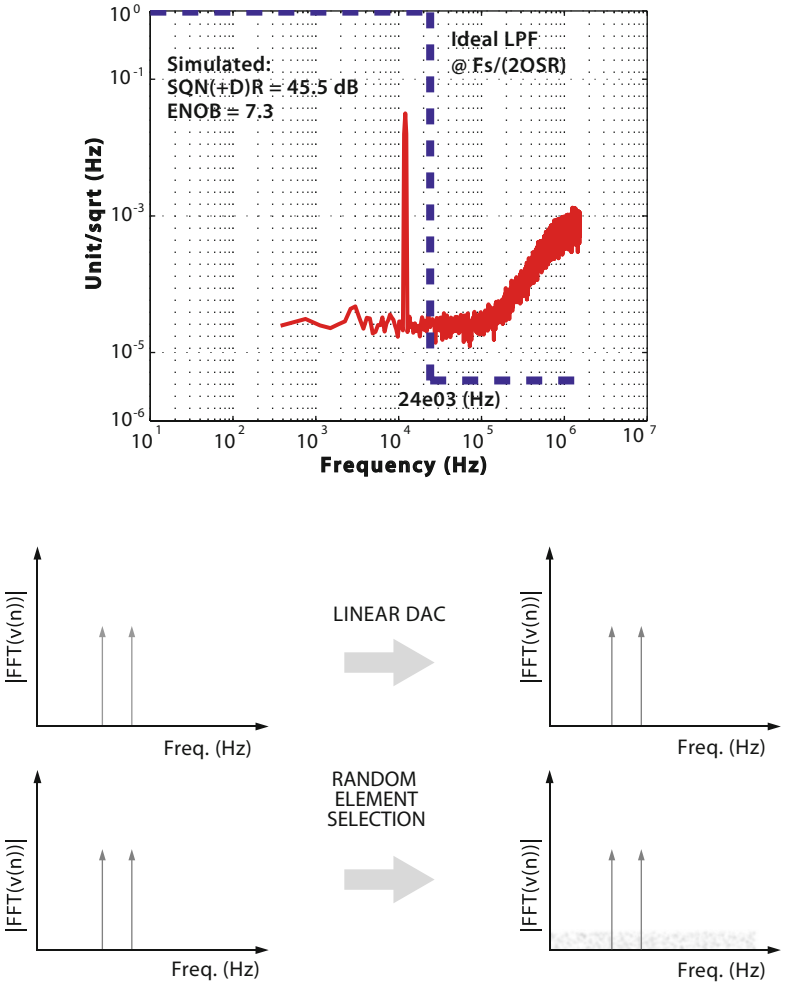


Fig. 5.7 (Top) Modulator output spectrum with RES-DEM enabled and (bottom) theoretical representation of the RES-DEM effects on the spectrum

5.5 Alternative DEMs Theory: DWA and ILA

In this section some simple alternative DEM algorithms are presented as these not only linearize the feedback DAC in a time-average sense by swapping the DAC elements around but swap the elements in such a way that the noise spectrum resulting from the elements swapping is not white but first-order noise shaped, so that it is suppressed in the signal passband. Important to note is that the DAC noise being shaped is the one resulting from the mismatch error introduced by the uneven spacing of the DAC levels rather than the quantization error. Although a multitude of different DEM algorithms exists in order to achieve noise shaping of the DAC noise (e.g., tree-structure, vector-based mismatch shaping, etc.), each having their advantages and disadvantages, in this book only two of the most basic and common strategies are presented [1, 3–5], namely, data weighting averaging (DWA) and individual level averaging (ILA). Both of these techniques utilize the unit element DAC structure already introduced in this chapter and figuratively illustrated in Fig. 5.8, but each uses a different strategy for the selection of the unit element for a given digital input code.

Important to note is that these DEM techniques assume the offset and gain errors of the DAC are acceptable and only the DAC nonlinearity error is of concern. Such assumption is typically valid in practical applications, and it is necessary to make since no element selection method can reduce the offset – OS – (i.e., DAC output at input code *0*, where no unit element is used) or affect the full-scale value – FS – when all elements are used. Thus, the effective gain (FS − OS)/*V* (i.e., where *V* is the full-scale digital input to the DAC) cannot be changed by any element selection logic either. Therefore, it should be assumed that these values are the correct ones and hence that the "ideal" *M*-element DAC input-output characteristic is described by the linear function:

$$w(k) = \mathrm{OS} + \frac{\mathrm{FS} - \mathrm{OS}}{M} k \qquad k = 0, 1, 2, \ldots, V \tag{5.1}$$

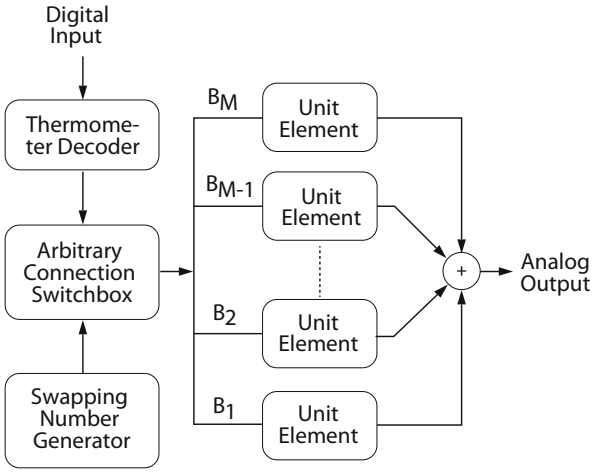


Fig. 5.8 Parallel unit element DAC

5.5.1 Data Weighting Averaging

Recalling what mentioned in Sect. 5.4, which is that every time all DAC elements are used once the mismatch error is set to zero, it is easy to appreciate the importance of using all elements as often and as equally as possible. This principle is at the basis of the data weighting averaging [1, 3–5] algorithm strategy, which aims to obtain the long-term average use of each unit element in the DAC the same by rotating the pattern of unit elements. As an example, consider a DAC with eight unit elements, and visualize them arranged in a circle as illustrated in Fig. 5.9.

Letting the first input code be 3, the elements that will be employed are 1, 2, and 3. If the following code is 4, it results that the succeeding four elements (4, 5, 6, and 7) will be selected. Allowing the subsequent code to be 6, note that a wrap around the circle is obtained, utilizing the elements 8, 1, 2, 3, 4, and 5. In other words, the number of DAC elements corresponding to the input digital code is implemented starting from the element following the last one used from the previous input code. Therefore, all components are used sequentially and as often as possible.

Similarly to MOD1, a disadvantage of the DWA algorithm is the occurrence of tone generation resulting whenever the DAC input is not a busy, random signal but a DC or a low-frequency periodic one. For example, by assuming the input to the eight-element DAC of Fig. 5.9 is a DC digital signal with a value of 6, it results that the element sets employed will be $S_1 = (1, 2, 3, 4, 5, 6)$, $S_2 = (7, 8, 1, 2, 3, 4)$, $S_3 = (5, 6, 7, 8, 1, 2)$, and $S_4 = (3, 4, 5, 6, 7, 8)$. Considering the subsequent cycle, it occurs that $S_5 = S_1$, $S_6 = S_2$, *etc*. Thus, the error $e(n)$ will be a periodic signal with a line spectrum, not a high-pass filtered noise. The tonal character of $e(n)$ can be changed by introducing a random, or pseudorandom, effect (e.g., random element skipping, bi-directional rotation, dither, etc.) into the process. Important to note is that the DWA-generated tones are in addition to limit cycles tones generated from the $\Sigma\Delta$-modulator and they may interact with each other.

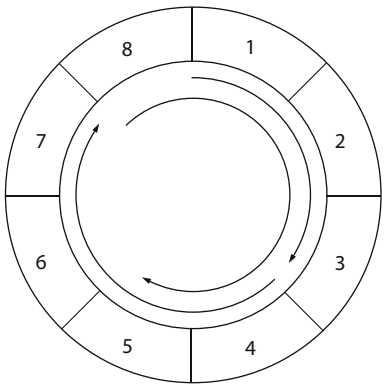


Fig. 5.9 Element rotation

Example

Consider a feedback DAC in a ΣΔ-modulator formed by unit elements with values $E[0:7] = [0.93,\ 1.05,\ 1.01,\ 0.94,\ 0.99,\ 1.06,\ 1.07,\ 0.95]$. The DAC output is to be generated from these elements according to the formula $\mathrm{DACout}(n) = \sum_{i=0}^{V(n)-1} E(i)$. Assuming the DAC receives an input sequence $V(n) = [2, 7, 1, 0, 8, 2, 4]$, calculate the DAC output sequence DACout(n) in the case a DWA scheme is employed.

Solution

Input $V(n)$	Elements selected	DACout(n)
2	DACout(0) = 0.93 + 1.05	= 1.98
7	DACout(1) = 1.01 + 0.94 + 0.99 + 1.06 + 1.07 + 0.95 + 0.93	= 6.95
1	DACout(2) = 1.05	= 1.05
0	DACout(3) = 0	= 0
8	DACout(4) = 1.01 + 0.94 + 0.99 + 1.06 + 1.07 + 0.95 + 0.93 + 1.05	= 8.0
2	DACout(5) = 1.01 + 0.94	= 1.95
4	DACout(6) = 0.99 + 1.06 + 1.07 + 0.95	= 4.07

5.5.2 Individual Level Averaging

Individual level averaging [1, 3–5] is another technique for introducing a first-order high-pass filtering of the DAC-generated mismatch noise. Instead of attempting to equalize the use of all elements for all codes over a long time period, ILA tries to equalize their usage separately for each input code over time. The simplest way to implement this technique is to use the DWA algorithm for each code – for a given input code k, the elements used by k places each time the code occurs are rotated and wrapped around after the *Mth* element has been used. Since the average error of all selections for a given code is zero, cycling through all possible selections sequentially results in the elimination of the average error (e.g., DC error) of the DAC.

Compared to the DWA method, ILA converges more slowly to the zero average condition, which reduces the effectiveness of the technique, especially for large M-sequences. On the other hand, it is less affected by any correlation between the mismatch shaping and the DAC input signal and is less likely to generate tones even for DC or periodic inputs.

Example

Consider a feedback DAC in a ΣΔ-modulator formed by unit elements with values $E[0:7] = [0.93,\ 1.05,\ 1.01,\ 0.94,\ 0.99,\ 1.06,\ 1.07,\ 0.95]$. The DAC output is to be generated from these elements according to the formula $\mathrm{DACout}(n) = \sum_{i=0}^{V(n)-1} E(i)$. Assuming the DAC receives an input sequence V

$(n) = [2, 7, 1, 0, 8, 2, 4]$, calculate the DAC output sequence DACout(n) in the case an ILA scheme is employed.

Solution

Input $V(n)$	Elements selected	DACout(n)
2	DACout(0) = 0.93 + 1.05	= 1.98
7	DACout(1) = 0.93 + 1.05 + 1.01 + 0.94 + 0.99 + 1.06 + 1.07	= 7.05
1	DACout(2) = 0.93	= 0.93
0	DACout(3) = 0	= 0
8	DACout(4) = 0.93 + 1.05 + 1.01 + 0.94 + 0.99 + 1.06 + 1.07 + 0.95	= 8.0
2	DACout(5) = 1.01 + 0.94	= 1.95
4	DACout(6) = 0.93 + 1.05 + 1.01 + 0.94	= 3.93

5.6 Alternative DEMs Simulations: DWA and ILA

The Simulink® model utilized in this paragraph contains a switch selecting four different input sources through the variable *dem_select*. The input sources are DC, arbitrary repeating sequence, and sinusoidal and uniform random number and will be applied to the three DEM algorithms discussed: RES, DWA, and ILA.

The RES, DWA, and ILA algorithms are coded as embedded MATLAB® functions and the source code can be viewed by double-clicking on the relevant block.

In the Toolbox open the model *testbench_DEM.mdl* found in the folder *5_MB_DAC*, and type – *load_par* – into the MATLAB Command Window to reset default parameters.

Type – *dac.dem_select = 4* – into the MATLAB Command Window to select the random noise input source.

Add some elements mismatch to the DAC by typing – *dac.elements = [1.05, 1, 0.95]* – into the MATLAB Command Window.

Run the simulation (i.e., noting that the DEM algorithms are not enabled, the noise spectrum appearing for each DEM should be flat since random noise is used as input and identical DACs are used for each algorithm. This is just a check to ensure correct initial settings).

Type – *DEM_SNDR_ac* – into the MATLAB Command Window and open the time scopes.

Type – *dac.dem_enable = 1* – into the MATLAB Command Window to enable the DEMs and rerun the simulation.

Similar results to that illustrated in Fig. 5.10 should be obtained, and, as it can be seen, these are in accordance with the theory discussed. Specifically, the RES algorithm has a white noise spectrum, while both DWA and ILA give first-order noise shaping of the DAC output noise, as a consequence of the scrambling operations. Note that, in this case, DWA achieved the lowest noise because it reuses all the elements the fastest and hence has the highest effective oversampling ratio for noise shaping.

As mentioned, one problem with all these simple DEM algorithms is that, like MOD1, they can generate additional tones when fed with periodic inputs, as this can lead to the same DAC elements being selected repeatedly. This can sometimes result in signal band tones either directly or via intermodulation with tones from the modulator (i.e., limit cycles and DC tones). Therefore, it is instructive to investigate such phenomena.

Select the DC input source by typing – *dac.dem_select = 1* – into the MATLAB Command Window.

Set the DC input value by typing – *dac.dem_dcin = 2* – into the MATLAB Command Window and run the simulation.

Type – *DEM_SNDR_ac* – into the MATLAB Command Window and open the time scopes.

Repeat the exercise with – *dac.dem_dcin = 0, 1, and 3.*

The results of this simulation should conclude that for DC input values of 1 and 2, the FFT of the sequence of DAC output errors $e(n)$ – *mismatch noise* – generated during the conversion comprises a single spectral line due to the repeating patterns that can be seen in the *Time Scope2*. For DC inputs 0 and 3 such repeating pattern does not occur since either no elements are used or all of them are (i.e., the DAC has three unit elements in this specific exercise). Therefore, as expected, every time all elements are used, the sum of all errors is zero (i.e., providing undetectable output spectrum due to the FFT parameter used). Further, it is instructive to note that for any N-clock periods the sum of all errors $|S(N)|$ is at most equal to $P/2$, where P is the sum of the absolute values of the errors in the three unit elements of the DAC. Note that P is a finite, and typically relatively small, number. Thus, the average value of the error samples $e(n)$ after N-clock periods satisfies:

$$|e(n)|_{\text{Average}} = |S(N)|/N \leq P/2N) \rightarrow 0 \quad \text{as} \quad N \rightarrow \infty \tag{5.3}$$

This analysis further highlights that the long-term average value of $e(n)$ is zero, which is highly desirable.

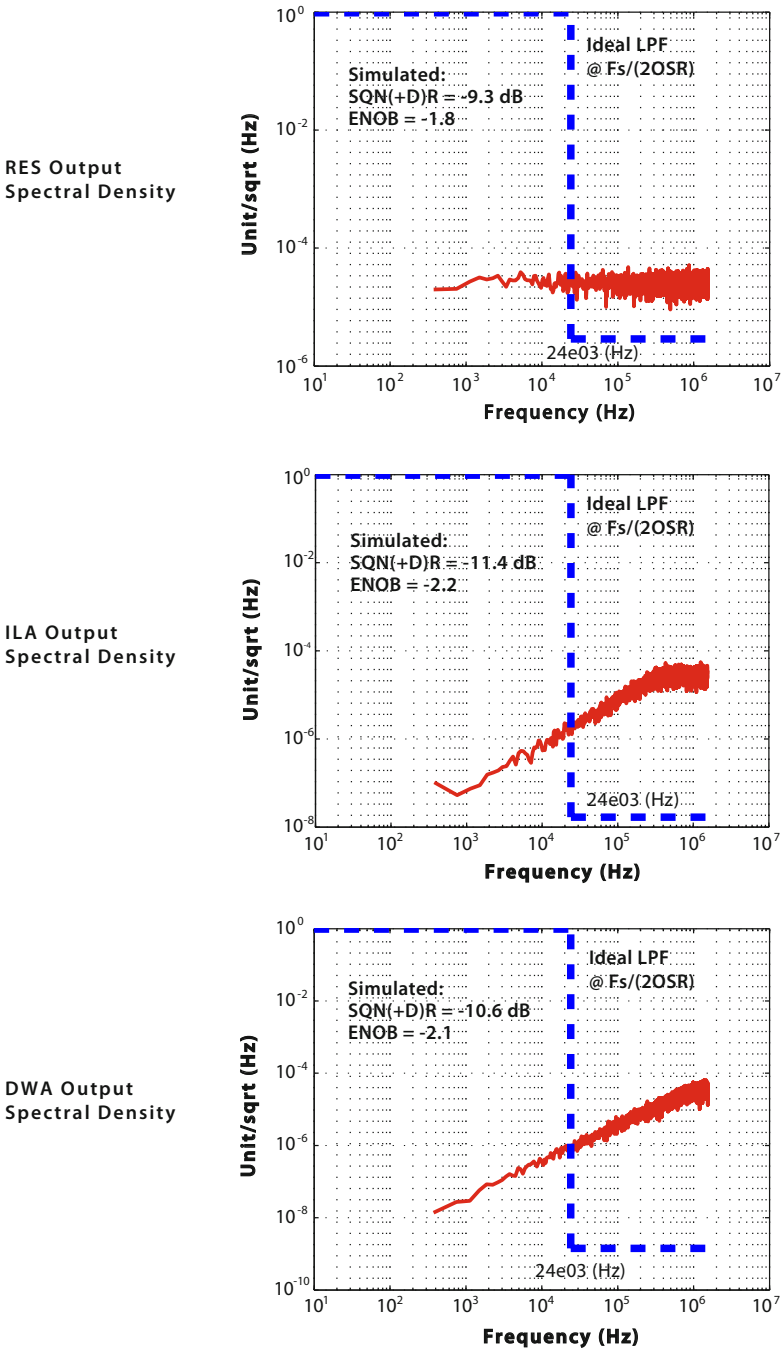


Fig. 5.10 (Top) FFT of RES, (middle) ILA, and (bottom) DWA DEM techniques

Example

Considering the exercises in Sect. 5.5, calculate the DAC output error sequence $e(n) = \text{DACout}(n) - V(n)$ for the RES, DWA, and ILA DEM schemes analyzed. Find the average error for each of the DEM schemes, and discuss the results.

Solution

RES			
DACout(0) = 1.01 + 1.07	= 2.08	$e(0) = 2.08–2$	= 0.08
DACout(1) = 0.93 + 1.05 + 1.01 + 0.99 + 1.06 + 1.07 + 0.95	= 7.06	$e(1) = 7.06–7$	= 0.06
DACout(2) = 1.07	= 1.07	$e(2) = 1.07–1$	= 0.07
DACout(3) = 0	= 0	$e(3) = 0$	= 0
DACout (4) = 0.93 + 1.05 + 1.01 + 0.94 + 0.99 + 1.06 + 1.07 + 0.95	= 8.0	$e(4) = 8.0–8$	= 0
DACout(5) = 0.93 + 1.06	= 1.99	$e(5) = 1.99–2$	= −0.01
DACout(6) = 0.94 + 1.06 + 1.07 + 1.05	= 4.12	$e(6) = 4.12–4$	= 0.12
DWA			
DACout(0) = 0.93 + 1.05	= 1.98	$e(0) = 1.98–2$	= −0.02
DACout(1) = 1.01 + 0.94 + 0.99 + 1.06 + 1.07 + 0.95 + 0.93	= 6.95	$e(1) = 6.95–7$	= −0.05
DACout(2) = 1.05	= 1.05	$e(2) = 1.05–1$	= 0.05
DACout(3) = 0	= 0	$e(3) = 0$	= 0
DACout (4) = 1.01 + 0.94 + 0.99 + 1.06 + 1.07 + 0.95 + 0.93 + 1.05	= 8.0	$e(4) = 8.0–8$	= 0
DACout(5) = 1.01 + 0.94	= 1.95	$e(5) = 1.95–2$	= −0.05
DACout(6) = 0.99 + 1.06 + 1.07 + 0.95	= 4.07	$e(6) = 4.12–4$	= 0.07
ILA			
DACout(0) = 0.93 + 1.05	= 1.98	$e(0) = 1.98–2$	= −0.02
DACout(1) = 0.93 + 1.05 + 1.01 + 0.94 + 0.99 + 1.06 + 1.07	= 7.05	$e(1) = 7.05–7$	= 0.05
DACout(2) = 0.93	= 0.93	$e(2) = 0.93–1$	= −0.07
DACout(3) = 0	= 0	$e(3) = 0$	= 0
DACout (4) = 0.93 + 1.05 + 1.01 + 0.94 + 0.99 + 1.06 + 1.07 + 0.95	= 8.0	$e(4) = 8.0–8$	= 0
DACout(5) = 1.01 + 0.94	= 1.95	$e(5) = 1.95–2$	= −0.05
DACout(6) = 0.93 + 1.05 + 1.01 + 0.94	= 3.93	$e(6) = 3.93–4$	= −0.07

Calculating the average error $E = \dfrac{\sum_{n=0}^{n=6} e(n)}{7}$ for each of the DEM schemes considered, it results:

$$\text{RES} \rightarrow E = \frac{0.08 + 0.06 + 0.07 + 0 + 0 - 0.01 + 0.12}{7} = 0.045$$

$$\text{DWA} \rightarrow E = \frac{-0.02 - 0.05 + 0.05 + 0 + 0 - 0.05 + 0.07}{7} = 0$$

$$\text{ILA} \rightarrow E = \frac{-0.02 + 0.05 - 0.07 + 0 + 0 - 0.05 - 0.07}{7} = 0.0022$$

The highest error is obtained using random selection, followed by ILA and DWA algorithms, respectively. The DWA algorithm achieves zero average error due to the sum of input codes being 24, which corresponds to an exact multiple of the number of the DAC elements. Therefore, all elements are used exactly three times each, meaning that the total error sums to zero. The RES algorithm is the slowest method to average the error, while ILA performs relatively well but takes more time to equally employ all elements than DWA.

Interesting to investigate is also the fact that even more exotic tones can be generated for periodic inputs to the DEM algorithms, such as modulator limit cycles.

Select the repeating input code sequence by typing – *dac.dem_select = 2* – into the MATLAB Command Window.

Run the simulation, type – *DEM_SNDR_ac* – into the MATLAB Command Window, and open the time scopes.

The FFT results should illustrate the presence of three tones instead of the single one seen in the previous exercise. Consequently, a clear periodic behavior should be seen in the time domain results. Comparing the periodic behavior of the DWA algorithm to the lowest tone's period of the input sequence and the period of the lowest tone seen for DC input in the previous exercise, it shouldn't be difficult to see how multiple high-frequency tones arising from DEM scrambling and those present in the input sequence can interact to create lower-frequency tones that may fall within the signal band. This should highlight why out-of-band tones, especially in the case of the DWA algorithm, are considered a hazard, since they can interact with other tones to move in-band.

Methods for suppressing the DEM-generated tones analyzed are not to be treated in this book; however, the reader should be aware of the issue and that dither can often help considerably as it creates a more noise-like input to the DEM, free from modulator tones and limit cycles for sine input (i.e., noise-in tends to give noise-out). Further, it should be noted that many other DEM algorithm exists which are capable of providing a better tonal behavior [4, 7].

5.7 DEM-DAC Within a Sigma-Delta Modulator

The previous sections examined the stand-alone behavior of the dynamic element matched DACs, primarily because if these methods were to be examined in situ in a ΣΔ-modulator, then confusion could have arisen as to whether any tones or noise shaping observed was due to the DEM or the modulator or an interaction of both.

In this section the DWA-DEM applied inside the loop filter of a ΣΔ-modulator is to be examined.

In the Toolbox open the model *mod2_2bit_DWA.mdl* found in the folder *5_MB_DAC*, and type – *load_par* – into the MATLAB Command Window to reset the model's parameters.

Type – *psdset.win = 'hann'* – into the MATLAB Command Window to set the window function for this exercise and run the simulation.

Add some elements mismatch to the DAC by typing – *dac.elements = [1.05, 1, 0.95]* – into the MATLAB Command Window.

To smooth the modulator spectrum plot to aid in determining the noise-shaping order, add some dither by typing – *mod.dither = 0.1* – into the MATLAB Command Window.

Ensure the *Manual Switch* in the feedback loop is in the *Top* position in order to select the ideal feedback DAC, and run the simulation.

Type – *mod_SNDR_ac* – into the MATLAB Command Window to obtain the FFT plot.

The FFT results should highlight a second-order noise-shaping characteristic and an SNDR of approximately 78 dB.

Set the *Manual Switch* in the feedback loop in the *Bottom* position in order to select the real feedback DAC, and run the simulation.

Type – *mod_SNDR_ac* – into the MATLAB Command Window to obtain the FFT plot.

The FFT results should highlight a second-order noise-shaping (i.e., although less evident) characteristic and an SNDR of about 40 dB.

Enable the DWA algorithm by typing – *dac.dem_enable = 1* – into the MATLAB Command Window and run the simulation.

Type – *mod_SNDR_ac* – into the MATLAB Command Window to obtain the FFT plot.

Figure 5.11 illustrates the FFT results for both the ideal case and the DWA enabled DAC case, respectively.

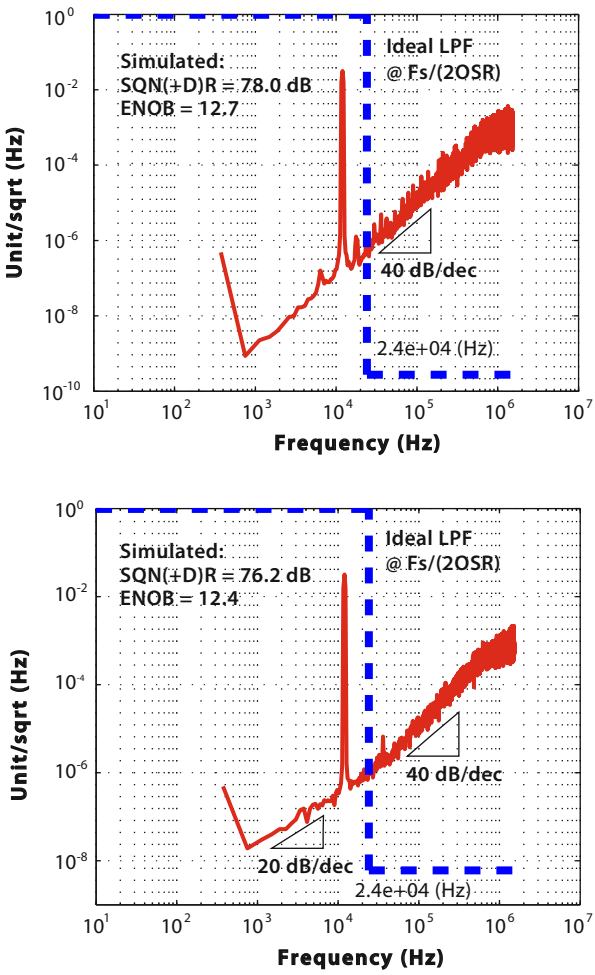


Fig. 5.11 FFT of (top) MOD2 with ideal DAC and (bottom) DWA enabled real DAC with ±5% mismatch

As it can be seen, the DWA enabled DAC results are very close in terms of SNDR to that of the ideal DAC case. Therefore, it can be said that DWA almost completely removes the effects of the DAC mismatches. Interesting to note is the slope of the noise spectrum in the passband of the DWA enabled DAC case that resembles the first-order noise-shaping characteristic of the DWA algorithm rather than the second-order noise-shaping slope of the second-order modulator. This highlights the fact that for low-frequency input signals to the modulator, a slightly poorer performance in terms of SNDR may be achieved than the ideal DAC case, as expected according to the theory previously discussed.

Repeating the above simulations for the Simulink® models *mod2_2bit_ILA.mdl* and *mod2_2bit_RES.mdl* should highlight that DWA provides the best suppression of the DAC errors in the signal passband, as already discussed, and thus the highest SNDR. The interested reader is also encouraged to repeat the simulations for a DAC

element mismatch closer to that of a typical device matching limits on a modern IC process (e.g., ±0.25%) in order to appreciate how, thanks to DEM techniques, DAC mismatches in multi-bit ΣΔ-modulators can be considered relatively not too problematic in achieving high-performance modulators.

5.8 Conclusions and Essential Takeaways

This chapter discussed the significant issue of mismatched elements in the feedback DACs of ΣΔ-modulators implementing multi-bit quantizers. It has been demonstrated how the nonlinearity resulting from even tiny DAC element mismatches can have catastrophic consequences for an analog modulator, causing both signal distortion (e.g., harmonic distortion) and quantization noise distortion (i.e., causing high white noise in the signal passband). On this regard, it has been shown how the implementation of dynamic element matching techniques, which scramble the mismatch error of the DAC's elements, can aid the linearization of the DAC in a time-average sense. Further, DEM algorithms that not only allow DAC linearization but also shape the noise resulting from the scrambling of the elements in order to leave only a small portion of it in the signal passband of a modulator have been presented and analyzed. Note that, until effective methods for addressing DAC linearity in multi-bit ΣΔ-ADCs were invented, modulators were limited to single-bit quantization and feedback with the attendant issues that result: high quantization noise, poorly defined quantizer gain affecting the NTF, difficulty in adding dither without further overloading the quantizer, etc. Compared to more traditional (and often expensive!) linearization techniques (e.g., element trimming, etc.), DEM algorithms have proven themselves as one of the most practical solutions, adding only very modest digital complexity to a modulator and exhibiting the additional advantage of being able to track any element drift caused by the aging of the analog components in an IC. Therefore, it shouldn't be surprising that DEM techniques are almost universally used in the design of high-performance and cheap ΣΔ-modulators implementing multi-bit quantizers.

Exercises

Q.1
Describe the advantages of multi-bit quantization in ΣΔ-modulators.

Q.2
What is the advantage of having a well-defined gain in multi-bit quantizers relatively to ΣΔ-modulator theory?

Q.3
Describe how dynamic element matching (DEM) techniques can be employed to improve the performance of high-order multi-bit ΣΔ-modulators.

Q.4
A five-level feedback DAC is to be designed for a Sigma-Delta modulator. The DAC elements have component values $E[0:3] = [1.1, 1.03, 0.97, 0.9]$. A constant digital value of 3 is the input to the DAC. Write down the sequence of analog output values for four clock cycles, assuming the elements are used starting from $E[0]$ *without* the use of DEM.

Q.5
Using the DAC of Q.4, write down the sequence of analog output values for four clock cycles, assuming the elements are used starting from $E[0]$ using data weighting averaging (DWA).

Q.6
Determine the mismatch error sequence found in part Q.4 and Q.5. Discuss the presence of limit cycles.

Q.7
Using the DAC of Q.4, write down the sequence of analog output values for four clock cycles, assuming the elements are used starting from $E[0]$ using individual level averaging (ILA) for the digital input sequence 1,3,1,3.

Q.8
In the Toolbox, simulate Q.5 and Q.7, and verify your results are in accordance with your answers in Q.5 and Q.7.

References

1. Schreier R, Temes GC. Understanding delta-sigma data converters. Hoboken: Wiley; 2005.
2. Oppenheim AV, Schafer RW. Discrete-time signal processing. Upper Saddle River: Prentice-Hall Inc; 1989.
3. Norsworthy SR, Schreier R, Temes GC. Delta-sigma data converters: theory, design and simulation. New York: Wiley-IEEE Press; 1996.
4. Geerts Y, Steyaert M, Sansen W. Design of multibit delta sigma A/D converters. New York: Kluwer; 2002.
5. Candy JC, Temes GC. Oversampling delta-sigma data converters: theory, design and simulation. New York: Wiley-IEEE Press; 1991.
6. Reiss JD. Understanding sigma-delta modulation: the solved and unsolved issues. JAES. 2008;56(1/2):49–64.
7. Casier H, Steyaert M, Van Roermund AHM. Analog circuit design: robust design, sigma delta converters, RFID. London: Springer; 2011.
8. De la Rosa JM, Del Rio RF. CMOS sigma-delta converters: practical design guide. London: Wiley; 2013.



MASH Sigma-Delta Modulators

6

So far only the possibility of creating stable, high-order Sigma-Delta modulators through a careful loop architecture choice and/or using multi-bit quantization has been considered. As demonstrated in previous chapters, although this approach can yield effective high-order loop designs, it also results in mathematically intensive (the Schreier's Toolbox helps considerably in this regard!) and extensive simulations that are required to check that instability does not occur for any envisioned modulator input. Moreover, especially for low oversampling values, it is almost impossible to obtain high SQNR performances in a single-bit quantizer modulator simply by increasing the order of the loop filter since stability considerations limit the permissible input signal amplitude for higher-order loops, which counteracts the improved noise suppress. On this regard, multi-bit quantizers allow, to some extent, to overcome such limitations but at the cost of increased complexity to ensure in-band linearity of the internal DAC, typically limiting the practical application to 4–5 bits at the most.

This chapter aims to provide a general overview of an alternative, popular approach for designing high-order modulators, particularly when the art of devising stable, high-order single-loop architectures was much less mature – which wasn't that long ago! These alternative modulators are known as *cascade* or *multi-stage noise shapers* (MASH) and rely on the cancellation, rather than the filtering, of the quantization noise. Therefore, by cascading unconditionally stable first- and/or second-order modulators, it will be demonstrated how stable higher-order noise-shaping converters can be realized.

6.1 MASH ΣΔ-Modulators Fundamentals

The concept of cascading unconditionally stable, low-order modulators is illustrated in Fig. 6.1. The output signal of the first stage is [1]:

© Springer International Publishing AG, part of Springer Nature 2019

I. Arnaldi, *Design of Sigma-Delta Converters in MATLAB®/Simulink®*,
https://doi.org/10.1007/978-3-319-91539-5_6

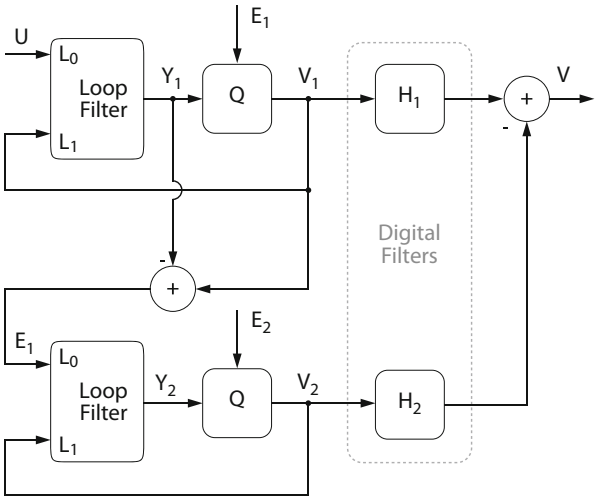


Fig. 6.1 Two-stage MASH structure

$$V_1(z) = \mathrm{STF}_1(z)U(z) + \mathrm{NTF}_1(z)E_1(z) \tag{6.1}$$

where STF_1 and NTF_1 are the signal and noise transfer functions, respectively, of the first stage. The input to the second stage is the quantization error E_1 obtained from the input stage and found in analog form by subtracting the input to its internal quantizer from its output. The quantization error E_1 is then processed by the second stage, in which the output signal results in:

$$V_2(z) = \mathrm{STF}_2(z)E_1(z) + \mathrm{NTF}_2(z)E_2(z) \tag{6.2}$$

The digital filter stages H_1 and H_2 at the outputs of the two modulator loops are designed such that in the overall output $V(z)$ of the system, the first stage error $E_1(z)$ is cancelled. According to Eqs. 6.1 and 6.2, this is achieved if such condition holds:

$$V(z) = H_1(z)V_1(z) - H_2(z)V_2(z) = 0 \tag{6.3}$$

The simplest and usually most practical choice for H_1 and H_2 which satisfies Eq. 6.3 is $H_1 = \mathrm{STF}_2$ and $H_2 = \mathrm{NTF}_1$. Since STF_2 is typically just a delay (e.g., z^{-1}), H_1 is then easily realized. Therefore, the overall output results in:

$$\begin{aligned} V(z) = {} & H_1(z)\mathrm{STF}_1(z)U(z) + [H_1(z)\mathrm{NTF}_1(z) - H_2(z)\mathrm{STF}_2(z)]E_1(z) \\ & - H_2(z)\mathrm{NTF}_2(z)E_2(z) \end{aligned} \tag{6.4}$$

$$V(z) = \mathrm{STF}_1(z)\mathrm{STF}_2(z)U(z) - \mathrm{NTF}_1(z)\mathrm{NTF}_2(z)E_2(z)$$

where the $E_1(z)$ term has been cancelled and $E_2(z)$ shaped by $\mathrm{NTF}_1(z)\mathrm{NTF}_2(z)$.

However, the main problem [2, 3] affecting MASH modulators is that if perfect cancellation of the analog and digital terms is not easily achieved (Eq. 6.3), due to imperfections in the realization of the analog transfer functions (e.g., capacitor matching, finite op-amp gain, non-unity quantizer gain, etc.), then E_1 will appear at the output multiplied by $\mathrm{STF}_{2d}(z)\mathrm{NTF}_{1a}(z) - \mathrm{STF}_{2a}(z)\mathrm{NTF}_{1d}(z)$ – where subscript a denotes the actual value of the analog transfer function and d the digital terms. This may result in a serious degradation of the SQNR performance of the modulator. It is beneficial, then, for MASH systems to use a low-distortion loop filter structure in all stages [4] which makes possible to obtain the first stage error $E_1(n)$ without any subtraction for entering it into the second stage (i.e., Silva-Steensgaard MOD2). Furthermore, the low-distortion property helps improving the performance of both stages.

An advantage of MASH structures is that the remaining error in the output V is the shaped quantization error $E_2(n)$ of the second stage operating *with* an input $E_1(n)$, which is itself noise-like. Thus, the second stage quantization error $E_2(n)$ is very similar to a true white noise; even in the case, the first stage noise contains tones. This important characteristic results in MASH modulators being less likely to need dithering when compared to single-stage architectures [1].

Another useful property of the MASH structure is that it often allows the use of a multi-bit quantizer in the second stage, without any DEM or other correction of the DAC nonlinearity [5]. This is true since the nonlinearity error of the second stage DAC (as part of V_2) is multiplied by $H_2(z)$ before being added into the output signal V. As shown in Eq. 6.4, $H_2(z)$ contains the NTF of the first stage. Since this $\mathrm{NTF}_1(z)$ is a high-pass filter function, the nonlinearity error of the second stage DAC is suppressed in the baseband [1].

Furthermore, since the input to the second stage contains the quantization error E_1 of the first stage, rather than the input signal, no harmonic distortion of the signal is generated in the second stage, and, especially for high OSR and small nonlinearity errors, the small added noise due to the DAC nonlinearity is usually tolerable [2, 3].

Example

Cascading two second-order modulators with $\mathrm{STF}_1(z) = \mathrm{STF}_2(z) = z^{-2}$ *and* $\mathrm{NTF}_1(z) = \mathrm{NTF}_2(z) = (1 - z^{-1})^2$ results in a fourth-order MASH modulator. Briefly state the advantage, in terms of stability, of this arrangement.

Solution

Starting from Eq. 6.4:

$$V(z) = \mathrm{STF}_1(z)\mathrm{STF}_2(z)U(z) - \mathrm{NTF}_1(z)\mathrm{NTF}_2(z)E_2(z)$$

$$V(z) = z^{-2} \cdot z^{-2} \cdot U(z) - \left(1 - z^{-1}\right)^2 \cdot \left(1 - z^{-1}\right)^2 \cdot E_2(z)$$

$$V(z) = z^{-4}U(z) - \left(1 - z^{-1}\right)^4 E_2(z)$$

resulting in a fourth-order modulator NTF whose stability resembles that of a second-order one.

6.2 A Third-Order 1+ 1 + 1 MASH ΣΔ-Modulator

The Simulink® model proposed to study MASH architectures comprises a modulator cascading three first-order (i.e., MOD1) loops to achieve an overall third-order noise shaping. Opening the file *"mod3_1plus1plus1MASH.mdl"* found in the folder *6_MASH* of the Toolbox, it can be seen that the model includes:

- Three cascaded MOD1 blocks implementing a 2-bit quantizer, except for the last stage which uses a single-bit one
- Difference operational blocks (highlighted in green in the model), used to derive the quantization errors E_1 and E_2 for the first two stages by subtracting these quantizer inputs from their outputs (i.e., quantizer output = quantizer input + quantization error)
- Digital logic (yellow highlighted blocks in the model) used to generate the overall modulator output by reassembling the *digital* outputs of each quantizer, which contain various combinations of the individual quantization errors shaped by the STFs and NTFs of the sub-modulators

The comments with red background on the Simulink® model show how the quantization errors are generated and combined algebraically in the digital domain (i.e., yellow blocks) to cancel the errors E_1 and E_2 from the first and second stages in order to leave only:

- The third stage quantization error, E_3, with third-order noise shaping
- The input signal processed by the STFs of all three modulator stages

Note that in order to achieve the exact cancellation of E_1 and E_2, the digital representations of each modulator's STF and NTF (e.g., as encoded in "*Digital Filter1*" and "*Digital Filter2*") must perfectly match the true STF and NTF of each analog loop, and this is where MASH modulators can fall down in practice, as already mentioned. Therefore, in the following exercises, the operation of this MASH converter is to be examined with the main focus of getting a practical feel for its sensitivity to some of the common analog/digital path matching errors.

6.2.1 Operation for DC Inputs

In the Toolbox open the model "*mod3_1plus1plus1MASH.mld*" found in the *6_MASH* folder.

Type – *load_par* – in the Matlab Command Window to initialize variables needed for the model to function.

Set the DC input level to zero by typing – $dc = 0$ – into the Matlab Command Window, and ensure the manual switch in the model selects the DC input source.

Run the simulation.

Open the Time Scopes and type – *mod_SNDR* – in the Matlab Command Window to obtain the output spectrum.

Save the result and repeat for the following DC input values: 0.5, 0.75, 0.875, 0.9375, 1, and 1.25.

The results of the simulations should highlight that:

- The modulator is stable for all inputs up to the maximum allowable full scale (e.g., ± 1), resembling the properties of first-order modulators, rather than single-loop third-order ones. This confirms the effectiveness of using a cascade of unconditionally stable low-order modulators to achieve high-order ΣΔ-converters with greater stability.
- Within the stable input range, the mean value of the filtered bit sequence of the modulator output corresponds to that of the DC input signal (i.e., time domain results), where the resultant percentage error is related to the power of 3, as expected in a third-order modulator. Therefore, the conversion can be considered accurate.
- The modulator output is periodic.
- Examining the FFT results, noise shaping is not apparent, and strong tones are present in the spectrum.

6.2.2 Operation for Sinewave Inputs

In the Toolbox open the model "*mod3_1plus1plus1MASH.mld*" found in the *6_MASH* folder.

Reset initial conditions by typing – *load_par* – into the Matlab Command Window, and ensure the manual switch in the model selects the sinewave input source.

(continued)

Type – *sinamp = 0.1* – in the Matlab Command Window to set the sinewave amplitude (i.e., this amplitude is the same used in the MOD3 chapter, Sect. 4.3.3, and aids results comparison in order to better understand MASH behavior).

Run the simulation.

Open the Time Scopes and type – *mod_SNDR* – in the Matlab Command Window to obtain the output spectrum.

The 1 + 1 + 1 MASH FFT output spectrum of Fig. 6.2 clearly illustrates the third-order noise-shaping characteristic with 60 dB per decade slope, hence confirming the operation of the MASH architecture.

To verify the SQNR achieved, it is useful to compare its value with the theoretical SQNR resulting from the formula [1]:

$$\begin{aligned} \text{SQNR} &= 6.02N + 1.76 + (20L + 10)\log_{10}(\text{OSR}) - 10\log_{10}\left(\frac{\pi^{2L}}{2L+1}\right) \\ &\cong 112.8\ (\text{dB}) \end{aligned} \tag{6.6}$$

where the modulator order is $L = 3$, the oversampling ratio is OSR = 64, and the number of bits corresponds to $N = 1$ (note that an error based only on E_3 is expected – last MASH stage – which is a 1-bit quantizer [1, 6]).

Equation 6.6 corresponds to the theoretical SQNR for a full-scale input – *sinamp = 1* (i.e., 0dBFS). Having used an amplitude of 0.1 for the simulation, corresponding to −20 dB, Eq. 6.6 needs to be modified appropriately in order to be able to compare the results obtained with the theory. Therefore, by subtracting the value of −20 dB to the results obtained (e.g., 112.8–20 = 92.8 dB), it is seen that the

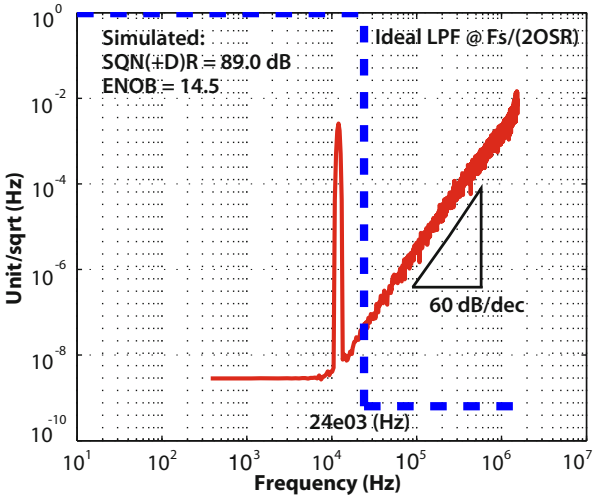


Fig. 6.2 FFT Results for 1 + 1 + 1 MASH with sinewave input

modulator differs by approximately 4.2 dB from the expected theoretical value, which is acceptable considering FFT accuracy errors and other imprecisions of the simulation.

Comparing the result with the one of the CIFF-MOD3 analyzed in Chap. 4 (i.e., 71.8 dB, Fig. 4.5), it is easily seen that the difference between simulated and theoretical SQNR values is much smaller for the MASH modulator. The main reason of such behavior is due to MASH architectures not suffering from a modified NTF (i.e., Lee's criterion), which is necessary to ensure stability in single-loop MOD3s (i.e., from the MASH output spectrum of Fig. 6.2, note that at high out-of-band frequencies, the spectrum does not flatten). Further, signal-dependent quantizer gain should also be considered, since the last stage of the MASH architecture under analysis implements a 1-bit quantizer. Being the input signal to the quantizer larger in single-loop MOD3 architectures than that of MASH modulators (i.e., recall that the input signal of the last stage of a $1 + 1 + 1$ MASH modulator is just the noise E_2 of the previous stage; thus the result is relatively small), it is seen that the MOD3 architecture will provide lower SQNR compared to MASH structures (i.e., greater input signals to the quantizer correspond to decreased quantizer gain and noise-shaping effectiveness). However, signal-dependent quantizer gain still affects MASH modulators as well, explaining the discrepancies from theoretical to simulated SQNRs [1, 5].

In order to appreciate the achievable SQNR within the stable input range for the $1 + 1 + 1$ MASH modulator and to compare it with the one of the CIFF-MOD3, it is useful to perform a sweep of sinewave amplitudes.

In the Toolbox open "*sweep_testbench.mdl,*" right click on the *bench* block, and enter in the *Model Reference Parameters*. Change the *Model Name* to "sweep*_mod3_1plus1plus1MASH.mld*," press *ok*, and save.

Reset initial conditions by typing – *load_par* – into the Matlab Command Window, and type -*simu.select* = 2 – to select the sinewave input source.

Set the sinewave frequency to 6 KHz by typing – *sinfreq* = *6e3* – into the Matlab Command Window (i.e., this frequency is the same used in the MOD3 chapter, Sect. 4.3.3, and aids results comparison in order to better understand MASH behavior).

Type – *target_mod* = *'sweep_mod3_1plus1plus1MASH'* – into the Matlab Command Window.

Type – *sweep_sinamp* – into the Matlab Command Window and run the simulation. Note it may take some time to complete.

Comparing the results of Fig. 6.3 with those of the CIFF-MOD3 with 1, 2, and 3 quantizer bits (e.g., Fig. 4.6, 4.8, and 4.9), it is easily seen that the 1 + 1 + 1 MASH modulator performs similarly to a 3-bit CIFF-MOD3, with the already mentioned advantages of full-input range, good stability, and generally higher SQNR

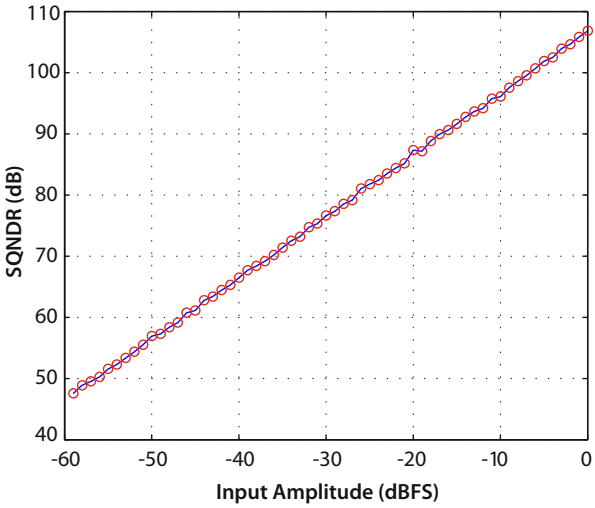


Fig. 6.3 SQNR vs. swept amplitude of a 6 KHz sine input for 1 + 1 + 1MASH

performance due to both the smaller signals present at each stage and the avoidance of having a reduced NTF aggressiveness.

However, the maximum SQNR at input amplitudes close to 0 dBFS is found at only 108 dB, which is approximately 4 dB lower than the expected theoretical value of 112 dB. This further highlights that MASH modulators are still affected by limited quantizer gain as well as the performance of each of its stages, where the first and the last are usually the ones having more impact on the overall performance.

6.2.3 The Effects of Finite Integrator Gain

In Chap. 4 Sect. 4.3.2, it has been shown that single-loop CIFF-MOD3 with single-bit quantization still performed well even for very modest integrator DC gains, as all three gains multiply together before quantization. Therefore, the next task proposes to investigate the tolerance of the 1 + 1 + 1 MASH modulator to finite DC gain in the integrators.

In the Toolbox open "*sweep_testbench.mdl*," right click on the *bench* block, and enter in the *Model Reference Parameters*. Change the *Model Name* to "sweep_*mod3_1plus1plus1MASH.mld*," press *ok*, and save.

Set the DC input level to zero by typing – $dc = 0$ – into the Matlab Command Window, and type – $simu.select = 1$ – to select the DC input source.

(continued)

Type – *target_mod = 'sweep_mod3_1plus1plus1MASH'* – into the Matlab Command Window.

Type – *sweep_dc_dc* – in the Matlab Command Window and run the routine. A DC sweep for a subset of the input range (i.e., ±0.1) will be performed. Once the simulation is completed, do not close the results window.

Set the DC gain of all integrators to 10 by typing – *mod.igain1 = 10; mod.igain2 = 10; mod.igain3 = 10;* – into the Matlab Command Window, and rerun the "*sweep_dc_dc*" routine.

Figure 6.4 illustrates the results in the case of finite integrator gain of 10.

Comparing the results with those of a MOD3 with the same integrator gains, it is seen that larger errors are present, including the overall slope of the output which does not match the slope of the input (i.e., output ≠ input). Further, much larger dead zones are seen, especially around DC = 0. This demonstrates the high sensitivity of MASH architectures to inexact cancellation of the analog and digital terms due to imperfections in the realization of the analog transfer functions in the digital filtering stage, where E_1 and E_2 appear at the output multiplied by their correspondent $STF_{ia}(z)$ and $NTF_{ia}(z)$, thus introducing errors in the conversion. From this analysis, a straightforward trade-off between MASH and single-loop MOD3 is clearly apparent. Although MASH modulators provide better stability and higher stable input range than MOD3 modulators, their sensitivity to real analog issues such as finite integrator gains may make their actual circuital design more challenging, since circuital imperfections are less tolerated for a satisfactory performance [1, 4].

To better understand the sensitivity to finite integrator gain in MASH modulators, it is instructive to investigate the DC gain errors in the sub-modulators with the sinewave input and a more realistic integrator DC gain.

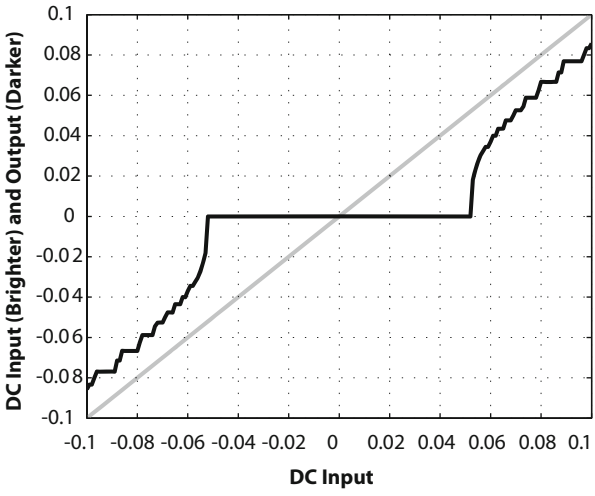


Fig. 6.4 1 + 1 + 1 MASH DC transfer function with finite integrator gain of 10

Type – *simu.select = 2* – into the Matlab Command Window to select the sinewave input source and control that – *target_mod = 'sweep_mod3_1plus1plus1MASH'*.

Set the third loop integrator DC gain to 50 by typing – *mod.igain3 = 50;* – into the Matlab Command Window, and rerun the "*sweep_sinamp*" routine.

Once the simulation is completed, do not close the results window.

Set only the second loop integrator DC gain to 50 by typing – *mod.igain3 = inf; mod.igain2 = 50;* - into the Matlab Command Window, and rerun the "*sweep_sinamp*" routine.

Again, do not close the resultant figure.

Finally, set only the first loop integrator DC gain to 50 by typing – *mod.igain3 = inf; mod.igain2 = inf; mod.igain1 = 50;* – into the Matlab Command Window, and rerun the "*sweep_sinamp*" routine.

The results should demonstrate that the first stage of MASH modulators is the most sensitive to finite integrator gain and the one which has bigger impact on the overall modulator performance, since it is its error that has to be cancelled [2, 3]. Therefore, it should be highlighted at this point that the tolerance available on finite gain (i.e., op-amp matching in real circuits) depends on how much the first stage SQNR needs to be increased, as its error needs to be cancelled by at least that much [1]. For this reason, among others, it is often useful to implement a second-order modulator as first stage of a MASH converter, as investigated in the next paragraph.

Example

Going from a 40 dB SQNR first stage to a 100 dB overall modulator SQNR target for a 1 + 1 + 1 MASH architecture with identical sub-modulators each having 40 dB SQNR in the signal band requires the first stage error to be reduced by, at least, 60 dB. What consideration should be taken into account regarding the cancellation accuracy and limits of CMOS technology?

Solution

A rough estimation can be found from:

$$10^{\left(\frac{60}{20}\right)} = 1000$$

$$\frac{1}{1000} \approx 0.001 \rightarrow 99.9\%$$

Accurate cancellation of, at least, 99.9% of E_1 by the modulator output (e.g., at H_1) to get it down to 0.1% of the first stage output error. This would be near typical matching limits of any current CMOS technology [6], hence being highly unpractical to realize!

6.2.4 A Third-Order 2 + 1 MASH ΣΔ-Modulator

The 2 + 1 MASH modulator [1, 7] achieves an overall third-order noise shaping by cascading a second-order and a first-order modulator, as the name implies. The second-order stage, in this case, is the Silva-Steensgaard modulator seen in Chap. 3, now implemented with a 2-bit quantizer to help stabilize the quantizer gain and hence the NTF. Further, note that this type of modulator is convenient to use in MASH structures because the quantization error is already available in analog form at the output of the second integrator (i.e., delayed by two clock cycles). This saves the need to add an additional analog differential stage to compute the error, with associated scope for matching errors (i.e., the Simulink® model "*mod3_2splus1MASH_v2.mdl*" in the Toolbox demonstrates how the quantization error could be obtained if it was not already available). The modulator of the second stage is a simple 1-bit MOD1.

To investigate the increased tolerance to finite integrator gains, a simple DC sweep simulation can be conducted.

Type – *load_par* - to reset initial conditions.

In the Toolbox open "*sweep_testbench.mdl*," right click on the *bench* block, and enter in the *Model Reference Parameters*. Change the *Model Name* to "*mod3_2plus1MASH.mld*," press *ok*, and save.

Type – *simu.select = 1* – into the Matlab Command Window to select the DC input source.

Type – *target_mod = 'sweep_mod3_1plus1plus1MASH'* – into the Matlab Command Window.

Set the integrator gain by typing – *mod.igain1 = 10; mod.igain2 = 10; mod.igain3 = 10;* – into the Matlab Command Window.

Type – *sweep_dc_dc* – into the Matlab Command Window and run the simulation. Note it may take some time to complete.

According to the result in Fig. 6.5, the 2 + 1 MASH modulator is much more tolerant than the 1 + 1 + 1 configuration to finite DC integrator gains, with a performance similar to that of the third-order CIFF modulator analyzed in Chap. 4. As already seen, the higher tolerance of the 2 + 1 MASH modulator is due to the high gain resulting from multiple integrators in front of the *first* quantizer (i.e., dead zone width proportional to $\approx 1/A^2$ in the first stage of the 2 + 1 MASH modulator). However, it should be noted that the first stage is still the most sensitive one, and poor integrator gain could still lead to poor modulator performance as it can be investigated by performing a simple sinewave sweep simulation. The interested reader is encouraged to perform all the DC and sinewave simulations seen throughout the Chaps. 2, 3, and 4 in order to study the 2 + 1 MASH modulator in detail.

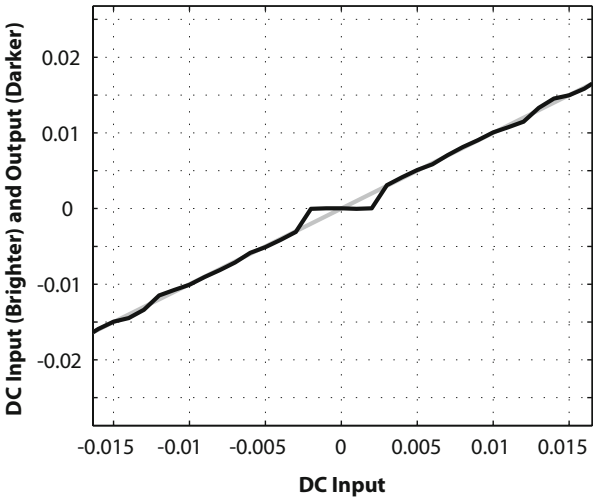


Fig. 6.5 2 + 1 MASH DC transfer function with finite integrator gain of 10

6.3 Additional Simulations for MASH Modulators

The simulations conducted so far only looked at the effects of finite integrator DC gain on the performance of the MASH modulator. Intuitively, other sources of error can also cause a poor modulator performance, even if the integrators would be made from ideal amplifiers. Such sources of errors are of particular interest in MASH modulators, where the analog and digital NTF need to cancel as perfectly as possible to obtain satisfactory performances. Therefore, a brief overview of those is here introduced.

6.3.1 Quantizer Gain

As examined in different occasions in previous chapters, the effective gain of the linearized approximation of a binary quantizer does vary with the signal level of its input [1–3] and therefore, in MASH architectures, can cause the analog and digital NTFs to be mismatched. However, since using more than 1-bit of quantization helps stabilizing the gain and hence the NTF, it is easily understood that it also aids the cancellation of the analog and digital terms in MASH structures. This is why all the examples seen in this chapter used 2-bit quantization in stages where quantization error was to be cancelled. The interested reader may wish to experiment with the Simulink® model – *mod3_2plus1_MASH_1bit* – where the 2-bit quantizer of the first stage has been replaced with a single-bit one in order to investigate how the SNR, and hence the third-order noise shaping, behaves as the amplitude of the input sinewave is increased. The results should show that the signal-dependent quantizer gain of the first stage greatly affects the SNR of the MASH modulator,

hence further highlighting the importance and the need to achieve a good performance in the first stage.

Note that in the Toolbox all the components are actually ideal, the cancellation difficulty being in achieving well-defined NTFs from the sub-modulators due to the signal-dependent quantizer gains causing their NTFs to differ from the ideal $(1+z)^{-1}$. Further, note that because this effect is due to system-level quantizer gain effects, it can affect even an *all-digital* MASH modulator that uses single-bit word length truncation [8, 9].

6.3.2 Signal Path Gain

Gain path errors (e.g., inputs to a signal differencing stage not having exact weightings of +1 and –1 in a real analog circuit due to small random component variations, etc.) are often another significant source of cancellation error between the analog and digital part of MASH modulators [4, 5, 7]. Important to note is that, although any real analog circuit incorporates many sub-paths with errors, their aggregate effect can be approximated, from a linearized model point of view, by condensing them into a single overall path gain error. These aggregated gain errors can be simulated by setting the "*mod.mgain1/2/3*" parameters visible in all the Simulink® MASH models provided in the Toolbox. The reader should compare SQNR sensitivity to gain errors in the different stages to sensitivity to different integrator gains seen in paragraph 6.2 and note that a similar trend is found, where gain errors around the first modulator stages are much more important than similar errors in the final modulators.

Important to consider is that if a modulator SQNR is degraded by *any* gain path errors approaching the component matching limits of the technology in which the modulator is going to be implemented [6, 7], then alarm bells should be ringing! That is, note that resistors and capacitors typically achieve a matching of about 0.25% accuracy in modern IC process and DT modulator designs [6], provided these are drawn in layout as best as possible (large geometries, dummy devices, etc.). Despite seemingly pretty accurate, it has been mentioned in the example in paragraph 6.2.3 that a 1 + 1 + 1 MASH converter with identical sub-modulators each having 40 dB SQNR in the signal band aiming for an overall SQNR of 100 dB means that the first stage quantization error must be reduced by, at least, 60 dB to make negligible contribution to the overall SQNR. However, noting that a 60 dB reduction only matches the overall SQNR target, it results that the error contribution of E_1 would need to be less than the target for it to actually have negligible contribution! Achieving a reduction greater than 60 dB requires E_1 to be reduced to less than 1/1000 in size at the output or, in other words, cancelled to an accuracy greater than 99.9%. This would allow for an aggregate signal path gain matching errors smaller than 0.1%, which would be a severe challenge in any IC process without resorting to expensive trimming or some sort of calibration technique.

Example

Assume that a 2 + 1 MASH architecture is used where the first modulator has 70 dB in-band SQNR. What reduction is expected in order to cancel E_1 to negligible levels, considering a 100 dB SQNR target? What consideration should be taken into account regarding the cancellation accuracy, signal path gain errors, and limits of CMOS technology?

Solution

Cancelling E_1 to negligible levels requires reducing it by more than 30 dB – cancelling to > 97% accuracy. Therefore the gain path errors should result in < <3%. This is a reasonable proposition for any modern CMOS technology (e.g., a total error of 1% should be easily achievable, giving about 10 dB margin below the target SQNR).

The interested reader may wish to examine the sensitivities of the 1 + 1 + 1 and 2 + 1 MASH ΣΔ-modulators to signal path gain errors and compare these to the typical matching limits (i.e., about 0.25% accuracy) given in the example above (i.e., there should be a good agreement in the results). The reader may also wish to examine the SQNR of the first stage sub-modulators (e.g., *Qout1*, *2*, etc., by using the routines – *Qout_1_1_1MASH_SNDR and Qout_2_1MASH_SNDR*) and see how these compare to the error sensitivities observed.

6.4 Conclusion and Essential Takeaways

This chapter examined the approach of designing high-order ΣΔ-ADCs by cascading a series of inherently stable low-order modulators. In these alternative converters, known as MASH, the quantization error of each stage is quantized by the succeeding stage/s and subtracted digitally. Thus, the order of the noise shaping equals the sum of the orders of the stages. Important to note is that the quantization noise cancellation depends on the precision of analog/digital signal paths.

Single-loop designs [1–3] offer good tolerance to errors such as finite DC integrator gain and signal path gain matching errors; however, they are difficult to stabilize due to having long chains of integrators in the NTF path. Further, the aggressiveness of the noise shaping in the signal passband often needs to be reduced in order to ensure stability. Despite some heuristic rules, such as Lee's criterion, exist for modifying the NTF via changing to loop topology/architecture to obtain stability, extensive simulation is still required in the design process to ensure that, for example, exceeding the stable input range does not result in the loop locking up indefinitely.

On the other hand, the main advantage of MASH converters [1–3] is their ability to offer unconditional loop stability and wide input range. Moreover, MASH architectures tend to further randomize the quantization noise and, thus, result in being less sensitive to limit cycles. However, matching errors between the analog

and digital signal processing paths typically make it very difficult in practice to obtain more than 30–40 dB SQNR increase over the first modulator in the MASH cascade. Therefore, they may not be a viable solution for all high SQNR applications, and for this reason MASH converters tend to be preferred as ΣΔ-DACs rather than ΣΔ-ADCs [10].

There are thus a wide range of trade-offs in choosing a suitable modulator topology for high SQNR applications requiring loop orders greater than the second order, and given the empirical/heuristic nature of much ΣΔ-modulator theory, experience and experiment still play a large role in choosing an appropriate architecture, especially for the sensitive first stage modulator of a MASH converter.

Exercises

Q.1

Sketch the general block diagram of the $1 + 1 + 1$ MASH ΣΔ-modulator, and find the output transfer function for $V(z)$.

Q.2

Find the output transfer function for $V(z)$ of the MASH ΣΔ-modulator shown in Fig. 6.6, and determine its order. What is the delay, in terms of STF(z), at the output?

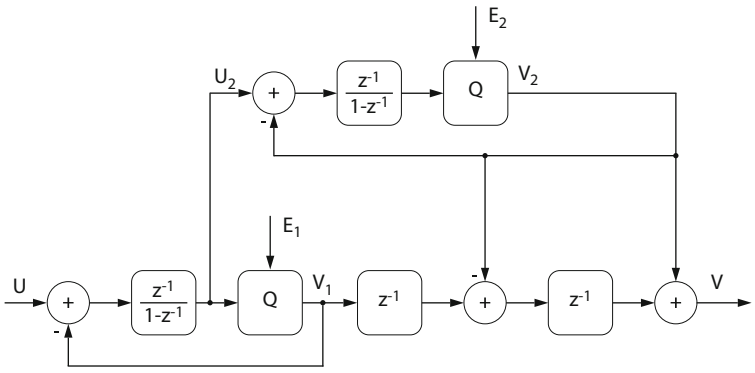


Fig. 6.6 ΣΔ-MASH modulator block schematic for Q.2

Q.3

Sketch a block diagram of a fourth-order MASH ΣΔ-modulator. Carefully choose the modulator architectures of the different stages, and explain your choices. (Note: Do not worry about achievable SNR, etc., but solve the exercise solely from a theoretical point of view.)

Q.4
Explain the practical implementation difficulties faced by MASH ΣΔ-modulator structures.

Q.5
Consider a 2 + 2 MASH ΣΔ-modulator. Discuss if you would require DEM logic to mitigate the effects of feedback DAC nonlinearity in any of the stages.

Q.6
Sketch the first stage of a 2 + 1 MASH ΣΔ-modulator using the simplest MOD2 architecture, demonstrating how the input to the second stage could be obtained if it was not already available as in the case of Silva-Steensgaard architectures.

Q.7
Assume that a 2 + 1 MASH architecture is used with a first stage in-band SQNR of 60 dB. What reduction is expected in order to cancel E_1 to negligible levels, considering a 100 dB SQNR target? What consideration should be taken into account regarding the cancellation accuracy, signal path gain errors, and limits of CMOS technology?

Q.8
Briefly state the main differences between single-loop and MASH ΣΔ-converters.

References

1. Schreier R, Temes GC. Understanding delta-sigma data converters. Hoboken: Wiley; 2005.
2. Norsworthy SR, Schreier R, Temes GC. Delta-sigma data converters: theory, design and simulation. New York: Wiley-IEEE Press; 1996.
3. Candy JC, Temes GC. Oversampling delta-sigma data converters: theory, design and simulation. New York: Wiley-IEEE Press; 1991.
4. Casier H, Steyaert M, Van Roermund AHM. Analog circuit design: robust design, sigma delta converters, RFID. London: Springer; 2011.
5. Geerts Y, Steyaert M, Sansen W. Design of multibit delta sigma A/D converters. New York: Kluwer Academic Publisher; 2002.
6. Reiss JD. Understanding sigma-delta modulation: the solved and unsolved isses. JAES. 2008;56(1/2):49–64.
7. De la Rosa JM, Del Rio RF. CMOS sigma-delta converters: practical design guide. London: Wiley; 2013.
8. Medeiro F, Verdù BP, Vazquez AR. Top down design of high performance sigma delta modulators. Boston: Kluwer Academic Publisher; 1999.
9. Bourdopoulos G, et al. Delta-sigma modulators: modelling, design and applications. London: Imperial College Press; 2003.
10. Graells SF. Integrated heterogeneous systems design. [Online] http://www.cnm.es/~pserra/uab/ihsd/ihsd-5-dsm.pdf. Date accessed 24 June 2017.



Continuous Time Sigma-Delta Modulators

7

This chapter is dedicated to introducing the continuous time (CT) implementation of Sigma-Delta modulators. Therefore, the significant differences between discrete time and continuous time modulators will be analyzed, as well as the differences concerning the actual design process. On this regard, the mathematical equivalence between CT and DT modulators is presented and used to demonstrate the inherent anti-aliasing characteristic of CT designs. Moreover, important non-idealities which are specific to CT modulators, as well as some practical issues derived from actual circuits, will be discussed and suggestions for simulation practices of CT modulators in MATLAB/Simulink® provided. By the end of this chapter, the reader should have a basic knowledge of all the fundamentals relative to continuous time Sigma-Delta converters and be able to develop basic, high-level CT modulator systems in the Simulink® environment.

7.1 DT-CT Loop Filter Equivalence

Due to a large amount of literature and mature software available to design discrete time ΣΔ-modulators, the proposed continuous time design procedure is to start with a DT-NTF, calculate the relative DT loop filter $H(z)$, and then transform it into a continuous time $H(s)$ equivalent. The specific non-idealities of CT modulators are then added to the modulator's model and their effect simulated by transforming the CT modulator back to its DT equivalent for the MATLAB/Simulink® environment. This last transformation is advantageous because simulations are much faster in the DT domain, due to the less computational power required. From an analytical modeling point of view, the advantage of developing the design in discrete time is that it results easier than a continuous one since CT modulators are mixed CT/DT systems [4, 5].

The transformation between discrete and continuous time domains is possible due to the fact that a noise shaper having the loop filter implemented with CT circuits still

© Springer International Publishing AG, part of Springer Nature 2019

I. Arnaldi, *Design of Sigma-Delta Converters in MATLAB®/Simulink®*,
https://doi.org/10.1007/978-3-319-91539-5_7

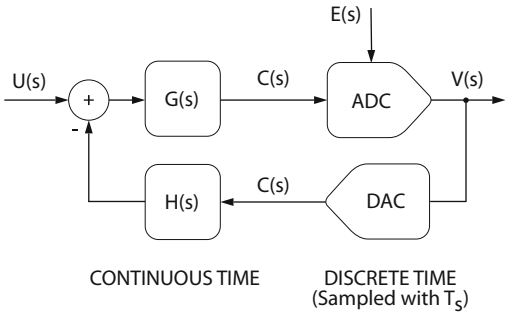


Fig. 7.1 Noise shaper with CT loop filter

remains a sampled system [4, 6], as shown in Fig. 7.1. Therefore, the quantizer output $V(s)$, as well as the DAC output $D(s)$, only changes on the edge of a sampling clock signal with period Ts. Assuming the input signal $U(s)$ is largely oversampled, the ΣΔ-modulator in Fig. 7.1 can be designed to be equivalent to its DT-based version (in terms of large signals) if for identical inputs $u(t)$ and $d(t)$ during a clock cycle the input to the quantizer $c(t)$ at the end of the current clock cycle is identical to the one in the DT version. To satisfy this condition, the impulse response of the filters connecting the input to the quantizer and the DAC output to the quantizer should have the same impulse response in the DT and CT variants. With this condition satisfied, the time domain signal sampled by the quantizer is identical in the DT and CT designs. Therefore, ideally, the time domain simulation results obtained for the DT loop filter should equal to that of the CT modulator. Important to note is that the equivalence described exploits the linear model of the ΣΔ-loop filter, which limits its applicability to the case of the input $u(t)$ having a large amplitude. Therefore, in the case of a quantization noise $E(s)$ no longer independent to $U(s)$, the equivalence between the CT and DT loop filters would not guarantee identical input signals to the quantizer, hence not allowing for an equivalent system [7, 8].

Given the limitation in the equivalence method described above, another approach can be taken in which the complete STF and NTF are mapped [4, 6, 8], instead of the attempt of the tight mapping between the two filters connected inside the loop (i.e., $G(z)$ and $G(z)H(z)$ individually). By designing the entire NTF(s) and STF(s) to show identical impulse responses, the partial contributions of the input signal and the quantization noise to the quantizer input are identical in both the DT and CT domains. This allows for a better comparison between the behavior of the DT and CT equivalent loop filter because the poles and zeros of the entire transfer functions are matched. Therefore, such method is generally preferred for a systematic analysis. No advantage is lost since the mapping does not actually guarantee the identical time domain behavior of the two designs for the entire input dynamic range. Important to note is that the direct mapping of NTF and STF assumes a constant DAC output during one clock cycle; therefore, any discrepancy from this assumption results in a degradation of the modulator performance.

The most common mathematical tools which allow transforming back and forth DT and CT transfer functions include the impulse invariant transform (IIT) [6, 9], the modified Z-transform [5, 8], and the state-space [4] time domain representation (i.e., used by the Schreier's Toolbox). Due to the practical and introductory nature of this book, as well as the complexity of the mathematical tools mentioned, these techniques are not discussed. However, it is shown how to perform such transformation by using the dedicated MATLAB routine available in the Schreier's Toolbox, and the interested reader is redirected to [4] to investigate the mathematics behind the state-space analysis technique and how the DT-to-CT mapping of the NTF and STF impulse responses is achieved.

7.2 DT-CT Modulators Overview

The main difference of the modulation process between discrete and continuous time ΣΔ-converters is the position of the sampler in the system [6, 10], which is inside the loop filter in the CT case and outside the loop in the DT one (Fig. 7.2). In DT implementations, the sampler circuit must feature good linearity and limited errors properties, as any error appears directly at the output of the ADC. On the other hand, in CT systems any error introduced by the sampler (e.g., settling error, nonlinearities, etc.) is suppressed by the gain of the preceding filters and hence shaped in the loop. Therefore, the circuit requirements of the sampler in CT modulators are largely relaxed [2].

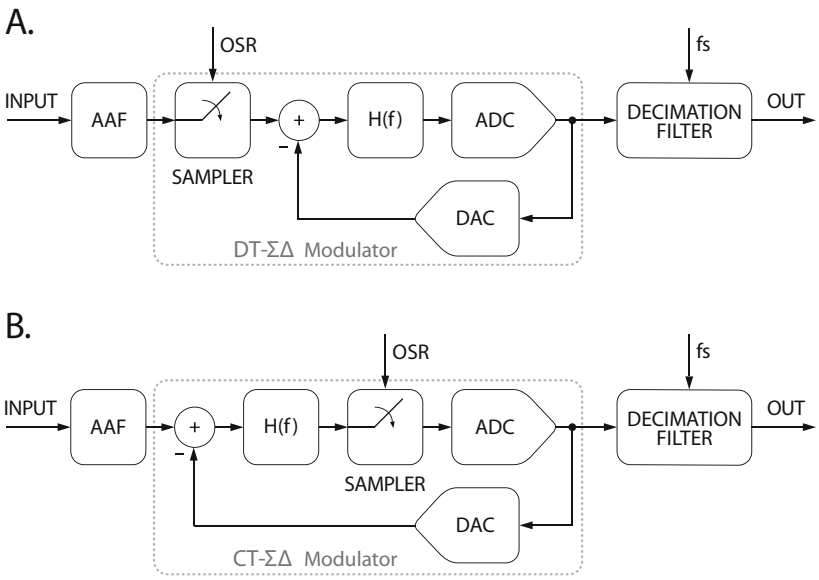


Fig. 7.2 (**a**) DT and (**b**) CT ΣΔ-modulators block schematics

Important to note is also that in a DT modulator, the overall circuit noise is mainly due to the thermal noise of the input switches, which is sampled on the input capacitors. This sets a limit on the minimum size of these capacitors. In CT modulators, as sampling occurs before the quantizer and any noise contribution of the sampler is suppressed in the loop, this limitation is not present. Furthermore, the white noise terms result aliased because of sampling in DT modulators, while in a CT design, these are, again, suppressed in the loop. Thus, CT modulators are generally expected to exhibit less noise [2, 6, 9].

Continuous time modulators are also expected to allow a higher speed of operation and a lower power consumption [6, 8] when compared to similar DT modulators. An important characteristic of DT modulators is the settling behavior of switched-capacitor filters since these are clocked systems and as such they must handle a charge packet in a defined time window [10]. Typically, as long as the feedback charge is transferred within a half clock period and the output value settles well, the output of the loop filter can be considered correct. Important to note is that the shape of the feedback DAC pulse is not important. However, it should be considered that the voltages of the circuit nodes change abruptly whenever switching occurs, and as such, fast (i.e., GBW >> F_s) and power-hungry active circuits capable of managing these abrupt changes are required. This limits both the power consumption and the highest bandwidth the modulator can achieve [1, 4]. In continuous time designs, thanks to the sampler position inside the loop, the input signal is processed by the CT loop filter resulting in smooth internal signals with no abrupt amplitude variations, allowing for relaxed speed requirements of the integrators. Note that the feedback DAC pulse is continuously fed into the loop filter during the entire clock period, and as such, only small timing errors are allowed. From a signal processing perspective, the DAC pulse is convoluted with the loop filter impulse response in the time domain to generate the quantizer input signal [6, 8]. Therefore, the feedback DAC pulse shape is important. It is this difference that makes CT modulators more sensitive to feedback DAC pulse characteristics, such as finite rise/fall time, jitter, etc.

Another difference is that CT modulators do not require an anti-alias filter, although in practice this is true only for narrowband signals [4]. According to Fig. 7.3a–c, and analyzing the system as shown in [8], it can be seen that the final STF results:

$$\mathrm{STF}(j2\pi f) = L_0(j2\pi f)\mathrm{NTF}\left(\mathrm{e}^{j2\pi f/f_s}\right) \tag{7.1}$$

In the frequency domain, the STF can be approximated by a $sinc^L$ ($\pi f/f_s$) function with nulls at multiples of the sampling frequency as shown in Fig. 7.3d. The attenuation achieved by the *sinc* function at the nulls approximate to $(2 * \mathrm{OSR})^L$, therefore providing an anti-aliasing feature of out-of-band noise and interference due to the sampling process [9]. However, a more detailed analysis [6] concludes by expressing the anti-aliasing properties of the model of Fig. 7.3c, such as:

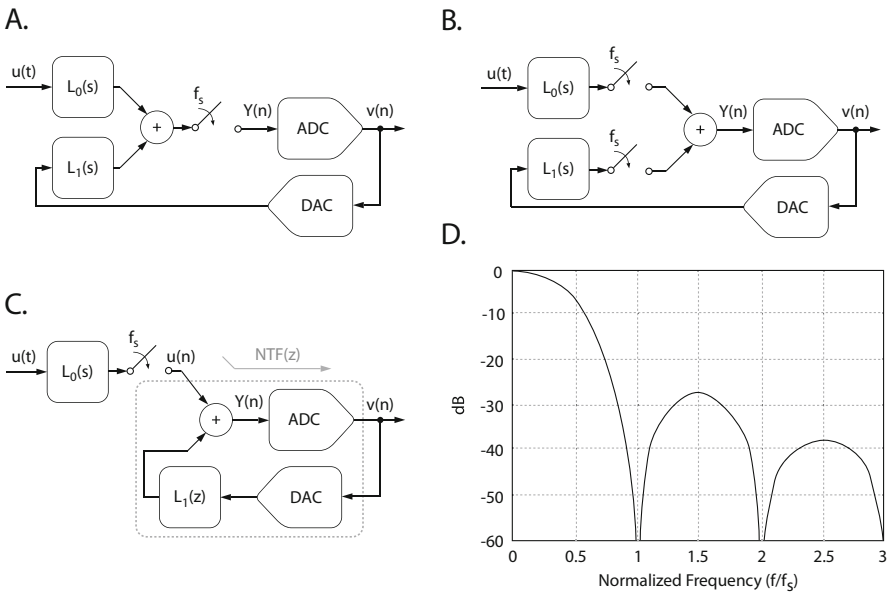


Fig. 7.3 Inherent AAF model of CT ΣΔ-modulators: (**a**) Block diagram of a generic CT modulator, (**b**) equivalent diagram of "**a**," and (**c**) equivalent diagram of "**b**" with a DT noise shaping loop. (**d**) STF frequency response of CT modulators

$$Y_s(\mathrm{OSR}f_s - f) = \frac{G(f)}{dG(\mathrm{OSR}f_s - f)}U(f) \tag{7.2}$$

where $Y_s(f)$ is the modulator output, $G(f)$ the loop filter function (i.e., low pass), and $U(f)$ the input signal. The frequency of interest is around the sampling frequency, such as $f \approx \mathrm{OSR}f_s$. Therefore, the numerator evaluates to a small value at high frequencies, while the denominator evaluates to a large number at low frequencies, resulting in a null around multiples of the sampling frequency. Thus, note that the worst aliasing occurs at the edge of the signal band, explaining why such anti-aliasing property is more effective for narrowband signals.

7.3 DT-CT Modulators Non-idealities

The most significant DT ΣΔ-modulator non-idealities [4, 7, 10, 11], grouped by the circuit portion which they affect (i.e., a switched-capacitor implementation is assumed), are summarized in Table 7.1. These errors, according to their effect on the modulator performance, can be distinguished in:

- *Errors that alters the STF and NTF* by changing their poles and zeros, such as finite amplifier DC gain (i.e., integrator leakage), capacitor mismatch, and

Table 7.1 DT ΣΔ-modulator non-idealities and their effect on the system

	Non-ideality	Effect
Integrators	Output swing	Stability, signal amplitude, noise floor, and distortion
	Finite GBW	Stability and noise floor
	Finite, nonlinear DC gain	Noise floor and distortion
	Finite slew rate	Noise floor and distortion
	Flicker (1/f) and thermal noise	Noise
Quantizer	Hysteresis, offset and gain error	Stability, signal amplitude, noise floor, and distortion
Capacitors	Mismatch and nonlinearity	Stability, noise floor, and distortion
Switches	Finite switch-on resistance, charge injection, clock feedthrough, nonlinearity	Stability, noise floor, and distortion
Multi-bit DAC	Offset, gain error, and nonlinearity	Stability, signal amplitude, noise floor, and distortion

Table 7.2 CT ΣΔ-modulator non-idealities and their effect on the system

	Non-ideality	Effect
Integrators	Finite, nonlinear DC gain	Noise floor and distortion
	Time constant error	Stability and noise floor
	Integration incomplete transient response	Noise floor and distortion
	Circuit element tolerances	Stability, signal amplitude, noise floor, and distortion
	Flicker (1/f) and thermal noise	Noise
Quantizer	Metastability and hysteresis	Signal amplitude, noise floor, and distortion
Feedback DAC	Delay (ELD)	Stability, signal amplitude, noise floor, and distortion
	Inter-symbol interference	
	Nonlinearity (i.e., multi-bit, etc.)	
Clock	Jitter	Noise and signal skirt

incomplete integrator (i.e., linear) settling error. As already seen, the effect of these errors strongly depends on the modulator design. For instance, MASH modulators are more sensitive to capacitor mismatch and finite DC gain than single-loop architectures.

- *Errors that introduce noise and distortion*, such as clock jitter, circuit noise (i.e., both thermal and flicker components), and harmonic distortion caused by the circuit nonlinearities. The errors of this group, generally, are those which can be modeled as additive noise sources at the modulator input; hence, they are not attenuated by the action of feedback.

Table 7.2 presents the case for CT modulators [2, 6, 8, 9], where circuit errors can be grouped into two main categories:

- *Building-block errors*, which typically alter the STF and NTF similarly to the DT case. These may include finite op-amp DC gain, integrator time constant error, integration incomplete transient response, circuit element tolerances, circuit noise, etc.
- *Architectural timing errors* such as quantizer metastability, excess loop delay (ELD), inter-symbol interference (ISI), clock jitter, etc., which affect noise and distortion in the system, as well as its stability.

One fundamental difference between the non-idealities of DT and CT ΣΔ-design is that CT modulators are sensitive to timing errors, and these represent a crucial part of a successful converter design. Further, the integrator circuital realization results more significant in CT designs, since many solutions exist. On this regard, it should be noted that DT switched-capacitor integrators exhibit inherently lower parameter variation since most circuit parameters are defined by capacitor ratios instead of absolute components values as in the case of CT circuits. Thus, although CT modulators are often designed and simulated in the DT domain, it should result clear that these are inherently different from their DT counterparts. For this reason, the next paragraphs introduce some of the most common CT modulator circuits and investigate in detail the most significant CT non-idealities. Further, suggestions to aid correct simulations of those are provided.

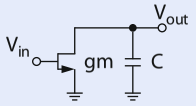
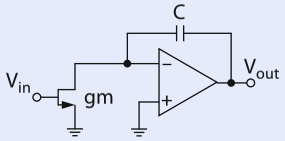
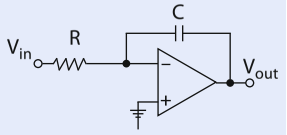
7.4 Integrators

Different implementations of CT loop filter integrators exist [2, 3, 6, 8], and Table 7.3 provides a summary of the most common as well as their main characteristics.

- *Gm-C:* A capacitor is fed by a voltage-controlled current produced by a transistor. Since no op-amp is required, this configuration allows for low power consumption and good adjustability of the gain by calibration of the transistor transconductance *gm*. However, since no feedback regulation is included, the linearity results poor.
- *Active gm-C:* By placing the capacitor in the op-amp feedback path, better linearity is achieved as the current is fed into a node with constant potential (i.e., virtual ground). Further, the parasitic capacitance at the output of the transistor is suppressed, since the voltage across it is constant. However, some nonlinearity persists, as the current is generated by a transistor with nonlinear *V/I* characteristic. Moreover, power consumption results increased due to the op-amp.
- *Active RC:* The *V/I* conversion is performed by a resistor located between the input and the virtual ground of an op-amp; hence, high linearity is achieved.

Important to note in the design of CT modulators is that since the first integrator of the loop filter typically dominates the overall converter performance, as in the DT

Table 7.3 Filter implementations comparison

	gm-C	Active gm-C	Active RC
			
Frequency range	Highest	High	High
Tunability	High	High	Low
Mismatch insensitivity	High	Medium	Low
Linearity	Low	High	Highest
Dynamic range	Medium	Low	High
Power consumption	Low	High	High
Low voltage applicability	Low	Medium	High

case [4, 6], it is common to encounter CT loop filters formed by a mixture of integrator arrangements. This is because after the first integration stage, the linearity requirements are typically relaxed. This allows for more power efficient solutions, such as the *gm-C* integrator, to be implemented after the first integration stage in order to achieve optimal system efficiency. Due to the introductory nature of this book, however, the focus of the discussion is only on the *active RC* integrators since these generally provide the best linearity as well as highest dynamic and frequency range, hence resulting in a very common choice for the critical first integration stage of modulators. However, given the importance of all the various CT integrator circuital solutions, the interested reader is advised to investigate [2, 3, 6, 8, 9] in order to improve the knowledge on the topic and especially on how the non-idealities have an effect on each circuital solution as well as their respective high-level models [5, 6, 8, 12].

7.4.1 Active RC Integrator: Amplifier's Finite DC Gain and GBW

The ideal transfer function of the active RC integrator is [2, 6]:

$$\mathrm{Int}(s) = \frac{1}{s\mathrm{RC}} = k_i \frac{f_\mathrm{s}}{s} \tag{7.3}$$

where f_s is the sampling frequency and k_i is the scaling coefficient.

Important to note is that in the case of finite DC gain (i.e., no ideal virtual ground condition), the input current does not flow only to the integration capacitor, which results in an alteration of the NTF zeros placement and hence in an increase of in-band noise due to the pole of the integrator shifting away from the ideal DC value. According to [6], as long as the DC gain is greater or equal to the OSR, the SNR would reduce approximately by only 1 dB. Therefore, a rule of thumb in the design of single-loop modulators is to ensure an amplifier gain of $A_\mathrm{DC} \approx \mathrm{OSR}$ in order to avoid SNR degradation [2, 6, 8, 9]. For the more advanced designer, however, it should be noted that the optimal DC gain specification would better to be found through behavioral simulations since other aspects influence its value (i.e., NTF aggressiveness, etc.).

Another consideration for an active RC integrator is the effect of finite GBW [8, 13] as it introduces non-dominant poles in the NTF, resulting in gain and stability margin reduction. The introduction of the non-dominant poles can be modeled as a delay and as such compensated for with the techniques used for excess loop delay (more on this later). Note that the smaller the GBW, the larger the delay and such effect is also dependent on the amplitude of the output (i.e., the larger the voltage output swing, the larger the delay). Therefore, the prediction and compensations of GBW effects on the overall performance result are difficult, and typically the critical bound is found through extensive behavioral simulation [6, 8, 13]. Important to note, however, is that the GBW requirements in CT modulators are generally much lower than the DT counterparts (e.g., GBW $\approx 4F_\mathrm{s}$), and in [13] it has been demonstrated

that for a high-order CT modulator with a GBW of approximately F_s minimal degradation results in terms of overall performance, assuming a satisfactory high OSR capable of maintaining linear operation of in-band signals is implemented.

Therefore, as a guideline [2, 6, 8, 9, 13], a GBW of at least $2F_s$ is generally suggested for the initial design in order to avoid major issues caused by parasitic capacitances, process, and temperature variations.

7.4.2 Active RC Integrator: Coefficients Variation

Another major issue regarding the design of active RC loop filter integrators for CT modulators is the disadvantage that coefficients are determined by the RC product, which is significantly and inevitably affected by CMOS process variations [1]. In modern CMOS technology, matching accuracy and process spread of resistors and capacitors are relatively poor, and variations of about 30% in RC time constant can be expected [2, 6, 8]. This greatly differs from DT modulators [4, 10], where coefficients rely on capacitor ratio which results rather accurate (i.e., $\approx \pm$ 1% tolerance).

According to Eq. 7.3 of the previous paragraph, it is easily seen that variations in the R or C values inevitably result in variations of the scaling coefficients k_i and consequently in an alteration of the NTF of the modulator. It should be noted that a too aggressive NTF leads to modulator instability, while a too gentle noise shaping limits the modulator's ability to achieve a high SNR [4]. Depending on the interaction between the several integrator stages of the modulator, as well as on the IC process variations and circuit layout, the coefficients of individual stages may shift together (e.g., worst case – no averaging effect, resulting in maximum loop gain deviation) or randomly. Positive increments of the coefficients cause the swing of internal nodes to increase accordingly, which may lead to clipping and distortion [2, 6]. From a design perspective [1, 2, 4, 6, 8], for a given technology if the component variation range is predictable, then the coefficients can be scaled accordingly in order to ensure that over worst cases instability is avoided. In the case of a too stringent tolerance range, reducing the out-of-band gain of the NTF may be the only option to ensure stability at the expenses of a poorer SNR.

A rule of thumb to ensure stability in ΣΔ-modulators is to use Lee's criterion (e.g., which sets the maximum OBG to $|\mathrm{NTF}_{\mathrm{MAX}}| \approx 1.5$); however, it should be noted that this method is empirical and derived from experience [4]. Although it can be used as a starting point when designing the high-level modulator model, extensive simulations are required in order to ensure correct modulator performance under worst-case conditions and process variations, and, in the case of instability, a new NTF would have to be developed. If both stability and performance are important and a satisfactory trade-off between the OBG and SNR performance cannot be found, then coefficient tuning may be implemented. In active RC integrators, this is commonly achieved by capacitor array or resistor ladders tuning techniques. More complex and accurate tuning methods also exist [1, 2, 9]; however, it should be noted

that any tuning solution increases circuit complexity, area, cost, and power consumption. Therefore, they are undesirable.

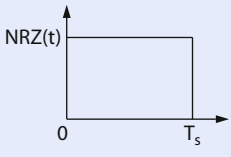
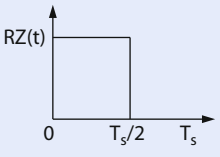
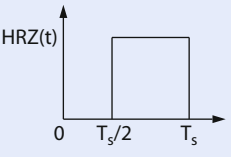
It should be clear that in any CT design, the coefficient variations have to be carefully simulated since they greatly affect the modulator performance. This can be easily done in the Toolbox by varying the value of the ABGC matrix which maps to the Simulink® models. For a practical example, please refer to the CT design example in Chap. 9.

7.5 Feedback DAC

In CT modulators both the shape and timing information of the feedback DAC pulses are significant, as these directly affect the system performance [6, 8]. Note that, since the DAC pulses are continuously being integrated, they cannot be treated as digital signals as in the case of DT modulators, where the actual shape of the feedback pulse does not affect the overall system performance, and only the final settling accuracy of the voltage in the first integrator is of interest. Common types of DAC pulses implemented in ΣΔ-modulators include [1, 3, 6, 8, 9] non-return-to-zero (NRZ), return-to-zero (RZ), and half-delayed return-to-zero (HRZ). The three DAC pulses and their Laplace transform are shown in Table 7.4.

NRZ pulses allow for a greater charge to be fed back at each cycle compared to the RZ and HRZ type, resulting in a greater input signal dynamic range and a more power-efficient DAC implementation while being more robust toward jitter. For RZ and HRZ implementations, the input signal needs to be scaled down, in order to maintain modulator stability, and this results in a reduction of the dynamic range. However, the advantage of using RZ or HRZ pulses is that they allow to partially solve some of the timing-related non-idealities affecting NRZ pulses. Additionally, RZ pulses allow being pulse-shaped [2, 3, 9] which is useful in order to reduce jitter noise, as explained later. In the next paragraphs, the CT non-idealities relative to these types of pulses are discussed and solutions to aid the simulation of those presented.

Table 7.4 DAC pulses and their s-domain responses

	NRZ	RZ	HRZ
$[\alpha\ \beta]$	[0 1]	[0 0.5]	[0.5 1]
Impulse response			
Transfer function	$R_{NRZ}(s) = \frac{1-\exp(-T_s s)}{s}$	$R_{RZ}(s) = \frac{1-\exp(-T_s s/2)}{s}$	$R_{HRZ}(s) = \frac{\exp\left(-\frac{T_s s}{2}\right)-\exp(-T_s s)}{s}$

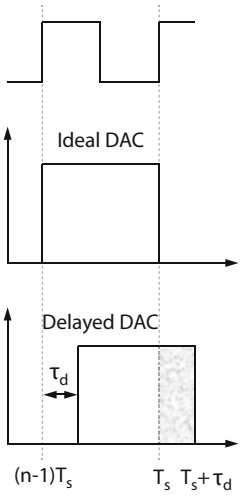


Fig. 7.4 Ideal, delayed, and linear combination of the NRZ pulse

7.5.1 Excess Loop Delay

The excess loop delay (ELD) is the delay occurring between the quantizer sampling instant and the feedback DAC output [4, 6, 14]. Ideally, the DAC should provide the feedback as soon as the input has been sampled, but in practice, a delay is inevitable due to the switching time of the transistors and the speed of the quantizer. Further, in the case of a multi-bit design, the DEM circuits required may add further delay. Figure 7.4 illustrates the ideal (i.e., $\alpha = 0$, and $\beta = 1$) and the delayed NRZ (i.e., $\alpha = \tau d$, and $\beta = 1 + \tau d$) pulse, where τ_d represents the excess loop delay.

It should be noted that in order to conduct the transformation of the DT-NTF into a CT-NTF during the high-level design of the modulator, a specific DAC pulse has to be specified; therefore, any deviation from this pulse in practical realizations causes ELD, which consequently creates a mismatch between the intended and actual noise transfer function [4, 6].

To analyze the effect of the ELD, the delayed NRZ pulse is modeled as the sum of two pulses, the one in the current clock period and the one extending to the next clock cycle (Fig. 7.4). According to [4, 6, 14], the pulse portion extending into the next clock period is equivalent to increasing the modulator order by one. Such increase of order pushes more quantization noise out of the signal band, resulting in an increase of out-of-band peaking. This has the effect of increasing the noise shaping performance at first, but then the achievable SNR results reduced due to the decreased stability margin of the loop which eventually may lead to instability. This phenomenon affects the HRZ DAC pulse as well, while for the RZ pulse, the sensitivity to ELD is greatly reduced since no pulse extends to the next clock period as long as the delay is not too large. However, even for RZ pulses, it should be noted that ELD degrades the noise performance of the modulator, but this can be restored by adjusting the filter coefficients in order to obtain the desired NTF.

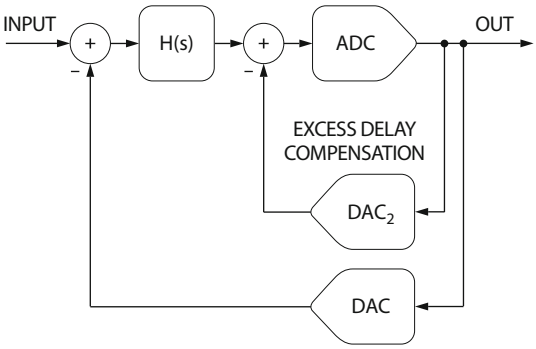


Fig. 7.5 Compensation of ELD effect with direct feedback method

A large variety of techniques can be found in the literature to counteract ELD [1, 2, 4, 6, 8, 9, 14] and some of the most common include the following:

- The use of a less sensitive pulse.
- The implementation of less sensitive ΣΔ-architectures (e.g., DT-CT hybrid modulators, where typically being the last integrator of the loop a DT one – the one before the quantizer – any delay that may have been created by the loop is restored to the right clock timing [3], etc.).
- The use of wideband op-amps, which have the disadvantage of increasing power consumption.
- The use of an additional feedback path directly connected to the quantizer input, often called 0th order path, in order to cancel the extra coefficient introduced by ELD in the NTF (Fig. 7.5). Important to note is that such technique is able to provide compensation only up to one clock cycle. Further, it has the disadvantage that an extra DAC and summation circuitry are needed, resulting in increased power consumption and area. However, this disadvantage can be mitigated if certain design strategies are adopted, such as the use of switched current feedback and current summation rather than conventional voltage summation, which allows avoiding a summing amplifier [2, 10].

The effect of ELD on a specific design can be analyzed at high-level by both, introducing a delay block in the feedback loop of the Simulink® model or by specifying the expected delay amount in the Schreier's Toolbox, which automatically computes the extra coefficient needed for compensation using an extra feedback path [4, 12].

Example

Suppose that in a modulator a 12-bit DR in a 50 MHz bandwidth is desired. Assuming an OSR of about 50, an $f_s = 5$ GHz, and a number of transistors in the system feedback loop of 2, find the excess loop delay in a 30 GHz process.

Solution

It can be assumed that ELD can be expressed as a fraction of the sampling period, such as:

$$\tau_D = \rho_D T_s$$

shown in Fig. 7.4 in the main text of the book for an ideal NRZ DAC pulse. The ELD τ_D depends on the switching speed f_T of the transistors, quantizer clock frequency f_s, the number of transistors in the system feedback loop n_i, and the loads on each transistor. For a rough approximation, we can assume that all transistors switch after $1/f_T$, which is:

$$\rho_D = \frac{n_i f_s}{f_T}$$

Therefore, using the data of the exercise, it results:

$$\rho_D = \frac{n_i f_s}{f_T} = \frac{2*5}{30} = 33\%$$

which is the amount of excess delay predicted.

7.5.2 Inter-symbol Interference

The asymmetric rise/fall time that may occur during switching transients creates a dynamic error known as inter-symbol interference (ISI). This error is signal-dependent and causes even harmonic distortion tones to appear in the output spectrum, which degrades the modulator SNR performance, especially in high-speed systems [8, 9]. Regarding circuital implementations of the modulator, one thing that should be considered is that, generally, at the output of the DAC, spikes are inevitable during pulse transitions due to the fast pulse transition and parasitic coupling [2, 3]. These spikes have a similar effect on the modulator performance to that of ISI phenomena and also cause a finite amount of signal-dependent error charge which results in pulse degradation [1].

To appreciate the effect of asymmetric pulses, the two NRZ sequences in Fig. 7.6a can be considered, where it is assumed that the fall time of the DAC signal is zero and the rise time is t_r.

As it can be seen, the top NRZ sequence corresponds to "1, 1, −1, −1" and the other to "−1, 1, −1, 1." Ideally, the area of the two sequences should sum up to zero for the four clock periods, since both have two +1 s and two −1 s. However, due to the unequal rise/fall time, the total energies of the two DAC feedbacks are different. Specifically for the top signal, the positive energy account for $2T_s - t_r + \frac{t_r}{4} = 2T_s - \frac{3t_r}{4}$, while the negative is $2T_s + \frac{t_r}{4}$, resulting in a total energy

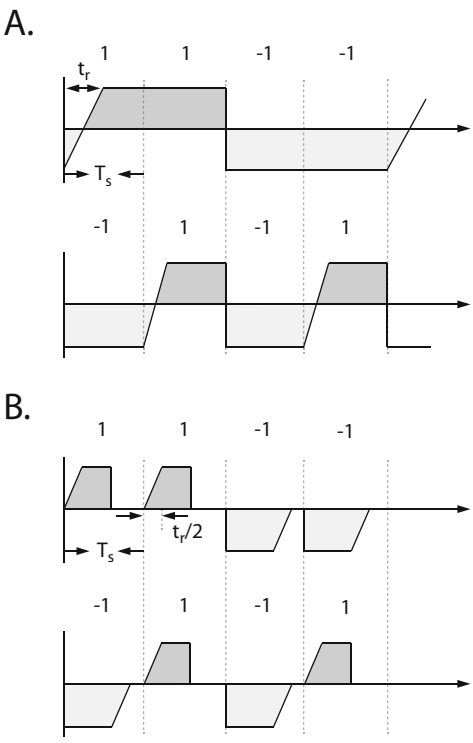


Fig. 7.6 (**a**) NRZ pulse sequence and (**b**) respective RZ pulse sequence of "a"

of $-t_r$. Analyzing the bottom signal in the same way, it results that the total energy is $2t_r$. Therefore, due to the unequal rise/fall times, the NRZ feedback energy deviates the ideal case and becomes signal-dependent. In other words, the NRZ feedback energy not only depends on the numbers of +1 s and −1 s but also on how those +1 s and −1 s switch. This has the effect of increasing the noise floor and harmonic distortion, as already mentioned. Important to note is that RZ and HRZ pulses are much less affected by this non-ideality. Figure 7.6b illustrates the RZ responses for the same sequences considered in the NRZ case. As it can be seen, RZ feedback pulses return to the zero level during each clock cycle, thus ensuring that the transition behavior is identical in each cycle and eliminating any pulse dependency on the input signal (e.g., no ISI) while introducing only a gain error to the ΣΔ-modulator. Despite the solution to use RZ/HRZ pulses to avoid the effect of unequal rise/fall time of the feedback DAC pulse, another solution is to use a differential circuitry which can produce symmetrical DAC pulses [7, 8]. Note that the maximum allowable asymmetry for a given sampling period, T_s, can be approximated according to [6, 8, 9]:

$$\tau \leq \frac{4T_s\sqrt{\text{OSR}}}{\text{SNR}_{\text{desired}}} \tag{7.4}$$

To study the effect of the unequal rise/fall time in a modulator at high level, a rate limiter block after the DAC can be used, which allows changing the feedback pulse area [5, 12].

Example

Assuming a maximum allowed asymmetry of 0.177 ns of a modulator with OSR = 32 and a frequency band of 24 KHz, calculate the achievable SNR.

Solution

The first step involves calculating the sampling period which is:

$$T_s = \frac{1}{24\ \text{KHz} \cdot 2 \cdot 32}$$

Important to note is that the frequency band has to be multiplied in order to achieve the Nyquist frequency. The next step is to rearrange Eq.7.4 in order to find the SNR, such as:

$$\text{SNR} = 20\log\left(\frac{4T_s\sqrt{\text{OSR}}}{\tau}\right) = 20\log\left(\frac{4\left(\frac{1}{24000 \cdot 2 \cdot 32}\right) \cdot \sqrt{32}}{0.177 \cdot 10^{-9}}\right) \approx 98.4\ \text{dB}$$

7.6 Clock Jitter

In an IC, clock signals [1, 2, 15] (i.e., usually generated by phase-locked loop (PLL) or voltage-controlled oscillator (VCO) circuits) are affected by jitter, which is the random variation in pulse position and width of clock transition edges caused by the various noise sources of the clock generating circuitry (e.g., thermal noise, power supply noise, etc.). The critical jitter-sensitive parts of a ΣΔ-modulator are the DAC output [16] and, in DT implementations, the input sample and hold circuit [4]. In all the other internal nodes in which sampling occurs, jitter is not of great concern, as these points are inside the loop and, as such, errors are suppressed by the high gain of the preceding blocks [6, 10]. Either with a normal distribution or appearing as accumulated jitter when the clock signal is VCO-generated [2], the variation due to sampling jitter affects the exact time a transition occurs in the DAC waveform, resulting in an incorrect amount of charge provided to the loop filter. Such phenomenon has a similar effect of that of the unequal rise/fall times previously described, in the sense that it only occurs at transition time. However, since it is randomly affecting the DAC output, no signal dependency is introduced in the feedback loop, and as such, no harmonic distortion is introduced, although a reduction of the dynamic range (DR) occurs [16]. The DR is reduced by both the leakage of the quantization noise inside the signal bandwidth caused by the modulation of the NTF with the random jitter (i.e., which appears as an additional quantization error to the modulator, and since quantization noise is shaped by the loop, its effect can often be neglected) and by the presence of the jitter noise power inside the signal bandwidth

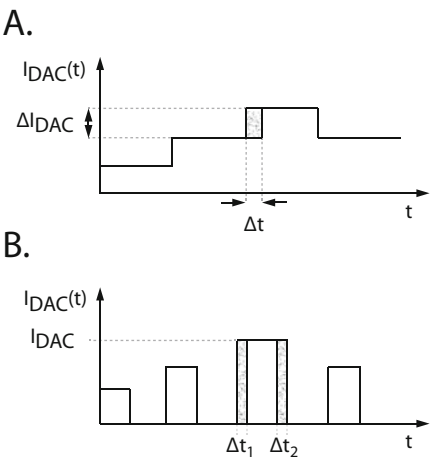


Fig. 7.7 Qualitative waveforms of DAC currents. (**a**) NRZ continuous time, (**b**) RZ continuous time

introduced at the DAC connection to the first integrator. The errors introduced by the latter case can be seen as a modulation of the feedback coefficients, resulting in a flatter spectrum of the in-band quantization noise, which translates in a poorer resolution [2, 6]. Since the jitter affects the system through modulation on the DAC pulse, it is apparent that the actual shape of the pulse would have a direct impact on the final modulator performance. Moreover, the performance bounded by jitter is actually determined by the quantity, which is the clock jitter normalized to clock period, explaining the reason why high-speed systems are more affected by this non-ideality [8, 9].

Figure 7.7 illustrates the NRZ and RZ pulses affected by jitter with standard deviation σ_j and zero mean. The shaded edges represent the random time uncertainty of the transition. Assuming the two waveforms are applied to the same modulator, note that the RZ waveform output value has to be doubled (i.e., considering a 50% duty cycle) to supply the same amount of charge of the respective NRZ pulse [1, 6, 17]. Comparing the two pulses, it can be noted that an NRZ pulse is affected by this random effect only when a transition occurs (i.e., which introduces a signal dependency of the jitter power in the DAC output). On the other hand, the RZ pulse has four times more jitter power due to two transitions happening in one clock cycle and the output power being doubled during the active phase. Therefore, it can be concluded that the loss of DR is signal-dependent for NRZ DACs [6], while it is signal-independent but 6 dB larger for RZ DACs. Further, in the case of an NRZ DAC, the integral charge error Q_j can be derived as [6, 8]:

$$Q_j = \Delta I_{\mathrm{DAC}} \Delta t = \pm k \frac{I_{\mathrm{FS}}}{2^N} \Delta t \tag{7.5}$$

where I_{FS} represents the full-scale DAC current, N the number of bits of the quantizer, Δt the time error, and k the current variation ΔI_{DAC} as an integer multiple of the unit current of the least significant bit. According to Eq. 7.5, the charge error introduced by the same time uncertainty Δt is halved for each additional bit, explaining why multi-bit modulators are less affected by jitter.

Although NRZ pulses perform better than RZ ones in both single-bit and multi-bit quantizer cases, it should be noted that these do not allow for pulse-shaping techniques to be applied, making the modulator jitter performance entirely dependent on the quality of the clock [1, 2], which usually translates into a more complex and expensive clock circuit design. On the other hand, RZ pulses allow to be pulse-shaped and may result in a cheaper solution compared to high-quality clock jitter circuitry [2, 9].

Pulse-shaping techniques are based on spreading the feedback pulse over several clock cycles, which consequently reduce the increase of in-band noise caused by jitter. It is shown in [6, 17] that such techniques allow single-bit CT modulators to achieve the same jitter performance of multi-bit implementations. The downsides of these techniques are the increase of both circuit complexity and power consumption, especially if the DR figure has to be preserved. Due to the introductory nature of this work, it is impractical to provide a detailed description and comparison of all the methods that can be used to minimize the jitter effect; however, a summary of some of the most common methods used and their main characteristics have been reported in Table 7.5.

Table 7.5 Jitter reduction techniques comparison

	Performance	Realization complexity	Advantages	Disadvantages
NTF design [6, 8]	Low	Simple	No extra circuitry	• Only in multi-bit design • Complicated NTF design • NTF peaking
Switched capacitor-resistor DAC [18]	High	Simple	Simple implementation	• Reduced dynamic range • High power consumption
Switched capacitor-switched resistor [19]	High	Complex	Retained dynamic range	• Large circuit overhead
Sine-shaped pulse [20]	High	Complex	Retained dynamic range	• High implementation cost
PWM-FIR [21]	Medium	Complex	Retained dynamic range	• Limited jitter rejection • High implementation cost
FIR [22, 23]	Medium	Complex	Retained dynamic range	• Limited jitter rejection • High implementation cost
Fixed length RZ [24]	High	Medium	Low cost, good performance	• Rely on RC time constant • Not verified on chip level

7.7 Continuous Time NTF Design

As already discussed, the suggested CT design approach resembles that of DT modulators, and, as such, the guidelines presented in Chap. 4, Sects. 4.6, 4.7, and 4.8, still apply. Therefore, the procedure reported below can be used to design CT modulators in the Toolbox.

Type – *load_par* – in the MATLAB Command Window to reset the default variables.

Type – *mex simulateDSM.c* – in the MATLAB Command Window. This routine is required by the Schreier's Toolbox to function more efficiently.

Open – *MakeModulator_CT.m* – and enter the following variables in the – *Step [1]*- portion of the code:

order = 3;
OSR = 64;
form = 'FF';
nLev = 2;
OBG = NaN;
opt = 1;
tdac = [0 1];

Save the code and type – *MakeModulator_CT* – in the MATLAB Command Window.

As it can be noted, the only difference with a DT design is that the DAC pulse has to be specified (i.e., *tdac,* where *[0 1]* represents an NRZ pulse – see Appendix B for timing descriptions). Once the code has been computed, the a, g, b, and c coefficients should be displayed in the MATLAB Command Window. These coefficients correspond directly to the multiplier gains of the relative DT models found in the *4_MOD3 - > Schreier's Models* directory (i.e., the one to use in this example is *mod3_CIFF_CRFF_1bit*), which can be used for simulations. Note that the coefficients computed – *MakeModulator_CT* – are derived by exploiting the equivalence between DT and CT loop filters, and thus they may differ from what – *MakeModulator* (i.e., as used in DT designs of previous chapters) – may compute. Once the code has been run, the poles and zeros of the NTF designed, the STF and NTF transfer curves, as well as the mapping of the impulse response and relative NTF should be displayed. These plots should be used to understand whether the NTF synthesized corresponds to what the designer aimed for or not, and the reader should be familiar with the results at this point of the book.

To conclude the high-level design, the reader should perform the simulations presented throughout the book on the respective Simulink® model, in order to investigate the behavior of the modulator designed. Important to note is that the

specific non-idealities of CT modulators have to be included. For a practical step-by-step example, please refer to the CT design example in Chap. 9.

Example

Simulate the CRFF modulator designed in Simulink®. Further, simulate the effects of jitter using the – NRZ_DAC – found in the folder – 7_CT_MOD – of the Toolbox and briefly discuss the results.

Solution

Open the Simulink® model – sweep_testbench – right click on the – Modulator – block and select the – Model Reference Parameter. Then select the model – mod3_CIFF_CRFF_1bit. Type – simu.select = 2; – in the MATLAB Command Window in order to select a sinewave input. Run the simulation and, once completed, type – mod_SNDR – in order to obtain the graph illustrated in Fig. 7.8.

Important to verify is that the SNDR achieved by the Simulink® model is close to what the Schreier's Toolbox predicted. To simulate the effects of jitter, enter in the modulator's model and add the – NRZ_DAC – block in the feedback path. Simulate the modulator again and note that the same results should be achieved. The final step involves entering in the – NRZ_DAC – block and change the – Gain – block following the – Random Number – block to – 1. This introduces the jitter error in the model as an additive white noise source. Simulating the modulator with default settings should give the results illustrated in Fig. 7.9.

As it can be seen, jitter greatly affects the achievable SNDR. Important to note is the rise in noise floor due to jitter.

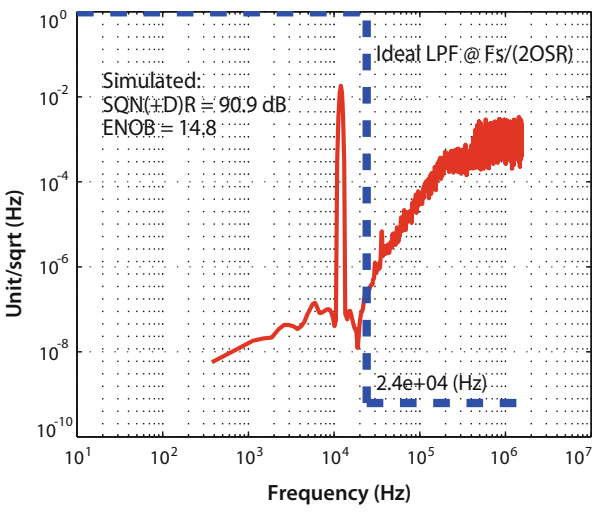


Fig. 7.8 SNR simulated

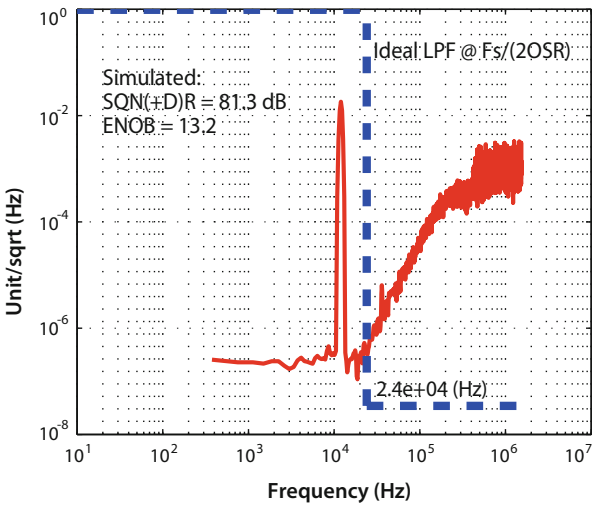


Fig. 7.9 SNR simulated when jitter is included

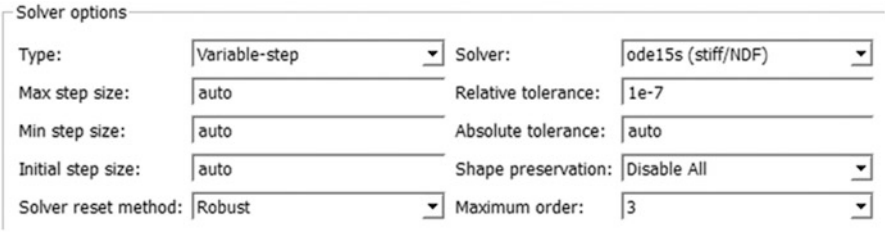


Fig. 7.10 Simulink® solver settings example

7.7.1 Additional Simulink® Simulation Information

Besides implementing the modulator by strictly using discrete time Simulink® operational blocks, there is also the possibility to use continuous time ones. Important to note is that in Simulink®, the modulator still operates as a hybrid CT-DT system, where the loop filter is solved in the Laplace domain and the quantizer and feedback branch in the z-domain. Further, note that choosing the correct solver engine is essential for both achieving precise numerical results and maximizing the model computational efficiency. In ΣΔ-modulator systems, a high discrepancy between the various time constants can lead to rapid variations in the model's performance results. Therefore, the first criterion for a correct solver choice is the average order of the system. If it largely employs integrators and high-order transfer functions, a variable-step solver should be chosen to observe hidden dynamics that may arise in that kind of environment. Furthermore, note that implicit differential solvers are recommended to achieve convergence. Generally, it is suggested employing the *"ode15s"* solver as shown in Figs. 7.8 and 7.10.

Note that a relative error tolerance of *1e-7* can be used in common simulations except in the presence of anomalous responses, where a lower tolerance has then to be used (i.e., important to check the output of the integrator to see how small, or big, the signal is. Then the tolerance should be adjusted accordingly). A third order should be enough to obtain satisfactory results along with a – *robust* – solver reset method. Further, it should be noted that the model behavior heavily relies on the variable-step feature. Hence, it is the task of the engine to accurately solve dynamic changes at the expense of higher computational cost. Moreover, note that the response of saturation blocks used in the integrators, quantizer, etc. depends mainly on the zero-crossing detection algorithm, which detects discontinuities in rapidly changing states of the system. In certain cases, a glitch at its solving routines may provoke abnormally elevated initial conditions on the "sample and hold" blocks that immediately would overload the modulator. Thus, it is fundamental to set the outputs initial values of sensitive blocks in order to avoid an incorrect performance.

To conclude, for the nonexpert user of MATLAB/Simulink®, it is advised to proceed with the DT approach explained earlier when designing CT modulators. On the other hand, the more advanced user is encouraged to experiment with actual CT systems in order to more accurately investigate modulator's behavior, assuming appropriate simulations settings are used.

Example

Design and simulate a first order modulator in Simulink® by using continuous time blocks.

Solution

Open the Simulink® model – CT_mod1 – provided in the Toolbox folder – 7_CT_MOD. Type – load_par – into the MATLAB Command Window. In the Simulink® model go to – Simulation – in the top bar and select – Configuration Parameters. Ensure that the settings discussed in this paragraph are used. Then, run the simulation and inspect the time domain results (i.e., scope) and the SNDR achieved (i.e., by typing – mod_SNDR – into the MATLAB Command Window). It should result that the modulator behaves as expected and for a 12 KHz, −3 dBFS input sinewave achieves an SNDR of approximately 54 dB.

7.8 Conclusions and Essential Takeaways

In this chapter, an introduction to continuous time ΣΔ-modulators has been provided, and the differences with the discrete time implementations have been discussed. Such differences have been summarized in Table 7.6 [4, 6, 8, 10]. Further, specific aspects relative to the design process of CT modulators have been investigated and some common circuital solutions presented. A summary of the non-idealities studied and the actions which may be taken to counteract their effects on the conversion process is presented in Table 7.7 [1, 2, 4, 6, 8].

Table 7.6 Comparison of CT and DT ΣΔ-modulators

	Continuous time	Discrete time
Sampling frequency	Potentially higher operation speed achievable	Lower
Power consumption	Lower	Higher
Anti-aliasing	Not required – inherent for narrowband applications	Required before the ADC
Sampling error	Shaped by the loop filter	Appears directly at the ADC output
Clock jitter	Sensitive to clock jitter in feedback DAC	Robust to clock jitter
Loop delay	Sensitive	Very little effect
Mismatch sensitivity	High – need additional calibration circuitry	Low – rely on precise capacitor ratios
CMOS compatibility	Low	High
Simulation/ prototype	Difficult/easy to prototype	Easy/difficult to prototype
Other advantages	SNR not limited by capacitor size Reduced op-amp speed requirements Reduced impact from supply and ground noise Less glitch and switching noise	Accurate transfer function since pole-zero locations is set by accurate capacitor ratios Switched capacitor integrator highly linear Only capacitive loads
Other disadvantages	Requires accurate RC time constants Require highly linear op-amps	Large capacitors for high SNR required

Table 7.7 Non-idealities affecting CT ΣΔ-modulators and common solutions

Non-ideality	Impacts	Solution
Overload	• More noise • Distortion • Unstable system	• Limited input range • Time constant scaling • Multi-bit quantizer Reset circuitry
Excess loop delay	• More noise • Unstable system	• Use simple circuits (e.g., 1-bit Quantizer, no DEM, etc.) • Use RZ DAC • Use compensation techniques
Clock jitter	• More noise	• Accurate clock source • Multi-bit quantizer • Jitter-insensitive DAC pulse shapes Jitter-insensitive ΣΔ-architectures (e.g., hybrid, etc.)
Unequal DAC rise/ fall time	• More noise • Distortion	• Use RZ/HRZ DAC pulses • Differential circuitry
Amplifier finite gain/GBW	• More noise • Distortion	• Design amplifier with enough gain and GBW
Amplifier finite slew rate	• More noise • Distortion	• Design amplifier with enough slew rate
Time constant variation	• Unstable system • Poorer performance	• Trimming • Resistance/capacitance tuning
Quantizer hysteresis	• More noise	• Reset circuitry

Exercises

Q.1
Why might we choose to employ a CT filter inside a modulator loop rather than a DT filter? Briefly discuss your statements.

Q.2
Design a second-order CIFB modulator having an OSR = 64, single-bit quantizer, and a frequency band of operation of 16 KHz. Simulate the modulator in Simulink®.

Q.3
Using the design of Q.2, simulate the effects of jitter using an NRZ DAC in Simulink®. Discuss the results.

Q.4
Using Simulink®, check that the – *RZ_DAC* – model found in the folder – *7_CT_MOD* – provides an RZ pulse. What can be said about the input and output of the model in relation to the RZ pulse (i.e., Hint: enter into the model to inspect the blocks arrangement)? What are the advantages and disadvantages of using such feedback DAC pulse type?

Q.5
Using the design of Q.2, simulate the effects of jitter using an RZ DAC in Simulink®. Discuss the results.

Q.6
Discuss the advantages and disadvantages of the gm-C, active gm-C, and active RC integrators.

Q.7
Explain the phenomenon of excess loop delay. Why excess loop delay is problematic only in CT ΣΔ-modulators although it exists also in DT designs?

Q.8
Considering a high-order modulator with a sampling period $T_s = 163\ \mu s$, OSR = 150, and a resolution of 16-bit, find the maximum asymmetry allowed in the feedback DAC pulse to avoid ISI.

Q.9
In order to design ΣΔ-modulators using continuous time blocks in Simulink®, it is necessary to transform the DT transfer function into a continuous time one (i.e., note that the coefficients derived by the Schreier's Toolbox with the method discussed would not always work with continuous time blocks!). One method mentioned in the book is the impulse invariant transform. Assuming an ideal NRZ feedback DAC

pulse is used, the shape of the DAC pulse is perfectly a rectangular pulse of magnitude 1 that lasts from α to β:

$$\widehat{r_{\mathrm{D}}} = \begin{cases} 1, & \alpha \le t \le \beta, \quad 0 \le \alpha \le \beta \le 1 \\ 0, \text{otherwise} \end{cases} \tag{7.6}$$

Using Laplace transform, the equation above can be expressed as:

$$\widehat{R}_{(\alpha,\beta)} = \frac{\exp(-\alpha s) - \exp(-\beta s)}{s} \tag{7.7}$$

What is found is a z-domain pole of multiplicity l at z_k which maps to one at s_k with the same multiplicity, such as:

$$s_k = \ln z_k \tag{7.8}$$

(this makes sense when you think of $z = \exp(sT_s)$). Poles at dc (i.e., $z_k = 1$) end up giving $0^l/0^l$ as the numerator of the s-domain equivalent, which necessitates l applications of l'Hôpital's rule; this has been done in the right column of Table 7.7.

For a second-order low-pass design of a DT ΣΔ-ADCs, the NTF(z) $= (z-1)^2$ such as:

$$H(z) = \frac{-2z+1}{(z-1)^2} \tag{7.9}$$

To obtain the CT equivalent, $H(z)$ is written as a partial fraction expansion first, such as:

$$H(z) = \frac{-2}{z-1} + \frac{-1}{(z-1)^2} \tag{7.10}$$

Thus $z_k = 1$, which means $s_k = 0$ from Eq. 7.8. Applying the first row of Table 7.8 to the first term of Eq. 7.10 and the second row to the second term with $(\alpha, \beta) = (0, 1)$ gives:

Table 7.8 s-domain equivalences for z-domain loop filter poles

Limit for $z_k = 1$	$\frac{r_0}{s-s_k}$ $r_0 = \frac{y0}{\beta-\alpha}$	$\frac{r_1 s + r_0}{(s-s_k)^2}$ $r_1 = \frac{1}{2}\frac{(\alpha+\beta-2)y0}{\beta-\alpha}$ $r_0 = \frac{y0}{\beta-\alpha}$
s-domain equivalent	$\frac{r_0}{s-s_k} \cdot \frac{y0}{z_k^{1-\alpha} - z_k^{1-\beta}}$ $r_0 = s_k$	$\frac{r_1 s + r_0}{(s-s_k)^2} \cdot \frac{y0}{z_k^{\left(z_k^{1-\alpha} - z_k^{1-\beta}\right)^2}}$ $r_1 = q_1 s_k + q_0$ $r_0 = q_1 s_k^2$ $q_1 = z_k^{1-\beta}(1-\beta) - z_k^{1-\alpha}(1-\alpha)$ $q_0 = z_k^{1-\alpha} - z_k^{1-\beta}$
z-domain pole	$\frac{y0}{z-z_k}$	$\frac{y0}{(z-z_k)^2}$

$$-\frac{2}{s}+\frac{-1+0.5s}{s^2} \tag{7.11}$$

$$\tilde{H}(s)=-\frac{1+1.5s}{s^2}$$

where in a CIFB second-order modulator, the first term refers to the feedback gain connecting at the modulator's input, while the term 1.5 to the feedback gain after the first integrator. Having H(s), design and simulate the second-order, 1-bit CIFB modulator in Simulink® using only CT blocks.

(i.e., Hint: to achieve correct scaling in Simulink®, use a gain block prior to the continuous time integrators with a gain of F_s).

Q.10

As discussed in Sect. 7.5, there are some advantages of using a return-to-zero DAC pulse in low-pass modulators. These DACs produce a rectangular pulse which lasts from 0 to $T_s/2$ only. Starting from the double integration modulator of Eq. 7.9, find the CT equivalent applying Table 7.7 with $(\alpha,\beta)=(0,0.5)$. Explain your results.

Q.11

A CT double integration modulator as in Q.8 is being design, but the NRZ feedback pulse is effected by excess loop delay, such as $(\alpha,\beta)=\tau_d, 1+\tau_D)$. Using Table 7.8, what is the equivalent $H(z)$ for such an $\widehat{H}(s)$ and DAC pulse?

Hint: The formulae in Table 7.9 only apply for a pulse with $\beta\leq 1$, but note that by using superposition it is possible to write a τ_d-delayed NRZ pulse as:

$$\widehat{r}_{(\tau_D,1+\tau_D)}(t)=\widehat{r}_{(\tau_D,1)}(t)+\widehat{r}_{(0,\tau_D)}(t-1) \tag{7.12}$$

that is, as a linear combination of a DAC pulse from τ_d to 1 and a one sampled-delayed DAC pulse from 0 to τ_d.

Table 7.9 z-domain equivalences for s-domain loop filter poles

Limit for $s_k=0$	$\frac{y0}{z-z_k}$ $y0=r_0(\beta-\alpha)$	$\frac{y1z+y0}{(z-z_k)^2}$ $y1=\frac{r_0}{2}[\beta(2-\beta)-\alpha(2-\alpha)]$ $y0=\frac{r_0}{2}(\beta^2-\alpha^2)$
z-domain equivalent	$\frac{y0}{z-z_k}\cdot\frac{r_0}{s_k}$ $y0=z_k^{1-\alpha}-z_k^{1-\beta}$	$\frac{y1z+y0}{(z-z_k)^2}\cdot\frac{r_0}{s_k^2}$ $y1=z_k^{1-\beta}[1-s_k(1-\beta)]-z_k^{1-\alpha}[1-s_k(1-\alpha)]$ $y0=z_k^{2-\alpha}(1+s_k\alpha)-z_k^{2-\beta}(1-s_k\beta)$
s-domain pole	$\frac{r_0}{s-s_k}$	$\frac{r_0}{(s-s_k)^2}$

Q.12
Briefly state the main advantages and disadvantages of the continuous time and discrete time ΣΔ-implementations.

Q.13
Briefly state the main non-idealities affecting continuous time ΣΔ-implementations as well as their effect and common counteraction used to limit their effects.

References

1. De la Rosa JM, Del Rio RF. CMOS sigma-delta converters: practical design guide. London: Wiley; 2013.
2. Casier H, Steyaert M, Van Roermund AHM. Analog circuit design: robust design, sigma delta converters, RFID. London: Springer; 2011.
3. Kester W. ADC architectures IV: sigma-delta ADC advanced concepts and applications. analog devices, 2008. [Online] http://www.analog.com/media/en/training-seminars/tutorials/MT-023.pdf.
4. Schreier R, Temes GC. Understanding delta-sigma data converters. Hoboken: Wiley; 2005.
5. Bourdopoulos G, et al. Delta-sigma modulators: modelling, design and applications. London: Imperial College Press; 2003.
6. Ortmanns M, Gerfers F. Continuous-time sigma-delta A/D conversion: fundamentals, performance limits and robust implementations. Berlin/Heidelberg: Springer; 2006.
7. Reiss JD. Understanding sigma-delta modulation: the solved and unsolved issues. JAES. 2008;56(1/2):49–64.
8. Cherry JA, Snelgrove WM. Continuous-time delta-sigma modulators for high- speed a/D conversion: theory, practice and fundamental performance limits. London: Kluwer Academic Publishers; 2002.
9. Breems L, Huijsing J. Continuous-time delta-sigma modulation for a/D conversion in radio receivers. London: Kluwer Academic Publishers; 2001.
10. Candy JC, Temes GC. Oversampling delta-sigma data converters: theory, design and simulation. New York: Wiley-IEEE Press; 1991.
11. Norsworthy SR, Schreier R, Temes GC. Delta-sigma data converters: theory, design and simulation. New York: Wiley-IEEE Press; 1996.
12. Medeiro F, Verdù BP, Vazquez AR. Top down design of high performance sigma delta modulators. Boston: Kluwer Academic Publisher; 1999.
13. Ortmanns M, Gerfers F, Manoli Y. Compensation of finite gain-bandwidth induced errors in continuous-time sigma-delta modulators. IEEE Trans, Circuits Syst I: Regul Pap. 2004;51:1088.
14. Keller M, et al. A comparative study on excess-loop delay compensation techniques for continuous time sigma delta modulators. IEEE, Circuits Syst I: Regul Pap. 1998;55:1887.
15. Kester W. The data conversion handbook. Burlington: Analog Devices Inc.; 2004.
16. Reddy K, Pavan S. Fundamental limitations of continuous time delta sigma modulators due to clock jitter. IEEE Trans Circuits Syst – I: Regul Pap. 2007;54(10):2184–94.
17. Hernandez L et al. Modelling and optimization of low pass continuous-time sigma- delta modulators for clock jitter noise reduction. In: Proceedings of the 2004 international symposium on circuits and systems, ISCAS, 2004.
18. Silva P. et al. Noise analysis of continuous time delta sigma modulators with switched capacitor feedback DAC. In: IEEE, circuits and systems, ISCAS international symposiums; 2006. p. 4.

19. Anderson M, Sundstorm L. Design and measurement of a CT delta-sigma ADC with switched capacitor switched resistor feedback. IEEE, Solid State Circuit J. 2009;44(2):473–83.
20. Ortmanns M, Gerfers F, Manoli Y. Jitter insensitive feedback DAC for continuous time sigma delta modulators. In: IEEE international conference, electronics circuits and system, 2001. p. 1049.
21. Sukumaran A, Pavan S. Low power design techniques for single-bit audio continuous time delta sigma ADCs using FIR feedback. IEEE J Solid-State Circuits. 2014;49(11):2515–25.
22. Oliaei O. Sigma–delta modulator with spectrally shaped feedback. IEEE Trans Circuits Syst II, Analog Digit Signal Process. 2003;50(9):518–30.
23. Nguyen K. A 106-dB SNR hybrid oversampling analog-to-digital converter for digital audio. IEEE J Solid State Circuits. 2005;40(12):2408–15.
24. Yavari M, Shoaei O. A 3.3V high resolution sigma delta modulator for digital audio. In: IEEE, 13th international conference on microelectronics, Rabat, Morocco, Oct 29–31, 2001.



DT ΣΔ-Modulator Design Example

8

This chapter presents the high-level design of a discrete time Sigma-Delta modulator, from specifications to system characterization. The aim of this exercise is to practice what is covered in the book and to further familiarize with the Simulink® Toolbox functions. The reader should be aware that the design choices proposed are not the only possible solution, and several alternative designs may be able to achieve the same, or even superior, performances. Therefore, thoughtful creativity is encouraged in the reader who wishes to experiment and challenge this simple design!

8.1 Modulator Specifications

The exercise aims to design, at high level, an audio band ΣΔ-modulator. The specifications to be met are:

- SNR $\geq$ 103 dB
- THD at full scale < -85 dB
- Stable input range $\geq$ 1.4 dBFS
- No limit cycles or tones in the audio band

The design procedure is divided as follows: Sect. 8.2 discusses the choice of the modulator architecture and its parameters (e.g., order, OSR, etc.), Sect. 8.3 presents the simulations that typically have to be performed while appraising the results in conjunction with the theory in order to validate the design, and the final Sect. 8.4, is where conclusions are drawn.

© Springer International Publishing AG, part of Springer Nature 2019

I. Arnaldi, *Design of Sigma-Delta Converters in MATLAB®/Simulink®*,
https://doi.org/10.1007/978-3-319-91539-5_8

8.2 Theoretical Design

8.2.1 Modulator Order and Oversampling Ratio

In order to achieve an SNR greater than 103 dB, the minimum effective number of bits required for the modulator is [1]:

$$\text{ENOB} = \frac{\text{SNR} - 1.76}{6.02} = 16.81 \tag{8.1}$$

Obviously, to obtain the desired resolution, various possibilities between low- and high-order modulators as well as different oversampling ratio are possible. To help identify the most suitable combinations, a summary of achievable SQNR for a respective order and OSR could be obtained by solving Eq. 8.2 (e.g., see Table A1, Appendix A). However, the graphs in Fig. A1 (i.e., Appendix A) illustrate approximately the same results derived from calculations, but with the difference that stability considerations are also taken into account, hence providing a lower and more realistic SQNR for a respective order and OSR value. Therefore, the use of Fig. A1 is encouraged for the choice of initial parameters:

$$\text{SQNR} = 6.02N + 1.76 + (20L + 10)\log(\text{OSR}) - 10\log\left(\frac{\pi^{2L}}{2L+1}\right) \tag{8.2}$$

According to Appendix A, third-order modulators seem a good compromise between the higher OSR which would be required by second-order loop filters and the higher power consumption, as well as larger area, required by fourth-order ones. However, due to the relatively low bandwidth of audio signals (i.e., a sampling frequency of 48 KHz is assumed), the use of a second-order modulator with OSR of 128 or 256, depending on the number of bits of the quantizer, would be highly possible in modern IC process technologies, and potentially it could result in a simpler and area-efficient solution and therefore be less expensive. Further, second-order modulators are less affected by the stability problems encountered for high-order loop filters [3]. It must also be noted that the theoretical SQNR is usually chosen to be 10–20 dB higher than the one desired, in order to provide some tolerance for the inevitable degradation caused by circuit non-idealities and to allow as much of the noise budget for thermal noise as possible [4, 5]. Additionally, the SQNR shown in Fig. A1 corresponds to modulators which are close to instability, hence providing a guidance on what is theoretically possible rather than practical [1]. It should also be considered that even though a larger area is usually discouraged, an increase of it could represent a significant decrease in power consumption, depending on the choice of the modulator fabrication technology [6] – i.e., in the case of a smaller technology, the increase in area may result less significant than the expected increase in OSR, and hence power consumption, needed for a lower-order ΣΔ-modulator. This is because power consumption could

be a more significant feature than in bigger process technologies. Therefore, a fourth-order modulator could be more convenient.

Following the discussion provided, in order to meet the SNR specification of the project, the design should aim for an SQNR of approximately 123 dB. To achieve this figure while maintaining the OSR and number of bits as low as possible, third-order modulators seem a good choice. Further, since no particular specifications to be met are given in terms of power budget, area, fabrication technology, etc., an OSR of 128 is chosen, resulting in a sampling ratio of [1]:

$$\text{Sampling ratio} = \text{Fs} * \text{OSR} = 48\ \text{KHz} * 128 = 6.144\ (\text{MHz}) \tag{8.3}$$

8.2.2 Quantizer Bit

It is required for this project to obtain a stable input range of at least ± 0.85, corresponding to 1.4 dBFS. Important to note that for a third-order modulator implementing a 1-bit quantizer, this specification can hardly be met (e.g., usually the maximum input range is approximately ± 0.7 as seen in Chap. 4) due to the consequent overload of the integrators and relative instability issues [7]. Therefore, the option to use a single-bit quantizer is discarded, although they would provide the simplest and most efficient power and area solution for the SNR considered.

Multi-bit quantizers [1, 8] allow for a greater input range (e.g., hence allowing higher peak SQNR to be obtained and resulting more tolerant to the use of dither) and are much more robust toward instability (i.e., as the feedback loop becomes more linear since the effective quantizer gain variation with signal is reduced, allowing for a more aggressive NTF to be implemented). They also relax the slew rate requirements of the op-amps (e.g., the width of a single DAC step halves for each additional bit) and are more robust toward jitter. However, for each additional bit, the area and power consumption of the quantizer doubles. Another major disadvantage is caused by the nonlinearity due to mismatches with the consequent multi-bit DAC. Since DAC errors are not shaped by the loop filter due to the DAC position in the loop, any error is added directly to the input signal, thus degrading the modulator SQNR performance and introducing further harmonic distortion. Therefore, dynamic element matching techniques to linearize the DAC would need to be implemented (e.g., RES, ILA, DWA, etc.), increasing the circuit complexity.

To meet the stable input range requirement for the project, a 2-bit mid-rise quantizer is used as it provides a convenient solution in order to maintain circuit complexity, area, and power consumption of the design as low as possible. Although some methods exist to predict stability for a certain input range when multi-bit quantizers are used, these are not strictly sufficient, and extensive simulations are still required to prove modulator's stability [1, 3, 6, 8, 9]. Therefore, it should be noted that the choice made has then to be carefully simulated in order to verify whether it allows to meet specifications or not.

To linearize the DAC and avoid the relative loss in SQNR and harmonic distortion problems, data weighting averaging DEM technique is to be implemented as it provides the most effective solution compared to RES and ILA techniques in terms of noise shaping, although it should be noted that ILA is claimed to be less tonal [1]. Therefore, extensive simulations need to be carried out in order to ensure that tones which may be generated will not appear in the signal band. If such is the case, other techniques or the use of dither should be considered.

8.2.3 Loop Filter Typology

The first architectural choice to be made is whether to use single-loop or cascaded modulators (MASH). Both solutions have been successfully implemented in audio applications with similar specifications to the one required for this exercise [6].

It can be easily seen that even some of the most basic MASH modulators (e.g., 2–1) require twice some of the component (e.g., quantizer, feedback DAC, etc.) of a single-loop ΣΔ-ADC of the same order, hence increasing power consumption, area, and circuit complexity of the overall system. Another major drawback of MASH modulators is that good matching of the analog NTF of the loop filter and that of the ECL (i.e., error cancellation logic) is required to avoid quantization noise leakage from the first stage, which otherwise dramatically reduces the achievable SQNR [10]. Despite matching problems, the major advantage of MASH modulators is that they are always stable hence allowing for a wide input range, which is not the case of high-order single-loop modulators [1]. Various techniques exist in order to make single-loop converters stable while improving their input range; however, these typically result in an increased circuit complexity, power consumption, and area. Therefore, the advantages in using single-loop architectures are greatly reduced when compared to their MASH alternative, especially in DT implementations.

For this project, since no constraints are given for power consumption, area, process technology, etc., and since circuital solutions which may aid in the choice of one loop topology over the other are not investigated as out of the scope of the book, it has been chosen to develop a single-loop modulator due to [1, 8]:

- Its stronger ability to achieve, in practice, high SNR since it does not suffer from the matching errors which severely affects MASH modulators.
- Since in both cases a multi-bit quantizer would have to be implemented (e.g., to maintain stability and improve signal range in single-loop modulators – to maintain signal-independent quantizer gain allowing for an accurate NTF digital cancellation in MASH), the disadvantages of this feature (e.g., need of DEM techniques, increased circuit complexity, etc.) make no significant difference in the selection of the loop structure.

8.2.4 Architecture Selection

For single-loop modulators, two main architectures exist [1]: chain of integrators with weighted feedforward summation (CIFF) and cascade of integrators with distributed feedback (CIFB). Both structures obtain the same noise-shaping characteristic, but the signal in the loop is treated differently. In CIFB modulators, the input and output of integrators contain both the signal and the quantization noise; hence, the integrators' input and output amplitude are large. In order to maintain stability, little gains are required, and therefore the input and output of the integrators need to be scaled down. In typical switch capacitor circuits for DT modulators, the gain coefficients are obtained by proportional capacitors; therefore, little gains result in big capacitors, which translates into more power consumption and bad output settle [6]. On the other hand, in CIFF modulators the input signal is not processed by the loop filter and, since the quantization noise power is typically much smaller than the signal power, larger integrator coefficients are allowed without causing saturation of the integrator outputs. This allows implementing smaller, power saving integrator capacitors [6]. Further, as the input signal is not processed by the loop filter, the output swing of the op-amp is reduced (i.e., especially if a multi-bit quantizer is used), and the introduction of harmonic distortion into the output signal caused by integrators' nonlinearities is avoided. Another advantage in terms of power consumption is that CIFF modulators require only one feedback DAC instead of the M DAC that may be required in a CIFB structure in each of the feedback paths. For these reasons, CIFF modulators are usually preferred when low power consumption and low signal distortion are required [1, 4, 8, 9]. It should be noted that although there are many advantages listed, CIFF modulators do have some drawbacks. First, the feedforward of integrator outputs require an accurate active adder before the quantizer, which increases area and power consumption while adding noise. Additionally, the closed-loop frequency response has a peak at high frequencies caused by the zeros in the signal path. The amplification of the out-of-band frequencies due to the high-frequency boost can overload the quantizer and drive the modulator into instability, especially in the case of input signals close to that frequency. In some cases, pre-filtering of the input signal may be necessary. Another disadvantage is that the NTF and STF are not independent – the selection of the NTF sets the magnitude of the high-frequency boost in the STF.

Since this exercise requires a relatively low signal distortion, the CIFF architecture is chosen. Further, to improve the SQNR response of the architecture, the optimization of the NTF [1] through zero spreading will be implemented (e.g., by 8 dB for a third-order modulator – see Table A3, Appendix A) by introducing a resonator between the third and second integrators, making the CIFF architecture an actual CRFF one. This improvement in SQNR has been considered because, in order to ensure stability, the maximum theoretical third-order NTF gain has to be reduced in CIFF architectures (e.g., by approximately 20 dB for this exercise), resulting in an increase of the baseband noise and thus leaving little tolerance for the target SNR considered. Therefore, in order to meet the SQNR of at least 123 dB, the theoretical SQNR achievable with the architecture considered results to be:

$$\mathrm{SQNR} = 139.92 - 20 + 8 = 127.92\ \mathrm{dB} \tag{8.4}$$

which seems appropriate, in terms of tolerance, from the specifications of the project.

8.2.5 Summary of Design Parameters

In this paragraph, the design choices made are briefly summarized. Table 8.1 reports an overview of the theoretical concepts discussed during the selection of the parameters for this exercise, while Table 8.2 details the design choices made as well as the theoretical performance expected for the modulator considered.

Table 8.1 Overview of parameters' design choices [1, 7, 9]

	SNR vs. OSR ratio	Idle tones/ linearity	Circuit design complexity	Power consumption/ area	Stability issues	Decimation filter simplicity
Low order, 1-bit, single loop	Bad	Bad	Good	Good	Good	Good
High order, 1-bit, single loop	Good	Good	Good	Medium	Bad	Good
MASH, 1-bit	Good	Bad	Bad	Bad	Good	Bad
Multi-bit	Good	Bad	Bad	Bad	Good	Bad
High order, multi-bit, single loop	Good	Bad	Bad	Bad	Bad	Bad

Table 8.2 Design parameters

Parameter	Symbol	Type	Value	Units
Architecture	–	CRFF	–	–
Input sample rate	Fs_{in}	–	48	KHz
Signal bandwidth	BW	–	24	KHz
SQNR	SNR	–	127.92	dB
Sample rate	Fs	–	6.144	MHz
Oversampling ratio	OSR	–	128	–
Quantizer	M	Mid-rise	2-bit	4 levels
Dither	–	–	None	–
DEM	–	DWA	–	–

8.3 High-Level Design

Figure 8.1 illustrates the system under consideration.

8.3.1 NTF Synthesis

To synthesize the coefficients for the CRFF loop filter, the parameters of Table 8.2 are used.

Type – *CRFF_DTExample* – in the MATLAB Command Window to load the variables (i.e., equivalent of the – *load_par* – routine, but with adapted specifications).
Type – *mex simulateDSM.c* – for Schreier's routines to function.
Type – *MakeModulator_DTExample.m* – to design the NTF and STF.

Once the code has been compiled, the NTF's poles-zeros and STF-NTF graphs should appear, as well as the z-domain transfer function and the coefficients before

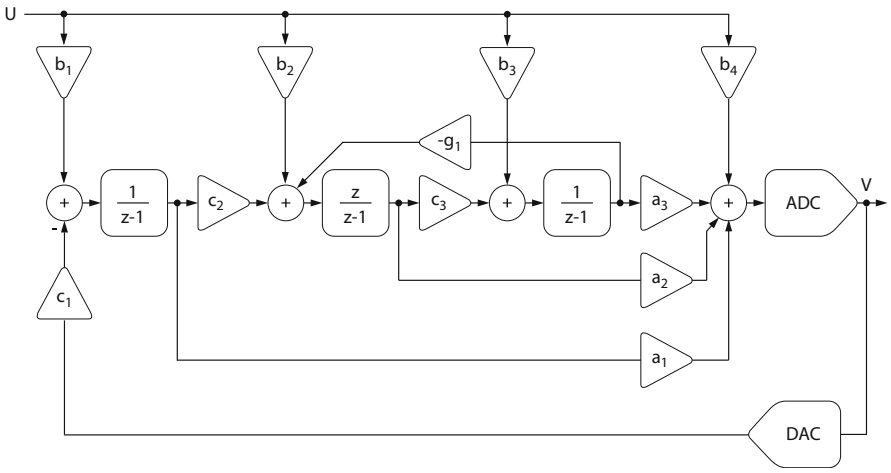


Fig. 8.1 Theoretical MOD3-CRFF architecture

and after dynamic range scaling. Note that dynamic range scaling has to be performed because it ensures roughly the same power level at all nodes, and so, it allows to avoid unnecessarily large noise gain from nodes with small signal levels [1]. On the other hand, non-scaled coefficients are those of a modulator whose internal states occupy an unspecified range. Thus, comparing the loop filter coefficients before and after dynamic range scaling (Table 8.3), it is easily seen that the integrator outputs are scaled down since the respective c_i coefficients are no

Table 8.3 Unscaled (left) and scaled NTF coefficients

	a_i	g_i	b_i	c_i	a_i	g_i	b_i	c_i
1	0.5615	0.00036142	1	1	1.2459	0.0016	0.4507	0.4507
2	0.2507	–	0	1	1.2378	–	0	0.4495
3	0.0457	–	0	1	1.0293	–	0	0.2191
4	–	–	1	–	–	–	1	–

Table 8.4 Unscaled (left) and scaled integrator output statistics

	MAX	MIN	Mean	Standard deviation	MAX	MIN	Mean	Standard deviation
Integrator 1	0.82	−0.754	0	0.227	0.37	−0.34	0	0.102
Integrator 2	1.549	−1.58	0	0.338	0.314	−0.32	0	0.069
Integrator 3	6.541	−6.668	−0.024	1.271	0.29	−0.296	0	0.056
SUM	8.91	−9.002	–	–	0.974	−0.956	–	–

longer equal to 1. Further, note that the b_1 coefficient is set equal to the feedback c_1 coefficient. This ensures a maximally flat STF over the passband (i.e., verifiable in the STF-NTF graph).

Moving to the Simulink® model of the modulator (e.g., *mod3_CRFF_2bit.mdl*), a first verification should be conducted by simply simulating the model and ensuring that the time domain results of the scopes display, indicatively, correct behavior (e.g., input sinewave = output sinewave, 2-bit quantization of the output signal, etc. Note that, obviously, the input signal should have an amplitude and frequency not too high nor too low, in order to avoid instability, etc.).

Having a first confirmation on the correct behavior of the modulator, the next step is to verify that the dynamic range scaling has been performed adequately. Therefore, the range of the signals at the output of the integrators should be investigated (i.e., using the routine – *integ_stats(INT1OUT)*, etc.). The results are summarized in Table 8.4 and, as it can be seen, the absolute value of the *Max/Min* signals at the integrator outputs are smaller and have similar values between each other in the scaled case. Thus, roughly the same power level at all nodes is ensured. The controlled gain achieved at each node would be perhaps even clearer by looking at the time scope of the integrators' output, noting that the signal amplitudes are roughly the same for each of the outputs. Important to note is also that, after dynamic range scaling, the sum of the integrator outputs should result in the full signal range $\pm$ 1 being used, as in this case (i.e., assuming a full-scale input. The data in Table 8.4 are for a −3 dBFS signal, hence the slightly lower values). Thus, the scaling can be considered successful. Further, in the model, note that the alternating delaying and non-delaying integrators used help to reduce the noise peaking characteristic of the CIFF/CRFF architectures, hence improving the stability of the modulator [6, 10].

Important to note is that the coefficient b_4 is set to 1 in the NTF synthesized since this improves the linearity of CRFF architectures. To understand this point, the interested reader is encouraged to repeat the simulations of the output of the integrators in the case of the coefficient b_4 set to 0 (e.g., to do this write, % , in front of the code in, *line 53*, of the routine – *MakeModulator_DTExample*). Repeating the simulations, the results should report a smaller signal range being used by the modulator after dynamic range scaling (e.g., sum of integrator output statistics should account for, roughly, ± 0.63), mainly due to the larger standard deviation in the last integrator which limits the uniform distribution of noise gain from nodes with small signal levels. This should also be noticeable by looking at the shapes of the integrator output statistic histograms, where the last integrator, instead of achieving its max value around zero, achieves that at the extremities of its range (i.e., *U*-shape). Therefore, to avoid distortion and other issues, the usable signal range needs to be smaller, especially in the last integrator [9]. Thus, a less efficient distribution of noise gains during dynamic range scaling of the loop filter coefficients is achieved, resulting in a poorer architecture linearity. Important to note is that this phenomenon has a minor effect on other modulator parameters such as the SQNR, and the difference between the $b_4 = 1$ and the $b_4 = 0$ case is, in most cases, negligible. Moreover, note that not having to realize the b_4 coefficients would have the advantage of a smaller and less power consuming design.

Continuing the NTF analysis, the stable input range of the modulator designed is found at -0.91 dBFS, as can be seen by typing – *umax* – into the Matlab Command Window or by calculating:

$$\text{Stable input range} \leq \frac{\text{umax}}{\text{nLev} - 1} = 0.9 \cong -0.91 \text{ dBFS} \tag{8.5}$$

where umax $= 2.7$ and the number of quantizer levels nLev $= 4$. Note that this value needs to be further verified through DC and sinewave simulations, as shown later.

To verify the SQNR the routine – *sweep_sinamp* – should be used initially, in order to be able to compare the results for different input amplitudes. Typically, a well-designed modulator should perform close to theoretical expectations *at least* up to a -3 dBFS input signal, such as:

$$\text{SQNR} = 127.92 - 3 = 124.92 \text{ dB} \tag{8.6}$$

Moreover, note that the routine – *sweep_sinamp* – should be performed for an input signal of frequency as close as possible to the signal bandwidth of the converter while trying to hit as best as possible an FFT bin. Figure 8.2 illustrates the results for a 12 KHz sinewave input signal, and, as it can be seen, the peak SQNR and the SQNR for a -3 dBFS input signal result, approximately, 123 dB and 119 dB, respectively. These values are only 5 dB lower than what expected from theoretical calculations; thus, the modulator performs satisfactorily considering that the lower values found can be attributed to FFT inaccuracies and others simulation's non-idealities. Another important thing to note is that the resolution used by the – *sweep_sinamp* – routine doesn't allow to accurately find and use the maximum

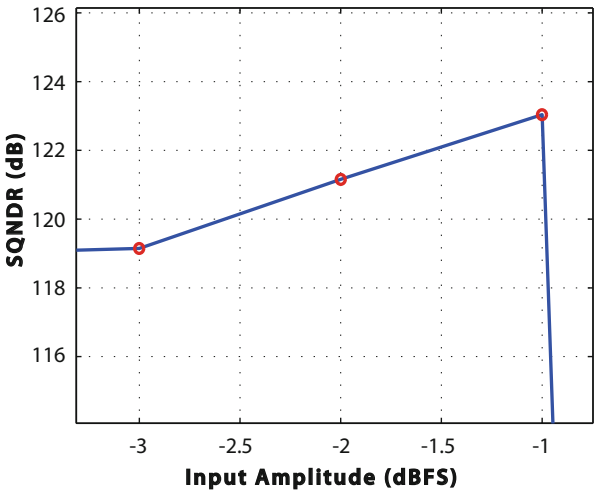


Fig. 8.2 12 KHz sine amplitude sweep SQNR peak

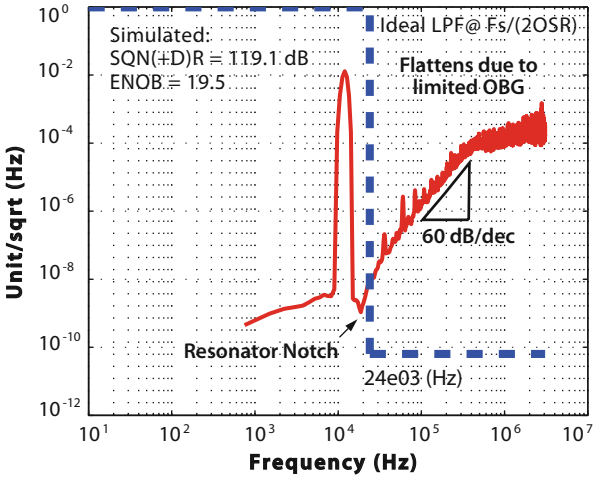


Fig. 8.3 PSD for a sinewave of 12 KHz, amplitude −3 dBFS

stable input range; therefore, the peak SQNR is found at −1 dBFS, instead of −0.91 dBFS, thus further justifying the lower peak SQNR value when compared to theoretical calculations.

A final initial verification of the NTF synthesized should also be conducted through the – *mod_SNDR* – routine. Figure 8.3 illustrates the power spectral density graph of a 12 KHz sinewave at −3 dBFS in amplitude, and it results clearly that the NTF behaves as expected, specifically:

- Slope of 60 dB/dec, hence third-order filter.
- At very high frequencies, the NTF flattens due to the reduced out-of-band gain (i.e., Lee's criterion applied).

- Notch at the edge of passband due to zero spreading and correct increment in SQNR, thanks to the resonator (e.g., about 8 dB) when compared to the same NTF with all zeros at DC (e.g., verifiable by setting the *g(1)* coefficient to zero in the Simulink® model).

8.3.2 DC Behavior

Once the design satisfies the simulations discussed in the previous paragraph, the modulator stability, accuracy, and tonal behavior for DC inputs should be investigated. Many different DC values should be tested, using both the – *mod_SNDR* – routine and the time scopes at each sensible node of the modulator. Note that particular attention should be paid to rational DC values, as more prone to creating long repeating sequences leading to tones.

Table 8.5 summarizes the results for some DC input values, and, as it can be seen, the maximum stable input is found at an amplitude of 0.91, hence confirming the results of Eq. 8.5. Further, the mean output of the modulator shows that the converter is able to accurately represent the analog DC inputs with an error of approximately 10^{-6} at DC = 0. This can be considered accurate enough for a converter aimed at audio applications, as the one of this exercise.

The major problem observed in the simulations is the presence of tones in the passband for DC values below, approximately, 0.5. Further, even when tones are not present in the baseband (e.g., DC = 0.85, etc.), they always appear at very high frequencies. This limits the robustness of the modulator to instabilities [1]. In fact, although such tones are out of the passband, and so are to be filtered out by the decimation filter, they can still lead to instability if they grow too much in magnitude and exceed the out-of-band gain limit set in the design. Given the magnitude of these high frequencies tones, as well as their presence in the passband, it results clear that dither has to be implemented in order to counteract this phenomenon. By trying various dither values, it results that in order to meet the specification of a stable input range of at least ±0.85, a maximum dither of 0.2 is allowed. It should be noted, though, that smaller dither values are sufficient to remove limit cycles for DC inputs. However, as discussed later, a dither of 0.2 is necessary in order to avoid tones in the baseband for any type of input signals (e.g., sinewave, etc.). Therefore, all the simulations that will be presented from now on use a 0.2 dither value, unless explicitly stated. An interesting thing to point out is that using an input DC value of 0.86 does not always lead to instability. This should highlight the difficulty of

Table 8.5 DC simulations

DC input	0	0.25	0.5	0.75	0.85	0.91	0.4367	$\pi/10$
Mean	−1.6557e−06	0.25	0.5	0.75	0.85	0.91	0.4367	0.31416
					Dithered input limit	*Un-dithered input limit*		

testing ΣΔ-modulators and suggests to the reader that it results wise to perform the same simulation few times when testing for variables which are around the modulator limits.

8.3.3 Stability and Tonal Behavior

Modulator stability and satisfactory tonal behavior within the stable input range have also to be confirmed for, at least, slow varying input signals such as sinewaves. A full set of simulations of multiples of the reference frequency of 12 KHz (e.g., 6 KHz, 3 KHz, etc.), as well as at amplitudes for each tested frequency of, for example, 0.0625, 0.125, 0.25, 0.5, 0.707, 0.85, 0.91, and 0.92, should be conducted [6, 9], both without and with dither. As for the DC tests, the routine to be used is the – *mod_SNDR* – coupled with time scopes investigations. Important to note is that since it is not possible to test the frequency of 24 KHz, as too close to the limits of the converter, the frequency of, for example, 19.5 KHz, can be used instead. This frequency value does not fall in an FFT bin, and so the numerical results it produces (e.g., SQNR, etc.) are to be affected by inaccuracies (i.e., note that choosing a sensible frequency which still performs relatively well, as in this case, is suggested in order to limit possible confusion between actual modulator's issues and simulations inaccuracies). However, testing the modulator with signal frequencies close to the converter bandwidth results essential to investigate NTF stability, especially in high-order CRFF architectures where the noise at high frequencies coupled with a high-frequency signal can often lead to instability.

Performing the range of simulations discussed with and without the use of dither, the stable input range is found at amplitudes of 0.91 and 0.85, respectively, hence in accordance with the DC results. Note that to find the minimum amount of dither needed to remove tones, it is advisable to start from the minimum dither value which removes tones in DC simulations and then keep increasing this value gradually until all tones disappear for sinewave inputs too. As already mentioned, the dither value to be used for this exercise found from the sinewave simulations here discussed is 0.2. With this amount of dither, no tones have been found to fall in the passband.

Obviously, with the use of dither, attention has also to be paid to the SQNR, since this results reduced from its initial value due to the increase in baseband noise. The SQNR for the reference sinewave signal of 12 KHz at −3 dBFS is then changed from 119 dB to 115.3 dB – approximately 4.5 dB of difference – when the dither of 0.2 is used. However, thanks to the tolerance in SQNR from the minimum project requirement, this decrease does not compromise the validity of the design, with the modulator still able to achieve an SQNR of 103 dB for an input up to approximately −18 dBFS (i.e., use – *sweep_sinamp* – to verify this point).

To ensure modulator stability within the stable input range found at ±0.85, further tests should always be conducted for a ramp, step, and chirp input signals [6, 9]. Figure 8.4 illustrates the results for the ramp and step signals (i.e., including dither). Note that the step used has an amplitude of 0.86, instead of 0.85, in order to test the modulator under the most stressful conditions possible (i.e., fast varying and

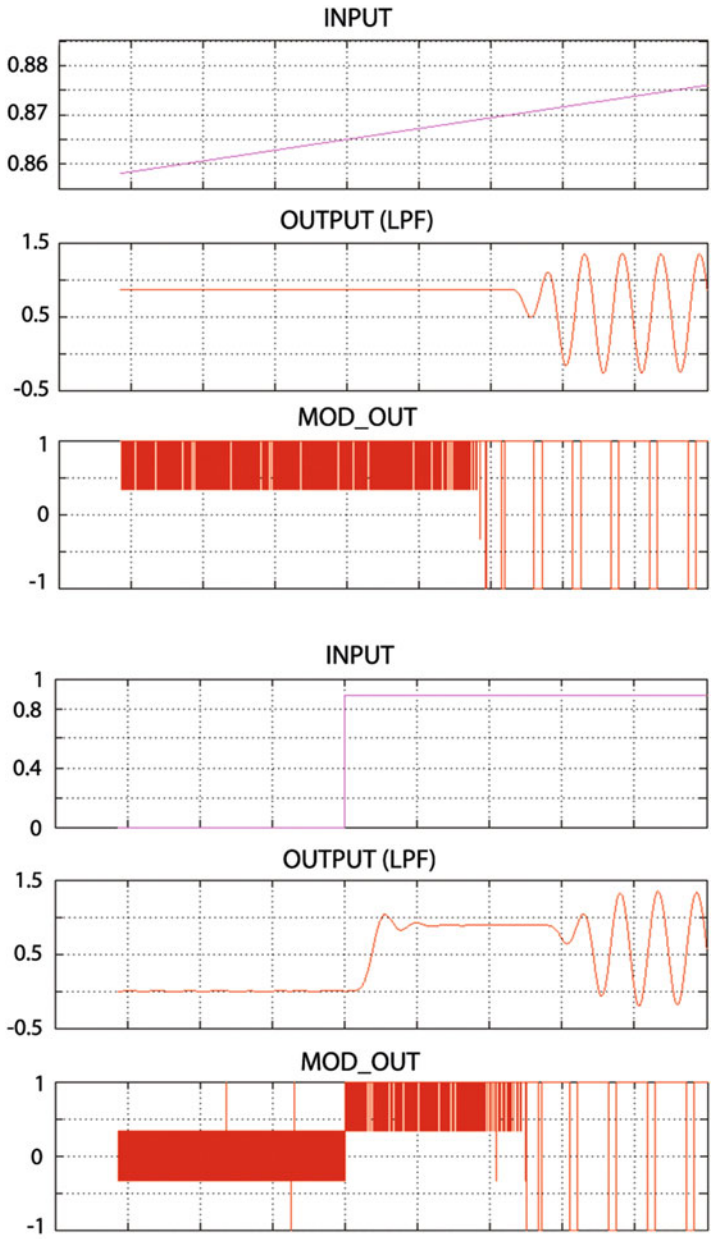


Fig. 8.4 (*Top*) ramp and (*bottom*) step time scope results

slightly large signal). As it can be seen, it takes several cycles for the modulator to become unstable in both the ramp and step cases. This further confirms that by using an amplitude of just ±0.85, the modulator should be ensured to remain stable.

8.3.4 Integrator Finite Gain and Saturation

The effect of low, finite DC integrators gain should also be investigated (e.g., using the – *sweep_dc_dc* – routine), especially in those designs where limited gain is available and a sensible architecture is implemented (e.g., first stage of MASH modulators, etc.). For the CRFF modulator designed, it is found that, without dither (e.g., *mod.dither = 0*) and up to an integrators gain of about 5 (e.g., *mod.igain1 = 5*, *mod.igain2 = 5, mod.igain3 = 5,* etc.), the width of the dead zone at zero DC depends on a power of the gain of 10^3, as expected from a third-order modulator [2]. Important to remember is that dither helps to recover dead zones and smooth the overall SQNR response; thus, the tolerance to finite integrators gain results improved in the modulator designed. Therefore, achieving a high enough DC integrators gain to minimize dead zones, and hence achieve a reasonable converter accuracy, shouldn't represent a significant issue for real implementations of the modulator designed.

The behavior during finite integrator saturation in the case of sinewave inputs should also be investigated by setting – *mod.isat1 = 1; mod.isat2 = 1; mod.isat3 = 1* – and running the – *sweep_sinamp* – routine while dither is applied. The results should show that the modulator still achieves a peak SQNR of, approximately, 117 dB (i.e., about 6 dB loss from the peak SQNR of 123 dB found earlier). Such loss is not particularly significant thanks to the tolerance from minimum specifications allowed in the initial phase of the design.

8.3.5 Feedback DAC and Element Mismatch

Due to the use of a multi-bit quantizer, it is necessary to test the modulator performance when feedback DAC non-idealities are taken into consideration. The set of simulations for different frequencies and amplitudes conducted in Sect. 8.3.3 should be repeated for the Simulink® model – *DAC_mod3_CRFF_2bit*. These test should be conducted according to the specifications of the technology of implementation available; however, an element mismatch of 0.25% is generally a good worst case scenario for modern IC process (e.g., set *dac.elements = [1.0005 1.0025 0.9975]* in the model).

Performing the simulations with a mismatch of 0.25% should reveal that the modulator is able to perform relatively well when DWA is enabled, with an SQNR at −3 dBFS of 114.5 dB. However, it should be noted that few, small tones might be seen when performing the simulations (e.g., at 6 KHz of amplitude 0.707 and 0.5, at 3 KHz of amplitude 0.85, etc.). However, the small tones are generally above 20 KHz, hence above the limit of the audible range. Moreover, additional

simulations would show that tones tend to disappear or become too small to be detected for smaller element mismatch errors (e.g., 0.025%, still achievable in today IC processes). It should also be noted that such small tones can also be an artifact of the FFT setting, rather than actual tones from the modulator. Therefore, the modulator and the DWA technique implemented can be considered adequate enough in terms of tonal behavior, and further simulations on the matters would rather be performed on actual circuital simulators instead of high-level software. If tones would still result in circuital simulations, alternatives to remove these may include the use of different DEM algorithms (i.e., ILA, etc.) or the use of more dither (i.e., which may require a redesign of the modulator to meet the specification of the stable input range, etc.).

8.3.6 Coefficients Mismatch

In this paragraph, the single coefficients variation is investigated in order to better understand their effect on the modulator. Moreover, note that a final test to verify if the modulator is capable of meeting the project specification should always be conducted by combining the effects of coefficients mismatch and all other non-idealities studied. Generally, it is a good practice to consider a design satisfactory if it meets minimum project specifications for, at least, an input signal amplitude of -3 dBFS [6, 9]. Thus, in all the results here presented, the reference sinewave of 12 KHz at -3 dBFS is used, as well as dither of 0.2.

The results obtained for a $\pm$ 50% variation in the g_1 coefficient value are illustrated in Fig. 8.5. As it can be seen, the SQNR tends to decrease by few dBs, and the change of the coefficient value clearly affects the notch in the NTF. Note that, as already mentioned in Chap. 4, it is the position and incidence of the notch to regulate the distribution of the power in the spectrum. Considering that the g_1 coefficient is relatively small compared to the others, and hence can be difficult to be realized in analog form (e.g., requiring a capacitor smaller than the minimum capacitor unit size in integrated switched-capacitor implementations [6]), it can be said that, according to the results obtained, even for a gross variation in its value, the modulator is sufficiently tolerant to its mismatch as the resultant SQNR is still able to largely meet the project specifications.

Figure 8.6 presents the effect of mismatch in the a_i coefficients. Important to note is that the a_i coefficients affect the NTF aggressiveness and overall modulator stability [1]. As it can be seen, for a variation of -50%, the OBG results diminished; however, the stability is compromised by the peak that appears at very high frequency due to the too small OBG. On the other hand, a variation of +50% from the original a_i coefficients value increases the OBG; hence, higher SQNR can be achieved at the expenses of a smaller stable input range.

Regarding the coefficients b_i, it should be known from the previous analysis that they affect the input signal scaling and system linearity. Coefficients c_i, instead, affect the system scaling and integrators saturation limit [1].

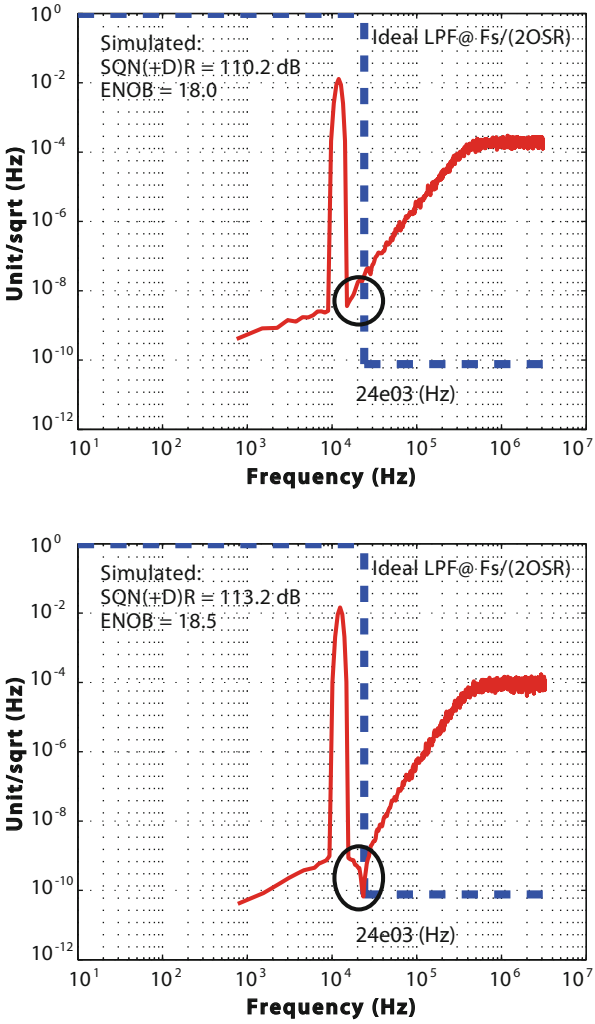


Fig. 8.5 Output spectrum of g_1 coefficient for a variation of (*top*) −50% and (*bottom*) +50%

Considering more realistic mismatch values of ±0.25%, since the modulator is a DT type and the coefficients are likely to be implemented by capacitors ratio, a final test should be performed including all the non-idealities studied in this exercise, in order to simulate a worst real case scenario [6, 9]. By setting the integrators gain to 100 (i.e., *mod.igain*), the integrator saturation to 1 (i.e., *mod.isat*), mismatching the quantizer levels (i.e., 0.668, −0.64, etc.), and mismatching all the coefficients and DAC elements randomly by ±0.25%, a similar results to that of Fig. 8.7 should be obtained.

Obviously various mismatch combinations exist, each providing a different response. Performing few simulations, it should result clear, however, that the

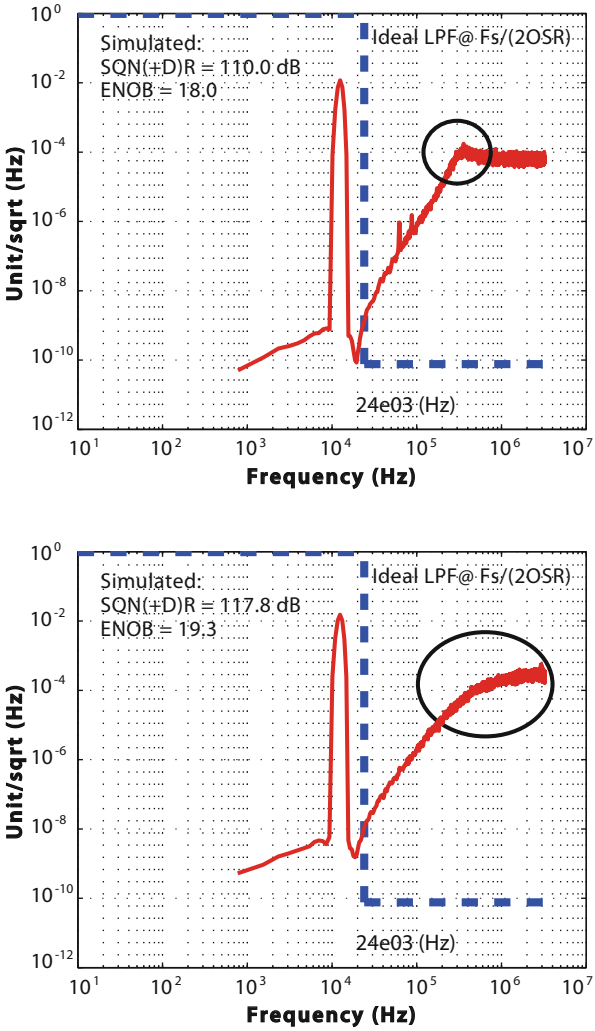


Fig. 8.6 Output spectrum of a_1 coefficient for a variation of (*top*) −50% and (*bottom*) +50%

modulator is able to meet the SQNR required, allowing approximately between 5 to 10 dB tolerance from minimum specifications when a full-scale input signal is used. This can be considered an adequate margin for the thermal noise budged to be managed in the case of circuital implementation of the design [4]. Important to note is that the maximum stable input range should also be verified when all non-idealities are considered, and it should be found that the modulator is able to meet the specifications.

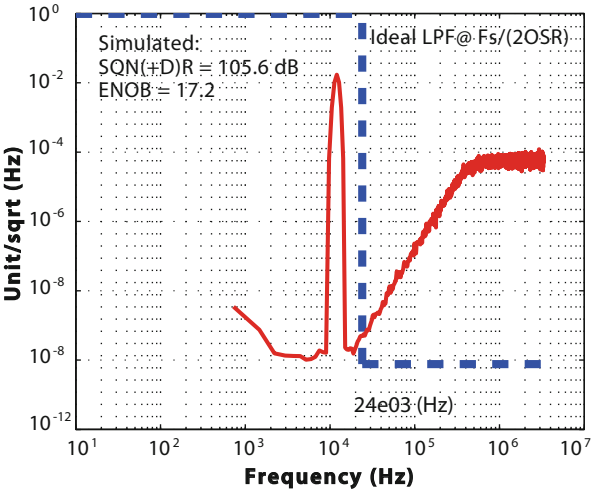


Fig. 8.7 12 KHz, −3 dBFS sinewave output spectrum when all non-idealities are applied

8.3.7 Total Harmonic Distortion

In order to roughly estimate the total harmonic distortion (i.e., THD) of the modulator, the procedure described below can be used.

Simulate the – mod3_CRFF_DTExample – model with a full-scale sinewave input and dither.
Type – *THD* – into the MATLAB Command Window.

The routine returns the power and frequency of the first five harmonics, as well as the THD. Note that the SNDR measured should be approximately 115 dB (i.e., run – *mod_SNDR*), while the THD should be around −96 dB. Therefore, being the THD less than the minimum required of −85 dB, the designed modulator can be considered successful in terms of THD. However, it should be noted that such estimation is very approximate and should be used only as a guideline. For a more precise estimation, it is suggested to test the THD at a circuit level and for a much larger number of harmonics.

8.4 Conclusions

Table 8.6 summarizes the characteristics and performance of the modulator designed. As it can be seen, the project specifications are met; therefore, the design can be considered successful from a high-level point of view. The next steps in the design process would be to perform calculations for thermal noise in order to be able to size the capacitors needed for the circuital realization of the coefficients. Then, the circuital modulator model should be designed and simulated.

Table 8.6 Final modulator performance

Parameter	Value	Comments
Order/architecture	3rd/ CRFF	
Bit	2	Mid-rise quantizer
Bandwidth (KHz)	24	
Sampling rate (MHz)	6.144	
OSR	128	
SNR at full scale (dB)	115 ≈108	Ideal MOD with dither With all non-idealities and dither
Maximum stable input	±0.85	−1.4 dBFS
THD (dB)	−96.2526	At full scale
Dither	0.2	
DEM	DWA	

Exercises

Q.1
Briefly state the main steps of the high-level design process of DT ΣΔ-converters.

Q.2
For the design example considered in this chapter, list at least other three different theoretical designs. Support your answer with some minimum analysis.

Q.3
Design the NTF of an audio, single-bit, second-order DT ΣΔ-modulator with OSR = 16 and OBG = 1.5 in the case of a CIFB and CRFB architecture. Note to set the input feedforward coefficients b_2 and b_3 to zero. Plot the magnitude response and pole-zero graphs for both the NTFs. What considerations can be made regarding the STF?

Q.4
Map, in Simulink®, the synthesized NTF in Q.3 to a CRFB topology. Simulate the modulator and estimate the in-band SQNR and the MSA.

Q.5
What changes would you make to the Simulink® schematic if the modulator in Q.4 is changed to CIFB?

Q.6
Test the CIFB modulator in Q.5 and state the main differences of the output spectrum when compared to the one of Q.4.

Q.7

List at least five reasons to convince your manager that a multi-bit design offers substantial improvement over the single-bit design developed in Q.4 and Q.6. How would you mitigate the errors arising due to the nonlinearity in the multi-bit feedback DAC?

References

1. Schreier R, Temes GC. Understanding delta-sigma data converters. Hoboken: Wiley; 2005.
2. Reiss JD. Understanding sigma-delta modulation: the solved and unsolved issues. JAES. 2008;56(1/2):49–64.
3. Kester W. The data conversion handbook. Burlington: Analog Devices Inc.; 2004.
4. Casier H, Steyaert M, Van Roermund AHM. Analog circuit design: robust design, sigma delta converters, RFID. London: Springer; 2011.
5. Bourdopoulos G, et al. Delta-sigma modulators: modelling, design and applications. London: Imperial College Press; 2003.
6. De la Rosa JM, Del Rio RF. CMOS sigma-delta converters: practical design guide. London: John Wiley & Sons, Ltd; 2013.
7. Candy JC, Temes GC. Oversampling delta-sigma data converters: theory, design and simulation. New York: Wiley-IEEE Press; 1991.
8. Geerts Y, Steyaert M, Sansen W. Design of multibit delta sigma A/D converters. New York: Kluwer Academic Publisher; 2002.
9. Medeiro F, Verdù BP, Vazquez AR. Top down design of high performance sigma delta modulators. New York: Kluwer Academic Publisher; 1999.
10. Norsworthy SR, Schreier R, Temes GC. Delta-sigma data converters: theory, design and simulation. New York: Wiley-IEEE Press; 1996.



CT ΣΔ-Modulator Design Example

9

This chapter presents the high-level design of a Continuous-Time Sigma-Delta modulator, from specifications to system characterization. The aim of this exercise is to practice what covered in the book and to further familiarize with the Simulink® Toolbox functions. The reader should be aware that the design choices proposed are not the only possible solution and several alternative designs may be able to achieve the same, or even superior, performances. Therefore, thoughtful creativity is encouraged in the reader who wishes to experiment and challenge this simple design!

9.1 Modulator Specifications

The exercise aims to design, at high level, an audio band ΣΔ-modulator. The specifications to be met are:

- 16-bit
- 20 KHz bandwidth
- 92 dB dynamic range
- 90 dB S/(N + THD)
- Minimum power consumption
- Clock frequency to be chosen as any binary divider of 20 MHz

The design procedure is divided as follows: Sect. 9.2 discusses the choice of the modulator architecture and its parameters (e.g., order, OSR, etc.), Sect. 9.3 presents the simulations that typically have to be performed while appraising the results in conjunction with the theory in order to validate the design, and the final Sect. 9.4, is where conclusions are drawn.

© Springer International Publishing AG, part of Springer Nature 2019

I. Arnaldi, *Design of Sigma-Delta Converters in MATLAB®/Simulink®*,
https://doi.org/10.1007/978-3-319-91539-5_9

9.2 Theoretical Design

9.2.1 Modulator Order, Oversampling Ratio, and Loop Filter

In this example, a first consideration to be made for the design is whether to use a single-loop or MASH architecture due to the dynamic range being a parameter of interest. In fact, since each of the single stages of a MASH modulator would have to satisfy such parameter, it should be noted that the design may result quite challenging. However, both architecture typologies have been successfully implemented in DT modulators of similar specifications, thanks to the reasonable matching accuracy achievable due to the integrator coefficients relying on capacitor ratios [1]. Unfortunately, such accuracy is hardly achievable in CT modulators [2], and so MASH architectures result also less practical to be implemented in CT domain, due to the difficulty encountered in matching the analog and digital NTFs. This issue could lead to the realization of an "over-designed" modulator, so to counteract the losses in SNR and DR due to mismatches. Therefore, a single-loop architecture is preferred for this exercise.

To obtain a 16-bit resolution, the target SNR required for the modulator is [1]:

$$\mathrm{SNR} = 6.02 * \mathrm{ENOB} + 1.76 = 98.08\ \mathrm{dB} \tag{9.1}$$

As for the DT example in Chap. 8, to choose the order, OSR, and number of quantizer bits, it is convenient to refer to Fig. A1 in Appendix A. For this exercise, it is suggested to consider only orders between the second and fourth since higher orders would, arguably, not be convenient in terms of achievable SNR per area/power consumption required. Important to note is also the achievable DR for the various parameter combinations under consideration. A summary of achievable DR can be drawn by solving [1]:

$$\mathrm{DR} = 10\log\left(\frac{3(2L+1)}{2\pi^{2L}}\right) + (20L+10)\log(\mathrm{OSR}) \tag{9.2}$$

The results of Eq. 9.2 for different parameter combinations are reported in Table A2, Appendix A. Therefore, according to the data in Appendix A, in order to satisfy the SNR and DR requirements of this exercise, high-order modulators appear to be the most suitable choice as they allow implementing lower OSR values compared to second-order ones, hence relaxing circuit requirements and overall power consumption. However, due to the relatively low bandwidth of audio signals (i.e., sampling frequency is assumed to be 48 KHz), the use of a second-order modulators with OSR of 128 or 256, depending on the number of bits of the quantizer, would be highly possible in modern IC process technologies and potentially could result in a simpler and area-efficient solution. Further, as discussed in Chap. 8, although a larger area is usually discouraged, an increase of it could represent a significant decrease in power consumption, depending on the choice of the modulator fabrication technology [3, 4].

Following the discussion presented and noting that the theoretical SQNR and DR values are usually chosen to be 10–20 dB higher than the one desired (i.e., in order to provide some tolerance for the inevitable degradation caused by circuit non-idealities and to allow as much of the noise budget for thermal noise as possible [4]), in order to meet the project specifications, it is suggested that the design should aim for an SQNR of approximately 108–118 dB while allowing some tolerance for the DR as well (e.g., >102 dB). To achieve these specifications while maintaining a good compromise between the OSR, area, and power consumption, a third-order modulator with an OSR of 64 is suggested, resulting in a sampling ratio of [5]:

$$\text{Sampling ratio} = Fs * \text{OSR} = 48\ \text{KHz} * 64 = 3.072\ \text{MHz} \tag{9.3}$$

9.2.2 Feedback DAC and Quantizer Considerations

To choose an efficient implementation in terms of quantizer bits, feedback DAC, and CT modulator non-idealities counteract actions, additional power consumpition considerations are required.

For a third-order modulator with an OSR of 64, according to Table A2 in Appendix A, the theoretical DR achievable is approximately 106 dB. Important to note is that this figure would result reduced after stability actions are taken to stabilize the high-order loop (i.e., Lee's criterion). Therefore, due to the relatively limited tolerance available (i.e., probably just about 10 dB after stability actions [4]), it is important to preserve as much of the DR figure as possible already at this design stage, as it would probably get worse if the modulator has to be implemented in a real circuit, due to analog non-idealities and process variations.

If the feedback DAC to be implemented uses RZ pulses, the DR would further reduce due to the properties of the pulse itself [2], as explained in Chap. 7. Therefore, very accurate pulse-shaping techniques which allow retaining as much of the DR as possible would have to be used. As already discussed, such techniques have large circuit overhead, are of complex realization, and have high implementation costs. Further, although very difficult to estimate at this stage, the circuit overhead could lead to a higher power consumption (e.g., if a switched capacitor DAC is used to pulse shaping, a larger slew rate – hence bias current – would be required in the first op-amp [4], etc.). Therefore, generally, in the literature [3, 6–8], it can be often found that lower power consumption is achieved if NRZ feedback pulses are used (i.e., note that, however, such power estimates are somehow misleading and of difficult interpretations as dependent on what has been considered during the estimations – some consider just the modulator, other also the clock generator which may be of great precision hence consuming more power, etc.). However, a significant reason which may lead to preferring NRZ pulses is that they allow for a greater charge to be fed back at each cycle compared to the RZ type, resulting in a greater input signal DR and a more power efficient DAC implementation. Therefore, due to the ease in preserving DR and the less power consumption achieved in terms of the modulator itself (i.e., without considering clock circuitry, etc.), NRZ pulses are

suggested for the exercise. The drawback is that such pulse type is more prone to jitter, ISI, and ELD errors; therefore actions are required to preserve the modulator performance.

The advantages and disadvantages of a single-bit and multi-bit quantizers have already been discussed at length throughout the book, and for this exercise, a 3-bit quantizer is suggested. This choice appears as a reasonable compromise between the benefits offered by a multi-bit operation, especially in terms of jitter sensitivity and more robust DR, and the increase in complexity of implementation. Moreover, such choice, besides allowing for a more robust modulator implementation in terms of SNR, DR, and NTF aggressiveness (i.e., NTF can be more freely controlled due to the lower in-band quantization noise and the smaller difference between successive outputs of the modulator), also allows for lower slew-rate requirements in the loop filter as the noise amplitude is smaller, hence resulting in less power dissipation from the integrators op-amps (i.e., especially in the first integrator [4]). This should help in achieving a low power consumption design as required by specifications. The downside of this choice is that DAC nonlinearity has to be addressed and, in order to keep the exercise short, the same DWA algorithm discussed in Chap. 8 is to be used.

To summarize, a greater area and power consumption from the multi-bit quantizer and subsequent DAC, including DWA, are required as NRZ feedback pulse is to be used; however, the increase in power consumption should result relatively small compared to the increase which would be caused by the overhead circuitry for accurate pulse shaping needed to retain DR if an RZ pulse was chosen. For this project since no constraints in terms of area are given, multi-bit solutions and NRZ pulse feedback seem to be a convenient choice, as modulator area is traded for, potentially, lower power consumption of the modulator itself.

9.2.3 Architecture Selection

The last design choice to be made as regards the architecture topology, and, according to the theory previously discussed in Chap. 4, a feedforward structure, or CRFF, is suggested for this project since the requirements in terms of THD and minimum power consumption are relatively stringent. Hence, after stability actions (e.g., Lee's criterion), note that the expected SQNR would be around $\text{SQNR} \approx 132 - 20 = 112$ dB, which is acceptable for the considerations discussed in Sect. 9.2.1. Further, investigating and comparing the properties of CRFF and CRFB structures more in detail (Fig. 9.1) and in particular how these could be realized in circuital implementations (i.e., note that in this exercise Active RC integrators are to be used), other two arguments can be pointed out in order to justify the architecture choice made.

The first consideration regards capacitors size [3, 4]. Important to note in Fig. 9.1 is that the integrator unity-gain frequencies are scaled so that the peak signal swing of the first integrator (i.e., the one with unity-gain frequency of ω_1) is approximately one half that of the other two integrators, so that the first integrator limits the last,

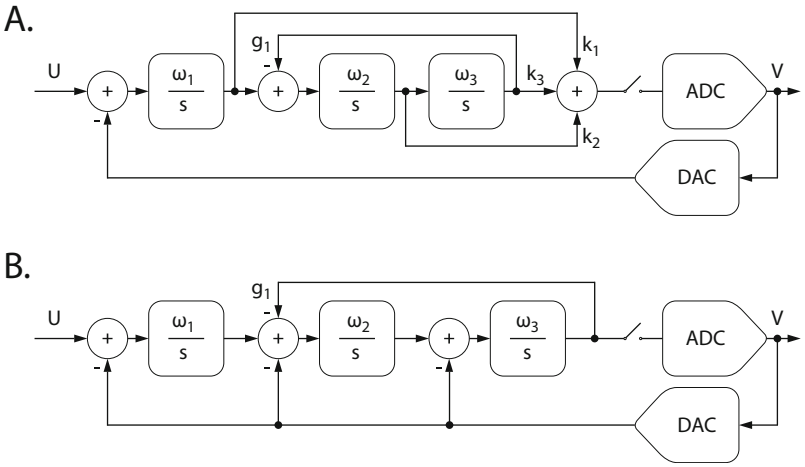


Fig. 9.1 (**a**) CRFF and (**b**) CRFB simplified block schematics

hence aiding recovery of the modulator from overload [9]. Note that no additional provisions are to be made to reset the integrating capacitors. For circuital implementations, the value of the integrating resistor in the first integrator of both architectures is dictated by thermal noise considerations, while the second and third integrators are allowed, theoretically, to use larger resistor values, and the size of the corresponding capacitors can be reduced subsequently. In a CRFF structure, the first integrator is the "fastest" (e.g., least delay due to higher ω_1) while in a CRFB structure is the "slowest" [3, 4, 10], thus the integrating capacitor of the first integrator in a CRFF architecture results much smaller than that in a CRFB structure. Therefore, using a CRFF structure for the CT implementation of the exercise should allow saving both area and power consumption in terms of integrators implementations.

The second argument is based on the fact that to limit noise and distortion, large bias currents in the first op-amp are needed; thus, poles resulting from the finite bandwidth of the op-amp are expected to be at relatively high frequencies [1, 2]. This is advantageous in CRFF modulators since the first integrator has the highest unity-gain frequency and needs to have the least "delay." In a CRFB architecture, actions to ensure that the "extra" poles in the last integrator are not so low that loop stability is compromised must be taken (e.g., the last integrator may need more current). Thus, the large bias currents in the first op-amp, which are needed anyway for noise-related reasons, are more efficiently used in a CRFF structure.

9.2.4 Summary of Design Parameters

Table 9.1 summarizes the design choices made and the theoretical performance expected.

Table 9.1 Design parameters

Parameter	Symbol	Type	Value	Units
Architecture	–	CRFF	–	–
Input sample rate	Fs_{in}	–	48	KHz
Signal bandwidth	BW	–	24	KHz
SQNR	SNR	–	132.87	dB
Sample rate	Fs	–	3.072	MHz
Oversampling ratio	OSR	–	64	–
Dynamic range	DR	–	106.81	dB
Quantizer	M	Mid-rise	3-bit	7-levels
Dither	–	–	None	–
DEM	–	DWA	–	–
Pulse DAC	NRZ	–	–	–

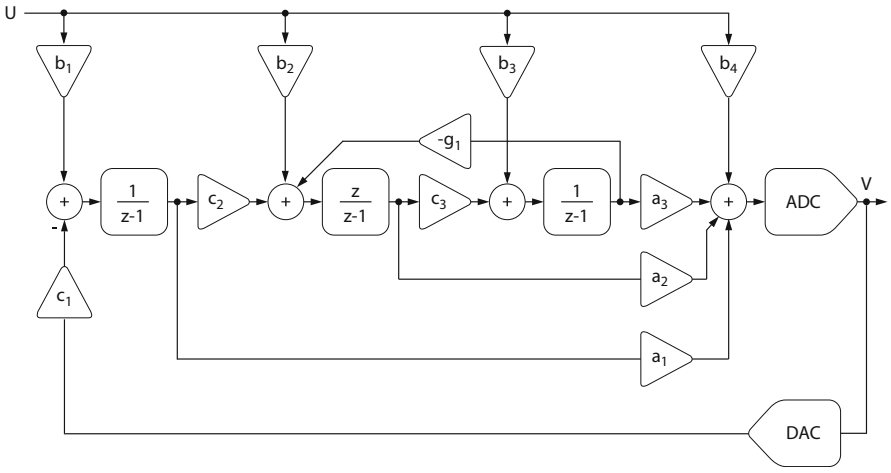


Fig. 9.2 Theoretical MOD3-CRFF architecture

9.3 High-Level Design

Figure 9.2 illustrates the system under consideration.

In order to design the proposed modulator, the out-of-band gain (OBG) of the NTF and the DAC timing of the system need to be specified. As mentioned, a more aggressive NTF than as predicted by Lee's criterion is allowed (e.g., larger OBG), since a multi-bit quantizer is used. However, although a larger OBG reduces the in-band quantization noise, it also leads to an increase in noise due to clock jitter [6, 11, 12]. Therefore, a trade-off exists between quantization and jitter noise. Figure 9.3 shows the peak SQNR and the peak signal-to-jitter-noise ratio (SJNR) as a function of the OBG for a maximally flat NTF of a modulator as the one

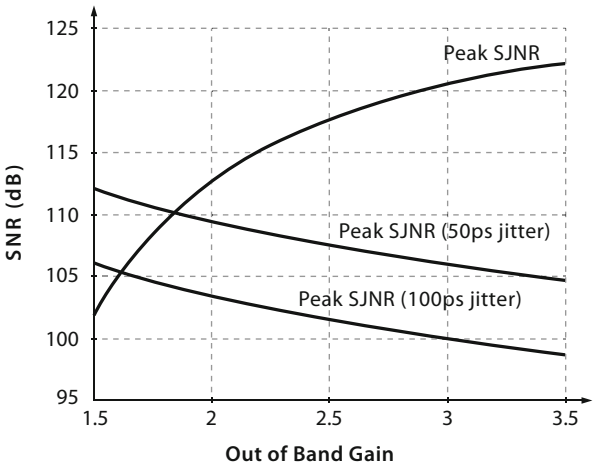


Fig. 9.3 Peak SQNR and SJNR as a function of out-of-band gain

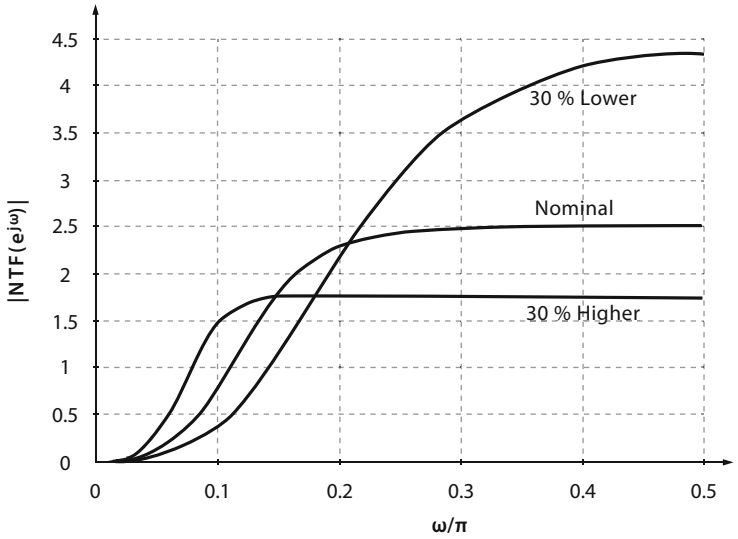


Fig. 9.4 NTFs with a systematic ±30% change in loop filter bandwidth

designed so far [6]. The trade-off between SQNR and SJNR is clearly visible. Further, it should be noted that a large OBG increases the modulator sensitivity to RC component variations and ELD. Since RC time constants are to be affected by process variations and ambient temperature, it is shown [6, 7] that an increase (decrease) in time constants from the nominal value causes the in-band quantization noise to increase (decrease) while simultaneously decreasing (increasing) the OBG (Fig. 9.4).

Therefore, time-constant variations in the loop filter either result in a poor rejection of the in-band quantization noise or in a decrease of the modulator stability

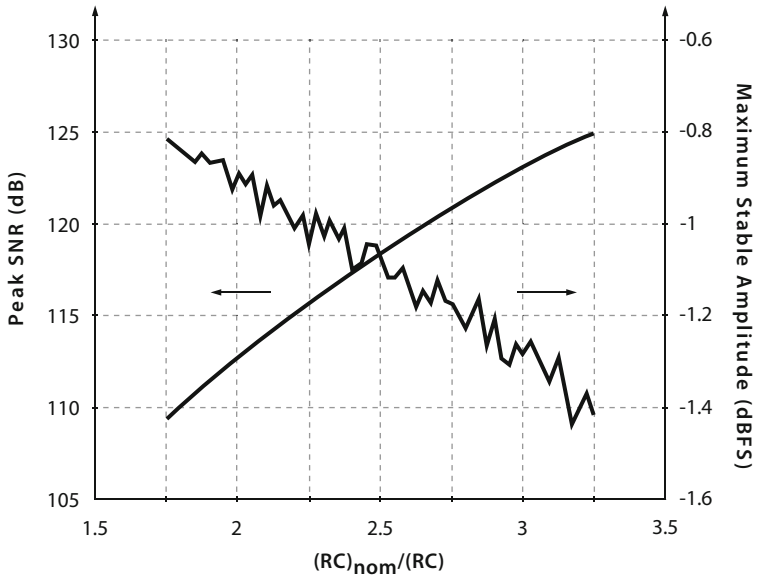


Fig. 9.5 Effect of systematic RC time-constant deviation on the in-band SQNR and MSA of a third-order NTF with OBG of 2.5

conditions (Fig. 9.5). To counteract the increased quantization noise when the time constants become larger, the NTF can be chosen so that the in-band noise satisfies the specification under this worst-case condition. Such strategy allows avoiding, to some extent, the need for an RC time-constant tuning loop [7].

Following the discussion provided, an OBG of 2.5 is suggested for this exercise such that the quantization noise is dominated by the noise due to clock jitter, which is subsequently dominated by the thermal noise of the loop filter. Therefore, at OBG = 2.5, the SQNR, SJNR, and the theoretical behavior of time-constant variations seem to be convenient in order to meet the project specifications under worst-case conditions.

Further, note that the maximum DAC timing for the system is assumed to be [0.5, 1.5], so that the DAC is updated half a clock cycle after the comparator samples its input (e.g., 50% duty cycle). This half-cycle delay would give the real comparator circuit sufficient time to resolve its input and also provides setup time for the DAC. It should be noted that the clock frequency in this project is probably low enough that a modern CMOS process would not require a feedback delay of this magnitude; however, it is nonetheless instructive to see how this delay can be incorporated into the design process. Further, note that, in principle, although any amount of delay (ELD) in the main loop filter can be accommodated by adding a sufficient number of properly timed feedback paths to the quantizer, the sensitivity of the modulator to errors in the coefficients increases as the feedback delay is increased. Therefore, a more accurate delay selection may lead to less sensitive designs [10].

9.3.1 NTF Synthesis

To synthesize the coefficients for the CRFF loop filter, the parameters of Table 9.1 are used.

Type – *CRFF_CTExample* – in the Matlab Command Window to load the variables (i.e., equivalent of the – *load_par* – routine but with adapted specifications).

Type – *mex simulateDSM.c* – for Schreier's routines to function.

Type – *MakeModulator_CTExample.m* – to design the NTF and STF.

Once the code has been compiled, the NTF's poles-zeros, STF-NTF, CT-DT NTF mapping, and SNR graphs should appear, as well as the s-domain transfer function and the coefficients before and after dynamic range scaling.

The successful CT-NTF realization is ensured by the matching of the DT-NTF impulse response with the CT-NTF sample response for the coefficients computed, as shown in Fig. 9.6. Note the half clock delay mismatch in between the first impulse due to the DAC timing that has been chosen and the fact that no ELD compensation is simulated at this stage. However, note that the Toolbox computes the ELD coefficient needed in order to compensate for the half clock cycle delay (e.g., see – *ABCDc*), as reported in Table 9.2.

Moving to the Simulink® model of the modulator – *mod3_CRFF_CTExample.mdl* – a first verification should be conducted simply by simulating the model and ensuring that the time domain results of the scopes display, indicatively, correct behavior (e.g., input sinewave = output sinewave, 3-bit quantization of the output signal, etc. Note that, obviously, the input signal should have an amplitude and frequency not too high nor too low, in order to avoid instability, etc.).

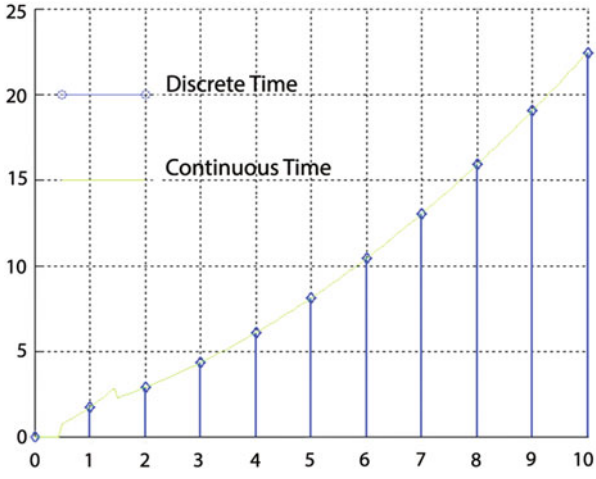


Fig. 9.6 Loop filter impulse/sample response

Table 9.2 Unscaled (left) and scaled NTF coefficients

	a_i	g_i	b_i	c_i	a_i	g_i	b_i	c_i
1	0.8415	0.0014	1	1	3.3196	0.0037	0.2535	0.2535
2	0.8628	–	0	1	5.0244	–	0	0.6775
3	0.2920	–	0	1	4.3631	–	0	0.3898
4	–	–	1	–	–	–	1	–
ELD coefficient	–				−0.7261			

Table 9.3 Unscaled (left) and scaled integrator outputs statistics

	Max	MIin	Mean	Standard deviation	Max	Min	Mean	Standard deviation
Integrator 1	0.343	−0.354	0	0.119	0.087	−0.090	0	0.030
Integrator 2	0.333	−0.343	0	0.105	0.057	−0.059	0	0.018
Integrator 3	0.483	−0.468	0.002	0.155	0.032	−0.031	0	0.010
SUM	1.159	−1.165	–	–	0.176	−0.18	–	–

Having a first confirmation of the correct behavior of the modulator, the next step is to verify that the dynamic range scaling has been performed adequately. As already discussed, dynamic range scaling has to be performed because it ensures roughly the same power level at all nodes, and so it allows to avoid having unnecessarily large noise gain from nodes with small signal levels [1]. Comparing the loop filter coefficients before and after dynamic range scaling (Table 9.2), it is easily seen that the integrator outputs are scaled down since the respective c_i coefficients are no longer equal to 1. Further, note that the b_1 coefficient is set equal to the feedback c_1 coefficient [10]. This ensures a maximally flat STF (i.e., Butterworth) over the passband (i.e., verifiable in the STF-NTF graph).

Further, the range of the signals at the output of the integrators should also be investigated (i.e., by using the routine – *integ_stats(INT1OUT)*, etc.). The results are summarized in Table 9.3 and, as it can be seen, the absolute value of the *Max/Min* signals at the integrator outputs are smaller after DR scaling and in accordance with a 3-bit quantizer, thus ensuring roughly the same power level at all nodes. Important to note is also that the coefficient b_4 is set to 1 in the NTF synthesized since this allows achieving less harmonic distortion [1]. Following the same procedure of Chap. 8, the interested reader is encouraged to repeat the simulations of the output of the integrators in the case of the coefficient b_4 set to 0 (e.g., to do this write – % – in front of the code in line 53 of – *MakeModulator_CTExample*), noting how the same conclusions discussed in the DT exercise are to be obtained. Further, in the model, note that the alternating delaying and non-delaying integrators used help to reduce the noise peaking characteristic of the CIFF/CRFF architectures, hence improving the stability of the modulator [2, 13].

It should be mentioned at this point that since DR scaling is performed in the Schreier's Toolbox using a DT model and since such model only computes the modulator's internal states at the sampling instants, it would be advisable in the case of CT modulators [1, 3, 4, 14] to conduct further simulations to ensure efficient DR scaling once a behavioral version of the CT modulator is developed (e.g., circuit schematic simulations, etc.).

Continuing the NTF analysis, the stable input range of the modulator designed is found to be -1.5 dBFS, as seen by typing – *umax* – into the Matlab Command Window or by calculating [1]:

$$\text{Stable input range} \leq \frac{\text{umax}}{\text{nLev} - 1} = 0.84 \cong -1.51 \text{ dBFS} \tag{9.4}$$

Where umax $= 5.04$ and the number of quantizer levels nLev $= 7$.

To verify the SQNR, the routine – *sweep_sinamp* – should be used, initially, in order to be able to compare the results for different input amplitudes. Typically, a well-designed modulator should perform close to the theoretical expectations *at least* up to a -3 dBFS input signal, such as:

$$\text{SQNR} = 132.87 - 8 - 3 = 121.87 \text{ dB} \tag{9.5}$$

Where the approximate value of -8 dB is due to the OBG $= 2.5$.

Moreover, note that the routine – *sweep_sinamp* – should be performed for an input signal of frequency as close as possible to the signal bandwidth of the converter while trying to hit as best as possible an FFT bin. Figure 9.7 illustrates the results for a 12 KHz sinewave input signal, and, as it can be seen, the peak SQNR and the SQNR for a -3 dBFS input signal results, approximately, 119 dB and 118 dB, respectively.

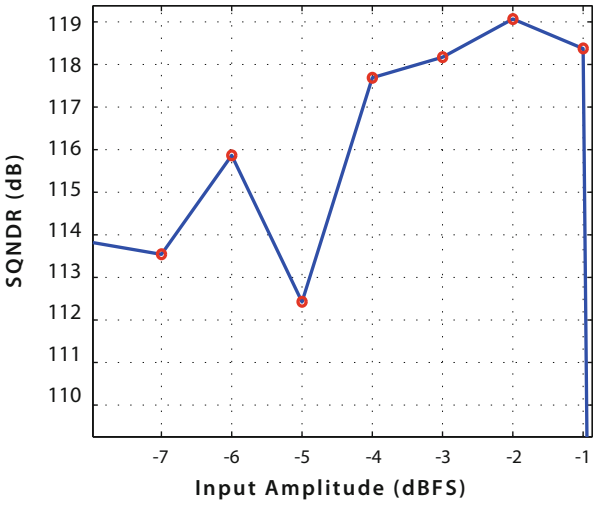


Fig. 9.7 12 KHz sine amplitude sweep SQNR peak

These values are only a few dB lower than what expected from theoretical calculations; thus the modulator performs satisfactorily considering that the low values found can be attributed to FFT inaccuracies and others simulation's non-idealities. Another important thing to note is that at least for slowly varying signals, the modulator seems to be able to maintain stability up to −1 dBFS. Note, however, that the value found from the *sweep_sinamp* routine is indicative and further tests are needed to confirm such value.

A final verification of the NTF synthesized should also be conducted through the – *mod_SNDR* – routine and the SNR graph obtained from the – *MakeModulator_CTExample* – routine at the beginning of the paragraph. As it can be seen in Fig. 9.8, the DR of the modulator has been found around 110 dB, thus meeting the target set for the project.

Figure 9.9 illustrates the power spectral density graph of a 12 KHz sinewave at −3 dBFS in amplitude, and it results clear that the NTF behaves as expected, specifically:

- Slope of 60 dB/dec, hence third-order filter.
- At very high frequencies, the NTF flattens due to the reduced out-of-band gain (i.e., Lee's criterion applied).
- Notch at the edge of passband due to zero spreading and correct increment in SQNR thanks to the resonator (e.g., about 8 dB) when compared to the same NTF with all zeros at DC (e.g., verifiable by setting the *g(1)* coefficient to zero in the Simulink® model).

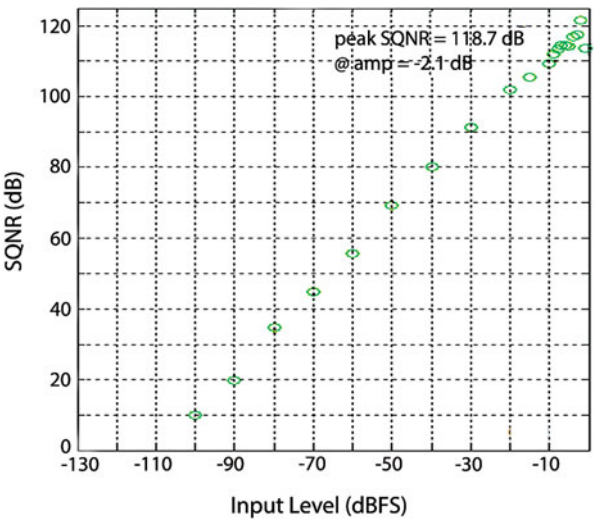


Fig. 9.8 12 KHz sine amplitude sweep SQNR peak

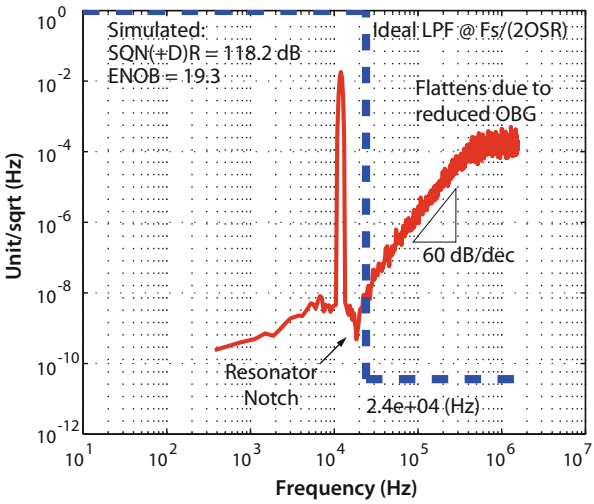


Fig. 9.9 PSD for a sinewave of 12 KHz, amplitude −3 dBFS

Table 9.4 DC simulations

DC input	0	0.25	0.5	0.75	0.8	0.83	0.88	0.4367	$\pi/10$
Mean	5.293e−07	0.25	0.5	0.75	0.8	0.83	0.88	0.4367	0.31416
						Dithered input limit	*Un-dithered input limit*		

9.3.2 DC Behavior

Once the design satisfies the simulations discussed in the previous paragraph, the modulator stability, accuracy, and tonal behavior for DC inputs should be investigated [14]. Many different DC values should be tested, using both the – *mod_SNDR* – routine and the time scopes at each sensible node of the modulator. Note that particular attention should be paid to rational DC values, as more prone to creating long repeating sequences leading to tones.

Table 9.4 summarizes the results for some DC input values, and, as it can be seen, the mean output of the modulator shows that the converter is able to accurately represent the analog DC inputs with an error of approximately 10^{-7} at DC = 0. This can be considered accurate enough for a converter aimed at audio applications, as the one of this exercise. The maximum stable input is found at an amplitude of 0.88, hence slightly higher than what found in Eq. 9.4. This result is probably due to the larger OBG used. In fact, as it will be shown later, for fast varying signals, the actual un-dithered input limit matches Eq. 9.4 quite accurately. Although no tones in the baseband should be detected, it is important to note that they always appear at very high frequencies. This limits the robustness of the modulator to instabilities [1]. In fact, although such tones are out of the passband and so are to be filtered out by the

decimation filter, they can still lead to instability if they grow too much in magnitude and exceed the OBG limit set in the design. Given the magnitude of these high-frequency tones, as well as the higher OBG used in this design, it results clear that dither has to be implemented in order to counteract this phenomenon. A quick investigation using a dither value of 0.1 has been conducted, and, according to the simulations, the maximum stable input has been found for a DC of 0.83 while eliminating all the tones.

9.3.3 Stability and Tonal Behavior

Modulator stability and satisfactory tonal behavior within the stable input range have also to be confirmed for, at least, slow varying input signals such as sinewaves. A full set of simulations of multiples of the reference frequency of 12 KHz (e.g., 6 KHz, 3 KHz, etc.), as well as at amplitudes for each tested frequency of, for example – 0.0625, 0.125, 0.25, 0.5, 0.707, 0.85, 0.91, and 0.92 – should be conducted [14], both without and with dither. As for the DC tests, the routine to be used is the – *mod_SNDR* – coupled with time scopes investigations. Important to note is that since it is not possible to test the frequency of 24 KHz, as too close to the limits of the converter, the frequency of, for example, 19.5 KHz can be used instead. This frequency value does not fall in an FFT bin, and so the numerical results it produces (e.g., SQNR, etc.) are to be affected by inaccuracies (i.e., note that choosing a sensible frequency which still performs relatively well, as in this case, is suggested in order to limit possible confusion between actual modulator's issues and simulations inaccuracies). However, testing the modulator with signal frequencies close to the converter bandwidth results essential to investigate NTF stability, especially in high-order CRFF architectures where the noise at high frequencies coupled with a high-frequency signal can often lead to instability.

Performing the range of simulations discussed with and without the use of dither, the stable input range is found at amplitudes of 0.88 and 0.83, respectively, hence in accordance with the DC results. Note that for sinewaves, the actual stable input range is actually slightly larger; however, the peak SQNR is found at the amplitudes reported. Further, as seen during DC simulations, the modulator is not always stable for constant inputs larger than the amplitudes considered. In order to remove all tones from the spectrum, a dither of 0.1 has been used, which decreases the achievable SQNR only by a few dB, hence not compromising the validity of the design thanks to the tolerance allowed from the minimum specifications (i.e., use – *sweep_sinamp* – to verify this point. Be careful that a good design should perform well *at least* up to a −3 dBFS input signal. For this exercise, note that the modulator hit the minimum SQNR required up to a −20 dBFS input signal).

To ensure modulator stability within the stable input range found of ±0.83, further tests should always be conducted for a ramp, step, and chirp input signals [4, 14]. Figure 9.10 illustrates the results for the ramp and step signals, respectively. Note that the step used has an amplitude of 0.86, instead of 0.83, in order to test the modulator under the most stressful conditions possible (i.e., fast varying and slightly

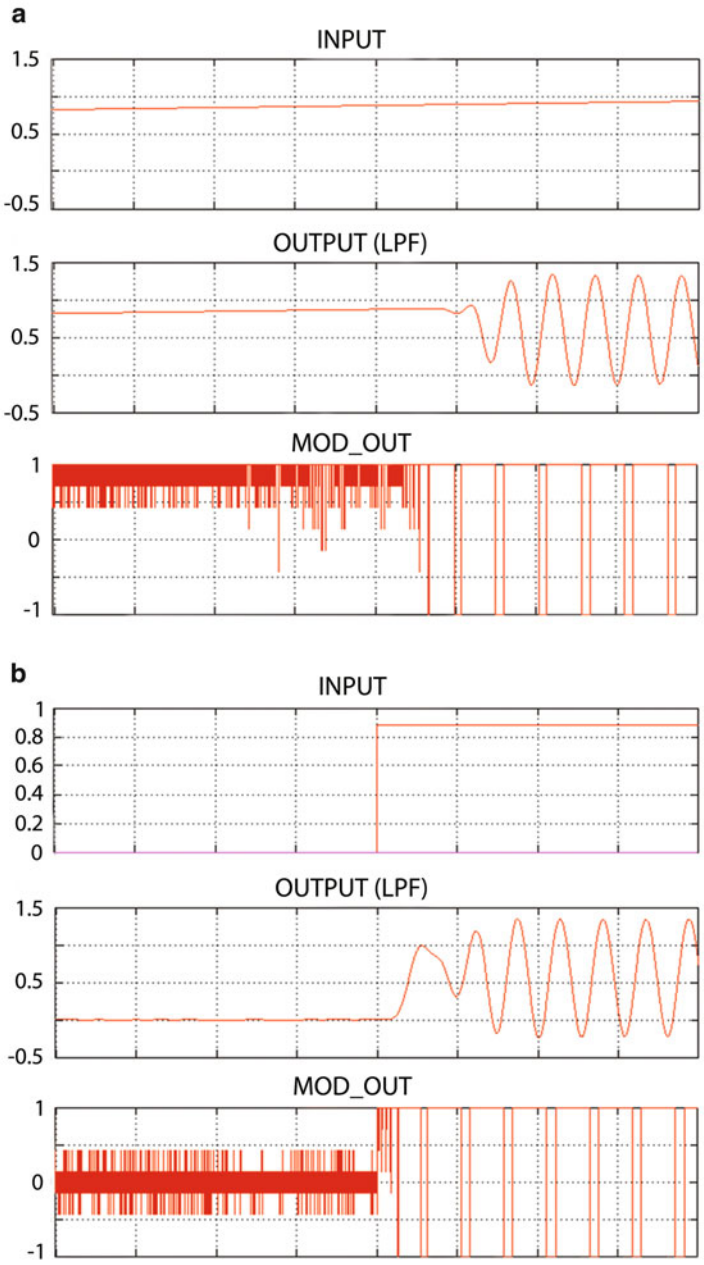


Fig. 9.10 (*Top*) ramp and (*bottom*) step time scope results

large signals). As it can be seen, it takes several cycles for the modulator to become unstable. This further confirms that by using an amplitude of just ±0.83, the modulator should be ensured to remain stable.

9.3.4 Integrator Finite Gain and Saturation

The effect of low, finite DC integrator gain should also be investigated (e.g., use the – *sweep_dc_dc* – routine), especially in those designs where limited gain is available and a sensible architecture is implemented (e.g., first stage of MASH modulators, etc.). For the CRFF modulator designed, it is found that without dither (e.g., *mod.dither = 0*) and up to an integrator gain of about 5 (e.g., *mod.igain1 = 5*, *mod.igain1 = 5, mod.igain1 = 5,* etc.), the width of the dead zone at zero (i.e., DC) depends on a power of the gain of 10^3, as expected by a third-order modulator [1]. Important to remember is that dither helps to recover dead zones and smooth the overall SQNR response; thus the tolerance to finite integrator gain results improved in the modulator designed. Therefore, achieving a high enough DC integrator gain to minimize dead zones and hence achieve a reasonable converter accuracy shouldn't represent a significant issue for real implementations of the modulator designed.

The behavior during finite integrator saturation in the case of sinewave inputs should also be investigated by setting – *mod.isat1 = 1, mod.isat2 = 1, mod.isat3 = 1*, – and running the – *sweep_sinamp* – routine, while dither is applied. The results should show that the modulator still achieves a satisfactory peak SQNR for the specifications set, thus not representing a significant problem for a real implementation of the modulator [4].

9.3.5 Feedback DAC and Element Mismatch

Due to the use of a multi-bit quantizer, it is necessary to test the modulator performance when the feedback DAC non-idealities are taken into consideration [10]. The set of simulations for different frequencies and amplitudes conducted in paragraph 9.9.3.3 should be repeated for the Simulink® model – *DAC_mod_CRFF_CTExample*. These tests should be conducted according to the specifications of the technology of implementation available; however, an element mismatch of 0.25% is generally a good worst-case scenario for modern IC process (e.g., set – *dac.elements = [1.005 1.0025 0.9975 0.998 1.0015 1.001 0.9977]* – in the model).

Performing the simulations with a mismatch of 0.25% should reveal that the modulator is able to perform relatively well when DWA and dither are enabled, with an SQNR at −3 dBFS of about 116 dB. Further, no significant tones should be detected, either in-band or outside the band of interest. Therefore, the modulator and the DWA technique implemented can be considered adequate in terms of tonal behavior, and further simulations on the matters would rather be performed on actual circuital simulations instead of at high level.

Table 9.5 Qualitative overview of loop filter coefficients

Coefficient	Defines	Influence
a_i	System poles	NTF aggressiveness and filter stability
g_i	NTF zeros	In-band attenuation
b_i	STF zeros	Interferer rejection
c_i	Interstage gains	System scaling and integrators saturation limit

9.3.6 Coefficients Mismatch

The effect of the combined coefficients mismatch and a DAC elements mismatch of ±0.25% while dither is applied should also be investigated [3]. As seen in Chap. 8 and as summarized in Table 9.5, it can be easily seen that the modulator performance is to be affected mainly by changes in 'a_i' coefficients, as these define the NTF.

Following the same test procedure conducted in Chap. 8, similar behavioral results should be obtained. Obviously, various mismatch combinations exist, each providing a different response. By performing few simulations, it should result clear, however, that the modulator is able to meet the SQNR required, allowing approximately between 5 and 10 dB tolerance from minimum specifications when a full-scale input signal is used. This can be considered an adequate margin for the thermal noise budget to be managed in the case of a circuital implementation of the design. Important to note is also that the maximum stable input range should be verified when all non-idealities are considered and it should be found that the modulator meets the specifications.

9.3.7 Inter-symbol Interference

The Simulink® model to be used for this exercise is *Complete_DAC_mod3_CRFF_CTExample,* where the effects of the unequal rise/fall time that may occur in the feedback DAC pulse, causing inter-symbol interference (ISI) , can be simulated by using the rate limiter provided (i.e., as suggested in [15]). Since such error is signal dependent and causes even harmonic distortion tones to appear in the output spectrum, a full set of simulations of multiples of the reference frequency of 12 KHz, as well as at amplitudes for each tested frequency of 0.0625, 0.125, 0.25, 0.5, 0.707, and 0.83, should be conducted, while a dither of 0.1 is implemented, and an unequal rise/fall time of 5, 10, 20, and 30% is applied (i.e., so according to the percentage value obtained out of the timing period of the modulator, the relative rate limiter slew-rate reduction should be calculated and used in the Simulink® model). It should result that no significant tones appear in the spectrum when dither is applied. The interested reader may wish to repeat the experiment without dither and see how much more tones appear due to ISI. Important to note is that the unequal rise/fall time of the DAC pulse tends to lower the SQNR of about 0.5–1.5 dB, depending on how much mismatch is used. However, the degradation is so small that does not constitute a problem for the modulator

designed. For this project and from the test conducted, however, it is advisable to not exceed an unequal rise/fall time of ≈25% in order to ensure a satisfactory modulator performance, especially in terms of tone behavior for worst-case conditions where other mismatches and non-idealities are included. Further, the reader should find that the results are in accordance with the theoretical predictions, where the maximum asymmetry allowed for the desired SNR results [2, 11]:

$$\tau \leq \frac{4T_s\sqrt{\text{OSR}}}{\text{SNR}_{\text{desired}}} = 88.13\ (\text{ns}) \tag{9.6}$$

where SNR desired = 118 (dB) and Ts = 1/Fs = 325 (ns). It is easily seen that for 25% asymmetry $\tau \leq 25\ \% * T_S = 81.225$ (ns), the modulator is still able to perform relatively well, while at greater asymmetries, the modulator tends to lose too much resolution, and significant tones tend to appear which, in some cases, may compromise the stability of the converter.

9.3.8 Excess Loop Delay

The effect of excess loop delay of the feedback DAC pulse on the modulator should also be studied by changing the amount of delay at which the feedback DAC pulse is supplied to the input of the first integrator while recording the output SQNR (e.g., *mod_SNDR*) for the different delay amounts. The model to be used for investigating ELD is *Complete_DAC_mod3_CRFF_CTExample*. Further, since the maximum ELD which can be compensated is up to one clock cycle, note that in the Simulink® model, an array of delay blocks has to be used to obtain fractional delays. The interested reader should refer to [16] for an explanation on how delays smaller than a clock cycle can be implemented in Simulink®.

Considering that the clock period is [2, 11, 12]:

$$T_S = \frac{1}{f_{\text{clk}}} = \frac{1}{3.072(\text{MHz})} \cong 325\ (\text{ns}) \tag{9.7}$$

The results summarized in Fig. 9.11 illustrate that the modulator is able to perform adequately in terms of SQNR (e.g., ≥112 dB) without ELD compensation up to, approximately, a 32% delay of the clock cycle (e.g., ≈100 ns). If in circuital implementations such limit should be exceeded, the use of an additional feedback path with the coefficient computed by the Schreier's Toolbox (Table 9.2) directly connected to the quantizer input (as discussed in Chap. 7) in order to cancel the extra coefficient introduced by ELD in the NTF can be used [1, 2]. As it can be seen, the compensated modulator is then able to perform adequately up to the maximum DAC timing considered in the initial design of [0.5, 1.5], which is, half clock cycle delay (e.g., 50% delay). Therefore, it can be said that the modulator does not seem to be too sensitive to ELD, and if an accurate enough clock source and a fast enough comparator are provided, compensation can be avoided. This saves a substantial amount of area as the secondary DAC needed for compensation can be omitted.

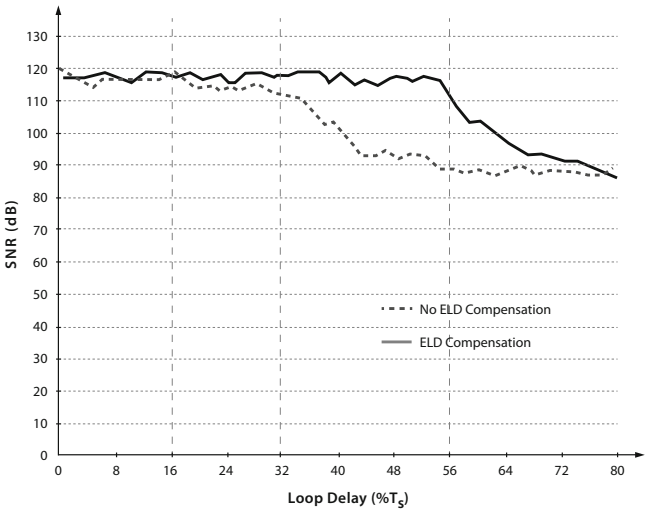


Fig. 9.11 Effect of ELD on the achievable SQNR with and without compensation

9.3.9 Jitter Behavior

The robustness of the modulator to jitter errors has to be investigated in CT modulators. From a high-level perspective, according to [17, 18], the effect of jitter can be modeled as an additive sequence at the input of a jitter-free modulator. In the case of the NRZ pulse considered for this work, the error sequence can be derived as [2, 19]:

$$e_{\text{jitter}}(n) = [y(n) - y(n-1)]\frac{\Delta T_{\text{DAC}}(n)}{T_s} \tag{9.8}$$

where $y(n)$ is the *nth* output sample of the modulator, T_s is the sampling time, and $\Delta T_{\text{DAC}}(n)$ is the clocking uncertainty of the *nth* edge of the DAC which is assumed as independent and identically distributed random variable.

In order to implement such jitter error sequence in Simulink®, the arrangement suggested in [18] can be used, as incorporated in the – *Complete_DAC_mod3_CRFF_CTExample* – model. The jitter model simply introduces random Gaussian noise after the DAC according to Eq. 9.8.

As it can be seen in Fig. 9.12, the modulator behavior to clock jitter is in accordance with the Fig. 9.3 used at the beginning of the design. Note that the slightly lower SJNR found is due to the use of a dither of 0.1. As it could be expected, the modulator is sensitive to jitter errors which substantially degrade the achievable SQNR since these errors are not processed by the loop filter due to the DAC position in the loop (e.g., more than 10 dB loss for 100 ps jitter). Further, note how the introduction of jitter in the system flattens the spectrum in the passband.

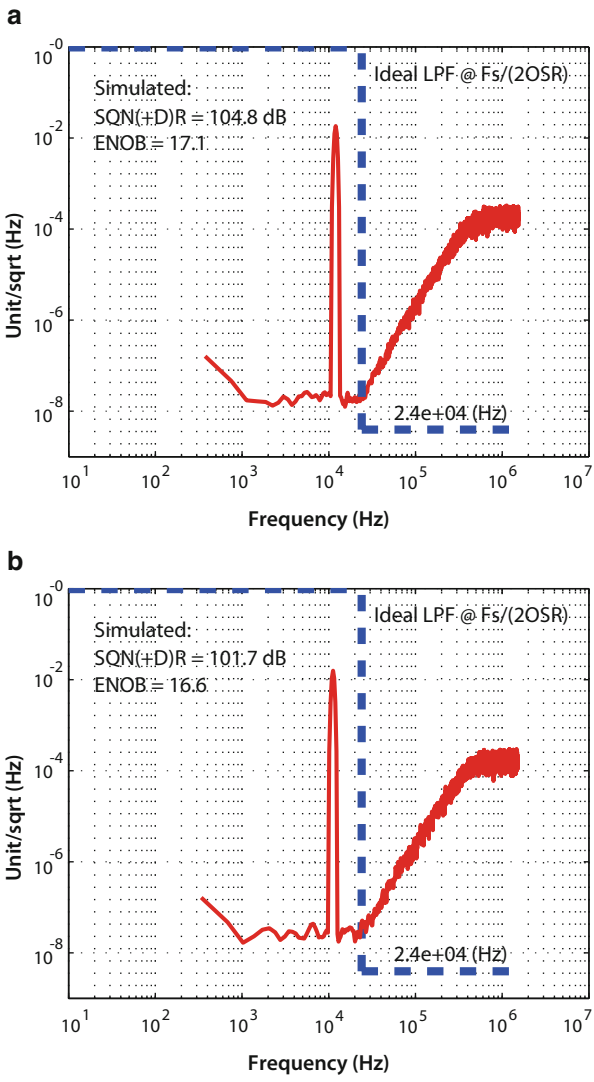


Fig. 9.12 (*Top*) SJNR for 50 ps jitter and (*bottom*) 100 ps Jitter 12 KHz sinewave, amplitude –3 dBFS

According to the results obtained and as expected from initial considerations of the design, it is advisable for this project to use a clock signal which does not exceed, approximately, 50 ps jitter in order to allow some tolerance for further degradations from other non-idealities.

9.3.10 Total Harmonic Distortion

To investigate the THD of the system, the routine – *THD.m* – should be run, by typing it into the Matlab Command Window. Note that THD should be run for a full-scale input (e.g., −1.6 dBFS), while dither is used in the modulator. Note that the THD routine's results are also dependent on the frequency of the sinewave input used. For a 12 KHz sinewave input, the results should display a THD of approximately −90 dB, which just meets the minimum required of −90 dB; therefore the designed modulator should be further tested at a circuit level to ensure its ability to meet such specification. However, as stated in Chap. 8, such estimation is very approximate and should be used only as a guideline. For a more precise estimation, it is suggested to test the THD at a circuit level and for a much larger number of harmonics.

9.4 Conclusions

A final test which includes as many non-idealities as possible should be conducted to ensure that the modulator is able to meet the project specifications under a worst-case scenario. Specifically, a dither of 0.1, a DAC mismatch of 0.25% (i.e., with DWA enabled), a coefficient mismatch of 30%, a jitter of 50 ps, and an unequal rise/fall of 20%, as well as a realistic integrator finite gain and saturation limit, should be set for such worst-case test. Assuming a good enough clock is available, ELD can be avoided in the simulation. Under these worst-case conditions, a full set of simulations of multiples of the reference frequency of 12 KHz and at different amplitudes, as well as for DC inputs, should be performed. As for the coefficient mismatch paragraph, note that various results are obtainable, depending on the random variation of the coefficients. Figure 9.13 illustrates one of the worst cases found, where the a_i coefficients were increased positively.

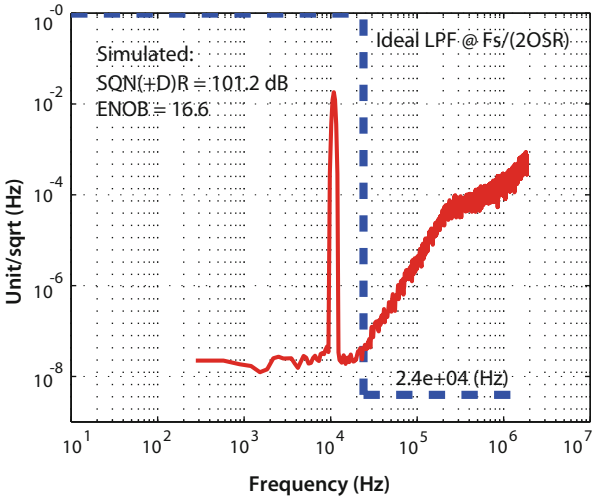


Fig. 9.13 Worst-case SQNR, 12 KHz at −3 dBFS

Table 9.6 Final modulator performance

Parameter	Value	Comments
Order/architecture	3rd/ CRFF	
Bit	3	Mid-rise quantizer
Bandwidth (KHz)	24	
Input sample rate (KHz)	48	
Sampling rate (MHz)	3.072	
OSR	64	
SNR at full scale (dB)	118	Ideal MOD with dither
	≈108	With all non-idealities and dither
Maximum stable input	±0.83	–1.6 dBFS
THD (dB)	−90	At full scale
Dither	0.1	
DEM	DWA	

As it can be seen, the modulator is still able to meet the desired resolution for the reference tone of 12 KHz at −3 dBFS. As already discussed, note the rise of noise in the passband due to clock jitter, the introduction of some tones at high frequencies due to mismatches, and the increase of noise at very high frequency due to the positive increase of the a_i coefficients. Besides this example, however, from the study conducted resulted that the modulator is able to meet the desired resolution where the average SQNR found for different conditions resulted to be around 108 dB for full-scale inputs, hence providing a tolerance from minimum specifications of approximately 10 dB which results appropriate in order to manage further degradation from the circuital realization and noise budget. Further, no tones were observed in the baseband for all different input tones tested.

To conclude, Table 9.6 summarizes the characteristics and performance of the modulator designed. As it can be seen, the project specifications are met; therefore the design can be considered successful from a high-level point of view. The next steps in the design process would be to perform calculations for thermal noise in order to be able to size the capacitors and resistors needed for the realization of the coefficients. Then, the circuital model of the modulator should be designed and simulated.

Exercises

Q.1
Briefly state the main steps of the high-level design process of a CT ΣΔ-converters.

Q.2
For the design example considered in this chapter, list at least other three different theoretical designs. Support your answer with some minimum analysis.

Q.3
Design the NTF of an audio, single-bit, fifth-order CT ΣΔ-modulator with OSR = 32, NRZ DAC pulse, and OBG = 1.5 in the case of a CRFB architecture. Note to set the input feedforward coefficients b_2, b_3, b_4, and b_5 to zero. Plot the pole-zero graphs.

Q.4
Map, in Simulink®, the synthesized NTF in Q.3 to a CRFB topology. Simulate the modulator, and estimate the in-band SQNR and the MSA. Briefly analyze your results.

Hint: If the Schreier's Toolbox is used to compute the coefficients, make sure the Simulink® model refers to the same coefficients nomenclature. Therefore, check the Schreier's Toolbox manual for the appropriate nomenclature.

Q.5
Calculate the maximum asymmetry allowed for the feedback DAC pulse in order to retain the SQNR found in Q.4 of 83 dB.

Q.6
What changes would you make to the Simulink® schematic if the modulator in Q.3 would require some sort of ELD compensation?

Q.7
List some of the reasons to convince your manager that a DAC pulse different from the NRZ would be beneficial in a CT modulator. How would you mitigate the errors arising due to the modulator being sensitive to timing and jitter errors?

References

1. Schreier R, Temes GC. Understanding delta-sigma data converters. Hoboken: John Wiley & Sons Inc; 2005.
2. Ortmanns M, Gerfers F. Continuous-time sigma-delta A/D conversion: fundamentals, performance limits and robust implementations. Berlin Heidelberg: Springer Berlin Heidelberg; 2006.
3. De la Rosa JM, Del Rio RF. CMOS sigma-delta converters: practical design guide. London: John Wiley & Sons, Ltd; 2013.
4. Casier H, Steyaert M, Van Roermund AHM. Analog circuit design: robust design, sigma delta converters, RFID. London: Springer; 2011.
5. Reiss JD. Understanding sigma-delta modulation: the solved and unsolved issues. JAES. 2008;56(1/2):49–64.
6. Pavan S, et al. A power optimized continuous-time ΔΣ ADC for audio applications. IEEE J Solid State Circuits. 2008;43(2):351–60.
7. Pavan S, Sankar P. A 110uW single bit audio continuous-time oversampled converter with 92.5dB dynamic range. In: IEEE, ESSCIRC '09, proceedings of, Issue Date: 14–18 September 2009.
8. Prasad A, et al. A 120 dB 300mW stereo audio A/D converter with 110 dB THD+N. In: Proceddings of the IEEE European solid-state circuits conference, p. 191–194, September 2004.

9. Geerts Y, Steyaert M, Sansen W. Design of multibit delta sigma A/D converters. New York: Kluwer Academic Publisher; 2002.
10. Norsworthy SR, Schreier R, Temes GC. Delta-sigma data converters: theory, design and simulation. New York: Wiley-IEEE Press; 1996.
11. Cherry JA, Snelgrove WM. Continuous-time delta-sigma modulators for high- speed A/D conversion: theory, practice and fundamental performance limits. London: Kluwer Academic Publishers; 2002.
12. Breems L, Huijsing J. Continuous-time delta-sigma modulation for A/D conversion in radio receivers. London: Kluwer Academic Publishers; 2001.
13. Candy JC, Temes GC. Oversampling delta-sigma data converters: theory, design and simulation. New York: Wiley-IEEE Press; 1991.
14. Medeiro F, Verdù BP, Vazquez AR. Top down design of high performance sigma delta modulators. Boston: Kluwer Academic Publisher; 1999.
15. Bourdopoulos G, et al. Delta-sigma modulators: modelling, design and applications. London: Imperial College Press; 2003.
16. MathWorks Inc. Variable fractional delay. [Online] https://it.mathworks.com/help/dsp/ref/variablefractionaldelay.html; 2016.
17. Hernandez L, et al. Modelling and optimization of low pass continuous-time sigma- delta modulators for clock jitter noise reduction. In: Proceedings of the 2004 international symposium on circuits and systems, ISCAS; 2004.
18. Rypestol L. Design and modelling of a high resolution, continuous-time delta-sigma ADC: in-depth noise considerations and optimization. NTNU, MSc Thesis, 2011. [Online] http://brage.bibsys.no/xmlui/bitstream/handle/11250/255454/438128_FULLTEXT01.pdf?sequence=2&isAllowed=y.
19. Reddy K, Pavan S. Fundamental limitations of continuous time delta sigma modulators due to clock jitter. IEEE Trans Circuits Syst I Regul Pap. 2007;54(10):2184–94.



Frontiers of Sigma-Delta Modulators 10

Since the introduction of Sigma-Delta modulation as Analog-to-Digital conversion technique, there has been an impressive amount of ICs published, comprising many diverse technology processes, architectures, and circuit techniques, as well as targeting a variety of different applications [1]. To conclude the book, this section provides a basic introduction on emerging Sigma-Delta ICs typically implemented in nanometer (i.e., beyond 180 nm) CMOS technologies [2]. These include sturdy MASH (SMASH), incremental, and time domain VCO-based modulators. Similarly to the other chapters of the book, the respective Simulink® models provided in the Toolbox of the architectures presented are described and relevant simulations suggested. The hope is to instill in the reader some curiosity as these Sigma-Delta modulator implementations are an active and vibrant area of contemporary research.

10.1 SMASH ΣΔ-Modulators

One of the most significant disadvantages of MASH ΣΔ-architectures is the issue of matching requirements in the loop analog filters and digital cancellation filters at the output of individual stages [3]. In modern designs, this problem is further worsened by both the implementation of low voltage technologies and the need to keep the circuit requirements as relaxed as possible (i.e. using low-gain op-amps to reduce power consumption, allowing maximum tolerance to the inaccuracies in analog circuitry components and digital filters if used, etc.). To address these problems, a relatively recent architectural development is the sturdy MASH (SMASH) modulator. SMASH modulators can be considered to have sort of hybrid properties between the stability problems of single-loop modulators and the gain sensitivity issues of MASH architectures [4].

Figure 10.1 shows a general form of the SMASH structure, and, as it can be seen, the main advantage is that the digital filters are removed [4–6]. The quantization noise of the first stage is fed to the second stage, while the output of the second stage

© Springer International Publishing AG, part of Springer Nature 2019

I. Arnaldi, *Design of Sigma-Delta Converters in MATLAB®/Simulink®*,
https://doi.org/10.1007/978-3-319-91539-5_10

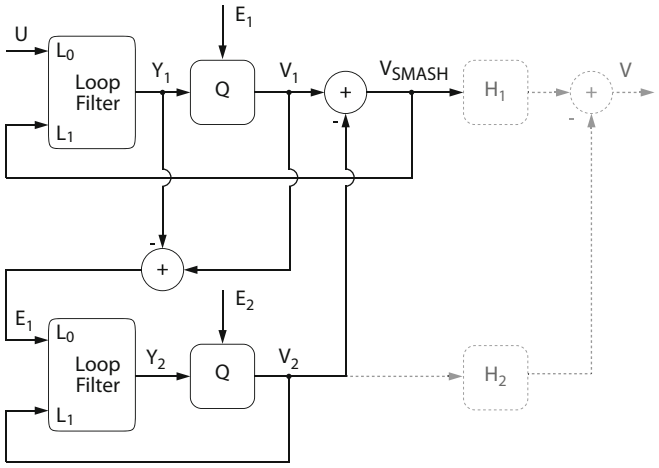


Fig. 10.1 General SMASH architecture and differences (dotted blocks) from traditional MASH

is directly subtracted from the output of the first-stage quantizer in the digital domain. Therefore, since the analog NTFs are not approximated in a digital recombination path for the quantizer outputs, it results that the NTF processing required for recombining the quantizer outputs is obtained by reusing the analog loops. In other words, rather than generating a term such as $\mathrm{NTF}_1 * E_2$ using a digital representation of NTF_1, as in MASH structures, the second-stage quantization error is actually processed using the first-stage modulator loop. Analyzing mathematically the block schematic of Fig. 10.1, it is seen that the quantization noise E_1 is noise shaped, such as:

$$V_{\mathrm{SMASH}} = \mathrm{STF}_1\mathrm{U} + \mathrm{NTF}_1(1 - \mathrm{STF}_2)E_1 - \mathrm{NTF}_1\mathrm{NTF}_2E_2 \tag{10.1}$$

According to Eq. 10.1, the quantization noise of both stages can be shaped to the same order if the following condition applies:

$$\mathrm{NTF}_2 = (1 - \mathrm{STF}_2) \tag{10.2}$$

Thus, if the second stage has unity STF, then the quantization noise of the systems is completely eliminated [5, 6]. As seen in previous chapters, the concept of unity STF is based on adding an extra feedforward path from the input node to the quantizer input of each stage, so that the integrators ideally process quantization error only, which reduces their output swings [4]. Note that, generally, there are no specific constraints on the signal and noise transfer function of the first loop.

Processing the second-stage quantization error using the first modulator loop ensures that the same NTFs are seen in the signal processing and recombination paths, resulting in a modulator much more tolerant to low integrator gain, path gain errors, etc. [4–6]. However, the downside of injecting later-stage quantization errors back into earlier modulator stages is that long loops of integrators are created and

stability is not guaranteed. Further, the price to be paid in SMASH structures to shape all quantization errors with the overall modulator order is an additional DAC at the input of the modulator and a slight increase in the swing of the integrators. Note that the DAC, generally, has to have a double full scale and one more bit than the largest of the resolutions of the ADCs in the cascade.

The reader is encouraged to experiment with the SMASH converter variation of the MASH converter model *"mod3_2plus1_MASH_v2"* provided in the Simulink® Toolbox.

In the Toolbox open the model *"mod3_2plus1SMASH.mdl"* found in the folder *10_SMASH*.

Type – *load_par* – in the MATLAB Command Window to initialize variables needed for the model to function and run the simulation. Analyze the time domain results and output spectrum.

In particular, the reader may wish to compare the SMASH modulator sensitivity to effects such as finite integrator gain and path gain errors with the respective MASH version (i.e., same order and stage's architecture). The reader should find it to be much less sensitive. Next, the reader should change the "*Gain5*" and "*Gain6*" in the Simulink® schematic both to 1 and investigate what happens to the loop stability, recalling that the MASH variant is always stable.

To conclude, it should be noted that removing the digital filters of traditional MASH architecture, hence reducing the sensibility of the SMASH architecture to mismatches, means that SMASH architecture can also be designed as CT system more easily. Therefore, SMASH CT modulators should, theoretically, be able to achieve higher operational bandwidths while reducing power consumption.

10.2 Incremental ΣΔ-Modulators

To achieve high-resolution, Sigma-Delta ADCs trade time for resolution without the need for accurate component matching. However, their input has to be a highly correlated narrowband signal [3]. In applications such as instrumentation and measurement (e.g., smart sensors, portable weight scales, digital multimeters, etc.), biomedical (e.g., electromyography (EMG), electroencephalogram (EEG), electrocardiogram (ECG), neural recording, etc.), biological, etc., the signal under conversion is typically weakly correlated, and sometimes it requires multiplexing. Further, the frequency band of the input signal can be extremely narrow, often only a few Hertz wide. Therefore, if in applications such as telecommunication, audio, etc., the output spectral properties of the digitized running input waveform are of interest, in other applications such as the one mentioned above, an accurate sample-by-sample mapping is typically required instead, as well as high absolute

accuracy and very good integral nonlinearity [1] performance. Therefore, it should be apparent that conventional ΣΔ-ADCs are not suitable for multichannel applications since the digital output is an average of many input samples. Without the one-to-one mapping between input and output samples, they can hardly be multiplexed between channels, with the exception of applying special techniques [7–10].

To allow multiplexing without the need of recurring to special techniques, which often is difficult to implement and not convenient in terms of area, cost, and power consumption, ΣΔ-ADCs can be implemented as incremental ΣΔ-modulators (IΣΔ). An incremental ΣΔ-ADC is structurally a ΣΔ-modulator in which the analog loop filter and digital filter are periodically reset at each conversion [7]. Therefore, unlike conventional modulators that obtain one digital output using many input samples, IΣΔ-ADCs are memory-less systems that convert one input sample to one digital output (i.e., eliminates the memory of the previous sample). Since the operation of such converters is intermittent, they act as high-resolution Nyquist-rate ADCs and are well suited to process multiplexed signals. Further, they can provide precise high-resolution conversion with low offset and gain errors and, for the measurement of DC input signals, have the advantage of requiring simpler digital post-filters compared to conventional ΣΔ-converters.

A general block diagram of IΣΔ-ADC is shown in Fig. 10.2. Instead of running continuously, the ΣΔ-modulator in incremental mode is clocked at an oversampling rate of $f_s \cdot \mathrm{OSR}$ for M cycles and then its loop filter is reset. The modulator output, at a rate of $f_s \cdot \mathrm{OSR}$, is sent to the digital filter. After M cycles, the output of the digital filter is sampled and then reset as the loop filter. The final digital output data rate is at $f_{\mathrm{rst,\ NYQ}} \approx (f_s \cdot \mathrm{OSR})/M$, which corresponds to the effective conversion rate. The number of cycles in each conversion, M, is equivalent to the OSR in the ΣΔ-ADC. A sample and hold block (i.e., sampler), clocked at the rate of $f_{\mathrm{sh,\ NYQ}}$, is sometimes placed preceding the IΣΔ-ADC to make sure the input is constant during each conversion. However, it is not always used in practice [13], especially considering the power and noise penalty from such a block. Further, note that an Anti-Aliasing Filter is also often required at the input to avoid aliasing issues.

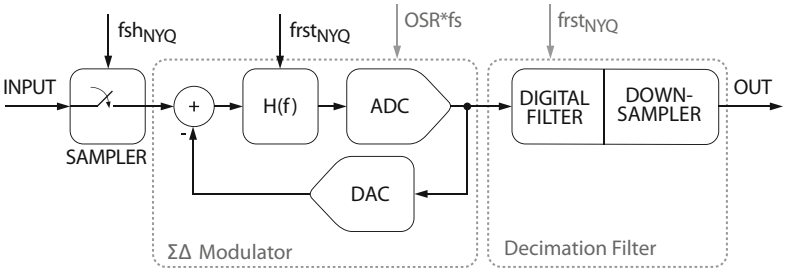


Fig. 10.2 General block diagram of an incremental ΣΔ-ADC

10.2.1 Digital Filter Design

One of the main differences between conventional and incremental ΣΔ-ADCs lies in the digital filter design [7]. In the first-order case, a *Sinc* or $Sinc^2$ filter is needed in conventional ΣΔ-converters, while a simple up-down counter is sufficient for IΣ-Δ-ADCs. In the case of high-order modulators, the digital filter in conventional ΣΔ-ADCs is commonly implemented as [3]:

- A $Sinc^L$ filter composed of a cascade of integrators and differentiators.
- A two-stage decimation filter composed of a cascade of a $Sinc^L$ stage and a *finite impulse response* (FIR) or an *infinite impulse response* (IIR) stage

On the other hand, the commonly adopted digital filter implementations in the IΣΔ-ADC are [10]:

- *Matched digital filter* realizes the digital filter as the replica of the analog filter that processes the feedback signal. This method takes advantage of one distinctive feature of the incremental mode – the conversion residue can be made available at the output of the last integrator by the end of each conversion. One major problem of this digital filter realization is that it requires exact matching with the analog filter.
- *Cascade of integrators* directly calculates the digital output from the modulated samples using a cascade of digital integrators without knowing the exact value of any analog coefficients. Assuming single-bit quantization, the first stage can be realized with a counter, while the subsequent stages are implemented by digital integrators.
- $Sinc^K$ *filter* provides suppression for 50/60 Hz power line frequency noise. As in the conventional ΣΔ-ADCs, it is normally desirable to let the *Sinc* filter be one order greater than the modulator in the incremental ΣΔ-ADCs (i.e., $K = L + 1$). However, there are two drawbacks that prevent its practical usage: (i) it needs more clock cycles per conversion to achieve the desired resolution; and (ii) it cannot provide a flat frequency response in the signal band.

10.2.2 Theoretical Resolution

Another major difference between the IΣΔ-ADCs and their conventional counterparts is the reduced theoretical resolution [7, 10]. The performance degradation of incremental ADC is the price for the ability to multiplex between multiple input channels. Due to the reset of the modulator and decimation filter, the length of the impulse response of the NTF for an incremental ADC is limited to M samples. To have a rough comparison, assuming the digital filter is formed by a cascade of

Table 10.1 Comparison of the SQNR improvements when doubling the OSR or M in conventional and incremental ΣΔ-modulators, respectively. A 1-bit quantizer is assumed in each case

Loop filter order	$L = 1$	$L = 2$	$L = 3$	$L = 4$
Conventional ΣΔ	9.03 dB	15.05 dB	21.07 dB	27.09 dB
Incremental ΣΔ	6.02 dB	9.54 dB	12.04 dB	13.98 dB

integrators, the SQNR of a *Lth*-order IΣΔ-ADC can be estimated through time domain analysis as:

$$\text{SQNR} \approx 6.02 + 6.02 \log_2\left(\frac{(M+L-1)!}{L!(M-1)}\right) + 1.76 \tag{10.3}$$

where M is the number of incremental cycles. It can be appreciated from Eq. 10.3 that for every doubling of M, the SQNR is incremented by approximately 6 dB for a first-order IΣΔ-ADCs, instead of the 9 dB of conventional modulators (i.e., see Chap. 1). As the loop filter order L increases, the performance degradation of the incremental mode compared to the conventional mode becomes more significant. Table 10.1 shows the SQNR improvement per doubling OSR or M of conventional modulators and IΣΔ-ADCs calculated for $L = 1, 2, 3, 4$, respectively.

Important to note is that, typically, first-order IΣΔ-ADC [8] can achieve very high resolution, but at the cost of extremely long conversion time, thus resulting in poor power efficiency. Therefore, high-order and multi-bit IΣΔ-ADCs have been developed to speed up the conversion rate [9]. Alternatively, ADC architectures which combine the IΣΔ-modulators with a Nyquist-rate ADC [10], such as extended counting (EC) [11] or extended range (ER) [12].

ADCs have been proposed to effectively enhance the resolution of IΣΔ-ADCs. On the other hand, the advantage of the increased quantization noise floor in the IΣΔ-ADC relaxes the requirements on jitter and ELD. Hence IΣΔ-ADC is less sensitive to these non-idealities than traditional modulators.

10.2.3 Incremental ΣΔ-Modulators Conclusions

To summarize, incremental ΣΔ-ADCs [7–10]:

- Focus on absolute precision instead of spectral performance.
- Reset before every A/D conversion.
- Interested in residual error – i.e., time domain analysis instead of frequency domain.

The reader is encouraged to experiment with the models "*INC_MOD*" provided in the Simulink® Toolbox.

In the Toolbox, use the models found in the folder *10_INCREMENTAL*.

Type – *load_par* – in the MATLAB Command Window to initialize variables needed for the model to function and run the simulation.

In particular, the reader may wish to analyze the modulator SNDR and Time Scopes between the modulator stages. Specifically, attention should be paid to the timing of operation. Additionally, the multi-bit implementation should also be investigated, noting how the conversion time results reduced.

10.3 Time Domain ADCs

All the ΣΔ-modulators investigated so far have used analog integrators and voltage comparators (i.e., quantizer), which are difficult to design in advanced CMOS technology nodes with reduced power supply voltages and small transistor intrinsic gains. However, thanks to the very fast switching speed and high transistor density of modern CMOS processes, it is possible to implement analog signal processing functions also by leveraging time as a signal domain [13]. Time domain ADCs differ from voltage domain ADCs in that they quantize the input signal using *time* as the reference quantity. In this context, voltage-controlled oscillators (VCO) are circuit elements which allow to integrate, or quantize, conventional signals into the time domain while offering several advantages, including [13–15]:

- Take advantage of Moore's law, since VCO-ADCs resolution versus sampling speed trade-off relationship directly improves with the reduced gate delays of modern CMOS processes.
- Reduced design complexity. In the case of ΣΔ-modulators ring, oscillators are commonly used, which can be implemented using simple inverter chains and digital gate circuits.
- Area and power efficient, due to the highly digital implementation.
- VCO-based quantizers are less sensitive to supply voltage reduction compared to conventional voltage quantizers. Further, they have the ability to potentially achieve first-order noise shaping of their quantization noise.
- Depending on the ADC-architecture, the VCO may allow achieving a first-order AAF.

Although virtually any VCO could be used in the design of ΣΔ-ADCs, sinusoidal ring oscillators are probably the most common due to their ability to oscillate even for very low input voltages, their low power dissipation at high frequency of oscillation, wide tuning range of given control voltage, and the possibility to easily achieve multiple output phases. They also are cost- and area-effective when compared, for example, to LC-type oscillators. Further, consisting of multiple delay stages in a closed loop, ring oscillators are simple to design since each delay stage can be realized as a simple inverting circuit [13, 16].

Besides the many advantages discussed, the most significant drawback of using VCOs is their severely nonlinear voltage-to-frequency conversion [16]. A VCO has typically only 6-bit of linearity which limits the overall SNDR to 40 dB. To limit this error, two simple strategies (among others) are to limit the voltage swing of the input signal to the VCO and to incorporate the VCO in a closed loop capable of providing enough gain to suppress the nonlinearity. Therefore, it should be apparent that ΣΔ-modulators are natural candidates for implementing such strategies.

10.3.1 Introduction to VCO Operation

Figure 10.3 illustrates the general block diagram and characteristics of a voltage-to-time converter implemented by using a VCO [13, 17].

As it can be seen, an ideal VCO output can be expressed as a sinusoidal signal with amplitude, phase, and frequency of:

$$\mathrm{VCO_{OUT}}(t) = A_{\mathrm{VCO}} \ \sin\theta_{\mathrm{VCO}}(t) \tag{10.4}$$

$$\theta_{\mathrm{VCO}}(t) = 2\pi \int_0^{\infty} f_{\mathrm{VCO}}(\tau)d\tau = 2\pi \int_0^{\infty} f_{\mathrm{c}} + K_{\mathrm{VCO}}x(\tau)d\tau \tag{10.5}$$

$$\omega_{\mathrm{VCO}}(t) = \omega_0 + K_{\mathrm{VCO}}V_{\mathrm{ctrl}} \tag{10.6}$$

where ω_0 is the output frequency corresponding to an input voltage $x(t) = V_{\mathrm{ctrl}} = 0$, while the free running center frequency, f_c, corresponds to the mid-range value between the maximum and minimum oscillating frequencies. Clearly, $\omega_{\mathrm{VCO}} = 2\pi f_{\mathrm{VCO}}$.

The oscillator tuning range is the difference between the maximum (e.g., ω_2) and minimum (e.g., ω_1) oscillating frequency; therefore if the allowed V_{ctrl} range is V_1 to V_2, then the tuning range achieved is the difference between ω_2 and ω_1.

The gain is defined as the ratio of tuning range to control voltage range:

$$K_{\mathrm{VCO}} \geq \frac{\omega_1 - \omega_2}{V_2 - V_1} \tag{10.7}$$

According to Eq. 10.5, it is seen that the phase of the VCO is the integral of the VCO frequency. Thus, the VCO behaves as a CT voltage-to-phase integrator [16]. As shown in Fig. 10.3, the instantaneous VCO output frequency $\omega_{\mathrm{VCO}}(t)$ is proportional to the applied input voltage $\mathrm{V_{ctrl}}(t)$ according to the voltage-to-frequency gain K_{VCO} [*Hz/V*]. The resulting VCO output phase $\theta_{\mathrm{VCO}}(t)$ is proportional to the time integral of the applied input voltage. Note that as long as the VCO oscillates, the VCO output phase will accumulate endlessly, even for a DC input. This implies that the VCO behaves as a CT integrator with infinite DC gain. The value of a simple integrator structure that provides infinite DC gain cannot be emphasized enough. Indeed, with transistor intrinsic gains dropping rapidly at

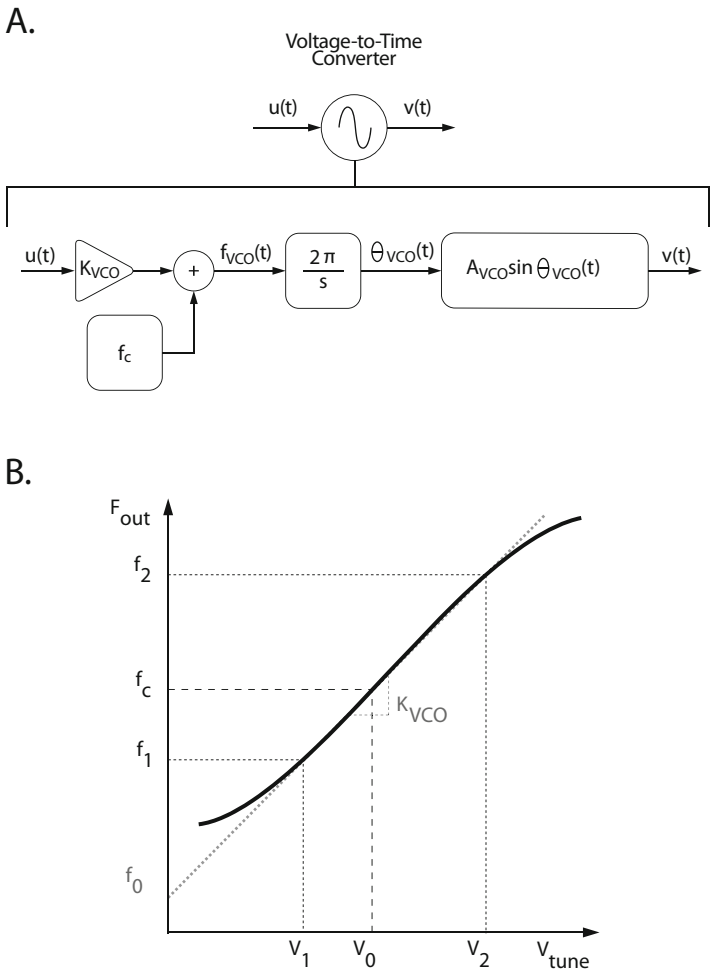


Fig. 10.3 (**a**) Ideal VCO block diagram and (**b**) transfer curve characteristic

each technology node, even the ability to achieve modest gains (e.g., $\geq$40 dB) with a conventional amplifier topology is highly challenging. Note, however, that the VCO integrator is not an ideal integrator even though it has infinite DC gain due to the VCO output frequency being a nonlinear function of the applied input voltage. Consequently, an input voltage signal that modulates the VCO control node will incur in a potentially high harmonic distortion, degrading the effective dynamic range of the VCO-based integrator.

Another property of interest is the possibility to implement VCOs also as quantizers [17], which can be distinguished in (Fig. 10.4):

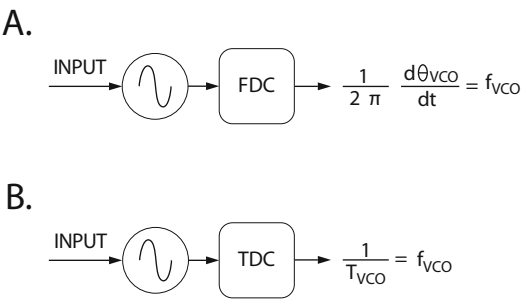


Fig. 10.4 VCO-based quantizer. (**a**) FDC: measures phase difference. (**b**) TDC: measures period and inverts

- *Frequency-to-digital converter (FDC) :* the phase difference between samples is measured to arrive at the VCO frequency.
- *Time-to-digital converter (TDC) :* the period of the VCO is measured and inverted to get the VCO frequency.

The nonlinear voltage-to-frequency conversion of the VCO is, again, a major issue, limiting the achievable quantizer SNDR to approximately 40 dB. One effective strategy to counteract this error has been mentioned to be the reduction of the VCO input swing. To achieve this, one approach is to perform VCO quantizer's residue cancellation as seen in Chap. 4, Sect. 4.5.4. Another approach is to use the VCO not only as a quantizer but also as an integrator by outputting phase instead of frequency. To further reduce nonlinearity, a dual-VCO-based integrator can be used to suppress VCO's dominant second-order distortion. Moreover, note that all these techniques can be incorporated into a VCO-ΣΔ-modulator in addition to the nonlinearity suppression provided by the ΣΔ-loop filter itself. Clearly, many different architectures to implement FDC and TDC exist [15–17]; however, due to the introductory nature of this chapter, these are not studied in detail, and only three of the most common VCO-quantizer arrangements will be briefly presented.

10.3.2 VCO Voltage-to-Frequency ΣΔ-Modulators

Figure 10.5 illustrates the FDC voltage-to-frequency VCO-quantizer architecture used in the ΣΔ-modulator discussed in this paragraph. As it can be seen, it comprises a ring VCO, a set of *M*-bit registers (i.e., typically D flip-flops), *M*-bit XOR gates, and an adder stage. The relative simplicity of the circuit is an important advantage for CT ΣΔ-modulators, allowing a high speed of operation with small latency.

The principle of operation of the VCO quantizer [16, 17] is to count the number of inverters switching state within the clock period by comparing samples of their current and previous states. Note that the inverter delay is set by the input tuning voltage. Therefore, the number of inverters switching state at each clock period represents the input tuning voltage, and the total number of inverters defines the quantizer resolution. A key constraint is that the maximum number of transitions in a

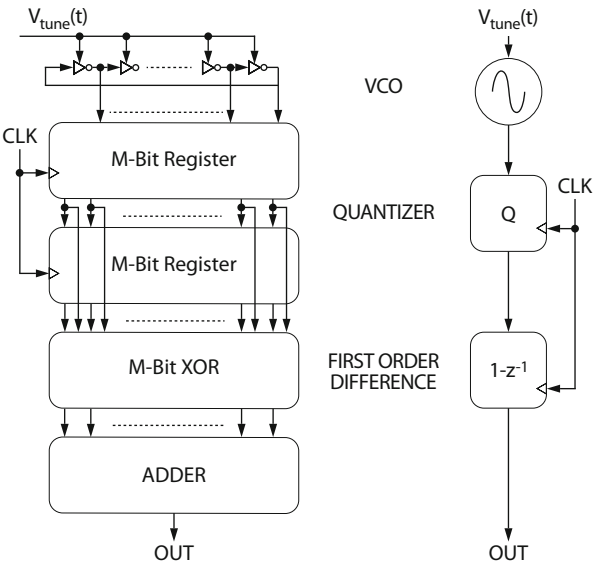


Fig. 10.5 VCO voltage-to-frequency quantizer

clock period cannot exceed the total number of inverters. This can also be expressed in terms of frequency as:

$$f_{\mathrm{VCO_{MAX}}} < \frac{f_s}{2} \tag{10.8}$$

where f_{VCO} is the VCO output frequency and f_s is the sampling frequency.

For a more accurate analysis, Fig. 10.6 illustrates the linear model of the VCO voltage-to-frequency quantizer [16, 17]. The ring VCO is modeled as a CT integrator from voltage input to output phase with gain $2\pi K_{\mathrm{VCO}}$. The sampling registers (i.e., quantizer) are modeled as a sampler on the integrator output, and the XOR operation is modeled as a first-order DT differentiator. Thus, the forward path in the quantizer is CT integrated and then DT differentiated. The quantizer essentially samples the oscillator phase, which corresponds to the integral of frequency. The first-order difference then converts the sampled phase back to a scaled version of frequency. Note that, in general, the XOR operation does not yield a first-order difference operation for arbitrary inputs, but does so in this case since the VCO phase monotonically increases as a function of time. An advantage of the differentiator is that when using the VCO quantizer in a $\Sigma\Delta$-modulator, the XOR outputs can be used to directly feed the DACs, eliminating the need for a DEM dedicated circuit typically required in multi-bit implementations, which saves area and power consumption. This is because, due to the digital differentiation at the VCO output, the elements in the feedback DACs are naturally selected in a barrel-shifted fashion, which implements a noise-shaping property identical to DWA DEM. As a result, the DAC mismatches are automatically shaped to the first order.

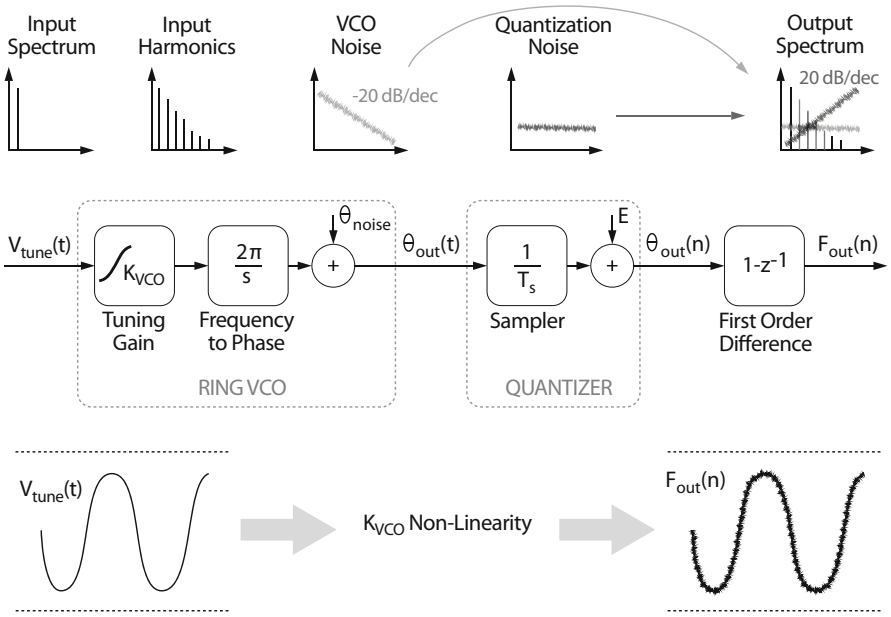


Fig. 10.6 Linear model of the VCO voltage-to-frequency quantizer

As shown in Fig. 10.6, there are three key non-idealities presented by the VCO-based quantizer [16, 17]: the K_{VCO} nonlinearity, VCO phase noise, and quantization noise. K_{VCO} nonlinearity corresponds to the nonlinear voltage-to-frequency characteristic of the oscillator, and VCO phase noise corresponds to the accumulated effect of thermal noise (and flicker noise, $1/f$) in the devices that comprise the delay stages. For the sake of simplicity, the VCO phase noise is shown to have a -20 dB/decade slope, but in reality, a steeper slope will occur at low-frequency offsets due to the impact of flicker noise, and a noise floor will occur at high-frequency offsets. Finally, quantization noise is a by-product of any quantizer implementation, and it is assumed and modeled as white noise. Important to note for VCO-based ΣΔ-modulators is that if the quantization noise is first order shaped to high frequencies, then the VCO phase noise becomes the dominant noise source of the VCO quantizer at low-frequency offsets.

The multi-bit case of the VCO voltage-to-frequency quantizer is also of relatively easy implementation. Multi-bit quantization (or equivalently, multi-phase) can be accomplished by sampling the output nodes of a multi-stage ring oscillator with an array of D flip-flops [17]. For example, to obtain a 5-bit output, it is sufficient to increase the number of delay stages of the ring oscillator to 31, followed by 31-bit registers, XOR logics, and summation block. This eliminates the need to generate an array of evenly spaced comparison voltages typically required in voltage quantizers. Note that since the VCO phases are full-swing logic signals, the quantizer is robust to voltage offsets in the digital registers. At the same time, only one VCO edge

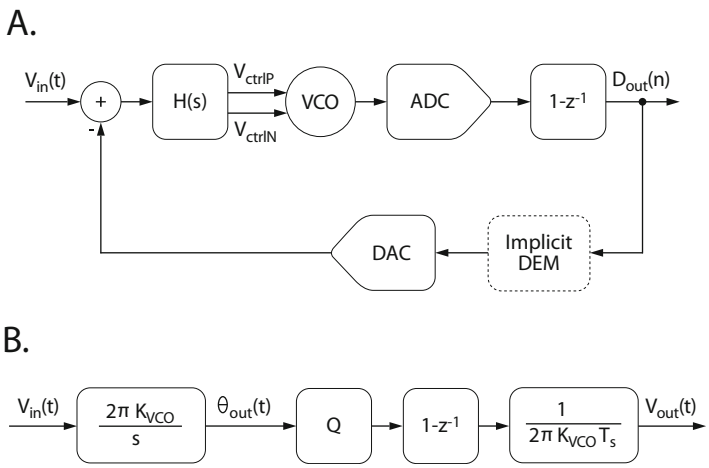


Fig. 10.7 ΣΔ-VCO-based ADC using frequency as key variable (**a**) block diagram and (**b**) linear model of (**a**)

transition at a given sampling instant occurs, while the rest of the VCO phases saturate to either *VDD* or *GND*. Consequently, the quantizer not only is less prone to generate meta-stable outputs but also has guaranteed monotonicity without requiring any calibration.

The VCO quantizer incorporated in a ΣΔ-modulator is illustrated in Fig.10.7.

Important to note when using the VCO quantizer with frequency as a key variable in a ΣΔ-modulator is that the output frequency representing the input voltage, f_{VCO}, has also to be converted back to voltage in order to feed the modulator feedback DACs. Dividing by $2\pi K_{VCO}T_s$ cancels the CT integrator gain and allows to insert the quantizer model in a CT ΣΔ-modulator without changing its NTF [14]. Note that, due to the frequency-dependent cancellation of the CT integration by the DT differentiation within the VCO, the VCO transfer function shows at $f_s/2$ a phase shift of 90°, corresponding to 50% delay in the modulator. This delay, if causing the modulator to become unstable, can be compensated with the techniques used for ELD compensation [17]. Further, since the VCO together with the differentiator acts only as quantizer, the overall transfer function gain from the VCO input to the ADC output is limited, resulting in a large VCO input which leads to a limited ADC linearity (i.e., K_{VCO}). Therefore, analog integrators are typically required in order to increase the loop order and limit the voltage-to-frequency nonlinearity. Since $H(s)$ is comprised of analog integrators, it is scaling unfriendly and power hungry [14, 15].

To conclude this introduction to VCO-based voltage-to-frequency ΣΔ-modulators, the reader is encouraged to experiment with the models "*FDC_MOD*" provided in the Simulink® Toolbox.

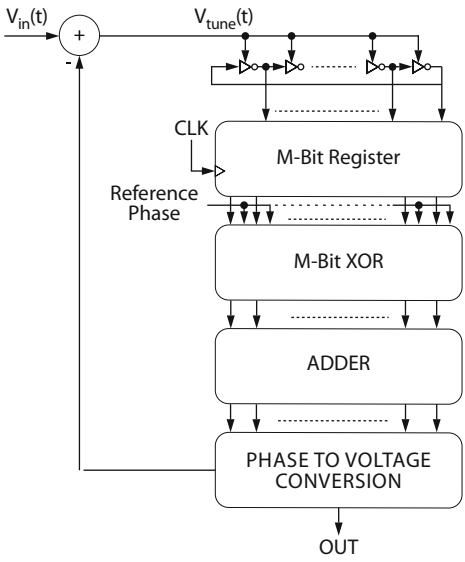


Fig. 10.8 VCO voltage-to-phase quantizer

In the Toolbox use the models found in the folder *10*_VCO -> VCO_*Frequency*.

Type – *load_par* – in the MATLAB Command Window to initialize variables needed for the model to function and run the simulation.

In particular, the reader may wish to analyze the modulator SNDR and Time Scopes between the VCO stages to investigate the accordance with the linear model described. In particular, attention should be paid to the spurious-free dynamic range (SFDR), since it is the VCO's harmonic distortion that limits the quantizer linearity. It should result that it is very difficult to achieve high SNDR even if the order of the modulator is increased, due to the VCO intrinsic nonlinearity. Additionally, the multi-bit implementation should also be investigated, noting the phase differences set in the VCO-models used and the increase in the modulator performance.

10.3.3 VCO Voltage-to-Phase ΣΔ-Modulators

The previous paragraph discussed that K_{VCO} nonlinearity poses a severe challenge toward achieving high converter SNDR, even for applications that embed the VCO quantizer within a high-gain loop filter of a ΣΔ-modulator structure. To overcome this issue, phase rather than frequency can be used as the key output variable [16, 17]. Figure 10.8 illustrates the respective FDC voltage-to-phase VCO-quantizer architecture. The quantizer comprises a ring VCO, one *M*-bit register

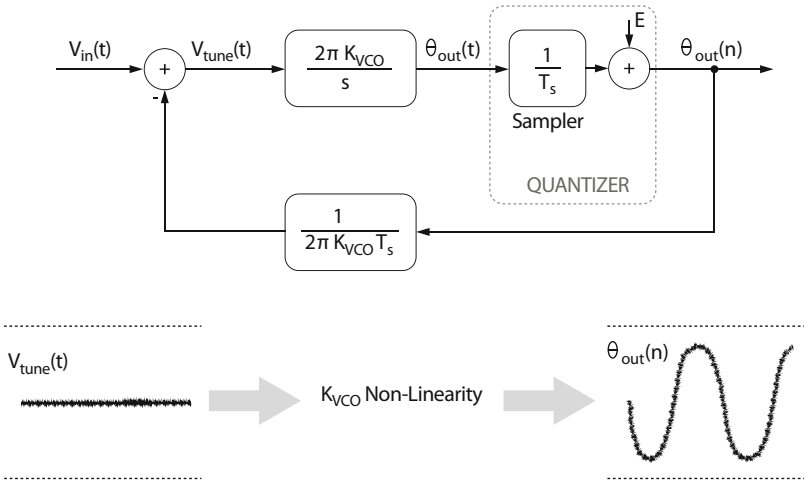


Fig. 10.9 Linear model and principle of operation of the VCO voltage-to-phase quantizer

(e.g., typically a D flip-flop), *M*-bit XOR gates, and an adder stage. As it can be seen, it is similar to the voltage-to-frequency architecture discussed in the previous paragraph, except that the output of the adder is phase instead of frequency and it is subtracted from the input after being converted into a voltage. Further, note the reference phases generated from the sampling clock to detect the phase change at the input of the XOR gates.

The main idea in the voltage-to-phase quantizer is that the control voltage of the ring VCO is only the error between the quantized output and the quantizer input (Fig. 10.9). Since the error has a much smaller dynamic range compared to the input signal, the K_{VCO} characteristics may be assumed linear, and the VCO keeps oscillating near its free running frequency. In other words, being phase the integral of frequency, the tuning voltage can be confined to a very small operating region while still enabling a complete exercise of the quantizer dynamic range. Important to note is that since there is no differentiation block in the loop, the VCO acts also as an integrator which has infinite DC gain. Therefore, the feedback loop of the quantizer effectively corresponds to a first-order CT ΣΔ-modulator, and hence the quantizer exhibits an inherent first-order noise-shaping property.

The multi-bit version of the voltage-to-phase quantizer is developed by implementing a multi-phase ring VCO [13, 17], where the inverter delay is set by the control voltage. Similarly to the single-bit case, the multi-phase output of the VCO is compared to a multi-phase reference and the phase error converted to a voltage and subtracted from the input.

Note that the reference multi-phase generated is dependent on the free running frequency and the number of the VCO stages. Therefore, for an *N*-bit ring VCO, there are $2 * (2^N - 1)$ possible output phase states while, for example, for a free running frequency of $f_s/4$, four reference phases would be required.

The VCO quantizer incorporated in a ΣΔ-modulator is illustrated in Fig. 10.10.

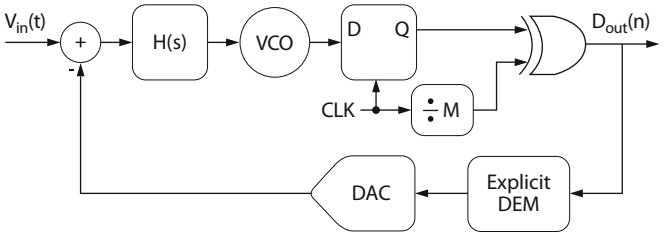


Fig. 10.10 ΣΔ-Modulator with VCO voltage-to-phase quantizer

As it can be seen, a feedback loop in which the quantizer phase is compared to a reference phase is implemented, and the resulting error is fed back through the modulator DAC to the input of the modulator loop. The drawback of using a voltage-to-phase VCO quantizer is that the DEM ability previously seen in the voltage-to-frequency quantizer is lost because the output is always thermometer coded. Therefore, any mismatch in the DAC would cause tones in the output, and, to avoid this, an explicit DEM has to be used. This leads to extra circuit complexity, area, power consumption, and also propagation delay. Note that this additional propagation delay may cause degradation in the quantization noise shaping and potentially can destabilize the loop. However, when high SNDR is desired, the improvement in the linearity achieved outweighs the loss of the DEM ability [16, 17].

To conclude this introduction to VCO-based voltage-to-phase ΣΔ-modulators, the reader is encouraged to experiment with the models "*FDC_MOD*" provided in the Simulink® Toolbox.

In the Toolbox use the models found in the folder *10*_VCO -> VCO_*Phase*.

Type – *load_par* – in the MATLAB Command Window to initialize variables needed for the model to function and run the simulation.

In particular, the reader may wish to analyze the modulator SNDR and Time Scopes between the VCO stages to investigate the accordance with the linear model described. Specifically, attention should be paid to the improved linearity when compared to the voltage-to-frequency quantizer. Additionally, the multi-bit implementation should also be investigated, noting the phase differences set in the VCO-models used and the increase in the modulator performance.

10.3.4 Dual-VCO ΣΔ-Modulators

In order to solve the issues of low VCO linearity and loss of intrinsic DEM capability of the VCO quantizers discussed in previous paragraphs, the dual-VCO quantizer proposed in [15] is introduced. Figure 10.11 illustrates the quantizer structure while being implemented in a ΣΔ-modulator.

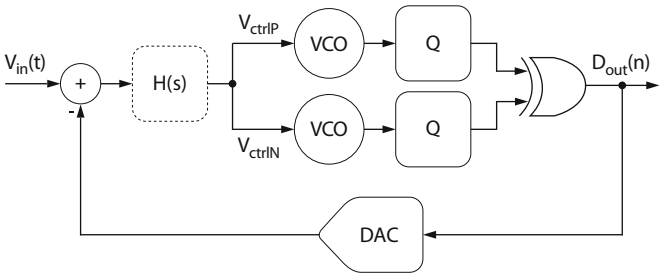


Fig. 10.11 ΣΔ-Modulator with dual-VCO quantizer

As it can be seen, a *differential* arrangement is used to form the dual-VCO quantizer while phase is being exploited as the key variable, so to retain the integral behavior of the VCOs. The phase output is measured by comparing the phase of one VCO with that of the other. The advantage is given by the fact that the feedback loop operates only on the difference between the two VCO phases and does not control the VCO center frequency. Therefore, since no correlation exists between the VCO center frequency and the sampling frequency, the center frequency can be chosen freely and does not have to be locked to a fraction of the sampling frequency as in the voltage-to-frequency and voltage-to-phase architectures previously seen. This allows lowering the VCO frequency to save power and reduces phase noise.

Another important advantage, derived from the fact that the sampling circuit samples irreverently with the VCO center frequency, is that the starting element in the VCO at sampling edge is randomly chosen. Therefore, even though phase is used as the key variable, a natural rotation of the DAC selection pattern is achieved, and DEM capability is not lost. The DEM scheme achieved is that of clocked averaging (CLA) [18] which modulates the DAC mismatch errors to the twice of the VCO center frequency and moves them out of the signal band.

Furthermore, because the dual VCO is used as an integrator, it provides a large loop gain to suppress the VCO nonlinearity. Its pseudo-differential structure also helps cancel out even-order distortions. Additionally, as seen in previous paragraphs, a large number of VCO stages can be used to increase the effective phase quantizer resolution and further suppress the signal swing at the VCO input. As a result, the analog filter $H(s)$ is no longer needed from a linearity point of view. Although not strictly relevant from a high-level point of view, note that the dual-VCO structure proposed in [15] also uses current instead of voltage as the input to the VCO in order to further reduce the VCO nonlinearity. Note that the strategy to use current can be applied also to the VCO quantizers investigated in previous paragraphs.

As for the other VCO quantizers presented, the reader is encouraged to experiment with the models of the dual-VCO ΣΔ-modulators provided in the Simulink® Toolbox.

In the Toolbox use the models found in the folder *10*_VCO -> Dual_VCO.

Type – *load_par* – in the MATLAB Command Window to initialize variables needed for the model to function and run the simulation.

In particular, the reader may wish to analyze the modulator SNDR and Time Scopes between the VCO stages and investigate the performance when compared to the models of previous paragraphs.

10.4 Conclusion and Essential Takeaways

This chapter introduced emerging ΣΔ-ADC architectures which exhibit attractive properties for implementations in modern nanometer CMOS technologies and applications where typical ΣΔ-modulators are historically not of common use. Important concepts covered include:

- *SMASH* allows to reduce the matching sensitivity of MASH architectures while saving area and power consumption by re-utilizing the analog loop filter to recombine the quantizer outputs, instead of using the digital filters of typical MASH structures. The main disadvantage, when compared to MASH architectures, is that stability is not always guaranteed.
- *Incremental* allows to implement ΣΔ-modulators in applications where the signal under conversion is typically weakly correlated and of a narrow frequency band, as well as in applications where multiplexing is required. Although the digital filter design for such architectures is typically facilitated, the main disadvantages are found in the longer conversion time required, the more complex implementation (especially in terms of timing), and the lower resolution achieved when compared to typical ΣΔ-modulators.
- *Time domain* allows to use time, instead of voltage, as the reference quantity for quantizing signals. This carries several advantages, including the availing of Moore's law, reduced design complexity of the quantizer, reduced area, power consumption, etc. Besides the many possible implementations of time domain ΣΔ-ADCs (e.g., PWM-controlled modulators, etc.), one of the most common has been discussed to be the one utilizing ring oscillators. The main disadvantage of using ring VCOs is due to their nonlinear voltage-to-frequency conversion. To overcome this issue, two main strategies have been presented, including (i) the use of phase instead of frequency as the reference parameter of the conversion and (ii) the implementation of a differentiated dual-VCO quantizer structure.

To conclude the book, it is my hope that the reader felt somehow inspired by the many possibilities of emerging ΣΔ-modulators. Further investigation of the concepts introduced is highly encouraged since the topics covered are of relevance in the current research community. A simple suggestion which can be used as a starting point is to try mixing the techniques discussed, for example, by implementing a

SMASH modulator using VCO quantizers or developing an incremental architecture formed by VCO-based modulators. Obviously many possibilities exist! As a word of courage, it is always worth to remember that actual simulation still plays a huge role in ΣΔ-technologies; therefore experimenting with tools such as either the Schreier's Toolbox or the Simulink® Sigma-Delta Toolbox which accompanies this book is probably a good method which may help to develop the next generation of ΣΔ-modulators in a fun and relatively easy way when compared to classical mathematical methods.

References

1. De la Rosa JM, Del Rio RF. CMOS sigma-delta converters: practical design guide. London: Wiley; 2013.
2. De la Rosa JM, et al. Next-generation delta-sigma converters: trends and perspectives. IEEE J Emerging Sel Top Circuits Syst. 2015;5(4):484–99.
3. Schreier R, Temes GC. Understanding delta-sigma data converters. Hoboken: Wiley; 2005.
4. Maghari N, Kwon S, Temes GC, Moon U. Sturdy mash Δ-Σ modulator. Electron Lett. 2006;42:1269–70.
5. Maghari N, Moon UK. Multi-loop efficient sturdy MASH delta-sigma modulators. IEEE international symposium on circuits and systems (ISCAS), 2008.
6. Maghari N, Kwon S, Temes GC, Moon U. Mixed-order sturdy Δ-Σ modulator. Proceddings of IEEE international symposium on circuit and system (ISCAS), p. 257–260, 2007.
7. Janos M, et al. Theory and applications of incremental Delta-sigma converters. IEEE Trans Circuit Syst I Regul Pap. 2004;51(4):678–90.
8. Temes GC, et al. Incremental data converters. Proceedings of the 19th international symposium on mathematical theory of networks and systems (MTNS), Jul. 2010.
9. Janos M, et al. Incremental delta-sigma structures for DC measurement: an overview. IEEE custom integrated circuit conference (CICC), 2006.
10. Carbone P, et al. Design, modelling and testing of data converters. Berlin: Springer Berlin Heidelberg; 2014.
11. Chen CH, et al. An incremental analog-to-digital converter with multi-step extended counting for sensor interfaces. IEEE international symposium on circuits and systems (ISCAS), 2016.
12. Garcia J, Rusu A. An extended-range incremental CT ΣΔ ADC with optimized digital filter. IEEE 13th international symposium on quality electronic design (ISQED), 2012.
13. Casier H, Steyaert M, Van Roermund AHM. Analog circuit design: robust design, sigma delta converters, RFID. New York: Springer; 2011. p. 177–201.
14. Straayer MZ, Perrott MH. A 12-bit, 10-MHz bandwidth, continuous-time ΣΔ ADC with a 5-bit, 950-MS/s VCO-based quantizer. IEEE J Solid State Circuits. 2008;43(4):805–14.
15. Lee K, Yoon Y, Sun N. A 10MHz-BW, 5.6mW, 70dB SNDR ΔΣ ADC using VCO-based integrators with intrinsic DEM. IEEE international symposium on circuit and systems (ISCAS), 2013.
16. Allam M, Aboushady H, Louerat M. Continuous-time ΣΔ modulators with VCO-based voltage-to-phase and voltage-to-frequency quantizers. IEEE 53rd international midwest symposium on circuit and systems (MWSCAS), Aug. 2010.
17. Yoder S, Ismail M, Khalil W. VCO-based quantizers using frequency-to-digital and time-to-digital converters. London: Springer; 2011.
18. Geerts Y, Steyaert M, Sansen W. Design of multibit delta sigma A/D converters. New York: Kluwer Academic Publisher; 2002.

Appendix A
Sigma-Delta Modulators Design Tools

Table A1 Theoretical SQNR (dB) for different modulators order, OSR, and quantizer bit

	OSR	32	64	128	256	32	64	128	256
	Bit	1				2			
Order	1	47.46	56.79	65.82	74.85	53.78	62.81	71.84	80.87
	2	70.14	85.19	100.24	115.29	76.16	91.21	106.26	121.31
	3	91.76	112.83	133.9	154.97	97.78	118.85	139.92	160.99
	4	113.01	140.10	167.19	194.29	119.03	146.12	173.21	200.31
	5	134.04	167.15	200.27	–	140.06	173.17	206.29	–
	6	154.93	194.06	–	–	160.95	200.08	–	–
	7	175.71	–	–	–	181.73	–	–	–
	8	196.41	–	–	–	202.43	–	–	–
	Bit	3				4			
Order	1	59.80	68.83	77.86	86.89	65.82	74.85	83.88	92.91
	2	82.18	97.23	112.28	127.33	88.20	103.25	118.30	133.35
	3	103.8	124.87	145.94	167.01	109.82	130.89	151.96	173.03
	4	125.05	152.14	179.23	206.33	131.07	158.16	185.25	–
	5	146.08	179.19	–	–	152.10	185.21	–	–
	6	166.97	206.10	–	–	172.99	–	–	–
	7	187.75	–	–	–	193.77	–	–	–
	8	–	–	–	–	–	–	–	–

$$\mathrm{SQNR} = 6.02N + 1.76 + (20L + 10)\log(\mathrm{OSR}) - 10\log\left(\frac{\pi^{2L}}{2L+1}\right)$$

© Springer International Publishing AG, part of Springer Nature 2019

I. Arnaldi, *Design of Sigma-Delta Converters in MATLAB®/Simulink®*,
https://doi.org/10.1007/978-3-319-91539-5

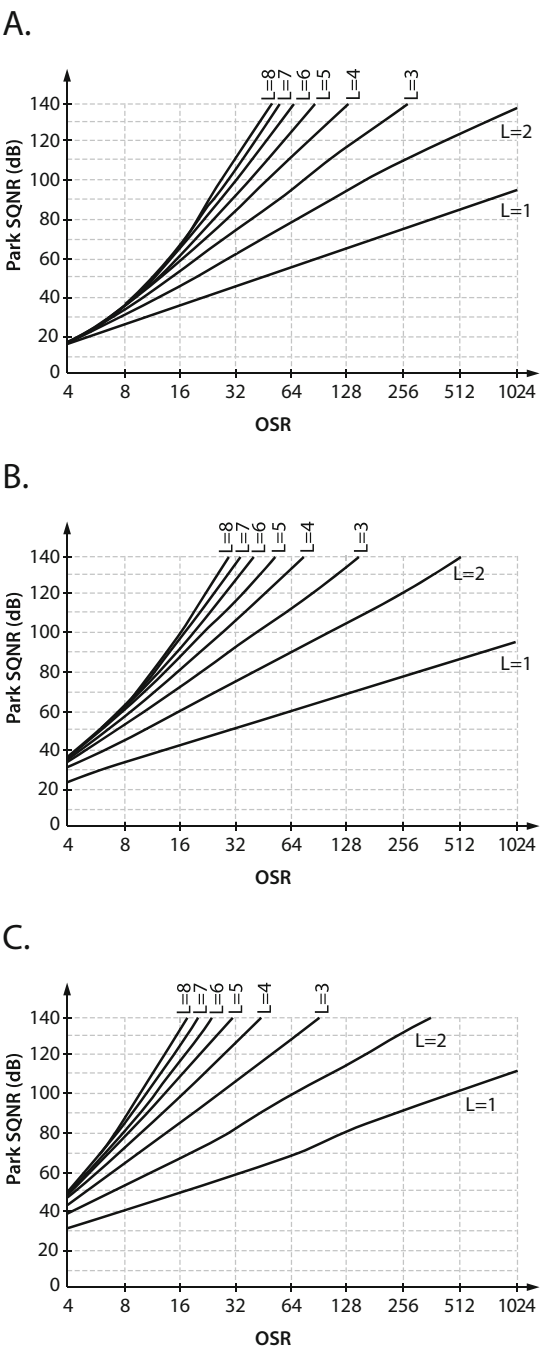


Fig. A1 Practical SQNR Limit for modulators with (**a**) 1-, (**b**) 2-, and (**c**) 3-bit quantizers, including stability considerations [1]

Table A2 Theoretical DR (dB) for various modulators order and OSR (i.e., 1-bit quantizer assumed)

	OSR	32	64	128	256
Order	1	40.65	49.68	58.71	67.74
	2	63.50	78.55	93.60	108.65
	3	85.30	106.38	127.45	148.52
	4	125.05	152.14	160.84	187.93
	5	127.75	160.86	193.98	–
	6	148.68	187.81	–	–
	7	169.49	–	–	–
	8	190.22	–	–	–

$$\mathrm{DR} = 10\log\left(\frac{3(2L+1)}{2\pi^{2L}}\right) + (20L+10)\log(\mathrm{OSR})$$

Reference

1. Schreier R, Temes GC. Understanding delta-sigma data converters. New Jersey: Wiley Inc.; 2005.

Appendix B
The Sigma-Delta Simulink® Toolbox

The *Sigma-Delta Simulink® Toolbox* is a time-domain behavioral simulator for Sigma-Delta (ΣΔ) modulators developed in the MATLAB®/Simulink® environment. The toolbox can be used to simulate any arbitrary ΣΔ-modulators architecture, implemented with both discrete time and continuous time circuit techniques since a complete list of the most important building blocks (i.e., integrators, DACs, etc.) is provided. The behavioral models of these building blocks take into account the most critical non-idealities of different circuit techniques affecting the modulator performance. This appendix provides an overview of the most significant features of the simulator.

B.1 Installing the Sigma-Delta Simulink® Toolbox

The *Sigma-Delta Simulink® Toolbox* can be freely downloaded from the *MATLAB® File Exchange* website:

https://www.mathworks.com/matlabcentral/fileexchange/64429-sigma-delta-simulink-toolbox

In order to design complete modulators (e.g., coefficients of high-order architectures, etc.), the Sigma-Delta Simulink® Toolbox makes use of some routines of the famous MATLAB® Schreier's Delta-Sigma Toolbox [1], which is also freely available from the *MATLAB® File Exchange* website:

http://www.mathworks.com/matlabcentral/fileexchange/19-delta-sigma-toolbox

Once the archives containing the toolboxes are downloaded, the following steps must be followed to install them:

- Uncompress the archive files to a directory on your own computer hard disk. Let us assume that the directory is named *Sigma_Delta_Toolbox*; then such folder will contain both the *Sigma-Delta* Simulink® *Toolbox* and the *MATLAB® Schreier's Delta-Sigma Toolbox.*
- Start MATLAB®.

© Springer International Publishing AG, part of Springer Nature 2019

I. Arnaldi, *Design of Sigma-Delta Converters in MATLAB®/Simulink®*,
https://doi.org/10.1007/978-3-319-91539-5

- Set the MATLAB® search path in order to add the *Sigma_Delta_Toolbox* directory. To do this, go to the *File* menu and select *Set Path*. The Set Path dialogue box opens, listing all folders on the search path. From this dialogue box, click the button *Add with Subfolders* and select the *Sigma_Delta_Toolbox* directory to add to the search path. In order to reuse the newly modified search path including *Sigma_Delta_Toolbox* directory and subdirectories, click *Save*, and finally click *Close*. This procedure must be done only the first time *Sigma_Delta_Toolbox* is installed in the hard disk.

The *Sigma-Delta Simulink® then such folder will contain bothToolbox* requires a MATLAB® version which includes:

- *Communications System Toolbox*
- *DSP System Toolbox*
- *Signal Processing Toolbox*
- *Optimization Toolbox*
- *Control System Toolbox*
- *Filter Design HDL Coder*
- *Simulink Control Design*
- *Fixed-Point Designer*

B.2 MATLAB Routines Summary

`load_par`	Initialize parameters used by the ΣΔ-modulator Simulink® models, including: • Simulation and PSD parameters • Modulator variables • Modulator model settings • DAC/DEM settings
`MakeModulator`	Example code to quickly design DT modulators by utilizing the functions of the *Schreier's Delta-Sigma Toolbox*. It works only for low-pass modulators and Butterworth NTFs approximations. Other types of Sigma-Delta modulators would require modifications of the code To make your own design, simply change the parameters in – *STEP* [1] – only, unless you can confidently code in MATLAB and know the Schreier's toolbox functions available

(continued)

`MakeModulator_CT`	Example code to quickly design CT modulators by utilizing the functions of the *Schreier's Delta-Sigma Toolbox*. It works only for low-pass modulators and Butterworth NTFs approximations. Other types of Sigma-Delta modulators would require modifications of the code. To make your own design simply change the parameters in – *STEP* [1] – only, unless you can confidently code in MATLAB and know the Schreier's toolbox functions available
`baseband_power`	Is the function used to compute the total baseband power. The routine works for low-pass modulators only
`baseband_sqnr`	Is the function used to compute the output signal power and baseband SQN(+D)R. If the model being simulated is ideal, the SQNR will actually be the SQNR. Otherwise, if non-idealities introducing distortion are present, the SQNR variable will actually be the SNDR. The routine works for low-pass modulators only
`sweep_dc_bbpwr`	DC sweep of the targeted ΣΔ-modulator Simulink® model and report/plot baseband powers. The routine uses the function `baseband_power` The script requires a workspace variable called *"target_mod"* and points to a Simulink® testbench of a modulator with a DC input block. Default DC step size for the sweep 1e-. This can be changed in the initialization script (e.g., *load_par -> simu.dcstep*), if required. Assuming no offset in the model, the script assumes the DC STF is symmetric around 0, so just sweep for the input range [0,1] and ignore [−1,0]
`sweep_dc_dc`	DC sweep of the targeted ΣΔ-modulator Simulink® model and report/plot the DC powers The script requires a workspace variable called *"target_mod"* and points to a Simulink® testbench of a modulator with a DC input block. Default DC step size for the sweep 1e-3. This can be changed in the initialization script (e.g., *load_par -> simu.dcstep*), if required. Assuming no offset in the model, the script assumes the DC STF is symmetric around 0, so just sweep for the input range [0,1] and ignore [−1,0]
`sweep_sinamp`	Sine sweep of the targeted ΣΔ-modulator Simulink® model and report/plot the SQN(+D)R vs. input signal amplitude. The routine uses the function `baseband_sqnr`

(continued)

	The script requires a workspace variable called *"target_mod"* and points to a Simulink® testbench of a modulator with a DC input block
`mod_SNDR`	Plot the PSD and compute the SQN(+D)R of the output of the ΣΔ-modulator Simulink® models of the toolbox. The routine uses the Welch method to compute the PSD. To change the frame size and window type, use the structure *"psdset,"* as found in *load_par*. Note that *"winbw"* control field (i.e., in *load_par*) determines how many bins each side of the signal frequency bin are assumed to contain signal power. So, set to span the main lobe width of whatever window you chose at least. The routine works for low-pass modulators only
`mod_SNDR_ac`	Same as `mod_SNDR` but it removes any DC component that may be present
`DEM_SNDR_ac`	Same as `mod_SNDR` but adapted to automatically calculate all the outputs of the "*testbench_DEM"* Simulink® model in the folder *5_MB_DAC*
`calc_quant_gain`	Estimate the quantizer gain based on in/out data. This script assumes that workspace variables called "*Qin"* and *"Qout*" exist in the Simulink® model and that these contain samples of a quantizer's input and output
`integ_stats`	Function plotting integrator histogram and its statistics based on the integrator outputs of the ΣΔ-modulator Simulink® model being tested
`THD`	Estimate the THD of the ΣΔ-modulator Simulink® model being tested

Simulink® Models Summary

First-order modulators – folder: *2_MOD1*	
`mod1`	DT first-order modulator model
`sweep_mod1`	DT first-order modulator model to be used in the – *sweep_testbench* – model in the main toolbox folder to perform the sweep routines
Second-order modulators – folder: *3_MOD2*	
`mod2`	DT second-order modulator model

(continued)

`sweep_mod2`	DT second-order modulator model to be used in the – *sweep_testbench* – model in the main toolbox folder to perform the sweep routines
`mod2_BoserWooley`	DT second-order modulator model. Boser-Wooley architecture. To be used in the – *sweep_testbench* – model
`mod2_SilvaSteensgaard`	DT second-order modulator model. Silva-Steensgaard architecture. To be used in the – *sweep_testbench* – model
Third-order modulator – folder: *4_MOD3*	
`mod3`	DT third-order modulator model
`mod3_ciff_1/2/3bit`	DT CIFF third-order modulator model. 1-/2- and 3-bit versions
`sweep_mod3_ciff_1/2/3bit`	DT third-order CIFF modulator mode. To be used in the – *sweep_testbench* – model. 1-/2- and 3-bit versions
Folder: Schreier's models	
`mod3_CIFB_CRFB_1/2/3 bit`	DT third-order modulator model of either a CIFB or CRFB architecture, to be used after the – *MakeModulator* – routine has been compiled. The model utilizes the *a*, *b*, *g*, *c* coefficients as computed by the Schreier's toolbox. The model has to be used in the – *sweep_testbench* – model. 1-/2-/3-bit versions available
`mod3_CIFF_CRFF_1/2/3 bit`	DT third-order modulator model of either a CIFF or CRFF architecture to be used after the – *MakeModulator* – routine has been compiled. The model utilizes the *a*, *b*, *g*, *c* coefficients as computed by the Schreier's toolbox. The model has to be used in the – *sweep_testbench* – model. 1-/2-/3-bit versions available

(continued)

Multi-bit DAC/DEM models – folder: *5_MB_DAC*	
`testbench_DEM`	Testbench to simulate individually the characteristics of random element selection (RES), individual level averaging (ILA), and data weighting averaging (DWA) dynamic elements matching (DEM) techniques
`mod2_2bit_NO_DEM`	Example model to simulate the effects of DAC mismatches in the case of no DEM
`mod2_2bit_RES`	Example model to simulate the effect of DAC mismatches in the case of RES algorithm applied
`mod2_2bit_ILA`	Example model to simulate the effect of DAC mismatches in the case of ILA algorithm applied
`mod2_2bit_DWA`	Example model to simulate the effect of DAC mismatches in the case of DWA algorithm applied
MASH modulator – folder: *6_MASH*	
`mod3_1plus1plus1MASH`	DT third-order MASH modulator model formed by three first-order modulators
`sweep_mod3_1plus1plus1MASH`	DT third-order MASH modulator model formed by three first-order modulators. To be used in the – *sweep_testbench* – model
`mod3_2plus1MASH`	DT third-order MASH modulator model formed by a second-order modulator and a first-order modulator stage. To be used in the – *sweep_testbench* – model
`mod3_2plus1MASH_1bit`	Same as `mod3_2plus1MASH` but with a 1-bit quantizer implemented in all stages in order to simulate the effects of not accurate cancellation of the analog and digital NTF due to the poorly defined quantizer gain

(continued)

`mod3_2plus1MASH_v2`	Same as `Mod3_2plus1MASH` but the input signal to the second stage is not directly taken from the Silva-Steensgaard architecture of the first stage but computed at the output of the first-stage quantizer
`Qout_111MASH_SNDR`,	These routines calculate the SNQ(+D)R of
`Qout_21MASH_SNDR`	each single stage forming the respective *1plus1plus* and *2plus1* MASH models. Same as `mod_SNDR`
CT modulator – folder: *7_CT_MOD*	
`CT_mod1`	Example of a CT first-order modulator Simulink® model. The model uses CT integrators
`NRZ_DAC`	Simulink® model of a DAC providing an non-return-to-zero pulse. The model allows to simulate the effect of jitter. To be used in the feedback path of a Sigma-Delta modulator
`RZ_DAC`	Simulink® model of a DAC providing a return-to-zero pulse. The model allows to simulate the effect of jitter. To be used in the feedback path of a Sigma-Delta modulator
`SCR_DAC`	Simulink® model of a DAC providing an switched-capacitor-resistor pulse. The model allows to simulate the effect of jitter. To be used in the feedback path of a Sigma-Delta modulator
DT design – folder: *8_DT_Design_Example*	
`CRFF_DTExample`, `MakeModulator_DTExample`	These routines are the equivalent to `load_par` and `MakeModulator`, respectively, but with adapted specification to demonstrate a specific design of a DT-CRFF modulator
`mod3_CRFF_DTExample`	DT third-order CRFF modulator model of the design example. To be used in the – *sweep_testbench* – model

(continued)

`DAC_mod3_CRFF_DTExample`	DT third-order CRFF modulator model of the design example which include the possibility to simulate DAC mismatches and implement DEM algorithms. To be used in the – *sweep_testbench* – model
CT design – folder: *9_CT_Design_Example*	
`CRFF_CTExample`, `MakeModulator_CTExample`	These routines are the equivalent to `load_par` and `MakeModulator`, respectively, but with adapted specification to demonstrate a specific design of a CT-CRFF modulator
`mod3_CRFF_CTExample`	CT third-order CRFF modulator model of the design example. To be used in the – *sweep_testbench* – model
`DAC_mod3_CRFF_CTExample`	CT third-order CRFF modulator model of the design example which includes the possibility to simulate DAC mismatches and implement DEM algorithms. To be used in the – *sweep_testbench* – model
`Complete_DAC_mod3_CRFF_CTEx`	Same as `DAC_mod3_CRFF_CTExamp le` but with specific CT non-idealities included (e.g., jitter, ELD, ISI, etc.). To be used in the – *sweep_testbench* – model
SMASH modulator – folder: *10_SMASH*	
`mod3_2plus1_SMASH`	DT third-order SMASH modulator model formed by a second-order modulator stage and a first-order modulator stage. To be used in the – *sweep_testbench* – model
Incremental modulator – folder: *10_INCREMENTAL*	
`mod1_inc`	DT first-order incremental modulator model
`mod2_SilvaSteensgaard_inc`	DT second-order incremental modulator model

(continued)

VCO modulator – folder: *10_VCO*	
`Diff_MOD`	DT first-order dual VCO modulator model
`VtF_VCO_Quantizer`	Model to simulate the voltage-to-frequency VCO-based quantizer
`VtF_mod1_1/2/3 bit`	Voltage-to-frequency first-order modulator model based on the 1, 2 and 3-bit VCO-quantizer
`VtF_mod2_SilvaSteensgaard`	Voltage-to-frequency second-order modulator model based on the 1-bit VCO-quantizer
`VtP_VCO_Quantizer`	Model to simulate the voltage-to-phase VCO-based quantizer
`VtP_mod_1/2/3 bit`	Voltage-to-phase first-order modulator model based on the 1-, 2-, and 3-bit VCO-quantizer
`VtP_mod2_SilvaSteensgaard`	Voltage-to-phase second-order modulator model based on the 1-bit VCO-quantizer

Reference

1. Schreier R, Temes GC. Understanding delta-sigma data converters. New Jersey: WileyInc.; 2005.

Index

A
Additive white noise source, 2
Algorithm
  DAC (*see* Digital-to-analog converter (DAC))
  DWA, 125
  ILA, 126
Analog terms, 139, 145
Analog transfer functions, 139, 145
Analog-to-digital conversion (ADC) technique, 225, 227, 231–242
  architectures, 242
  concepts, 242
  incremental (*see* Incremental ΣΔ-modulators)
  SMASH (*see* Sturdy MASH (SMASH) modulators)
  time domain (*see* Time domain ADCs)
Analog-to-digital converter (ADC), 71
  Nyquist-rate, 8
  quantization error, 2
Anti-aliasing filter (AAF), 2, 6, 7, 153, 156
Architectures
  CIFB, 96, 97
  CIFF, 98, 99, 101
  CRFB/CRFF, 104
  CT sigma-delta modulator, 204, 205
  DT ΣΔ-modulator, 185, 186
  MOD2
    Boser-Wooley modulator, 71
    selection, 74
    signal and noise transfer, 70
    Silva-Steensgaard modulator, 73, 74

B
Boser-Wooley modulator, 71

C
Cancellation, 139, 140, 145, 146, 148–150
Cascade, 137, 140, 141, 150, 151
Cascaded Integrators Feedback (CIFB), 79
Cascaded Integrators Feedforward (CIFF), 79
Cascaded Resonators Feedback (CRFB), 79
Cascaded Resonators Feedforward (CRFF), 79
Chain of Integrators with Feedback paths (CIFB), 96, 97
Chain of Integrators with Feedforward paths (CIFF), 98
CIFF ΣΔ-modulator, 89–91, 94
Clock Jitter, 168–170
Clocked averaging (CLA), 241
Coefficients variation, 162
Continuous time (CT)
  AAF model, 157
  DT ΣΔ-modulators, 175
  loop filter, 153, 154
  modulation process, 155–157
  non-idealities, 157, 158, 175
  NTF design, 171, 172
  Simulink® Simulation, 173, 174
Continuous time (CT) sigma-delta modulator
  high-level design (*see* High-level design)
  performance, 222
  specifications, 201, 221
  SQNR, 221
  theoretical design
    architecture selection, 204, 205
    feedback DAC, 203, 204
    order, 202
    OSR, 202, 203
    parameters, 206
    single-loop architecture, 202

© Springer International Publishing AG, part of Springer Nature 2019

I. Arnaldi, *Design of Sigma-Delta Converters in MATLAB®/Simulink®*,
https://doi.org/10.1007/978-3-319-91539-5

D
DAC pulse, 156, 163, 166, 167
Data weighting averaging (DWA)
  average error, 130
  elements selection, 126
  FFT, 128, 129
  first-order noise shaping, 128
  MATLAB® functions, 127
  tone generation, 125
  unit elements, 125
  ΣΔ-modulator, 132, 133
DC (inputs / behavior), 191
  CT sigma-delta modulator, 213
DC input signal
  MOD1
    conversion accuracy, 24, 26, 27
    idle tone generation, 27, 28
    Matlab, 24
    time domain and FFT, 25
    tonal behavior, 29–31
    tone prevention, 31, 32
    z-domain, 25
  MOD2
    idle tone, 54
    time domain and FFT, 55
    tonal behavior, 55, 57
Dead zones, 36, 37, 42, 46, 60
Decimation filter, 7, 8, 13
Delaying feedback loop, 23
Design, 79, 80, 106, 181
  CT (*see* Continuous time (CT) sigma-delta modulator)
  DT ΣΔ-modulator (*see* DT ΣΔ-modulator)
Digital filter, 138, 145
Digital filter design, incremental ΣΔ-ADC, 229
Digital signal processors (DSPs), 1
Digital terms, 139, 145, 148
Digital-to-analog converter (DAC)
  mismatches, 118, 120
  MOD2 FFT, 119
  nonlinearity, 116, 120
  Simulink® model, 117
  thermometer-code operation, 117
Discrete time (DT)
  loop filter, 153, 154
  modulation process, 155–157
  non-idealities, 157, 158
Discrete time (DT) sigma-delta modulator, 187
  architectures, 185, 186
  design parameters, 186
  high-level design (*see* High-level design)
  MASH, 184
  order and OSR, 182
  parameters' design choices, 186
  performance, 199
  quantizer bit, 183
  Simulink® Toolbox functions, 181
  specifications, 181
Discrete-time (DT) modulators, 2
Dither
  DC error sweep, 32
  dead zones, 37
  Matlab, 37
  source, 23
  Toolbox, 32
Downsampling, 7
Dual-VCO ΣΔ-modulators, 241
Dynamic element matching (DEM), 124
  average error, 130
  tones, 131
  ΣΔ-modulator, 132, 133
Dynamic Range (DR), 10

E
Effective Number of Bit (ENOB), 11
Elements selection, 123, 127
Example, 201
  CT sigma-delta modulator (*see* Continuous-time (CT) sigma-delta modulator)
Excess loop delay (ELD), 164, 165, 218

F
Fast Fourier transform (FFT), 9, 21
Feedback DAC
  Clock Jitter, 168–170
  CT modulators, 163
  CT sigma-delta modulator, 216
  DT ΣΔ-modulator, 194, 195
  ELD, 164, 165
  ISI, 166, 168
  linearity, 113, 114
  mismatch effects, 119
  multi-bit quantization, 114, 117
  RES, 122
  SNDR, 119
Feedback pulse, 170, 178
Filter
  anti-aliasing, 228
  digital, 227–229, 242
  loop, 225, 228, 234, 242
Finite integrator gain
  DC sweep, 36
  dead zones, 36, 38
  dither, 37

FFT, 35
integrator transfer function, 35
MOD2, 59, 60
time domain waveforms, 34, 35
First-order Sigma-Delta modulator (MOD1)
DC input
conversion accuracy, 24, 26, 27
idle tone generation, 27, 28
Matlab, 24
time domain and FFT, 25
tonal behavior, 29–31
tone prevention, 31, 32
z-domain, 25
finite integrator gain, 34–38
inputs and output blocks, 23
mid-rise quantizer, 45
noise-shaping, 45
quantization noise, 45
sampling frequency, 23
simulations, 34
SIMULINK® model, 22
sinewave inputs
amplitude sweep, 40, 42, 43
FFT, 38
low-pass filter, 39
time domain and FFT, 40
stability, 33
z-domain, 43
Fractional-N phase-locked-loop (PLL), 11
Frequency-to-digital converter (FDC), 234, 238

G
Gain
control, 188
DT modulators, 185
integrator finite gain and saturation, 194
large noise, 187
NTF, 185
quantizer, 183, 184

H
Half-delayed return-to-zero (HRZ), 163
High-level design
CT sigma-delta modulator
CMOS process, 208
coefficients mismatch, 217
DAC timing, 208
DC input values, 213
ELD, 208, 218
feedback DAC and element mismatch, 216
integrator finite gain and saturation, 216
ISI, 217, 218
jitter behavior, 219, 220
loop filter bandwidth, 207
MOD3-CRFF architecture, 206
NTF synthesis, 209–212
out-of-band gain, 206
SQNR and SJNR, 207
stability and tonal behavior, 214, 216
THD, 221
DT ΣΔ-modulator
coefficients mismatch, 195–197
DC behavior, 191
feedback DAC and element mismatch, 194, 195
integrator finite gain and saturation, 194
NTF synthesis, 187–190
stability and tonal behavior, 192, 194
THD, 198
theoretical MOD3-CRFF architecture, 187
High-order modulators, 95, 96
High-order ΣΔ modulator, 141, 150
HSPICE™, 12

I
Idle tone generation, 27, 28, 54
In-band noise, 100
Incremental ΣΔ-modulators
ADC, 227, 228
anti-aliasing filter, 228
applications, 227
conversion, 228
data, 228
digital filter design, 229
high-order and multi-bit, 230
implementation, 228
Simulink® Toolbox, 230, 231
SQNR, 230
time domain analysis, 230
Individual level averaging (ILA)
average error, 130
FFT, 128, 129
first-order noise shaping, 128
MATLAB® functions, 127
M-sequences, 126
Instability, 80, 89, 90, 99
Integrator finite gain and saturation, 216
Integrators
active gm-C, 159
active RC, 159

Integrators (*cont.*)
amplifier's finite DC gain, 161
GBW, 161
coefficients variation, 162
CT loop filter, 159
Gm-C, 159
Inter-symbol interference (ISI), 166, 168

J
Jitter behavior, CT sigma-delta modulator, 219, 220

L
Lee's criterion, 80, 82, 105, 108, 109
Loop filter, 5, 6
CIFB, 96, 98
CIFF, 98
CT sigma-delta modulator, 202–204
DT-CT equivalence, 153, 154
MOD1, 32
Loop filter, DT ΣΔ-modulator, 184
Loop stability, 150
Low-order modulators, 137, 141, 150

M
1+1+1 MASH
DC input, 141
finite integrator gain, 144–146
Simulink® model, 140
sinewave input, 142–144
STF and NTF, 140
2 + 1 MASH, 147
Matlab
DC input, 141
sinewave input, 142
MATLAB®, 12
Mid-rise quantizer, 23, 45
Mismatch
CT sigma-delta modulator
coefficients mismatch, 217
element mismatch, 216
DT ΣΔ-modulator
coefficients, 195, 196
element mismatch, 194, 195
sensitivity, 175
Mismatch error, 125
Mismatch noise, 126, 128
Multi-bit quantization, 89, 91, 108
Multi-bit quantizers, 115–120
DAC
mismatches, 118, 120
MOD2 FFT, 119
nonlinearity, 115, 116, 120
Simulink® model, 117
single-bit, 115
thermometer-code operation, 117
modulator stability, 114
SQNR, 114
vs. single-bit, 113–115
Multi-stage noise shapers (MASH)
analog transfer functions, 139
cascading unconditionally stable, 137, 139
low-order modulators, 137
property, 139
quantization error, 138, 139
quantizer gain, 148, 149
signal path gain, 149, 150

N
Noise
in-band, 101, 104, 108
level, 102
peak, 83
Noise coupling, 102, 103
Noise level, 31, 32
Noise power, 29, 30, 32, 45, 48, 65
Noise shaping, 2, 4, 17, 44, 61, 68, 83, 86, 100, 101
Noise transfer function (NTF)
constraints, 104, 105
ΣΔ-modulators, 105
noise suppression, 104
optimization, 100, 101
zero/poles mapping, 104
zeros, 100, 101
Non-delaying integrator, 23
Non-idealities, 157–159, 163, 174
Nonlinearity, 87, 109
Non-return-to-zero (NRZ), 163
NTF synthesis
CT sigma-delta modulator, 209–212
Nyquist, 3, 6, 16

O
Overload Level (OL), 11
Oversampling, 3, 4, 13, 16
Oversampling ratio (OSR), 182, 183, 245, 247

P
Pattern noise, 28
Performance metrics, 8, 10
Phase-locked loop (PLL), 168

Q
Quantization error, 2, 4, 14, 138–140, 147, 149
Quantization noise, 2, 4, 5, 13, 45, 137, 150
Quantizer bit, DT ΣΔ-modulator, 183, 245
Quantizer transfer curve (QTC), 62

R
Random element selection (RES)
  average error, 130
  DAC, 121
  DEM, 122, 123
  elements selection, 123
  FFT, 128, 129
  MATLAB® functions, 127
  white noise spectrum, 128
Return-to-zero (RZ), 163, 178

S
Schreier's Delta-Sigma Toolbox, 106–108
Schreier's models, 253
Schreier's Toolbox, 243
Second-order Sigma-Delta modulator (MOD2)
  architectures
    Boser-Wooley modulator, 71
    selection, 74
    signal and noise transfer, 70
    Silva-Steensgaard modulator, 73, 74
  DC inputs
    idle tone, 54
    time domain and FFT, 55
    tonal behavior, 55, 57
  finite integrator gain, 59, 60
  mid-rise quantizer, 69
  noise shaping, 68
  Simulink® model, 51, 53
  sinewave inputs
    amplitude sweep, 64–68
    FFT, 62
    integrator saturation, 63, 64
    Matlab, 61
    PSD, 63
    QTC, 62
  SQNR, 69
  stability, 58, 59
  z-domain, 68
Sigma-Delta (ΣΔ) modulators
  analytical results, 11, 12
  applications, 1
  block diagram, 6
  DR, 17
  DT, 2
  First-order, 7
  frequency domain analysis, 5
  low-pass (LP), 4
  NTF, 5
  performance metrics, 12
  PSD, 18
  SNDR, 17
  SNR, 17
  STF, 5
Sigma-Delta Simulink® Toolbox, 249
  CT, 255, 256
  DAC/DEM models, 254
  DT, 255
  incremental modulator, 256
  installation, 249
  MASH, 254, 255
  MATLAB routines, 250–252
  MATLAB® version, 250
  MOD1, 252
  MOD2, 252
  MOD3, 253
  Schreier's models, 253
  SMASH, 256
  VCO modulator, 257
Signal to Quantization Noise Ratio (SQNR), 9
Signal transfer function (STF), 5, 18
Signal-to-Noise and Distortion Ratio (SNDR), 9
Signal-to-Noise Ratio (SNR), 9
Signal-to-quantization noise ratio (SQNR), 107
  linear model, 63
  NTF, 91, 100
  peak, 93
  Sinewave inputs, 87
  stability, 105
  vs. swept amplitude, 65–67, 88
Silva-Steensgaard modulator, 73, 74
Simulink/Matlab (Toolbox/model), 181, 188, 194, 217, 227, 243
Simulink® model, 12, 80, 117, 127, 140
  MOD1, 22
  MOD2, 51, 53
Sinewave inputs, 38–40, 42, 43, 61–68
  CIFF-MOD3, 86, 87
  MOD1

Sinewave inputs (*cont.*)
amplitude sweep, 40, 42, 43
FFT, 38
low-pass filter, 39
time domain and FFT, 40
MOD2
amplitude sweep, 64–68
FFT, 62
integrator saturation, 63, 64
Matlab, 61
PSD, 63
QTC, 62
SQNR curve, 87
SQNR *vs*. swept amplitude, 88
STF, 88, 89
Single-bit *vs*. multi-bit quantizer, 113–115
Single-loop designs, 150
Spectre™, 12
Spurious-Free Dynamic Range (SFDR), 10
Stability, 80, 88, 89, 92, 100, 101, 105, 107
MOD1, 33
MOD2, 58, 59
SQNR, 246
Stage
digital filtering, 145
finite integrator gain, 146, 149
MASH structure, 138
Sturdy MASH (SMASH) modulators
architecture and differences, 226
concept of unity STF, 226
condition, 226
digital filters, 227
disadvantages, 225
low voltage technologies, 225
quantization error, 226
Simulink® Toolbox, 227
structure, 225
System level, 21
System-level modeling, 12
System-level simulation, 12

T
Third-order modulator (MOD3)
CIFF, 89, 90, 93
DC inputs, 83–85
finite integrator gain, 84, 85
NTF gain, 82
Simulink® model, 80
Sinewave inputs, 86–89
stability analysis, 80
Time domain ADCs
advantages, 231
LC-type oscillators, 231
VCO (*see* Voltage controlled oscillator (VCO))
VCO operation, 232, 234
Time-to-digital converter (TDC), 234
Tonal behavior
CT sigma-delta modulator, 214, 216
Total harmonic distortion (THD), 10, 198

V
Voltage controlled oscillator (VCO)
characteristics, 232
CT integrator, 232
dual-VCO quantizer, 241
FDC and TDC, 234
gain, 232
input voltage, 232
output, 232
voltage-to-frequency quantizer, 235–237
voltage-to-phase, 238, 240
Voltage-to-frequency ΣΔ-modulators, 235–237
Voltage-to-phase ΣΔ-modulators, 238, 240

W
White noise model, 31
Windowing function, 9