UNDERSTANDING
DELTA-SIGMA
DATA CONVERTERS

UNDERSTANDING DELTA-SIGMA DATA CONVERTERS

SECOND EDITION

SHANTHI PAVAN
RICHARD SCHREIER
GABOR C. TEMES

IEEE Press Series on Microelectronic Systems

IEEE PRESS

WILEY

For general information on our other products and services or for technical support, please contact our Customer Care Department within the United States at (800) 762-2974, outside the United States at (317) 572-3993 or fax (317) 572-4002.

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print may not be available in electronic formats. For more information about Wiley products, visit our web site at www.wiley.com.

CONTENTS

PREFACE

The earlier incarnation of this book[1] was aimed at describing the principles and operation of delta-sigma ($\Delta\Sigma$) modulators, as used in analog-to-digital (A/D) and digital-to-analog (D/A) converters, in simple conceptual terms, without relying on complicated mathematics. It also provided practical design information for both industrial and academic designers of $\Delta\Sigma$ converters. The book was well received, dubbed the green book, and sold many copies internationally. It was translated into Japanese, and reprinted in China. It is cited currently about 170 times annually in the literature. In view of this continued popularity, why did we embark on creating this new avatar?

The answer is that twelve years have gone by since the green book was published. The interest of $\Delta\Sigma$ converter designers has shifted significantly during this period, in the wake of many new applications for data converters at the extreme ends of the frequency spectrum. Continuous-time $\Delta\Sigma$ ADCs with GHz clocks, both for lowpass and bandpass signals, were required for wireless applications. At the other extreme of the spectrum, multiplexed ADCs with very narrow (sometimes only 10 Hz wide) signal bands, but very high accuracy, were needed e.g. in the interfaces of biomedical or environmental sensors. Often, the optimal converter for these specifications is the incremental ADC, which is basically a $\Delta\Sigma$ ADC that is periodically reset and restarted.

To reflect the changed needs of designers, this book includes much new material on both theory and design techniques. The emphasis of topics in the existing material has also been changed. New chapters have been added on the cascade (MASH) architecture, on

---

[1]Understanding Delta-Sigma Data Converters, R. Schreier and G. C. Temes, IEEE Press and Wiley-Interscience, 2005.

DAC mismatch effects and their mitigation, as well as expanded chapters on continuous-time $\Delta\Sigma$ ADCs and their nonidealities, on circuit design techniques for both sampled-data and continuous-time ADCs, and on incremental ADCs.

During the past decade, several new books that deal with special aspects of $\Delta\Sigma$ ADCs have been published. A recent book by de la Rosa and del Rio[2] provides an encyclopedic collection of practical information on $\Delta\Sigma$ ADCs. It is a valuable addition to the literature, highly recommended for designers. By contrast, the purpose of our work (as the title implies) is to give a basic understanding of the operation of these converters, and to provide general design techniques. We can think of several possible scenarios for using this book in a classroom setting. Chapters 1 through 6 form the core theory. A semester-long course focusing on discrete-time $\Delta\Sigma$ ADCs should, in addition, cover chapters 7, 12, 13 and 14. A course focusing on CT$\Delta\Sigma$Ms would cover Chapters 1-6, 8-11, and 14.

Several colleagues, from academia and industry, reviewed drafts of the book at various stages. It is our pleasure to acknowledge their assistance. Thanks are due to Trevor Caldwell (Analog Devices), Rakshit Datta (Texas Instruments), Ian Galton (University of California at San Diego), John Khoury (Silicon Laboratories), Victor Kozlov (Analog Devices), Saurabh Saxena (Indian Institute of Technology Madras), and Nan Sun (University of Texas at Austin). Their careful and astute comments have, in our opinion, helped improve the quality of the book. Amrith Sukumaran's editorial assistance is also appreciated.

To stick to limits imposed by space and time, some topics had to be omitted altogether, while others had to be given short shrift. Nevertheless, we hope that this book will be useful both for teaching and for self-education purposes.

<div align="right">

SHANTHI PAVAN

Chennai, India

RICHARD SCHREIER

Toronto, Canada

GABOR C. TEMES

Corvallis, USA

</div>

---

[2]CMOS Delta-Sigma Converters, J. M. de la Rosa and R. del Rio, IEEE Press and Wiley-Interscience, 2013.

CHAPTER 1

THE MAGIC OF DELTA-SIGMA MODULATION

The aim of this introductory chapter is to motivate the need for oversampling data converters, and to give a bird's-eye view of the topics covered in this book. Towards the end of the chapter, we give a brief overview of the origins of $\Delta\Sigma$ data conversion and trends in this exciting area.

1.1   The Need for Oversampling Converters

Computational and signal processing tasks are now performed predominantly by digital means, since digital circuits are robust and can be realized by extremely small and simple structures that can in turn be combined to obtain very complex, accurate, and fast systems. Every year, the speed and density (of transistors) of digital integrated circuits (ICs) increase, thereby enhancing the dominance of digital methods in almost all areas of communications and consumer products. Since the physical world nevertheless remains stubbornly analog, data converters are needed to interface with the digital signal processing (DSP) core. As the speed and capability of DSP cores increases, so too must the speed and accuracy of the converters associated with them. This presents a continual challenge to the lucky few engineers dedicated to the design of data converters!

Figure 1.1 illustrates the block diagram of a signal processing system with analog input and output signals, plus a central digital engine. As shown, the analog input signal (usually after some amplification and filtering) enters an analog-to-digital converter

1
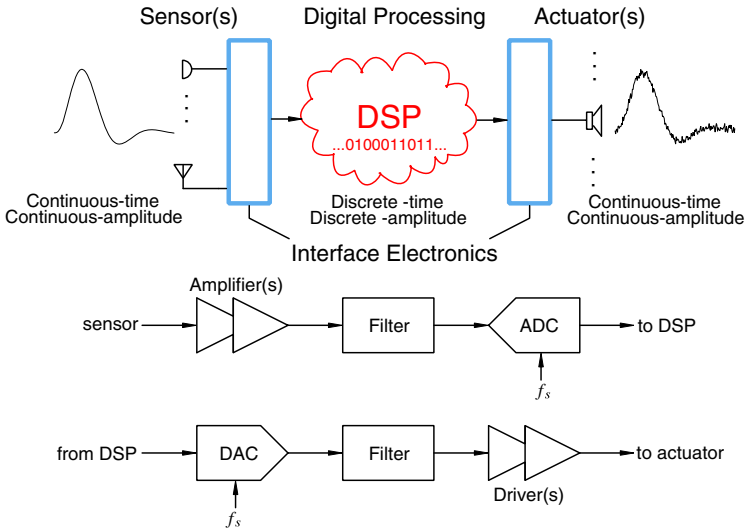
Figure 1.1   ADCs and DACs interface the real world to the virtual world.

(ADC), that transforms the input signal into a digital data stream. This stream is processed by the DSP core, and the resulting digital output signal is reconverted into analog form by a digital-to-analog converter (DAC). The DAC output is usually also filtered and amplified to obtain the final analog output signal.

Data converters (both ADCs and DACs) can be classified into two main categories: Nyquist-rate and oversampled converters. In the former category, there exists a one-to-one correspondence between the input and output samples. Each input sample is separately processed, regardless of the earlier input samples; in other words, the converter has no memory. Applying a digital input word containing bits $b_1, b_2, \cdots, b_N$ to a Nyquist-rate DAC ideally results in an analog output

$$V_{out} = V_{ref}(b_1 2^{-1} + b_2 2^{-2} + \cdots + b_N 2^{-N}), \tag{1.1}$$

where V_{ref} is the reference voltage, regardless of any previous input word. The accuracy of conversion can be evaluated by comparing the actual value of V_{out} with the ideal value given by (1.1).

As the name implies, the sampling rate f_s of a Nyquist-rate converter can be as low as Nyquist's criterion requires, i.e., twice the bandwidth B of the input signal. (For practical reasons, the actual rate is usually somewhat higher than this minimum value.)

In most cases, the linearity and precision of a Nyquist-rate converter is determined by the matching accuracy of the analog components (resistors, current sources, or capacitors) used in the implementation. For example, in the N-bit resistor-string DAC shown in Figure 1.2, the resistors must have a relative matching error less than 2^{-N} to guarantee an integral nonlinearity (INL)[1] less than 0.5 LSB. Similar matching requirements prevail for ADCs and DACs constructed from current sources or switched-capacitor (SC) branches.

---

[1]INL is simply the difference between the actual output and the ideal output.
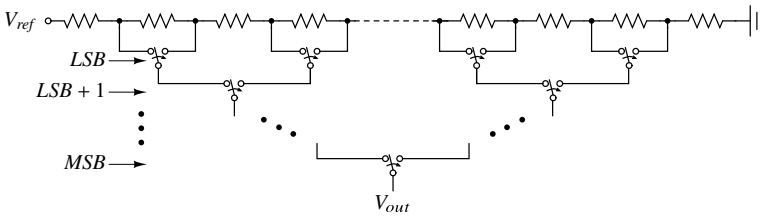
Figure 1.2   A resistor-string DAC. LSB denotes the least significant bit and MSB the most significant bit of the digital input.

Practical conditions restrict the matching accuracy to about 0.02%, and hence the effective number of bits (ENOB) to about 12, for such converters.

In many applications (e.g., digital audio), higher resolution and linearity are required, even as much as 18 or 20 bits. The only Nyquist-rate converters capable of such accuracy are the integrating or counting ones. These, however, require at least 2^N clock periods to convert a single sample, and hence, they are too slow for most signal processing applications.

Oversampling data converters are able to achieve over 20 ENOB resolution at reasonably high conversion speeds by relying on a trade-off. They use sampling rates much higher than the Nyquist rate, typically higher by a factor between 8 and 512, and generate each output utilizing numerous preceding input values. Thus, the converter incorporates memory elements in its structure. This property destroys the one-to-one relation between input and output samples. With oversampling converters, only a comparison of the complete input and output waveforms can be used to evaluate the converter's accuracy, either in the time or in the frequency domain.

A common measure of a converter's accuracy is the signal-to-noise ratio (SNR) for a sine-wave input. The relationship between ENOB and SNR (expressed in dB) for an ideal Nyquist converter with a full-scale sine-wave excitation is $SNR = 6.02\,ENOB + 1.76$. The inverse relationship is often applied to oversampling converters to convert the SNR into an effective number of bits.

As will be shown in later chapters, the implementation of oversampling converters requires a considerable amount of digital circuitry, in addition to some analog stages. Both need to be operated faster than the Nyquist rate. However, the accuracy requirements on the analog components are relaxed compared to those associated with Nyquist-rate converters. The price paid for high accuracy thus includes faster operation and added digital circuitry; both of these are getting cheaper as digital IC technology advances. Hence, the trade-off offered by $\Delta\Sigma$ converters continues to improve. As a result, they are gradually taking over in many applications previously dominated by Nyquist-rate converters.

1.2   Nyquist and Oversampling Conversion by Example

To better understand the difference between Nyquist and oversampled analog-to-digital conversion, consider the following illustrative examples.

1.2.1 The Coffee Shop Problem

A student visits a coffee shop on campus every morning to get her fix of caffeine, so that she can get through the day. A coffee grande at the campus cafe costs $3.47. What are the ways in which the student can pay this rather inconvenient sum? (The old-fashioned cafe does not accept credit cards). The "Nyquist" way of paying would be for the student to carry coins of the right denominations every day. She could, however, pay with a $5 bill and expect to shop assistant to return $1.53. The cafe, unfortunately, is severely short of small coins, and the shop assistant is not in a position to entertain this practice. Nevertheless, the shop assistant and the student come to an understanding that will allow the latter to pay with a $5 bill, while at the same time not under or overpay the cafe. It exploits the fact that the student visits the cafe *every day*. This is the $\Delta\Sigma$ way, described below.

The agreement between the two parties is the following. On any day, if the student owes the cafe more than $2.50, she hands a $5 bill to the shop assistant. When instead, she owes less than $2.50, she pays nothing. The student keeps track of how much she owes the cafe. The transactions for the first three days are shown in Figure 1.3.
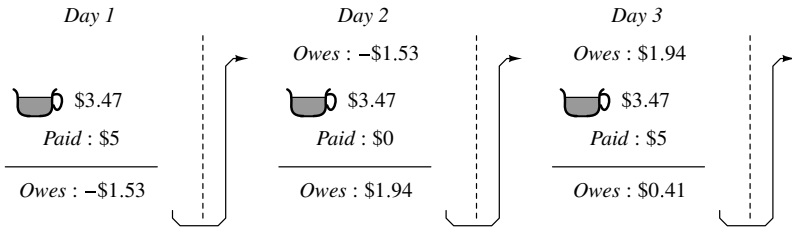
Figure 1.3   The $\Delta\Sigma$ way of paying $3.47 for a coffee grande, with only $5 bills.

On the first day, the student pays $5, as agreed upon. She notes, at the end of the day, that she owes −$1.53 to the cafe. The minus sign indicates that the student has overpaid.

While ordering at the cafe on the second day, the student reminds the shop assistant of the overpayment the previous day. The student needs to only pay $(3.47 − 1.53)=$1.94. As agreed upon, she pays nothing, again noting that the cafe is owed $1.94.

On the third day, the student needs to pay $5.41, and as per the understanding with the shop assistant, hands over a $5 bill. She notes that the cafe is owed $0.41. This continues every day *ad infinitum*.
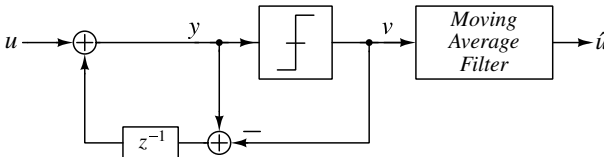
Figure 1.4   The algorithm of Figure 1.3. The u represents the cost of a coffee grande, and $v[n]$ is the payment made by the student on the nth day.

When the scheme above is cast into a signal flow diagram, Figure 1.4 results. In the figure, u presents the price of the coffee grande; $y[n]$, which is the input of the quantizer, represents the total amount owed by the student while ordering on the nth day; $v[n]$, which

is the quantizer output, represents the students payment on the nth day, and takes the value 0 or 5. Therefore, $(y[n] - v[n])$ is the amount owed by the student to the cafe after making the payment on the nth day. The z^{-1} block in Figure 1.4 denotes a delay of one day.
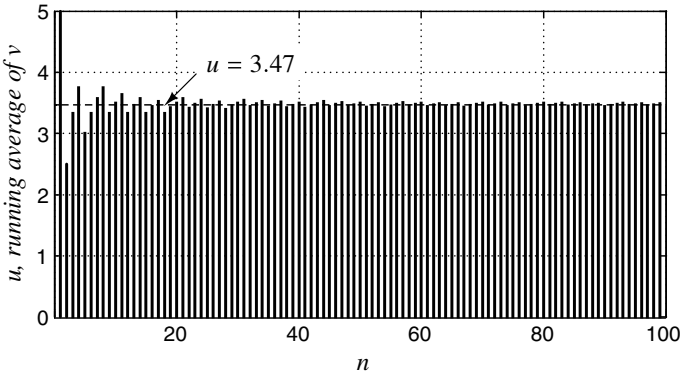
Figure 1.5   The running average of v approaches u for large n.

Figure 1.5 shows the running average of v, given by

$$\frac{1}{n}\sum_{k=1}^{n} v[k]. \tag{1.2}$$

The running average represents the price paid by the student per unit coffee grande, on average during the preceding days. As n becomes large, we see it approaches u, which is $3.47.

In the beginning of this discussion, it might have seemed surprising that the student would be able to pay an inconvenient sum of $3.47 with only $5 bills. The $\Delta\Sigma$ way exploits the fact that u remains substantially the same from sample to sample. It uses feedback to make v approximate u *on average*. An individual sample of v has no meaning – one can determine u from v only by averaging many samples. Why does this scheme work? It is perhaps easier to see this by redrawing the diagram of Figure 1.4 as in Figure 1.6. We see that $y[n]$ is the total amount owed by the student (from the beginning of time) after grabbing her coffee for the day. As long as this is bounded, it must follow that the average of the accumulator's (Σ) input must be close to zero. Since the input to the accumulator is the difference (Δ) between the input and feedback sequence, it should follow that v and u would be equal, on average. Thus, by embedding a (very coarse) 2-level quantizer into a negative feedback loop, and sufficiently averaging the output sequence, the digital estimate $\hat{u}$ can be a very good representation of u.
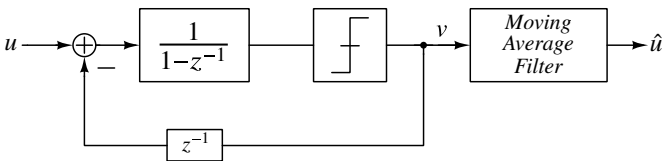
Figure 1.6   The system of Figure 1.4, redrawn.

The feedback loops of Figures. 1.4 and 1.6, both equivalent, represent a first-order $\Delta\Sigma$ modulator. The first structure is called the error-feedback structure, while the latter is the more traditional (and immediately recognizable) error accumulating structure.

1.2.2 The Dictionary Problem

A student visiting a bookstore begins to wonder about the thickness of that venerable tome, the *Webster's International Dictionary of the English Language*. An immediate way of finding the thickness is to get hold of a 6-inch ruler (this is a bookstore, after all) and measure the dictionary's thickness, as illustrated in Figure 1.7. Since the ruler has markings at every eighth of an inch, the worst-case error in measuring the thickness would amount to one-sixteenth of an inch. This is the "Nyquist" way, where the distance between successive marks on the ruler would correspond to the LSB. Measurement uncertainty (quantization error, in data-conversion parlance) can only be reduced by using a ruler with more finely spaced markings. The effort involved in making such a ruler is decidedly higher, not to mention the difficulty in discerning the marking that best corresponds to the height of the tome. Note, however, that the measurement is made in one shot – meaning that one use of the ruler is sufficient for measurement.

Figure 1.7   Measuring the thickness of *Webster's Dictionary* the Nyquist way.

The student finds focusing on the finely spaced levels a strain on the eyes, and he begins to wonder if it is at all possible to measure the book's thickness without having to look at the marks on the ruler at all. In other words, is it possible to find the thickness to within one-sixteenth of an inch (or even better) using only the fact that this is a 6-inch ruler? At first, this may seem like an impossible task – how is it possible to measure to within a fraction of an inch with a scale whose only "marking" is 6 in?

The student, being resourceful, exploits the fact that the bookstore has any number of copies of the *Webster's Dictionary* that he can put at his disposal. He contrives the following algorithm, which, he reasons, should allow him to determine the dictionary's thickness to arbitrary precision. The algorithm involves a sequence of operations, and proceeds as follows as illustrated in Figure 1.8.

Markings are made on the wall, from the floor up, at intervals of 6 inches using the (6-inch) ruler. The student places a copy of the *Webster's Dictionary* on the floor. The action of placing the book causes the level corresponding to the top of the stack of dictionaries (which contains just one instance at this time) to cross the lowest (6-inch) mark on the wall, which is at the floor level. The result of this experiment, denoted by v, is decreed to be 6 (corresponding to the 6-inch ticks on the wall).

A copy of *Webster's* is placed on the first, as shown in Figure 1.8(b). Since the action of adding the second copy causes the height of the stack to cross a marking on the wall, the result of this experiment is also deemed to be 6. This mode of operation continues *ad infinitum*. v at the end of every step, therefore, is 6 if the addition of a new copy causes
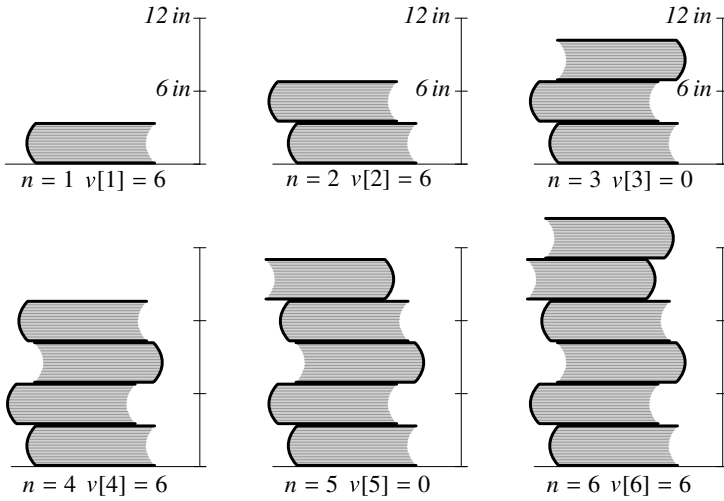
Figure 1.8   Measuring the thickness of *Webster's Dictionary* the $\Delta\Sigma$ way.

the stack to cross a new 6-inch mark, and zero otherwise. Denoting the thickness by u, the height of the stack in the nth instance is given by

$$\sum_{k=1}^{n} u = nu. \tag{1.3}$$

This is compared with the *next* 6-inch mark on the wall, whose height is given by

$$\sum_{k=1}^{n-1} v[k]. \tag{1.4}$$

Thus,

$$v[n] = \begin{cases} 6, & \sum_{k=1}^{n} u \geq \sum_{k=1}^{n-1} v[k], \\ 0, & otherwise. \end{cases}$$

The student argues that at the end of n operations,

$$0 < \sum_{k=1}^{n} v[k] - \sum_{k=1}^{n} u < 6, \tag{1.5}$$

since the height of the stack and the mark immediately above the top of the stack can differ by at most 6 inches. This means that

$$\frac{1}{n} \sum_{k=1}^{n} v[k] - \frac{6}{n} < u < \frac{1}{n} \sum_{k=1}^{n} v[k]. \tag{1.6}$$

An estimate of u can therefore be obtained by simply averaging the sequence $v[n]$. As n approaches infinity, the average of the output sequence approaches the true height of our venerable tome, which is about 3.42 inches.

When the student's scheme is translated into the language of electrical engineering, the diagram shown in Figure 1.9 results. The input u is summed in a delay-free integrator.
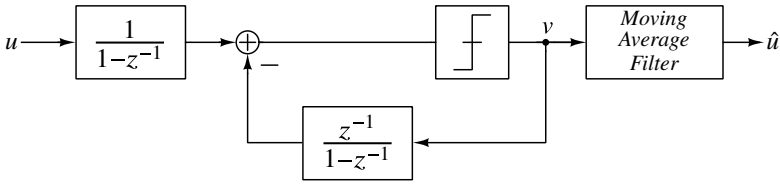
Figure 1.9   Equivalent representation of the algorithm in Figure 1.8.

The output sequence v is summed (Σ) using a delayed integrator, since the current decision depends on the sum of the previous decisions. The difference (Δ) between the two accumulated results is quantized to one of two levels (0 and 6 in our example). The resulting output sequence v is averaged (by a moving-average filter) to estimate the input u. The averaging filter acts on a digital input and is, therefore, a digital filter.
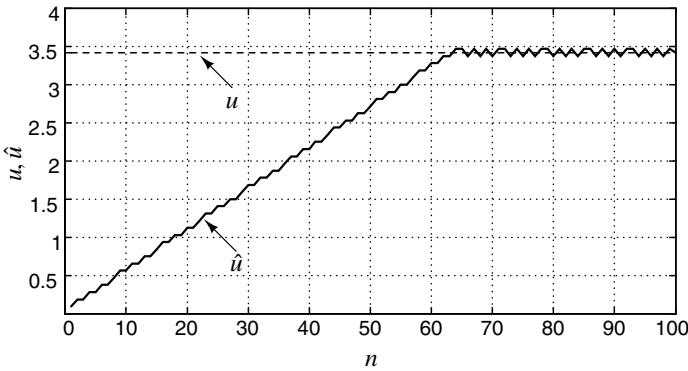
Figure 1.10   First 100 samples of the output of the moving average filter in Figure 1.11, for $u = 3.42$. A 64-tap filter (with all taps equal) is assumed.

Figure 1.10 shows the first hundred samples of $\hat{u}$ at the output of a 64-tap moving-average filter. In steady state, $\hat{u}$ happens to be within 0.05 inches of u. At first sight, it indeed seems remarkable that one can resolve to a small fraction of an inch with a scale marked only at 6 inches!

It is instructive to compare the Nyquist and $\Delta\Sigma$ ways of measurement. The former is a one-shot process, with the accuracy of measurement depending on the fineness and precision of the marks on the ruler. The latter, in contrast, is an iterative process. It involves feedback, since the outcome $v[n]$ of the nth iteration depends on the results of previous experiments. The $\Delta\Sigma$ method relies on the fact u does not change between successive iterations. This means that u is heavily *oversampled*. Moreover, $v[n]$ is not representative of u; u can only be inferred by averaging the outcomes of a large number of iterations. Measurement accuracy generally improves as n is increased. Averaging 1000 samples reduces the error to 0.006 inches.

A practical problem with the realization of Figure 1.9 is that the outputs of both integrators keep increasing with n. In our bookstore example, the pile of dictionaries in Figure 1.8 would risk hitting the ceiling due to lack of headroom. Likewise, electronic integrators have limits on their maximum allowable output. This can be circumvented by

simply moving the integrators into the loop, as shown in Figure 1.11. In the figure, $\hat{u}$ is a digital representation of u, and the system converts the continuous valued input u into a quantized output. This is achieved by embedding a coarse quantizer (which, in our example, has only two levels – 0 and 6 inches) in a negative feedback loop. The feedback loop of Figure 1.11 is called a $\Delta\Sigma$ modulator (or $\Delta\Sigma$ converter). More precisely, it represents a first-order, 2-level $\Delta\Sigma$ modulator. The integrator, whose output is quantized, is often referred to as the *loop filter*.

The discussion in this section was a (hopefully) gentle introduction to the basic idea behind $\Delta\Sigma$ modulation. A more detailed development of the first-order $\Delta\Sigma$ loop, its analysis and alternative ways of realizing the same functionality are given in Chapter 2.
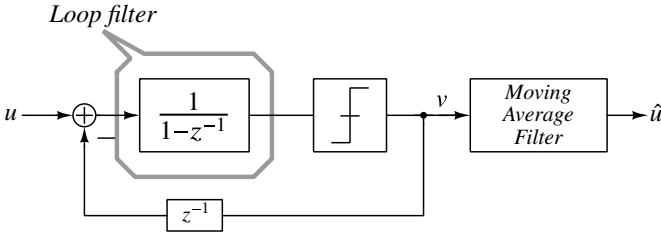
Figure 1.11 Addressing "headroom problems" of the system of Figure 1.9 by moving the integrators into the loop. Averaging v yields an estimate $\hat{u}$ of u.

The reader might wonder why the measurement must proceed in the iterative fashion shown in Figure 1.8. Why not stack 64 dictionaries, and measure the height of the stack (to the nearest 6 inch mark) and divide by 64? To understand this, we denote the error introduced by the quantizer of Fig 1.11 in the nth iteration by $e[n]$. It is easy to see that

$$v[n] = u[n] + e[n] - e[n-1].\tag{1.7}$$

The output of the M-tap moving-average filter (with weights being equal) is given by

$$\hat{u} = \frac{1}{M}\sum_{k=r+1}^{M+r} v[k] = u + \frac{1}{M}\left(e[r+M+1] - e[r]\right).\tag{1.8}$$

It is easy to see that $\hat{u}$ is what one would obtain by stacking up M dictionaries, measuring the height of the stack to the nearest 6 inches, and dividing the result by M. From the equation above, we observe that the estimation error in $\hat{u}$ is due to the e in the first and last of the 64 (assuming $M = 64$) samples being processed by the moving-average filter. This suggests that quantization error can be reduced by weighting $v[n]$ non-uniformly – that is, by attaching more importance to the middle set of samples than those toward the end. This intuition is confirmed by filtering the output sequence of the modulator with a 64-tap moving-average filter with a triangular impulse response. From Figure 1.12, we see that the peak-to-peak excursion of the output of such a filter is much smaller than that in the case where all the samples of $v[n]$ are equally weighted. Thus, there is merit to observing the height of the stack every time an additional dictionary is added, as this enables the use of arbitrary moving-average filters. Recall that measuring the height of a stack of 64 dictionaries (to the closest 6-inch mark) and dividing by 64 is equivalent to uniformly weighting the samples of v. To summarize, there is more to choosing the post-filter that processes the modulator's output than simply averaging the output. To understand how
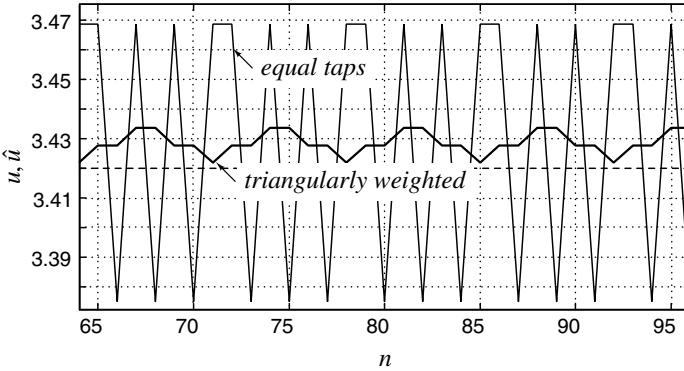
Figure 1.12   Outputs of moving-average filters with equal weights, and a triangularly weighted response.

one designs a post-filter, it is helpful to examine a $\Delta\Sigma$ modulator in the frequency domain, which we will do going forward. Before that, we wish to draw the reader's attention to the following.

The example above considered the modulator's input u to be constant. In practice, the input to be digitized has a nonzero bandwidth (which is much smaller than the sampling rate). Then, the output of the digital post-filter (which is a sequence at the sampling rate) can be downsampled, so that the output sample rate can equal the Nyquist rate corresponding to the input signal. Figure 1.13 shows the system model of an ADC employing
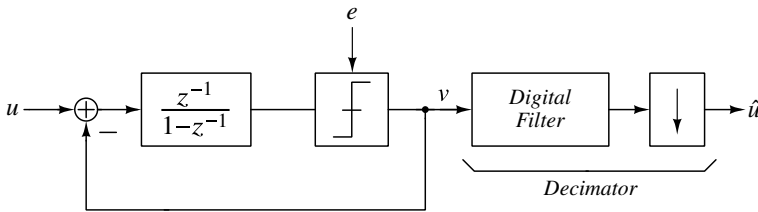
Figure 1.13   System model of an ADC with a first-order $\Delta\Sigma$ modulator.

a first-order $\Delta\Sigma$ modulator. The delay element in the feedback path of the modulator of Figure 1.11 has been pushed into the forward path. The (benign) consequence of this is to delay the input by one sample. The combination of the the digital post-filter and downsampler is called the *decimation filter* or decimator.

The output noise due to the quantization error in the $\Delta\Sigma$ modulator is $q[n] = e[n] - e[n-1]$. In the z-domain, this becomes $Q(z) = (1-z^{-1})E(z)$, and in the frequency domain, after z is replaced by $e^{j\omega}$, the power spectral density (PSD) of the output noise is found to be

$$S_q(\omega) = 4\sin^2\left(\frac{\omega}{2}\right) S_e(\omega). \tag{1.9}$$

Here, $S_e(\omega)$ is the one-sided PSD of the quantization error (noise) of the internal ADC. For "busy" (i.e., rapidly and randomly varying) input signals, one may approximate e with white noise of mean-square value $\Delta^2/12$, where Δ is the step size of the quantizer, and thus

obtain

$$S_e(\omega) = \frac{\Delta^2}{12\pi}. \tag{1.10}$$

The filtering function $(1 - z^{-1})$ is called the noise transfer function (NTF). The squared magnitude of the NTF as a function of frequency is illustrated in Figure 1.14. As the
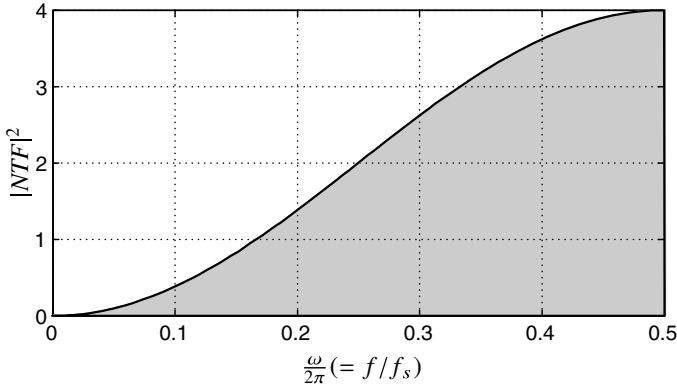
Figure 1.14   Noise-shaping function for the $\Delta\Sigma$ modulator shown in Figure 1.13.

figure shows, the NTF of the $\Delta\Sigma$ modulator is a highpass filter function. It suppresses e at frequencies around 0, but the NTF also enhances e at frequencies around $\omega = \pi$.

We introduce next the *oversampling ratio*

$$OSR = \frac{f_s}{2f_B}, \tag{1.11}$$

where f_B is the maximum signal frequency, which is the signal bandwidth. *OSR* defines how much faster we sample in the oversampled modulator than in a Nyquist-rate converter.

It turns out that the in-band component of quantization noise at the output of the modulator is given by

$$q_{rms}^2 = \frac{\pi^2}{3} \frac{e_{rms}^2}{OSR^3}. \tag{1.12}$$

As expected, the in-band noise decreases with increasing OSR. However, this decrease is relatively slow; doubling the OSR reduces the noise only by 9 dB, and hence it enhances the ENOB by only about 1.5 bits.

The discussion in this chapter is intended merely as an introduction, a whetting of the appetite – the topics of sampling, oversampling and the first-order $\Delta\Sigma$ modulator are covered in detail in Chapter 2.

1.3 Higher-Order Single-Stage Noise-Shaping Modulators

As the reader might have anticipated, a way to increase the resolution (i.e., the ENOB) of a $\Delta\Sigma$ modulator is to use a higher-order loop filter. By adding another integrator and feedback path to the modulator of Figure 1.13, the structure of Figure 1.15 results. Linearized
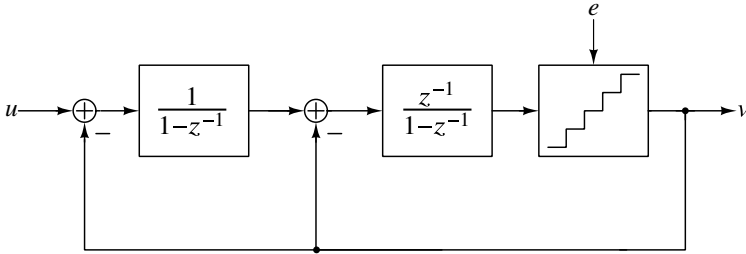
Figure 1.15   A second-order $\Delta\Sigma$ modulator.

analysis yields

$$V(z) = z^{-1}U(z) + (1 - z^{-1})^2 E(z). \tag{1.13}$$

This indicates that the NTF is now $(1 - z^{-1})^2$ in the z-domain, which applies a shaping function of $(2\sin(\omega/2))^4$ to the PSD of e. It follows that the in-band noise power is (to a good approximation for $OSR \gg 1$)

$$q_{rms}^2 = \frac{\pi^4 e_{rms}^2}{5\,OSR^5}. \tag{1.14}$$

Doubling OSR, therefore, results in about 2.5 bits of additional resolution. This is a much more favorable trade-off than that of the first-order modulator. A more detailed analysis of second-order $\Delta\Sigma$ modulators, and alternative ways of realizing them, are discussed in Chapter 3.

In principle, by adding more integrators and feedback branches to the loop, even higher-order NTFs can be obtained. For an Lth-order loop filter resulting in $NTF(z) = (1 - z^{-1})^L$, the in-band noise power is approximately

$$q_{rms}^2 = \frac{\pi^{2L} e_{rms}^2}{(2L + 1)\,OSR^{2L+1}}, \tag{1.15}$$

and the number of bits added to the resolution by doubling the OSR is given by $(L + 0.5)$.

From the discussion above, it appears as if using a $\Delta\Sigma$ loop with an appropriately chosen (very high-order) NTF can attain arbitrarily high SNRs, even for small OSR. This sounds too good to be true, and as the wise reader should suspect, something that *sounds* too good to be true *is probably* too good to be true. It turns out that for high-order loops, stability considerations, which have thus far been ignored, reduce the achievable resolution to a lower value than that given by the equations above. For high-order single-bit modulators, the difference is substantial, amounting to more than 60 dB for a fifth-order modulator. Higher-order $\Delta\Sigma$ modulators, their stability, trade-offs involved in their design, and various means of realizing them will be discussed in detail in Chapter 4.

1.4   Multi-Stage and Multi-Quantizer Delta-Sigma Modulators

The philosophy behind using a high-order loop to suppress in-band quantization noise is to *divide* noise by a large loop-gain, obtained by incorporating more integrators in the loop. An alternative strategy to accomplish the same objective is to *cancel* the quantization error

by measurement and subtraction. It turns out that this approach eases the stability problems associated with high-order modulators. The resulting structures are called cascade modulators, and also referred to as multi-stage or MASH (for Multi-stAge noise-SHaping) modulators. This, and other techniques based on this fundamental idea, form the subject of Chapter 5.

The basic concept behind a cascade modulator is illustrated in Figure 1.16. The output
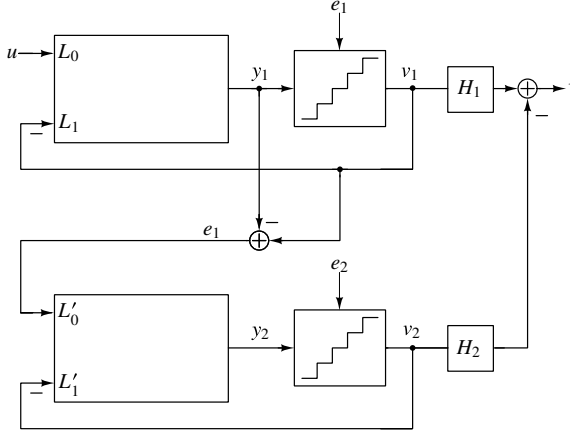
Figure 1.16   A multi-stage delta-sigma modulator.

signal of the first stage is given by

$$V_1(z) = STF_1(z)U(z) + NTF_1(z)E_1(z), \tag{1.16}$$

where STF_1 and NTF_1 are the signal and noise transfer functions, respectively, of the first stage. The second stage is added to improve the SNR beyond what NTF_1 can provide.

As shown in Figure 1.16, the quantization error e_1 of the input stage is found in analog form by subtracting the input to its internal quantizer from its output. e_1 is then fed to another $\Delta\Sigma$ loop forming the second stage of the modulator, and converted into digital form. Hence, the output signal of the second stage in the z-domain is given by

$$V_2(z) = STF_2(z)E_1(z) + NTF_2(z)E_2(z), \tag{1.17}$$

where STF_2 and NTF_2 are the signal and noise transfer functions, respectively, of the second stage. 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 of the system, the first-stage error e_1 is canceled. By the equations above, this is achieved if the condition

$$H_1(z)NTF_1(z) = H_2(z)STF_2(z) \tag{1.18}$$

holds. The simplest (and usually most practical) choice for H_1 and H_2 that satisfies (1.18) is $H_1 = k \cdot STF_2$ and $H_2 = k \cdot NTF_1$, where k is constant chosen to give unity signal gain. Since STF_2 is often just a delay, H_1 is easily realized. The overall output is then given by

$$V(z) = k \cdot STF_1(z)STF_2(z)U(z) + k \cdot NTF_1(z)NTF_2(z)E_2(z). \tag{1.19}$$

In a typical case, both stages of the MASH modulator may contain a second-order loop, and their transfer functions may be given by $STF_1 = z^{-1}$, $STF_2 = 0.5z^{-2}$ and $NTF_1 = NTF_2 = (1 - z^{-1})^2$. Choosing $k = 2$, the output we obtain is then

$$V(z) = z^{-2}U(z) + 2(1 - z^{-1})^4 E_2(z). \tag{1.20}$$

Thus, the noise-shaping performance is essentially that of a fourth-order single-loop converter, but the stability behavior is that of a second-order one.

If the condition (1.18) is not exactly satisfied, for example, due to imperfections in the realization of the analog transfer functions, then $E_1(z)$ will appear at the output multiplied by $k \cdot [STF_2NTF_{1a} - NTF_1STF_{2a}]$, where the subscript a denotes the actual value of the analog transfer function. This is not surprising, since the efficacy of *any* technique based on cancellation is always degraded by mismatch. As will be shown in Chapter 5, mismatch can result in a serious deterioration of the noise performance of the converter.

1.5   Mismatch Shaping in Multi-Bit Delta-Sigma Modulators
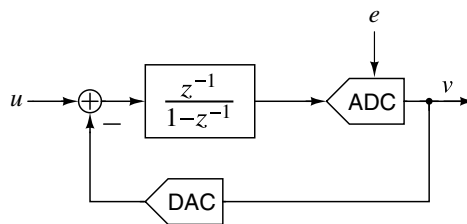
Figure 1.17   The quantizer in a $\Delta\Sigma$ modulator is implemented as a cascade of ADC and DAC.

A quantizer is implemented as a cascade of an ADC and DAC, as shown in Figure 1.17. The DAC appears in the feedback path of the $\Delta\Sigma$ modulator, and its nonlinearities result in comparable nonlinearities for the overall conversion. This occurs because the in-band part of the DAC output signal is forced by the feedback loop to follow the input signal u very accurately. Hence, if the DAC is nonlinear, its input will be distorted to give an accurate output. Since the DAC input is the output of the converter, the converter output is distorted.

It was this fact that forced early designers of $\Delta\Sigma$ modulators to use single-bit internal DACs in the $\Delta\Sigma$ loops. A single-bit DAC has the immensely important virtue of *inherent linearity*. Since the input to a one-bit DAC only takes on two values, the transfer characteristic of the DAC can be represented by two points in the input–output plane. Thus, a straight line that passes through those points models a 1-bit DAC exactly. In other words, the DAC is *exactly* described by an equation of the form $v_d = kv + offset$, where k and *offset* are constants. Since a system that obeys such a model does not introduce distortion, a 1-bit DAC is said to be *inherently linear*.

In contrast, single-bit ADCs (which are essentially comparators) have an ill-defined gain factor, as will be shown in Chapter 2. Also as Chapters 3 and 4 will show, loops containing one-bit quantizers must remain stable over a wide range of loop gains. This

consideration results in a reduction of the allowable input signal swing, and hence a reduction in the achievable SNR.

For a multi-bit quantizer, the loop is inherently more stable because the quantizer gain is well-defined, and its no-overload range is increased. In fact, linear analysis can be used to design the modulator so that its stability is guaranteed. Furthermore, since the quantization noise decreases by 6 dB for each bit added to the quantizer, and since aggressive high-order noise-shaping functions can used, multi-bit modulators can have very high ENOB even at low OSR values. Hence, there is strong motivation to solve the problem of DAC nonlinearity inherent in the use of multi-bit quantization. While brute-force techniques, such as element trimming, have been used earlier, the techniques currently in favor use auxiliary digital circuitry to manipulate the elements of the DAC so as to reduce the in-band portion of the error signal introduced by DAC nonlinearities. These techniques are conceptually very similar to the noise shaping used in $\Delta\Sigma$ modulators, and are often described with the term *mismatch shaping*. As with noise shaping, the effectiveness of mismatch shaping increases with increasing OSR. For very low OSR values ($OSR < 8$), digital techniques can be used to determine and then correct the nonlinearities of the DAC.

The fundamental principles behind addressing DAC mismatch in multi-bit delta-sigma modulators will be covered in detail in Chapter 6.

1.6   Continuous-Time Delta-Sigma Modulation

At the beginning of this chapter, we saw that an ADC converts a continuous-time analog signal (which is continuous in time and amplitude) to a digital one (where time and amplitude are quantized). A discrete-time $\Delta\Sigma$ modulator acts on a sampled version of the analog signal, and its role is to quantize these samples. A continuous-time $\Delta\Sigma$ modulator (CT$\Delta\Sigma$M), on the other hand, works with the continuous-time input $u(t)$. There are many ways of understanding a CT$\Delta\Sigma$M – and the development below is appropriate for an introductory chapter such as this. A more general development, building on previous chapters discussing discrete-time $\Delta\Sigma$ design, is given in Chapter 8.

Figure 1.18(a) shows a first-order lowpass filter. The opamp is ideal. Regardless of the (continuous-time) input $u(t)$, the average capacitor current $\overline{i_c(t)}$ must be zero – otherwise, the voltage across the capacitor would become unbounded. Thus, $\overline{i_1(t)} = \overline{i_2(t)}$, which results in $\overline{u(t)} = -\overline{v(t)}$.

Next, the output of the opamp is sampled at a rate f_s, as shown in Figure 1.18(b). The resulting *sequence* $v[n]$ is zero-order held, before being fed back through the resistor. Assuming that the feedback loop is functional, the average capacitor current is still zero, meaning that the average of $u(t)$ is still equal to the average of $v(t)$. The latter now refers to the output of the ZOH. $\overline{v(t)}$, however, equals the average value of the *sequence* $v[n]$. By inserting a sampler into the loop, therefore, we are now in a position to relate the average of the input *waveform* $u(t)$ to the average of the output *sequence*. Now, if the input signal varies very slowly (in relation to the sampling period), $u(t)$ and its local average are largely the same. Under this circumstance, $u(t) \approx -v[n]$. Half the battle of analog-to-digital conversion has been won – we have accomplished discretization of time.

The next progression is to quantize $v[n]$ before feeding it back through the ZOH. *Assuming* the loop is still stable, and that $u(t)$ is slowly varying, $u(t)$ is still approximately
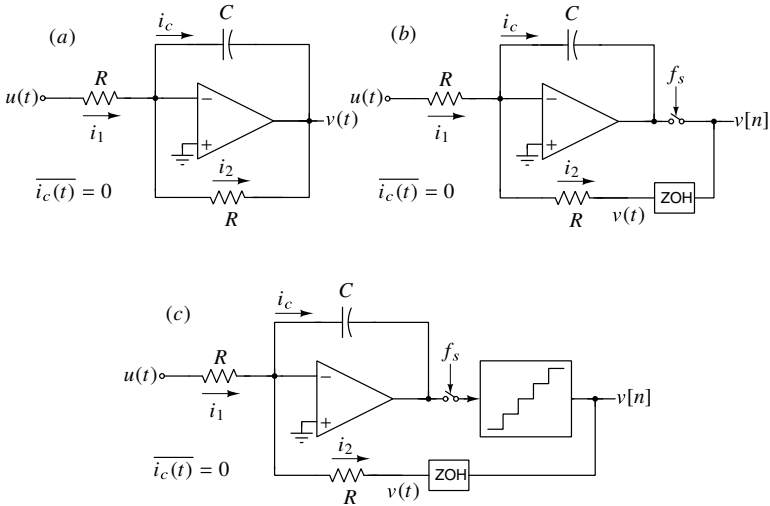
Figure 1.18   (a) A first-order lowpass filter. (b) Sampling the opamp's output and feeding back the sampled sequence using a zero-order hold (ZOH). (c) Quantizing the output sequence before feeding it back.

equal to $-\overline{v[n]}$. $v[n]$ is now not only discrete in time, but also in amplitude. It can, therefore, be represented in digital form. Note that, as in the discrete-time case, it is only the *average* of $v[n]$ that approximates u; the individual samples have no meaning in isolation. The system of Figure 1.18(c) is a first-order CTΔΣM – its output sequence has to processed by an appropriately chosen digital filter, so as to properly average $v[n]$ and yield an estimate of $u(t)$. As in the discrete-time case, higher-order loop filters can better reject in-band quantization noise.

It is instructive to examine the output of the first-order CTΔΣM when $u(t) = \cos(2\pi f_s t)$, that is, for a sine wave whose frequency equals the sample rate[2]. Since the virtual ground voltage is zero, and $\overline{u(t)} = 0$, it follows that $\overline{i_1(t)} = 0$. This means that $\overline{i_2(t)}$ has to be zero. This, in turn, implies that $\overline{v[n]} = 0$, indicating that the CTΔΣM does not respond to an input at its sampling frequency! This remarkable property of a CTΔΣM distinguishes it from all other ADC families, where an input at f_s cannot be distinguished from an input at dc. This ability of a CTΔΣM to respond differently to inputs at dc and f_s is referred to as *implicit anti-aliasing.* Chapter 8 gives a detailed discussion of the fundamentals of continuous-time ΔΣ modulation.

However, a CTΔΣM is the victim of many non-idealities – excess delay in the quantizer, time-constant variations in the loop filter, clock jitterand so forth. Chapter 9 analyzes the effects of these nonidealities and how they may be circumvented. Chapter 10 gives circuit design considerations for the blocks that make up a CTΔΣM.

The block diagram of a generic single-loop CTΔΣM is shown in Figure 1.19. It is a bag of contrasts. The loop filter processes $u(t)$ and $v(t)$, and it should be linear and time-invariant. The output of the filter is sampled and quantized. Sampling is a time-

---

[2]When a Nyquist converter without an anti-alias filter up front is subject to a sinusoidal input at its sampling frequency, the output is a dc sequence, indicating that the tone at f_s aliases to dc.
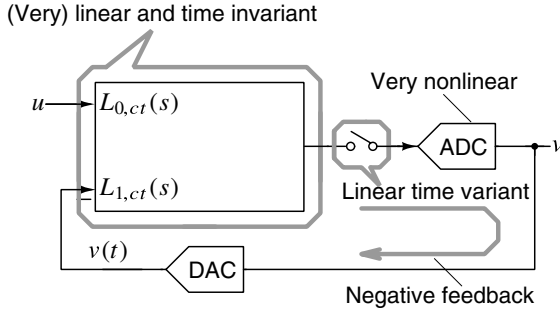
Figure 1.19   Block diagram of a generic single-loop CT$\Delta\Sigma$M.

varying operation. The quantizer is nonlinear; in fact, it is extremely so, due to the abrupt steps in its transfer curve. To add to the excitement, all these blocks are enclosed in a negative-feedback loop. Thus, a CT$\Delta\Sigma$M may be described as a part linear, part nonlinear, part time-invariant, part time-variant, part continuous-time, and part discrete-time system incorporating negative feedback. Understanding a CT$\Delta\Sigma$M, therefore, exposes one to a variety of topics – from signal processing and systems theory to precision circuit design. Apart from its obvious pedagogical value, a CT$\Delta\Sigma$M is also highly relevant in practice. It is what we would like to refer to as *a system for all seasons*.

1.7   Bandpass Delta-Sigma Modulators

Up to now, it was assumed that the signal energy was concentrated in a narrow band at low frequencies, centered at dc. In applications such as RF communication systems, the signal is concentrated in a narrow band of width f_B around a center frequency f_0, where f_B is much smaller than f_s while f_0 is not. In such cases, $\Delta\Sigma$ modulation may still be effective, but now the noise transfer function NTF must have a bandstop, rather than highpass, character, with zeros located at or around f_0.
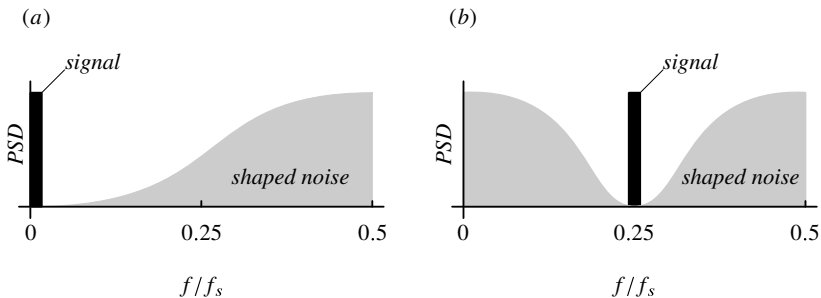
Figure 1.20   Conceptual output spectra for (a) lowpass and (b) $f_s/4$ bandpass modulators.

Figure 1.20 compares the conceptual output spectra of a lowpass and a bandpass $\Delta\Sigma$ modulator. A simple way to obtain the NTF of a bandpass $\Delta\Sigma$ modulator is to find first an appropriate lowpass NTF, and then perform a z-domain mapping on it. For example, the transformation $z \rightarrow -z^2$ maps the frequency range around dc (i.e., $z = 1$) to the

ranges around $\pm f_s/4$ (i.e., $z = \pm j$). Hence, the resulting NTF will have small values near $f_o = f_s/4$, and will suppress the quantization noise there. This bandstop noise-shaping makes it possible to achieve a high signal-to-noise ratio (SNR) for signals whose energy is restricted to frequencies near $f_s/4$.

Note that the mapping doubles the order of the lowpass NTF and transforms the zeros of the NTF from near $z = 1$ to the vicinity of the points $z = \pm j$, as illustrated in Figure 1.21. Other techniques for finding the NTF of bandpass $\Delta\Sigma$ modulators will be discussed in
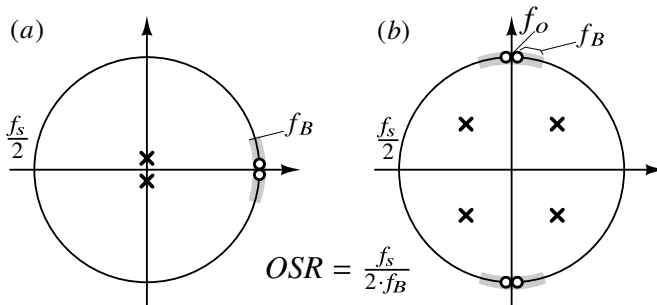
Figure 1.21   Pole-zero locations of (a) a lowpass NTF and (b) a bandpass NTF.

detail in Chapter 11 along with circuit design techniques for bandpass $\Delta\Sigma$ modulators.

1.8   Incremental Delta-Sigma Converters

In our discussions so far, we evaluated the in-band quantization noise of a delta-sigma modulator by integrating its spectral density over the signal bandwidth. This is justified only if the digital filter following the modulator has a brick-wall response. Such a requirement can only be approximated in practice, and like all sharp filters, the impulse response of the filter can be extremely long. This means that the $\Delta\Sigma$ modulator and accompanying post-filter have significant memory. This makes a conventional $\Delta\Sigma$ modulator unsuitable in applications where the ADC is multiplexed among multiple sensors, or when the ADC has to be operated in an intermittent fashion.

A different ADC scheme that applies the noise-shaping algorithm of $\Delta\Sigma$ ADCs, but only within the sample-by-sample operation of a Nyquist-rate ADC, is the *incremental* $\Delta\Sigma$ ADC. This family of ADCs is closely related to their conventional cousins, and it forms the subject of Chapter 12.

1.9   Delta-Sigma Digital-to-Analog Converters

The motivation for using $\Delta\Sigma$ modulation to realize high-performance DACs is the same as for ADCs: it is difficult, if not impossible, to reliably achieve an untrimmed linearity and accuracy better than about 14 bits for DACs operated at the Nyquist rate. With $\Delta\Sigma$ modulation, this task becomes feasible. A $\Delta\Sigma$ DAC system is illustrated in Figure 1.22. By operating a fully digital $\Delta\Sigma$ modulator loop at an oversampled clock rate, a data stream
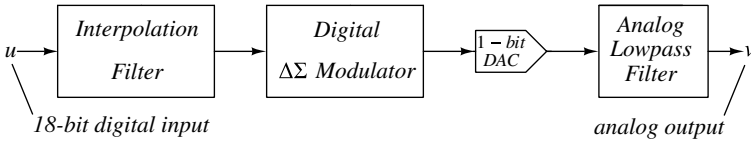
Figure 1.22   A ΔΣ DAC system.

with (say) 18-bit word length can be changed into a high-speed single-bit digital signal such that the baseband spectrum is preserved. The large amount of truncation noise generated in the loop is shaped in order to make the in-band noise negligible. The single-bit digital output signal can then be converted with high (ideally, perfect) linearity into an analog signal using a simple two-level DAC circuit. The out-of-band truncation noise can be subsequently removed using analog lowpass filters.

As in the case of analog ΔΣ loops, using single-bit truncation can lead to instability, and hence limits the effectiveness of the noise shaping. Using multi-bit (typically, 2–5 bit) truncation improves the noise shaping and makes the task of the analog post filter much easier. The linearity of the DAC for in-band signals can be achieved by using the same mismatch shaping techniques used in the internal DACs of analog multi-bit ΔΣ ADCs.

Also as in the case of ADCs, bandpass ΔΣ DACs can be designed. Noise-shaping now suppresses the truncation noise in a narrow band located around a nonzero center frequency f_0, which need not be much smaller than the clock frequency f_s.

ΔΣ DACs will be discussed in detail in Chapter 13.

1.10   Decimation and Interpolation

The digital filter that follows the ΔΣ ADC has the crucial function of eliminating shaped noise. Assuming an ideal brick-wall filter, the filtered output of the modulator has a small bandwidth in relation to the sampling rate. This allows the output of the digital filter to be downsampled to yield an output sequence at the Nyquist rate. A brick-wall characteristic can only be approximated in practice, which means that the filter characteristic should be very sharp to prevent degradation of the in-band SQNR due to aliasing that occurs when samples are dropped. The combination of digital filtering and sample-dropping is performed by a *decimation filter*. In the same vein, a filter with similar requirements (called the interpolation filter) occurs in the beginning of a ΔΣ DAC signal chain.

Design considerations for decimation and interpolation filters are given in Chapter 14.

1.11   Specifications and Figures of Merit

The primary specifications of any ADC include its power consumption P, signal bandwidth (*BW*), and effective number of bits (*ENOB*). Clearly, there are combinations of *ENOB*, *BW* and *P* specifications that are hard, or even impossible, to satisfy, and others that are relatively easy. To quantify the degree of difficulty, it is common to compute a figure of merit (FoM) that reflects the power efficiency of the ADC. There are two commonly used

FoMs. The Walden FoM [1] is defined as

$$FoM_W = \frac{P}{2^{ENOB} \cdot f_N}.$$ (1.21)

Here P is the power needed by the ADC, $ENOB$ is the effective number of bits, and f_N is the Nyquist frequency. The dimension of FoM_W is Joules, and it gives the amount of energy needed for each conversion step (LSB step). Note that a smaller FoM_W indicates a more efficient ADC.

An alternative, proposed originally by Rabii and Wooley [2] and presented in a modified form in the first edition of this book, has been dubbed the Schreier FoM. It is defined as

$$FoM_S = \frac{DR \cdot BW}{P}$$ (1.22)

or

$$FoM_S \,(dB) = DR \,(dB) + 10 \cdot \log_{10}\left(\frac{BW}{P}\right),$$ (1.23)

where DR denotes the dynamic range of the ADC. A larger FoM_S indicates a more efficient ADC.

The motivation for the definition of the FoM_S is given next. We assume that the technology determines the full-scale signal power, and hence, the DR will be determined by the variable in-band noise power q_{rms}^2. We also assume that the noise is white, so q_{rms}^2 is proportional to the signal bandwidth BW, and hence, DR is proportional to $1/BW$. As a consequence, for a given ADC with given P, the product $DR \cdot BW$ is a constant.
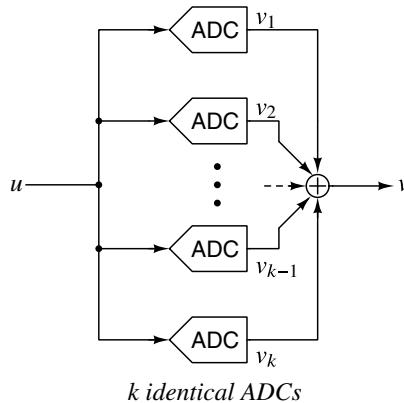
k identical ADCs

Figure 1.23   A multi-path $\Delta\Sigma$ ADC system.

For a fixed BW, the necessary power P is proportional to the required DR. To show this, assume that the ADC is realized in the multi-path configuration as shown in Figure 1.23. The component ADCs are identical, and ADC_i generates an output signal $v_i[n]$ and a noise output $q_i[n]$. The signals are fully correlated, and hence the output signal power will be $k^2 \cdot v_{rms}^2$. The noise outputs are uncorrelated, and hence the total noise output power will be $k \cdot q_{rms}^2$. The overall dynamic range of the ADC will hence be k times the DR of the individual component ADCs. If each ADC needs a power P, the total power used will be $k \cdot P$. Since both the DR and P are proportional to k, they must also be proportional to

each other. Note that this result agrees with the observation that the thermal noise power can be decreased by reducing the impedance level of the circuitry. Thus, increasing all C, g_m and $1/R$ values by a factor $k > 1$ will change DR into $k \cdot DR$, but it will increase all currents, and thus for fixed bias voltages P also, by the same factor k.

For a fixed DR, the power P needed is proportional to the required BW. To show this, assume initially that the power is kept constant, and the BW is increased by a factor l. Since, with fixed P, the product $DR \cdot BW$ is constant, the new dynamic range is DR/l. To restore DR to its original value, by the argument of the preceding paragraph, P must be replaced by $l \cdot P$. (An alternative derivation of the proportionality between BW and P can also be based on the structure of Figure 1.23, assuming that each ADC has a BW/l wide sub-band of the overall band.)

Finally, consider what happens if both DR and BW are changed. If DR is replaced by $k \cdot DR$, and BW by $l \cdot BW$, by the arguments presented above, the power P needs to be changed to $k \cdot l \cdot P$. This shows that $DR \cdot BW/P$ is a characteristic constant of a given converter, and hence can be used to compare the efficiencies of competing configurations and circuits.

The FoMs introduced above can be used for evaluating the energy efficiency of ADCs, but they do not tell the whole story about their practical usefulness. The key missing parameters are the cost and system impact of the chosen architecture. The cost aspects include the technology needed for their implementation, the silicon area occupied by the ADC, the number of pins required on the package, the production tests needed, and the fabrication yield. The system aspects include the prefiltering (anti-aliasing) needed, and the out-of-band signal transmission of the ADC. The latter determine how effectively the ADC can suppress large unwanted out-of-band "blockers" present in its input signal. For a meaningful comparison of the available ADC algorithms and configurations, these properties also need to be taken into account!

1.12   Early History, Performance, and Architectural Trends

The way $\Delta\Sigma$ modulation is developed in this book is not how it historically came about. The origins can be traced to Δ-modulation, which was a technique intended to be used to encode speech into digital form, so as to enable electronic switching in telephony. The prevalent technique at the time was to digitize speech at its Nyquist rate and quantize it to the required resolution of 8 bits. Since building an 8-bit ADC was a challenge, people began to wonder if using oversampling (so as to induce significant correlation between successive samples) could simplify the design of the quantizer.

The basic idea behind Δ-modulation is the following. If the input signal to be digitized is slowly varying in relation to the sampling rate, successive samples are so similar that one might as well transmit only the quantized *difference* (Δ) between successive samples. This way, the dynamic range of the transmitted signal can be significantly smaller than that of the signal itself, reducing the number of levels needed in the quantizer. In a Δ modulator, the quantizer is the simplest possible, namely the two-level one.

A naive attempt at transmitting the quantized difference is shown in Figure 1.24(a). The transmitter puts out a two-level sequence $\pm\Delta$, whose sign is dependent on the sign of the input slope. Since v contains the first difference of u, the receiver should be an inte-

grator. Figure 1.24(b) compares the sinusoidal input u and its estimate $\hat{u}$ at the receiver's output, when $OSR = 512$. We see significant error between the two waveforms, since the low-frequency component of quantization noise is greatly amplified by the integrator in the receiver. Since v is a two-level sequence, it might be thought of as an ADC – however, as seen from Figure 1.24(b), its performance leaves a lot to be desired.
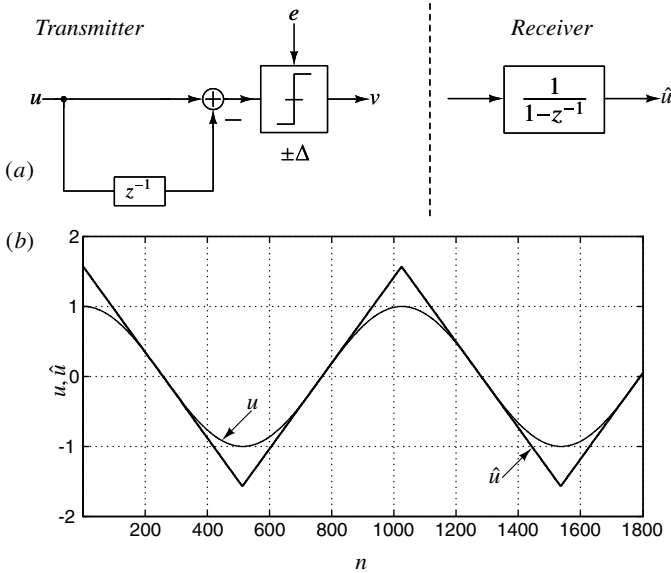
Figure 1.24   (a) A simple-minded attempt at quantizing the difference between successive samples of a highly oversampled signal. (b) Input and reconstructed waveforms. The reconstruction error has a large low-frequency component.

The Δ-modulator, which is a fundamental improvement over the idea discussed above, derives the delayed version of the input by integrating v, as shown in Figure 1.25(a). As a consequence, the quantization error e is in the feedback loop. It can be thought of as quantizing the difference between the input and its predicted value. In fact, the name Δ modulator is derived from the fact that the output is based on the difference (Δ) between a sample of the input and a predicted value of that sample. In the general case, the loop filter may be a higher-order circuit, which generates a more accurate prediction of the input sample $u[n]$, in order to subtract from the actual $u[n]$. This type of modulator is sometimes called a *predictive encoder.*

Referring to Figure 1.25,

$$v[n] = u[n] - u[n-1] + e[n] - e[n-1]. \tag{1.24}$$

We see that the quantization noise is also first-order shaped, and the reconstruction error at the receiver will therefore be much smaller. This is confirmed by the waveforms in Figure 1.25(b) – for the same input and step size, $\hat{u}$ is a much better approximation to u when compared to the system of Figure 1.24(a). Since v is a two-level sequence from which u can be reconstructed, it follows that the Δ-modulator can also be thought of as an ADC. It, however, has several disadvantages. The loop filter (integrator for the first-order loop shown) is in the feedback path, and hence, its nonidealities limit the achievable
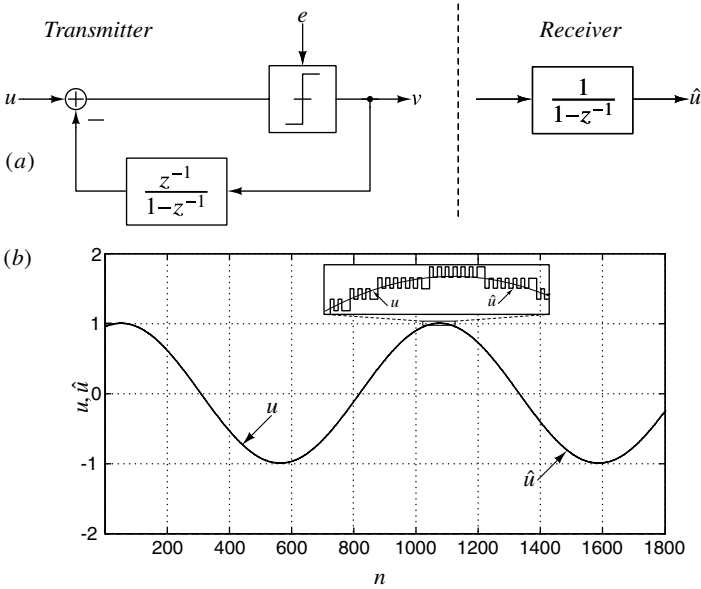
Figure 1.25   (a) The Δ-modulator. (b) Reconstruction error is greatly reduced due to noise-shaping of the quantizer's error.

linearity and accuracy. The integrator in the receiver has a high gain in the signal band, and hence, it will amplify the nonlinear distortion of the transmitted waveform as well as any noise picked up by the signal between the modulator and demodulator. It can also produce an arbitrary dc offset in the output. A delta-modulator, therefore, does not work reliably with a dc input.

The Δ-modulator of Figure 1.25 is also called an error-feedback structure. It was proposed in 1952 by de Jager [3], and in a different form by Cutler [4].

The $\Delta\Sigma$ modulator of Figure 1.13 is an alternative oversampling structure that avoids the shortcomings of the Δ-modulator. It is again a feedback loop, containing a loop filter as well as an internal low-resolution quantizer, but the loop filter is now in the forward path of the loop. As seen earlier, the $\Delta\Sigma$ modulator's output is given by

$$v[n] = u[n-1] + e[n] - e[n-1]. \tag{1.25}$$

Thus, the digital output contains a delayed, but otherwise unchanged, replica of the analog input signal u and a differentiated version of the quantization error e. Since the signal is not changed by the modulation process, the demodulation operation does not need an integrator as was the case for the delta modulator. Hence, the amplification of in-band noise and distortion at the receiver does not take place. Furthermore, the differentiation of the error e suppresses it at frequencies that are small compared to the sampling rate f_s. In general, if the loop filter has a high gain in the signal band, the in-band quantization "noise" is strongly attenuated, a process now commonly called noise shaping.

The $\Delta\Sigma$ modulator can be obtained from the Δ-modulator by cascading an integrator or summing block with the delta modulator. Hence, the structure of Figure 1.13 came to be called a sigma-delta ($\Sigma\Delta$) modulator. Alternatively, one can observe the differencing at

the input, followed by the summation in the loop filter, and hence call the structure a delta-sigma ($\Delta\Sigma$) modulator. Other systems with higher-order loop filters, multi-bit quantizers and the like are most properly called noise-shaping modulators, but it is common to extend the term $\Delta\Sigma$ modulator (or $\Sigma\Delta$ modulator) to these systems as well.

Although the basic idea of using feedback to improve the accuracy of data conversion has been around for about 50 years, the concept of noise-shaping was probably first proposed (along with the name delta-sigma modulation) in 1962 by Inose et al. [5]. They described a system containing a continuous-time integrator as the loop filter, and a Schmitt trigger as the ADC, that achieved (nearly) 40 dB SNR and had a signal bandwidth of about 5 kHz. Since the trade-off between analog accuracy and higher speed plus additional digital hardware was not particularly attractive at the time, further research on this topic was relatively sparse for a while.

Twelve years later, Ritchie proposed the use of higher-order loop filters [6]. Useful theory, as well as analysis and design techniques were developed by Candy and his collaborators at Bell Laboratories [7, 8, 9, 10, 11]. Candy and Huynh also proposed the MASH concept for the digital modulators used in $\Delta\Sigma$ DACs [12]. In 1986, Adams described an 18-bit $\Delta\Sigma$ ADC that used a third-order continuous-time loop filter, and a 4-bit quantizer with trimmed resistors performing as the DAC [13]. The MASH configuration was first applied in $\Delta\Sigma$ ADCs by Hayashi et al. [14] in 1986.

Using a multi-bit internal quantizer in a $\Delta\Sigma$ loop with digital linearity correction was proposed by Larson et al. [15] in 1988; the use of dynamic matching (randomization) was also introduced for the internal DAC of a $\Delta\Sigma$ ADC by Carley and Kenney in 1988 [16]. Various mismatch-shaping algorithms were suggested subsequently by Leung and Sutarja [17], Story [18], Redman-White and Bourner [19], Jackson [20], Adams and Kwan [21], Baird and Fiez [22], Schreier and Zhang [23], and Galton [24].

Bandpass $\Delta\Sigma$ modulators were motivated for their potential applications in wireless communications, and emerged in the late 1980s [25, 26, 27].

Current design trends in $\Delta\Sigma$ converters are aimed at extending the signal frequency range without any reduction in SNR. This will open up new applications in digital video, wireless and wired communications, radar and so on. Higher speed can often be achieved by using high-resolution (typically, 5-bit) internal quantizers, and a multistage (2- or 3-stage) MASH architecture. To correct for the nonlinearity of the internal DAC and for quantization noise leakage, digital correction algorithms have been proposed [28] for $\Delta\Sigma$ ADCs. A great deal of effort is also being applied to improving the performance of bandpass $\Delta\Sigma$ ADCs [29, 30, 31, 32, 33].

Over the last decade, there has been a significant wave of research, and commercial deployment of continuous-time $\Delta\Sigma$ ADCs. The benefits of such converters are many. As discussed earlier in this chapter, they feature the remarkable property of inherent anti-aliasing. It turns out that this makes them robust when they are part of a large digital chip with significant substrate noise. The input impedance of these ADCs is (usually) resistive, making them easy to drive. Reference generation circuitry is also generally simpler to design when compared to the corresponding effort needed in the case of a Nyquist ADC.

Technological trends (finer line widths, accompanied with lower breakdown voltages) stimulated research into $\Delta\Sigma$ modulators needing only low supply voltages [34]. Also applications opening up in portable devices motivated the development of low-power design techniques for $\Delta\Sigma$ data converters. Finally, applications in the instrumentation

and measurements area, including biomedical sensor interfaces motivated the development of low-frequency and very-high-accuracy ADCs, often realized by periodically reset $\Delta\Sigma$ modulators (which, as we discussed earlier, are called *incremental data converters*).

As noise-shaping theory and practice continue to mature, $\Delta\Sigma$ data converters can be expected to expand their range of application further.

References

[1] R. H. Walden, "Analog-to-digital converter survey and analysis," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 4, pp. 539–550, 1999.

[2] S. Rabii and B. A. Wooley, "A 1.8-V digital-audio sigma-delta modulator in 0.8-μm CMOS," *IEEE Journal of Solid-State Circuits*, vol. 32, no. 6, pp. 783–796, 1997.

[3] F. De Jager, "Delta modulation: A method of PCM transmission using the one unit code," *Phillips Research Reports*, vol. 7, pp. 542–546, 1952.

[4] C. C. Cutler, "Transmission systems employing quantization," Mar. 8, 1960. US Patent 2,927,962.

[5] H. Inose, Y. Yasuda, and J. Murakami, "A telemetering system by code modulation-$\Delta\Sigma$ modulation," *IRE Transactions on Space Electronics and Telemetry*, vol. 3, no. SET-8, pp. 204–209, 1962.

[6] G. Ritchie, J. C. Candy, and W. H. Ninke, "Interpolative digital-to-analog converters," *IEEE Transactions on Communications*, vol. 22, no. 11, pp. 1797–1806, 1974.

[7] J. C. Candy, "A use of limit cycle oscillations to obtain robust analog-to-digital converters," *IEEE Transactions on Communications*, vol. 22, no. 3, pp. 298–305, 1974.

[8] J. C. Candy and O. J. Benjamin, "The structure of quantization noise from sigma-delta modulation," *IEEE Transactions on Communications*, vol. 29, no. 9, pp. 1316–1323, 1981.

[9] J. C. Candy, "A use of double integration in sigma delta modulation," *IEEE Transactions on Communications*, vol. 33, no. 3, pp. 249–258, 1985.

[10] J. C. Candy, B. A. Wooley, and O. J. Benjamin, "A voiceband codec with digital filtering," *IEEE Transactions on Communications*, vol. 29, no. 6, pp. 815–830, 1981.

[11] J. C. Candy, "Decimation for sigma delta modulation," *IEEE Transactions on Communications*, vol. 34, no. 1, pp. 72–76, 1986.

[12] J. C. Candy and A.-N. Huynh, "Double interpolation for digital-to-analog conversion," *IEEE Transactions on Communications*, vol. 34, no. 1, pp. 77–81, 1986.

[13] R. W. Adams, "Design and implementation of an audio 18-bit analog-to-digital converter using oversampling techniques," *Journal of the Audio Engineering Society*, vol. 34, no. 3, pp. 153–166, 1986.

[14] T. Hayashi, Y. Inabe, K. Uchimura, and T. Kimura, "A multistage delta-sigma modulator without double integration loop," in *Digest of Technical Papers, IEEE International Solid-State Circuits Conference*, vol. 29, pp. 182–183, IEEE, 1986.

[15] L. E. Larson, T. Cataltepe, and G. C. Temes, "Multibit oversampled Σ-Δ A/D convertor with digital error correction," *Electronics Letters*, vol. 24, no. 16, pp. 1051–1052, 1988.

[16] L. R. Carley and J. Kenney, "A 16-bit 4th order noise-shaping D/A converter," in *Proceedings of the IEEE Custom Integrated Circuits Conference*, pp. 21–7, IEEE, 1988.

[17] B. H. Leung and S. Sutarja, "Multibit Σ-Δ A/D converter incorporating a novel class of dynamic element matching techniques," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 39, no. 1, pp. 35–51, 1992.

[18] M. J. Story, "Digital to analogue converter adapted to select input sources based on a preselected algorithm once per cycle of a sampling signal," Aug. 11, 1992. US Patent 5,138,317.

[19] W. Redman-White and D. Bourner, "Improved dynamic linearity in multi-level Σ-Δ converters by spectral dispersion of D/A distortion products," in *European Conference on Circuit Theory and Design*, pp. 205–208, IET, 1989.

[20] H. S. Jackson, "Circuit and method for cancelling nonlinearity error associated with component value mismatches in a data converter," June 22, 1993. US Patent 5,221,926.

[21] R. W. Adams and T. W. Kwan, "Data-directed scrambler for multi-bit noise shaping D/A converters," Apr. 4, 1995. US Patent 5,404,142.

[22] R. T. Baird and T. S. Fiez, "Linearity enhancement of multibit ΔΣ A/D and D/A converters using data weighted averaging," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 42, no. 12, pp. 753–762, 1995.

[23] R. Schreier and B. Zhang, "Noise-shaped multbit D/A convertor employing unit elements," *Electronics Letters*, vol. 31, no. 20, pp. 1712–1713, 1995.

[24] I. Galton, "Noise-shaping D/A converters for ΔΣ modulation," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, vol. 1, pp. 441–444, IEEE, 1996.

[25] T. Pearce and A. Baker, "Analogue to digital conversion requirements for HF radio receivers," in *Proceedings of the IEE Colloquium on System Aspects and Applications of ADCs for Radar, Sonar, and Communications*, 1987.

[26] P. H. Gailus, W. J. Turney, and F. R. Yester Jr, "Method and arrangement for a sigma delta converter for bandpass signals," Aug. 15, 1989. US Patent 4,857,928.

[27] R. Schreier and M. Snelgrove, "Bandpass sigma-delta modulation," *Electronics Letters*, vol. 25, no. 23, pp. 1560–1561, 1989.

[28] X. Wang, U. Moon, M. Liu, and G. C. Temes, "Digital correlation technique for the estimation and correction of DAC errors in multibit MASH ΔΣ ADCs," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, vol. 4, pp. IV–691, IEEE, 2002.

[29] W. Gao and W. M. Snelgrove, "A 950 MHz second-order integrated LC bandpass sigma–delta modulators," in *Digest of Technical Papers, IEEE Symposium on VLSI Circuits*, 1997.

[30] G. Raghavan, J. Jensen, J. Laskowski, M. Kardos, M. G. Case, M. Sokolich, and S. Thomas III, "Architecture, design, and test of continuous-time tunable intermediate-frequency bandpass delta-sigma modulators," *IEEE Journal of Solid-State Circuits*, vol. 36, no. 1, pp. 5–13, 2001.

[31] P. Cusinato, D. Tonietto, F. Stefani, and A. Baschirotto, "A 3.3-V CMOS 10.7-MHz sixth-order bandpass ΣΔ modulator with 74-dB dynamic range," *IEEE Journal of Solid-State Circuits*, vol. 36, no. 4, pp. 629–638, 2001.

[32] R. Schreier, J. Lloyd, L. Singer, D. Paterson, M. Timko, M. Hensley, G. Patterson, K. Behel, J. Zhou, and W. J. Martin, "A 50 mW bandpass ΔΣ ADC with 333 kHz BW and 90 dB DR," in *Digest of Technical Papers, IEEE International Solid-State Circuits Conference*, vol. 1, pp. 216–217, IEEE, 2002.

[33] H. Shibata, R. Schreier, W. Yang, A. Shaikh, D. Paterson, T. C. Caldwell, D. Alldred, and P. W. Lai, "A dc-to-1 GHz tunable RF ΔΣ ADC achieving DR=74 dB and BW=150 MHz at $f_0 = 450$ MHz using 550 mW," *IEEE Journal of Solid-State Circuits*, vol. 47, pp. 2888–2897, Dec 2012.

[34] M. Keskin, U.-K. Moon, and G. C. Temes, "A 1-V 10-MHz clock-rate 13-bit CMOS ΔΣ modulator using unity-gain-reset op amps," *IEEE Journal of Solid-State Circuits*, vol. 37, no. 7, pp. 817–824, 2002.

CHAPTER 2

SAMPLING, OVERSAMPLING, AND NOISE-SHAPING

Analog-to-digital converters (ADCs) and digital-to-analog converters (DACs) play a crucial role in the signal conditioning needed to interface the real world, whose signals are continuous both in time and amplitude, to the virtual world, where they are represented as quantities discrete in time and amplitude.
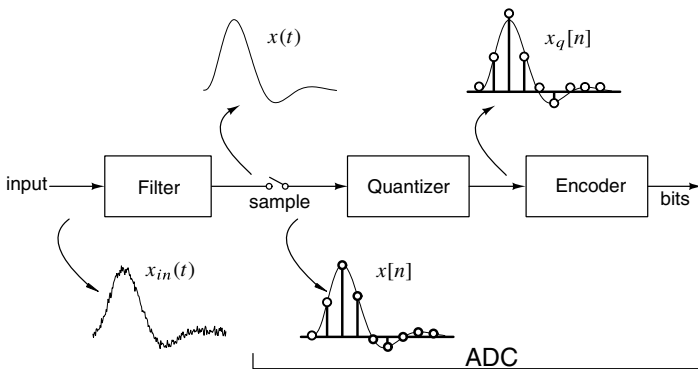
Figure 2.1   Representative signal chain of an analog front end.

The block diagram of the signal chain of a typical analog front end is shown in Figure 2.1. The input signal $x_{in}(t)$ has a bandwidth of B Hz. In principle, it can be perfectly

reconstructed from its samples provided that the sampling rate f_s is at least $2B$. In practice, the input is usually corrupted by noise, which can have frequency components outside the frequency range $[0, B]$. Thus, $x_{in}(t)$ must be filtered by an *anti-alias* filter before sampling. The filter eliminates out of band noise, which would otherwise alias into the signal band after sampling, and degrade the quality of the samples of $x_{in}(t)$.

The samples of the filter output, $x[n] = x(nT_s)$ (where $T_s = 1/f_s$) are then quantized in amplitude to yield $x_q[n]$, which has a discrete number of levels. A digital code can be assigned to each of the levels – thereby yielding a *digital signal* – one that is discrete both in time (due to sampling) and amplitude (due to quantization).

2.1   A Review of Sampling

Since $x[n]$ is a result of sampling $x(t)$, the Fourier transform of $x[n]$ should be related to that of $x(t)$. To determine this relationship, we proceed as follows. We first form a continuous-time signal $x_s(t)$ by multiplying $x(t)$ with the Dirac delta train $\sum_n \delta(t - nT_s)$ (which is periodic with T_s). Thus,

$$x_s(t) = x(t) \sum_{n=-\infty}^{\infty} \delta(t - nT_s) = \sum_{n=-\infty}^{\infty} \underbrace{x(nT_s)}_{\equiv x[n]} \delta(t - nT_s). \tag{2.1}$$

One way of obtaining the Fourier transform of $x_s(t)$ is to convolve the transform of $x(t)$ (denoted by $X_c(f)$) with that of the Delta train [1].

Recalling that

$$\mathcal{F}\{ \sum_{k=-\infty}^{\infty} \delta(t - kT_s)\} = \frac{1}{T_s} \sum_{n=-\infty}^{\infty} \delta (f - nf_s), \tag{2.2}$$

where $f_s = 1/T_s$, we obtain

$$\mathcal{F}\{x_s(t)\} = X_s(f) = \frac{1}{T_s} \sum_{n=-\infty}^{\infty} X_c (f - nf_s). \tag{2.3}$$

Applying the Fourier transform to both sides of (2.1), but term by term on the right-hand side this time, we can also express $X_s(f)$ as

$$X_s(f) = \sum_{n=-\infty}^{\infty} x[n]e^{-j2\pi f T_s n}. \tag{2.4}$$

From (2.3) and (2.4), we have

$$\sum_{n=-\infty}^{\infty} x[n]e^{-j2\pi f T_s n} = \frac{1}{T_s} \sum_{n=-\infty}^{\infty} X_c (f - nf_s). \tag{2.5}$$

The discrete-time Fourier transform of the sequence $x[n]$, denoted by $X_d(e^{j\omega})$, can then be obtained by replacing $2\pi f T_s$ with ω in the relation above. This yields

$$X_d(e^{j\omega}) = \mathcal{F}\{x[n]\} = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n} = f_s \sum_{n=-\infty}^{\infty} X_c\left(\frac{f_s\omega}{2\pi} - nf_s\right). \qquad (2.6)$$

Clearly, $X_d(e^{j\omega})$ is periodic with period 2π.

Given $X_c(f)$, the discrete-time Fourier transform $X_d(e^{j\omega})$ of $x[n]$ can be obtained by the following process, as illustrated in Figure 2.2.
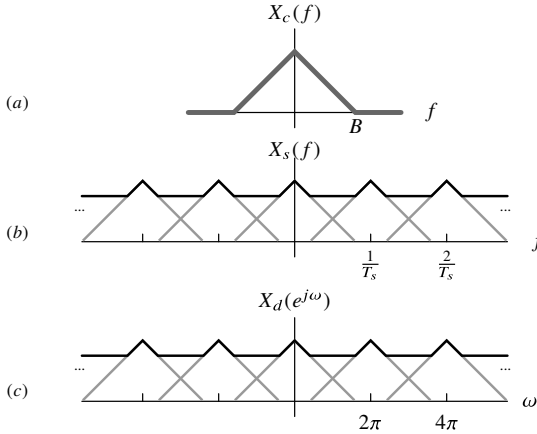
Figure 2.2   Fourier transforms of (a) $x(t)$, (b) $x_s(t)$, and (c) $x[n]$.

- Form copies of $X_c(f)$, shifted by integer multiples of f_s.

- Add these copies and scale the result by f_s (Figure 2.2(b)).

- Scale the frequency axis so that f_s corresponds to 2π and label the axis with ω (Figure 2.2(c)).

From the figure above, it is apparent that choosing $f_s < 2B$ causes the shifted copies of $X_c(f)$ to overlap, resulting in *aliasing*. The minimum sampling frequency that prevents aliasing is, therefore, $f_{s,min} = 2B$, which is referred to as the *Nyquist rate*.

A continuous-time tone at f Hz, after sampling, appears as a discrete-time tone at $\omega = 2\pi f T_s$. Due to the periodicity of $X_d(e^{j\omega})$, two continuous-time tones with frequencies of the form f and $\pm f + m f_s$, where m is an integer, cannot be distinguished from each other after being sampled at f_s.

It is natural to wonder what benefits (if any) accrue if a signal is sampled at a rate higher than $2B$. A number that compares the actual sampling rate f_s to the Nyquist rate $2B$ is called the *oversampling ratio* (OSR), and is defined as

$$OSR \equiv \frac{f_s}{2B}. \qquad (2.7)$$

Figure 2.3 shows the benefit to be had, with regard to the anti-alias filter, by using a larger
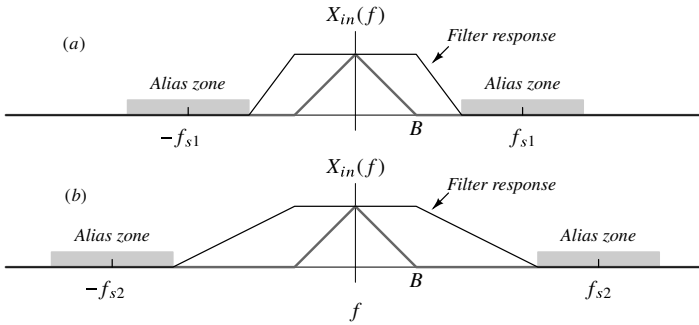
Figure 2.3   Constraints on the response of the anti-alias filter, for two sampling rates.

OSR. Without the filter, signals in the frequency range $[f_s - B, f_s + B]$ will alias into the desired signal band $|\omega| < 2\pi(B/f_s)$ after sampling. The filter, therefore, should attenuate all signals in the alias zones. The filter's magnitude response should be 1 for $|f| < B$ and 0 for $|f - f_s| < B$. By increasing the OSR, it is seen that the transition band of the filter is wider, thereby relaxing the design of the filter. With very high OSRs, anti-alias filtering can be trivial.

Apart from a relaxed anti-alias filter, a high OSR also plays a crucial role in reducing in-band quantization noise, as we will see in the rest of this book. The operation following anti-alias filtering and sampling is quantization, as shown in Figure 2.1.

2.2 Quantization

Quantization is a nonlinear memoryless operation, whose symbol is shown in Figure 2.4(a). A convention that we follow throughout this book is to denote a quantizer's input by y, and its output by v.[1] The transfer curve is a staircase that is usually uniform, so that any two adjacent output levels differ by a fixed spacing Δ. The staircase straddles a straight line with slope k for a range of inputs[2] and saturates thereafter.

In practice, quantizers are implemented with *bipolar inputs*, with the transfer curve being an odd function of y. Depending on the number of steps, denoted by M, two types of transfer curves are possible, as shown in Figure 2.4(b) and (c). The first, where $y = 0$ coincides with a step (rise) of v, is called a mid-rise characteristic. In the second case, $y = 0$ occurs in the middle of a flat portion (tread) of the curve, so the quantizer is called a mid-tread quantizer. Unless otherwise noted, the quantizers considered in this text will be symmetric bipolar quantizers with $\Delta = 2$. This common value for Δ allows the quantization levels of both quantizer types to be integer values: odd integers for the mid-rise quantizer and even integers for the mid-tread quantizer.   The difference $e = v - y$ is is called the quantization error, or (not entirely correctly) the quantization noise.

The transfer curve that relates the error to the input is shown in Figure 2.4(c). It can be seen from the figure that as long as y is between $-(M + 1)$ and $(M + 1)$, the error

---

[1]A mnemonic for the notation that y is the quantizer input and v is its output is to think of the quantizer as losing information and to picture the letter v as being the letter y minus its tail.
[2]For mathematical convenience, we typically use $k = 1$.
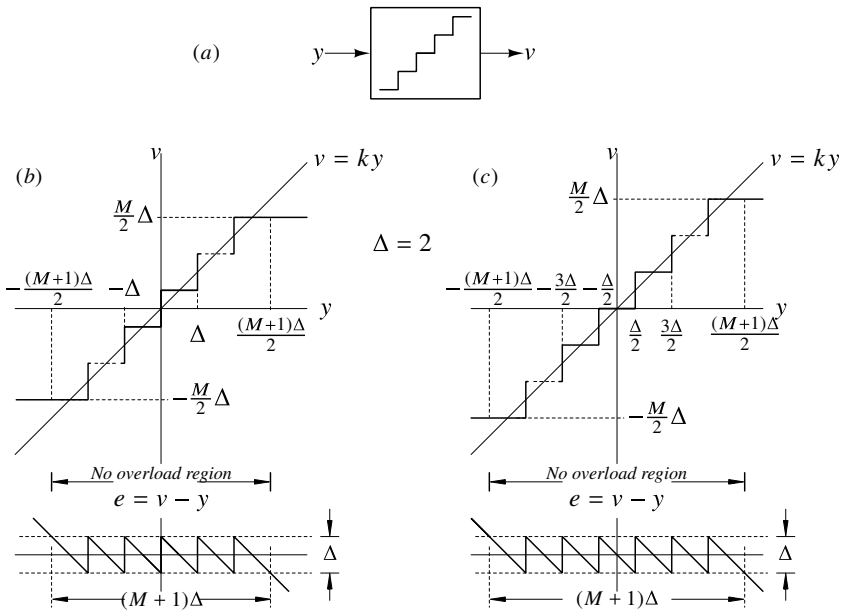
Figure 2.4   (a) Quantizer symbol. Transfer and error curves for a symmetric bipolar M-step (b) mid-rise and (c) mid-tread quantizer, with $\Delta = 2$ and $k = 1$.

e lies between -1 and 1. We call the range of y where this condition is satisfied, as the no-overload input range, or simply the input range. The difference between the lowest and highest levels is called the full-scale (FS) of the quantizer. Table 2.1 summarizes these and other properties of the quantizers of Figure 2.4(b) and (c).

Parameter	Value
Input step size (LSB size)	2
Number of steps	M
Number of levels	$M + 1$
Number of bits	$\lceil \log_2(M + 1) \rceil$
No-overload input range	$[-(M + 1), (M + 1)]$
Full-scale	$2M$
Input thresholds	$0, \pm 2, \cdots, \pm(M - 1), \quad M$ odd (mid-rise)
	$\pm 1, \pm 3, \cdots, \pm(M - 1), \quad M$ even (mid-tread)
Output levels	$\pm 1, \pm 3, \cdots, \pm M, \quad M$ odd (mid-rise)
	$0, \pm 2, \pm 4, \cdots, \pm M, \quad M$ even (mid-tread)

Table 2.1   Properties of the symmetric quantizers of Figures 2.4(b) and (c), with $\Delta = 2$.

The ideal quantizer is a deterministic device; v, and hence also the error e, are fully determined by the input y. However, as shown in Figure 2.4, e is a "complicated" function of y. The difficulties associated with the strongly nonlinear nature of the quantizer have prompted engineers to make several assumptions regarding the nature of quantization error. Loosely speaking, these assumptions are permissible when y stays within the input range of the quantizer, and changes by sufficiently large amounts from sample to sample so

that its position within a quantization interval is essentially random (such a signal is also called a *busy signal*). A detailed (and more appropriate) discussion on quantization, various approximations for the properties of quantization error, and the assumptions behind them, are given in [2]. The simplifying assumptions are the following:

1. e is assumed to be an additive "noise" sequence.

2. e is assumed to be independent of y.

3. e is assumed to be uniformly distributed in $[-1, 1]$.

4. Having made three dubious assumptions about e, it does not hurt to make one more – namely, that e is a white sequence.

Even though none of the assumptions above are guaranteed to be true, it is amazing how far we can go with them. First, let us compute the mean and variance of e using assumption 3 above.

$$\bar{e} = \frac{1}{2} \int_{-1}^{1} e \, de = 0 \tag{2.8}$$

and

$$\overline{e^2} = \frac{1}{2} \int_{-1}^{1} e^2 \, de = \frac{1}{3}. \tag{2.9}$$

The implications of assumption 4 are shown in the frequency domain in Figure 2.5. The input is sampled at the Nyquist rate, and the spectrum of $y[n]$ extends from $[-\pi, \pi]$. After quantization, y is corrupted by e, whose (double-sided) spectral density is flat and equal to $\Delta^2/(24\pi)$.
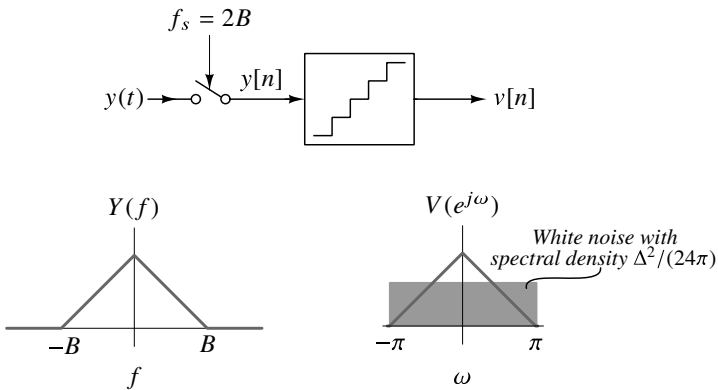
Figure 2.5   Spectral view of the additive quantization noise assumption. $y(t)$ is sampled at the Nyquist rate ($OSR = 1$).

To summarize, the quantizer's input and output are related by

$$v = y + e, \tag{2.10}$$

where e is assumed to be a uniformly distributed random variable when y is busy.

What is the SNR at the output of an N-bit (2^N-level) quantizer for a sinusoid exercising the full range? The peak-to-peak amplitude of such a sinusoid is $2^N\Delta$, resulting in a signal power of $2^{2N-3}\Delta^2$. The mean-square noise due to quantization is $\Delta^2/12$ (assuming that the error is uniformly distributed). The peak SQNR (in dB), therefore, is given by $10\log(2^{2N}(12/8)) = 6.02N + 1.76\,\text{dB}$.

It is important to note that (2.10) is *always valid*; an approximation is made only when e is assumed to have specified properties such as a uniform distribution or a white spectrum. Again, such approximations are only justifiable under the conditions stated earlier.

It is easy to conceive of inputs for which these conditions are not satisfied, and hence for which the approximation gives wildly wrong results. A constant y, or a periodic y with a frequency harmonically related to f_s are examples that immediately come to mind.
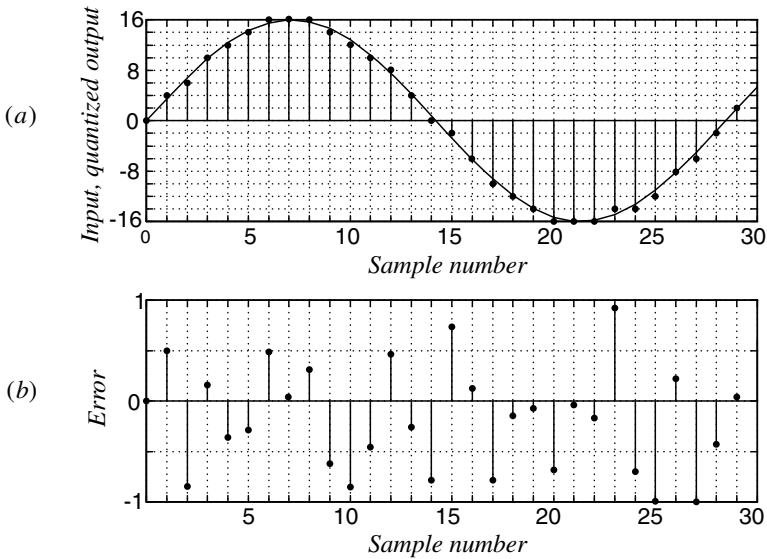
Figure 2.6   (a) A sine wave quantized by a symmetrical 16-step quantizer, and (b) the corresponding quantization error sequence.

To illustrate these points, Figure 2.6 shows a full-scale sine wave sampled and then quantized by a 16-step symmetrical bipolar quantizer with $\Delta = 2$. The frequency f of the input is moderately low compared to f_s and does not have a simple harmonic relation to f_s. As a result, the quantization error sequence appears fairly random, although a careful examination of the error sequence as the input passes through its peaks does reveal a non-zero correlation between adjacent samples. The mean-square value of the error in this example is found to be 0.30, which is close to the expected value of $\Delta^2/12 = 1/3$.

A fast Fourier transform (FFT) of v is plotted in Figure 2.7. The spectrum consists of one large spike representing the input sine wave, plus many smaller spikes distributed evenly along the frequency axis, representing the frequency components of the quantization error. Judging by these results, the white-noise approximation appears to be reasonable in this situation.
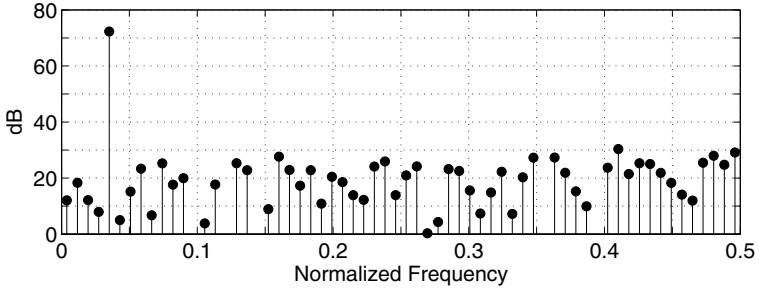
Figure 2.7   A 256-point FFT of the quantized sine wave of Figure 2.6.
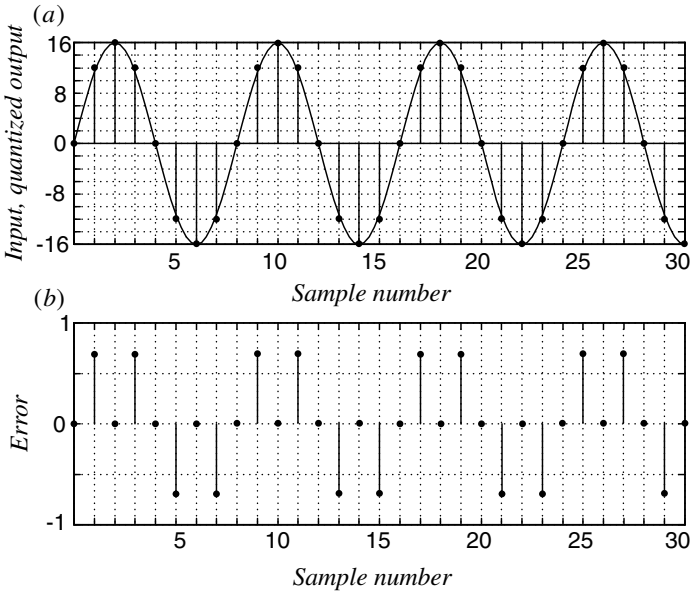
Figure 2.8   (a) A $f = f_s/8$ sine wave, sampled at f_s, and quantized by a symmetrical 16-step quantizer, and (b) the corresponding quantization error sequence.
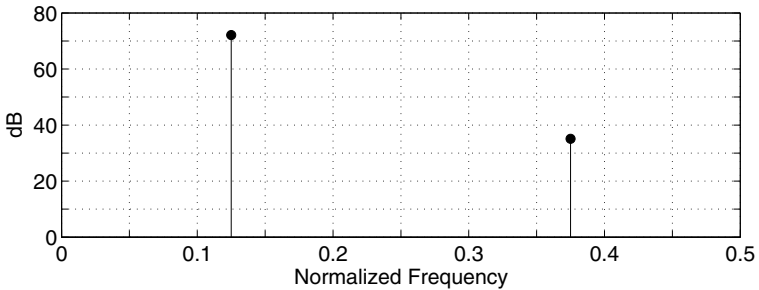
Figure 2.9   A 256-point FFT of the quantized sine wave of Figure 2.8.

Now consider what happens when the full-scale sine wave with frequency $f = f_s/8$ shown in Figure 2.8 is quantized by the same 16-step quantizer. The quantization error, shown in the lower part of the figure, is now periodic, and since it assumes only three values, its distribution is far from uniform. The mean-square value of this error sequence is found to be 0.23, or only about 70% of the expected value. The FFT shown in Figure 2.9 represents an even more serious departure from our normal assumptions in that the spectrum now consists of only two spikes! The quantization noise energy is fully concentrated in one tone at $f_s/8$ (coincident with the signal itself) and a second tone at $3f_s/8$ (the third harmonic of the signal).

2.2.1 Quantizer Modeling

As illustrated in Fig 2.10, the quantizer's transfer curve straddles a straight line with slope $k = 1$. We therefore modeled the quantizer (assuming it is not overloaded, and that y is busy, so that e is uniform) as a system whose gain k is *unity*, and whose input is corrupted by quantization noise. It seemed natural to use $k = 1$. But why is this justified? After all, one can draw many straight lines "within" the quantizer characteristic, as shown in Figure 2.10(a).
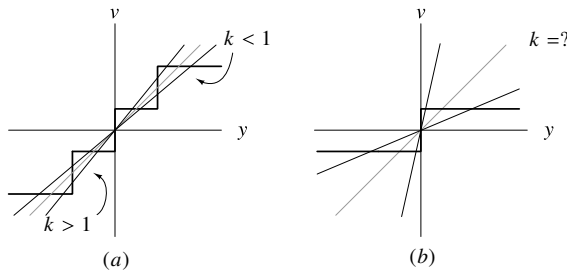
Figure 2.10   (a) Many straight lines "fit" the quantizer characteristic. (b) What is the best-fit line for a 2-level quantizer?

This question becomes more problematic as the number of quantizer steps is reduced. As seen in Figure 2.10(b), any number of "best-fit" lines can be drawn for a 2-level quantizer. All three lines in the figure result in the same maximum error of $\Delta/2$, although with different no-overload ranges. Clearly, a more systematic way of determining the gain is necessary.

A natural way of determining the gain of a quantizer is to ask the question – what is the slope k of the straight line that results in the smallest average squared error between the quantizer's output v and $k \cdot y$? In other words, we should attempt to minimize

$$\sigma_e^2 = \lim_{N \to \infty} \frac{1}{N} \sum_{n=0}^{N} e^2[n] = \lim_{N \to \infty} \frac{1}{N} \sum_{n=0}^{N} (v[n] - ky[n])^2. \tag{2.11}$$

To determine k, we first introduce the notation for an *inner product*, or a scalar product. For real sequences a and b, the inner product is defined as

$$\langle a, b \rangle = \lim_{N \to \infty} \left[\frac{1}{N} \sum_{n=0}^{N} a[n]b[n] \right] = E[ab]. \tag{2.12}$$

Since $e = v - ky$, the average power of e can be written as

$$\begin{aligned} \sigma_e^2 &= \langle e, e \rangle \\ &= \langle v - ky, v - ky \rangle \\ &= \langle v, v \rangle - 2k \langle v, y \rangle + k^2 \langle y, y \rangle. \end{aligned} \tag{2.13}$$

This is minimized for

$$k = \frac{\langle v, y \rangle}{\langle y, y \rangle}. \tag{2.14}$$
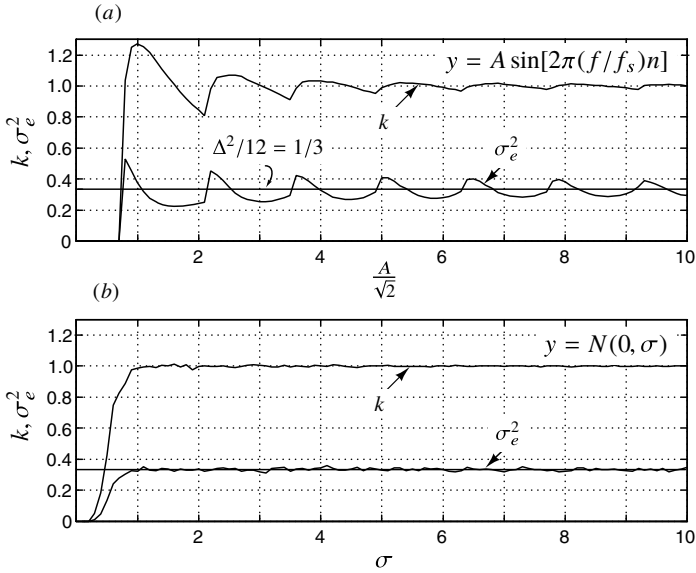
Figure 2.11   Computed gain and noise variance for a mid-tread quantizer ($\Delta = 2$) as a function of strength: (a) Sine-wave input and (b) Gaussian noise with variance σ^2.

Figure 2.11 shows the computed k and σ_e^2 for a mid-tread quantizer with $\Delta = 2$, for two kinds of inputs. A sinusoidal input with $f/f_s = 7/256$ is used to generate the results in part(a) of the figure. For large amplitudes (as long as the quantizer is not overloaded), the gain hovers around unity, and $\sigma_e^2 \approx \Delta^2/12$. k deviates from unity at small amplitudes, before becoming 0 for $A < 1$. This makes sense, given that the quantizer is of the mid-tread type.

The situation is somewhat different when y is a zero mean, white Gaussian sequence with a standard deviation denoted by σ. Thanks to the "busy" nature of y, k and σ_e^2 are virtually 1 and 1/3, respectively, for $\sigma > 1$. This makes sense, when we bear in mind that the excursions of a Gaussian distribution are about $\pm 3\sigma$ about the mean. So, even when $\sigma = 1$, v is a 4-level sequence.

2.2.2   Overloaded Quantizers

What should we expect to happen to k and σ_e^2 when the input is so large that the quantizer starts to become overloaded? Since v saturates when y exceeds the input range, k should reduce. Further, when the quantizer is overloaded, e exceeds $\Delta/2$; so we should expect the decrease in k to be accompanied by an increased σ_e^2.
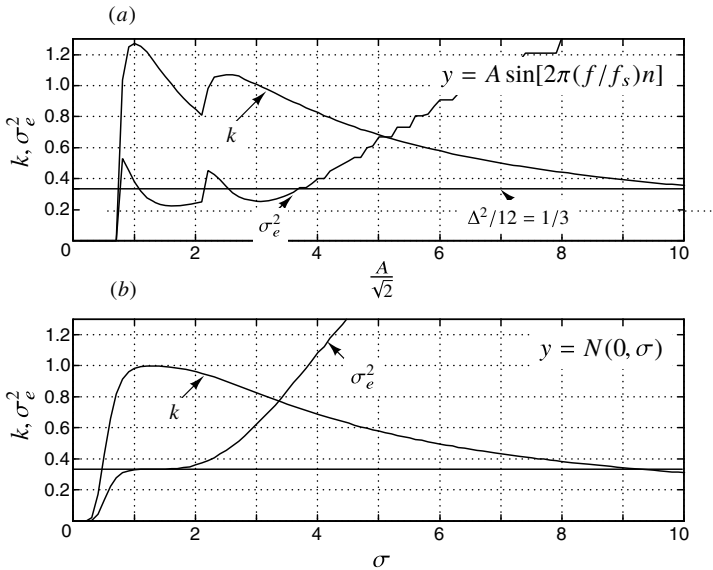
Figure 2.12   Computed gain and noise variance for a five-level quantizer ($\Delta = 2$) as a function of strength: (a) Sine-wave input and (b) Gaussian noise with variance σ^2.

This intuition is confirmed by the simulation results shown in Figure 2.12. The quantizer's input range is $[-5, 5]$. For the sine-wave input, k and σ_e^2 begin to deviate from the curves in Figure 2.11(a) at $A/\sqrt{2} = 3.5$. For the Gaussian input, we see that this occurs at $\sigma \approx 5/3 = 1.67$.

What is the gain for a 2-level (binary) quantizer? In this case, (2.14) simplifies to

$$k = \frac{\langle v, y \rangle}{\langle y, y \rangle} = \frac{E[|y|]}{E[y^2]}. \tag{2.15}$$

Clearly, the optimal value of k for the linear model of a binary quantizer is dependent on the statistics of its input y. As a sanity check, let k be the gain associated with an input y according to (2.15). If the input is modified to $\hat{y} = 10\,y$, then $E[|\hat{y}|] = 10\,E[|y|]$ and $E[\hat{y}^2] = 100\,(E[y^2])$, so that $\hat{k} = k/10$. Thus, the effective gain of a binary quantizer is reduced by a factor of 10 when its input is amplified by a factor of 10. This makes physical sense, since v remains the same when y is increased tenfold.

When a system containing a binary quantizer is replaced by its linear model, the estimate of the quantizer's gain k should be found from extensive numerical simulations. Otherwise, misleading results may be obtained from the linear model.

2.2.3 Quantizer Modeling with Two Inputs
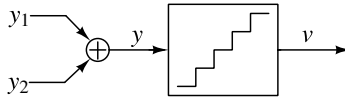
Figure 2.13 v is the quantized version of $y_1 + y_2$.

How does one model a quantizer with two inputs y_1 and y_2, as shown in Figure 2.13? We proceed in a manner similar to that in the single input case by writing

$$v = k_1 y_1 + k_2 y_2 + e \tag{2.16}$$

and determining those values k_1 and k_2 that minimize the error e between v and $k_1 y_1 + k_2 y_2$ in the mean-square sense. The best fit k_1 and k_2 (assuming $\langle y_1, y_2 \rangle = 0$) are given by

$$k_1 = \frac{\langle v, y_1 \rangle}{\langle y_1, y_1 \rangle}, \quad k_2 = \frac{\langle v, y_2 \rangle}{\langle y_2, y_2 \rangle}. \tag{2.17}$$
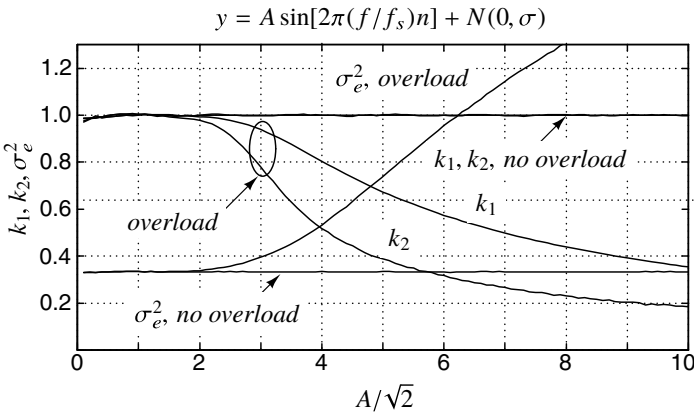
Figure 2.14 Computed gain and noise variance for quantizers (with $\Delta = 2$) for $y = A \sin[2\pi(f/f_s)n] + N(0, 1)$ as A is varied, without (∞ levels) and with overload (5 levels).

When the number of levels is very large, so that the quantizer does not saturate, and when y is busy, we can expect k_1 and k_2 to be both equal and close to 1. The mean-square error, by the same token, should be $\Delta^2/12$. This intuition is confirmed by the results in Figure 2.14, which shows k_1, k_2, and σ_e^2 for a quantizer with $\Delta = 2$. The input is given by

$$y = \underbrace{A \sin[2\pi(f/f_s)n]}_{y_1} + \underbrace{N(0, \sigma)}_{y_2}, \tag{2.18}$$

where y_1 is a sinusoid with $f/f_s = 7/256$, and y_2 is Gaussian noise with $\sigma = 1$. In the figure, $A/\sqrt{2}$ is swept from 0.1 to 10. We see that $k_1 \approx k_2 \approx 1$ and $\sigma_e^2 \approx (1/3)(= \Delta^2/12)$.

When the number of levels is reduced to 5, the quantizer begins to saturate – occasionally, when A is small, but more and more frequently for larger values of A. This causes
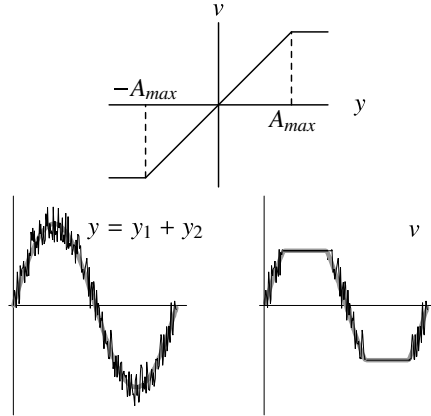
Figure 2.15   Sinusoid and noise passed through a saturating nonlinearity – the effective gain for the sine wave is higher than that for noise.

k_1 and k_2 to reduce, and σ_e^2 to increase. An estimate of A at which this begins to happen is $5 - 3\sigma = 2$. Interestingly, we see that k_1 and k_2 are *not the same* when the quantizer overloads. We also notice that $k_1 > k_2$. Why does this make sense?

Consider the sum of a sine wave ($y_1 = A\sin[2\pi(f/f_s)n]$) and noise ($y_2 = N(0, 1)$) exciting a saturating nonlinearity, as shown in Figure 2.15. Since $A > A_{max} \gg 1$, the output saturates for a portion of the period of the sine wave, during which time the "incremental gain" for the noise y_2 is zero. The average gain for y_2 should, therefore, be smaller than unity.

The average gain for y_1 is also smaller than unity, since the output, when saturated, is smaller than what it would have been without saturation. However, in contrast to y_2, which is completely attenuated, only a small portion of y_1 is "lost" due to saturation. Thus, the effective gain k_1 for the sinusoid, though smaller than 1, should be greater than the gain k_2 for noise.[3]

2.3   Quantization Noise Reduction by Oversampling

The quantizer's output, after encoding, is a digital signal that is used by the DSP for processing. The DSP is, therefore, using v as an approximate representation of y (to within e). Put another way, the DSP's estimate $\hat{y}[n]$ of $y[n]$ is $v[n]$. Thus, as we saw in the previous section, the mean-square value of the error between the estimate $\hat{y}[n]$ and $y[n]$ is $\Delta^2/12$.

Now, suppose that $v[n]$ is known to be the result of quantizing a slowly varying sequence $y[n]$, as shown in Figure 2.16. This corresponds to sampling y with a large oversampling ratio ($OSR \gg 1$). Can we estimate y from v with a mean-square error less than

[3] This phenomenon where the gain of one signal is affected by the other, when both of them are passed through a nonlinear device, is not altogether unfamiliar. This is similar to *desensitization* in RF amplifiers, where an amplifier's gain for a small desired signal is reduced in the presence of a much larger, undesired interfering signal. This occurs since the interferer periodically drives the compressive amplifier into the lower gain regions of its transfer curve, resulting in a smaller average gain for the desired signal.
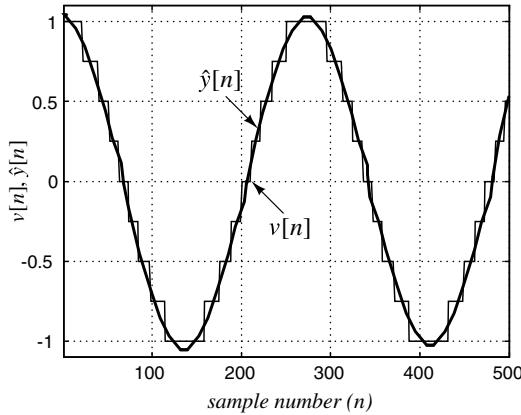
Figure 2.16   Estimating the quantizer input from its output sequence.

$\Delta^2/12$? In other words, can we make a better guess of what y is? This should indeed be possible – the intuition is the following.

Since the input is oversampled, the difference between successive samples should be small. v, on the other hand, exhibits step jumps; these *must* be arising when y crosses quantizer thresholds. A better estimate $\hat{y}$ can, therefore, be obtained by "smoothing" the sequence v, as shown in black in Figure 2.16. Mathematically, this is equivalent to filtering the digital sequence $v[n]$ with a *digital* filter, as shown in Figure 2.17(a).
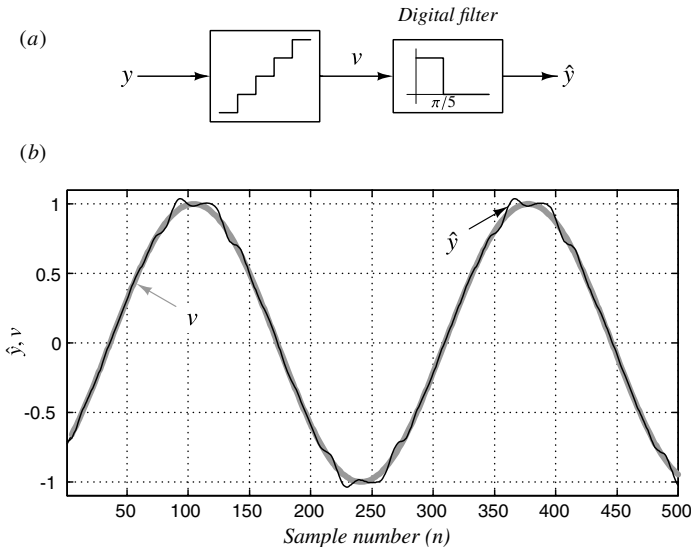
Figure 2.17   (a) Improved estimation of y by digitally filtering v. (b) Comparison of y and $\hat{y}$.

The results of the numerical experiments of Figures 2.16 and 2.17(b) illustrate the idea. A low-frequency sine wave y is quantized with a step size of $\Delta = 0.25$, yielding v. The mean-square difference between v and y is 4.6×10^{-3}, which is close to $\Delta^2/12 (=$

5.2×10^{-3}). To estimate y from v, the latter is filtered with a sharp digital lowpass filter with a cutoff frequency of $\pi/5$. When the filtered output $\hat{y}$ is compared with y, the error $\hat{y} - y$ is greatly reduced, as Figure 2.17(b) shows. $\overline{(\hat{y} - y)^2}$ is 7×10^{-5}, about 5.5 times smaller than without filtering. This makes sense – for quantization noise that is white, the digital filter should be expected to let through 20% of the quantization noise. In reality, the assumption of "whiteness" does not quite hold, and the mean-square error is somewhat lower.

The spectral picture of the system, which exploits oversampling to reduce quantization noise, is shown in Figure 2.18. The spectrum of y occupies the range $[-\pi/OSR, \pi/OSR]$, while quantization noise extends over the $[-\pi, \pi]$ range, with a (double-sided) spectral density $\Delta^2/(24\pi)$. The digital filter cuts off noise outside the signal band, thereby reducing the power of the quantization noise in $\hat{y}$ by a factor of OSR.
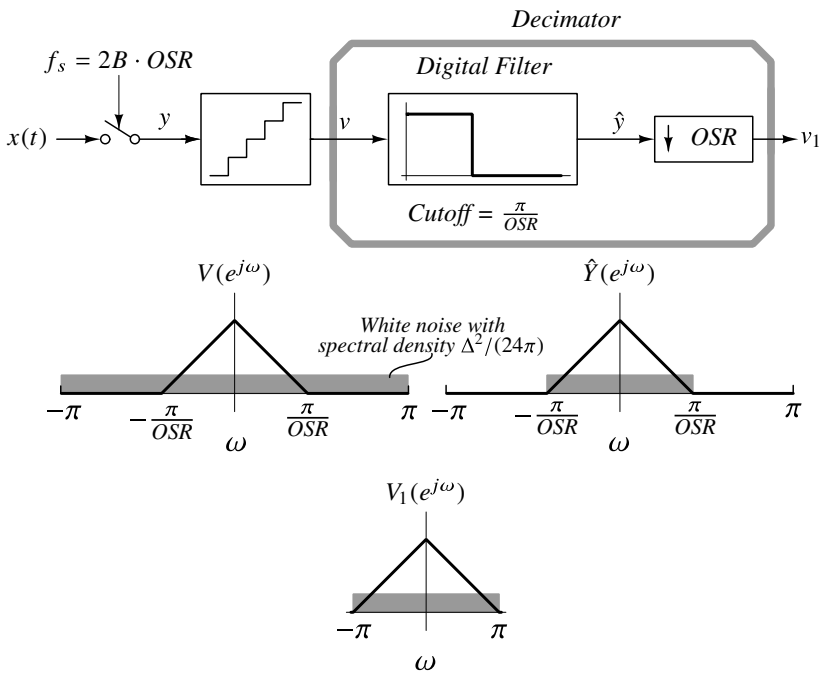
Figure 2.18   Signal chain that employs oversampling to reduce quantization noise, and spectra of v, $\hat{y}$, and v_1.

That is not all. Since $\hat{y}$ occupies the range $[-\pi/OSR, \pi/OSR]$, it can be downsampled by OSR, resulting in the sequence v_1, which is at the Nyquist rate. The combination of the digital filter and down sampler is called the *decimation filter*. However, when compared to using the same quantizer with Nyquist sampling, our ADC based on oversampling results in a reduced quantization noise.

To summarize the discussion above, we saw that by simply oversampling a band-limited signal, quantizing it, and filtering the digital sequence by an ideal lowpass filter with cutoff frequency π/OSR, we were able to reduce the in-band quantization noise power by a factor of OSR. The net effect was to make it seem as if the resolution of the quantizer had

improved, since the peak SQNR increased by $10 \log_{10}(OSR)$ dB. Thus, doubling the *OSR* results in a 3-dB (or half-bit) improvement of the quantizer resolution. We should keep in mind, however, that the cost of increasing the resolution came with the requirement of high-speed digital processing, in the form of the decimation filter.

Having tasted blood, the natural tendency is to now ask the question – can we do better than a mere half a bit every time we double the *OSR*? It turns out that we can, and we turn to our good friend, negative feedback, for inspiration.

2.4 Noise-Shaping
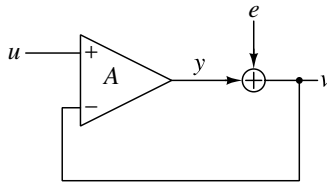
Figure 2.19   A simple negative-feedback amplifier.

Consider the feedback system with an amplifier, as shown in Figure 2.19. Think of *e* as being the output noise of the amplifier. By inspection, we have

$$v = \left(\frac{A}{1+A}\right) u + \left(\frac{1}{1+A}\right) e. \tag{2.19}$$

As the gain of the amplifier increases, we see that v starts approaching u, and the transfer function from e to v diminishes. In the limit of $A \to \infty$, $v = u$ and e does not affect the output.
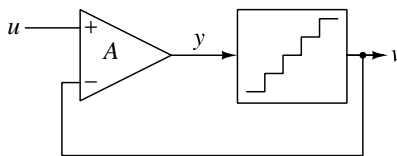
Figure 2.20   Associating e with quantization error.

Now, what if we assumed e to be quantization noise? In other words, what if the amplifier was noiseless, but we associated e with quantization noise as shown in Figure 2.20? Can we eliminate quantization noise altogether, by embedding it in a negative feedback loop, and making A sufficiently large? Achieving a large A seems easy to do – as analog designers, we are completely used to designing high-gain amplifiers. Can elimination of quantization noise be so easy?

Unfortunately, a scheme that sounds too good to be true *is* usually too good to be true, and the system of Figure 2.20 is no exception. To see why, consider a practical quantizer. Any such physically feasible device will take a finite time to operate – in other words, the quantized output will only be available a small time after the quantizer has

"looked" at the input. In other words, the amplifier will be able to use the quantizer output (and therefore adjust its own output to reduce the error between u and v) only in the next sample. Mathematically, therefore, we should insert a one-sample delay in the amplifier output, as shown in Figure 2.21, since the quantizer output can be "seen" by the rest of the circuit only in the next cycle. This concept is not entirely unfamiliar to us – the fact that there cannot be a "delay free loop" is a common idea in sequential digital state machine design.
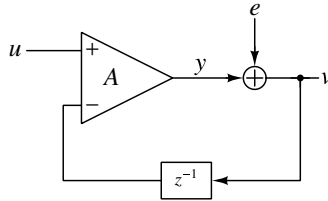
Figure 2.21   A physically realizable discrete-time feedback loop should have at least one sample delay. e is the quantization error.

Analysis of Figure 2.21 in the z-domain yields

$$V(z) = (U(z) - z^{-1}V(z))A + E(z),$$

which reduces to

$$V(z) = \underbrace{\left(\frac{A}{1 + Az^{-1}}\right)}_{\text{Signal Transfer Function (STF)}} U(z) + \underbrace{\left(\frac{1}{1 + Az^{-1}}\right)}_{\text{Noise Transfer Function (NTF)}} E(z). \qquad (2.20)$$

Anticipating things to come, we call the transfer function from the u to v the Signal Transfer Function (STF), and that from e to v the Noise Transfer Function (NTF). As $A \to \infty$, the STF approaches unity, while the NTF tends to zero. As is usually the case for the transfer functions associated with a single system, the STF and NTF have the same denominator, which is the characteristic polynomial of the system. The pole location of the system, found by determining the root of the denominator polynomial, is given by $z = -A$.
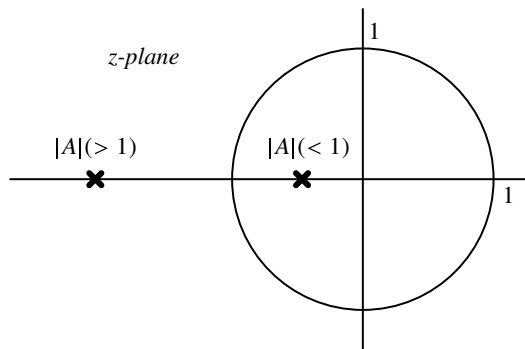
Figure 2.22   Pole locations of the system of Figure 2.21 for small and large A.

For a discrete-time system to be stable, all its poles must lie within the unit circle. In our system, the only way to ensure stability is to make $|A| < 1$, as shown in Figure 2.22.

We find ourselves in a catch-22 situation – we need $|A| \gg 1$ so that the magnitude of the NTF is small. If we do this, the system is unstable, as the pole lies outside the unit circle. If we attempt to stabilize the loop, noise suppression is lost!

From the discussion above, it is apparent that making A large at *all* frequencies is not workable – it appears as if we were being too greedy by attempting to eliminate quantization noise altogether. An aspect we have not exploited so far is that the spectrum of the input sequence u is confined to low frequencies due to oversampling. Thus, rather than try to suppress quantization noise across all frequencies, what if we were content to eliminate it only over the signal bandwidth $[0, \pi/OSR]$? Equivalently, rather than make A high at all frequencies, what if A was made high only at low frequencies? This calls for replacing the frequency-independent gain A by a block with a frequency-dependent gain. This gain should be infinite at low frequencies, so that the NTF has a small magnitude at low frequencies. The lowest order system with these characteristics is an integrator.
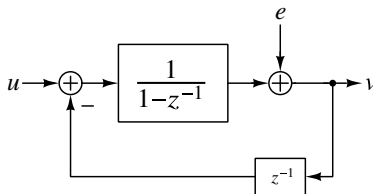
Figure 2.23   A negative feedback system where the output quantization noise is attenuated at low frequencies.

The resulting system is shown in Figure 2.23. Using $A = 1/(1 - z^{-1})$ in (2.20), we see that

$$V(z) = \underbrace{1}_{STF} U(z) + \underbrace{\left(1 - z^{-1}\right)}_{NTF} E(z). \tag{2.21}$$

The STF is unity. The NTF, $(1 - z^{-1})$, is a first-order high pass response with a zero of transmission at dc ($\omega = 0$, or, $z = 1$). This makes sense, since the gain of the "forward amplifier" is infinite at dc.

The magnitude response of the NTF in linear and log scales is shown in Figure 2.24. The shaded portion in part (a) of the figure shows the in-band component of the noise, and extends from dc to $\omega = \pi/OSR$. The log plot places in evidence the first-order nature of the highpass response, increasing at a rate of 20 dB/decade.

Let us determine the variance of the in-band quantization noise.

$$IBN = \frac{\Delta^2}{24\pi} \int_{-\frac{\pi}{OSR}}^{\frac{\pi}{OSR}} |(1 - e^{-j\omega})|^2 \, d\omega = \frac{\Delta^2}{12\pi} \int_0^{\frac{\pi}{OSR}} 4\sin^2\left(\frac{\omega}{2}\right) d\omega$$

$$\approx \frac{\Delta^2}{12\pi} \int_0^{\frac{\pi}{OSR}} \omega^2 \, d\omega = \frac{\Delta^2}{36\pi} \frac{\pi^3}{OSR^3}.$$

We see that the in-band noise power is proportional to OSR^{-3}. Doubling the OSR decreases the *in-band* noise power by 9 dB, corresponding to an effective increase in resolution of 1.5 bits. Recall that by simply oversampling (but without noise-shaping), resolution
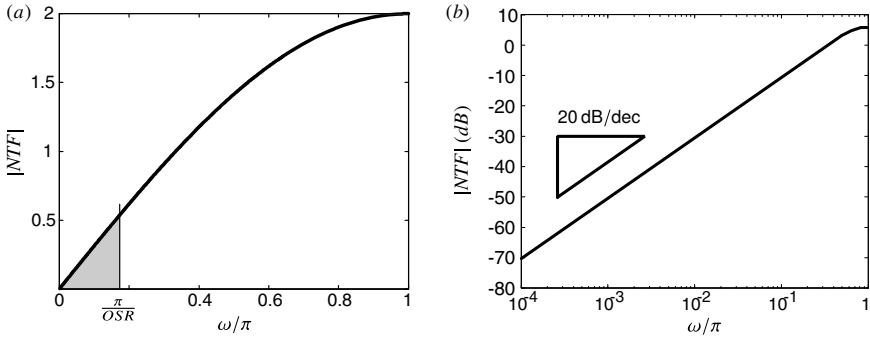
Figure 2.24   Magnitude of the NTF of a first-order noise-shaping quantizer on the (a) linear and (b) log scales.

only increased by 0.5 bit when the OSR doubled. In either case, arbitrarily high accuracy can, in principle, be achieved by using a sufficiently high value of *OSR*, but the *OSR* value needed is much lower when oversampling is combined with noise-shaping. Spectrally, quantization noise is highpass filtered, or "shaped" out of the signal band. This, therefore, is a *first-order noise-shaped converter*, also known as a first-order $\Delta\Sigma$ converter. In this book, we also affectionately (and interchangeably) refer to this system as MOD1.
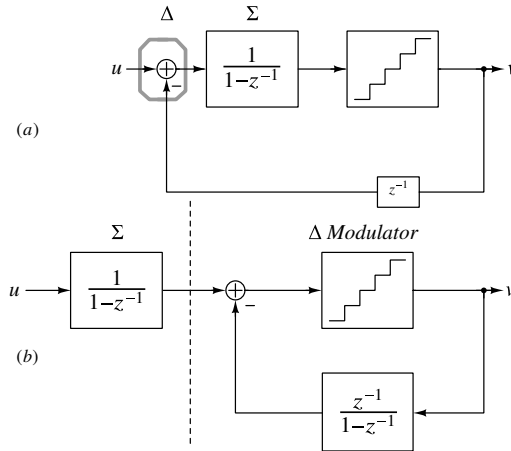
Figure 2.25   Justifying the name (a) $\Sigma\Delta$ modulator and (b) $\Delta\Sigma$ modulator.

ΣΔ or ΔΣ ?

A (friendly) bone of contention in the design community is the choice of nomenclature. Should this be called a ΔΣ modulator, or a ΣΔ modulator? Some designers argue that ΣΔ is apt due to the following. From Figure 2.25(a), we see that the Δ operation occurs first, followed by the Σ. In science and engineering, a composite operation is named in the reverse order of the individual operations. For example, to determine the root mean-square value of a waveform, we first square it, determine its mean and then compute the result's square root. In a similar vein, therefore, this should be called a ΣΔ modulator.

The proponents of ΔΣ argue that the modulator historically resulted from cascading an integrator (Σ) and a Δ modulator. Going by the reverse-order convention, it makes sense to call this the ΔΣ modulator. Another convention is that the first name used in the literature is the one that should be used, and since Inose and Yasuda [3] chose the name ΔΣ, that is the convention we have adopted. Besides, ΔΣ simply sounds better.
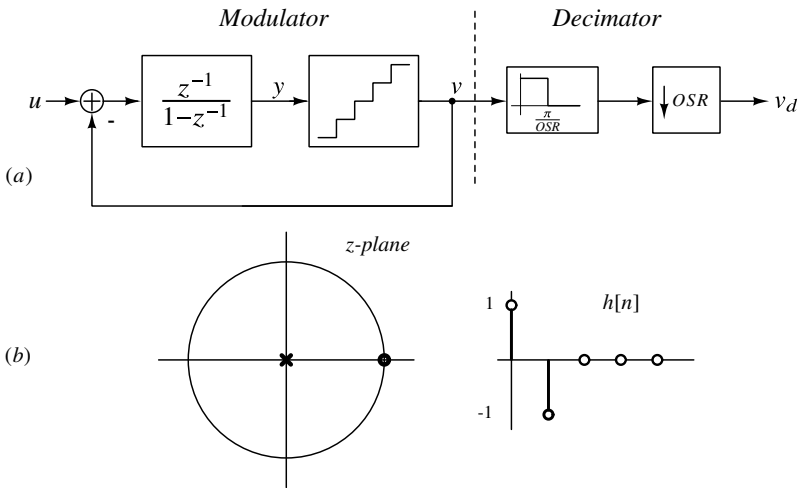
Figure 2.26   (a) Signal chain with a first-order ΔΣ loop. (b) Pole-zero map of the system and the impulse response corresponding to the NTF.

The system diagram of a first-order ΔΣ converter is shown in Figure 2.26. It consists of an "analog" part (often called the modulator), which is the negative-feedback loop incorporating the integrator, quantizer and subtractor. The decimator is comprised of a "sharp" digital lowpass filter, whose output is downsampled by a factor of OSR, yielding a Nyquist rate digital signal w.

The equations governing the operation of the modulator are

$$
\begin{aligned}
y[n] &= y[n-1] + u[n-1] - v[n-1], \\
v[n] &= Q\{y[n]\}.
\end{aligned}
\tag{2.22}
$$

The output can be expressed as a sum of the input u and the quantization error $e = v - y$. Therefore,

$$V(z) = z^{-1}U(z) + (1 - z^{-1})E(z).$$

There is *no approximation* involved in the equations above; the approximation is to assume that $e[n]$ is a white sequence.

- Quantization error is "first-order shaped" out of the signal band, with a transfer function $(1 - z^{-1})$, which has a zero at dc.

- Assuming white quantization noise, the in-band noise power is proportional to OSR^{-3}. Doubling the OSR, therefore, increases resolution by 9 dB. For an M-step quantizer, the peak SQNR for a full-scale input is given by

$$SQNR_{peak} = \frac{9M^2 OSR^3}{2\pi^2}. \tag{2.23}$$

This indicates that many design choices can be used to achieve a desired peak $SQNR$. A many-level quantizer can be used in a $\Delta\Sigma$ loop with sampling with a low OSR; alternatively, increasing the OSR allows one to reduce M. The attractive aspect of the latter approach is the greatly simplified complexity of the quantizer (and subsequently, the $\Delta\Sigma$ loop) as M is reduced.

In the limit, $M = 1$, resulting in a 2-level, or *binary*, or *single-bit* quantizer. At first glance, it would almost sound unbelievable that a 1-bit quantizer can be used to digitize a quantity with high precision. With oversampling, negative feedback and appropriate digital processing (in the form of a decimation filter), we have seen how this is possible. Such is the magic of feedback.

- Denoting the impulse response corresponding to the NTF by $h[n]$, we see that $h[0] = 1$. This makes sense due to the following. The output v due to an impulse in e at $n = 0$ is comprised of two parts: $e[0] = 1$ and $y[0]$. As we discussed earlier, one cannot implement a delay-free loop. This means that the output of the loop filter (y) at $n = 0$ must be zero – resulting in $h[0] = 1$. In the transform domain, this constraint translates as

$$NTF(z) = h[0] + h[1]z^{-1} + h[2]z^{-2} + \cdots + h[n]z^{-n} + \cdots . \tag{2.24}$$

Setting $z = \infty$, we see that
$$NTF(\infty) = h[0] = 1.$$

This (or its time domain equivalent) is a *fundamental* constraint that applies to *any* realizable $\Delta\Sigma$ loop. Evaluating (2.24) at $z = 1$, and recognizing that $NTF(1) = 0$, we see that, for any NTF with a zero at dc, $\sum_{n=0}^{\infty} h[n] = 0$.

- While the in-band noise decreases, the total quantization noise power in v is *more* than that introduced by the quantizer (which is $\Delta^2/12$). The computation of the total quantization noise power (i.e, over the entire band $[0,\pi]$) is simplified using Parseval's theorem, and is given by

$$\frac{\Delta^2}{12\pi} \int_0^\pi |NTF(e^{j\omega})|^2 \, d\omega = \frac{\Delta^2}{12} \sum_{n=0}^{\infty} h^2[n] = 2\frac{\Delta^2}{12}. \tag{2.25}$$

It is instructive to compare the output sequence of an oversampling converter (without noise-shaping), with that of a first-order $\Delta\Sigma$ modulator. Figure 2.27(a) shows the quantizer
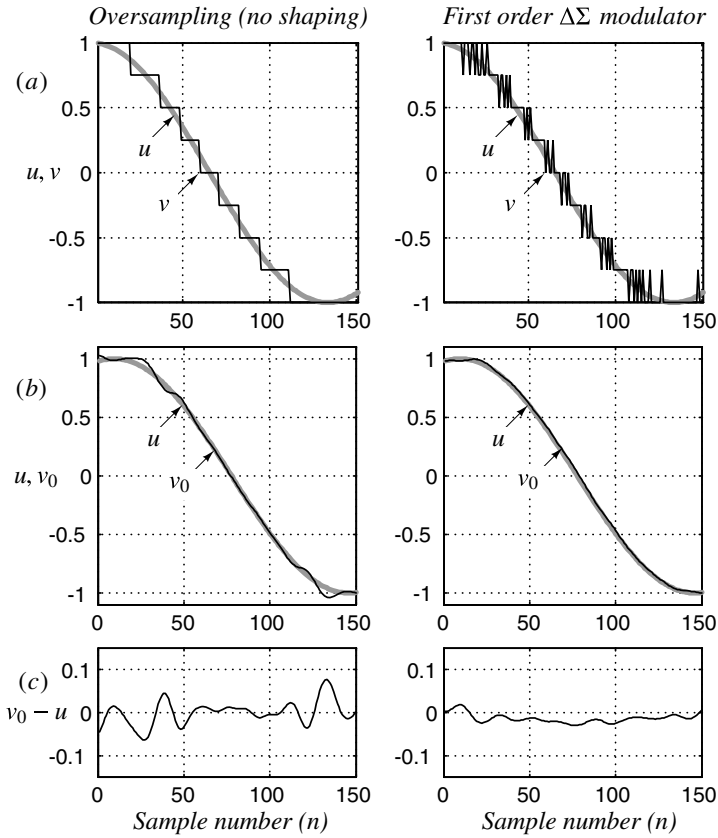
Figure 2.27   (a) u and v (b) u and v after being filtered with a digital lowpass filter. (c) Error between the input and filtered output.

outputs in both cases. The output of the $\Delta\Sigma$ modulator is seen to consist of many more transitions between levels – this makes sense, as the gain of the quantization noise at high frequencies ($\omega = \pi$) is 2 (as opposed to one in the case with only oversampling). Part (b) of the figure shows the waveforms after digitally filtering the sequences – as seen in Figure 2.27(c), it is apparent that the quantization error is greatly suppressed by noise shaping.

ADCs, DACs, and Quantizers

We saw in Figure 2.1 that an ADC *conceptually* samples an input, and then quantizes and assigns a digital word to each level of the quantizer output. While this block diagram of an ADC is convenient for analysis, a practical ADC is not implemented in this way. Rather, quantization and encoding are interwoven in ways that depend on the specific ADC architecture. In other words, a sampled and quantized version of the input ($x_q[n]$ in Figure 2.1) is simply not available. Sometimes, the input is also not *explicitly* sampled before being quantized. This is understandable because, in theory, interchanging the order of sampling and quantization has no bearing on the ADC output. Furthermore, the input has dimensions (usually of voltage or current), while the output is dimensionless.

A DAC, in contrast, produces a continuous-time *waveform* in response to a digital sequence. The dimensions of the output correspond to that of a physical quantity (e.g., voltage, current, or charge) while the input is dimensionless.

A quantizer, like the one in Figure 2.26, is a device whose input and output are dimensioned quantities. So, just how does one realize the quantizer in Figure 2.26? The common way of doing this is shown in Figure 2.28(a), where an ADC and DAC are cascaded. The resulting first-order $\Delta\Sigma$ modulator is shown in Figure 2.28(b). The output of the ADC, which is a digital word, is processed by the decimation filter, while the DAC output is subtracted from u and processed by the integrator.
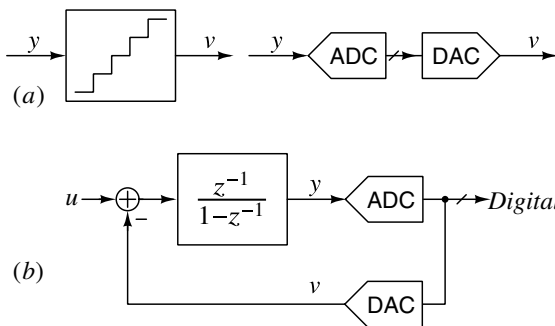
Figure 2.28   (a) A quantizer's practical realization by a ADC–DAC cascade and (b) a practical first-order $\Delta\Sigma$ modulator.

2.4.1   The Effects of Finite DC Gain of the Integrator

Ideally, the dc gain of the integrator in the $\Delta\Sigma$ loop should be infinite. With circuit nonidealities, however, one can only ensure a large (but not infinite) dc gain, which we denote by A. Referring to Fig.2.26, the transfer function of a practical integrator is modified to[4]

$$L_0(z) = \frac{pz^{-1}}{1 - pz^{-1}}, \tag{2.26}$$

where $p = A/(1 + A)$. The NTF is, therefore,

$$NTF(z) = 1 - pz^{-1}. \tag{2.27}$$

From this expression, it is apparent that the NTF's zero shifts from $z = 1$ to $z = p$, inside the unit circle. Thus the NTF gain at dc changes from its ideal value of zero to $(1 - p) = 1/(A + 1)$, and the modulator loses its ability to achieve infinite precision with dc signals. For busy input signals, the additional noise power resulting from the "filling in" of the noise notch at low frequencies can be estimated by integrating $|NTF(e^{j\omega})|^2 \approx A^{-2} + \omega^2$ across the band of interest and comparing the result against the $A = \infty$ case. If $A > OSR$, the additional noise is less than $1.2\,\text{dB}$, and hence this effect is rarely serious. Although this argument suggests that high opamp gain is not a critical requirement, the reader should be aware that the above argument assumes linear opamp gain and neglects the phenomenon of dead-zones. Low opamp gain can be problematic if the gain is sufficiently nonlinear. Also the finite-gain effect has serious consequences for the cascade (MASH) architectures discussed in Chapter 5.

2.4.2   Effect of Quantizer Nonidealities

As seen in Figure 2.29, a practical quantizer is realized as a cascade of an ADC and a DAC. How do the nonidealities of these constitutive blocks affect the quantizer, and MOD1 itself?

An ideal ADC should have uniformly spaced thresholds. In practice, however, these are shifted from their ideal values, causing the treads to have varying widths. This nonideality can be modeled as an error sequence e_{adc} at the input of the ADC, as shown in Figure 2.29(a).

All output steps of a DAC should ideally be equal. In reality, this is not the case, due to variability in components used in the DAC. This nonideality is likewise modeled as an an error signal e_{dac} added to the DAC output, as illustrated in Figure 2.29(b).

The model of MOD1, including these errors, is given in Figure 2.29(c). From the figure, it is apparent that e_{adc} should not be a cause for worry; after all, e_{adc} adds to the quantization error introduced by the ADC, and it will therefore be shaped by the negative-feedback loop just like the quantization error.

e_{dac} is problematic, however, as it cannot be distinguished from the input u and so degrades the in-band SQNR. This is in line with our general experience that the transfer function of a negative feedback system is largely determined by the feedback element (when the loop gain is large). In $\Delta\Sigma$ parlance, e_{adc} is noise-shaped whereas e_{dac} is not.

---

[4]This particular form of the integrator transfer function is chosen so as to have a "clean" NTF of the form $(1 - pz^{-1})$.
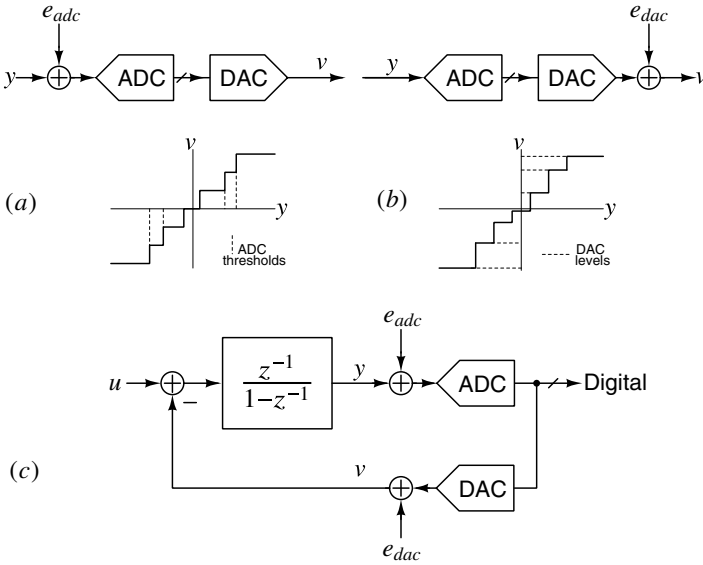
Figure 2.29 (a) Nonidealities (threshold shifts) in the ADC can be modeled as an additive sequence e_{adc}. (b) Nonuniform levels of the DAC can be modeled as an error e_{dac} at its output. (c) Modeling ADC and DAC errors on MOD1.

2.4.3 The Single-Bit First-Order Delta-Sigma Modulator
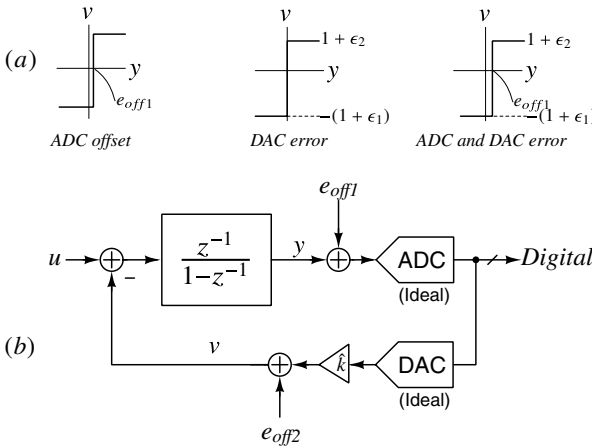
Figure 2.30 (a) Quantizer characteristics with ADC and DAC nonidealities in a 2-level modulator and (b) MOD1 with practical nonidealities.

With a single-bit modulator, the ADC threshold error (e_{off1}) is equivalent to a dc offset. The DAC levels, ideally supposed to be ± 1, are of the form $-(1 + \epsilon_1)$ and $(1 + \epsilon_2)$. The resulting quantizer characteristic is shown in Figure 2.30(a). Thus, v can be related to y by

$$v = \hat{k} \left[sign(y - e_{off1}) \right] + e_{off2}, \qquad (2.28)$$

where

$$\hat{k} = 1 + \frac{\epsilon_1 + \epsilon_2}{2} \qquad (2.29)$$

and

$$e_{off2} = \frac{\epsilon_2 - \epsilon_1}{2}. \qquad (2.30)$$

The resulting MOD1 model is shown in Figure 2.30(b). e_{off1} is of no consequence as it reduces to zero when referred to the input (due to the infinite dc gain of the integrator). e_{off2}, which adds to u, appears as an offset in the input. $\hat{k}$, which is close to unity since $|\epsilon_1, \epsilon_2| \ll 1$, alters the in-band gain of the STF to $1/\hat{k}$.

From the discussion above, we see that component mismatch in a single-bit delta-sigma modulator causes offset and gain errors, both of which are benign (unlike in the case with more than two levels). This is a *big* motivating factor that favors the use of a two-level quantizer. In fact, this property is so special that the term *inherent linearity* has been coined to describe it. There are other benefits – scaling the output of the integrator has no effect on v, since $v = sign(y)$. A downside, however, is that a higher OSR is needed to achieve a desired in-band SQNR with 2-level quantization than with more than 2 levels of quantization. This is usually not a problem at low signal bandwidths. A detailed description of the trade-offs that need to be borne in mind while designing with a single-bit quantizer is deferred to Chapter 4.

2.5   Nonlinear Aspects of the First-Order Delta-Sigma Modulator

So far, we have built intuition on the linear aspects of the first-order $\Delta\Sigma$ loop. In reality, however, the quantization noise model is not quite correct, and we should not be surprised to see deviations from the behavior we expect with linear analysis. Some of these strange aspects are discussed next. The true behavior of MOD1 is easily determined by time-domain simulations of the difference equations (2.22). Simulation is an important tool in $\Delta\Sigma$ modulator design because the linear model is imperfect and can hide important effects which only become apparent when the true nonlinear nature of the modulation process is taken into account.

All that is needed to perform such calculations are the samples of the input signal and the initial condition of the integrator. Once the output sequence has been computed, its spectrum can be found using an FFT. However, to achieve the high numerical accuracy needed, several precautionary measures must be used; these are discussed in Appendix A.

As a demonstration that MOD1 really does shape quantization noise, Figure 2.31 plots the spectrum of the output of MOD1 that employs a 2-level quantizer, when simulated with a full-scale sine-wave input. This figure clearly shows a noise-shaped characteristic, the hallmark of $\Delta\Sigma$ modulation, and the 20 dB/decade slope of the noise is consistent with first-order shaping. However, the simulated SQNR of 55 dB for $OSR = 128$ is 5 dB less than the value of 60 dB predicted by the linear model. In the discussion in the rest of this section, we assume that $M = 1$ unless mentioned otherwise.

Figure 2.32, which plots the simulated SQNR versus the input amplitude for two different test frequencies, gives a clue as to the source of this SQNR discrepancy. As this figure demonstrates, the simulated SQNR of MOD1 is an erratic function of the input
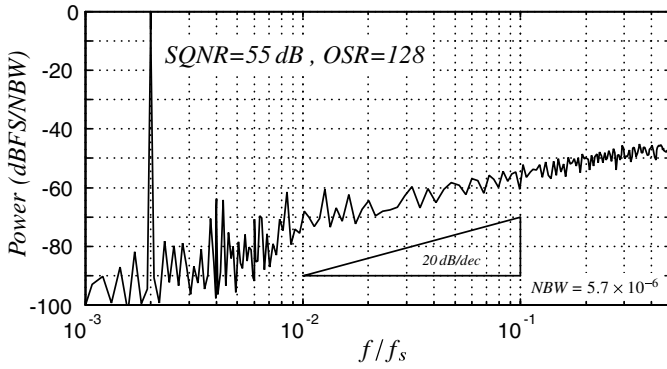
Figure 2.31   Simulated MOD1 output spectrum with a full-scale sine-wave input.
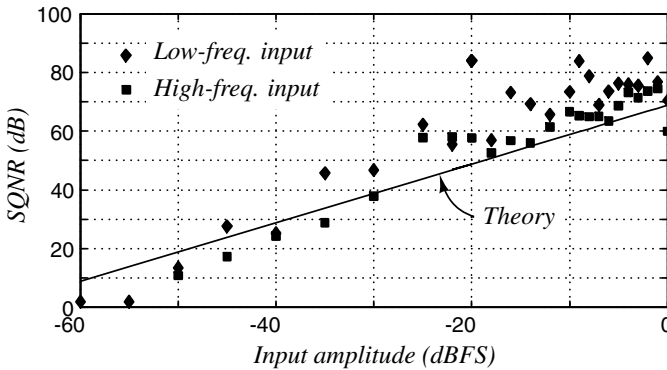
Figure 2.32   Simulated SQNR for MOD1 with $OSR = 256$.

amplitude and frequency. Clearly, the dynamics of MOD1 are not as simple as the linear model would lead us to believe.
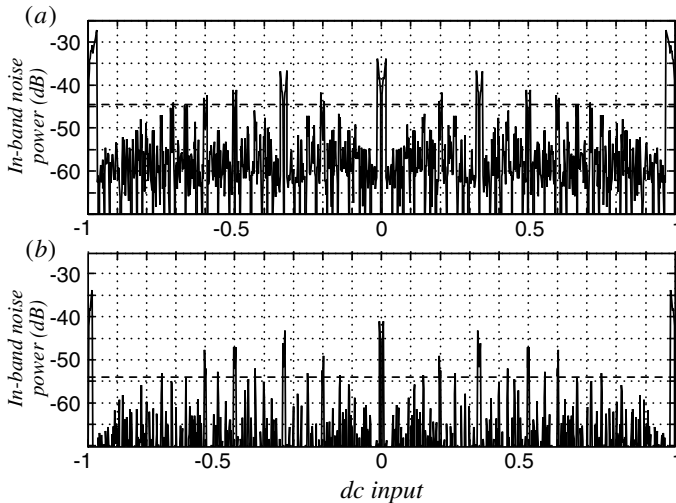
Figure 2.33   In-band quantization noise power versus the dc input level for MOD1. The broken line marks the noise power averaged over all input values. (a) $OSR = 32$ and (b) $OSR = 64$.

As a further demonstration of the complexity lurking in MOD1, Figure 2.33 plots the simulated in-band noise power as a function of the value of a dc input for two values of OSR. Also shown are the mean-square levels of the in-band noise, averaged over all input values. Both plots show that MOD1 exhibits increased in-band noise in the vicinity of certain input values, in particular ±1 and 0. Comparing Figure 2.33(a) with Figure 2.33(b), we see that as OSR is increased, the widths and absolute heights of the anomalous regions decrease, but relative to the mean noise level the noise spikes are higher. Candy and Benjamin [4] showed that the central noise peaks have a height $-20 \log(\sqrt{2} \cdot OSR)$ dB and width OSR^{-1}. Later, Friedman [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.

This and other exotic behavior is a direct consequence of the nonlinear difference equations which govern MOD1. Although a thorough study is beyond the scope of this text, it is possible to state some important results and give some indications of the underlying mechanisms.

2.6   MOD1 with DC Excitation

As mentioned earlier, the linear model of the quantizer is valid only under specific conditions (large and fast random variations) on the input y to the quantizer, which can only be satisfied if the input u to the loop meets similar conditions. In this section, we shall discuss the behavior of MOD1 for an important case in which these conditions are violated, namely the case of a constant input signal.
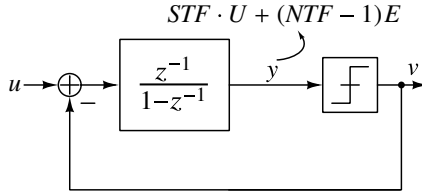
Figure 2.34 y is linearly related to u and $e = v - y$.

2.6.1   Idle Tone Generation

Consider MOD1 with a two-level quantizer, and excited by a dc input u, as shown in Figure 2.34. Despite the nonlinear quantizer, we saw earlier that y is linearly related to u and $e = v - y$, so we can write

$$Y(z) = STF(z)U(z) + (NTF(z) - 1)E(z),$$

where $STF(z) = z^{-1}$ and $NTF(z) = (1 - z^{-1})$. In the time domain, after recognizing that $v[n] = sign(y[n])$, this translates to

$$y[n] = u[n - 1] + (y[n - 1] - sign(y[n - 1])). \tag{2.31}$$

The exotic behavior of MOD1 can be captured in the single first-order nonlinear difference equation above.

If MOD1 is stable, in the sense that $y[n]$ does not reach $\pm\infty$ as $n \to \infty$, it must follow that $\bar{u} = \bar{v}$. If this were not true, the average difference between u and v would keep accumulating and eventually become infinite.

For example, assume that $u = 1/2$. If $y[n]$ is positive, then $v[n] = 1$ and y is decremented by $-(u - v) = 1/2$. If $y[n]$ is negative, then $v[n] = -1$ and y is incremented by 1.5. Starting at $y[0] = 0.1$, the operation of MOD1 with $u = 0.5$ is tabulated in the table below. Clearly, for $n = 4$, the same conditions exist as for $n = 0$. Hence, the output is periodic with a period of 4.

n	0	1	2	3	4
$y[n]$	0.1	−0.4	1.1	0.6	0.1
$v[n]$	1	−1	1	1	1

Table 2.2   MOD1 operation for $u = 1/2$.

The average value of v for a full period is $(1 - 1 + 1 + 1)/4 = 1/2$, which is identical to the input u. This is expected, since MOD1 is capable of converting a dc input signal with unlimited accuracy, provided that it is allowed to operate for unlimited time, and is followed by a perfect lowpass filter. For the input under consideration, the output is periodic with period 4, and so contains tones at $f_s/4$ and its harmonics, which will be removed by the lowpass filter.

The reader should repeat the calculation for different values of $y[0]$ to verify that the output will again be periodic with a period of 4, and that $|y[n]| \leq 2$. Over a period of 4 samples, the output will always contain three $+1s$ and one -1. The order in which they occur will depend on $y[0]$.

Consider now the more general case where the input is a constant rational number: $u = a/b$. Assume that $0 < a < b$, and that a and b are odd positive integers with no common factor. The initial value $y[0]$ is given, with $|y[0]| < 1$. Let the first b output samples of v contain $(a + b)/2$ samples of $+1$, and $(-a + b)/2$ samples of -1. Then the average value of v in this set is a/b, the same as u. The total net input to the integrator for the first b samples is zero. Hence, after the bth sample, the value of y will be again back at $y[0]$. The v sequence will, therefore, be repeated during the next b samples, and a periodic sequence with a period b will be generated.

If $u = a/b$ with one of a or b even so that $(a + b)$ is odd, it can be easily shown that again a periodic sequence results, with a period $2b$ containing $(a + b)$ $+1s$ and $(b - a)$ $-1s$. For a negative input u, the output is the negative of the above given the positive input $-u$.

Assume now that MOD1 has an output that is periodic with a period p, containing m samples of $+1$ and $(p - m)$ samples of -1 in each period. The average output for each period will be $(2m - p)/p$, a rational number. Hence, the (constant) input must equal this value, and must also be rational.

Thus, a periodic output with a constant input u ($|u| < 1$) implies that u is rational. It follows that if u is constant but irrational, so the output of MOD1 cannot be periodic.

The periodic sequences generated by rational dc inputs are sometimes called pattern noise, idle tones, or limit cycles. They do not represent instability of the loop; their amplitude does not change with time but is a complicated function of u. Their frequency also depends on the input, as discussed above.

It is important to examine the effect of limit cycles on the in-band noise of MOD1. In the numerical example discussed above, the output spectrum contains a line at dc, corresponding to the average value of v. This represents the digital equivalent of the dc input u, and is thus the desired signal. In addition, there will be spectral lines at $f_s/4, 2f_s/4$, and $3f_s/4$ that represent noise. Since the cutoff frequency f_B of the digital lowpass filter following the loop typically satisfies $f_B = f_s/(2 \cdot OSR) \ll f_s$, these lines are in the stop band of the filter, and are hence harmless.

Unfortunately, the situation is not always so rosy. Assume that the input is the rational dc value $u = 1/100$. Now the argument presented above shows that the output signal will contain tones at $f_s/200$ and its harmonics. Several of these will usually be within the passband of the digital filter, and hence deteriorate the SNR. (A detailed analysis shows that the output sequence will contain an alternation of the values $+1$ and -1 for 101 periods; the -1 that would have occurred at sample 102 is changed to $+1$ due to the accumulation of u in the integrator so that two $+1s$ occur in a row. Then the alternation resumes until sample 199, when again a -1 becomes a $+1$ so that two $+1s$ occur in succession. A quick calculation confirms that the average of the output over 200 samples is $(101 - 99)/200 = 2/200 = u$, as expected. The output spectrum can be visualized by observing that the output is equivalent to a sampled sine wave of frequency $f_s/2$ modulated by a square wave with a frequency of $f_s/200$ and a 50.5% duty cycle.)

As mentioned earlier, both the frequency and power of the tones are functions of the dc input u, so is the in-band noise that they introduce. Figure 2.33 shows the variation of in-band noise power for MOD1 with u for $OSR = 32$ and for $OSR = 64$. As the figure shows, large peaks occur near simple rational values, such as $u = 0, \pm 1/2$, and $\pm 1/3$.

In some applications, such as in digital audio, idle tones cannot be tolerated, since the human hearing apparatus can detect tones even 20 dB below the level of any white noise present. The prevention of tone generation is, therefore, an important aspect of $\Delta\Sigma$ modulator design, one that often motivates the use of high-order modulators or *dither*, an added pseudo-random signal.

Idle tones may also be generated in the presence of slowly varying input voltages, which stay near a critical level (a simple rational value) long enough for a limit cycle to become apparent. This is particularly likely to occur for low-order modulators, such as MOD1.

2.6.2 Stability of MOD1

Linear analysis would predict unconditional stability for MOD1, since the pole of the system is located at $z = 0$, well within the unit circle. This prediction, however, does not take into account the actual signal processing performed by the quantizer. Hence, time-domain considerations, taking into account nonlinearities, are required.

Consider the stability of the loop under dc excitation, assuming a two level quantizer ($M = 1$). It is obvious that the loop becomes unstable, and specifically y becomes unbounded, if $|u| > 1$. For example, if $u = 1.3$, the DAC will try desperately to balance u by feeding back a signal $+1$ every time. Even so, a net input of 0.3 will enter the integrator in every clock period, until y grows so large that the circuit becomes dysfunctional.

Vice versa, if $|u| < 1$ and the initial value of y satisfies $|y[0]| < 2$, then the loop will remain stable, with $|y|$ bounded by 2. This stability condition is sufficient even for time-varying u and is readily established as follows. Despite the nonlinear quantizer, we have seen that y is linearly related to u and $e = y - v$, allowing us to write

$$Y(z) = STF(z)U(z) + (NTF(z) - 1)E(z),$$

where $STF(z) = z^{-1}$ and $NTF(z) = (1 - z^{-1})$. In the time domain, after recognizing that $v[n] = sign(y[n])$, we can translate to

$$y[n] = u[n-1] + (y[n-1] - sign(y[n-1])), \qquad (2.32)$$

indicating that

$$|y[n]| \le |u[n-1]| + |y[n-1] - sign(y[n-1])|. \qquad (2.33)$$

If $|y[0]| < 2$, then $|y[0] - sign(y[0])| < 1$. Consequently, if $|u| < 1$, then $|y[1]| < 2$. Continuing the recursion, $|y[n]| < 2$ for all n.

If $y[0] > 2$ and $|u| < 1$, then the modulator output will contain a string of $+1$s (if $y[0] > 2$) or -1s (if $y[0] < -2$) and $|y|$ will monotonically decrease until it is less than 2, at which point the condition of the preceding paragraph will hold and y will remain bounded by 2. Thus, it is clear that MOD1 is stable with arbitrary inputs less than or equal to 1 in magnitude, and that it is able to recover from any initial condition.

2.6.3 Dead-Zones

Another interesting (and undesirable) phenomenon, which is a consequence of finite integrator gain and the nonlinear nature of the quantizer, is the occurrence of dead-zones in $\bar{v}$ for small values of u.

We first analyze MOD1 with an ideal integrator and extend the intuition built here to the case of a lossy integrator. A two-level quantizer is assumed. Figure 2.35(a) shows

(a)

(b)

$$a = p/(1 + p)$$
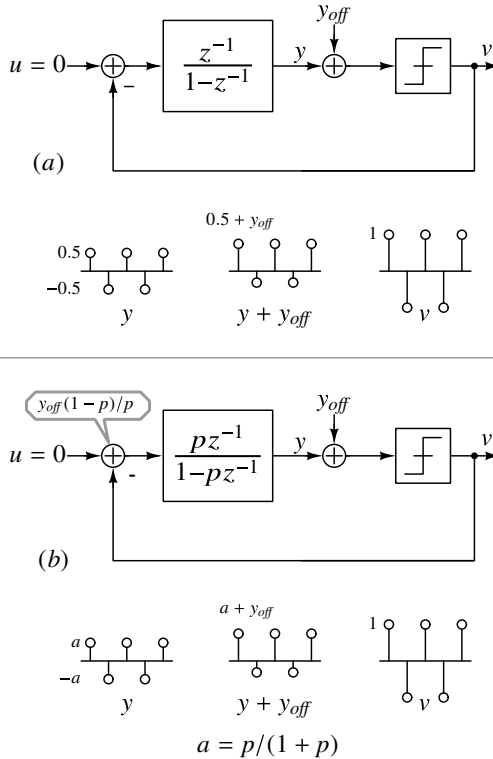
Figure 2.35   (a) MOD1 with $u = 0$. A large y_{off} does not disturb v. (b) With a damped integrator, y_{off} can be referred to the input.

MOD1 with $u = 0$. Assume that initially $y[0] = -1/2$. y_{off}, which is initially zero, is an offset we deliberately add at the output of the integrator. Since $u = 0$, $\bar{v}$ *must be zero*; or else y will eventually go to $\pm\infty$. It is easy to see that v is the repeating sequence $\cdots, 1, -1, 1, -1, \cdots$, and can be expressed as $-\cos(\pi n)$. When fed back, the resulting y in steady state is

$$y[n] = \left.\frac{z^{-1}}{1 - z^{-1}}\right|_{z=e^{j\pi}} \cos(\pi n) = -\frac{1}{2}\cos(\pi n).$$

Since $sign(y[n]) = v[n]$, this forms a consistent solution to the equations describing MOD1. Let us now increase y_{off}, as shown in the figure. It is apparent that as long as $|y_{off}| < 0.5$, v does not change, since v is the sign of $y + y_{off}$.

We next examine MOD1 when the integrator is lossy, as shown in Figure 2.35(b), with $u = 0$ and $y[0] = 0$. Further, we assume that y_{off} is initially zero. As in the discussion above, we see that v is the repeating sequence $\cdots, 1, -1, 1, -1, \cdots$. y in steady state is

$$y[n] = \left.\frac{pz^{-1}}{1 - pz^{-1}}\right|_{z=e^{j\pi}} \cos(\pi n) = -\frac{p}{1 + p}\cos(\pi n).$$

Again, since $sign(y[n]) = v[n]$, this is a consistent solution. We now notice that as long as $|y_{off}| < p/(1 + p)$, the sequence v does not change and $\bar{v}$ *remains* zero, since v is the sign of $y + y_{off}$.

The integrator, however, is lossy and has a dc gain of $A = p/(1 - p)$. This means that adding y_{off} at the output of the integrator is equivalent to adding $y_{off}(1 - p)/p$ at MOD1's input. Since $|y_{off}| < p/(1 + p)$ has no effect on $\bar{v}$, it follows that small dc inputs

$$|u| < \frac{1 - p}{1 + p} \approx \frac{1}{2A}$$

result in $\bar{v} = 0$.
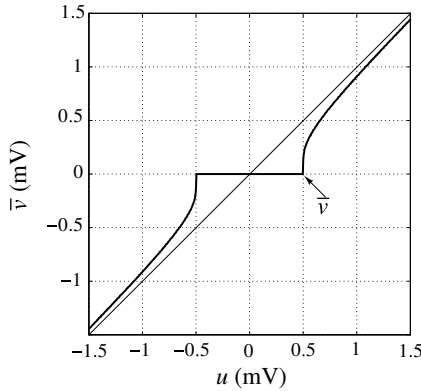
Figure 2.36 $\bar{v}$ for small dc u for MOD1. The width of the deadzone is $1/A$.

The conclusion is that with finite opamp gain, inputs smaller than $1/(2A)$ in normalized value will have no effect on the output. For example, if $A = 1000$ and $V_{ref} = 1\,V$[5], then dc signals less than $0.5\,\text{mV}$ will not register in the output. There is thus a *dead-zone* or *dead-band* around $u = 0$ if MOD1 uses a leaky integrator. It can be shown that similar dead-zones exist around all rational values of u, and that with the exception of the dead-zones around $u = \pm 1$, these other dead-zones are narrower than the one around $u = 0$. Figure 2.36 plots the simulated average output, v, of MOD1 for dc values around zero when $A = 1000$. This figure shows that the width of the dead-zone is as predicted, and that the error quickly decays to a value much less than $0.5\,\text{mV}$ for inputs outside the dead-zone.

Another consequence of the finite amplifier gain concerns the limit cycles of MOD1. The limit cycles displayed by 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, if the NTF zero is inside the unit circle, the resulting limit cycles are stable, or attracting, because sufficiently small shifts in the input lead to small changes in the integrator state, and no change occurs in the output pattern. This is a detrimental effect, since limit cycles are usually undesirable.

---

[5]The feedback values are $\pm V_{ref}$.

2.7   Alternative Architectures: The Error-Feedback Structure
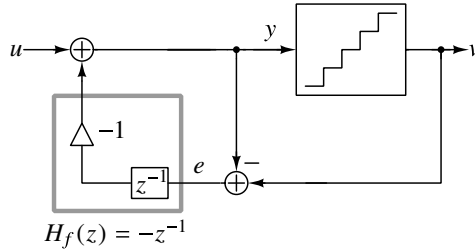
Figure 2.37   MOD1 implemented using an error-feedback architecture.

As an example of a circuit that looks simple and hence attractive, but is nonetheless problematic for analog $\Delta\Sigma$ loops, we will next consider the so-called *error-feedback struc-ture* shown in Figure 2.37. We saw this in the context of our coffee shop example in Chapter 1. Here, the quantization error e is obtained in analog form by subtracting the internal ADC's input from the DAC output; e is then fed back to the input through a filter H_f. The output signal in the z-domain is

$$V(z) = E(z) + U(z) + H_f(z)E(z). \tag{2.34}$$

Hence, $STF(z) = 1$ and $NTF(z) = 1 + H_f(z)$. To obtain $NTF(z) = (1 - z^{-1})$, clearly $H_f(z) = (1 - z^{-1}) - 1 = -z^{-1}$ is needed. This is readily realized, as shown in Figure 2.37.

Despite its attractive simplicity, the structure is problematic for analog implementa-tion, since it is sensitive to variations of its parameters. Assume that the subtractor real-izing $(v - y)$ has a +1% error, so that $1.01\,e$ is fed back. Then the actual NTF will be $(1 - z^{-1}) - 0.01z^{-1}$. At very low frequencies, instead of 0, the magnitude of e will then equal 0.01, or −40 dB. Hence, even with careful analog design, the achievable ENOB will typically be less than 12 bits for ADCs with a single-bit quantizer, even for $OSR \approx 1000$. By contrast, comparable parameter changes still allow 15-bit resolution for the structure of Figure 2.26 at $OSR = 1000$. The error-feedback structure is thus of limited utility in $\Delta\Sigma$ ADCs – however, it has found favor in noise-coupled ADCs, the principle of which will be presented in Chapters 3 and 5.

The error-feedback structure is very useful, and is often applied, in the digital loops required in $\Delta\Sigma$ DACs, where the subtraction is exactly realized. This topic will be dis-cussed in detail in Chapter 13.

2.8   The Road Ahead

So far in this text, we have seen how oversampling a signal can reduce in-band quantization noise. The benefit of *only* oversampling is modest, at 0.5 bit for every doubling of the *OSR*. We found that we could do better by shaping quantization noise out of the signal band by using negative feedback. The result of this was the first-order $\Delta\Sigma$ modulator (MOD1), which yielded 1.5 more bits for every doubling of the *OSR*. A natural question to ask is if we can do better than MOD1. This leads us to the second-order $\Delta\Sigma$ modulator, which is discussed in the next chapter.

References

[1] A. V. Oppenheim, R. W. Schafer, and J. R. Buck, *Discrete-Time Signal Processing*. Prentice-Hall, 1989.

[2] R. M. Gray and D. L. Neuhoff, "Quantization," *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2325–2383, 1998.

[3] H. Inose, Y. Yasuda, and J. Murakami, "A telemetering system by code modulation-$\Delta\Sigma$ modulation," *IRE Transactions on Space Electronics and Telemetry*, vol. 3, no. SET-8, pp. 204–209, 1962.

[4] J. C. Candy and O. J. Benjamin, "The structure of quantization noise from sigma-delta modulation," *IEEE Transactions on Communications*, vol. 29, no. 9, pp. 1316–1323, 1981.

[5] V. Friedman, "The structure of the limit cycles in sigma delta modulation," *IEEE Transactions on Communications*, vol. 36, no. 8, pp. 972–979, 1988.

CHAPTER 3

SECOND-ORDER DELTA-SIGMA MODULATION

In our analysis of MOD1, we saw its in-band SQNR can be improved by employing a quantizer with more levels. This way, the step size of the quantizer is reduced in relation to its full-scale. In the frequency domain, increasing the number of levels reduces the spectral density of the quantization noise (normalized to full-scale) over the entire frequency range $[0, \pi]$, which in turn enhances the SQNR.

Another way of improving MOD1's resolution is to use a quantizer whose *in-band quantization noise spectral density* is small, rather than increasing the number of levels (which reduces noise spectral density at *all* frequencies). The simplest quantizer that can do this is MOD1 itself. This suggests that replacing the quantizer in MOD1 by another instance of MOD1 should enhance noise-shaping. The resulting modulator is shown in Figure 3.1(a) – the portion enclosed in the gray box is the first-order modulator, which has taken the place of the quantizer.

We see that

$$V(z) = Y_1(z) + (1 - z^{-1})E(z),\tag{3.1}$$

which is of the same form as that associated with a quantizer, except that $E(z)$ has been replaced by its first-order noise-shaped version $(1 - z^{-1})E(z)$, as shown in Figure 3.1(b). This means that the input, output, and the quantization error of the arrangement in Figure 3.1 are related by

$$V(z) = U(z) + (1 - z^{-1})^2 E(z).\tag{3.2}$$

$$V(z) = Y_1(z) + (1 - z^{-1})E(z)$$

(*a*)

$$MOD1$$

(*b*)

$$(1 - z^{-1})E(z)$$
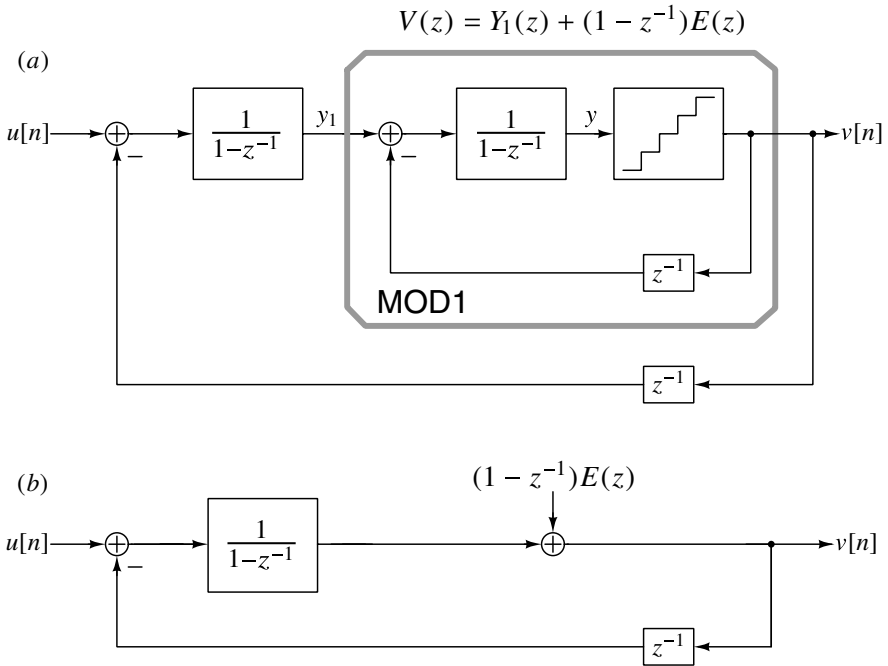
Figure 3.1 (a) Synthesis of a second-order $\Delta\Sigma$ modulator and (b) equivalent representation.

Quantization noise is therefore shaped out of the signal band by a *second-order* highpass filter. This $\Delta\Sigma$ loop is, therefore, called the second-order $\Delta\Sigma$ modulator, and is affectionately referred to as MOD2. An equivalent modulator, where the delays have been pushed into the integrators, is shown in Figure 3.2.

$$h[n]$$
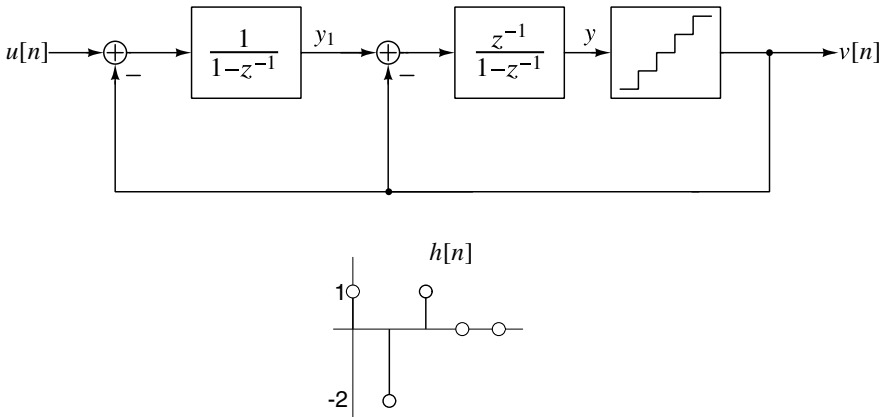
Figure 3.2 MOD2 with feedback delays pushed into the integrators, and the impulse response corresponding to the NTF.

Straightforward analysis of Figure 3.2 shows that

$$STF = z^{-1},$$
$$NTF(z) = (1 - z^{-1})^2. \tag{3.3}$$

As expected from our discussion on the infeasibility of delay-free loops in Section 2.4, we see that the first sample of the impulse response corresponding to the *NTF* is $h[0] = 1$. Due to the zeros at $z = 1$, $\sum_{n=0}^{\infty} h[n] = 1 - 2 + 1 = 0$.

At low frequencies, the magnitude of the *NTF* is approximately ω^2. The in-band noise is thus given by

$$IBN \approx \frac{\Delta^2}{12\pi} \int_0^{\frac{\pi}{OSR}} \omega^4 d\omega = \frac{\Delta^2}{60\pi} \left(\frac{\pi}{OSR}\right)^5, \tag{3.4}$$

which is proportional to OSR^{-5}. Doubling the OSR, therefore, increases the SQNR by 15 dB or 2.5 bits – compare this with 1.5 bits in the first-order case, and only 0.5 bit without noise-shaping.

The peak in-band SQNR achievable with an *M*-level quantizer is

$$SQNR = \frac{15M^2(OSR)^5}{2\pi^4}. \tag{3.5}$$
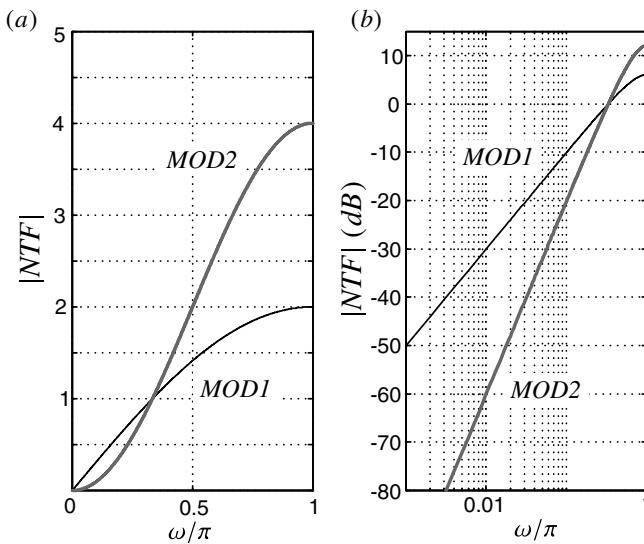
Figure 3.3 NTF magnitudes of MOD1 and MOD2 on (a) linear and (b) log scales.

Figure 3.3 compares the magnitudes of the NTFs of MOD1 and MOD2. The log plot places in evidence the second-order nature of the NTF of MOD2: we see a 40 dB/dec. slope at low frequencies.

The reduction in in-band gain of the NTF of MOD2 when compared to that of MOD1 is accompanied by an increase in the out-of-band gain. The gain at $\omega = \pi$ for MOD1's NTF is 2, while that in MOD2 is 4. One should therefore expect to see a lot more variation from sample to sample in MOD2's output, as confirmed by the waveforms in Figure 3.4.
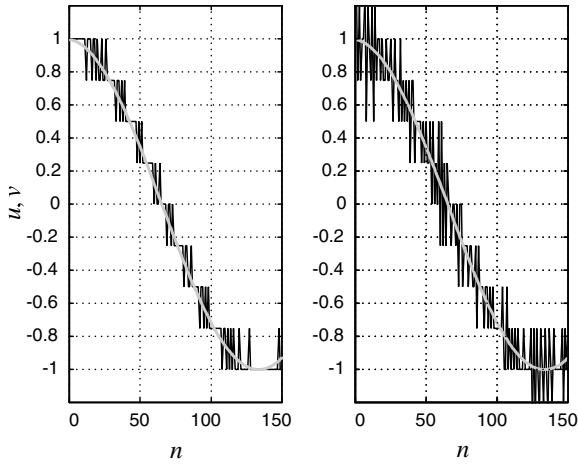
Figure 3.4   Representative input and output sequences for (a) MOD1 and (b) MOD2, $\Delta = 1/8$.

Though the *in-band* quantization noise in MOD2 is reduced compared to MOD1, the total quantization noise (i.e integrated over the entire bandwidth $[0, \pi]$) has increased, and is given by

$$\frac{\Delta^2}{12\pi} \int_0^{\pi} |NTF(e^{j\omega})|^2 \, d\omega = \frac{\Delta^2}{12} \sum_{n=0}^{\infty} h^2[n] = 6\frac{\Delta^2}{12}. \tag{3.6}$$
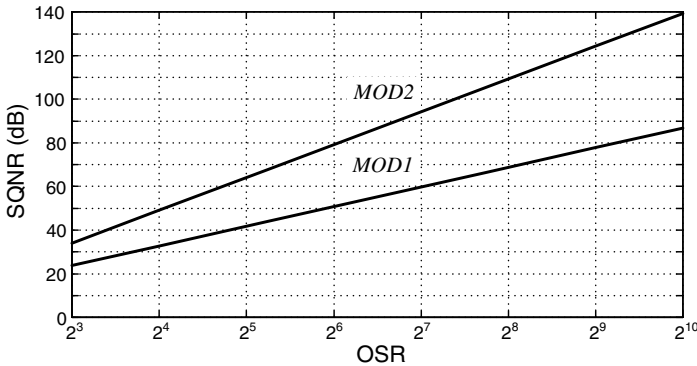
Figure 3.5   Theoretical SQNR versus OSR for MOD1 and MOD2 with a two-level ($M = 1$) quantizer.

Figure 3.5 compares the theoretical SQNR versus OSR performance of MOD2 with that of MOD1 for a two-level quantizer. As an example of the achievable ENOB, let $OSR = 128$. Then, (3.5) yields SQNR = 94.2 dB, which corresponds to nearly 16-bit resolution. If the ADC is used to convert audio signals, so that $f_B = 20$ kHz, the clock rate needed will be $f_s = 2 \cdot OSR \cdot f_B = 5.12$ MHz, a highly practical value. To achieve the same resolution with MOD1, (2.23) predicts that an $OSR = 1800$ would be needed. This necessitates the use of $f_s = 72$ MHz, which would make the implementation unnecessarily difficult.
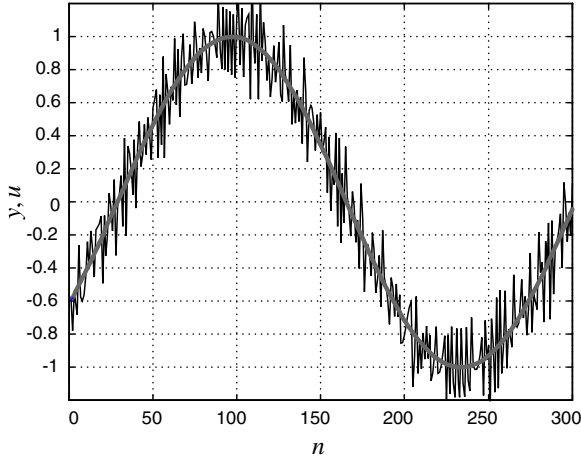
Figure 3.6   The quantizer input of MOD2 consists of u with shaped noise riding over it. $\Delta = 1/8$.

It is instructive to examine the signal that drives the quantizer embedded in MOD2. Refer to Figure 3.2, where y is seen to be $(v[n] - e[n])$. Since the magnitude of the STF at low frequencies is unity, $v[n]$ consists of the input plus quantization noise shaped by $NTF(z)$. $Y(z)$ is thus given by

$$
\begin{aligned}
V(z) &= \underbrace{STF(z)}_{=z^{-1}} U(z) + NTF(z)E(z) \\
Y(z) &= V(z) - E(z) \\
&= z^{-1}U(z) + (NTF(z) - 1)E(z).
\end{aligned}
\tag{3.7}
$$

Since $NTF(z) - 1 = -2z^{-1} + z^{-2}$, the mean square noise on y is $(\Delta^2/12)\left[1^2 + 2^2\right] = 5(\Delta^2/12)$. Representative waveforms are shown in Figure 3.6.

If binary quantization is used, MOD2 has the same inherent linearity property as MOD1. Like MOD1, the linearity of MOD2 is not jeopardized by comparator non-idealities, such as offset or hysteresis, since these errors are injected into the loop at the same point as the quantization noise and are thus also attenuated by the NTF. Slight shifts in the loop filter coefficients are similarly benign, since these translate into small shifts in the pole locations of the NTF and STF, rather than into nonlinearities.

3.1   Simulation of MOD2

As with MOD1, the validity of the assumptions that lead to analytical SQNR predictions for MOD2 are best checked by performing simulations based on the modulator's difference equations. With a many-level quantizer, the output sequence straddles u, as shown in Figure 3.4.   With a two-level quantizer, the output is binary as shown in Figure 3.7, which illustrates the input and output waveforms for MOD2 with a half-scale sine-wave input, with a two-level quantizer. As is typical in $\Delta\Sigma$ systems, these waveforms provide very little insight into the system's behavior when viewed in the time domain. Only crude
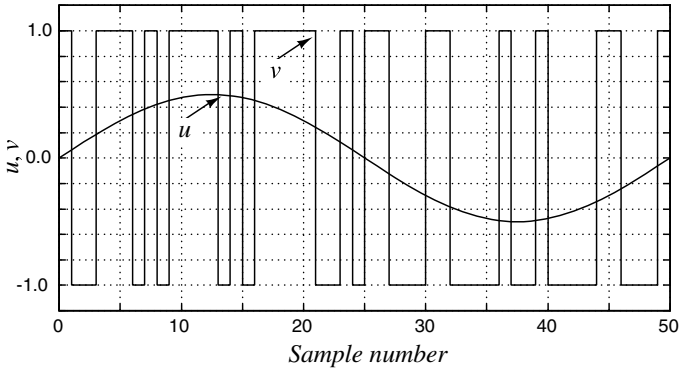
Figure 3.7   MOD2 input and output waveforms for a two-level quantizer.

observations can be made, such as the increased tendency for the output to be +1 when the input is positive and −1 when the input is negative. However, by viewing the modulator output in the frequency domain, a much clearer picture emerges.
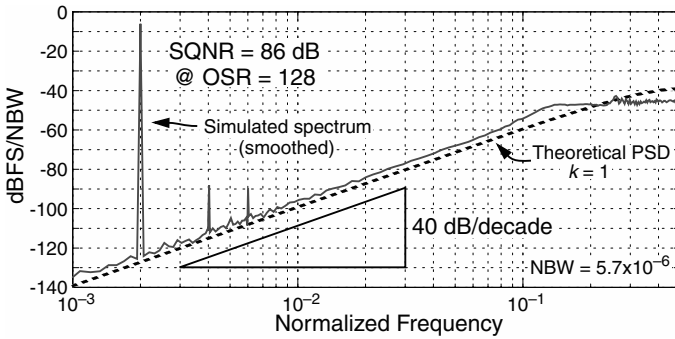
Figure 3.8   Output spectrum of MOD2 with a −6 dBFS sine-wave input.

The spectrum shown in Figure 3.8 clearly exhibits noise-shaping, and the 40 dB/dec. slope confirms the second-order nature of the shaping. For an oversampling ratio of 128, integrating the PSD yields $SQNR = 86$ dB and extrapolating from this value to the SQNR for a full-scale input results in a predicted peak SQNR of 92 dB. This is in close agreement with the theoretical result of 94 dB predicted by (3.5). However, Figure 3.8 displays two features that are not in agreement with our model. First, the second and third harmonics of the signal are distinctly visible, having amplitudes of −88 dBFS and −90 dBFS, respectively. Since white quantization noise cannot produce harmonics of the signal, this simulation result is inconsistent with the white-noise model. Second, the figure shows a plot of MOD2's NTF, scaled by $(\Delta^2/12) \cdot 2NBW$, so that it should align with the observed PSD.[1] The theoretical curve is similar to the observed PSD, but has a corner frequency that is higher than that of the observed PSD. The higher corner frequency makes the theoretical

---

[1]NBW is an abbreviation for noise bandwidth. See Appendix A for a discussion of the importance of NBW in the context of a PSD plot.

PSD somewhat lower at low frequencies and somewhat higher at high frequencies than the observed PSD.
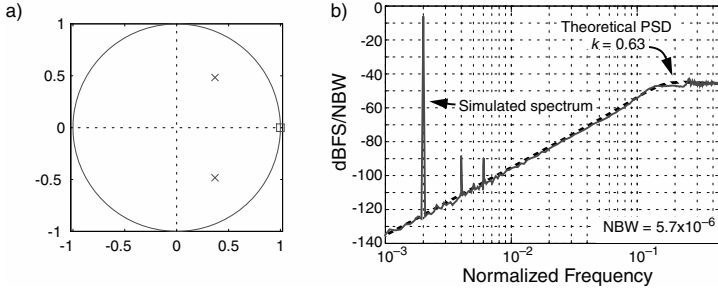
Figure 3.9   (a) Pole-zero plot of the NTF of MOD2 if the quantizer gain is $k = 0.63$; (b) comparison of corresponding PSD with simulation.

To account for the harmonics, a nonlinear quantizer model is needed. Such a model will be presented in the next section. To account for the shift in NTF shape, it is sufficient to evaluate the effective quantizer gain from simulation. Applying (2.14) to the simulation data yields $k = 0.63$, and recomputing the NTF for this value of k results in the pole-zero and PSD plots shown in Figure 3.9. Now the agreement between the observed PSD and that "predicted" by the linear model is very good.

When the input amplitude is lowered, simulations show that the optimal value of the quantizer gain changes slightly, and is approximately $k = 0.75$ for inputs that are less than -12 dBFS. Thus, simulations indicate that a more accurate NTF for MOD2 with small inputs can be derived if it is assumed that $k = 0.75$, rather than one. What is the effective NTF if k is not 1?

We denote the NTFs associated with quantizer gains of 1 and k by NTF_1 and NTF_k, respectively. Denoting the transfer function from v to y by $-L_1(z)$, we see that

$$NTF_1(z) = \frac{1}{1 + L_1(z)}.$$ (3.8)

If the gain of the quantizer was k instead, with quantization noise being added after this gain, the resulting NTF would be

$$NTF_k(z) = \frac{1}{1 + kL_1(z)}.$$ (3.9)

Using (3.8) and (3.9), we obtain

$$NTF_k(z) = \frac{NTF_1(z)}{k + (1 - k)NTF_1(z)}.$$ (3.10)

Applying (3.10) with $k = 0.75$ yields the improved estimate of MOD2's NTF:

$$NTF(z) = \frac{(1 - z^{-1})^2}{1 - 0.5z^{-1} + 0.25z^{-2}}.$$ (3.11)

Since this NTF has an in-band gain 2.5 dB above that of $NTF_1(z) = (1 - z^{-1})^2$, the in-band noise power seen in simulation should be about 2.5 dB higher than that predicted from NTF_1.
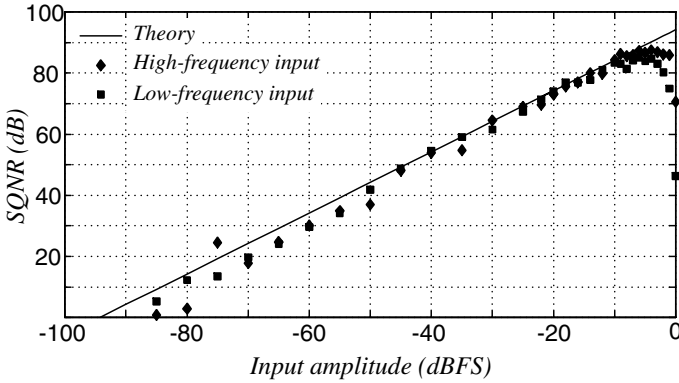
Figure 3.10 Simulated SQNR of MOD2 with *OSR* = 128.

When the input amplitude exceeds half of full-scale, the optimal quantizer gain decreases, degrading the noise-shaping further and increasing the difference between the expected and simulated SQNR. To illustrate this, Figure 3.10 plots the SQNR of MOD2 as a function of the input amplitude, for two different test frequencies, and compares the simulation results against the predictions of the unity-gain white-noise quantizer model. The simulation data follows that expected from theory quite closely over the middle portion of the amplitude range. For small inputs, the observed SQNR is somewhat lower than predicted, with the result that 0 dB SQNR requires a larger input than expected. For large input amplitudes, the SQNR of MOD2 saturates, peaking for inputs in the vicinity of −5 dBFS, and then dropping abruptly as the input amplitude approaches full-scale. The largest degradation in SQNR occurs for low-frequency full-scale signals because these signals apply large values for extended periods of time to the input of MOD2. Comparing Figure 3.10 with the corresponding one for MOD1 (Figure 2.32), we see that the SQNR of MOD2 is more well-behaved than MOD1, except for the aforementioned saturation under large-signal conditions.

3.2 Nonlinear Effects in MOD2

The simulations of MOD2 presented above confirm the essential insights provided by the linear model, but also exhibit anomalies that can only be explained with nonlinear models. Unfortunately, the dynamics of MOD2 with a binary quantizer are much more complex than those of MOD1 and defy exact analysis. In light of the intractable dynamics, it is reasonable to use approximate and/or empirical techniques to explain the observed behavior of MOD2.

3.2.1 Signal-Dependent Quantizer Gain

As Section 2.2.1 described, the gain of a binary quantizer is indeterminate. Moreover, the simulations of the previous section suggest that, in the case of MOD2, the quantizer's gain is in fact signal dependent. A model for MOD2 that includes this signal-dependent gain (a nonlinear effect) is depicted in Figure 3.11.
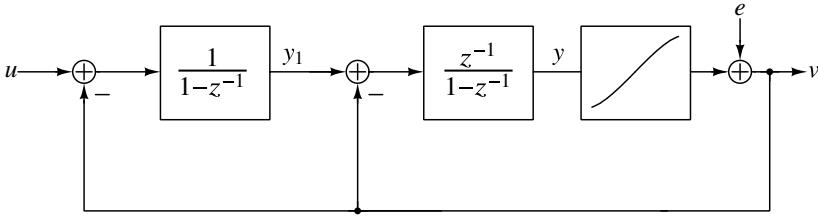
Figure 3.11   MOD2 model incorporating quantizer nonlinearity.

In this figure, the quantizer is replaced by a weak nonlinearity, the quantizer transfer curve (QTC), plus additive noise. The QTC is determined by computing the average quantizer output as a function of the average quantizer input, while the dc input signal is swept across its range. Figure 3.12 shows the QTC determined by simulations. This figure shows that the QTC is compressive, meaning it exhibits reduced gain when the magnitude of its input is large. Figure 3.12 also plots a cubic approximation to the quantizer nonlinearity, $v = k_1 y + k_3 y^3$, and lists the (k_1, k_3) coefficients of this approximation. We can use these coefficients to estimate the distortion induced by the QTC as follows.
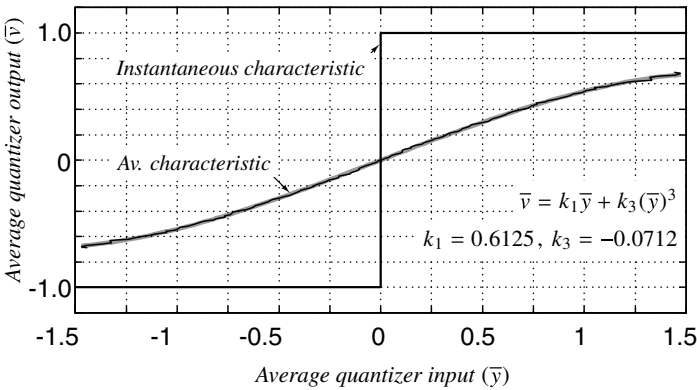
Figure 3.12   The quantizer's "average" transfer characteristic is obtained by plotting $\bar{v}$ as a function of $\bar{y}$.

First, determine the effective NTF of the modulator by using $k = k_1$ as the effective the quantizer gain:

$$NTF_{k_1} = \frac{(1 - z^{-1})^2}{1 - 0.775z^{-1} + 0.3875z^{-2}}. \tag{3.12}$$

Next, since the distortion term $k_3 y^3$ is added at the same place in the loop as the quantization error, we conclude that the spectrum of the distortion is shaped by NTF_{k_1}. Thus, distortion is greatest for frequencies where the NTF gives the least protection, that is, when the distortion term lies at the edge of the passband. For the spectrum of Figure 3.9, the input signal is located at $f = f_s/500$, and thus the third harmonic is at a normalized frequency of $3f = 0.006$, where the attenuation provided by NTF_{k_1} is about 53 dB.

For a small low-frequency sine-wave 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 sine-wave, but of amplitude A/k_1. The amplitude of the third harmonic induced by the QTC is $k_3(A/k_1)^3/4$, from which it follows that the third harmonic distortion is

$$HD_3 = \left| \frac{k_3 A^2}{4k_1^3} NTF_{k_1}(z) \right|, \tag{3.13}$$

where $z = e^{j2\pi(3f)}$. Evaluating (3.13) under the conditions of Figure 3.9 ($A = 0.5, f = 1/500$) yields $HD_3 = -87$ dB. Since this calculated value is about 5 dB short of the observed value of -82 dB, the distortion estimate provided by the calculation above can only be considered a rough approximation. The discrepancy is greatest when the input is large and the distortion is severe.
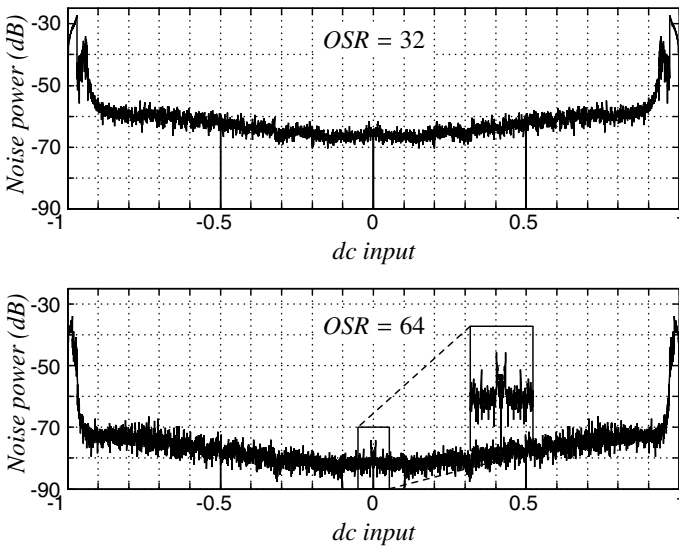
Figure 3.13   Simulated in-band quantization noise power for MOD2 as a function of dc input level.

      Returning now to our empirical study of MOD2's behavior as its input is varied, Figure 3.13 plots the in-band quantization noise power of MOD2 (with binary quantization) versus the dc input level for oversampling ratios of 32 and 64. Comparing these plots with those for MOD1 (Figure 2.33), we immediately note that the numerous spikes associated with periodic behavior in MOD1 are now absent, and thus conclude that MOD2 is more resistant to tones than MOD1. However, as the expanded view around $u = 0$ in Figure 3.13 illustrates, MOD2 still exhibits vestiges of this undesirable behavior.

      A second feature apparent in the plots of Figure 3.13 is the U-shape of the in-band noise power, which culminates in very large noise peaks as $|u| \to 1$. Even over the range $|u| < 0.7$, (i.e., for inputs less than -3 dBFS), the in-band noise varies by about 6 dB. The bulk of this behavior is accounted for by the signal-dependent quantizer gain (and hence signal-dependent NTF) of MOD2. As the input increases in magnitude, the quantizer gain decreases, reducing the loop gain and hence the effectiveness of the noise-shaping. As a

result, the noise at the output of MOD2 increases with signal level. In the terminology of stochastic processes, 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.

3.3   Stability of MOD2

The preceding section used a combination of simulation and quasi-linear modeling to explain several of the nonlinear aspects of MOD2. In this section, results from the literature regarding the stability of MOD2 with a dc input are presented.
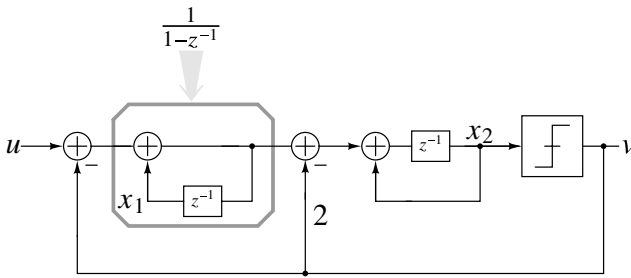
Figure 3.14   Model used to establish bounds on state variables in MOD2.

Hein and Zakhor [1] have shown that if MOD2 is constructed as in Figure 3.14, so that the difference equations are

$$v[n] = sign(y[n]) = sign(x_2[n]),$$

$$x_1[n + 1] = x_1[n] - v[n] + u,$$

$$x_2[n + 1] = x_1[n] + x_2[n] - 2v[n] + u,$$

then for dc inputs u satisfying $|u[n]| < 1$ the following bounds apply:

$$|x_1| \leq |u| + 2, \tag{3.14}$$

$$|x_2| \leq \frac{(5 - |u|)^2}{8(1 - |u|)}. \tag{3.15}$$

Figure 3.15 plots these bounds along with values found by simulation. As this figure shows, the analytic bounds are fairly tight for $|u| < 0.7$, but the analytic bound on x_2 "blows up" more quickly than do the simulation results as $|u| \to 1$. Since x_2 is (a delayed version of) the quantizer input, Figure 3.15 once again demonstrates that the input to the quantizer in MOD2 becomes large as the input approaches full-scale.

According to (3.14) and (3.15), the internal states of MOD2 are guaranteed to be bounded for dc inputs less than 1 in magnitude, although the bound on x_2 does become arbitrarily large as $|u| \to 1$. Since MOD2 tracks the low-frequency content of its input,
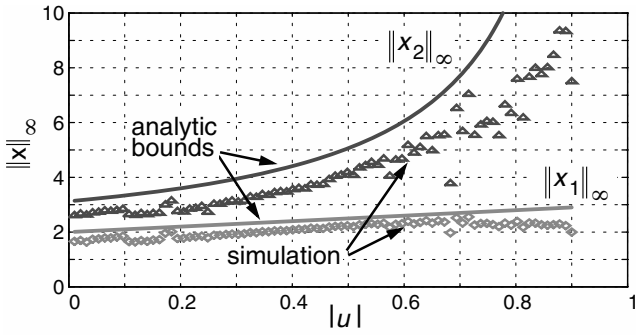
Figure 3.15 Comparison of analytic state bounds with simulation for MOD2.
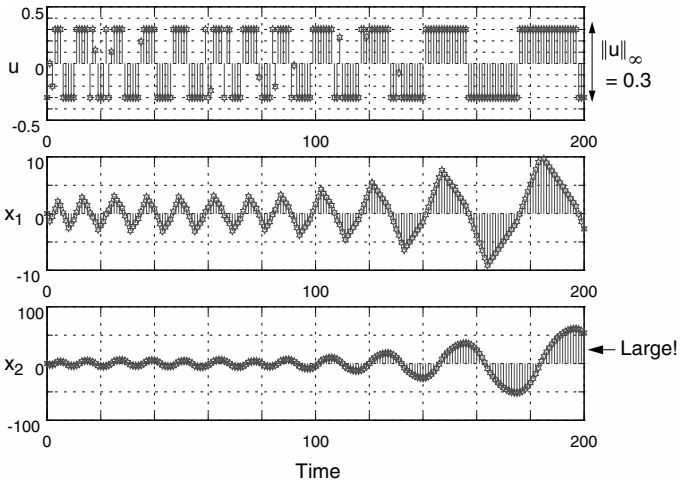
Figure 3.16 A hostile input with $|u| \leq 0.3$ can drive MOD2 "unstable".

one might assume that arbitrary time-varying inputs satisfying $|u[n]| < 1$ for all n would lead to bounds similar to those for dc inputs. However, as Figure 3.16 illustrates, an input waveform with $|u| \leq 0.3$ can lead to large internal states if the waveform is chosen appropriately. The input waveform shown in Figure 3.16 predominantly oscillates between -0.3 and $+0.3$, and it does so with just the right phase and period so as to drive the modulator state into ever-increasing oscillations. Furthermore, the values at the transitions are chosen so as to maximize the excursion of the internal oscillations. Fortunately, such a waveform is unlikely to be encountered in practice, and even if it were, it is even more unlikely that it would be precisely synchronized with the modulator's internal dynamics the way the waveform in Figure 3.16 is. It is known that MOD2 is stable for arbitrary inputs less than 0.1 in magnitude, but the upper limit on the input amplitude for which stable operation is guaranteed is not known.

The preceding discussion shows that MOD2 is "less stable" than MOD1. Although the stability of MOD2 with dc inputs less than 1 in magnitude has been rigorously established, the states may become arbitrarily large as $|u| \to 1$. It is therefore wise to limit the input of MOD2 to 0.9, if possible, so that the state of the second integrator is not unduly large. Unfortunately, even though such an input limit will keep the modulator state reasonable for dc and slowly varying inputs, it is possible for the modulator state to become much larger than intended. It is therefore important to include means for detecting overly large states and for placing the modulator in a "good" state.

3.3.1 Dead-Zones

Section 2.4.1 showed that when the dc gain of the integrator is finite, the zero in MOD1's NTF shifts from $z = 1$ to $z = p$, where $(1 - p)$ is (approximately) inversely proportional to the integrator's dc gain, A. This shift in the NTF zero creates dead bands in the response of MOD1 to dc inputs; the most troublesome of these dead bands is the one centered on $u = 0$, the width of which was calculated to be $2/A$.
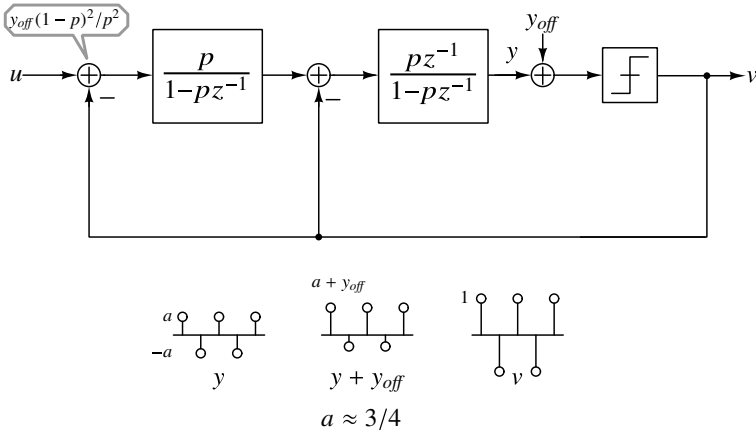
Figure 3.17   Origins of the dead-zone around $u = 0$ in MOD2 when the integrators have finite gain $A = p/(1 - p)$.

How does finite integrator gain cause dead-zones in MOD2? The analysis proceeds as with MOD1, and is illustrated using Figure 3.17. y_{off} is an externally added offset, and it is initially zero. The transfer function from v to y is

$$- L_1(z) = - \frac{p^2 z^{-1}}{(1 - pz^{-1})^2} - \frac{pz^{-1}}{1 - pz^{-1}}. \tag{3.16}$$

When $u = 0$, it is easy to see that v is the periodic sequence $\cdots, -1, 1, -1, 1, \cdots = \cos[\pi n]$. y, in steady state, is given by

$$y[n] = - L_1(z)|_{z=-1} \cos[\pi n] = \frac{p(2p + 1)}{(p + 1)^2} \cos[\pi n] \approx \frac{3}{4} \cos[\pi n]. \tag{3.17}$$

The approximation in the equation above is valid since $p \approx 1$. Since $y_{off} = 0$, the quantizer input is $(3/4) \cos[\pi n]$, and $sign(y[n]) = v[n]$, which is consistent with the constraint imposed by closing the loop.

If we now deliberately introduce a nonzero offset y_{off}, we see that the output sequence v is not disturbed as long as $|y_{off}| < (3/4)$. Adding y_{off} at the quantizer input is equivalent to adding

$$u_{eq} = \frac{y_{off}}{\underbrace{\frac{p^2}{(1 - p)^2}}_{\text{dc gain from } u \text{ to } y}} \tag{3.18}$$

at the modulator's input. Since $p/(1 - p) = A$, we see that the average of the output sequence of MOD2 will remain zero as long as

$$|u_{eq}| < \frac{3}{4} \frac{1}{A^2}. \tag{3.19}$$

Thus, the impact of finite integrator gain is drastically reduced because the open-loop gain of the loop filter is proportional to A^2.
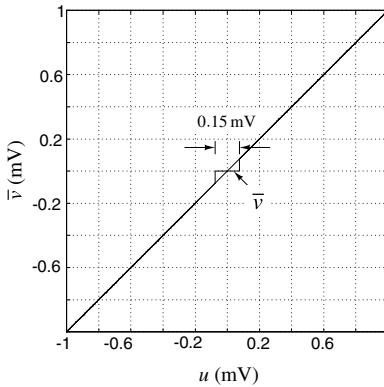
Figure 3.18 $\bar{v}$ for small dc u for MOD2. The width of the deadzone is $1.5/A^2$.

To verify the preceding calculation, in Figure 3.18, we plot the average value of the output of MOD2 for small dc inputs; the poles of the loop filter are located at $z = 0.99$ with

$A \approx 100$. The observed width of the dead band is identical to our estimate. Furthermore,we see that, as with MOD1, inputs outside the dead band are accurately represented by the average output of MOD2.

The gain-squaring nature of the two-integrator cascade found in MOD2 is responsible for MOD2's increased tolerance of finite opamp gain and reduced susceptibility to tones relative to MOD1. Also, as noted earlier, MOD2 possesses a vastly superior SQNR/OSR trade-off. Aside from the added analog complexity, the main drawback of MOD2 is reduced stability, which is manifested in a practical input range that is about 80% to 90% of the potential full-scale range of the converter.

The following section presents alternative realizations of MOD2, as well as a modified version of MOD2 that offers improved SNR and better stability.

3.4 Alternative Second-Order Modulator Structures

A number of alternative structures exist that can perform a second-order modulation, and that also give a unity-gain STF (albeit with a delay of one or two clock periods), as well as the same NTF as the structure shown in Figure 3.1. In devising such structures, care must be taken to avoid delay-free loops (which are not realizable), and to preserve reasonable robustness against the unavoidable nonideal practical effects such as element inaccuracies and finite opamp gain.

Second-order modulator structures that implement lowpass STFs exist, as do structures that implement NTFs other than plain second-order differentiation. In dealing with such NTFs, care must be taken to ensure that the resulting modulator is stable. For example, converting the delay-free integrators of Figure 3.1 into delaying-integrators and removing the feedback delay alters the NTF and yields a marginally stable system.

3.4.1 The Boser–Wooley Modulator
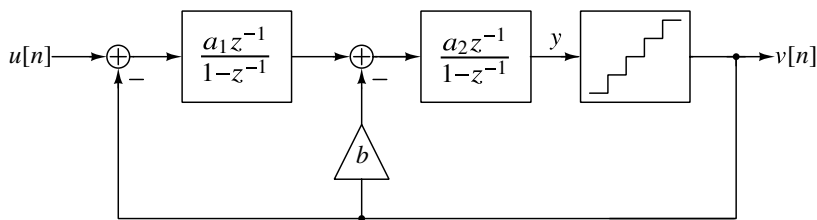
Figure 3.19   A second-order modulator with delaying integrators.

A second-order modulator that contains two delaying integrators [2] is shown in Figure 3.19. Using delaying integrators is desirable because it allows the opamps in each integrator to settle independently of each other, thereby relaxing their speed requirements [3].

Presuming unity gain for the quantizer, we can use linear analysis to show that the STF and NTF are

$$STF(z) \quad = \quad \frac{a_1 a_2 z^{-2}}{D(z)},$$

$$NTF(z) \quad = \quad \frac{(1 - z^{-1})^2}{D(z)}, \tag{3.20}$$

where
$$D(z) = (1 - z^{-1})^2 + a_2 b z^{-1}(1 - z^{-1}) + a_1 a_2 z^{-2}. \tag{3.21}$$

.

To achieve $STF = z^{-2}$ and $NTF(z) = (1 - z^{-1})^2$, we need to ensure that the conditions $a_1 a_2 = 1$ and $a_2 b = 2$ are satisfied. Since we have 3 parameters and only 2 constraints, there are many possible solutions. For example, $a_1 = a_2 = 1$, $b = 2$, or $a_1 = 0.5$, $a_2 = 2$, and $b = 1$, can be used. In the actual design process, dynamic range scaling (to be discussed in Section 4.7) removes any ambiguity in finding the parameters needed to implement a given NTF and STF.

3.4.2 The Silva–Steensgaard Structure
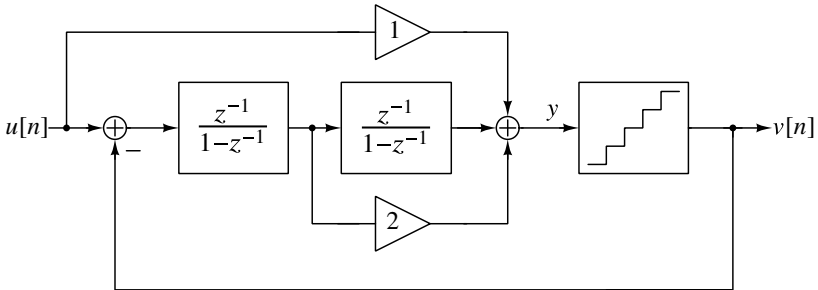
Figure 3.20   A second-order modulator with feedforward paths.

Another useful second-order structure is shown in Figure 3.20 [4, 5]. The distinguishing features of this circuit are the direct feedforward path from the input to the quantizer and the single feedback path from the digital output. Linear analysis confirms that the output is given in the z-domain by

$$V(z) = U(z) + (1 - z^{-1})^2 E(z), \tag{3.22}$$

as before. The input signal to the loop filter is, however, different: it contains only the shaped quantization noise:

$$U(z) - V(z) = -(1 - z^{-1})^2 E(z). \tag{3.23}$$

Also, as can be seen from (3.23), the output of the second integrator will give $-z^{-2} E(z)$ directly. This is advantageous if the modulator is the input stage of a MASH structure (discussed in Chapter 5).

3.4.3 The Error-Feedback Structure

$$H_f(z) = -2z^{-1} + z^{-2}$$
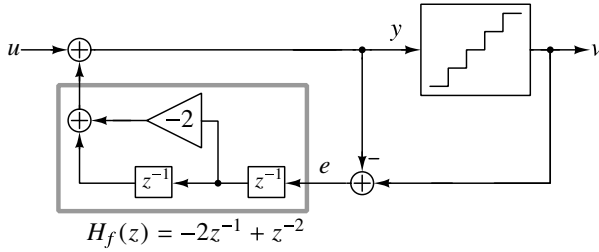
Figure 3.21 MOD2 implemented using an error-feedback architecture.

The error-feedback structure, shown in Figure 3.21, is the second-order analog of the corresponding structure for MOD1 (Figure 2.37). The output signal in the z-domain is

$$V(z) = E(z) + U(z) + H_f(z)E(z). \tag{3.24}$$

Hence, $STF(z) = 1$ and $NTF(z) = 1 + H_f(z)$. To obtain $NTF(z) = (1 - z^{-1})^2$, $H_f(z) = (1 - z^{-1})^2 - 1 = z^{-2} - 2z^{-1}$. As discussed in connection with Figure 2.37, this structure is not particularly suited to analog realization, but it is commonly used in digital $\Delta\Sigma$ loops.

3.4.4 The Noise–Coupled Structure

$$V(z) = Y_1(z) + (1 - z^{-1})E(z)$$
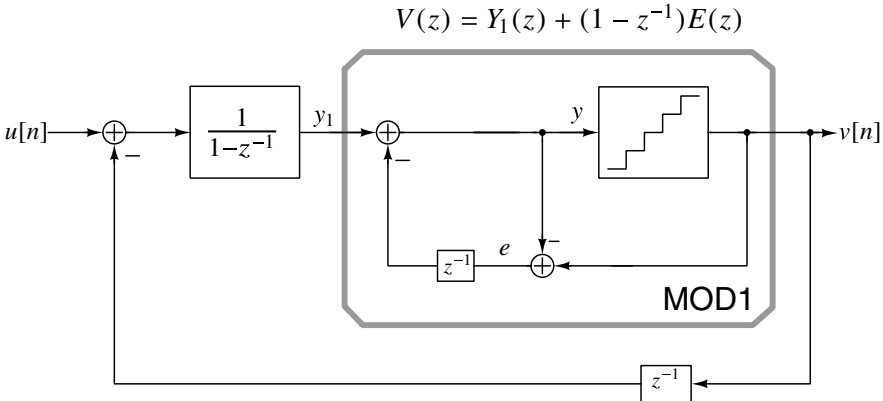
Figure 3.22 The noise-coupled MOD2 structure.

MOD2 was realized by replacing the quantizer in MOD1 with a first-order noise-shaped quantizer (which is MOD1 itself). If MOD1 is implemented as an error-feedback structure, as shown in Figure 3.22, a noise-coupled MOD2 structure results. When compared to an analog realization of Figure 3.21, this system is more robust with respect to practical nonidealities, thanks to the first integrator. At the same time, it inherits the inherent simplicity of the error-feedback structure. This technique has been applied to high-order modulators, to realize an NTF with an extra order of noise-shaping without the associated active circuitry.

3.5 Generalized Second-Order Structures
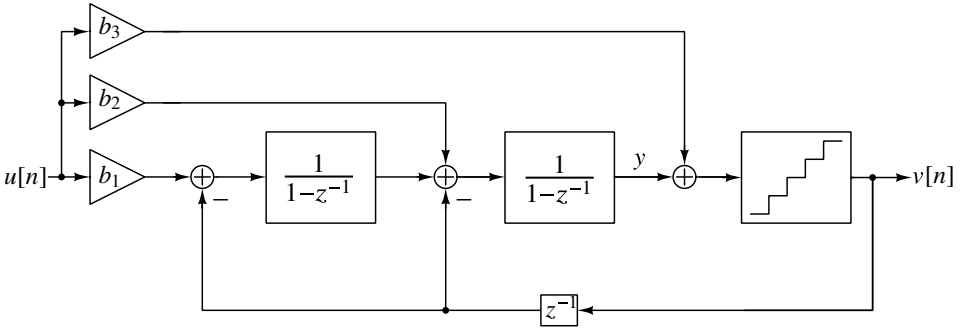
Figure 3.23 A second-order modulator with feed-in paths.

The structure of Figure 3.1 gives $STF(z) = 1$ and $NTF(z) = (1 - z^{-1})^2$. More general STFs can be obtained by feeding u not only to the input of the first integrator but also to the inputs of the second integrator and the quantizer (Figure 3.23). The STF then becomes

$$STF(z) = b_1 + b_2(1 - z^{-1}) + b_3(1 - z^{-1})^2. \tag{3.25}$$

The STF now has two zeros, and a double pole at $z = 0$. In this way, a "free" second-order FIR signal pre-filter can be incorporated into the ADC.
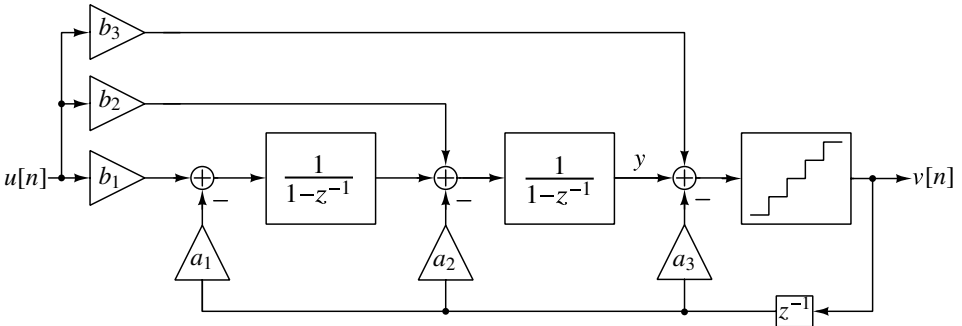
Figure 3.24 A second-order modulator with feed-ins and feedback paths.

Similarly, by feeding the output signal v back to all three blocks in the forward path (Figure 3.24), two nonzero poles can be generated in both the *STF* and the *NTF*. Thus, more general STFs and NTFs can be obtained. The new functions are

$$STF(z) = \frac{B(z)}{A(z)},$$

$$NTF(z) = \frac{(1 - z^{-1})^2}{A(z)}, \tag{3.26}$$

where

$$B(z) = b_1 + b_2(1 - z^{-1}) + b_3(1 - z^{-1})^2, \tag{3.27}$$

and

$$A(z) = 1 + (a_1 + a_2 + a_3 - 2)z^{-1} + (1 - a_2 - 2a_3)z^{-2} + a_3 z^{-3}. \tag{3.28}$$

Since the feedback term a_3 increases the NTF order to 3, but does not increase the number of in-band NTF zeros, this term is rarely used.

By incorporating both multiple feedforward and feedback features into the second-order structure, more flexibility is obtained for enhancing stability and improving dynamic range.

3.5.1 Optimal Second-Order Modulator

Given the variety of modulator architectures shown thus far, the reader might well wonder which architecture is the best. One part of the answer to this question deals with the optimal NTF, while the second part of the answer deals with the optimal topology. The STF is a secondary consideration because the STF merely filters the signal; it plays no role in determining the peak SQNR. Also, since the choice of the topology is tied more closely to practical considerations than to fundamental mathematical limits, this section will only consider the problem of optimizing the NTF.

The first step in finding an optimal second-order modulator is to find the second-order NTF that yields the highest SQNR, or equivalently, the NTF that minimizes the in-band noise. For high values of OSR, the magnitude of $NTF(z) = (1 - z^{-1})^2 / A(z)$ in the signal band is approximately $K\omega^2$, where $K = 1/A(1)$. By shifting the NTF zeros from $z = 1$ to $z = e^{\pm j\alpha}$, the magnitude of the NTF in the passband becomes $|K(\omega - \alpha)(\omega + \alpha)| = |K(\omega^2 - \alpha^2)|$. The integral of the square of this quantity over the passband is a measure of the in-band noise, and it can be minimized by choosing α such that

$$I(\alpha) = \int_0^{\frac{\pi}{OSR}} (\omega^2 - \alpha^2)^2 \, d\omega \tag{3.29}$$

is minimized. The solution to this optimization problem can be obtained by differentiating $I(\alpha)$ with respect to α, and equating the result to 0. This gives

$$\alpha_{opt} = \frac{1}{\sqrt{3}} \frac{\pi}{OSR}. \tag{3.30}$$

Since the ratio $I(0)/I(\alpha_{opt}) = 9/4$, the expected SQNR improvement is $10 \log(9/4) = 3.5 \, \text{dB}$. Note that these results assume that the quantization noise is white, and that $|A(e^{j\omega})| = 1$ in the $[0, \pi/OSR]$ frequency range.

An exhaustive search of the NTF design space for the NTF with the highest peak SQNR yields the NTF whose SQNR curve is depicted in Figure 3.25. The denominator of this optimal NTF is

$$A_{opt}(z) = 1 - 0.5z^{-1} + 0.16z^{-2}. \tag{3.31}$$

Compared to the SQNR curve of MOD2 shown in Figure 3.10, the SQNR curve associated with this NTF is more linear and supports signals closer to full-scale without saturating. As a result, the peak SQNR for this NTF (with $OSR = 128$) is around 94 dB, which is about 6 dB higher than that of MOD2.
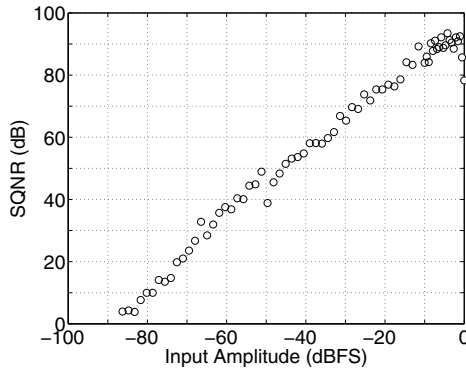
Figure 3.25   SQNR for an optimal second-order NTF for $OSR = 128$.

3.6   Conclusions

This chapter examined the second-order modulator, MOD2, and several of its variants. Like MOD1, MOD2 is theoretically able to achieve arbitrarily high resolution for dc inputs, and is immune to a variety of imperfections. In contrast to MOD1, which displays a 9 dB increase in SQNR for every doubling of OSR, the SQNR of MOD2 increases at 15 dB per octave. As a result, MOD2 is able to achieve a given level of performance with a lower sampling rate than MOD1 would require. Due to the gain-squaring provided by the two-integrator cascade, MOD2 is also more robust in the face of finite opamp gain than MOD1. Furthermore, the quantization noise at the output of MOD2 is less likely to contain tones than the noise at the output of MOD1. In all these areas, MOD2 is markedly superior to MOD1. The primary drawbacks associated with MOD2 are increased hardware complexity (both analog and digital) and a slight decrease in the allowable signal range. Since increasing the order of the NTF seems to have done us a whole lot of good, it seems natural to explore $\Delta\Sigma$ loops that realize higher order NTFs. This forms the subject of the next chapter.

References

[1] S. Hein and A. Zakhor, "On the stability of sigma delta modulators," *IEEE Transactions on Signal Processing*, vol. 41, no. 7, pp. 2322–2348, 1993.

[2] B. E. Boser and B. Wooley, "The design of sigma-delta modulation analog-to-digital converters," *IEEE Journal of Solid-State Circuits*, vol. 23, no. 6, pp. 1298–1308, 1988.

[3] R. Gregorian and G. C. Temes, *Analog MOS Integrated Circuits for Signal Processing*. Wiley-Interscience, 1986.

[4] J. Silva, U. Moon, J. Steensgaard, and G. Temes, "Wideband low-distortion delta-sigma ADC topology," *Electronics Letters*, vol. 37, no. 12, pp. 737–738, 2001.

[5] J. Steensgaard-Madsen, *High-Performance Data Converters*. Ph.D. dissertation, The Technical University of Denmark, 1999.

CHAPTER 4

HIGH-ORDER DELTA-SIGMA MODULATORS

We derived the second-order $\Delta\Sigma$ modulator from the first-order structure by replacing the quantizer in MOD1 by another instance of itself. In the same spirit, using MOD1 in place of the quantizer in MOD2 results in a third-order modulator, with an NTF $(1 - z^{-1})^3$, as shown in Figure 4.1(a). Higher order modulators, with NTFs of the form $(1 - z^{-1})^L$, can be generated in a similar fashion.

(a)

(b)
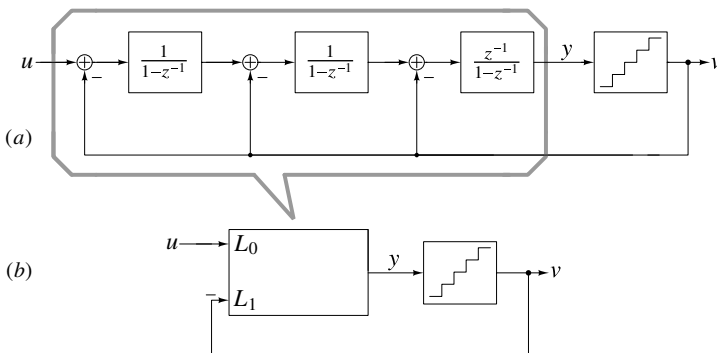
Figure 4.1 (a) A third-order modulator with $NTF = (1 - z^{-1})^3$ derived by replacing the quantizer in MOD2 with an instance of MOD1. (b) Generic representation of a (high-order) $\Delta\Sigma$ modulator.

The generic block diagram of a high-order modulator is shown in Figure 4.1(b). The loop filter processes two inputs – the input to be digitized u, and the modulator output v. The filter's output y drives the quantizer. The transfer functions from u and v to y are denoted by $L_0(z)$ and $-L_1(z)$, respectively. In terms of z-transforms,

$$Y(z) \;=\; L_0(z)U(z) - L_1(z)V(z), \tag{4.1}$$

$$V(z) \;=\; Y(z) + E(z). \tag{4.2}$$

If $V(z)$ is expressed as

$$V(z) = STF(z)U(z) + NTF(z)E(z), \tag{4.3}$$

it is easy to see that

$$STF(z) \;=\; \frac{L_0(z)}{1 + L_1(z)}, \tag{4.4}$$

$$NTF(z) \;=\; \frac{1}{1 + L_1(z)}. \tag{4.5}$$

A few observations are in order.

a. If the modulator has to be physically realizable, it cannot be delay free. Thus, as in the case of MOD1 and MOD2, the first sample of the impulse response corresponding to $NTF(z)$ must be unity. In the frequency domain, this translates to the constraint

$$NTF(z = \infty) = 1. \tag{4.6}$$

Equivalently, from (4.5),
$$L_1(z = \infty) = 0, \tag{4.7}$$

which means that the first sample of the impulse response of $L_1(z)$ must be zero.

b. The STF and NTF have the same denominator, which is the characteristic equation of the system.

c. From (4.5), we see that the NTF goes to zero when the $L_1(z)$ is infinite. This means that the poles of the loop filter are the zeros of the NTF. The loop filter of a modulator with an Lth order NTF of the form $(1 - z^{-1})^L$ should therefore have L poles at $z = 1$. This indicates that there must be L integrators in the loop, and at low frequencies $(z \rightarrow 1)$, $L_1(z)$ should approach $1/(1 - z^{-1})^L$.

d. The STF at dc $(z = 1)$ is usually chosen to be unity. From (4.4), we then have $\lim_{z \to 1} L_0(z) = \lim_{z \to 1} L_1(z)$.

e. Using $STF(1) = 1$, and assuming a low-frequency input, the input to the quantizer can be seen to be $Y(z) \approx U(z) + (NTF(z) - 1)E(z)$. Making the usual (additive white) assumption about quantization noise, we see that the variance of noise on y is $(\Delta^2/12)(\|h\|_2^2 - 1)$, where

$$\|h\|_2^2 = \left(\sum_{n=0}^{\infty} h^2[n] \right). \tag{4.8}$$

The in-band quantization noise for an Lth order NTF $(1 - z^{-1})^L$ is given by

$$IBN \approx \frac{\Delta^2}{12\pi} \int_0^{\frac{\pi}{OSR}} \omega^{2L} d\omega = \frac{\Delta^2}{12\pi(2L+1)} \left(\frac{\pi}{OSR}\right)^{2L+1}. \tag{4.9}$$

Doubling the OSR of such a modulator increases its resolution by $(L + 0.5)$ bits. It thus seems as if using $NTF(z) = (1 - z^{-1})^L$ with a sufficiently high L can yield arbitrarily large SNRs. Now, this sounds too good to be true – and we *know* that something that sounds too good to be true, *is usually* too good to be true. Not surprisingly, $\Delta\Sigma$ modulators are no exception to this rule. It turns out that Lth order NTFs of the form discussed above become unstable even for small inputs, thereby severely restricting the ADCs usable signal range. This brings us to the question of signal-dependent stability of a $\Delta\Sigma$ modulator, which we discuss next.

4.1   Signal-Dependent Stability of Delta-Sigma Modulators

Before launching into a detailed discussion on this topic, let us recall the following:

a. The expression for in-band noise (4.9) was based on the assumption that quantization can be modeled as a uniformly distributed, additive, white-noise source. As we saw in Section 2.2.1, this is largely true when the input signal exciting the quantizer does not overload it. For an M-level quantizer, this means that its input should not exceed the range $[-M, M]$.

b. The input to the quantizer in a $\Delta\Sigma$ loop consists of two parts – the (low-frequency) input u, and quantization error shaped by $(NTF(z) - 1)$ riding over it, as shown in Figure 4.2.
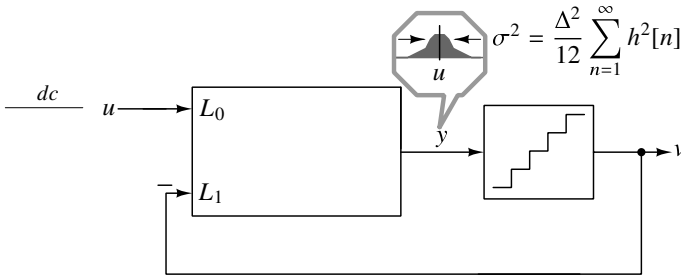
Figure 4.2   For a dc input, y consists of dc plus shaped noise with variance $(\Delta^2/12) \sum_{n=1}^{\infty} h^2[n]$.

Consider now the modulator of Figure 4.2 excited by a small dc input u, for NTFs of the form $(1 - z^{-1})^L$, where L is large. The variance of the quantizer input y is given by $(\Delta^2/12)(\|h\|_2^2 - 1)$, and is tabulated for $L = 1, \cdots, 4$ in Table 4.1. We see that the variance increases (dramatically) with the high-frequency gain of the NTF. This makes sense – the increased gain of the NTF at $\omega = \pi$ means that quantization noise is amplified to a larger degree as it goes around the loop.

Order (L)	NTF	Gain at $\omega = \pi$	$\|h\|_2^2 - 1$
1	$(1 - z^{-1})$	2	1
2	$(1 - z^{-1})^2$	4	5
3	$(1 - z^{-1})^3$	8	19
4	$(1 - z^{-1})^4$	16	69

Table 4.1   Gain at π and variance of shaped noise ($\Delta = 2$) at the quantizer input for $\Delta\Sigma$ modulators with NTFs of the form $(1 - z^{-1})^L$.

As u (which we assume to be dc) is slowly increased, there comes a point when the quantizer begins to saturate – infrequently at first, but more and more often as u continues to increase. When the quantizer saturates, the effective gain for u and the shaped quantization noise both fall, and the variance of the fitting error (between y and v) increases beyond $\Delta^2/12 = 1/3$. What happens to the loop? To build intuition, we assume that the number of quantizer steps (M) is large.

Stated formally, the problem is the following: we wish to examine the behavior of the modulator of Figure 4.3(a) as the dc input u is increased from zero. The quantizer is assumed to have M steps, and saturates if its input exceeds the range $[-(M + 1), (M + 1)]$. We make the following observations.
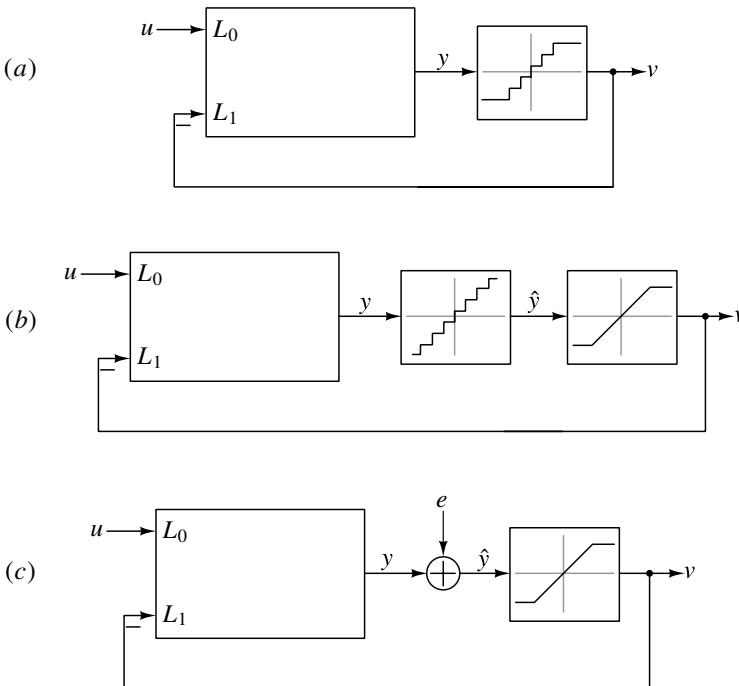
Figure 4.3   (a) $\Delta\Sigma$ modulator with a saturating quantizer; (b) modeling the saturating quantizer by cascading one with an infinite range followed by a saturating unity gain block; and (c) quantization noise modeled as a uniformly distributed white sequence e, with $|e| < \Delta/2$. When $|u|$ is large, saturation introduces an additional error.

a. The effect of saturation can be separated from the process of quantization by thinking of the saturating quantizer as a cascade of a saturating nonlinearity following a quantizer with an infinite range, as shown in Figure 4.3(b). The (fictitious) output of the infinite-range quantizer is denoted by $\hat{y}$.

b. From the discussion in Section 2.2.1, $\hat{y}$ can be thought of as $y + e$, where e is a uniformly distributed white noise sequence with variance $\Delta^2/12 = 1/3$. The resulting system is shown in Figure 4.3(c). The *only* approximation made here is the nature of e.

The phenomenon we wish to understand can therefore be equivalently recast as follows: how does the system of Figure 4.3(c) behave as a function of u (assumed to be dc), where e is a zero mean, uniformly distributed noise with variance $1/3$?

We first consider the case when $e = 0$. What is the allowed range of u, for which the loop "works" (i.e., $u = v$)? For inputs that do not saturate the output, namely for $|u| \leq M$, it is easy to see that $v = u$ and $\hat{y} = y = u$. The system is stable. Why? The NTF is $(1 - z^{-1})^L$ by design, which means that the modulator has L poles at $z = 0$.

What happens when $|u|$ exceeds M (with e still being zero)? For large positive u, v saturates to M as the negative feedback loop attempts to make u and v equal. Since v cannot exceed M, feedback drives y to ∞ in desperation. Mathematically, at low frequencies, $L_0(z) = L_1(z) \propto z^{-1}/(1 - z^{-1})^L$. Thus, as $z \to 1$, $Y(z) \approx (U(z) - V(z))/(1 - z^{-1})^L$, indicating that $(u - v)$ is integrated L times. If v saturates, $(u - v)$ is nonzero, causing y to reach ∞. It is thus seen that the usable range u, even when $e = 0$, is given by $|u| < M$. When e is non-zero, this range is further decreased, as explained below.

Consider the system of Figure 4.3(c) with a dc u chosen so that $|u| < M$. We now introduce the additive noise e as shown in the figure. e "goes around" the loop, and *if $|u|$ is small enough so that the quantizer is not overdriven*, the transfer function from e to v is simply the NTF. For what range of u can saturation be avoided? Referring to Figure 4.3(c), and using $Y(z) = STF(z)U(z) + (NTF(z) - 1)(V(z) - Y(z))$, we have

$$
\begin{aligned}
y[n] &= u + (h[n] - \delta[n]) * (v[n] - y[n]) \\
&= u + \sum_{k=-\infty}^{n} (v[n-k] - y[n-k]) \cdot (h[k] - \delta[k]). \quad (4.10)
\end{aligned}
$$

For an M-step quantizer without overload, $|v[n-k] - y[n-k]|$ (which is the quantization error) should be less than $(\Delta/2) = 1$. Thus,

$$
\max_n y = \max_n\{u + \sum_{k=-\infty}^{n} (|v[n-k] - y[n-k]|) \cdot |h[k] - \delta[k]|\} \leq \max_n u + \sum_{k=1}^{\infty} |h[k]|. \quad (4.11)
$$

A sufficient (but not necessary) condition to avoid overload, therefore, is to limit $\max_n\{y\}$ to $(M+1)$, since the quantizer will be overloaded if y exceeds the range $[-(M+1), (M+1)]$. This is assured if [1]

$$
|u|_{max} = \max_n |u[n]| \leq M + 2 - \underbrace{\sum_{k=0}^{\infty} |h[k]|}_{\triangleq \|h\|_1}, \quad (4.12)
$$

where $\|h\|_1$ is called the 1-norm of $h[n]$.

For $NTF(z) = (1 - z^{-1})^L$, $\|h\|_1 = 2^L$. Thus, from (4.12), it is seen that using an L-bit $((2^L - 1)$-step) quantizer results in $|u|_{max} = 1$! If, however, an $(L + 1)$-bit quantizer is used, we would have $|u|_{max} = 2^L + 1$, indicating that the usable range *without overloading the quantizer* is about half the quantizer range.

Extensive simulations performed for dc, sinusoid, and noise input signals [2] showed that for $L = 5$ and $M > 2^5$, the condition (4.12) is a very tight one. Thus, signal levels only slightly higher than the value given by (4.12) could cause instability. This indicates the practical value of this condition.

For small M, however, the condition (4.12) is overly restrictive. For example, our friend MOD2 is known to be stable with binary quantization and dc inputs $|u| < 0.9$. However, for MOD2, $\|h\|_1 = 4$, and thus (4.12) yields $|u|_{max} = 3 - 4 = -1$, meaning (4.12) cannot guarantee stable operation of MOD2 for *any* input. The problem with (4.12) is that it requires the quantizer to never be overloaded, whereas in practice a modulator can work properly even if the quantizer is occasionally overloaded.

To better understand this, we begin by recognizing that Figure 4.3(c) is a nonlinear system, excited by a dc input u and noise e. We then use a tool familiar to analog designers – small signal analysis. u is assumed to set the quiescent operating point of the system, while e is the *incremental input*.
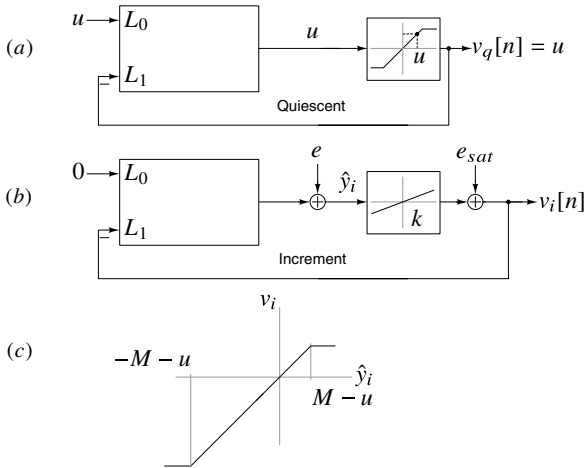
Figure 4.4   (a) u sets the quiescent operating point. (b) e is the *incremental* input. The effective gain (k) of the saturating element depends on the nature of e, as well as the operating point u. (c) Linearization of the saturating element around its operating point.

Since u is dc, and satisfies $|u| \le M$, the input and output of the nonlinear element are both u. The quiescent output is $v_q[n] = u$, as shown in Figure 4.4(a). In the incremental model, shown in Figure 4.4(b), u is set to zero (this is the "bias"), and the saturating nonlinearity is replaced by a gain k. The input and output of the saturating nonlinearity are denoted by $\hat{y}_i$ and $v_i[n]$, respectively. To determine k, the nonlinear element must be linearized around its operating point u. Figure 4.4(c) shows the shifted characteristic that must be used for this purpose. How do we determine k? As discussed in Section 2.2.1, it should be chosen to make the output of the nonlinearity v_i the best linear approximation of

$\hat{y}_i$, namely $k = \langle \hat{y}_i, v_i \rangle / \langle \hat{y}_i, \hat{y}_i \rangle$. The fitting error associated with this process is denoted by e_{sat}.

When the modulator saturates, k becomes smaller than unity. This makes intuitive sense: v_i with saturation is smaller than what it would have been otherwise. The transfer function from e to v_i is seen to be

$$NTF_k(z) = \frac{1}{1 + kL_1(z)} = \frac{NTF(z)}{k + (1 - k)NTF(z)}. \tag{4.13}$$

In our example, $NTF(z) = (1 - z^{-1})^3$. As k changes, the poles are the roots of the characteristic polynomial $(1 - k)(z - 1)^3 + kz^3$. Figure 4.5 shows the loci of the roots as k falls
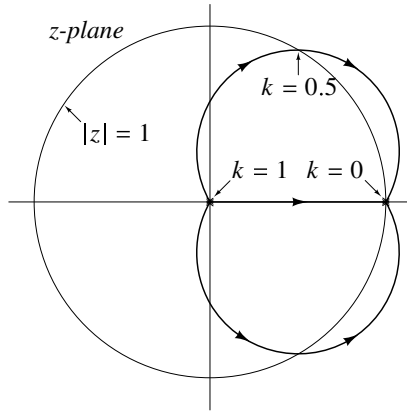
Figure 4.5 Root locus plot for a third-order $\Delta\Sigma$ modulator with $NTF = (1 - z^{-1})^3$, as k falls from one to zero. The system becomes unstable for $k < 0.5$.

from one to zero. Without saturation, $k = 1$, and the system has three poles at $z = 0$. When $k = 0$, they must be at $z = 1$. As the gain reduces to $k = 0.5$, the system becomes unstable. This is hardly surprising, since most high-order negative-feedback systems are only conditionally stable.

The linear model's behavior gives insights into what can be expected of the modulator as u increases. When $|u|$ is sufficiently close to M, so that the quantizer saturates, the effective gain for the shaped noise decreases. This causes the poles of the linearized system (Figure 4.4b) to move toward the unit circle. Further, e_{sat} is no longer zero. As u is further increased, both these effects are accentuated – k falls further, and the variance of e_{sat} increases even more. Both of these increase the excursions of $\hat{y}_i$, saturating the loop further (causing k to reduce) until the system becomes unstable, in the sense that $\hat{y}_i$ and other states in the loop filter become infinite. Now, the loop filter's output can become infinite only if u does not equal $\bar{v}$, indicating that noise-shaping is lost. The summary of the discussion above is the following:

a. A very loose bound on $|u|$ is the maximum quantizer output M, since the quantizer output can never balance the input in this case.

b. The range of $|u|$ for stable modulator operation, assuming an M-step quantizer, is at most $M + 2 - \|h\|_1$. For u in this range, the quantizer does not saturate. However, this is a very restrictive limit.

c. The modulator remains stable even when $|u|$ is increased beyond this limit, by letting the quantizer saturate occasionally. The amount by which $|u|$ can exceed $M + 2 - \|h\|_1$ depends on M and the NTF (through $\|h\|_1$).

It is thus apparent that signal-dependent stability should be *expected*. The largest input for stable operation, normalized to the quantizer's full scale output, is called the maximum stable amplitude (MSA) of a $\Delta\Sigma$ modulator, and is defined as

$$MSA \triangleq \frac{\max |u|}{M}. \tag{4.14}$$

The "root cause" for instability is saturation, and not the process of quantization: using an infinite-level quantizer would not destabilize the modulator, in the sense that the state variables become unduly large.

As discussed earlier, the MSA depends on the variance of the shaped noise exciting the quantizer, which in turn is dependent on the maximum gain of the NTF. Thus, for a $\Delta\Sigma$ loop with an NTF of the form $(1 - z^{-1})^L$ and an M-step quantizer, the MSA should reduce as L increases. While we discussed dc inputs in this section, the MSA should also be expected to depend on input frequency; we saw a similar effect in connection with a second-order modulator in Section 3.3.

In concluding this section, it should be reiterated that while there are impressive research results available on the stability of high-order delta-sigma modulators, extensive behavioral simulations are still advisable before implementing them. The lower the resolution of the quantizer used, the more suspicious the designer should be about unforeseen instability!

4.1.1   Estimating Maximum Stable Amplitude

How does one estimate the maximum stable amplitude of a $\Delta\Sigma$ modulator? Simulation of the difference equations describing the modulator is the best approach. One simulation approach is the following. u is an in-band sine wave, whose amplitude is stepped. For each amplitude, the in-band SQNR is computed. When the amplitude exceeds the MSA, the magnitude of the quantizer input approaches infinity. Consequently, noise-shaping is lost, and the SQNR degrades dramatically. A disadvantage of the sinewave method is that multiple long simulations (one for each amplitude) are necessary to estimate the MSA.

An alternative technique [3] is to excite the modulator with a ramp that slowly varies from 0 to full scale, say, over a million samples, as shown in Figure 4.6(a). The magnitude of the quantizer input is monitored. When u exceeds the MSA, the modulator becomes unstable, and y approaches infinity. The value of u when this happens is the MSA.

Figure 4.6(b) shows a plot of $20 \log |y|$ as a function of u for a third-order modulator with $NTF(z) = (1 - z^{-1})^3$, and a 9-level quantizer. From this figure, it is apparent that the MSA is about -4 dBFS. Simulations show that this method yields results that are close to what one gets with the sine-wave method, while being quicker. We also note that the MSA (at least for slow inputs) is much higher than what the $1/8 (= -18$ dBFS) value obtained from (4.12).
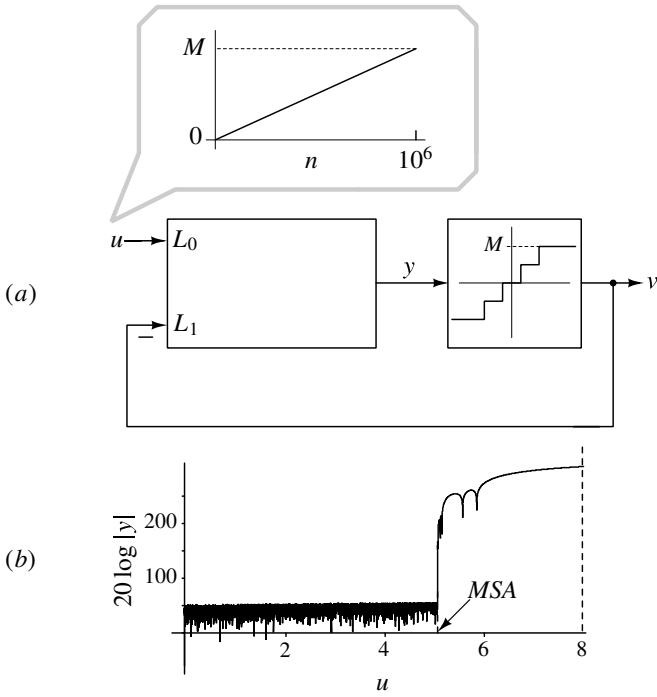
Figure 4.6   (a) Estimating MSA by exciting the $\Delta\Sigma$ modulator with a slow ramp. (b) $20\log|y|$ for a third-order loop with $NTF(z) = (1 - z^{-1})^3$, and a 9-level quantizer.

4.2   Improving MSA in High-Order Delta-Sigma Converters

In the previous section, we saw that $\Delta\Sigma$ loops with NTFs of the form $(1 - z^{-1})^L$ become unstable at relatively small fractions of the modulator's full scale. This happens since the large gain (2^L) of the NTFs at high frequencies greatly amplifies quantization noise as it circulates around the loop. As a consequence, the quantizer is overloaded even for small inputs, reducing its effective gain and thereby resulting in instability. How can we improve the input stable range, while still maintaining the order of noise-shaping?

Consider the magnitude response of the NTF $(1 - z^{-1})^3$, which has too much gain at high frequencies. This restricts the input range of the modulator, as discussed at length in the previous section. What we would like to do is to improve the MSA, while retaining the ω^3 noise-shaping at low frequencies. The MSA can only be improved by preventing quantizer overload, which has to be accomplished by reducing the NTF's high-frequency gain. This can be conceptually achieved by multiplying the NTF by a lowpass transfer function G (whose gain at dc is much larger than the gain at $\omega = \pi$), as shown in Figure 4.7.
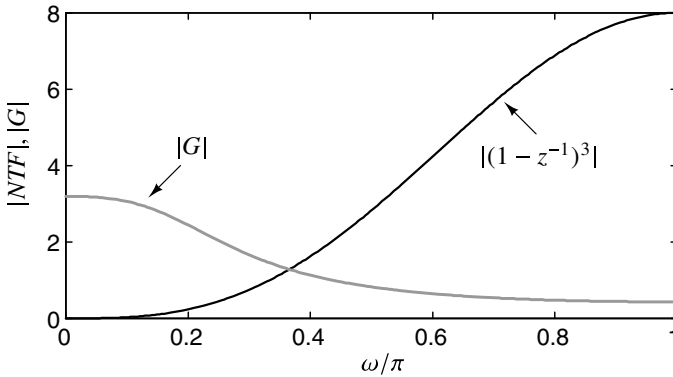
Figure 4.7   Magnitude responses of $(1 - z^{-1})^3$ and the general shape of a lowpass characteristic G. $|G \cdot NTF|$ has a lower gain at $\omega = \pi$, when compared to $|NTF|$.

What are the pole locations of such a lowpass filter? To get intuition about this, let us recall a graphical method of finding the magnitude response of a transfer function

$$H(z) = \frac{(z - z_1)}{(z - p_1)(z - p_2)} \tag{4.15}$$

at a complex frequency z, as shown in Figure 4.8. It is straightforward to see that $H(z)$ can be determined by drawing vectors r_1, r_2 and r_3 to z from p_1, p_2 and z_1, respectively, and using

$$H(z) = \frac{r_3}{r_1 r_2}. \tag{4.16}$$

Getting back to our problem, we assume that $G(z)$ is a second-order lowpass filter of the form $1 / \left[(1 - p_1 z^{-1})(1 - p_1^* z^{-1}) \right]$. The modified NTF is, therefore, $(1 - z^{-1})^3 G(z)$. The poles of $G(z)$ are located at p_1 and p_1^*. Why does the particular form for $G(z)$ make sense?
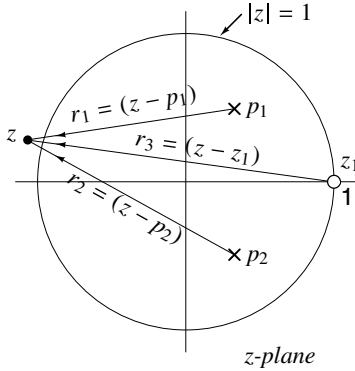
Figure 4.8   Geometrical interpretation of $H(z)$ for a transfer function of the form in (4.15).

It is needed to ensure that the modified NTF is physically realizable (which demands that $(1 - z^{-1})^3 G(z)$ evaluate to one as $z \to \infty$).

Since we need the gain of $G(z)$ to be higher at $z = 1$ compared to $z = -1$, it follows that p_1 (and p_1^*) should be closer to $z = 1$ when compared to $z = -1$, as shown in Figure 4.9(a). Thanks to the lowpass filter, the gain of the modified NTF at $\omega = \pi$ has now reduced – from 8 to $8(1/|r_\pi|)^2$. A higher order $G(z)$ can be used, in principle, to better
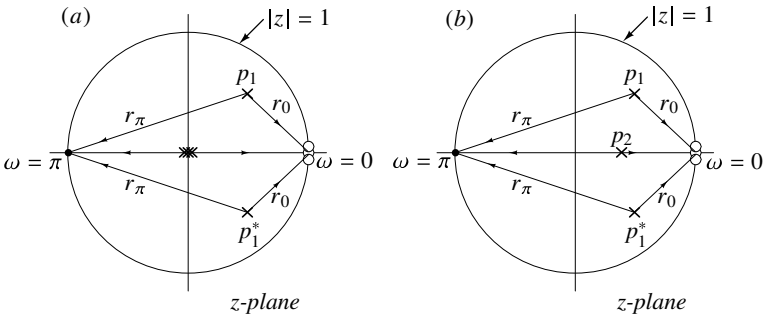
Figure 4.9   (a) Pole-zero map of $(1 - z^{-1})^3 G(z)$. For the gain at $\omega = \pi$ to be smaller than unity, p_1 and p_1^* must be closer to $z = 1$ than $z = -1$. (b) The lowpass filter can be realized without increasing order by moving the poles of the NTF to suitable positions in the z-plane.

shape the NTF.

What consequence does $G(z)$ have on the in-band performance? Since the poles p_1, p_1^* are close to $z = 1$, it follows that $G(1) = (1/|r_0|)^2 > 1$. Further, while the order of noise-shaping is preserved, the in-band NTF has increased from ω^3 to $k_1 \omega^3$, where $k_1 = (1/|r_0|)^2 > 1$. Thus, while the stable range has increased (due to a reduction of the NTFs high-frequency gain), it is accompanied by an increased in-band noise.

While the idea of multiplying an NTF by an auxiliary lowpass transfer function does enhance the stable input range, it carries with it the disadvantage of increasing the order of the modulator. This is easily remedied as described below. Rather than introduce new poles (p_1 and p_1^* in our example) to realize the lowpass filter as we have done, one could

move the three existing poles of the NTF (at $z = 0$) to appropriate locations in z-plane, as shown in Figure 4.9(b). In other words, the NTF is chosen to be of the form

$$NTF(z) = \frac{(1 - z^{-1})^N}{D(z)},$$ (4.17)

where $D(z)$ is such, that the NTF's poles are moved away from $z = 0$ to appropriate locations in the z-plane (close to $z = 1$). The transfer function of the lowpass filter is, therefore, $1/D(z)$, where the roots of $D(z)$ are chosen to reduce out-of-band gain, just like $G(z)$ earlier. The "no delay-free loop" condition dictates that $D(z)$ is of the form $D(z) = \prod_k (1 - z^{-1} p_k)$.

As discussed earlier in this section, the roots of $D(z)$ should be closer to $z = 1$ compared to $z = -1$ so that $|D(e^{j\pi})| > 1$. As a result, in-band noise-shaping is marginally degraded (as illustrated in Figure 4.10), but the result is an increased stable input range. The improvement in MSA depends, of course, on the actual pole positions chosen for the NTF. A variety of possibilities exist – we will explore some in the section to follow.
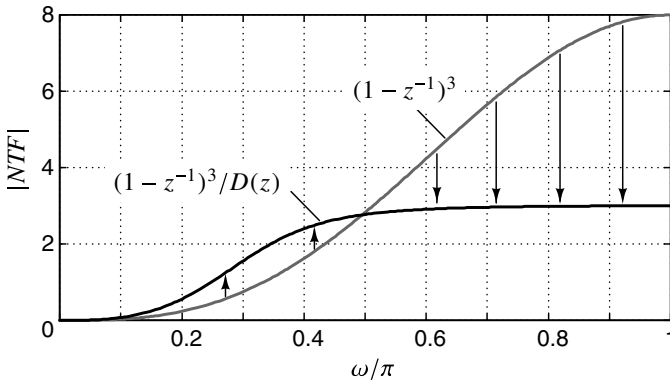
Figure 4.10   The lowpass filter reduces the NTF's gain at high frequencies at the expense of an increase in in-band quantization noise.

Let us now summarize some key points with regard to high-order $\Delta\Sigma$ loops.

a. Overloading the quantizer destabilizes the $\Delta\Sigma$ loop.

b. Since the quantizer input equals the modulator input plus shaped noise, it follows that the stable input range of the modulator *must be smaller* than the quantizer range.

c. More shaped noise in a stable modulator means a higher likelihood of instability and a reduced stable input range.

d. An NTF with more shaped noise (higher out-of-band gain) will also have less in-band noise. From the observations above, we see that an aggressive NTF (i.e., one that attempts to attenuate in-band noise to a larger degree) will have a smaller stable input range.

At this juncture, we would like to draw the attention of the reader to a recurring thread in all our discussions. Ponder this – when we compare MOD1 and MOD2, the

in-band performance of the latter was much better (ω^2 versus ω). However, the gain at $\omega = \pi$ was "worse" (4 versus 2). This trend continued as we conceived of high-order $\Delta\Sigma$ modulators with NTFs of the form $(1 - z^{-1})^L$ earlier in this chapter. In this section, when we tamed the high-frequency gain of a high-order NTF by moving pole positions, the in-band performance degraded! It seems as if the in-band and out-of-band performances of an NTF are always going in opposite directions: is this a coincidence with the NTFs we have been considering, or is there something fundamental lurking here? It turns out to be the latter, and we discuss this in Section 4.5.

Having gained intuition about the stability and NTF trade-offs associated with a high-order modulator, we now proceed to the topic of systematically choosing a noise transfer function. In other words, we attempt to answer the question: how does one go about determining an NTF that achieves a desired in-band SQNR?

4.3 Systematic NTF Design

As we have seen earlier, an NTF is a highpass transfer function that must be designed so as to achieve the desired in-band SQNR. In a typical application, the signal bandwidth, sampling rate (usually constrained by the system) and the desired in-band SQNR are known. Increasing the number of levels in the quantizer has practical implementation difficulties, and designers usually restrict the resolution of the quantizer to 16 levels. How do we go about the design? This is best illustrated by an example. Let us aim to design a third-order NTF with $OSR = 64$ that achieves better than 115 dB peak SQNR with a 16-level quantizer. We proceed in the following stepwise fashion [4].

a. We pick a prototype highpass filter family - one can, here, borrow from the rich literature on IIR digital filters. Commonly used filters belong to the Butterworth, Chebyshev, inverse Chebyshev, and elliptic families. A practical advantage to using these "readymade" highpass filters, as opposed to inventing one's own, is that coefficients for these approximations are readily obtained from MATLAB. In this example, we (arbitrarily) pick a Butterworth filter with a 3-dB corner of $\pi/8$. Recall that a Butterworth design is fully specified by its corner frequency.

b. We obtain the transfer function from MATLAB. The relevant code fragment and the output are the following:

```
[b,a]=butter(3,1/8,'high')
```

$$H(z) = \frac{0.6735 - 2.0204z^{-1} + 2.0204z^{-2} - 0.6735z^{-3}}{1 - 2.2192z^{-1} + 1.7151z^{-2} - 0.4535z^{-3}}.$$

As per standard practice, the filter coefficients are scaled so that the passband gain of the high-pass filter is 1. A question to ponder is the following. An NTF is a highpass filter, but can any highpass transfer function be an NTF? In other words, how do we know that the $H(z)$ above is a valid NTF? We have addressed this question before – any physically realizable NTF, when evaluated at $z = \infty$ must reduce to unity. This is the frequency domain constraint imposed by the "no delay-free loop" rule.

In our example above, $H(z = \infty) = 0.6735$, indicating that it is not physically implementable. It should therefore be scaled by $1/0.6735$. The resulting NTF is given

by

$$NTF(z) = \frac{(1 - 3z^{-1} + 3z^{-2} - z^{-3})}{1 - 2.2192z^{-1} + 1.7151z^{-2} - 0.4535z^{-3}}. \tag{4.18}$$

The (constant) gain of the NTF at high frequencies is called the out-of-band gain (OBG). In our example, the OBG = $1/0.6735 = 1.48$. Figure 4.11 shows the magnitude responses of $H(z)$ and $NTF(z)$.

How does the in-band gain of the NTF in (4.18) compare with that of $NTF(z) = (1 - z^{-1})^3$? As $z \to 1$, the denominator of (4.18) evaluates to 0.0424: the in-band gain of our NTF is $\omega^3/0.0424$, which is about 24 times higher than that of $(1 - z^{-1})^3$. This is not surprising, as the OBG is much smaller.
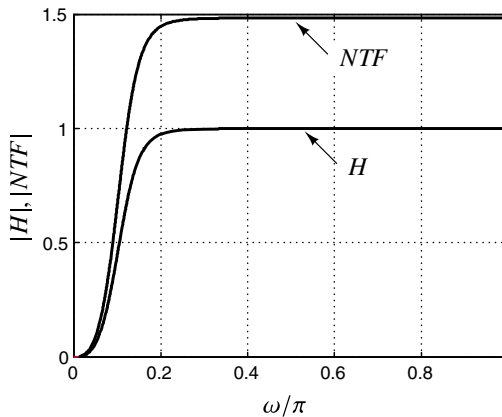
Figure 4.11   Magnitude responses of H and NTF. The OBG of the NTF is 1.48, and the resulting SQNR is about 102 dB.

c. Next, we find the transfer function of a loop filter using $1/(1 + L_1(z)) = NTF(z)$. The result is

$$L_1(z) = \frac{0.7808z^{-1} - 1.285z^{-2} + 0.5465z^{-3}}{1 - 3z^{-1} + 3z^{-2} - z^{-3}}. \tag{4.19}$$

One way of realizing the $\Delta\Sigma$ modulator with the desired NTF is to use the special case of Figure 4.1 where $L_0 = L_1 = L$, as shown in Figure 4.12.

d. We then simulate the equations describing the modulator, determine the MSA, and thereby the peak SQNR. In our example, we obtain the MSA to be about 85% of full scale, and a peak SQNR of 102 dB, a good 13 dB short of our design goal.

e. Since the SQNR is not adequate, we conclude that the highpass filter is not doing a sufficiently good job of attenuating low frequencies. The cutoff frequency of the Butterworth highpass filter should, therefore, be increased. In our example, we increase the 3-dB corner of the filter from $\pi/8$ to $\pi/4$. Going through step (b) above, we see that the OBG of the resulting NTF will be higher than that we had earlier. Since the OBG increases, the MSA should reduce.
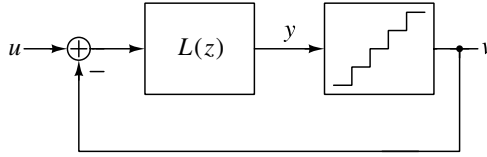
Figure 4.12   A $\Delta\Sigma$ modulator structure where $L_o = L_1 = L$.

f. After completing steps (c) and (d), we find that the OBG and MSA are 2.25 and 80% of full-scale, respectively, and that the peak SQNR is about 116 dB. We have thereby achieved our goal.

4.4 Noise Transfer Functions with Optimally Spread Zeros

In the preceding sections, we dealt with NTFs of the form $(1 - z^{-1})^N/D(z)$, where all the zeros of the NTF occur at $z = 1$. In the signal band, $|NTF|^2 \approx k_1\omega^{2N}$, where, as seen earlier, $k_1 > 1$.

$$\text{In-band noise} = \frac{\Delta^2}{12\pi}\int_0^{\frac{\pi}{OSR}} k_1\omega^{2N}\,d\omega. \tag{4.20}$$
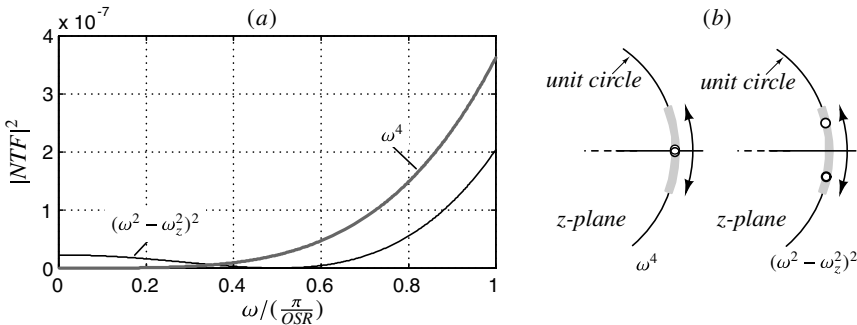
Figure 4.13   (a) Squared magnitudes of second-order NTFs with both zeros at $z = 1$ and with optimized zeros, and (b) the corresponding zero locations in the z-plane.

Figure 4.13(a) shows the squared magnitude of a second-order NTF in the signal band. It is apparent that most of the contribution to the in-band noise is from frequencies around the band edge. We can do better by making the NTF zeros complex (of the form $e^{\pm j\omega_z}$), as shown in Figure 4.13. The corresponding squared magnitude response is now given by $k_1(\omega^2 - \omega_z^2)^2$, as seen in Figure 4.13(a). But what ω_z should one use for best results? The optimal ω_z is the one that minimizes the integral

$$\frac{\Delta^2}{12\pi}\int_0^{\frac{\pi}{OSR}} k_1(\omega^2 - \omega_z^2)^2\,d\omega. \tag{4.21}$$

Straightforward analysis shows that the optimum corresponds to

$$\omega_z = \frac{\pi}{OSR}\frac{1}{\sqrt{3}}. \tag{4.22}$$

The in-band noise is lower by 3.5 dB.

From the discussion above, it is to be expected that even greater benefits can be obtained by optimizing the location of the zeros of higher-order NTFs. The principle of optimization, however, remains the same: 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.

The resulting values for the zeros (normalized to the signal band limit) are given, for NTFs with orders from 1 to 8, in Table 4.2. 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, e.g., for 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.

Order	Zero Locations Relative to Band Edge	SQNR Improvement (dB)
1	0	0
2	$\pm 1/\sqrt{3} = \pm 0.577$	3.5
3	$0, \pm \sqrt{3/5} = \pm 0.775$	8
4	$\pm 0.340, \pm 0.861$	13
5	$0, \pm 0.539, \pm 0.906$	18
6	$\pm 0.23862, \pm 0.66121, \pm 0.93247$	23
7	$0, \pm 0.40585, \pm 0.74153, \pm 0.94911$	28
8	$0, \pm 0.18343, \pm 0.52553, \pm 0.79667, \pm 0.96029$	34

Table 4.2   Zero placement for minimum in-band noise.

4.5 Fundamental Aspects of Noise Transfer Functions

So far in this book, we have been exposed to NTFs of the form $(1 - z^{-1})^N/D(z)$, as well as those with optimized zeros. We saw, repeatedly, that good in-band performance was accompanied by high out-of-band gain. Was this incidental, or is there something more fundamental lurking here? It turns out to be latter, as the alert reader would suspect. In this section, we examine this correlation in more detail.

4.5.1 The Bode Sensitivity Integral

Before we discuss NTFs of $\Delta\Sigma$ modulators, let us review the following facts from control theory. Consider the discrete-time feedback system shown in Figure 4.14, with input x and output v. e is a disturbance injected at the output of the loop filter L.
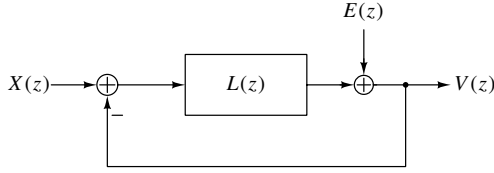
Figure 4.14   Sensitivity of a feedback loop is defined as the transfer function from the disturbance e to the output y.

$$V(z) = \frac{L(z)}{1 + L(z)} X(z) + \frac{1}{1 + L(z)} E(z). \tag{4.23}$$

If $L(z) = \infty$, $V(z) = X(z)$ and the loop rejects $E(z)$. In other words, the loop is *insensitive* to $E(z)$. Since $L(z)$ cannot be ∞ at all frequencies, the loop can reject the disturbance e effectively only at frequencies where the loop gain is high. The *sensitivity function*, defined as

$$S(e^{j\omega}) = \frac{1}{1 + L(e^{j\omega})} \tag{4.24}$$

quantifies how effectively the loop rejects e. In a $\Delta\Sigma$ loop, the sensitivity is the same as the NTF. From our discussions earlier, the first sample of the impulse response corresponding to $NTF(z)$ has to be unity. In the frequency domain, this is equivalent to $NTF(\infty) = 1$. Further, since the numerator and denominator polynomials of the NTF can be expressed as a product of first- and second-order factors, $NTF(z)$ can be written as

$$NTF(z) = \frac{(1 + b_1 z^{-1})(1 + b_2 z^{-1} + b_3 z^{-2}) \cdots}{(1 + a_1 z^{-1})(1 + a_2 z^{-1} + a_3 z^{-3}) \cdots}. \tag{4.25}$$

For a stable modulator, the poles must lie inside the unit circle. The zeros of the NTF lie on the unit circle. It is easily shown (by integrating $\log[z/(z + a_1)]$ on the unit circle) that if $|a_1| \leq 1$,

$$\int_0^\pi \log(|1 + a_1 e^{-j\omega}|)\, d\omega = 0. \tag{4.26}$$

Equation (4.26) tells us that the area above zero in the log magnitude plot of $(1 + a_1 e^{-j\omega})$ is equal to the area below zero, as shown in Figure 4.15. Having accepted (4.26), it is straightforward to see that

$$\int_0^\pi \log(|1 + a_2 e^{-j\omega} + a_3 e^{-j2\omega}|)\, d\omega = 0 \tag{4.27}$$

if the roots of $1 + a_2 z^{-1} + a_3 z^{-2}$ lie within (or on) the unit circle.

The NTF can be expanded as a ratio of first- and second-order polynomials as shown in (4.25), and the integral of the log magnitude of the NTF is seen to be

$$\int_0^\pi \log|NTF(e^{j\omega})|\, d\omega = \int_0^\pi \log\left|\frac{(1 + b_1 e^{-j\omega})(1 + b_2 e^{-j\omega} + b_3 e^{-2j\omega}) \cdots}{(1 + a_1 e^{-j\omega})(1 + a_2 e^{-j\omega} + a_3 e^{-3j\omega}) \cdots}\right|\, d\omega = 0. \tag{4.28}$$

Since the NTF in a $\Delta\Sigma$ loop is the same as the sensitivity of the feedback loop, the equation above is equivalent to
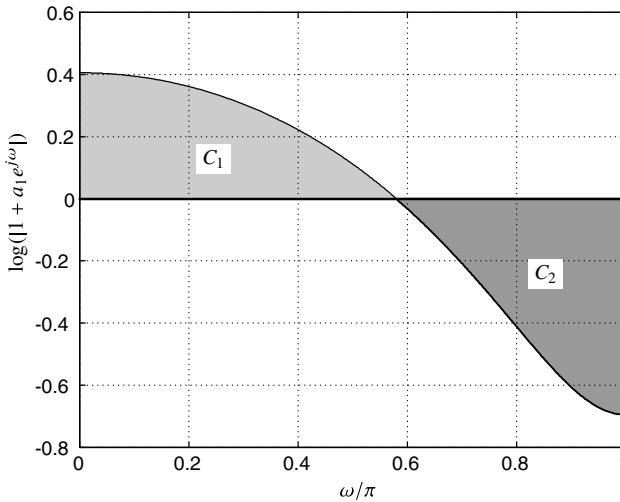
Figure 4.15   For $|a_1| < 1$, the area above zero (C_1) and that below zero (C_2) are equal.

$$\int_0^\pi \log(|S(e^{j\omega})|)d\omega = 0. \tag{4.29}$$

This integral relationship, well known in control theory, is called the *Bode sensitivity integral*. This is a direct consequence of the infeasibility of physically realizing a delay-free feedback loop [5].

Thus, in a $\Delta\Sigma$ modulator, good in-band performance can *only* be obtained at the expense of poor out-of-band performance. Any attempt to reduce the in-band gain leads to an increased gain at high frequencies.

The Bode sensitivity integral gives us a different kind of insight into why using a high-order NTF is more effective in reducing in-band quantization noise. Figure 4.16 shows the log magnitudes of two NTFs with the same out-of-band gain, but with different orders. The signal bandwidth is marked with the dashed line. A higher order NTF enables a narrower transition band. This increases the area above the $0\,$dB line, which means that more "negative area" is available. Further, since the transition band is narrower, it follows that less of the negative area is "wasted" in this band. This allows the in-band gain to be lower, resulting in better in-band performance.

4.6   High-Order Single-Bit Delta-Sigma Data Converters

So far in this book, we have seen that combining oversampling with negative feedback can dramatically enhance the effective resolution of the coarse quantizer embedded in the loop. Several possible approaches can yield the desired SQNR in the signal bandwidth. One can reduce the number of quantizer levels, while increasing the sampling rate (equivalent to operating at a higher OSR), or by increasing the order of noise-shaping. The advantage of using a quantizer with fewer levels is the reduction in the hardware required to realize the
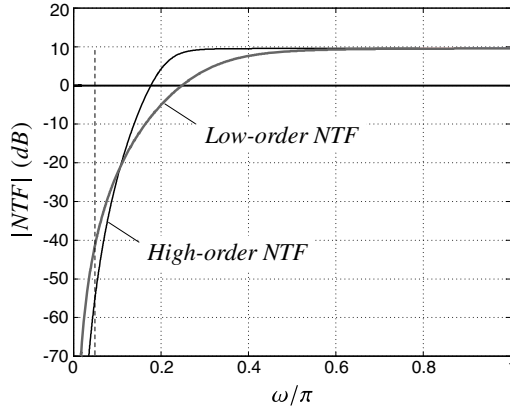
Figure 4.16   High order NTFs have a larger "positive area" and a sharper transition band.

modulator. The simplest quantizer that can be used is one that has two levels, also called the 1-bit quantizer.

Apart from being very easy to implement, a single-bit quantizer is inherently linear in the sense discussed in Section 2.4.1, where errors in the threshold or levels cause (benign) offset and gain errors in the loop. Since the quantizer's output is simply the sign of its input, it follows that the loop filter's output can be scaled without affecting the modulator's output, as shown in Fig. 4.17. It turns out that this can potentially simplify the design of operational amplifiers used in the integrators of the loop filter.
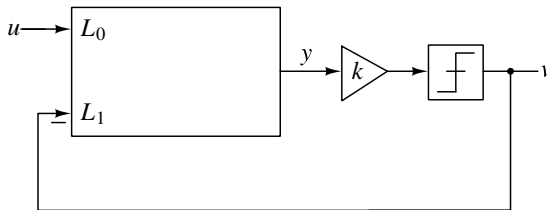
Figure 4.17   The output sequence of a single-bit modulator is not affected by scaling the loop filter output by a positive constant k.

What should we expect from past experience based on the additive quantization noise model? When compared with a multi-level design, we should expect to use a larger OSR to achieve the same in band SQNR, since the error added by a single-bit quantizer is much higher. For the same reason, the MSA should reduce in the two-level case, when compared to a multibit loop with the same NTF. Further, the error waveform that needs to be processed by the loop filter is much larger in magnitude. This means that, to achieve the same level of overall performance, the loop filter of such a modulator has to be lot more linear than its counterpart in a multi-bit design.

The conclusions above regarding single-bit modulators were based on the intuition we developed from our study of multi-bit $\Delta\Sigma$ modulation. While these are largely valid, one should not be surprised if a single-bit modulator behaves in unexpected ways: after all, a 1-bit quantizer is *always* saturated, which means that the additive white quantization noise model is particularly questionable.

In our discussion on the stability of a multibit modulator, we saw that the NTF is all one needs to know to understand its stability. In the case of a binary (1-bit) modulator, an important question is "What NTF properties are necessary and sufficient for stable operation?" Unfortunately, a simple and exact answer to this question is not known. The proven results are generally either too restrictive (too conservative), or apply only for specific modulators with constant inputs. A widely-used approximate criterion is the (modified) Lee's rule [6, 7].

> A 1-bit modulator is likely to be stable if $\max_{\omega}(|NTF(e^{j\omega})|) < 1.5$.

The quantity $\max_{\omega}(|NTF|)$ is the maximum gain of the NTF over all frequencies, also known as the *infinity-norm* of *NTF*, for which the mathematical notation is $\|NTF\|_{\infty}$. In the original statement of the condition, the limit on $\|NTF\|_{\infty}$ was given as 2, but as experience with higher order modulators was gained, the rule of thumb was revised to use a limit of 1.5. For moderate-order modulators (order 3 or 4), slightly higher values may be tolerable, while for very high-order modulators (7 or more) a more conservative $\|NTF\|_{\infty} = 1.4$ may be more appropriate.

Note that this criterion is neither necessary (as we have seen in the stable MOD2 with $NTF(z) = (1 - z^{-1})^2$, where $\|NTF\|_{\infty} = 4$ was allowed) nor sufficient (the criterion says nothing about a limit on the input signal). Nevertheless, due to its simplicity, it is of some use. Although Lee's rule is a helpful rule of thumb for predicting a priori instabilities in single-bit modulators, it has no solid theoretical foundations, and needs to be confirmed by extensive simulations.

Note that the maximum of $NTF(e^{j\omega})$ usually occurs at $\omega = \pi$, since this point is farthest from the zeros (which are clustered around $z = 1$) and closest to the poles. An exception may occur if the $NTF(z)$ has high-Q poles, in which case the peak value may occur near the dominant (highest Q) pole.
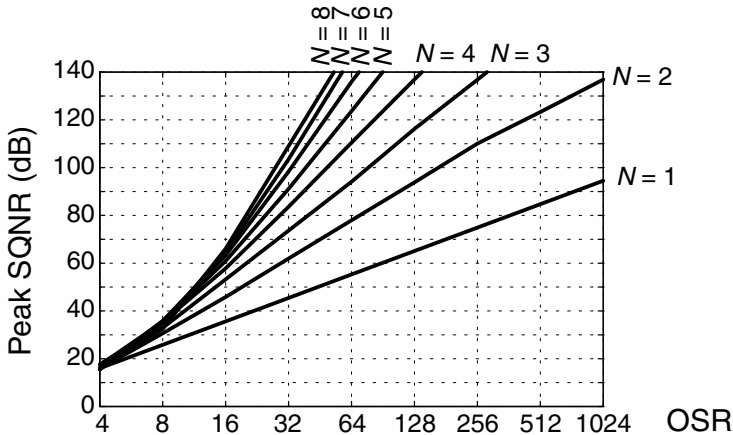
Figure 4.18   Empirical SQNR limit for 1-bit modulators of order N.

Figures 4.18 through 4.20 provide some additional design information [8]. These curves show the achievable peak signal-to-quantization-noise ratio (SQNR) for modulators of orders $N = 1 - -8$ employing optimal zero placement, with 1- to 3-bit internal quantization. The curves include the effects of the reduction of the input u necessary to
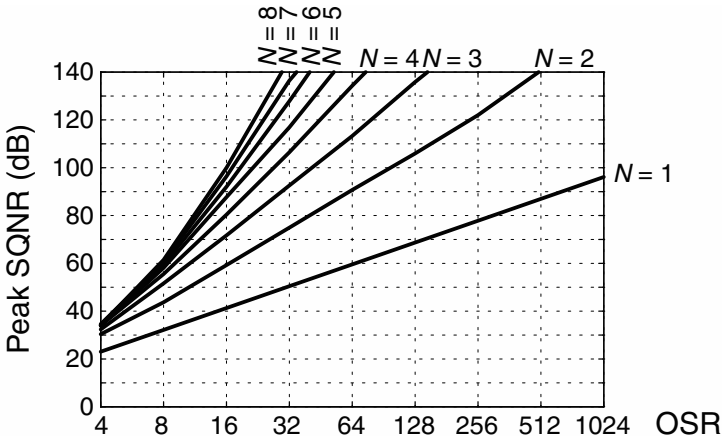
Figure 4.19 Empirical SQNR limit for modulators with 2-bit quantizers of order N.
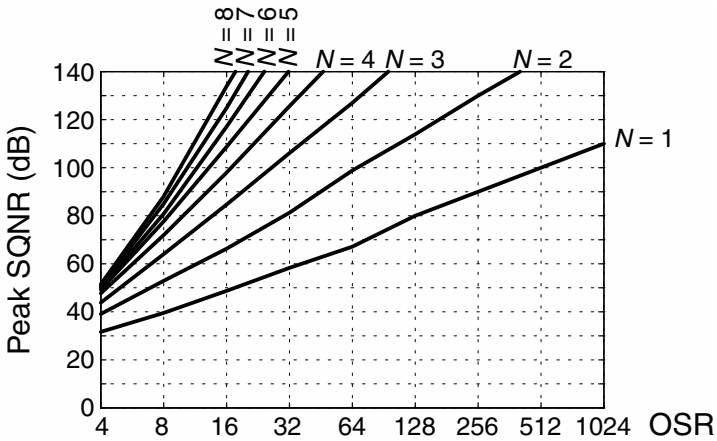
Figure 4.20 Empirical SQNR limit for modulators with 3-bit quantizers of order N.

satisfy the stability conditions. Hence, they accurately predict the actual performance of the real (nonlinear as opposed to linearized) modulator.

4.7   Loop Filter Topologies for Discrete-Time Delta-Sigma Converters

Having understood various trade-offs regarding the choice of the NTF, it is time to implement the loop filters $L_0(z)$ and $L_1(z)$ in Figure 4.2. In this section, some basic configurations will be described: while some of these can be considered as straightforward generalizations of the second-order modulator MOD2 discussed in Chapter 3, others are different. Since a desired NTF may be realized in many ways, this begs the question: which topology should one choose, and why? We attempt to throw light on this below [4, 9, 10].

4.7.1   Loop Filters with Distributed Feedback: The CIFB and CRFB Families
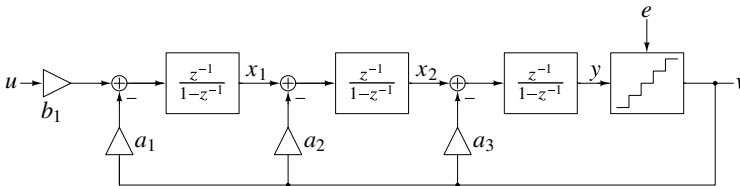
Figure 4.21   A third-order NTF realized as a cascade of integrators with feedback (CIFB) structure. All NTF zeros are at $z = 1$.

Figure 4.21 shows a third-order $\Delta\Sigma$ modulator derived along the same lines as that in Figure 4.1. The loop-filter consists of a cascade of three delaying integrators, with the quantizer output fed back into each of the integrators with different weight factors. This is the CIFB structure, and it is easily extended to higher orders. It is straightforward to see that

$$L_0(z) = \frac{b_1 z^{-3}}{(1 - z^{-1})^3} \ , \quad L_1(z) = \frac{a_1 z^{-3}}{(1 - z^{-1})^3} + \frac{a_2 z^{-2}}{(1 - z^{-1})^2} + \frac{a_3 z^{-1}}{(1 - z^{-1})}. \tag{4.30}$$

L_0 and L_1 have three poles at dc ($z = 1$). The NTF and STF are given by

$$NTF(z) \quad = \quad \frac{(1 - z^{-1})^3}{(1 - z^{-1})^3 + a_3 z^{-1}(1 - z^{-1})^2 + a_2 z^{-2}(1 - z^{-1}) + a_1 z^{-3}}, \tag{4.31}$$

$$STF(z) \quad = \quad \frac{b_1 z^{-3}}{(1 - z^{-1})^3 + a_3 z^{-1}(1 - z^{-1})^2 + a_2 z^{-2}(1 - z^{-1}) + a_1 z^{-3}}. \tag{4.32}$$

The denominators of both transfer functions are identical, since they are associated with the same system. The NTF satisfies the condition $NTF(z = \infty) = 1$, necessary for physical realizability. It has three zeros at dc, corresponding to the three dc poles of $L_1(z)$. $a_1, \cdots, a_3$ are chosen to achieve the desired poles (as dictated by the NTF).

The dc gain of the STF, given by $STF(1)$, is (b_1/a_1). While this is apparent from the equations above, the gain can be inferred intuitively from Figure 4.21 as follows. In *any*

stable negative feedback loop, the average input of any integrator should be zero. Why? If this was not true, the nonzero dc would keep accumulating, causing the integrator's output to reach infinity, thereby contradicting our assumption of a stable system. Applying this principle to the first integrator of Figure 4.21, we see that at dc, $b_1\bar{u} = a_1\bar{v}$, resulting in a dc gain of (b_1/a_1).

Provided that b_1 and a_1 are chosen to be equal so that the dc gain of the STF is unity, and provided that the input u is slowly varying, $Y(z) \approx U(z) + (NTF(z) - 1)E(z)$. y, as we have seen before, consists of shaped quantization noise riding on the input. When the amplitude of u is equal to the MSA, the peak-to-peak swing of y is approximately the no-overload range of the quantizer. Under these circumstances, what are the swings at the first and second integrator outputs (x_1 and x_2)?

> ### Example: Third-order maximally-flat NTF with OBG = 2.25
>
> The NTF (found using the techniques of Section 4.3) is given by
>
> $$NTF = \frac{(1 - z^{-1})^3}{1 - 1.467z^{-1} + 0.8917z^{-2} - 0.1967z^{-3}}.$$
>
> Equating coefficients of like powers of z^{-1} of the denominator of the NTF above to those in the general form of equation (4.31), we obtain $a_1 = 0.228$, $a_2 = 0.957$, and $a_3 = 1.533$.
>
> How does the in-band RMS quantization noise of this NTF compare with that obtained with an NTF of $(1 - z^{-1})^3$? As seen from Figure 4.21, a_1 represents the gain of the "triple integral" path of $L_1(z)$ – that is, it is the coefficient of $z^{-3}/(1 - z^{-1})^3$. At low frequencies, therefore, the NTF, which is $1/(1 + L_1(z))$, is approximately $1/L_1(e^{j\omega}) \approx \omega^3/a_1$. It is thus seen that the magnitude of the NTF in the signal band is mostly dictated by the loop filter path with the highest order of integration. In our example, reducing the OBG to 2.25 has resulted in an NTF whose low-frequency gain (and in-band RMS noise) is larger by a factor of $1/a_1 = 4.4$.
>
> One might wonder why the second- and first-order paths of $L_1(z)$ are necessary at all, if the in-band NTF only depends on a_1. These paths are needed to stabilize the negative-feedback loop. We also see that a_3 is larger than a_2, and significantly larger than a_1. Why does this make intuitive sense? The loop gain around every stable negative-feedback loop consists of a "quick and dirty" fast path, which is necessary to give the loop a good sense of gross errors, and make course corrections. The "precision path", which is needed to ensure steady state accuracy (which in the $\Delta\Sigma$ context translates to the low frequency gain of the NTF), should have high dc gain and is achieved by cascading integrators. Due to the cascade, the high-order path is necessarily slow. Without a fast path that provides a sufficient amount of quick feedback, the high gain but slow path will cause instability. This means that the first-order path should have a sufficiently large gain. This is why a large a_3 (which is the gain of the first-order path) makes sense.

To determine this, we use superposition. If only u were present (with $e = 0$), $v = u$. x_1 and x_2 should be a_2u and a_3u, respectively. Why? Recall that the dc input to any integrator in a stable negative-feedback system must be zero. Extending this argument, the low-

frequency component of any integrator's input should be very small. Thus, $x_1 = a_2v = a_2u$ and $x_2 = a_3v = a_3u$, since $v = u$ in the absence of e. With only e present, v is e shaped to third order. x_1 being the accumulated version of v, is second-order shaped. x_2 is comprised of first- and second-order shaped noise. By a similar token, $x_3(= y)$ consists of all orders of shaped noise. In summary,

$$x_1 = a_2u + \textit{second-order shaped noise,}$$
$$x_2 = a_3u + \textit{first- and second-order shaped noise,}$$
$$x_3 = y = u + \textit{all orders of shaped noise.}$$

Since u is a sinusoid with the maximum stable amplitude, the peak-to-peak swing of y is virtually the no-overload range of the quantizer. In practice, the latter is chosen to be the largest range that can "fit" in the supply voltage used to design the modulator. This choice makes sense, as it maximizes the step size of the ADC and simplifies its design. The MSA is typically a large fraction (about 85%) of the quantizer range. This means that implementing the $\Delta\Sigma$ modulator as in Figure 4.21 can potentially cause the range of x_1 and x_2 to exceed that of the quantizer (which we assumed to be the maximum possible in the given supply voltage). In our example modulator above, with $a_3 = 1.53$ and $a_2 = 0.95$, and with a sinusoidal input having amplitude of 85% of full-scale, simulations show that x_2 will swing much more than x_3 does ($x_{2,max} \approx 21$ and $x_{3,max} \approx 15$).
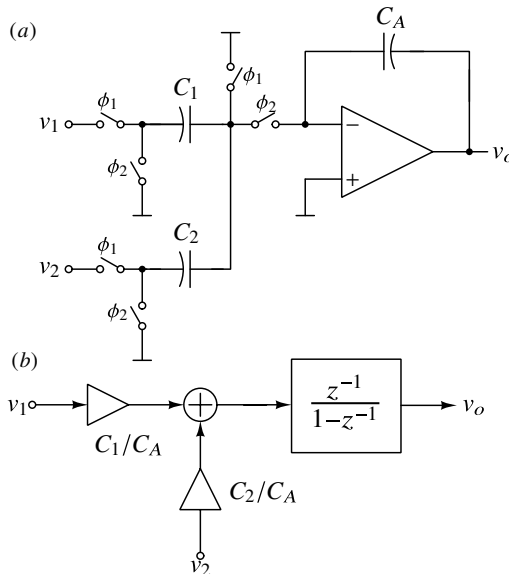
Figure 4.22   (a) A summing, delaying switched-capacitor integrator; (b) equivalent macro model.

Large internal swings are problematic in practice, as they cause the opamps realizing the integrators to saturate, which severely degrades performance, and may even destabilize the modulator. To prevent premature saturation of internal states, they must be *scaled* without affecting the transfer functions L_0 and L_1 of the loop filter. This process is called *dynamic-range scaling*. We describe this in more detail below, with special reference to discrete-time integrators realized using switched capacitors.

Figure 4.22(a) shows a single-ended delaying switched-capacitor integrator that performs weighted addition of inputs v_1 and v_2. The transfer functions of the integrator from v_1 and v_2 to v_o are $(C_1/C_A)z^{-1}/(1-z^{-1})$ and $(C_2/C_A)z^{-1}/(1-z^{-1})$, respectively. The integrator's macro-model is shown in Figure 4.22(b). Thus, the output can be scaled by varying C_A, or C_1 *and* C_2. Analysis (see Chapter 7) shows that the input paths are corrupted by noise whose mean square value is inversely proportional to C_1 and C_2, respectively.
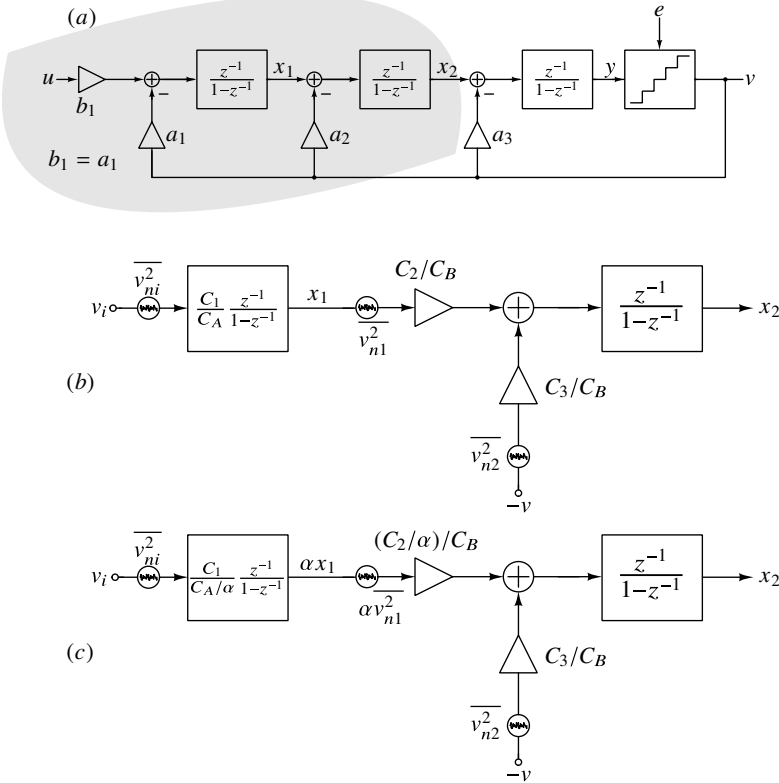
Figure 4.23   (a) Portion of a signal flow graph; (b) x_1 is scaled by α, but transfer functions from v_i to x_2 and x_3 remain unchanged.

Consider the portion of the modulator of Figure 4.23(a) shown in gray. If the dc gain of the modulator's STF is chosen to be unity, $b_1 = a_1$. The input integrator, then, processes $(u - v)$, which we denote by v_i. The gain $a_1(= b_1)$ can subsequently be pushed through the summer. The macro-model of the resulting circuit built using switched-capacitor integrators is shown in Figure 4.23(b). $\overline{v_{ni}^2}$ represents the input-referred mean square noise of the first integrator. C_1/C_A implements b_1. The second integrator processes the weighted sum of x_1 and v. The input capacitors of this integrator are denoted by C_2 and C_3, and the integrating capacitor is denoted by C_B. The input referred thermal noise of the two input paths are represented by $\overline{v_{n1}^2}$ and $\overline{v_{n2}^2}$.

Suppose that we wish to scale x_1 by a factor α, while at the same time ensure that the transfer functions from v_i and v to x_2 do not change. Reducing the feedback capacitor

of the first integrator C_A by a factor α scales x_1 by α. To keep x_2 the same, the input capacitor C_2 of the subsequent block (that senses x_1) should be reduced by α as shown in Figure 4.23(c).

Scaling x_1 has interesting consequences for noise at x_2. Since C_2 has reduced, $\overline{v_{n1}^2}$ increases by α, but the transfer function from the noise source to x_2 has reduced by the same factor. Thus, the contribution of v_{n1} to the mean square noise at v_o has *reduced* by α. Scaling x_1 by $\alpha > 1$, therefore, has the desirable consequence of reducing noise, as well as the total capacitance in the network. If α is too large, however, the first integrator will saturate – something that should be avoided. It is thus seen that α should be chosen to be as large as possible without causing saturation. To achieve a low noise and an area-efficient design, this exercise should be applied to all opamp outputs. As mentioned earlier, this procedure is called *dynamic-range scaling*, and it is an essential part of the design process.
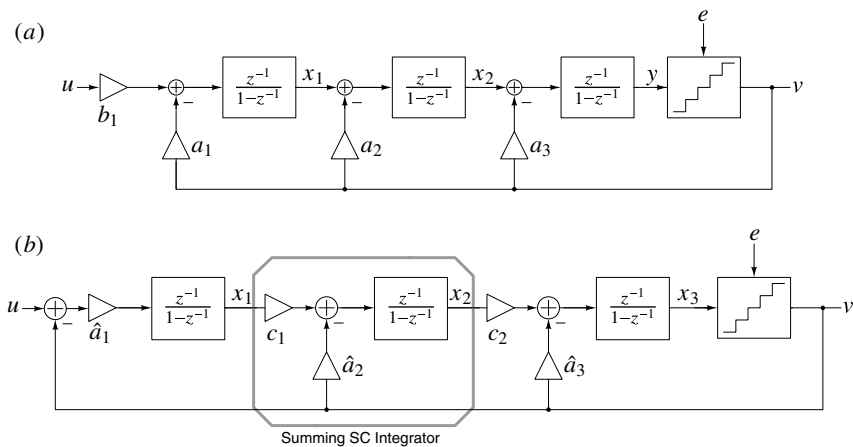
Figure 4.24   (a) Prototype CIFB modulator and (b) incorporating dynamic-range scaling. The dc gain of the STF is constrained to be unity.

Figures 4.24(a) and (b) show the structure of the CIFB modulator of Figure 4.21, without and with dynamic-range scaling. In the latter, scaling is achieved through coefficients c_1 and c_2. The STF at dc is unity. Figure 4.25(a) shows the integrator outputs of the CIFB modulator of Figure 4.24(a). The quantizer is assumed to have 16 levels (with step-size of 2). The input is a sinusoid with an amplitude 80% of full-scale. As we deduced earlier, every integrator output has an input component. We see that x_2 can saturate in a practical realization. After dynamic-range scaling (see Figure 4.25(b)), where x_1 and x_2 were restricted to a peak magnitude of 12, there is no danger of saturation.

What is the STF of the modulator of Figure 4.24(b)? By inspection, $L_0(z) = \hat{a}_1 c_1 c_2 z^{-3}/(1 - z^{-1})^3$. Thus,

$$STF(z) = \frac{L_0(z)}{1 + L_1(z)} = \frac{\hat{a}_1 c_1 c_2 z^{-3}}{D(z)}, \tag{4.33}$$

where $D(z)$ is the denominator polynomial of the NTF. As discussed earlier in this chapter, $1/D(z)$ is a lowpass response, chosen so as to "tame" the high-frequency gain of $(1 - z^{-1})^3$. Thus, the magnitude response of the STF must have a lowpass shape whose details are
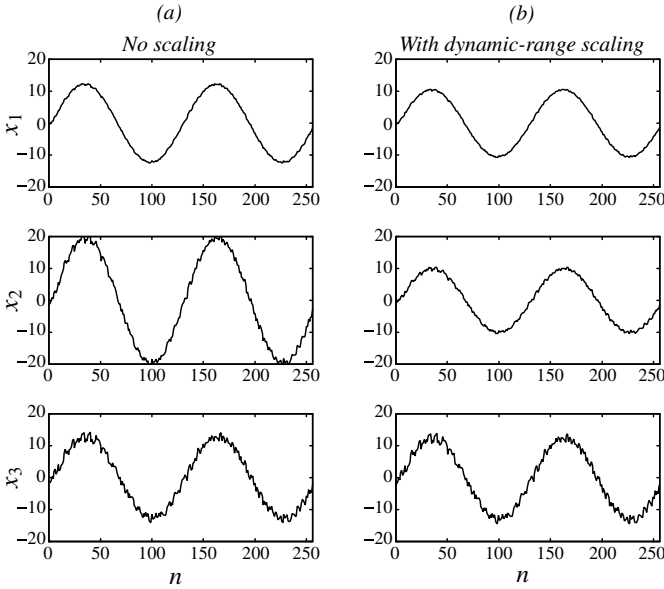
Figure 4.25 Integrator outputs of the CIFB modulator of Figure 4.21. (a) Before dynamic-range scaling and (b) after dynamic-range scaling. The input is a sinusoid with an amplitude 80% of full-scale.

dictated by the particular choice of $D(z)$. It is important to see that once the architecture (CIFB in this case) and NTF are frozen, there is no freedom with respect to the STF – one has to accept whatever shape results from (4.33).

What are the problematic aspects of a CIFB design that motivate alternative methods of realizing the loop filter? First, a CIFB loop has multiple feedback paths into the loop filter. This means that an Nth order $\Delta\Sigma$ modulator will need N feedback DACs. Another problem with the CIFB architecture is area efficiency, particularly so when a quantizer with many levels is used. The intuition behind this is the following. Consider the case of a dc input u, chosen to be slightly less than the maximum stable amplitude of the modulator. Since the number of quantizer levels is large, the peak-to-peak shaped noise is small compared to u, and is neglected in this analysis. Referring to Figures 4.24(a) and (b), we see that a_3 and $\hat{a}_3$ should be equal. Since x_2 has been scaled so that its peak swing is the same as that of x_3, and neglecting shaped quantization noise riding on x_2, c_2 should equal $\hat{a}_3$ to ensure that the dc input into the succeeding integrator is zero. Since $c_2 = \hat{a}_3 = a_3$, the gain of the double integration path in L_1 is $\hat{a}_2\hat{a}_3$, which should equal a_2. Thus $\hat{a}_2 = a_2/a_3$. Reasoning along similar lines as above, $c_1 = \hat{a}_2 = a_2/a_3$ and $\hat{a}_1 c_1 c_2 = a_1$, resulting in $\hat{a}_1 = a_1/a_2$. Recall that $\hat{a}_1$ *has* to be much smaller than unity – this is a consequence of stabilizing the NTF, as we saw in Section 4.2. As a consequence, $\hat{a}_1$ is *necessarily* small.

As will be seen in Chapter 7, the input referred noise of the $\Delta\Sigma$ modulator is largely dictated by the size of the input capacitor used in the first integrator, with a high-resolution design demanding the use of a large input capacitor. Therefore, a small $\hat{a}_1$ necessitates an even larger integrating capacitor, thereby greatly increasing the area occupied by the modulator. Another undesirable consequence of the small $\hat{a}_1$ in the first integrator is that noise and distortion added further down in the loop filter are not adequately attenuated

when referred to the modulator's input. Moreover, the large input components present at the output of every integrator cause harmonic distortion in the $\Delta\Sigma$ modulator's output due the unavoidable nonlinearities of the integrators.
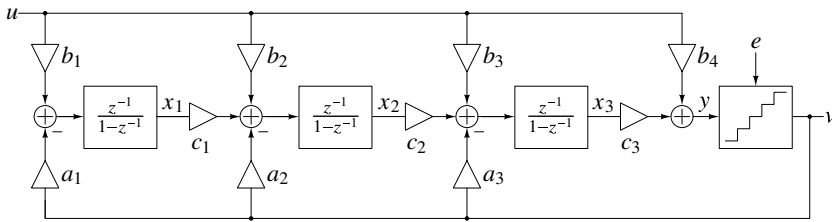
Figure 4.26   A CIFB structure with feedforward.

Having established "states consisting of the input component" as the root cause of the CIFB modulator's problems, several ways of keeping the states free of the input can be conceived. One of them is to add feedforward from u to the output of every integrator, as shown in Figure 4.26. Since y is comprised of u and shaped noise, the loop filter can be spared the burden of having to generate u by *assisting* it through the feedforward path b_4, whose gain is chosen to be unity. This way, x_3 is devoid of u. In a similar manner, choosing $b_3 = a_3$ and $b_2 = a_2$ ensures that the input components of the signals being injected by the feedback paths are supplied by the feedforward paths. Thus, adding feed-in paths into the unscaled modulator of Figure 4.21 not only dramatically reduces the swings at the outputs of the integrators but also ensures that they remain largely independent of the amplitude of u (as long as the modulator is stable). This means that x_1, x_2, and x_3 can be scaled by factors larger than unity by reducing the sizes of the feedback capacitors. This reduces capacitor area and noise contributed by the second and third integrators, as discussed earlier in this section.

Adding feedforward modifies the STF, which is given by

$$\frac{L_0(z)}{1 + L_1(z)} = \frac{b_4(1 - z^{-1})^3 + b_3 c_3 z^{-1}(1 - z^{-1})^2 + b_2 c_2 c_3 z^{-2}(1 - z^{-1}) + b_1 c_1 c_2 c_3 z^{-3}}{D(z)}.$$

(4.34)

As expected, the numerator polynomial is altered by the feed-in paths, and causes peaking in the STF. This can be problematic if the input consists of signals with large out-of-band amplitudes, in the sense that the gain imparted by the STF can destabilize the $\Delta\Sigma$ loop.
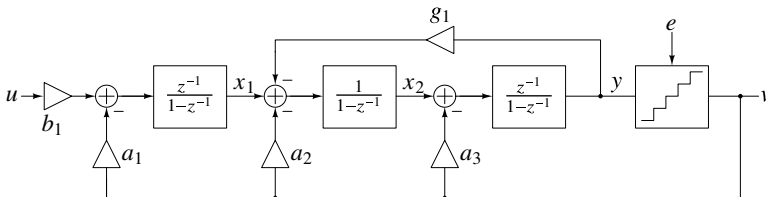
Figure 4.27   The CRFB $\Delta\Sigma$ modulator structure. Feed-in paths and scaling coefficients are not shown.

The CIFB structures that we have discussed so far can realize NTF zeros only at dc. We have already seen that a much higher SQNR can be achieved by placing these zeros at nonzero frequencies on the unit circle. This requires $L_1(z)$ to have complex poles, which is easily achieved by modifying the CIFB architecture, as indicated in Figure 4.27. This loop is capable of realizing three NTF zeros, one at dc and a complex conjugate pair on the unit circle. The first integrator contributes the dc pole to $L_1(z)$. The second and third integrators, together with the feedback path with gain $-g_1$, form a resonator with two complex poles which are the zeros of $z^2 - (2 - g_1)z + 1$. These poles will be on the unit circle at frequencies $\pm\omega_1$, where ω_1 satisfies $\cos(\omega_1) = 1 - (g_1/2)$. For the usual case when $\omega_1 \ll 1$, $\omega_1 \approx \sqrt{g_1}$. This modulator configuration is called a cascade of resonators with distributed feedback (CRFB) structure. As in the CIFB case, input feed-in paths can be added to the outputs of all integrators. While not shown in Figure 4.27, x_1, x_2, and x_3 will need to be scaled for dynamic range, through the coefficients c_i, as in Figure 4.24.

The astute reader would have noticed that the resonator loop in Figure 4.27 contains a *non-delaying* integrator. This is necessary to ensure that the resonator poles lie on the unit circle. For $\Delta\Sigma$ ADCs operating at high sampling rates (to achieve a large signal bandwidth), it is advantageous to have a delay associated with every integrator, since this reduces the speed requirements of the amplifiers used. In such situations, both integrator blocks in the resonator have transfer functions $z^{-1}/(1 - z^{-1})$. It is easy to see that the transfer function of the resonator then becomes

$$R(z) = \frac{z^{-2}}{1 - 2z^{-1} + (1 + g_1)z^{-2}}. \tag{4.35}$$

The poles are now outside the unit circle, at $z = 1 \pm j\sqrt{g_1}$. For $\omega_1 \ll 1$, $\omega_1 \approx \sqrt{g_1}$. The resonator by itself is unstable, as can be inferred from its pole locations. However, local oscillations are prevented since it is embedded in a strong negative-feedback system.

In designing the CRFB circuit, the value of g_1 can immediately be determined from ω_1, as shown above. The rest of the parameters (a_i and b_i) can readily be found by calculating $L_o(z)$ and $L_1(z)$, first from the specified STF and NTF, and then in terms of the a_i, b_i, and g_i from the circuit diagram, and matching the coefficients of like powers of z^{-1}. A less laborious way is to use the software tools described in Appendix B.

4.7.2 Loop Filters with Distributed Feedforward and Input Coupling: The CIFF and CRFF Structures
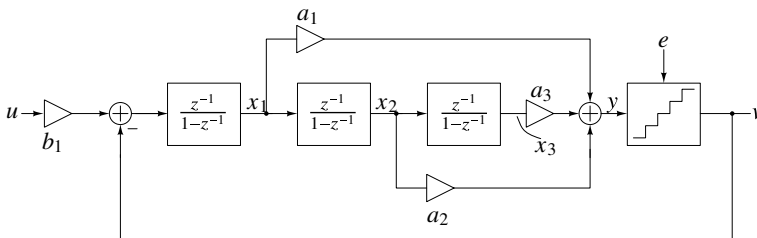
Figure 4.28   A third-order NTF realized using a cascade of integrators with feed forward (CIFF) structure.

The third-order NTF transfer function, realized using a CIFB structure in Figure 4.21 can also be realized by using feedforward in the loop filter, as shown in Figure 4.28. By inspection, $L_1(z)$ is given by

$$L_1(z) = a_1 \left(\frac{z^{-1}}{1 - z^{-1}} \right) + a_2 \left(\frac{z^{-1}}{1 - z^{-1}} \right)^2 + a_3 \left(\frac{z^{-1}}{1 - z^{-1}} \right)^3. \tag{4.36}$$

If the STF has to be one at dc, $b_1 = 1$ (so that the dc input into the first integrator is zero). The a_1, a_2, a_3 coefficients are determined from the desired $L_1(z)$.

As discussed in the feedforward implementation of MOD2 (Section 3.4.2), the dc components of x_1 and x_2 are zero because these states connect directly to the subsequent integrator. Since y consists of u and shaped noise, the input component of y must be the contribution of x_3. This in turn means an increased capacitor area. To avoid this, one can "assist" the loop filter by feeding u to its output, as shown in Figure 4.29. This way, all the integrators process only the shaped quantization noise, thereby resulting in reduced harmonic distortion. It is easy to see that $L_0(z) = 1 + L_1(z)$, which means that the STF is
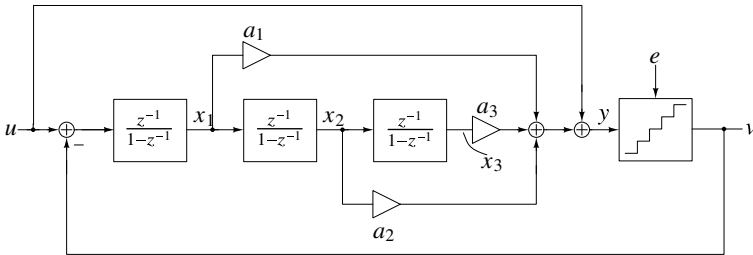
Figure 4.29   A low distortion CIFF structure, accomplished using input feedforward.

unity at all frequencies.

The "fast path" around the quantizer is that with the lowest order of integration. In the CIFF case, this corresponds to the first integrator. It turns out that after dynamic-range scaling, the first integrator's gain is large (since it only processes shaped quantization noise, and its output does not have a signal component). This is useful, since the noise and distortion added by the successive stages are rendered small when referred to the input. What is the other advantage of a CIFF implementation? Such a modulator would need only one feedback DAC, thereby simplifying design.

An aspect of a CIFF loop filter is that the feedback path of the modulator is part of the "fast" as well as the "precision" paths of the feedback loop. This can render the implementation of a high-speed design a challenge, especially when the loop filter is implemented in continuous time.

As (4.36) shows, all the three poles of $L_1(z)$ lie at dc, for the structures of Figures 4.28 and 4.29. Hence, so do all zeros of the NTF. To obtain optimized zeros, resonators must be created by internal feedback within the loop filter. The resulting modulator is illustrated in Figure 4.30. This is called the cascade of resonators with feed forward (CRFF) structure.
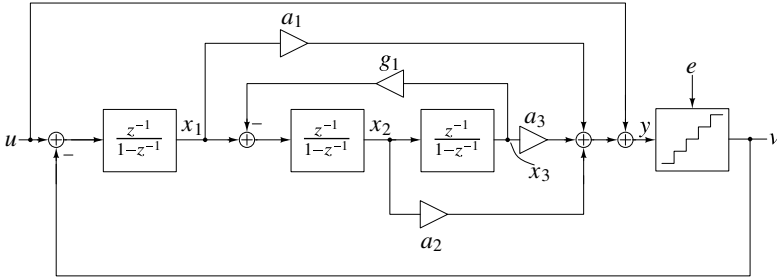
Figure 4.30   A low distortion CRFF structure. Internal feedback through g_1 realizes the complex zeros of the NTF.
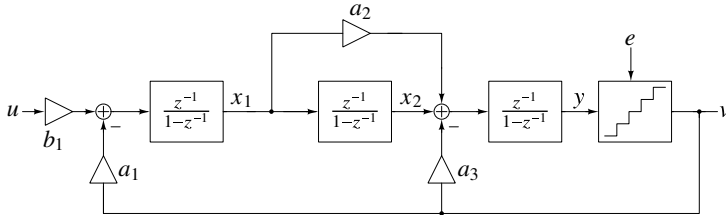
Figure 4.31   A $\Delta\Sigma$ modulator with feed-forward and multiple feedback paths.

4.7.3   Loop Filters with Feedforward and Multiple Feedback: The CIFF-B Structure

Having understood various trade-offs associated with CIFB and CIFF loop filters, we are now in a position to combine multiple feedback and feedforward. The aim of this exercise is to create a topology (which we call the cascade of integrators with feedforward and feedback, or CIFF-B) that inherits the benefits of its parents. Figure 4.31 shows a third-order $\Delta\Sigma$ modulator that uses a CIFF-B loop filter. It uses two DACs; as opposed to one and three for the CIFF and CIFB loops, respectively. It also has multiple feedback paths through coefficients a_1 and a_3. The fast feedback around the quantizer is through a_3, while a_1 controls the gains of the second- and third-order paths. The advantage of this is that the fast and precise paths of the feedback loop are decoupled, allowing them to be optimized for speed and precision, respectively, the same as in a CIFB loop. This is particularly important when the loop filter is implemented in continuous-time, and we will revisit this aspect in Chapter 8. The second-order path is realized using feedforward, through the first and third integrators. Since x_1 is the input to the second integrator, it must have a very small input component. This means that its gain will be large after scaling the modulator for dynamic range. This is a useful attribute, as we saw in connection with the CIFF loop filter. Not only does this lead to a smaller value for the feedback capacitor of the first integrator, but it also reduces the effect of noise and distortion added by subsequent stages of the loop filter. Though x_1 is largely independent of u, x_2 will have an input component, which is needed to ensure that the low-frequency input to the third integrator is small. It is easy to see that for a dc input u, x_2 consists of first-order shaped noise riding on a dc of value $a_3 u$. Complex zeros of the NTF can be realized by adding internal feedback across two integrators, as in the CIFF and CIFB cases.

From the discussion in this section, it is apparent that several methods of implementing the loop filter can be conceived. Every topology has its advantages and drawbacks, reinforcing the notion that engineering is all about trade-offs, and that there is no free lunch.

4.8   State-Space Description of Delta-Sigma Loops

Working with the loop filter description in transfer function form is convenient for analysis and building intuition. Computer simulation of the modulator's behavior, however, is best done by modeling the loop filter in state-space form. This approach is described extensively elsewhere [11], and the purpose of this section is to draw the reader's attention to a few aspects concerning the simulation of a $\Delta\Sigma$ modulator. The discussion that follows uses a second-order structure for illustration.

Figure 4.32 shows MOD2, where the block diagram of each delaying integrator is explicitly shown. The *output* of every delay element is a state. From the figure, it is seen
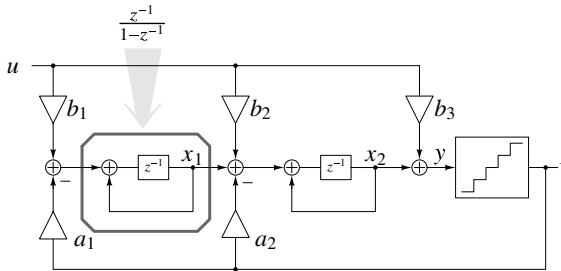
Figure 4.32   MOD2: the integrators are implemented with delays.

that the inputs to the delay elements can be related to the outputs by

$$
\begin{aligned}
x_1[n + 1] &= x_1[n] && + b_1 u[n] - a_1 v[n] \\
x_2[n + 1] &= x_1[n] + x_2[n] + b_2 u[n] - a_2 v[n] \\
y[n] &= x_2[n] + b_3 u[n].
\end{aligned}
$$

In matrix form,

$$
\underbrace{\begin{bmatrix} x_1[n+1] \\ x_2[n+1] \end{bmatrix}}_{\text{next state}} = \underbrace{\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}}_{A_d} \underbrace{\begin{bmatrix} x_1[n] \\ x_2[n] \end{bmatrix}}_{\text{present state}} + \underbrace{\begin{bmatrix} b_1 & -a_1 \\ b_2 & -a_2 \end{bmatrix}}_{B_d} \underbrace{\begin{bmatrix} u[n] \\ v[n] \end{bmatrix}}_{\text{inputs}} \tag{4.37}
$$

$$
y[n] = \underbrace{\begin{bmatrix} 0 & 1 \end{bmatrix}}_{C_d} \begin{bmatrix} x_1[n] \\ x_2[n] \end{bmatrix} + \underbrace{\begin{bmatrix} b_3 & 0 \end{bmatrix}}_{D_d} \begin{bmatrix} u[n] \\ v[n] \end{bmatrix}. \tag{4.38}
$$

A_d, B_d, C_d, D_d, are the discrete-time state-space matrices. In the general case of an nth order modulator, the dimensions of the A_d, B_d, C_d, D_d matrices are $n \times n, n \times 2, 1 \times n$ and 1×2, respectively. Observe that $D_d[1, 2] = 0$, indicating that the current loop filter output $y[n]$ does not depend on the current quantizer output $v[n]$. This is the state-space equivalent of the "no delay-free loop" criterion.

A shorthand notation of conveying this information, extensively used in the $\Delta\Sigma$ toolbox (see Appendix B), is to put the state matrices together into a single $(n+1)\times(n+2)$ matrix *ABCD* given by

$$ABCD = \left[\begin{array}{c|c} A_d & B_d \\ \hline C_d & D_d \end{array}\right]. \tag{4.39}$$

Once the state-space representation is known, simulation proceeds as follows. Knowing the states and u at instance n, (4.38) is used to obtain $y[n]$. The modulator output $v[n]$ is obtained by quantizing $y[n]$. The states at the next instant $(n+1)$ are determined using (4.37), and the process repeats.

4.9 Conclusions

In this chapter, we discussed high-order modulators. Special attention was devoted to the stability of high-order delta-sigma loops, both with multi-bit and with single-bit quantizers. For multi-bit loops, the quantizer's gain varies only slightly with its input signal, and hence tight theoretical bounds can be found for the signal range that ensure stable loop operation.

For single-bit loops, the equivalent gain of the quantizer varies strongly with the value of its input. For this reason, linearized stability analysis becomes a difficult task.

The optimization of the noise transfer function zeros and poles, already discussed for second-order loops in Chapter 3, was generalized here for higher order modulators. The most commonly used loop architectures were also described, analyzed, and compared.

References

[1] J. G. Kenney and L. R. Carley, "Design of multibit noise-shaping data converters," *Analog Integrated Circuits and Signal Processing*, vol. 3, no. 3, pp. 259–272, 1993.

[2] Y. Yang, R. Schreier, and G. Temes, "A tight sufficient condition for the stability of high order multibit delta-sigma modulators," *Oregon State University Research Report*, 1991.

[3] L. Risbo, *Sigma-Delta Modulators: Stability Analysis and Optimization*. Ph.D. dissertation, Technical University of Denmark, 1994.

[4] S. R. Norsworthy, R. Schreier, and G. Temes, *Delta-Sigma Data Converters: Theory, Design, and Simulation*. IEEE Press, New York, 1997.

[5] C. Mohtadi, "Bode's integral theorem for discrete-time systems," *IEE Proceedings on Control Theory and Applications*, vol. 137, no. 2, pp. 57–66, 1990.

[6] W. Lee, *A Novel Higher Order Interpolative Modulator Topology for High Resolution Oversampling A/D Converters*. Ph.D. dissertation, Massachusetts Institute of Technology, 1987.

[7] K. C. Chao, S. Nadeem, W. L. Lee, and C. G. Sodini, "A higher order topology for interpolative modulators for oversampling A/D converters," *IEEE Transactions on Circuits and Systems*, vol. 37, no. 3, pp. 309–318, 1990.

[8] R. Schreier, "An empirical study of high-order single-bit delta-sigma modulators," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 40, no. 8, pp. 461–466, 1993.

[9] J. Steensgaard-Madsen, *High-Performance Data Converters*. Ph.D. dissertation, The Technical University of Denmark, 1999.

[10] J. Silva, U. Moon, J. Steensgaard, and G. Temes, "Wideband low-distortion delta-sigma ADC topology," *Electronics Letters*, vol. 37, no. 12, pp. 737–738, 2001.

[11] B. C. Kuo, *Digital Control Systems*. Holt, Rinehart and Winston, 1980.

CHAPTER 5

MULTI-STAGE AND MULTI-QUANTIZER DELTA-SIGMA MODULATORS

$\Delta\Sigma$ converters rely on oversampling and noise-shaping to reduce the power of the quantization noise in the signal band. The SQNR (signal-to-quantization noise ratio) can be increased by increasing the OSR (oversampling ratio), and/or the quantizer resolution, and/or the order or aggressiveness of the loop filter. In this chapter, we discuss $\Delta\Sigma$ modulators that combine an alternative strategy – noise cancellation with noise-shaping.

5.1 Multi-Stage Modulators

As discussed in the previous chapter, the SQNR of a $\Delta\Sigma$ modulator can be increased by raising its OSR, and/or the order L of the loop, and/or the number of levels M in the quantizer. However, there are practical limits to all these parameters. Higher OSR requires more power, and this is limited by the speed allowed by the available IC technology. Raising the order of the loop filter is subject to stability considerations, which limit the maximum permissible input signal amplitude for higher order loops. This counteracts the expected improved noise suppression. Finally, the SQNR can also be increased by using more bits in the internal quantizer, but this requires a flash ADC and necessitates additional circuitry to ensure the in-band linearity of the internal DAC. (This subject will be discussed in Chapter 6.) As a result, the complexity of the quantizer grows exponentially with the number of bits used. Hence, the quantizer resolution is seldom higher than 4 or 5 bits.

A different strategy is to cancel the quantization noise in addition to noise filtering us-ing a multi-stage structure for the modulator. The quantized outputs can then be combined in a way that reduces the overall noise power. This chapter is discusses modulators that use such schemes.

5.1.1   The Leslie–Singh Structure [1]
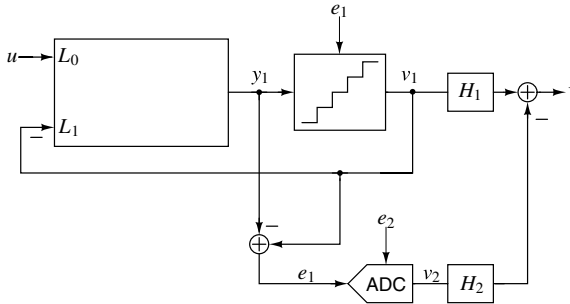
Figure 5.1   The $L - 0$ cascade (Leslie–Singh) structure.

A simple two-stage delta-sigma ADC is illustrated in Figure 5.1. It contains an Lth-order $\Delta\Sigma$ modulator as its first stage, and a static (i.e., zero-order) ADC as its second stage. The outputs of the two stages, v_1 and v_2, are digitally filtered and combined to obtain the overall output v.

As shown, the quantization error $e_1[n]$ of the input stage is extracted in analog form by subtracting the input signal y_1 of the internal quantizer from its output v_1. The error e_1 is then converted into digital form by a multi-bit (e.g., 10-bit) ADC that forms the second stage of the modulator. This introduces another quantization error $e_2[n]$, which however can be much smaller than $e_1[n]$, since the second-stage ADC (not being in a feedback loop) is allowed to have arbitrary latency, and hence it can be realized as a low-complexity multi-bit pipeline structure.

Next, the outputs v_1 and v_2 of the two stages are filtered by the digital stages H_1 and H_2, respectively, and added. Usually $H_1(z) = z^{-k}$. This simply implements a delay that equals the latency of the second-stage ADC. Also, H_2 can be chosen as the digital equivalent of the NTF of the first stage. Then, subtracting the output of H_2 from that of H_1 produces the output

$$\begin{aligned} V(z) &= H_1(z)V_1(z) - H_2(z)V_2(z) \\ &= z^{-k}\left[STF_1(z)U(z) + NTF_1(z)E_1(z)\right] - NTF_1(z)z^{-k}\left[E_1(z) + E_2(z)\right] \\ &= z^{-k}\left[STF_1(z)U(z) - NTF_1(z)E_2(z)\right]. \end{aligned} \tag{5.1}$$

In comparing the output $V(z)$ with the first-stage output $V_1(z)$, it is clear that (apart from the delay of k clock periods) the difference is that $E_1(z)$ is replaced by $E_2(z)$ in $V(z)$. As explained above, $E_2(z)$ may be much smaller than $E_1(z)$ since it is much cheaper to construct a multi-bit pipeline ADC than a multi-bit loop quantizer for the first stage. Hence, this technique can enhance the SQNR by as much as 25 to 30 dB.

To obtain $e_1[n]$ by simple subtraction, the operation of the quantizer must be delay free, which may not be practical. In this case, the signal y_1 must be delayed before the subtraction is carried out. To avoid the subtraction altogether, the input signal of the second stage can be chosen as $y_1[n]$, the input signal of the first-stage ADC, instead of $e_1[n]$. It is given by

$$Y_1(z) = V_1(z) - E_1(z) = STF_1(z)U(z) + [NTF_1(z) - 1]E_1(z). \tag{5.2}$$

We keep $H_1(z) = z^{-k}$, but choose the other filter function as $H_2(z) = NTF_1(z)/(NTF_1(z) - 1)^1$, so that the overall output now becomes

$$
\begin{aligned}
V(z) &= z^{-k}[STF_1(z)U(z) + NTF_1(z)E_1(z)] \\
&\quad - \frac{NTF_1(z)}{NTF_1(z) - 1}z^{-k}\{STF_1(z)U(z) + [NTF_1(z) - 1]E_1(z) + E_2(z)\}.
\end{aligned} \tag{5.3}
$$

Assuming perfect cancellation of like terms,

$$V(z) = \frac{z^{-k}STF_1(z)}{1 - NTF_1(z)}U(z) + \frac{z^{-k}NTF_1(z)}{1 - NTF_1(z)}E_2(z) \tag{5.4}$$

results. In the signal band, $|NTF \ll 1|$, and hence the SQNR obtained with the new $V(z)$ is very close to the one obtainable with the $V(z)$ given in (5.1). A disadvantage of using $y_1[n]$ as the input to the second stage is that it contains $u[n]$ as well $e_1[n]$, and hence the second stage must be able to handle a larger input signal. Also the second stage must have low distortion, to avoid generating harmonics of $u[n]$.

Consider next one of the low-distortion structures discussed in Section 4.7 being used as the first stage. Suppose that in the CIFB modulator of Figure 4.26 the conditions $b_i = a_i$ for all $i \leq N$ and $b_{N+1} = 1$ hold. Then $STF(z) = 1$, and the output signal of the last integrator is

$$
\begin{aligned}
X_N(z) = Y(z) - b_N U(z) &= STF(z)U(z) - [1 - NTF(z)]E(z) - b_n U(z) \\
&= [1 - NTF_1(z)]E(z). \tag{5.5}
\end{aligned}
$$

This signal can be used as the input signal of the second-stage ADC. It does not contain u, and hence the second stage has a smaller input signal, and need not be very linear. Note, however, that the time domain signal $x_N[n]$ now contains a linear combination of delayed versions of $e[n]$, which can be larger than $e[n]$, and hence it should be scaled appropriately.

It can easily be seen that similar conclusions apply to the other low-distortion structures: it is possible to extract $y_1[n] - u[n] \approx e[n]$, and use it as the input to the second stage. As an example, Figure 5.2 shows a second-order low-distortion CIFF modulator. Simple analysis shows that its noise transfer function is $(1 - z^{-1})^2$, its signal transfer function is 1, and the output signal of its second integrator is $X_2(z) = z^{-2}E_1(z)$. Hence, $X_2(z)$ can be used directly as the input to the second stage of the structure.

It should be noted that canceling, rather than filtering out, the noise $e_1[n]$ is an inherently ill-conditioned operation. Thus, a small error in the transfer functions involved may result in a large "leakage" of the $e_1[n]$ noise. Also, the scaling of the signal fed to the second stage needs to be carefully considered. It should not overload the second stage, but should secure a good dynamic range for it.

---

[1]$H_2(z)$, as written, is non-causal. Multiplying $H_2(z)$ and $H_1(z)$ by z^{-1} yields a realizable pair of filters and preserves the desired noise cancellation.
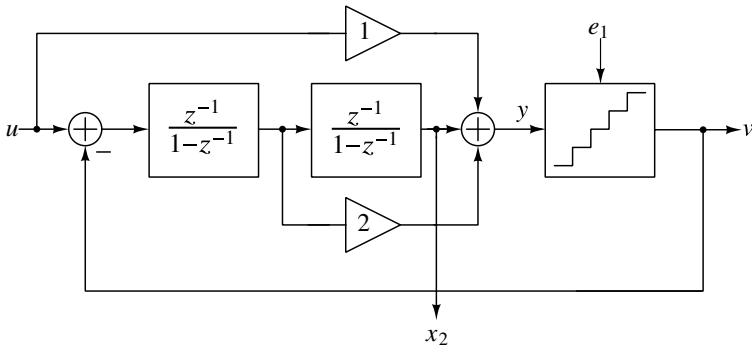
Figure 5.2   Low-distortion CIFF modulator used as first MASH stage.

5.2   Cascade (MASH) Modulators
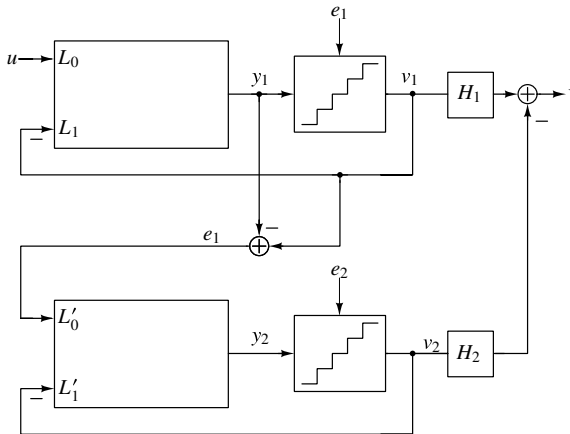
Figure 5.3   A two-stage MASH structure.

An obvious extension of the Leslie–Singh modulator, which historically preceded it, is the cascade modulator, also called multi-stage or MASH (for Multi-stAge noise-SHaping) modulator [2, 3, 4]. Here, the second stage is realized by another delta-sigma modulator. The basic concept is illustrated in Figure 5.3. The output signal of the first stage is given by

$$V_1(z) = STF_1(z)U(z) + NTF_1(z)E_1(z), \tag{5.6}$$

where STF_1 and NTF_1 are the signal and noise transfer functions, respectively, of the first stage.

As shown in Figure 5.3, the quantization error e_1 of the input stage is found in analog form by subtracting the input to its internal quantizer from its output. It is then fed to another $\Delta\Sigma$ loop forming the second stage of the modulator, and converted into digital form. Hence, the output signal of the second stage in the z-domain is given by

$$V_2(z) = STF_2(z)E_1(z) + NTF_2(z)E_2(z), \tag{5.7}$$

where STF_2 and NTF_2 are the signal and noise transfer functions, respectively, of the second stage. 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 condition $E_1(z)$ is canceled. By (5.6) and (5.7), this is achieved if

$$H_1 \cdot NTF_1 - H_2 \cdot STF_2 = 0 \tag{5.8}$$

holds. The simplest (and usually most practical) choice for H_1 and H_2 that satisfies (5.8) is $H_1 = STF_2$ and $H_2 = NTF_1$. Since STF_2 is often just a delay, H_1 is easily realized. The overall output is then ideally given by

$$V = H_1 V_1 - H_2 V_2 = STF_1 \cdot STF_2 \cdot U - NTF_1 \cdot NTF_2 \cdot E_2. \tag{5.9}$$

In a typical case, both stages of the MASH may contain a second-order loop, and their transfer functions may be given by

$$STF_1(z) = STF_2(z) = z^{-2} \tag{5.10}$$

and

$$NTF_1(z) = NTF_2(z) = (1 - z^{-1})^{-2}. \tag{5.11}$$

Then, the overall output will be

$$V(z) = z^{-4} \cdot U(z) - (1 - z^{-1})^4 E_2(z). \tag{5.12}$$

Thus, the noise-shaping performance is that of a fourth-order single-loop converter, while the stability is that of a second-order one, since both internal feedback loops are of order two.[2]

If the condition (5.8) is not exactly satisfied due to imperfections in the implementation of the analog transfer functions, then E_1 will appear at the output, multiplied by $STF_2 NTF_{1a} - NTF_1 STF_{2a}$, where the subscript a denotes the actual value of the analog transfer function. As will be shown in Section 5.3, this may result in a serious deterioration of the noise performance of the converter.

As discussed in the preceding section, it is advantageous for MASH systems as well to use a low-distortion loop filter structure in all stages. This makes it possible to obtain the first-stage error $e_1[n]$ without any subtraction, for entering it into the second stage. In addition, of course, the low-distortion property improves the performance of both stages.

An advantage of the MASH configuration 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. Hence, the second-stage quantization error $e_2[n]$ is very similar to a true white noise. This remains valid even if the first-stage noise contains tones. Figure 5.4 shows the simulated output spectra of the input stage (V_1) and the overall modulator (V) for a 2-2 MASH with single-bit quantization. V_1 contains the third harmonic near $f = 0.01$, which is greatly reduced in V. Thus, a MASH modulator is less likely to need dithering than a single-stage one.

---

[2]In practice, the e_1 input to the second modulator stage needs to be scaled to fit within the stable input range. For a second-order, single-bit first stage, the usual scaling factor is $k = 1/4$. If multi-bit quantization is used in the first stage, the scaling factor can be greater than 1. The inverse of this scaling factor $1/k$ needs to be included in H_2 in order to cancel e_1. For $k > 1$, this reduces e_2 in the output, and thus enhances the SQNR.

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 dynamic or other correction of the DAC nonlinearity [5]. This is because 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 above, $H_2(z)$ contains the NTF of the first stage. Since this $NTF_1(z)$ is a highpass filter function, the nonlinearity error of the second-stage DAC is suppressed in the baseband.

Also, since the input to the second stage contains the quantization error $e_1[n]$ 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 error) the small added noise due to the second-stage DAC nonlinearity is usually tolerable.
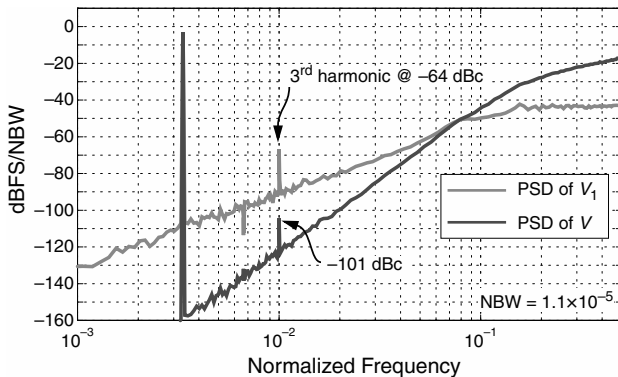
Figure 5.4   Output spectra for a 2-2 MASH modulator.

The principle of quantization error cancellation implemented by the two-stage MASH structure of Figure 5.3 can be extended. Just as the second stage of the MASH is used to cancel the quantization error $e_1[n]$ of the first stage, a third stage can be added to cancel $e_2[n]$, the quantization error of the second stage (Figure 5.5). The cancellation conditions can be found exactly as for the two-stage MASH. They are

$$
\begin{aligned}
H_1 \cdot NTF_1 - H_2 \cdot STF_2 &= 0, \\
H_2 \cdot NTF_2 - H_3 \cdot STF_3 &= 0.
\end{aligned}
\tag{5.13}
$$

Under these conditions, e_1 and e_2 are canceled in the overall output signal:

$$
\begin{aligned}
V &= [STF_1 \cdot U + NTF_1 \cdot E_1] \cdot H_1 - [STF_2 \cdot E_1 + NTF_2 \cdot E_2] \cdot H_2 \\
&\quad + [STF_3 \cdot E_2 + NTF_3 \cdot E_3] \cdot H_3 \\
&= STF_1 \cdot H_1 \cdot U + NTF_3 \cdot H_3 \cdot E_3.
\end{aligned}
\tag{5.14}
$$

Using (5.13) to express H_3, we can rewrite V as

$$
V = STF_1 \cdot H_1 \cdot U + \frac{H_1 \cdot NTF_1 \cdot NTF_2 \cdot NTF_3}{STF_2 \cdot STF_3} \cdot E_3.
\tag{5.15}
$$

As discussed earlier, H_1 and the signal transfer functions usually contain only simple delays, or have a flat gain in the signal band, and hence they will not shape either the signal or the noise significantly. However, the NTFs provide suppression over the baseband. Hence,

under ideal conditions, the quantization errors of the first two stages are canceled, while that of the third stage is filtered by the product of the NTFs of all three stages. Thus, if all three stages contain second-order loop filters, the overall NTF of the structure will be equivalent to that of a sixth-order modulator, but without the troublesome stability problems inherent in such a high-order loop.
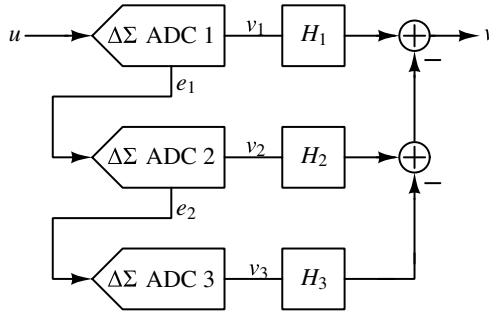
Figure 5.5   A 3-stage MASH ADC.

Since the three-stage MASH is normally used only when it is necessary to provide very high SQNR performance, the leakage of the poorly filtered quantization noise of the first stage due to imperfect matching between the analog transfer functions $(NTF_{1,2,3}), STF_{1,2,3}$ and the digital ones $(H_{1,2,3})$ is a very critical issue here, and such leakage limits the practically achievable resolution. The topic of noise leakage will be discussed in the next section.

5.3   Noise Leakage in Cascade Modulators

In high-order single-stage modulators, the imperfect matching of the passive loop filter elements (usually capacitors) and the finite gain of the active ones (usually opamps) will change the coefficients of the NTF and STF, but will usually not affect the SQNR performance significantly. This is because the quantization error is suppressed by filtering, and as long as the gain L_1 of the loop filter remains sufficiently large in the signal band, $|NTF| \approx |1/L_1| \ll 1$ will continue to hold there. It can be easily shown,for example, that for opamp gains as low as OSR/π, the SQNR decreases by only a few decibels from its ideal value for high-order single-stage ADCs.

In a two-stage MASH structure, by contrast, a large SQNR is achieved by accurate cancellation of the first-stage quantization error e_1, which is shaped only by a low-order NTF_1. As (5.13) shows, this requires accurate matching between the nominally identical mixed-signal (analog and digital) transfer functions $H_1 \cdot NTF_1$ and $H_2 \cdot STF_2$. For the designer, it is important to know how precise the analog circuit needs to be to obtain good performance for the cascade, namely how accurately the components need to be matched, and what is the minimum acceptable gain for the opamps, etc., in order to keep the leakage of e_1 acceptably low. For a three-stage MASH, the leakage of e_2 also needs to be analyzed. Even for relatively simple structures, the equations describing the leakage can become very complex.

As is usual for delta-sigma modulators, accurate behavioral simulation is the most reliable technique for predicting the effects of all nonidealities on the SQNR of a MASH modulator. However, under some (usually valid) conditions, useful and simple results can be obtained using linear analysis and approximations, as will be shown next.

Referring back to (5.14), the transfer functions from e_1 and e_2 to the overall output v are

$$
\begin{aligned}
H_{l1} &= H_1 \cdot NTF_1 - H_2 \cdot STF_2, \\
H_{l2} &= H_2 \cdot NTF_2 - H_3 \cdot STF_3,
\end{aligned}
\tag{5.16}
$$

respectively. Ideally, both of these leakage transfer functions are identically zero, but since the NTF and STF functions are realized using imperfect analog components, they will be inaccurate. Hence, H_{l1} and H_{l2} will be nonzero, allowing e_1 and e_2 to leak into v. We can usually justify the following simplifying assumptions:

1. The leakage of e_2 is less important than that of e_1 . This is because the terms in H_{l2} represent higher-order noise-shaping than those in H_{l1} . As an example, in a 2-2-1 MASH, the noise-shaping due to H_{l1} is at most of order 2, while that of H_{l2} is of order 4. Also, often e_2 is smaller than e_1 if a multi-bit second-stage quantizer is used.

2. In H_{l1}, the effect of an imperfect NTF_1 dominates that of the imperfect STF_2, even though the gain error is the same for both. This is because the second stage is followed by the noise-shaping block $H_2 = NTF_1$. Hence, error signals due to imperfect STF_2 are inherently noise shaped. The same is not true for errors due to an imperfect NTF_1, since the H_1 block has unity gain in the signal band.

3. In view of assumption 2, we can set $STF_2 = H_1 = 1$ in (5.16), as a first approximation. Then
$$
|H_{l1}| \approx |NTF_1 - H_2| = |NTF_{1a} - NTF_{1i}|
\tag{5.17}
$$
results, where the subscripts a and i denote actual and ideal functions, respectively.

4. From (5.3), $NTF_1 = 1/(1 + L_1)$, where L_1 is the gain of the loop filter from the quantizer output to its input. Assuming only small errors, $|L_1| \gg 1$ holds for both the ideal and actual functions. Then, (5.17) can be further approximated by
$$
|H_{l1}| \approx \left| \frac{1}{L_{1i}} - \frac{1}{L_{1a}} \right|.
\tag{5.18}
$$

Equation (5.18) is much simpler to evaluate than the complete original relations (5.14) or (5.16). It can be used for both two- and three-stage MASH modulators.

As an illustration, consider the simple case of a 1-1 or 1-1-1 MASH modulator. The loop filter for the first stage is just a delaying integrator, with an ideal transfer function

$$
I_1(z) = \frac{az^{-1}}{1 - z^{-1}}.
\tag{5.19}
$$

If the integrator is realized using switched-capacitor (SC) circuitry such as shown in Figure 5.6, then a relative error D in the nominal capacitance ratio C_1/C_2 will change the
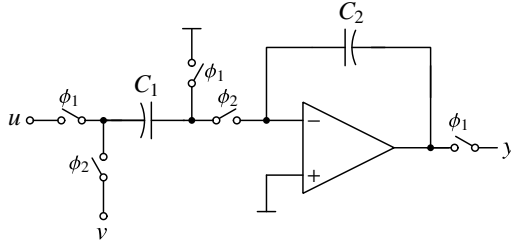
Figure 5.6 A delaying switched-capacitor integrator used in a first-order $\Delta\Sigma$ modulator.

factor a, and the finite dc gain A of the op amp will change both a and the value of the pole (ideally at $p = 1$). The resulting actual transfer function is

$$I_a(z) = \frac{a'z^{-1}}{1 - p'z^{-1}}, \tag{5.20}$$

where, as a simple analysis shows, for $D \ll 1$ and $(a/A) \ll 1$,

$$a' \approx a \left[1 - D - \frac{(1 + a)}{A} \right] \tag{5.21}$$

and

$$p' \approx 1 - \frac{a}{A}. \tag{5.22}$$

Since here $L_1(z) = -I(z)$, by (5.18), the leakage transfer function is

$$
\begin{aligned}
|H_{l1}| &= \left| \frac{z - 1}{a} - \frac{z - p'}{a'} \right| \\
&\approx \left| \frac{1}{a'} \right| \cdot \left| \frac{a}{A} + (z - 1) \cdot \left[D + \frac{1 + a}{A} \right] \right| \\
&\approx \left| \frac{1}{A} + (z - 1) \cdot \left[\frac{D}{a} + \frac{1 + (1/a)}{A} \right] \right|.
\end{aligned} \tag{5.23}
$$

As (5.23) shows, there is an unfiltered leakage component approximately equal to E_1/A, and a first-order-filtered component given approximately by $(z-1) \cdot \left[\frac{D}{a} + (1 + 1/a)/A \right] E_1$. For a high specified SQNR, a very-high-gain opamp with fast settling is required to reduce the unfiltered leakage to a sufficiently low level. If the OSR is low, then the second component will also be significant, requiring $D \ll 1$. Hence, the matching accuracy of the capacitors must also be very high.

Errors in the path coupling the first and second stages will also add to $H_{l1} \cdot E_1$. However, their effects on the output V will be at least first-order filtered, since the error signal will pass through the H_2 filter.

For a second-order first stage, the leakage of e_1 can be reduced. The calculations, however, become much more complicated. Consider, e.g., the 2-0 MASH (Leslie–Singh) modulator (Figure 5.1), with the ideal output given in (5.1). Assume that the first stage is realized by the low-distortion modulator structure shown in Figure 5.2, built from two cascaded integrators. The ideal transfer functions of the integrators are given by (5.19) and their actual transfer functions by (5.20) through (5.22).

As a result of these changes, and the inaccuracies in the other coefficients b_i in the loop, there will be leakage of the first-stage quantization error E_1 to the output V of the overall ADC system. It is useful to represent the parasitic leakage transfer function by its Taylor-series expansion around $z = 1$ [6]:

$$H_{l1}(z) = A_0 + A_1(1 - z^{-1}) + A_2(1 - z^{-1})^2 + \cdots, \tag{5.24}$$

where, assuming $A \gg 1$ and $D \ll 1$, the coefficient values are

$$
\begin{aligned}
A_0 &= \frac{1}{A^2}, \\
A_1 &= \left(\frac{1}{a_1} + \frac{1}{a_2}\right)\frac{1}{A}, \\
A_2 &= \frac{1}{a_1 a_2} - 1 + 2\left(1 - \frac{1}{a_1 a_2} - \frac{1}{a_2^2}\right)A + \frac{2D}{a_1 a_2}.
\end{aligned}
\tag{5.25}
$$

The first term in the series expansion of H_{l1} represents the unfiltered leakage. Since it is inversely proportional to A^2, it is usually very small. The second term gives the linearly filtered error leakage, the third the quadratically filtered leakage and so on. For $OSR \gg 1$ and typical opamp gains and matching errors, the linear and quadratic terms containing A_1 and A_2 tend to dominate H_{l1}, since A_0 is normally very small, and since high-order filtering suppresses the terms beyond the quadratic one.

The derivation given above ignored leakage due to the errors in the coupling branch and in the second stage. In this case, these errors contribute only to the quadratic and high-order terms (A_2, A_3, etc.), since H_2 is here a second-order highpass filter.

As an illustration, for $A = 1000$ and $D = 0.5\%$, we find that $A_0 = 10^{-6}$ and the values of coefficients $A_1 - A_4$ are between 0.001 and 0.2. The multipliers $(1 - z^{-1})^L$ introduce highpass filtering into the terms in H_{l1}, which reduces their effects on the inband noise. The reduction increases rapidly with increasing L and OSR. For example, with $OSR = 64$, the linear term ($L = 1$) is reduced by a factor around 1/30, the quadratic term by about 1/1000, and the cubic one by about 1/30,000. Hence only the first three terms are significant.

5.4   The Sturdy-MASH Architecture

The high sensitivity of the cascade $\Delta\Sigma$ ADC to analog circuit imperfections can be reduced, and the noise cancellation logic eliminated, by a modification of the MASH architecture. The block diagram of the modified modulator is shown in Figure 5.7 [7, 8]. As the figure reveals, there are two differences between the MASH and sturdy-MASH structures: (1) in the modified structure, the output of the second stage is coupled back into the first loop; (2) the noise cancellation logic (H_1 and H_2) is absent in the unmodified structure.

As a result of these changes, the output of the modulator is now given by the relation

$$V = STF_1 \cdot U - NTF_1 \cdot NTF_2 \cdot E_2 + NTF_1 \cdot (1 - STF_2) \cdot E_1. \tag{5.26}$$

Comparing this equation with equations (5.8) and (5.9) reveals that the condition (5.8) for canceling e_1 in the output is now replaced by $(1 - STF_2) = 0$. On the one hand, this is
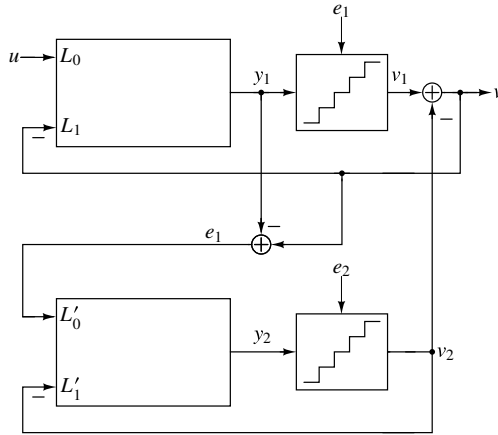
Figure 5.7   Block diagram of a sturdy-MASH modulator.

a bad deal, since STF_2 cannot be a delay-free function, so this condition is impossible to satisfy even with ideal circuitry. On the other hand, in the signal band, the magnitude of the error $|1 - STF_2|$ can be reduced by choosing it with properties similar to those of the noise transfer functions. Thus, selecting $STF_2 = 1 - NTF_2$ yields

$$V = STF_1 \cdot U - NTF_1 \cdot NTF_2 \cdot (E_1 + E_2). \tag{5.27}$$

As a result, the high-sensitivity noise cancellation is replaced by low-sensitivity noise shaping, while still retaining the improved stability properties of the MASH scheme. For its robust performance, the modified scheme was named Sturdy-MASH or SMASH.
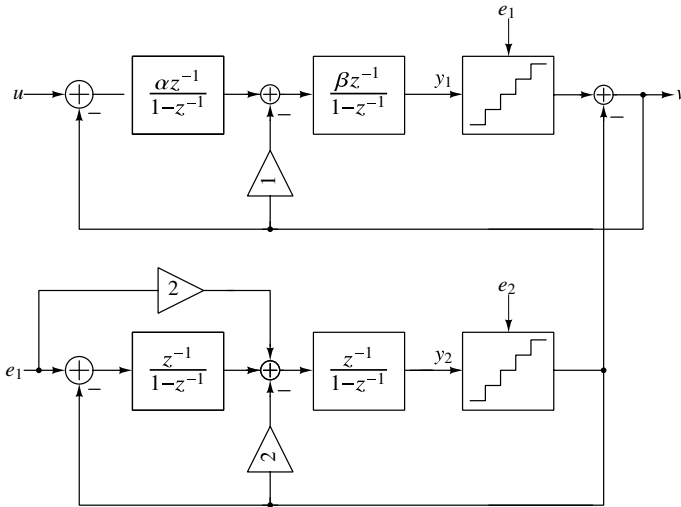
Figure 5.8   A 2 + 2 SMASH modulator.

Figure 5.8 illustrates a 2+2 SMASH modulator with $NTF_1 = NTF_2 = (1 - z^{-1})^2$.

As a comparison of (5.9) and (5.27) shows, E_2 is replaced by $E_1 + E_2$ in a SMASH ADC. Since e_1 and e_2 are uncorrelated noises, their powers add, and thus for $e_1 \approx e_2$, the noise increment is around 3 dB. Notice also that the added noise e_2 is inserted at the output of the first quantizer Q_1, and hence it is suppressed by the loop filter before reaching the input of Q_1. Thus, it does not tend to overload Q_1. See [8] for a description of a SMASH modulator that uses 35-dB opamps to achieve SNDR = 74 dB with an OSR =16; [9] and [10] describe additional modifications that can improve the performance of SMASH even further.

5.5 Noise-Coupled Architectures
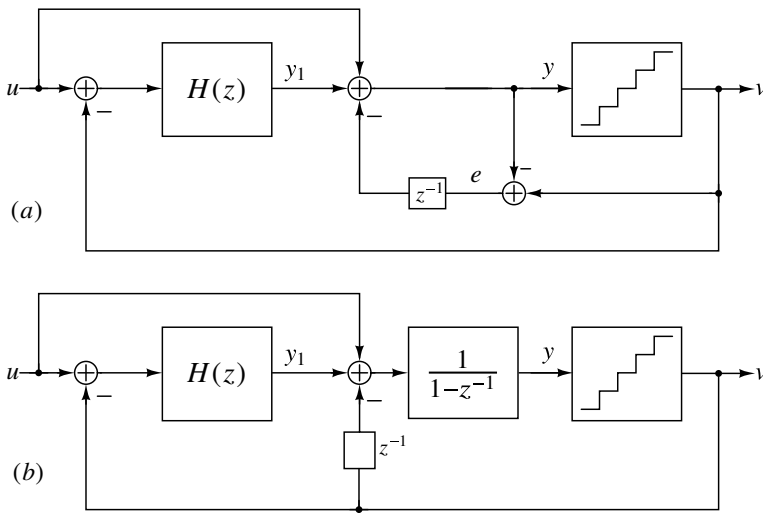
Figure 5.9   Noise-coupling in a $\Delta\Sigma$ ADC: (a) Actual circuit; (b) an alternative implementation.

Another strategy of enhancing the noise-shaping performance of the $\Delta\Sigma$ ADC is *noise-coupling* [11, 12], illustrated in Figure 5.9. Part (a) shows the actual circuit, and part (b) an alternative implementation. The quantization error e is acquired by subtracting the input of the quantizer from its D/A converted output. The difference is delayed, and fed back to the input of the quantizer. The effect is the replacement of $E(z)$ by $(1 - z^{-1})E(z)$. Figure 5.9(b) shows that this is equivalent to inserting an additional integrator into the loop filter. There is now no need for a fast multi-input adder at the input of the quantizer. This feature is particularly useful for low-distortion feed-forward modulators, which normally require an extra opamp with low feedback factor to carry out the summation of several signals.

The input signal of the quantizer may be somewhat increased by the added branch, since it now contains the first difference $e[n] - e[n-1]$ of the quantization error $e[n]$. However, $e[n]$ and $e[n-1]$ are only weakly correlated. Hence, the linear range of the quantizer is reduced at most by about 3 dB. On the positive side, the filtered error acts as a dither signal at the input of the quantizer. This dither converts tones and harmonics

into random noise, and improves the SFDR and THD parameters significantly. Hence, noise-coupled converters often achieve SFDR values over 100 dB, and are well-suited for applications where linearity is a key requirement.
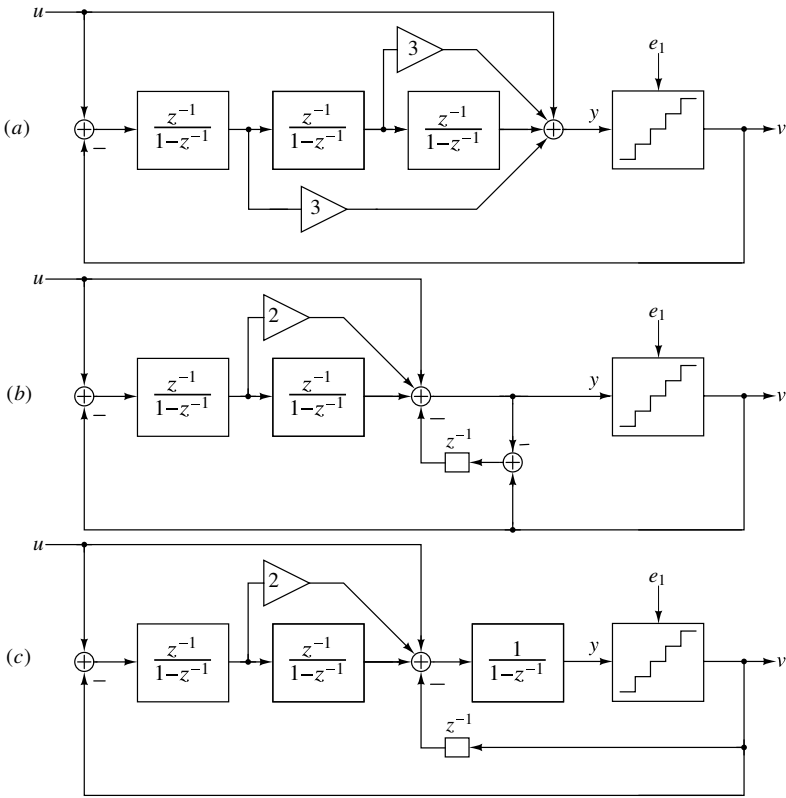
Figure 5.10   A third-order $\Delta\Sigma$ ADC: (a) Without noise-coupling; (b) and (c) with first-order noise-coupling.

It is possible to enhance the effect of noise-coupling by using more complex circuitry in the coupling path. As an illustration, Figure 5.10(a) shows a third-order feed-forward modulator without noise-coupling, and Figures 5.10(b) and (c) show modulators with first-order noise coupling.

Figure 5.11 shows modulators with second-order noise-coupling. Noise-coupling allows a reduction of the number of opamps, and hence lowers the power dissipation. First-order noise-coupling can reduce the number of active stages by 1; second-order coupling by 2.

Since the noise-coupling network follows the input stages of the loop filter, it does not affect the sensitivity to element value variations and offset errors. Simulations indicate that the performance of the modulator remains minimally affected when the opamp dc gains are as low as 30 dB and the element errors are as large as 5%.
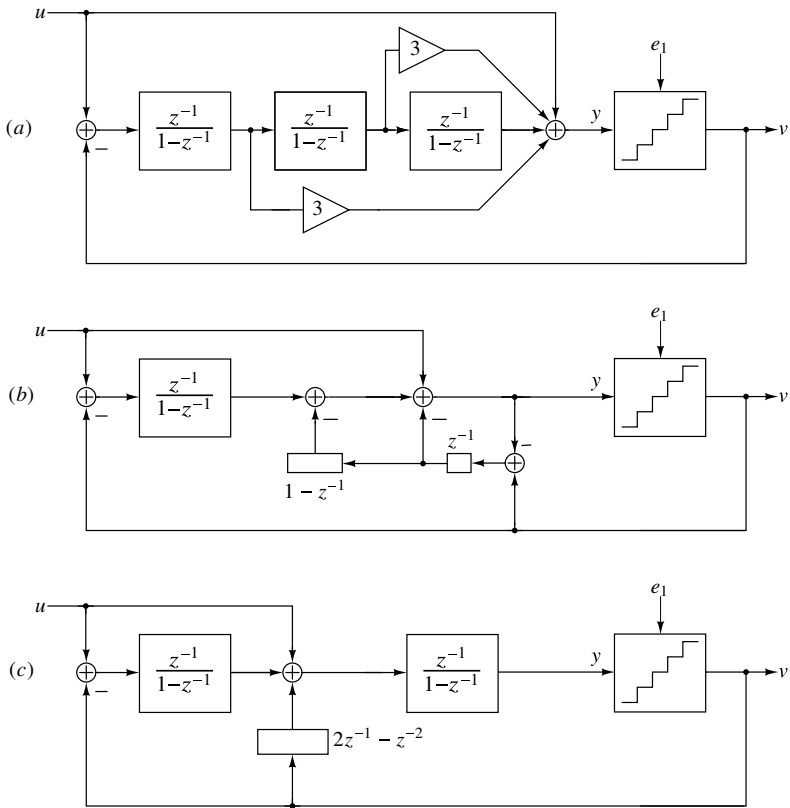
Figure 5.11   A third-order $\Delta\Sigma$ ADC: (a) Without noise-coupling; (b) and (c) with second-order noise-coupling.

5.6 Cross-Coupled Architectures

Noise-coupling is an effective way of improving the performance of split and time-interleaved $\Delta\Sigma$ ADCs. Figure 5.12(a) shows a split modulator. This structure was used to enable digital calibration of the ADC [13]. The two halves receive the same input signal u, and their outputs are added. Assuming $STF_1 = STF_2 = 1$ and $NTF_1 = NTF_2 = NTF$, the overall output is given by

$$V = U + NTF \cdot \frac{E_1 + E_2}{2}. \tag{5.28}$$

Since e_1 and e_2 are uncorrelated due to the mismatches between the two half circuits and also to noise, the SQNR of the full circuit is improved by 3 dB compared to that of each half. For a prescribed SQNR, this allows cutting the capacitance and transconductance values in the split circuit to half. This in turn results in the split circuit needing about the same amount of power as a single-path $\Delta\Sigma$ ADC.

Cross-coupling the quantization errors e_1 and e_2 results in the structure of Figure 5.12(b). The output is now

$$V = U + NTF \cdot (1 - z^{-1})\frac{E_1 + E_2}{2}. \tag{5.29}$$

Thus, an additional first-order noise-shaping results.

Further improvement can be achieved by time interleaving the two half circuits, with a half clock period shift (Figure 5.12(c)). The output signal of the resulting modulator is now given by

$$V = U + NTF \cdot (1 - z^{-1/2})\frac{E_1 + E_2}{2}. \tag{5.30}$$

Since at frequencies much lower than the clock rate $|1 - z^{-1/2}| \approx |1 - z^{-1}|/2$ holds, the SQNR of the time interleaved ADC will be about 6 dB higher than that of the cross-coupled modulator of Figure 5.12(b). The noise transfer functions of the three modulators of Figure 5.12 are compared for a second-order loop filter in Figure 5.13.

References [15] and [14] describe experimental results for single-path and time-interleaved noise-coupled modulators.

5.7 Conclusions

In this chapter, multi-stage and multi-quantizer modulators were discussed, and their advantages and drawbacks relative to the single-stage ones analyzed. The inherent ill conditioning of some of these modulators was pointed out, and techniques were given for estimating the noise leakage due to the imperfections of the analog components. Recent research in this area also explores other configurations, such as the 0-L MASH, where the first stage is a memoryless converter, and the second stage (a $\Delta\Sigma$ modulator) converts the error of the first one [16, 17, 18].
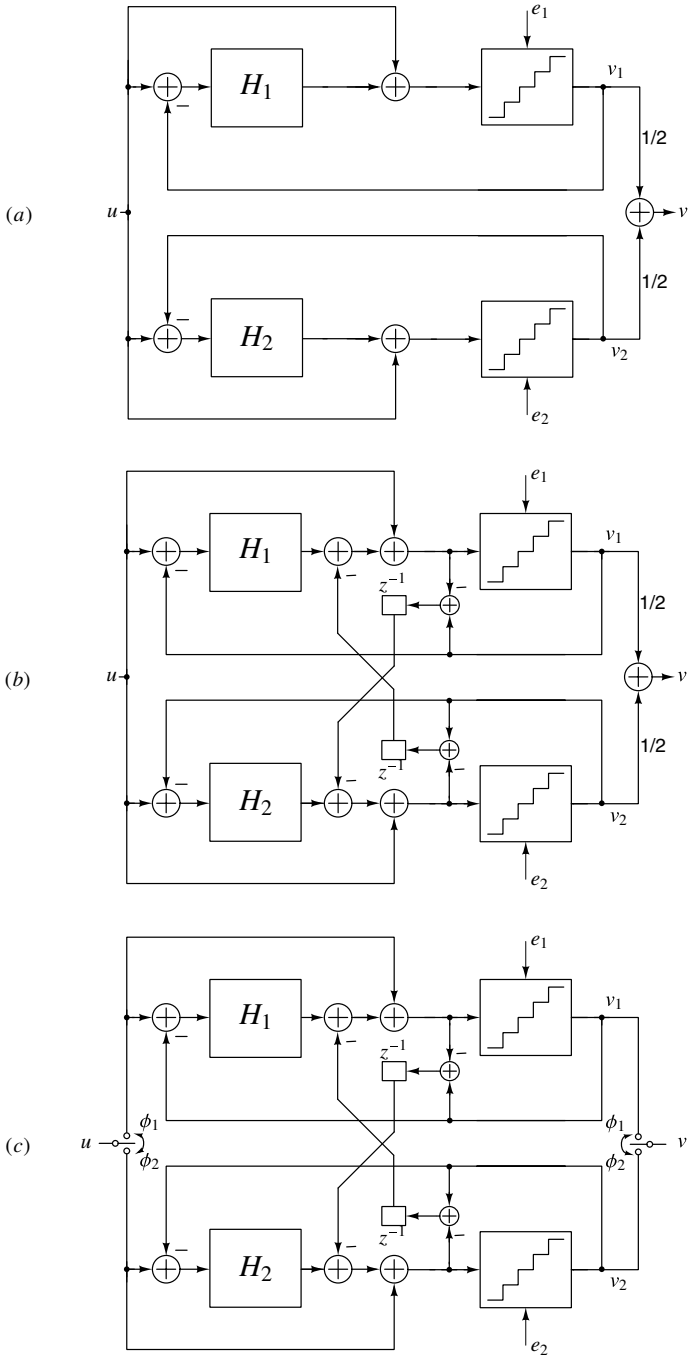
Figure 5.12 (a) Split modulator; (b) noise-coupled split (NCS) circuit; (c) noise-coupled time interleaved (NCTI) modulator.
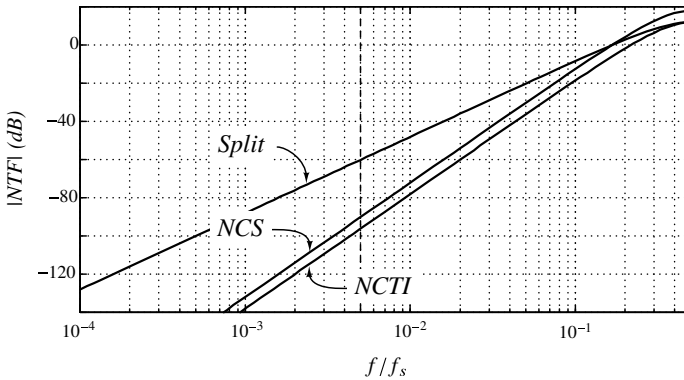
Figure 5.13 Noise transfer functions of split modulators.

References

[1] T. Leslie and B. Singh, "An improved sigma-delta modulator architecture," in *IEEE International Symposium on Circuits and Systems*, pp. 372–375, IEEE, 1990.

[2] T. Hayashi, Y. Inabe, K. Uchimura, and T. Kimura, "A multistage delta-sigma modulator without double integration loop," in *Digest of Technical Papers, IEEE International Solid-State Circuits Conference*, vol. 29, pp. 182–183, IEEE, 1986.

[3] Y. Matsuya, K. Uchimura, A. Iwata, T. Kobayashi, M. Ishikawa, and T. Yoshitome, "A 16-bit oversampling A-to-D conversion technology using triple-integration noise shaping," *IEEE Journal of Solid-State Circuits*, vol. 22, no. 6, pp. 921–929, 1987.

[4] J. C. Candy and A.-N. Huynh, "Double interpolation for digital-to-analog conversion," *IEEE Transactions on Communications*, vol. 34, no. 1, pp. 77–81, 1986.

[5] B. P. Brandt and B. Wooley, "A 50-MHz multibit sigma-delta modulator for 12-b 2-MHz A/D conversion," *IEEE Journal of Solid-State Circuits*, vol. 26, no. 12, pp. 1746–1756, 1991.

[6] P. Kiss, J. Silva, A. Wiesbauer, T. Sun, U.-K. Moon, J. T. Stonick, and G. C. Temes, "Adaptive digital correction of analog errors in MASH ADCs. II. Correction using test-signal injection," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 47, no. 7, pp. 629–638, 2000.

[7] N. Maghari, S. Kwon, G. Temes, and U. Moon, "Sturdy mash Δ-Σ modulator," *Electronics Letters*, vol. 42, no. 22, pp. 1269–1270, 2006.

[8] N. Maghari, S. Kwon, and U.-K. Moon, "74 dB SNDR multi-loop sturdy-MASH delta-sigma modulator using 35 dB open-loop opamp gain," *IEEE Journal of Solid-State Circuits*, vol. 44, no. 8, pp. 2212–2221, 2009.

[9] N. Maghari and U.-K. Moon, "Multi-loop efficient sturdy MASH delta-sigma modulators," in *IEEE International Symposium on Circuits and Systems*, pp. 1216–1219, IEEE, 2008.

[10] N. Maghari, S. Kwon, G. C. Temes, and U. Moon, "Mixed-order sturdy MASH Δ-Σ modulator," in *IEEE International Symposium on Circuits and Systems*, pp. 257–260, IEEE, 2007.

[11] K. Lee, M. Bonu, and G. Temes, "Noise-coupled ΔΣ ADCs," *Electronics Letters*, vol. 42, no. 24, pp. 1381–1382, 2006.

[12] K. Lee and G. C. Temes, "Enhanced split-architecture Δ-Σ ADC," *Electronics Letters*, vol. 42, no. 13, pp. 737–739, 2006.

[13] J. McNeill, M. C. Coln, and B. J. Larivee, "Split ADC architecture for deterministic digital background calibration of a 16-bit 1-MS/s ADC," *IEEE Journal of Solid-State Circuits*, vol. 40, no. 12, pp. 2437–2445, 2005.

[14] K. Lee, J. Chae, M. Aniya, K. Hamashita, K. Takasuka, S. Takeuchi, and G. C. Temes, "A noise-coupled time-interleaved delta-sigma ADC with 4.2 MHz bandwidth, 98 dB THD, and 79 dB SNDR," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 12, pp. 2601–2612, 2008.

[15] K. Lee, M. R. Miller, and G. C. Temes, "An 8.1 mW, 82 dB delta-sigma ADC with 1.9 MHz BW and 98 dB THD," *IEEE Journal of Solid-State Circuits*, vol. 44, no. 8, pp. 2202–2211, 2009.

[16] A. Gharbiya and D. Johns, "A 12-bit 3.125 MHz bandwidth 0–3 MASH $\Delta\Sigma$ modulator," *IEEE Journal of Solid-State Circuits*, vol. 44, no. 7, pp. 2010–2018, 2009.

[17] Y. Chae, K. Souri, and K. Makinwa, "A 6.3 μW 20 bit incremental zoom-ADC with 6 ppm INL and 1 μV offset," *IEEE Journal of Solid-State Circuits*, vol. 48, no. 12, pp. 3019–3027, 2013.

[18] Y. Dong, R. Schreier, W. Yang, S. Korrapati, and A. Sheikholeslami, "A 235 mW CT 0-3 MASH ADC achieving -167 dBFS/Hz NSD with 53 MHz BW," in *Digest of Technical Papers, IEEE International Solid-State Circuits Conference (ISSCC)*, pp. 480–481, 2014.

CHAPTER 6

MISMATCH-SHAPING

6.1 The Mismatch Problem

The inherent linearity of a one-bit DAC allows highly linear $\Delta\Sigma$ ADCs and DACs to be constructed without using highly accurate components. Unfortunately, one-bit quantization puts severe constraints on the achievable SQNR at moderate OSR, and also makes continuous-time modulators unduly sensitive to jitter. Multi-bit quantization can solve both problems but renders the associated multi-bit DAC sensitive to element mismatch. For example, simulations show that the two elements used to make a 3-level DAC must match to within 0.01% in order to achieve distortion lower than $-90\,\text{dBc}$. Since obtaining this degree of matching is difficult, efficient techniques for making multi-bit DACs linear despite element mismatch have great practical value.

One solution to the mismatch problem is calibration. Factory calibration, such as laser-trimming of thin-film resistors at the time of manufacture, is able to achieve the requisite matching but is vulnerable to aging and packaging shifts. Background or foreground (on-demand) calibration of current sources circumvents packaging stress but requires analog calibration circuitry. In contrast, digital correction (Figure 6.1) dispenses with the analog calibration hardware and instead corrects DAC errors with a lookup table (LUT), that ensures the digital data fed to the decimation filter accurately reflects the analog output of the DAC [1]. As illustrated in Figure 6.1, this technique effectively puts the DAC in the forward path of the loop since the LUT and the DAC outputs are equal. As a result, the DAC element errors (ee) are shaped by the NTF. The drawbacks of both analog calibration

and digital correction are that the DAC errors need to be measured accurately, and that system performance will degrade if those errors drift.
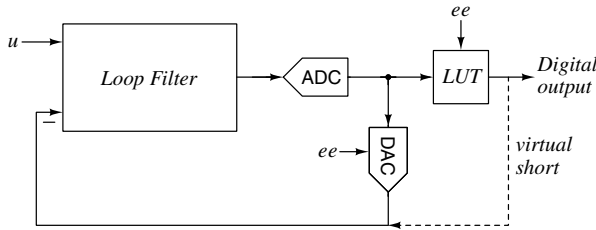
Figure 6.1   Digital correction of DAC errors in a $\Delta\Sigma$ ADC [1].

This chapter describes techniques that shape mismatch-induced errors. The remarkable feature of these methods is that they are blind – no knowledge of the actual errors is needed, and thus slowly-changing errors are accommodated automatically.

6.2   Random Selection and Rotation

Consider a 3-level DAC constructed from two nominally-equal elements. Without loss of generality, we can assume that the DAC is unipolar, and that the average value of the two elements is unity.[1] Under these assumptions the nominal outputs of the DAC corresponding to input codes 0, 1, and 2 are likewise 0, 1, and 2. If the elements are mismatched, such that one has value $1 + \epsilon$ and the other has value $1 - \epsilon$, then the endpoints are unaffected, but the middle level is either ϵ too high or ϵ too low. If the same element is always used to construct the middle level, then the DAC acts as a static nonlinearity and consequently produces distortion. (Since a 3-level DAC's transfer characteristic can be exactly described by a quadratic, the distortion is purely second order.) However, if the element used to construct the middle level is instead chosen randomly, then element mismatch turns into white noise. Having realized that we can whiten the error caused by element mismatch in a two-element DAC, the natural question for an apostle of $\Delta\Sigma$ is "Can we shape it?"

One interpretation of the $\Delta\Sigma$ gospel is that it is acceptable to make an error now, as long as we make up for it later. When responding to the middle code, choosing element 1 results in an error of ϵ, whereas choosing element 2 results in an error of $-\epsilon$. Thus, when responding to the middle code, we should choose element 2 after choosing element 1, and vice versa. To see that the error is shaped, consider the hypothetical input sequence

$$dac\ input = \{0, 0, 1, 0, 2, 1, 1, 1, 2, ...\}. \tag{6.1}$$

If we follow the alternating selection rule above, then the error sequence is

$$dac\ error = \{0, 0, \epsilon, 0, 0, -\epsilon, \epsilon, -\epsilon, 0, ...\}, \tag{6.2}$$

and thus the integrated error is

$$integrated\ error = \{0, 0, \epsilon, \epsilon, \epsilon, 0, \epsilon, 0, 0, ...\}. \tag{6.3}$$

---

[1]In essence, we define "unity" as the average of the DAC's two elements. Of course, this average value will vary from DAC to DAC, but such variation is equivalent to variability in the DAC's full-scale. We can ignore this variation when considering DAC linearity, just as we did with one-bit DACs.

Note that consecutive ϵ values in the integrated error start when element 1 is used in response to code 1 and terminate on the next code 1 when element 2 is used. Since this integrated error sequence is bounded, we can conclude that the actual error, obtained by differentiating the integrated error, is (at least) first-order shaped.
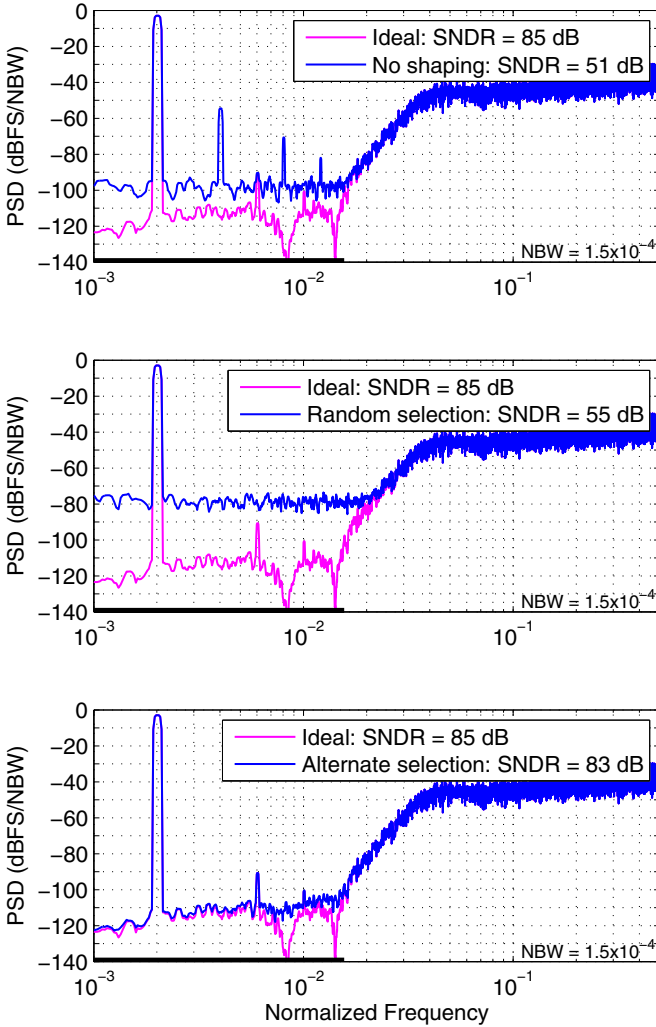
Figure 6.2   Average output PSDs for a 2-element DAC with 1% element mismatch.

Figure 6.2 compares the aforementioned element-selection schemes in the context of a two-element DAC driven by a three-level, fifth-order $\Delta\Sigma$ modulator with $||H||_\infty = 1.5$ and operated at $OSR = 32$. With a perfect DAC and a -3-dBFS input, the simulated SNDR is 85 dB. Figure 6.2(a) shows that if the elements have $\sigma = 1\%$ variation and the standard static selection strategy is used, then the SNDR degrades to 50 dB. The spectrum also contains several even-order harmonics, including a strong (-53-dBFS) second harmonic. Random selection (Figure 6.2(b)) eliminates the distortion terms, but the SNDR

is still 30 dB worse than the ideal SQNR. Figure 6.2(c) demonstrates that our alternate se-
lection strategy restores the SNDR to within 2 dB of ideal and completely eliminates the
DAC-induced harmonics. (The small third-order harmonic is present in the modulator data
itself.) For this 2-element example, dynamic element selection appears to work very well.
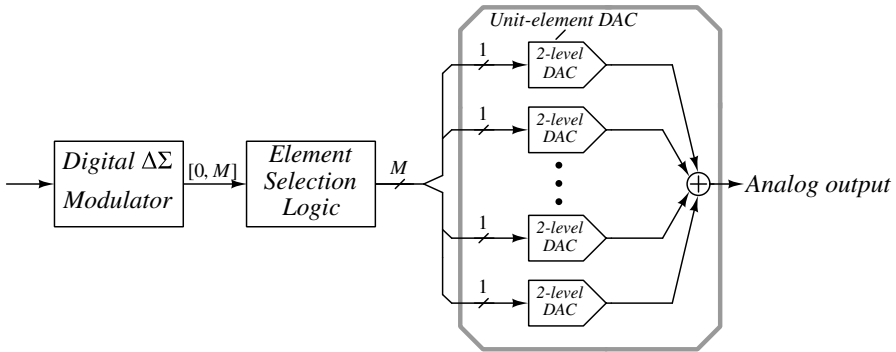
Figure 6.3   A $\Delta\Sigma$ DAC system.

To study the M-element case, a more aggressive fifth-order digital modulator with
$||H||_\infty = 2.5$ operated at a reduced OSR of 16 will be used. The $\Delta\Sigma$ DAC arrangement is
illustrated in Figure 6.3, and is evaluated for the effect of 1% DAC element variation.
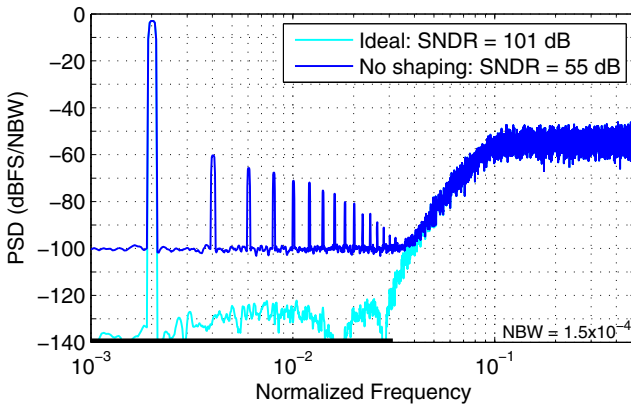
Figure 6.4   Average PSD for a 16-element DAC with 1% element mismatch – no shaping.

Starting with Figure 6.4, we see that if a static element selection strategy is used, then
mismatch degrades the SNDR from an ideal value of 101 dB to 55 dB and creates many
large harmonics. Using random selection (Figure 6.5) eliminates the distortion caused by
DAC mismatch and creates a flat noise floor. But at −62 dBFS, the in-band noise power is
still 40 dB higher than the ideal case.

Let us take a moment to compare this simulation result with a theoretical estimate.
For an M-element DAC whose elements have independent errors with a standard deviation
σ_{ee}, the variance of the difference between the DAC output for code m and the line defined
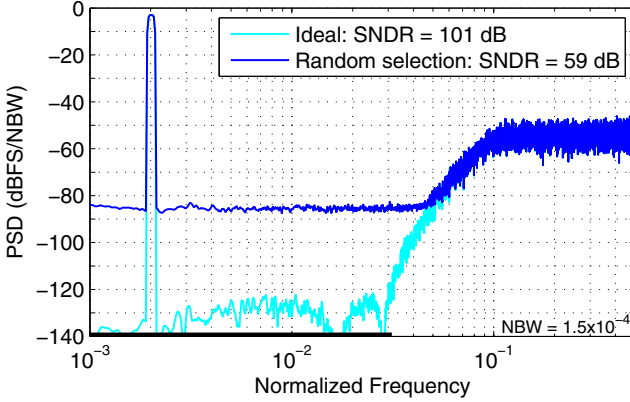
Figure 6.5   Average PSD for a 16-element DAC with 1% element mismatch– random selection.

by the endpoints is[2]

$$\sigma_m^2 = \frac{2m(M - m)}{M} \sigma_{ee}^2. \tag{6.4}$$

For a signal that spans many levels, the average value of σ_m^2

$$\frac{1}{M + 1} \sum_{m=0}^{M} \sigma_m^2 = \frac{(M - 1)}{3} \sigma_{ee}^2 \approx M\sigma_{ee}^2/3 \tag{6.5}$$

can be used to estimate the power of the mismatch noise. For $\sigma_{ee} = 1\%$, $M = 16$ and $OSR = 16$, and assuming the mismatch noise is white, the noise due to element mismatch has an in-band power relative to the power of a full-scale sine wave of

$$MNP = \frac{M\sigma_{ee}^2/3}{(OSR)(M/2)^2/2} = \frac{8\sigma_{ee}^2}{3(M)(OSR)} = -60 \text{ dBFS}. \tag{6.6}$$

Since the signal power is -3 dBFS, our SNDR estimate is therefore 57 dB, which is close to the 59 dB result given in Figure 6.5.

To understand how to shape the mismatch of M elements, interpret the $\Delta\Sigma$ philosophy as "If you make an error in the current cycle, then try to make the negative of that error in the next cycle." Since a DAC has no error for codes 0 and M, the negative of the error associated with using certain elements equals the error of using the other elements. Thus, if we select elements 1 to $v[0]$ at $t = 0$, then at $t = 1$ we ought to select the remaining elements. However, we are only allowed to select $v[1]$ elements. Thus, we select elements $v[0] + 1$ to $v[0] + v[1]$, and in so doing, we commit the error of not selecting the remaining elements. To make up for that error, we continue selecting elements sequentially in subsequent cycles until all elements 1 to M have been used once, at which point the accumulated error is

[2]Defining e_i as the value of element i, the error for code m is

$$\sum_{i=1}^{m} \left(e_i - \frac{1}{M} \sum_{i=1}^{M} e_i \right) = \left(1 - \frac{m}{M} \right) \sum_{i=1}^{m} e_i - \frac{m}{M} \sum_{i=m+1}^{M} e_i.$$

(6.4) follows directly.
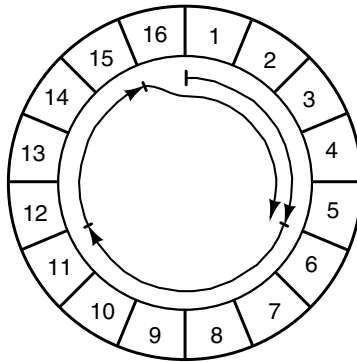
Figure 6.6 Rotational element selection for the sequence {5, 6, 4, 6}.

zero and selection returns to the beginning of the element array. We expect this *rotational element selection* strategy, depicted in Figure 6.6, to produce first-order shaped noise. This expectation is confirmed in Figure 6.7, where we see that the noise due to mismatch now has the characteristic 20-dB/decade slope of first-order shaping.
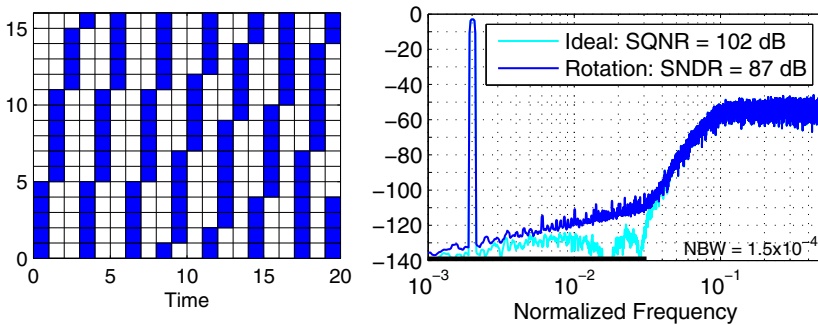
Figure 6.7 Example usage pattern and spectrum for rotation.

Also visible in Figure 6.7 are harmonics of the signal as well as spurs that also tend to follow a first-order slope. Although the spurs in Figure 6.7 are all below −100 dBFS, simulation with a small (−30 dBFS) input in the vicinity of $f \approx 1/64$ reveals a mismatch-induced second harmonic (H_2) that approaches −80 dBFS. Since reducing *OSR* would make the harmonic even larger, element rotation is less attractive when *OSR* is low. Furthermore, since the preceding spectra are ensemble averages, the designer must also leave sufficient margin to achieve adequate production yield.

To quantify the required margin, Figure 6.8 plots the cumulative distribution function (CDF) of H_2 obtained by a Monte Carlo simulation. The results indicate that for 99.9% yield, 12 dB of margin is needed above the median H_2 value. Schemes that can reduce mismatch-induced spurs are therefore clearly of interest.

Before addressing the spur problem, let's try to quantify the noise. According to Figure 6.7, with 1% variation on 16 elements, rotation yields an in-band mismatch noise power (*MNP*) of −90 dBFS at *OSR* = 16. To obtain a theoretical estimate of *MNP*, first
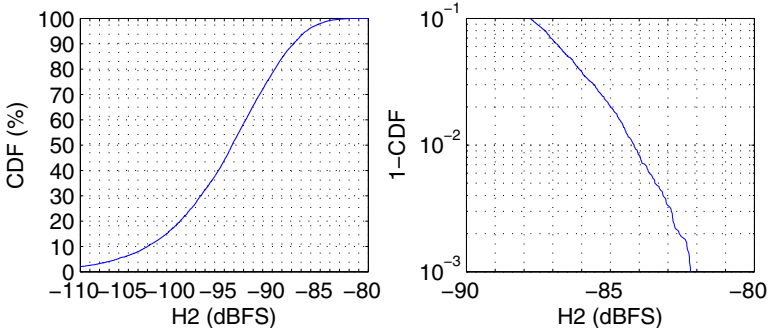
Figure 6.8 H_2 cumulative distribution function for a -3 dBFS signal at $f_s/(4OSR)$. ($OSR = 16$, $M = 16$, $\sigma_{ee} = 1\%$, rotational selection.)

recall that the in-band power of white noise with power P that is first-order shaped is

$$\frac{P}{\pi} \int_0^{\frac{\pi}{OSR}} \omega^2 d\omega = \frac{\pi^2 P}{3(OSR)^3}. \tag{6.7}$$

At each instant, the integrated mismatch error is equal to the sum of the mismatch errors of the m elements that have been selected one time more than the other elements. The power of this signal is thus given by averaging (6.4) over m to obtain the same result as (6.5), i.e., $P = M\sigma_{ee}^2/3$. If we make the dubious assumption that the integrated error is white noise, then the in-band mismatch noise power (*MNP*), relative to the power of a full-scale sine wave ($M^2/8$) is

$$MNP = \frac{\pi^2 M \sigma_{ee}^2}{9(OSR)^3(M^2/8)} = \frac{8\pi^2 \sigma_{ee}^2}{9M(OSR)^3} = -76 \, \text{dBFS}, \tag{6.8}$$

which over-estimates the simulated value by 14 dB. Clearly, this analytical calculation is too conservative for design purposes and thus simulations are required to achieve adequate accuracy.

Element rotation was introduced in the technical literature in 1995 [2] where it was dubbed *data-weighted averaging* (DWA) to contrast with the less effective *individual-level averaging* (ILA) [3] element-selection scheme. Although the term DWA is currently in common use and appeals to the engineer's fondness for TLAs (three-letter acronyms), we consider it to be less descriptive than "rotation." Furthermore, since element rotation was described in a patent two years earlier [4], our preference does not violate the convention of giving naming priority to the first inventor.

6.3 Implementation of Rotation

In a $\Delta\Sigma$ DAC system, the complexity of the element selection logic (ESL) is somewhat secondary because latency is not particularly problematic. However, in a $\Delta\Sigma$ ADC system, the latency of the DAC feedback typically needs to be a fraction of the clock period, and thus for high-speed applications the ESL must be simple. Fortunately, implementing rotation is easy.
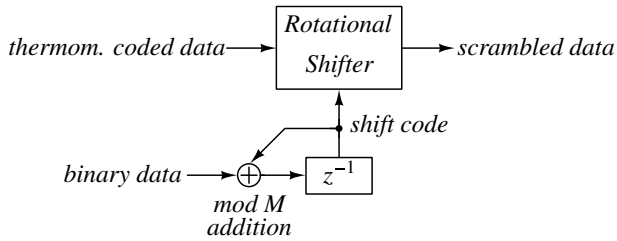
Figure 6.9   Implementation of element rotation.

Figure 6.9 shows an implementation of rotation in which the thermometer-coded output of the flash ADC is applied to a rotational shifter whose shift code is the output of a digital integrator. Think of the output of the integrator as a pointer that points to the start of the unused elements. At the end of each cycle, the pointer is incremented by the number of elements selected, modulo M, so that the pointer always points to the start of the unused elements. Since the current pointer value is independent of the current data, updating the pointer is not a time-critical operation.
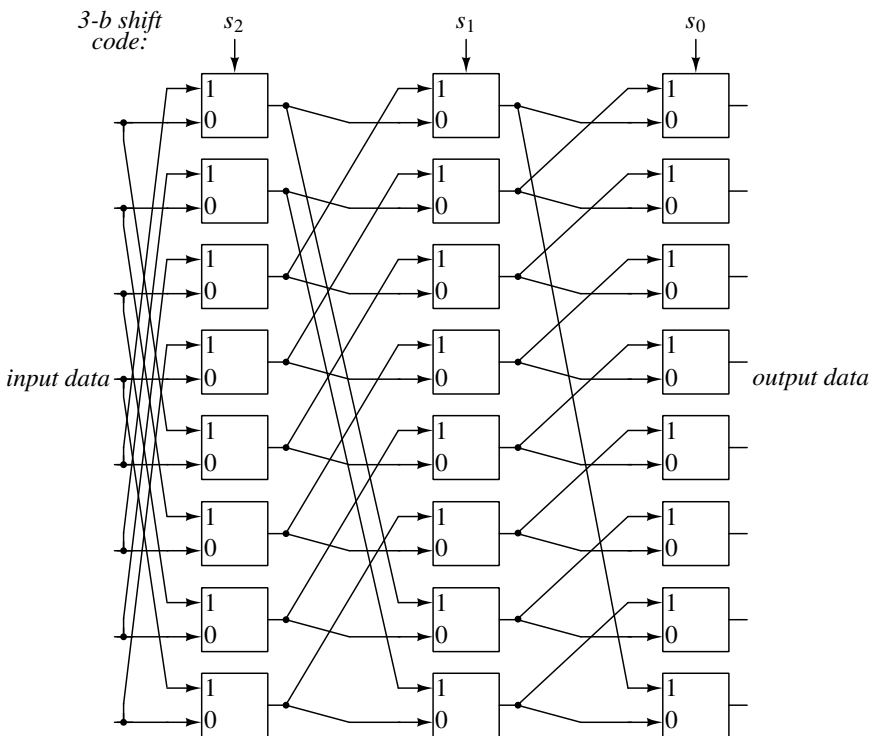
Figure 6.10   Logarithmic shifter ($M = 8$).

The delay of the shifter does eat into the time available for comparator regeneration, and can therefore be time-critical in a high-speed design. Figure 6.10 depicts a fast rotator structure whose delay is $t_d \lceil \log_2 M \rceil$, where t_d is the delay of a 2-input MUX. Two opti-

mizations can be used to reduce the delay of the MUX itself. First, as shown in Figure 6.11, eliminate the inverter needed to make a non-inverting MUX. Second, since the shift code (i.e., the pointer) is typically available in advance of the data, use large devices connected to the S and $\overline{S}$ signals. This arrangement increases the drive strength of the MUX without increasing the load capacitance presented by subsequent MUXes, and thereby minimizes the delay of the shifter.

$* = Large\ devices$
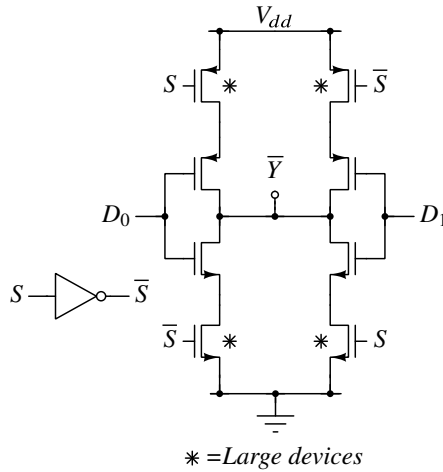
Figure 6.11   MUX with reduced D-Y delay.
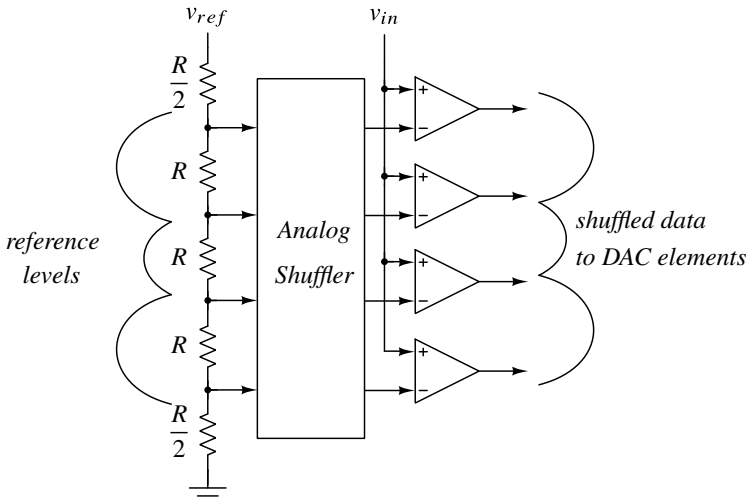
Figure 6.12   Shuffling of reference levels ($M = 4$).

Even with such optimizations, the shifter adds delay to the critical regeneration-plus-DAC-setup time and so can be a speed bottleneck. Instead of rotating the comparator outputs after regeneration, the comparator inputs (i.e., the reference levels) can be rotated

in advance of initiating regeneration as depicted in Figure 6.12 [7]. This *reference-shuffling* technique maximizes the time available for regeneration, but requires an analog shifter. The analog shifter can be constructed using the logarithmic shifter topology, but a structure that uses M M-input analog MUXes, configured either as an arbitrary-connection permutation (Figure 6.13(a)) or hardwired for rotation (Figure 6.13(b)), is usually faster.
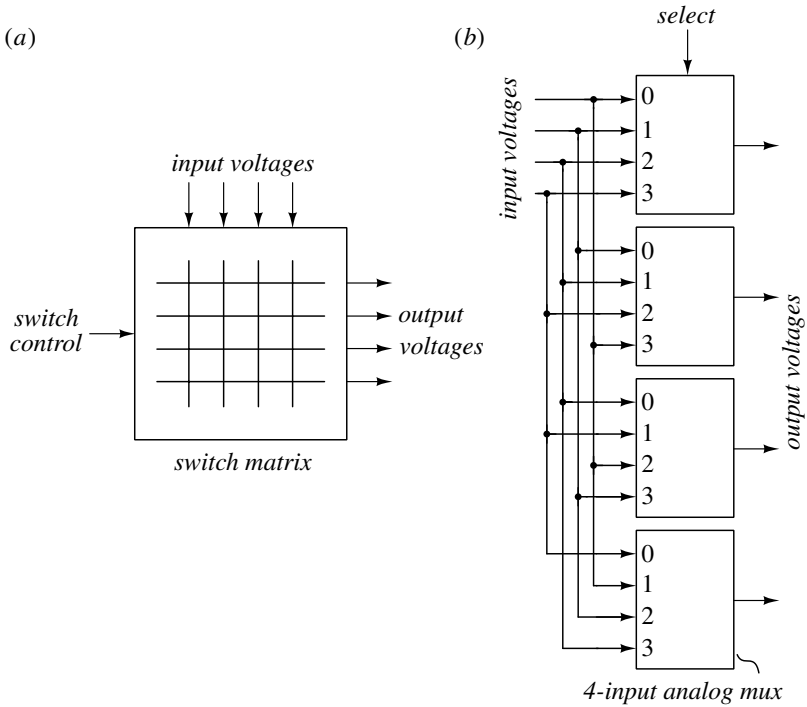
Figure 6.13   Analog shufflers ($M = 4$): (a) Arbitrary permutation (b) hardwired rotation.
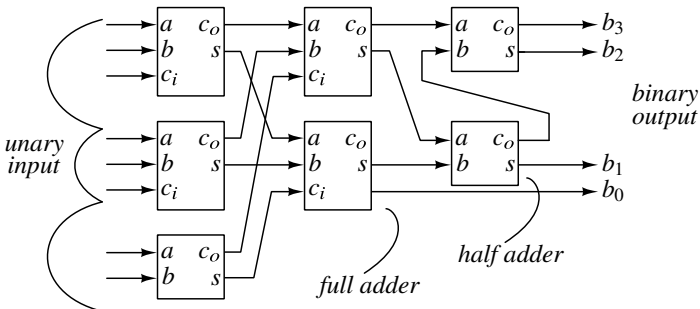
Figure 6.14   Unary-to-binary converter ($M = 8$).

Note that with reference shuffling, the conversion to binary must be done on shuffled data, necessitating the use of a *unary-to-binary converter*, rather than a thermometer-to-binary converter. A unary-to-binary converter is simply an adder with M 1-bit inputs and

is most conveniently implemented with a tree of full- and half-adder cells as shown in Figure 6.14.

It is rather fortuitous that implementing the rotation scheme is easier than the less-effective random-selection strategy. To see why, note that fully-random selection requires circuitry that supports all $M!$ possible permutations, whereas rotation uses only M permutations. If we content ourselves to support the latter number instead, then the preceding structures with a random pointer can be used to implement partially-random selection.

6.4   Alternative Mismatch-Shaping Topologies

This section gives brief descriptions of several alternatives to rotation that provide reduced tonal behavior at the expense of increased complexity and reduced mismatch suppression, culminating in an arrangement that has been rigorously *proven* to be tone-free.
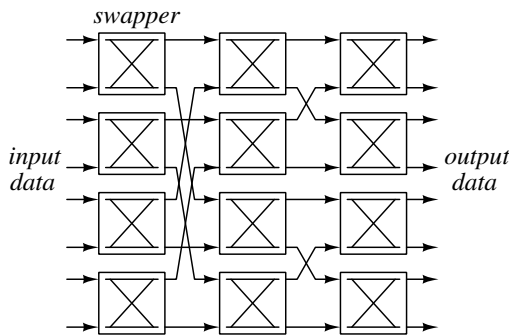
Figure 6.15   Butterfly shuffler.

6.4.1   Butterfly Shuffler

The butterfly shuffler depicted in Figure 6.15 consists of $\log_2 M$ columns of $M/2$ swapper cells arranged in a manner similar to the butterfly operations of an FFT [8]. Each swapper cell operates independently according to the following two rules:

   a. If the two incoming bits are the same, just pass them on.

   b. If the input bits are different, route the '1' to the bottom output if the most recent lone '1' was routed to the top output, and vice versa.

This scheme equalizes the average 1s-density of the cell's two outputs, and the multiple layers of swapper cells ensure equal 1s-density among all of the shuffler's outputs, thereby accomplishing first-order shaping.

In contrast to a rotational shuffler which has $\log_2 M$ bits of state, there are $M/2 \log_2 M$ bits of state in a butterfly shuffler. This extra state information leads to more exotic element usage patterns (Figure 6.16) and a reduced likelihood of periodic behavior compared to plain rotation. The reduced periodicity is accompanied by a reduction in the visible harmonics.
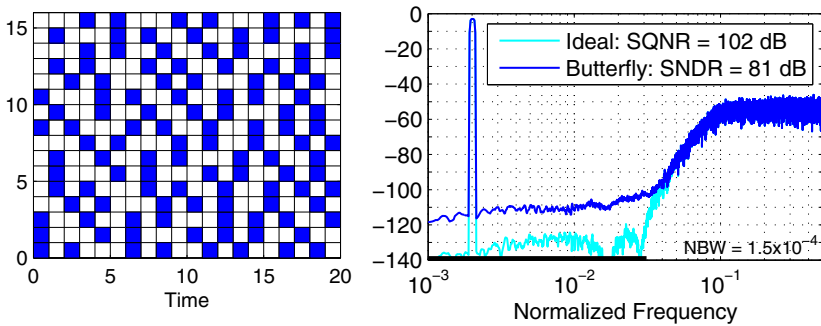
Figure 6.16 Example butterfly shuffler usage pattern and spectrum.

6.4.2 A-DWA and Bi-DWA
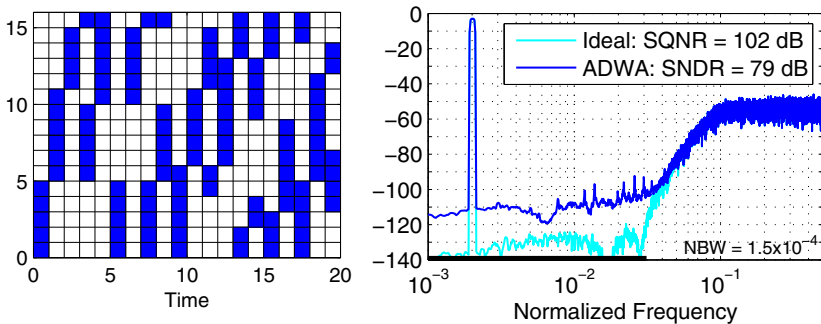
Figure 6.17 Example A-DWA usage pattern and spectrum (increment = 5).

The reason why plain rotation produces spurs is that it is deterministic. One can envision adding randomness by renumbering the elements whenever they have all been used equally, but such a scheme is complex to implement in hardware. Instead, *advancing data-weighted averaging* (A-DWA) [9] increments the starting value of the pointer each time it completes a full rotation. Figure 6.17 shows a simulated usage pattern and spectrum using the recommended increment of $\lfloor M/3 \rfloor = 5$. The simulated spectrum exhibits tones that are less distinct than rotation, but the tonality and SNDR are worse than those of a butterfly shuffler.

Other ideas have been tried, but a particularly simple and effective one, dubbed *bi-directional data-weighted averaging* (Bi-DWA)[10], involves interleaving forward and backward rotation as illustrated in Figure 6.18. Since the mismatch spectrum in Figure 6.18 has clear shaping and is remarkably free of tones, and since further simulations indicate these properties are robust, the 6-dB SNDR penalty compared to plain rotation would seem to be a price worth paying.
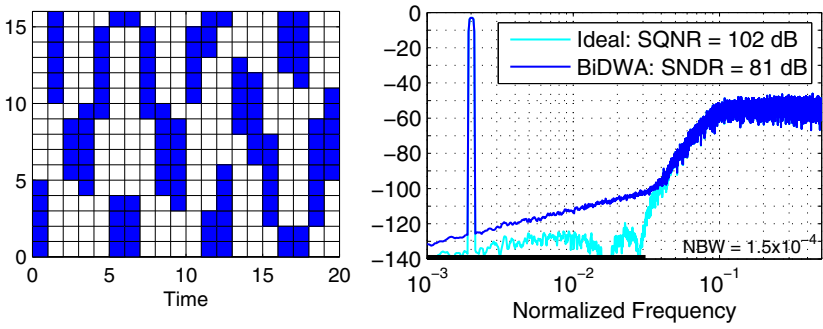
Figure 6.18   Example Bi-DWA usage pattern and spectrum.
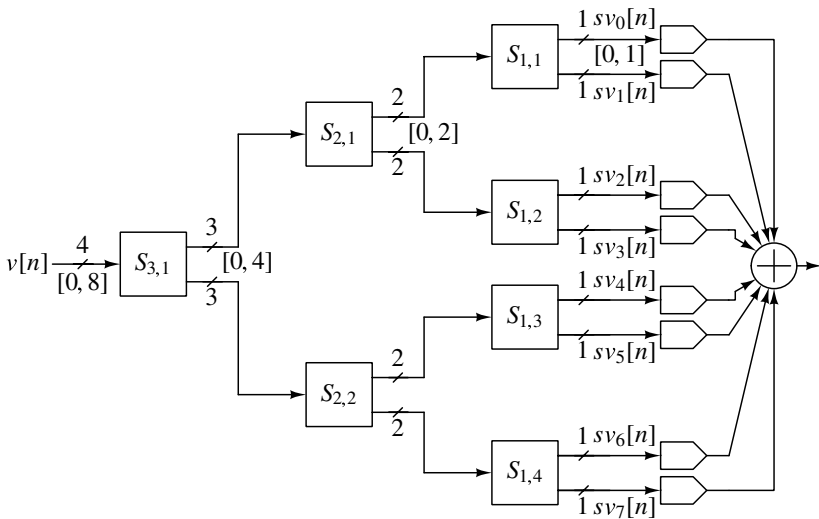
Figure 6.19   Tree-structured 8-element mismatch-shaping DAC.

6.4.3 Tree-Structured ESL

Figure 6.19 depicts the final first-order mismatch-shaping system that we consider. In this system, incoming $(m + 1)$-bit data representing values in $[0, 2^m]$ is successively split into two reduced-width data streams until the bit width is unity. The resulting 2^m one-bit sv_i signals select which of the 2^m DAC elements are enabled. (sv stands for *selection vector*.)
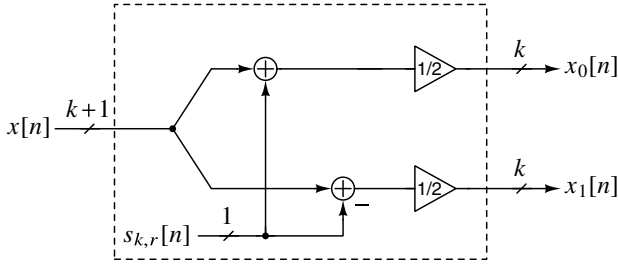
Figure 6.20   Signal processing equivalent of a switching block.

Each of the *switching blocks*, $S_{k,r}$, in Figure 6.19 implements the signal-processing shown in Figure 6.20. (The k subscript indicates the layer number of the block, while r gives the block location within the kth layer. The layer numbers, k, go from right to left so that the bit-width of $S_{k,r}$'s outputs is k.) From Figure 6.20 we see that the sum of each block's two outputs is equal to its input. The $s_{k,r}[n]$ signal is chosen to ensure that the division by 2 results in an integer and is also chosen to be a shaped sequence. Specifically, abbreviating $s_{k,r}[n]$ with $s[n]$,

$$s[n] = \begin{cases} 0, & x[n] \text{ even,} \\ +1, & x[n] \text{ odd and previous non-zero sample of } s = -1, \\ -1, & x[n] \text{ odd and previous non-zero sample of } s = 1, \end{cases} \quad (6.9)$$

which is the same rule we used in our alternate-selection strategy for a 2-element DAC.
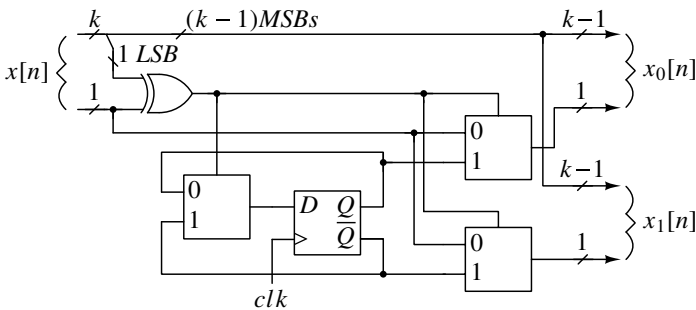
Figure 6.21   Example switching block implementation.

Figure 6.21 shows an implementation of a switching block that follows (6.9) [11]. In this diagram, a $(k + 1)$-bit digital signal representing values in $[0, 2^k]$ consists of a k-bit signal representing values in $[0, 2^k - 1]$ plus a 1-bit signal representing the values 0 and 1. This *redundant LSB representation* eliminates the adders depicted in Figure 6.20 and hence achieves both simplicity and speed.
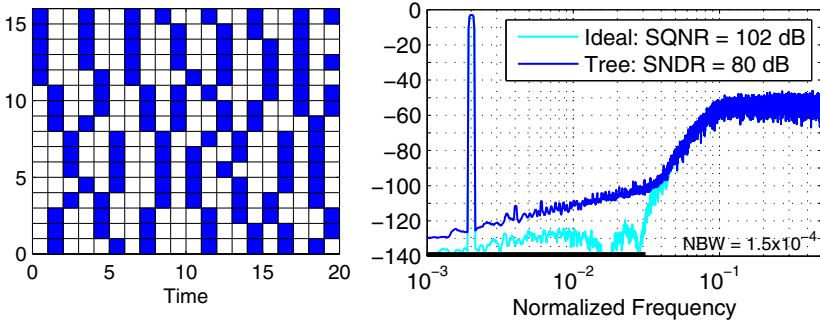
Figure 6.22   Simulation of first-order tree-structured mismatch shaping.

Since the sum of the outputs of a switching block equals its input, the sum of the components of sv equal the shuffler input v, and thus the nominal output of the DAC is v. Also, since each component of sv is equal to v/M plus a linear combination of the s sequences used within the switching blocks, and since the s sequences are first-order shaped, the mismatch-induced noise is first-order shaped. Figure 6.22 shows an example usage pattern and spectrum. For the element usage plot, the elements are numbered in bit-reversed order to emphasize the initial similarity with rotation. Looking at the spectrum, we see some evidence of harmonics and an SNDR that is 7 dB worse than rotation.

However, this is not the end of the story. Recall that the alternate-selection strategy chooses element 2 after choosing element 1, and vice versa, whereas our understanding of rotation suggests that after completing a rotation, meaning after using both elements equally, it is allowable to renumber the elements. Such renumbering is difficult to implement when the number of elements is large, but with only two elements the hardware is trivial. For the tree-structured mismatch shaper, the s sequences are now chosen according to

$$s[n] = \begin{cases} 0, & x[n] \text{ even,} \\ +1, & x[n] \text{ odd and } ss[n] = -1, \\ -1, & x[n] \text{ odd and } ss[n] = +1, \\ r[n], & x[n] \text{ odd and } ss[n] = 0, \end{cases} \tag{6.10}$$

where $ss[n] = \sum_{i=0}^{n-1} s[i]$ and $r[n]$ is a random bit taking on values of ± 1 with 50% probability. (Note that r, s, and ss are local to each switching block.) The reader can verify that these rules imply the sum sequence ss is bounded by 1 in magnitude, namely $|ss[n]| \leq 1$, and thus we know that s is first-order shaped. More impressive is the fact that the PSD of s has been proven to be smooth, and thus the mismatch noise is guaranteed to be devoid of tones [12].

Figure 6.23 confirms this amazing property, but also indicates the price is a 6-dB SNR degradation compared to the similarly-effective Bi-DWA scheme. The logic that implements a switching block operating according to (6.10) is depicted in Figure 6.24 [11].
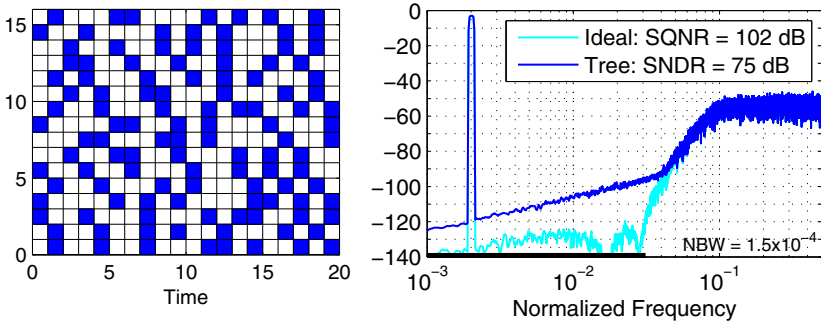
Figure 6.23   First-order tree-structured mismatch-shaping, with dither.
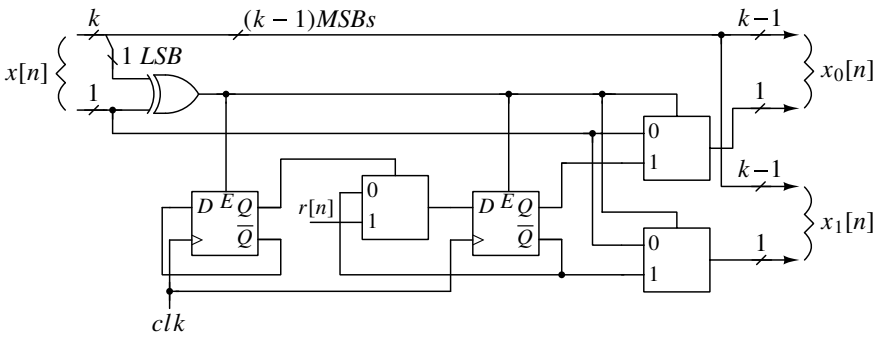
Figure 6.24   Logic implementing a dithered switching block.

6.5 High-Order Mismatch-Shaping

Having seen several ways to first-order shape mismatch-induced noise, the reader may wonder if high-order shaping is also possible. This section shows that high-order shaping of mismatch is indeed possible, but is subject to stability constraints that are at least as severe as those associated with binary $\Delta\Sigma$ modulation.

6.5.1 Vector-Based Mismatch-Shaping
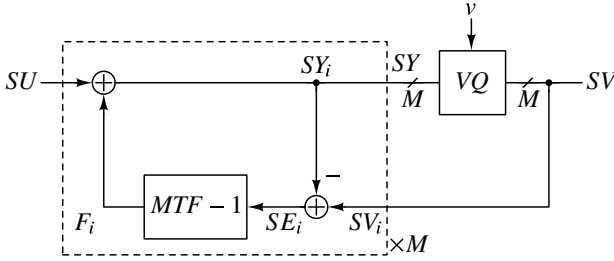
Figure 6.25   Vector-based mismatch-shaping.

Figure 6.25 depicts a system that is able to shape mismatch via an arbitrary transfer function $MTF(z)$ [13]. This system consists of M identical filters whose M outputs are quantized to M 1-bit signals sv_i that are in turn fed back to the M filters. Since each loop is an error-feedback modulator with a common input SU, $\Delta\Sigma$ theory tells us that

$$SV(z) = SU(z) + MTF(z)SE(z), \tag{6.11}$$

where the bold font denotes signals that are (row) vectors. We require the M-input quantizer, hereafter called the *vector quantizer* (VQ), to obey

$$sv[n] \cdot [1\ 1\ \dots 1] = \sum_{i=1}^{M} sv_i[n] = v[n], \tag{6.12}$$

where $\cdot$ denotes the dot product,[3] so that the nominal output of the DAC is v. (This constraint is indicated diagrammatically in Figure 6.25 by the v input to VQ.) As before, we can assume without loss of generality that the average element value is unity. Thus, the DAC output is

$$D(z) = SV(z) \cdot ([1\ 1\ \dots 1] + ee), \tag{6.13}$$

where ee, the *element error* vector, contains the deviation of individual elements from their average and thus satisfies

$$ee \cdot [1\ 1\ \dots 1] = 0. \tag{6.14}$$

Now, by virtue of (6.12)

$$SV(z) \cdot [1\ 1\ \dots 1] = V(z) \tag{6.15}$$

---

[3] $sv[n] \cdot [1\ 1\ \dots 1] = sv[n][1\ 1\ \dots 1]^T = \sum_i sv_i$.

while from (6.11) and (6.14),

$$SV(z) \cdot ee = SU(z)([1\,1\ldots1] \cdot ee) + MTF(z)(SE(z) \cdot ee)$$
$$= MTF(z)(SE(z) \cdot ee). \tag{6.16}$$

Thus, the DAC output is given by

$$D(z) = V(z) + MTF(z)(SE(z) \cdot ee), \tag{6.17}$$

which shows that the output of the DAC consists of the desired signal plus a term shaped by $MTF(z)$. As long as we can ensure that se is bounded, element errors will therefore result in noise that is shaped by $MTF(z)$.

To maximize the likelihood that se is bounded, the scheme used to quantize sy to sv subject to (6.12) is chosen to minimize the instantaneous se signal. Specifically, the $v[n]$ elements in $sv[n]$ that correspond to the largest values of $sy[n]$ are set to 1 and the other elements are set to 0. To remove commonality in the components of sy, $su[n]$ can be set to the negative of the smallest value of $f[n]$ so that $sy[n]$ consists of positive numbers and at least one zero. This choice is quite arbitrary; other choices such as the negative of the average value of $f[n]$ would also serve the purpose of removing commonality.

To make the discussion above more concrete, let's work through the operation of the element selection logic for $MTF(z) = 1 - z^{-1}$ with an 8-element DAC driven by the sequence $v[n] = \{1, 1, 2, 3, 4\}$ and starting from $sy[0] = [0\,0\,0\,0\,0\,0\,0\,0]$. To simplify the discussion, we will assume for the moment that su is zero.

Since all the components of $sy[0]$ are equal, there is no preference for choosing one element over another. Therefore, to satisfy $\sum_i sv_i[0] = v[0] = 1$, we choose the first element:

$$sv[0] = [1\,0\,0\,0\,0\,0\,0\,0]. \tag{6.18}$$

Since $MTF(z) - 1 = -z^{-1}$, the recursion equation with $su = 0$ is

$$sy[n+1] = -se[n] = sy[n] - sv[n], \tag{6.19}$$

and thus

$$sy[1] = [\,-1\,0\,0\,0\,0\,0\,0\,0\,]. \tag{6.20}$$

With $v[1] = 1$ we are again faced with ambiguity regarding which element among elements 2 to 8 to select, and so we choose the first of those elements:

$$sv[1] = [0\,1\,0\,0\,0\,0\,0\,0]. \tag{6.21}$$

Continuing in this vein, we find

$sy[2]$	$=$	[	-1	-1	0	0	0	0	0 0	]
$sv[2]$	$=$	[	0	0	1	1	0	0	0 0	]
$sy[3]$	$=$	[	-1	-1	-1	-1	0	0	0 0	]
$sv[3]$	$=$	[	0	0	0	0	1	1	1 0	]
$sy[4]$	$=$	[	-1	-1	-1	-1	-1	-1	-1 0	]
$sv[4]$	$=$	[	1	1	1	0	0	0	0 1	]

demonstrating that the resulting usage pattern is identical to element rotation. As is also apparent in this example, choosing su as the negative of the minimum of sy keeps all

components of sy positive and prevents them from growing in magnitude together. In fact, with $MTF(z) = 1 - z^{-1}$, the components of sy are either 0 or 1, implying that a single bit is sufficient for these signals.

Let's now use (6.17) to estimate the in-band mismatch noise.[4] Each of the M components of se is the quantization error sequence of a 1-bit modulator. In the current context, the quantization levels are 0 and 1. If we assume that the quantization error is uniformly distributed in $[-0.5, 0.5]$, then the quantization error power is $1/12$. If we further assume that these error sequences are white and uncorrelated with each other, then the power of the $(SE(z) \cdot ee)$ term is $M\sigma_{ee}^2/12$. Following the derivation for (6.8), the in-band mismatch power relative to the power of a full-scale sine wave is therefore

$$MNP = \left(\frac{M\sigma_{ee}^2}{12}\right)\left(\frac{\pi^2/3}{OSR^3}\right)\frac{8}{M^2} = \frac{2\pi^2\sigma_{ee}^2}{9M(OSR)^3} = -82\,\text{dBFS}. \tag{6.22}$$

This estimate is 6 dB closer to the simulated value of -90 dBFS than the estimate of (6.8). However, since the estimate is high by 8 dB, simulation is still recommended to quantify the effectiveness of the rotation scheme.

In addition to showing how element rotation can be derived, the preceding example also suggests a benign way to add dither to the element selection process. Simply adding a random value to each component of sy allows ties between elements to be broken in a random way. Unfortunately, when this scheme, or high-order MTFs, are used, the element usage pattern typically becomes indecipherable, and thus the structure illustrated in Figure 6.25 needs to be implemented explicitly.

The most complex block in this system is the vector quantizer. The $\Delta\Sigma$ toolbox function `simulateMS` (described on page 514) uses a sort operation to determine element priority. Sorting is a standard software operation, but a hardware sort can be prohibitively complex. The hardware becomes more manageable with partial sort [14]. As an alternative to sorting, the selection of the $v[n]$ largest components of $sy[n]$ can be achieved by finding the threshold $r[n]$ that yields $v[n]$ ones from M digital comparisons between $r[n]$ and the components of $sy[n]$. This method avoids the sorting operation but requires iteration to find $r[n]$.

Since the ESL effectively consists of M 1-bit $\Delta\Sigma$ modulator loops with extra constraints on the one-bit quantizers, the stability of the ESL is typically worse than a plain one-bit $\Delta\Sigma$ modulator. Thus, when high-order mismatch-shaping is desired, it is wise to take precautions such as restricting the DAC input to a fraction of the DAC full-scale (for example, by augmenting the DAC with extra elements), as well as implementing saturation and reset logic.

To demonstrate the effectiveness of high-order mismatch-shaping, we in Figure 6.26 show the expected performance of our example 16-element DAC with 1% element variation when a second-order MTF with $\|MTF\|_\infty = 1.5$ and zeros optimized for $OSR = 16$ is employed. We find that the SNDR is 4 dB higher than rotation and the spectrum appears to be free of spurs. At higher OSR, high-order mismatch-shaping gives increased noise suppression relative to first-order shaping, but at lower OSR the improvement is negligible.

---

[4]The authors thank Nan Sun for suggesting this improved method.
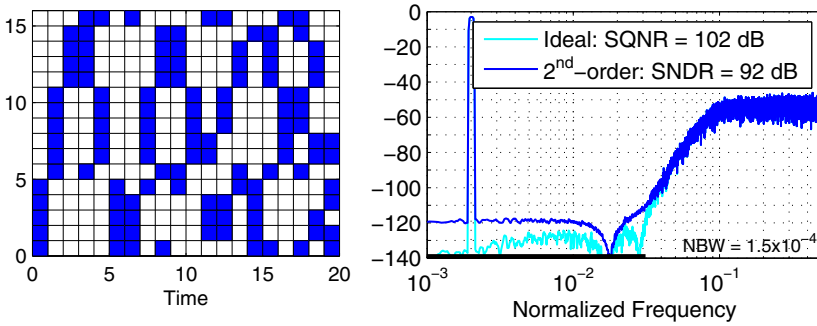
Figure 6.26   Example second-order mismatch-shaping usage pattern and spectrum.

6.5.2 Tree Structure

High-order mismatch-shaping can be accomplished with the tree structure of Figure 6.19 if, as before, the $s_{k,r}[n]$ signal of Figure 6.20 satisfies the constraint that the division by two results in a nonnegative integer less than 2^{k-1} and also satisfies

$$S_{k,r}(z) = MTF(z)E_{k,r}(z), \tag{6.23}$$

where $MTF(z)$ is the desired mismatch transfer function and $e_{k,r}[n]$ is a bounded sequence.
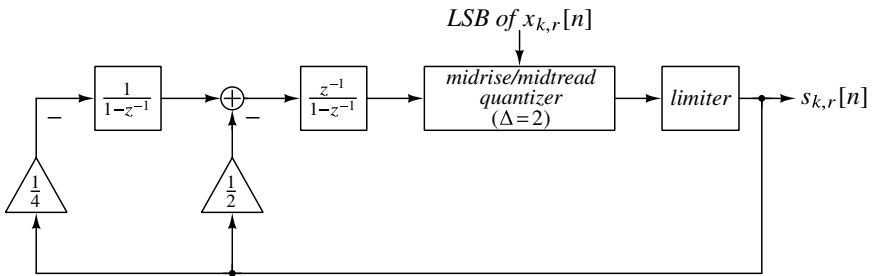
Figure 6.27   Second-order switching sequence generator.

As with the vector style, an $s_{k,r}$ signal can be generated by a digital $\Delta\Sigma$ modulator containing a modified quantizer. For example, Figure 6.27 shows a second-order modulator whose quantizer generates even values if $x[n]$ is even, and odd values if $x[n]$ is odd. To ensure the switching block's outputs are in the range $[0, 2^{k-1}]$, the values of s are further restricted to the range $[-L, L]$ where $L = min(x[n], 2^{k-1} - x[n])$. If we assume that the gain of the quantizer is unity, then the mismatch transfer function implemented by this structure is

$$MTF(z) = \frac{(z-1)^2}{z^2 - 1.25z + 0.5} \tag{6.24}$$

which has $\|MTF\|_\infty \approx 1.5$.

Figure 6.28 confirms the second-order shaping of mismatch, but unfortunately the SNDR is disappointingly low.
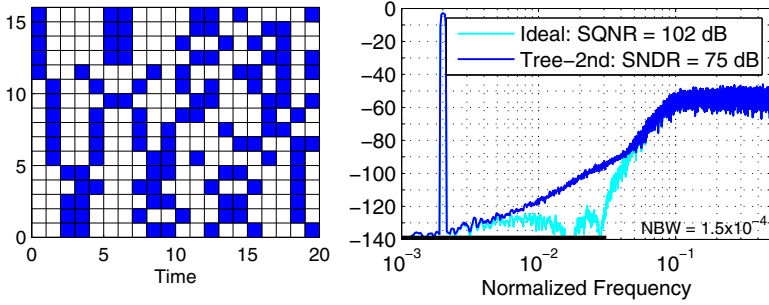
Figure 6.28   Example usage pattern and spectrum for second-order tree-structured mismatch-shaping.
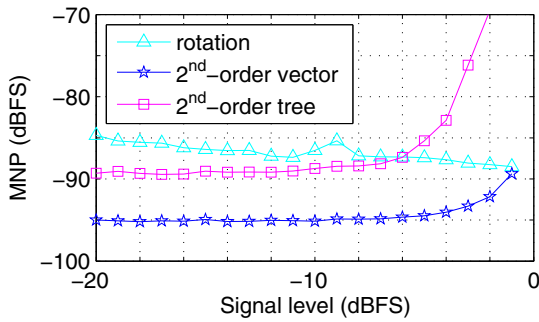
Figure 6.29   Simulated in-band mismatch noise power (*MNP*) vs. signal level. (*OSR* = 16.)

To gain perspective, Figure 6.29 compares the simulated in-band mismatch noise power (*MNP*) as a function of signal level for three styles of mismatch-shaping. For rotation, we find that *MNP* tends to decrease as the signal level increases, whereas for both second-order mismatch shapers *MNP* increases sharply above a critical signal level (-6 dBFS for the tree structure, -1 dBFS for the vector-based shaper). Our choice to use a -3-dBFS input level for the preceding comparisons therefore shows the second-order tree structure in an unduly unfavorable light. If we consider low signal levels, *MNP* with the second-order tree-structured mismatch-shaper is actually a few dB lower than with rotation. The vector-based shaper exhibits a further 5-dB advantage. Much of the difference between the two second-order shapers is due to the difference in their MTFs, but the vector-based approach appears to fare slightly better even when identical MTFs are used. We remind the reader that *MNP* improves with increased *OSR* at the rate of 15 dB per octave for both second-order shapers, whereas for rotation the rate of improvement is only 9 dB per octave. Thus, both second-order shapers are especially attractive when $OSR > 16$.

6.6   Generalizations

The focus thus far has been on DACs constructed with M nominally equal one-bit DAC elements. This section briefly describes two generalizations of mismatch-shaping, namely the use of tri-level and non-unit elements.

6.6.1   Tri-Level Elements
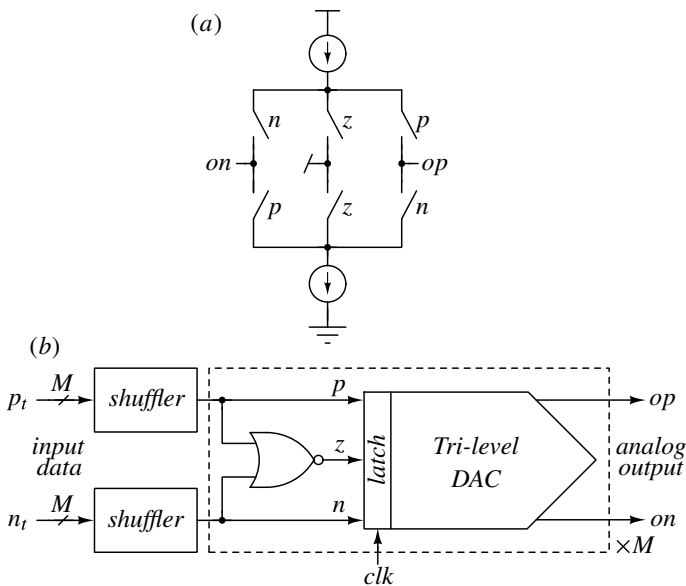
Figure 6.30   Tri-level DAC and tri-level mismatch-shaping.

Consider a tri-level DAC element that produces levels -1, 0, and $+1$ in response to the one-hot control signals n, z, and p, respectively, using a structure such as that depicted in Figure 6.30(a). Such tri-level elements are useful for two reasons. First, the spacing of the quantization levels in a quantizer that uses tri-level elements is half of what can be achieved with two-level elements (i.e., $\Delta = 1$ vs. $\Delta = 2$), and thus, for the same number of DAC elements, the quantization noise is reduced by 6 dB. Second, and more fundamental, is the fact that the thermal noise of a tri-level element is zero when the z signal is active.

To combine the benefits of tri-level elements and mismatch-shaping, the arrangement of Figure 6.30(b) can be used [15]. In this system, positive data is encoded as an M-bit thermometer code p_t with $n_t = 0$, and vice-versa for negative data. The p_t and n_t data are shuffled to produce the P and N data that drive the DAC elements; each bit in z is produced by NORing the corresponding n and p bits.

6.6.2 Non-Unit Elements
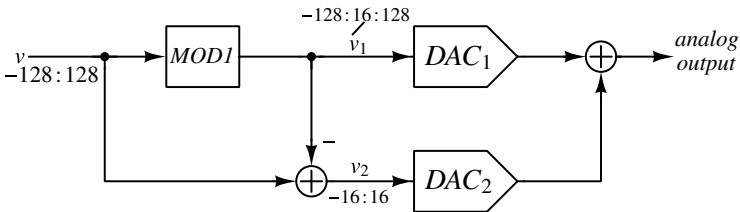
Figure 6.31   Segmented scrambling.

When M is large, the amount of digital hardware in the element selection logic is also large. Hardware requirements can be reduced by using the *segmented scrambling* technique [8]. The concept is illustrated in Figure 6.31 for a 257-level DAC whose output is constructed with 16 elements of weight 16 (DAC1) and 32 elements of weight 1 (DAC2). The incoming data V is decomposed into $V = V_1 + V_2$, where V_1 is quantized to multiples of 16 by a first-order $\Delta\Sigma$ modulator (MOD1), and V_2 is the difference between the input and output of MOD1. The properties of MOD1 guarantee $|V_2| \leq 16$ for $V \in [-128, 128]$, and thus the 32-element DAC2 has sufficient range to convert V_2 into analog form. Employing mismatch-shaping in DAC1 and DAC2 shapes intra-DAC mismatch. Since inter-DAC mismatch makes the output of the system proportional to $V_1 + (1 + \epsilon)V_2 = V + \epsilon V_2$, and since $V_2 = (1 - z^{-1})E_1$, where E_1 is MOD1's quantization error, inter-DAC mismatch is also shaped.

The segmented scrambling technique can be applied recursively. For example, [16] takes this approach to the extreme with a fully-segmented DAC using 14 levels of recursion to shape the mismatch of 28 elements whose weights are $\{2^{13}, 2^{13}, 2^{12}, 2^{12}, \ldots, 2, 2, 1, 1\}$. This DAC is able to construct 2^{14} levels using only 28 elements. However, since the total element weight is $\approx 2^{15}$, the usable output range is only half of the theoretically available range. Other modifications, such as the use of dithering (especially in the primary modulator), or the use of higher-order modulation, can be applied provided the range of the secondary DAC(s) is increased appropriately.

Table 6.1   Output waveforms of a 1-bit DAC and its model.

$v[n-1]$	$v[n]$	DAC output	Model output
-1	-1	w_{LL}	$w_0 - w_1 - w_2 + w_e$
-1	$+1$	w_{LH}	$w_0 + w_1 - w_2 - w_e$
$+1$	$+1$	w_{HH}	$w_0 + w_1 + w_2 + w_e$
$+1$	-1	w_{HL}	$w_0 - w_1 + w_2 - w_e$

6.7   Transition-Error Shaping

Thus far we have been concerned with DAC errors that afflict both discrete-time (DT) and continuous-time (CT) DACs. In this section we consider errors resulting from nonlinear transition dynamics. Such errors can be of critical importance in CT DACs but are irrelevant in DT DACs whose outputs settle fully each clock period. We start by developing a model of *transition error* and then proceed to show how this error can be shaped using extensions of the techniques described earlier.
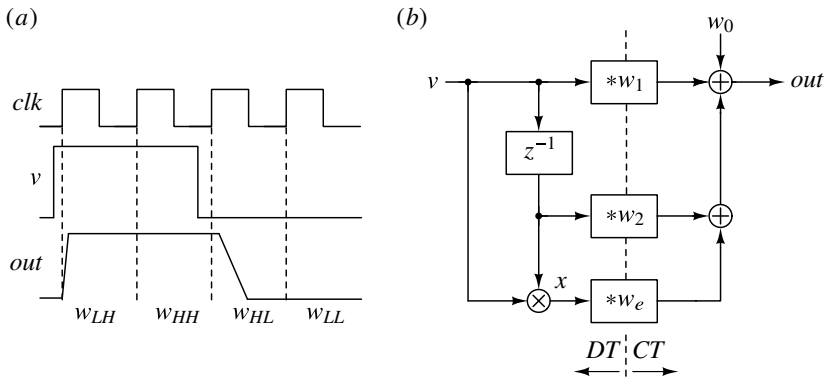
Figure 6.32   Waveforms in a 1-bit CT DAC and associated signal-processing model. (Note that $*$ denotes convolution in the model.)

Figure 6.32(a) defines $w_{LH}(t)$, $w_{HH}(t)$, $w_{HL}(t)$, and $w_{LL}(t)$ as the output *waveforms* of a 1-bit DAC in response to low-high, high-high, etc., input data. These waveforms are functions of time that span one clock period. If the internal nodes of the DAC settle by the end of each clock period, then the output of the DAC can be constructed by stitching together the $w_{LH}(t)$, $w_{HH}(t)$, etc., waveforms and thus these waveforms provide a complete, albeit potentially nonlinear, description of the DAC. As the first step in quantifying the nonlinearity, Figure 6.32(b) shows a model of a 1-bit CT DAC whose four waveform parameters ($w_0(t)$, $w_1(t)$, $w_2(t)$ and $w_e(t)$) will be selected such that the model produces the correct output waveform in response to the input data.[5]

Table 6.1 tabulates the DAC and model outputs in response to the four possible one-bit input pairs. In keeping with the conventions used in the earlier chapters, the input alphabet

---

[5]The model of Figure 6.32(b) can be derived by performing a least-squares fit of the linear model $w_{lin} = w_0 + w_1 v[n] + w_2 v[n-1]$ to the w_{LH}, w_{HH}, w_{HL} and w_{LL} responses and then computing the residual $w_{actual} - w_{lin}$. We spare the reader these details by starting with the model and matching it to the w_{XX} waveforms.

for the v input is $\{-1, +1\}$. In order for the model output to match the actual DAC output, we require

$$
\begin{bmatrix}
+1 & -1 & -1 & +1 \\
+1 & +1 & -1 & -1 \\
+1 & +1 & +1 & +1 \\
+1 & -1 & +1 & -1
\end{bmatrix}
\begin{bmatrix}
w_0 \\
w_1 \\
w_2 \\
w_e
\end{bmatrix}
=
\begin{bmatrix}
w_{LL} \\
w_{LH} \\
w_{HH} \\
w_{HL}
\end{bmatrix},
\tag{6.25}
$$

which can be inverted to yield the model waveforms in terms of the DAC waveforms

$$
\begin{bmatrix}
w_0 \\
w_1 \\
w_2 \\
w_e
\end{bmatrix}
= \frac{1}{4}
\begin{bmatrix}
+1 & +1 & +1 & +1 \\
-1 & +1 & +1 & -1 \\
-1 & -1 & +1 & +1 \\
+1 & -1 & +1 & -1
\end{bmatrix}
\begin{bmatrix}
w_{LL} \\
w_{LH} \\
w_{HH} \\
w_{HL}
\end{bmatrix}.
\tag{6.26}
$$

Having computed the model parameters in terms of the DAC's output waveforms, we are now in a position to discuss the nature of a CT DAC's nonlinearity. First note that the DAC model is linear in the mathematical sense except for two terms, specifically w_0 and w_e. The w_0 term, which is analogous to a dc offset in the DT case, creates spurs at dc and at multiples of the clock frequency. Such offset and clock feed-through terms can usually be ignored since they do not corrupt the information-bearing components of the signal. The w_e term is the more problematic nonlinear term. In order for the DAC to be linear, we require $w_e = 0$, which, from the last row of (6.26), is equivalent to the requirement

$$
w_{LL} + w_{HH} = w_{LH} + w_{HL}.
\tag{6.27}
$$

One interpretation of this result is that a linear CT DAC must obey the principle of super-position. Another useful insight from (6.27) is that a balanced differential DAC is auto-matically linear because in such a DAC $w_{LL} = -w_{HH}$ and $w_{LH} = -w_{HL}$. Important sources of imbalance in a differential current-mode DAC include V_T mismatch in the switches and mismatched delay in the switch control signals. The job of the DAC designer is therefore to design DAC circuits that make these error sources sufficiently small.

The spectrum of the nonlinear component of the DAC's output is obtained by con-volving $w_e(t)$ with the discrete-time sequence

$$
x[n] = v[n] \cdot v[n-1].
\tag{6.28}
$$

Since $x[n]$ is $+1$ if the DAC does not switch and -1 if the DAC does switch, the x sequence reflects the occurrence of switching events, or transitions. We therefore refer to the error obtained by convolving x with w_e as the *transition error*. Since convolution in the time domain is multiplication in the frequency domain, the spectrum of the transition error is equal to the spectrum of the x sequence multiplied by the Fourier transform of w_e. Transition error can therefore be shaped by shaping the x sequence.

The arrangement shown in Figure 6.33 can be used to shape the x sequences of a multi-element DAC. As with vector-based mismatch-shaping, all the labeled signals ex-cept v are M-element vectors, and the vector quantizer obeys $\sum_i sv_i = v$. This system can be divided into a linear loop filter and a nonlinear "quantizer" as indicated by the dashed boxes. With this partitioning, we can recognize the system as M implementations of MOD1 with inputs xu and outputs xv, but containing a peculiar quantizer. If the loop is stable, $\Delta\Sigma$ theory tells us that with a constant input xu, the output xv contains a dc term equal to xu plus first-order shaped noise. Since xv is a vector of the transition (x)
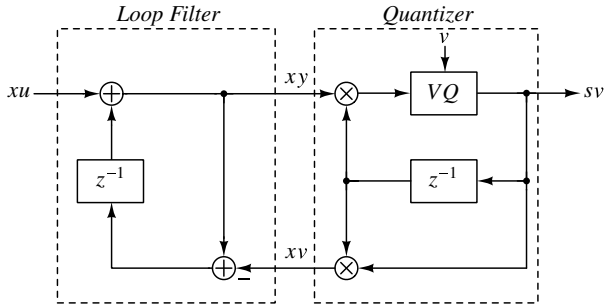
Figure 6.33   Element selection logic for first-order transition-error shaping.

sequences, we conclude that transition error is first-order shaped. To verify the plausibil-ity of stable operation, note that if xy_i is high (indicating that there have been too many transitions for element i), then the associated comparator is biased towards making the associated bit stay the same.

As with vector-based mismatch-shaping, the stability of this transition-error-shaping loop depends on the shaping function and on the input data v. However, in contrast with vector-based mismatch-shaping, the xu input also affects the stability of the loop, and, unlike the time-varying scalar su signal in vector-based mismatch-shaping, xu is a vector and must be constant. The choice of an appropriate value for xu is an open problem. Using $xu = 0$ sets the target transition rate to 50%, which is not supportable when v is near the ends of its $[-M, M]$ range since only a few elements can switch. On the one hand, the problem is worse when xu is negative because such a setting increases the target switching rate. On the other hand, making xu positive decreases the target switching rate but makes it difficult for the system to track a fast-changing signal.
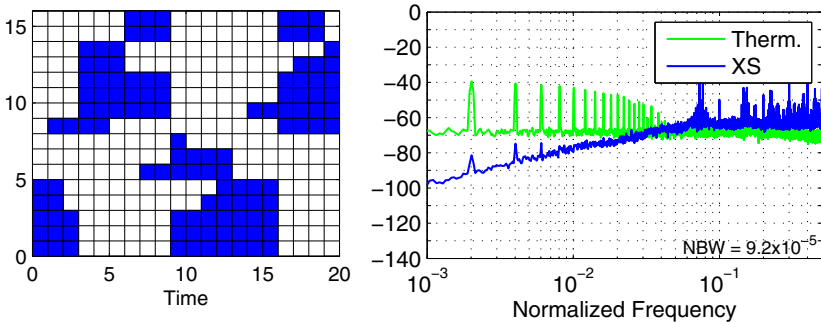
Figure 6.34   Example usage pattern for first-order transition-error shaping (XS) and comparison of transition vector spectra. (−3-dBFS input, $xu = 0.7$.)

Figure 6.34 depicts simulation results for $xu = 0.7$ when the input is a −3-dBFS low-frequency ($f = 0.002$) sine wave. The element usage plot shows that the large positive value of xu results in a blotchy usage pattern, which is expected since the average number of elements that switch per cycle is a relatively small value, namely $M(1 - xu)/2 = 2.4$. The spectrum indicates that transition error is shaped. The most encouraging aspect of this simulation is that the second-harmonic is attenuated by more than 30 dB compared to

plain thermometer coding. Since further simulations indicate that the system tolerates -3-dBFS inputs up to $f \approx 0.1$, our choice of $xu = 0.7$ appears to be reasonable. On the less encouraging side, the frequency at which the shaped transition error crosses the unshaped error is $f \approx 0.03$, so transition-error shaping offers little benefit for $OSR < 16$; also the large spur energy in the vicinity of $f \approx 0.07$ is worrisome.
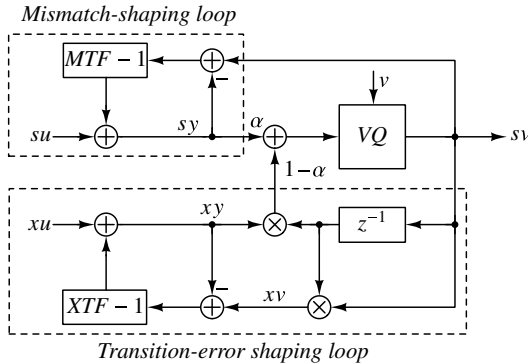
Figure 6.35   Combined mismatch-shaping and transition-error shaping.

As a final example of a mismatch-shaping system at the forefront of technology, Figure 6.35 depicts a system that combines both mismatch-shaping and transition-error shaping [17]. In this system, the output of the mismatch-shaping loop and the output of the transition-error-shaping loop are added with weights α and $(1 - \alpha)$, respectively. The designer selects α according to the relative magnitude of the two error sources.
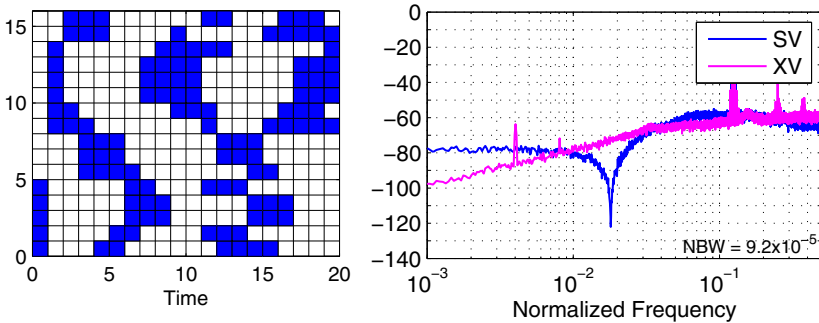
Figure 6.36   Simulated performance of combined mismatch- and transition-error shaping. ($\alpha = 0.5$; $xu = 0.5$; -3-dBFS input at $f = 0.002$.)

To demonstrate that this arrangement is workable, Figure 6.36 plots the usage pattern along with spectra of the sv and xv signals for a system combining first-order transition-error shaping with second-order mismatch-shaping. Since xu is positive, the usage pattern again has a blotchy appearance, but now both the spectra of the selection vector and the transition vector are shaped.

6.8 Conclusions

In this chapter we examined a variety of techniques for enhancing the linearity of a multi-bit DAC consisting of M one-bit DAC elements. An important feature of these mismatch-shaping schemes is that they operate without knowledge of the actual element errors. We found that choosing the elements randomly turns the errors caused by static element mismatch into white noise, whereas selecting the elements in a rotational fashion accomplishes first-order shaping. Using the Bi-DWA or the dithered tree-structure variants of first-order mismatch-shaping was found to reduce or eliminate tones, respectively, at the expense of mismatch suppression. We showed that high-order shaping is possible with more complex hardware containing multiple modified $\Delta\Sigma$ loops.

In addition to static element mismatch, we also discussed the dynamics of a one-bit CT DAC, and saw that transition error can also be shaped. Simultaneous shaping of transition error and mismatch error can be accomplished by combining a mismatch-shaping loop with a transition-error-shaping loop. Transition-error shaping is in its infancy and the ambitious reader is encouraged to develop it further. For a recent review of such techniques, and for a more extensive set of references, consult [18].

The $\Delta\Sigma$ Toolbox contains several functions related to element selection: `ds_therm` (thermometer-coded selection), `simulateMS` (vector-based mismatch-shaping, including rotation), `simulateSwap` (butterfly shuffler), `simulateTSMS` (tree-structured shaper), `simulateBiDWA` (bi-directional data-weighted-averaging), `simulateXS` (transition-error shaping), and `simulateMXS` (combined mismatch- and transition-error shaping).

References

[1] M. Sarhang-Nejad and G. C. Temes, "A high-resolution multibit $\Sigma\Delta$ ADC with digital correction and relaxed amplifier requirements," *IEEE Journal of Solid-State Circuits*, vol. 28, pp. 648–660, June 1993.

[2] R. T. Baird and T. S. Fiez, "Improved $\Delta\Sigma$ DAC linearity using data weighted averaging," *Proceedings of the 1995 IEEE International Symposium on Circuits and Systems*, vol. 1, pp. 13–16, May 1995.

[3] B. H. Leung and S. Sutarja, "Multi-bit Σ-Δ A/D converter incorporating a novel class of dynamic element matching," *IEEE Transactions on Circuits and Systems II*, vol. 39, pp. 35–51, Jan. 1992.

[4] H. S. Jackson, "Circuit and method for cancelling nonlinearity error associated with component value mismatches in a data converter," U.S. patent number 5221926, June 22, 1993 (filed July 1, 1992).

[5] M. J. Story, "Digital to analogue converter adapted to select input sources based on a preselected algorithm once per cycle of a sampling signal," U.S. patent number 5138317, Aug. 11, 1992 (filed Feb. 10, 1989).

[6] W. Redman-White and D. J. L. Bourner, "Improved dynamic linearity in multi-level $\Sigma\Delta$ converters by spectral dispersion of D/A distortion products," *IEE Conference Publication European Conference on Circuit Theory and Design*, pp. 205–208, Sept. 5-8, 1989.

[7] Yang, W. Schofield, H. Shibata, S. Korrapati, A. Shaikh, N. Abaskharoun, and D. Ribner, "A 100mW 10MHz-BW CT $\Delta\Sigma$ modulator with 87dB DR and 91dBc IMD," *Proceedings of the 2008 IEEE International Solid-State Circuits Conference*, pp. 498–499, Feb. 2008.

[8] R. W. Adams and T. W. Kwan, "Data-directed scrambler for multi-bit noise-shaping D/A converters," U.S. patent number 5404142, April 4, 1995 (filed Aug. 1993).

[9] D-H. Lee and T-H. Kuo, "Advancing data weighted averaging technique for multi-bit sigma-delta modulators," *IEEE Transactions on Circuits and Systems II*, vol. 54, no. 10, pp. 838–842, Oct. 2007.

[10] I. Fujimori, L. Longo, A. Hairapetian, K. Seiyama, S. Kosic, J. Cao, and S. L. Chan, "A 90-dB SNR 2.5-MHz output-rate ADC using cascaded multibit delta-sigma modulation at 8× oversampling ratio," *IEEE Journal of Solid-State Circuits*, vol. 35, no.12, pp. 1820–1828, Dec. 2000.

[11] J. Welz and I. Galton, "Simplified logic for first-order and second-order mismatch-shaping digital-to-analog converters," *IEEE Transactions on Circuits and Systems II*, vol. 48, no. 11, pp. 1014–1027, Nov. 2001.

[12] J. Welz and I. Galton, "A tight signal-band power bound on mismatch noise in a mismatch-shaping digital-to-analog converter," *IEEE Transactions on Information Theory*, vol. 50, no. 4, pp. 593–607, Apr. 2004.

[13] R. Schreier and B. Zhang, "Noise-shaped multibit D/A convertor employing unit elements," *Electronics Letters*, vol. 31, no. 20, pp. 1712–1713, Sept. 28, 1995.

[14] A. Yasuda, H. Tanimoto, and T. Iida, "A third-order $\Delta - \Sigma$ modulator using second-order noise-shaping dynamic element matching," *IEEE Journal of Solid-State Circuits*, vol. 33, pp. 1879–1886, Dec. 1998.

[15] K.Q. Nguyen and R. Schreier, "System and method for tri-level logic data shuffling for oversampling data conversion," U.S. patent number 07079063, July 18, 2006 (filed Apr. 18 2005).

[16] K. L. Chan and I. Galton, "A 14b 100 MS/s DAC with fully segmented dynamic element matching," in *Proceedings of the IEEE International Solid-State Circuits Conference*, pp. 2390–2399, Feb. 2006.

[17] L. Risbo, R. Hezar, B. Kelleci, H. Kiper, and M. Fares, "Digital approaches to ISI-mitigation in high-resolution oversampled multi-level D/A converters," *IEEE Journal of Solid-State Circuits*, vol. 46, no.12, pp. 2892–2903, Dec. 2011.

[18] A. Sanyal and N. Sun, "Dynamic element matching techniques for static and dynamic errors in continuous-time multi-bit $\Delta\Sigma$ modulators," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 5, no. 4, pp. 598–611, Dec. 2015.

CHAPTER 7

CIRCUIT DESIGN FOR DISCRETE-TIME DELTA-SIGMA ADCS

This chapter examines circuits for switched-capacitor $\Delta\Sigma$ ADCs. A simple low-speed 1-bit second-order modulator is used to illustrate the primary design considerations. More advanced circuits, techniques, and analyses follow.

7.1 SCMOD2: A Second-Order Switched-Capacitor ADC

To implement MOD2, whose standard block diagram is given below in Figure 7.1, we need circuits that do integration, summation, 1-bit quantization and 1-bit feedback. Before we learn how to make such circuits, let us pick some target ADC specifications.
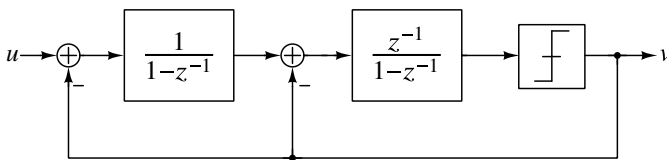
Figure 7.1   Standard MOD2 block diagram.

Table 7.1 lists our proposed targets. An ADC having a bandwidth of 1 kHz could be used as an on-chip voltage monitor or as part of a low-speed calibration engine. Since

choosing a clock rate of 1 MHz yields an oversampling ratio of 500, the SQNR available from a second-order modulator is about 120 dB, and thus a 100-dB SNR target is quite feasible. To ease the circuit design, we will use a 1.8-V supply. Although the core supply voltage of current nanometer CMOS processes is 1.0 V or lower, most such technologies support 1.8-V devices operating from the IO supply.[1] For this reason, the circuits of this design example are practical even in modern CMOS processes.

Table 7.1 Specifications for SCMOD2.

Parameter	Symbol	Value	Units
Bandwidth	f_B	1	kHz
Sampling frequency	f_s	1	MHz
Signal-to-Noise Ratio	SNR	100	dB
Supply voltage	V_{dd}	1.8	V

7.2 High-Level Design

7.2.1 NTF Selection
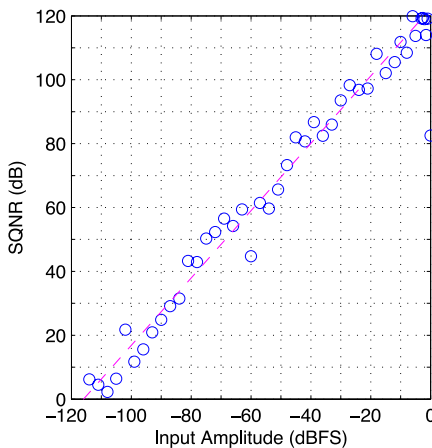
Figure 7.2 Simulated SQNR versus input amplitude.

The standard version of MOD2 having $NTF\,(z) = (1-z^{-1})^2$ is in common use, but we recommend using a less aggressive NTF so that the modulator is more well-behaved when the input is close to full-scale. The code fragment below creates an NTF and evaluates its performance using an SQNR versus amplitude curve (Figure 7.2). Since the ideal peak SQNR is 120 dB, quantization noise is 20 dB below our target noise level. This margin is at the generous end of the 10–20 dB range that typically separates the ideal SQNR from the target SNR.

---

[1]It is common for a CMOS IC to have a secondary supply, the IO (input/output) supply, that is used to interface to other ICs. Analog designers often take advantage of the IO supply.

```
% Create a second-order NTF
order = 2;
osr = 500;
M = 1;
ntf = synthesizeNTF(order,osr);
% Plot the SQNR vs. amplitude curve
[sqnr, amp] = simulateSNR(ntf,osr,[],[],M+1);
plot(amp,sqnr,'-o','Linewidth',1);
...
```

7.2.2   Realization and Dynamic-Range Scaling

When converting a block diagram into a circuit, we need to ensure that the signal magnitude at each node is within the range of the amplifier driving that node. Unfortunately, the block diagram in Figure 7.1 provides no information regarding the signal swing at the integrator outputs. Even if we had this information, we have done nothing to ensure that these swings are compatible with our circuits. To remedy this omission, we need to determine the swing of each integrator output, and then scale each stage such that its output is within the anticipated range of our opamps.
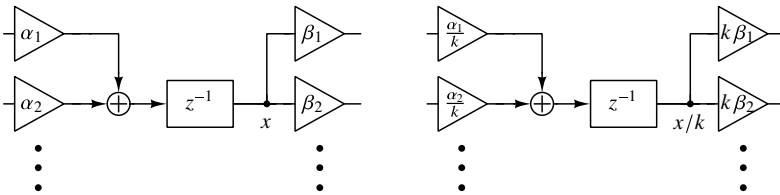
Figure 7.3   State scaling.

As described in Section 4.7.1 and illustrated in Figure 7.3, to scale a particular state (x) down by a factor of k, simply divide all incoming coefficients by k and multiply all outgoing coefficients by k.

The code fragment below realizes the aforementioned NTF with a CIFB topology and performs dynamic range scaling as described above using the functions of the $\Delta\Sigma$ toolbox (Appendix B). Figure 7.4 shows the associated block diagram. In this diagram, all co-
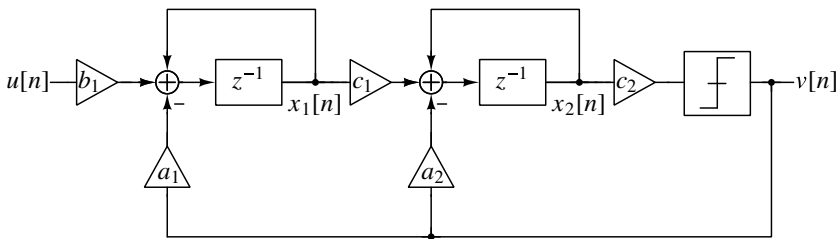
Figure 7.4   Second-order CIFB modulator.

efficients except c_2 will translate into capacitor ratios. The c_2 coefficient turns out to be immaterial, since a 1-bit quantizer only cares about the sign of its input, and c_2 is positive.

```
...
form = 'CIFB';
swing = 0.5;     %Amplifier output swing, Vp
umax = 0.9*M;    %Scale system for inputs up to 0.9 of full-scale
[a,g,b,c] = realizeNTF(ntf,form);
b(2:end) = 0;
ABCD = stuffABCD(a,g,b,c,form);
ABCD = scaleABCD(ABCD,M+1,[],swing,[],umax);
[a,g,b,c] = mapABCD(ABCD,form);
% Yields a = [0.1131 0.1829]; b = 0.1131; c=[0.4517 4.2369]
```

7.3 Switched-Capacitor Integrator
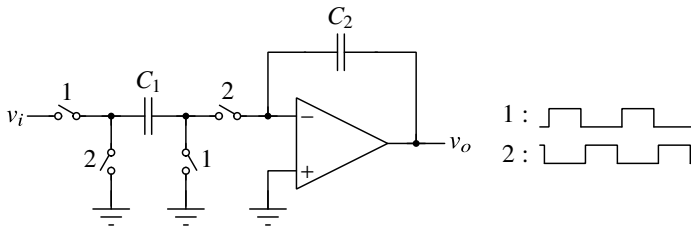
Figure 7.5   Switched-capacitor integrator.

Figure 7.5 depicts our first building block: a switched-capacitor (SC) integrator. In this circuit, each clock period is divided into two phases and the circuit toggles between the two configurations defined by the phase switches. During phase 1, the switches labeled "1" are on and the switches labeled "2" are off. During phase 2, the reverse is true. Furthermore, no phase-1 switch is on at the same time as a phase-2 switch.
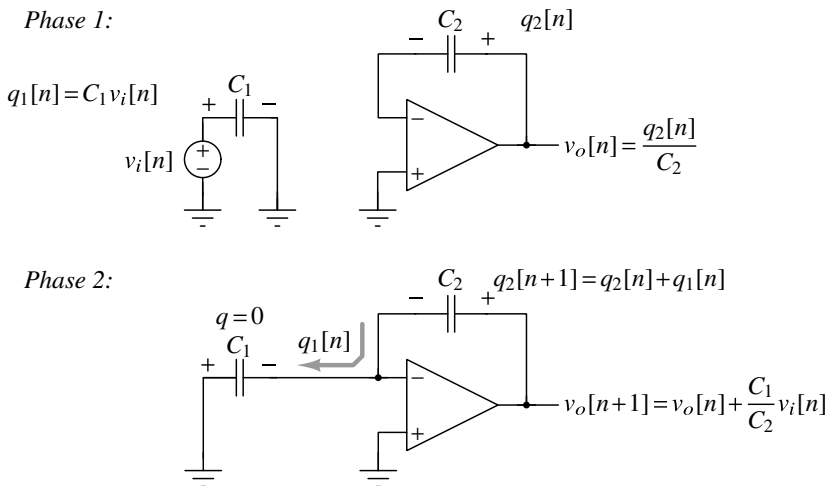
Figure 7.6   Phase 1 and phase 2 configurations of the integrator.

To understand how this circuit works, consider the two configurations depicted in Figure 7.6. During phase 1, C_1 is charged to the input voltage $v_i[n]$ and C_2 holds whatever charge it had during the preceding phase 2. During phase 2, C_1 is connected to ground on the left side and to virtual ground on the right. If the gain of the opamp is infinite, then the charge on C_1 is driven to zero. By virtue of the circuit's topology, however, the charge from C_1 must accumulate on C_2, and thus

$$q_2[n + 1] = q_2[n] + q_1[n]. \tag{7.1}$$

Taking the z-transform of (7.1) yields

$$\frac{Q_2(z)}{Q_1(z)} = \frac{z^{-1}}{1 - z^{-1}}. \tag{7.2}$$

Since $Q_1(z) = C_1 V_i(z)$ and $Q_2(z) = C_2 V_o(z)$, we conclude that the circuit in Figure 7.5 implements a delaying integrator with a scale factor C_1/C_2:

$$\frac{V_o(z)}{V_i(z)} = \frac{C_1}{C_2} \frac{z^{-1}}{1 - z^{-1}}. \tag{7.3}$$

The circuit in Figure 7.5 is often described as being *parasitic-insensitive*, since parasitic capacitances from the switch nodes to ground do not affect the transfer function. To understand this property, first consider a parasitic capacitance from the left side of C_1 to ground. This parasitic capacitor is charged to v_i on phase 1 and then discharged to ground on phase 2. Although this sounds similar to what happens with C_1, the discharge path is only through the phase-2 switch, and therefore none of the charge on this parasitic capacitor is transferred to C_2. Thus, parasitic capacitance on the left side of C_1 does not alter the integrator's transfer function.

Next consider a parasitic capacitor from the right side of C_1 to ground. The top of this capacitor is alternately connected to ground and then to virtual ground. As a result, this parasitic capacitor never holds any charge and so plays no role in the integrator's transfer function. By following similar reasoning with other nodes in the circuit, the reader can verify that parasitic capacitance from any node to ground does not alter (7.3).

Let's now advance to the circuit shown in Figure 7.7. This circuit consists of two switched-capacitor branches connected to the same amplifier. By applying the principle of superposition to this circuit, we quickly arrive at the relation

$$V_o(z) = \left(\frac{C_1}{C} V_1(z) + \frac{C_2}{C} V_2(z)\right) \frac{z^{-1}}{1 - z^{-1}}. \tag{7.4}$$

The circuit in Figure 7.7 therefore implements both summation and integration. In fact, by connecting v_2 to $+V_{ref}$ when $v[n]$ is -1 and to $-V_{ref}$ when $v[n]$ is $+1$, the second branch implements a one-bit feedback DAC with the polarity called for in Figure 7.1. The circuit in Figure 7.7 thus implements all the functions needed in MOD2, except for quantization!

Figure 7.8 shows a candidate switched-capacitor topology and its associated timing diagram alongside the difference equations for Figure 7.4. (The dashed lines connecting v to switches represent 1-bit DACs, i.e., a pair of switches connected to $\pm V_{ref}$ and controlled by v.) The first difference equation indicates that $v[n]$ depends on $x_2[n]$, which is consistent
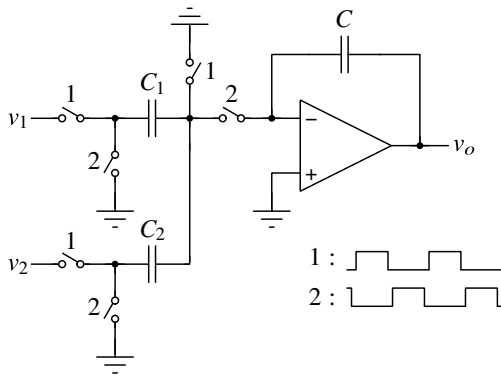
$$V_o(z) = \left(\frac{C_1}{C} V_1(z) + \frac{C_2}{C} V_2(z) \right) \frac{z^{-1}}{1-z^{-1}}$$

Figure 7.7   SC integrator with two inputs.
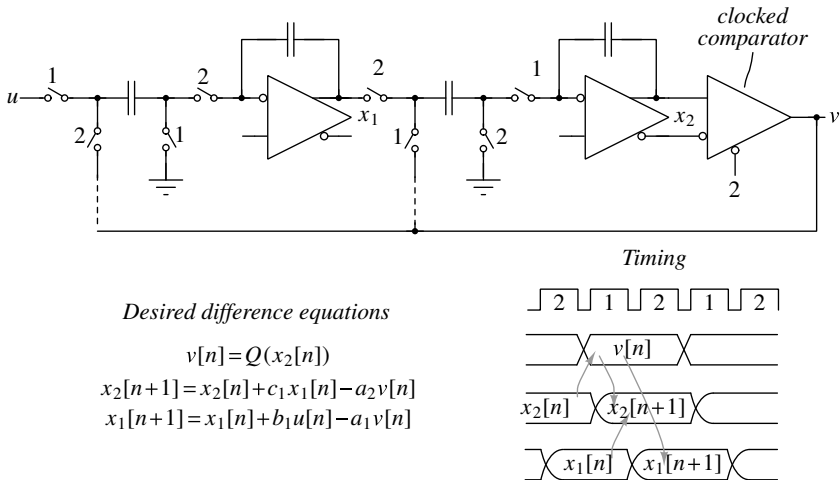
Desired difference equations

$$v[n] = Q(x_2[n])$$
$$x_2[n+1] = x_2[n] + c_1 x_1[n] - a_2 v[n]$$
$$x_1[n+1] = x_1[n] + b_1 u[n] - a_1 v[n]$$

Figure 7.8   Candidate topology for MOD2 and associated timing check.

with both the schematic and the timing diagram since $x_2[n]$ is available at the end of phase 2 and $v[n]$ is produced by the comparator on the falling edge of the phase-2 clock. The next equation shows that $x_2[n + 1]$ depends on both $x_1[n]$ and $v[n]$, which is again consistent with the schematic and timing diagram since, as illustrated, $v[n]$ is available on the next phase 1 and $x_1[n]$ was sampled on the preceding phase 2. Finally, we see that $x_1[n + 1]$ depends on $v[n]$, which is again consistent with the circuit and the timing diagram, since $x_1[n + 1]$ is produced on phase 2 when $v[n]$ is present. (The dependence of $x_1[n + 1]$ on $u[n]$ is not important since u is only connected to the first integrator and so we are free to assign arbitrary timing labels to the u waveform.) [2]

7.3.1   Integrator Variations
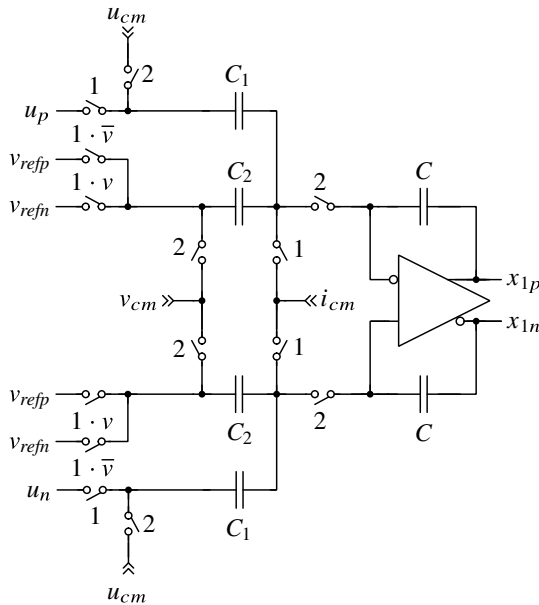
Figure 7.9   Differential integrator with separate input and DAC capacitors.

Figure 7.9 shows a differential version of the combined integrator, summer, and feedback DAC. This diagram shows three different common-mode nodes driven by voltages i_{cm}, u_{cm}, and v_{cm}. These voltages represent degrees of freedom for the designer. For

---

[2]This methodology deviates from standard practice in the design of switched-capacitor filters. In the standard methodology, the designer chooses the end of either phase 1 or phase 2 as the dividing line between time n and time $n + 1$, and then handles the circuit using z-domain descriptions of each block. The circuit of Figure 7.8 presents two challenges to the standard methodology. First, since there is no time when $v[n]$ and $x_2[n]$ are both present, we cannot make a consistent division between time n and time $n + 1$. Second, the z-domain description of a block is not unique. For example, if we choose the end of phase 2 as the dividing line between time n and time $n + 1$, then the second integrator is described with a delaying transfer function, whereas if we choose the end of phase 1 as the dividing line between time n and time $n + 1$, then the transfer function is non-delaying. In fact, an integrator may be considered delaying for one input and non-delaying for another input. For these reasons, we have chosen to abandon the standard methodology and instead verify the difference equations directly.

example, i_{cm} is the input common-mode voltage of the amplifier, and the designer is free to choose this voltage based on the needs of the amplifier. Likewise, u_{cm} is the common-mode voltage of the input signal u and v_{cm} is the common-mode voltage of the reference voltage. These two voltages have no effect on the operation of the circuit provided

$$u_{cm} = (u_p + u_n)/2 \tag{7.5}$$

and

$$v_{cm} = (v_{refp} + v_{refn})/2. \tag{7.6}$$

Mismatch between the voltage supplied at u_{cm} and the actual common-mode voltage of the input, or between the voltage supplied at v_{cm} and the actual common-mode voltage of the reference, introduces a common-mode error at the input of the amplifier. The designer should ensure that the amplifier has enough input common-mode range to accommodate such errors.

Next, observe that the opamp in a switched-capacitor circuit only drives capacitors. Thus, as the circuit settles in response to a phase change, the opamp's output current decays to zero. This property allows the opamp to have an arbitrarily large output resistance, and thus a simple transconductance with a high output resistance makes a perfectly serviceable high-gain opamp for a switched-capacitor circuit. The acronym applied to this kind of amplifier is OTA, which stands for *operational transconductance amplifier*.
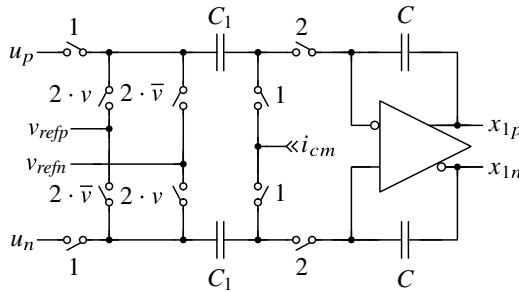
Figure 7.10   Differential integrator with shared input and DAC capacitors.

Figure 7.10 shows a simplified version of the preceding integrator in which the same capacitors are used for sampling the input (on phase 1) and applying feedback (on phase 2). This arrangement has ~3 dB lower noise than the circuit in Figure 7.9 (noise is discussed in Section 7.18) but requires the input full-scale voltage to equal the reference voltage and is also susceptible to mismatch in their common-mode voltages.

Figure 7.11 shows an integrator that also uses the same capacitors for the feedback DAC as are used to sample the input, but accepts a single-ended input. Furthermore, this circuit uses $V_{ref} = V_{dd}$.[3] Since one of the applications envisioned for our ADC is a voltage monitor, a single-ended input with a range of $[V_{ss}, V_{dd}]$ seems like a perfect fit. For this reason, we would like to adopt this topology for the first integrator.

---

[3]In practice, using the supply voltage as the reference requires heavy filtering to achieve high conversion accuracy in the face of supply noise. A suitable circuit utilizing a two-step coarse/fine charging scheme and an external capacitor to effectively filter the supply voltage with a 2-Hz lowpass filter is described in [1].
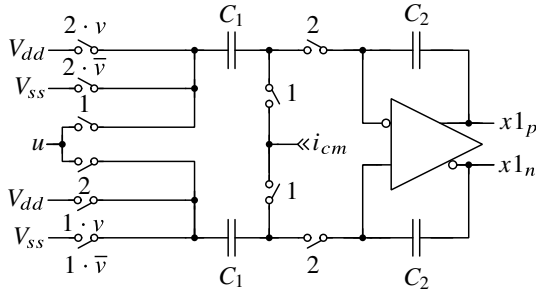
Figure 7.11   Differential integrator with single-ended-to-differential conversion.

However, some care must be taken at this point since the circuit in Figure 7.11 uses slightly different timing than was assumed in Figure 7.8. In particular, Figure 7.8 shows $v[n]$ being fed back to the first integrator during phase 2, whereas in Figure 7.11 $v[n]$ is sampled on both phase 2 and the preceding phase 1. Fortunately, as indicated in the timing diagram of Figure 7.8, $v[n]$ is available at both times, and thus we are able to use the circuit in Figure 7.11 as the first integrator in our system. As an aside, we note that the situation would not have worked out so well if we had chosen the CRFB topology.
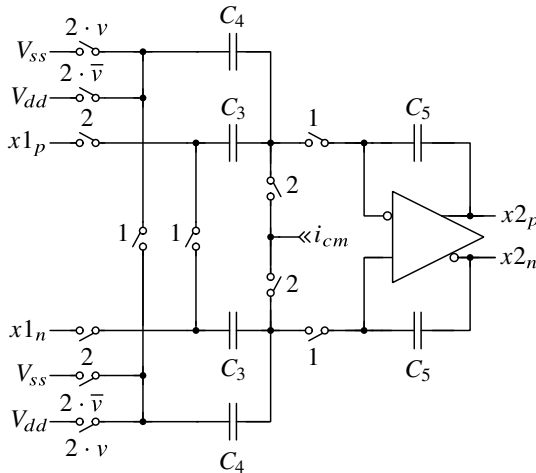
Figure 7.12   Second integrator in SCMOD2.

Figure 7.12 shows a version of Figure 7.9 in which the input common-mode and reference common-mode switches have been replaced with differential shorting switches. This simple connection automatically finds the required common-mode voltage, and since one switch replaces two, the resistance of the switch can be double that of the two original switches. Since this circuit decouples the input and feedback weightings, and since the second integrator in our example modulator uses the unrelated coefficients a_2 and c_1, we will use this circuit for the second integrator.

Figure 7.13 shows an integrator in which the DAC capacitors are neither shorted together nor connected to v_{cm} during phase 2. Instead, the DAC capacitors are connected to
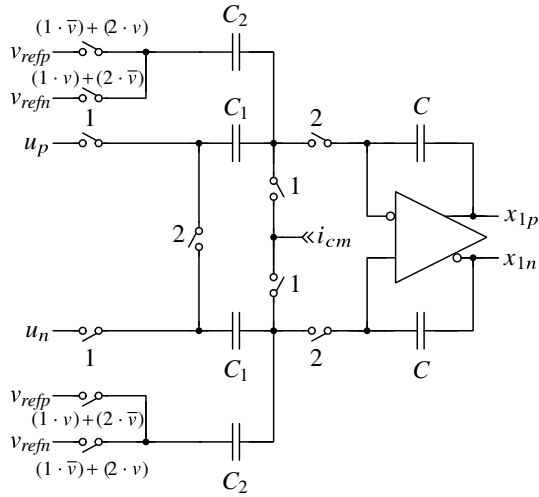
Figure 7.13   Integrator with double-sampled references.

the opposite reference polarity. Doing so effectively doubles the reference voltage and so results in lower noise than the circuit of Figure 7.9.
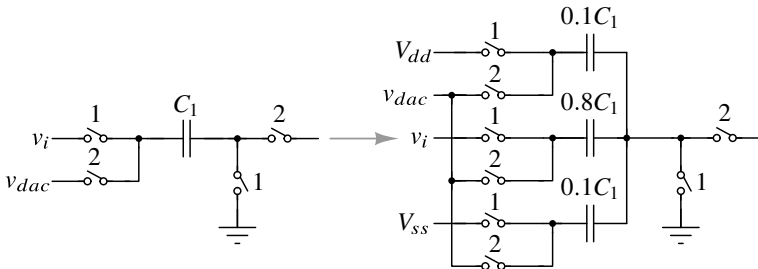
Figure 7.14   Input voltage scaling and shifting.

As a finale for our foray into integrator variations, Figure 7.14 shows an input structure that attenuates the input by 20% in order to prevent the modulator from being overloaded. The price is a 2-dB SNR hit and a loss of dc accuracy since the $V_{\text{fullscale}}/V_{\text{ref}}$ ratio now depends on capacitor matching.

7.4 Capacitor Sizing

The capacitance ratio in the first stage can be computed using either

$$a_1 = \frac{C_1 V_{ref}}{C_2} = \frac{C_1 V_{dd}}{C_2},\tag{7.7}$$

or

$$b_1 = \frac{C_1 V_{FS}}{M C_2} = \frac{C_1 V_{dd}}{C_2}.\tag{7.8}$$

Since $a_1 = b_1$, both calculations yield the same result.[4]

The absolute value of C_1 is determined by a thermal noise constraint. The mean-square noise voltage yielding an SNR of 103 dB (100 dB plus 3 dB margin) relative to the power of a full-scale ($V_p = V_{dd}/2$) sine wave is

$$\overline{v_n^2} = \frac{(V_{dd}/2)^2/2}{10^{SNR/10}} = \frac{(0.9)^2/2}{10^{(103/10)}} = (4.5\,\mu\text{V})^2. \tag{7.9}$$

The in-band input-referred mean-square noise voltage associated with the first integrator is approximately[5]

$$v_n^2 = \frac{kT}{OSR \cdot C_1}. \tag{7.10}$$

Equating (7.9) to (7.10) yields $C_1 = 0.4\,\text{pF}$, and plugging this value into (7.8) gives $C_2 = 6.5\,\text{pF}$. We now have the capacitances needed in the first integrator.

The c_1 coefficient specifies the weighting factor connecting the first integrator to the second:

$$c_1 = \frac{C_3}{C_5}. \tag{7.11}$$

And, similar to (7.7) for the a_1 coefficient, a_2 is related to the feedback capacitor C_4 and the 1-b DAC's differential reference voltage ($\pm V_{dd}$) via

$$a_2 = \frac{C_4 V_{dd}}{C_5}. \tag{7.12}$$

However, since the oversampling ratio is high, the in-band thermal noise of the second integrator is so heavily attenuated by the gain of the first integrator that the second-stage noise constraint is unimportant. Instead, we set the smallest capacitor (C_4) to 10 fF and compute the other capacitors using (7.11) and (7.12). Admittedly, 10 fF is an arbitrary value. If the process supports smaller values with adequate accuracy, then choosing a smaller capacitance would save some power, but as we will see, the savings are small. The calculations for the capacitors are summarized in the code fragment below.

```
% Compute capacitor sizes
Vdd = 1.8;
Vref = Vdd;
FullScale  = Vdd;
DR = 100 +3;    % Dynamic range in dB, plus 3-dB margin
k = 1.38e-23; T = 300; kT = k*T;
% First stage values based on kT/C noise
v_n2 = (FullScale/2)^2/2 / undbp(DR); % = kT/(osr*C1)
C1 = kT/(osr*v_n2);
C2 = C_1/b(1)*FullScale/M;
% Second-stage values based on C4 = 10f
```

[4]DAC coefficients, such as a_1, include the DAC reference voltage, since the v signal is unit-less whereas our state-staling endowed the x_1 signal with dimensions of volts. According to the conventions of the $\Delta\Sigma$ toolbox, u is also normalized to match v, and thus the expression relating b_1 (which multiplies u) to capacitor ratios includes the full-scale voltage.

[5]Section 7.18 shows that the noise of a differential integrator is approximately $4kT/C_1$. The OSR factor appears in (7.10) because we are interested in the in-band portion of the noise, and the factor of 4 disappears because the circuit of Figure 7.11 does single-ended to differential conversion.

```
C4 = 10e-15;
C5 = C4 * Vref / a(2);
C3 = C5 * c(1);
% Yields C1=410f, C2=6.49p, C3=44f, C4=10f, C5=98f
```

7.5   Initial Verification
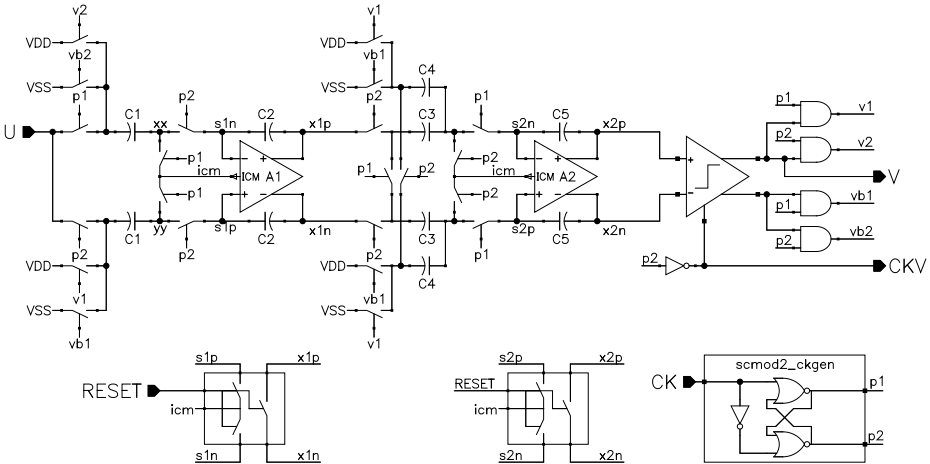
Figure 7.15   MOD2 schematic.

Having manually verified the viability of the chosen topology, we need to simulate our schematic (Figure 7.15) to ensure that it implements the desired difference equations. Since a $\Delta\Sigma$ modulator can be difficult to debug with closed-loop simulations, it is wise to do open-loop simulations before closing the loop and verifying the modulator as a whole.

To verify the loop filter, we recommend checking that its impulse response matches the expected response, which can be computed using the $\Delta\Sigma$ toolbox function `impL1`. To perform this check on our circuit, we need to replace the quantizer with a block that supplies an impulse $v = \{1, 0, 0, \ldots\}$ to the feedback path. Unfortunately, a single-bit DAC can only accept values of $v = \pm 1$. To overcome this limitation, we instead simulate the loop filter twice, first with the DAC input sequence

$$v_1 = \{-1, +1, -1, +1, -1, +1, \ldots\}, \tag{7.13}$$

and then with

$$v_2 = \{+1, +1, -1, +1, -1, +1, \ldots\}. \tag{7.14}$$

Since the loop filter is linear, its response to an impulse ($\delta = (v_2 - v_1)/2$) is given by subtracting the response due to v_1 from the response due to v_2 and then dividing by two. Figure 7.16 compares the predicted responses with the results from this check. Since the simulation results (solid lines) pass through the predicted points (marked by ×), we can be confident that the loop filter and feedback DAC are operating properly.

The first closed-loop check we recommend is a short transient simulation with a dc input. Figure 7.17 shows the simulated input, output and internal signals over 40 clock
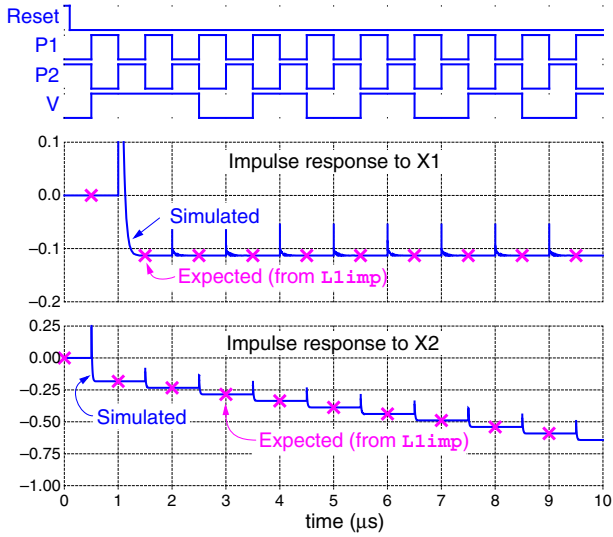
Figure 7.16   Impulse response check.
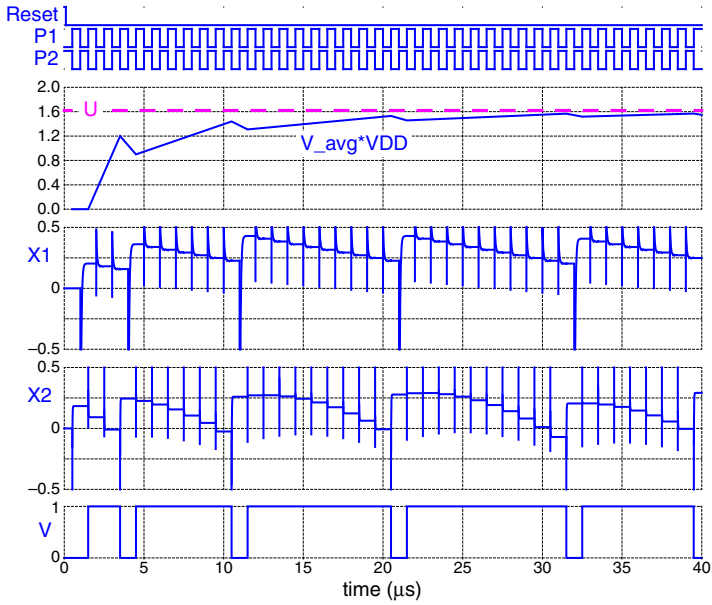
Figure 7.17   Short transient simulation with a dc input.

periods with a dc input at 90% of V_{dd}. Note that the settled integrator outputs are within the prescribed ± 0.5-V limits and that the running average of v, scaled by V_{dd}, approaches the input signal.[6] These observations increase our confidence that the circuit is behaving as intended.
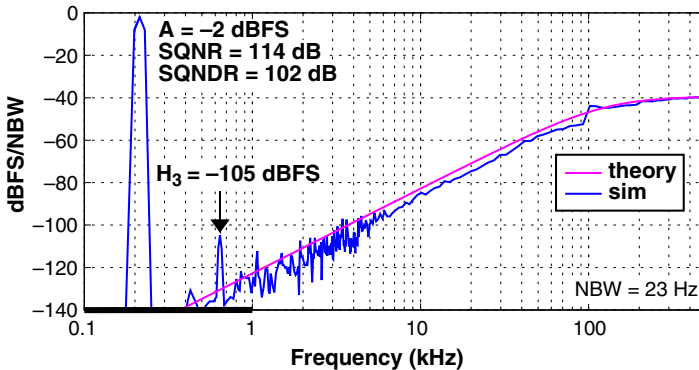
Figure 7.18   Spectrum from a behavioral simulation.

The primary evidence that a modulator really works, however, is a closed-loop simulation with a sine-wave input. Since the OSR is so high, such a simulation needs to be very long in order to accurately determine the modulator's SQNR. A good practice is to do a short simulation to verify that the modulator is stable and demonstrates noise-shaping before running a long simulation, but in Figure 7.18 we present the results of the long (2^{16} clock cycle) simulation only. Noise-shaping with the 40-dB/decade slope characteristic of a second-order null is clearly evident. Also included in Figure 7.33 is the SQNR (114 dB) and the level of the third harmonic ($H_3 = -105$ dBFS) calculated from the simulation data. The SQNR is 6 dB lower than that obtained from the $\Delta\Sigma$ toolbox, but is sufficiently high to confirm that our simulation tolerances are adequate for observing an SQNR well above 100 dB. Similarly, the value of H_3 indicates that distortion terms larger than -105 dBFS are unlikely to be due to the simulation setup. Let's now move on to some transistor-level circuits.

7.6   Amplifier Design

Figure 7.19 depicts *folded-cascode* amplifiers having either NMOS or PMOS input pairs. The folding connection between the output of the differential pair and the source of the opposite-polarity cascodes provides a wide input common-mode range, while the cascoded current sources in the output legs provide a high output impedance and hence high opamp gain. Next we examine some design considerations for this circuit.

To support a differential output of 0.5 V peak, both o_p and o_n must be able to swing over a 0.5-V range. Since the supply voltage is 1.8 V, this leaves 0.65 V for each of the NMOS and PMOS cascoded current sources. To minimize the noise contributed by the

---

[6]Since the input is unipolar, in this plot we interpret the two values of v as $\{0, 1\}$ rather than $\{-1, +1\}$.
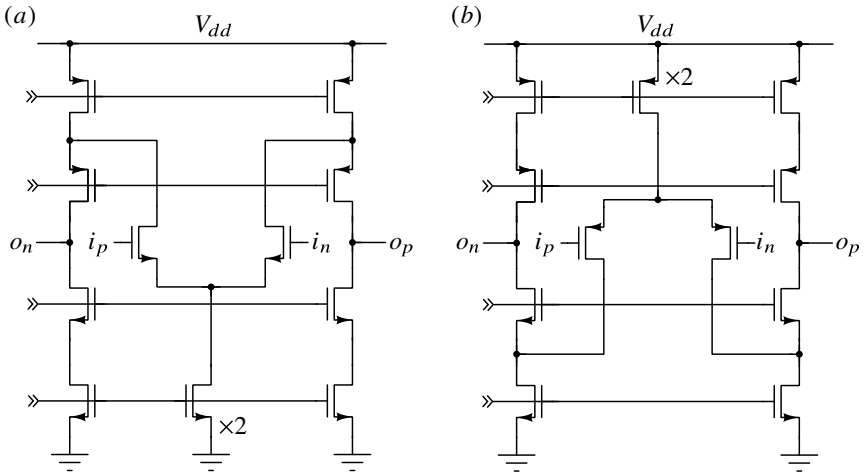
Figure 7.19   Folded-cascode opamps: (a) NMOS input; (b) PMOS input.

current sources, most of this voltage (400 mV) will be allocated to the current sources, leaving 250 mV for the cascode devices.
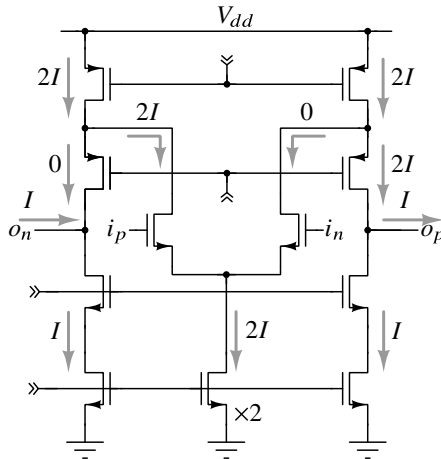
Figure 7.20   Slew currents.

Our next consideration is slewing. Typically, at the start of each charge-transfer phase, the amplifier input terminals are driven far enough apart that the current in the differential pair switches fully to one side. As shown in Figure 7.20, the magnitude of the output current under these conditions is I, where I is the bias current in each half of the differential pair. (I is also assumed to be the standing current in the output cascodes.) Clearly, I must be large enough to transfer the charge from the input capacitor(s) to the integrating capacitor in the allotted time. Let's allocate half of a clock phase (i.e., one quarter of a clock period) for slewing. Since the voltage on the left side of the input capacitor C_1 can

change by as much as $V_{dd} = 1.8$ V, we therefore need

$$I > \frac{C_1 V_{dd}}{T/4} = \frac{0.4\,\mathrm{pF} \cdot 1.8\,\mathrm{V}}{0.25\,\mu\mathrm{s}} = 3\,\mu\mathrm{A}. \tag{7.15}$$

$$\beta = \frac{C_2}{C_1 + C_2}$$

$$C = C_3 + \frac{C_1 C_2}{C_1 + C_2}$$

$$R = \frac{1}{\beta g_m}$$
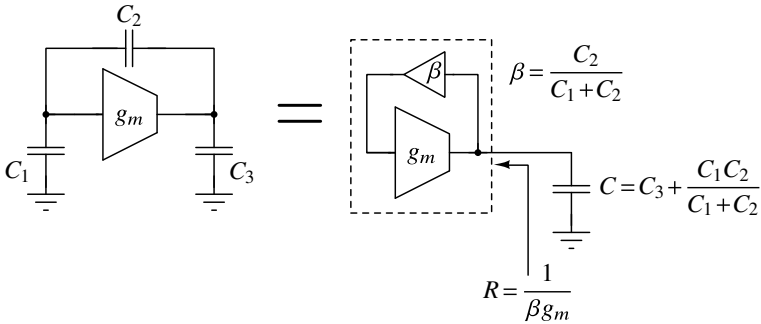
Figure 7.21   Time-constant calculation.

Allocating half of a clock phase for slewing leaves the other half for ordinary linear settling. Figure 7.21 shows the small-signal model of an integrator in the charge-transfer phase and an equivalent circuit from which we immediately see that the time-constant is

$$\tau = RC = \frac{C_1 + C_3 + C_1 C_3 / C_2}{g_m}. \tag{7.16}$$

If we require linear settling to provide, say, 100 dB of attenuation of the initial condition, then

$$T/4 = \tau \ln(10^5) \approx 12\tau, \tag{7.17}$$

which gives

$$g_m = \frac{C_1 + C_3 + C_1 C_3 / C_2}{T/48} = 20\,\mu\mathrm{A/V}. \tag{7.18}$$

Note that the g_m value listed above is that associated with the amplifier half-circuit and thus represents the transconductance of the individual transistors in the differential pair. Since we have already established that the minimum drain current of these transistors is $I_d = 3\,\mu$A, the required g_m/I_d ratio of the input transistors is $20/3 = 7\,\mathrm{V}^{-1}$, which is well within the practical range of a transistor biased in moderate inversion. Since simulations of individual transistors in this process indicate that a g_m/I_d ratio as high as $18\,\mathrm{V}^{-1}$ can be achieved, we could allow more time for slewing at the expense of linear settling and thereby arrive at a more optimal current target. However, since it is still early in the design phase, we will leave some slack in anticipation of some surprises.

7.6.1   Amplifier Gain

Our final amplifier consideration is amplifier gain. Figure 7.22 shows the steps used to analyze the effects of finite amplifier gain on a switched-capacitor integrator. As in the infinite-gain case, we start with phase 1. During phase 1, C_1 charges to the input voltage while C_2 holds its charge $q_2[n]$. However, now the voltage on the left side of C_2 is $-v_o[n]/A$

Phase 1:

$$q_2[n] = C_2(1 + 1/A)v_o[n]$$

$$q_1[n] = C_1 v_i[n]$$

Phase 2:

$$q_2[n+1] = q_2[n] + q$$

$$q_1[n] - q$$

$$v_o[n+1] = \frac{q_2[n+1]}{C_2(1 + 1/A)}$$
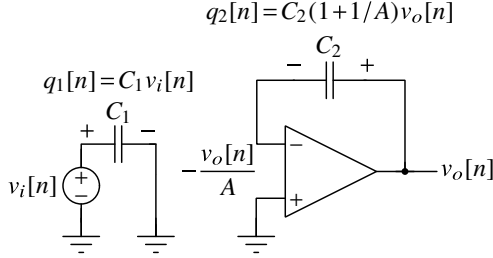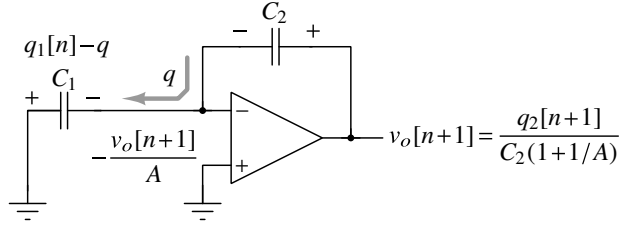
Figure 7.22   Analysis of finite opamp gain.

while the voltage on the right side of C_2 is $v_o[n]$. Thus, the relationship between q_2 and v_o is

$$q_2 = C_2(1 + 1/A)v_o. \tag{7.19}$$

In phase 2 a charge q flows as indicated, reducing the charge on C_1 to

$$q_1[n] - q = C_1 v_o[n+1]/A, \tag{7.20}$$

whence

$$q = q_1[n] - C_1 v_o[n+1]/A. \tag{7.21}$$

This charge is added to C_2, so that

$$
\begin{aligned}
q_2[n+1] &= q_2[n] + q \\
&= q_2[n] + q_1[n] - \frac{C_1 v_o[n+1]}{A} \\
&= q_2[n] + q_1[n] - \frac{C_1 q_2[n+1]}{C_2(A+1)},
\end{aligned} \tag{7.22}
$$

which implies that

$$q_2[n+1] = \frac{q_2[n] + q_1[n]}{1 + \epsilon}, \tag{7.23}$$

where

$$\epsilon = \frac{C_1}{C_2(A+1)}. \tag{7.24}$$

Applying (7.19) to (7.23) and taking the z-transform yields

$$\frac{V_o(z)}{V_i(z)} = \frac{C_1/C_2}{\left(1 + \frac{1}{A}\right)(1 + \epsilon)\left(z - \frac{1}{1+\epsilon}\right)}. \tag{7.25}$$
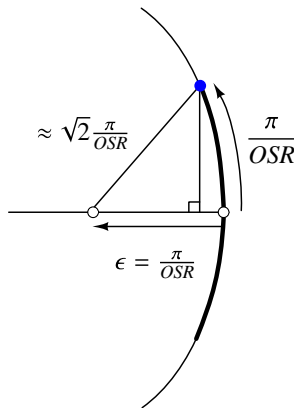
Figure 7.23   NTF zero shift.

From (7.25) we can see that finite amplifier gain has two effects: a small reduction in the integrator's gain constant and an inward shift of the integrator's pole ($z_p \approx 1 - \epsilon$). Since the change in the integrator's gain constant is equivalent to a coefficient error, this change typically has a negligible impact on the in-band attenuation provided by the NTF. In contrast, the pole shift is more problematic because the integrator pole becomes an NTF zero. As illustrated in Figure 7.23, a shift in an NTF zero reduces the NTF attenuation at the passband edge by ~3 dB when the shift is ~ π/OSR. According to this argument, we want the opamp gain to satisfy

$$A > \left(\frac{OSR}{\pi}\right)\left(\frac{C_1}{C_2}\right) - 1 = 19\,\text{dB}. \tag{7.26}$$

This gain requirement is remarkably lax. Unfortunately, it is predicated on two highly optimistic assumptions. The first assumption is that it is sufficient to treat the modulator as a purely linear system. Unfortunately, a low-order modulator such as ours is susceptible to the nonlinear phenomenon of *dead bands*. As discussed in Section 3.3.1, a dead band is a range of inputs that yield the same periodic output sequence and hence the same post-decimation output. Usually the worst dead band is associated with the output pattern $\{+1, -1\}$. In our modulator with an input of zero, this feedback pattern produces a periodic sequence $x_2 = \{+80\,\text{mV}, -80\,\text{mV}\}$ at the output of the second integrator.[7] Following the methodology of Section 3.3.1, we find that with a small input, δ, the x_2 sequence is shifted up by $A^2\delta$ and the output sequence is therefore unchanged if

$$\delta < \frac{80\,\text{mV}}{A^2}. \tag{7.27}$$

Since the output sequence has an average of zero, an input whose magnitude is δ or less will be indistinguishable from an input of zero. In order to resolve, say, a 10 μV input, we would therefore need

$$A > \sqrt{\frac{80\,\text{mV}}{10\,\mu\text{V}}} = 39\,\text{dB}. \tag{7.28}$$

---

[7]For the structure in Figure 7.4, the steady-state signal at x_2 with $v = \{+1, -1\}$ is $\pm(a_2 - a_1 c_1/2)\,/\,2$.

The second and more pertinent assumption is that the opamp gain is constant. In practice, an opamp's gain varies with its input voltage and this variability causes distortion. Behavioral modeling can be used to quantify the distortion resulting from a given gain curve, but it is more direct, and usually easier, to rely on transistor-level simulations of the amplifier in the context of the modulator.

We can nonetheless put an upper bound on the required gain by noting that finite amplifier gain causes incomplete charge transfer from the input capacitors. Since the magnitude of the associated input-referred error signal is no more than $v_{o,max}/A$, we know that the distortion of a signal near the stable limit ($0.9\,V_{dd} = 1.6\,$V in our case) is less than $0.5\,$V$/A$. If the amplifier gain stays above $80\,$dB, then, no matter how nonlinear the amplifier is, we can be certain that all distortion terms are less than $0.5\,$V$/(1.6\,$V$\times 10^{-80/20}) = -90\,$dBc.

Even if there is no explicit distortion requirement for the ADC, we need to ensure that the loop filter is sufficiently linear that distorted out-of-band quantization noise does not fill in the noise notch. As an estimate of the required linearity, observe in the spectrum of Figure 7.18 that the out-of-band quantization noise density is above the in-band density by nearly $90\,$dB, and thus distortion at the -90-dB level would have an appreciable effect on the in-band noise.

Based on these considerations, we should aim for at least $80\,$dB of amplifier gain and then follow up with modulator simulations to verify that the amplifier's linearity is adequate. It is important to reiterate that an estimate of the required amplifier gain based on purely linear theory is woefully inadequate.

7.6.2 Candidate Amplifier
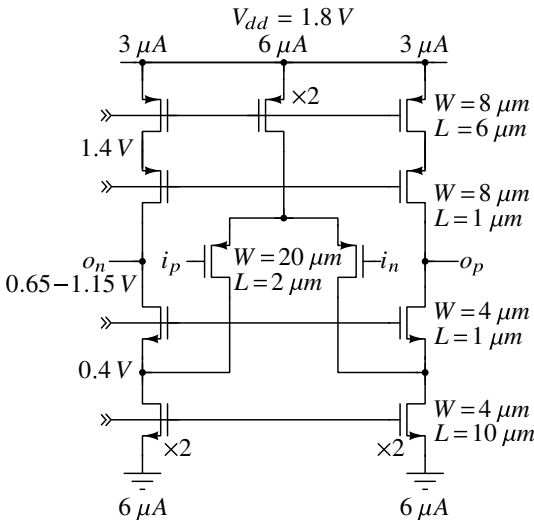
Figure 7.24   Transistor sizes and bias points.

Figure 7.24 shows the schematic of the amplifier intended for the first integrator, an-
notated with bias currents and selected node voltages. Transistors were sized to achieve
the target saturation voltages with 100 mV of margin at the slow/hot corner while the gate
areas of the input pair and the current source devices were made large enough that the $1/f$
noise corner was below 100 Hz. The total current consumption of this amplifier is $12\,\mu A$.
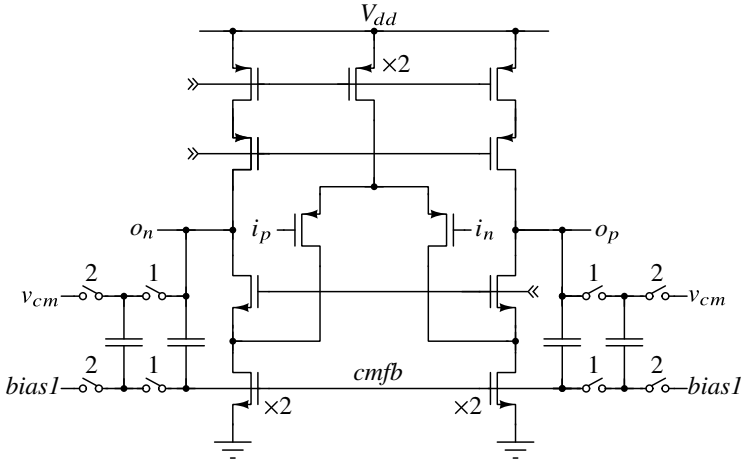
Figure 7.25   Switched-capacitor common-mode feedback.

When an amplifier of this kind is used within a modulator, common-mode feedback
(CMFB) is typically provided by the switched-capacitor network shown in Figure 7.25. In
this circuit, the switched capacitors set the dc voltage on the main CMFB capacitors and
the high gain of the CMFB path ensures that the common-mode of the output is regulated
to within a fraction of a mV. However, since this network takes a few clock cycles to
settle, we advocate doing ac simulations using the ideal common-mode feedback depicted
in Figure 7.26 to check the gain and stability of the amplifier.
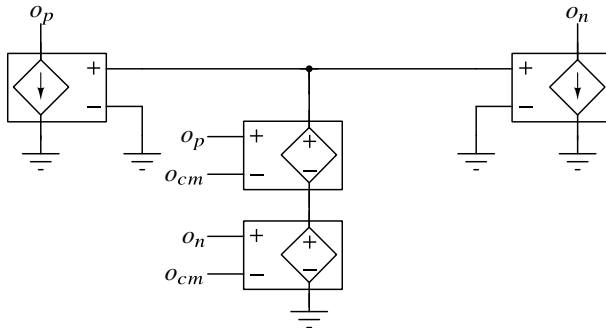
Figure 7.26   Ideal common-mode feedback.

Figure 7.27 plots the gain of the amplifier versus the differential output voltage in
both the nominal case and the slow-hot corner. In the nominal case, the amplifier gain is
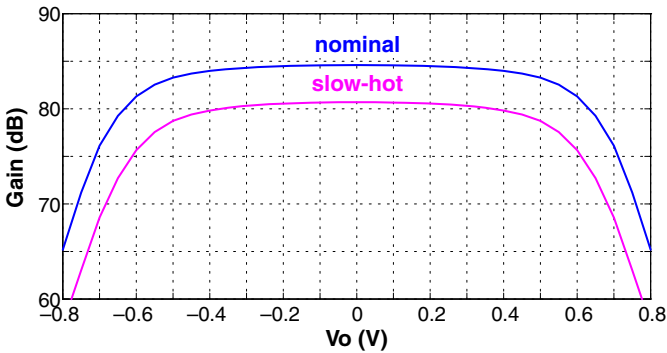
Figure 7.27 Amplifier gain versus output voltage.

greater than 83 dB for $|V_o| <= 0.5$ V, whereas in the slow-hot corner the minimum gain over this output range is about 5 dB lower.
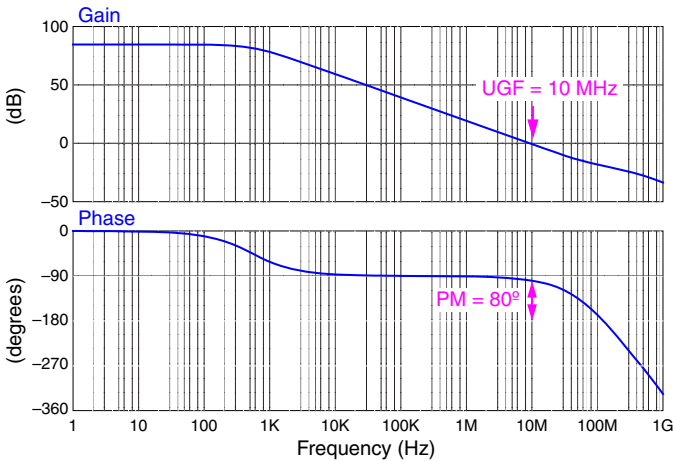
Figure 7.28 Amplifier gain and phase with a 0.5-pF load.

Figure 7.28 shows Bode plots of the amplifier with an output load capacitance of 0.5 pF. The unity-gain frequency (UGF) of 10 MHz provides $2\pi(10\,\text{MHz})/(1\,\text{MHz}/4) \approx$ 16 time constants of settling in a quarter-period of the 1-MHz clock, so the amplifier should have no trouble settling in the allotted time.[8] Also, since the phase margin is a generous 80°, the settling behavior should also be free from ringing.

The second amplifier can be a scaled-down version of the first amplifier. The capacitors in the second stage are so much smaller than those in the first stage that a scaling factor of 10x appears reasonable. However, such aggressive scaling offers diminishing returns. Scaling the first stage by factor of four yields convenient device widths that are close to

---

[8]This quick calculation assumes the amplifier is connected in unity-gain feedback and drives an output load of 0.5 pF. Thus, a 10-MHz UGF corresponds to a settling time constant of $1/(2\pi(10\,\text{MHz}))$. The actual operating condition is somewhat different (the feedback factor is 0.94 and $C_L = 0.4$ pF plus CMFB capacitance), but this fact does not change the conclusion that the amplifier is plenty fast.

the minimum allowed in the process, and the combined power consumption of the two amplifiers is no more than 15% higher than if a 10x scaling factor were applied.

7.7   Intermediate Verification
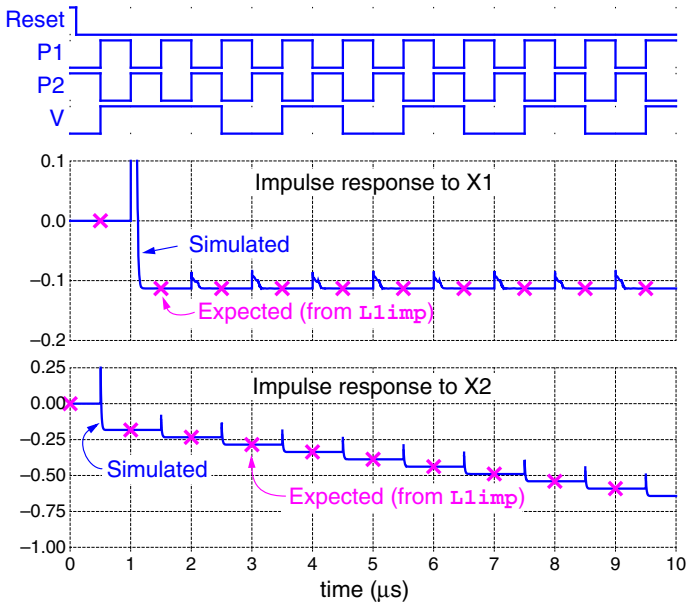
Figure 7.29   Impulse response check – A1 transistorized.

Let's perform a few checks with the first amplifier transistorized. Figure 7.29 shows that the impulse response still passes through the expected points and Figure 7.30 confirms that the modulator still appears to operate properly with a dc input. Such quick checks help establish that the system is ready for more detailed simulations.

Before doing such simulations, let's look more closely at the output voltages of the first integrator. Figure 7.31 shows a zoomed-in view of the single-ended waveforms present at the output of the first integrator when the feedback is $v[n] = -1$. Two features of these waveforms deserve mention.

First note that when x_{1p} changes from 0.9 V to 0.8 V over the course of a clock period, it begins this -0.1-V transition by stepping *up* by nearly 0.6 V! To understand why such counterintuitive behavior happens, consider the half-circuit in Figure 7.32. This figure illustrates the transition from phase 1 to phase 2 when $v[n] = -1$ and $u[n] = V_{dd}$. In this scenario, the voltage on the left side of C_1 increases by V_{dd} and this positive-going step propagates via voltage division to all capacitors in the network virtually instantaneously (limited by the conductance of the switches). After this initial charge redistribution event, the amplifier takes over and drives v_1 to zero and v_o to -0.11 V. Note that although the
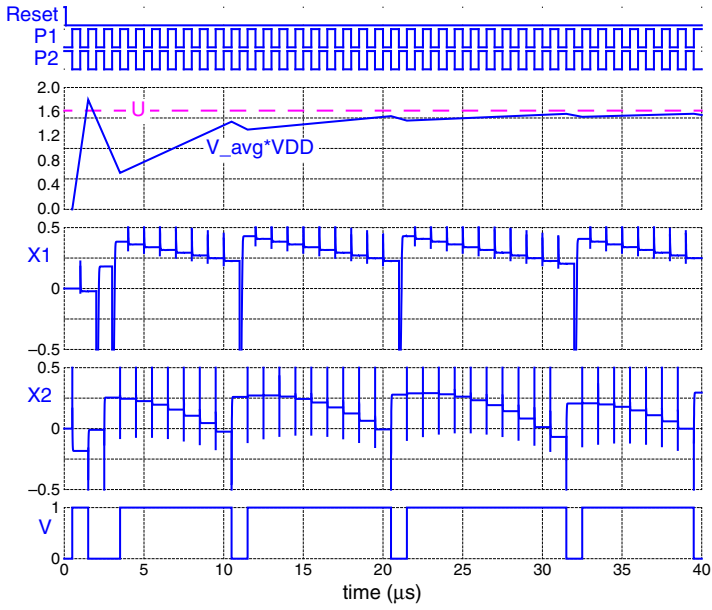
Figure 7.30   Short simulation with dc input – A1 transistorized.
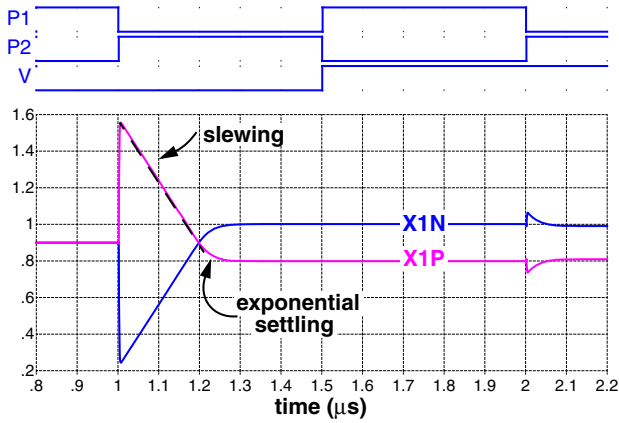
Figure 7.31   A1 output voltages.
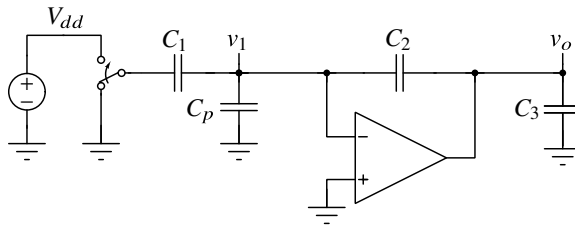
$$\Delta v_1 = \frac{C_1 V_{dd}}{C_1 + C'_p}, \text{ where } C'_p = C_p + \frac{C_2 C_3}{C_2 + C_3}$$

$$\Delta v_o = \frac{C_2 \Delta v_1}{C_2 + C_3}$$

Figure 7.32 V_{ss} to V_{dd} switching event.

initial x_{1p} voltage, at about 1.5 V in Figure 7.31, is well outside the linear range of the amplifier, such excursions are immaterial as long as the amplifier settles.

The slewing portion of the charge transfer operation is controlled by the 3-μA bias current of the transistors in the differential pair and fits within our quarter-clock-period budget. Once the inputs of the amplifier fall within the differential pair's linear range, the amplifier outputs converge exponentially to their final values.
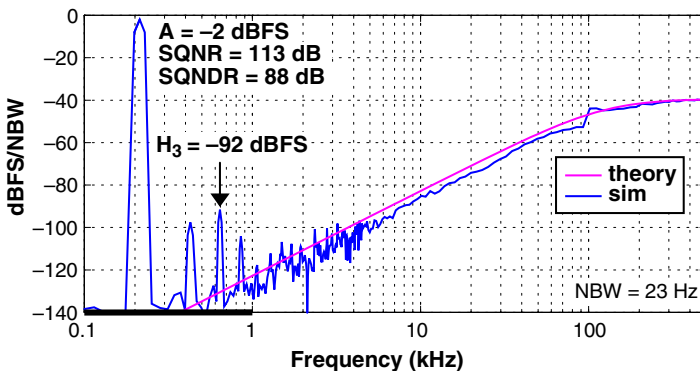
Figure 7.33   Spectrum with the first amplifier transistorized.

Figure 7.33 shows the results of a long modulator simulation with the first amplifier and its common-mode feedback transistorized. Since the quantization noise is still sharply shaped, we conclude that the amplifier is sufficiently linear to prevent distortion of the out-of-band quantization noise from degrading the SQNR. However, the amplifier is not so linear that harmonics of the input signal are negligible. Nonetheless, we deem the −92-dBFS third-harmonic distortion to be adequate. A similar simulation at the slow-hot corner yields a 2-dB degradation in the SNDR, validating our 80-dB target gain.

To verify that the noise of the amplifier is acceptable, we can do AC noise analysis of node x in the circuit shown of Figure 7.34. Node x measures the combined voltage across
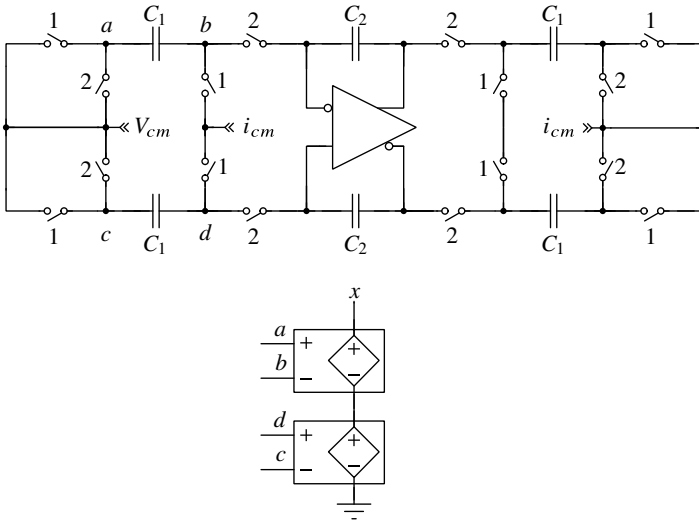
Figure 7.34   Simulation test bench for measuring the noise across the C_1 capacitors.

the input capacitors[9]

$$v(x) = \frac{v(a) - v(b)}{2} - \frac{v(c) - v(d)}{2}, \tag{7.29}$$

so the noise at node x, n_x, is the noise that gets sampled on these capacitors. Since this noise is sampled, we sum the squared values of n_x at all alias frequencies and then take the square root to obtain the sampled-data noise density shown in Figure 7.35(a). Integrating these noise densities (squared, and then taking the square root) yields the curves shown in Figure 7.35(b). Let's compare these simulation results with our kT/C-based theory.

In phase 1, where the amplifier does not play a role, we expect the noise referred to the single-ended input to be $0.5kT/(OSR \cdot C_1) = 3.3\ \mu V_{rms}$ [10]. The simulated value is within 10% of this prediction. In phase 2, the noise of the amplifier yields a value that is 4 dB higher than straight kT/C noise. Approximately 1 dB of this increase is due to the $1/f$ noise of the amplifier. The sum of the phase-1 and phase-2 noise is $\sqrt{3.6^2 + 5.8^2} = 6.8\ \mu V_{rms}$, which corresponds to a level of -99.4 dBFS. So, despite leaving 3 dB of SNR margin, we are now 1.6 dB short of our 100-dB SNR target. (Remember that we only guarantee that the modulator works for inputs below -1 dBFS.) If we were committed to meeting the SNR target, we would be obliged to scale the capacitors and the amplifier by at least a factor of $\mathtt{undbp}(1.6) = 1.4$.[11] Also, if we were concerned about $1/f$ noise, we would adopt a technique such as chopping [2]. To keep the design process moving forward, we choose to leave the capacitors and amplifier as they are.
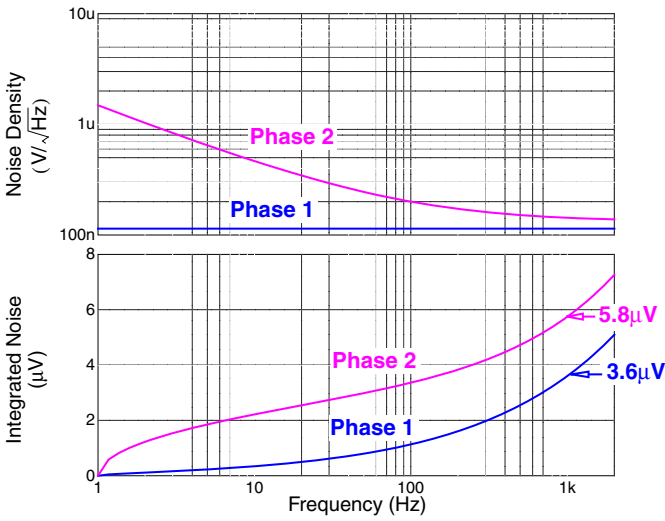
Figure 7.35 In-band noise density and integrated noise.
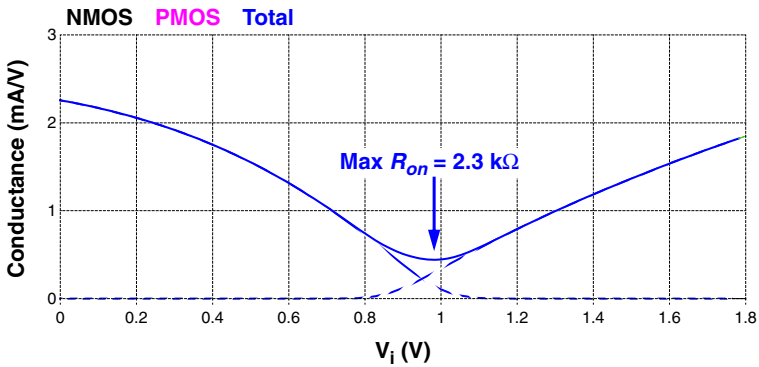
Figure 7.36 Switch conductance versus input voltage for $W_n = 1\,\mu$m and $W_p = 4\,\mu$m (slow-cold corner).

7.8   Switch Design

Clearly, switches that connect to V_{dd} need PMOS devices, and switches that connect to V_{ss} need NMOS devices. Switches that pass both high and low voltages require transmission gates. Figure 7.36 plots the worst-case conductance of a switch consisting of a 1-μm-wide NMOS in parallel with a 4-μm-wide PMOS. This 4:1 ratio was chosen to balance the NMOS and PMOS conductances at the voltage extremes.

From this plot, we see that below $V_i = 0.8$ V the PMOS device contributes little to the switch conductance. If we choose a low amplifier input common-mode voltage, say 0.5 V, then the switches at the virtual ground can be implemented with NMOS devices only. (This is another benefit of choosing an amplifier with a PMOS input pair.) We also see the switch conductance varies over a 5:1 range. The reader may well worry that such a dramatically nonlinear characteristic will cause nonlinearity in the modulator. However, nonlinear switch resistance is unimportant as long as the resistance is low enough that the circuit settles.
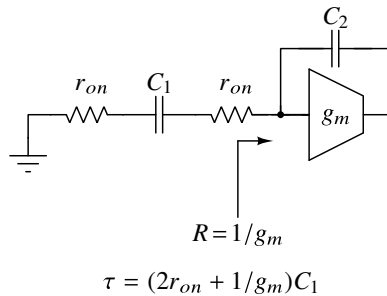
$$\tau = (2r_{on} + 1/g_m)C_1$$

Figure 7.37   Effect of switch resistance on settling time.

Figure 7.37 quantifies the effect of switch resistance on settling time. To have a negligible impact on the settling time, we want $2r_{on} \ll 1/g_m$. Since the worst-case resistance of a 4/1 transmission gate (2.3 kΩ) plus the worst-case resistance of a 1-μm NMOS at $V_i = 0.5$ V (0.6 kΩ) is a small fraction of $1/g_m = 50$ kΩ, this combination can be used for the switches associated with the input sampling network. Other switches are sized similarly, subject to the process-constrained minimum width of 0.5 μm.

7.9   Comparator Design

Figure 7.38 shows the schematic of a popular comparator. This *StrongARM* comparator [3] operates as follows. When ck is low, the comparator is in its reset state, with the differential pair turned off and all nodes above the differential pair pulled to V_{dd}. When ck rises, the current in the differential pair pulls down on the p and q nodes, activating the NMOS devices M_3 and M_4 which in turn pull down on the $\overline{R}$ and $\overline{S}$ nodes. The imbalance

---

[9]The divisions by 2 account for the double-sampling of the single-ended input.

[10]This calculation uses $T = 55°C = 328$ K, since an IC is usually warmer than room temperature.

[11]undbp$(x) = 10^{x/10}$ is a function from the $\Delta\Sigma$ toolbox.

in the pull-down currents from the differential pair is amplified by the positive feedback provided by M_5 and M_6 until either $\overline{R}$ or $\overline{S}$ goes low, which in turn sets the RS latch to the result of the comparison. The RS latch may be constructed with cross-coupled NAND gates, as illustrated in Figure 7.38, or it may be implemented using either of the latches shown in Figure 7.39.
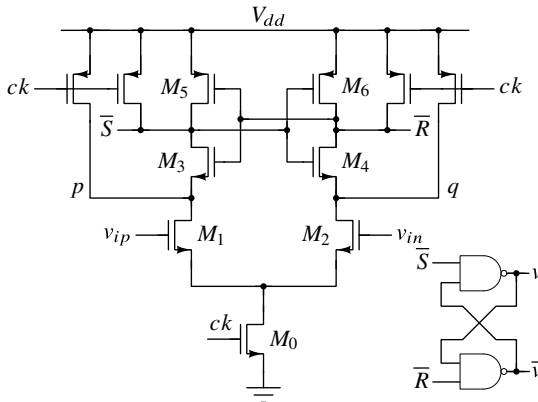
Figure 7.38 StrongARM comparator with RS latch [3].
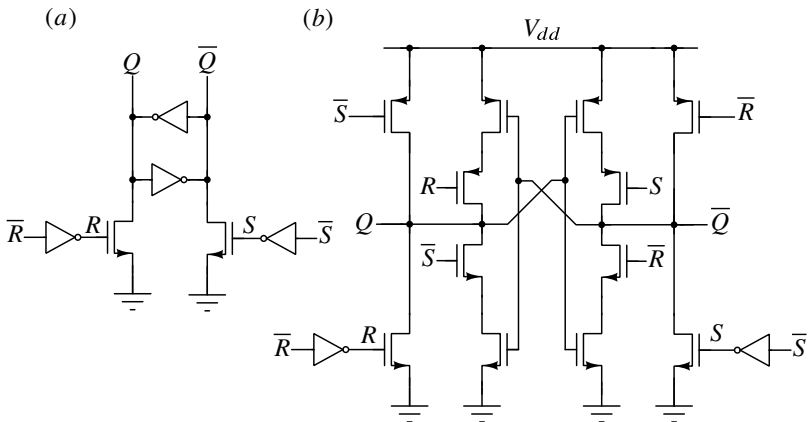
Figure 7.39 Alternative RS latches: (a) jam latch, and (b) symmetric latch.

Figure 7.40 shows the schematic of the comparator that we will study via simulation. Figure 7.41 plots the internal waveforms of the comparator in response to a 1-mV input. We see that the P and Q signals plummet rapidly but the Rb and Sb signals hover around $V_{dd}/2$ for ~100 ps before one of them goes low. The output of the latch changes after another ~100 ps, with the rising transition leading the falling transition courtesy of the cross-coupled NAND gates. The simulated power consumption of the comparator is 0.2 μW when clocked at 1 MHz.

Figure 7.42 superimposes the results of several simulations with ever-smaller inputs. From these curves we see that the time it takes the comparator to resolve a small differen-
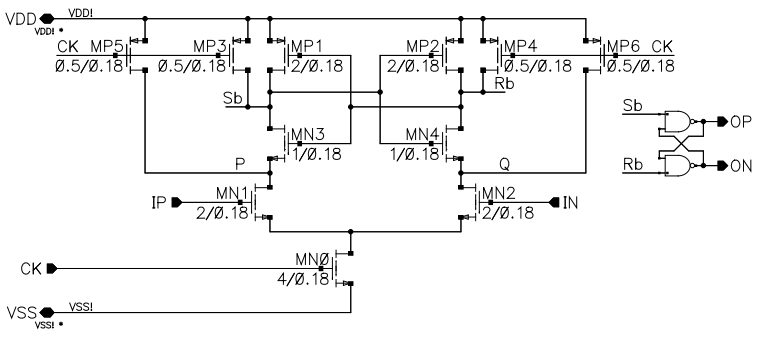
Figure 7.40   Comparator schematic.
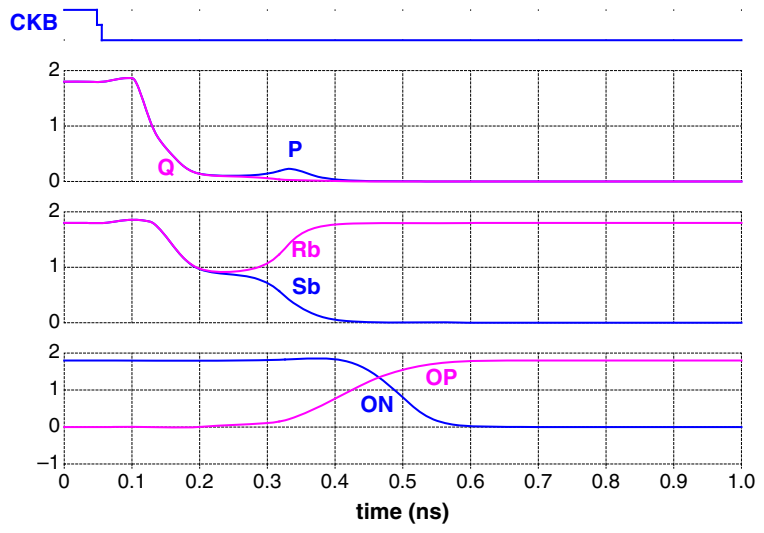
Figure 7.41   Comparator response to a 1-mV input.

tial input is longer than the time needed to resolve a large input, and the time appears to increase by a fixed amount for each decade decrease in v_i. This *metastable* behavior is an important phenomenon and we take a moment to examine it further.
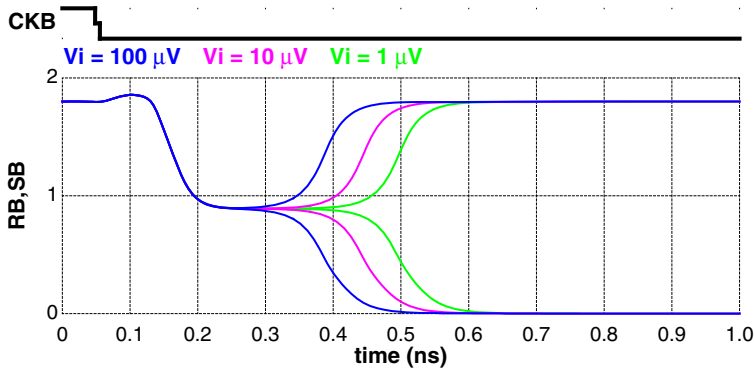
Figure 7.42   Comparator transient simulation with small inputs.

$$\dot{v}_x = -g_m v_y / C$$
$$\dot{v}_y = -g_m v_x / C$$
$$v_d = v_x - v_y$$
$$\dot{v}_d = g_m v_d / C$$

$$\rightarrow v_d(t) = v_o \exp\left(\frac{t}{\tau}\right), \ (\tau = C/g_m)$$
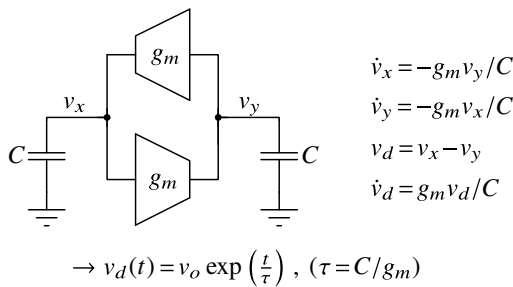
Figure 7.43   Derivation of the regeneration time-constant.

Figure 7.43 contains an abbreviated analysis of the small-signal equivalent of a pair of cross-coupled inverters operating near the metastable point. According to the analysis, any initial condition is amplified with an exponential whose time-constant is

$$\tau_{\text{regen}} = \frac{C}{g_m}. \tag{7.30}$$

Consequently, we expect that reducing the input differential voltage by a factor of k will increase the delay of the comparator by $(\tau_{\text{regen}})(\ln k)$. To check this prediction, Figure 7.44 plots the nominal delay of the comparator as a function of the input voltage and lists the value of the regeneration time constant obtained by fitting a line to the small-input data. The quality of the fit provides qualitative support of our analysis.

As a final point, we note that it is common practice to size the gates connected to $\overline{R/S}$ such that their switching point is below the $\overline{R/S}$ metastable voltage in order to prevent the latch outputs from changing until the comparator has resolved.

Other comparator parameters that are typically of interest include offset, hysteresis, and noise. However, in a single-bit modulator, comparator offset is irrelevant, since it merely results in a corresponding offset at the output of the second integrator. Similarly,
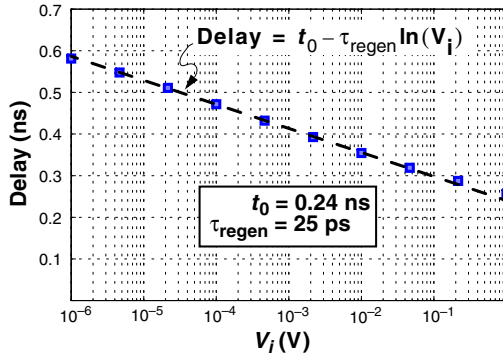
Figure 7.44   Comparator delay versus V_i.

comparator noise is usually insignificant compared to the quantization noise and thus can be ignored. Last, hysteresis can cause changes in the modulator's dynamics, but again the effect of hysteresis on the in-band performance is usually negligible.

7.10 Clocking
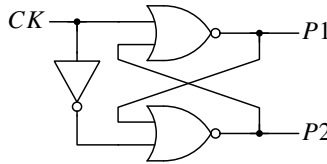
Figure 7.45   A simple non-overlapping clock generator.

Figure 7.45 shows the schematic of a simple non-overlapping clock generator. The cross-coupled NOR gates in this circuit ensure that P1 goes low before P2 can go high, and vice versa. However, there is more to clocking the switches than just ensuring no overlap.
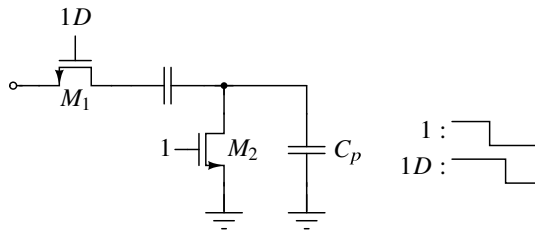
Figure 7.46   Early/late clock phases.

Figure 7.46 shows the sampling arrangement that exists in most switched-capacitor circuits. Let's consider the effect of *charge injection* from the switches M_1 and M_2. The channel charge in M_2 is signal-independent since the drain and source voltages of M_2 are signal-independent whereas the channel charge in M_1 is signal-dependent. In our idealized

view of switched-capacitor circuits, M_1 and M_2 turn off simultaneously. However, if M_2 is still on when M_1 turns off, part of M_1's channel charge is transferred to the sampling capacitor and will be added to the integrating capacitor on the subsequent phase 2. Since this charge is nonlinearly related to the signal, distortion results. However if M_1 is on when M_2 turns off, then a fixed amount of charge is injected, which only introduces a dc offset. Consequently, it is standard practice to delay turning off M_1 until M_2 has been turned off.
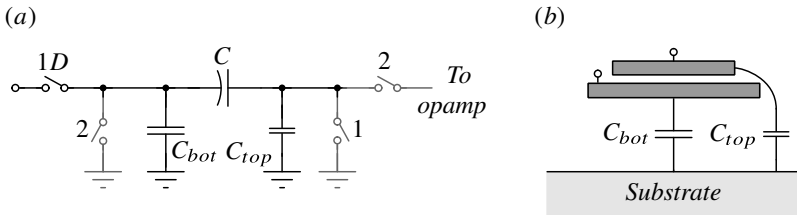
Figure 7.47   (a) Bottom-plate sampling. (b) Capacitor parasitics.

Figure 7.47 shows a cross section of a common capacitor structure. As this figure shows, the parasitic capacitance associated with the bottom plate is much greater than that associated with the top plate. Orienting the capacitor as shown in Figure 7.47(a) minimizes the capacitance on the summing node and also shields the summing node from the substrate. The top plate of the integrating capacitor is usually connected to the summing node for the same reasons.
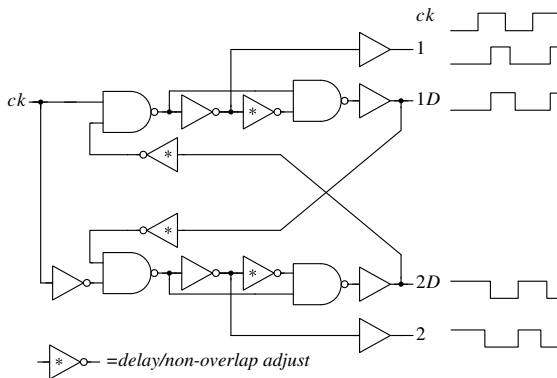
Figure 7.48   Professional clock generator.

Figure 7.48 shows the schematic of a more elaborate clock generator that generates the required clock phases. In this circuit, the non-overlap and delay times can be adjusted via the inverters marked with asterisks.

Since transmission gates require complementary clock signals, non-overlap between the active portions of these control signals and the control signals of switches on the other phase must be ensured. In a high-speed design, it is helpful to align such complementary clocks using a structure such as that depicted in Figure 7.49 in order to maximize the time available for settling.
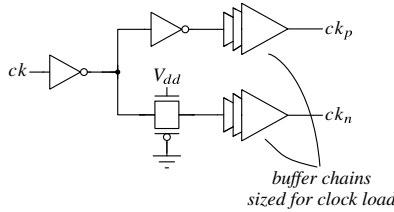
Figure 7.49   Aligning complementary clocks.

7.11 Full-System Verification

Since we started the design process by building and verifying a behavioral model, a good implementation strategy is to design and verify individual blocks in isolation, then in the behavioral version of the modulator, and then together with other blocks in the modulator. Short simulations (impulse response check, dc input) should be performed before launching long spectrum simulations. It can be tempting, especially when time is at a premium, to assemble everything and then try to debug the modulator as a whole. After all, this has to be done eventually. However, this approach is not recommended, since debugging a full modulator using long transient simulations is very time-consuming and usually provides little helpful information. It is not unusual for some unforeseen problem to appear during the verification process, and unfortunately, even this simple design fell into the typical category.
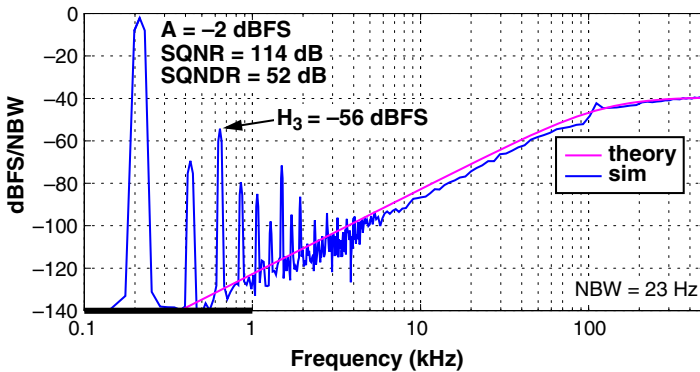
Figure 7.50   A problematic spectrum.

Figure 7.50 shows a spectrum obtained when the design was mostly transistorized. (The clock generator and bias are behavioral.) Since the spectrum exhibits numerous harmonics, with the worst at the -56-dBFS level, clearly a disaster has occurred. Some clues can be gleaned from the spectrum – for example, since the noise-shaping follows the expected shape down to very low frequency without flattening out, we expect the amplifiers are not the source of the problem – but nothing conclusive can be deduced from just this spectrum. By systematically doing short simulations of the modulator with individual elements transistorized, the problem was tracked down to the switches attached to the right sides of the C_1 capacitors (nodes XX and YY in Figure 7.15).

a)

b)
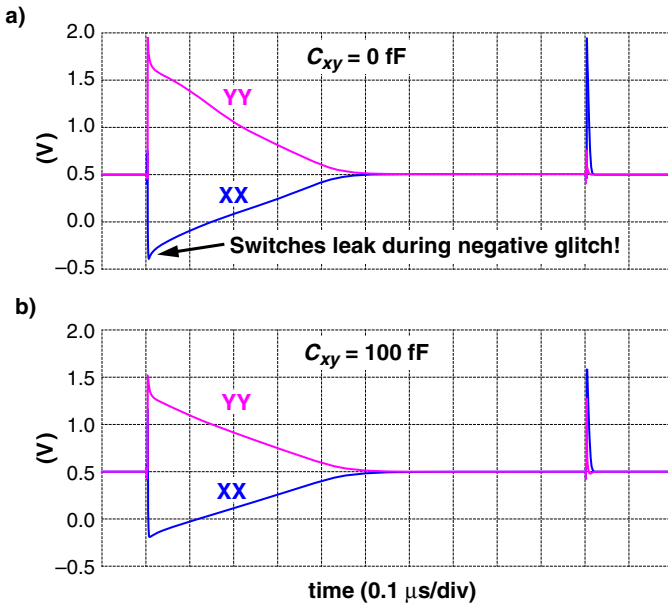
Figure 7.51   Voltage waveforms at the right sides of the C_1 capacitors.

Figure 7.51(a) shows a close-up of the voltages on the XX and YY nodes at a time when the difference between the input and the feedback is large. We see large glitches on these nodes, so large that the XX node goes negative enough to partially turn on the phase-1 NMOS switch and thereby lose some of the charge that was supposed to go to the integrating capacitor. Although every switched-capacitor circuit depends on there being no charge lost to switches that are supposed to be off, this is rarely a problem since the glitches are usually small enough that the off switches stay off. In our case, however, the combination of a large V_{dd}-V_{ss} transition, a low value of ICM, and small parasitics relative to the size of the switched capacitor gave rise to this unusual situation.
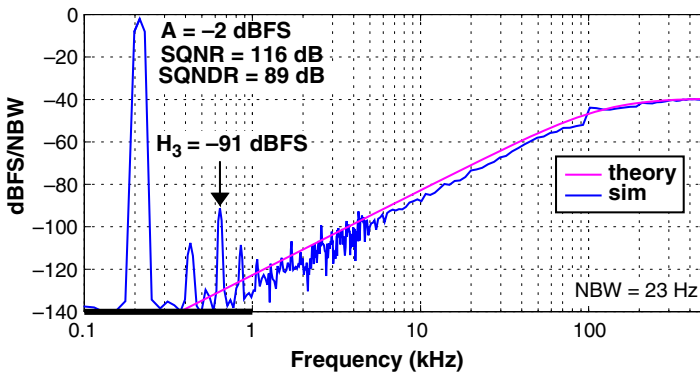
Figure 7.52   Spectrum with most blocks transistorized.

To tame this problem, it was sufficient to add 100 fF of capacitance between the XX and YY nodes. Figure 7.51(b) shows that the negative-going glitch has been reduced by a few hundred mV and now Figure 7.52 shows the spectrum is much improved. This remedy is workable, but comes at the cost of increased noise and a higher opamp load. An alternative which avoids these problems is to turn the troublesome switches off with a negative gate voltage, but this solution requires a negative voltage to be generated. Last, we note that adopting a multi-bit architecture would also avoid the problem since the feedback and the input are unlikely to differ by V_{dd}.
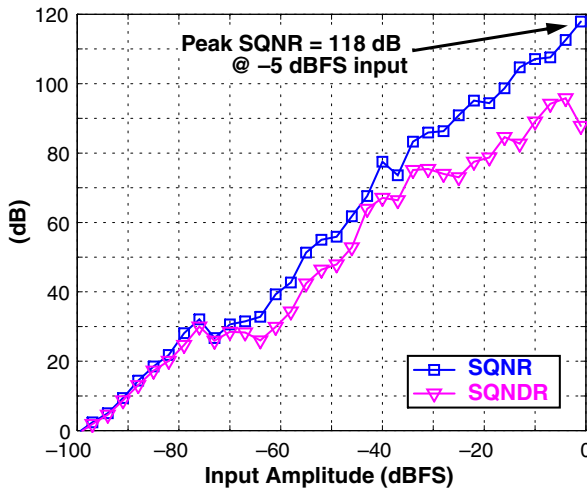
Figure 7.53   Simulated SQNR of the transistorized modulator.

Given the unusual nature of this problem, we choose the brute-force solution and present the simulated SQNR and SQNDR of the patched design in Figure 7.53. According to the simulations, the power consumed by the modulator is $P = 40\,\mu$W. Combining the 40-μW power consumption with the 1-kHz bandwidth and 98-dB DR yields a 172-dB figure-of-merit. The latter number is respectable, but is admittedly more than 10 dB below the current state of the art for this signal bandwidth.

Now that we have gone through a first-pass at the design, we content ourselves to list the steps needed to bring the design up to industrial standards. A commercial design would need to have power-down and debug features added, and would need to be simulated over process, temperature, and voltage corners. Monte Carlo checks (especially of the biasing), as well as verification of reliability and aging, are also recommended. And, of course, all these need to be well documented so that others can modify the design for their specific requirements. Lucky for us, we have the option of stopping here and moving on to fresh territory.

Figure 7.54   Fourth-order CRFB modulator.

Desired difference equations

$$v[n] = Q(x_4[n])$$
$$x_1[n+1] = x_1[n] - a_1 v[n] + b_1 u[n] - g_1 x_2[n]$$
$$x_2[n+1] = x_2[n] - a_2 v[n] + c_1 x_1[n+1]$$
$$x_3[n+1] = x_3[n] - a_3 v[n] + c_2 x_2[n] - g_2 x_4[n]$$
$$x_4[n+1] = x_4[n] - a_4 v[n] + c_3 x_3[n+1]$$

Timing

Figure 7.55   SC circuit and timing diagram implementing the system in Figure 7.54.

7.12 High-Order Modulators

7.12.1 Architecture

The design procedure for a high-order modulator is very similar to the design procedure we followed in our low-order example. Consider the fourth-order CRFB system depicted in Figure 7.54. The difference equations associated with this structure are listed in Figure 7.55. Figure 7.55 also shows a simplified switch-level implementation of this modulator and the associated timing diagram. Using reasoning similar to that applied in the second-order example, let's verify that the schematic is able to implement the difference equations.

First, we see that $v[n]$ is obtained by quantizing $x_4[n]$ at the end of phase 1. Next, $x_1[n+1]$ and $x_3[n+1]$ are evaluated during phase 2, based on the values of $x_2[n]$ and $x_4[n]$ sampled during the preceding phase 1, and on $v[n]$ as it becomes available during phase 2. Then $x_2[n+1]$ and $x_4[n+1]$ are evaluated during phase 1, based on the available value of $v[n]$ and on the $x_1[n+1]$ and $x_3[n+1]$ values sampled during the preceding phase 2. The process then repeats. Note that each amplifier settles individually – there is no need for series settling even in such a high-order modulator.

7.12.2 Capacitor Sizing
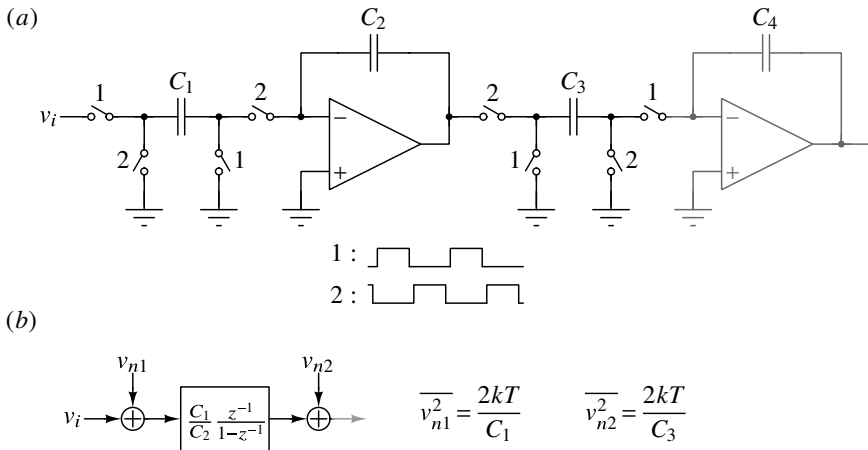
Figure 7.56 Example front-end fragment: (a) simplified schematic; (b) block diagram with noise sources.

As with our low-order example, the coefficients computed after dynamic-range scaling correspond to capacitor ratios, while absolute capacitor sizes are dictated by noise. However, since a high-order modulator is unlikely to use an oversampling ratio that is as high as in our low-order example, we cannot necessarily ignore the contribution of back-end stages to the thermal noise. For example, suppose that $C_1 = 1\,\text{pF}$, and $C_2 = 2\,\text{pF}$, $OSR = 30$ and that we want to make the input-referred noise of C_3 in Figure 7.56 such that the combined input-referred noise of C_1 and C_3 at the passband edge is no more than 1 dB above that due to C_1 alone. To achieve this goal, the input-referred noise of C_3 must be no

more than $\mathrm{undbp}(1) - 1 = 0.25$ times the noise of C_1. Since the gain of the first integrator at the passband edge is

$$A = \left| \frac{C_1/C_2}{e^{j\pi/OSR} - 1} \right| \approx \frac{C_1}{C_2} \frac{OSR}{\pi} \approx 5, \tag{7.31}$$

we therefore require

$$C_3 = \frac{C_1}{(0.25)(5^2)} \approx \frac{C_1}{6}. \tag{7.32}$$

Since the power consumed by an integrator stage is roughly proportional to its capacitive load, the power consumed by the second integrator (INT2) under the assumptions above will be approximately one-sixth of the power needed by the first integrator (INT1). If, on one hand, we had allocated too little of the noise budget to INT2, then C_3, and hence the power consumed by INT2, would have been unduly large. On the other hand, allocating too much of the noise budget to INT2 leaves less for INT1 and its power consumption increases. To find the optimal noise allocation, we can proceed as follows.

Assume that the power consumed by INT1 is proportional to C_1 and likewise that the power consumed by INT2 is proportional to C_3. If we further assume that the proportionality constants are the same, then the objective function

$$f(C_1, C_3) = C_1 + C_3 \tag{7.33}$$

is a measure of the total power consumption.

A specification on the total in-band noise can be captured with the function

$$g(C_1, C_3) = \frac{1}{C_1} + \frac{\alpha^2}{C_3}. \tag{7.34}$$

Here, α^2 is a constant that refers INT2's noise power to the input of INT1. If we follow our earlier strategy of using A_p, the gain of INT1 at the passband edge, to input-refer INT2's noise, then $\alpha^2 = 1/A_p^2$. This approach is appropriate when the application is sensitive to the peak noise density in the passband. However, if the integrated noise is more relevant than the spot noise, then using the mean-square value of the attenuation function provided by INT1 is more appropriate, in which case $\alpha^2 = 1/(3A_p^2)$.

The optimization problem at hand is to find C_1 and C_3 that minimize f subject to an equality constraint on g. This problem, as well as the more general problem involving more than two stages, are readily solved by using the Lagrange multiplier method:

$$\nabla f + \lambda(\nabla g) = 0, \tag{7.35}$$

$$(1,1) - \lambda \left(\frac{1}{C_1^2}, \frac{\alpha^2}{C_3^2} \right) = 0, \tag{7.36}$$

which gives

$$\begin{aligned} C_1 &= \sqrt{\lambda}, \\ C_3 &= \alpha\sqrt{\lambda}. \end{aligned} \tag{7.37}$$

Thus, we see that no matter what the value of the noise constraint is, the minimum power consumption is achieved when the C_3/C_1 ratio is α. A convenient method is therefore to ratio the capacitors as indicated and then scale them in unison to achieve the required

noise. For example, if we assume that $C_1 = 1\,\text{pF}$ in our previous example consumes the entirety of the noise budget (i.e., the noise constraint is $g = 1\,(\text{pF})^{-1}$), then, to account for the second stage, we initially set

$$C_3 = \alpha C_1 = 0.2 C_1 = 0.2\,\text{pF}, \tag{7.38}$$

and compute

$$g = \frac{1}{C_1} + \frac{\alpha^2}{C_3} = \frac{1}{1\,\text{pF.}} + \frac{0.2^2}{0.2\,\text{pF.}} = 1.2\,(\text{pF})^{-1}. \tag{7.39}$$

To achieve the noise target, we therefore need to scale C_1 and C_3 by 1.2:

$$C_1 = 1.2\,\text{pF}, \tag{7.40}$$
$$C_3 = 0.24\,\text{pF}. \tag{7.41}$$

7.12.3   Combining the Noise from Multiple SC Branches
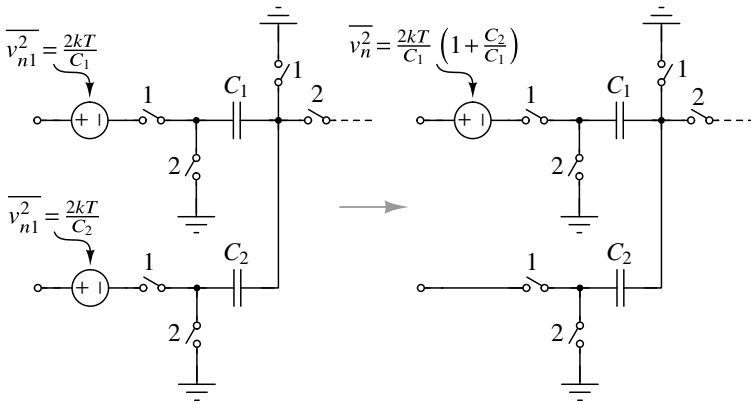
Figure 7.57   How to refer the noise from a second SC branch back to the first.

An important step in the procedure above is to refer the noise from multiple SC branches to a single branch. Figure 7.57 shows a pair of switched-capacitor branches and their associated noise sources. To combine these noise sources into a single source attached to the top branch, first convert the noise voltages into noise charges and add them:

$$\overline{q^2} = \overline{q_1^2} + \overline{q_2^2} = 2kTC_1 + 2kTC_2. \tag{7.42}$$

Next, refer this noise charge back to the input side of C_1 as illustrated in Figure 7.57:

$$\overline{v_n^2} = \overline{q^2}/C_1^2 = \frac{2kT}{C_1}\left(1 + \frac{C_2}{C_1}\right). \tag{7.43}$$

7.13   Multi-Bit Quantization

When single-bit quantization is used, the quantizer is just a single comparator and each DAC is just a single switched-capacitor branch. To implement multi-bit quantization, the

loop filter's output can be digitized using either multiple comparators or multiple comparisons.
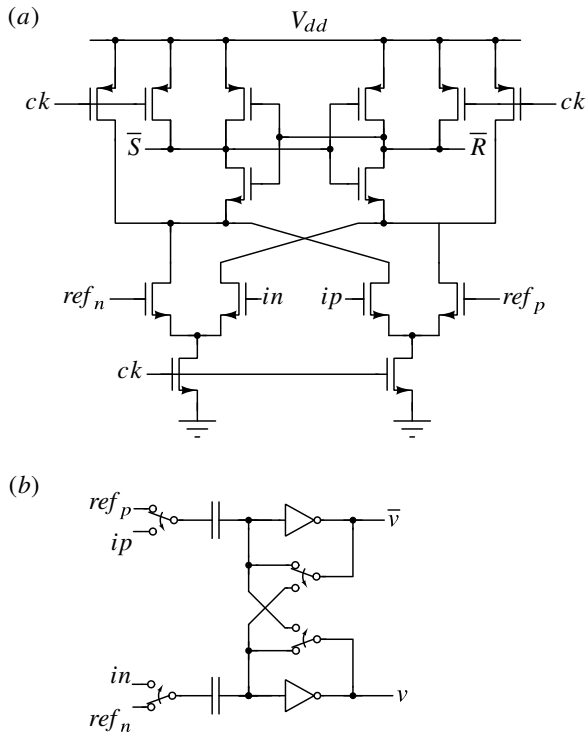
Figure 7.58   Dual-differencing comparators: (a) based on the StrongARM latch, and (b) auto-zeroing.

Let M be the number of steps in the quantizer transfer function. If we choose to perform quantization in one clock cycle, then we need M comparators to compare the loop filter's output to M reference voltages. Since both the loop filter's output and the reference levels are differential signals, we need M *dual-differencing comparators* (Figure 7.58).

The circuit in Figure 7.58(a) replaces the differential pair of the StrongARM comparator with two differential pairs connected in parallel. One differential pair compares i_n with v_{refn} and the other compares i_p with v_{refp}. The difference between the common-mode voltages of these signals should be within the linear range of the differential pairs. The alternative arrangement where i_p and i_n connect to one differential pair, and v_{refp} and v_{refn} connect to the other, is likely to be disastrous, since these two differences are likely to be well outside the linear range of the differential pairs.

The circuit in Figure 7.58(b) uses a pair of inverters that are biased at their trip points during the reference-sampling phase and then connected in positive feedback just after the sampling capacitors are connected to the input signal. Again, v_{refp} should be compared with i_p and v_{refn} should be compared with i_n for best performance. One particularly attractive feature of the circuit is *auto-zeroing*, whereby the offsets of the inverters are suppressed by their open-loop gain.
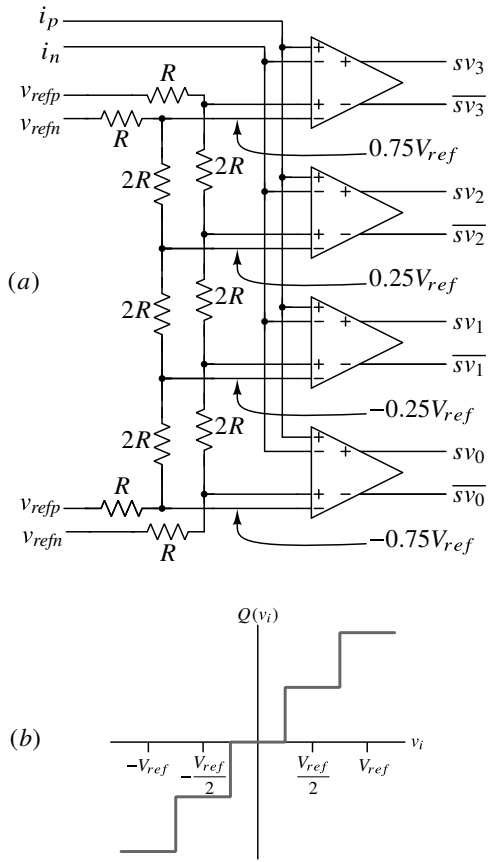
Figure 7.59   A four-step quantizer.

Figure 7.59 shows how $M = 4$ dual-differencing comparators can be used to construct a flash quantizer. The analog input signal is applied to one pair of inputs in all comparators while different reference voltages are applied to the other pairs. The reference voltages for the comparators can be generated using a resistor string connected between the v_{refp} and v_{refn} voltages. (Figure 7.59 shows two such strings for diagrammatic clarity. In practice, a single string is sufficient, but the wiring is more tangled.)
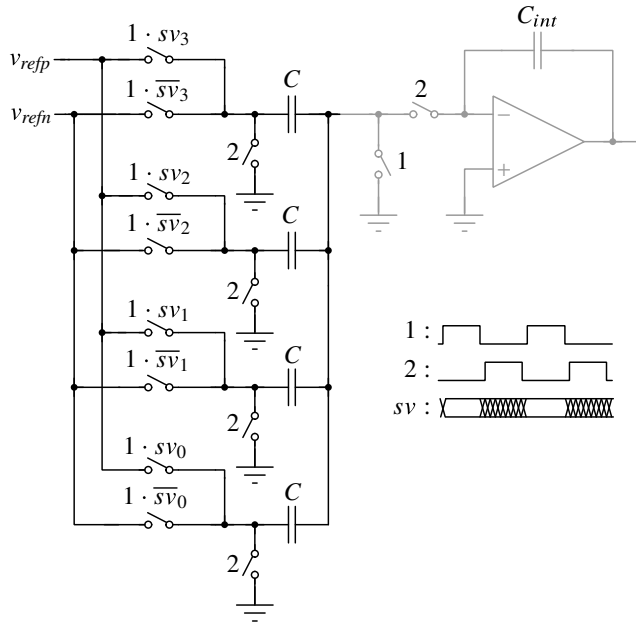
Figure 7.60   A four-element DAC.

Figure 7.60 shows the DAC counterpart to the flash quantizer. This *unary-coded* DAC consists of M identical switched-capacitor branches, each of which is controlled by one of the M comparators. To implement a *binary-coded* DAC, the capacitors would be weighted by powers of two.

In contrast to a flash ADC which resolves $M + 1$ levels using M comparators in a single clock cycle, a *successive-approximation register* (SAR) ADC resolves 2^m levels using a single comparator and $m + 1$ clock cycles. Figure 7.61 illustrates a single-ended implementation of a 3-bit version of such an ADC. This circuit consists of a binary-weighted capacitor array, one comparator, and some control logic. It operates as follows.

In the sampling phase, the comparator is auto-zeroed, and the input voltage is sampled onto the capacitor array. Next, the sampling and autozero switches are opened, and the SAR logic sets the DAC switches such that the MSB capacitor is connected to V_{ref} while the other capacitors are connected to ground. If the sampled value of v_i is above $V_{ref}/2$, then the voltage at node x will be below the threshold of the comparator, and once the comparator resolves this fact, the SAR logic will set the MSB of the data word to 1. If the MSB (b_2) is 1, then the 4C capacitor stays connected to V_{ref}; otherwise, the 4C capacitor is switched to ground when the 2C capacitor is connected to V_{ref}. The comparator is clocked again, and the next data bit is determined. Based on this bit (b_1), the 2C capacitor is either
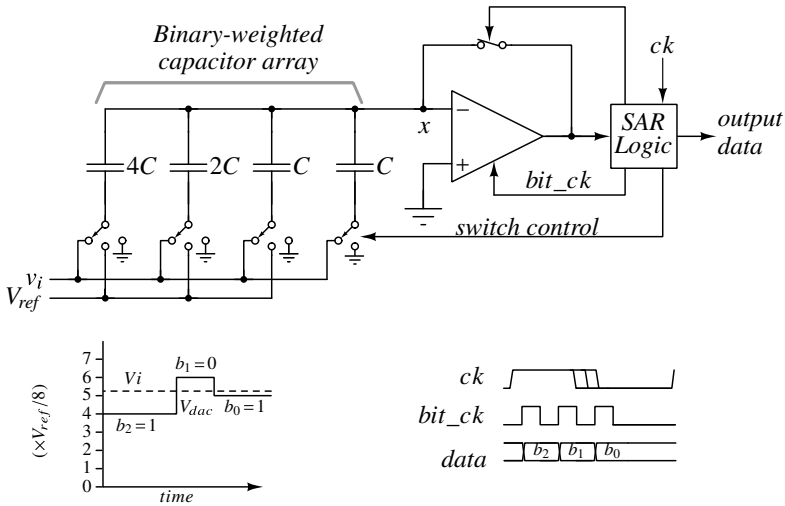
Figure 7.61   Three-bit SAR quantizer (single-ended version).

left connected to V_{ref} or is switched back to ground when the LSB capacitor is connected to V_{ref}. The final comparison resolves the LSB (b_0).

The earliest implementations of SAR ADCs used a high-rate clock to drive the SAR logic, but nowadays the SAR logic generates the bit clock asynchronously in response to each comparator decision. Many technological improvements have been made in recent years, including the use of capacitors in the attofarad range [4]-[7].

The SAR architecture can yield a very power-efficient ADC, especially when the resolution is low, and thus a SAR ADC is a good fit for the quantizer in a $\Delta\Sigma$ ADC. However, since the speed of a SAR ADC is lower than that of a flash ADC, a flash ADC is still needed when maximizing the sample rate is of paramount importance. Interpolating between these two architectural extremes yields such arrangements as multi-bit SAR [8] and two-step [9] ADCs.

7.14   Switch Design Revisited

Section 7.8 considered only simple NMOS, PMOS or transmission-gate switches. We saw that switches having a wide input voltage range exhibit a highly nonlinear conductance, and mentioned that this effect does not cause distortion as long as the circuit settles. A hidden premise in this assertion is that the input is a sample-and-held waveform, which is valid for all signals within the ADC itself but is usually not valid for the input to the ADC. If the ADC's input is a continuous-time signal, then nonlinear conductance of the input switch can limit the high-frequency linearity of the ADC.

Figure 7.62 illustrates the concept behind a solution to the problem of input-dependent switch resistance. In this *bootstrapped switch*, the V_{gs} of the NMOS switch is fixed at V_{dd}, thereby making the switch's on-conductance independent of the input voltage. Figure 7.63 shows an implementation that prevents over-stress on any device [10].
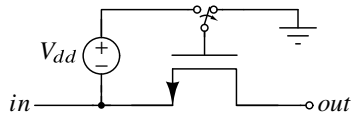
Figure 7.62   Bootstrapped switch concept.
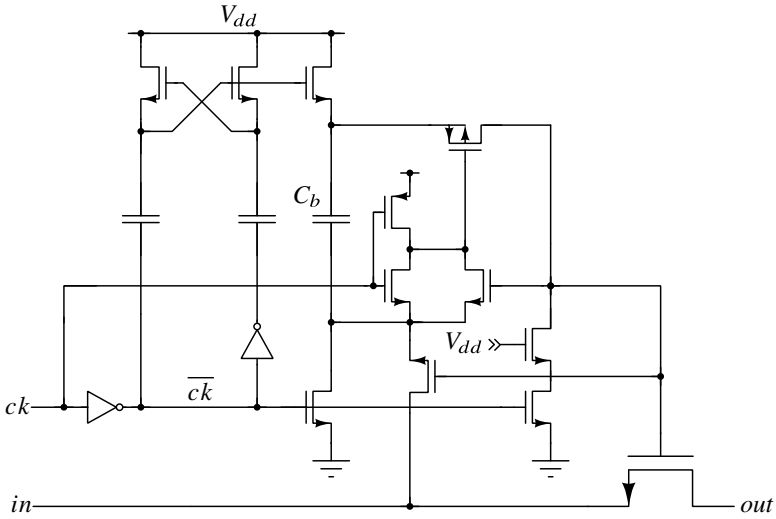
Figure 7.63   Abo's implementation of a bootstrapped switch [10].
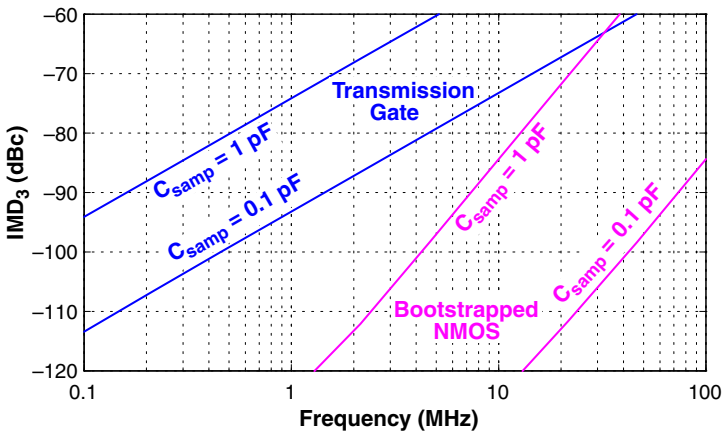
Figure 7.64   Distortion with regular and bootstrapped switches

To demonstrate how effective this technique can be, Figure 7.64 plots the simulated IMD3 of the voltage across a switched capacitor driven by a pair of $0.25\text{-}V_p$ sine waves centered at $V_{dd}/2 = 0.9\,\text{V}$. One pair of curves uses a transmission-gate switch containing a $1\text{-}\mu\text{m}$ NMOS in parallel with a $4\text{-}\mu\text{m}$ PMOS while the other uses a bootstrapped $1\text{-}\mu\text{m}$ NMOS. In both cases, reducing the size of the sampling capacitor relative to the switch reduces distortion, but the transmission gate fares so poorly compared to the bootstrapped switch that good high-frequency performance, say, -90-dBc IMD_3 at $10\,\text{MHz}$ would require a transmission gate that is more than 100 times the size of a bootstrapped switch.

Other switch tricks include switching the back-gate (to reduce the nonlinearity of C_{db}) or bootstrapping the back-gate (to hide C_{db}) [11].

7.15   Double Sampling
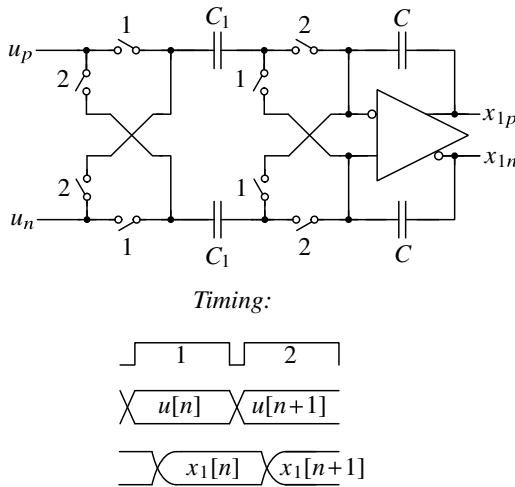
Figure 7.65   Double-sampling integrator [12].

Figure 7.65 shows the structure of a *double-sampling* integrator [12]. Since this circuit does integration on both phase 1 and phase 2, the sample rate is double the clock rate. In the context of a $\Delta\Sigma$ ADC, doubling the sample rate (for a given bandwidth) provides a sizable SQNR improvement and thus double-sampling is very attractive. Furthermore, since double-sampling makes use of the idle phase of the opamp, a double-sampled ADC is more efficient than one that does not use double sampling.

To derive the transfer function of this integrator, first note that it suffices to only analyze what happens when the circuit transitions from phase 1 to phase 2, since the same thing happens during the other transition. Next note that the input common-mode voltage (v_{icm}) of the amplifier is irrelevant, and thus we can assume that $v_{icm} = 0$. Figure 7.66 summarizes the analysis. On phase 1 the input capacitors sample $v_i[n]$, and on phase 2 the sum of $v_i[n]$ and $v_i[n + 1]$ is accumulated on the feedback capacitors with a C_1/C_2 factor.

Phase 1

$$q_1[n] = C_1 \frac{-v_i[n]}{2} \qquad q_2[n] = C_2 \frac{v_o[n]}{2}$$
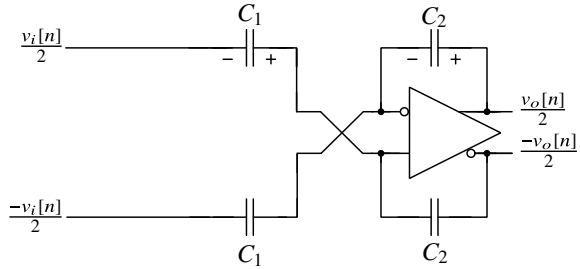
Phase 2

$$q_1[n+1] = C_1 \frac{-v_i[n+1]}{2} \qquad q_2[n+1] = C_2 \frac{v_o[n+1]}{2}$$
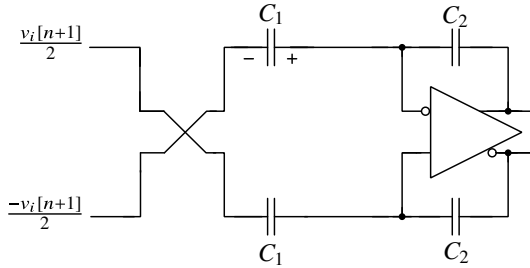
$$\Delta q_2 = \Delta q_1 = \frac{C_1}{2}(v_i[n+1] + v_i[n])$$

$$\rightarrow v_o[n+1] = v_o[n] + \frac{C_1}{C_2}(v_i[n+1] + v_i[n])$$

$$\rightarrow V_o(z) = \frac{C_1}{C_2} \frac{1 + z^{-1}}{1 - z^{-1}} V_i(z)$$

Figure 7.66   Analysis of the double-sampling integrator.

Thus, the transfer function of this integrator is

$$\frac{V_o(z)}{V_i(z)} = \left(\frac{C_1}{C_2}\right)\left(\frac{1 + z^{-1}}{1 - z^{-1}}\right). \tag{7.44}$$

The advantages of the double-sampling integrator shown in Figure 7.65 are significant, but this circuit as drawn provides no means to set the input common-mode voltage. Adding small conventional switched-capacitor branches to the summing nodes solves this problem.
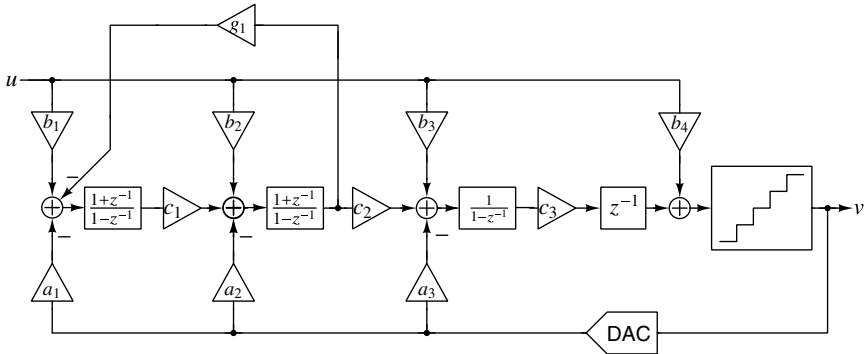
Figure 7.67   Loop filter with double-sampled integrators.

Figure 7.67 shows the structure of a $\Delta\Sigma$ ADC whose loop filter employs such double-sampled integrators. Note that it is not possible for every integrator in the loop filter to use the double-sampled integrators of Figure 7.65, since the L_1 loop gain would have $L_1(-1) = 0$ and this constraint would not allow the loop to support an arbitrary NTF. To overcome this limitation, it suffices for the last integrator to have the more conventional $1/(1 - z^{-1})$ integrator transfer function. This integrator can be implemented with a pair of ping-ponged standard strays-insensitive switched-capacitor branches. Note that since this integrator is preceded by several other integrators, mismatch between the ping and pong branches is not problematic.

7.16   Gain-Boosting and Gain-Squaring

The amplifiers in our low-speed example system had the luxury of using long-channel devices to achieve high dc gain. In high-speed applications, however, the large parasitic capacitances of such long-channel devices make the amplifier unacceptably slow. In this section we examine two techniques that enhance an amplifier's gain without drastically compromising its speed.

The first technique operates at the transistor level. Figure 7.68(a) shows a *gain-boosted* cascode [13]. This circuit uses an auxiliary opamp to enhance the g_m of the cascode by a factor equal to the opamp gain. The gain of the composite amplifier is therefore increased by the same factor. Figure 7.68(b) illustrates the differential version of this arrangement.

(*a*)   (*b*)
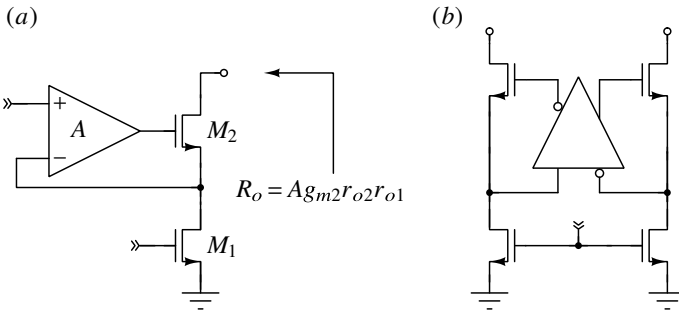
Figure 7.68   Gain-boosted cascode (a) single-ended (b) differential [13].

$R_o = Ag_{m2}r_{o2}r_{o1}$
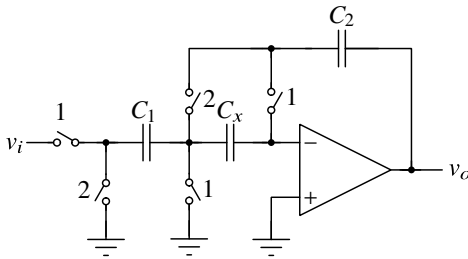
Figure 7.69   Gain-squaring integrator [14].

The second technique operates at the amplifier and switch level. The *gain-squaring* circuit shown in Figure 7.69 uses capacitor C_x to sample the voltage at the negative input terminal of the opamp during phase 1, and then puts this capacitor in series with the amplifier during the integration phase [14]. If C_x dominates the input capacitance of the amplifier, then the effective gain of the amplifier is increased, and both the amplifier's offset and $1/f$ noise are canceled.

7.17   Split-Steering and Amplifier Stacking

In simulation, our example second-order modulator promised a FOM of 172 dB with a bandwidth of 1 kHz. The current FOM record-holder is a fourth-order 1-bit feedforward $\Delta\Sigma$ modulator implemented in a 0.35-μm process [15]. When clocked at 640 kHz, this ADC achieves $FOM = 185$ dB for the same 1-kHz bandwidth as our paper design. Since a 13-dB improvement in FOM corresponds to a 20x improvement in power efficiency, the efficiency of this ADC is truly remarkable. Let's take a quick look at the circuit techniques used to achieve this record-setting efficiency.

Figure 7.70(a) shows one half of a pseudo-differential inverter-based amplifier. This circuit has three attributes that make it more efficient than the folded-cascode amplifier we used in our example modulator. First, since the input is applied to both NMOS and PMOS devices sharing a common bias current, the transconductance is double that of a single device operating with the same bias current. Second, since all of the bias current is used to realize transconductance, there is no current wasted on other amplifier functions, such as folding to increase swing and extend the common-mode input range. Last, unlike a class-A amplifier such as a folded cascode, the slew current is not limited by a fixed bias
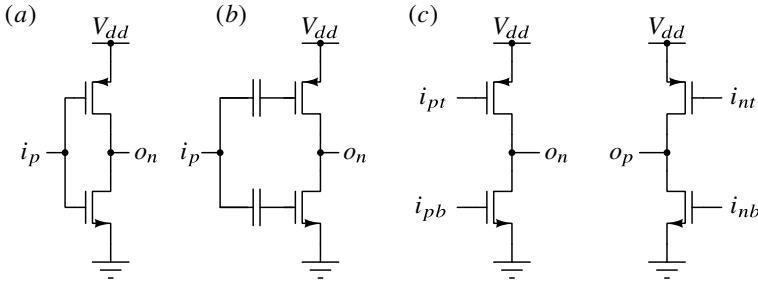
Figure 7.70   (a) Inverter-based amplifier. (b) With level-shifting capacitors. (c) With separate inputs.

current. The disadvantages of this inverter-based amplifier are that the gain is limited to the self-gain of a single MOS device, that the available output swing is limited to the sum of the threshold voltages of the MOS devices, and that the bias point varies with supply, process and input common-mode. In fact, if the threshold voltages are large and the supply voltage is low, then the transistors may essentially be off!

Gain can be improved (at the expense of output swing) by adding cascodes and by gain-boosting, while the other two disadvantages can be addressed by the arrangement in Figure 7.70(b). In this circuit, level-shifting capacitors allow the bias voltages of the NMOS and PMOS devices to be set independently, and thereby allow the bias current to be made independent of process and supply voltage. Unfortunately, the level-shifting capacitors contribute kT/C noise and add attenuation. To avoid a noise penalty, the level-shifting capacitors therefore need to be large.

The circuit of Figure 7.70(c) does away with the level-shifting capacitors to yield an amplifier with two pairs of inputs. This amplifier can be used in the *split-steering integrator* shown in Figure 7.71. This integrator takes advantage of the fact that a standard SC integrator can support an arbitrary input common-mode. In this integrator, the common-mode of the top input pair is set to VT while that of the bottom pair is set to VB. A bias circuit creates these two voltages such that the desired operating point is established despite process, temperature, and supply-voltage variations. At low supply voltage, the VT voltage can even be lower than the VB voltage. Common-mode feedback is achieved via these bias voltages.

In a switched-capacitor circuit, the signal power is proportional to $(V_{dd})^2$, and the noise power is inversely proportional to C. Thus, for a given SNR, increasing V_{dd} by a factor k allows C, and hence g_m, to be made k^2 times smaller. This trade-off indicates that FOM can be improved by using a large supply voltage. We have seen that the split-steering integrator of Figure 7.71 makes efficient use of the supply current to realize a given transconductance, but can we take advantage of a higher supply voltage to increase the transconductance efficiency further?

Figure 7.72 depicts a pair of integrators (shown single-ended for simplicity) that have been stacked so that the supply current runs through both amplifiers. Since the composite transconductance is double that of a single amplifier, this arrangement provides a way to exploit a high supply voltage. However, there are two penalties associated with amplifier stacking. The first is that the output swing of each integrator is halved, and thus the integrating capacitor needs to be doubled and the input-referred noise of subsequent stages
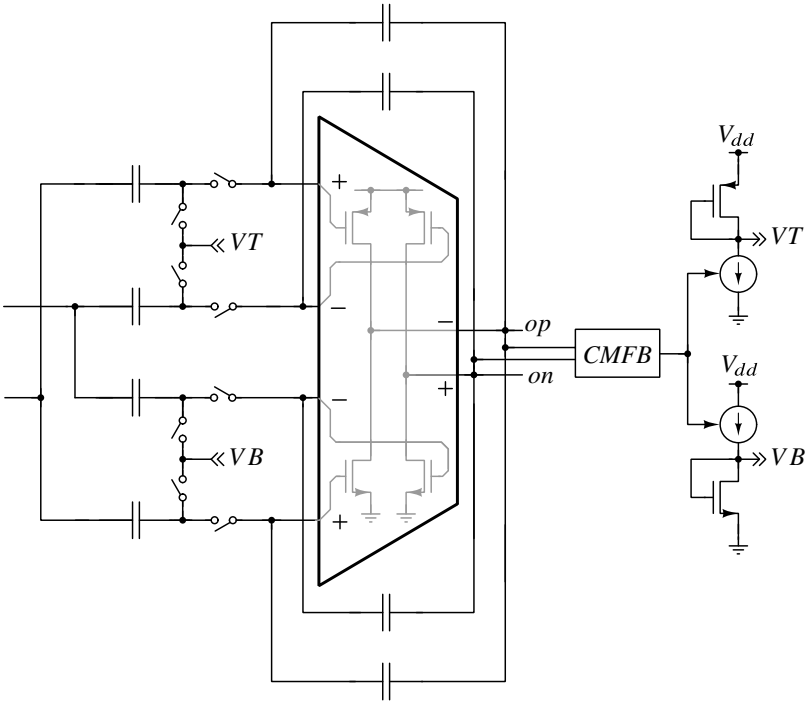
Figure 7.71   Split-steering integrator [15].
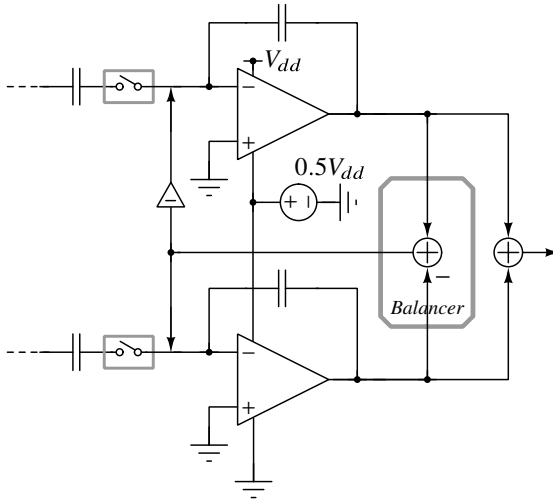
Figure 7.72   Stacked integrator [15].

becomes more significant. With $OSR = 320$, the in-band gain of the integrator is nonetheless so large that the power needed by the backend stages remains negligible. The second disadvantage is that additional circuitry, namely the *balancer* in Figure 7.72, is needed to ensure that the two integrators behave as one. In [15], the balancer is implemented with a small passive switched-capacitor subtractor and a non-noise-critical amplifier. In that work, the noise contribution of the balancer is only 1% of the ADC's total noise.
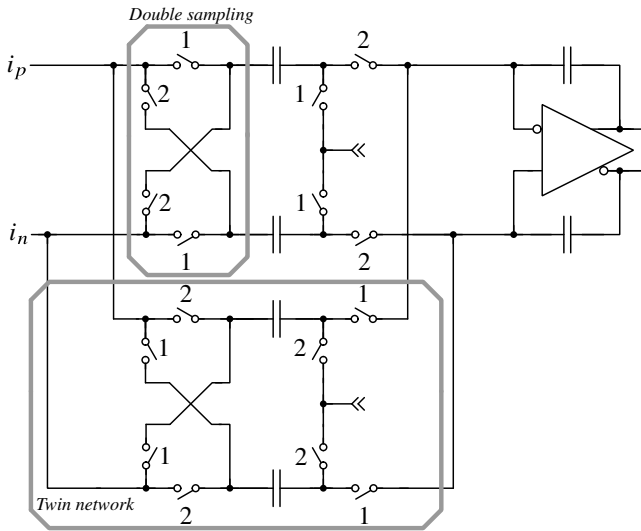
Figure 7.73   Twin double-sampling [15].

The final trick employed in this ADC is *twin double-sampling*, illustrated in Figure 7.73. The double-sampling switch arrangement on the left side of the capacitors effectively doubles the signal swing, thereby allowing the sampling capacitance to be reduced. Splitting the sampling capacitance into two halves driven in counterphase allows both clock phases to be used for integration. Noise analysis shows that for the same total sampling capacitance as the fully double-sampling integrator (Figure 7.65), the twin double-sampling configuration achieves the same low-frequency input-referred noise but requires an integrating capacitor that is only half as big.[12]

Figure 7.74 shows the measured in-band noise power as a function of the dc input. Based on this data and on sine-wave testing, a dynamic range of 136 dB is observed. The total power consumption is 13 mW from a 5.4-V supply. Most of the power is consumed by the first integrator (55%) and clock generation (40%), with the remaining 5% consumed by the other three integrators and the comparator. Chopping was used to suppress $1/f$ noise.
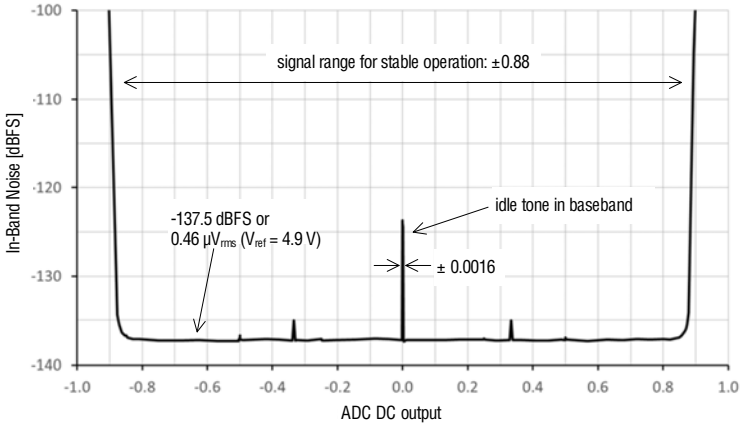
Figure 7.74   Measured noise power versus dc input, from [15].
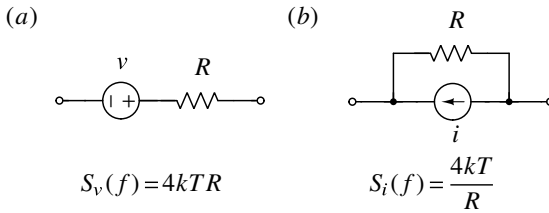
Figure 7.75   Resistor thermal noise models.
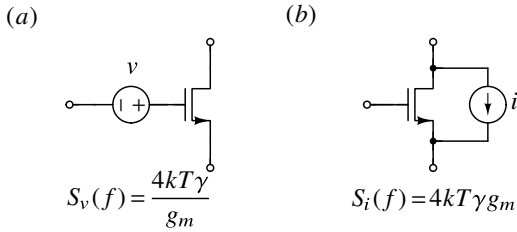
Figure 7.76   Transistor thermal noise models.

7.18   Noise in Switched-Capacitor Circuits

In this section we take a more detailed look at noise in switched-capacitor circuits. For ease of reference, the thermal noise models of a resistor and a MOS transistor in saturation are given in Figure 7.75 and Figure 7.76, respectively. For a MOS transistor used as a switch, the noise model of a resistor applies. As illustrated in Figure 7.75, thermal noise can be modeled by a voltage source in series with the resistor or with a current source in parallel with the resistor. A similar duality applies to the transistor: thermal noise can be modeled with a voltage source in series with the gate or with a current source across the drain and source terminals. The noise sources in each case are white, meaning the expressions for the power spectral densities $S_v(f)$ and $S_i(f)$ are independent of f. The single-sided power spectral density (PSD) of the resistor's voltage noise is

$$S_v(f) = 4kTR, \tag{7.45}$$

where $k = 1.38 \times 10^{-23}$ J/K is Boltzmann's constant and T is the temperature in Kelvin. Similarly, the PSD of the transistor's current noise is

$$S_i(f) = 4kT\gamma g_m, \tag{7.46}$$

where γ is a device-dependent fitting parameter. The theoretical value of γ is $\frac{2}{3}$, but there are more than a few reports of measured values in the 1–2 range.

$$S_v(f) = \frac{4kTR}{1+(2\pi f RC)^2}$$

$$\overline{v^2} = \int_0^\infty S_v(f)\, df = \frac{kT}{C}$$
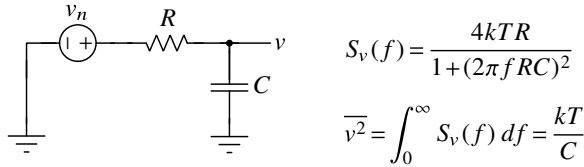
Figure 7.77   Analysis of the noise across a capacitor.

White noise has infinite power because the integral of $S(f)$ over all frequencies is unbounded. However, in a real circuit, capacitance band-limits the noise and makes the noise power finite. For example, Figure 7.77 shows a noisy resistor connected to a capacitor and provides the key steps that lead to the result we used in our first-order noise analysis, namely that the mean-square voltage across the capacitor is[13]

$$\overline{v^2} = \frac{kT}{C}. \tag{7.47}$$

The above result is a consequence of the fact that the noise bandwidth of a single-pole filter is

$$NBW_1 = \frac{1}{4RC}. \tag{7.48}$$

[12]The author is grateful to Matthias Steiner for sharing this observation. To understand why the fully double-sampling integrator is at a disadvantage, note that noise on the sampling capacitors sees a $1 + z^{-1}$ transfer function to the output. This transfer function amplifies the noise density at low frequencies by a factor of 4.

[13]This result can also be obtained by appeal to the *equipartition of energy* physical principle. According to this principle, the mean energy associated with any degree of freedom in a system at thermal equilibrium is $kT/2$. Since the energy of a capacitor C charged to a voltage v is $Cv^2/2$, equipartition of energy tells us that $\overline{Cv^2}/2 = kT/2$, which is identical to (7.47). Similar results apply to the mean-square noise current in an inductor, or the mean-square displacement of a mass in mass-spring system.

Since the voltage noise of the resistor is proportional to R while the noise bandwidth is inversely proportional to R, multiplying NBW_1 by the $4kTR$ noise density of the resistor yields the resistor-independent result of (7.47).

Sampling the voltage across the capacitor yields a discrete-time sequence that has the same power as (7.47). If the sample rate is low ($f_s < 1/RC$ is sufficiently low since the initial condition associated with the previous sample is attenuated by at least 2π time-constants), then the correlation between successive samples is close to zero and thus the discrete-time sequence is essentially white.

Since sampling the voltage on a capacitor C driven through a resistance R is equivalent to opening a switch whose on-resistance is R, we arrive at the conclusion that the mean-square noise voltage associated with the charging phase of a switched capacitor C is kT/C.
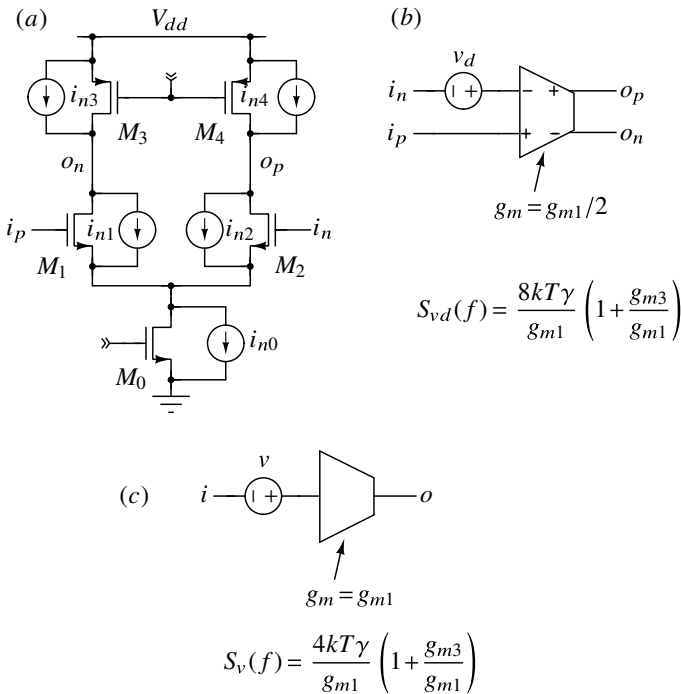
Figure 7.78   Opamp noise sources.

To analyze the noise associated with the charge-transfer phase, we need a noise model for the amplifier. Figure 7.78 a shows a simple differential CMOS opamp and its internal noise sources. We will assume that the circuit is symmetrical, that is, M_1 matches M_2 and M_3 matches M_4. By shorting the output nodes to ground and analyzing the resulting circuit, we find that the differential output current is

$$i_d = \frac{i_{n1} + i_{n2} + i_{n3} + i_{n4}}{2}. \tag{7.49}$$

(The noise associated with M0, the current source for the differential pair, splits equally between M1 and M2 and thus does not appear in i_d. Also note that if the current-sources are

cascoded, the noise of the cascodes is attenuated by a $g_m r_o$ factor and hence is negligible.) Since the PSDs of i_{n1} and i_{n2} are $4kT\gamma g_{m1}$ while the PSDs of i_{n3} and i_{n4} are $4kT\gamma g_{m3}$, the PSD of i_d is one quarter of the sum of these PSDs:

$$S_{i_d} = 2kT\gamma(g_{m1} + g_{m3}). \tag{7.50}$$

As indicated in Figure 7.78(b), this spectral density can be reflected to the input by dividing by $g_m^2 = (g_{m1}/2)^2$ to yield

$$S_{v_d} = \frac{8kT\gamma}{g_{m1}}\left(1 + \frac{g_{m3}}{g_{m1}}\right). \tag{7.51}$$

To simplify analysis, we often use the half-circuit model in Figure 7.78c, wherein

$$S_v = \frac{4kT\gamma}{g_{m1}}\left(1 + \frac{g_{m3}}{g_{m1}}\right). \tag{7.52}$$

Note that when performing noise analysis on a half-circuit, we need to double the single-ended noise power to obtain the differential noise power.

Absorbing the parenthesized term of (7.52) into γ leads to the definition

$$\gamma_{\text{amp}} \equiv \gamma\left(1 + \frac{g_{m3}}{g_{m1}}\right). \tag{7.53}$$

In the best case, where $\gamma = \frac{2}{3}$ and the noise from the current-source devices is negligible, γ_{amp} is less than 1: $\gamma_{\text{amp}} = \gamma = \frac{2}{3}$. However, since

$$g_m = \frac{2I_d}{V_{\text{eff}}}, \tag{7.54}$$

the V_{eff} of the current source devices must be many times that of the differential pair in order for the noise from the current source devices to be negligible. A more realistic assumption is that the V_{eff} ratio is 2, which implies $g_{m3}/g_{m1} = \frac{1}{2}$ and hence $\gamma_{\text{amp}} = \frac{2}{3}(1 + \frac{1}{2}) = 1$. For a folded-cascode opamp that has V_{eff} of all its current-source devices set to double that of the differential pair, $\gamma_{\text{amp}} = \frac{5}{3}$. Thus, typically $\gamma_{\text{amp}} \geq 1$. With a model of amplifier noise in hand, we can now analyze the effect of amplifier noise during the charge-transfer phase.

Figure 7.79 illustrates the procedure. Noise comes from the switches (represented by R_1 and R_2 in Figure 7.79(a) and the amplifier. The switch and amplifier noise sources are independent, so we will calculate their effects individually and then combine them in the mean-square sense.

For switch noise, we begin by merging R_1 and R_2 into $R = R_1 + R_2$ as shown in Figure 7.79b. Since the impedance looking to the right of R is $1/g_m$, the circuit is equivalent to an RC circuit in which the resistor is replaced by $R_{\text{eq}} = R + 1/g_m$. Consequently, the integrated noise across the capacitor due to the switches is

$$\overline{v^2} = \frac{S_{v_1}}{4R_{\text{eq}}C_1} = \frac{kTR}{(R + 1/g_m)C_1}. \tag{7.55}$$

According to this result, making $R \ll 1/g_m$ will minimize switch noise.

Next, consider amplifier noise. As shown in Figure 7.79c, now looking to the right of the resistor we see a circuit whose Thevenin equivalent is a resistance $1/g_m$ in series with a
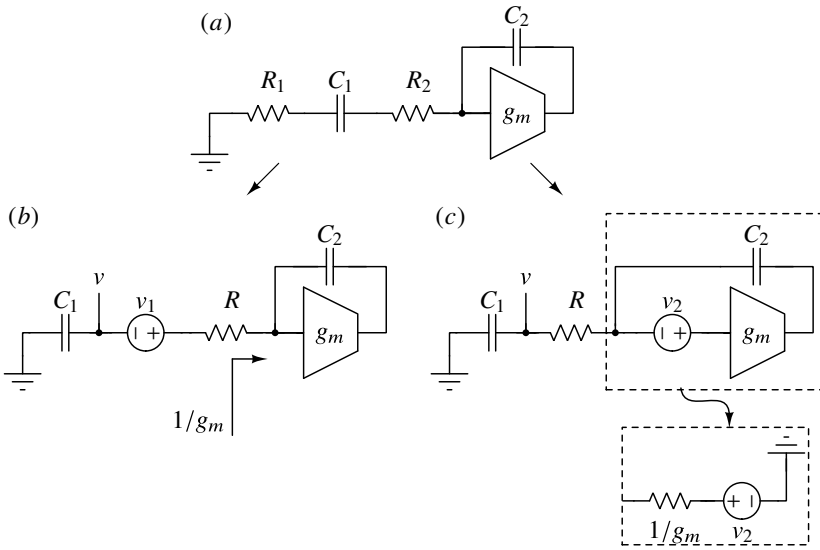
Figure 7.79   Analysis of phase-2 noise.

voltage source whose PSD is $S_{v_2} = 4kT\gamma_{\text{amp}}/g_m$. Again the circuit collapses into a single resistor and capacitor driven by a voltage source. Thus, we can immediately write

$$\overline{v^2} = \frac{S_{v_2}}{4R_{\text{eq}}C_1} = \frac{kT\gamma_{\text{amp}}/g_m}{(R + 1/g_m)C_1}. \tag{7.56}$$

In contrast to (7.55), minimizing the amplifier noise requires $R \gg 1/g_m$.

Combining (7.55) and (7.56) gives the noise due to both the amplifier and the switches in the charge transfer phase:

$$\overline{v^2} = \left(\frac{kT}{C_1}\right)\left(1 + \frac{\gamma_{\text{amp}} - 1}{1 + g_m R}\right). \tag{7.57}$$

If $\gamma_{\text{amp}} = 1$, the mean-square noise in the charge-transfer phase is the same as in the charging phase, namely kT/C_1, and thus the total from both phases is $2kT/C_1$.

As a final analytical exercise, let's consider power consumption. If we make the definition

$$x \equiv g_m R, \tag{7.58}$$

then the total noise from phase 1 and phase 2 is

$$\overline{v^2} = \left(\frac{kT}{C_1}\right)\left(2 + \frac{\gamma_{\text{amp}} - 1}{1 + x}\right), \tag{7.59}$$

while the settling time-constant is (from Figure 7.37)

$$\tau = \frac{C_1}{g_m}(1 + x). \tag{7.60}$$

The ADC specifications place constraints on both $\overline{v^2}$ and τ, so we ask "What value of x minimizes power consumption subject to constraints on $\overline{v^2}$ and τ?" To simplify the prob-

lem, assume that the power needed to drive the switches is negligible. Thus, power consumption is only related to g_m, which by (7.60) is

$$g_m = \frac{C_1}{\tau}(1 + x).$$ (7.61)

Substituting the value of C_1 dictated by (7.59) gives

$$g_m = \left(\frac{kT}{\tau \overline{v^2}}\right)\left(2(1 + x) + \gamma_{\text{amp}} - 1\right).$$ (7.62)

Clearly, g_m, and hence the power consumption, is minimized if $x = 0$, i.e. if the switch resistance is a small fraction of $1/g_m$. In practice, the desire to minimize switch resistance will be balanced by the power consumption associated with driving large switches and thus, our first-cut target of $R = 0.1/g_m$, that is $x = 0.1$, provides a reasonable starting point.

7.19 Conclusions

The first half of this chapter made a first pass at the design of a single-bit second-order switched-capacitor $\Delta\Sigma$ ADC. Design considerations for the amplifiers, comparator, clock generator, and switches were discussed, and example circuits were given and validated via simulation. Those simulations indicated that the design should achieve $DR = 98\,\text{dB}$, $BW = 1\,\text{kHz}$, and $P = 40\,\mu\text{W}$, which corresponds to $FoM = 172\,\text{dB}$. The second half of this chapter described various architectures and circuit techniques that can be used in more demanding applications, including multi-bit quantization, high-order loop filters, double-sampling, and gain-boosting. This material was capped with a discussion of the split-steering and amplifier stacking techniques used by an ADC that achieved a record-setting $FoM = 185\,\text{dB}$ ($DR = 136\,\text{dB}$, $BW = 1\,\text{kHz}$, and $P = 13\,\text{mW}$). To close the chapter, we took a more detailed look at noise in switched-capacitor circuits. The analysis justified our kT/C noise estimates as well as our recommendation that switch resistance be a small fraction of $1/g_m$.

References

[1] Y. Yang, A. Chokhawala, M. Alexander, J. Melanson, and D. Hester, "A 114 dB 68 mW chopper-stabilized stereo multi-bit audio A/D converter," *ISSCC Digest of Technical Papers*, pp. 64–65, Feb. 2003.

[2] C. Enz and G. C. Temes, "Circuit techniques for reducing the effects of opamp imperfection," *Proceedings of the IEEE*, vol. 84, pp. 1584–1614, Nov. 1996.

[3] J. Montanaro, R. Witek, K. Anne, A. Black, E. Cooper, D. Dobberpuhl, P. Donahue, and T. Lee, "A 160-MHz 32-b 0.5-W CMOS RISC microprocessor," *IEEE Journal of Solid-State Circuits*, vol. 31, pp. 1703–1714, Nov.1996.

[4] P. J. A. Harpe, B. Busze, K. Philips, and H. de Groot, "A 0.47-1.6 mW 5-bit 0.5-1 GS/s time-interleaved SAR ADC for low-power UWB radios," *IEEE Journal of Solid-State Circuits*, vol. 47, pp. 1594–1602, July 2012.

[5] D. Stepanović and B. Nikolic, "A 2.8 GS/s 44.6 mW time-interleaved ADC achieving 50.9 dB SNDR and 3 dB effective resolution bandwidth of 1.5 GHz in 65-nm CMOS," *IEEE Journal of Solid-State Circuits*, vol. 48, pp. 971–982, April 2013.

[6] P. Harpe, E. Cantatore, and A. van Roermund, "A 10b/12b 40 kS/s SAR ADC with data-driven noise reduction achieving up to 10.1b ENOB at 2.2 fJ/conversion-step," *IEEE Journal of Solid-State Circuits*, vol. 48, pp. 3011–3018, Dec. 2013.

[7] L. Kull, T. Toifl, M. Schmatz, P. Francese, C. Menolfi, M. Braendli, M. Kossel, T. Morf, T. Andersen, and Y. Leblebici, "A 3.1 mW 8b 1.2 GS/s single-channel asynchronous SAR ADC with alternative comparators for enhanced speed in 32 nm digital SOI CMOS," *IEEE Journal of Solid-State Circuits*, vol. 48, pp. 3049–3058, Dec. 2013.

[8] C. H. Chan, Y. Zhu, S. W. Sin, U. Seng-Pan, and R. P. Martins, "A 5.5 mW 6-b 5GS/s 4-interleaved 3b/cycle SAR ADC in 65nm CMOS," *ISSCC Digest of Technical Papers*, pp. 1–3, Feb. 2015.

[9] R. Jewett, K. Poulton, K. Hsieh, and J. Doernberg, "A 12b 128 MSample/s ADC with 0.05 LSB DNL," *ISSCC Digest of Technical Papers*, pp. 138–139, Feb. 1997.

[10] A. M. Abo and P. R. Gray, "A 1.5-V, 10-bit, 14.3-MS/s CMOS pipeline analog-to-digital converter," *IEEE Journal of Solid-State Circuits*, vol. 34, pp. 599–606, May 1999.

[11] J. Brunsilius, E. Siragusa, S. Kosic, F. Murden, E. Yetis, B. Luu, J. Bray, P. Brown, and A. Barlow, "A 16b 80MS/s 100mW 77.6dB SNR CMOS pipeline ADC," *ISSCC Digest of Technical Papers*, pp. 186–188, Feb. 2011.

[12] D. Senderowicz, G. Nicollini, S. Pernici, A. Nagari, P. Confalonieri, and C. Dallavale, "Low-voltage double-sampled $\Sigma\Delta$ converters," *IEEE Journal of Solid-State Circuits*, vol. 32, pp. 1907–1919, Dec. 1997.

[13] K. Bult and G. J. G. M. Geelen, "A fast-settling CMOS opamp for SC circuits with 90-dB dc gain," *IEEE Journal of Solid-State Circuits*, vol. 25, pp. 1379–1384, December 1990.

[14] K. Nagaraj, T. R. Viswanathan, and K. Singhal, "Reduction of finite-gain effect in switched-capacitor filters," *Electronics Letters*, vol. 21, no. 15, pp. 644–645, July 1985.

[15] M. Steiner and N. Greer, "A 22.3 b 1 kHz 12.7 mW switched-capacitor $\Delta\Sigma$ modulator with stacked split-steering amplifiers," *ISSCC Digest of Technical Papers*, pp. 284–286, Feb. 2016.

CHAPTER 8

CONTINUOUS-TIME DELTA-SIGMA MODULATION

So far in this book, we have built intuition and gained an understanding of the basic concepts behind $\Delta\Sigma$ modulation. Specifically, given the bandwidth of the input signal and the desired SQNR, we now know how to choose an NTF and an OSR that achieve these specifications. Further, we understand the trade-offs behind various loop-filter topologies that could be used to realize the NTF. We have seen, in detail, the many ways in which the loop-filter could be implemented. Since the input to the modulator is an oversampled *sequence*, the loop-filter has to be realized using discrete-time circuitry. As we saw in the previous chapters, the basic building block of the loop-filter is the integrator, most often implemented using opamps, switches, and capacitors in a manner similar to that in Figure 4.22. Such switched-capacitor integrators have many benefits: their coefficients, governed by capacitor ratios, are robust over process/temperature variations and are insensitive to stray capacitances.

Unfortunately, however, the design of such integrators becomes increasingly challenging in low-voltage CMOS technologies due to several difficulties. Turning the switches on and off is problematic at low supply voltages. As we saw in Chapter 4, the output of the integrator needs to settle in a half-clock period, necessitating wideband opamps with good settling behaviour. Thus, attempting to realize $\Delta\Sigma$ modulators with large signal bandwidth (or equivalently, high clock rate) results in high power dissipation, or may simply not be feasible in a given process technology. What if we realized the loop-filter with continuous-time circuitry? To better understand this, we go back to our good friend MOD1, and attempt to realize the loop-filter in continuous-time (CT) [1, 2, 3].

8.1   CT-MOD1
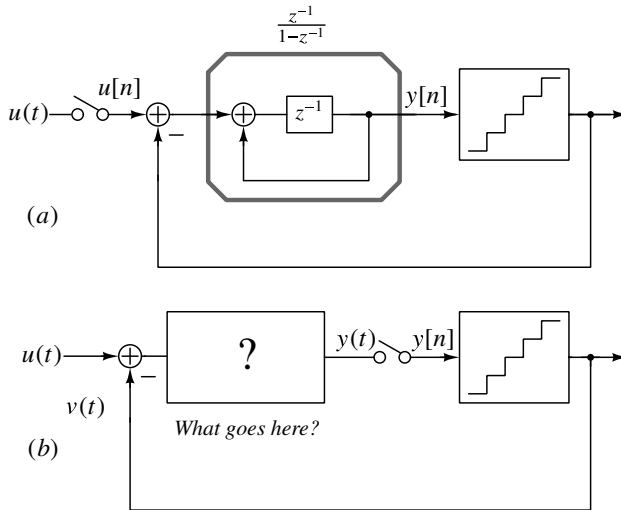
Figure 8.1   (a) MOD1 – u is sampled up-front, and the loop-filter processes the difference between the output sequence and $u[n]$. (b) The loop-filter output is sampled and quantized. The quantizer's output *waveform* is compared with the input waveform.

The difference between the discrete-time and continuous-time approaches to MOD1 is shown in Figure 8.1. In the former, the input is sampled up-front. The difference between the input *sequence* and the fed back *sequence* is processed by the DT loop-filter. The philosophy behind the CT realization is the following. The input is *not* sampled up-front; rather the quantizer's output *waveform* is subtracted from it, and processed by a continuous-time loop-filter. The filter's output is sampled, quantized, and fed back. As usual, the quantizer is realized as an ADC-DAC cascade. The ADC converts y to a digital output (which also forms the output of the modulator), while the DAC produces a continuous-time feedback waveform from the digital output code. What is the transfer function of the CT loop-filter in CT-MOD1, so that the NTF is $(1 - z^{-1})$? Before we get to this, we need to discuss a few aspects of DACs.

A DAC takes an input sequence and puts out a waveform that is related (linearly) to the sequence. Every DAC is associated with a pulse shape $p(t)$, allowing us to express the feedback waveform as

$$v(t) = \sum_n v[n]p(t - nT_s).\tag{8.1}$$

Some commonly used pulse shapes are listed below.

- NRZ DAC : $p(t) = 1$, $0 < t < T_s$.

- RZ DAC : $p(t) = 2$, $0 < t < 0.5T_s$.

- Impulsive DAC : $p(t) = \delta(t)$.

How is the spectrum of $v(t)$ related to that of $v[n]$? To determine this, we proceed as shown in Figure 8.2(a). We first form a continuous-time waveform $v_1(t)$ consisting of
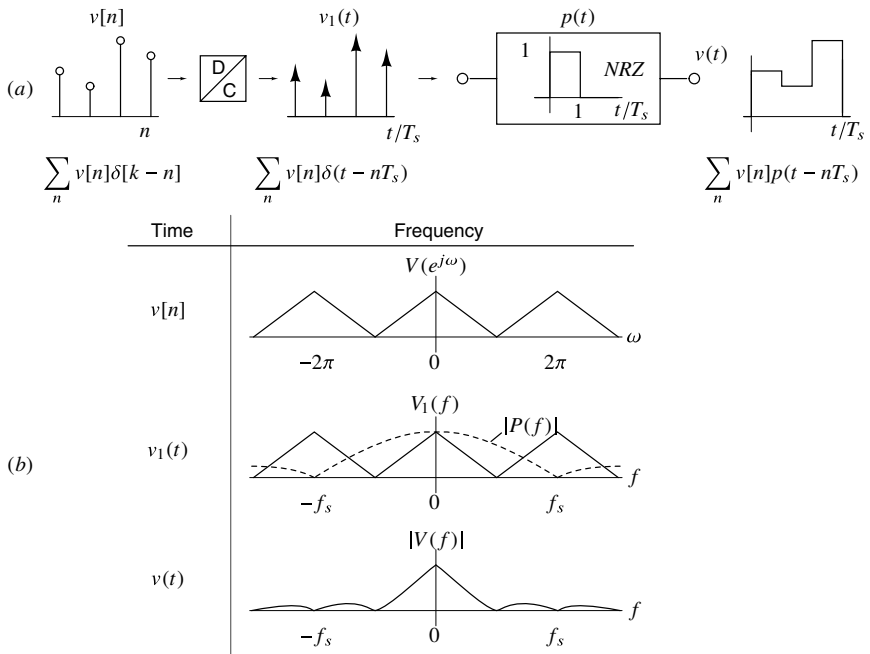
Figure 8.2 (a) Relating $v[n]$ to $v(t)$: the latter can be thought of as the output of a filter with impulse response $p(t)$, when excited by the Dirac impulse sequence $v_1(t)$. (b) Spectra of $v[n]$, $v_1(t)$ and $v(t)$.

Dirac impulses, related to $v[n]$ as

$$v_1(t) = \sum_n v[n]\delta(t - nT_s). \tag{8.2}$$

The Fourier transforms of $v[n]$ and $v_1(t)$ are denoted by $V(e^{j\omega})$ and $V_1(f)$, respectively. We then have

$$
\begin{aligned}
V(e^{j\omega}) &= \sum_n v[n]e^{-j\omega n}, \\
V_1(f) &= \sum_n v[n]e^{-j2\pi f T_s n}.
\end{aligned}
\tag{8.3}
$$

From the equations above, it is apparent that $V_1(f) = V(e^{j2\pi f T_s})$. The DAC output $v(t)$ can be thought of as the output of a linear time-invariant filter with impulse response $p(t)$, excited by $v_1(t)$. The Fourier transform $V(f)$ of the DAC's output waveform is thus given by

$$V(f) = P(f)V_1(f) = P(f)V(e^{j2\pi f T_s}). \tag{8.4}$$

The model of the quantizer (the ADC-DAC cascade) is shown in Figure 8.3. Sampling
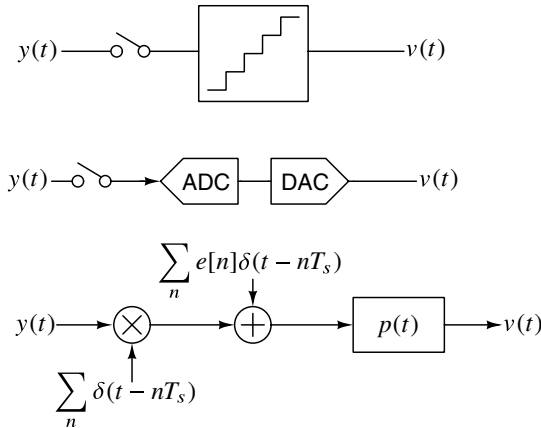
Figure 8.3   Quantizer model, appropriate to analyze a CTΔΣM.

is modeled by multiplying the input $y(t)$ with a Dirac impulse train. Quantization error is modeled by adding $e(t) = \sum_n e[n]\delta(t - nT_s)$. The DAC pulse is modeled by a continuous-time filter with impulse response $p(t)$. The resulting model for the continuous-time ΔΣ realization is shown in Figure 8.4(a). Part (b) of the figure is for the purist; it is usually drawn as shown in Figure 8.4(c).

What should we replace the discrete-time integrator of a DT-MOD1 with, so that the CT design of Fig 8.4(c) achieves the same NTF? We proceed by equating the loop impulse responses in both cases, as shown in Figure 8.5. For simplicity, we assume that the sampling period of CT-MOD1 is $T_s = 1$. We break the loop at the quantizer input in both cases. When the DT loop-filter is excited with an impulse, the resulting output sequence is

$$l_{dt}[n] = 0, 1, 1, 1, \cdots. \tag{8.5}$$
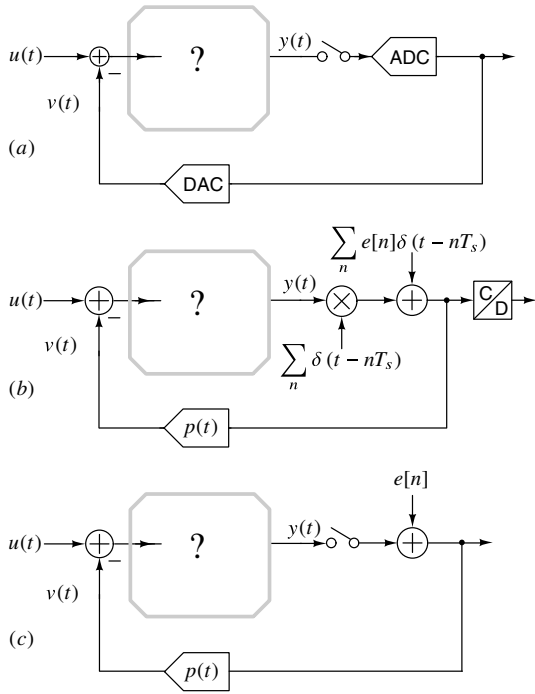
Figure 8.4   (a) A first-order CT $\Delta\Sigma$ modulator. (b) A purist's view: C/D denotes a continuous-time to discrete-time converter. (c) A commonly used depiction of the purist's model.
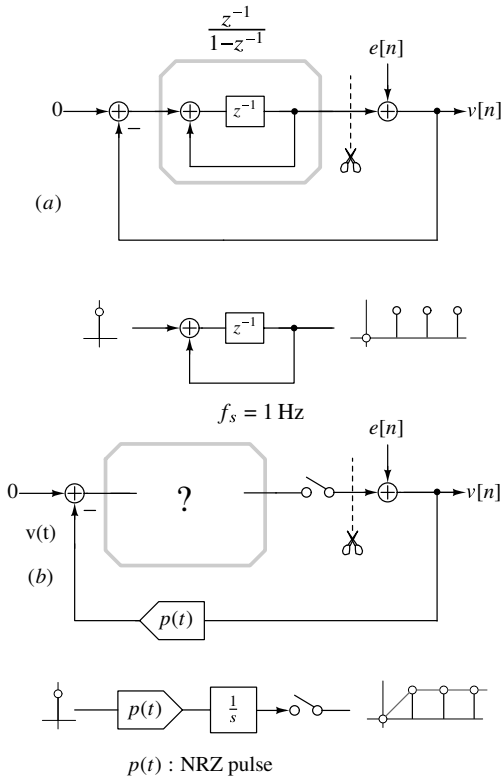
Figure 8.5   Equating loop impulse responses in DT- and CT-MOD1.

In the CT design, the corresponding sequence is

$$l_{ct}(n) = p(t) * l(t)|_{t=n}, \tag{8.6}$$

where $l(t)$ is the impulse response of the loop-filter, and $*$ denotes convolution. To realize an NTF of $(1 - z^{-1})$, $l(t)$ should be chosen so that $l_{ct}(n) = l_{dt}[n]$. If $p(t)$ is assumed to be an NRZ pulse, it is easy to see that using a continuous-time integrator achieves our objective (Figure 8.5(b)).

The resulting $\Delta\Sigma$ modulator, which is an avatar of MOD1 in continuous-time, is shown Figure 8.6(a); we call it CT-MOD1. How does the discussion above change if the sampling rate is increased to m Hz? To achieve the same NTF, the loop-filter must be frequency scaled by a factor m, as shown in Figure 8.6(b). It is always convenient (and advisable) to work with a normalized modulator (where $f_s = 1$ Hz) and frequency scale it at the very end.
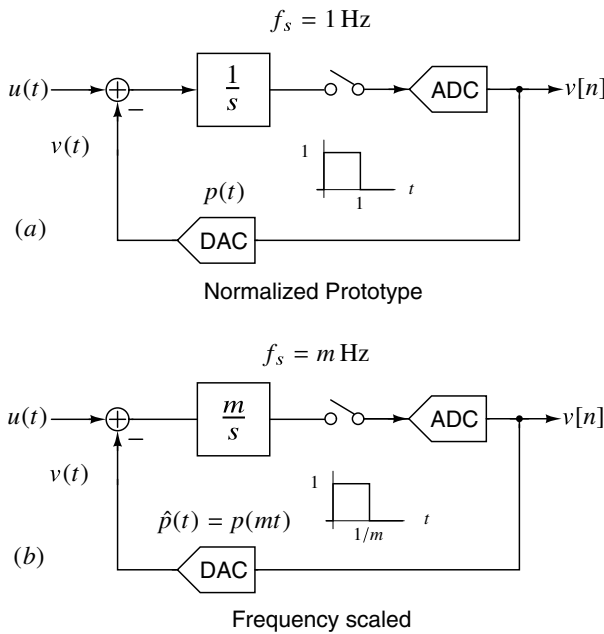
Figure 8.6   (a) CT-MOD1 and (b) frequency scaled so that $f_s = m$ Hz.

What happens to the NTF of CT-MOD1 when the DAC pulse is modified? As shown in Figure 8.7, using an RZ or impulsive DAC in place of an NRZ one modifies $l_{ct}(t)$ – however, the samples $l_{ct}(n)$ remain unchanged, indicating the NTF of CT-MOD1 is unaffected by the pulse shape, *as long as*

$$\int_0^1 p(t) \, dt = 1 \tag{8.7}$$

and $p(t) = 0, \ t > 1$.

To reiterate, the only feature of the DAC pulse shape that is relevant to the NTF of CT-MOD1 is the pulse's area. From the discussion above, we see that a continuous-time loop-filter can mimic the behavior of a discrete-time structure as far as the NTF is
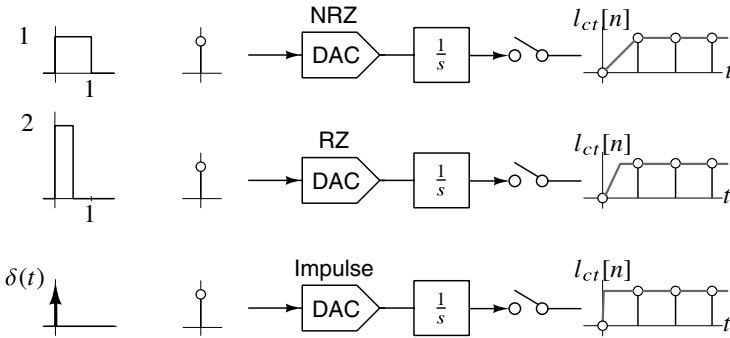
Figure 8.7 $l_{ct}(n)$ for different DAC pulse shapes.

concerned. How is the input (which is continuous-time) affected by the loop? The analysis of the STF is not quite as straightforward as in the discrete-time case, since $u(t)$ is CT, while the output $v[n]$ is a discrete-time sequence. We examine this next.

8.2 STF of CT-MOD1

When attempting to characterize a linear system, we excite it with a complex sinusoid $e^{j2\pi ft}$ and examine the output. We use the same strategy with CT-MOD1. The modulator is a feedback loop consisting of continuous-time and sampled parts, complicating analysis. To simplify matters, it is rearranged as shown in Figure 8.8. The idea behind this series of transformations is to separate CT-MOD1 into continuous-time and discrete-time parts as shown in part (c) of the figure. To determine the STF, we use $u(t) = e^{j2\pi ft}$. Thus,

$$y_1(t) = \frac{1}{j2\pi f} e^{j2\pi ft}, \tag{8.8}$$

$$y_1[n] = \frac{1}{j2\pi f} e^{j2\pi fn}. \tag{8.9}$$

The transfer function from $y_1[n]$ to $v[n]$, which is the same as that from $e[n]$ to $v[n]$, is simply the NTF of the loop, given by $(1 - z^{-1})$.

The output sequence due to $u = e^{j2\pi ft}$ is therefore

$$v[n] = \underbrace{\frac{1}{j2\pi f}}_{loop\text{-}filter} \underbrace{(1 - e^{-j2\pi f})}_{NTF} e^{j2\pi fn}. \tag{8.10}$$

The equation above can be interpreted as follows. $v[n]$ can be thought of as being obtained by exciting a continuous-time *linear time invariant (LTI)* filter with transfer function

$$STF(f) = \underbrace{\frac{1}{j2\pi f}}_{loop\text{-}filter} \underbrace{(1 - e^{-j2\pi f})}_{NTF} = e^{-j\pi f} sinc(f) \tag{8.11}$$

with u and sampling its output at 1 Hz, as shown in Figure 8.9. $STF(f)$ is called the signal
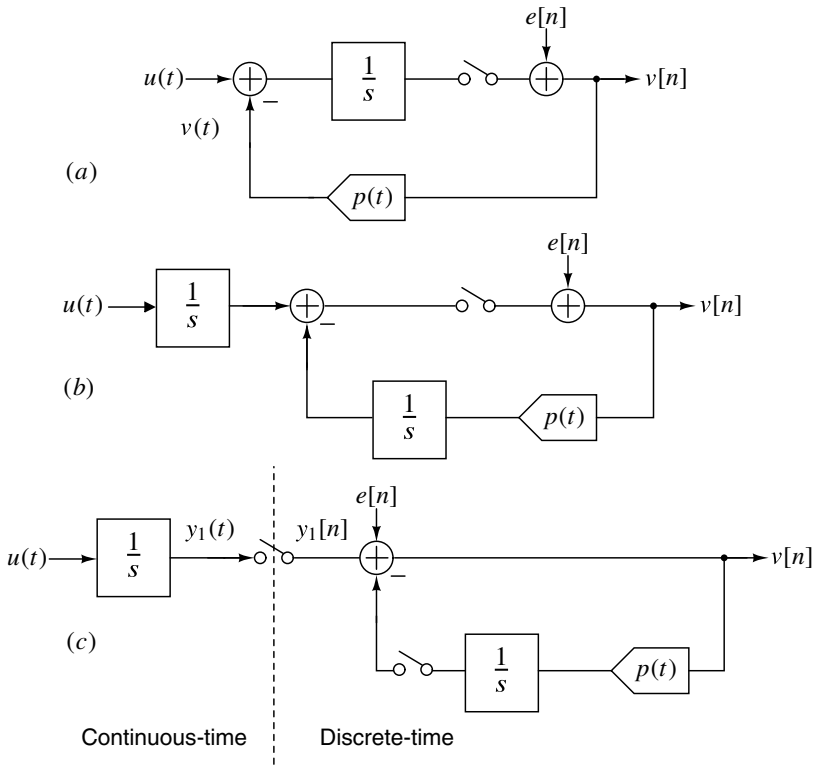
Figure 8.8 Steps in evaluating the STF (a) CT-MOD1, (b) moving the integrator up-front, and (c) separation into CT and DT parts.
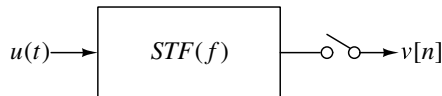
Figure 8.9 Interpretation of the STF of CT-MOD1 – the input is filtered by a continuous-time filter with frequency response $STF(f)$ before being sampled.
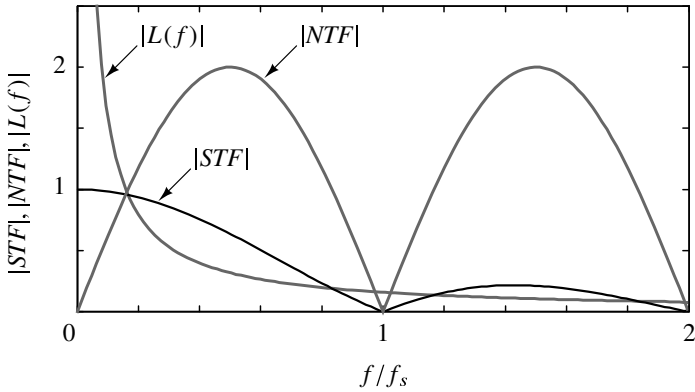
Figure 8.10   Magnitude responses of the loop-filter, NTF and the resulting STF.

transfer function of the CTΔΣM [1].

Figure 8.10 shows the magnitude responses of the loop-filter, the NTF and the STF. The STF's dc gain is seen to be unity (as expected); more interestingly, the STF has nulls at all nonzero integral multiples of the sampling frequency. This means that all tones that can potentially alias to dc after sampling are inherently eliminated by CT-MOD1. The response of $STF(f)$ at all the alias-zones is small, though not zero, as shown in Figure 8.11. Thus, CT-MOD1 possesses what we call the "inherent anti-aliasing" property, where the modulator doubles up as an anti-alias filter. This remarkable property of CT-MOD1 makes it unnecessary to use an explicit anti-alias filter up-front.
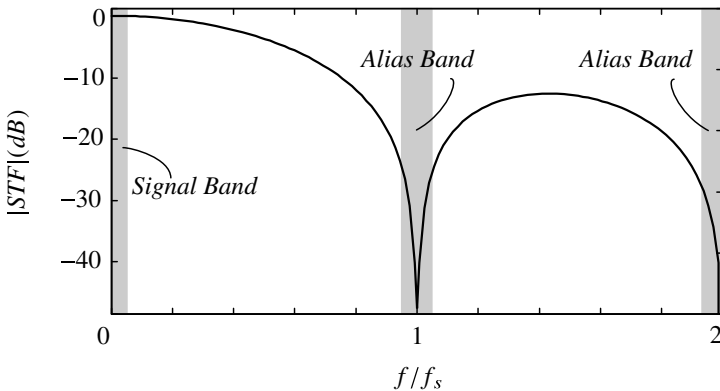
Figure 8.11   The STF of CT-MOD1 has nulls at integral multiples of the sampling frequency.

The model for CT-MOD1, assuming additive quantization noise, can be thought of as shown in Figure 8.12. The input is first filtered by a continuous-time filter with a transfer function $STF(f) = \exp(-j\pi f)sinc(f)$. It is then sampled, with the samples being corrupted by shaped quantization noise. In the time domain, the STF "filter" has a rectangular impulse response as shown in the figure.

From the analysis in this section, we see that u is prefiltered before sampling – this makes sense, since the sampling operation in CT-MOD1 occurs *after* the loop-filter.
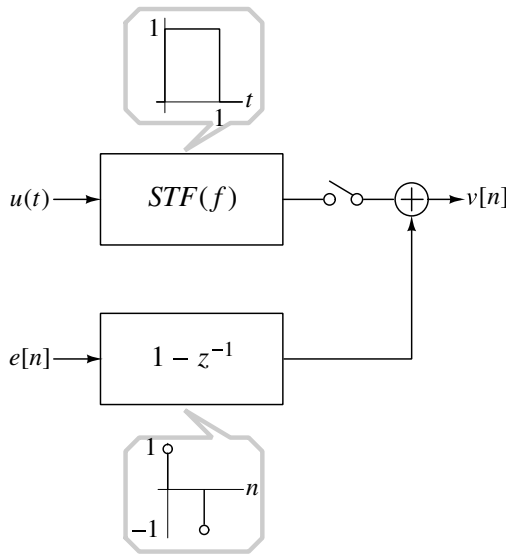
Figure 8.12   Model for CT-MOD1, assuming an additive quantization noise model.

Is there a more intuitive way to see that the STF of CT-MOD1 has notches at multiples of 1 Hz? As shown in Figure 8.13, the first step is to realize that the average value of the integrator's input $\overline{v_x(t)} = 0$. Since $u(t) = \cos(2\pi t)$, $\overline{u(t)} = 0$. This means that $\overline{v(t)}$, which is the average value of the feedback waveform, should be zero. This in turn implies that the $\overline{v[n]} = 0$. Referring to Figure 8.12, if $\overline{v[n]} = 0$ with $u = \cos(2\pi f_s t)$, it *must* mean that the amplitude of the sinusoid at the output of $STF(f)$ is zero. Why? A nonzero amplitude of the sinusoid at f_s, after sampling, would result in a nonzero $\overline{v[n]}$. Thus, we see that CT-MOD1 does not respond to tones at multiples of the loop's sampling frequency.
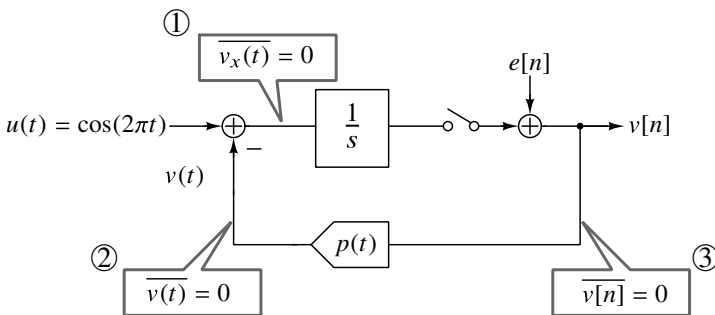
Figure 8.13   Intuitive explanation of zero dc output for an input at the sampling frequency.

8.2.1 Summary of CT-MOD1

By proper choice of the continuous-time loop-filter, the NTF of CT-MOD1 can be made equal to that of its discrete-time cousin. The loop-filter has to be chosen so that its impulse

response, when convolved by the DAC pulse shape $p(t)$ and sampled, matches the impulse response of the DT loop-filter. This is sometimes called *impulse invariance*. The NTF is independent of pulse shape, provided the area under $p(t)$ is 1, and that the pulse does not extend beyond 1 s. The STF has nulls at multiples of the sampling rate; CT-MOD1 therefore features the remarkable property of inherent anti-aliasing!

While deriving CT-MOD1 from its DT prototype, we simply replaced a discrete-time integrator with a continuous-time one, and doing so yielded the desired NTF for MOD1. Is this serendipity, or is there a fundamental principle lurking here?

The impulse response of a discrete-time system consists of a sum of complex exponential sequences of the form z_l^k, where z_l's denote its pole locations. The impulse response of a continuous-time system, on the other hand, is comprised of a sum of complex exponentials of the form $e^{s_l t}$, where s_l's are the system poles. Since we derived the CT loop-filter by matching the samples of its impulse response with that of the discrete-time loop-filter, it follows that $z_l^k = e^{s_l k}$. Equivalently,

$$s_l = \ln(z_l). \tag{8.12}$$

The loop-filter of MOD1 has a pole at $z_1 = 1$. The pole of CT-MOD1's loop-filter, therefore, should be located at $s_1 = \ln(1) = 0$.

The loop-filter of a high-order DT modulator with $NTF = (1 - z^{-1})^N / D(z)$ will have N poles at $z = 1$; the discussion above indicates that the loop-filter of a CT $\Delta\Sigma$ modulator that realizes the same NTF should have N integrators (N poles at $s = 0$).

8.3   Second-Order Continuous-Time Delta-Sigma Modulation
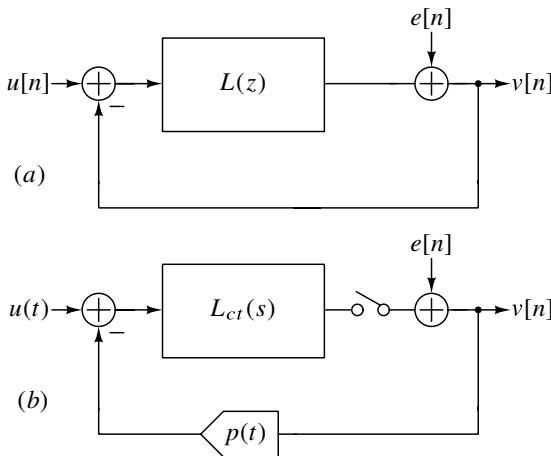
Figure 8.14   Replacing $L(z)$ in a CIFF MOD2 with a continuous-time loop-filter. $p(t)$ denotes the DAC pulse shape.

MOD2 was an attempt to improve the noise-shaping performance of MOD1, while CT-MOD1 was a continuous-time implementation of MOD1. The next logical destination

is CT-MOD2, where the loop-filter of MOD2 is implemented in continuous-time, as shown in Figure 8.14. MOD2 is assumed to be realized in CIFF form.
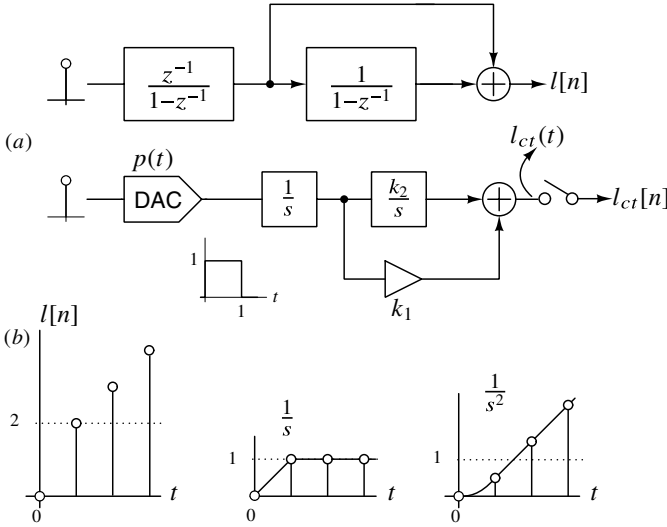
Figure 8.15   (a) The sampled pulse response of the CT loop-filter has to match $l[n]$. (b) $l[n]$ and the sampled pulse responses of the $1/s$ and $1/s^2$ paths.

The NTF of MOD2 is $(1 - z^{-1})^2$. Thus, $L(z)$ is given by

$$L(z) = \frac{1}{NTF(z)} - 1 = \frac{1}{z - 1} + \frac{z}{(z - 1)^2}. \tag{8.13}$$

The impulse response corresponding to $L(z)$ is

$$
\begin{aligned}
l[n] &= \begin{bmatrix} 0 & 1 & 1 & 1 & \cdots \end{bmatrix} \\
&+ \begin{bmatrix} 0 & 1 & 2 & 3 & \cdots \end{bmatrix} \\
&= \begin{bmatrix} 0 & 2 & 3 & 4 & \cdots \end{bmatrix}.
\end{aligned}
$$

The structure of the CT loop-filter is shown in Figure 8.15(a), and has a transfer function of the form

$$L_{ct}(s) = \frac{k_1 s + k_2}{s^2}. \tag{8.14}$$

Assuming that the $p(t)$ is an NRZ pulse, we can write the sampled pulse responses of the $1/s$ and $1/s^2$ paths as

$$
\frac{1}{s} \rightarrow \begin{bmatrix} 0 & 1 & 1 & 1 & \cdots \end{bmatrix},
$$

$$
\frac{1}{s^2} \rightarrow \begin{bmatrix} 0 & 0.5 & 1.5 & 2.5 & \cdots \end{bmatrix}.
$$

CT transfer function	z-transform of sampled pulse response
$1/s$	$1/(z-1)$
$1/s^2$	$0.5(z+1)/(z-1)^2$
$1/s^3$	$(1/6)(z^2+4z+1)/(z-1)^3$
$1/s^4$	$(1/24)(z^3+11z^2+11z+1)/(z-1)^4$

Table 8.1 z-transforms of sampled pulse responses of CT transfer functions of the form $1/s^l$, for an NRZ DAC pulse.

k_1 and k_2 that result in $l_{ct}(n) = l[n]$ can be determined by solving the following.

$$\begin{bmatrix} 0 & 0 \\ 1 & 0.5 \\ 1 & 1.5 \\ \vdots & \vdots \end{bmatrix} \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \\ 3 \\ \vdots \end{bmatrix}. \tag{8.15}$$

Clearly, the set of equations above is *overdetermined*, since there are more equations than unknowns. Yet, the solution is unique, and is given by

$$k_1 = 1.5 \text{ and } k_2 = 1.$$

Another way of arriving at the same result is to equate the weighted transforms of the sampled pulse responses of the $1/s$ and $1/s^2$ paths to $L(z)$.

Looking up the relevant transforms from Table 8.1, we obtain

$$\frac{k_1 z^{-1}}{1 - z^{-1}} + \frac{k_2(0.5z^{-1} + 0.5z^{-2})}{(1 - z^{-1})^2} = \frac{z^{-1}}{1 - z^{-1}} + \frac{z^{-1}}{(1 - z^{-1})^2}. \tag{8.16}$$

k_1 and k_2 can be determined by multiplying both sides of (8.16) by $(1 - z^{-1})^2$ and equating coefficients of like powers of z^{-1}.

The resulting modulator, CT-MOD2, is shown in Figure 8.16(a). To determine the STF, we proceed in the same manner as we did for CT-MOD1.

$$STF(f) = \underbrace{\left(\frac{1.5(j2\pi f) + 1}{(j2\pi f)^2} \right)}_{\text{loop-filter } L(s)} \underbrace{(1 - e^{-j2\pi f})^2}_{NTF} = (1 + 1.5(j2\pi f))e^{-j2\pi f} sinc^2(f). \tag{8.17}$$

The dc gain of the STF is unity. The STF is the product of $L(s)$ and the NTF evaluated at $e^{j2\pi f}$. Just like in CT-MOD1, the STF has nulls at multiples of the sampling frequency, resulting in implicit anti-aliasing, as shown in Figure 8.16(b). The STF has a zero at $s = -2/3$, due to the feedforward nature of the loop-filter. This causes the STF to asymptotically roll off as $1/f$ at high frequencies. The impulse response corresponding to $STF(f)$ is shown in Figure 8.16(c). It is a weighted combination of a triangular pulse (with height 1 and width of 2 seconds) and its first derivative.

It is important to observe that the loop-filter of CT-MOD2 does *not* result from simply replacing the discrete-time integrators of MOD2 with continuous-time ones; the $1/s$ and
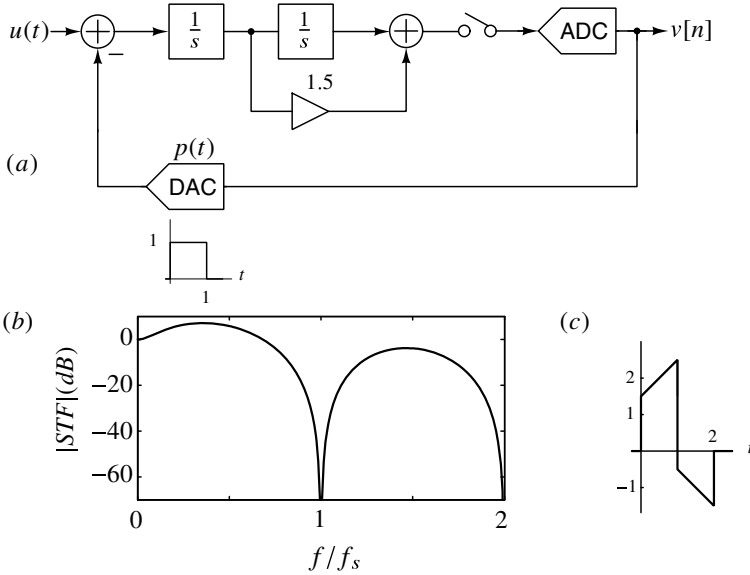
Figure 8.16 (a) CT-MOD2 for an NRZ feedback DAC. (b) The magnitude response of the STF, and (c) impulse response corresponding to $STF(f)$.

$1/s^2$ paths must be appropriately weighted, with these coefficients being dependent on the DAC pulse shape.

CT-MOD2 can also be realized as a CIFB structure, as shown in Figure 8.17(a). The STF in this case is given by

$$STF(f) = \underbrace{\frac{1}{(j2\pi f)^2}}_{loop\text{-}filter} \underbrace{(1 - e^{-j2\pi f})^2}_{NTF} = e^{-j2\pi f} sinc^2(f). \tag{8.18}$$

As in the case of a CIFF modulator, notches are seen in the STF at multiples of f_s, again placing in evidence (Figure 8.17(b)) the alias-rejection properties of a CT$\Delta\Sigma$M. Further, the magnitude response asymptotically rolls off as $1/f^2$ at high frequencies. The impulse response corresponding to $STF(f)$ is a triangular pulse as shown in the inset.

8.3.1 Influence of the DAC Pulse Shape

In our discussions of CT-MOD1, we found that its NTF was unaffected by the DAC pulse, as long as the pulse had an area of unity, and did not spill over beyond 1 s. What happens with CT-MOD2?

The pulse response of the loop-filter is the sum of the pulse responses of the $1/s$ and $1/s^2$ paths. The former, as we saw earlier, does not depend on pulse shape (provided that the area of $p(t)$ is 1, and its duration is less than 1 s). The response of the $1/s^2$ path is given by

$$l_2(t) = tu_1(t) * p(t) = \int_0^t p(\tau)(t - \tau)d\tau, \tag{8.19}$$

(a)

(b)
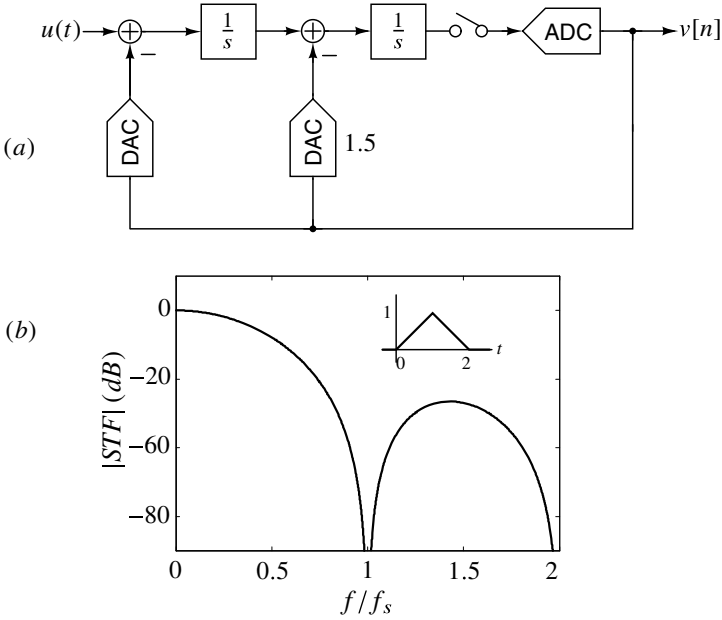
Figure 8.17   (a) CT-MOD2 with a CIFB loop-filter and (b) magnitude of the STF. The inset shows the impulse response corresponding to the $STF(f)$.

where $u_1(t)$ is the unit step function, and $*$ denotes convolution. Since the pulse duration is restricted to 1 s, $l_2(t)$ for $t \geq 1$ can be expressed as

$$l_2(t) = \int_0^1 p(\tau)(t - \tau)d\tau = t \underbrace{\int_0^1 p(\tau)d\tau}_{=1} - \int_0^1 \tau p(\tau)d\tau. \qquad (8.20)$$

Recalling that the average delay of $p(t)$ is given by

$$t_d = \frac{\displaystyle\int_0^1 \tau p(\tau)d\tau}{\underbrace{\displaystyle\int_0^1 p(\tau)d\tau}_{=1}}, \qquad (8.21)$$

we obtain

$$l_{ct}(n) = n - t_d , \ n \geq 1. \qquad (8.22)$$

It is thus apparent that the pulse response of the $1/s^2$ path depends only on two features of $p(t)$: namely its area and delay. Therefore, the NTF of CT-MOD2 will be independent of the details of the DAC pulse as long as the area *and delay* of the pulse remain the same.

Figure 8.18 shows $l_{ct}(t)$ for CT-MOD2 for three DAC pulses: NRZ, RZ, and the raised cosine pulse. The area and delay of all the pulses are 1 and 0.5, respectively. We see

that the waveforms, though different for $0 < t < 1$, are identical after the DAC pulse has died down. Consequently, $l_{ct}(n)$ and the NTFs remain the same for all these DAC pulses.
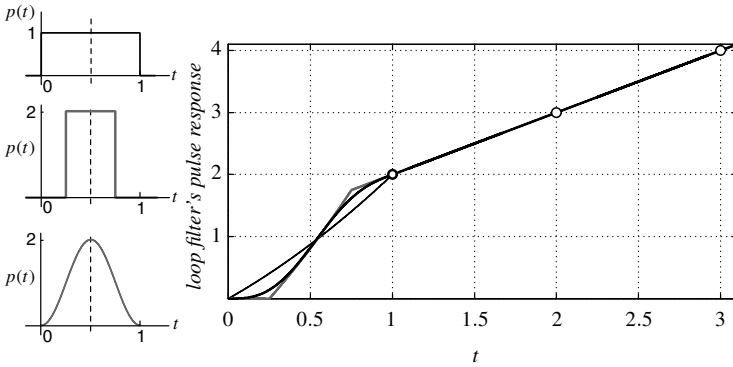
Figure 8.18 The pulse response of CT-MOD2's loop-filter is independent of the DAC pulse as long as the area and delay is maintained.

8.4 High-Order Continuous-Time Delta-Sigma Modulators

Having understood CT versions of MOD1 and MOD2, we proceed to the next logical destination – the design of high-order CT$\Delta\Sigma$M s. Figure 8.19(a) shows a discrete-time prototype, on which the continuous-time prototype is based. As usual, the loop-filter's transfer functions from u and v are denoted by $L_0(z)$ and $L_1(z)$, respectively.
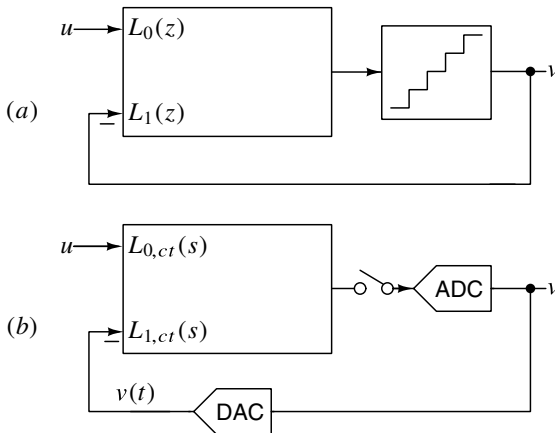
Figure 8.19 Block diagrams of (a) discrete-time and (b) continuous-time $\Delta\Sigma$ modulators.

The desired Nth order NTF is of the form $(1 - z^{-1})^N/D(z)$. The low-frequency gain of the modulator's STF is usually set to unity. Then, we see that

$$STF(z = 1) = 1 = \frac{L_0(z = 1)}{1 + L_1(z = 1)}, \tag{8.23}$$

which tells us that $L_0(z)$ and $L_1(z)$ must approach each other as $z \to 1$. Since the NTF has N zeros at $z = 1$, $L_1(z)$ must have N dc poles. For the STF to be 1 at dc, therefore, $L_0(z)$ must also have N dc poles. Recall that in our discussion of CT-MOD1, we reasoned that matching the pulse response of a CT loop-filter to that of a discrete-time one is only possible when its poles (s_l) of the former are related to those (z_l) of the latter as $s_l = \ln(z_l)$. From the arguments above, it is clear that $L_{0,ct}$ and $L_{1,ct}$ must both have N poles at $s = 0$, that is, must contain N integrators.
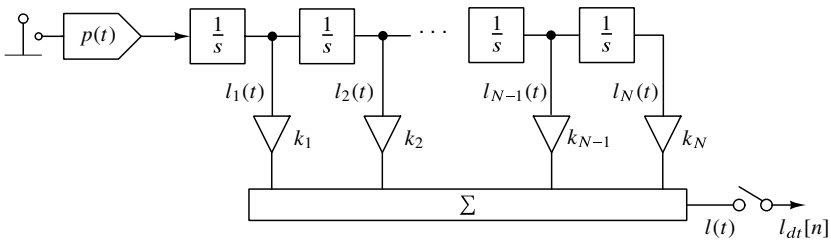
Figure 8.20   A possible realization of $L_{1,ct}$.

A possible realization of $L_{1,ct}(s)$ is shown in Figure 8.20. It can be expressed as a linear combination of paths of the form $1/s^i$, with $i = 1, \cdots, N$. The gain coefficients of these paths $k_1, \cdots, k_N$ must be chosen so that the filter's sampled output when driven by the DAC pulse $p(t)$ matches $l_{dt}[n]$, the impulse response corresponding to $L_1(z)$. A systematic way of determining coefficients is the following.

a. From the desired NTF, determine $L_1(z) = 1/NTF(z) - 1$.

b. Find $l_{dt}[n]$, the impulse response corresponding to $L_1(z)$.

c. Determine the pulse responses of the individual $1/s^i$ paths $x_i[n] = x_i(t)|_{t=n}$ for $i = 1, \cdots, N$.

d. Solve $\begin{bmatrix} x_1 & x_2 & \cdots & x_N \end{bmatrix} \begin{bmatrix} k_1 \\ k_2 \\ \vdots \\ k_N \end{bmatrix} = \begin{bmatrix} l_{dt} \end{bmatrix}$. The x_i's and l_{dt} are column vectors. The

system of equations is overdetermined; however, as in CT-MOD2, the equations admit a unique solution.

e. One way of realizing $L_{0,ct}$ is to add u to the DAC's output as shown in Figure 8.21. This satisfies the requirement that $L_{0,ct}$ must equal $L_{1,ct}$ as $s \to 0$. The loop-filter's output is sampled, quantized, and fed back through the DAC. Since the loop-filter is a cascade of integrators with feedforward, this corresponds to a CIFF CTΔΣM, where $L_{0,ct}(s) = L_{1,ct}(s)$.
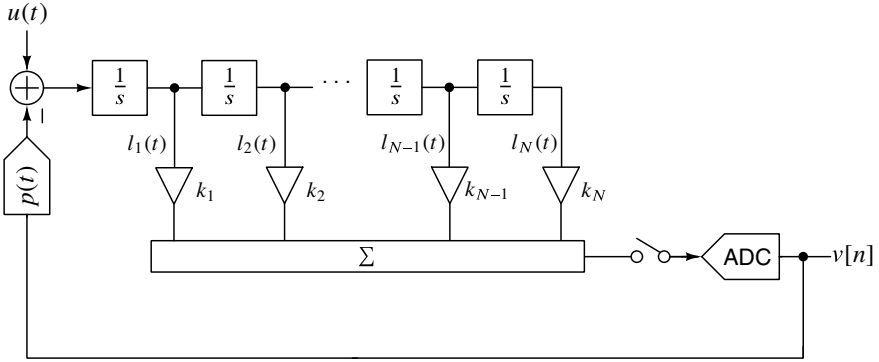
Figure 8.21   Completing the loop and adding u to realize an Nth order CIFF CT$\Delta\Sigma$M.

The STF of the modulator of Figure 8.21 can be found by using the same process we applied to CT-MOD1, and not surprisingly, this yields

$$STF(f) = L_{0,ct}(j2\pi f)NTF(e^{j2\pi f}).\tag{8.24}$$

The dc gain is unity, and thanks to the nulls of the NTF, the STF has excellent rejection around all integer multiples of the sampling rate that is commensurate with the in-band attenuation provided by the NTF.

8.4.1   Influence of DAC Pulse Shape [4]

Earlier in this chapter, we discussed the effect of DAC pulse shape on the NTFs of CT-MOD1 and CT-MOD2, assuming that the pulse died down after 1 s. We found that the former was agnostic to pulse shape as long as its area was unity, while the latter's NTF depended only on *two* features of the pulse shape – namely, its area and delay. What happens in an Nth order modulator?

For simplicity, we first consider a three-integrator chain, whose impulse response is $(t^2/2)u_0(t)$, where $u_0(t)$ denotes the unit step function. The output $x_3(t)$, for $t > 1$, obtained by convolving the impulse response with $p(t)$, is given by

$$x_3(t) = p(t) * \frac{t^2}{2}u_0(t) = \int_0^1 p(\tau)\frac{(t-\tau)^2}{2}u_0(t-\tau)d\tau \ , \quad t > 1.\tag{8.25}$$

This simplifies to

$$x_3(t) = \left[\int_0^1 p(\tau)d\tau\right]\frac{t^2}{2} - \left[\int_0^1 \tau p(\tau)d\tau\right]t + \frac{1}{2}\int_0^1 \tau^2 p(\tau)d\tau \ , \quad t > 1.$$

As seen above, the pulse response at the output of the chain is a polynomial in t, with coefficients dependent only on the details (or features) of $p(t)$. These are the *moments* of the pulse, defined as follows.

DAC type	Laplace transform	μ_0	μ_1	μ_2	μ_3
$\delta(t)$ Impulse	1	1	0	0	0
NRZ	$\frac{1-e^{-s}}{s}$	1	$\frac{1}{2}$	$\frac{1}{3}$	$\frac{1}{4}$
RZ	$2\frac{1-e^{-s/2}}{s}$	1	$\frac{1}{4}$	$\frac{1}{12}$	$\frac{1}{32}$
Exponential	$\frac{1}{1+s\tau_d}$	1	τ_d	$2\tau_d^2$	$6\tau_d^3$

Table 8.2   Moments of some commonly used pulses.

$$\underbrace{\mu_0}_{area} = \int_0^\infty p(\tau)d\tau,$$

$$\underbrace{\mu_1}_{\mu_0\cdot delay} = \int_0^\infty \tau p(\tau)d\tau,$$

$$\mu_2 = \int_0^\infty \tau^2 p(\tau)d\tau,$$

$$\vdots = \vdots$$

$$\mu_l = \int_0^\infty \tau^l p(\tau)d\tau.$$

Here μ_0 is the area (or "mass") of the pulse, μ_1/μ_0 is the average delay (or the "center of mass"), μ_2/μ_0 is the "moment of inertia", and so forth. Since the duration of $p(t)$ is 1 s, the upper limits of the integrals above can be replaced by 1. The moments of some commonly used DAC pulses are given in Table. 8.2.

Rewriting the equation above using the moments of $p(t)$, we obtain

$$x_3(t) = \frac{\mu_0}{2}t^2 - \mu_1 t + \frac{\mu_2}{2} , \quad t > 1. \tag{8.26}$$

From the discussion above, we observe that the output $x_3(t)$ (and therefore, the samples $x_3[n]$) are only dependent on three moments of $p(t)$, for $t \geq 1$.

In general, the pulse response of the $1/s^N$ path, denoted by $x_N(t)$, is given by

$$x_N(t) = \frac{t^{(N-1)}u_1(t)}{(N-1)!} * p(t) = \frac{1}{(N-1)!} \int_0^t p(\tau)(t-\tau)^{(N-1)}d\tau. \tag{8.27}$$

For $t \geq 1$, this simplifies to

$$x_N(t) = \frac{1}{(N-1)!} \int_0^1 p(\tau)(t-\tau)^{(N-1)} d\tau = \sum_{l=0}^{N-1} \frac{(-1)^l}{(N-1)!} \binom{N-1}{l} \mu_l t^{N-l-1}, \quad (8.28)$$

indicating that the sampled pulse response depends only on N features of $p(t)$, namely the $0, \cdots, (N-1)$ moments of the pulse.

Since the sampled pulse response of the Nth order loop-filter is given by

$$y[n] = \sum_{i=1}^{N} k_i x_i[n], \quad (8.29)$$

it must follow that the Nth order NTF is completely determined by the N moments of $p(t)$. In other words, the NTF remains the same, *even if the DAC pulse shape is modified*, as long as the $0, \cdots, (N-1)$ moments remain unchanged. What is the practical utility of this observation? To see this, we examine the following problem. Suppose that the continuous-time loop-filter transfer function that resulted in a desired NTF was known, for a given DAC pulse shape $p(t)$. What would the transfer function have to be to achieve the same NTF, if the pulse shape was modified to $q(t)$, as shown in Figure 8.22?
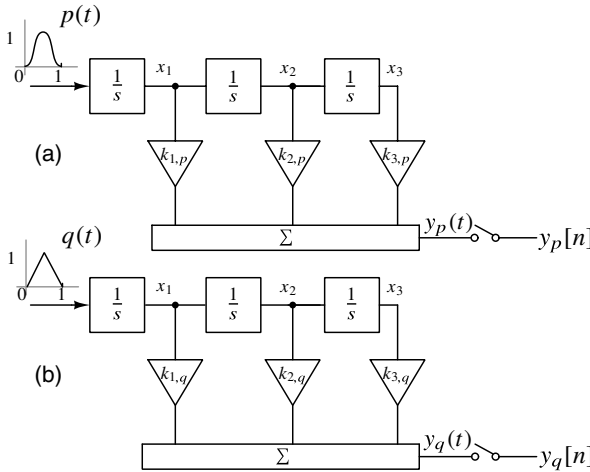
Figure 8.22   How should the coefficients of the loop-filters be related, so that both modulators have the same NTF?

This is illustrated with our third-order example below. In the discussion that follows, we denote the coefficients when the DAC pulse is $p(t)$ by $k_{1,p} \cdots k_{3,p}$, and the corresponding moments by $\mu_{0,p} \cdots \mu_{2,p}$. We have, for $t \geq 1$,

$$x_3(t) = \frac{\mu_{0,p}}{2} t^2 - \mu_{1,p} t + \frac{\mu_{2,p}}{2},$$

$$x_2(t) = \mu_{0,p} t - \mu_{1,p},$$

$$x_1(t) = \mu_{0,p}.$$

The loop-filter output is given by

$$y_p(t) = \frac{k_{3,p}\mu_{0,p}}{2}t^2 + (k_{2,p}\mu_{0,p} - k_{3,p}\mu_{1,p})t + \left(k_{1,p}\mu_{0,p} - k_{2,p}\mu_{1,p} + \frac{k_{3,p}\mu_{2,p}}{2}\right).$$

When the pulse is modified to $q(t)$,

$$y_q(t) = \frac{k_{3,q}\mu_{0,q}}{2}t^2 + (k_{2,q}\mu_{0,q} - k_{3,q}\mu_{1,q})t + \left(k_{1,q}\mu_{0,q} - k_{2,q}\mu_{1,q} + \frac{k_{3,q}\mu_{2,q}}{2}\right).$$

If the area of the pulses is chosen to be the same, so that $\mu_{0,p} = \mu_{0,q} = 1$ the equations above can be simplified as follows.

$$\begin{aligned}
y_p(t) &= \frac{k_{3,p}}{2}t^2 + (k_{2,p} - k_{3,p}\mu_{1,p})t + \left(k_{1,p} - k_{2,p}\mu_{1,p} + \frac{k_{3,p}\mu_{2,p}}{2}\right), \\
y_q(t) &= \frac{k_{3,q}}{2}t^2 + (k_{2,q} - k_{3,q}\mu_{1,q})t + \left(k_{1,q} - k_{2,q}\mu_{1,q} + \frac{k_{3,q}\mu_{2,q}}{2}\right).
\end{aligned}$$

If the NTFs have to be the same, $y_p(t) = y_q(t)$ for $t \geq 1$, resulting in

$$\begin{aligned}
k_{3,q} &= k_{3,p}, \\
k_{2,q} &= k_{2,p} + k_{3,p}\left(\mu_{1,q} - \mu_{1,p}\right), \\
k_{1,q} &= k_{1,p} + \left(\mu_{1,q} - \mu_{1,p}\right)(k_{2,p} + \mu_{1,q}k_{3,p}) - \frac{k_{3,p}}{2}\left(\mu_{2,q} - \mu_{2,p}\right). \quad (8.30)
\end{aligned}$$

If the zeroth, first, and second moments of $p(t)$ and $q(t)$ are equal, then the same coefficients can be used for both DAC pulses – in line with the discussion earlier in this section. In a practical Nth order NTF (where out-of-band gains are restricted to values much smaller than 2^N), it turns out that the coefficients of the $1/s^i$ paths for $i \geq 3$ are small. In the third-order example above, therefore, $k_{3,p} \ll k_{2,p}, k_{1,p}$. Then, from (8.30) we see that if $p(t)$ and $q(t)$ were chosen so that only their zeroth and first moments were the same, the NTFs should remain largely similar even if $\mu_{2,p} \neq \mu_{2,q}$.

> The observation above has the key implication that *the NTF of a practical high-order CTDSM is largely insensitive to the exact nature of the pulse shape, as long as the area and the delay ("center of mass") remain the same.*

Simulation results given below for a fourth-order modulator confirm the intuition gained using our analysis. Figures 8.23(a) and (b) show the discrete-time prototype and the CIFF implementation of the CTΔΣM, respectively. NTFs were determined for four DAC pulse shapes – the NRZ, delayed RZ, raised cosine, and the delayed impulse. All these pulses have the same area and average delay, resulting in $\mu_0 = 1$ and $\mu_1 = 0.5$. Maximally flat NTFs with out of band gains of 1.5 and 3, respectively, were used as examples. The out-of-band gains (OBG) represent the limits used in practice – the former for single-bit designs, and the latter being a typical upper limit for a multi-bit design. The coefficients of the discrete-time modulator and the corresponding CTΔΣMs (with an NRZ DAC) are given in Table 8.3.

CTΔΣM coefficients computed assuming an NRZ DAC were used to determine the NTF with *all the pulse shapes* of Figure 8.23. Figure 8.24 shows the NTFs, where we see that they are largely unchanged, even for seemingly drastic changes in the DAC pulse.
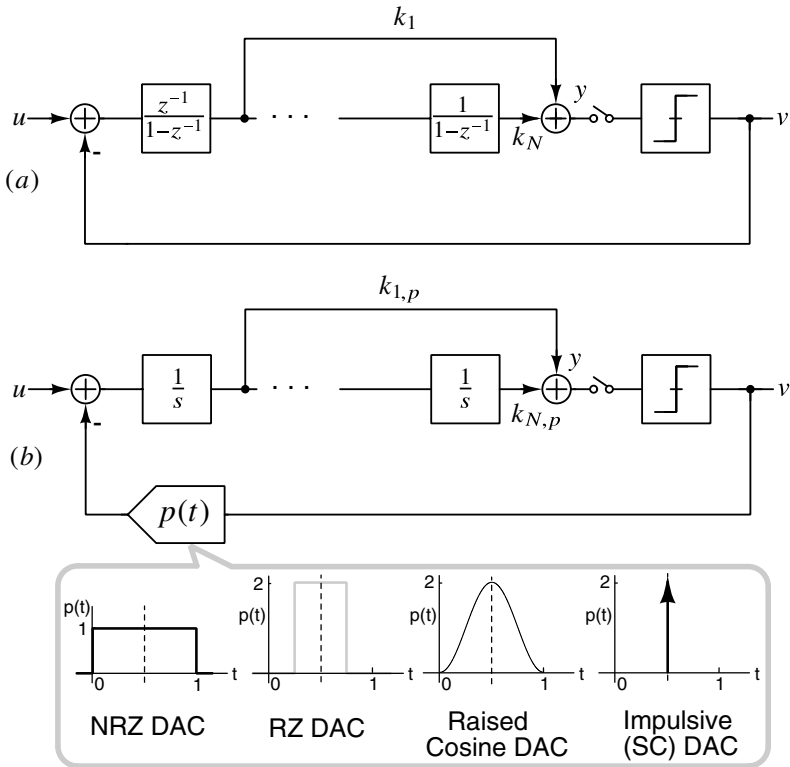
Figure 8.23   (a) The discrete-time prototype and (b) CT-$\Delta\Sigma$ modulators with various feedback DAC pulses – all pulses have the same area and delay.

	Pulse/OBG	k_1	k_2	k_3	k_4
DT (CRFF)	-/1.5	0.5556	0.2500	0.0524	0.0061
CT$\Delta\Sigma$M	NRZ/1.5	0.6713	0.2495	0.0555	0.0061
DT (CRFF)	-/3.0	1.1994	0.8890	0.5423	0.1584
CT$\Delta\Sigma$M	NRZ/3.0	1.3851	1.1862	0.6215	0.1584

Table 8.3   Fourth-order modulator coefficients for maximally-flat NTFs, with OBGs of 1.5 and 3.

The NTF magnitudes are virtually indiscernible when $OBG = 1.5$, and are only slightly different with $OBG = 3$. This makes sense due to the following. The magnitude of an Nth-order NTF at low frequencies is approximately ω^N/k_N, where k_N is the gain of the $1/s^N$ path of $L_{1,ct}$. Since a higher OBG means lower in-band gain for the NTF, k_N increases with increasing OBG, as confirmed by Table. 8.3. From (8.30), it is apparent that if $k_{1,q}$ was incorrectly chosen to be $k_{1,p}$, then this coefficient would be in error by a quantity proportional to $k_{4,p}$ (which increases with OBG). It therefore follows that the NTF would be less sensitive to the pulse shape when its out of band gain is small.
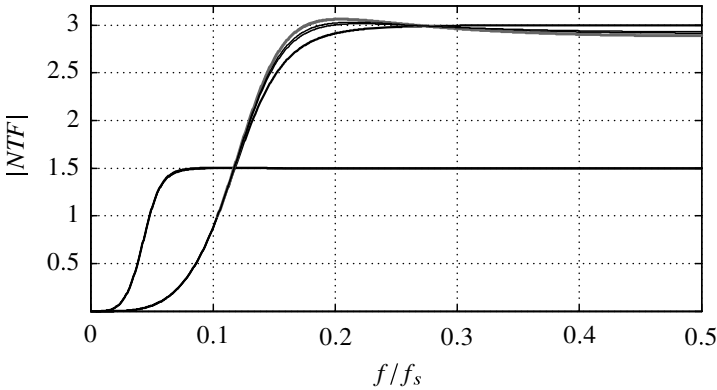
Figure 8.24   NTFs for fourth-order CTΔΣMs with $OBG = 1.5$ and $OBG = 3$ – the same coefficients (corresponding to those with an NRZ DAC) are used with all pulse shapes.

In this section, we discussed a systematic procedure to determine the transfer function of the loop-filter of a CTΔΣM, given a desired NTF. As can be expected from our experience with realizing CT-MOD1 and CT-MOD2, the loop-filter can be implemented in many ways, while still realizing the same NTF. We will consider some of these next.

8.5   Loop-Filter Topologies

8.5.1   The CIFB Family
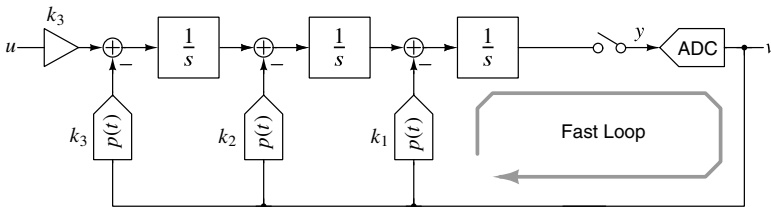
Figure 8.25   A third-order CTΔΣM realized as a CIFB structure.

We begin our discussion with the third-order CTΔΣM realized as a Cascade of Integrators with Feed Back (CIFB), shown in Figure 8.25. As in the discrete-time case, three

DACs are necessary. The "fast path" around the loop is through the innermost DAC, with coefficient k_1, while the "precise path" is through the DAC with weight k_3. The CIFB structure, therefore, decouples the fast and precise parts of the loop. This is a useful attribute, especially when clock rates are high. Inspection of Figure 8.25 reveals that

$$L_{0,ct}(s) = \frac{k_3}{s^3},$$

$$L_{1,ct}(s) = \frac{k_3}{s^3} + \frac{k_2}{s^2} + \frac{k_1}{s}.$$

The STF, given by $L_{0,ct}(j2\pi f)NTF(e^{j2\pi f})$, rolls of as $1/f^3$ at high frequencies. In applications (like wireless transceivers) where the modulator may be subject to input signals with significant out of band content, the inherent band limiting of the CIFB structure is an advantage, as it can potentially simplify the design of the filter that precedes it.

What are the disadvantages of the CIFB structure that motivate the search for other ways of realizing the loop-filter? As in the discrete-time case, the output of every integrator consists of the input component. The reasoning is the following. The low-frequency content of the input of every integrator must be very small. This means, unfortunately, that the output of every *preceding* integrator must contain a large input component, so as to nullify the input component injected by the feedback DAC. For instance, the only way for the dc content of the second integrator's input can be zero is if the dc output of the input integrator "balances" the dc output of the second DAC (whose weight is k_2). Consequently, the output of this integrator must consist of $k_2 \cdot u$ in addition to shaped quantization noise. In a similar fashion, the output of the second integrator must contain u with strength k_1. This is problematic on two counts. After *dynamic-range scaling*, whose motivation we discussed in detail with reference to the discrete-time case, the unity-gain frequency of the first integrator becomes small. Since a small unity-gain frequency means reduced gain in the signal band, the effects of noise and distortion further down the loop-filter are not adequately attenuated when referred to the modulator's input. The low unity-gain frequency also necessitates a large integrating capacitor in the input integrator, increasing the area occupied by the modulator.
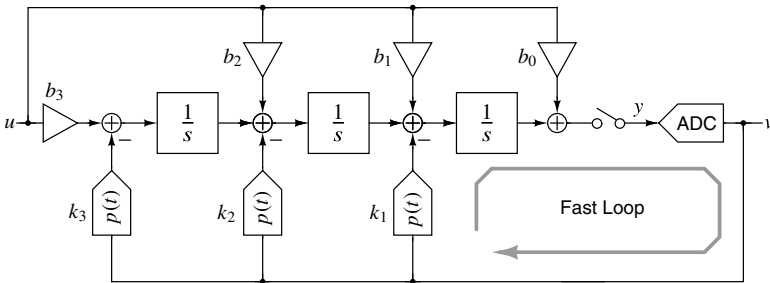
Figure 8.26 A CIFB structure with input feed-ins.

Recognizing the root cause of the CIFB loop's problems to be the input component injected by the feedback DACs at the outputs of the first and second integrators, it is apparent that assisting the integrators by adding input feed-ins should mitigate the problem. Figure 8.26 shows the CIFB CT$\Delta\Sigma$M with feed-ins. If the integrator outputs have to be devoid of the input component (assumed to be dc), $b_0 = 1$, $b_1 = k_1$ and $b_2 = k_2$. As the

input frequency increases, the "assistance" offered by the feed-in paths is not perfect, since v (and therefore the DAC feedback waveforms) consists of a phase shifted version of u. With feed-ins,

$$L_{0,ct}(s) = \frac{b_3}{s^3} + \frac{b_2}{s^2} + \frac{b_1}{s} + b_0. \tag{8.31}$$

The STF at high frequencies is now $b_0 NTF(e^{j2\pi f})$. The price to be paid to address the drawbacks of the CIFB structure is apparently the loss of the filtering nature of the STF.

8.5.2 The CIFF Family
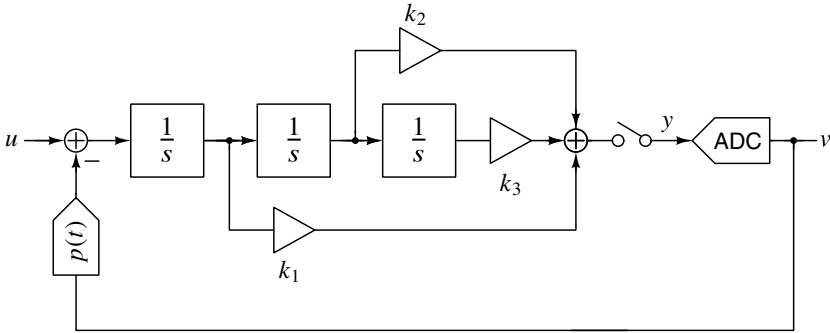
Figure 8.27   A third-order CT$\Delta\Sigma$M, with the loop-filter realized as a CIFF structure.

The loop-filter can also be realized as a cascade of integrators with feedforward (CIFF) structure, shown in Figure 8.27. Such a design needs only one feedback DAC. Further, the outputs of all integrators except the last are devoid of the input, which results in reduced output swings when compared with their CIFB counterparts. This, after dynamic-range scaling, translates into a high unity-gain frequency for the input integrator. This is beneficial, as nonidealities like noise and distortion added further down the loop are smaller when referred to the modulator's input. Faster integrators also means lower capacitor values in the loop-filter, saving area. There is a price to be paid for these benefits. Referring to Figure 8.27, we see that

$$L_{0,ct}(s) = L_{1,ct}(s) = \frac{k_3}{s^3} + \frac{k_2}{s^2} + \frac{k_1}{s}, \tag{8.32}$$

indicating that the STF can roll off only as $1/f$ at high frequencies. It is straightforward to see that for the same NTF, the STFs of the CT$\Delta\Sigma$Ms of Figures 8.25 and 8.27 are related as

$$STF_{CIFF}(s) = \left(1 + \frac{k_2}{k_3}s + \frac{k_1}{k_3}s^2\right) STF_{CIFB}(s). \tag{8.33}$$

This makes intuitive sense – adding feedforward paths introduces zeros in the transfer function. The STF of a CIFF modulator, therefore, peaks outside the signal band, which can be problematic in wireless applications. A further disadvantage, stemming from the fact that a CIFF design needs only one DAC, is that the "fast" and "precise" paths of the $\Delta\Sigma$ loop have the input integrator and DAC as constituents. This could be troublesome in high-speed designs: the necessity of having to close the loop favors simple circuitry

(single-stage amplifiers and small DAC delays), which is at odds with ways of achieving high linearity (multi-stage, high-gain amplifiers and special techniques to linearize the feedback DACs).

8.5.3 The CIFF-B Family

A particularly useful topology, that combines the benefits of the CIFF and CIFB loops is the CIFF-B structure, named (rather unimaginatively) after its parents.
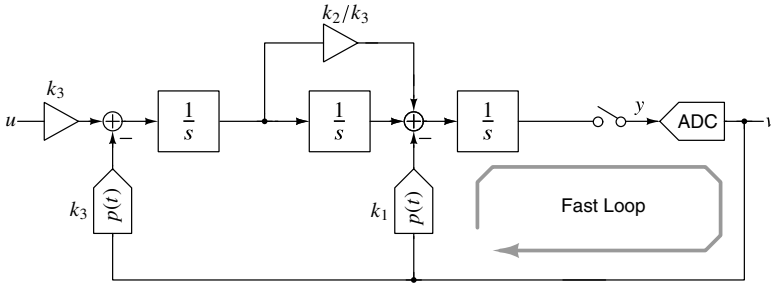
Figure 8.28   The CIFF-B loop-filter structure.

For this structure (Figure 8.28), we see that

$$L_{0,ct}(s) = \frac{k_3}{s^3} + \frac{k_2}{s^2},$$

(8.34)

leading us to conclude that the STF rolls of as $1/f^2$ at high frequencies. The STF is not quite as good a filter as in the CIFB case, but is not as "peaky" as that associated with a CIFF CT$\Delta\Sigma$M. As in the CIFB case, the fast and precise loops are decoupled, which is beneficial in high-speed designs. Due to the feed-forward path, the low-frequency swing at the output of the first integrator is small. This means that, after dynamic range scaling, the unity-gain frequency of the input integrator is larger than in the CIFB structure. This results in reduced distortion and noise from the rest of the loop-filter when referred to the input, as in a CIFF CT$\Delta\Sigma$M.

8.6 Continuous-Time Delta-Sigma Modulators with Complex NTF Zeros

While discussing the properties of NTFs in Chapter 4, we found that spreading the zeros of the NTF across the signal band (rather than bunching them all at dc) improves the in-band SQNR. The optimal locations of the zeros were obtained by minimizing the in-band noise with respect to those zeros. The optimal NTF zeros, being on the unit circle, are of the form $z_k = e^{j\theta_k}$. The poles of the continuous-time loop-filter, based on our reasoning in Section 8.1, are thus located at

$$p_k = \ln(z_k) = j\theta_k.$$

(8.35)

Since complex NTF zeros appear in conjugate pairs, it follows that the continuous-time loop-filter must have conjugate poles on the imaginary axis of the s-plane; implemented as a resonator by adding a negative feedback loop around two integrators.
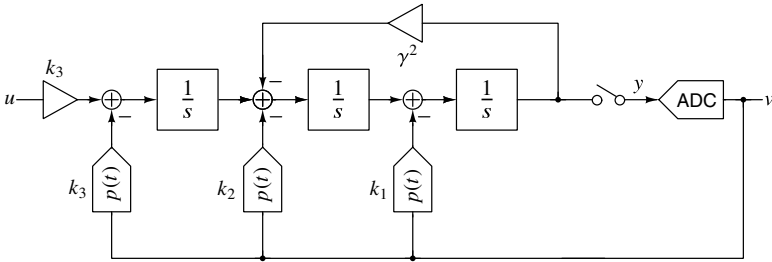
Figure 8.29   A third-order CRFB CT$\Delta\Sigma$M . The feedback through γ^2 implements the complex NTF zeros.

Figure 8.29 shows a third-order CT$\Delta\Sigma$M based on multiple feedback paths, implementing complex NTF zeros. Such loop-filters are called Cascade of Resonators with Feed Back (CRFB) structures. CRFF and CRFF-B modulators can be derived in an analogous fashion.

8.7   Modeling of Continuous-Time Delta-Sigma Modulators for Simulation

Taking a cue from discrete-time modulators, it would appear that a state-space description of the loop-filter is the most apt way of representing a CT$\Delta\Sigma$M for simulation. This is true, but there is more to this than meets the eye. We illustrate this with the second-order example shown in Figure 8.30. The outputs of the two integrators are the states, denoted by x_1 and x_2. From the figure, we have
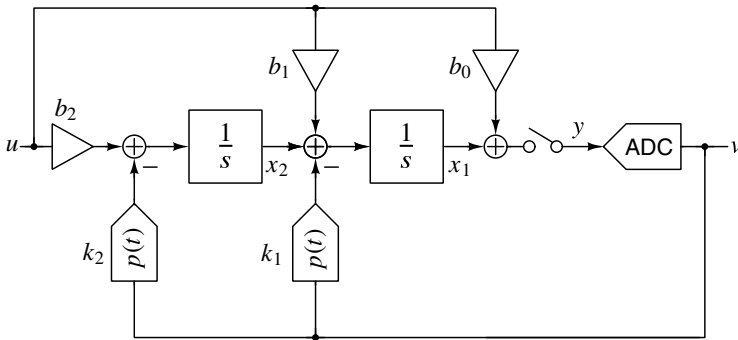
Figure 8.30   Example second-order CT$\Delta\Sigma$M illustrating state-space representation.

$$
\begin{aligned}
\dot{x}_1 &= x_2 + b_1 u - k_1 v, \\
\dot{x}_2 &= b_2 u - k_2 v, \\
y &= x_1 + b_0 u.
\end{aligned}
$$

In matrix form,

$$
\underbrace{\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix}}_{\text{derivative of state}} = \underbrace{\begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}}_{A_c} \underbrace{\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}_{\text{present state}} + \underbrace{\begin{bmatrix} b_1 & -k_1 \\ b_2 & -k_2 \end{bmatrix}}_{B_c} \underbrace{\begin{bmatrix} u \\ v \end{bmatrix}}_{\text{inputs}}
$$

$$
y = \underbrace{\begin{bmatrix} 1 & 0 \end{bmatrix}}_{C_c} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \underbrace{\begin{bmatrix} b_0 & 0 \end{bmatrix}}_{D_c} \begin{bmatrix} u \\ v \end{bmatrix}.
$$

The dimensions of the matrices for an Nth order modulator are the following.

$$
A_c : N \times N , \quad B_c : N \times 2 , \quad C_c : 1 \times N , \quad D_c : 1 \times 2.
$$

How does one simulate a loop that operates both in the continuous-time and sampled domains? One approach is to realize that rather than entire waveform $y(t)$, the output v depends on its *sampled* version $y[n]$. If the input is slowly varying, and assuming an NRZ feedback DAC pulse, the operation of the continuous-time loop-filter can be discretized as illustrated in Figure 8.31. Part (a) of the figure shows a slowly varying $u(t)$ and its zero-order-held (ZOH) approximation. The latter can be expressed as

$$
\hat{u}(t) = \sum_n u[n]p(t - n), \tag{8.36}
$$

where $p(t)$ is an NRZ pulse, and $u[n]$ is the sequence obtained by sampling $u(t)$ at 1 Hz (recall that the sampling rate of the modulator is also 1 Hz). $\hat{u}(t)$ can be thought of as the output of a filter with impulse response $p(t)$, excited by $u[n]$. Referring to Figure 8.31(b), we see that the continuous-time loop-filter is excited by two sequences $u[n]$ and $v[n]$, and samples of its output $y(t)$ are of interest. The system enclosed in the box is evidently a linear one, with two discrete-time sequences as inputs, and an output sequence $y[n]$. In principle, therefore, it can be replaced by a discrete-time system, whose state matrices are denoted by A_d, B_d, C_d and D_d. Given the state matrices of the continuous-time loop-filter, determining the DT state representation is straightforward, as we see below. The continuous-time filter is governed by

$$
\dot{x}(t) = A_c x(t) + B_c \begin{bmatrix} u(t) \\ v(t) \end{bmatrix}, \tag{8.37}
$$

$$
y(t) = C_c x(t) + D_c \begin{bmatrix} u(t) \\ v(t) \end{bmatrix}. \tag{8.38}
$$

The natural response of the states is $e^{A_c t}$. $u(t)$ is approximated by its piecewise-constant cousin $\hat{u}(t)$, and $v(t)$ is piecewise-constant anyway due to the NRZ pulse. Then, the states at time $(n + 1)$ can be related to $x[n]$ and $u[n], v[n]$ in the following manner.

$$
x[n + 1] = e^{A_c} x[n] + \underbrace{\int_0^1 e^{A_c \tau} B_c \begin{bmatrix} u[n] \\ v[n] \end{bmatrix} d\tau}_{\text{convolution integral}},
$$

$$
= e^{A_c} x[n] + A_c^{-1}(e^{A_c} - I)B_c \begin{bmatrix} u[n] \\ v[n] \end{bmatrix}. \tag{8.39}
$$

The first term in the equations above represents the evolution of the states from n to $(n+1)$, while the second represents convolution of the piecewise-constant inputs with the impulse
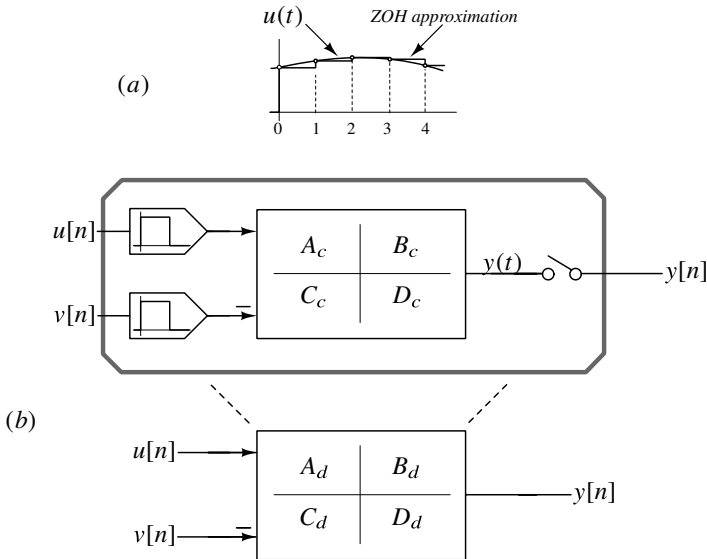
Figure 8.31   (a) A slowly varying $u(t)$ is approximately the same as its ZOH version $\hat{u}(t)$. (b) The continuous-time loop-filter and the two NRZ DACs can be replaced by its discrete-time equivalent.

responses from u and v to the states. I denotes an $(N \times N)$ identity matrix. The output at time n is given by

$$y[n] = C_c x[n] + D_c \begin{bmatrix} u[n] \\ v[n] \end{bmatrix}. \tag{8.40}$$

From (8.39) and (8.40), we see that the state matrices of the equivalent discrete-time system are given by

$$
\begin{aligned}
A_d &= e^{A_c}, \\
B_d &= A_c^{-1}(e^{A_c} - I)B_c, \\
C_d &= C_c, \\
D_d &= D_c.
\end{aligned}
\tag{8.41}
$$

 Since the CT$\Delta\Sigma$M has now been discretized, it can be simulated by the very same routines used for a discrete-time modulator.

How does the preceding discussion change when the pulse shape of the feedback DAC is modified? In that case, (8.39) can be rewritten as follows, where B_c is expressed as $[B_{c1} \;\; B_{c2}]$. B_{c1} and B_{c2} are the first and second columns of B_c, and affect the state transfer functions from u and v, respectively. $p_{dac}(t)$ denotes the DAC pulse shape and is assumed

to be 0 beyond $t = 1$.

$$
\begin{aligned}
x[n+1] &= e^{A_c}x[n] + \underbrace{\int_0^1 e^{A_c\tau}B_{c1}u[n]\,d\tau}_{\text{convolution integral}} + \underbrace{\int_0^1 e^{A_c\tau}B_{c2}p_{dac}(1-\tau)v[n]\,d\tau}_{\text{convolution integral}} \\
&= e^{A_c}x[n] + \underbrace{A_c^{-1}(e^{A_c} - I)B_{c1}}_{B_{d1}}u[n] + \underbrace{\int_0^1 e^{A_c\tau}B_{c2}p_{dac}(1-\tau)\,d\tau}_{B_{d2}}v[n].
\end{aligned}
$$

$$(8.42)$$

From the equations above, it is seen that incorporating arbitrary feedback DAC pulses is straightforward, where all that is needed is to modify B_d in (8.41) according to

$$
B_d = \begin{bmatrix} B_{d1} & B_{d2} \end{bmatrix},
\tag{8.43}
$$

where the first and second columns of B_d are given by [2, 5]:

$$
\begin{aligned}
B_{d1} &= A_c^{-1}(e^{A_c} - I)B_{c1}, \\
B_{d2} &= \int_0^1 e^{A_c\tau}B_{c2}p_{dac}(1-\tau)\,d\tau.
\end{aligned}
$$

8.8 Dynamic-Range Scaling
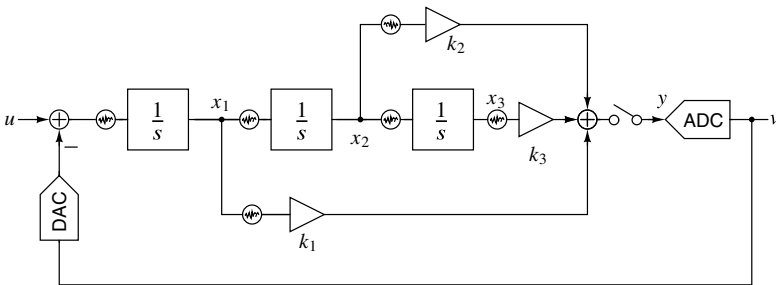
Figure 8.32   A third-order CIFF CTΔΣM: every integrator generates noise, modeled by an input-referred noise source.

Consider the third-order CIFF CTΔΣM of Figure 8.32, excited by a low-frequency input u, whose amplitude is chosen to be close to the MSA of the modulator. We assume that the number of quantizer levels is large. The loop-filter's output y consists of u and shaped noise. What observations can we make regarding the states x_1, x_2, and x_3? The low-frequency component of y must largely be the contribution of the third-order path of the loop-filter: thus, $k_3 x_3 \approx u$. Since $k_3 \ll 1$ (to ensure a large MSA in relation to full scale), it must follow that the peak swing of x_3 must greatly exceed the modulator's full scale. On the other hand, the peak-to-peak swing of x_1, which is the integrated version of shaped quantization noise, is bound to be much smaller than full scale. The reason is the

following. Since we assumed many quantizer levels, the peak-to-peak swing contributed by shaped noise at y (and v) is only a few levels. This must be the contribution of the fast $(1/s)$ path of the loop-filter. Since k_1 is of the order of unity, it follows that x_1's swing has to be small. In an ideal world (without noise, and with infinite supply voltages), the large differences in the peak-to-peak swings of the states is of no consequence.

However, we need to consider two practical realities. First, every integrator generates thermal noise, modeled as an input-referred noise source in Figure 8.32. Next, every integrator will saturate if its output attempts to exceed a certain threshold. This is a consequence of the limits enforced by the (finite) supply voltage. There are many choices of the loop-filter internal states that result in the same input–output transfer function. For instance, the CT$\Delta\Sigma$M of Figure 8.33 has the same NTF and STF as the design in Figure 8.32. However, the state $\hat{x}_1$ in the former is a scaled version (by a factor α) of the state x_1 in the latter. This is accomplished by increasing the gain of the first integrator by α, and reducing the gains of all blocks that sense $\hat{x}_1$ by the same factor. This way, the transfer function and the other states of the loop-filter remain unchanged.

Since α is arbitrary, a good question to ask is if there is method to choosing it. What happens, for instance, if α is too small? Referring to Figure 8.33, we see that this mandates a large gain k_1/α from $\hat{x}_1$ to y, thereby greatly amplifying the input-referred thermal noise of the gain element. This suggests that α should be chosen to be large to reduce thermal noise at the loop-filter's output.

Now, what if α was made too large? This is problematic too, since the integrator will saturate if its output attempts to increase beyond a certain limit (dictated by the supply voltage). When an integrator saturates, its output no longer responds to changes in its input, effectively cutting it out from the modulator. Given our experience with the deleterious effects of quantizer saturation, one should expect that a saturating integrator is most likely to destabilize the modulator. The conclusion from the discussion above is that one should always attempt to keep the state variables as large as possible in magnitude, while avoiding saturation. This way, thermal noise from the loop-filter is amplified to the least extent possible. Further, increasing the gains of the integrators increases their unity-gain frequency, and reduces the capacitance needed for their implementation. This results in a reduction in the active area occupied by the CT$\Delta\Sigma$M. Scaling the states in a manner that they are as large as possible (without being any larger!) is called *dynamic-range scaling*, and this should be an integral part of the design process.
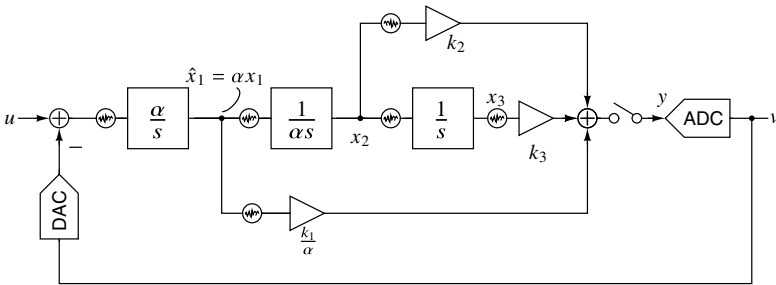
Figure 8.33   Scaling x_1 by α without affecting the loop-filter's transfer function.

As we saw above, dynamic-range scaling does not affect the input–output behavior of the filter. How does this reflect in its state-space description? The original state equations are

$$\dot{x}(t) = A_c x(t) + B_c \begin{bmatrix} u(t) \\ v(t) \end{bmatrix}, \tag{8.44}$$

$$y(t) = C_c x(t) + D_c \begin{bmatrix} u(t) \\ v(t) \end{bmatrix}. \tag{8.45}$$

We denote the scaled states by $\hat{x}$. Since each state can be scaled by a different factor, $\hat{x} = Tx$, where T is a diagonal transformation matrix. Thus, $x = T^{-1}\hat{x}$. Substituting this in the state equations above, we obtain

$$T^{-1}\dot{\hat{x}} = A_c T^{-1}\hat{x} + B_c \begin{bmatrix} u \\ v \end{bmatrix},$$

$$y = C_c T^{-1}\hat{x} + D_c \begin{bmatrix} u \\ v \end{bmatrix}.$$

The state matrices of the scaled loop-filter are thus seen to be

$$\hat{A}_c = TA_c T^{-1}, \ \hat{B}_c = TB_c, \ \hat{C}_c = C_c T^{-1}, \ \hat{D}_c = D_c. \tag{8.46}$$

8.9 Design Example

In this section, we illustrate the concepts we have discussed so far by attempting the design of a third-order CTΔΣM whose NTF is maximally flat, with $OBG = 2.5$. The modulator employs a 16-level quantizer and operates with $OSR = 64$. The signal bandwidth is 500 kHz. We assume that a CIFF loop-filter and an NRZ DAC are used. We make extensive use of the ΔΣ toolbox. We proceed in the following step-by-step fashion.

a. Determine the NTF.
   ```
   ntf = synthesizeNTF(3,64,0,2.5,0)
   ```
   which yields

$$NTF(z) = \frac{(z-1)^3}{(z-0.417)(z^2 - 0.8778z + 0.3804)}.$$

b. Next, we determine $L_1(z) = 1/NTF(z) - 1$.
   ```
   L1 = 1/ntf - 1
   ```
   which yields

$$L_1(z) = \frac{1.7052(z^2 - 1.322z + 0.4934)}{(z-1)^3}.$$

c. We then determine the impulse response l of the DT loop-filter
   ```
   l = impulse(L1,10);
   ```

d. $L_{1,ct}(s)$ is of the form $\frac{k_1}{s} + \frac{k_2}{s^2} + \frac{k_3}{s^3}$. We need to determine k_1, k_2, and k_3.

e. We first find the pulse response samples (x_1, x_2, x_3) of the $1/s$, $1/s^2$ and $1/s^3$ paths.
   ```
   x1 = impulse(c2d(tf([1],[1 0]),1),10);
   ```

```
x2 = impulse(c2d(tf([1],[1 0 0]),1),10) ;
x3 = impulse(c2d(tf([1],[1 0 0 0]),1),10);
```

f. Determine $\mathbf{K} = \begin{bmatrix} k_1 & k_2 & k_3 \end{bmatrix}^T$ by solving $\begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix} \mathbf{K} = l$

```
K = [x1 x2 x3]\l;
```

This yields $k_1 = 1.2244$, $k_2 = 0.8638$, $k_3 = 0.2930$.
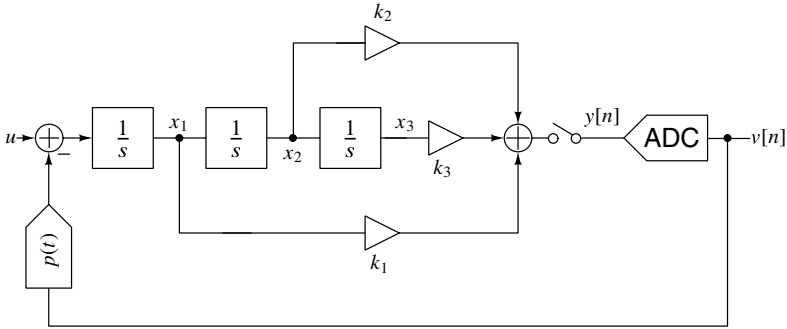
Figure 8.34   Third-order CTΔΣM. x_1, x_2, x_3 are the state variables.

g. The loop-filter is a CIFF design (Figure 8.34), and we describe it in state-space form as follows.

$$A_c = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, B_c = \begin{bmatrix} 1 & -1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, C_c = \begin{bmatrix} k_1 & k_2 & k_3 \end{bmatrix}, D_c = \begin{bmatrix} 0 & 0 \end{bmatrix}.$$

h. Next, we create the CT loop-filter
   ```
   sys_ct=ss(Ac,Bc,Cc,Dc);
   ```
   and determine the corresponding discrete-time loop-filter.
   ```
   sys_dt=c2d(sys_ct,1);
   ```
   This yields

$$A_d = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0.5 & 1 & 1 \end{bmatrix}, B_d = \begin{bmatrix} 1 & -1 \\ 0.5 & -0.5 \\ 0.1667 & -0.1667 \end{bmatrix}$$

$$C_d = \begin{bmatrix} 1.225 & 0.864 & 0.293 \end{bmatrix}, D_d = \begin{bmatrix} 0 & 0 \end{bmatrix}.$$

i. We then simulate the difference equations describing the modulator.
   The sinusoidal input is at one-fourth the signal bandwidth, with an amplitude of 0.8 times full scale.
   ```
   u = 0.8*15*sin(2*pi*(0.25/OSR)*(0:1:2^15));
   ABCD = [Ad Bd; Cd Dd];
   [v,xn,xmax,y] = simulateDSM(u,ABCD,16,zeros(3,1));
   ```
   simulateDSM yields v, the final states xn, their maxima $xmax$, and the sampled loop-filter output y.

j. The power spectral density (PSD) of v is shown in Figure 8.35.
   ```
   psd(v,Nfft, fs, hanning(Nfft,'periodic'));
   ```
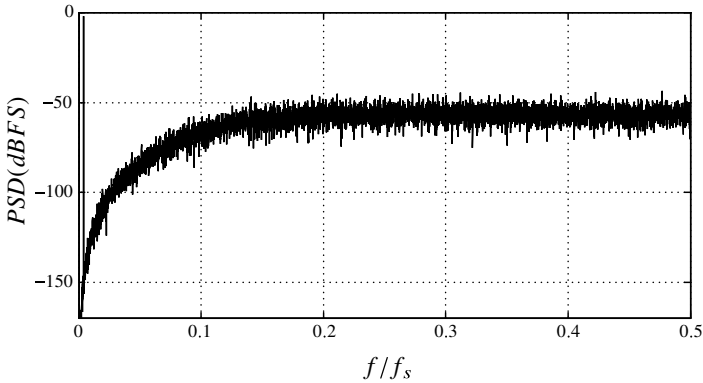
Figure 8.35   Power spectral density of the output sequence.

The quantization noise seems shaped, as we expect. But how do we know that the NTF is indeed the one that we set out to realize? The problem with inferring the NTF (shape and OBG) by eyeballing the PSD is its noisy nature, arising due to the noise-like properties of the quantization error. In simulation, however, this error is explicitly available, since we have access to both v and y. A useful "trick" to eliminate noise in the PSD (during simulations), and thereby verify the NTF, therefore, is to divide $PSD(v)$ by $PSD(v - y)$. This should yield $|NTF(e^{j\omega})|^2$ without noise, as we see next.

k. Verify the NTF
```
[P1,f]= psd(v,Nfft, fs, hanning(Nfft,'periodic'));
[P2,f]= psd((v-y),Nfft, fs, hanning(Nfft,'periodic'));
plot(f,10*log10(P1./P2);
```
The resulting NTF, plotted on a dB scale, is shown in Figure 8.36. Determining $|NTF(e^{j\omega})|^2$ as $PSD(v)/PSD(v - y)$ clearly places in evidence the nature of the NTF, and allows one to accurately determine OBG.
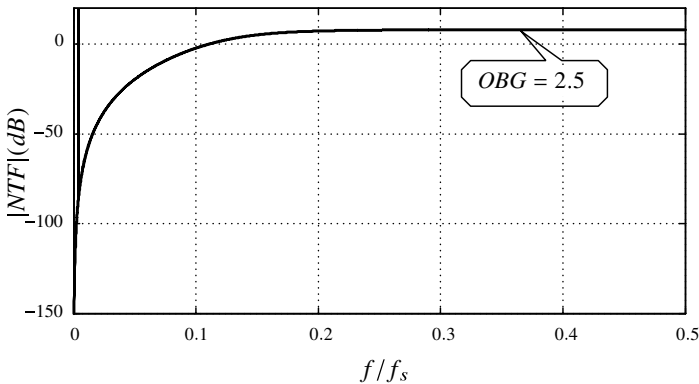
Figure 8.36   Determining the NTF by computing $PSD(v)/PSD(v - y)$.

l. Finally, we scale for dynamic range and frequency.
  simulateDSM also yields the maxima of the states, which turn out to be $x_{1,max} = 2.605$, $x_{2,max} = 2.905$ and $x_{3,max} = 43.32$. (Recall that the DAC output can go

from -15 to 15). Assuming that we would like to limit the magnitude of the states to 10, 12, and 14, respectively, it is straightforward to see that the integrator unity-gain frequencies should be scaled by $\alpha = 4.6$, $\beta = 0.89$, and $\gamma = 0.067$, respectively. Further, the integrator outputs should now be weighted by $\hat{k}_1 = 0.26$, $\hat{k}_2 = 0.21$, and $\hat{k}_3 = 1.05$, as shown in Figure 8.37. Finally, to operate with a sampling frequency f_s, the bandwidths of all integrators are multiplied by f_s.
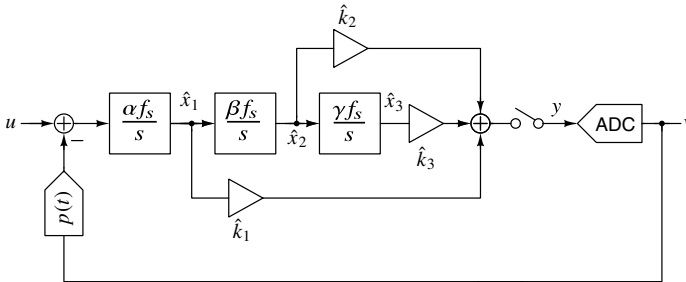
Figure 8.37   The third-order CIFF modulator, after dynamic range and frequency scaling.

8.10 Conclusions

In this chapter, we discussed the basic ideas and attributes of continuous-time delta-sigma modulation. The philosophy behind a CT$\Delta\Sigma$M is to emulate the behavior of the loop-filter in a discrete-time $\Delta\Sigma$ converter using continuous-time circuitry. This is accomplished by choosing the CT loop-filter to be impulse-invariant with respect to the DT prototype. By virtue of sampling occuring inside the loop, CT$\Delta\Sigma$Ms feature implicit anti-aliasing. As with DT $\Delta\Sigma$ converters, many choices exist for the realization of the loop-filter – each with its associated trade-offs. Finally, we saw how a CT$\Delta\Sigma$M can be simulated by discretizing the continuous-time loop-filter, so as to leverage the tools already designed to simulate discrete-time converters.

References

[1] J. C. Candy, "A use of double integration in sigma delta modulation," *IEEE Transactions on Communications*, vol. 33, no. 3, pp. 249–258, 1985.

[2] R. Schreier and B. Zhang, "Delta-sigma modulators employing continuous-time circuitry," *IEEE Transactions on Circuits and Systems: Fundamental Theory and Applications*, vol. 43, no. 4, pp. 324–332, 1996.

[3] J. A. Cherry, *Theory, Practice, and Fundamental Performance Limits of High Speed Data Conversion Using Continuous-Time Delta-Sigma Modulators*. Ph.D. dissertation, Carleton University, 1998.

[4] S. Pavan, "Continuous-time delta-sigma modulator design using the method of moments," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, no. 6, pp. 1629–1637, 2014.

[5] S. R. Norsworthy, R. Schreier, and G. Temes, *Delta-Sigma Data Converters: Theory, Design, and Simulation*. IEEE Press, New York, 1997.

CHAPTER 9

NONIDEALITIES IN CONTINUOUS-TIME DELTA-SIGMA MODULATORS

In the previous chapter, we understood the fundamental principles of continuous-time $\Delta\Sigma$ modulation. In particular, we learned how one could design a continuous-time loop-filter that results in a desired NTF. Unfortunately, many of the assumptions we made regarding the operation of the modulator are not true in practice. For example, no real quantizer can make a decision instantaneously – there must be a delay associated with it. The ADC thresholds and DAC levels will deviate from their desired values due to element mismatch. Integrators in the loop-filter are not ideal either. In the most optimistic case, component inaccuracy shifts the unity-gain frequencies. More realistically, the integrators have finite dc gain, and their transfer functions have parasitic poles and zeros. Since they are built with transistors, the integrators are also somewhat nonlinear.

Finally, any practical sampling clock will be jittery. While one might argue that clock jitter is "not the CT$\Delta\Sigma$'s problem," it turns out that the choice of modulator architecture has a dramatic impact on how it responds to jitter. This, therefore, warrants a detailed study. In this chapter, we examine the primary nonidealities in a CT$\Delta\Sigma$M – namely excess delay, time-constant variations, and clock jitter, and discuss how to address these problems.

9.1 Excess Loop Delay

So far in this book, we have considered the quantizer as having zero delay, so that the quantized output is available at the same instant at which the input is sampled. In practice,

t_d > ADC delay + DEM logic delay + DAC setup time
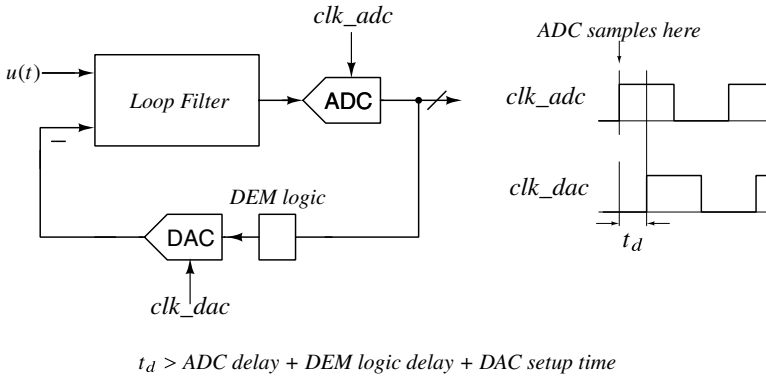
Figure 9.1   The excess delay problem in continuous-time $\Delta\Sigma$ modulators.

however, there is a delay due to the following. As seen in Figure 9.1, the quantizer is implemented as a cascade of an ADC followed by a DAC. The former samples the loop-filter's output on the rising edge of its clock, denoted by clk_adc. As described in Section 7.9, a nonzero time is needed to resolve the analog input, and the output sequence of the ADC is only available after a delay. The DAC, which converts the ADC output sequence back into a waveform, can therefore only be clocked at a later time, as shown in the figure. As we saw in Chapter 6, it is common to insert digitial circuitry (called dynamic element matching (DEM) logic)) between the ADC and DAC to shape mismatch-induced noise out of the signal band. The delay t_d of the DAC clock with respect to the ADC clock, therefore, should be large enough to accommodate the delay of the ADC and DEM plus the DAC setup time. In most cases, the DAC needs to be explicitly clocked to prevent the variable nature of the regeneration and propagation delays from introducing unwanted jitter.

Before getting into a detailed analysis of the effects of excess loop delay, let us ponder what delay will do. Like in any feedback loop, adding delay is bound to degrade modulator stability. We should also expect that high-order loops fare worse than loops with low order. Finally, since the magnitude of the loop-gain does not change with delay, we expect that quantization noise suppression within the signal band should not be affected (provided, of course, that the modulator remains stable).

9.1.1   CT-MOD1 : The First-Order Continuous-Time Delta-Sigma Modulator

Consider the normalized first-order CT$\Delta\Sigma$M, with an excess delay t_d, as shown in Figure 9.2. An NRZ DAC is assumed. The pulse response of the loop-filter is obtained by breaking the loop and sampling the output of the integrator. It is seen to be

$$
\begin{aligned}
l[n] &= \{0, 1 - t_d, 1, 1, \cdots\} \\
&= \underbrace{\{0, 1, 1, 1, \cdots\}}_{ideal\ response} - \underbrace{\{0, t_d, 0, 0, \cdots\}}_{error}.
\end{aligned}
\tag{9.1}
$$
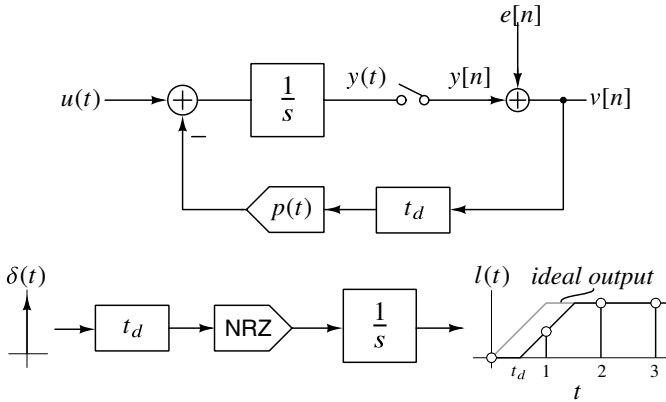
Figure 9.2 CT-MOD1 with delay t_d.

The loop-gain $L(z) = \mathcal{Z}\{l[n]\}$ and NTF are given by

$$L(z) = \frac{z^{-1}}{1 - z^{-1}} - t_d z^{-1},$$

$$NTF(z) = \frac{1}{1 + L(z)} = \frac{1 - z^{-1}}{1 - t_d z^{-1} + t_d z^{-2}}.$$

From these expressions, it is apparent that the order of the system has now increased to two, with the pole locations depending on t_d. Analysis shows that the modulator poles move toward the unit circle as delay increases, and lie on the unit circle when $t_d = 1$, as shown in Figure 9.3. The onset of instability as delay increases is hardly surprising. Additionally, we see that in the signal band, the magnitude of the NTF remains ω (independent of t_d), as we intuitively expected.
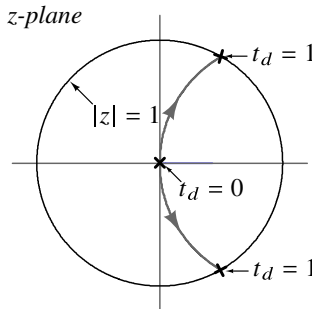
Figure 9.3 Locus of pole locations of a first-order CTΔΣM as a function of excess loop delay (t_d).

While analyzing the influence of delay on CT-MOD1 is informative, a more fruitful exercise is to understand how one can mitigate the effects of excess delay. From (9.1), we see that adding a path whose sampled pulse response is $\{0, t_d, 0, 0, 0 \cdots\}$ in parallel with the integrator can restore the loop's NTF. One way of realizing the parallel path is to simply have a gain t_d as shown in Figure 9.4. From our experience with circuits, we realize that this is not so surprising after all – a feedforward path adds a "zero" to the loop-gain

function, thereby improving phase margin, and stabilizing the system. It is important to note that only the samples of the compensated loop-filter are equal to ideal samples (i.e., without delay) – this is not true of the continuous-time waveform at the loop-filter's output. Incorporating the direct path into the modulator leads to the systems shown in Figure 9.5.
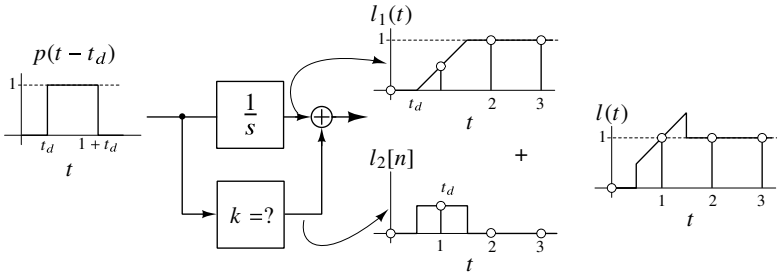
Figure 9.4   Using a direct path across the integrator to mitigate excess delay.

The specific implementation shown in part (b) of the figure is commonly used – with the shaded portion being referred to as the "direct feedback path around the quantizer". Note that since the direct feedback is sampled, the DAC that implements this path does not need to be clocked.
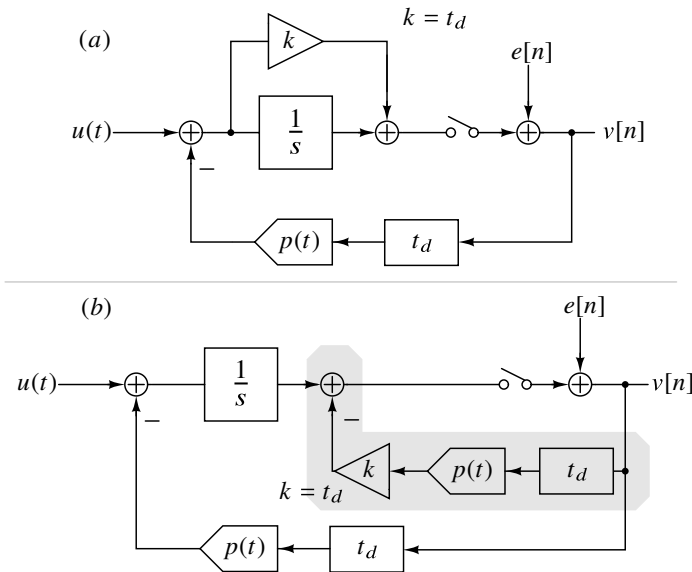
Figure 9.5   Alternative implementations of the direct path in CT-MOD1.

The shape of the DAC pulse influences the sampled response of the loop-filter. Other commonly used pulses are the return-to-zero (RZ) and impulsive shapes. From Figure 9.6, it is seen that the RZ DAC can tolerate half-clock cycle delay, while the impulsive DAC can tolerate almost a whole-clock cycle.
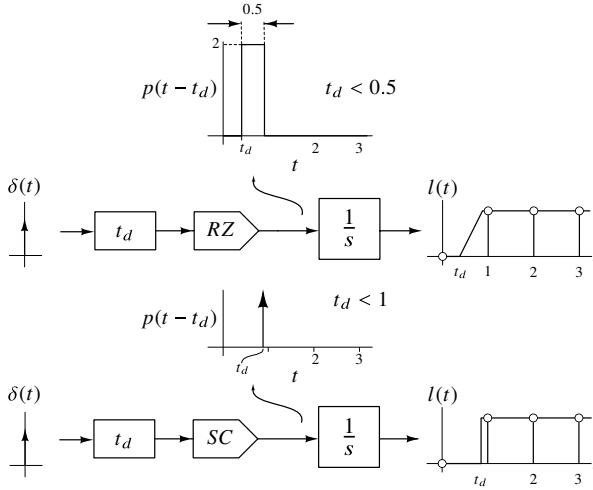
Figure 9.6   Insensitivity to excess delay using other pulse shapes (a) RZ and (b) impulse.

9.1.2 CT-MOD2 : The Second-Order Continuous-Time Delta-Sigma Modulator
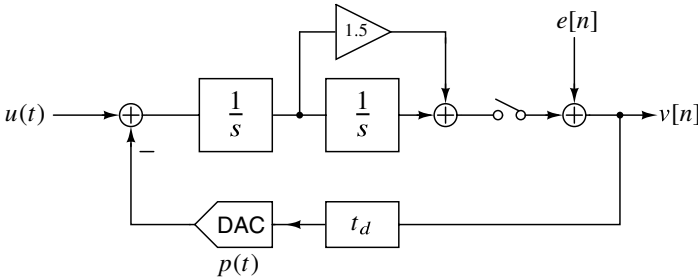
Figure 9.7   A second-order CTΔΣM with excess delay.

Next, we analyze the effect of excess delay in a second-order modulator with an NRZ DAC, shown in Figure 9.7. Ideally, $t_d = 0$, and $NTF(z) = (1 - z^{-1})^2$, which corresponds to a loop-gain function $L(z) = (2z - 1)/(z - 1)^2$. The transfer function of the loop-filter is

$$L_c(s) = \frac{1.5}{s} + \frac{1}{s^2}. \tag{9.2}$$

With excess delay, the z-transforms of the sampled pulse responses of the $1/s$ and $1/s^2$ paths are given by

$$\frac{1}{s} \quad \rightarrow \quad \frac{1 - t_d}{z - 1} + z^{-1}\frac{t_d}{z - 1}, \tag{9.3}$$

$$\frac{1}{s^2} \quad \rightarrow \quad \frac{(0.5 - t_d + 0.5t_d^2)z + 0.5(1 - t_d^2)}{(z - 1)^2} + z^{-1}\frac{t_d(1 - 0.5t_d)z + 0.5t_d^2}{(z - 1)^2}. \tag{9.4}$$

$L(z)$ and the NTF can be determined as functions of t_d from the equations above. As in the first-order case, excess delay causes the order of the system to increase by 1. The locus of the roots as t_d changes, shown in Figure 9.8, indicates that the modulator is unstable for delays greater than 30% of the clock period, with the MSA reducing dramatically as t_d approaches 0.3.
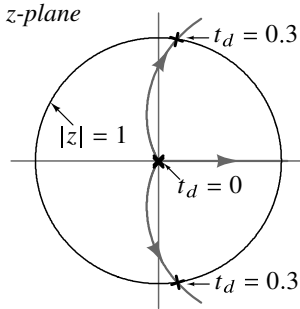
Figure 9.8   Locus of the poles of a second-order CTΔΣM as t_d changes.

From the discussion above, it is clear that a second-order modulator is a lot less tolerant of loop delay compared to a first-order design. How does one restore the NTF of the second-order loop? Using the first-order case as inspiration, we add a direct path around the quantizer with gain $\hat{k}_0$, as shown in Figure 9.9. As usual, the sampled pulse response of the loop-filter should equal the impulse response of the discrete-time prototype $L(z)$.

The sampled responses of the direct, $1/s$ and $1/s^2$ paths are given by

Direct path $\rightarrow$ z^{-1},

$$\frac{1}{s} \rightarrow \frac{1 - t_d}{z - 1} + z^{-1}\frac{t_d}{z - 1},$$

$$\frac{1}{s^2} \rightarrow \frac{(0.5 - t_d + 0.5t_d^2)z + 0.5(1 - t_d^2)}{(z - 1)^2} + z^{-1}\frac{t_d(1 - 0.5t_d)z + 0.5t_d^2}{(z - 1)^2}.$$
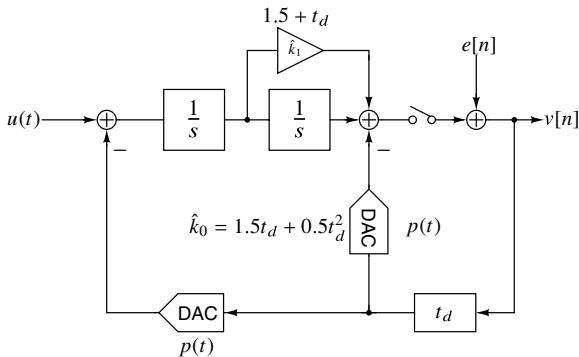
Figure 9.9   Restoring the NTF of a second-order CTΔΣM using a direct path.

To restore the NTF [1],

$$\hat{k}_0 z^{-1} + \hat{k}_1 \left[\frac{1 - t_d}{z - 1} + z^{-1} \frac{t_d}{z - 1} \right] +$$

$$\hat{k}_2 \left[\frac{(0.5 - t_d + 0.5t_d^2)z + 0.5(1 - t_d^2)}{(z - 1)^2} + z^{-1} \frac{t_d(1 - 0.5t_d)z + 0.5t_d^2}{(z - 1)^2} \right] = \frac{2z - 1}{(z - 1)^2}.$$

Equating coefficients on both sides, we obtain the set of equations below.

$$\begin{align}
0.5t_d^2 \hat{k}_2 - t_d \hat{k}_1 + \hat{k}_0 &= 0, \tag{9.5} \\
(0.5 - t_d + 0.5t_d^2)\hat{k}_2 + (1 - t_d)\hat{k}_1 + \hat{k}_0 &= 2, \\
-(0.5 + t_d - t_d^2)\hat{k}_2 + (1 - 2t_d)\hat{k}_1 + 2\hat{k}_0 &= 1.
\end{align}$$

The resulting solution is

$$\begin{align}
\hat{k}_2 &= 1, \\
\hat{k}_1 &= 1.5 + t_d, \\
\hat{k}_0 &= 1.5t_d + 0.5t_d^2. \tag{9.6}
\end{align}$$

Observe that the gain needed in the direct path increases with t_d. This makes sense, since a higher delay results in an increased phase shift of the loop-gain function – thereby needing a "stronger" zero to stabilize. The gain of the $1/s^2$ path does not change after compensation – this is intuitively satisfying, since the in-band NTF (and magnitude of the low-frequency loop-gain) has not changed.

The process of mitigating the effect of excess loop delay in an Nth order CT$\Delta\Sigma$M is similar, and is summarized below.

a. Determine the discrete-time equivalents of the direct, $1/s$, $1/s^2$, $\cdots$, $1/s^N$ paths driven by the delayed DAC pulse, and denote them by $\hat{L}_0(z), \cdots, \hat{L}_N(z)$, respectively.

b. Determine the coefficients of these paths $\hat{k}_0, \hat{k}_1, \cdots, \hat{k}_N$ so that $\hat{k}_0 \hat{L}_0(z) + \hat{k}_1 \hat{L}_1(z) + \cdots + \hat{k}_N \hat{L}_N(z) = (1/NTF(z)) - 1$. This will result in a set of $(N + 1)$ simultaneous equations in as many variables, which when solved, yield $\hat{k}_0, \hat{k}_1, \cdots, \hat{k}_N$.

The $\Delta\Sigma$ toolbox function `realizeNTF_ct` automates this procedure. From the intuition gleaned from the first- and second-order examples, one should expect that the gain of the direct path should be an increasing function of t_d. Further, the gain of the Nth order path should not change with delay. While the idea behind excess delay compensation is straightforward, the algebra seems daunting, even in the second-order case. Further, the whole process needs to be repeated if the DAC pulse changes. It, therefore, seems as if only a particularly brave-hearted (or masochistic) reader would attempt to analytically solve the excess delay problem in a high-order CT$\Delta\Sigma$M, with an arbitrary DAC pulse. Fortunately, as the next section shows, this is not as difficult as the analysis leading to equation (9.6) would indicate.
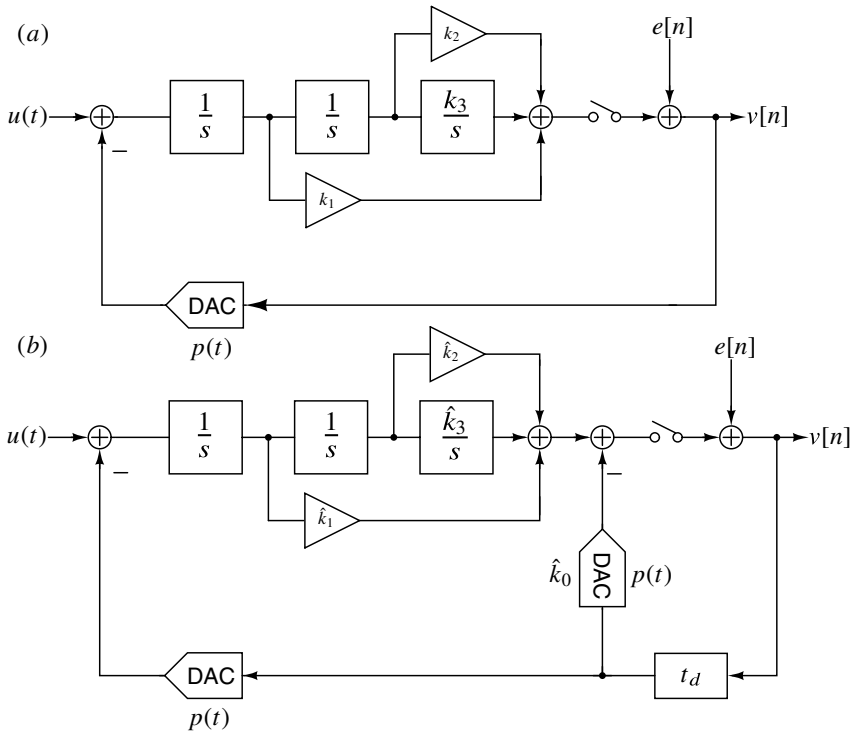
Figure 9.10   The excess delay compensation problem.

9.1.3   Excess Delay Compensation in High-Order Continuous-Time Delta-Sigma Modulators with Arbitrary DAC Pulse Shapes [2, 3]

A third-order CIFF modulator will be used as an example to illustrate the basic idea. The problem we wish to solve is as shown in Figure 9.10, in which the quantizer is replaced by an additive noise sequence $e[n]$. Part (a) of the figure shows a noise-shaping loop with k_1, k_2, and k_3 chosen so as to result in a desired NTF. How should $\hat{k}_0$, $\hat{k}_1$, $\hat{k}_2$, and $\hat{k}_3$ be chosen so that the CT$\Delta\Sigma$M of Figure 9.10(b) has the same NTF, even though there is an excess delay t_d?

Equivalently, one could pose the question in terms of the pulse response of the loop-filter, as shown in Figure 9.11. The problem now reduces to choosing $\hat{k}_0$, $\hat{k}_1$, $\hat{k}_2$, and $\hat{k}_3$, so that $y[n]$ is the same in both cases.
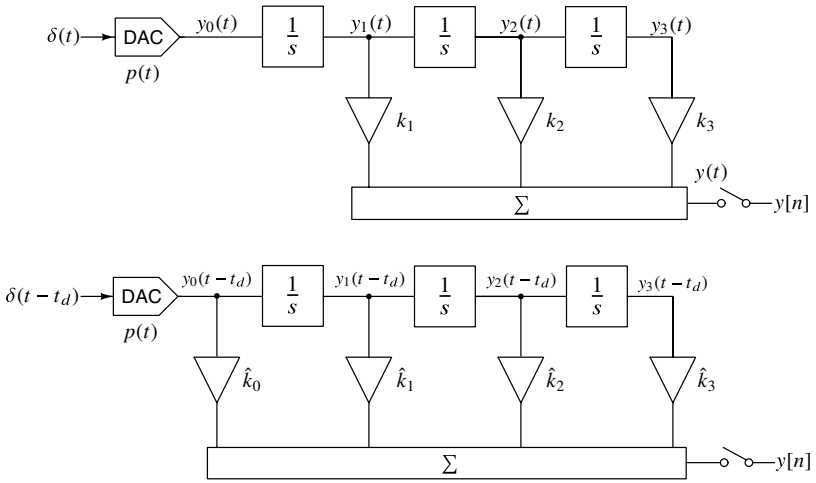
Figure 9.11   Solution by equating the pulse responses of the ideal and delayed loop-filter outputs.

We consider two cases. In the first, we assume that the delayed DAC pulse $p(t - t_d)$ does not extend beyond $t = 1$. This situation applies to CT$\Delta\Sigma$Ms with an RZ DAC with less than a half-cycle delay, or an impulsive DAC with delay less than one cycle. $p(t)$ is otherwise arbitrary.

We first examine the output of the $1/s^3$ path, without and with delay, as shown in Figure 9.12. As far as the loop's NTF is concerned, only the samples of the waveforms $y_3(t)$ and $y_3(t-t_d)$ at multiples of 1s are relevant. From the inset in the figure, it is apparent that for $t \geq 1$, $y_3(t)$ can be obtained from $y_3(t - t_d)$ by "looking t_d ahead". To this end, we apply the Taylor series to $y_3(t - t_d)$ at t as follows.

$$
\begin{aligned}
y_3(t) &= y_3(t - t_d + t_d) \\
&= y_3(t - t_d) + t_d \underbrace{\frac{d}{dt} y_3(t - t_d)}_{y_2(t-t_d)} + \frac{t_d^2}{2} \underbrace{\frac{d^2}{dt^2} y_3(t - t_d)}_{y_1(t-t_d)} + \frac{t_d^3}{6} \underbrace{\frac{d^3}{dt^3} y_3(t - t_d)}_{y_0(t-t_d) \equiv 0} \\
&= y_3(t - t_d) + t_d y_2(t - t_d) + \frac{t_d^2}{2} y_1(t - t_d). \tag{9.7}
\end{aligned}
$$
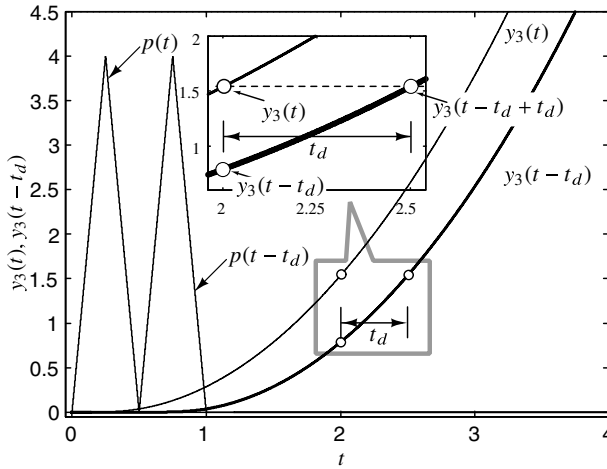
Figure 9.12   Pulse responses of a three integrator cascade with $(y_3(t - t_d))$ and without $(y_3(t))$ excess delay.

Note that terms of order three and beyond in the series expansion above are zero, since $p(t - t_d) = 0$ for $t \geq 1$. What (9.7) is telling us is that given t_d and $y_3(t - t_d)$, we can determine $y_3(t)$ if we have access to the derivatives of $y_3(t - t_d)$. From Figure 9.11, it is immediately apparent that the derivatives of $y_3(t - t_d)$ are simply the outputs of the preceding integrators. Thus, the output of the delay-free $1/s^3$ path can be obtained from the delayed output by adding appropriate portions of the $1/s^2$ and $1/s$ paths, as seen from (9.7). The same idea can be applied to obtain the ideal outputs of the $1/s^2$ and $1/s$ paths.

From Figure 9.11(a), the delay-free response of the loop-filter is given by $y(t) = k_3 y_3(t) + k_2 y_2(t) + k_1 y_1(t)$. Using the discussion above, the same output can be obtained from the delayed outputs as follows.

$$
\begin{aligned}
k_3 y_3(t) &= k_3 y_3(t - t_d) + k_3 t_d y_2(t - t_d) + 0.5 k_3 t_d^2 y_1(t - t_d) \\
+ k_2 y_2(t) &= k_2 y_2(t - t_d) + k_2 t_d y_1(t - t_d) \\
+ k_1 y_1(t) &= k_1 y_1(t - t_d)
\end{aligned}
$$

$$
y(t) = k_3 y_3(t - t_d) + (k_2 + k_3 t_d) y_2(t - t_d) + (k_1 + k_2 t_d + 0.5 k_3 t_d^2) y_1(t - t_d).
$$

It is important to note that the equation above is valid for all $t \geq 1$.

As far as the NTF is concerned, only the samples $y[n]$ are relevant. Thus, since the outputs of the delay-free and delayed paths are zero for $t = 0$, and equal for all times $t = 1, 2, \cdots, n$, we conclude that the NTF will be restored if the coefficients of the loop-filter are modified as follows.

$$
\begin{aligned}
\hat{k}_3 &= k_3, \\
\hat{k}_2 &= k_2 + k_3 t_d, \\
\hat{k}_1 &= k_1 + k_2 t_d + 0.5 k_3 t_d^2, \\
\hat{k}_0 &= 0.
\end{aligned}
$$

Thus, no direct path is necessary for *any* DAC shape or delay where the delayed pulse does not extend beyond $t = 1$. The NTF of the modulator can be restored to the one without

delay by appropriately tuning coefficients. A useful mnemonic for the delay-compensation formulae above, and the rationale behind it, is the following.

Let us assume that the continuous-time loop-gain function needed to yield a desired NTF (no excess delay) is

$$L_c(s) = \frac{k_3}{s^3} + \frac{k_2}{s^2} + \frac{k_1}{s}.$$

With excess delay, the loop-gain is $L_c(s)e^{-st_d}$. Delay can be compensated by multiplying $L_c(s)$ by e^{st_d}. To do this, when DAC pulses such that $p(t - t_d) = 0$ for $t \geq 1$, the $1/s^l$ path of the loop-gain should be replaced by a path whose transfer function is obtained by multiplying $1/s^l$ by e^{st_d}, where the exponential is expanded up to the $(l - 1)$th power as shown below.

$$\frac{1}{s^l} \rightarrow \frac{1}{s^l}e^{st_d} = \frac{1}{s^l} + t_d\frac{1}{s^{l-1}} + \cdots + \frac{t_d^{l-1}}{(l-1)!}\frac{1}{s}.$$

Thus, in the third-order case, $\hat{L}_c(s)$, which restores the NTF, is given by

$$
\begin{aligned}
\hat{L}_c(s) = L_c(s)e^{st_d} &= \frac{k_3(1 + st_d + 0.5s^2t_d^2)}{s^3} + \frac{k_2(1 + st_d)}{s^2} + \frac{k_1}{s} \\
&= \frac{k_3}{s^3} + \frac{k_2 + k_3t_d}{s^2} + \frac{k_1 + k_2t_d + 0.5k_3t_d^2}{s}.
\end{aligned}
\tag{9.8}
$$

An aspect of the preceding technique is that transformations between the s and z domains are avoided. The coefficients can be calculated without frightening algebra (see, for comparison, the analysis that led to (9.6)). Contrary to the impression given by the analysis in the preceding subsection, the compensated loop-filter's coefficients are independent of pulse shape (provided, of course, that $p(t - t_d) = 0$ for $t \geq 1$).

What happens in the more practical case, where $p(t - t_d)$ extends beyond $t = 1$? This occurs, for example, when an NRZ DAC is used and the the loop delay is positive (but less than one clock cycle). From the analysis based on the Taylor series used in this section, it is apparent that choosing the loop-gain function $\hat{L}_c(s)$ as in (9.8) does ensure that the pulse response of $\hat{L}_c(s)$ will equal that of $L(s)$ beyond $t \geq 2$, since $p(t - t_d) = 0$ for $t \geq 2$. Thus, the only difference between the samples of the pulse response of $L_c(s)$ and $\hat{L}_c(s)$ is at $t = 1$. This difference, therefore, should be made up by the direct path.

Let us now apply our techniques to the second-order modulator of Figure 9.9. The loop-gain function needed to achieve an NTF of $(1 - z^{-1})^2$ is given by

$$L_c(s) = \frac{1}{s^2} + \frac{1.5}{s} \Rightarrow k_2 = 1, k_1 = 1.5.$$

To compensate for an excess delay of t_d, we need to do the following.

$$
\begin{aligned}
\frac{1}{s^2} &\rightarrow \frac{1}{s^2} + t_d\frac{1}{s}, \\
\frac{1.5}{s} &\rightarrow \frac{1.5}{s}.
\end{aligned}
$$

Thus,

$$\hat{L}_c(s) = \frac{1}{s^2} + \frac{1.5 + t_d}{s} \Rightarrow \hat{k}_2 = 1, \hat{k}_1 = 1.5 + t_d.$$

Since an NRZ DAC with $t_d < 1$ is used, a direct path is necessary. To determine the gain of the direct path, the pulse responses of $L_c(s)$ and the delayed response of $\hat{L}_c(s)$ should be determined. It is straightforward to see that

$$y[1] = k_1 + 0.5k_2 = 2 \quad , \quad \hat{y}[1] = (1 - t_d)\hat{k}_1 + 0.5(1 - t_d)^2 \hat{k}_2$$
$$\Rightarrow \hat{k}_0 = y[1] - \hat{y}[1] = 1.5t_d + 0.5t_d^2.$$

We see that, with the Taylor series based technique described in this chapter, translating back and forth between the s and z domains is avoided. Simple formulae, amenable even to hand-calculations, and valid for *arbitrary* DAC pulse shapes, yield the coefficients of the compensated filter in terms of those of the delay-free design.

Design Example

A fourth-order maximally-flat NTF with an out-of-band gain of 1.5 is chosen as the target to be implemented. We assume that the coefficients corresponding to an ideal NRZ DAC with no delay are known. We will use the techniques developed in this chapter to determine the coefficients of the compensated modulator when an excess delay of a half-clock cycle is introduced. The NTF, obtained from the $\Delta\Sigma$ toolbox, is given by

$$NTF(z) = \frac{(z - 1)^4}{z^4 - 3.194z^3 + 3.892z^2 - 2.136z + 0.4444}.$$

We can compute the transfer function that yields the desired NTF for an NRZ DAC and no excess delay as

$$L_1(s) = \frac{0.6713s^3 + 0.2495s^2 + 0.0555s + 0.0061}{s^4}. \tag{9.9}$$

If we assume a CIFF topology, $k_1 = 0.6713$, $k_2 = 0.2495$, $k_3 = 0.0555$, and $k_4 = 0.0061$. For an excess delay of $t_d = 0.5$, the coefficients of the loop-filter have to be modified to

$$\hat{k}_4 = k_4 = 0.0061 , \ \hat{k}_3 = k_3 + k_4 t_d = 0.0585,$$
$$\hat{k}_2 = k_2 + k_3 t_d + k_4(t_d^2/2) = 0.2780,$$
$$\hat{k}_1 = k_1 + k_2 t_d + k_3(t_d^2/2) + k_4(t_d^3/6) = 0.8031. \tag{9.10}$$

With the coefficients above, the output of the loop-filter with the delayed DAC at $t = 1$ is 0.423. It is easy to show that the sample of the ideal pulse response at $t = 1$ for an NTF $(z - 1)^N/B(z)$ should be $(N +$ the coefficient of z^{N-1} in $B(z))$. For our NTF, it is 0.8060, which means that the direct path should have a gain of $(0.8060 - 0.423) = 0.37$.

9.1.4 Summary

In this section, we found that excess delay can render a CT$\Delta\Sigma$M unstable. Even if the modulator is stable, excess delay can reduce the stable input range. Modulators with high-order and/or high out-of-band gains are more sensitive to the effects of excess delay. Fortunately, the deleterious consequences of delay can be easily addressed by tuning the loop-filter's coefficients and by adding a direct path around the quantizer. Intuitively, coefficient tuning and the direct path, in amplifier parlance, add zeros (or move existing ones) to the loop-gain function and restore stability. Given the excess delay t_d, the modified coefficients (that restore the NTF) can be easily determined. The direct path can be implemented in many ways, some of which we will see in Chapter 10.

9.2   Time-Constant Variations of the Loop Filter

As seen in Chapter 8, the NTF of a CTΔΣM is derived from that of a discrete-time proto-type by using the impulse-invariant transformation. Denoting the impulse response of the loop-filter of the discrete-time prototype by $l_1[n]$, the impulse response of the continuous-time loop-filter $l_{ct,1}(t)$ has to be chosen so that

$$p(t) * l_{ct,1}(t)|_{t=nT_s} = l_1[n]. \qquad (9.11)$$

We saw several ways of arriving at the loop-filter coefficients that satisfy the equation above. In practice, however, the unity-gain frequencies of the integrators are dependent on component values. Figure 9.13 shows the simplified schematics of two commonly used
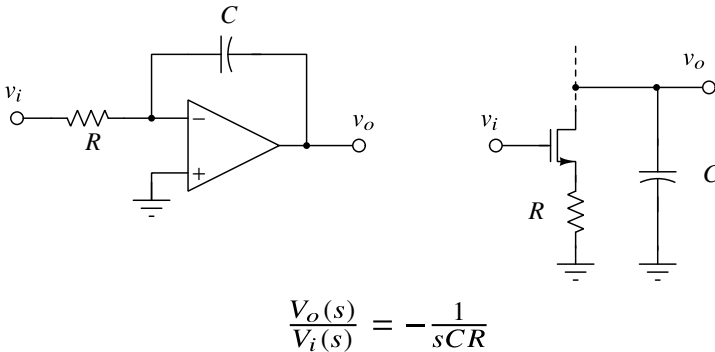
$$\frac{V_o(s)}{V_i(s)} = -\frac{1}{sCR}$$

Figure 9.13   Integrators realized using active-RC and Gm-C techniques.

integrators, namely the active-RC and Gm-C structures. Assuming an ideal opamp in the former, and that $g_m R \gg 1$ in the latter, we see that

$$\frac{V_o(s)}{V_i(s)} = -\frac{1}{sCR}. \qquad (9.12)$$

Since R and C vary with process and temperature, $l_{ct,1}(t)$ is bound to deviate from its nominal waveform, thereby modifying the NTF from the one that is desired. In this section, our aim is to get some intuition about the consequences of RC variation on modulator performance.

Denoting the transfer function of the loop-filter by $L_1(s)$, we see that decreasing every RC product by a factor k_p causes $L_1(s)$ to become $L_1(s/k_p)$, where $k_p > 1$. As a consequence, the magnitude of $L_1(s)$ in the signal band increases, as shown in Figure 9.14. Since the in-band loop-gain increases, the NTF must have a lower magnitude at low frequencies, indicating better noise-shaping. Thus, Bode's sensitivity integral predicts that the NTF must be worse off at out-of-band frequencies. One should therefore expect the maximum stable amplitude to be somewhat lower than that for the nominal value of the RC product. In the same vein, time-constants larger than nominal should increase in-band quantization noise and reduce the gain of the realized NTF at high frequencies.

Figure 9.15 shows the NTF magnitudes for a third-order CTΔΣM as the unity-gain frequencies of the constituent integrators vary by ±30%. In the time domain, the increased out-of-band gain manifests as a "more frantic wiggling" of the output sequence when $k_p > 1$, and vice versa, as shown in Figure 9.16.
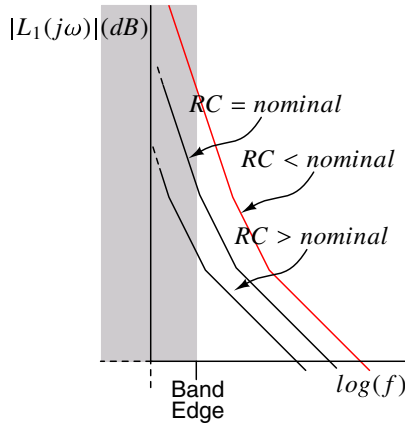
Figure 9.14   Magnitude of the loop-filter transfer function for the nominal RC, low RC ($k_p > 1$), and high RC ($k_p < 1$) .
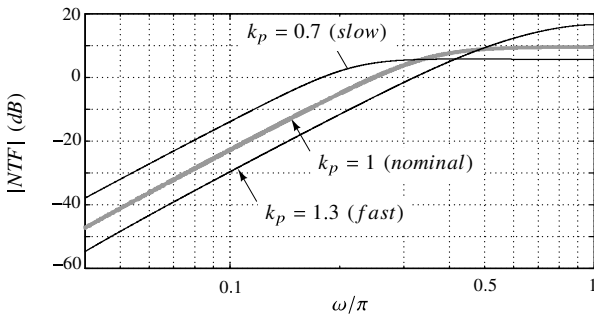
Figure 9.15 $|NTF|$ for a third-order CT$\Delta\Sigma$M , as the RC products vary. The nominal NTF is maximally flat with an out-of-band gain of 3.
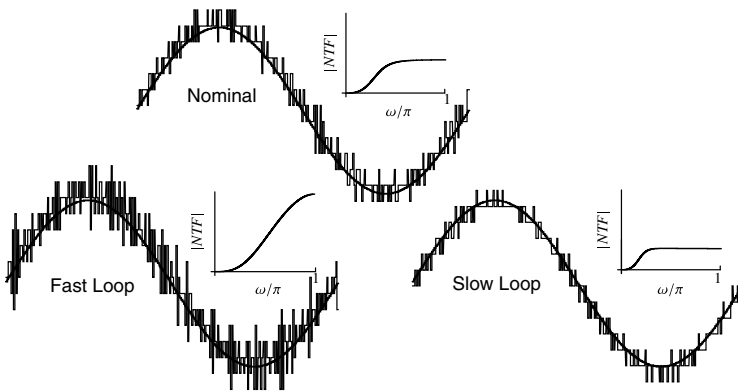
Figure 9.16   Time-domain illustration of the effect of RC variation.

The analysis above notwithstanding, we must realize that a high-order negative feedback loop is conditionally stable, and should not be surprised if a CTΔΣM becomes unstable for large deviations of the RC product from its nominal value. It therefore becomes necessary to tune RC time-constants close to their nominal values in the face of process, voltage, and temperature (PVT) variations. There are many ways of doing this – one method is shown in Figure 9.17. A current I is integrated on a digitally controlled capacitor bank for a time duration T_s. The voltage developed across the bank is compared to a reference generated by passing I through a resistor R. The decision of the comparator, therefore indicates if RC/T_s is greater or lesser than 1. The logic then varies the digital code controlling the capacitor bank in a successive-approximation fashion, so that RC/T_s approaches unity. The capacitors used in the integrators of the CTΔΣM are scaled versions of that used in the tuning circuit. The code developed in the tuning circuit is applied to all the capacitor banks in the modulator.

$$I\frac{T}{C} = IR$$
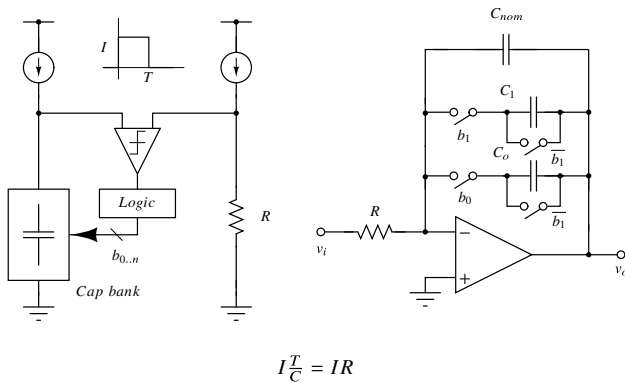
Figure 9.17   An example replica RC tuning loop with a digitally programmable capacitor bank.

9.3   Clock Jitter in Delta-Sigma Modulators

9.3.1   The Discrete-Time Case

We first examine the effect of clock jitter in a discrete-time ΔΣ modulator. The continuous-time input u is sampled up-front, as shown in Figure 9.18(a). Ideally, the edges of the sampling clock should occur exactly at integer multiples of T_s. In practice, however, there are deviations in the timing of the edges, as shown in Figure 9.18(b). These timing errors, denoted by the sequence $\Delta t[n]$, are termed jitter. For simplicity, we assume that $\Delta t[n]$ is a white sequence with an rms value $\sigma_{\Delta t}$. Also, we will assume that the jitter is small, so that the error sequence due to jitter is given by

$$e_j[n] = \left.\frac{du}{dt}\right|_{nT_s} \Delta t[n]. \tag{9.13}$$

We let u be a sinusoid with amplitude A and frequency f_{in}. Thus,

$$e_j[n] = 2\pi A f_{in} \cos(2\pi f_{in} n T_s)\Delta t[n]. \tag{9.14}$$
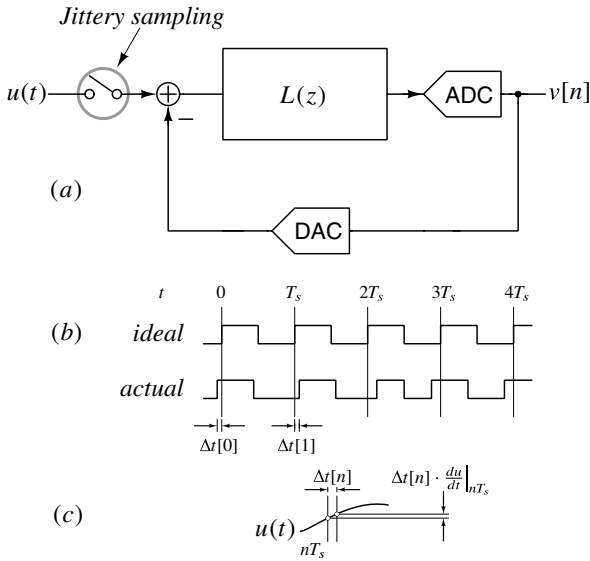
Figure 9.18   Clock jitter in a discrete-time $\Delta\Sigma$ modulator.

Since $\Delta t[n]$ is white[1], $e_j[n]$ is also white, and has a mean-square value of $2(\pi A \sigma_{\Delta t} f_{in})^2$. Of this power, only a fraction $1/OSR$ lies in the signal band. The in-band SNR due to jitter is thus given by

$$SNR_{jitter} = \frac{OSR}{4\pi^2 (f_{in}\sigma_{\Delta t})^2}.$$ (9.15)

From the discussion above, it is seen that in a discrete-time $\Delta\Sigma$ modulator, clock jitter degrades performance by corrupting the input even before it is processed by the converter. Since the discrete-time circuitry in the modulator is usually designed to settle well within half of a clock period, clock jitter does not influence the performance of the modulator itself. In a CT$\Delta\Sigma$M, however, the mechanism of degradation is quite different, as we will see below.

9.3.2   Clock Jitter in Continuous-Time Delta-Sigma Modulators

Consider the CT$\Delta\Sigma$M shown in Figure 9.19(a).   The ADC and DAC are clocked by *clk_adc* and *clk_dac* respectively. As discussed earlier in this chapter, *clk_dac* has to be delayed with respect to *clk_adc* to give the ADC enough time to resolve its input. Clock jitter affects the performance of the ADC and DAC. The ADC, sampling with a jittery clock, can be modeled as one working with a jitter-free clock, but with an error e_{adc} added at its input, as shown in Figure 9.19(b). In a similar fashion, the output of a jittery DAC can be modeled by an additive error e_{dac} at the output of a jitter-free DAC. The resulting model for the CT$\Delta\Sigma$M, which incorporates the effects of clock jitter, is given in

---

[1]We also assume that $\Delta t[n]$ is stationary, i.e., its statistics are independent of time. In contrast, $e_j[n]$ is white but not stationary.
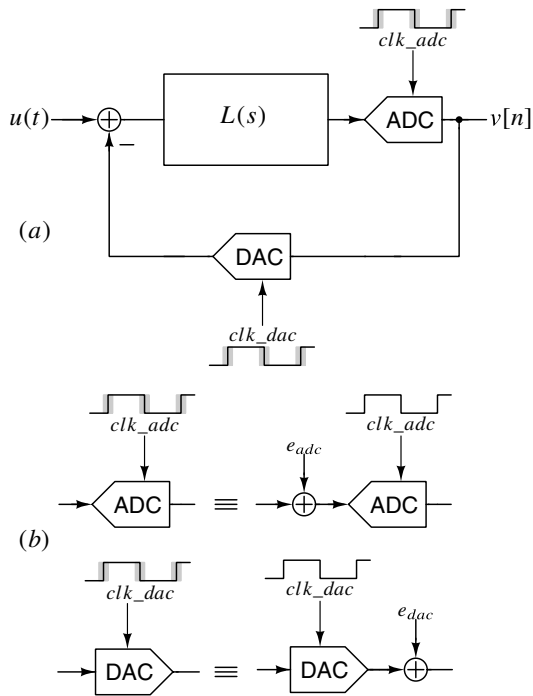
Figure 9.19   Clock jitter in a CT$\Delta\Sigma$M.

Figure 9.20. It is immediately apparent that e_{adc} is shaped by the modulator's NTF, just
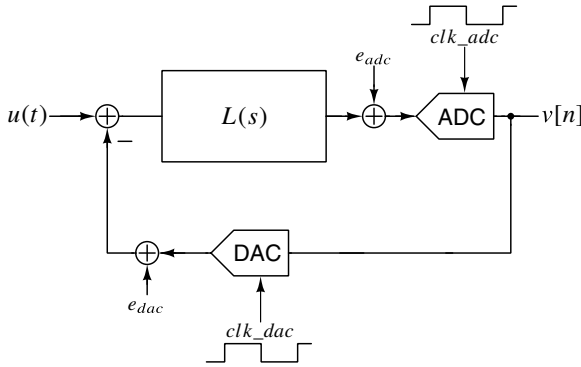
Figure 9.20   Modeling jitter induced errors in a CTΔΣM.

like quantization noise, and has virtually no effect on the in-band spectrum of the CTΔΣM
.

The story is different, however, with the error induced by jitter at the DAC output. As seen in Figure 9.20, e_{dac} adds to the input to the modulator. The low-frequency content of e_{dac} is responsible for the degradation of the modulator's in-band SNR, and therefore merits a more careful analysis [4, 5].

We first consider the case of an NRZ DAC, as shown in Figure 9.21. Without jitter, its output waveform has transitions that occur at multiples of T_s. The difference between the ideal and jittery output waveforms is shown in the lower part of the figure – it consists of a sum of slivers, whose width and height at nT_s are given by $\Delta t[n]$ and height $(v[n]-v[n-1])$, respectively.   We see that jitter in a particular clock edge introduces an error at that edge only if the modulator output changes in that cycle.

Since we intend to determine the in-band noise due to clock jitter, the e_{dac} can be thought of as an equivalent error *sequence* at the DAC input, as reasoned below. Consider an error pulse of width $\Delta t[n]$ and height $(v[n] - v[n - 1])$. At frequencies much smaller than $f_s = 1/T_s$, its spectrum is identical to that of a pulse with width T_s, and height $e_j[n] = (v[n] - v[n - 1])(\Delta t/T_s)$. Thus, the waveform $e_{dac}(t)$ at the DAC output can be replaced by an equivalent noise sequence $e_j[n]$ at the DAC input, as shown in Figure 9.21(b).

The in-band noise spectrum of v, therefore, consists of two parts – one due to shaped quantization noise, and the other due to clock jitter mixing with the sequence $(v[n] - v[n - 1])$. Since we assumed $\Delta t[n]$ to be a white sequence, $e_j[n]$ is also white, and has a mean square value

$$\sigma_{ej}^2 = \sigma_{dv}^2 \frac{\sigma_{\Delta t}^2}{T_s^2},\tag{9.16}$$

where σ_{dv}^2 denotes the mean-square value of $(v[n] - v[n - 1])$.

Assuming that u is in the signal band, where the $|STF| \approx 1$, we can write

$$v[n] = u[n] + e[n] * h[n].$$
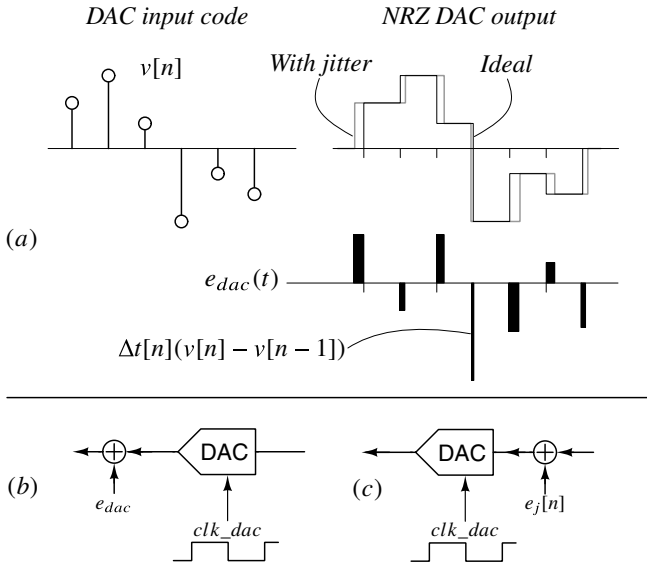
Figure 9.21   (a) DAC input sequence, and output waveforms with and without clock jitter. (b) Error waveform. (c) Equivalent model, accurate at low frequencies.

In the expression above, as usual, $e[n]$ and $h[n]$ denote the quantization noise and the impulse response corresponding to the NTF, respectively. Thus,

$$v[n] - v[n-1] = u[n] - u[n-1] + (e[n] - e[n-1]) * h[n].$$

Since we assumed u to be within the signal band, $u[n] \approx u[n-1]$. Thus,

$$v[n] - v[n-1] \approx (e[n] - e[n-1]) * h[n].$$

σ_{dv}^2 is more easily found in the frequency domain as shown below. $e[n]$ is assumed to be white. Since the step size is 2, its mean-square value is $1/3$. Thus,

$$\sigma_{dv}^2 \approx \frac{1}{3\pi} \int_0^\pi |(1 - e^{-j\omega}) NTF(e^{j\omega})|^2 d\omega.$$

Since e_j is white, only a fraction $(1/OSR)$ of its power lies in the signal band. The in-band noise due to jitter (J) is thus given by

$$
\begin{aligned}
J &= \sigma_{dv}^2 \frac{\sigma_{\Delta t}^2}{T_s^2} \frac{1}{OSR} \\
&\approx \frac{\sigma_{\Delta t}^2}{T_s^2} \frac{1}{3\pi OSR} \int_0^\pi |(1 - e^{-j\omega}) NTF(e^{j\omega})|^2 d\omega.
\end{aligned}
$$

This seemingly formidable expression can be separated into three components as indicated below:

$$J = \underbrace{\frac{\sigma_{\Delta t}^2}{T_s^2}}_{jitter} \underbrace{\frac{1}{OSR}}_{\text{in-band component}} \underbrace{\frac{1}{3\pi} \int_0^\pi |(1 - e^{-j\omega}) NTF(e^{j\omega})|^2 d\omega}_{\text{mean sq. value of transitions}}. \tag{9.17}$$

From (9.17) above, we see that the gain of modulator's NTF around $\omega = \pi$ has a significant bearing on the in-band noise. This makes sense, as it is the high-frequency gain of the NTF that dictates how v 'wiggles' around u. From Chapter 4, we know that modulators with a higher out-of-band gain have lower in-band quantization noise; however, as the discussion above indicates, this causes the noise due to clock jitter to increase.

What happens when the number of quantizer levels (M) is increased? The maximum stable amplitude can be expressed as $\alpha(M-1)$, where α depends on the details of the NTF. The peak signal-to- jitter-noise ratio (SJNR) is $\alpha^2(M-1)^2/J$, indicating that increasing M is an effective way of reducing the susceptibility of the modulator to clock jitter. This makes sense, since the height of the DAC transitions are reduced in relation to its full-scale output.

Let us take a moment to compare the mechanisms by which clock jitter degrades the performance of discrete-time and continuous-time $\Delta\Sigma$ modulators (assuming an NRZ feedback DAC). In the former, clock jitter "mixes" with the derivative of the input. In the latter, it "mixes" with the changes in the feedback waveform, which not only consists of the input but also contains shaped quantization noise.

9.3.3   Clock Jitter in Single-Bit Continuous-Time Delta-Sigma Modulators

The expression for J in (9.17) assumed that quantization error can be modeled as an additive white sequence. As we have seen in earlier chapters, this is not quite true in a 1-bit modulator. Figure 9.22 shows the ideal and jittered DAC waveforms in a 1-bit CT$\Delta\Sigma$M with an NRZ DAC. Since the height of the transition is always 2, the error introduced by jitter at the nth clock edge, assuming $v[n]$ differs from $v[n-1]$, is a pulse with width $\Delta t[n]$ and height 2. As in the multi-bit case, $e_{dac}(t)$ at the output of the DAC can be modeled by

Probability of a transition = p
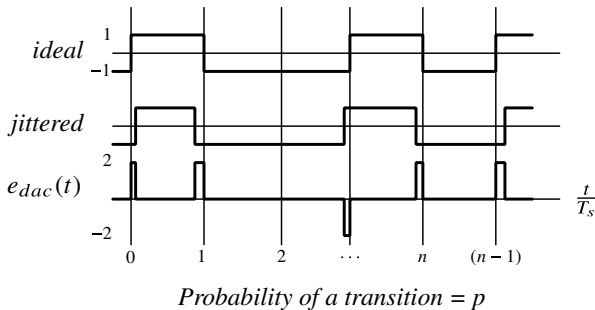
Figure 9.22   Modeling DAC error due to jitter in a single-bit CT$\Delta\Sigma$M with an NRZ DAC.

an equivalent error sequence e_j at the DAC input, where $e_j[n] = (\Delta t[n]/T_s)(v[n]-v[n-1])$. Denoting the probability that v makes a transition by p, we see that the in-band noise due to jitter is

$$J = \left(\frac{\sigma_{\Delta t}}{T_s}\right)^2 \frac{4p}{OSR}. \tag{9.18}$$

What value do we use for p? When $u \approx 0$, p should be high, as v transitions between ± 1 in an attempt to make the $\bar{v}$ equal to u. As u increases, the number of transitions in v
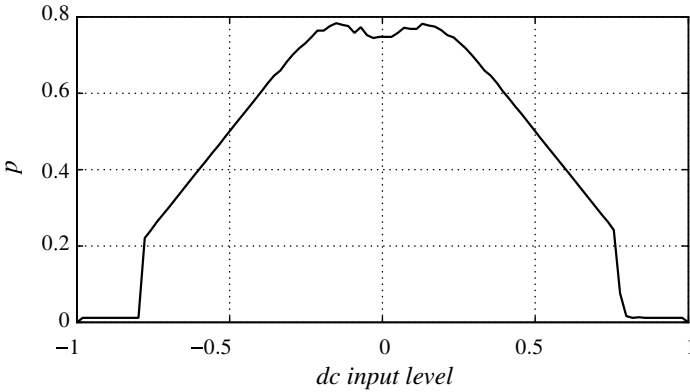
Figure 9.23   Probability of the output of a third-order single-bit CTΔΣM making a transition as a function of dc input.

should reduce, since v should have more 1's than -1's. When u increases further, so as to destabilize the modulator, the quantization error is much larger than the full scale, and this should lead to a dramatically reduced p. Figure 9.23, which shows p as a function of input dc level for a third-order single-bit CTΔΣM, confirms this intuition. For small u, $p \approx 0.8$, and falls linearly for $|u| > 0.2$ until $|u| \approx 0.8$, beyond which the modulator becomes unstable. It is thus seen that using $p = 0.8$ provides a good estimate of the in-band noise due to clock jitter.

Example : Jitter Noise in CTΔΣMs with 1 and 4 Bit Quantizers

Suppose that we need to design a CTΔΣM with an in-band SQNR of 110 dB in a 25 kHz bandwidth. Several combinations of order, OSR, and number of quantizer levels can be used to achieve the desired SQNR. We consider two modulators: one that uses a 1-bit quantizer, and another that employs a 4-bit quantizer. Further, we restrict the orders of both modulators to three. The NTFs of both modulators are chosen to be maximally flat with an out-of-band gain of 1.5. Clock jitter is assumed white, with an rms value of 25 ps. Calculations show that the two-level modulator needs twice the OSR as its 16-level counterpart to achieve the same peak SQNR. We also see that the SNR due to jitter is about 28 dB worse in the single-bit case than in the multi-bit one. This makes sense – the step size, relative to full scale, in the latter is smaller by a factor of 15 (23.5 dB), and the MSA is higher by about 2 dB. When compared to T_s, rms jitter is smaller by a factor of 2 (6 dB), but OSR is also smaller by a factor of 2. The net increase in SJNR due to multi-bit operation, therefore, is seen to be $(25.5 + 6 - 3) = 28.5$ dB.

	2 levels	16 levels
Order	3	3
NTF's OBG	1.5	1.5
OSR	128	64
f_S	6.4 MHz	3.2 MHz
T_S	156.25 ns	312.5 ns
Maximum stable amplitude	0.8 FS	FS
Peak SQNR	110 dB	110 dB
Peak SJNR	88 dB	116 dB

9.3.4 Continuous-Time Delta-Sigma Modulators with RZ DACs

Earlier in this section, we discussed the effect of clock jitter in CTΔΣMs with NRZ feedback DACs. We now analyze what happens when the DAC is of the RZ kind. Before we dive into the details, we pause for a bit and ponder over why RZ DACs are relevant in the first place. To see this, remember that the output waveform of any practical DAC (NRZ or RZ) will have nonzero rise and fall times. It is also the case that the rise and fall times will not necessarily be identical.

Section 6.7 described that such asymmetry is a source of a nonlinear *transition error*, but let us recall that discussion. Consider the output waveforms of a *practical* 1-bit NRZ DAC for two periodic input sequences, $\cdots, 1, -1, 1, -1, \cdots$ and $\cdots, 1, 1, -1, -1, \cdots$, as shown in Figure 9.24(a). Both sequences have an average value of zero. The DAC has a rise time t_r, and its fall time is assumed to be zero. The average value of the output waveform for the first input sequence is seen to be $-t_r/2$. For the second sequence, it is $-t_r/4$. Therefore, we see that the average of the output waveforms for the two zero-mean input sequences is not the same. This is enough to demonstrate the inherent nonlinearity of an NRZ DAC in the presence of rise–fall asymmetry. Intuitively, this nonlinearity can be explained as follows. In a 1-bit DAC, every positive transition must be followed by a negative transition (not necessarily at the next clock edge). If the transitions are identical, the errors made during the positive and negative transitions are equal in magnitude but opposite in sign. On average, therefore, their effect is zero – indicating no degradation of

in-band (low-frequency) performance. If the transitions are not symmetric, the errors due to them do not cancel. Further, the occurrence of transitions is dependent on the signal. Thus, the average error is not zero, but more important, it depends on the signal in a nonlinear manner.
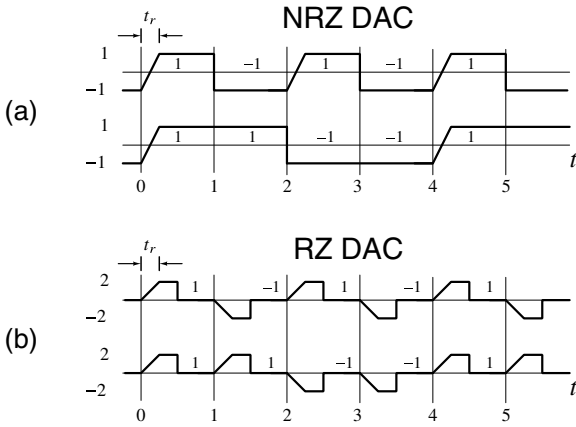
Figure 9.24   (a) Output of an NRZ DAC with inputs $\cdots, 1, -1, 1, -1, \cdots$ and $\cdots, 1, 1, -1, -1, \cdots$. Due to rise-fall asymmetry, the average value of the DAC output waveform is not the same in both cases. (b) Output of an RZ DAC with rise–fall asymmetry.

An RZ DAC, whose output waveforms are shown in Figure 9.24(b), does not have the problems highlighted above. This is because the output waveform is associated with a rising and falling transition in *every* clock cycle, independent of the input sequence. The inherent linearity of an RZ DAC, despite the rise–fall asymmetry, is the prime motivation for its use. The price paid for this, however, is the increased sensitivity to clock jitter, as we show next.
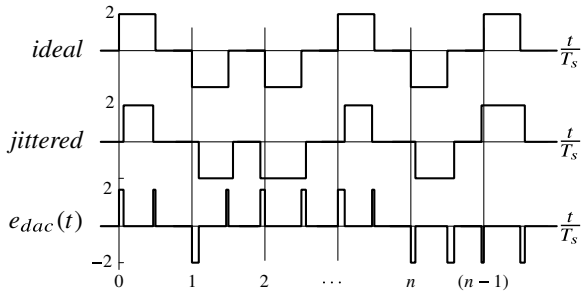
Figure 9.25   Error due to jitter in a single-bit CT$\Delta\Sigma$M with an RZ DAC.

Figure 9.25 shows the output waveforms of a single-bit RZ DAC without and with clock jitter. $e_{dac}(t)$ consists of slivers of height 2. Further, there are two such slivers in every clock period - this is due to the up and down going transitions associated with the pulse. Assuming (conservatively) that random jitter is white and affects all edges, we see

that the in-band noise due to jitter is

$$J = \left(\frac{\sigma_{\Delta t}}{T_s}\right)^2 \frac{8}{OSR}. \tag{9.19}$$

For a 1-bit modulator, therefore, an RZ DAC performs about 4 dB worse than an NRZ DAC for the same clock jitter. What happens in the multi-level case? In contrast to an NRZ DAC, where the heights of the transitions in the DAC waveform are a few levels, an RZ DAC's output goes all the way from 0 to $2 \cdot v[n]$ and back in every cycle. In the jitter scenario we have assumed (namely, white jitter), this puts an RZ DAC at a significant disadvantage with respect to jitter.

Further, while the RZ DAC is itself very linear, such a DAC increases the demands placed on the linearity of the loop-filter. This is due to the following. The loop-filter in a CTΔΣM processes the difference between the input and feedback *waveforms*. The RZ DAC results in a feedback waveform with twice the peak-to-peak amplitude when compared to its NRZ counterpart. As a result, the error waveform is much larger in magnitude in the former, even though the low-frequency content of both waveforms is the same. The loop-filter, therefore, has to be much more linear in the RZ case.

In [6], Adams proposes interleaving two RZ DACs to obtain the linearity of an RZ DAC while avoiding the jitter sensitivity of a single RZ DAC. This is often called the dual-RZ DAC.

9.3.5 Real Clock Sources and Phase Noise

So far in this section, we have gained an understanding of the mechanisms through which clock jitter degrades the performance of a CTΔΣM. Our analysis assumed white jitter – something that is not quite true in practice. It turns out that the output of a practical clock source can be expressed as

$$v_{clk} = \sin(2\pi f_s t + \phi(t)), \tag{9.20}$$

where $\phi(t)$ is small, and varies slowly when compared to $2\pi f_s t$. $\phi(t)$ results due to noise processes in the clock source, and perturbs the phase of the clock from its ideal trajectory of $2\pi f_s t$. It is, therefore, referred to as *phase noise*.

For small $\phi(t)$, (9.20) can be written as

$$v_{clk} \approx \sin(2\pi f_s t) + \phi(t) \cos(2\pi f_s t). \tag{9.21}$$

The power spectral density of v_{clk} is given by

$$P_{clk}(f) = \frac{1}{4}(\delta(f - f_s) + \delta(f + f_s)) + \frac{1}{4}(S_\phi(f - f_s) + S_\phi(f + f_s)), \tag{9.22}$$

where $S_\phi(f)$ represents the power spectral density of $\phi(t)$.

It is thus seen that in the presence of phase noise, the spectrum of the clock source can be thought of as consisting of the carrier, with power $1/2$, and the spectrum of $\phi(t)$, which is translated around $\pm f_s$. It turns out that a large part of $\phi(t)$ varies slowly in relation to $2\pi f_s t$. $S_\phi(f)$ is thus lowpass in nature, and reduces with frequency, before flattening off. When v_{clk} is measured on a spectrum analyzer, the power at negative frequencies folds

atop that at positive frequencies, resulting in a spectrum similar to that in Figure 9.26. As the figure shows, $S_\phi(\Delta f)$ is the ratio of the power of v_{clk} in a 1 Hz bandwidth around $(f_s + \Delta f)$ to the power of v_{clk} (=1/2). In practice, therefore, $S_\phi(\Delta f)$ is specified (usually in dBc) in terms of the power spectral density of v_{clk} at a frequency Δf offset from f_s.
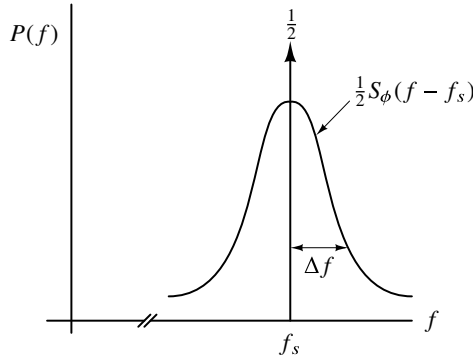
Figure 9.26   Power spectral density of v_{clk} as observed on a spectrum analyzer.

How does $\phi(t)$ manifest in the time domain? Without noise, $\phi(t) = 0$ and the rising edges of v_{clk} occur precisely at integer multiples of $1/f_s = T_s$. Phase noise causes a deviation in the zero-crossings of v_{clk} from their noise-free values. It is easy to see that $\phi(t)$ displaces the rising edges of v_{clk} by

$$\Delta t[n] = \frac{\phi[nT_s]}{2\pi} T_s. \tag{9.23}$$

We have earlier seen that the effect of clock jitter on a CT$\Delta\Sigma$M with an NRZ feedback DAC can be modeled by adding an error sequence $e_j[n] = (v[n] - v[n-1])(\Delta t[n]/T_s)$ to the modulator output. Assuming an in-band input $u(t) = A\cos(2\pi f_{in}t)$ and that $|STF| \approx 1$, we see that

$$v[n] - v[n-1] \approx 2\pi A f_{in} T_s \sin[2\pi f_{in}nT_s] + (e[n] - e[n-1]) * h[n], \tag{9.24}$$

where $h[n]$ is the impulse response corresponding to the NTF. Using (9.23), we can express $e_j[n]$ as

$$e_j[n] = \underbrace{A(f_{in}/f_s)\phi[nT_s]\sin[2\pi f_{in}T_sn]}_{e_{j1}=\text{input signal component}} + \underbrace{[(e[n] - e[n-1]) * h[n]] \cdot (\phi[nT_s]/2\pi)}_{e_{j2}=\text{shaped quantization noise component}},$$

$$\tag{9.25}$$

where e_{j1} is due to the interaction of jitter with the input signal, while e_{j2} models the mixing of shaped quantization noise with jitter. The spectrum of e_j is illustrated with the simplified sketch shown in Figure 9.27. Since multiplication in the time domain corresponds to convolution in the frequency domain, it follows that the spectral density of e_{j1} is that of $\phi[nT_s]$, scaled by $(A^2/2)(f_{in}/f_s)^2$ and translated around f_{in}. The input tone has an amplitude A. Therefore, the power spectral density of e_{j1} at a frequency Δf offset from f_{in} in relation to that of the input (= $A^2/2$) is simply $(f_{in}/f_s)^2 S_\phi(\Delta f)$. Hence, the effect of the close-in phase noise of the clock source is to broaden the line spectrum that we would expect for a sinusoidal input.
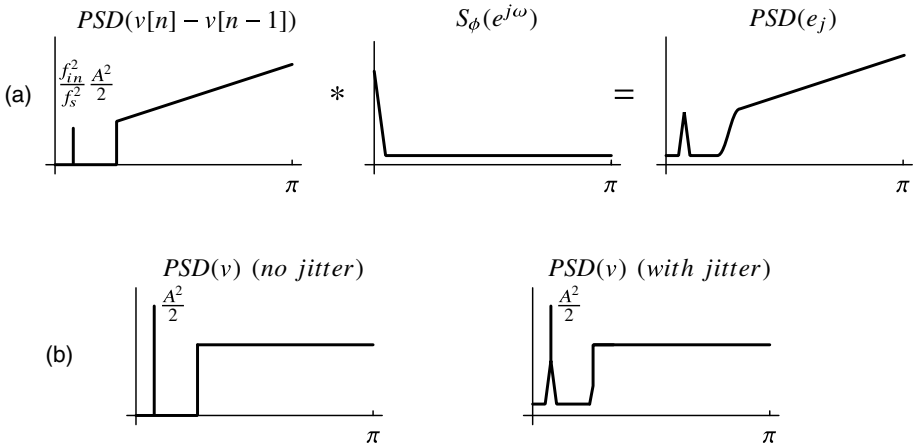
Figure 9.27   Simplified sketch illustrating the effect of clock phase noise on the PSD of a CTΔΣM.

The PSD of e_{j2} is the result of convolution of the spectra of the first difference of the shaped quantization noise and the phase-noise sequence. It is the in-band power of e_{j2} that we are interested in. From Figure 9.27(a), we see that the majority of the (white) in-band noise due to jitter is contributed by the far-out phase noise convolving with the shaped noise at high frequencies. Without jitter, $PSD(v)$ should have very little in-band noise and shaped out-of-band noise, as shown in Figure 9.27(b). With jitter, however, the in-band spectrum is corrupted by sidebands around the input tone, as well as an increased noise floor.
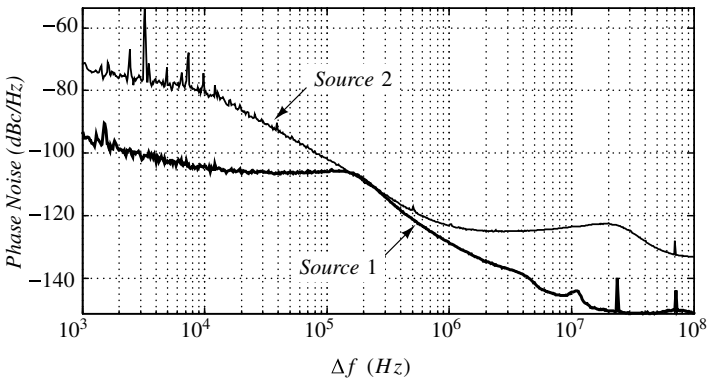
Figure 9.28   Measured phase noise of two 6 GHz clock sources as a function of frequency offset.

Figure 9.28 shows the measured phase noise plots of two 6 GHz clock sources as a function of frequency offset. It is apparent that the first source should result in a much smaller in-band noise due to jitter, since its phase noise at large frequency offsets is about 20 dB lower than that for source 2. Are any (or both) of these clock generators suitable for a single-bit CTΔΣM operating with OSR= 50, and targeting an SNDR of 75 dB?

Source-1 has a far-out phase noise spectral density of about -150 dBc/Hz. Over a 6 GHz bandwidth ($= f_s$), this corresponds to -52.2 dBc. The rms phase error is $\phi_{rms} = \sqrt{10^{-5.22}} = 2.45 \times 10^{-3}$ radians. In the time domain, this corresponds to an rms (white) jitter of $\phi_{rms}/(2\pi f_s) = 65\times10^{-15}$ s. Using (9.18) with $p = 0.8$, and assuming that the MSA is -3 dBFS, the peak signal-to-jitter-noise ratio is calculated to be 74 dB. To the jitter noise, we must also add thermal and quantization noise, which will degrade the in-band SNDR further. The conclusion is that both these clock sources are incapable of achieving the performance we seek from our 1-bit CTΔΣM with an NRZ DAC. Architectural changes are necessary to reduce the susceptibility of this modulator to clock jitter. Several alternatives exist; we will examine some of these in the next section.

9.4 Addressing Clock Jitter in Continuous-Time Delta-Sigma Modulators

From our discussion on the manifestation of jitter in RZ and NRZ DACs, it is apparent that the shape of the DAC pulse has a significant bearing on the jitter sensitivity of a CTΔΣM. One approach to mitigating the effect of jitter, therefore, is to choose the DAC pulse shape in a way that jittery clock edges have little or no effect on the low frequency content of the feedback DAC waveform. This, for example, can be accomplished by using an impulsive feedback DAC, as we show below. Figure 9.29(a) shows the outputs of a
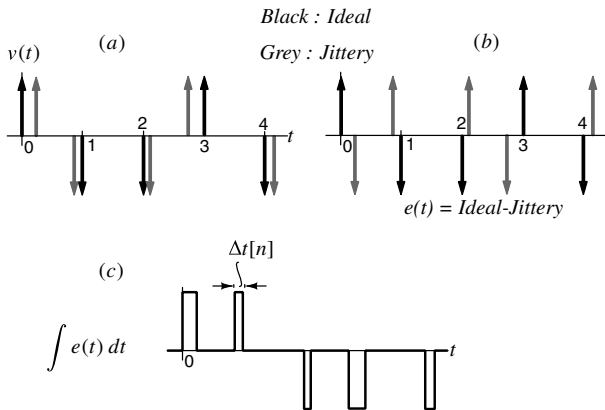
Figure 9.29 Effect of clock jitter on an impulse DAC waveform. (a) Outputs of a jitter-free and jittery DAC, (b) $e(t)$, and (c) integral of $e(t)$.

jitter-free and jittery impulsive 1-bit DAC. The difference between these two waveforms, which corresponds to the error caused by jitter, is shown in part(b) of the figure. The in-band components of $e(t)$ are responsible for the degradation of the modulator's SNR. To better understand the low-frequency power spectral density of $e(t)$, consider the integral of $e(t)$, shown in Figure 9.29(c). The area of each pulse is $v[n]\Delta t[n]$. As we concluded when we evaluated the jitter error of an NRZ DAC, the low-frequency spectral density of this waveform is the same as that of the sequence $v[n](\Delta t[n]/T_s)$. If jitter is assumed to be white, it follows that the power spectrum of the *integral* of $e(t)$ is also white. The PSD of $e(t)$ must hence be proportional to ω^2, indicating that noise due to clock jitter is, to first order, shaped out of the signal band. An impulsive DAC is, therefore, less susceptible

to clock jitter than its NRZ counterpart. Intuitively, this is because the area of the pulse (which dictates the low-frequency content of the DAC waveform) is not affected by jitter. The fact that the pulse position is altered by jitter is of secondary consequence, as this error has a highpass spectrum.
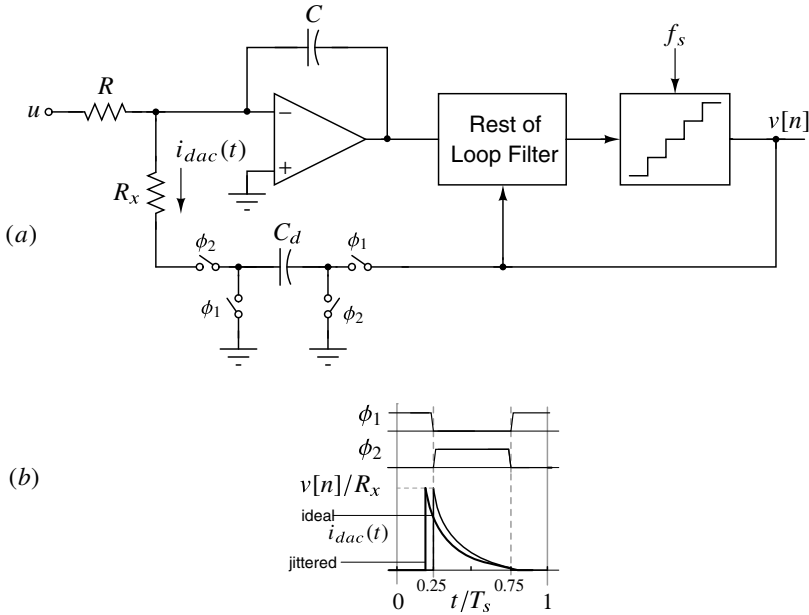
Figure 9.30 (a) CTΔΣM with a switched-capacitor feedback DAC. (b) $i_{dac}(t)$ with and without jitter.

In reality, it is impossible to realize an impulse. A practical alternative is to approximate the impulse by an exponentially decaying pulse [7]. How can one implement a DAC with such a pulse shape? One way is shown in Figure 9.30. The sampling rate of the modulator is assumed to be $f_s (= 1/T_s)$. The capacitor C_d is charged to $v[n]$ during ϕ_1. In ϕ_2, it is discharged into the virtual ground of the operational amplifier though the resistor R_x. If the opamp is ideal, the DAC current (without jitter) is given by

$$i_{dac}(t) = \sum_n v[n]p(t - nT_s), \qquad (9.26)$$

where

$$p(t) = \frac{1}{R_x} \exp\left(\frac{-(t - \frac{1}{4})}{R_x C_d}\right) \quad , \quad \frac{1}{4} \le t/T_s \le \frac{3}{4} \qquad (9.27)$$

and zero elsewhere.

How does this DAC fare with a jittery clock? Figure 9.30(b) shows $i_{dac}(t)$. Assuming white jitter on both edges of ϕ_1 and ϕ_2, we obtain the difference in the areas of the current pulses that are fedback with and without jitter is:

$$e(t) = \frac{v[n]}{R_x} \exp\left(\frac{-T_s}{2R_x C_d}\right)\left(\Delta t[n] - \Delta t\left[n + \frac{1}{2}\right]\right). \qquad (9.28)$$

From this equation, it is clear that the error due to jitter reduces exponentially with the discharge time constant $R_x C_d$. It is thus tempting to use a small R_x, so as to hasten the discharging of the capacitor. This, however, has a very undesirable consequence. Since the capacitor is charged to $v[n]$ at the end of ϕ_1, the initial current injected by the DAC during ϕ_2 is $v[n]/R_x$. This current has to be supplied by the opamp, which mandates that it be extremely linear (in turn increasing its power dissipation).

Choice of C_d in a Switched-Capacitor DAC

How should C_d should be chosen in the CT$\Delta\Sigma$M of Figure 9.30, so as to achieve an STF with a dc gain of 1? This means that $\bar{v} = u$, for a dc u. The average current flowing through the input resistor is seen to be u/R. Since the average current flowing through the integrating capacitor has to be zero, it follows that $\overline{i_{dac}(t)} = u/R$. Assuming that the capacitor is completely discharged during ϕ_2, it is straightforward to see that

$$\overline{i_{dac}(t)} = -\bar{v} f_s C_d. \tag{9.29}$$

Thus, the condition $|STF(0)| = 1$ dictates that $f_s C_d = 1/R$, indicating that the switched-capacitor resistor in the feedback path must equal the input resistor.

To summarize, a switched-capacitor (SC) feedback DAC mitigates the effect of clock jitter at the expense of the modulator's linearity. The fundamental reason for this is the high peak-to-average ratio of the exponentially decaying DAC pulse. It turns out that an SC DAC also severely compromises the alias rejection of the modulator when the opamp is not ideal [8]. In view of the several problems that afflict an SC DAC, it is not as attractive as it may at first seem.

9.5 Mitigating Clock Jitter Using FIR Feedback

A particularly elegant way of addressing the clock jitter problem in a CT$\Delta\Sigma$M is to use FIR feedback [9, 10]. We illustrate with the single-bit example shown in Figure 9.31. The 2-level output sequence v is filtered by an N-tap lowpass FIR filter with transfer function $F(z)$, before exciting the main feedback DAC. The DAC has an NRZ pulse shape. For simplicity, the tap weights of $F(z)$ are assumed to be identical. Since $v = \pm 1$, the magnitude of transitions in v is 2. With $F(z)$ in place, this is reduced to $2/N$. Thus, the magnitudes of the steps in the feedback DAC waveform (denoted by $v_1(t)$ in Figure 9.31(a)) are N times smaller than what they would have otherwise been. Since noise due to clock jitter is proportional to the height of the transitions in the DAC output, it follows that the in-band mean-square noise due to jitter is reduced by $20\log(N)$ dB. This argument assumes that the jitter is common to all the DAC elements. An implementation should, therefore, use a common clock for all elements – i.e. not a tree of clock buffers.

The FIR DAC has other important benefits. Since $F(z)$ is a lowpass filter, the input component of v is not affected, though the power of the shaped noise is reduced. The DAC's output $v_1(t)$, therefore, has a reduced high-frequency content that closely follows the input u. The error processed by the loop-filter, which is $u(t) - v_1(t)$, is therefore much smaller. This relaxes the linearity requirements of the loop-filter, just like in a CT$\Delta\Sigma$M
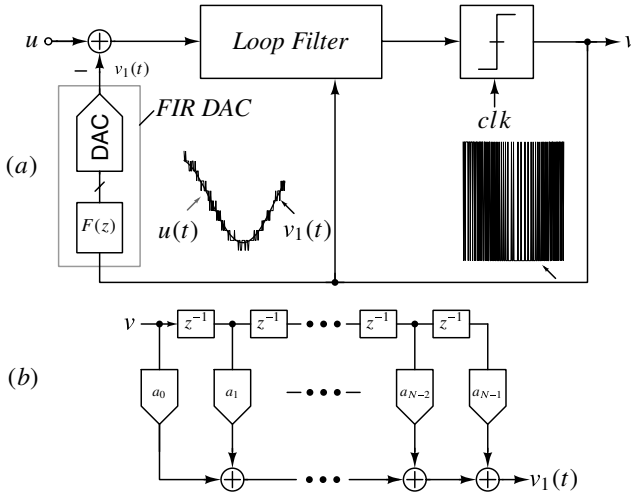
Figure 9.31   Using an FIR feedback DAC to mitigate noise due to clock jitter in a single-bit modulator.

with a multi-bit DAC. Intuitively, since $v_1(t)$ "looks" like the output of a multi-bit DAC, one should expect similar benefits with respect to clock jitter and loop-filter linearity.

Implementing the FIR DAC as drawn in Figure 9.31(a) is problematic when the DAC levels are not equally spaced (due to mismatched components). Recognizing that $v[n]$ is a two-level sequence, a linear DAC-filter combination can be realized using the semi-digital approach [11], as shown in Figure 9.31(b). Here, the delays are implemented digitally, while the individual DAC outputs (which are assumed here to be currents) are weighted and summed in the analog domain. It is easy to see that DAC mismatch modifies the transfer function of the filter but does not cause nonlinearity.
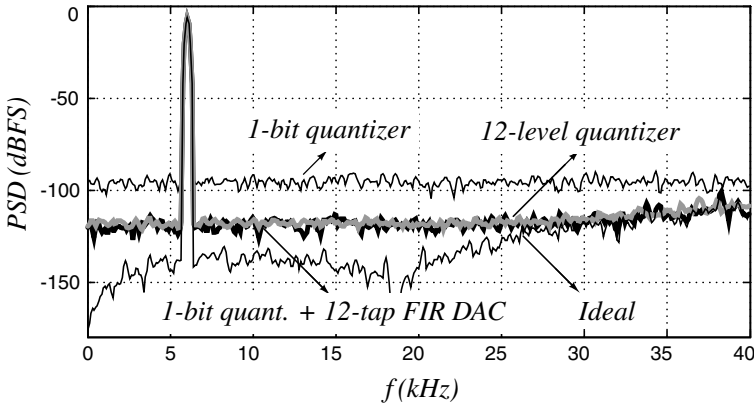
Figure 9.32   Comparison of the PSDs of various CTΔΣMs with clock jitter, assumed white ($f_s = 6.144$ MHz, $\sigma_{\Delta t} = 160$ ps). The ideal (jitter-free) spectrum is also shown for comparison.

Figure 9.32 compares the PSDs of single-bit, multi-bit, and single-bit+FIR DAC third-order CTΔΣMs with clock jitter. The PSD without jitter is also shown. The input is a -6 dBFS sinusoid. The multi-bit modulator employs a 12-level quantizer, and is clocked at 1/3 the clock rate of the single-bit design. To achieve the same in-band quantization noise as the single-bit design, the out-of-band gain of the NTF has to be increased to 2.8. With jitter, the performance of the single-bit design is significantly worse than that of the multi-bit one, as expected. When a 12-tap FIR DAC is used, however, the noise due to jitter reduces by $20\log_{10}(12) = 21.5$ dB, and the performances of the 1-bit and multi-bit designs are nearly the same. The assumption here is that NTF of the single-bit modulator has been restored after incorporating the FIR DAC, as discussed next.

Having sung praises of the FIR DAC's advantages, we must not lose sight of the fact that the FIR filter introduces delay in the ΔΣ loop, and most likely renders the modulator unstable. One of the key design challenges with an FIR DAC, therefore, is to compensate the loop for the effect of the FIR filter. Stated formally, the problem is as follows. A prototype modulator with an NRZ DAC has a known (desired) NTF. An FIR filter $F(z)$ is inserted before the DAC, to take advantage of several of its properties described above. The questions we wish to answer are the following.

   a. Is it possible to restore the NTF of the loop to that of the prototype?

   b. If so, how should the loop-filter be modified to restore the NTF?

Fortunately, it turns out that the first question can be answered in the affirmative. In fact, as we will see below, the NTF of loop with the FIR DAC can be restored *exactly*. In a manner reminiscent of the process of excess loop delay compensation, it turns out that this can be done by modifying the loop-filter coefficients and by adding a *direct path FIR filter* around the quantizer.

To illustrate the process of compensation, we use a normalized third-order CIFF modulator with an NRZ DAC. The method of moments [3], which we discussed in Chapter 8, allows us to quickly and simply determine the modified coefficients. We denote the coefficients and DAC pulse moments of the prototype modulator by $k_1, \cdots, k_3$ and $\mu_0, \cdots, \mu_2$, respectively. As mentioned earlier, the prototype's coefficients are assumed to have been chosen to achieve a desired NTF. The main feedback DAC of the prototype is modified to a 4-tap FIR DAC with equal tap weights (each being 0.25), as shown in Figure 9.33(a). We claim that the loop can be compensated by modifying the coefficients to $\tilde{k}_1, \cdots, \tilde{k}_3$ and by adding the compensating FIR DAC (whose transfer function is denoted by $F_c(z)$).

Consider the pulse response of the loop-filter at the point $\tilde{y}_1(t)$ in Figure 9.33(a). The FIR DAC driving the loop-filter can be thought of as a modified NRZ DAC with a pulse shape 4 seconds wide, with height of 0.25, as shown in part(c) of the figure. The moments of this pulse are denoted by $\tilde{\mu}_0, \cdots, \tilde{\mu}_2$. Since the modulator is of third-order, the relevant moments of the DAC pulse are $\tilde{\mu}_0 = 1$, $\tilde{\mu}_1 = 2$, and $\tilde{\mu}_2 = 16/3$. From the theory of moments in Chapter 8, $y_1(t)$ for $t \geq 4$ can be expressed as

$$\tilde{y}_1(t) = \tilde{k}_3 \left(\frac{\tilde{\mu}_0}{2} t^2 - \tilde{\mu}_1 t + \frac{\tilde{\mu}_2}{2} \right) + \tilde{k}_2 (\tilde{\mu}_0 t - \tilde{\mu}_1 t) + \tilde{k}_1 \tilde{\mu}_0 \ , \quad t \geq 4. \tag{9.30}$$

For the prototype loop-filter, $y(t)$ (Figure 9.33(b)) is given by

$$y(t) = k_3 \left(\frac{\mu_0}{2} t^2 - \mu_1 t + \frac{\mu_2}{2} \right) + k_2 (\mu_0 t - \mu_1 t) + k_1 \mu_0 \ , \quad t \geq 1. \tag{9.31}$$
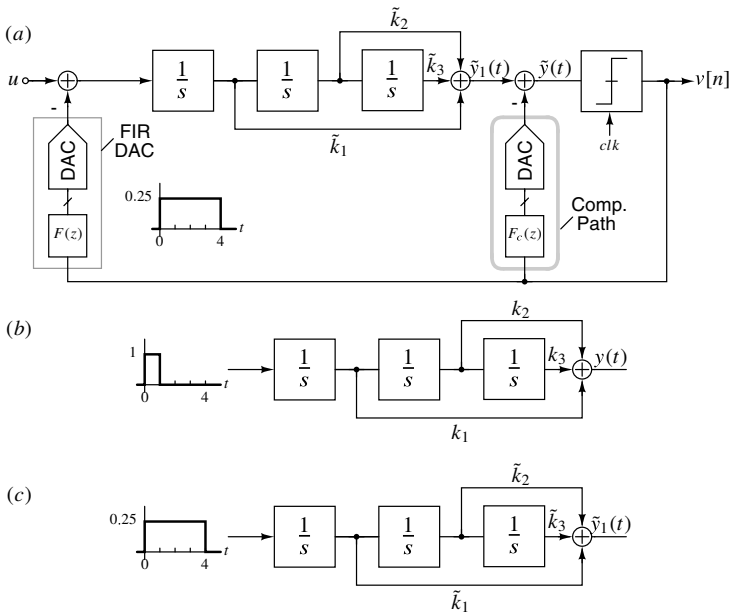
Figure 9.33   (a) A third-order CT$\Delta\Sigma$M with a 4-tap FIR DAC compensated for the delay of the FIR DAC. $F_C(z)$ is also a 4-tap FIR filter. (b) Determining the pulse response $y(t)$ of the loop-filter without the FIR DAC. (c) $\tilde{y}_1(t)$ can be made to equal $y(t)$ for $t \geq 4$ by modifying coefficients. $F_C(z)$ is needed to compensate for the difference in the sampled pulse response for $t < 4$.

where $\mu_0 = 1$, $\mu_1 = 1/2$ and $\mu_2 = 1/3$. If $\tilde{k}$'s are chosen according to

$$
\begin{aligned}
\tilde{k}_3 &= k_3, \\
\tilde{k}_2 &= k_2 + k_3 \, (\tilde{\mu}_1 - \mu_1) = k_2 + 1.5k_3, \\
\tilde{k}_1 &= k_1 + (\tilde{\mu}_1 - \mu_1) \, (k_2 + \tilde{\mu}_1 k_3) - 0.5k_3 \, (\tilde{\mu}_2 - \mu_2) = k_1 + 1.5k_2 + 0.5k_3.
\end{aligned}
\tag{9.32}
$$

Hence, $\tilde{y}_1(t)$ and $y(t)$ are equal for $t \geq 4$.
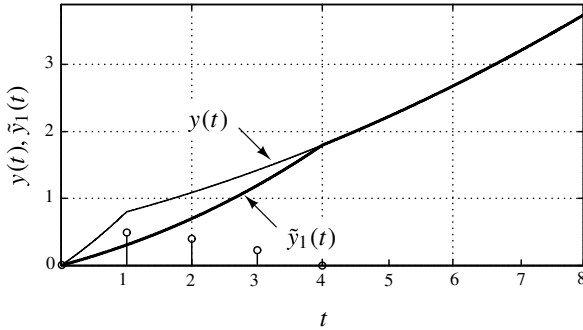
Figure 9.34   Loop filter pulse responses of the NRZ prototype, the (coefficient tuned) loop-filter with a 4-tap FIR DAC, and the response of the compensation filter. The main FIR DAC taps are all assumed to be equal. The direct path DAC with response $F_c(z)$ should make up for the difference $(\tilde{y}_1(t) - y(t))$.

Figure 9.34 shows $\tilde{y}_1(t)$ and $y(t)$ in our 4-tap example. From this, it is immediately clear that tuning $\tilde{k}_1, \cdots \tilde{k}_3$ can *only* ensure that the sampled pulse response matches that of the prototype beyond $t \geq 4$. There are just not enough degrees of freedom to achieve the desired pulse response for $t < 4$. One possible way of achieving the desired pulse response for $t < 4$ is to use a four-tap compensation filter $(F_c(z))$, in a direct path around the quantizer, as shown in Figure 9.33(a). The taps of $F_c(z)$ can be computed using the following steps.

a. Using $k_1, \cdots, k_3$ from the prototype and the coefficients of the main FIR DAC, use (9.32) to compute $\tilde{k}_1, \cdots, \tilde{k}_3$.

b. Determine the pulse response of the prototype loop-filter and the coefficient tuned loop-filter with the FIR feedback DAC. These responses will be identical beyond $t = M$, where M denotes the number of FIR DAC taps.

c. Determine the difference between the pulse responses in step (b) above, that lasts for a duration M. The direct path filter taps are the samples of this difference at times $1, \cdots, (M - 1)$.

It is not necessary for the compensation path to appear directly around the quantizer. It can be moved to the input of the third or second integrators. For an M-tap FIR filter with all

equal taps, it is straightforward to see that (9.32) results in

$$\tilde{k}_3 = k_3 \; ,$$

$$\tilde{k}_2 = k_2 + \frac{(M-1)}{2}k_3,$$

$$\tilde{k}_1 = k_1 + \frac{(M-1)}{2}k_2 + \frac{(M-1)(M-2)}{12}k_3. \qquad (9.33)$$

From the discussion above, it appears as if choosing a large number of FIR taps is beneficial, since it results in better filtering of the feedback sequence. This, apart from reducing jitter sensitivity, should also reduce the magnitude of the error signal that is processed by the loop-filter and thereby improves linearity. What then, is the limit to the number of FIR taps? What prevents one from using, for instance, a 200-tap filter? The delay introduced by the filter cannot be a problem, since its effect can be exactly compensated by coefficient tuning and a direct-path FIR DAC, as discussed earlier.

In our analysis of jitter immunity achieved by FIR feedback, we did not consider the effect of jitter noise injected by the compensation DAC. The latter depends on the specific loop-filter architecture, and the location of the compensation DAC. Analysis shows that these are important considerations that limit the number of taps that one can use. As for linearity improvement, it turns out that the phase shift between u and the FIR DAC output $v_1(t)$ (Figure 9.31) increases with the number of taps. This means that the magnitude of $(u - v_1(t))$ will *increase* if a large number of taps are used. Yet another consequence of FIR feedback is a change in the STF. As seen from (9.33), the loop-filter coefficients k_1 and k_2 have to increase to compensate for the delay of the main FIR DAC. This means a higher gain of the transfer function from u to y ($L_{0,ct}$) at high frequencies, thereby increasing STF peaking. Finally, on the practical front, increasing the number of taps necessitates more flip-flops, and result in a much smaller "unit DAC". The former increases switching power, while the latter results in increased area when resistive DACs are used. Bearing these constraints in mind, it appears that there is little to be gained by choosing an FIR-DAC with more than 10–15 taps.

To summarize, using a single-bit quantizer with FIR feedback combines the benefits of single-bit and multi-bit operation. The ADC design is power efficient, since only one comparator is used. The FIR DAC is inherently linear, even when the weights of the filter deviate from their nominal values. Thus, element mismatch does not result in distortion, unlike in a multi-bit DAC. Also, the FIR-DAC output waveform resembles that of a multi-bit DAC, and benefits from a small step size, just like a multi-bit CTΔΣM. While we have argued the advantages of FIR feedback assuming equal tap weights, one could do marginally better with taps optimized to minimize in-band noise due to jitter. An FIR DAC, therefore, is a very effective way of reducing jitter sensitivity. As we have seen above, given $F(z)$, the loop's NTF can be restored exactly. Finally, while a single-bit ADC with an FIR DAC is particularly useful in practice, FIR feedback can be applied to multi-bit ADCs as well.

9.6   Comparator Metastability

In our discussion on excess delay in CTΔΣMs, we assumed that the ADC needs a nonzero time to make a valid decision. It turns out that reality is a bit more complicated. We begin our discussion by studying the behavior of a 1-bit quantizer (the comparator). The output v of an ideal comparator is simply the sign of its input y. A practical comparator has offset, and a delay that depends on the magnitude of its input. Offset is the result of transistor and capacitor mismatch, and details of the latch construction. As we discuss below, the time taken by the comparator to give a valid output depends on the magnitude of the differential input. How does this influence the operation of the CTΔΣM?

Assuming an NRZ feedback DAC is used, its pulse width is modulated by the signal-dependent delay of the ADC. As a result, the in-band SNR is degraded, like when a jittery clock is used.
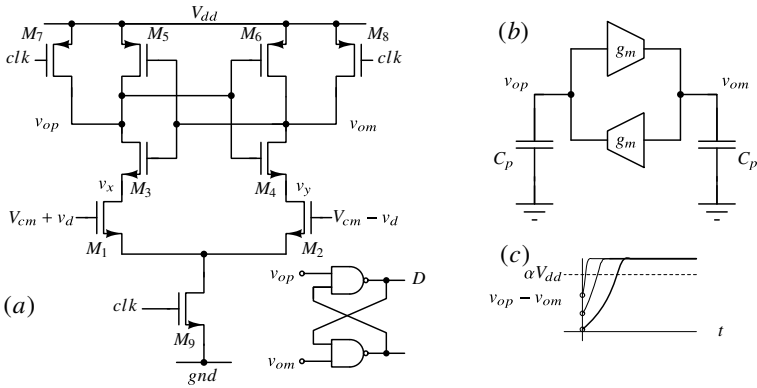
Figure 9.35   (a) The StrongARM latch. (b) Equivalent circuit during regeneration and (c) evolution of $(v_{op} - v_{om})$ during regeneration.

Figure 9.35(a) shows a commonly used comparator circuit, often referred to as a StrongARM latch [12]. When clk is low, $M_{7,8}$ pull v_{op} and v_{om} to the supply. When clk goes high, M_9 turns on. Subsequently, $M_{1,2}$ turn on and are initially in saturation. The differential current $g_m v_d$ developed during this phase creates an imbalance between v_x and v_y. $M_{3,4}$ are the next devices to turn on and are initially in saturation. The parasitic capacitances at $v_{op,om}$ differentially appear negative at $v_{x,y}$, causing a voltage gain at $v_{op,om}$. All the while, the common-mode of $v_{op,om}$ and $v_{x,y}$ keeps reducing. Finally, $M_{5,6}$ turn on and $M_{1,2}$ go into the triode region, and regeneration begins. The equivalent circuit during regeneration is shown in Figure 9.35(b). When the transistors are in saturation, the inverters can be modeled by transconductors. C_p represents the parasitic capacitance at the regenerating nodes. The difference $v_{op} - v_{om}$ evolves according to

$$v_{op}(t) - v_{om}(t) = (v_{op}(0) - v_{om}(0)) \exp\left(\frac{t}{\tau}\right), \tag{9.34}$$

where $\tau = C_p/g_m$. Eventually, the inverters saturate and v_{op}/v_{om} reach supply/ground depending on the sign of v_d.

The v_{op} and v_{om} are connected to an RS-latch, so that the decision made by the latch is held when *clk* is low. Functionally therefore, the latch samples the differential input $2v_d$ at the rising edge of *clk*, and determines its sign.

$v_{op} - v_{om}$ must be at least αV_{dd} for the RS-latch to recognize a change in the logical states of v_{op} and v_{om}. It is thus seen that the comparator needs more time to resolve a small v_d, and the delay is of the form

$$t_{delay} = \tau \ln\left(\frac{\alpha V_{DD}}{2\beta v_d}\right), \tag{9.35}$$

where β is the gain from v_d to $v_{op}(0) - v_{om}(0)$.
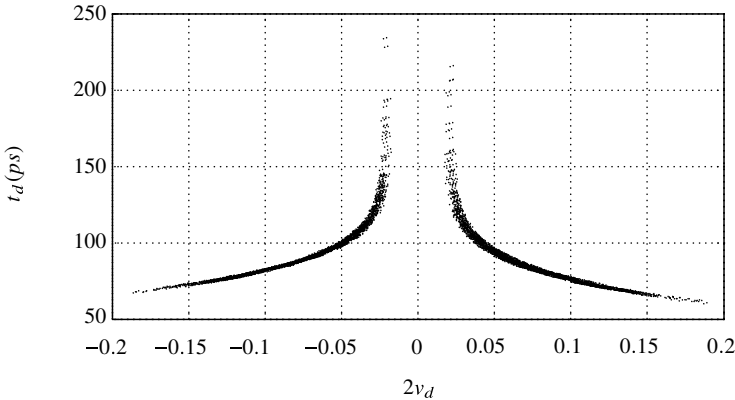
Figure 9.36   Comparator delay as a function of v_d.

Figure 9.36 shows the simulated delay of a comparator embedded in a second-order single-bit CT$\Delta\Sigma$M operating at 1 GS/s. On the *x*-axis is the quantizer input ($2v_d$) at the sampling instant, while the *y*-axis shows the corresponding decision delay. A couple of observations are in order. As expected, the delay increases as the magnitude of v_d decreases. Further, the variation in delay is larger, when v_d is smaller. The reason is the following. When the latch tracks the input, the differential current $g_m v_d$ is integrated on the parasitic capacitors for a small time interval that is determined by device details, as we described earlier. This integrated version of v_d is what is regenerated. Since v_d is the loop-filter's output, it varies with time – if it is large in magnitude, changes in $v_d(t)$ during the integration phase of the latch have only a small effect on $(v_{op}(0) - v_{om}(0))$. If, on the other hand, v_d is very small, the details of the waveform during the integration phase play a significant role on the voltage developed across the parasitic capacitances of the latch. For instance, as small v_d with a large negative slope is likely to result in a smaller $(v_{op}(0) - v_{om}(0))$ than one with a large positive slope. These differences in the developed $(v_{op}(0) - v_{om}(0))$ contribute to the larger spread of the comparator delay. While the analysis above described the mechanism of input dependent delay in a specific latch circuit, similar mechanisms come to play in all latches.

Referring to Figure 9.35(a), the output of the RS-latch D can be directly used to excite the DAC in a single-bit CT$\Delta\Sigma$M. Since the ADC's delay t_d is dependent on y, the edge of the DAC feedback waveform is modulated by the loop-filter in a nonlinear fashion. If an NRZ DAC is used, this error can be modeled by adding a thin pulse to the DAC output,

with its height being equal to the transition in the DAC output waveform, and a width t_d. As in the case of clock jitter, this error can be modeled by the additive sequence [5]

$$e_m[n] = \frac{t_d}{T_s}(v[n] - v[n-1]) \qquad (9.36)$$

at the DAC's input, as shown in Figure 9.37. The power of e_m in the signal band is given

$$e_m[n] = t_d f_s(v[n] - v[n-1])$$
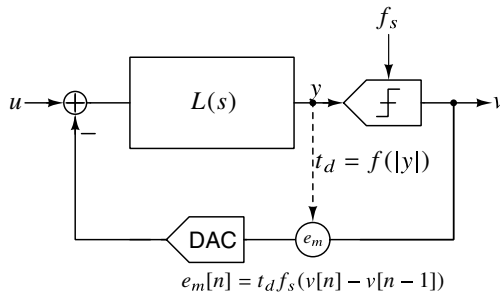
Figure 9.37   Modeling signal-dependent comparator delay in a 1-bit CTΔΣM with an NRZ feedback DAC.

by

$$J_m = \frac{4p}{OSR}\left(\frac{\sigma_{td}}{T_s}\right)^2, \qquad (9.37)$$

where $p \approx 0.8$ is the probability that D makes a transition. σ_{td}^2 represents the variance of the comparator's signal dependent delay, and is assumed to be uncorrelated with the transitions in the feedback waveform. For the comparator of Figure 9.35(a), whose delay versus input amplitude curve is shown in Figure 9.36, σ_{td} turns out to be about 18 ps. Assuming that $u = A\cos(2\pi f_{in}t)$, we can express the in-band SNR due to metastability as

$$SNR_{metastability} = 10\log\left(\frac{A^2 OSR \cdot T_s^2}{8p\sigma_{td}^2}\right). \qquad (9.38)$$

Figure 9.38 shows the PSD of CT-MOD2 with the StrongARM comparator of Figure 9.35(a), with $u = 0.1\cos(2\pi f_{in}t)$. The loop-filter is ideal. The signal-dependent delay of the comparator causes an increased low-frequency noise floor, and it dramatically degrades the in-band SNR. The SNR found from simulation is 27.9 dB. The PSD for a modulator with an ideal comparator is also shown for comparison. Using (9.38) with $A = 0.1$ and $\sigma_{td} = 18$ ps yields an estimate of 27.8 dB, in good agreement with results from transistor-level simulations.

We thus see that signal-dependent delay of the comparator could be a serious problem in CTΔΣMs designed for high speed/precision. A single-bit modulator is particularly bad in this respect, since an error in the DAC pulse width results in an error proportional to the modulator's full scale.

A way of overcoming this problem is to clock the DAC long enough after the ADC, so that the probability that the ADC has made a decision is very high. In other words, we could deliberately introduce excess delay into the ΔΣ loop so as to give enough time for
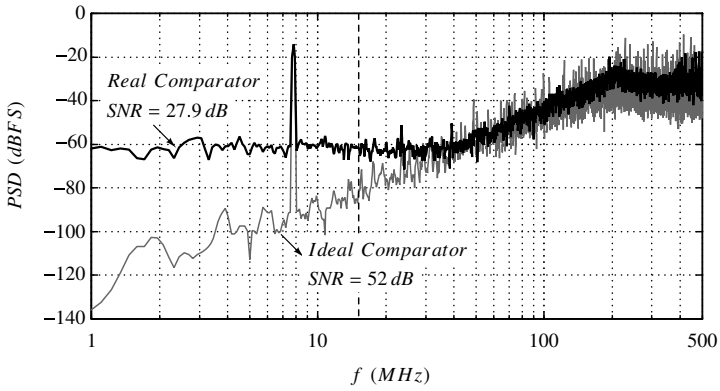
Figure 9.38   PSD of a second-order CT$\Delta\Sigma$M with ideal and real comparators: $f_s = 1$ GHz and OSR = 32.
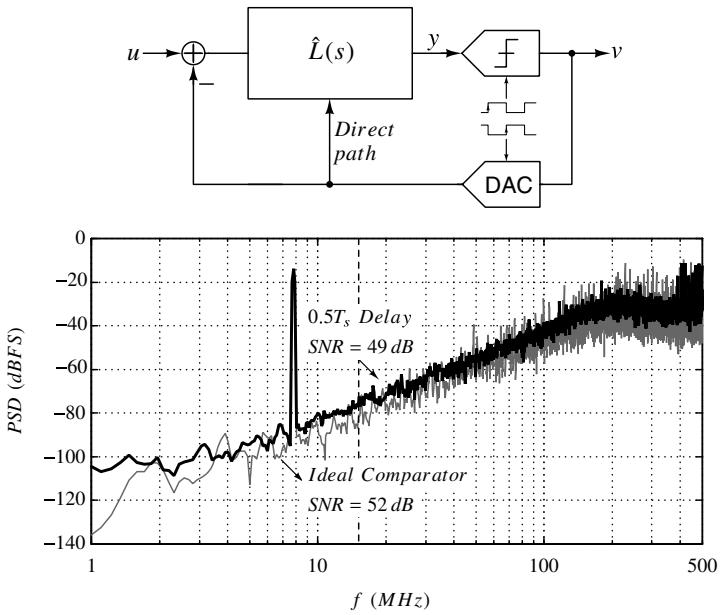
Figure 9.39   Mitigating comparator metastability by clocking the DAC after half-cycle delay.

the ADC to make a decision. This way, the input to the DAC, when it samples the ADC output, is very close to V_{dd} or ground. Consequently, the latch in the DAC has very little delay, which also implies a very small data-dependent jitter.

The deliberately introduced delay is not problematic, as the loop can be compensated, and its NTF restored by coefficient tuning and by introducing a direct path around the quantizer, as discussed earlier in this chapter. The question at hand is – how much time should the DAC wait, after the ADC has been clocked? A very convenient (and robust) choice is to clock the DAC a half-clock cycle later [13], as shown in Figure 9.39. As seen from the PSD in Figure 9.39, the in-band SNR is only 3 dB short of what can be achieved by an ideal comparator.

Other strategies that mitigate the effect of signal-dependent comparator delay observe that since metastability manifests in a manner similar to clock jitter, techniques that address the latter will also be effective for the former. A case in point is the use of a multi-bit quantizer, which is beneficial on two counts. First, since the OSR needed to achieve a desired SQNR over a given signal bandwidth is reduced, jitter due to metastability is a smaller fraction of the clock period. Further, only the output of the comparator "closest" to y experiences a delay different from that of the other comparators. Thanks to multi-bit operation, this output can influence only a fraction of the feedback waveform, alleviating problems due to signal-dependent delay. Similarly, using an impulsive DAC (implemented by using switched-capacitor techniques) also addresses comparator metastability, since the amount of charge delivered into the loop-filter is independent of the ADC delay.
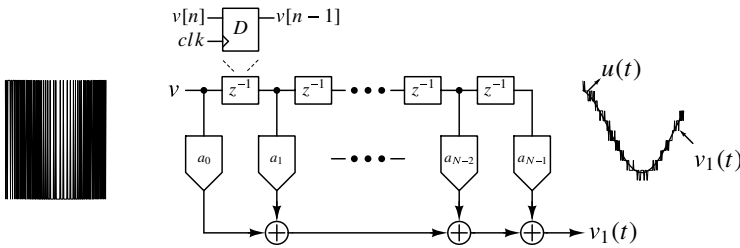
Figure 9.40   An FIR DAC mitigates the effect of comparator metastability due to the cascade of latches, as well as by reducing the contribution of an individual latch to $v_1(t)$.

The FIR feedback DAC [9, 10], as we have seen, is a attractive way of mitigating the effect of clock jitter. It should, therefore, be expected to address the problem of comparator metastability. The chain of latches are particularly effective in reducing signal-dependent delay, as illustrated in Figure 9.40. v is the output of the single-bit quantizer, and exhibits significant signal-dependent delay. However, since $v[n]$ is weighted by $a_0(<1)$, its contribution to $v_1(t)$ is reduced by this factor. One could deliberately choose $a_0 = 0$, which is equivalent to introducing a 1-cycle delay into the loop, to avoid direct dependence on $v[n]$. Since $v[n-1], \cdots, v[n-N+1]$ are derived from $v[n]$ through a chain of flip-flops, successive regeneration renders data-dependent jitter a non-issue.

9.7   Conclusions

In this chapter, we examined the influence of the primary nonidealities on the performance of CT$\Delta\Sigma$Ms, namely excess loop delay, time-constant variations in the loop-filter, clock jitter, and comparator metastability. Excess delay, as in any feedback loop, can render the modulator unstable. However, this can be addressed by a combination of coefficient tuning and adding a direct path around the quantizer.

Time-constant variations in the loop-filter modify the NTF and can spell potential trouble, but these can be (easily) addressed by RC-tuning loops.

Clock jitter is a serious concern in CT$\Delta\Sigma$Ms (significantly more so than their discrete-time counterparts). We gained an intuitive understanding of the mechanisms by which clock jitter degrades the performance of a CT$\Delta\Sigma$M and examined various approaches that address the effects of jitter. The use of an NRZ DAC pulse, along with FIR feedback was seen to be a good way of addressing this problem. The loop can be compensated for delay introduced by the FIR DAC, and we saw that the NTF can be restored exactly by coefficient tuning, and through the use of a direct path compensating FIR DAC.

The signal-dependent delay of the comparator degrades in-band SNDR in a manner similar to clock jitter. We described various ways of mitigating this problem.

References

[1] J. Cherry and W. M. Snelgrove, "Excess loop delay in continuous-time delta-sigma modulators," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 46, no. 4, pp. 376–389, 1999.

[2] S. Pavan, "Excess loop delay compensation in continuous-time delta-sigma modulators," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 55, no. 11, pp. 1119–1123, 2008.

[3] S. Pavan, "Continuous-time delta-sigma modulator design using the method of moments," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, no. 6, pp. 1629–1637, 2014.

[4] J. A. Cherry, *Theory, Practice, and Fundamental Performance Limits of High Speed Data Conversion using Continuous-time Delta-Sigma Modulators*. Ph.D. dissertation, Carleton University, 1998.

[5] J. Cherry and W. M. Snelgrove, "Clock jitter and quantizer metastability in continuous-time delta-sigma modulators," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 46, no. 6, pp. 661–676, 1999.

[6] R. Adams and K. Q. Nguyen, "A 113-dB SNR oversampling DAC with segmented noise-shaped scrambling," *IEEE Journal of Solid-State Circuits*, vol. 33, no. 12, pp. 1871–1878, 1998.

[7] M. Ortmanns, F. Gerfers, and Y. Manoli, "A continuous-time $\Sigma\Delta$ modulator with reduced sensitivity to clock jitter through SCR feedback," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 52, no. 5, pp. 875–884, 2005.

[8] S. Pavan, "Alias rejection of continuous-time modulators with switched-capacitor feedback DACs," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 58, no. 2, pp. 233–243, 2011.

[9] B. M. Putter, "$\Sigma\Delta$ ADC with finite impulse response feedback DAC," in *Digest of Technical Papers, IEEE International Solid-State Circuits Conference (ISSCC)*, pp. 76–77, 2004.

[10] O. Oliaei, "Sigma-Delta modulator with spectrally shaped feedback," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 50, no. 9, pp. 518–530, 2003.

[11] D. K. Su and B. A. Wooley, "A CMOS oversampling D/A converter with a current-mode semidigital reconstruction filter," *IEEE Journal of Solid-State Circuits*, vol. 28, no. 12, pp. 1224–1233, 1993.

[12] A. Abidi and H. Xu, "Understanding the regenerative comparator circuit," in *Proceedings of the IEEE Custom Integrated Circuits Conference (CICC)*, pp. 1–8, IEEE, 2014.

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

CHAPTER 10

CIRCUIT DESIGN FOR CONTINUOUS-TIME DELTA-SIGMA MODULATORS

In the chapters so far, we have covered the architectural aspects of continuous-time $\Delta\Sigma$ modulators. We now know how to choose an NTF and the oversampling ratio to achieve a desired in-band SQNR, and how to pick an appropriate loop-filter topology. We understand the effect of practical nonidealities like excess delay, time-constant variations and quantizer metastability on modulator performance, and how to mitigate these problems. In this chapter, we explore circuit design techniques for the various building blocks of CT$\Delta\Sigma$Ms. As one should expect, a practical implementation will most likely introduce "new" nonidealities. What should we expect these nonidealities to be?

The building blocks will be realized using transistors, which need time to operate, and which are fundamentally noisy and nonlinear. First, this means that the finite delay associated with the transistors introduces undesired poles and zeros in the loop-filter transfer function, which will modify the NTF. Next, thermal and flicker noise of the transistors introduce noise over and above that due to quantization. Finally, the loop-filter, which is expected to be perfectly linear, is no longer so. As we see later in this chapter, this can significantly degrade the in-band signal-to-noise ratio.

While a thorough awareness of the underlying theory is fundamental, understanding and mitigating implementation-related nonidealities is key to realizing a CT$\Delta\Sigma$M that works as intended. This chapter is focused on the design of the circuit blocks needed to realize a CT$\Delta\Sigma$M, their main nonidealities, and what one can do to mitigate them.

10.1 Integrators

Integrators can be implemented in many styles. Figure 10.1 shows three popular alternatives that realize an inverting integrator. The figure shows single-ended circuits – in reality, most signal paths are realized in fully differential form. These diagrams should be interpreted as the single-ended equivalent of a fully differential realization.
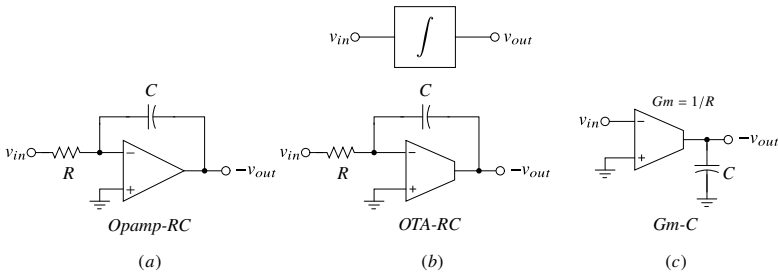
Figure 10.1 Three techniques to realize an integrator.

Figure 10.1(a) shows an active-RC integrator. Assuming an ideal opamp,

$$V_{out}(s) = -\frac{1}{sCR}V_{in}(s). \tag{10.1}$$

What are the good attributes of such an integrator? If the opamp is ideal, its inverting terminal is a virtual ground. v_{in} is thus converted into a current v_{in}/R in a very linear fashion, thanks to the linearity of R. If C is linear, then v_{out} is linearly related to v_{in}; in other words, if the opamp is ideal, a perfectly linear integrator is possible. Parasitic capacitances, which are bound to exist at every node, are harmless due to the following. A parasitic at the output of the opamp does not affect v_{out}, since the opamp is a voltage controlled voltage source whose output impedance is zero. Parasitic capacitance at the virtual ground node has no consequence either, since the voltage across it does not vary. Further, since the output impedance of the integrator is zero, driving subsequent integrators is not a problem.
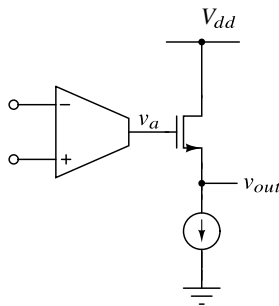
Figure 10.2 The limited swing problem in a CMOS opamp that achieves a low output impedance.

If the active-RC structure has all that one can wish for in an integrator, what is the need to invent other topologies, like those in Figure 10.1(b) and (c)? It turns out that designing an opamp that achieves a low output impedance is problematic, as illustrated in Figure 10.2. The figure shows a conceptual design of an opamp with a low output impedance, achieved by using a common-drain output stage. We denote the threshold and overdrive of all transistors (assumed identical) by V_T and ΔV, respectively. v_a can go as high as $V_{dd} - \Delta V$ before the transconductor goes into the triode region. This means that v_{out} can go as high as $V_{dd} - V_T - 2\Delta V$ while maintaining all devices in saturation (needed to achieve high gain). On the lower side, the least v_{out} can be is ΔV, below which the current source biasing the output stage is driven into the linear region. The peak-to-peak swing of v_{out}, therefore, is $V_{dd} - V_T - 3\Delta V$. If we assume that $V_{dd} = 1.2\,\text{V}$, $V_T = 0.5\,\text{V}$, and $\Delta V = 100\,\text{mV}$, this turns out to be $400\,\text{mV}$. It is thus seen that the common-drain stage, needed to achieve the low output impedance needed in the opamp, severely restricts its output swing. A reduced integrator swing necessitates larger capacitor values than would otherwise be necessary. More important, since the swing at the loop-filter output is reduced, the step size of the ADC is now smaller, which complicates its design. Thus, an opamp-RC integrator, while having several attractive properties, also has significant disadvantages. This is where the OTA-RC integrator, where the opamp is replaced by an operational transconductance amplifier (OTA), scores.

An ideal OTA is a voltage-controlled current source, whose transconductance is infinite. Thus, the virtual ground node of the OTA-RC integrator, shown in Figure 10.1(b), is zero (as in the opamp-RC case). This means that the integrator is linear (assuming linear passives). The output impedance of the integrator is zero – in this case, achieved through the use of strong negative feedback. Thanks to this, the integrator is insensitive to parasitic capacitances, and can drive other integrators. Finally, since a common-drain amplifier is not needed, the output can swing to within ΔV of the rails. The peak-to-peak swing possible is thus $V_{dd} - 2\Delta V$, which is a big improvement when compared to the opamp-RC structure.[1]

The impressive performance of an OTA-RC integrator is achieved through negative feedback. By nature, this limits the bandwidth of operation to a fraction of the intrinsic speed of the OTA. The transconductance-capacitance (or Gm-C) integrator attempts to mitigate the speed issue by using an open-loop structure, as shown in Figure 10.1(c). Here, v_{in} is converted into a current by a transconductor G_m (chosen to be $1/R$), which is integrated on the capacitor C, to generate the output voltage. The input impedance of the integrator is infinite – so cascading integrators is easy.

Unfortunately, the Gm-C structure is problematic on several fronts. There are many, many ways of implementing a transconductor. Those that use open-loop techniques have limited linearity, which is strongly dependent on device characteristics. Others use feedback to linearize the transconductor, which reduces speed of operation. Further, the integrator is sensitive to parasitic capacitances. Gm-C integrators are useful only when speed is the primary consideration, with linearity being less important.

---

[1]Since the opamp-RC integrator severely restricts output swing, an opamp in the sense of Figure 10.1(a) is rarely used. The "opamp" name and symbol are often (mis)used, in the sense that one actually means "OTA" when talking about an opamp. We are guilty of the same indiscretion.

From the discussions above, the OTA-RC structure appears to be the best choice, and as such warrants a more detailed study. We begin with the simplest possible OTA structure, namely the single-stage OTA.

10.1.1   The Single-Stage OTA-RC Integrator

(a)   (b)
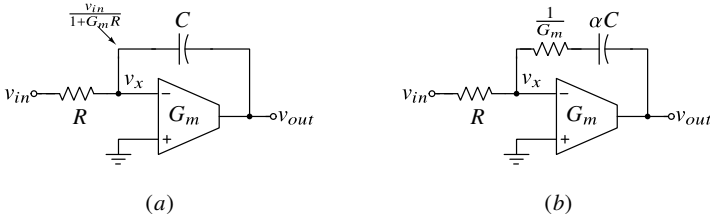
Figure 10.3   (a) A single-stage OTA-RC integrator and (b) use of a zero-canceling resistor to eliminate the RHP zero.

Figure 10.3(a) shows an OTA-RC integrator based on a single-stage OTA. The OTA has a finite G_m. Other nonidealities like parasitic capacitances at the input and output, and finite output resistance are neglected. Unlike in the ideal case, the virtual "ground" node is no longer at ground. Straightforward analysis shows that

$$v_x = \frac{v_{in}}{1 + G_m R} \tag{10.2}$$

and

$$\frac{V_{out}(s)}{V_{in}(s)} = - \underbrace{\frac{\alpha}{sCR}}_{\substack{\text{unity-gain} \\ \text{freq. shift}}} \underbrace{\left(1 - \frac{sC}{G_m}\right)}_{\text{RHP zero}}, \tag{10.3}$$

where $\alpha = G_m R/(G_m R + 1)$. The finite G_m causes a shift in the unity-gain frequency, reducing it by a factor α. Further, we see that the transfer function of the integrator has a zero in the right-half plane (RHP). Intuitively, this is due to multiple paths from the input to the output – one through the transconductor, and the other through the integrating capacitor. The RHP zero adds phase lag, which is equivalent to adding excess loop delay.

The shift in the unity-gain frequency can be addressed by modifying the capacitor or resistor values by α. The RHP zero can be eliminated by introducing a resistor of value $1/G_m$ in series with the integrating capacitor. The result, shown in Figure 10.3(b), achieves a transfer function $(-1/sCR)$.

The single-stage OTA has some practical problems. The output conductance of the transistors results in an integrator with finite dc gain, which reduces further with loading. Further, the simplest possible transistor-level implementation of a single-stage OTA, namely the differential pair, has a limited output swing. Figure 10.4 shows the simplified schematic of a fully differential single-stage OTA based on the differential pair. Since integrators in the loop-filter need to be cascaded, the input and output common-mode voltages have to be the same. This means that v_o cannot exceed the threshold voltage of the NMOS transistor, no matter how high the supply, which is a severe limitation. Finally, as we will see later in this chapter, nonlinearity can be a significant problem unless $G_m R$ is very large.

Simply increasing G_m to achieve linearity is not power efficient. A way of addressing the limitations of a single-stage OTA is to use a two-stage design, which we discuss next.
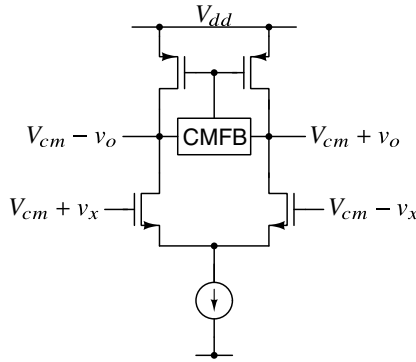
Figure 10.4   Fully differential single-stage OTA based on a differential pair.

10.2   The Miller-Compensated OTA-RC Integrator
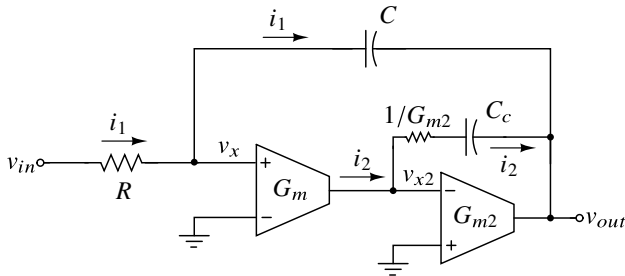
Figure 10.5   An OTA-RC integrator employing a two-stage Miller-compensated OTA.

A Miller-compensated OTA can be used in place of a single-stage OTA, as shown in Figure 10.5. A second transconductor G_{m2} is cascaded with the first. For stability purposes, a compensating capacitor C_c is placed across G_{m2}, and the resulting RHP zero is canceled by the resistor $1/G_{m2}$ in series with C_c.

Why is the Miller-compensated OTA an improvement over a single-stage OTA? For one, the dc gain of the integrator is due to two gain stages, and is therefore much higher. The common-mode voltage of the internal node is not constrained in any way, unlike in a one-stage OTA. Thanks to this, the output can swing to within one overdrive voltage of the rails. Further, the use of cascodes at the output of the first stage improves gain without degrading the maximum swing at the output of the second stage. Since the dc gain that can be achieved with a two-stage OTA is inherently more than in a single-stage design, cascading integrators is not as problematic.

How does the performance of an integrator with a Miller-compensated design compare with that which uses a single-stage OTA? We give an intuitive explanation below.

Referring to Figure 10.5, v_x and v_{x2} must be very small. This means that the voltages across C and C_c are approximately the same. Denoting the current through C by i_1, it follows that the current through the compensating capacitor is

$$i_2 = i_1 \frac{C_c}{C}. \tag{10.4}$$

Since $i_1 \approx v_{in}/R$, and i_2 is the output current of the first stage, it follows that

$$v_x \approx \frac{v_{in}}{G_m R} \frac{C_c}{C}. \tag{10.5}$$

With a single-stage OTA, recall that $v_x \approx v_{in}/(G_m R)$. It is thus seen that using a Miller-compensated OTA can be thought of as using a single-stage OTA whose G_m is higher by a factor C/C_c. Another way of interpreting the result above is the following. For the integrator to be ideal, v_x must be zero. From (10.5), this means that $G_m R(C/C_c) \gg 1$. Thus,

$$\underbrace{\frac{G_m}{C_c}}_{\text{OTA's UGB}} \gg \underbrace{\frac{1}{RC}}_{\text{Integrator's UGB}}, \tag{10.6}$$

which is intuitively satisfying.

From our approximate analysis, we concluded that the benefits of using a Miller-compensated OTA are in inverse proportion to C_c. It is thus tempting to set it to zero. This is not practical, however, because our analysis neglected parasitic capacitances at the outputs of the first and second stages. These parasitics will degrade the phase margin (or render the integrator unstable) unless C_c is made sufficiently large. To first-order, the transfer function of the integrator based on a Miller-compensated OTA can be shown to be

$$\frac{V_{out}(s)}{V_{in}(s)} \approx -\frac{1}{sCR} \frac{1}{\left(1 + \frac{sC_c}{G_m}\right)}. \tag{10.7}$$

The effect of finite OTA bandwidth is to add an extra pole to the integrator transfer function. If parasitic capacitors not modeled in Figure 10.5 are considered, more poles and zeros make their appearance in the transfer function. Further, the output resistance of G_m and G_{m2} results in an integrator with a finite dc gain. It is thus seen that fixing the problems associated with a one-stage OTA causes the integrator transfer function to become a high-order one. A natural question that arises is how all these poles influence the loop's NTF, and what one can do about it. This, and related issues, are discussed in Section 10.8.

10.3   The Feedforward-Compensated OTA-RC Integrator

A two-stage OTA can also be compensated using feedforward. The basic idea is shown in Figure 10.6. Rather than add a compensating capacitor across the second stage, as in Miller compensation, a third transconductor G_{m3} senses v_x and pumps current into the output node. This provides the "fast path" of the feedback loop. The cascade of G_m and G_{m2} forms the high dc-gain (and slow) path. Thanks to the two-stage design, the integrator has high dc gain, as in the Miller-compensated OTA. At first sight, it might seem that the feedforward transconductor increases power dissipation. It turns out that G_{m3} can be implemented by reusing the bias current needed anyway to realize G_{m2}.

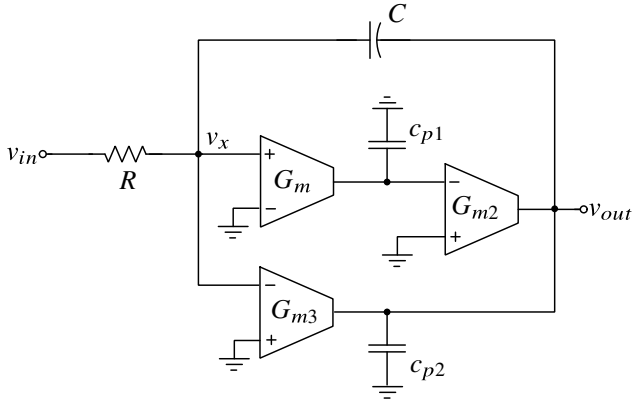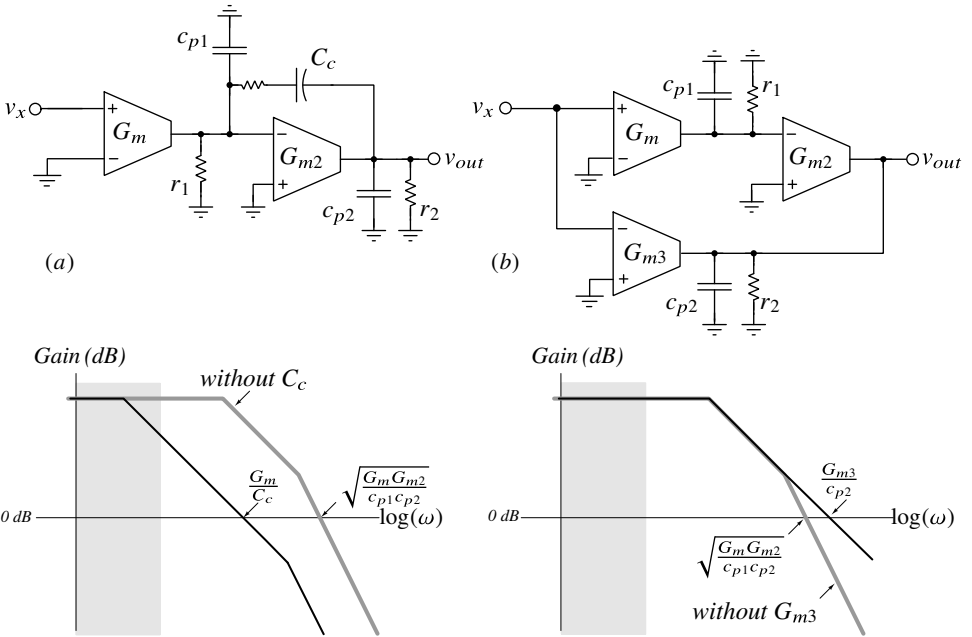**Figure 10.6**   An active-RC integrator using a feedforward-compensated OTA.

Figure 10.7   Comparison of (a) two-stage Miller-compensated and (b) feedforward-compensated OTAs. r_1 and r_2 model the output resistances of the transconductors.

How does a feedforward-compensated OTA fare in comparison with a Miller-compensated one? Without compensation, the frequency responses of both designs are the same. As shown in Figure 10.7, their unity-gain frequencies are $\sqrt{G_m G_{m2}/c_{p1} c_{p2}}$. With Miller compensation, C_c has to be chosen so that the unity-gain frequency G_m/C_c is smaller than the pole due to the second stage, which is approximately at $G_{m2}/(c_{p1} + c_{p2})$. The open loop-gain of the OTA, therefore, begins to roll off at 20 dB/dec at a frequency much lower than G_m/C_c. With feedforward compensation, G_{m3} should be chosen such that the magnitude response rolls of at 20 dB/dec when it crosses 0 dB. The unity-gain frequency of the compensated amplifier is G_{m3}/c_{p2}, which is higher than $\sqrt{G_m G_{m2}/c_{p1} c_{p2}}$. From Figure 10.7, it is apparent that the feedforward-compensated structure achieves a much larger bandwidth (for the same power dissipation) when compared to its Miller-compensated counterpart. This makes sense, since the philosophy behind achieving stability by Miller compensation is to *slow down* the first stage by adding C_c. Deliberately adding a large capacitor and charging/discharging it increases the Miller OTA's power dissipation.

Given that feedforward compensation achieves a much higher bandwidth for the same power dissipation, why is it not used more often? For instance, it is the Miller OTA that is the workhorse of discrete-time $\Delta\Sigma$ modulators. Graduate courses on analog integrated circuits, where one is introduced to opamp design, analyze Miller-compensated OTAs in great depth. Feedforward, however, is given short shrift, if covered at all. Most commercial discrete opamps are also Miller-compensated. It is only natural to wonder why this is so.

Since feedforward causes zeros in the transfer function of the integrator, it results in slow-settling transients. This is generally unacceptable in switched-capacitor circuits, where the samples of the OTA output are of interest. A Miller OTA does not have such problems, though it is slower than a feedforward OTA – making it an appropriate choice for use in a discrete-time $\Delta\Sigma$ modulator. In a CT$\Delta\Sigma$M, the entire output waveform is relevant, and slowly settling transients do not matter. This means that feedforward OTAs can be used.

Observing the responses in Figure 10.7 also reveals that if the OTAs are enclosed in a negative feedback loop, the Miller amplifier becomes less stable as the feedback factor is increased. Unity feedback thus forms the "worst case" for such an OTA. However, the feedforward structure becomes less stable when the feedback factor is *reduced* – and a feedback factor of 1 results in the most stable amplifier. This makes a feedforward-compensated opamp less suitable as a general-purpose discrete device.

While we discussed a second-order feedforward-compensated OTA, high-order structures can be conceived as well [1]. An example is shown Figure 10.8(a). The transfer function is

$$A(s) = \frac{G_{m1}}{sC_{p1}} + \frac{G_{m2}G_{m3}}{s^2 C_{p1}C_{p2}} + \frac{G_{m2}G_{m4}G_{m5}}{s^3 C_{p1}C_{p2}C_{p3}} + \cdots . \tag{10.8}$$

For stability, the magnitude response must cross the 0-dB level at about 20 dB/decade, like in the second-order case. The advantage with multi-stage feedforward is that the transition region of the magnitude response can be tailored to be narrow, allowing high gain to be maintained over a wider bandwidth before rolling off, as seen in Figure 10.8(b). As with all high-order systems, such OTAs are conditionally stable. When the modulator is over-driven, stages can saturate and precipitate instability due to overloading of internal stages. While careful and extensive simulations are needed to ensure stability over all operating
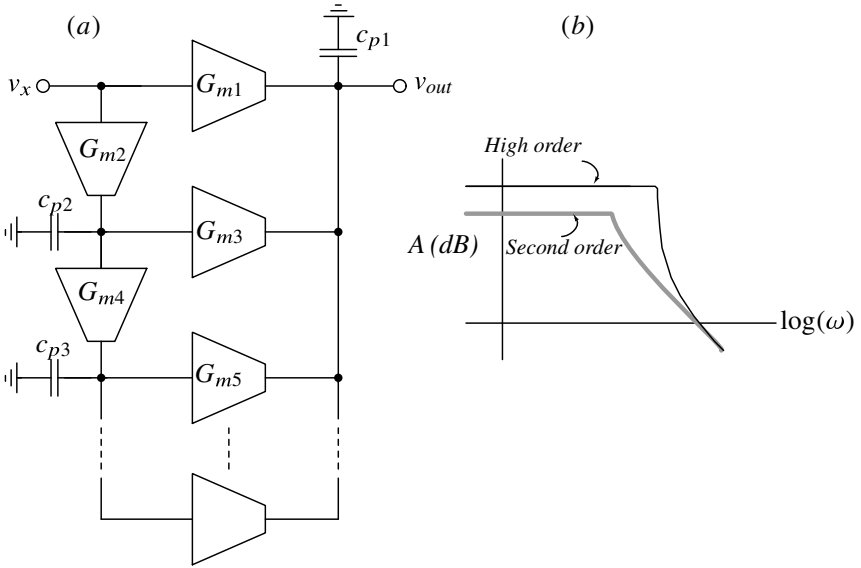
Figure 10.8   (a) A multipath feedforward OTA and (b) magnitude responses of second and high-order OTAs.

conditions, a useful practice is to ensure that the first-order path saturates last. This way, saturation of internal nodes will still result in a stable system as it recovers from overload.

10.4   Stability of Feedforward Amplifiers

When a feedforward OTA is enclosed in a loop with unity-gain feedback, the loop-gain function is of the form

$$LG(s) = \frac{k_1}{s} + \frac{k_2}{s^2} + \cdots + \frac{k_n}{s^n}. \tag{10.9}$$

The magnitude and phase plots of the loop-gain function for a third-order example (with $k_1 = 2.5, k_2 = 0.5, k_3 = 0.1$) are shown in Figure 10.9. Around the unity-gain frequency, the magnitude response rolls off at 20 dB/decade, and the phase lag is 90°. The phase margin for this particular design is, therefore, almost 90°. Interestingly, at $\omega_1 = 0.2$ rad/s, the phase lag of the loop-gain function is 180°, and its magnitude is 12.5 (which is $\gg 1$).

The fact that the system is stable when $|LG|$ is (much) greater than unity with $\angle LG = 180°$ is somewhat disorienting. After all, we know from Barkhausen's criterion that a feedback system is unstable if $|LG| = 1$ and $\angle LG = 180°$. Our situation *seems* much worse, with $|LG| \gg 1$ when the loop-gain's phase is 180°. Yet, we *know* that the closed-loop system is stable (as established using the Nyquist criterion or root-locus methods). How do we resolve this paradox?

First, we examine the intuition behind the Barkhausen criterion. Consider the feedback loop shown in Figure 10.10(a). Assume that at ω_1, $LG(j\omega_1) = 1$ and $\angle LG(j\omega_1) = 180°$. To understand why the system is unstable, we perform a *gedanken* experiment. We replace the adder of Figure 10.10(a) with a one-pole two-throw switch, as shown in part
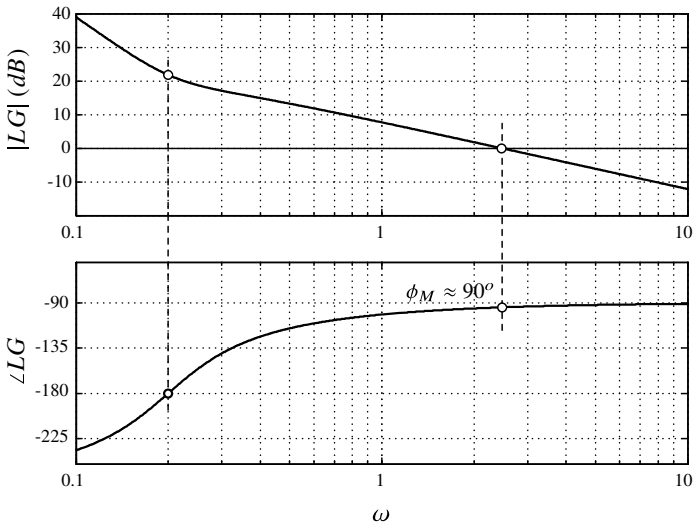
Figure 10.9   Magnitude and phase responses of the loop-gain of an example third-order feedforward-compensated loop.
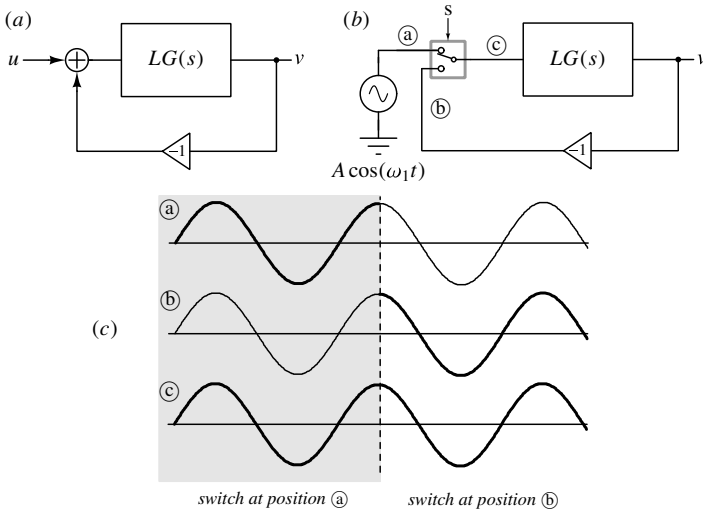
Figure 10.10   (a) Feedback system. (b) Loop is initially opened, and excited at a frequency ω_1 for which $|LG(j\omega_1)| = 1$ and $\angle LG(j\omega_1) = 180°$. The position of the switch is then changed to close the loop, and the system oscillates at ω_1. (c) Waveforms at ⓐ, ⓑ, and ⓒ.

(b) of the figure. Initially, the switch is at position ⓐ, where it is excited by a sinusoid $A\cos(\omega_1 t)$. Since $LG(j\omega_1) = -1$, the sinusoid at ⓑ, in steady state, is also $A\cos(\omega_1 t)$, which is exactly identical to the signal at ⓐ, as shown in Figure 10.10(c). The position of the switch is then changed to ⓑ. As far as the amplifier is concerned, it makes no difference whether its input excitation is from the independent source or from to its own (inverted) output (since the two outputs are indistinguishable). Thus, the system continues to oscillate at ω_1 even after the switch has changed state.
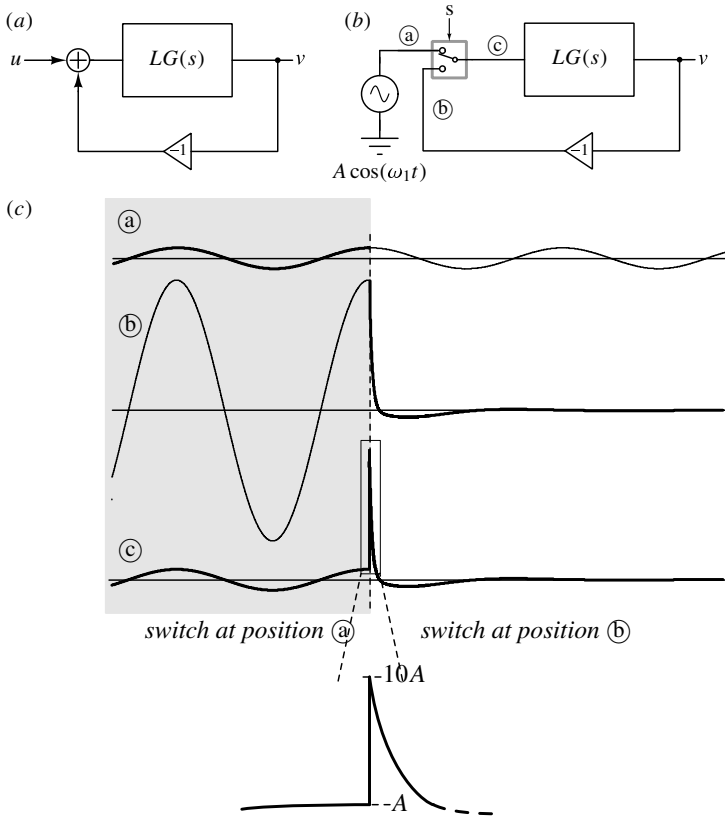
Figure 10.11   (a) Feedback system. (b) Loop is initially opened, and excited at a frequency ω_1 for which $|LG(j\omega_1)| = 12.5$ and $\angle LG(j\omega_1) = 180°$. The position of the switch is then changed to close the loop – the loop is eventually "quenched". (c) Waveforms at ⓐ, ⓑ, and ⓒ. A system with $|LG(j\omega_1)| \gg 1$ and $\angle LG(j\omega_1) = 180°$ can be stable if the "fast" feedback is sufficiently strong.

Based on the arguments above, we begin to wonder how it is even possible for the system to be stable when the loop-gain is *greater* than 1, when its phase is 180°. Referring to Figure 10.11(b), when the switch is at position ⓐ, the signal at ⓑ in our example system is amplified 12.5 times. Thus, the signals at ⓐ,ⓑ, and ⓒ are as shown in gray in Figure 10.11. One would then be tempted to think that the amplitude of the sinusoid in the loop would grow without bound when the loop is closed. After all, the fedback signal is 12.5 times the input, and this should regeneratively build up to infinity as it circulates

around the loop. However, we *know* that this is not so. What then, is the fallacy in our argument?

The key point is to realize the implication of saying $LG(j\omega_1) = 12.5\angle180°$. This statement means that if the open loop system is excited by a sinusoid at ω_1, the output is -12.5 times larger **in steady state**. At the risk of sounding redundant, we emphasize that it **does not** mean that the output will be $-12.5\times$ the input instantaneously. In our example, the moment we throw the switch to position ⓑ, the input at ⓒ experiences a step jump (from A to $10A$) as shown in Figure10.11(c). The instantaneous response of the amplifier will be dominated by the first-order path which has a transfer function $2.5/s$. Thus, the output at v immediately after the switch changes state is a ramp, that after inversion causes the signal at ⓑ to *reduce*, as seen in Figure10.11(c). Note that this is in the opposite direction of what we would have concluded if we assumed that steady state behavior was reached instantaneously. The reduction in the amplifiers output is fed back to the input, and the sinusoid that was circulating in the loop is "quenched" and dies down to zero, as it should in a stable system without a source. The quick feedback in the right direction is due to the high gain of the first-order path of the loop-gain. If this is not high enough, the correction (after the switch is thrown to ⓑ) would not be enough, and the loop would break into oscillation. This is consistent with the frequency domain argument that reducing the gain of the $1/s$ path will result in a smaller phase margin, and will eventually cause instability.

10.5   Device Noise in Continuous-Time Delta-Sigma Modulators

The components of the loop-filter, namely resistors and transistors, inject thermal (and $1/f$) noise into the CT$\Delta\Sigma$M. We first examine the noise of the building block of the loop-filter – namely the integrator. Figure10.12 shows the noise sources in an active-RC integrator that
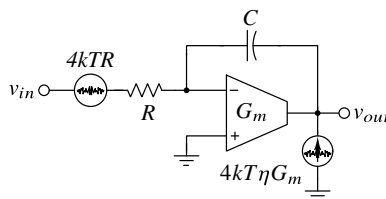
Figure 10.12   Noise sources in an active-RC integrator that uses a single-stage OTA.

uses a single-stage OTA. The output current noise spectral density of the OTA is $4kT\gamma G_m$, where γ is dependent on design details. The input-referred noise voltage spectral density of the integrator can be shown to be

$$S_v(f) \approx 4kT\left(R + \frac{\gamma}{G_m}\right). \tag{10.10}$$

Recalling that $G_mR \gg 1$ for a good integrator, it follows that $S_v \approx 4kTR$. As we will see going forward, we need to worry only about the in-band noise spectrum. What does it cost to reduce the in-band thermal noise of the integrator by 3 dB? R must be reduced by a factor of 2. For everything else to remain the same, G_m and C should increase by the same factor. Since all impedances have reduced by $2\times$, power dissipation increases twofold.
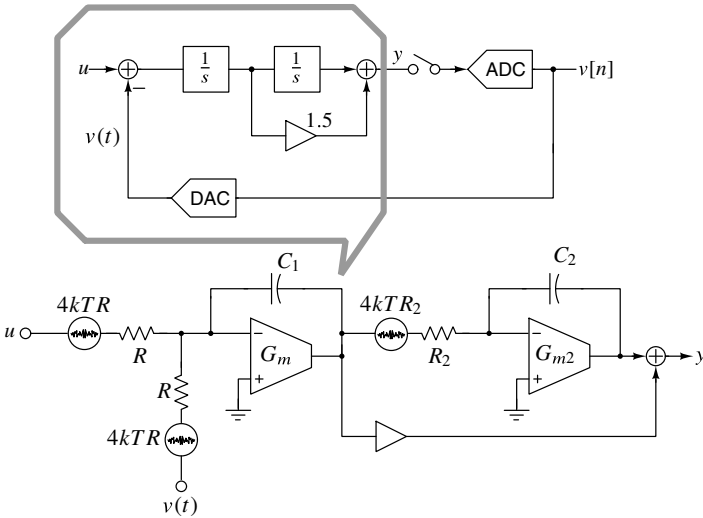
Figure 10.13 An example second-order CIFF CTΔΣM with noise sources. A resistive NRZ feedback DAC is assumed.

How does the noise of the loop-filter affect the in-band SNR of the modulator? We illustrate this with a second-order CIFF example, shown in Figure 10.13. The active-RC integrators use single-stage OTAs. The feedback DAC is assumed to be resistive. The effect of all noise sources can be referred to the input of the modulator and consolidated into an equivalent noise voltage, whose spectral density we denote by $S_{eq}(f)$. The equivalent model of the modulator, assuming additive quantization noise, is shown in Figure 10.14. The STF has a dc gain of unity; and nulls at multiples of f_s, thanks to the CTΔΣM's inherent anti-aliasing property. The peaking in the STF is due to the CIFF loop-filter architecture.
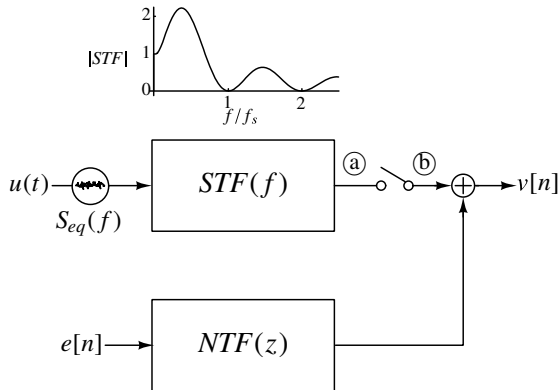
Figure 10.14 Equivalent CTΔΣM model including thermal and quantization noise sources. $S_{eq}(f)$ represents the equivalent input referred noise spectral density of the loop-filter and feedback DAC.
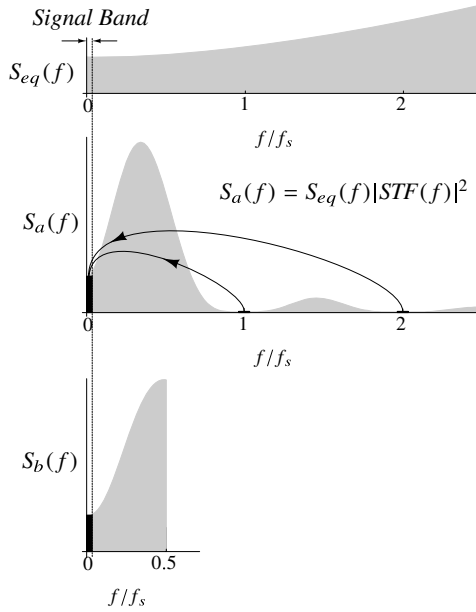
Figure 10.15   Noise spectral densities at various points in Figure 10.14.

Figure 10.15 shows the noise spectral densities at various points in Figure 10.14. At point ⓐ, $S_{eq}(f)$ has been shaped by the STF, and the resulting noise density is

$$S_a(f) = S_{eq}(f)|STF(f)|^2. \tag{10.11}$$

Since the continuous-time output at ⓐ is sampled to yield the sequence at ⓑ, it follows that the thermal noise component of $v[n]$ is

$$S_b(f) = f_s \sum_{k=-\infty}^{\infty} S_a(f - kf_s). \tag{10.12}$$

We are only interested in the in-band noise spectral density, since out-of-band noise will be eliminated by the decimation filter anyway. From Figure 10.15, we see that noise at ⓐ from around multiples of f_s will alias into the signal band after sampling at ⓑ. However, due to the inherent anti-aliasing property of the CTΔΣM, noise around multiples of f_s is virtually eliminated by the action of the STF. Thus, sampling does not increase the in-band noise, even though there is aliasing. As far as thermal noise calculations are concerned, therefore, only $S_{eq}(f)$ in the signal band is relevant.

As a corollary, if the STF did not have nulls around multiples of f_s, we would have to account for noise that folds into the signal band from around these frequencies. The STF will not have nulls around f_s when the loop-filter is time varying, which is the case when a switched-capacitor feedback DAC is used.

Coming now to the specific case of our CIFF CTΔΣM of Figure 10.13, the input-referred noise spectral density at low frequencies (neglecting noise from the OTAs) is given

by

$$S_{eq}(f) \approx \underbrace{8kTR}_{\substack{input\ and\ DAC \\ resistors}} + \underbrace{4kTR_2(2\pi f RC_1)^2}_{\substack{second\ integrator's\ noise \\ shaped\ by\ gain \\ of\ the\ first\ integrator}}. \tag{10.13}$$

From the expression above, it is clear that the noise due the second integrator becomes negligible when referred to the modulator's input due to the high gain of first integrator in the signal band. This means that the rest of the loop-filter can be impedance scaled with no consequence on the in-band thermal noise, thereby reducing power dissipation. Another implication of this observation is that increasing the order of the NTF can be done with very little extra power, since the additional integrator can be impedance scaled.

Since the noise of the rest of the loop-filter (within the signal bandwidth) is reduced by the gain of the first integrator when referred to the input, the CIFF and CIFF-B structures are more tolerant of noise (and distortion) from the rest of the loop when compared to their CIFB counterparts.

10.5.1 Thermal versus Quantization Noise

The mean square in-band noise of a CT$\Delta\Sigma$M is comprised of two parts: a thermal component, denoted by N_{th} and (shaped) quantization noise component, which we denote by N_q. Due to oversampling, N_{th} is largely flat in the signal bandwidth (if we neglect $1/f$ noise). Suppose that we want to achieve a CT$\Delta\Sigma$M with a desired peak SNR. The maximum signal amplitude (MSA) is determined by the NTF and the number of levels in the quantizer. The peak SNR for a sinusoidal input is given by

$$SNR_{max} = \frac{(MSA^2/2)}{N_{th} + N_q}. \tag{10.14}$$

Clearly, many choices of N_{th} and N_q will result in the same peak SNR. A natural question that arises during design is how one should partition the noise budget into thermal and quantization components. Figure 10.16 shows three potential strategies. In strategy A, the budget is dominated by the quantization noise component. One could conceivably make N_{th} and N_q equal as in strategy B, or let thermal noise dominate, as in strategy C. Which
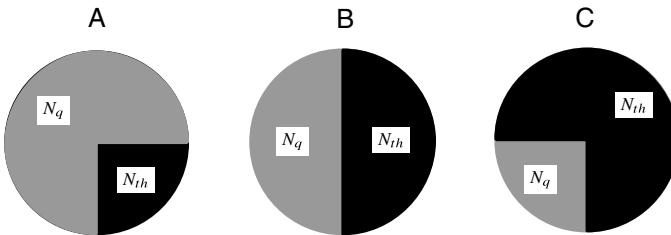
Figure 10.16   Possible ways of partitioning thermal and quantization noise components to achieve a desired peak SNR.

of these yields a modulator with the lowest power dissipation? To answer this question, let us arbitrarily pick strategy A and ponder what would happen to the power dissipated in the CT$\Delta\Sigma$M when we varied the relative contributions of N_{th} and N_q. If we decide to

increase N_{th} by a factor of 2, N_q would have to decrease by somewhat less than a factor of two. Now, increasing N_{th} by 2× is accomplished by impedance scaling the loop-filter by the same factor, and this reduces its power dissipation by a factor of 2. How do we reduce N_q? This can be done in a variety of ways – for instance, the NTF can be made more aggressive by increasing its out-of-band gain. Alternatively, the OSR and/or the order of the NTF can be increased slightly. The former does not result in any increase in power dissipation. Increasing OSR/order impact the power dissipation only slightly. Overall, therefore, increasing N_{th} by two reduces the CTΔΣM's power dissipation by a factor of two. Extending this argument further, it seems advantageous to allocate more and more of the noise budget to thermal noise (while reducing N_q). Beyond a certain point however, reducing in-band quantization noise begins to become difficult too. Thus, a power-efficient design is one where thermal noise accounts for a large part (around 75%) of the total noise budget. A good rule of thumb is to keep N_q about 12 dB lower than the thermal noise.

There are two other reasons to keep $N_{th} \gg N_q$:

- N_{th} is repeatable. With multi-bit quantization, N_q can be variable, necessitating comparator calibration and extra margin on the specifications.

- N_q can contain harmonics.

10.6   ADC Design
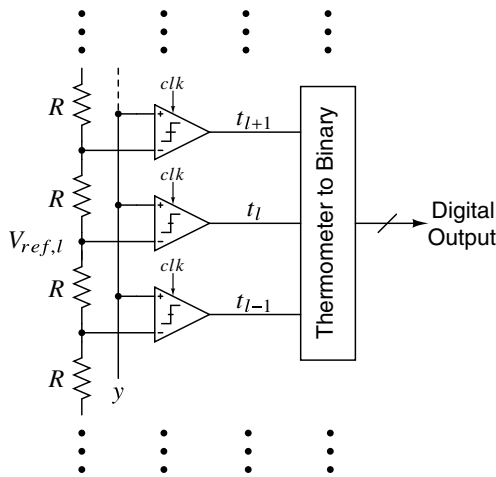
Figure 10.17   Block diagram of a flash ADC.

The ADC used in the quantizer can be realized in many ways. Which ADC architecture is particularly suited to a ΔΣ loop? Given that the ADC is going to be embedded in a strong negative feedback loop, the natural choice is to use the flash architecture. Flash ADCs use parallelism to achieve high-speed operation at the expense of hardware complexity and power dissipation. The operating principle of a flash ADC is shown in

Figure 10.17. An array of clocked comparators compares the input y with a set of references, usually generated using a resistor ladder. The comparison occurs at a time instant defined by the clock. Each comparator in the array generates a logical output t that is 1 if y (at the sampling instant) is greater than its reference (and vice versa). An M-step flash ADC employs M comparators, and the output of the comparator array is the so-called thermometer code, which is subsequently converted into binary form, and forms the output of the modulator. While Figure 10.17 showed a single-ended diagram for simplicity, practical realizations are fully differential.
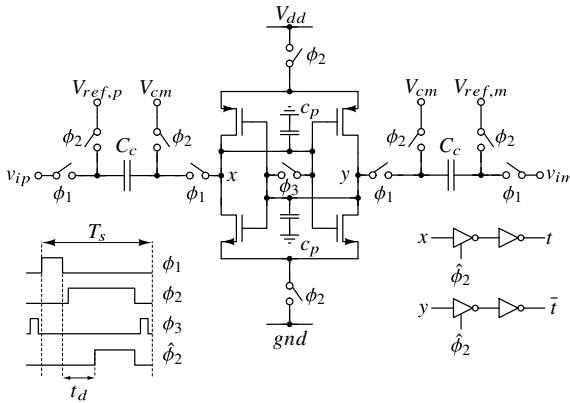
Figure 10.18   A comparator based on a sense amplifier.

The basic building block of the flash ADC is the clocked comparator. Several comparator circuits can be conceived. Figure 10.18 shows one derived from a sense amplifier. It consists of two clocked CMOS inverters connected back-to-back, and operates in three non-overlapping phases. C_c are the reference storage capacitors, while c_p denotes the parasitic capacitance from nodes x and y to ground. During ϕ_1, which is the sampling phase, the reference storage capacitors are connected in series with the differential inputs v_{ip} and v_{im}. The transisors are off. The differential voltage developed across x and y is given by

$$v_{xy} = \frac{C_c}{C_c + c_p} \left[(v_{ip} - v_{im}) - (V_{ref,p} - V_{ref,m}) \right]. \tag{10.15}$$

The sampling instant of the comparator is defined by the falling edge of ϕ_1. Regeneration occurs in ϕ_2, and by the end of this phase, the potentials at x and y reach V_{dd} or gnd. The reference storage capacitors are also refreshed during ϕ_2. ϕ_3 is a short phase that clears memory of the latch (and prevents hystersis) by shorting x and y. The next comparison cycle then begins. Since the voltage at x/y is a valid logic level only during the latter part of ϕ_2, the output is held on CMOS inverters that are clocked using $\hat{\phi}_2$ to generate the logical output t valid for a whole clock cycle. The delay of the comparator is that between the falling edge of ϕ_1 and the rising edge of $\hat{\phi}_2$.

In practice, threshold voltage mismatch of the MOS devices forming the regenerative pair causes static offset. Differences in parasitic capacitance on nodes x and y result in dynamic offset, which can often be much larger than the static offset. It turns out that dynamic offset depends on the difference between the common-mode voltage of x and y at the end of ϕ_1, and the natural threshold of the regenerating inverter. In a sense-amplifier-

based comparator, this difference can be made small by appropriate choice of V_{cm} and inverter geometry.

A disadvantage of the sense-amplifier-based comparator is the complexity of generating and distributing the various clocks needed. In high-speed designs, the power dissipated in the clock distribution network can become significant.
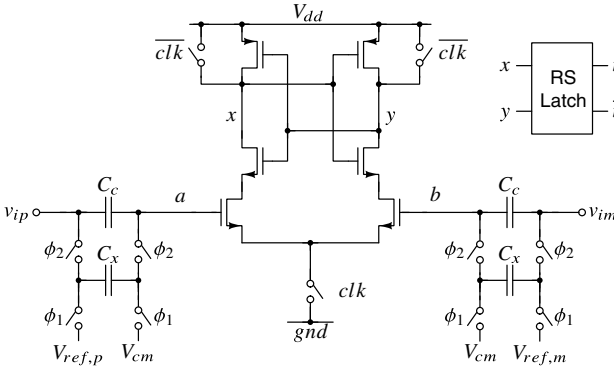
Figure 10.19   A comparator based on the StrongARM latch.

A comparator with simpler clocking, based on the StrongARM latch, is shown in Figure 10.19. In steady state, the reference storage capacitors C_c have voltages $(V_{ref,p}-V_{cm})$ and $(V_{ref,m} - V_{cm})$ across them. Thus,

$$v_a - v_b = (v_{ip} - v_{im}) - (V_{ref,p} - V_{ref,m}). \tag{10.16}$$

As described in Section 7.9, when *clk* is low, the nodes x and y are pulled to V_{dd}. Regeneration begins when *clk* goes high. x and y attain valid logic levels during the latter part of this phase. The decision is held for a whole clock cycle by the RS-latch. C_x, chosen to be much smaller than C_c, serves to refresh the reference voltage stored across the latter. ϕ_1 and ϕ_2 are non-overlapping clocks (and can even be operated at a lower clock rate). An advantage of this way of subtracting references is that no switches appear in series with the high-speed signal path. Since $C_x \ll C_c$, the ϕ_1 and ϕ_2 switches can be small, thereby reducing parasitic capacitances at a and b.

The "sampling instant" of the StrongARM structure is somewhat fuzzy, but it can be expected to occur shortly after *clk* goes high. While simplified clocking and ease of reference subtraction is a definite advantage, the StrongARM latch suffers from an increased dynamic offset when compared to the sense-amplifier-based structure. The reason is the following. When *clk* goes high, the pull-down network is activated. Thus, the the common-mode voltage of x and y, which was V_{dd} at the end of the previous phase, drops. Mismatch in parasitic capacitance at x and y will result in a differential voltage to be developed across these nodes, which manifests as comparator offset.

Figure 10.20 shows another comparator based on the StrongARM latch, where reference subtraction is accomplished through additional transistors connected in parallel with the input transistors. Many other variants exist.

As discussed in earlier chapters, comparator offset is of no consequence in a single-bit modulator. In a multi-bit CT$\Delta\Sigma$M, however, offsets modify the shape of the quantizer's
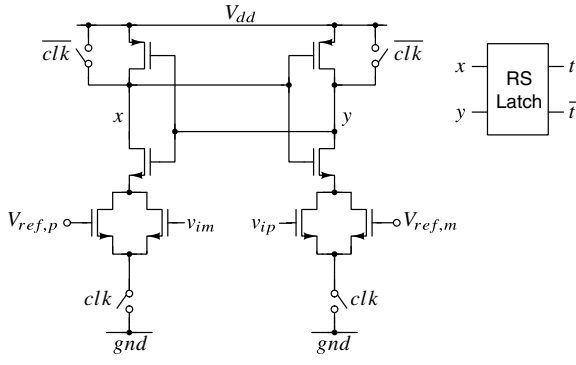
Figure 10.20   Another comparator based on the StrongARM latch.
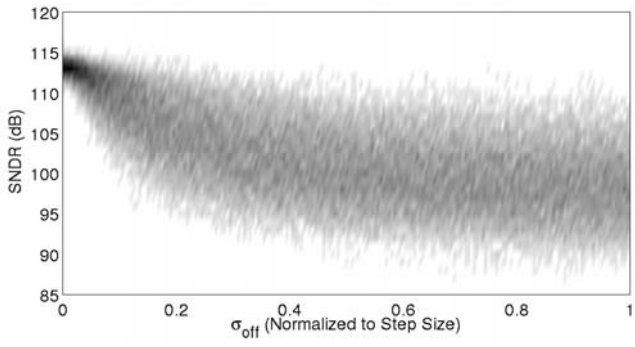
Figure 10.21   In-band SNDR of a third-order, 15-level CT$\Delta\Sigma$M in the presence of random offset in the comparator thresholds.

transfer curve. As a result, the in-band quantization noise and the maximum stable ampli-tude change. Figure 10.21 shows the simulated in-band SNDR of a third-order modulator with varying levels of comparator offset. The quantizer has 15 levels, $OSR = 64$, and the NTF is maximally flat, with an out-of-band gain of 2.5. The input is a $-6\,$dBFS tone in the signal band. Comparator offsets are assumed to be Gaussian distributed, with σ_{off} denot-ing the standard deviation of offset normalized to the nominal step size. 200 Monte Carlo simulations are performed for each value of σ_{off}. It is apparent that the SNDR can degrade significantly (by about 20 dB) when σ_{off} is large. It is prudent, therefore, to ensure that comparator offsets are kept small. Figure 10.21 indicates that $\sigma_{off} = 0.05$ is a good value to target, in our example.
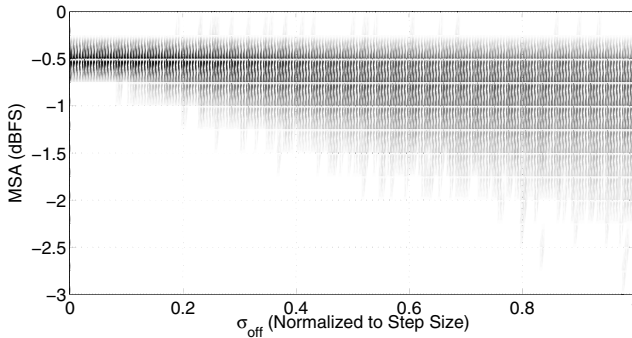
Figure 10.22   MSA of the CTΔΣM with varying σ_{off}.

Figure 10.22 shows the MSA of the CTΔΣM as a function of comparator offset. We see that this is also affected, though not quite as much as the in-band SNDR.

From the discussion above, we see the following. While flash ADC errors are less critical (when compared to feedback DAC nonidealities), worst-case comparator offsets do have to be restricted to a fraction of the nominal step size. It is not uncommon, therefore, to correct for these offsets. To do so, a means to control offset is necessary. One way of doing this is shown in Figure 10.23. The preamplifier output ($v_a - v_b$), which drives the Stron-gARM latch, is the amplified difference ($v_{ip} - v_{im}) - (V_{ref,p} - V_{ref,m}$). I_{off} is a (digitally) programmable current source that modifies the input-referred offset of the preamplifier. During power on, for instance, I_{off} can be set so as to minimize the comparator offset.

10.7   Feedback DAC Design

The closed-loop transfer function of a negative feedback system is critically dependent on the characteristics of the feedback block. A CTΔΣM, whose in-band SNDR is critically influenced by the noise and linearity of the feedback DAC, is no exception to this rule. In this section, we discuss various ways in which the DAC can be implemented, and their relative merits.

Figure 10.23 Offset canceled comparator employing a preamplifier and StrongARM latch.

Figure 10.24 (a) Conceptual single-ended schematic of a CTΔΣM using a resistive DAC and (b) practical implementation of a differential unit element, feeding into an OTA-RC integrator.

10.7.1   Resistive DACs

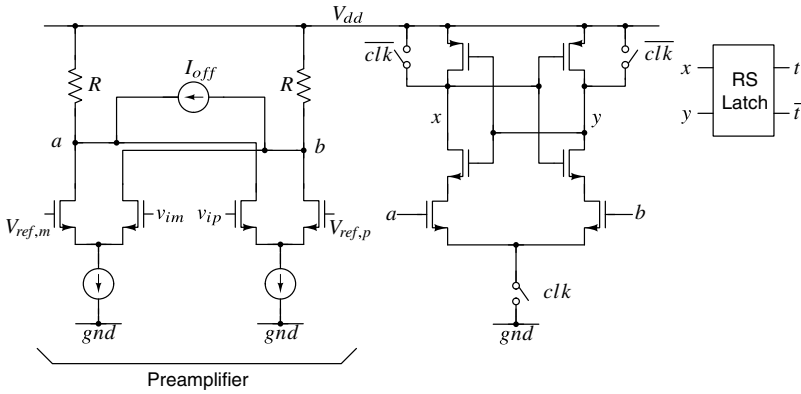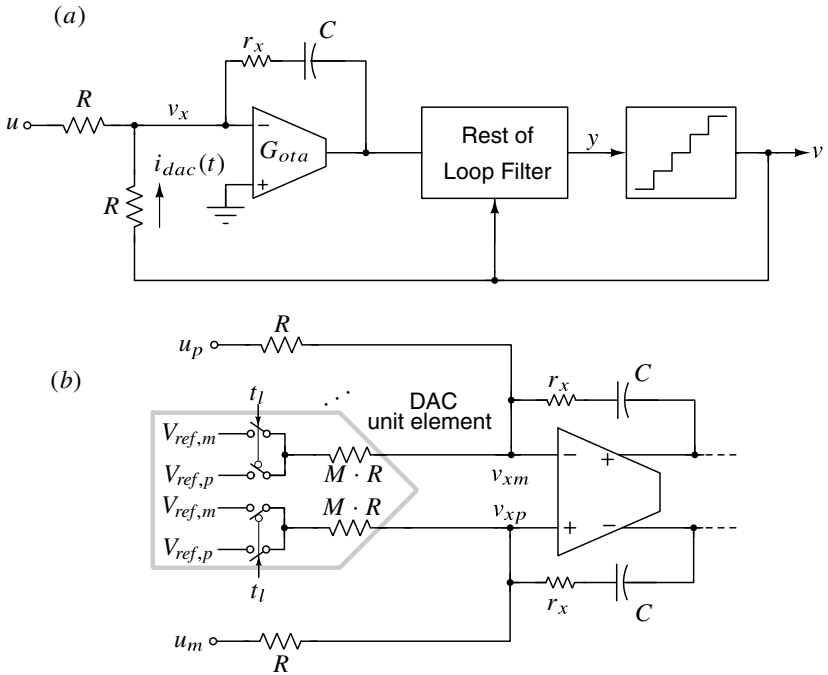As we saw earlier in this chapter, the OTA-RC structure is a compelling choice for implementing the integrators comprising the loop-filter. A natural approach to realize a DAC is to take advantage of the virtual ground facilitated by the OTA, as shown in the conceptual single-ended schematic of Figure 10.24(a). Most practical CTΔΣM designs, however, are differential, and Figure 10.24(b) shows how the DAC connects to the OTA. If we assume an M-step quantizer, the ADC output is an M-bit thermometer code. Each thermometer bit t drives a differential DAC *unit element*, which is formed by a pair of resistors each of value $M \cdot R$. Depending on the state of t_l (representing the lth thermometer bit output), the resistors are connected to the positive or negative references ($V_{ref,p}/V_{ref,m}$). If t_l is held for a whole clock period, an NRZ pulse shape results. Such DACs are also referred to as switched-resistor DACs.

We will denote the supply voltage by V_{dd}; the common-mode voltage at the opamp input is chosen to be $0.5\,V_{dd}$. In the best case where $V_{ref,p} = V_{dd}$ and $V_{ref,m} = 0$, the full-scale differential current of the DAC is V_{dd}/R. The spectral density of the current noise is given by $8kT/R$. When referred to the modulator's input, this translates into a noise voltage spectral density of $8kTR$.

What are the merits of a resistive DAC? For a given full-scale current, such a DAC adds the smallest possible thermal noise. Modulators that use such DACs, therefore, tend to be very power efficient. DAC layout tends to be clutter-free, thanks to the simplicity of the unit element.

What are the problems associated with a switched-resistor DAC? The loop-gain around the input OTA is reduced due to the resistive loading at its virtual ground node. As a result, the low-frequency noise of the OTA, when referred to the CTΔΣM input, is gained up by a factor of two. Further, the reduced loop-gain degrades the linearity of the integrator, and must be tackled by appropriate OTA design.

Often, high-resistivity polysilicon resistors are not available in a fabrication process. This results in physically large resistors, particularly in low-bandwidth designs. The distributed parasitic capacitance of the resistor can then be significant, adding excess loop-delay, over and above the delay introduced by the rest of the quantizer. This can be addressed by any of the ways discussed in Chapter 9.

In a multi-bit modulator, a mismatch in the resistances of the unit elements will degrade the in-band SNR, and this has to be addressed by calibration or dynamic element-matching (DEM) techniques. This is a vast and important topic in itself, and was covered in detail in Chapter 6.

Another form of nonlinearity can degrade even the performance of a single-bit switched-resistor DAC. We alluded to this while discussing the usefulness of RZ DACs in Chapter 9. There, we referred to this phenomenon as transition error, more commonly known as inter-symbol interference (ISI). This is an instance of *dynamic nonlinearity*, caused by the difference between the rise and fall times of the NRZ feedback waveform. The root cause is the mismatch between the resistance and timing of the pull-up and pull-down switches in the DAC unit element, as explained with the help of Figure 10.25. The DAC switches are assumed to have resistances r_m and r_p, and a parasitic capacitance c_p is present at the junction of the switches. The OTA is assumed to be ideal. We first analyze the single-ended current i_m. Ideally, this should be $\pm I_o$, where $I_o = V_{dd}/(2R)$, as shown in
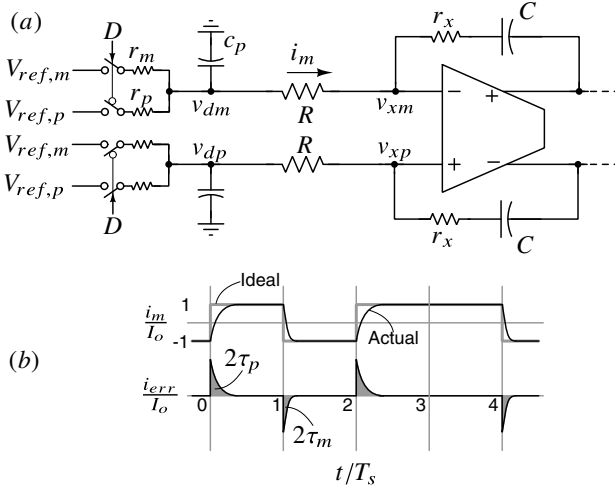
Figure 10.25   (a) A single-bit switched-resistor DAC, with unequal resistances for the pull-up and pull-down switches. c_p represents (undesired) parasitic capacitors. (b) Current waveforms.

Figure 10.25(b). Due to nonzero switch resistance, the waveform has differing rise and fall time-constants $\tau_p = r_p c_p$ and $\tau_m = r_m c_p$, respectively. The error current $i_{err}(t)$, which is the difference between the ideal and actual current is a train of exponentially decaying pulses, as shown in the figure. It is easily seen that the following sequences take on a value of $2(-2)$ at rising(falling) edges and zero elsewhere.

$$t_{up}[n] = 0.5(v[n] - v[n-1] + |v[n] - v[n-1]|),$$
$$t_{dn}[n] = 0.5(v[n] - v[n-1] - |v[n] - v[n-1]|).$$

$p_r(t)$ and $p_f(t)$, which are the deviations from the ideal DAC waveform at the rising and falling edges, respectively, are given by $p_r(t) = 2\exp(-t/\tau_p)$ and $p_f(t) = 2\exp(-t/\tau_m)$. The error current can be expressed as

$$i_{err}(t) = I_o \sum_n t_{up}[n]p_r(t - nT_s) + t_{dn}[n]p_f(t - nT_s) \tag{10.17}$$

$$= \frac{I_o}{2} \sum_n \underbrace{\{v[n] - v[n-1]\}}_{linear}(p_r(t - nT_s) + p_f(t - nT_s))$$

$$+ \frac{I_o}{2} \sum_n \underbrace{\{|v[n] - v[n-1]|\}}_{nonlinear}(p_r(t - nT_s) - p_f(t - nT_s)).$$

From the equations above, it is seen that the single-ended error current waveform has a nonlinear component dependent on $|v[n] - v[n-1]|$ and the difference between the rise and fall transition errors, p_r and p_f. The nonlinear charge injected by the DAC in one clock period, normalized to the ideal charge, is given by

$$\alpha = \frac{\tau_m - \tau_p}{T_s}. \tag{10.18}$$

In a two-level modulator, v transitions rapidly between -1 and 1 when the input is small,
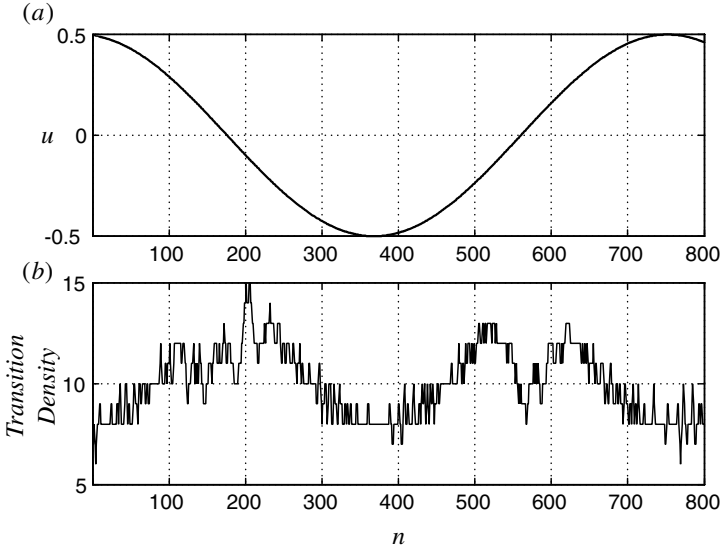
Figure 10.26   (a) Input to the third-order 2-level modulator and (b) number of transitions in the preceding 16 samples.

and less frequently when u is large (see Figure 9.23). When u is a sinusoid, the error due to ISI is large when u crosses zero and small at the peaks. Figures 10.26(a) and (b) show the input and output transition density, respectively, for a third-order CT$\Delta\Sigma$M whose input u is 6 dB below full-scale. The transition density refers to the number of transitions in v in the preceding 16 samples. From this, it is apparent that the error due to ISI should have a strong second harmonic content. This observation is confirmed by the PSD in Figure 10.27 ($\alpha = 10^{-3}$).
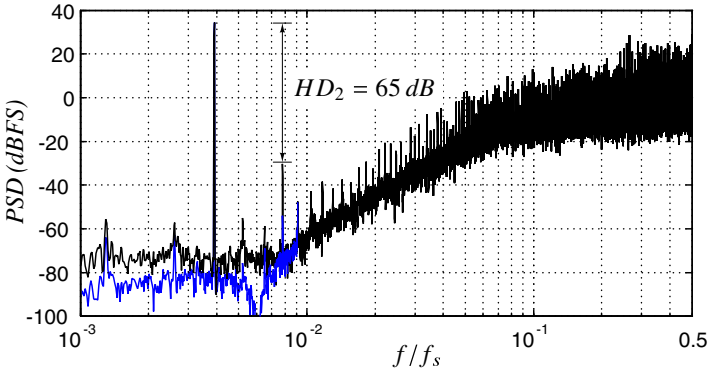
Figure 10.27   Simulated PSD of a third-order single-bit CT$\Delta\Sigma$M with and without ISI. An NRZ DAC, with $\alpha = 10^{-3}$ is assumed.

The discussion above considered ISI induced distortion in the single-ended DAC current, from which it is apparent that ISI can dramatically degrade the performance of a CT$\Delta\Sigma$M that employs an NRZ DAC. Fortunately, however, differential operation comes to the rescue. Referring to Figure 10.25, we see that the two integrating resistors are identi-

cal. Then, the nonlinear portion of the DAC current is a common-mode component that is completely rejected due to differential symmetry. In practice, mismatch in the differential halves, or their timing, would cause some of this to "leak through".

How does ISI affect the output of a multi-bit DAC? The error current added by each unit element is still given by (10.17). If the DAC elements were directly driven by the thermometer output of the ADC, the total nonlinear error due to ISI would be proportional to $|v[n] - v[n-1]|$, and would result in an increased in-band noise floor and harmonic distortion. However, the use of dynamic element-matching techniques modifies the number of transitions (as seen in Chapter 6), and it can potentially change the spectrum of the error current.

A final aspect of a resistive NRZ DAC that merits discussion is the reference generator. It is apparent that noise on the references, when referred to the CT$\Delta\Sigma$M input, appears as-is. The reference generator must therefore be designed appropriately. Further, if we assume identical unit elements and a zero opamp-offset, then the current drawn from $V_{ref,p}/V_{ref,m}$ will be independent of the modulator output v.

10.7.2 Return-to-Zero and Return-to-Open DACs
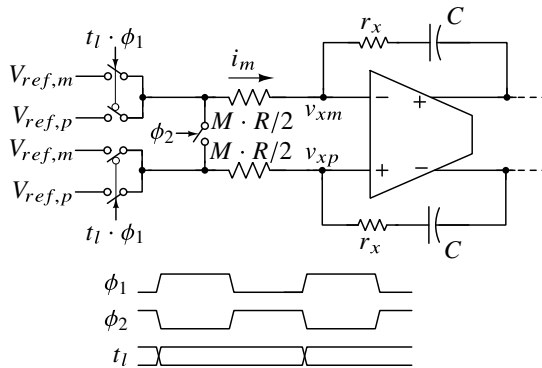
Figure 10.28 The differential unit element of a return-to-zero DAC.

The problem of ISI associated with an NRZ DAC is the motivation to use a return-to-zero DAC. The basic idea is shown in Figure 10.28. t_l is the lth bit of the thermometer code. During ϕ_1, the resistors are connected to $V_{ref,p}/V_{ref,m}$ depending on t_l. During ϕ_2, the resistors are shorted (or connected to V_{cm}), causing the current to return to zero for the latter half of the clock cycle. Since an RZ waveform has a rising and falling edge in every clock cycle (independent of t_l), unequal rise/fall times do not result in nonlinearity. To deliver a charge equal to that in the NRZ case, the resistors need to be reduced by a factor of two. When compared to an NRZ DAC, therefore, the OTA should be designed to handle larger currents without causing distortion.

Since the resistors are smaller, the input-referred thermal noise spectral density of the RZ DAC is twice as high as that of its NRZ counterpart. This makes sense – during ϕ_2 (which lasts half the clock period), the RZ DAC simply injects noise without contributing to the signal component.

The return-to-open (RTO) DAC aims to remedy this problem by dispensing with the ϕ_2 switch of Figure 10.28 altogether. This way, the resistors do not contribute noise in ϕ_2. The average input-referred noise spectral density contributed by the resistors is $8kTMR$, like with an NRZ DAC. Unfortunately, however, the periodic switching of the DAC resistance modulates the loop-gain around the OTA, rendering the integrator a periodically time-varying system. This degrades the alias-rejection of modulator, and causes OTA noise from higher frequencies to alias into the signal band.

To summarize, the RZ and RTO DACs are attempts to solve the ISI problem associated with the NRZ DAC. The price paid for this is the increased demand on the linearity of the OTA, thermal noise (in the RZ case), or a compromised alias rejection (with the RTO DAC). Further, as we saw in Chapter 9, a jittery clock adversely affects the performance of the modulator. The reference buffer needs to supply pulsed currents due to the RZ nature of the current pulse. This necessitates stronger bypassing in the buffer.

10.7.3 Current-Steering DACs
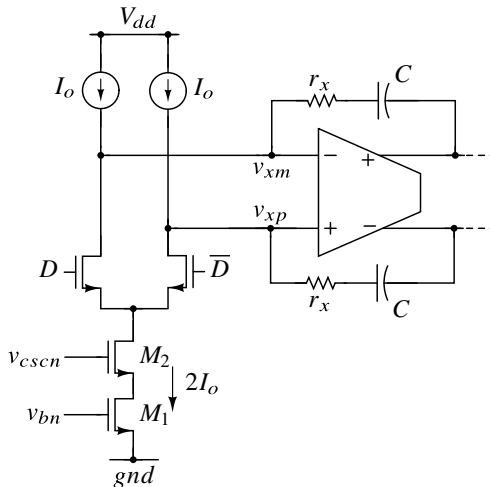
Figure 10.29   Unit element of a current-steering DAC.

An alternative to relying on the virtual ground of the OTA, as we did with a switched-resistor DAC, is to steer the current generated by a current source. This forms the basis for the current-steering DAC, whose unit cell is shown in Figure 10.29. M_1 and M_2 form a cascoded current source, whose current $2I_o$ is steered into v_{xm} or v_{xp} depending on the sign of D. The I_o current sources are needed to balance the common-mode component of the current injected by $M_{1,2}$.

Why would one want to use a current-steering DAC in the first place? For one, the DAC does not load the virtual ground node, at least in principle. This has two benefits – the gain from the OTA noise to the output is now unity (as opposed to 2 with a switched-resistor DAC). The loop-gain around the first OTA is higher, resulting in improved integrator linearity. Further, a current-steering DAC's full-scale, and hence that of the ADC, is adjustable

via the v_{bn} voltage. In applications that demand a high dynamic range, a variable-gain amplifier (VGA) is often used to broaden the dynamic range of the signal chain, and in such applications a variable ADC full-scale can eliminate the need for a VGA.

Last, since the v_{bn} bias voltage is connected to a transistor gate, bias noise can be filtered with a simple RC section. A mega-ohm resistor plus a 100 pF capacitor provides a noise bandwidth in the kHz range.
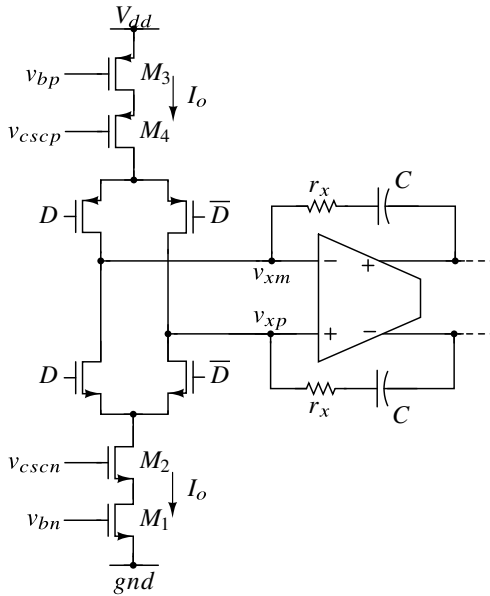
Figure 10.30   Unit element of a complementary current-steering DAC.

Referring to Figure 10.29, the net current flowing into the integrating capacitors is $\pm I_o$. Since one of these differential currents I_o is obtained by subtracting I_o from $2I_o$, the unit element injects more noise than is fundamentally necessary. Excess noise in the basic current-steering DAC is remedied by the complementary unit cell shown in Figure 10.30. However, the current noise injected by this cell is still larger than that due to a switched-resistor DAC supplying the same differential current. To see why, notice that the overdrives of three devices must "fit" within $V_{dd}/2$. The most optimistic scenario is to allocate *all* of this headroom to M_1/M_3. Then, the spectral density of the noise current from each source would be

$$S_{Io}(f) = 4kT\gamma \frac{2I_o}{V_{dd}/2},\tag{10.19}$$

where γ in modern processes ranges from $1-2$. If the same current was due to a resistor, the noise current would have been

$$S_{Io}(f) = 4kT \frac{I_o}{V_{dd}/2},\tag{10.20}$$

indicating that the current-steering DAC is at least 3 dB worse than a switched-resistor NRZ DAC. In practice, the spectral density of the noise added by the former is higher, since the overdrive voltage of M_1/M_2 can only be a fraction of $V_{dd}/2$.

Other differences between a switched-current and a switched-resistor DAC relate to the switches. In the latter, the switches are operated in triode mode, whereas in the former, they are usually operated in saturation. As a result, a current-steering DAC's switches can be much smaller than those in a resistive DAC. The downside for the current-steering DAC is that the gate-drive voltage (V_{on}) of the switches may need to be managed. For example, in order for the NMOS switches to be in saturation, $V_{on} < V_{cm} + V_T$. If $V_{on} = V_{dd}$ and $V_{cm} = V_{dd}/2$, this means that we require $V_T > V_{dd}/2$. If this condition cannot be guaranteed over PVT, then circuitry is needed to set V_{on} appropriately.

Last, we note that switch mismatch in a resistive DAC results in both static and dynamic (ISI) errors. In current-steering DAC, however, the static error caused by switch mismatch is attenuated by the high impedance of the current source.

In all other ways, a current-steering DAC and a resistor DAC are similar. Their sensitivity to jitter is identical, as is ISI due to timing asymmetry.

10.7.4 Switched-Capacitor DACs
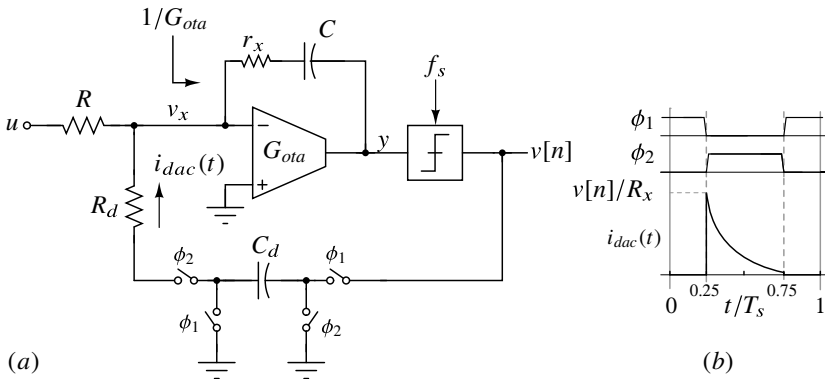
Figure 10.31   (a) A first-order CT$\Delta\Sigma$M with a switched-capacitor feedback DAC and (b) relevant waveforms.

In Chapter 9, we saw that using a switched-capacitor (SC) feedback DAC [2] offered one way to address the problem of clock jitter in CT$\Delta\Sigma$Ms. The idea is illustrated using the first-order single-bit modulator of Figure 10.31(a). The sampling rate and period are denoted by f_s and T_s, respectively. The integrator is of the OTA-RC type. r_x is the zero-canceling resistor, and nominally equals $1/G_{ota}$. During ϕ_1, the DAC capacitor C_d is charged to v. During ϕ_2, it is discharged into the OTA's virtual ground. For an ideal OTA ($G_{ota} \to \infty$), i_{dac} is $v[n]/R_d$ at the beginning of ϕ_2, and it decays exponentially with a time-constant $R_d C_d$. If this is chosen to be much smaller than $T_s/2$, C_d is discharged at the end of ϕ_2. The charge injected by the feedback DAC in one clock period is given by $C_d v[n]$. Since $R_d C_d \ll T_s/2$, a jittery clock does not modify this charge. Thus, as explained in detail in Section 9.4, using an SC DAC reduces the sensitivity to clock jitter.

The SC DAC has an exponentially decaying pulse shape. The peak current in a clock period is given by $v[n]/R_d$, while the average current is $v[n]C_d/T_s$. The peak-to-average ratio of the pulse is $T_s/R_d C_d$. The price for achieving good jitter immunity, therefore, is a

feedback current with large peaks. This in turn necessitates an OTA with sufficiently high linearity.

Let us review the basic properties of an SC-DAC, assuming that G_{ota} is infinite. How should C_d be chosen so as to achieve an STF whose dc gain is unity? The average current though the integrating capacitor is zero. Since we are assuming an ideal OTA (i.e, $v_x = 0$), this means that

$$\underbrace{\frac{\bar{u}}{R}}_{\text{Average current due to } u} - \underbrace{\frac{\bar{v}C_d}{T_s}}_{\text{Average DAC current}} = 0. \tag{10.21}$$

Thus, $C_d f_s = 1/R$ to achieve $STF(0) = 1$. This is intuitively satisfying – unity-gain is achieved when the switched-capacitor "feedback resistor" is made equal to the input resistor.

How should C be chosen to achieve an NTF equal to $(1 - z^{-1})$? To determine this, we need the transfer function $L_1(z)$ from v to the sampled output at y to be $z^{-1}/(1 - z^{-1})$. This is easily seen to be $(C_d/C)z^{-1}/(1 - z^{-1})$, implying that C should be made equal to C_d.

How does the modulator respond when excited by a tone at f_s? An input tone at f_s can potentially alias to the dc component of v. To determine the $\bar{v}$, we again use the observation that the average current through C is zero. This means that

$$\underbrace{\frac{\overline{\cos(2\pi f_s t)}}{R}}_{=0} - \underbrace{\frac{\bar{v}C_d}{T_s}}_{\text{Average DAC current}} = 0, \tag{10.22}$$

which yields $\bar{v} = 0$.

To summarize, with an ideal OTA, the modulator of Figure 10.31 has a dc gain of unity, $NTF = (1 - z^{-1})$, inherent anti-aliasing, *and* reduced sensitivity to clock jitter. What happens when the transconductance of the OTA is finite?

Figure 10.32(a) shows the integrator with the SC DAC. As far as the DAC is concerned, the integrator can be replaced by its Thevenin equivalent circuit, as shown in Figure 10.32(b). Assuming $RG_{ota} \gg 1$ (needed anyway to realize a good integrator), the Thevenin voltage and resistance are $u/(RG_{ota})$ and $1/G_{ota}$, respectively. The discharge time-constant of the DAC capacitor is now given by $(R_d + 1/G_{ota})C_d$, which should be chosen to be much smaller than $0.5T_s$ to achieve good jitter immunity.

What is the dc gain of the STF? To determine this, we proceed as follows. We excite the modulator with a 1 V dc input. Referring to Figure 10.32(a), $\overline{i_c(t)} = 0$. Since $i_c(t) = -G_{ota}v_x(t)$, if follows that $\overline{v_x(t)} = 0$. The average current drawn from the input is thus given by

$$\overline{i_{in}(t)} = \overline{\frac{u - v_x(t)}{R}} = \frac{1}{R}. \tag{10.23}$$

It therefore follows that $\overline{i_{dac}(t)}$ should be $-1/R$. How is $\overline{i_{dac}(t)}$ related to $\bar{v}$? To determine this, we use Figures 10.32(b) and (c). During ϕ_1, C_d is charged to $v[n]$. In this phase, $i_{dac}(t) = 0$, and the node x is at a potential of $1/(RG_{ota})$. During ϕ_2, C_d is flipped around and connected to x through R_d. As a result, v_x initially dips, but by the end of this phase, C_d loses its initial charge, and v_x approaches the potential $1/(RG_{ota})$, as shown
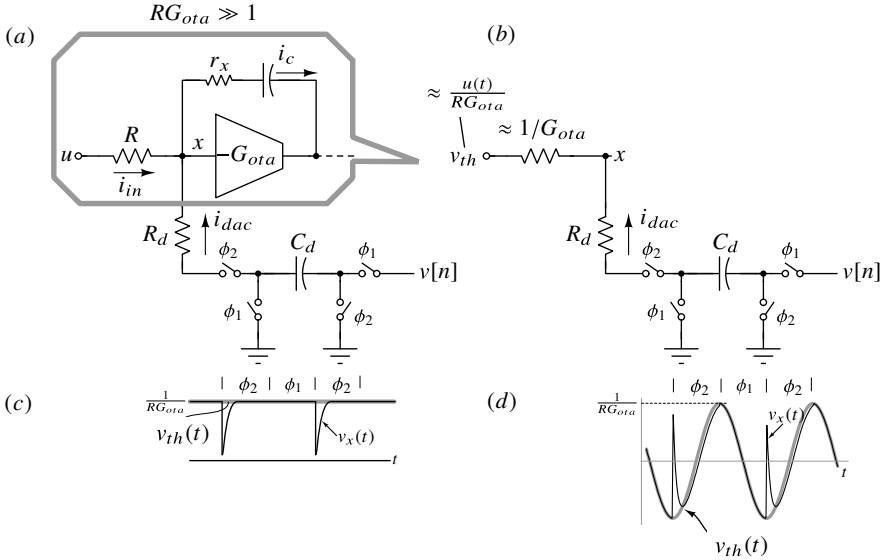
Figure 10.32 (a) Input integrator of a CT$\Delta\Sigma$M with a switched-capacitor feedback DAC. (b) The input resistor and OTA can be replaced by its Thevenin equivalent. (c) $v_{th}(t)$ and $v_x(t)$ for a dc input and (d) $v_{th}(t)$ and $v_x(t)$ for a sinusoidal input.

in Figure 10.32(c). The charge transferred by the DAC over the complete clock cycle is, therefore,

$$Q_{dac}[n] = \underbrace{-C_d v[n]}_{\text{initial charge}} - \underbrace{\frac{C_d}{RG_{ota}}}_{\text{final charge}}, \tag{10.24}$$

which means that

$$\overline{i_{dac}(t)} = f_s C_d \left(-\overline{v} - \frac{1}{RG_{ota}} \right). \tag{10.25}$$

Since $\overline{i_{dac}(t)} = -1/R$ and $f_s C_d = 1/R$, it follows that

$$\frac{\overline{v}}{u} = STF(0) = \left(1 - \frac{1}{RG_{ota}} \right). \tag{10.26}$$

It is thus seen that the dc gain is *approximately* one, with the deviation from unity being proportional to $1/(RG_{ota})$.

To determine the alias rejection at f_s, we need to find the response of the modulator for an input at the sampling frequency. To this end, we excite the modulator with $u(t) = \cos(2\pi f_s t)$. The average current drawn from the input is

$$\overline{i_{in}(t)} = \frac{\overline{\cos(2\pi f_s t) - v_x(t)}}{R} = 0. \tag{10.27}$$

Since $\overline{i_c(t)}$ and $\overline{i_{in}(t)}$ are zero, it follows that $\overline{i_{dac}(t)}$ should be zero. To relate $\overline{i_{dac}(t)}$ to $\overline{v}$, we use Figures 10.32(b) and (d). v_{th} is now a sinusoid with amplitude $1/(RG_{ota})$. During ϕ_1, $i_{dac}(t) = 0$. During ϕ_2, C_d, which was charged to $v[n]$ during ϕ_1, is connected to x through R_x. This causes a glitch in the virtual ground node, as shown in Figure 10.32(d).

Since the discharge time-constant of the DAC capacitor is much smaller than $T_s/2$ (needed for good immunity to clock jitter), the voltage across C_d tracks v_{th} by the end of ϕ_2 and attains a potential of $1/(RG_{ota})$. The charge transferred by the DAC over the complete clock cycle is, therefore,

$$Q_{dac}[n] = \underbrace{-C_d v[n]}_{\text{initial charge}} - \underbrace{\frac{C_d}{RG_{ota}}}_{\text{final charge}} , \tag{10.28}$$

which means that

$$\overline{i_{dac}(t)} = f_s C_d \left(-\overline{v} - \frac{1}{RG_{ota}} \right). \tag{10.29}$$

Since $\overline{i_{dac}(t)}$ must be zero, it follows that

$$\overline{v} = STF(j2\pi f_s) = -\frac{1}{RG_{ota}}. \tag{10.30}$$

We thus see that with a practical OTA, the "inherent anti-aliasing" property of the CTΔΣM is compromised [3]. The intuition is the following. The DAC capacitor samples the virtual ground of the OTA. Since G_{ota} is finite, the virtual ground node contains a component at the input frequency, which aliases into the signal band after sampling. Alias rejection can be improved by increasing the OTA's transconductance, thereby increasing RG_{ota}. Unfortunately, this is not power efficient – improving the rejection by 20 dB in this manner would need the power dissipation of the OTA increase by a factor of 10!

Does the use of a multi-stage OTA (which can lead to an increased G_{ota}) help improve alias rejection? Unfortunately, this is not the case, for the reason described below. When the CTΔΣM is excited by an input at f_s, the swing at the virtual ground node depends on the OTA's transconductance *at* f_s. A multi-stage OTA, as we saw earlier in this chapter, improves only the low-frequency gain of the OTA, making it an ineffective tool to address the alias-rejection problem.

The "average" arguments presented above enabled us to determine the alias rejection of the modulator at f_s. What happens when the input frequency is close to f_s? Intuitively, we should expect it to be approximately $1/(RG_{ota})$, but a more careful analysis is needed. The key aspect of a CTΔΣM with SC feedback is that the loop-filter becomes a linear periodically time varying (LPTV) system. It turns out that this degrades alias rejection, as explored in more detail in Appendix C.

In summary, while the SC feedback DAC is an intuitively appealing idea for combating clock jitter, it presents many practical implementation challenges. For one, the linearity needed of the first integrator is greatly increased due to the high peak-to-average ratio of the feedback waveform. Next, the inherent anti-aliasing feature, the hallmark of continuous-time ΔΣ modulation, is limited to about 20 dB.

10.8 Systematic Design Centering

So far, we have seen how to choose an NTF and the number of quantizer levels to achieve a desired in-band SQNR. We have also discussed the considerations that go into the choice

of loop-filter, as well as various approaches to designing an integrator. We understand how thermal and quantization noise levels must be chosen relative to one another. The next step in the design process is to design the various building blocks – namely the OTAs (assuming active-RC integrators), ADC and DAC, and then to put the modulator together. Errors in the ADC thresholds are either corrected, or modeled as additional noise at its input. Errors in the DAC levels need special attention, and this topic is addressed in detail in Chapter 6. As far as the NTF is concerned, therefore, the quantizer can simply be modeled by its delay. The next task is to understand the effect of finite gain and bandwidth of the OTAs on the loop's NTF and more important, to mitigate these effects.

Due to finite dc gain and bandwidth, the integrators of the loop-filter are no longer ideal. Further, loading also modifies their transfer functions. Consequently, the NTF that is actually realized will be different from that which was originally intended. This raises the following questions:

- Is it possible at all to restore the NTF to the one that we wanted in the first place?

- If yes, how does one do this?
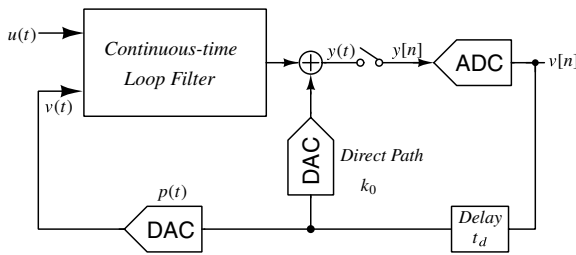
Figure 10.33   An example second-order CIFF CTΔΣM to illustrate the idea behind coefficient tuning.

We attempt to answer the questions above using a second-order CIFF CTΔΣM as an example (Figure 10.33). Without loss of generality, we assume that the sampling rate is 1 Hz. The DAC pulse shape and excess loop delay are denoted by $p(t)$ and t_d, respectively. The direct path, with gain k_0, compensates for t_d.

How does one determine the coefficients of the continuous-time loop-filter to achieve the desired NTF? As we have seen in Chapter 8, we need to match the sampled pulse response of the continuous-time loop with the impulse response of the discrete-time pro-totype. We denote the latter by $l[n]$ and its z-transform by $L(z)$. To determine the pulse response of the continuous-time filter, the loop is opened, as shown in Figure 10.34, and excited by the DAC pulse, which is delayed by t_d, to account for excess loop delay. $l_0(t), l_1(t)$, and $l_2(t)$ and their sampled versions can then be found. With ideal integrators, and an NRZ pulse shape,

$$l_0[n] = \begin{bmatrix} 0 & 1 & 0 & \cdots \end{bmatrix}^T,$$

$$l_1[n] = \begin{bmatrix} 0 & 1 - t_d & 1 & \cdots \end{bmatrix}^T,$$

$$l_2[n] = \begin{bmatrix} 0 & 0.5(1 - t_d)^2 & 1.5 - t_d & \cdots \end{bmatrix}^T.$$
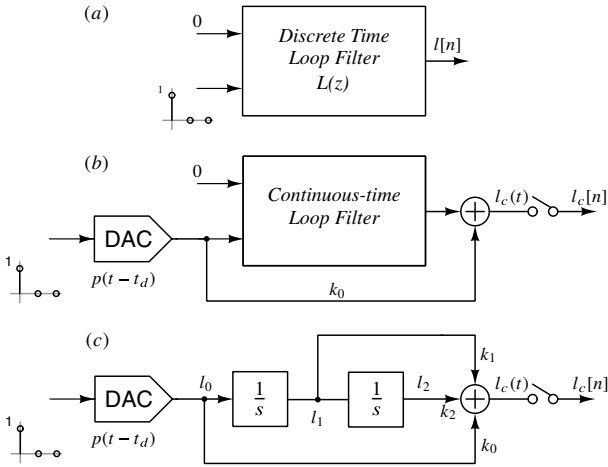
Figure 10.34   (a) Discrete-time loop-filter. (b) Exciting the continuous-time loop-filter with the DAC pulse. (c) Matching the impulse response of the continuous-time loop-filter with that of the discrete-time prototype by adjusting k_0, k_1 and k_2.

Denoting $K = \begin{bmatrix} k_0 & k_1 & k_2 \end{bmatrix}^T$, where k_0, k_1, and k_2 denote the gains of the direct, first, and second-order paths respectively, we have

$$\begin{bmatrix} l_0[n] & l_1[n] & l_2[n] \end{bmatrix} K = l[n] \, , \ n \in [0, N]. \tag{10.31}$$

The set of $(N + 1)$ equations in three unknowns has a unique solution, as we have seen in Chapter 8, regardless of N. The advantage of this numerical way of finding the coefficients, rather than work with the z-transforms of l_0, l_1, l_2, and $L(z)$ is that the former can be applied to a practical design, where they are readily available from the results of a transient analysis in a circuit simulator. However, working with their transforms requires the precise knowledge of the poles of the continuous-time system – which is no easy task, since the integrators are high-order systems in reality.
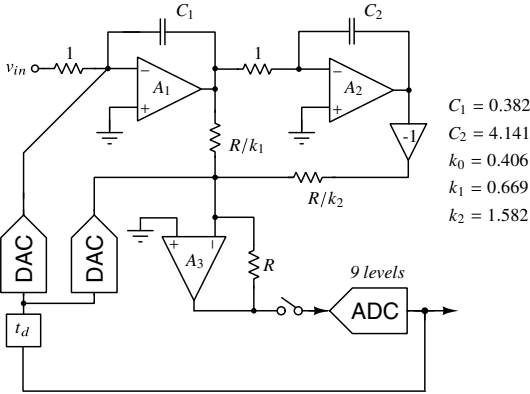
Figure 10.35   Component values of the CTΔΣM after dynamic-range scaling, assuming ideal OTAs.

After determining coefficients, we perform dynamic-range scaling assuming ideal OTAs, as discussed in Section 8.8. The resulting normalized second-order CTΔΣM that employs a 9-level quantizer and achieves $NTF = (1 - z^{-1})^2$ is shown in Figure 10.35. The next step is to design the OTA. In view of its several advantages, we use a two-stage feedforward-compensated design, whose macro-model is shown in Figure 10.36. The NTF

$$A(s) = A_{dc} \frac{N(s)}{D(s)}$$
$$N(s) = (1 + \frac{s}{\omega_z})$$
$$D(s) = (1 + \frac{s}{\omega_{p1}})(1 + \frac{s}{\omega_{p2}})$$
$$A_{dc} = 4120, \omega_z = 6.43, \omega_{p1} = 0.1875, \omega_{p2} = 0.0625$$
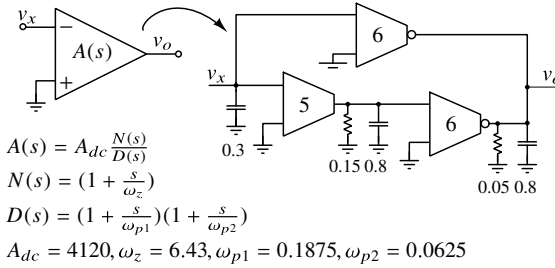
Figure 10.36 Macromodel of the two-stage feedforward-compensated OTA. (Trans)conductances (in Mho's) and capacitance (in Farads) are marked.

of the loop with real OTAs can then be determined. As seen in Figure 10.37, it is nowhere close to what we intended. Some of the NTF's poles have apparently moved close to the unit circle, causing significant peaking in its magnitude response. This makes sense – given that the integrators have become slower, one should expect that the NTF has degraded due to the extra delay added by the loop-filter.
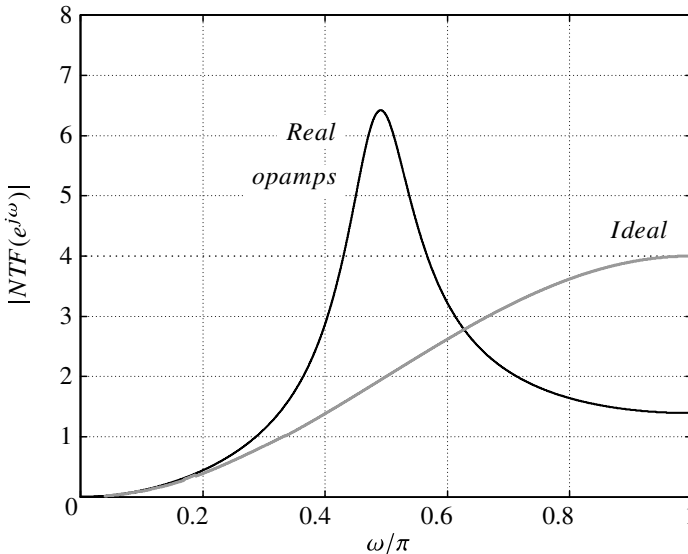
Figure 10.37 The modulator's NTF with ideal and real OTAs.

How do we "fix" the loop-filter so as to get back our original NTF? One way is to design OTAs that are a whole lot faster, but this is not a power efficient solution. Another approach is to adjust the component values so that the desired NTF is achieved with the

n	5	15	25
k_0	0.8803	0.9670	1.2136
k_1	0.7579	0.7000	0.5350
k_2	1.8707	1.9308	2.0348

Table 10.1   Modulator coefficients obtained by solving (10.32) for different values of n.

OTAs we currently have. In our second-order example of Figure 10.35, we attempt to vary k_0, k_1, and k_2 so as to make the NTF approach $(1 - z^{-1})^2$ as closely as possible.

A tempting, albeit incorrect, approach to determine K is the following. As before, we determine the sampled pulse responses of the direct, first- and second-order paths – this time with the real OTAs. These are obtained from a transient simulation of the schematic after layout parasitic extraction. K is found by fitting the sampled pulse response of the continuous-time loop-filter to that of the discrete-time prototype according to

$$\underbrace{\begin{bmatrix} l_0[n] & l_1[n] & l_2[n] \end{bmatrix}}_{\substack{\text{known from simulation} \\ \text{(schematic or layout)}}} K = l[n]. \tag{10.32}$$

The same as earlier, there are $(n+1)$ equations in 3 unknowns. With ideal integrators, the solution for K was unique regardless of n. With real OTAs, however, it turns out that this is no longer true, as Table 10.1 shows. The coefficients vary significantly, and consequently, so does the resulting NTF, as seen in Figure 10.38. What are the "right" coefficients to use? All in all, this way of finding K does not inspire confidence.
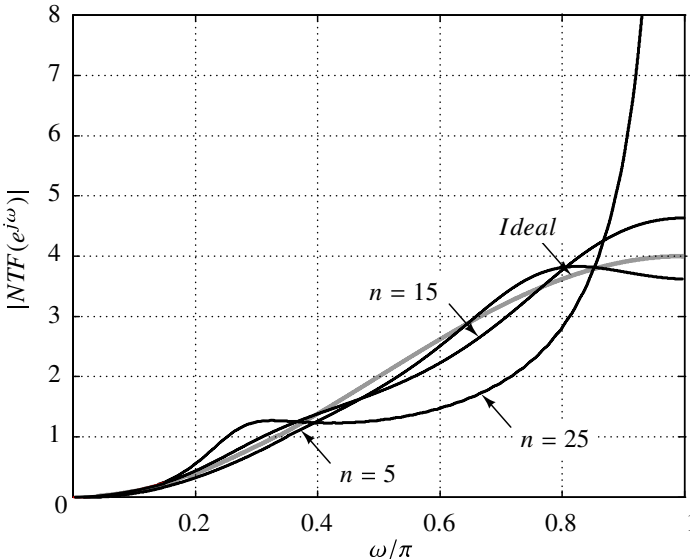
Figure 10.38   Computed NTFs with K determined using $n = 5, 15, 25$ in (10.32).

Why does this happen? In reality, the OTA-RC integrator has finite gain, as well as multiple poles and zeros due to the OTA's internal parasitics. Thus, the loop-filter in our second-order example, which should ideally be of second-order, is a high-order system.

Equation (10.32), which attempts to fit the pulse response of the (high-order) continuous-time response to that of the second-order discrete-time prototype, can only do an approximate job. Thus, we should *not* expect a unique solution for K. More important, it turns out that the set of equations (10.32) is ill-conditioned. As a result, K varies wildly with N, and is not appropriate. The problems with this technique are solved by the closed-loop fitting method [4] discussed next.

10.8.1 Closed-Loop Fitting

The idea behind this technique is to attempt to fit $NTF(z)(1 + L(z))$ to unity, rather than the open-loop pulse response of the continuous-time filter to $l[n]$, as described earlier. The NTF of the CT$\Delta\Sigma$M is related to its equivalent discrete-time loop filter transfer function by

$$\underbrace{NTF(z)}_{h[n]} = \cfrac{1}{1 + \cfrac{L(z)}{\underbrace{k_0 l_0[n] + k_1 l_1[n] + k_2 l_2[n]}}}. \qquad (10.33)$$

Writing this in the time domain, we have

$$h[n] + (k_0 l_0[n] + k_1 l_1[n] + k_2 l_2[n]) * h[n] = \delta[n], \qquad (10.34)$$

where $h[n]$ denotes the impulse response corresponding to the NTF, and $*$ denotes convolution. Denoting $h_0[n] = l_0[n] * h[n]$, $h_1[n] = l_1[n] * h[n]$, $h_2[n] = l_2[n] * h[n]$, we obtain

$$h[n] + k_0 h_0[n] + k_1 h_1[n] + k_2 h_2[n] = \delta[n]. \qquad (10.35)$$

Thus,

$$\begin{bmatrix} h_0 & h_1 & h_2 \end{bmatrix} K = \delta[n] - h[n]. \qquad (10.36)$$

The set of equations (10.36) can be solved to determine K. The coefficients obtained with different N are shown in Table 10.2.

N	5	15	25
k_0	0.9023	0.9003	0.8988
k_1	0.7420	0.7423	0.7425
k_2	1.9093	1.9010	1.8951

Table 10.2 Modulator coefficients obtained by solving (10.36) for different values of N.

Figure 10.39 shows the magnitudes of the NTF calculated with coefficients obtained using (10.36) for $N = 5$, 15, and 25. They are almost indistinguishable, and close to what we desire. This indicates that the proposed technique does a good job of approximating the desired NTF. The inset in Figure 10.39 compares the in-band behavior of the tuned NTF with that of the ideal NTF (whose slope must be 40 dB/decade). Below $\omega/\pi \approx 0.005$, the tuned NTF exhibits first-order behavior, due to integrator finite gain.

Why is it that the open-loop fitting method is virtually unusable, but the closed-loop technique robust? $l_0[n]$, $l_1[n]$, and $l_2[n]$ are very sensitive to the position of those poles of $L(z)$ that are close to the unit circle. For instance, if the integrators were ideal, $l_2[n] \propto n$ for large n, while finite gain integrators result in a $l_2[n] \to 0$ (for large n). Since the least-squares solution of (10.32) minimizes the norm of $[l_0 \quad l_1 \quad l_2]K - l$, and the error in l_2
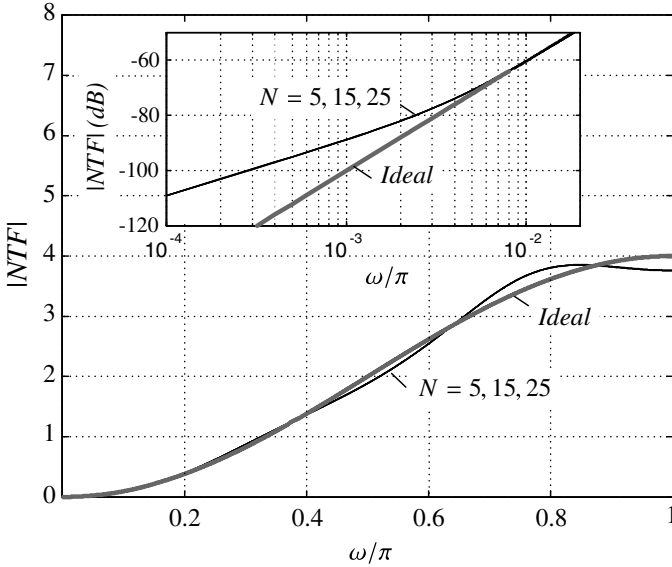
Figure 10.39   The magnitudes of the NTF: the ideal NTF and the NTFs with coefficients tuned as per (10.36) for N =5, 15, 25. The inset compares the in-band behavior of the tuned NTF with that of the ideal NTF – below $\omega/\pi \approx 0.005$ the tuned NTF exhibits first-order behavior due to finite integrator gain.

due to finite integrator gain increases greatly with n, the coefficient k_2 increases with N (confirmed by the trend in Table 10.1). To reduce the error for large n (by using a large k_2), k_1, and k_0 also have to change with n. It is thus seen that the primary reason for the undesirable behaviour of the coefficients extracted using (10.32) is the sensitivity of l_0, l_1, and l_2 to the locations of their poles that are close to the unit circle.

Yet, h_0, h_1, and h_2 in (10.36) are less sensitive to changes in l_0, l_1, and l_2 due to the following. For simplicity, consider an NTF with all its zeros at $z = 1$. If the integrators were ideal, $h_i[n] = l_i[n] * h[n]$ would be FIR, since the zeros of the NTF would cancel the poles of $L_i(z)$. If the locations of those poles of $L_i(z)$ near the unit circle are perturbed by Δz, the pole-zero cancellation is not exact, but the change in h_i is negligible (even though the effect on l_i is dramatic). Figure 10.40 shows l_2 and h_2 in the second-order example of Figure 10.34 for two cases: one where the OTA's dc gain is infinite, and another where it is 35. Though there is a significant difference in l_2, there is virtually no change in h_2.

To summarize, finite bandwidth effects of the OTAs in the loop-filter can dramatically alter the NTF, or even render it unstable. Fortunately, this can be mitigated by tuning component values so that the sampled pulse response of the loop-filter mimics the impulse response of the discrete-time prototype, while considering finite OTA bandwidths. The tuning procedure suggested in (10.36) above is convenient and robust. The $l_i[n]$ can be obtained from a short transient simulation of the schematic or layout extracted netlist, thereby accounting for layout parasitics and nonidealities of the DAC pulse shape. Thanks to convolution of the l_i's with h, the coefficients obtained are largely independent of the number of samples used in the least-squares fit of (10.36). Tuning coefficients using this method, therefore, should result in an NTF that is close to the desired one. It is possible
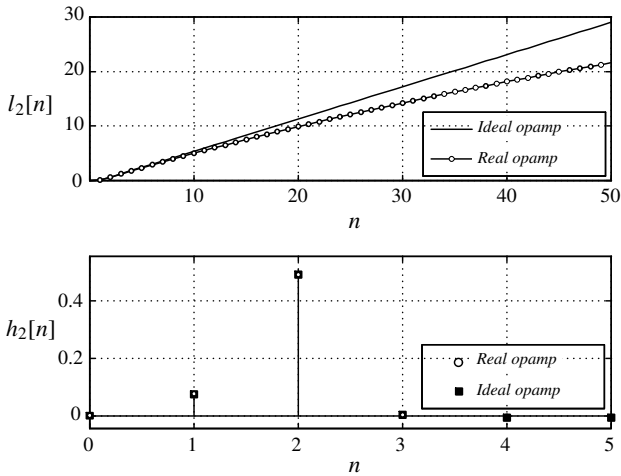
Figure 10.40 $l_2[n]$ and $h_2[n]$ with ideal and real opamps. Though $l_2[n]$ changes significantly due to opamp nonidealities, $h_2[n]$ changes very little.

that the act of tuning k_0, k_1, and k_2 modify the l_i due to changes in the loading of the integrators. This is a second-order effect, and it can be mitigated, if necessary, by going through another tuning iteration.

A question that arises from the discussion above is the following. Given that finite bandwidth effects in the OTAs can be "fixed" by appropriate coefficient tuning, can power dissipated in the CTΔΣM be reduced by deliberately using slow OTAs? While this is a valid argument, one should be aware that overdoing it has several undesirable consequences. The design becomes less robust on two counts. First, the NTF is more sensitive to variations in the OTA bandwidth. It also becomes more sensitive to parasitic capacitances. The latter makes sense due to the following. An active-RC integrator is insensitive to stray capacitors only when the OTA is ideal so that the potential of the virtual ground node is zero. With a finite-bandwidth OTA, this is no longer true – and less so for lower OTA bandwidths. Another potential problem with using low OTA bandwidths is distortion. As we will see in Section 10.9, the nonlinear currents injected by the OTAs is proportional to the cubes of the voltages at their internal nodes. Since reduced OTA bandwidth means higher swings at the virtual ground (and other internal nodes), the in-band noise due to weak loop-filter nonlinearities will increase. OTA bandwidths should therefore be chosen as a compromise between these conflicting requirements: power dissipation, on the one hand, and distortion on the other. As engineers, we are no strangers to such trade-offs.

10.9   Loop-Filter Nonlinearities in Continuous-Time Delta-Sigma Modulators

Until this point, we have assumed that the loop-filter is perfectly linear. In practice, it is weakly nonlinear. The resulting CTΔΣM model is shown in Figure 10.41, where quantization noise is assumed to be additive. The natural question that arises is how nonlinearity degrades the modulator's performance, and what (if anything) one can do to address it.
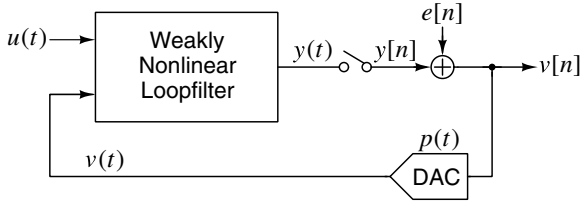
**Figure 10.41**   Model of a CTΔΣM with a weakly nonlinear loop-filter.
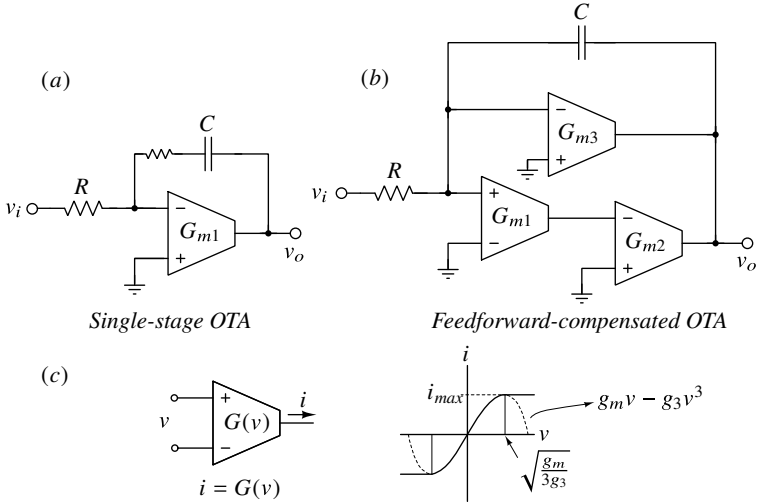
Figure 10.42   Simplified models of weakly nonlinear (a) single-stage and (b) feedforward-compensated OTAs. (c) Simplified model of the weakly nonlinear transconductor.

Before we dive deeper, let us understand the ways in which nonlinearity manifests in the loop-filter. The OTAs are comprised of transconductors, as shown in Figure 10.42. If we assume fully differential operation and weak nonlinearity, the output current of every transconductor can be related to its input voltage as follows.

$$i = G(v) = \begin{cases} g_m v - g_3 v^3 & , \ |v| \le \sqrt{\frac{g_m}{3g_3}} \\ \pm i_{max} & , \ otherwise. \end{cases}$$

Weak nonlinearity means that the voltage at the input of every transconductor is sufficiently small, so that the third-order distortion component $|g_3 v^3| \ll |g_m v|$. These assumptions are a simplification, but they yield useful insights about the modulator performance in the presence of nonlinearity. The question we wish to answer is: given g_m and g_3 for every transconductor in the loop-filter, what is the in-band SQNR of the CT$\Delta\Sigma$M ? To gain intuition without being drowned in notation, we illustrate with CT-MOD1, shown Figure 10.43(a). The integrator uses a single-stage OTA. It has finite dc gain, with input and output parasitic capacitances c_m and c_o, respectively, as shown in Figure 10.43(b).
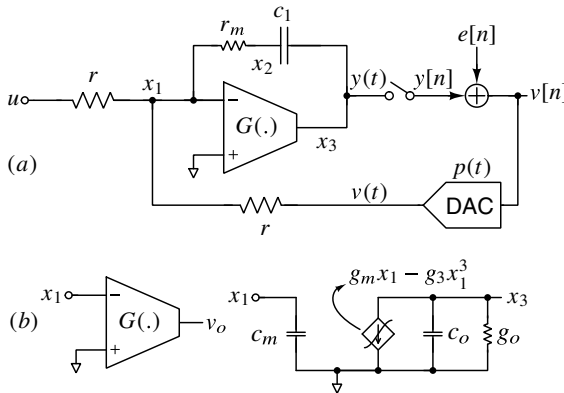
Figure 10.43   (a) CT-MOD1 with a weakly nonlinear integrator and (b) OTA model.

The system of Figure 10.43(a) is a weakly nonlinear one excited by two inputs – $u(t)$ and $e[n]$. We need to determine $v[n]$. To do this, we first write the nodal equations governing the operation of the loop-filter as follows. The voltages at the internal nodes are

denoted by x_1, x_2, x_3.

$$\overbrace{\begin{bmatrix} c_m & 0 & 0 \\ 0 & c_1 & -c_1 \\ 0 & -c_1 & c_1 + c_o \end{bmatrix}}^{C\dot{x}} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} + \overbrace{\begin{bmatrix} (2g + g_m) & -g_m & 0 \\ -g_m & g_m & 0 \\ g_m & 0 & g_o \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}}^{Gx} + \overbrace{\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ -g_3 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1^3 \\ x_2^3 \\ x_3^3 \end{bmatrix}}^{G_3 x^3}$$

$$= \underbrace{\begin{bmatrix} g & g \\ 0 & 0 \\ 0 & 0 \end{bmatrix}}_{\begin{bmatrix} F_1 & F_2 \end{bmatrix}} \begin{bmatrix} u(t) \\ v(t) \end{bmatrix}$$

$$v(t) = \underbrace{\sum_n y[n]p(t - nT_s)}_{y_{dac}(t)} + \underbrace{\sum_n e[n]p(t - nT_s)}_{e(t)}.$$

In matrix form, the equations describing the modulator can thus be expressed as

$$\begin{aligned} C\dot{x} + Gx + G_3 x^3 &= \begin{bmatrix} F_1 & F_2 \end{bmatrix} \begin{bmatrix} u(t) & v(t) \end{bmatrix}^T, \\ v(t) &= y_{dac}(t) + e(t), \end{aligned} \qquad (10.37)$$

where

- x is the column vector of node voltages in the loop-filter;

- C is the capacitance matrix, G and G_3 are the conductance matrices and F_1 and F_2 are the input matrices;

- x^3 denotes the column vector of the cubes of the node voltages;

- $y_{dac}(t) = \sum_n y[n]p(t - nT_s)$ and $e(t) = \sum_n e[n]p(t - nT_s)$. In this particular example, $y[n] = x_3[nT_s]$. In general, it can depend on other node voltages.

The equations (10.37) represent a set of coupled nonlinear differential equations that represent the operation of the modulator, which is a system excited by $u(t)$ and $e(t)$. The term that introduces nonlinearity is G_3, which in general, renders the equations hard to solve. However, under the assumption of weak nonlinearity, where the terms in $G_3 x^3$ are small in magnitude when compared to those in Gx and $C\dot{x}$, (10.37) can be solved in an approximate manner. The intuition behind the solution is the following familiar observation.

Consider a perfectly linear amplifier with gain k_1, excited by an input u. Denote its output by $y = k_1 u$. If the amplifier is now excited by an input αu, the output is $\alpha y = \alpha k_1 u$. What happens when the amplifier is weakly nonlinear, with a transfer curve given by $y = k_1 u + k_3 u^3$? When u to such an amplifier is scaled by α, the output is given by

$$\hat{y} = \underbrace{\alpha k_1 u}_{linear\ component} + \underbrace{\alpha^3 k_3 u^3}_{third\ order\ component}. \qquad (10.38)$$

We see that $\hat{y}$ consists of a "linear" term that scales as α, and a component (arising from the nonlinearity) that scales as α^3. For a more general amplifier characteristic with a gently

saturating odd nonlinearity, the equation above is a good approximation in the sense that higher order nonlinear components can be neglected as long as u is sufficiently small.

Returning to CT-MOD1, and in the spirit of the discussion above, we assume that the $x(t)$ (the node voltage vector) can be expressed as the sum of linear and nonlinear components, according to

$$x(t) \approx \underbrace{x^{(1)}(t)}_{\text{linear component}} + \underbrace{x^{(3)}(t)}_{\substack{\text{third order} \\ \text{nonlinear component}}}. \tag{10.39}$$

The nonlinear component $x^{(3)}(t)$, which is a consequence of G_3, must consist largely of third-order distortion components, since the OTA exhibits cubic nonlinearity. The key point behind the solution is to ask what happens to v if the inputs to the system (u and e) were scaled by α. If the loop-filter was perfectly linear, x and v should also simply scale by α. Due to nonlinearities in the loop-filter, however, the linear part of v and x scale by α, while the third-order distortion components will scale by α^3. Thus,

$$x(t) \approx \alpha x^{(1)}(t) + \alpha^3 x^{(3)}(t). \tag{10.40}$$

Using this in (10.37), with scaled inputs αu and αe, we have

$$C\left[\alpha \dot{x}^{(1)} + \alpha^3 \dot{x}^{(3)}\right] + G\left[\alpha x^{(1)} + \alpha^3 x^{(3)}\right] + G_3\left[\alpha x^{(1)} + \alpha^3 x^{(3)}\right]^3$$
$$= \begin{bmatrix} F_1 & F_2 \end{bmatrix}\left[\alpha u(t) \quad \alpha y_{dac}^{(1)}(t) + \alpha^3 y_{dac}^{(3)}(t) + \alpha e(t)\right]^T.$$

Equating the coefficients of the first and third powers of α on both sides of the equation above, we obtain

$$C\dot{x}^{(1)} + Gx^{(1)} = \begin{bmatrix} F_1 & F_2 \end{bmatrix}\left[u(t) \quad y_{dac}^{(1)}(t) + e(t)\right]^T, \tag{10.41}$$

$$C\dot{x}^{(3)} + Gx^{(3)} + G_3(x^{(1)})^3 = \begin{bmatrix} F_1 & F_2 \end{bmatrix}\left[0 \quad y_{dac}^{(3)}(t)\right]^T. \tag{10.42}$$

The systems of equations above are linear. The first set corresponds to CT-MOD1, where the OTA is replaced by its linear counterpart, obtained by setting $g_3 = 0$ as shown in Figure 10.44(a). Solving this set yields $x^{(1)}(t)$, and thereby $y_{dac}^{(1)}$ and $v^{(1)}[n]$. In other words, $v^{(1)}[n]$ is the output of CT-MOD1 with an input $u(t)$, and its loop-filter's nonlinearity turned off.

Equations (10.42) also represent a linear CT-MOD1; however, u and e for this modulator are set to zero. Instead, its internal nodes are excited by the currents that would have been generated by $x^{(1)}(t)$ acting on the cubic nonlinearities in the loop-filter, as shown in Figure 10.44(b). $x^{(1)}(t)$ is known, since these voltages correspond to the node voltages of CT-MOD1 where the OTA is linear. Solution of this modulator yields $x^{(3)}(t)$, and the output sequence $v^{(3)}[n]$. $v^{(3)}$ depends on $[x^{(1)}]^3$. $x^{(1)}$ is a linear function of u and e. Thus, $[x^{(1)}]^3$ consists of signal distortion, noise floor increase due to mixing of shaped quantization noise with itself, and cross-products (e.g., $u \cdot e^2$ and $u^2 \cdot e$).

The output of the weakly nonlinear system of Figure 10.41 is thus given by

$$v[n] \approx v^{(1)}[n] + v^{(3)}[n], \tag{10.43}$$

with $v^{(1)}[n]$ and $v^{(3)}[n]$ being obtained from the systems of Figures 10.44(a) and (b), respectively. While the discussion above illustrated the mechanism by which the in-band
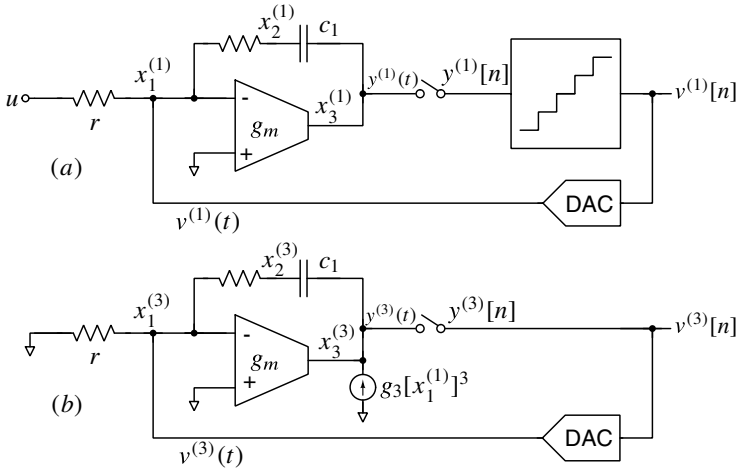
Figure 10.44   (a) CT-MOD1 with loop-filter nonlinearity turned off. (b) Nonlinear currents are injected into CT-MOD1, with $u = 0$, and the quantizer bypassed.

performance of CT-MOD1 degrades due to the nonlinear OTA, the same principles apply to a high-order CTΔΣM.

In summary, to determine the effect of weak loop-filter nonlinearities on CTΔΣM performance, we proceed as follows [5].

a. Determine the output sequence of the CTΔΣM $v^{(1)}[n]$ with all nonlinear effects removed, namely by setting $g_3 = 0$ for all nonlinear elements.

b. Nonlinear currents $g_3[x_1^{(1)}]^3$ is injected into a linear "modulator", with input u set to zero, and the quantizer bypassed. The output sequence of this modulator is $v^{(3)}[n]$.

c. Compute the PSD of $v^{(1)}[n] + v^{(3)}[n]$ to estimate the in-band SNR of the modulator – this now accounts for quantization noise as well as weak nonlinearities in the loop-filter.

From the discussion above, we see that the in-band PSD of $v^{(1)}[n]$ must correspond to that of the ideal modulator, while that of $v^{(3)}[n]$ models the degradation due to nonlinear effects in the loop-filter. Since $v^{(3)}[n]$ is obtained by analyzing CT-MOD1 by *injecting nonlinear currents* into it, this way of analysis is called *the method of current injection.*

Figure 10.45(a) shows a third-order CIFF CTΔΣM that employs a nine-level quantizer, and samples at 6.144 MHz. The OTAs are two-stage feedforward-compensated designs. The component values are given for a modulator normalized to have a 1-Hz sampling rate and integrating resistors of 1 Ω. Part (b) of the figure shows the ideal PSD, where the in-band SNDR is about 125 dB in a 24 kHz bandwidth. With weakly nonlinear OTAs, the SNDR is degraded to 91 dB. SPICE and the current injection method are in good agreement. A similar agreement is seen in the SNDR plot (Figure 10.46).
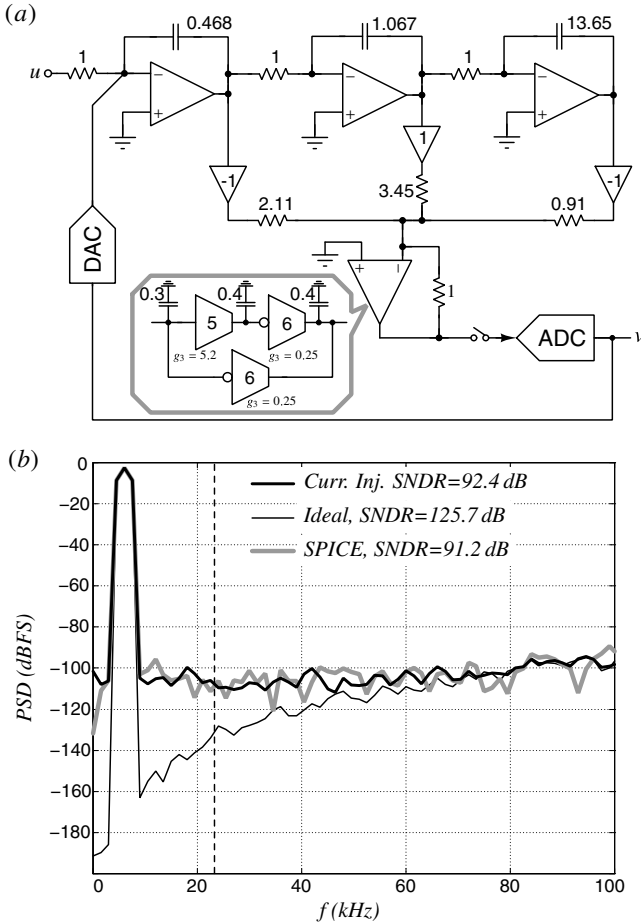
Figure 10.45 (a) A third-order CIFF CTΔΣM, with components normalized for a 1-Hz sampling rate and impedance level of 1 Ω. (b) Low-frequency PSD of the ideal modulator, compared with those obtained from SPICE and using the method of current injection.
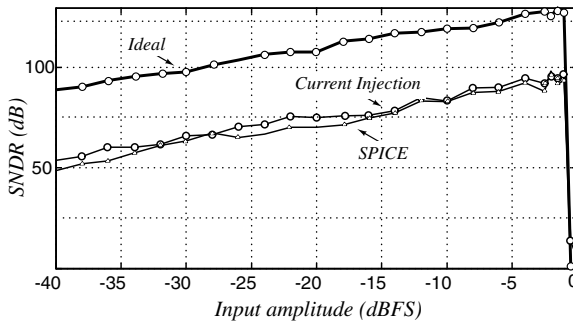
Figure 10.46 SNDR of the ideal CTΔΣM compared with a modulator with weak loop-filter nonlinearities. The current injection method is in good agreement with SPICE simulations.

10.9.1   Circuit Techniques to Improve Loop-Filter Linearity

Earlier in this section, we saw how the input signal and quantization noise interact through nonlinearities in the loop-filter and degrade the SNDR of a CTΔΣM. The key to low distortion operation is to reduce the nonlinear currents injected by the transconductors that comprise the loop-filter. This can be accomplished in several ways. A "brute-force" approach is to use (multi-stage) OTAs with large transconductances, which results in smaller swings on the internal nodes of the OTAs. As a result, the strengths of the nonlinear currents injected by the transconductors that comprise the OTA are reduced, resulting in a smaller "noise" due to nonlinearity.
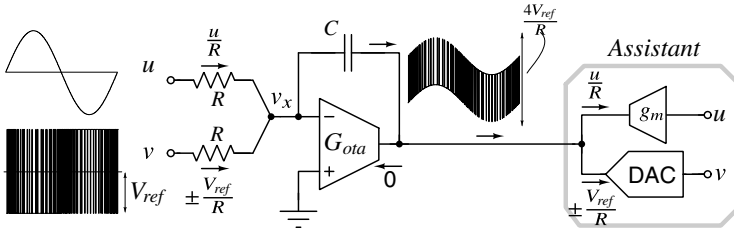

Figure 10.47   Principle of the assisted opamp integrator.

Another technique, exploiting feedforward, to reduce distortion due to nonlinearity, is discussed below. For illustration, we use an OTA-RC integrator that forms the input integrator of a single-bit CTΔΣM, as shown in Figure 10.47. For the time being, let us assume that the circuitry marked "assistant" does not exist. The switched-resistor NRZ DAC feeds back a rail-to-rail waveform. If the OTA were ideal ($G_{ota} \rightarrow \infty$), the virtual ground potential would be zero, and the current through the integrating capacitor would be $(u + v)/R$, which would in turn be sunk by the OTA. In practice, v_x would need to swing due to the finite transconductance of the OTA. Finite bandwidth of the OTA would make matters worse – the rail-to-rail steps in the feedback waveform would necessitate large swings at OTA's virtual ground. Such large swings, by the intuition gained from the "current injection" method, result in significant nonlinear currents that degrade CTΔΣM linearity.

One can avoid the problem of a large virtual ground swing by recognizing the following. The current that the OTA needs to sink is known, since the feedback sequence v and the modulator input u are readily accessible. Thus, the current $(u + v)/R$ can be generated by the "assistant" circuitry, shown to the right of Figure 10.47. A transconductor $g_m = 1/R$ generates the input component u/R. The DAC component v/R is generated by a current-steering DAC. Thus, the assistant supplies the current that the OTA would otherwise be called upon to sink, so no current flows into the OTA [6]. Thanks to this, v_x remains zero, and speed and distortion problems are avoided. In practice, the OTA would have to only sink the mismatch between assistant and input currents.

How do noise and distortion of the assistant impact the linearity of the integrator? These are not problematic, since noise and distortion are injected at the *output* of the OTA. When referred to the input u, these errors are reduced by a factor of about RG_{ota} (which, in a good integrator, needs to be large anyway).

How does the use of assistance impact the stability of the integrator? It is easy to see that it does not – if u and v are set to zero, the assistant DAC and transconductor are

removed from the picture, and the integrator reduces to the one without assistance. This means that the pole locations of the integrator are not affected by the assistant circuitry. Last, the assistant circuitry dissipates power. When a single-bit DAC is used, the increase in power consumption due to the assistant is fairly small since the opamp is relieved from supplying the difference between the instantaneous input and feedback currents, which is large. However, when multi-bit quantization or FIR DACs are used, the currents generated in the assistant can be larger than those supplied by the opamp without assistance.

10.10   Case Study of a 16-Bit Audio Continuous-Time Delta-Sigma Modulator

This section describes the design of a CTΔΣM attempting to achieve 16-bit resolution in a 24 kHz bandwidth [7]. The process technology used is a 180-nm CMOS process, supporting a supply voltage of 1.8 V. The first aspect that must be carefully deliberated is the architecture to adopt. The process technology is decidedly fast for this application – even for $OSR = 128$, the resulting sampling frequency is only $f_s = 6.144$ MHz. Many potential choices of order, sampling rate and number of quantizer levels present themselves. How does one navigate one's way through this dizzying maze of possible design choices? Unfortunately, there is no easy answer to this question. The discussion that follows is one valid approach, justified by the results eventually obtained with measurements. Having said this, it is recommended that one should be able to rigorously and strongly defend his/her design choices.

High resolution CTΔΣMs have traditionally been realized using multi-level quantizers. To see why this is so, let us examine the arguments that favor such an approach.

a. *Lower Sampling Rate:* The sampling rate needed to achieve a desired in-band SQNR decreases with increasing number of quantizer levels. Further, the NTF can be made more aggressive, resulting in a further reduction in the sampling rate.

b. *Reduced Clock Jitter Sensitivity:* The reduced step-size in the feedback waveform (assuming an NRZ DAC pulse) results in reduced sensitivity to clock jitter, as discussed in Section 9.5.

c. *Improved Loop Filter Linearity:* With a multi-level DAC, the difference between the input and feedback waveform is small. This means that the peak magnitude of the signal processed by the loop filter is small, which results in improved linearity for a given power dissipation.

The arguments above are indeed compelling. However, there are reasons why we do not favor the use of a multi-bit quantizer. When a flash ADC is used to digitize the loop-filter's output, the complexity of the ADC used in the quantizer increases exponentially with the number of bits. Further, as seen in the discussion accompanying Figure 10.21, random offset in the comparators needs to be restricted to a small fraction of the step size. This will most likely need some form of offset correction, increasing power dissipation and design complexity. Even though the comparators in the quantizer lend themselves to low-power operation, clock generation and distribution can consume significant current. This, unfortunately, is difficult to estimate during the architectural design phase. Moreover, mismatch in the unit elements of the feedback DAC degrade the in-band SNDR of the modulator,

necessitating mismatch correction, like calibration or dynamic element matching (DEM). This further increases the power dissipation and design time of the quantizer. In contrast, using a single-bit quantizer, where the feedback DAC is inherently linear, dramatically simplifies the quantizer design. Comparator offset is also not problematic. The output of the loop-filter can be scaled without affecting the output sequence, simplifying the design of the integrator that drives the comparator. However, the full-scale two-level feedback waveform places increased demands on the linearity of the loop-filter, and the sensitivity of the modulator to clock jitter.

From the discussion above, it is seen that a multi-bit loop complicates the quantizer design at the expense of a simplified loop-filter. The opposite is true in a single-bit modulator. Recognizing this, several recent works have attempted to alleviate the linearity and clock jitter problems associated with a single-bit design. One approach is to use integrators based on opamp assistance, as discussed earlier in this chapter. An assisted opamp integrator addresses the linearity issue but does not remedy problems due to clock jitter.

Another possible approach is to use a single-bit ADC and an FIR feedback DAC. As we saw in Section 9.5, the reduced step-size in such a DAC not only reduces jitter sensitivity of the modulator, but also relaxes the linearity requirements of the loop-filter. In practice, the filter and DAC combination are implemented in a semi-digital fashion, as shown in Figure 9.31(b), which makes the FIR DAC inherently linear in spite of mismatch. Due to the single-bit quantizer, the ADC design is simple and consumes very little power. A modulator employing a single-bit quantizer and an FIR DAC, therefore, combines the best features of single-bit and multi-bit operation. FIR feedback adds delay, and will destabilize the modulator if it is not properly compensated. In the previous chapter, we saw how the NTF of a loop with FIR feedback can be restored exactly.

Based on the considerations above, we decide to go with a single-bit modulator, with an FIR feedback DAC. Further, to reduce idle-tone problems, we choose to implement a third-order loop. A maximally flat NTF with an out-of-band gain of 1.5 and optimally spread zeros yields a peak SQNR of 110 dB for an OSR of 128. Deeming this adequate, we proceed to choose the loop-filter topology.
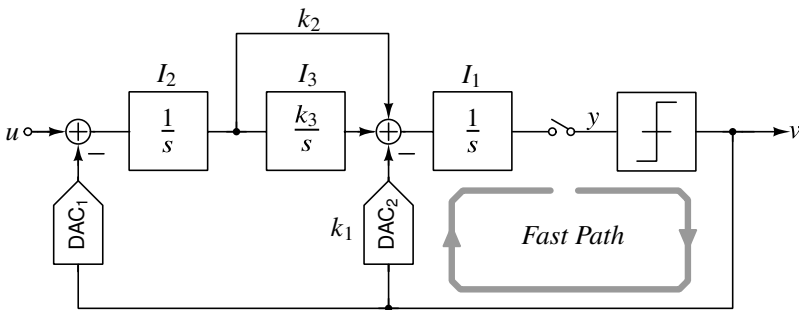

Figure 10.48   Single-loop prototype, $f_s = 1\,Hz$.

The architecture of the prototype modulator on which the FIR-CTΔΣM is based is the single-loop design shown in Figure 10.48. Weak feedback around I_1 and I_3, needed to achieve complex NTF zeros, is not shown. The third-order loop-filter is realized as a

cascade of integrators with feedforward and feedback (CIFF-B). As discussed in Chapter 8, this architecture has several advantages. The fast path around the quantizer (through DAC_2 and I_1) and the high-gain path (through I_1, I_2, and I_3) can be independently optimized, like in a CIFB design. Due to feedforward, the output of I_2 has virtually no signal at the input frequency. This means that its gain in the signal band (after dynamic range scaling) will be large. Thanks to this, nonidealities in the rest of the loop-filter will be significantly attenuated when referred to the modulator input, just like in a CIFF design. The CIFF-B architecture, therefore, inherits the appealing aspects of both its parents. The gain of the loop-filter from v to y is

$$L_{1,ct}(s) = \frac{k_1}{s} + \frac{k_2}{s^2} + \frac{k_3}{s^3}.$$ (10.44)

$$\hat{k}_2 = k_2 + k_3\left(\tfrac{N-1}{2}\right)$$
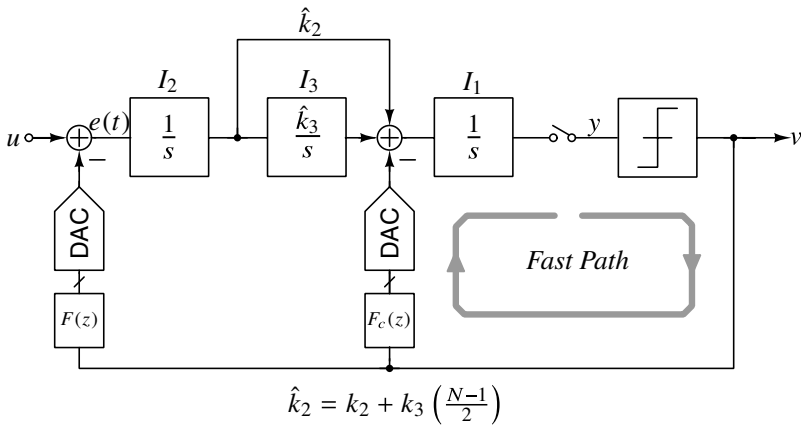
Figure 10.49   Normalized CTΔΣM prototype incorporating the FIR DAC. $F_c(z)$ represents the compensation DAC. All taps are assumed to be identical.

The outermost feedback DAC in the prototype is then replaced by an N-tap FIR DAC, whose transfer function is denoted by $F(z)$, as shown in Figure 10.49. All taps of the FIR filter are made identical for ease of layout. Further, the dc gain of $F(z)$ should be 1, to ensure that the in-band STF of the CTΔΣM is unity. A compensation FIR DAC with transfer function $F_c(z)$ is added at the input of I_1. We wish to stabilize the loop so that the NTFs of the modulators in Figures 10.48 and 10.49 are identical. As we have seen in Chapter 9,

a. $\hat{k}_3 = k_3$,

b. $\hat{k}_2 = k_2 + \frac{k_3}{2}(N-1)$,

c. $F_c(z)$ is an N-tap FIR filter.

where all hatted quantities refer to the modulator with the FIR DAC.

Thus, by tuning k_2, and adding $F_c(z)$ at the input of I_1, the NTF can be restored *exactly*. Once this is recognized, the coefficients of F_c can be determined analytically, or by using the numerical techniques of Section 10.8.

10.10.1   Choice of Number of Taps in the FIR DAC

It is tempting to increase the number of taps in the main FIR DAC without limit, since a longer FIR DAC results in better filtering of the quantization noise. Due to this, one would tend to conclude that the magnitude of the error signal $e(t)$ (in Figure 10.49) would reduce as N increases. As a result, the loop-filter processes a smaller signal, and better linearity should be expected. This is true for small N– better filtering of the quantization noise does indeed reduce $|e(t)|$. However, beyond a certain point, $|e(t)|$ no longer decreases due to the following. As seen earlier, $\hat{k}_2$ increases with N. Further, it also turns out that the dc gain of $F_c(z)$ increases with N. Both of these are undesirable. A larger $\hat{k}_2$ increases the peaking in the signal transfer function, causing the input component of the fed back signal $v(t)$ to be larger in magnitude and shifted in phase with respect to $u(t)$. This means that even though the quantization noise component of $v(t)$ is smaller due to better filtering, the peak $|e(t)|$ starts to become *larger* when N is increased beyond a certain value. The increased dc gain of $F_c(z)$ is problematic in a practical implementation, as the input signal component injected by the compensation DAC necessitates a lower unity-gain frequency for I_3 (after dynamic range scaling). Further, the power dissipation of the clock generation and distribution circuitry increases with the number of taps.
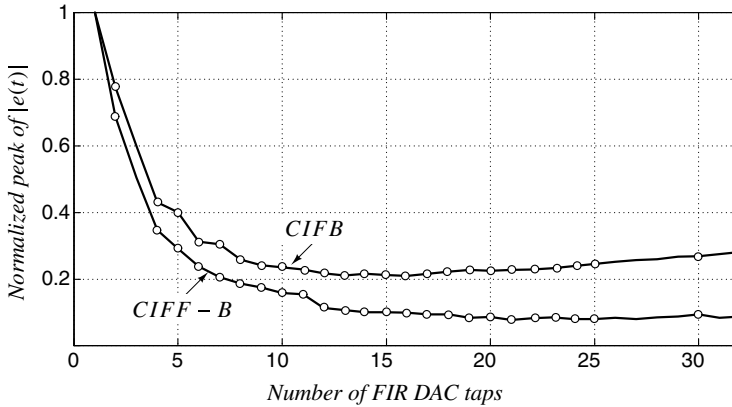
Figure 10.50   Peak magnitude of the input to the loop-filter as a function of number of taps in the FIR filter, for the CIFB and CIFF-B architectures.

The "optimal" number of taps to be used in the FIR DAC, therefore, is dependent on the loop-filter topology (which influences the STF) and the input signal frequency. In a CIFF-B design, it is a trade-off between the amount of STF peaking one is willing to tolerate, the increased power dissipation due to the extra taps (without a corresponding decrease in $|e(t)|$) and the dc gain of $F_c(z)$, which has implications for the design of I_3. Simulation results of $e(t)$ (normalized to the value that would be obtained without an FIR filter), as shown in Figure 10.50 for CIFB and CIFF-B loop-filters, are used as a guide to decide that 12-tap FIR DACs represent a reasonable choice considering the trade-offs involved.

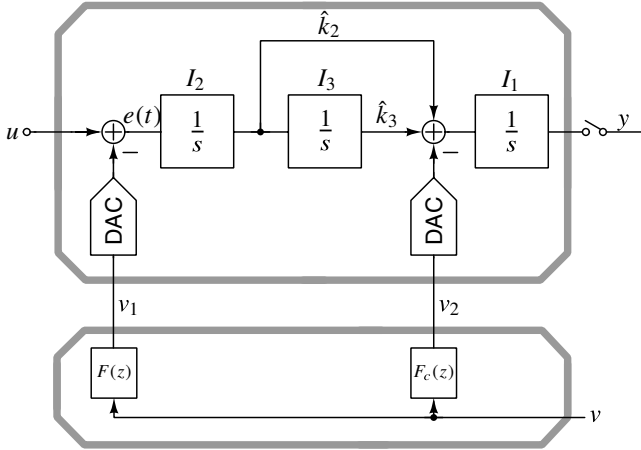## 10.10.2 State-Space Modeling and Simulation with an FIR DAC

Figure 10.51   Treating the loop-filter and FIR DACs as a composite system.

How does one model and simulate a CTΔΣM with an FIR DAC using the ΔΣ toolbox? To see this, we redraw the loop-filter of our modulator as shown in Figure 10.51. Conceptually, it consists of a continuous-time part that is driven by the outputs of the main FIR filter $F(z)$ and the compensating FIR filter $F_c(z)$, denoted by v_1 and v_2, respectively. The state matrices of the continuous-time loop-filter of Figure 10.51 are given by

$$A_c = \begin{bmatrix} 0 & \hat{k}_2 & \hat{k}_3 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad , \quad B_c = \begin{bmatrix} 0 & 0 & -1 \\ 1 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

$$C_c = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \quad , \quad D_c = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}.$$

The filter and DACs can be, as usual, discretized. We denote the resulting state vector and matrices by x and A_d, B_d, C_d, D_d, respectively.

The FIR filters, both of which have N-taps, introduce $(N - 1)$ new states into the system. The FIR DACs can be represented as a single-input, two-output system as shown in Figure 10.52. The states of this system are denoted by

$$x_{fir} = \begin{bmatrix} x_{f1} & \cdots & x_{f,(N-1)} \end{bmatrix}^T \tag{10.45}$$

and the corresponding state equations are
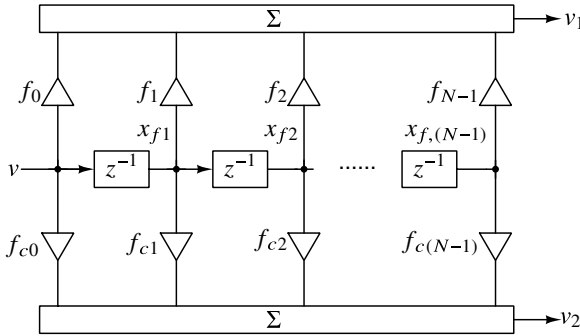
Figure 10.52   State-space representation of the main and compensation FIR DACs.

$$
x_{fir}[n+1] = \overbrace{\begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix}}^{A_{fir}} x_{fir}[n] + \overbrace{\begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}}^{B_{fir}} v[n],
\tag{10.46}
$$

$$
\begin{bmatrix} v_1[n] \\ v_2[n] \end{bmatrix} = \underbrace{\begin{bmatrix} f_1 & f_2 & \cdots & f_{N-2} & f_{N-1} \\ f_{c1} & f_{c2} & \cdots & f_{c,(N-2)} & f_{c,(N-1)} \end{bmatrix}}_{C_{fir}} x_{fir}[n] + \underbrace{\begin{bmatrix} f_0 \\ f_{c0} \end{bmatrix}}_{D_{fir}} v[n].
$$

The loop-filter and FIR DACs form a larger discrete-time system, whose state-space representation can be derived from the state matrices of the individual systems as shown below.

The discrete-time equivalent of the loop-filter is written as

$$
x[n+1] = A_d x[n] + B_{d1} u + B_{d23} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix},
\tag{10.47}
$$

$$
y[n] = C_d x[n],
$$

where B_{d1} represents the first column of B_d and B_{d23} is a matrix that is comprised of the second and third columns of B_d. Using (10.46) in (10.47), the state equations of the composite structure in Figure 10.51 can be expressed as

$$
\begin{bmatrix} x[n+1] \\ x_{fir}[n+1] \end{bmatrix} = \begin{bmatrix} A_d & B_{d23}C_{fir} \\ 0 & A_{fir} \end{bmatrix} \begin{bmatrix} x[n] \\ x_{fir}[n] \end{bmatrix} + \begin{bmatrix} B_{d1} & B_{d23}D_{fir} \\ 0 & B_{fir} \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix},
$$

$$
y[n] = \begin{bmatrix} C_d & 0 \end{bmatrix} \begin{bmatrix} x[n] \\ x_{fir}[n] \end{bmatrix}.
\tag{10.48}
$$

The zero entries in the matrices above are submatrices of appropriate order. With the state-space representation of the loop-filter in hand, the power of the `simulateDSM` routine in the $\Delta\Sigma$ toolbox can now be harnessed.
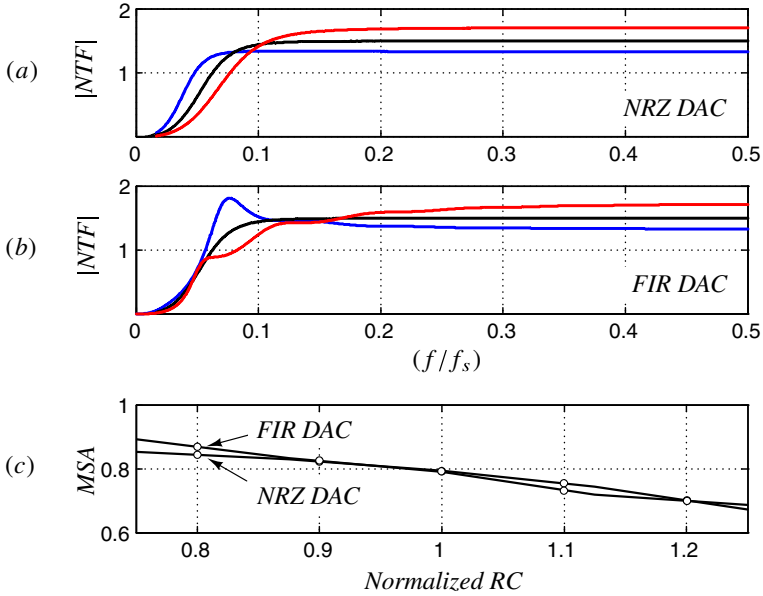
Figure 10.53   NTF magnitudes as a function of ±25% time-constant variation for (a) a conventional CTΔΣM and (b) a CTΔΣM with a 12-tap FIR feedback DAC. (c) The maximum stable amplitude as a function of time-constant variation.

10.10.3   Effect of Time-Constant Variations

A concern regarding a CTΔΣM-based FIR DAC is the effect of time-constant variations on the NTF, and the maximum stable amplitude (MSA). To examine this, simulations were run on CTΔΣMs with and without an FIR DAC. Under nominal conditions, both modulators were designed to have the same maximally flat NTF with an out-of-band gain of 1.5. All time-constants were then changed over a ±25% range from their nominal values. The resulting NTFs and MSAs for the conventional and FIR-CTDSM are shown in Figure 10.53. It is seen that the high-frequency gains are similar in both cases. Since the high frequency gain of the NTF largely influences the MSA, it makes sense that both designs have similar MSAs as time-constants are varied. From this, we conclude that a CTΔΣM with an FIR DAC is not any more sensitive to time-constant variations than its NRZ counterpart.

10.10.4   Modulator Architecture

The NTF of the CTΔΣM is chosen to be maximally flat, with an out-of-band gain of 1.5, as per Lee's rule. The full-scale is 3.6 V (peak-to-peak differential), corresponding to an external reference voltage of 1.8 V. Simulations show that the maximum stable amplitude is about −3 dBFS.

A simplified single-ended schematic of the third-order CTΔΣM is shown in Figure 10.54. As discussed earlier, a CIFF-B loop-filter is used. Negative resistors indicate inversion in the differential version. Active-RC integrators are used for low noise and high linearity. FIR DACs inject currents proportional to the reference into the virtual grounds
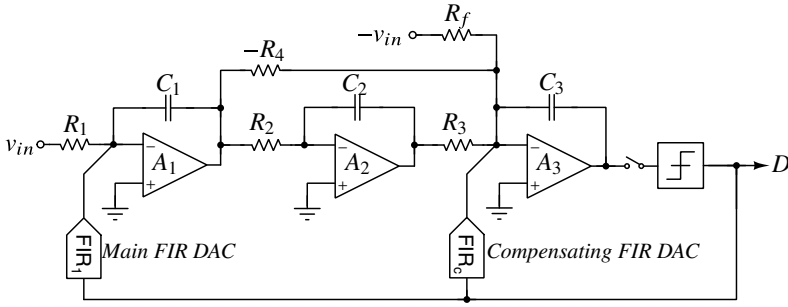
$$R_1 = 30K\Omega, R_2 = 1M\Omega, R_3 = 500K\Omega, R_4 = 1.02M\Omega, R_f = 400K\Omega$$
$$C_1 = 10.4pF, C_2 = 3.8pF, C_3 = 1.25pF$$

Figure 10.54   Simplified single-ended schematic of the modulator.

of the opamps. Resistive DACs are used for low-noise operation. All tap weights of the main FIR DAC (FIR_1) are equal. As a result, the unit resistors are $12R_1$. The input-referred thermal noise (of the differential loop-filter) due to R_1 and FIR_1 is about $7\,\mu V$ (rms). The impedance levels of the second and third integrators can be significantly increased without affecting the in-band noise. As seen from Figure 10.54, R_2 and R_3 are about 32 and 16 times larger than R_1. Thanks to this, A_2 and A_3 can also be impedance scaled, reducing power dissipation.
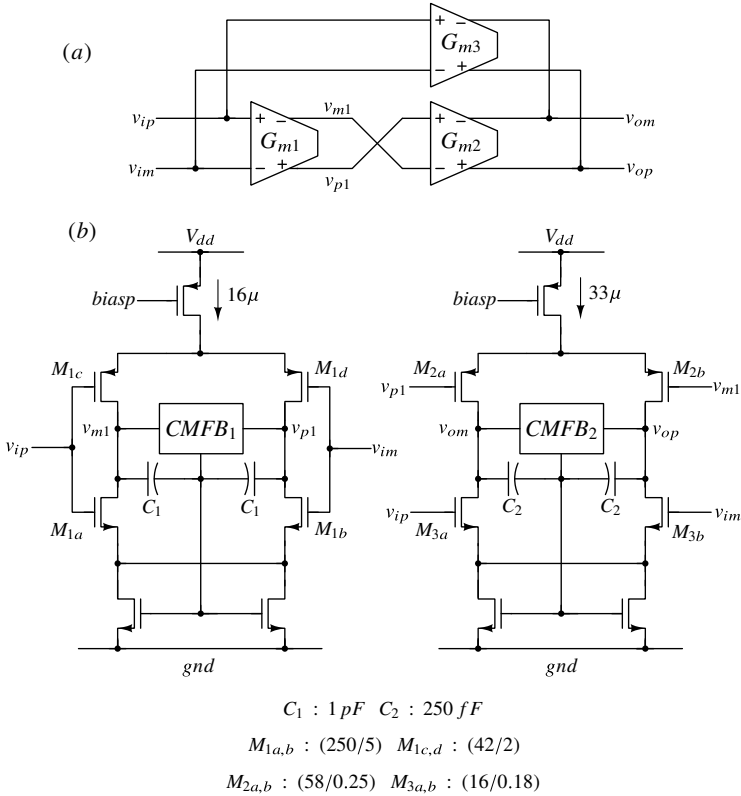
R_f feeds the input into the third integrator formed by A_3-R_3-C_3. Without R_f, the output of A_2 would consist of a component proportional to v_{in}. Using R_f to cancel the low frequency output of FIR_c (which is largely proportional to v_{in}) avoids this problem, enabling the use of a much smaller C_2. Weak feedback around A_2 and A_3 by adding a large resistor (not shown in Figure 10.54 for clarity) from the ouput of the latter to the inverting terminal of the former moves two zeros of the NTF from dc to an optimal location in-band. To avoid the use of a large resistor, a T-network is employed by reusing the CMFB sense resistors of A_3.

Thanks to single-bit operation, the loop-filter's output can be scaled without affecting the output sequence. This is advantageous in practice, since the opamp (A_3) need not be designed to handle large swings at its output.

All resistors and capacitors are realized as switchable banks, to counter RC time-constant variations. The opamps are two-stage feedforward-compensated structures. Thanks to the low frequency of operation (in relation to the speed of the process), excess loop delay is negligible. Since the FIR DACs are realized in semi-digital fashion, they are inherently linear: resistor mismatch only alters the transfer function of the FIR filter, which is virtually inconsequential to modulator performance. Details of the individual blocks are given below.

10.10.5   Opamp Design

The noise and linearity of the first opamp in the loop-filter are critical. The simplified block diagram and schematic of A_1 are shown in Figures 10.55(a) and (b), respectively. A two-

$$C_1 : 1\,pF \quad C_2 : 250\,fF$$
$$M_{1a,b} : (250/5) \quad M_{1c,d} : (42/2)$$
$$M_{2a,b} : (58/0.25) \quad M_{3a,b} : (16/0.18)$$

Figure 10.55 (a) Macromodel and (b) simplified schematic of the two-stage feedforward compensated OTA used in the first integrator.

stage feedforward-compensated architecture is chosen to achieve high dc gain and unity-gain bandwidth. The first stage, whose signal path is formed by transistors $M_{1,a} - M_{1,d}$, reuses the input stage current, resulting in low noise for a given power dissipation. The large sizes of the input devices are dictated by considerations of $1/f$ noise. The second gain stage G_{m2}, is formed by $M_{2a,b}$. $M_{3a,b}$ form G_{m3}.

Thanks to the 12-tap FIR DAC, the linearity requirements of the opamp are greatly relaxed, thereby enabling the use of a relatively small bias current in the opamp's second stage. The current in the second stage is reused to realize G_{m2} and G_{m3}. The common-mode voltage at the output of each stage is stabilized by separate loops. C_1 and C_2 are needed to compensate these common-mode feedback (CMFB) loops.
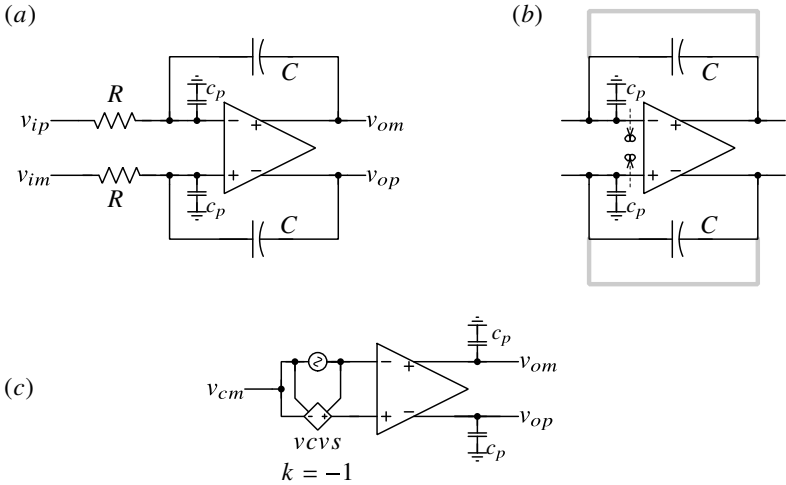

Figure 10.56   (a) OTA-RC integrator, with the OTA's input parasitic capacitance c_p.   (b) Determining the loop-gain at frequencies much higher than the unity-gain frequency of the integrator. (c) Setup to simulate the OTA's gain.

A question that arises during the OTA design phase is – "What feedback factor should one use to analyze OTA stability?" In our modulator, the OTA forms part of the integrator, as shown in Figure 10.56(a). The parasitic input capacitance of the OTA, which is not negligible, is denoted by c_p. As we have seen earlier in this chapter, the unity-gain frequency of the integrator ($\approx 1/RC$) has to be much smaller than the unity-gain bandwidth of the OTA. Therefore, as far as the feedback loop enclosing the OTA is concerned, the integrating capacitors C can be treated as short-circuits at frequencies around the OTA's unity-gain bandwidth, as shown in Figure 10.56(b). By a similar argument, the influence of the resistors on the loop-gain at high frequencies can be neglected. It is thus seen that the OTA should be stable in a unity-feedback configuration.

The input parasitic capacitance c_p loads the second stage of the OTA. To determine loop-gain, the loop has to be conceptually broken to the right of c_p, as shown in Figure 10.56(b). To simulate the loop-gain, therefore, it is appropriate to use the setup shown in part (c) of the figure. In the design of Figure 10.55, $c_p \approx 0.5\,\text{pF}$.

The simulated magnitude and phase responses of the OTA are shown in Figure 10.57. The dc gain is about 70 dB, and the unity-gain bandwidth is seen to be about 89 MHz. The
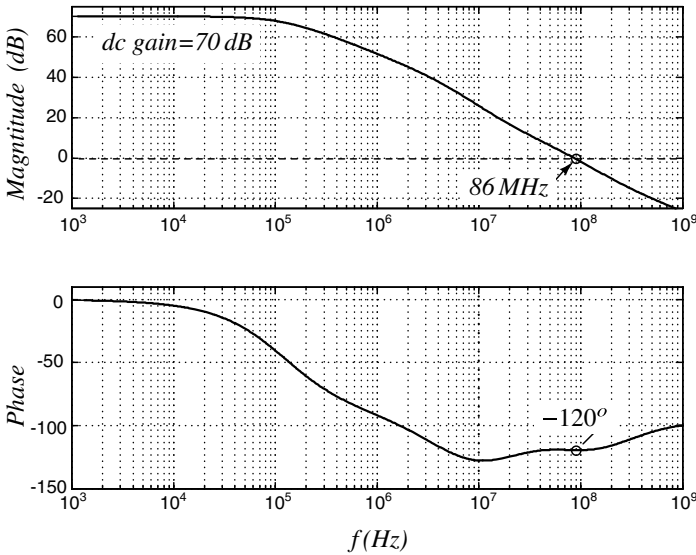
Figure 10.57   Magnitude and phase plots of the first OTA.

phase margin is about 60 degrees. Thus, an integrator realized using the OTA will have a high-frequency parasitic pole at about 89 MHz. This is approximately equivalent to a delay of about $1/(2\pi \cdot 89\,\text{MHz}) \approx 1.8\,\text{ns}$.
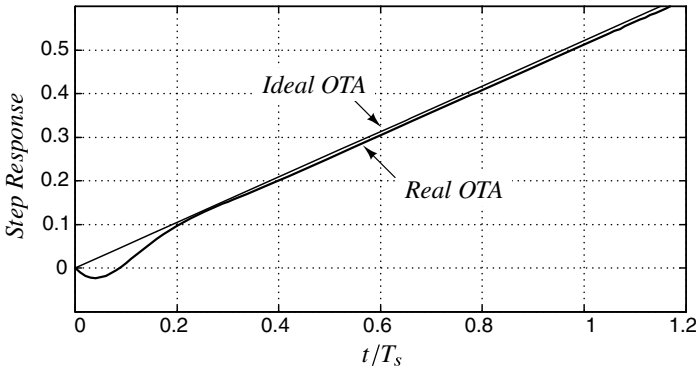
Figure 10.58   Step response of the first integrator. The response obtained with an ideal OTA is also shown for comparison.

The OTA is embedded into the integrator. The response to a unit-step is shown in Figure 10.58. The response with an ideal OTA is also shown for comparison. The undershoot when a real OTA is used is due to the right-half-plane zero in the integrator transfer function caused by the feedforward effect of the integrating capacitor C. The delay with respect to the output of an ideal integrator is about 2 ns, consistent with the OTA's 89 MHz unity-gain bandwidth.

10.10.6   ADC and FIR DACs

A sense-amplifier-based comparator was used to make the single-bit decision on the loop-filter output. The circuit is similar to that shown in Figure 10.18, except that the reference storage capacitors (C_c) and the corresponding switches are not needed (due to 1-bit operation). C^2MOS flip-flops were used to realize the digital portion of the FIR DACs. The main and compensating FIR DACs share the same flip-flops.

One tap of the FIR DAC
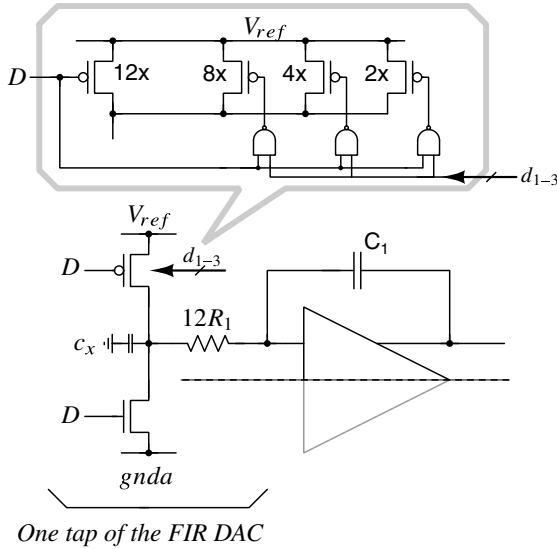
Figure 10.59   Using a digitally programmable PMOS device enables programming to reduce distortion due to rise-fall asymmetry.

The FIR DACs are resistive and are implemented in a semi-digital fashion. Rise-fall asymmetry can cause even-order distortion due to inter-symbol-interference (ISI), as in a conventional NRZ DAC. Straightforward analysis shows that the use of FIR filtering *does not* mitigate this problem. Figure 10.59 shows a single-ended section of one tap of the main FIR DAC. D is the single-bit decision of the quantizer. As we discussed in Section 10.7.1, ISI is primarily due to mismatch between the resistances of the pull-up and pull-down transistors, and causes even-order distortion in the current injected by the DAC unit element. Fortunately, this distortion is canceled due to differential operation. In practice, mismatches will cause a small fraction of ISI-induced distortion to leak into the differential output. In view of the extremely low levels of distortion desired in this work, and the lack of reliable information on resistor matching, the modulator was designed so that the single ended feedback current waveforms have inherently low distortion. This is accomplished by ensuring roughly equal on-resistances for the pull-up and pull-down devices under nominal conditions. To account for process variations, the PMOS switches are realized as a 3-bit bank, as shown in Figure 10.59.
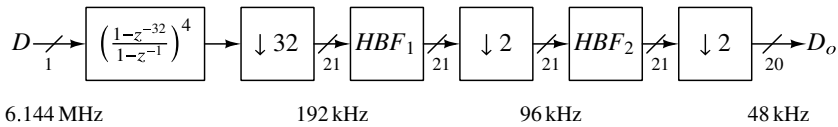
D ⟶ $\left(\dfrac{1-z^{-32}}{1-z^{-1}}\right)^{4}$ ⟶ ↓ 32 ⟶ HBF_1 ⟶ ↓ 2 ⟶ HBF_2 ⟶ ↓ 2 ⟶ D_o

6.144 MHz   192 kHz   96 kHz   48 kHz

Figure 10.60   Block diagram of the decimation filter.

10.10.7   Decimation Filter

The block diagram of the decimation filter is shown in Figure 10.60. The output of the decimation filter is twenty-bits wide, and is at the Nyquist rate (48 kHz). The first stage is a fourth-order 32-tap CIC filter, implemented as a Hogenauer structure (see Chapter 14.) This is followed by two halfband FIR lowpass filters, each of whose outputs are downsampled by a factor of 2. The filter orders are 16 and 60, respectively. The twenty-bit decimation-filter output is transferred off-chip through a serial peripheral interface (SPI). The decimator consumes 100 μW from the 1.8 V supply.

10.11   Measurement Results
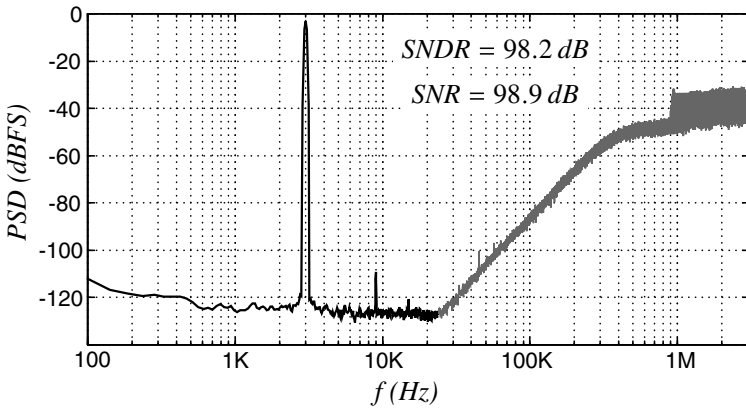

**Figure 10.61**   Measured PSD of the modulator output for the input amplitude that results in peak SNDR.

The CT$\Delta\Sigma$M, decimator and an SPI interface (to facilitate data transfer to/from the chip) were fabricated in a 0.18 μm CMOS process. The active area of the chip, including the decimator, is 1.25 mm × 1 mm.

Figure 10.61 shows the PSD of the modulator at the amplitude that results in the peak SNDR. The peak SNR, SNDR, and dynamic range (DR) are 98.9 dB, 98.2 dB, and 103 dB, respectively. In audio applications, an oft-quoted performance metric is the A-weighted SNR. The design achieves a peak A-weighted SNR and SNDR of 102.3 dB and 101.5 dB, respectively. The third harmonic distortion is seen to be about −106 dB. The PSD computed at the output of the decimation filter gives virtually the same performance, indicating that the decimator is working as intended.
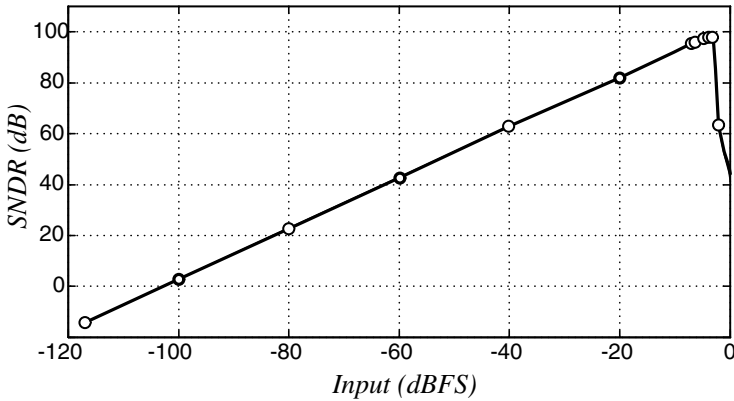
Figure 10.62   SNDR as a function of input amplitude.

Figure 10.62 shows the plot of SNDR as a function of input amplitude. The modulator dissipates 280 μW from a 1.8 V supply, which translates to a Schreier FoM of 182.3 dB.
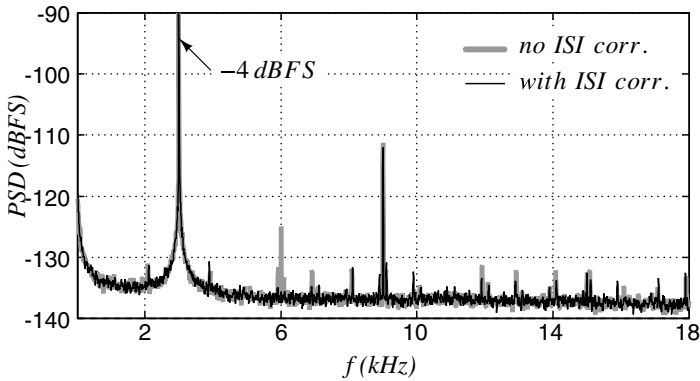
Figure 10.63   PSD with and without ISI correction.

Figure 10.63 shows the measured spectra of a -4 dBFS input with and without ISI correction. The reduction in HD_2 with correction is apparent. The inherently low HD_2 (even without correction) indicates good resistor matching in this process.

10.12   Summary

In this chapter, we discussed circuit design considerations for CT$\Delta\Sigma$Ms. We began by considering various options for realizing the integrators in the loop-filter. We concluded that an OTA-based active-RC structure is an excellent choice to realize the input integrator. Gm-C integrators, which are faster but less linear, are (only) suitable as inner integrators. We studied several choices for realizing the OTA. We found that with feedforward-compensated

multi-stage structures, it is possible to realize of high bandwidths in a power-efficient manner.

We then examined thermal noise in CTΔΣMs and concluded that *if the loop-filter is time-invariant*, the input-referred in-band (thermal) noise spectral density of the modulator is virtually the same as the in-band noise spectral density of the loop-filter. We saw that this was due to the "inherent anti-aliasing" property of a CTΔΣM (with a time-invariant loop-filter).

The in-band noise of a CTΔΣM consists of two independent components – the shaped quantization error and thermal noise. How one should partition a given total noise budget into these two components was discussed next. Since reducing quantization noise "costs" much less power than reducing thermal noise, we concluded the latter should account for most of the noise budget in a power efficient CTΔΣM design. A rule of thumb is to keep (in-band) quantization noise at least 10-12 dB below the (in-band) thermal noise.

We then looked into choices for the design of comparators. We saw that comparator offset can be problematic, especially when the SQNR-SNR margin is small. This often necessitates additional circuitry for comparator offset correction.

The next topic of discussion was the design of feedback DACs. We examined the merits of various DAC pulse shapes, and several methods of implementing them. In particular, we studied trade-offs associated with resistive and current-steering DACs. We discussed properties of CTΔΣMs with switched-capacitor feedback and found that such DACs not only degrade the modulator's linearity due to the high peak-to-average ratio of the feedback waveform but also severely degrade its alias-rejection.

Once the modulator is put together with "real" blocks, the NTF that is realized is bound to be different from the one that was originally intended. Fortunately, this can be remedied by tuning coefficients. We described a robust numerical technique to accomplish this.

We then gained an understanding of the effects of weak loop-filter nonlinearities on CTΔΣM performance, and discussed ways of addressing them. Finally, we described the design of a third-order, single-bit audio CTΔΣM designed to achieve 16-bit performance in a 24 kHz bandwidth. The modulator, which uses a 12-tap FIR feedback DAC, achieves a peak SNDR of 98.2 dB. Implemented in a 180 nm CMOS process, it consumes 280 μW from a 1.8 V supply, yielding FoM_s = 182.3 dB.

References

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

[2] M. Ortmanns, F. Gerfers, and Y. Manoli, "A continuous-time ΣΔ modulator with reduced sensitivity to clock jitter through SCR feedback," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 52, no. 5, pp. 875–884, 2005.

[3] S. Pavan, "Alias rejection of continuous-time modulators with switched-capacitor feedback DACs," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 58, no. 2, pp. 233–243, 2011.

[4] S. Pavan, "Systematic design centering of continuous-time oversampling converters," *IEEE Transactions on Circuits and Systems II: Express Briefs,*, vol. 57, no. 3, pp. 158–162, 2010.

[5] S. Pavan, "Efficient simulation of weak nonlinearities in continuous-time oversampling converters," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 57, no. 8, pp. 1925–1934, 2010.

[6] S. Pavan and P. Sankar, "Power reduction in continuous-time delta-sigma modulators using the assisted opamp technique," *IEEE Journal of Solid-State Circuits*, vol. 45, no. 7, pp. 1365–1379, 2010.

[7] A. Sukumaran and S. Pavan, "Low power design techniques for single-bit audio continuous-time delta sigma ADCs using FIR feedback," *IEEE Journal of Solid-State Circuits*, vol. 49, no. 11, pp. 2515–2525, 2014.

CHAPTER 11

# BANDPASS AND QUADRATURE DELTA-SIGMA MODULATION

Previous chapters described $\Delta\Sigma$ converters in which the highest frequency of interest was a small fraction of the sample rate. This chapter shows how $\Delta\Sigma$ converters can also be used to digitize narrowband signals containing frequencies that are an appreciable fraction of the sampling rate. The resulting *bandpass* and *quadrature bandpass* converters preserve many of the advantages of ordinary lowpass $\Delta\Sigma$ converters and are particularly attractive in wireless receiver systems.

## 11.1   The Need for Bandpass Conversion

Figure 11.1 shows five digital receiver architectures. In the *superheterodyne* architecture, the incoming radio-frequency (RF) signal is repeatedly filtered, amplified, and downconverted before being digitized and sent to a digital signal processor (DSP). This architecture is able to achieve a high degree of selectivity without using high-Q filters since filtering is applied repeatedly at progressively lower frequencies. Also, because unwanted signals, which in a wireless receiver can be much larger than the wanted signal, are removed prior to digitization, and since the wanted signal is at low frequency, the ADC requirements are modest. The price paid for these advantages is complexity: such receivers typically have several stages of analog filtering and mixing before analog-to-digital conversion.

The *direct-conversion* system architecture depicted in Figure 11.1(b) is considerably simpler since there is only one downconversion operation, although that downconversion
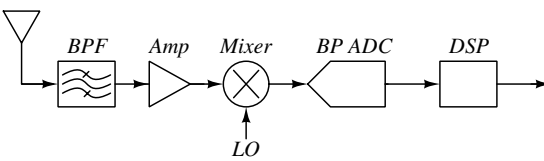
(a) Superheterodyne Receiver
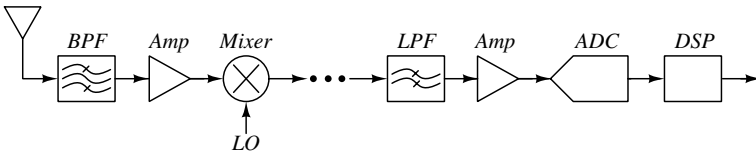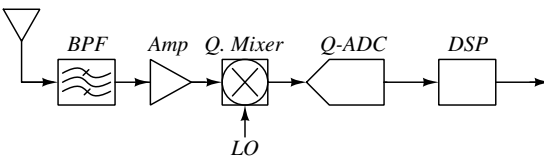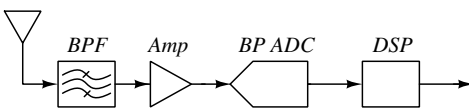

(b) Direct Conversion Receiver
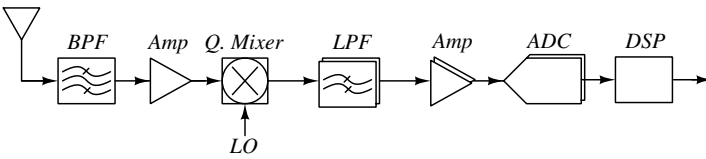
(c) Receiver using a Bandpass ADC at the first IF

(d) Receiver using a Quadrature ADC at the first IF


(e) Receiver using an RF Bandpass ADC

Figure 11.1   Receiver architectures.

does require a quadrature mixer. One disadvantage of this approach is that the receiver's ability to distinguish signals located above the local oscillator (LO) frequency from those below the LO frequency is limited by imperfect quadrature in the mixer and imbalance in the subsequent baseband circuits. Another disadvantage is that the receiver is vulnerable to dc offset, $1/f$ noise and even-order distortion products since the desired signal is at baseband. With adaptive digital signal processing it is possible to remove the dc offset while losing only a few kHz of bandwidth, and to correct I/Q mismatch with sufficient accuracy to achieve 70–80 dB of steady-state image rejection. However, the signal processing associated with wideband I/Q mismatch correction is complex and the $1/f$ noise and even-order distortion problems remain.

Using a bandpass ADC (Figure 11.1(c)) or quadrature bandpass ADC (Figure 11.1(d)) to digitize the first IF (intermediate frequency) reduces the complexity of a superheterodyne architecture to that of a direct conversion receiver without incurring the power and complexity penalties of adaptive digital processing. Furthermore, since the signal is at an IF, $1/f$ noise and dc offset are unimportant, and even-order distortion is less problematic.

Lastly, Figure 11.1e depicts the ultimate in architectural simplicity. This architecture dispenses with analog downconversion entirely and digitizes the RF signal using an RF bandpass ADC. In addition to its raw simplicity, this architecture also allows fast frequency-hopping because reprogramming a bandpass ADC is typically faster than changing the LO frequency. The primary barrier to using a bandpass ADC as a receiver is the converter's center-frequency range. Bandpass ADCs with center frequencies in the low GHz range have been reported in the technical literature [1]-[4], but since the current commercial limit is 450 MHz [5], bandpass conversion is currently more suited to IF rather than RF digitization. Nonetheless, the progress made over the last decade suggests that commercial parts capable of digitizing GHz RF signals will be available soon.
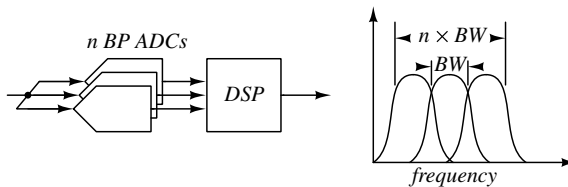
Figure 11.2   Frequency-interleaving.

As a final motivation for bandpass conversion, consider the *frequency-interleaved* system depicted in Figure 11.2. Just as time-interleaved ADCs achieve wideband operation by staggering the sampling times of multiple low-speed ADCs, a frequency-interleaved ADC staggers the center frequencies of multiple bandpass ADCs to digitize a wide swath of frequencies. Both forms of interleaving are sensitive to mismatch, but the resulting impairments are qualitatively different. In a time-interleaved ADC mismatch causes spurs and noise, whereas in a frequency-interleaved ADC, mismatch is either benign (if the bands are not stitched together), or merely results in a non-flat frequency response. For a receiver, linearity and spurious-free dynamic range are of paramount importance whereas transfer-function flatness is secondary and thus frequency-interleaving offers a promising way to construct ADCs for wideband receivers.

11.2 System Overview
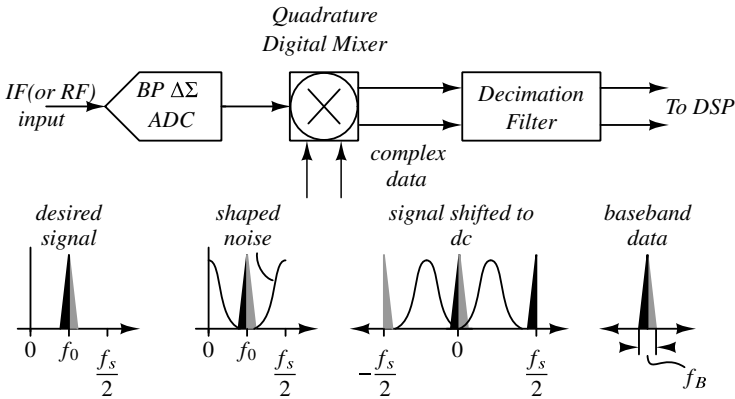
Figure 11.3   A bandpass $\Delta\Sigma$ ADC system.

Figure 11.3 depicts a bandpass $\Delta\Sigma$ ADC system as well as the spectra of its key signals. As this figure indicates, the input to the ADC is either an IF or RF signal and the output of the ADC contains the desired signal surrounded on either side by shaped quantization noise. The digital output of the bandpass modulator is mixed to dc by a digital quadrature mixer, and then lowpass-filtered and decimated by a quadrature lowpass digital decimation filter to produce complex baseband digital data.

The oversampling ratio of a bandpass ADC is defined in the same way as it is for a standard (lowpass) ADC, namely

$$OSR = \frac{f_s}{2f_B},\tag{11.1}$$

where f_s is the sampling frequency and f_B is the width of the band of interest. Note that the center frequency f_0 does not appear in (11.1), and thus the oversampling ratio can be large even if the ratio f_s/f_0 is not.

As indicated in Figure 11.3, the output data rate needed to support a bandwidth f_B is also f_B, since the output signal is complex. Thus, in a bandpass ADC system, the sample rate can therefore be reduced by a factor as large as $2 \cdot OSR$, rather than the OSR factor that applies in the lowpass case.

Just as a bandpass modulator can exploit the narrowband character of its input, a *quadrature $\Delta\Sigma$ modulator* can exploit the additional information available in a quadrature signal.[1] Figure 11.4 illustrates the main signal-processing operations that occur inside a quadrature $\Delta\Sigma$ ADC system. A quadrature signal, such as that produced by a quadrature mixer, is applied to a quadrature $\Delta\Sigma$ modulator which in turn produces a digital quadrature output containing the desired signal plus shaped quantization noise. The distinguishing feature of a quadrature $\Delta\Sigma$ modulator is that its quantization-noise stopband need only exist

---

[1] A quadrature signal consists of two real signals, commonly denoted either by I (for in-phase) and Q (for quadrature phase), or by re (for real) and im (for imaginary). In contrast to a real signal, the spectrum of a quadrature signal need not be symmetric about zero frequency, that is, positive and negative frequencies are distinct. Section 11.6 discusses quadrature signals and quadrature filters in more detail.
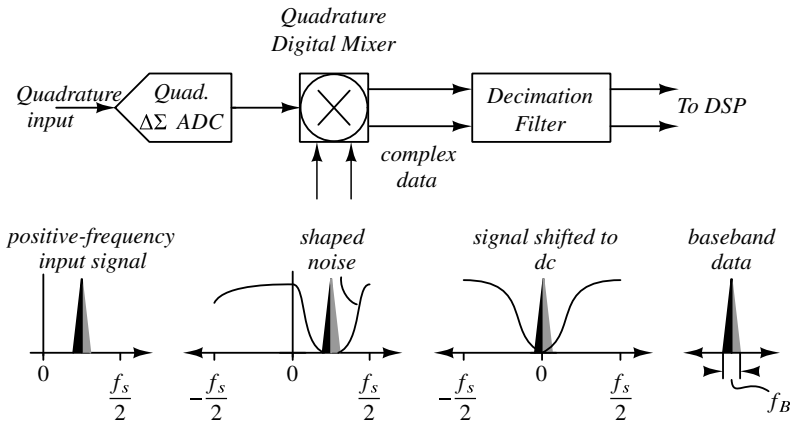
Figure 11.4   A quadrature $\Delta\Sigma$ ADC system.

at positive (or negative) frequencies. In a sense, a quadrature converter is more efficient than a bandpass converter because no power is wasted digitizing the negative-frequency content of the input. As in a bandpass $\Delta\Sigma$ ADC system, a quadrature modulator's output is mixed to baseband by a digital quadrature mixer and filtered by a quadrature decimation filter to produce Nyquist-rate baseband data.

The Nyquist band for a quadrature system is $[-f_s/2, f_s/2]$, and thus the total information bandwidth is f_s. In order for $OSR = 1$ to correspond to no oversampling, the oversampling ratio of a quadrature system is defined as

$$OSR = \frac{f_s}{f_B}. \tag{11.2}$$

In other words, for a given signal bandwidth and sampling rate, the OSR of a quadrature modulator is twice that of a real modulator. Since doubling the OSR can improve the SQNR of a $\Delta\Sigma$ modulator dramatically, quadrature bandpass $\Delta\Sigma$ modulators have a significant SQNR advantage over their real counterparts. Lastly, note that since the minimum output data rate is f_B, decimation by a factor of OSR is appropriate for a quadrature system.

Having completed our overview of bandpass and quadrature bandpass $\Delta\Sigma$ ADC systems, we now delve into a discussion of the modulators themselves. The essence of the design steps are the same as with lowpass modulators, namely select the NTF and number of quantizer levels, choose the topology, realize the NTF with the chosen topology, do dynamic-range scaling, and finally convert each block to a transistor circuit. The upcoming sections describe what is different, first for bandpass modulators and then for quadrature modulators. Circuit details and measurement results from a recent high-speed continuous-time bandpass ADC serve as a buffer between the two high-level discussions.

11.3   Bandpass NTFs

Figure 11.5(a) depicts a representative bandpass NTF. Note that in order to have n zeros in the passband there need to be n zeros at negative frequency, and thus a $2n$th-order bandpass modulator is analogous to a lowpass modulator of order n. Lee's rule for stability is as
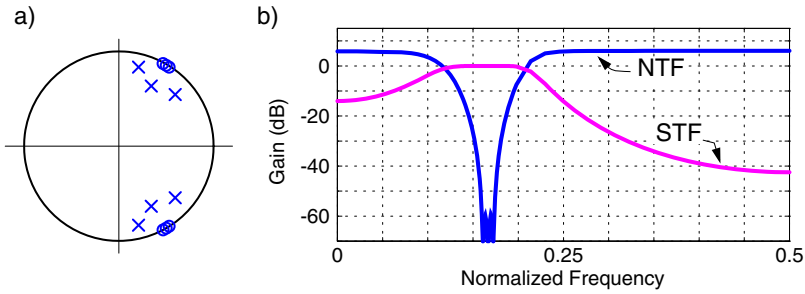
Figure 11.5   (a) Pole-zero and (b) NTF/STF magnitude plots for an $f_s/6$ bandpass modulator.

effective in the context of binary bandpass modulation as it is in the context of binary lowpass modulation. Zero optimization is likewise as useful for bandpass NTFs as it is for lowpass NTFs.

By supplying a nonzero value for the optional f_0 argument to the $\Delta\Sigma$ toolbox function synthesizeNTF, a bandpass NTF can be created that yields a maximally flat all-pole STF when a single feed-in is used for the input signal. The simulateDSM function can then simulate the behaviour of the modulator given the input and number of quantizer levels. The code fragment below illustrates these operations.

```
% Create bandpass NTF
osr = 32;
f0 = 1/6;
ntf = synthesizeNTF(6,osr,1,[],f0);
% Realize it with the CRFB topology
form = 'CRFB';
[a,g,b,c] = realizeNTF(ntf,form);
% Use a single feed-in
b(1) = abs( b(1) + b(2)/c(1)*(1-exp(-2i*pi*f0)) );
b(2:end) = 0;
ABCD = stuffABCD(a,g,b,c,form);
% Simulate the modulator
M = 16;
N = 2^15;
ftest = round((f0+0.25/osr)*N)/N;
u = undbv(-1)*M*sin(2*pi*ftest*(0:N-1));
v = simulateDSM(u,ABCD,M);
```

Figure 11.6 plots a portion of the output data along with the input signal and Figure 11.7 shows the associated spectrum. As with lowpass modulation, the correspondence between the input and the output in the time domain is coarse at best, but in the frequency domain it becomes apparent that the conversion is highly accurate, to 1 part in 10^5 for this example. See Appendix B for more information on how to apply other toolbox functions to bandpass systems.

11.3.1   *N*-Path Transformation

Figure 11.8(a) shows a pair of identical linear time-invariant systems $H(z)$ operating in a time-interleaved fashion. As may be verified by constructing the output in response to
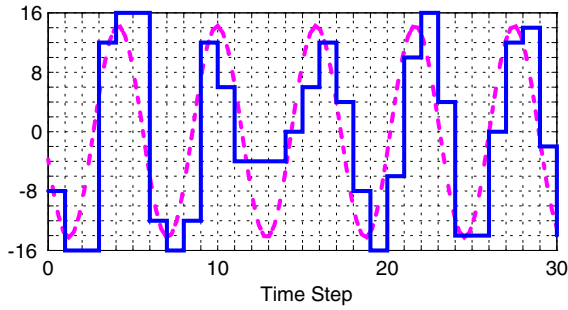
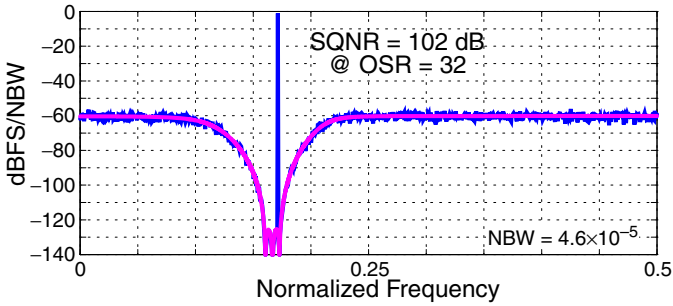**Figure 11.6**   Example input and output data for a bandpass modulator.

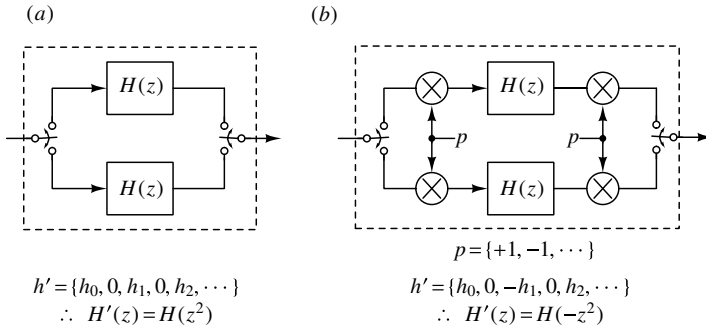**Figure 11.7**   Bandpass modulator output spectrum.

Figure 11.8   Two-path systems.

impulse inputs at arbitrary times, the composite system is time-invariant and has a transfer function $H'(z) = H(z^2)$. Interleaving N copies of the original system realizes a transfer function $H'(z) = H(z^N)$, and hence a transformation of the form $z \rightarrow z^N$ is called an *N-path transformation*.

In a similar vein, Figure 11.8(b) depicts an arrangement wherein the inputs and outputs of the paths have alternating polarity. Despite the time-varying nature of the commutating switches and polarity reversals, the composite system is again time-invariant and has a transfer function $H'(z) = H(-z^2)$. Interleaving N copies of the original system with alternating polarity on each of the path inputs and outputs realizes a transfer function $H'(z) = H(-z^N)$, and a transformation of the form $z \rightarrow -z^N$ is also considered to be an *N*-path transformation.
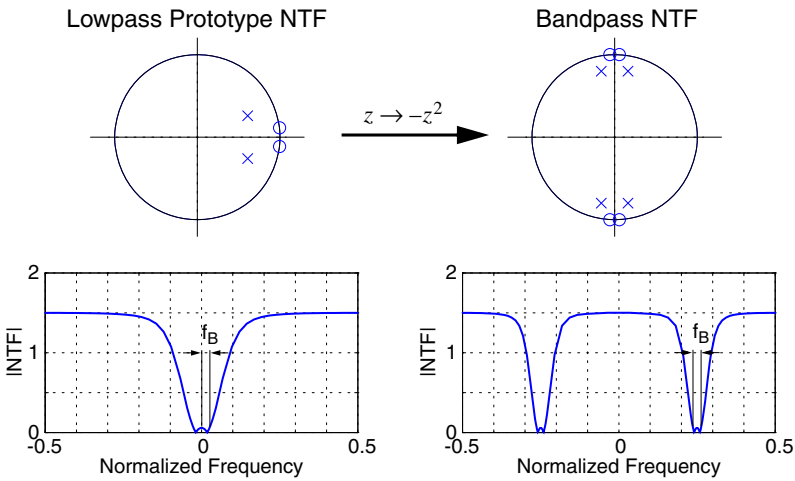

**Figure 11.9**   Applying a two-path transformation ($z \rightarrow -z^2$) to a lowpass NTF.

The substitution $z \rightarrow -z^2$ transforms a lowpass NTF (with n zeros near $z = 1$) into a bandpass NTF with n zeros near $z = j$ and n zeros near $z = -j$. This $(2n)$th-order NTF has the same gain versus frequency profile as the original NTF, except that the frequency axis is compressed by a factor of two and the response replicated as illustrated in Figure 11.9. Since this NTF is obtained via a two-path transformation, the resulting bandpass modulator is exactly equivalent to two copies of the original lowpass modulator operating on subsampled data with alternating polarities, as depicted in Figure 11.10.

This equivalence shows that a $(2n)$th-order bandpass modulator derived from an nth-order lowpass modulator via the $z \rightarrow -z^2$ transformation has exactly the same stability properties and SNR curve as the lowpass modulator operated at the same OSR. Furthermore, the limit-cycle characteristics of a bandpass modulator with an $f_s/4$ sine wave input correspond to the interleaved limit-cycle characteristics of two lowpass modulators with dc inputs of $A \cos \phi$ and $A \sin \phi$, where A is the amplitude of the sine wave and ϕ is its phase relative to the sampling clock.

The $z \rightarrow -z^2$ transformation can be applied to the NTF of a lowpass modulator to yield a bandpass NTF that can then be realized with any of the structures shown in Chapter 4. An alternative method is to apply the $z \rightarrow -z^2$ transformation directly to a lowpass

$f_s/4$ *Bandpass* $\Delta\Sigma$ *Modulator*

$p = \{+1, -1, \cdots\}$
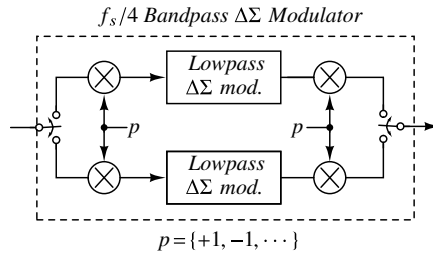
Figure 11.10   System-level $z \to -z^2$ transformation.
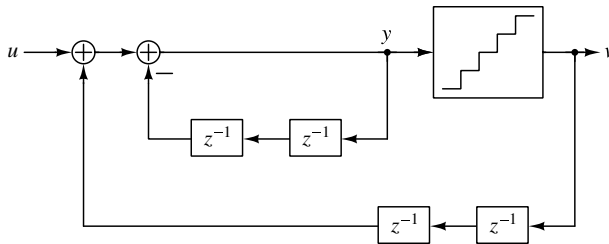
Figure 11.11 $z \to -z^2$ transformation applied directly to MOD1.

modulator at the block-diagram level, simply by replacing each delay element with two delay elements and an inversion. As an illustration, Figure 11.11 shows the $f_s/4$ bandpass analog of MOD1. The "integrator" now contains a pair of delays and an inversion in its feedback path, while the feedback path from the quantizer also contains two delays but loses the inversion that is normally present at the first summation. This structure is actually simpler than what would result from mapping the NTF onto any of the general-purpose structures presented in Chapter 4.

(*a*) $MTF(z) = 1 + z^{-1}$ (*b*) $MTF(z) = 1 + z^{-2}$

time   *time*
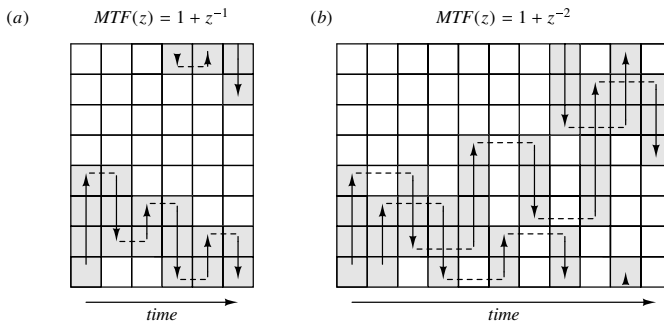
Figure 11.12 N-path mismatch-shaping usage patterns.

As a closing observation regarding the N-path transformation, note that this transformation can be applied to mismatch-shaping logic as well as to a noise-shaping loop. For example, transforming the rotation scheme of Section 6.2 via $z \to -z$ to implement a mismatch transfer function $MTF = 1 + z^{-1}$ yields the element usage pattern depicted in Figure 11.12(a). In this system, the block of 1s associated with each sample starts with

the last element selected and proceeds backwards into the most-recently-selected elements. This element usage pattern can be created by alternately flipping the thermometer-coded output of the flash ADC, from top to bottom, while also alternately adding or subtracting the binary code from the shift control signal. To implement $MTF = 1 + z^{-2}$, two copies of this algorithm need to be interleaved, yielding the element usage pattern shown in Figure 11.12(b).

11.4   Architectures for Bandpass Delta-Sigma Modulators

11.4.1   Topology Choices

Bandpass modulators possess the same architectural variety as lowpass modulators, and the trade-offs between the different structures are also essentially the same. For example, bandpass modulators can be implemented in single-loop or cascade form, with a similar trade-off between improved stability and increased sensitivity to analog nonidealities such as coefficient errors and finite opamp gain. Likewise, the loop-filter of a bandpass modulator can be constructed using any of the conventional forms found in lowpass modulators, including feedback, feedforward, and hybrid topologies, with similar trade-offs between internal dynamic range and STF quality.
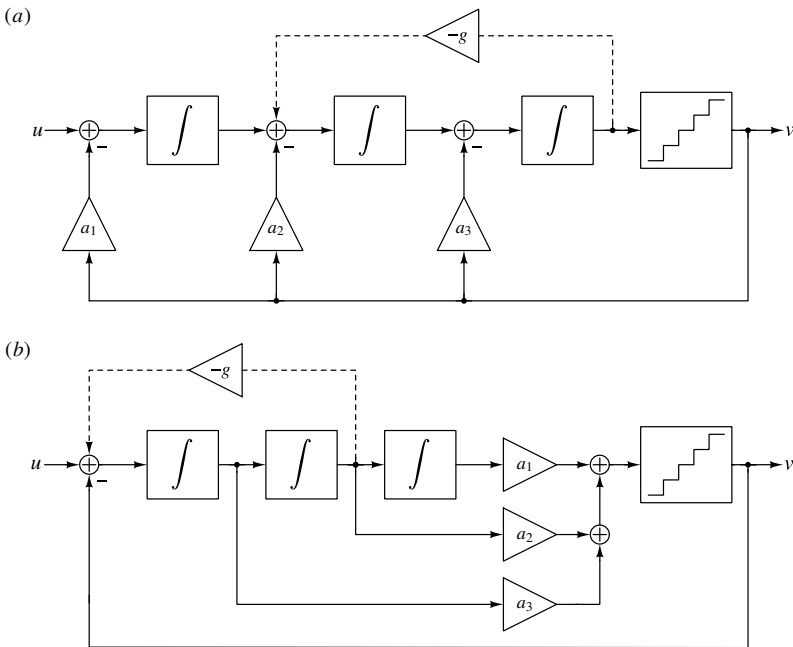
Figure 11.13   Basic loop topologies for lowpass modulators: (a) feedback (b) feedforward.

Figure 11.13 contrasts the feedback and feedforward topology extremes for lowpass modulators. In the feedback topology, the quantizer output signal is fed back to the input of every integrator in the loop filter, whereas in the feedforward topology, the output

signal of every integrator is fed forward to the input of the quantizer. The integrators may be any combination of continuous-time integrators (e.g., Gm-C or active-RC integrators) or discrete-time integrators (including delaying, nondelaying or half-cycle-delaying switched-capacitor integrators), provided the coefficients and timing are chosen appropriately. To shift the loop-filter poles to nonzero frequencies, it suffices to add internal feedback paths such as the one shown with dashed lines in Figure 11.13(a). These loop-filter topologies are readily used in the construction of bandpass modulators. For example, Figure 11.14 shows the structure of the loop-filter of a fourth-order bandpass modulator
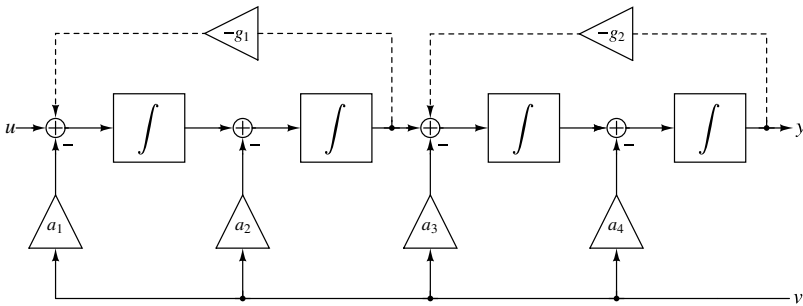
Figure 11.14   Loop filter of a fourth-order bandpass modulator employing the standard feedback topology.

employing the feedback topology.

When f_0 is a substantial fraction of the sampling rate, the resonator output may be taken from the first integrator as shown in Figure 11.15. The integrators in this figure are
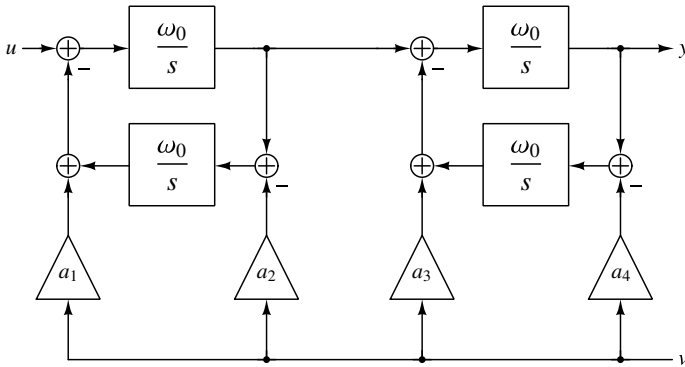
Figure 11.15   Loop filter of a fourth-order bandpass modulator employing a feedback topology with bandpass resonators.

shown as continuous-time blocks for convenience. Taking the resonator output from the first rather than the second integrator output changes the transfer function of the resonator from $\omega_0^2/(s^2+\omega_0^2)$, which is a lowpass response, to $s\omega_0/(s^2+\omega_0^2)$, which is a bandpass one. Since the bandpass response has a null at dc, it is clear that a lowpass modulator cannot make use of these bandpass resonators, whereas a bandpass modulator can. Since the $n/2$

resonators in an nth-order bandpass modulator may either be of the lowpass or bandpass variety, there are $2^{n/2}$ possible lowpass/bandpass resonator combinations for each of the loop-filter topologies (feedback, feedforward, or hybrid).
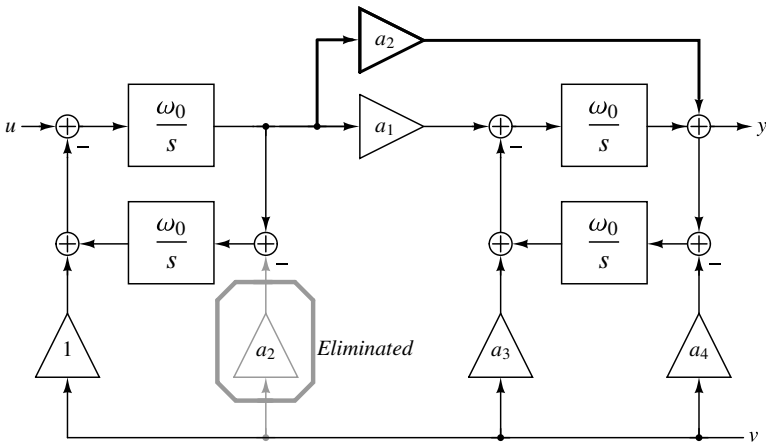
Figure 11.16 Eliminating a feedback DAC by adding a feedforward path.

Figure 11.16 illustrates how adding a feedforward path, and thus connecting the output of one resonator to both of the integrators in the next resonator, can eliminate one of the feedback coefficients (i.e., one of the feedback DACs) in a bandpass modulator. Since the transfer function from V to Y is the same in Figure 11.16 as that of Figure 11.15, the noise transfer function of a modulator employing the loop-filter of Figure 11.16 will be the same as that of a modulator employing the loop filter of Figure 11.15. (The signal transfer functions will not be the same, however.) This transformation may be applied to each resonator section except the last one, thereby cutting the required number of DACs by nearly 50%. As will be seen in Section 11.5, this transformation is helpful in the construction of a bandpass modulator that employs one or more LC tanks.
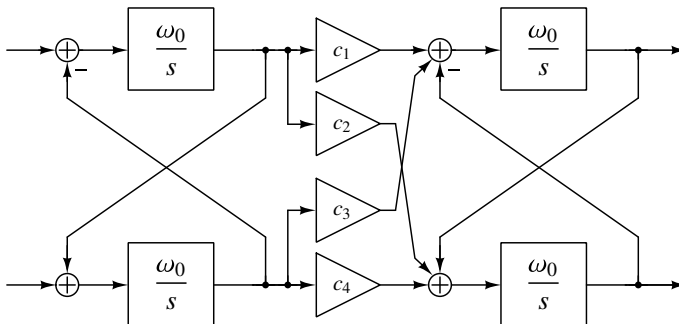
Figure 11.17 Internal structure of a more general loop-filter for a bandpass modulator.

Figure 11.17 shows a portion of a loop-filter which encompasses all of the variants. Each resonator section is coupled to the next through four arbitrary gain blocks, so the choice of a lowpass versus a bandpass section is simply a special case wherein all coef-

ficients are zero except for one. The feedback DACs are not shown, and these could be added to any or all of the integrator summing junctions, according to whether a feedback, feedforward, or hybrid modulator topology is used.

11.4.2 Resonator Implementations

The primary difference between the realizations of a lowpass modulator and a bandpass modulator is that a lowpass modulator requires good integrators, whereas a bandpass modulator needs good resonators. The degradation caused by a finite quality factor (Q) in the resonators of a bandpass modulator is analogous to the degradation caused by finite dc gain in the integrators of a lowpass modulator: both reduce SQNR and increase susceptibility to tonal behavior. The SQNR degradation is significant when Q falls below f_0/f_B. Thus, in order to take full advantage of a high value of *OSR*, the Q of each resonator should be high. Conversely, when the signal is not especially narrowband, that is, when f_0/f_B is not very high, the Q requirements for nearly ideal operation are relaxed. The resonant frequency of the resonator must be accurate for similar reasons. An f_0 error that is an appreciable fraction of f_B, say, 20%, is usually close to the level of significance. This section presents several resonator circuits that have been used in the construction of bandpass $\Delta\Sigma$ ADCs, and comments on the ability of each to achieve an accurate and high-Q resonance.

(*a*)

(*b*)
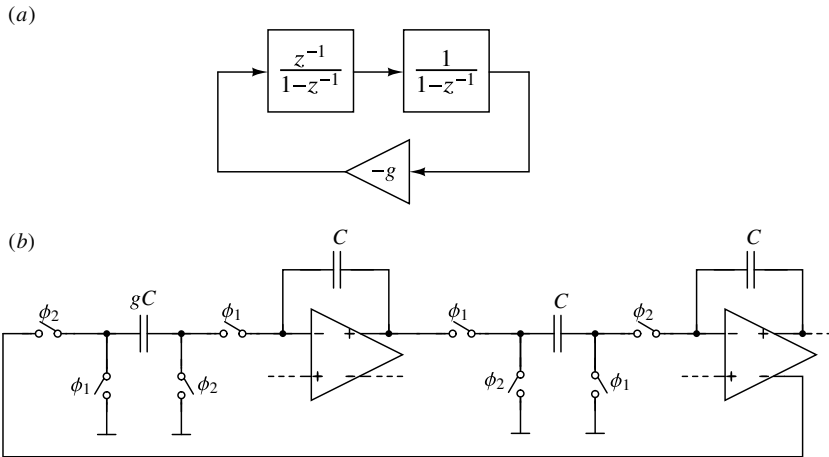
Figure 11.18   (a) The LDI loop. (b) A switched-capacitor implementation.

Figure 11.18(a) depicts the *lossless discrete integrator* (LDI) loop, which may be realized in switched-capacitor form as shown in Figure 11.18(b). The structure of this circuit is such that the poles are the roots of the characteristic equation

$$1 + \frac{gz}{(z-1)^2} = 0. \tag{11.3}$$

The roots of (11.3) are $z_p = \sigma \pm j\sqrt{1 - \sigma^2}$, where $\sigma = 1 - g/2$. Clearly, for $|\sigma| \leq 1$ (i.e., $0 \leq g \leq 4$), the poles of the LDI loop lie on the unit circle and thus the Q of the resonator is ideally infinite. Finite opamp gain limits Q, but since $Q > 100$ is readily achieved with typical opamp gain, finite resonator Q is usually not problematic.

The frequency of the resonance is given by $\omega_0 = \cos^{-1}(\sigma) = \cos^{-1}(1 - g/2)$, which shows explicitly the dependence of ω_0 on capacitor ratios. The sensitivity of ω_0 to capacitor ratio errors is an increasing function of ω_0, but even at the relatively high value of $\omega_0 = \pi/2$, a 1% shift in capacitor ratios translates to only a 0.6% shift in ω_0. Since capacitor matching is typically much better than 1%, the ω_0-accuracy of an LDI-based resonator is usually sufficient.

The LDI loop provides a good way to implement a switched-capacitor resonator possessing an arbitrary resonant frequency but does so using two opamps. When the resonant frequency is $f_s/4$, the 2-path transformation described in Section 11.3.1 leads to circuits, such as that depicted in Figure 11.19, that are able to implement a resonator with a single
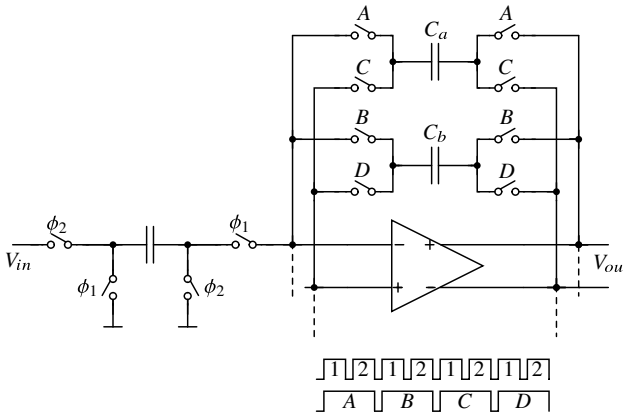

Figure 11.19   A 2-path switched-capacitor $f_s/4$ resonator.

opamp. Note that since such circuits implement the desired center frequency by virtue of their topology rather than through the use a particular set of capacitor ratios, capacitor errors do not translate into center-frequency errors.

Although the center frequency of the circuit shown in Figure 11.19 is insensitive to capacitor ratios, mismatch in the paths (in particular, the C_a and C_b capacitors) causes the circuit to be *periodically time-varying*, instead of time-invariant. The time-varying nature of the circuit in turn causes mixing of the signal with $f_s/4$ and its harmonics, and it is the mixing of the signal with $f_s/2$ that results in the appearance of an *image signal*, a frequency-inverted copy of the signal centered on f_0. Another source of difficulty in this circuit stems from the use of clocks whose frequency is $f_s/4$. These large-amplitude clocks can leak into the signal band and produce a tone at the band-center.

The circuit depicted in Figure 11.20 avoids these problem to a large degree. Since the C_a and C_b path capacitors in this circuit are only used for charge storage, and since the conversion from charge to voltage is performed by the (path-independent) C_o capacitors, the time-varying nature of the circuit is essentially hidden. (In practice, the opamp gain must be high enough to ensure adequate charge-transfer efficiency.) Also, since this circuit does not use $f_s/4$ clocks, this spur-generating mechanism is not an issue.

Figure 11.21 shows the structure of a g_m-C resonator. Since the center frequency is given by $\omega_0 = g_m/C$, and since the value of g_m/C implemented with on-chip capacitors
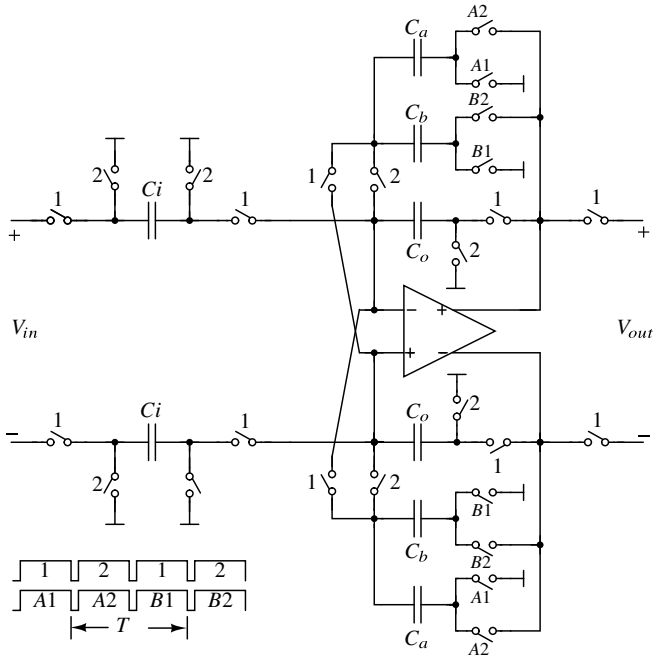
Figure 11.20   A 2-path resonator with reduced sensitivity to capacitor mismatch [6].
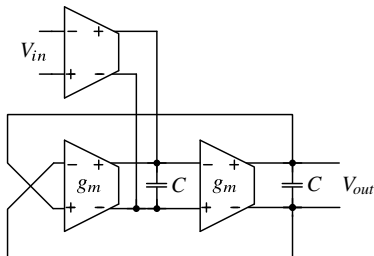
Figure 11.21   A g_m-C resonator.

and transconductors typically has 30% variability, the center frequency of a g_m-C resonator will be poorly controlled unless some means for tuning is provided. A common method for tuning a g_m-C filter is to adjust all the g_m elements of the filter along with those of a *reference filter* until the reference filter has the desired response [7]-[9]. However, since a resonator can be converted into an oscillator with only a small amount of positive feedback, it often suffices to measure the oscillation frequency of the resonator itself and adjust g_m (or C) directly. Since this calibration must be done off line, the designer must ensure that the drift of g_m over temperature is sufficiently small. If the drift cannot be made sufficiently small, a continuous-tuning method involving a (scaled) copy of the resonator is the next best choice.

Once the problem of resonator tuning has been addressed, the next set of concerns revolve around the resonator's Q. Nonidealities such as finite output impedance and non-zero phase shift in the transconductors limit resonator Q. Techniques such as cascoding can boost output impedance, while the phase shift can be reduced by using a wide-band g_m, or compensated by adding a small resistor in series with the capacitors.
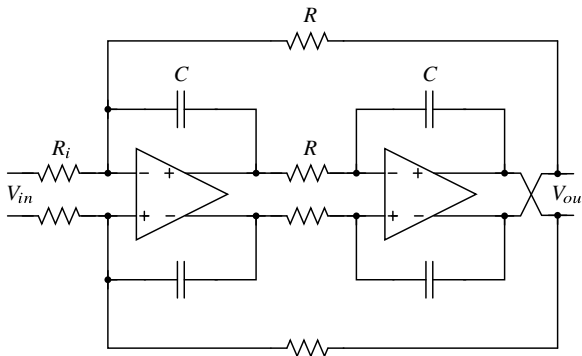
Figure 11.22   An active-RC resonator.

Figure 11.22 shows the structure of an active-RC resonator. Here the center frequency is given by $\omega_0 = 1/(RC)$, and once again the highly variable nature of the RC product necessitates the use of tuning. Tuning may be accomplished by adjusting R (continuously via MOS devices, or in discrete steps using a resistor array), by adjusting C (here an array is most practical), or by a combination of the two approaches. Once again, configuring the resonator as an oscillator is straightforward and eliminates the need for a replica block, but can only be done when the converter is off line.

Figure 11.23   A resonator based on an LC tank.

The last resonator we consider is the LC tank driven by a current source, shown in Figure 11.23. This resonator possesses three attributes that make its use in a bandpass converter highly advantageous. First, note that inductors and capacitors are noiseless. A resonator based on an LC tank therefore enjoys a significant noise advantage over the preceding resonator circuits and this noise advantage is achieved with no power consumption. Also an LC tank provides low distortion since inductors and capacitors are typically highly lin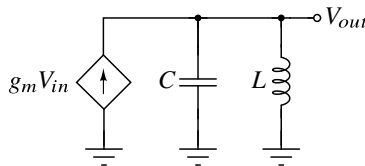ear. Finally, inductors do not consume voltage headroom, and thus using inductors maximizes the available signal swing. It is very rare in circuit design to obtain noise, distortion, and power advantages simultaneously, but such is the case with a bandpass $\Delta\Sigma$ ADC made with LC tanks.

There are disadvantages to using LC tanks within a bandpass modulator, but the aforementioned advantages provide a powerful motivation to overcome the disadvantages. The first disadvantage is that inductors resist integration: on-chip inductors possess only a few nanohenries of inductance and thus are only useful at frequencies above 1 GHz or so. For lower frequencies where external inductors are required, a bandpass ADC would typically use an LC tank only as its first resonator, where the noise and distortion advantages are most compelling. The tolerance of external inductors is also a potential problem, but this problem is readily overcome by including an on-chip capacitor array to tune the tank's resonant frequency.

The second disadvantage relates to programmability. Since it is impractical to tune inductance electronically, an LC-based bandpass modulator typically supports not much more than an octave of tuning range. Multiple tanks are needed to support wider ranges.

The final disadvantage associated with using LC tanks in a bandpass modulator results from the fact that an LC tank is a second-order system and so requires two degrees of freedom for control purposes. In a feedback topology, we would like to use a current DAC in parallel with the capacitor and a voltage DAC in series with the inductor. Although current DACs with high output resistance are quite practical, voltage DACs with low output resistance are not, since even one ohm of resistance can reduce tank Q significantly. Similarly, if we use a feedforward topology instead of a feedback one, we would need to sense the capacitor voltage and the inductor current in order to feed these states into subsequent stages. Again, sensing the capacitor voltage is easy, but sensing the inductor current without degrading tank Q is difficult. We now summarize three methods that have been proposed to overcome this controllability/observability problem.
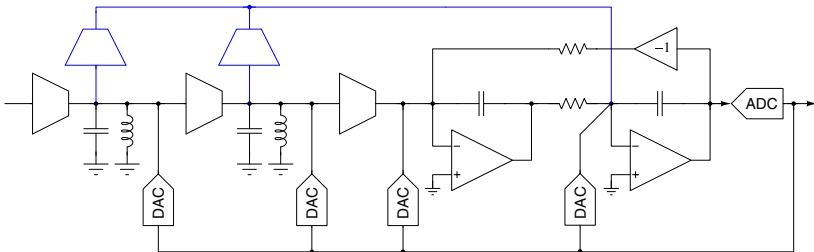
Figure 11.24 An LC-based bandpass ADC with feedforward paths into an active-RC back-end [10].

The most elegant was already introduced in the block diagram of Figure 11.16. Figure 11.24 provides a circuit-level representation of the concept in the context of a 3-

resonator feedback system. By adding a feedforward element from each front-end tank to the second integrator in the back-end active-RC resonator, two degrees of freedom are created that compensate for the two degrees of freedom lost by not having voltage-mode DACs.
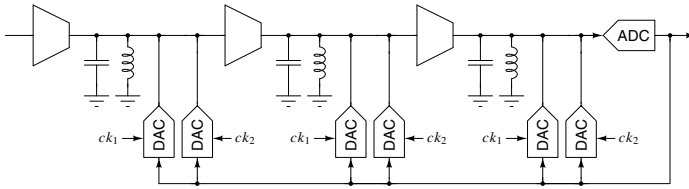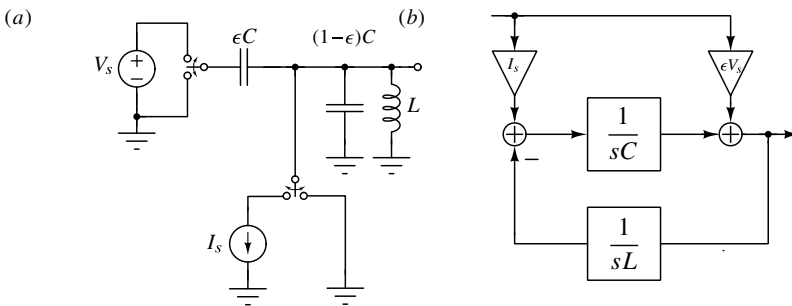

Figure 11.25   Using multiple feedback DACs provides full control over the NTF in an LC-based bandpass ADC [11].

A second method (Figure 11.25) uses pairs of DACs with different timing to restore the missing degrees of freedom. In the notation of Appendix B, timing such as $[0, 0.5]$ plus $[0.5, 1]$ ("return-to-zero" and "delayed return-to-zero") has been suggested [11].

Figure 11.26   Voltage feedback to an LC tank; (a) circuit (b) model [2].

A more recent approach is illustrated in Figure 11.26. In this arrangement only a small fraction of the tank capacitance is connected to a voltage DAC and thus nonzero output resistance in the voltage DAC is less critical.

11.5 Bandpass Modulator Example

Figure 11.27 shows the block diagram of a 65-nm CMOS IC which is capable of digitizing signals from DC to 1 GHz with bandwidths up to 100 MHz [1]. The incoming signal can be routed through either a low-noise amplifier (LNA) or a programmable attenuator. The LNA has 12 dB of gain range, while the attenuator provides a further 27 dB of gain control. Both blocks provide a 50-Ω termination when enabled. The output of the LNA/attenuator is digitized by a highly programmable continuous-time lowpass/bandpass ADC whose digital output is down-converted and filtered by an on-chip digital block. The IC also includes a synthesizer for generating the 2–4 GHz ADC clock.

The architecture of the ADC in bandpass mode is shown in Figure 11.28. The ADC uses a sixth-order continuous-time feedback topology with 16-step quantization and $[1, 2]$
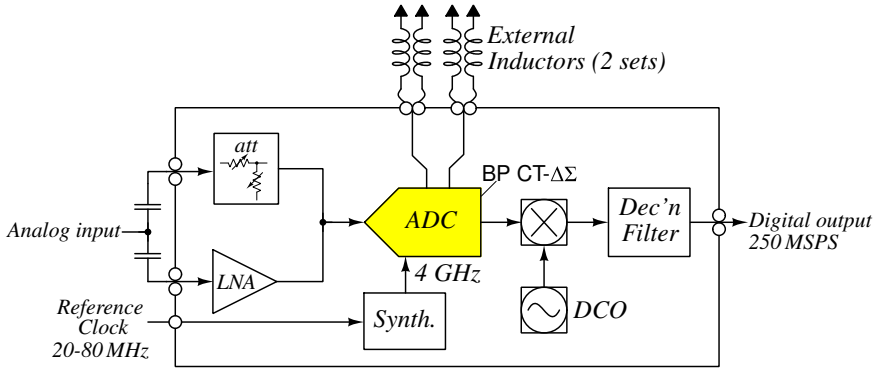
Figure 11.27   An IC containing a highly programmable ΔΣ ADC [1].
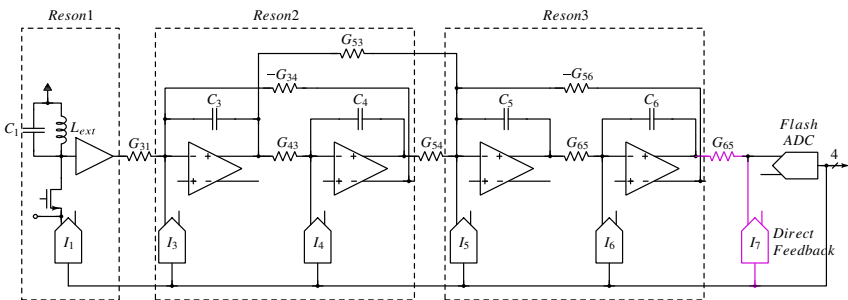
Figure 11.28   Simplified ADC architecture in bandpass mode.

feedback timing. The first resonator is an LC tank while the remaining two resonators are active-RC. The G_{53} component supplies the missing degree of freedom to support arbitrary sixth-order NTFs with five feedback DACs. The I_7 feedback DAC and associated resistor (highlighted) implement the direct feedback term needed to compensate for the chosen DAC timing.

In order to support a wide variety of clock rates, center frequencies and signal bandwidths, the ADC has a high degree of programmability. Programmable parameters include the LSB currents of the DACs, the LSB size of the flash ADC, all integrating capacitors, and every conductance. Each of these parameters is controlled with 8-bit resolution. Even the inductors can be chosen from one of two pairs via the associated cascode devices. In addition to the bandpass topology shown in Figure 11.28, the ADC can be configured as a 6th-order lowpass modulator by replacing the LC tank with an active-RC resonator. The lowpass mode is used for center frequencies from dc to 200 MHz while the bandpass mode uses one set of inductors for 200–500 MHz and the other set for 500–1000 MHz.
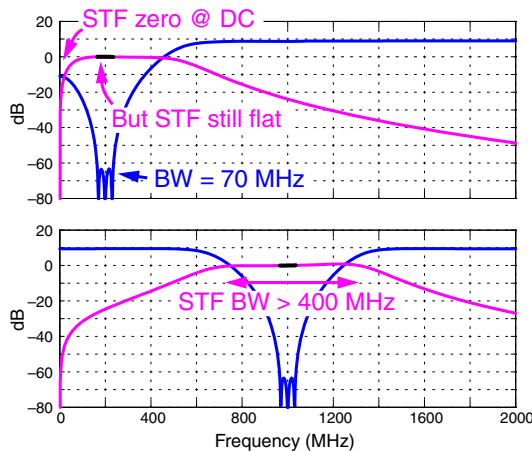
Figure 11.29   Example NTF/STF for $f_0 = 200$ MHz and 1 GHz.

Figure 11.29 illustrates the theoretical NTF and STF for center frequencies of 200 MHz and 1 GHz. In each case, the out-of-band gain was chosen to provide 65 dB of quantization noise attenuation for a bandwidth of 70 MHz. The resulting ~10 dB of out-of-band NTF gain is reasonable with 16-step quantization. The STF has a zero at dc courtesy of the inductor in the LC tank. At low center frequencies this zero makes it difficult to position the NTF/STF poles such that the STF is flat. The top plot shows that an acceptable STF can be achieved for f_0 as low as $f_s/20 = 200$ MHz. The lower plot shows that when $f_0 = f_s/4 = 1$ GHz, the STF is extremely flat and wideband. Such a wide and flat STF response is desirable because it yields a low group-delay variation.

11.5.1   LNA

Figure 11.30 shows a simplified schematic of the LNA. The LNA uses common-gate (CG) transistors to provide a 50-Ω match and programmable-width common-source (CS) tran-
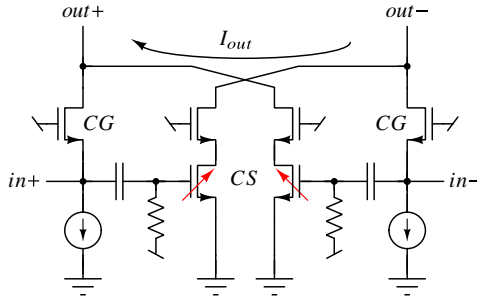
Figure 11.30   Variable-gain noise-canceling LNA.

sistors to provide 12 dB of gain range. This LNA topology takes some advantage of the noise-canceling principle [12][13].
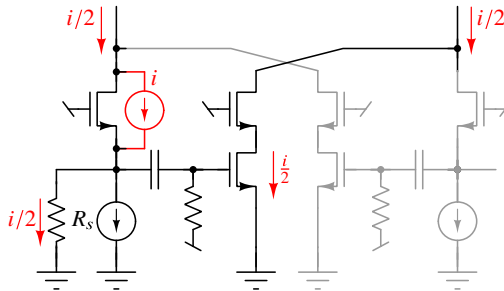
Figure 11.31   Analysis of the noise from a common-gate transistor at the 6-dB gain setting.

   To understand noise cancellation in this circuit, consider the operation of the circuit at the 6-dB gain setting, where the transconductances of the CG and CS transistors are both $1/R_s$. As illustrated in Figure 11.31, since the source of the CG transistor presents an impedance of $1/g_m$, and since $R_s = 1/g_m$, half of the noise current i injected into the source of the CG transistor flows through R_s to ground and then back into the *out+* terminal through the load, while the other half of the noise current just recirculates through the CG transistor. At the 6-dB gain setting the transconductance of the CS transistor is $1/R_s$, and since the gate voltage of the CS transistor is $R_s i/2$, the CS transistor also carries a current $i/2$, which is drawn from the *out−* terminal. Thus, at the 6-dB gain setting the noise of the CG transistor appears as a common-mode signal, which is rejected by subsequent stages. Of course, the noise of the CS transistor is not cancelled, and, unlike the standard noise-canceling circuits, the cancellation of the noise of the CG transistor only occurs at the 6-dB gain setting.

11.5.2   Attenuator

The attenuator (Figure 11.32) takes advantage of the virtual ground provided by the cascode transistor to yield a wideband programmable attenuator with noise properties superior to those of an attenuator with impedance matching at both input and output. To see why, consider the simplified schematic shown in Figure 11.33 where $0 \leq x \leq 1$ represents

Figure 11.32   Programmable attenuator.

$$I_{out} = \frac{xGV_s}{2}$$

$$NF = 10 \log_{10} \left(\frac{4-2x}{x} \right)$$
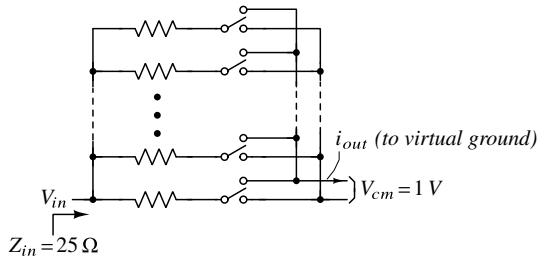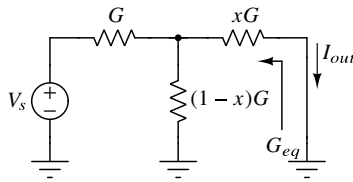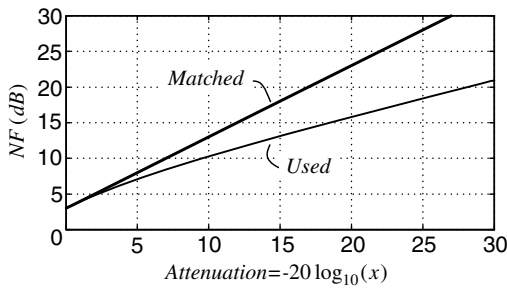

Figure 11.33   Attenuator noise figure.

the fraction of the programmable conductance connected to the output node. Since the transfer function from V_s to I_{out} is $xG/2$, the density of the I_{out} noise due to the source conductance G is

$$n_s = \left(\frac{4kT}{G}\right)\left(\frac{xG}{2}\right)^2 = kTGx^2. \tag{11.4}$$

The conductance seen looking into the output is the series combination of xG and $(2-x)G$:

$$G_{eq} = \frac{xG(2-x)G}{2G} = x(1 - x/2)G. \tag{11.5}$$

Since the network is passive, the total I_{out} noise density is $4kTG_{eq}$ and the noise factor is therefore

$$F = \frac{4kTG_{eq}}{n_s} = \frac{4 - 2x}{x}. \tag{11.6}$$

In contrast, the noise factor of a matched attenuator terminated with a matching resistor is

$$F_{\text{matched}} = \frac{2}{x^2}. \tag{11.7}$$

The plot in Figure 11.33 compares these two noise expressions. At 0 dB of attenuation the noise figure of both arrangements is 3 dB. However, the noise figure of the matched arrangement increases dB for dB of attenuation whereas the noise figure of the integrated attenuator increases less quickly. At an attenuation of 12 dB the matched system has $NF =$ 15 dB whereas the noise figure of the integrated attenuator is 11.5 dB. As the figure shows, the difference becomes more pronounced at higher attenuation settings. The integrated attenuator is an example of a common phenomenon, namely that building more of the signal chain into the ADC can yield a better trade-off between fundamental parameters such as noise figure and attenuation than traditional arrangements.

At the 0-dB gain setting of the LNA or the 0-dB attenuation setting of the attenuator the (trans-)conductance from the input to the virtual ground is $1/50\,\Omega$. Since the first feedback DAC has a full-scale of 4 mA at its maximum setting, the full-scale of the ADC under these conditions is therefore $4\text{ mA} \cdot 50\Omega = 200\text{ mV}_\text{p}$ or -4 dBm, which is 1/5 that of typical commercial ADCs. Decreasing the full-scale of the first feedback DAC or engaging the LNA can be used to reduce the effective full-scale of the ADC even further.

11.5.3 Amplifiers

The amplifier requirements depend on the location of the amplifier in the modulator loop. For example, the amplifier used within the first active-RC resonator associated with low-pass mode needs to have high gain and large swing since it is used in the most critical stage. The gain and swing requirements in subsequent stages are less stringent, but since those stages must process signals up to 1 GHz whereas the active-RC version of the first resonator only processes signals up to 250 MHz, the back-end amplifiers need higher bandwidth than the front-end amplifiers. Two amplifier variants were designed to cover these disparate requirements. The A1 amplifier of the first resonator in lowpass mode needs at least 60 dB of gain from dc to 250 MHz whereas the A3 amplifier used within the second and third resonators requires 40 dB of gain from dc to 1 GHz. The A1 amplifier uses the +2.5-V IO supply to provide large swing whereas the A3 amplifier operates from the 1-V core supply. Both amplifiers make heavy use of feedforward compensation.
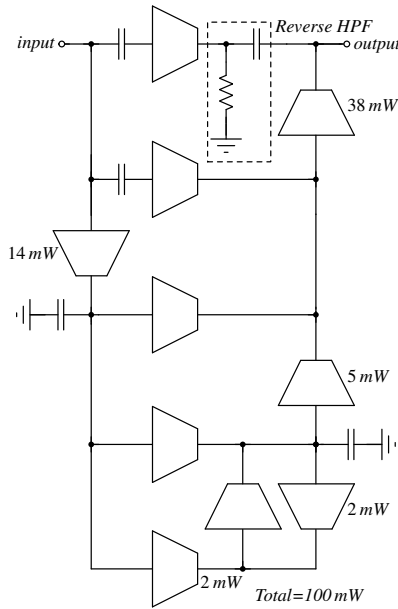
Figure 11.34   Structure of the A1 fifth-order feedforward amplifier.

Figure 11.34 shows the structure of the fifth-order A1 amplifier. In keeping with the principle of feedforward amplifier design, there are 1st-, 2nd-, 3rd-, 4th-, and 5th-order paths connecting the input to the output. The unity-gain frequencies of the paths are designed to provide a smooth transition from 5th-order roll-off with large phase lag to near 1st-order roll-off with a phase lag substantially less than 180°. Since the low-frequency gain of the amplifier is determined by the gain of the longest path, the low-frequency noise of the amplifier is dominated by the noise of the first g_m on that path while the burden of driving the load is borne by the last amplifier on that path. As indicated in Figure 11.34 these two blocks are responsible for roughly half of the amplifier's 100-mW power consumption. Thus the amplifier is reasonably power-efficient despite its complex structure.
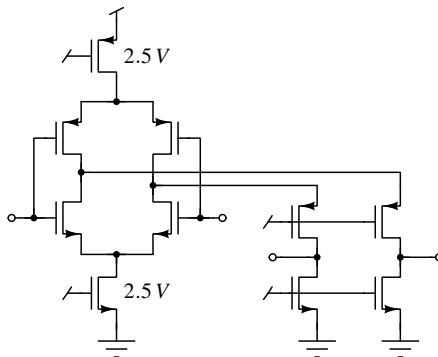
Figure 11.35   A1's input g_m.

As an example of the transistor-level implementation of a g_m stage used within A1, Figure 11.35 shows the topology used in the input g_m stage. This stage consists of a complementary differential pair operating on the 2.5-V supply connected to a folding stage. The complementary pair maximizes the g_m/I_{bias} ratio while the folding stage interfaces to the 1-V stages used in the lower portion of the amplifier.
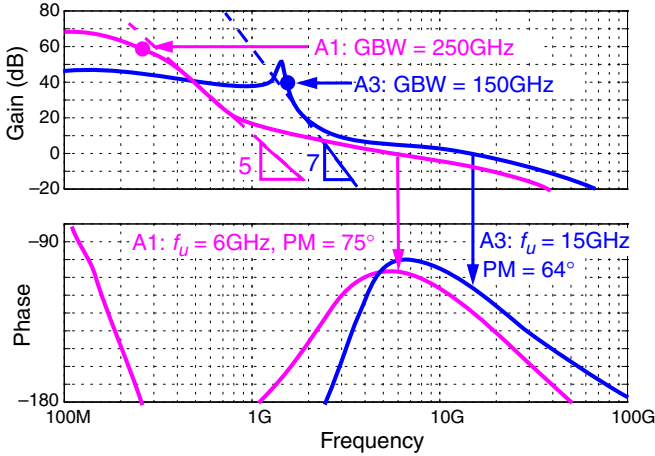
Figure 11.36   Amplifier frequency response.

Figure 11.36 compares the frequency responses of the 5th-order A1 and 7th-order A3 amplifiers. As this figure shows, A1 maintains 60 dB of gain out to 250 MHz, corresponding to a gain-bandwidth product of 250 GHz. Thanks to the high-order roll-off, the actual unity-gain frequency is a more practical f_u = 6 GHz and the phase margin is 75°. In contrast, the higher frequency A3 amplifier provides ~40 dB of gain out to 1.5 GHz (i.e., an equivalent GBW of ~150 GHz) and achieves f_u = 15 GHz with a phase margin of 64°. These simulation results demonstrate the utility of the feedforward technique in achieving high gain over a wide bandwidth without requiring an impractically high unity-gain frequency.

11.5.4   Measurements

Figure 11.37 shows measured STFs and noise spectral density (NSD) for several f_0 settings at a clock frequency of 3 GHz and a bandwidth of 75 MHz. First, note that the STFs are quite flat and broadband. The measurements indicate an STF variation of less than 0.5 dB over 100 MHz. The NSDs demonstrate the flexible nature of the ADC. With L = 43 nH, the ADC's center frequency can be tuned from 150 MHz to 220 MHz, and using L = 20 nH allows the center frequency to vary from 220 MHz to 380 MHz. For a given inductor size, the in-band noise (IBN) tends to decrease as the center frequency increases because the LC tank provides higher gain and hence more attenuation of back-end noise as the center frequency goes up. The voltage swing on the LC tank increases as well, so there is an upper limit on the center frequency that can be supported by a given inductance. The U-shaped in-band NSD is due to the fact that the gain of the LC tank is less at the edge of the passband and thus back-end noise contributes more to the total noise at the passband edges. The
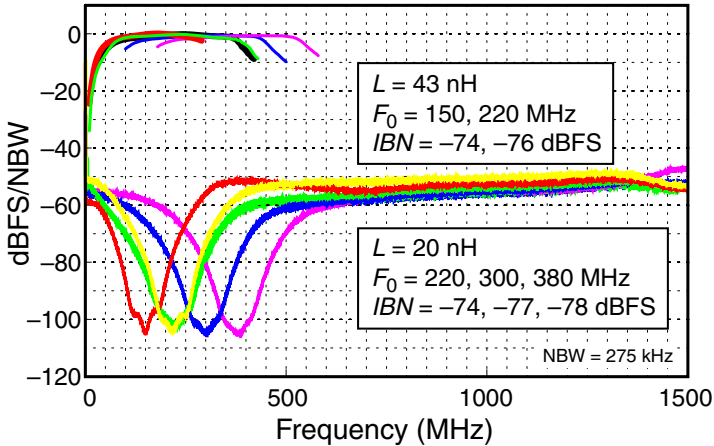
Figure 11.37   Measured STFs and noise spectral density for several f_0 settings.

depth of the U depends on the ADC's configuration, but at $BW = 75$ MHz the observed NSD variation across the passband is about 5 dB. Since in Figure 11.37 $NBW = 275$ kHz, the conversion from the units of the vertical axis to dBFS/Hz is $10 \log_{10}(NBW) = 53$ dB and thus the NSD minima in Figure 11.37 are at $-105 - 53 = -158$ dBFS/Hz. With optimized settings (attenuation $= 12$ dB, $f_0 = 350$ MHz, $BW = 50$ MHz, $L = 20$ nH, etc.), NSDs as low as -161 dBFS/Hz have been attained.
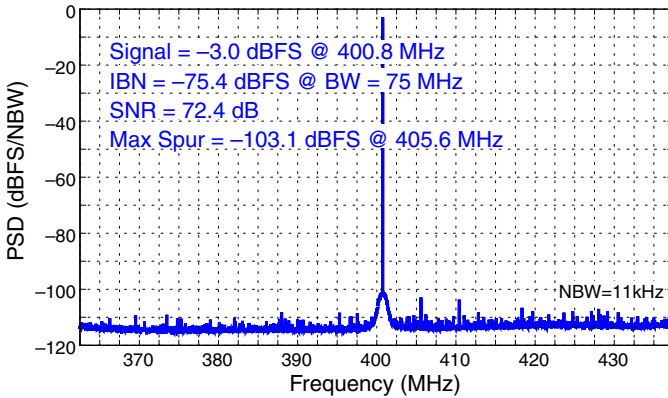
Figure 11.38   Single-tone spectrum at $f_0 = 400$ MHz.

Figure 11.38 shows the in-band spectrum observed after downconversion and decimation with a -3-dBFS input at approximately 400 MHz. Observe the clean spectrum (the largest spur is at -100 dBc), and note that even though the signal is at 400 MHz, an SNR of 72 dB is achieved with a bandwidth of 75 MHz.

To properly demonstrate linearity in a bandpass system, a two-tone test is required. Figure 11.39 shows a two-tone result in which the IMD3 terms are below -87 dBc. With $FS = -4$ dBm, this level of distortion with -8-dBFS tones yields an input-referred third-order intercept of $IIP3 = -12 + 87/2 = +31$ dBm.
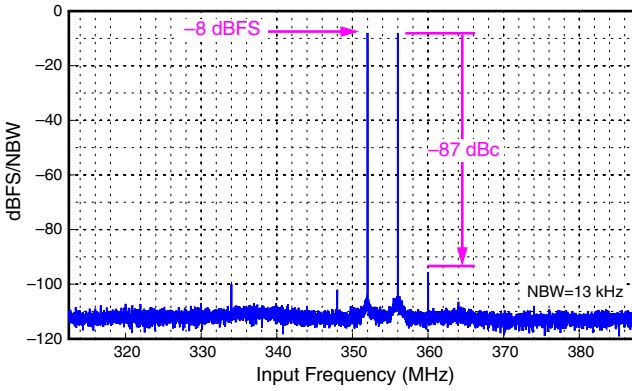
Figure 11.39   Two-tone spectrum at f_0 = 350 MHz.
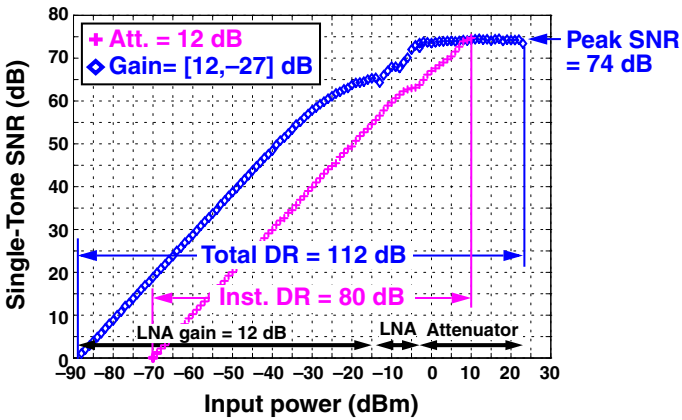
Figure 11.40   SNR versus input power (BW = 75 MHz, f_0 = 400 MHz, f_s = 4 GHz).

As a final measure of the ADC's performance, Figure 11.40 shows the single-tone SNR as a function of input power with and without gain control. With a fixed attenuation of 12 dB, the ADC demonstrates an instantaneous dynamic range of 80 dB, and a peak SNR of 74 dB with a 75-MHz bandwidth. Engaging the LNA extends the lower input limit by 18 dB and allowing increased attenuation adds another 14 dB on the high end (blue) for an overall dynamic range of 112 dB.
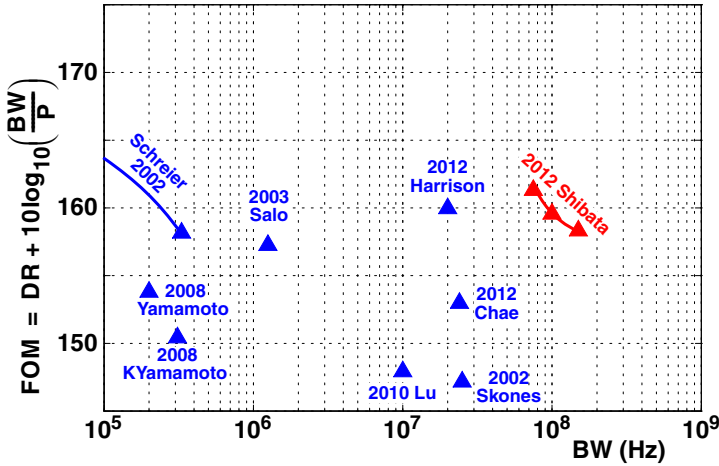
Figure 11.41   Figure-of-merit plot for bandpass ADCs.

Table 11.1 lists the characteristics of the ADC and Figure 11.41 puts this ADC into context by comparing its figure-of-merit (FoM) with other bandpass converters. As the figure shows, the ADC achieves a respectable FoM (159 dB) with state-of-the-art bandwidth.

Table 11.1   ADC summary.

Parameter	Value	Notes
Z_{in}	50 Ω	
f_0	200-400 MHz	
f_s	2-4 GHz	
Full-scale	−16 to +23 dBm	
BW	up to 100 MHz	<3-dB NSD degradation
NSD	< −157 dBFS/Hz	BW = 75 MHz; 12-dB att.
Current	110, 620, 20 mA	2.5, 1.0, −2.5 V supplies
Power	1 W	Includes digital filter
Technology	65 nm CMOS	

11.6 Quadrature Signals

This section reviews quadrature signal processing in preparation for the upcoming section on quadrature $\Delta\Sigma$ modulation. As will be explained in greater detail shortly, quadrature signals are produced by quadrature mixers, which are themselves useful because of their *image-rejection* properties.

A quadrature signal v is an abstract signal composed of two real signals, v_x and v_y, viewed as a single complex entity $v = v_x + jv_y$.[2] Since a quadrature signal has a non-zero imaginary part, its Fourier transform need not be symmetric about zero frequency. In other words, with quadrature signals positive frequencies and negative frequencies contain independent information.

11.6.1 Quadrature Mixing
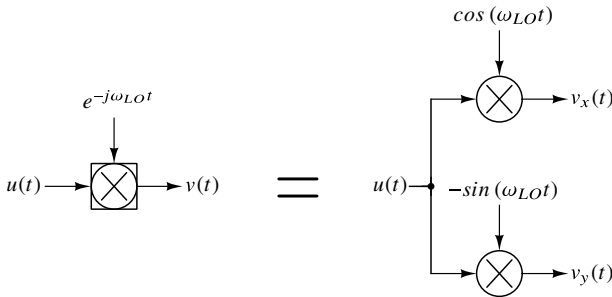
Figure 11.42   Quadrature mixing.

Quadrature mixing is the most common way to make quadrature analog signals. In a quadrature downconversion mixer, a real (or quadrature) signal is multiplied by the quadrature signal $e^{-j\omega_{LO}t}$, which we will refer to as the LO (for local oscillator). The LO consists of two real signals, $\cos\omega_{LO}t$ and $-\sin\omega_{LO}t$, as illustrated in Figure 11.42. Suppose that the input to such a mixer is the real signal $u(t) = A\cos((\omega_{LO} + \omega_{IF})t)$. Then the output of the mixer is

$$
\begin{aligned}
v(t) &= A\cos(\omega_{LO} + \omega_{IF})t \times e^{-j\omega_{LO}t} \\
&= A\left[\frac{e^{j(\omega_{LO}+\omega_{IF})t} + e^{-j(\omega_{LO}+\omega_{IF})t}}{2}\right]e^{-j\omega_{LO}t} \\
&= \frac{A}{2}e^{j\omega_{IF}t} + \frac{A}{2}e^{-j(2\omega_{LO}+\omega_{IF})t}.
\end{aligned}
\tag{11.8}
$$

[2] Since this text has a strong circuits emphasis, denoting the components of a quadrature signal by I and Q may lead to confusion, given that I is used to represent current, while Q is used to represent quality factor or charge. Similarly, denoting the components re and im leads to confusion when dealing with transforms. (The imaginary part of the Laplace transform of a quadrature signal is *not* the Laplace transform of the imaginary component of the quadrature signal.) Instead of using either of the two prevailing conventions, we adopt the notation (inspired by the Cartesian representation of a complex number) that a quadrature signal v is decomposed as $v = v_x + jv_y$, and refer to the components of v as the x and y components, respectively. The Laplace transform of v is then $V = V_x + jV_y$, where V_x and V_y are the Laplace transforms of the x and y components of v.

Removing the second term in the expression above with a lowpass filter leaves a frequency-shifted version of the original signal, centered at the angular frequency ω_{IF}.

A quadrature downconversion mixer is useful because it performs a frequency translation operation that distinguishes between signal frequencies above the LO and signal frequencies below the LO, whereas a conventional mixer does not. In practice, the ability of a quadrature mixer to distinguish between frequencies offset from the LO by equal positive and negative amounts is limited by mismatch between the two real mixers and imperfect quadrature in the two components of the LO. The *image-rejection ratio* (IRR) specifies the signal power appearing at ω_{IF} relative to the signal power appearing at $-\omega_{IF}$ as a result of an input at $\omega_{LO} + \omega_{IF}$. For small errors, *IRR* is approximately [14]

$$IRR = 6 - 10 \log_{10} \left[\left(\frac{\Delta A}{A} \right)^2 + (\Delta\phi)^2 \right], \tag{11.9}$$

where $\Delta A/A$ is the relative amplitude imbalance and $\Delta\phi$ is the phase error (in radians). Figure 11.43 indicates that an amplitude imbalance of 2% (0.17 dB), or a phase error of
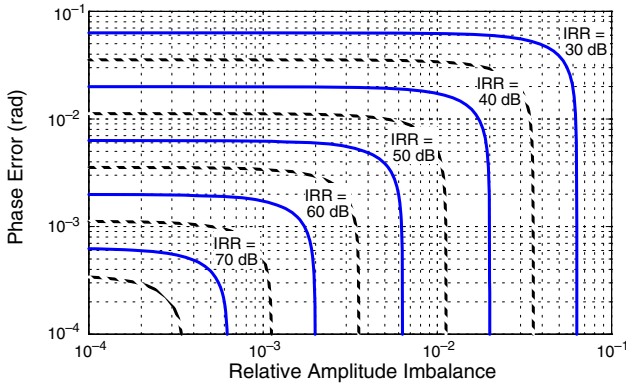
Figure 11.43   Image-rejection ratio as a function of amplitude imbalance and phase error.

0.02 rad (1.1 degree), is sufficient to limit IRR to 40 dB. Higher image suppression requires proportionally greater amplitude and phase accuracy.

11.6.2   Quadrature Filters

A quadrature signal may be filtered using a *quadrature filter*. The transfer function (H) of a quadrature filter differs from that of a real filter in that the poles and zeros of H need not come in complex-conjugate pairs, that is, H may have an asymmetric frequency response. Formal manipulation of such transfer functions in symbolic form is straightforward; realizing a quadrature filter is more cumbersome. One way to implement a quadrature filter H starts with decomposing H into $H = H_x + jH_y$, where H_x and H_y are real transfer
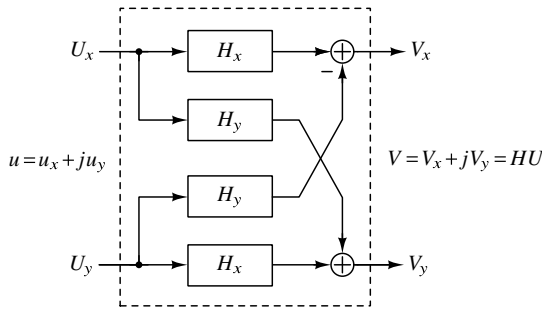
Figure 11.44   A quadrature filter.

functions. The output of the filter is

$$
\begin{aligned}
V &= HU \\
&= (H_x + jH_y)(U_x + jU_y) \\
&= (H_xUx - H_yU_y) + j(H_xUy + H_yU_x) \\
&= V_x + jV_y,
\end{aligned}
\tag{11.10}
$$

which indicates that a quadrature filter may be implemented with the lattice structure shown in Figure 11.44. This figure depicts a two-input/two-output linear system in which the transfer function from the input U_x to the output V_x is equal to that from U_y to V_y, while the transfer function from U_x to V_y is the negative of that from U_y to V_x. In practice, these symmetries are not exact and the reader may well wonder what impact such an imperfection has.

(a)   (b)
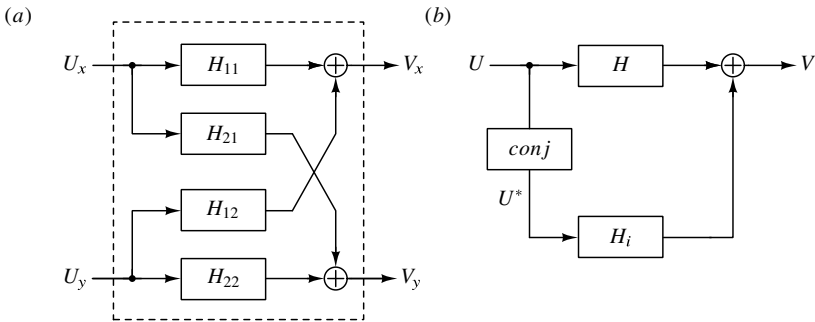
Figure 11.45   Mismatch in a quadrature filter creates an image response.

To address this question, Figure 11.45(a) shows an arbitrary two-input/two-output real linear system, whose inputs and outputs are to be interpreted as quadrature signals. As depicted in Figure 11.45(b), this system can be represented with two complex filters: one (H) operating on the unaltered signal, U, and the other (H_i) operating on its conjugate, U^*. To derive the equivalence, simply write the output of the second system in expanded form:

$$
\begin{aligned}
V &= HU + H_iU^* \\
&= (H_xU_x - H_yU_y) + j(H_xU_y + H_yU_x) + (H_{i,x}U_x + H_{i,y}U_y) + j(-H_{i,x}U_y + H_{i,y}U_x) \\
&= (H_x + H_{i,x})Ux + (H_{i,y} - H_y)U_y + j((H_y + H_{i,y})Ux + (H_x - H_{i,x})U_y). \tag{11.11}
\end{aligned}
$$

Thus,

$$
\begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix} = \begin{bmatrix} H_x + H_{i,x} & -H_y + H_{i,y} \\ H_y + H_{i,y} & H_x - H_{i,x} \end{bmatrix},
\tag{11.12}
$$

or inversely

$$
\begin{bmatrix} H_x & H_y \\ H_{i,x} & H_{i,y} \end{bmatrix} = \frac{1}{2} \begin{bmatrix} H_{11} + H_{22} & H_{21} - H_{12} \\ H_{11} - H_{22} & H_{21} + H_{12} \end{bmatrix}.
\tag{11.13}
$$

The upshot of (11.11–11.13) is that path mismatch ($H_{11} \neq H_{22}$ and/or $H_{12} \neq -H_{21}$) causes the output of a quadrature filter to contain the conjugate of the input, multiplied by the *image transfer function* $H_i = H_{i,x} + jH_{i,y}$, where $H_{i,x} = H_{11} - H_{22}$ and $H_{i,y} = H_{12} + H_{21}$. Since taking the conjugate of the input reflects its Fourier transform about $f = 0$, i.e. $(x(t) \leftrightarrow X(f)) \Rightarrow (x^*(t) \leftrightarrow X^*(-f))$, the image transfer function is responsible for transferring signal energy from positive frequencies to negative frequencies, and vice versa. We will see shortly that in a quadrature $\Delta\Sigma$ modulator this mirroring action can be highly detrimental.

At this point, it is helpful to consider two examples. First, let us suppose that we want to implement a quadrature filter with transfer function

$$
H(s) = \frac{\omega_0}{s - j\omega_0}.
\tag{11.14}
$$

Since this is a first-order transfer function with a single pole at $s = j\omega_0$, the resulting filter will therefore be a positive-frequency resonator. The H_x and H_y components of H are found by multiplying both numerator and denominator by the complex-conjugate of the denominator:

$$
H(s) = \frac{\omega_0}{s - j\omega_0} \left(\frac{s + j\omega_0}{s + j\omega_0} \right) = \frac{\omega_0 s + j\omega_0^2}{s^2 + \omega_0^2}.
\tag{11.15}
$$

Thus, the required filters are

$$
H_x(s) = \frac{\omega_0 s}{s^2 + \omega_0^2} \quad \text{and} \quad H_y(s) = \frac{\omega_0^2}{s^2 + \omega_0^2}.
\tag{11.16}
$$

These two second-order filters, as well as the computations of (11.10), can be implemented with only two real integrators configured as shown in Figure 11.46.
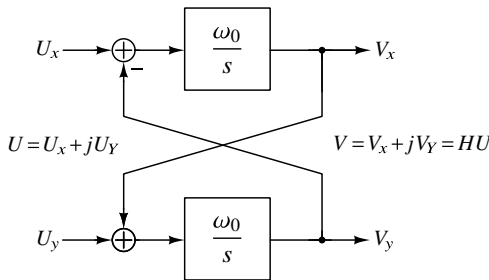
Figure 11.46   A quadrature resonator, $H(s) = \omega_0/(s - j\omega_0)$.

As a second example of quadrature filtering, consider the quadrature differential circuit shown in Figure 11.47(a). Our goal is to find the complex transfer function from u to
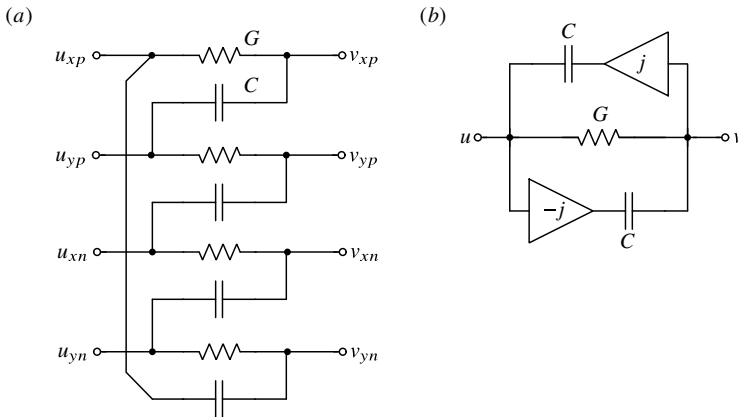
Figure 11.47   (a) A polyphase filter. (b) Equivalent quadrature quarter-circuit.

v. The brute-force method is to start by writing the four KCL equations associated with the output nodes:

$$
\begin{aligned}
(G + sC)V_{xp} &= GU_{xp} + sCU_{yp}, \\
(G + sC)V_{yp} &= GU_{yp} + sCU_{xn}, \\
(G + sC)V_{xn} &= GU_{xn} + sCU_{yn}, \\
(G + sC)V_{yn} &= GU_{yn} + sCU_{xp}.
\end{aligned}
\tag{11.17}
$$

Next we use the definition of a quadrature differential signal, namely

$$
V = (V_{xp} - V_{xn}) + j(V_{yp} - V_{yn}),
\tag{11.18}
$$

to convert the four equations in (11.17)) to one:

$$
(G + sC)V = (G - jsC)U,
\tag{11.19}
$$

from which we find the transfer function

$$
H = \frac{G - jsC}{G + sC}.
\tag{11.20}
$$

A more direct method is to analyze the equivalent *quadrature quarter-circuit*. The rules for constructing a quarter-circuit from a circuit with four-way symmetry are as follows:

a. If four equal elements connect corresponding phases of two signals (e.g., the conductance G in Figure 11.47(a)), then those elements are represented with a single element connecting the quadrature signals.

b. If the elements connect phases that are offset by $90°$ (e.g., the capacitance C), then those elements are represented with a pair of elements driven by $\pm j$ voltage buffers as shown in Figure 11.47(b). The j buffer is attached to the signal (v in Figure 11.47(a)) whose positive x phase is coupled to the other signal's positive y phase. The $-j$ buffer is connected to the other phase.

c. If the elements connect phases that are offset by $180°$, then those elements are represented with either a pair of elements driven by -1 voltage buffers, or, as is often done when constructing a differential half-circuit, with a single negative element.

Applying KCL to the output node of the circuit shown in Figure 11.47(b) yields

$$G(U - V) + sC(-jU - V) = 0 \qquad (11.21)$$

by inspection, from which (11.20) follows. With practice, an equivalent quadrature quarter-circuit can be visualized without drawing it explicitly and can thereby be analyzed quite quickly.
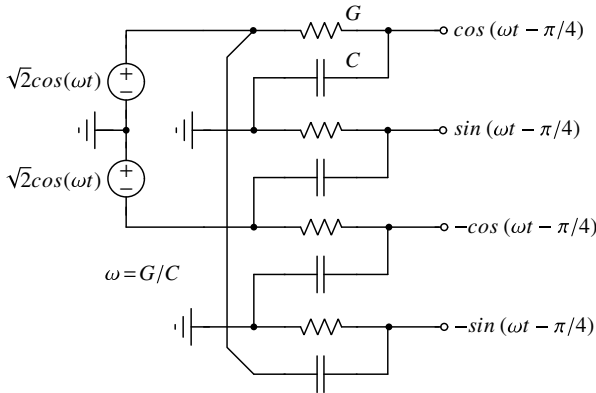
Figure 11.48   Using a polyphase circuit to generate a quadrature signal.

Since H has a zero at a negative frequency, $s = -j\omega$, where $\omega = G/C$, applying the real signal $u = \sqrt{2}\left(e^{j\omega t} + e^{-j\omega t}\right)$ to the circuit as shown in Figure 11.48 yields an output with only positive-frequency content. This circuit is commonly used to create the quadrature LO phases for a quadrature mixer from a differential sine wave. Since the quadrature is perfect at only one frequency, several polyphase filters can be cascaded to widen the frequency range.

11.7   Quadrature Modulation

As with other modulator types, the starting point in the design of a quadrature modulator is the NTF. The causality constraint $(h(0) = 1)$ is the same as that of a real modulator, and optimized zeros are as useful in quadrature systems as they are in real systems. The stability-imposed constraints on the out-of-band NTF gain appear to be similar between real and quadrature systems. The only real difference (pun intended) is that the pole/zero distribution of a quadrature NTF need not be symmetric about the real axis. Figure 11.49(a) shows the pole-zero plot of a quadrature NTF intended for an $f_0 = f_s/4$, $OSR = 32$ application. Observe that the NTF zeros are located only in the positive-frequency passband. Figure 11.49(b) plots the associated NTF magnitude. The frequency response resembles a lowpass response that has been shifted by $f_s/4$, and indeed one way to obtain a quadrature NTF is to start with a lowpass NTF and rotate its poles and zeros by multiplying them by $e^{j2\pi f_0}$.
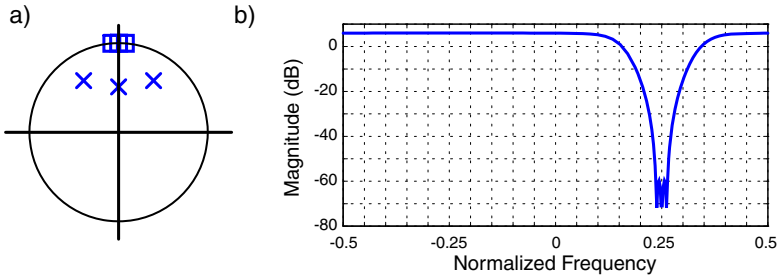
Figure 11.49   (a) Pole-zero and (b) magnitude plots for a quadrature NTF with $OSR = 32$.
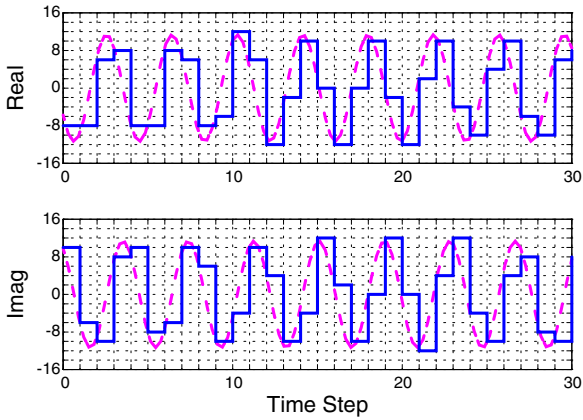
Figure 11.50   Real (I) and imaginary (Q) components of the output data from a quadrature modulator containing 16-step quantizers.

Figure 11.50 shows the simulated quadrature output data of such a modulator when the input is a −3 dBFS quadrature sine wave. As with the bandpass example, the correspondence between the input and output waveforms appears very coarse in the time domain, but a much clearer picture emerges in the frequency-domain plot of Figure 11.51, from which an SQNR of nearly 100 dB is obtained.

Figure 11.51   Simulated output spectrum for a quadrature modulator.

Nonidealities such as finite resonator Q and resonance frequency shift have deleterious effects that are similar in magnitude to those found in real systems and so are usually not problematic. However, quadrature errors caused by mismatch in the two cha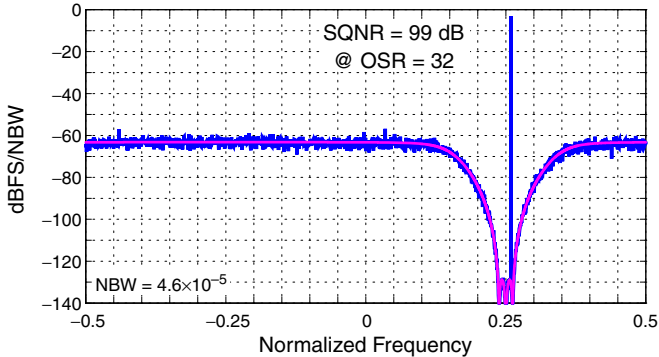nnels can be a serious source of degradation. To see this, observe that in the spectrum of Figure 11.51 the level of the quantization noise in the passband (around $f_s/4$) is nearly 65 dB below the level of the quantization noise in the image band (around $-f_s/4$). Path mismatch on the order of 0.1% (caused, for example, by mismatch in the full-scale outputs of the DACs that feed back to the first quadrature resonator) is sufficient to reflect enough image band noise to degrade the SQNR by more than 6 dB. Since much more stringent matching would be needed to ensure negligible performance degradation, path mismatch can easily be the dominant error source in a quadrature modulator.

Two methods for countering path mismatch have been described in the literature. The first involves adding one or more image zeros (and corresponding image poles) to the NTF so that the noise present in the image band is reduced. The depth of the image notch is adjusted to achieve the desired immunity to path mismatch. In addition to the increased hardware complexity, reducing the mismatch sensitivity in this way comes at the price of a reduction in the suppression of quantization noise and possibly a reduction in the stable input range. The second method for combatting mismatch applies only to DAC mismatch, and involves the use of *quadrature mismatch-shaping* [15].

The structure of a single-loop quadrature modulator follows that of real modulators, namely a loop-filter attached to a quantizer whose output is fed back to the loop-filter via DACs. The loop-filter consists of quadrature resonators such as those shown earlier in Figure 11.46. The usual variety of feedback and feedforward topologies, as well as single-loop and multi-loop architectures, are all applicable to quadrature modulators. For example, a feedback topology is depicted in Figure 11.52, while [16] describes the use of a feedforward topology. Note that structure in Figure 11.52 indicates the use of a single pair of DACs for each feedback path, that is, each feedback coefficient is assumed to be
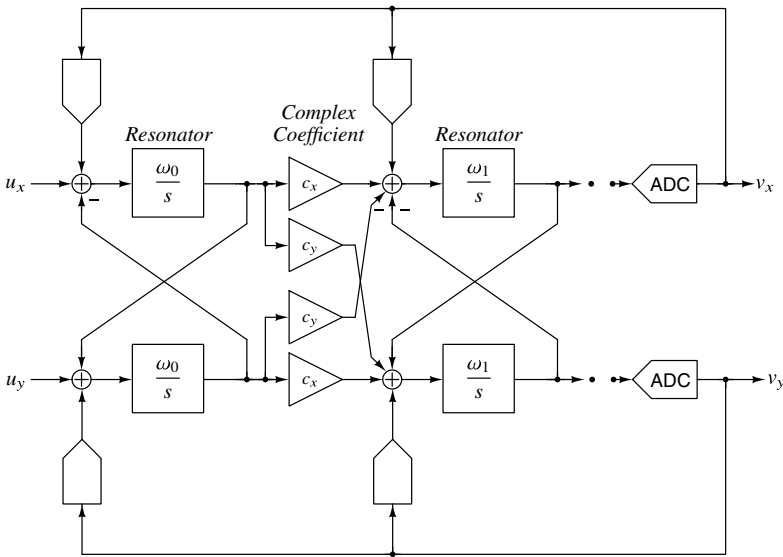
Figure 11.52   A quadrature modulator employing the feedback topology.

real. A complex feedback coefficient, which would require two DAC pairs, can be forced to be real by rotating it via multiplication by $e^{j\phi}$ and then multiplying the incoming and outgoing interstage coupling coefficients of that stage by $e^{j\phi}$ and $e^{-j\phi}$, respectively. Since complex interstage coefficients are usually less troublesome to implement than complex DACs, this operation usually simplifies the loop-filter.
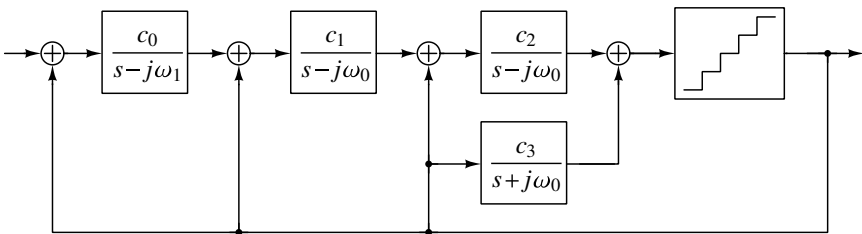
Figure 11.53   A quadrature modulator with a parallel path for the image resonator.

If the NTF contains image zeros, it is not advisable to simply tack the image resonator onto the end of a cascade of positive-frequency resonators, since the image resonator typically attenuates in-band signals. A more practical topology is shown in Figure 11.53 in which the image resonator(s) are placed in parallel with the in-band resonators. This topology has the added benefit of putting an STF zero in the image band.

As suggested in Figure 11.52, a quadrature DAC can be implemented with a pair of independent real DACs. Figure 11.54 compares this implementation with one that uses a single current-mode DAC consisting of elements with 4-way switching. Figure 11.54 also shows the constellations using two elements (so that the total DAC current is the same). In the first architecture, one DAC element is dedicated to the x component and the other is dedicated to the y component. In the second architecture, both elements can contribute to
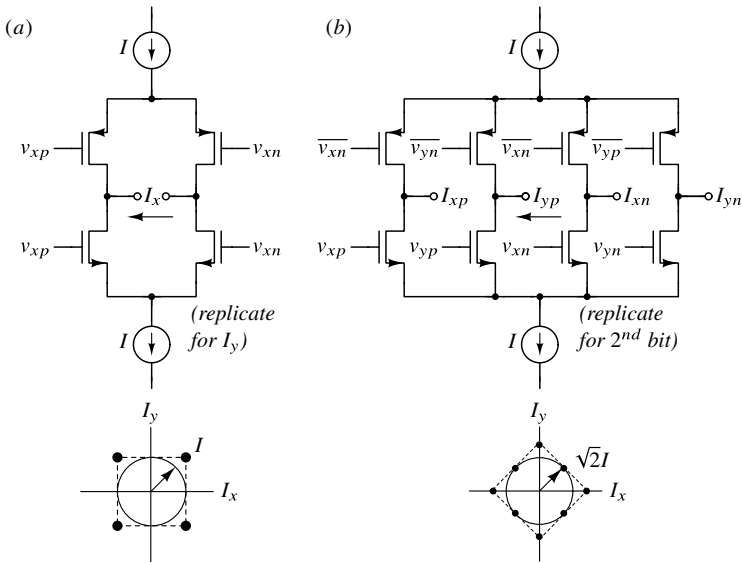
Figure 11.54   Comparison of quadrature DAC implementations.

the x and y components and thereby support a signal amplitude that is 3 dB higher than the first architecture. This extra range signal range can equate to a 3-dB FOM improvement for the DAC.
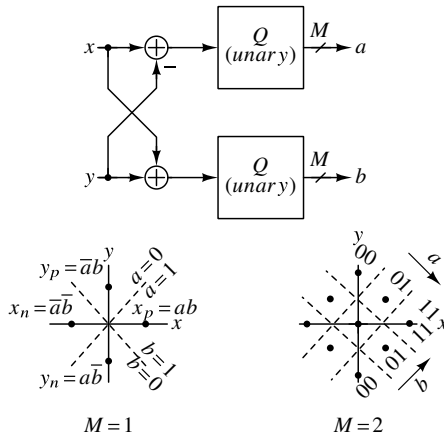
Figure 11.55   Quadrature quantizer for 4-way DAC elements.

The quantizer in a quadrature modulator can be implemented with a pair of real quantizers. In this case, the unary codes produced by a pair of flash ADCs can be applied to a set of 2-way DAC elements such as those shown in Figure 11.54(a). To interface to a set of 4-way elements, the arrangement depicted in Figure 11.55 may be used. Here, a pair of real quantizers are driven by the sum and difference signals and their unary outputs a and b are decoded to produce the drive signals for individual DAC elements. For example, as

illustrated in Figure 11.55 for $M = 1$ element, the required decoding is

$$\begin{aligned}
x_p &= a \cdot b, \\
y_p &= \bar{a} \cdot b, \\
x_n &= \bar{a} \cdot \bar{b}, \\
y_n &= a \cdot \bar{b}.
\end{aligned} \tag{11.22}$$

For arbitrary M it suffices to decode pairs of bits from the a and b signals using (11.22). Note that the pairing and ordering of the bits in the unary signals is irrelevant, and thus the signals may have their bits scrambled. The reader may wish to verify these claims for $M = 2$ using the diagram in Figure 11.55.
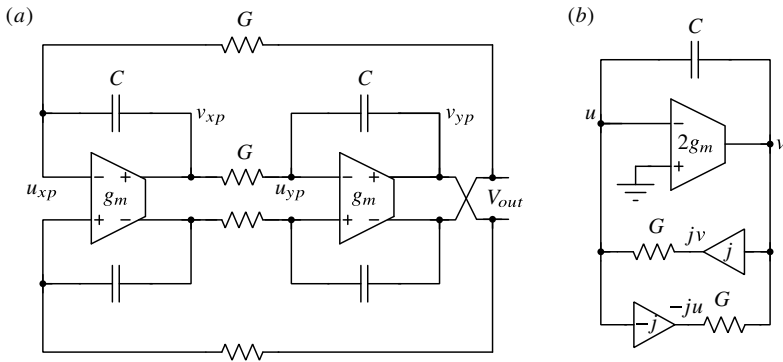
Figure 11.56   A high-Q quadrature resonator (a) differential circuit; (b) equivalent quadrature quarter-circuit.

Section 11.4.2 made note of the significant advantages associated with using an LC tank within a bandpass ADC. Unfortunately, a quadrature equivalent of an LC tank does not appear to exist. Despite the numerous indications that quadrature ADCs are natural extensions of real ADCs, it appears that passive quadrature resonances do not exist in nature. As some compensation for this disappointing fact, Figure 11.56 will be used to show that an active-RC resonator has a particularly attractive realization in quadrature form. Normally, the amplifiers in an active-RC resonator need high gain at the resonance frequency in order to ensure a high-Q resonance. Achieving high gain is difficult in an active-RC resonator because the amplifier must drive both a resistor and a capacitor and conventional wisdom is that in such cases the amplifier needs to provide a low output impedance. However, as we will show shortly, the circuit in Figure 11.56 is able to achieve a high-Q resonance by using a plain transconductor.

Applying KCL at the U and V nodes of the quadrature quarter circuit in Figure 11.56(b) gives

$$(sC + G)U = (sC + jG)V \tag{11.23}$$

and

$$(sC + G)V = (sC - jG - 2g_m)U. \tag{11.24}$$

Eliminating U,

$$
\begin{aligned}
(sC + G)^2 V &= (sC - jG - 2g_m)(sC + G)U, \\
(sC + G)^2 V &= (sC - jG - 2g_m)(sC + jG)V, \\
((sC)^2 + 2sCG + G^2)V &= ((sC)^2 - jG(sC) - 2g_m(sC) + jG(sC) + G^2 + 2jg_mG)V, \\
(2sC(G + g_m) - 2jg_mG)V &= 0.
\end{aligned}
\tag{11.25}
$$

Thus, we see that the pole of the system is at

$$
s = \frac{jG}{C(1 + G/g_m)}.
\tag{11.26}
$$

This is a pole that is on the $j\omega$ axis, i.e. has infinite Q, regardless of the value of g_m. The only effect of finite g_m is that the pole is displaced from its ideal location. For reasonable values of g_m, this displacement can be overcome by tuning either G or C. This analysis contains a few assumptions that deserve mention, namely negligible phase shift in the transconductor, negligible loading from subsequent stages, and input signals injected as currents. If the designer is unable to satisfy these assumptions, the associated degradation needs to be checked to ensure that it is acceptable.

11.8 Polyphase Signal Processing

Having seen how quadrature signal processing endows a mixer with image rejection and increases the bandwidth of a $\Delta\Sigma$ ADC, the reader may wonder if going beyond a quadrature representation offers further advantages. One way to extend the concept of quadrature signal processing is to observe that a quadrature signal consists of two phases and therefore consider a signal consisting of three phases: a, b and c. If we combine the three phases according to

$$
z = a + qb + q^2 c,
\tag{11.27}
$$

where

$$
q = e^{j\phi}, \phi = \frac{2\pi}{3},
\tag{11.28}
$$

then we have a way to represent a complex signal with three real signals. We now show that two very helpful properties emerge from this generalization: elimination of the $3LO$ term in a square-wave mixer and rejection of third-order distortion.

Consider the 3-phase LO signal illustrated in Figure 11.57. Since each phase of the LO is non-overlapping with the other two, the mixing operation can be performed by a passive mixer which switches the input signal into each phase of the output according to which phase of the LO is active. Taking the Fourier transform of the composite LO signal reveals that the spectrum consists of the desired $e^{j\omega_{LO}t}$ fundamental component plus terms spaced by multiples of $6\omega_{LO}$ on either side. Thus, the LO signal is devoid of many spurious terms including the image ($-LO$) and $\pm 3LO$ components, and the RF filtering needed in advance of the mixer can be relaxed. A lesser advantage of three-phase mixing compared to quadrature mixing is that the fundamental has a magnitude of $3/\pi$, which is only 0.4 dB below the total power of the LO signal and thus the noise penalty associated with the non-fundamental components of the LO is small. In contrast, quadrature square-wave mixing entails a 1-dB penalty.
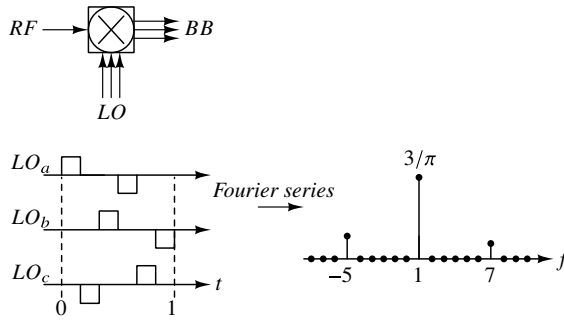
Figure 11.57   Three-phase mixing.

The second major advantage of three-phase signal processing relates to distortion cancellation. We know that differential circuits reject second-order distortion; it turns out that three-phase circuits reject third-order distortion. To see why, consider the third-order nonlinearity

$$f(x) = 4x^3 - 3x. \tag{11.29}$$

Subjecting a sine wave $\cos(\omega t + \theta)$ to this nonlinearity simply triples the argument,[3] yielding a distortion term $\cos(3\omega t + 3\theta)$. Applying this nonlinearity to the three-phase signals

$$
\begin{aligned}
a &= \cos(\omega t), \\
b &= \cos(\omega t - \phi), \\
c &= \cos(\omega t + \phi),
\end{aligned}
\tag{11.30}
$$

yields distortion terms

$$
\begin{aligned}
a_3 &= \cos(3\omega t), \\
b_3 &= \cos(3\omega t - 3\phi), \\
c_3 &= \cos(3\omega t + 3\phi).
\end{aligned}
\tag{11.31}
$$

Since $3\phi = 2\pi$, using (11.27) gives

$$z_3 = a_3 + q b_3 + q^2 c_3 = (1 + q + q^2)\cos(3\omega t) = 0. \tag{11.32}$$

The third-order distortion terms therefore appear as a common-mode signal that is rejected when the complex signal is formed.

A similar analysis of the distortion products under two-tone excitation shows that although the sum terms ($3\omega_1$, $2\omega_1 + \omega_2$, $\omega_1 + 2\omega_2$ and $3\omega_2$) cancel, unfortunately the difference terms ($2\omega_1 - \omega_2$ and $2\omega_2 - \omega_1$) do not. As a result, polyphase signal processing is not quite the panacea for distortion that (11.32) might suggest. Nonetheless, for receiver scenarios in which a single large interferer is present, polyphase signal processing affords some relief.

Polyphase circuits are a natural extension of quadrature signal processing and play to one of the strengths of VLSI circuits, namely replication of matched elements. As with

---

[3]This convenient property comes from the fact that $f(x)$ is a Chebyshev polynomial. Expanding a nonlinearity into a weighted sum of Chebyshev polynomials rather than a plain power series simplifies analysis of harmonic distortion.
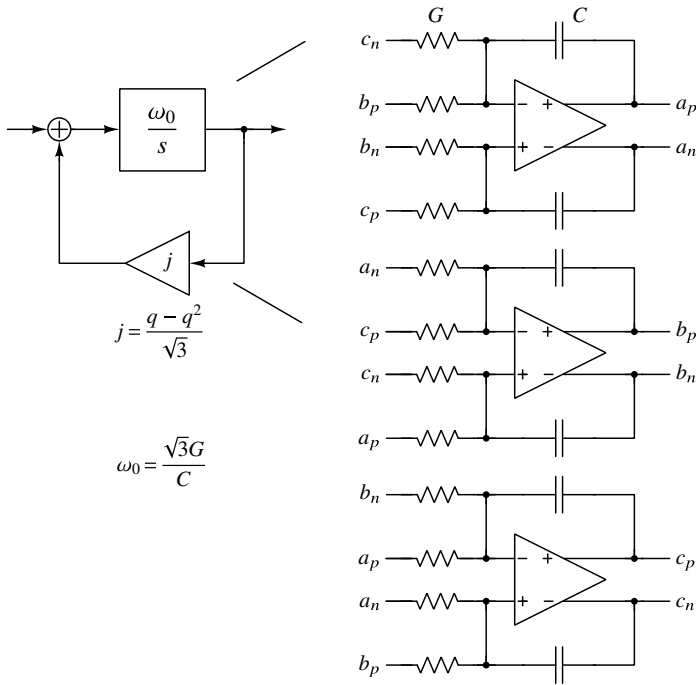
Figure 11.58   A 6-phase resonator.

differential signal processing or quadrature signal processing, the advantages of polyphase signal processing come with no power penalty, to a first-order approximation. The circuits are also aesthetically pleasing, as evidenced by Figure 11.58, which depicts a 6-phase positive-frequency resonator. Such a circuit could be used in the loop-filter of a polyphase bandpass $\Delta\Sigma$ ADC.

11.9 Conclusions

The delta-sigma ADCs described in this chapter can be used to digitize narrowband bandpass and narrowband quadrature signals. No ADC architecture is able to focus its resolving power on a particular frequency band the way bandpass ADCs can. In addition to the standard advantages of $\Delta\Sigma$ modulation such as robustness, high linearity, and, for continuous-time implementations, easy interfacing, inherent anti-aliasing, and a small and readily-adjustable full-scale, the advantages of bandpass modulation include

a. Simplified realization of a superheterodyne receiver.

b. Perfect I/Q balance.

c. Immunity to dc offset and $1/f$ noise.

d. Even-order distortion products of in-band signals fall out-of-band.

e. (Soon) Digitization of RF signals without analog mixing.

Efficient realizations of high-performance bandpass ADCs are possible, especially if a physical resonance, such as that of an LC tank, is exploited in the construction of the loop-filter. A bandpass ADC supporting center frequencies up to several hundred MHz is commercially available now, whereas quadrature bandpass ADCs are not yet available in stand-alone form.

References

[1] H. Shibata , R. Schreier, W. Yang, A. Shaikh, D. Paterson, T. Caldwell, D. Alldred, and P.W. Lai, "A DC-to-1 GHz tunable RF $\Delta\Sigma$ ADC achieving DR = 74 dB and BW =150 MHz at f_0 = 450 MHz using 550 mW," *IEEE Journal of Solid-State Circuits*, vol. 47, no. 12, pp. 2888–2897, Dec. 2012.

[2] J. Harrison, M. Nesselroth, R. Mamuad, A. Behzad, A. Adams, and S. Avery, "An LC bandpass $\Delta\Sigma$ ADC with 70 dB SNDR over 20MHz bandwidth using CMOS DACs," *International Solid-State Circuits Conference Digest of Technical Papers*, pp. 146–148, Feb. 2012.

[3] J. Ryckaert, J. Borremans, B. Verbruggen, L. Bos, C. Armiento, J. Craninckx, and G. Van der Plas, "A 2.4 GHz low-power sixth-order RF bandpass $\Delta\Sigma$ converter in CMOS," *IEEE Journal of Solid-State Circuits*, vol. 44, no. 11, pp. 2873–2880, Nov. 2009.

[4] L. Luh, J. Jensen, C. Lin, C. Tsen, D. Le, A. Cosand, S. Thomas, and C. Fields, "A 4 GHz 4th-order passive LC bandpass Delta-Sigma modulator with IF at 1.4 GHz," *Symposium on VLSI Circuits Digest of Technical Papers*, pp. 168–169, Feb. 2006.

[5] *AD6676 Wideband IF receiver subsystem datasheet*, Analog Devices, Norwood, MA, Nov. 2014.

[6] A. Hairapetian, "An 81 MHz IF receiver in CMOS," *IEEE Journal of Solid-State Circuits*, vol. 31, no. 12, pp. 1981–1986, December 1996.

[7] D. Senderowicz, D. A. Hodges, and P. R. Gray, "An NMOS integrated vector-locked loop," *Proceedings IEEE International Symposium on Circuits and Systems*, pp. 1164–1167, 1982.

[8] H. Khorramabadi, and P. R. Gray, "High-frequency CMOS continuous-time filters," *IEEE Journal of Solid-State Circuits*, vol. 19. pp. 939–948, Dec. 1984.

[9] F. Krummenacher and N. Joehl, "A 4-MHz CMOS continuous-time filter with on-chip automatic tuning," *IEEE Journal of Solid-State Circuits*, vol. 23. pp. 750–758, June 1988.

[10] J. Van Engelen and R. Van De Plassche, *Bandpass sigma delta modulators-stability analysis, performance and design aspects*, Norwell, MA: Kluwer Academic Publishers 1999.

[11] O. Shoaei and, W. M. Snelgrove, "A multi-feedback design for LC bandpass delta-sigma modulators," *IEEE International Symposium on Circuits and Systems*. vol. 1, pp. 171–174, May 1995.

[12] F. Bruccoleri, E. A. M. Klumperink, and B. Nauta, "Wide-band CMOS low-noise amplifier exploiting thermal noise canceling," *IEEE Journal of Solid-State Circuits*, vol. 39, pp. 275–282, Feb. 2004.

[13] R. Bagheri, A. Mirzaei, S. Chehrazi, M. E. Heidari, M. Lee, M. Mikhemar, W. Tang, and A. A. Abidi, "An 800-MHz,6-GHz software-defined wireless receiver in 90-nm CMOS," *IEEE Journal of Solid-State Circuits*, vol. 41, no. 12, pp. 2860–2876, Dec. 2006.

[14] B. Razavi, *RF Microelectronics*, Englewood Cliffs, NJ: Prentice-Hall, 1997.

[15] R. Schreier, "Quadrature mismatch-shaping," *Proceedings, IEEE International Symposium on Circuits and Systems*, vol. 4, pp. 675–678, May 2002.

[16] K. Philips, "A 4.4 mW 76 dB complex ΣΔ ADC for Bluetooth receivers," *International Solid-State Circuits Conference Digest of Technical Papers*, pp. 64–65, Feb. 2003.

[17] H. Chae, J. Jeong, G. Manganaro, and M. Flynn, "A 12 mW low power continuous-time bandpass ΔΣ modulator with 58 dB SNDR and 24 MHz bandwidth at 200 MHz IF," *International Solid-State Circuits Conference Digest of Technical Papers*, pp. 148–149, Feb. 2012.

[18] C. Y. Lu, J. F. Silva-Rivas, P. Kode, J. Silva-Martinez, and F. S. Hoyos, "A sixth-order 200 MHz IF bandpass Sigma-Delta modulator with over 68 dB SNDR in 10 MHz bandwidth," *IEEE Journal of Solid-State Circuits*, vol. 45, no. 6, pp. 1122–1136, Jun. 2010.

[19] T. Yamamoto, M. Kasahara, and T. Matsuura, "A 63 mA 112/94 dB DR IF bandpass ΔΣ modulator with direct feedforward compensation and double sampling," *IEEE Journal of Solid-State Circuits*, vol. 43, pp. 1783–1794, Aug. 2008.

[20] K. Yamamoto, A. C. Carusone, and F. P. Dawson, "A Delta-Sigma modulator with a widely programmable center frequency and 82-dB peak SNDR," *IEEE Journal of Solid-State Circuits*, vol. 43, pp. 1772–1782, July 2008.

[21] T. Salo, S. Lindfors, and K. A. I Halonen, "A 80-MHz bandpass ΔΣ modulator for a 100-MHz IF receiver," *IEEE Journal of Solid-State Circuits*, vol. 37, no. 7, pp. 1798–1808, July 2002.

[22] M. Inerfield, W. Skones, S. Nelson, D. Ching, P. Cheng, and C. Wong, " High dynamic range InP HBT delta-sigma analog-to-digital converters," *IEEE Journal of Solid-State Circuits*, vol. 38, pp. 1524–1532, Sept. 2003.

CHAPTER 12

INCREMENTAL ANALOG-TO-DIGITAL CONVERTERS

In contrast with all other converters discussed in this book, the incremental A/D converters (IADCs) described in this chapter are *Nyquist-rate* data converters. For such converters, each digital word generated at the output depends only on the samples of the analog input during the conversion interval; the behavior of the input outside this interval is immaterial. This property is obtained by resetting the $\Delta\Sigma$ modulators within the IADC. IADCs are typically used to convert narrowband signals with very high accuracy. They are often used in biomedical as well as instrumentation and measurement applications.

12.1 Motivation and Trade-Offs

In many instrumentation and measurement applications, integrated sensor interface circuits are required to prepare the analog sensor output for digital signal processing. Typical applications include digital voltmeters, image sensors, and biosensors. In some cases, such as image sensors and electroencephalograms, a single interface should also be shared among many sensors. Often these sensors are battery-powered devices, and hence power dissipation in the interface circuitry is of great concern. The interface usually requires a low-noise amplifier, a noise-suppressing anti-aliasing filter, and an analog-to-digital converter. In a typical application, the specifications of the ADC may include one or more of the following requirements:

a. High absolute accuracy (over 20 bits).
b. Small offset and gain errors (a few μV).
c. Low output noise (a few μV).
d. High linearity (over 16 bits).
e. Low power (a few μW).
f. Ease of multiplexing for multi-sensor systems.

The available ADC configurations for such high-accuracy requirements include dual-slope Nyquist-rate converters and delta-sigma modulators. However, dual-slope converters are very slow. They require many clock periods to obtain an output word: the necessary number of clock periods is 2^{N+1} for N-bit accuracy. $\Delta\Sigma$ ADCs are much faster than the dual-slope ones, but they are also more complicated. They need digital post-filters, and generally exhibit gain and offset errors. They are also subject to idle tones and instability. Since $\Delta\Sigma$ ADCs rely on both analog and digital memory to achieve high accuracy, they can only be shared among multiple sensors if all memory elements are replicated many times. Also, due to their elaborate digital filters, they have significant latency between their analog inputs and digital outputs.

A different ADC scheme, one that applies the noise-shaping algorithm of $\Delta\Sigma$ ADCs, but only within the sample-by-sample operation of a Nyquist-rate ADC, is the *incremental ADC* (IADC) discussed in this chapter. IADCs are well suited for satisfying the six requirements listed above. Figure 12.1 shows the basic block diagram of the IADC.
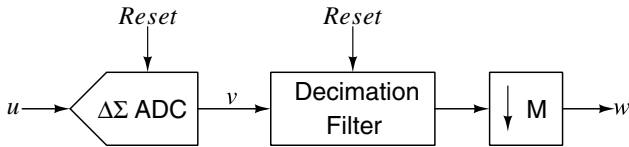
Figure 12.1   IADC block diagram.

The IADC scheme is similar to that of a single-stage $\Delta\Sigma$ ADC. The main difference is that the reset switches are turned on between conversions, rather than only during start-up or in response to overload. In an IADC, the reset switches discharge or reset all memory elements (capacitors in the $\Delta\Sigma$ modulator, storage registers in the decimation filter). This changes the character of the ADC from a continuously running converter to an intermittently operated one. This feature allows easy multiplexing, and also the use of sleep mode to reduce power dissipation, and allows an easy speed–power trade-off. Figure 12.2 illustrates the region of typical applications for IADCs as compared with those of other ADC schemes.

12.2   Analysis and Design of Single-Stage IADCs

Figure 12.3 illustrates a *third-order $\Delta\Sigma$ CIFF ADC*, converted into an IADC by the addition of the switches that reset the integrators after every M clock periods. Note that the circuit uses a unity-gain feedforward path connecting the ADC input to the quantizer input. This connection has two beneficial effects. As discussed for $\Delta\Sigma$ ADCs, the loop-filter only

Figure 12.2   ADC operating regions.

(*a*)

(*b*)

Figure 12.3   Third-order IADC.

needs to process the quantization error. This allows relaxed linearity specifications for the amplifiers. Also the input signal (almost) immediately appears at the output of the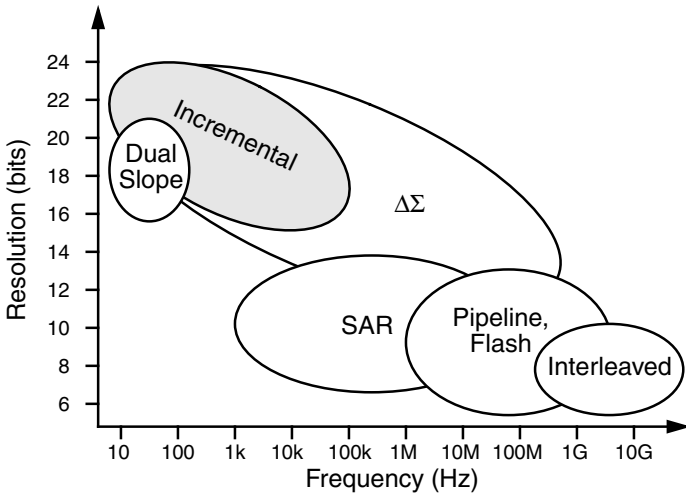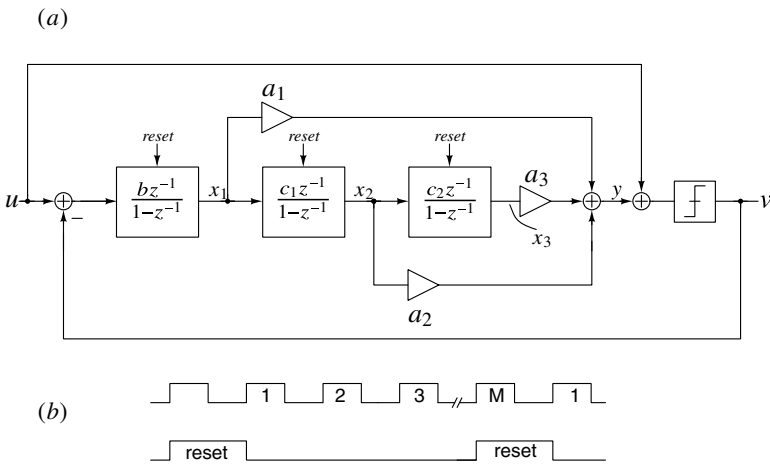 loop. As will be shown later, this improves the SNR of the converter, since the digital filter following the loop assigns a decreasing scale factor to the loop output samples $v[k]$.

Time-domain analysis shows that output signal of the third integrator after the Mth clock period is

$$x_3[M] = bc_1c_2 \sum_{n=2}^{M} \sum_{l=1}^{n-1} \sum_{k=0}^{l-1} (u[k] - v[k] \cdot V_{ref}). \tag{12.1}$$

Here $v[k]$ is the digital output after the kth clock period, and V_{ref} the reference voltage of the feedback DAC. Assuming that u is held constant during all M clock periods, we have

$$\frac{M(M-1)(M-2)}{6} \frac{u}{V_{ref}} - \sum_{n=2}^{M} \sum_{l=1}^{n-1} \sum_{k=0}^{l-1} v[k] = \frac{1}{bc_1c_2} \frac{x_3[M]}{V_{ref}}. \tag{12.2}$$

For a stable $\Delta\Sigma$ ADC, $|x_3| < V_{ref}$ can be obtained. Then, the term on the RHS of (12.2) is much smaller than those on the LHS, and hence the approximation

$$\frac{u}{V_{ref}} \approx G \sum_{n=2}^{M} \sum_{l=1}^{n-1} \sum_{k=0}^{l-1} v[k], \tag{12.3}$$

where

$$G = \frac{6}{M(M-1)(M-2)} \approx \frac{6}{M^3} \tag{12.4}$$

can be used to find an estimate of u. The estimation error corresponds to an LSB voltage

$$V_{LSB} = G \frac{2}{bc_1c_2} V_{ref}. \tag{12.5}$$

The equivalent number of bits of the ADC is then given by $ENOB = \log_2(2V_{ref}/V_{LSB})$. Equations (12.2–12.5) suggest the following design process for the third-order IADC:

a. Design a third-order low-distortion CIFF $\Delta\Sigma$ ADC using the Delta-Sigma toolbox. Carry out dynamic-range scaling to prevent overloading the integrators and the quantizer. This gives the values of branch factors a_1, a_2, and a_3, as well as the integrator gain factors b, c_1, and c_2.

b. Use (12.4) and (12.5) to find the lowest value of M needed to meet the SQNR specifications. As discussed in earlier chapters, it is advisable to make $SQNR \gg SNR_{spec}$ because, for power efficiency, most of the noise budget should be assigned to the thermal noise.

c. As (12.3) suggests, the digital estimate of u/V_{ref} can be obtained by using three digital accumulators and a multiplier.

Note that the resolution of the internal quantizer appears only indirectly in the error formula (12.5), through the bc_1c_2 product. Since this factor, after dynamic-range scaling, is inversely proportional to the step size of the quantizer, higher quantizer resolution gives smaller V_{LSB}, as expected.

An alternative analysis can be based on z-domain arguments. Since the loop output is

$$V(z) = STF(z) \cdot U + NTF(z) \cdot E(z), \tag{12.6}$$

choosing the transfer function of the digital decimation filter as $H(z) = 1/NTF(z)$ yields the overall digital output $W(z)$:

$$W(z) = H(z) \cdot V(z) = STF(z) \cdot \frac{U}{NTF(z)} + E(z). \tag{12.7}$$

For the case of a low-distortion $\Delta\Sigma$ ADC with a "maximally flat" noise transfer function $NTF(z) = (1 - z^{-1})^3$, the transfer function $H(z)$ can be realized by three cascaded accumulators. To keep the dc gain of H equal to 1, the scale factor G, defined in (12.4), must also be included. This leads to

$$w[M] = \frac{u}{V_{ref}} + G \cdot e[M]. \tag{12.8}$$

Thus, the final output $w[M]$ of the digital filter gives the estimate of u/V_{ref}. The error of the estimation is $G \cdot e[M]$. Here $e[M]$ is the last value of the quantization error of the internal quantizer. It satisfies $|e[M]| < \Delta/2$, where Δ is the quantizer step size. Unlike (12.5), the error formula (12.8) allows finding the oversampling ratio M even before the block-level design of the $\Delta\Sigma$ ADC loop is completed.

12.3 Digital Filter Design for Single-Stage IADCs

The analog loop design process for a single-stage IADC is essentially the same as for a single-stage $\Delta\Sigma$ ADC. However, the design of the decimation filter is different – in fact, it is usually much simpler. It was shown in Section 12.2 that for the third-order IADC the digital estimate of u/V_{ref} can be obtained from the triple summation of the digital output of the quantizer, multiplied by $G = 6/[M(M-1)(M-2)]$. In the general case of an Lth-order IADC, L accumulators are needed, and the scale factor is $G = L!/[M(M-1)(M-2)(M-L+1)]$. For improved dynamic range, in the third-order IADC the scaler can be split into factors $1/M$, $2/(M-1)$, and $3/(M-2)$, and each factor assigned to an accumulator. To avoid the costly division required by these factors, it is also advantageous to choose $M = 2^n$, where n is an integer, and to use the approximation

$$\frac{1}{M-k} \approx \frac{1 + \frac{k}{M} + \left(\frac{k}{M}\right)^2 + \left(\frac{k}{M}\right)^3 + \cdots}{M}, \quad k = 1, 2. \tag{12.9}$$

Here, the multiplication by $1/M = 2^{-n}$ requires only shifting the binary point by n places, so all factors can be easily and cheaply found. If not needed, the higher-order terms $(k/M)^2$ and $(k/M)^3$ may be neglected in the approximation.

An alternative realization of the decimation filter can be based on the finite-length convolution of the output sequence $\{v[k]\}$ of the loop with the M values of the finite impulse response $\{h[k]\}$ of the filter [8]. The impulse response $\{h[k]\}$ is the inverse z-transform of the transfer function $H(z)$ of the digital filter. It can easily be obtained by applying the impulse sequence $\{1, 0, 0, \cdots\}$ to the known structure (here, a cascade of accumulators) of the filter. For $L = 1$, this process gives $h[k] = 1$ for all M values $k = 0, 1, 2, \cdots (M-1)$.

For $L = 2$, the result is $h[k] = k + 1$; for $L = 3$, $h[k] = (k + 1)(k + 2)/2$. The scale factor needed to make the sum of all elements $h[k]$ equal to one (and thus $H(1) = 1$) must also be included.

A more sophisticated design technique for the digital filter that can minimize the weighted sum of the thermal and quantization noises is described in [8]. It assumes that the thermal noise is white with mean-square value $\gamma k_B T/C_{in}$. Here k_B is the Boltzmann constant, T is the temperature in Kelvin, and γ is a scale factor determined by the circuitry of the input branch [1]. Typically, $\gamma \approx 5$. Then, it can be shown [8] that the mean-square value of the output thermal noise is given by

$$P_t = \frac{\gamma k_B T}{C_{in}} h^T S^T S h. \tag{12.10}$$

Here h is an M-element column vector whose kth element is $h[k]$, the kth sample of the impulse response of the decimation filter. S is an $M \times M$ lower triangular matrix

$$S = \begin{bmatrix} s[0] & 0 & 0 & \cdots & 0 \\ s[1] & s[0] & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ s[M-1] & s[M-2] & s[M-3] & \cdots & s[0] \end{bmatrix}, \tag{12.11}$$

where $s[k]$ is the kth sample of the impulse response of the signal path from the input to the output of the loop. For the low-distortion loop, S becomes the unit matrix, and hence $P_t = (\gamma k_B T/C_{in})|h|^2$. To minimize the thermal noise, $h[k]$ should be chosen so as minimize P_t, subject to the condition that the dc gain of the digital filter should be one. This condition translates into

$$e \cdot h = 1, \tag{12.12}$$

where $e = [1\ 1\ 1 \ldots 1]^T$ is an M-element column vector. For the low-distortion case, this gives the minimum P_t for $h[k] = 1/M$ for $k = 0, 1, \ldots, (M - 1)$. Thus, all tap weights of the optimized decimation filter are the same for thermal noise minimization if the low-distortion architecture is used.

The estimation of the power of the contribution of the *quantization error* in the output is similar to the one performed above for the thermal noise. It will be assumed that the samples $e[k]$ behave as a zero-mean noise with uncorrelated samples, and that they have a mean-square value of $\Delta^2/12$, where Δ is the step size of the quantizer. (Note that this assumption rests on conditions which ensure their randomness, and may necessitate the use of a dither signal in the loop.) Let $\{n[k]\}$ be the impulse response of the quantization noise transfer function, from the quantizer to the output of the loop. It is the inverse transform of the noise transfer function $NTF(z)$ of the loop, windowed by the reset pulse. Then, defining the $M \times M$ matrix N generated from the $n[k]$ samples the same way as S was generated from the $s[k]$, the power P_q of the output quantization noise can be expressed in the form

$$P_q = \frac{\Delta^2}{12} h^T N^T N h. \tag{12.13}$$

To minimize the output quantization noise power, P_q given by (12.13) needs to be minimized, subject to constraint (12.12). This task can be performed analytically, using

the Lagrange multiplication method [8]. The resulting optimum impulse response of the decimation filter is given by

$$h_{opt} = \frac{Re}{e^T Re},$$ (12.14)

where $R = [N^T N]^{-1}$ and e is the unit-element vector defined above. Due to the structure of N, the matrix $N^T N$ cannot be singular, and thus R always exists. Alternatively, available software (e.g., the MATLAB function `quadprog`) can be used to find h_{opt}. It can be predicted that for an Lth-order loop the first L elements of h_{opt} will be zero or very small because the last L output samples of the loop will contain quantization error samples that will not be canceled by subsequent samples. Hence, these must have small weight factors in the optimum solution.

The optimization of the digital filter transfer function $H(z)$ for both quantization and thermal noises is based on minimizing the sum of P_t and P_q subject to the prescribed dc gain of the filter. Specifying as before $H(1) = 1$, and defining the matrix

$$O = \frac{\gamma k_B T}{C_{in}} S^T S + \frac{\Delta^2}{12} N^T N,$$ (12.15)

our task becomes finding h so as to achieve

$$\min_h (P_t + P_q) = \min_h (h^T O h)$$ (12.16)

subject to $e \cdot h = 1$. For this general case, the process described for the minimization of P_q remains applicable, and h_{opt} is still given by (12.14), but now $R = [O^T O]^{-1}$ holds, where O is given by (12.15). The optimum impulse response $\{h[k]\}$ of the digital filter will now be a compromise between the ones applicable in the extreme cases discussed above. As an example, the MATLAB™ code fragment below follows the example of [8] to yield the responses shown in Figure 12.4.
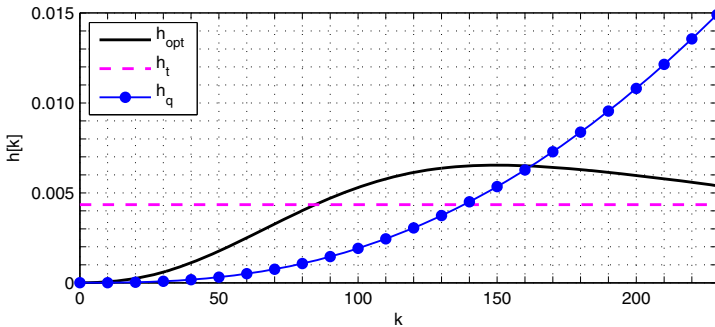
Figure 12.4   Optimal impulse responses of the digital filter of a third-order IADC.

```
%% Modulator description from [8]
M = 230;        % Decimation factor
Cin = 2e-12;    % Input capacitance
Vref = 1;       % Reference voltage
Vfs = sqrt(2);  % Full-scale input voltage
```

```
nlev = 5;         % Number of quantization levels
% Coefficients for low-distortion CIFF topology
a = [1.0398 0.4870 0.0967];
g = 0;
b = [1 0 0 1];
c = [1 1];

%% Calculation of optimal impulse response
ABCD = stuffABCD(a,g,b,c,'CIFF');
[ntf stf] = calculateTF(ABCD);
n = impulse(ntf,M); s = impulse(stf,M);
N = zeros(M,M);      S = zeros(M,M);
for i = 1:M
    N(i:M+1:M*(M+1-i)) = n(i);
    S(i:M+1:M*(M+1-i)) = s(i);
end
delta = 2*Vref/(nlev-1);
gamma = 5;
k = 1.38e-23;        % Boltzmann constant
T = 300;
t2 = gamma*k*T/Cin;
q2 = delta^2/6;      % Assumes 1 LSB of dither
O = t2*(S'*S) + q2*(N'*N);
R = inv(O'*O);
e = ones(M,1);
% Optimal impulse response
h_opt = R*e / (e'*R*e);
% Optimal impulse response for quantiztion noise only
h_q = inv(N'*N)*S*e;
h_q = h_q/sum(h_q);
% Optimal impulse response for thermal noise only
h_t = e'/M;
```

The curves show the optimum $\{h[k]\}$ responses for minimizing the thermal noise power P_t (dashed line), the quantization noise power P_q (dotted curve), and the total output noise (continuous curve). Note that the areas under the three curves are the same, but the individual properties differ, as discussed above. As expected, the response for minimum thermal output noise is constant, and the response for minimum quantization output noise is similar to a quadratic parabola. For optimum total noise, the curve initially follows the quantization noise response, since this determines the noise introduced at the end of the conversion. After that, it approaches the thermal noise response.

The decimation filter DF performs the convolution of the loop output data $\{v[k]\}$ with the FIR impulse response $\{h[k]\}$, discussed above. It needs to be implemented in an economical way. Since the output w of the DF is down-sampled by M, only the last result of the convolution needs to be calculated. The M coefficients $h[k]$, $k = 0, 1, \cdots, (M-1)$, can be stored, and a simple multiply-accumulate (MAC) stage may be used to carry out the calculation of w. Since the IADC quantizer usually has low resolution, the loop output will be integers with small magnitude, making the MAC operations trivial.

In some applications, the decimation filter needs to suppress one or more interferers (e.g., line noise). This requires transmission zeros at the frequency of the interferer and its harmonics. The simple cascade-of-integrators decimation filter does not provide any

notches other than those at multiples of f_s/M. In this case, the design may be based on the sinc function, which can provide transmission zeros at arbitrary frequencies [6]. Figure 12.11 in Section 12.5.1 shows an example response from that work.

12.4 Multiple-Stage IADCs and Extended Counting ADCs

As was the case for $\Delta\Sigma$ ADCs, the SQNR of the IADC may be improved by a variety of changes: increasing the order L, or the oversampling ratio M, or the resolution of the internal quantizer, all raise the SQNR. However, these measures are all limited by practical effects. For wideband ADCs, the OSR M may be limited to a low value by the bandwidth of the amplifiers, or by the allowable power dissipation. For low oversampling ratios, the SQNR cannot be significantly improved by raising the order of the loop-filter, and high SQNR may only be obtained by using impractically high quantizer resolution.

The problems presented by low OSR may be solved by utilizing the multi-stage (MASH) architecture, discussed in Chapter 5. Here, the quantization error e_1 of the first stage is obtained in analog form, and canceled by the output of the second stage. Similarly, the error e_2 of the second stage can be canceled by the output of the third stage, and so on. The digital outputs of all stages are then combined using error-canceling filters $H_1, H_2, \cdots$. Thus, high-order noise shaping may be obtained, while using only low-order individual loops. In addition, if the first loop contains a multi-bit quantizer, the error e_1 will be smaller than the full-scale voltage of the circuitry. Then e_1 may be amplified by a gain $A > 1$ before entering it into the second stage, and an attenuation $1/A$ can also be applied to the second-stage output, which further reduces the final error.

While MASH was developed originally for $\Delta\Sigma$ DACs and ADCs, it is applicable to IADCs as well. Reference [3] describes a two-stage MASH IADC, where the second stage operates all the time, from the second clock period until period $(M+1)$. Many IADC stages may also be cascaded; [7] describes a 12-bit IADC containing eight stages, and operating at an oversampling ratio of only three!

An economical MASH IADC can be obtained by recalling from (12.8) that the total conversion error of the first loop after digital filtering is given by the scaled last quantization error $e[M]$ generated in the loop. In general, $e[M]$ needs to be obtained by subtracting the input of the first-stage quantizer from its output. However, for the low-distortion structure with a maximally flat $NTF(z)$, (12.2) shows that $e[M]$ can be found simply from the output $x_3[M]$ of the last integrator in the first loop. Hence, an efficient MASH IADC can use a second stage which is inactive until the clock period $(M-1)$, and then it converts and scales $x_3[M]$ while the output of the first stage is processed by the decimation filter. This second stage will thus produce the N_{LSB} least significant bits of the overall output word. The second stage may be realized by a Nyquist-rate ADC (e.g., a successive-approximation ADC), and the operation can be fully pipelined if $N_{LSB} < (M-1)$. Multi-stage IADCs based on the principle described in this paragraph [9],[12] are often called *extended-range* or *extended-counting* ADCs. As an example, Figure 12.5 shows the block diagram of the extended-counting ADC described in [12]. It used a low-distortion second-order IADC as the first stage, and a SAR ADC as the second stage. It achieved SNDR > 86 dB in a 0.5-MHz bandwidth.
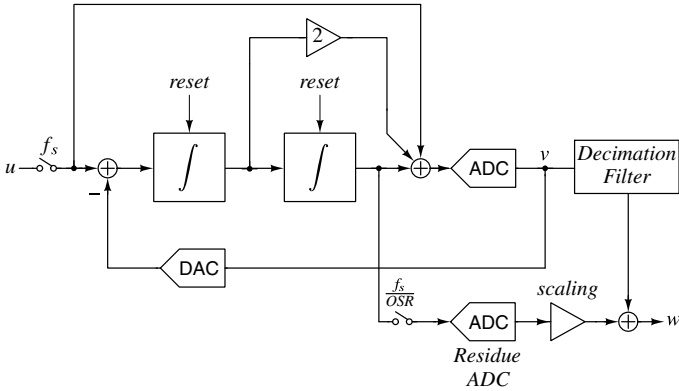
Figure 12.5   A 2-0 extended-counting ADC [12].

12.5   IADC Design Examples

12.5.1   Third-Order Single-Bit IADC

As an example for the design of a single-stage IADC, the 22-bit data converter described in [6] will be discussed. The block diagram of the noise-shaping loop was shown earlier in Figure 12.3; the switched-capacitor circuit implementing it is illustrated in Figure 12.6. In order to avoid dynamic as well as static nonideal effects introduced by a multi-bit DAC,
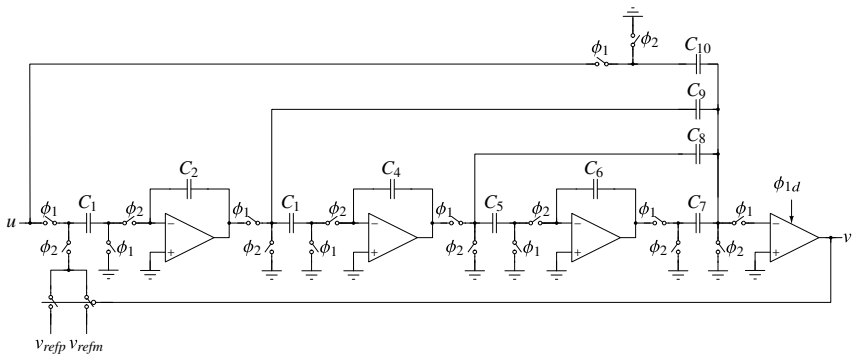

Figure 12.6   Single-ended switched-capacitor schematic of the system of Figure 12.3 [6].

single-bit quantization was used. The coefficients chosen were $a = [1.4\ 0.99\ 0.47]$, $b = 0.5674$, and $c = [0.5126\ 0.3171]$.

Figure 12.7 illustrates the mean square of the quantization noise as a function of the dc input signal u normalized to V_{ref} for $M = 1024$. As expected, when $|u|$ approaches V_{ref} the quantizer overloads, and the noise becomes large. However, there are no idle tones, since the reset operations prevent the occurrence of periodic signals with long periods, and the digital filter suppresses high-frequency tones.

In order to allow large input signals close to $|u| = V_{ref}$, the input stage includes an attenuator with a gain of 2/3. To make this gain factor accurate despite element mismatches,
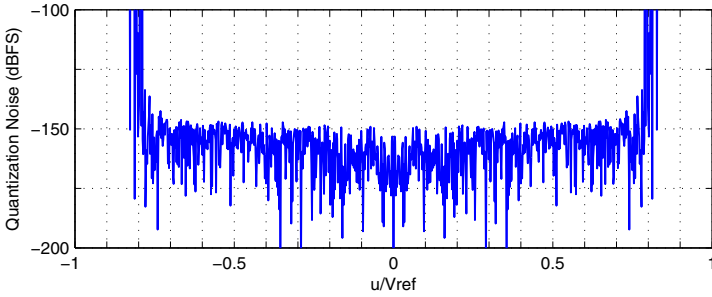
Figure 12.7 Quantization noise power as a function of u/V_{ref} for $M = 1024$.

a dynamic element matching scheme was used. The circuit is shown in Figure 12.8. In
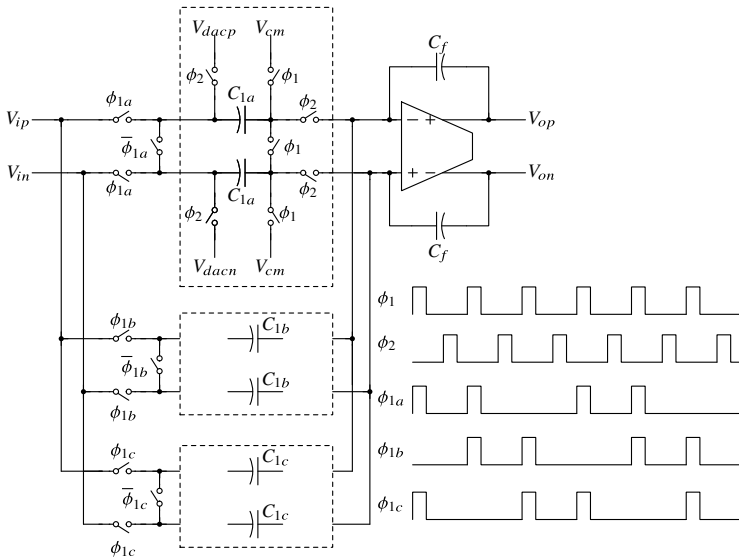
Figure 12.8 The rotating capacitor input circuit. The dotted frames contain replicas of the circuit containing C_{1a}.

a single clock period all six switched input capacitors deliver a charge proportional to the DAC output V_{dac}, but only four of them deliver the $C_1 \cdot u$ charge. This is equivalent to a scale factor of $2/3$ for u. By rotating the roles of the capacitors, the mismatch errors are converted into an out-of-band periodic noise.

To cancel offset, the device used an enhanced form of chopping, named *fractal sequencing*. Note that simple chopping is inadequate in the cascade-of-integrators circuit used here. To illustrate this, assume that a 1-mV offset exists at the input of the first integrator, and assume unity gain factors for all three integrator stages. Then the output sequence of the first integrator (in mV) will be $\{1, -1, 1, -1, \ldots\}$; the second stage output is $\{1, 0, 1, 0, \ldots\}$, and the third one $\{1, 1, 2, 2, 3, 3, \ldots\}$ – diverging with time. In fractal sequencing, the control signal INV ensures that the input signal is always integrated with the

same sign, while the input offset is toggled in such a way that after M oversampled cycles the offset at the output of the last integrator is canceled.

The sequence of simple chopping $\{+ - + - \ldots\}$ is $S_1 = (+-)$. Here, a $+$ denotes no inversion of the signal, while $-$ denotes inversion, and the parentheses indicate that this pattern repeats indefinitely. The method of creating the fractal sequences S_k is based on the recursion relation $S_{k+1} = [S_k, -S_k]$. Thus, from the simple chopping sequence $S_1 = (+-)$, higher-order sequences can be obtained as shown in (12.17) below. The desired sequence for the IADC is S_L, where L is the number of the cascaded integrators. $L = 3$ in this device.

$$
\begin{aligned}
S_1 &= (+-), \\
S_2 &= [S_1, -S_1] = (+ - -+), \\
&\;\;\vdots \\
S_{k+1} &= [S_k, -S_k].
\end{aligned}
\tag{12.17}
$$
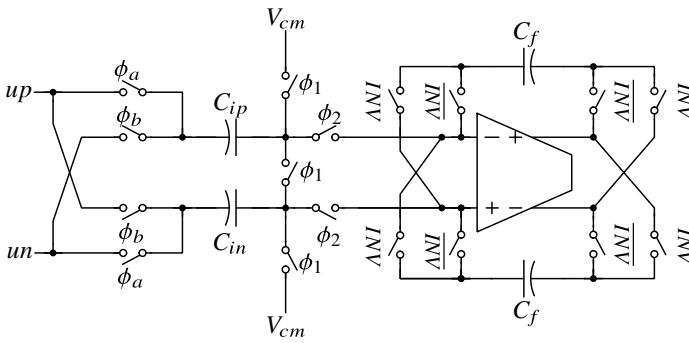
Figure 12.9   Offset compensation using fractal sequencing.

Figure 12.9 shows the input integrator using fractal sequencing. The switches INV and $\overline{\text{INV}}$ are operated by the S_3 fractal sequence. To maintain a consistent integration polarity, $\phi_a = \phi_1$ and $\phi_b = \phi_2$ when INV is low, but $\phi_a = \phi_2$ and $\phi_b = \phi_1$ when INV is high. Note that the chopping frequency used in the fractal sequence may be a subharmonic of the IADC clock. Figure 12.10 shows the normalized integrator output voltages after fractal sequencing using $f_{chop} = f_s/64$.

The digital filter used a modified sinc transfer function

$$
H(z) = \prod_{i=1}^{4} \frac{1 - z^{-M_i}}{M_i(1 - z^{-1})},
\tag{12.18}
$$

where $M_i = \{512, 512, 512 - 2^6, 512 + 2^6\}$, which provided wide notches around the line frequency (Figure 12.11). These notches suppress line-frequency noise even in the presence of clock-frequency or line-frequency variations.
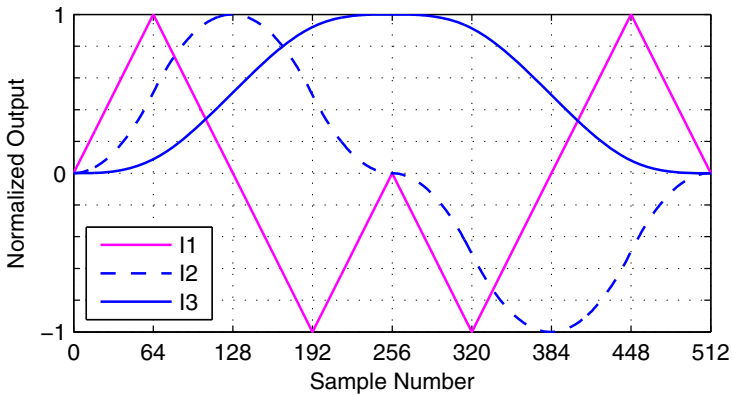
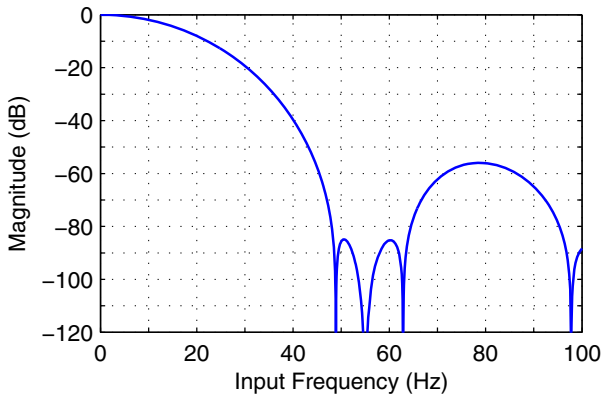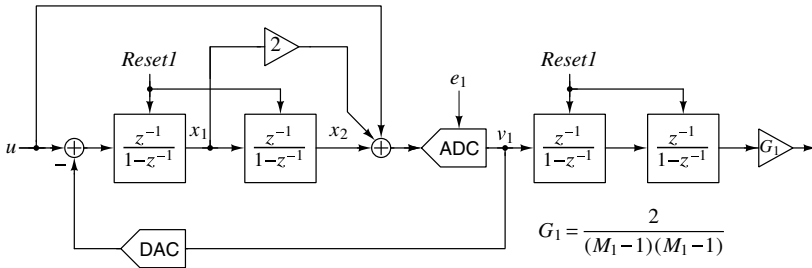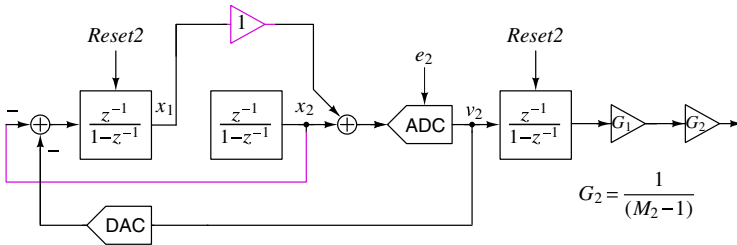**Figure 12.10**   Integrator output voltages after fractal sequencing.

Figure 12.11   Decimation filter gain response.

12.5.2 Two-Step IADC

Step 1: M_1 clock periods

Step 2: M_2 clock periods

Figure 12.12   Block diagram of the two-step IADC.

Our second example is a two-stage two-step IADC. This incremental ADC was proposed for low-bandwidth, micro-power sensor interface circuits. The two-step operation extends the order of a conventional IADC from N to $(2N - 1)$, while requiring only the circuitry of an Nth-order IADC. Figure 12.12 shows the block diagrams of the circuit during the two steps. In step 1, the circuit is a second-order feedforward IADC for M_1 clock periods. At the end of this step, the second integrator holds $x_2[M_1]$, which is the analog form of the quantization error. In step 2, the circuit is reconfigured. The second integrator now acts as a S/H input stage, and the rest of the circuit becomes a first-order converter IADC1, which converts $x_2[M_1]$ into digital form.

Figure 12.13 illustrates a simplified switched-capacitor implementation of the circuit during step 1. For a specified total number of clock periods $M = M_1 + M_2$, it is easy to show[1] that the optimum allocation for quantization noise is $M_1 = (2/3)M$ and $M_2 = (1/3)M$. In the implemented IADC, $M = 192$ was used, and hence $M_1 = 128$ and $M_2 = 64$.

The implemented third-order IADC achieved a measured dynamic range of 99.8 dB and an SNDR of 91 dB for a 2.2-V_{pp} input and 250-Hz bandwidth. Fabricated in 65-nm CMOS, the IADC's core area was 0.2 mm^2, and consumed only 10.7 μW. The FOMs were 0.76 pJ/conversion step and 173.5 dB, both among the best reported results.

Figure 12.14 compares the SQNR vs. OSR characteristics of the implemented two-step IADC2 with a single-step IADC2 and an IADC3. For the same total number of clock periods, the two-step circuit is nearly as accurate as the IADC3, but needs one less opamp.

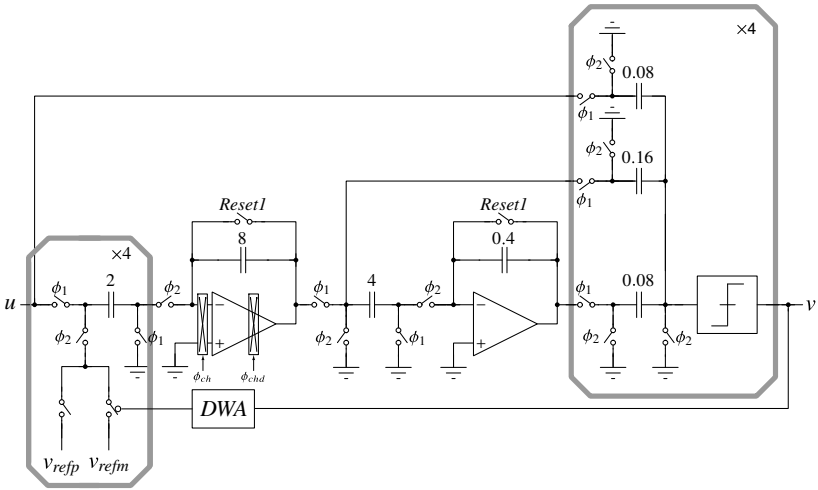[1]Minimize $1/M_1^2 + 1/M_2$ subject to $M_1 + M_2 = M$.

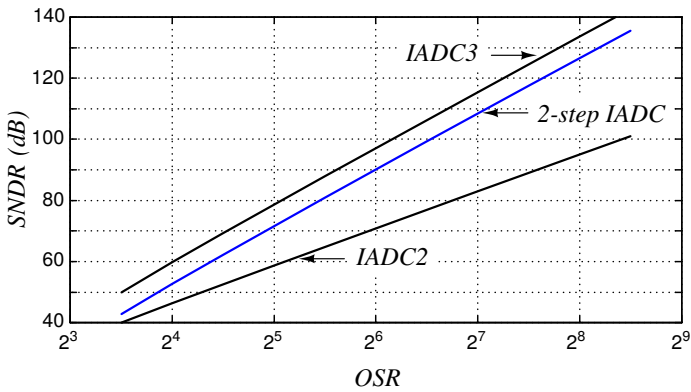**Figure 12.13**   Circuit diagram of the two-step IADC during step 1.

Figure 12.14   SQNR vs. OSR characteristics for IADCs.

In general, the two-step operation allows the approximate realization of a $(2N-1)$th-order IADC with only N amplifiers.
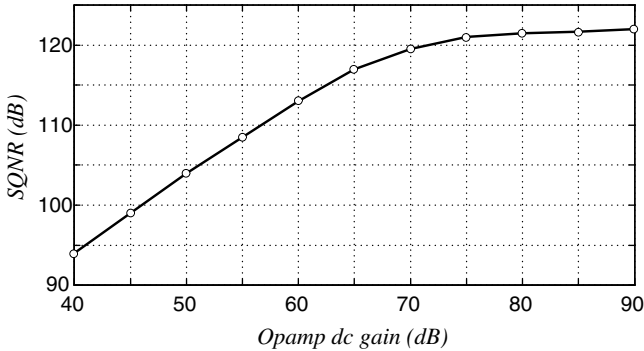
Figure 12.15   SQNR versus opamp gain for a two-step IADC.

As Figure 12.15 demonstrates, the operation is not overly sensitive to the dc gain of the amplifiers used. Last, Figure 12.16 shows the SNR and SNDR as a function of the amplitude of the input signal.
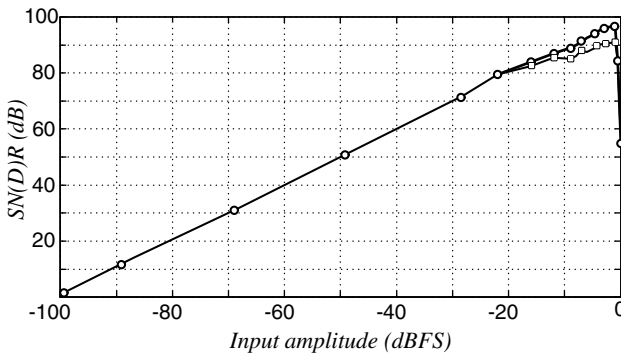
Figure 12.16   SNR and SNDR versus input signal amplitude for the two-step IADC.

12.6   Conclusions

By periodically resetting all memory elements of a delta-sigma ADC, it can be converted into a Nyquist-rate converter. The result is the incremental A/D converter (IADC). The number of clock periods between resets determine the oversampling ratio. Compared to the delta-sigma ADC, the IADC provides lower SNR, but it is easy to multiplex, has lower latency, and needs a much simpler digital filter. It is also less vulnerable to idle tones and instability. For these reasons, the IADC is often the best choice for sensor interface applications.

As with the delta-sigma ADC, the IADC can be realized in multi-stage and multi-step structures, which can provide highly efficient realizations for micro-power interfaces.

References

[1] R. J. van de Plassche, "A sigma-delta modulator as an A/D converter," *IEEE Transactions on Circuits and Systems*, vol. 25, no. 7, pp. 510–514, July 1978.

[2] J. Robert, G. C. Temes, V. Valence, R. Dessoulavy, and P. Deval, "A 16-bit low voltage A/D converter," *IEEE Journal of Solid-State Circuits*, vol. 22, no. 2, pp. 157–163, April 1987.

[3] J. Robert and P. Deval, "A second-order high-resolution incremental A/D converter with offset and charge injection compensation," *IEEE Journal of Solid-State Circuits*, vol. 23, no. 3, pp. 736–741, March 1988.

[4] J. Márkus, "Higher-order incremental delta-sigma analog-to-digital converters," Ph.D. thesis, Budapest University of Technology and Economics, 1999.

[5] J. Márkus, J. Silva, and G. C. Temes, "Theory and applications of incremental delta sigma converters, " *IEEE Transactions on Circuits and Systems-I*, vol. 51, no. 4, pp. 678–690, April 2004.

[6] V. Quiquempoix, P. Deval, A. Barreto, G. Bellini, J. Márkus, J. Silva, and G. C. Temes, "A low-power 22-bit incremental ADC," *IEEE Journal of Solid-State Circuits*, vol. 41, no. 7, pp. 1562–1571, July 2006.

[7] T. C. Caldwell, "Delta-sigma modulators with low oversampling ratios," Ph.D. thesis, University of Toronto, 2010.

[8] J. Steensgaard, Z. Zhang, W. Yu, A. Sárhegyi, L. Lucchese, D. I. Kim, and G. C. Temes, "Noise-power optimization of incremental data converters," *IEEE Transactions on Circuits and Systems I*, vol. 55, no. 5, pp. 1289–1296, June 2008.

[9] R. Harjani and T. A. Lee, "FRC: A method for extending the resolution of Nyquist-rate converters using oversampling," *IEEE Transactions on Circuits and Systems-II*, vol. 45, no. 4, pp. 482–494, April 1998.

[10] P. Rombouts, W. de Wilde, and L. Weyten, "A 13.5-b 1.2-V micropower extended counting A/D converter," *IEEE Journal of Solid-State Circuits*, vol. 36, no. 2, pp. 176–183, Feb. 2001.

[11] J. De Maeyer, P. Rombouts, and L. Weyten, "A double-sampling extended-counting ADC," *IEEE Journal of Solid-State Circuits*, vol. 39, pp. 411–418, March 2004.

[12] A. Agah, K. Vleugels, P. B. Griffin, M. Ronaghi, J. D. Plummer, and B. A. Wooley, "A high-resolution low-power incremental $\Sigma\Delta$ ADC with extended range for biosensor arrays," *IEEE Journal of Solid-State Circuits*, vol. 45 , pp. 1099–1110, June 2010.

[13] W. Yu, M. Aslan, and G. C. Temes, "82 dB SNDR 20-channel incremental ADC with optimal decimation filter and digital correction," *IEEE Custom Integrated Circuits Conference*, pp. 1–4, Sept. 21, 2010.

[14] C.-H. Chen, J. Crop, J. Chae, P. Chiang, and G. C. Temes, "A 12-Bit, 7 μW/channel, 1 kHz/channel incremental ADC for biosensor interface circuits," *IEEE International Circuits and Systems Symposium*, May 2012.

[15] C.-H. Chen, Y. Zhang, T. He, P. Y. Chiang, and G. C. Temes, "A micro-power two-step incremental analog-to-digital converter," *IEEE Journal of Solid-State Circuits*, vol. 50, no. 8, pp. 1796–1808, Aug. 2015.

CHAPTER 13

DELTA-SIGMA DACS

Up to this point, the concepts of oversampling and noise-shaping were only applied to ADCs, and not to digital-to-analog converters (DACs). In fact, delta-sigma DACs are commercially as important as their ADC counterparts, if not more so, and their implementation is often just as difficult as the implementation of a $\Delta\Sigma$ ADC. This chapter will be devoted to the specific issues involved in the design of delta-sigma DACs.

The motivation to use noise-shaping in D-to-A conversion is the same as in A-to-D conversion. For a DAC with a 3-V full-scale and an 18-bit resolution, the LSB voltage is only about $12\,\mu$V. Hence, the permissible deviation of the DAC output levels from their ideal values is of the order of $12\,\mu$V, which cannot be achieved in a conventional DAC without expensive trimming and/or extremely long conversion time. Therefore, the trade-off earlier discussed in connection with delta-sigma ADCs, wherein oversampling and additional digital hardware are applied to allow the use of robust and simple analog circuitry, is attractive for high-accuracy DACs as well. The actual structures implementing this trade-off will be discussed next.

## 13.1   System Architectures for Delta-Sigma DACs

Figure 13.1 illustrates the basic system diagram of the $\Delta\Sigma$ DAC. As indicated, the front end (containing a digital interpolation filter and a noise-shaping loop) contains digital circuitry, while the output stages (the internal DAC and the reconstruction filter) are analog.

IF: Interpolation filter   DAC: Digital-to-Analog Converter

NL: Noise-shaping Loop   LPF: Lowpass Filter
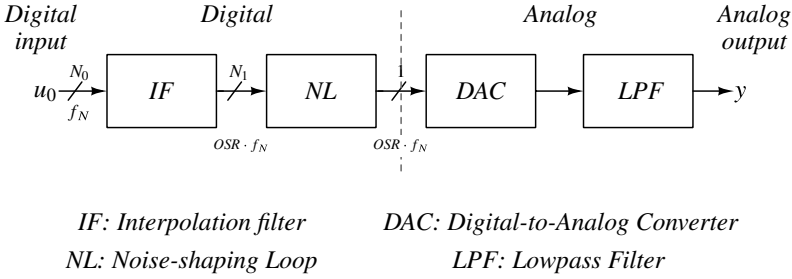
Figure 13.1   Block diagram of a $\Delta\Sigma$ DAC.

The spectra of the signals processed by the system are shown in Figure 13.2. The
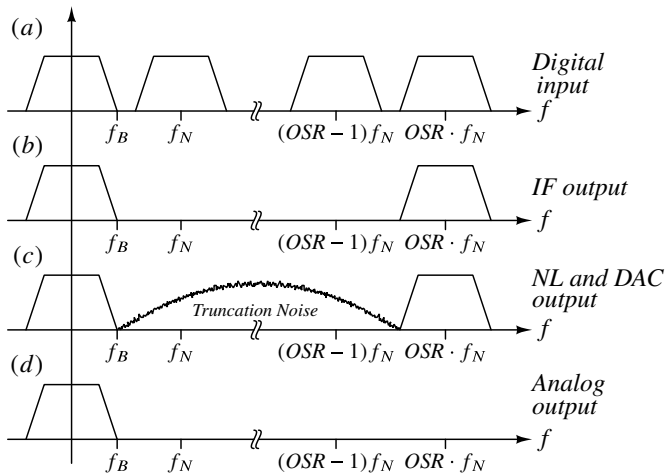
Figure 13.2   Signal and noise spectra in a $\Delta\Sigma$ DAC.

input signal $u_0[n]$ is a multi-bit data stream with a word length N_0 (typically, 15–24 bits) sampled near the Nyquist rate f_N. Its spectrum is illustrated in Figure 13.2(a).

The interpolation filter (IF) has two roles:

- to raise the sampling frequency to $OSR \cdot N$ and thereby allow subsequent noise-shaping, and

- to suppress the spectral replicas centered at $f_N, 2f_N, \cdots, (OSR-1)f_N$.

The purpose of this sideband suppression is to reduce digitally the out-of-band power of the input of the noise-shaping loop without affecting the baseband signal spectrum. This improves the dynamic range of the noise-shaping loop, since larger signals can thus be accommodated. Also the task of the analog output filter is eased, since it needs to suppress less out-of-band noise. Thus, the filter's linearity requirements can be somewhat more relaxed due to the reduced amount of intermodulated out-of-band noise folding down into the signal band. The suppression need not be very accurate, since the truncation error

generated in the noise-shaping loop will introduce unwanted noise in the same frequency range anyway. The ideal spectrum of the IF output signal is shown in Figure 13.2(b). The word length of this signal can remain about the same as that of the input data $u_0[n]$.

The noise-shaping loop reduces the word length of its input signal to a few (1–6) bits. If a single-bit NL output is used, then (as discussed in Chapter 2 for the internal DAC in a $\Delta\Sigma$ ADC) the linearity requirements for the DAC following the NL can be relaxed. If the output data are multi-bit, then the techniques discussed in Chapter 6 may be utilized to filter out or cancel the unavoidable DAC nonlinearity errors, and thus to achieve linear conversion. (The pros and cons of using multi-bit DAC loops will be discussed in Section 13.3). In any case, the NL output must contain a faithful reproduction of the input signal $u_0[n]$ in the baseband, but it will also include the filtered truncation noise caused by the reduction of the word length in the loop. The spectrum of the NL output signal is schematically illustrated in Figure 13.2(c).

The next block in the system is the embedded DAC. As discussed above, it may have a single-bit input, in which case its output will be a two-level analog signal. The structure of such a 1-bit DAC will be very simple, and its linearity will be theoretically perfect (although some practical precautions need to be observed to achieve good linearity). However, the high slew rate of the single-bit DAC output signal, and the large amount of out-of-band noise power it contains, make the design of the subsequent analog smoothing filter (LPF) a difficult task.

By contrast, for a multi-bit DAC, additional circuitry is required for the filtering or cancellation of the DAC nonlinearity error, which results in a more complex DAC. However, the reduced slew rate and out-of-band noise power of the DAC output signal allows reduced performance requirements, and hence simpler implementation, for the smoothing filter. Usually, the overall trade-off in complexity, chip area and power dissipation favors the multi-bit structure.

Ideally, the DAC will reproduce the digital signal at its input in an analog form without any distortion. Hence, the output spectrum of the DAC will be, except for a constant factor corresponding to the reference voltage or current of the DAC (and for a $sinc(fT_s)$ frequency-dependent factor corresponding to the frequency response of a zero-order hold), the same as that shown in Figure 13.2(c) for the output signal of the noise-shaping loop.

Finally, the task of the analog smoothing or reconstruction filter is to suppress most of the out-of-band noise power contained in its input signal. Hence, the ideal spectrum of its output signal should be as shown in Figure 13.2(d). As already mentioned, it is relatively easy to achieve good noise suppression without introducing additional distortion for a multi-bit DAC output signal, but the task is usually quite difficult for a single-bit signal. The design of the analog post-filter will be discussed in Section 13.5.2.

13.2   Loop Configurations for Delta-Sigma DACs

As was the case the $\Delta\Sigma$ ADCs, there is also a wide variety of loop architectures available for the designer of $\Delta\Sigma$ DACs. The function of the loop is similar to that of the noise-shaping loop of the $\Delta\Sigma$ ADC, namely to reduce the resolution of the input signal[1] to a

---

[1]For an analog signal, the resolution can be regarded as infinite.

few bits without significantly affecting its in-band spectrum in the process. Since the reduced word length means that quantization or truncation error is introduced, the loop must suppress the power spectrum of this added noise in the signal band. The only significant differences between ADC and DAC loops are as follows. In the DAC loop, all signals are digital, and hence, no internal data conversion is required. For the same reason, the signal processing in the loop can be made highly accurate, and we need not take any analog imperfections into account when predicting the actual behavior of the loop. As we will see, this permits the use of some efficient configurations that are impractical for ADC loops. Some typical loop configurations will be discussed next.

13.2.1 Single-Stage Delta-Sigma Loops

All loop architectures discussed for ADCs in Section 4.7 remain applicable for delta-sigma DACs. Thus, the structure containing cascaded integrators with distributed feedback and input coupling (CIFB), illustrated in Figure 4.26; the circuit using cascaded resonators with distributed feedback (CRFB), shown in Figure 4.27; as well as the structures containing cascaded integrators or resonators with feedforward coupling (CIFF), shown in Figures 4.28 and 4.30, respectively, can also be used in delta-sigma DAC loops. Of course, the component blocks are now accumulators rather than integrators, and they are implemented by digital adders and multipliers, rather than opamps, capacitors and switches as in the ADC loops.

The designer is still faced with some of the same problems (e.g., stability issues) as were encountered for analog loops, and also with some new ones. In finding the proper configuration and order for the loop, and calculating the coefficients needed, the noise-shaping and signal transfer specifications must be satisfied, and stability needs to be ascertained under all anticipated conditions. Also, the conditions for optimum dynamic range must be met, and the over- or underflow of any block avoided. Finally, the word lengths of all coefficients and operations should be carefully determined so that, on the one hand, the required accuracy in signal transfer and noise suppression is maintained, and, on the other hand, the complexity of the circuit is minimized subject to these accuracy conditions.

Qualitatively, the sensitivity considerations discussed for ADC loops remain valid for DAC loops, except that the errors generated here are due to coefficient truncations and to the round-off errors of the digital operations (additions and multiplications), rather than element-matching errors and finite opamp-gain effects. Thus, the coefficient and round-off errors must be kept small in all signal paths connecting to the input node, but they may increase progressively as the signal propagates towards the output of the loop. Hence, the word length needed may vary considerably with the location of the block in the loop.

Hardware can also be saved by choosing simple coefficients that contain only a few terms, each term an integer power of 2. This may alter the signal and noise transfer functions slightly, but the effects on the NTF and STF are usually small and, in the case of the STF, can often be corrected by the blocks preceding or following the loop. The signal transfer function of the low-distortion architecture discussed in Chapter 4 tends to be less susceptible to coefficient truncation than that of competing architectures.

13.2.2   The Error-Feedback Structure

A configuration that is not practical for ADC loops, but is highly efficient for DACs, is illustrated in Figure 13.3 for a 1-bit loop. Here, rather than feeding back the MSB retained
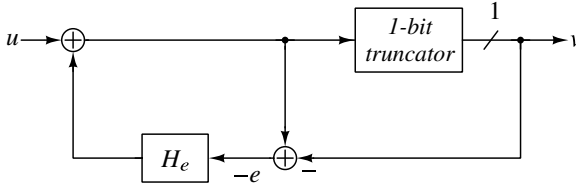

Figure 13.3   Error-feedback structure.

in the output signal as was done in the delta-sigma loop discussed in Section 13.2.1, the discarded LSBs (representing the truncation error $e[n]$) are filtered and fed back to the input. The loop-filter H_e used to filter $e[n]$ is now located in the feedback path.

In an ADC loop, this structure would be overly sensitive to the imperfections of the analog loop-filter and of the analog subtraction needed to generate $e[n]$, since the errors generated in either enter the input terminal directly. Hence, this architecture is never used in an ADC. In a digital realization, however, if sufficient accuracy is used in the digital implementation of the H_e filter, the circuit will perform well. Linear analysis shows that the output is given by

$$V(z) = U(z) + [1 - H_e(z)]E(z). \tag{13.1}$$

Hence, the STF is 1, and the NTF equals $[1 - H_e(z)]$.
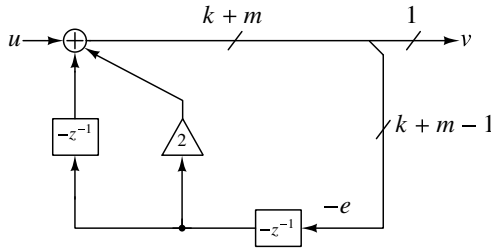
Figure 13.4   Second-order error-feedback noise-shaping loop.

For low-order loops, the error-feedback loop can usually be very simply realized. For a first-order loop, $NTF = 1 - z^{-1}$, and hence $H_e(z) = z^{-1}$, meaning, it is simply a delay. For a second-order loop with a double zero of the NTF at dc,

$$H_e(z) = 1 - (1 - z^{-1})^2 = z^{-1}(2 - z^{-1}). \tag{13.2}$$

Thus, the loop can be realized from two delays, a shift in the binary point to implement the factor 2, and two adders (Figure 13.4).

Higher-order error-feedback loops can, of course, also readily be designed, subject to stability considerations. As in the delta-sigma type loop, instability causes the input signal $y[n]$ of the quantizer (here, the truncator) to grow beyond the operating range of the
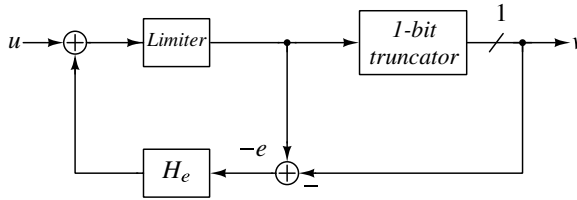
Figure 13.5 Error-feedback with limiter.

digital logic. Depending on the arithmetic used, this may just cause the saturation of $y[n]$ at its largest possible value, or it may cause a wraparound, where the output $v[n]$ suddenly decreases with increasing $y[n]$ at overflow. While saturation is usually acceptable, wraparound causes large errors, and hence, it must be prevented, e.g., by including a digital limiter in the loop (Figure 13.5) at the input of the truncator [1]. The limiter should saturate before overflow can occur.

13.2.3 Cascade (MASH) Structures

To achieve high-order noise-shaping without the stability problems inherent in the design of higher-order loops, cascade structures may be used for $\Delta\Sigma$ DACs as well as for $\Delta\Sigma$ ADCs [2]. (Cascaded DAC stages were, in fact, proposed before ADC ones.) Figure 13.6 illustrates the architecture of a two-stage cascade DAC. In a typical structure,
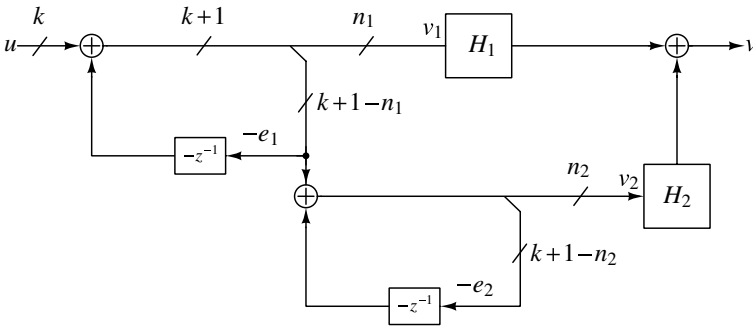

Figure 13.6 Cascade structure for a second-order noise-shaping loop.

both stages may contain second-order loop-filters, resulting in a fourth-order noise-shaping overall, while preserving the robust stability properties of second-order loops.

A design issue, one that did not occur for MASH ADCs but appears for cascade DACs, concerns the optimum location of the internal DACs in the structure. Assume, at first, that all signal processing in the structure of Figure 13.6 is performed digitally. As explained in the discussions of Section 5.2, the post-filter H_1 usually replicates the signal transfer function STF_2 of the second stage. Often, STF_2 is simply a single or multiple delay, and hence, H_1 can easily be implemented digitally without increasing the word length n_1 of the first stage output v_1. By contrast, H_2 usually reproduces the noise transfer function of the first stage, and hence, if implemented digitally, it increases the word length n_2 of v_2. Adding $H_1 \cdot V_1$ and $H_2 \cdot V_2$ digitally will increase the output word length even

further. Hence, such a structure will produce a multi-bit output $v[n]$, which then needs to be accurately converted in a multi-bit, and hence complex, internal DAC.

An alternative is to use a separate DAC in each stage, and combine their outputs using analog circuitry, as illustrated schematically in Figure 13.7. This allows the use of less
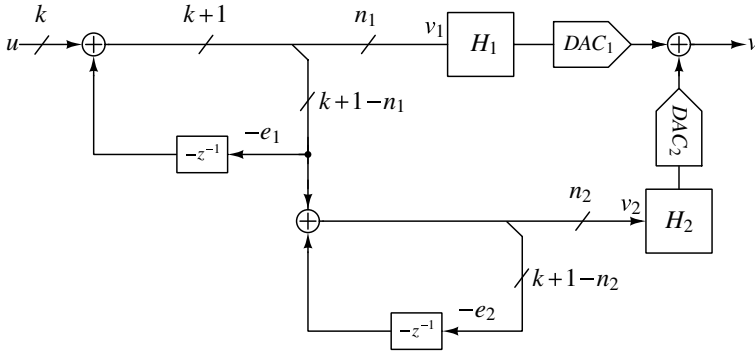
Figure 13.7   A cascade DAC using analog recombination.

complicated DACs. A mismatch between the gains of the two paths caused by analog errors will introduce leakage of the first-stage truncation error, but the mismatch will not affect the linearity of the signal conversion, which is limited only by the linearity of the first-stage DAC.

It is also possible to place DAC_2 ahead of the H_2 filter, which then must be realized by analog circuitry. This has two advantages: first, the resolution of DAC_2 can be reduced, since it only needs to convert V_2, not $H_2 \cdot V_2$, which has longer words. In fact, for $n_1 = n_2 = 1$, both DACs can be single-bit ones. For multi-bit DACs, H_2 is now going to shape the noise introduced by the inherent nonlinearity of DAC_2, along with the truncation noise of the second stage. A disadvantage of this modified scheme is that the analog implementation of H_2 cannot reproduce the digital NTF_1 as accurately as a digital filter could. (Note, however, that any zeros of H_2 at dc can be very accurately realized in an analog circuit, by having series capacitors in the signal path.)

Another option is to split the H_2 block into a digital stage preceding DAC_2, and an analog one following it. This way, the larger truncation noise receives full shaping by NTF_2 and H_2, and the much smaller noise caused by DAC_2 errors will be shaped only by the analog part of H_2. This scheme will realize the required replica of NTF_1 more accurately than a fully analog H_2 can.

13.3   Delta-Sigma DACs Using Multi-Bit Internal DACs

The parameters of the digital noise-shaping loops used in $\Delta\Sigma$ DACs are much more accurately controlled than those of the analog loops required in $\Delta\Sigma$ ADCs, and some of the basic arguments for using multi-bit quantization in ADCs (such as the ones based on opamp slew rate, power dissipation, nonlinearity, clock jitter, etc.) are not valid for DAC loops. Nevertheless, the stability considerations presented in Section 4.1 remain valid, and an additional powerful reason for multi-bit operation emerges: the relaxed requirements

on the analog smoothing filter LPF following the internal DAC. For a single-bit DAC, the input signal of this filter is a two-level fast-slewing analog signal (usually voltage), with most of its power contained in the large high-frequency quantization noise. This fast signal needs to be filtered by the lowpass filter so that almost all its high-frequency noise is removed. This must be accomplished without distorting the signal or even the out-of-band noise. (Distorting the noise will cause the folding down of the large noise spectrum from the region around $f_s/2$ into the signal band.) In addition, due to the steep slopes of the two-level analog signal, any clock jitter is translated into substantial amplitude noise at the output of the filter. In conclusion, the analog problems due to single-bit truncation which appeared in the noise-shaping loop in $\Delta\Sigma$ ADCs do not disappear in $\Delta\Sigma$ DACs; they are just shifted to the analog post-filter!

In earlier single-bit implementations [3], to overcome these difficulties, the smoothing filter was realized as a cascade combination of a high-order switched-capacitor (SC) filter, an SC buffer stage, and a continuous-time post filter. It required a considerable chip area and dc power. The motivation for paying such a high price for the one-bit system was to avoid the inherent nonlinearities of a multi-bit internal DAC.

In recent years, various techniques (dual quantization, mismatch-error shaping, and digital correction) have become available for reducing the DAC nonlinearity effects, and hence multi-bit DAC structures are being favored over single-bit ones. These DAC linearization methods are similar to their counterparts used in ADCs, which were discussed in Chapter 6. In the next section, these schemes will be examined.

13.3.1 Dual-Truncation DAC Structures

The general principle of dual-truncation DACs is similar to that of dual-truncation ADCs: use single-bit truncation in the D/A conversion of the signal, and use multi-bit truncation where only the truncation errors are converted. A simple implementation, which is similar to the Leslie–Singh structure of Section 4.5.1, is shown in Figure 13.8 [4]. As shown,
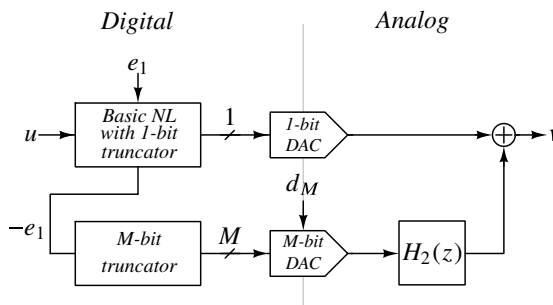
Figure 13.8   A dual-truncation DAC system.

the signal $u[n]$ is reduced to a single-bit data stream in a noise-shaping loop. This can be converted linearly in a 1-bit DAC. The large truncation error $-e_1$ is truncated to M bits ($M > 1$), and converted in an M-bit internal DAC. It is then filtered and added to the output of the 1-bit DAC to cancel $e_1[n]$. The spectrum of the nonlinearity error d_M of the M-bit DAC is shaped by the analog filter $H_2(z)$, which duplicates the NTF of the 1-bit loop and suppresses its in-band power.
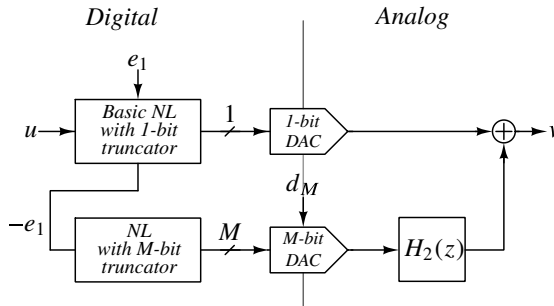
Figure 13.9   A dual-truncation MASH structure.

A more sophisticated and effective structure, shown in Figure 13.9, uses a noise-shaping loop in both stages, with 1-bit truncation in the first stage and M-bit truncation in the second one.
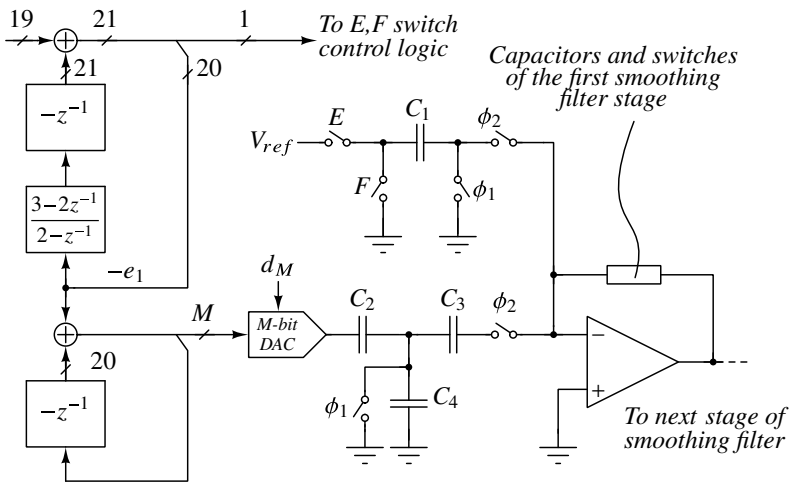

Figure 13.10   A third-order dual-truncation MASH noise-shaping stage.

Figure 13.10 shows the implementation of a third-order DAC [4] based on this structure. In the diagram, the switched-capacitor branch containing C_1 realizes the 1-bit DAC, while C_2, C_3, C_4, and their switches perform as the analog filter $H_2(z)$. Both loops use error-feedback; the first loop has a second-order loop filter with a pole at $z = 0.5$ for improved stability, while the second loop uses a simple first-order filter.

It is also possible to realize a single-stage dual-truncation DAC (Figure 13.11). This is similar to the Hairapetian ADC structure [5]. The single-bit output is fed to a 1-bit DAC, and is also fed back to all but the last stage in a cascade of integrators. An M-bit output is also generated; it is converted in a multi-bit DAC, and also entered into the last integrator. The input signal in this structure is converted by the 1-bit DAC in a potentially linear fashion, while the M-bit circuitry is used to cancel the large 1-bit truncation error in the output $v[n]$, and replace it with a much smaller M-bit error. An error cancellation logic, consisting of the analog filters H_4 and H_5, carries out this operation. Mismatch
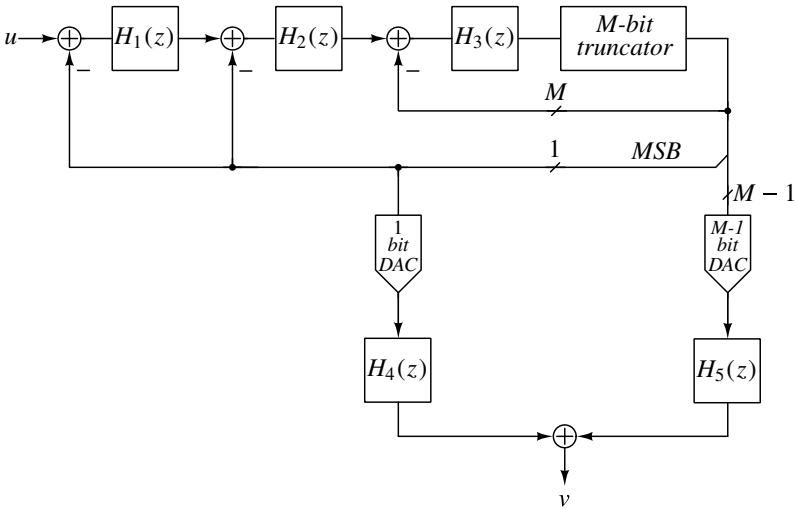
Figure 13.11   A single-stage dual-truncation D/A loop.

between the gains of these filters will degrade the cancellation of the 1-bit truncation error and hence the SNR, but will not introduce nonlinear signal distortion.

13.3.2 Multi-bit Delta-Sigma DACs with Mismatch Error Shaping

As mentioned earlier, the mismatch-error-shaping techniques discussed in Chapter 6 (data-weighted averaging, individual level averaging, vector-based mismatch shaping, tree-structure element selection) remain applicable to the internal DACs in multi-bit D/A converters. Again, however, there are new possibilities and trade-offs to consider for $\Delta\Sigma$ DACs.

In a multi-bit $\Delta\Sigma$ ADC, the number of bits N used in the output is generally limited to about 4, since for $N = 5$ the internal ADC already needs 32 comparators with associated circuitry, and hence requires substantial supply power and chip area. For $N = 2$ to 4, the complexity of the DAC itself, and that of digital circuitry implementing the necessary mismatch shaping, are both relatively low, and no special schemes are needed to simplify them.

By contrast, in a multi-bit $\Delta\Sigma$ DAC, no internal ADC is required, and hence, values of N higher than 4 may be chosen. However, since the complexity of the DAC and its error-correction circuitry grow exponentially with N, they may then require too much chip area and bias power. We may use 2^N as a complexity index; generally, for $N > 4$, its value is impractically high. This problem, and its solution, will next be discussed in terms of a second-order 6-bit delta-sigma DAC.

An obvious solution to having too many bits (here, 6 bits) in the DAC input signal is to use segmentation, that is, to split the 6-bit input data stream into two 3-bit segments: an MSB signal and an LSB one. The two segments can then be separately encoded into thermometer-coded words, scrambled, and converted into analog signals (Figure 13.12).
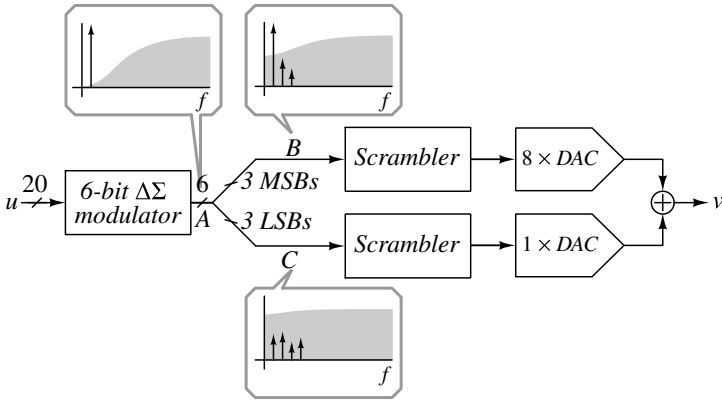
Figure 13.12   Segmentation.

The weighted sum of the two analog outputs provides the overall output signal. The effective complexity index of this system is $2 \cdot 2^3 = 16$, which is lower (by a factor of 4) than that of a direct realization of the 6-bit DAC, which is $2^6 = 64$.

The problem with this approach is that both the MSB and LSB segments contain large distortion components, which ideally cancel if the two are recombined exactly, with the weight factor 8 needed to scale the MSB analog output. However, if this factor is inaccurate, then the unfiltered quantization noise and distortion components contained in the MSB and LSB outputs will not cancel perfectly, and this will significantly degrade the linearity and SNR performance. This degradation will occur even if the scramblers in both paths use mismatch shaping, since the noise is already contained in the scrambler inputs B and C, and not generated by the internal DACs.

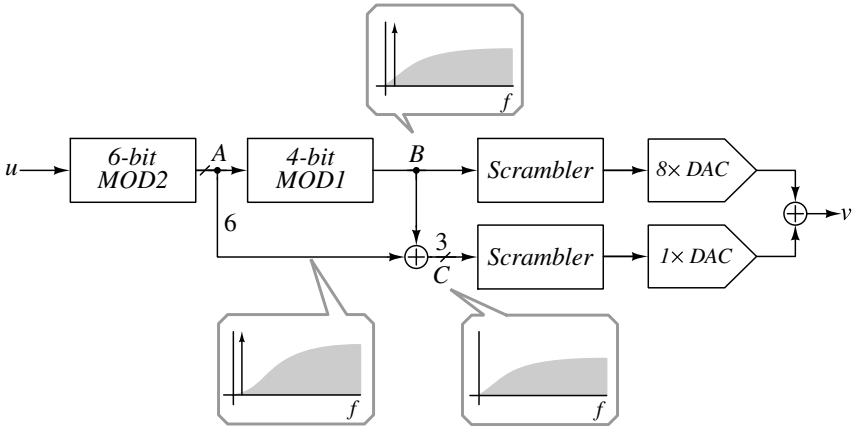**Figure 13.13**   Noise-shaped segmentation.

A way to overcome this accuracy problem is illustrated in Figure13.13 [6]. An additional first-order $\Delta\Sigma$ loop is cascaded with the main modulator, and it compresses the word length of its 6-bit input A into 4 bits. Denoting the NTF of MOD1 by H_1, and its quantization error by E_1, the two segmented signals are then the 4-bit output $B = A + H_1 E_1$ of

the first-order loop, and $C = -H_1E_1$, which is the negative of its shaped 3-bit quantization error. C is generated by subtracting the input A of MOD1 from its output B. Both signals B and C are next thermometer-coded, scrambled and D/A converted, and then added, using a scale factor 4 for B to make up for the shift of the binary point when C is thermometer coded. Ideally, the analog output is therefore $B + C = A$, as required. The complexity index is $2^4 + 2^3 = 24$, which is still much lower than the value $2^6 = 64$ associated with the unsegmented system.

In the system of Figure 13.13, both B and C are noise-shaped signals, and hence, if there is an error in the analog scale factor 4, so that C is not completely canceled, the resulting output error will be only some additional shaped noise. For sufficiently high oversampling ratio (e.g. 128), a 1% DAC element matching error will still allow a 110 dB SNR [6].
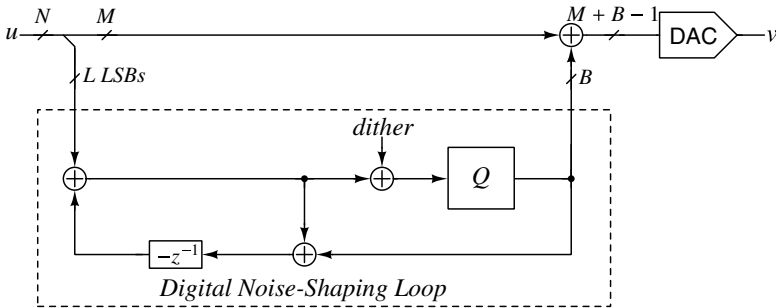

Figure 13.14   A hardware-reduced first-order modulator with dither.

Another segmentation scheme for $\Delta\Sigma$ DACs is illustrated in Figure 13.14 [7]. Here, the L LSBs of the input data stream are compressed to shorter (B-bit, $B < L$) words by an error-feedback noise-shaping loop, and fed to a digital adder. The M MSBs, by contrast, are directly entered into the adder. Since the addition is digital, it can be highly accurate. For a 6-bit input, the 4 LSBs may be compressed into 2 bits, and combined with the 2 MSBs to result in a 4-bit DAC circuit. For sufficiently large OSR, the accuracy can be satisfactory. (Note that this system is just a first-order modulator; it does not split the data into an MSB stream plus a noise-shaped LSB stream like the system of Figure 13.13 does.)

13.3.3 Digital Correction of Multi-Bit Delta-Sigma DACs
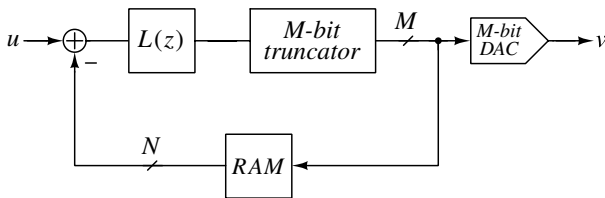

Figure 13.15   A digitally-corrected M-bit DAC.

As mentioned in Section 6.1, a power-up calibration is readily available for multi-bit $\Delta\Sigma$ DACs. The block diagram of the calibration scheme is shown in Figure 13.15.

The RAM stores the digital equivalents of the actual analog outputs of the DAC for all possible input codes. The feedback loop forces the in-band spectral components of the RAM output to follow the digital input $u[n]$, and since the inputs of the RAM and the DAC are the same, the output of the DAC follows that of the RAM. In conclusion, the in-band part of the DAC output signal will be the analog version of the input signal $u[n]$.
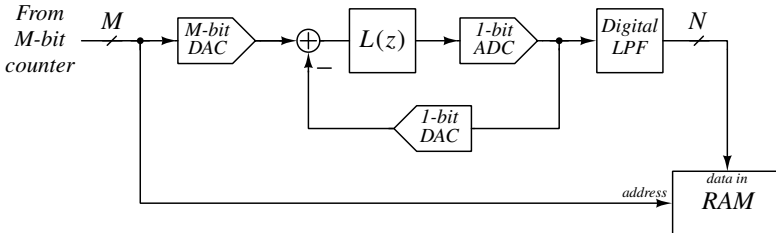
Figure 13.16   Calibration scheme for digital correction.

As discussed in Section 6.1, the calibration (i.e., the storing of the appropriate numbers in the RAM) can be performed at power-up, using an auxiliary 1-bit delta-sigma ADC (Figure 13.16). In the calibration process, a digital counter produces sequentially all input codes for the DAC. For an M-bit DAC, the counter will thus count up to 2^M. Each code from the counter is held at the DAC input for at least 2^N clock periods, where N is the required DAC linearity (in bits). The DAC output is converted by the ADC into a 1-bit data stream, whose dc average is linearly related to the DAC output. A digital lowpass filter recovers this dc value, which is then stored in the RAM at the address given by the counter output.

Background calibration is also possible. In a classical scheme [9] implemented for a current-switching DAC, the DAC contains two more unit current sources than necessary for conversion. One is used as a reference. In each clock period, a new unit source is selected for calibration, and the reference current is copied into it, while the remaining sources perform the data conversion. Thus, by selecting the calibrated source in a rotating pattern, every source can be recalibrated in each 2^M clock periods.

Another background calibration scheme for current-mode DACs was described in [10]. Here, the current sources are measured and adjusted against a reference source, using an auxiliary DAC and a 1-bit $\Delta\Sigma$ ADC.

A charge-based calibration scheme similar in principle to that of [9], but suitable for switched-capacitor DACs, was described in [11]. Here, the charges delivered by the unit-element capacitors are adjusted sequentially, using a variable reference voltage for each element, until all charges match a fixed reference charge. This scheme also requires extra unit elements.

13.3.4   Comparison of Single-Bit and Multi-Bit Delta-Sigma DACs

Comparison of single-stage $\Delta\Sigma$ DACs with single-bit or multi-bit internal truncation shows the following relative advantages of the two schemes:

Single-bit truncation Much simpler internal DAC structure can be used, without the need for thermometer coding, unit elements and digital mismatch-shaping logic.

Multi-bit truncation Several advantages can be obtained, including:

a. Simpler digital noise-shaping loop, since more aggressive NTF may be used, and since the truncation noise is reduced by at least $N - 1$ bits.

b. Less (or no) dithering, since tones are less likely to be generated, and since typically the amplitude of dithering is about 1/2 LSB, which is smaller in a multi-bit quantizer.

c. Much simpler analog smoothing filter, since the slewing and out-of-band noise in the DAC output are both reduced. Also, the sensitivity to clock jitter is reduced, due to the smaller step-size in the DAC output signal.

Generally, the advantages of multi-bit truncation outweigh those of single-bit truncation, and hence, it is preferable to design $\Delta\Sigma$ DACs with multi-bit internal DACs.

As an illustration, [3] describes a $\Delta\Sigma$ audio DAC using single-bit internal truncation, while [8] discusses a 5-bit DAC with comparable performance. The 1-bit DAC needed a 5th-order noise-shaping loop; the 5-bit DAC required only a 3rd-order loop. The 1-bit DAC used an analog smoothing filter containing a 4th-order switched-capacitor (SC) filter, followed by an SC buffer stage and a 2nd-order continuous-time active filter. By contrast, in the 5-bit system, the SC analog filter was effectively merged with the DAC itself, and required no extra operational amplifiers. However, the mismatch-shaping applied in the 5-bit DAC needed a fairly elaborate digital circuitry.

13.4   Interpolation Filtering for Delta-Sigma DACs
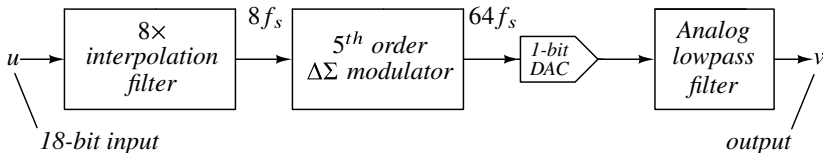

**Figure 13.17**   An 18-bit D/A converter architecture.

Efficient implementation of the digital interpolation filter (IF) preceding the noise-shaping loop (Figure 13.1) usually requires a multi-stage structure. A typical architecture and the role of the individual filter stages will next be discussed. We use as an example the IF of a classical 18-bit audio DAC [3]. The block diagram of the DAC is illustrated in Figure 13.17 and the structure of the IF in Figure 13.18.

The IF contains three cascaded finite-impulse-response (FIR) filter stages, followed by a digital sample-and-hold register. (A similar DAC IF was implemented in the more recent audio DAC discussed in [8].)   Figure 13.19 shows the spectra of the signals appearing at the inputs and outputs of the individual IF stages, as well as the final output of the DAC.

As explained in Section 13.1, the purpose of the IF is to take advantage of the increased clock frequency, and to suppress all unnecessary replicas of the signal spectrum occurring between the baseband and $f_s/2$. This will improve the dynamic range of the
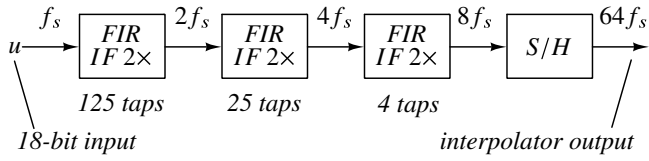
**Figure 13.18**   Interpolation filter architecture.
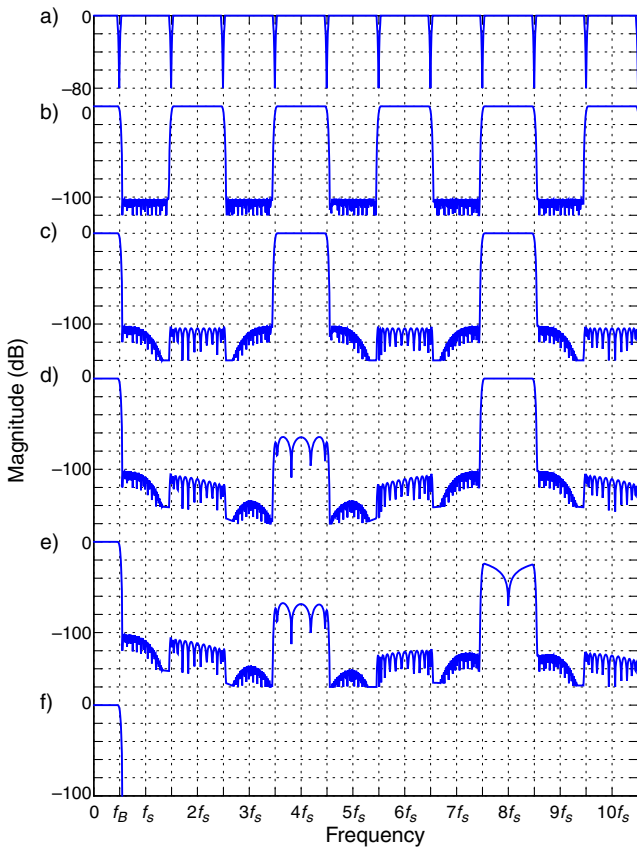
Figure 13.19   Spectra within a $\Delta\Sigma$ DAC system.

noise-shaping loop, and ease the selectivity and linearity requirements of the analog output filter. As was also mentioned in Section 13.1, the unwanted sidebands need not be totally erased, since truncation noise will be introduced in their place anyway in the noise-shaping loop NL.

In principle, it is possible to raise the sampling frequency immediately to $OSR \cdot f_s$, and then to carry out all filtering at this elevated clock rate. However, this would require all digital circuitry to function at high speed, and it would hence dissipate an unnecessarily large amount of power. It would also generate more digital activity, and thus more digital noise than necessary. Therefore, it is preferable to perform the increasing of the clock frequency and the filtering in parallel steps, with most of the signal processing performed at a low clock rate.

The first stage of the filter is operated at $2f_s$ (Figure 13.18), and is used to suppress the odd-order images. Thus, it removes the first replica of the baseband, assumed to extend approximately from f_B to $3f_B$, as well as the third replica between $5f_B$ and $7f_B$, and so forth. The operation is illustrated in Figures 13.19(a) and (b), where the first curve shows the spectrum of the Nyquist-sampled input signal, and the second the desired spectrum of the first-stage output. Notice that the requirements of this stage are very demanding: it needs to have a flat passband with extremely small (here, about 0.001 dB) gain variation in the 0 to f_B frequency range, and a very sharp cutoff in order to suppress the adjacent image, which is quite close. In the filter discussed in [3], this stage was realized by a 125-tap half-band FIR filter. (Half-band filters are FIR structures, which allow every second tap weight (except the center one) to be zero, and hence are very economical. A half-band filter can only realize, however, a frequency response which has a skew symmetry around its midpoint at $f_s/4$, as shown in Figure 13.20. Thus, the pass- and stop-band
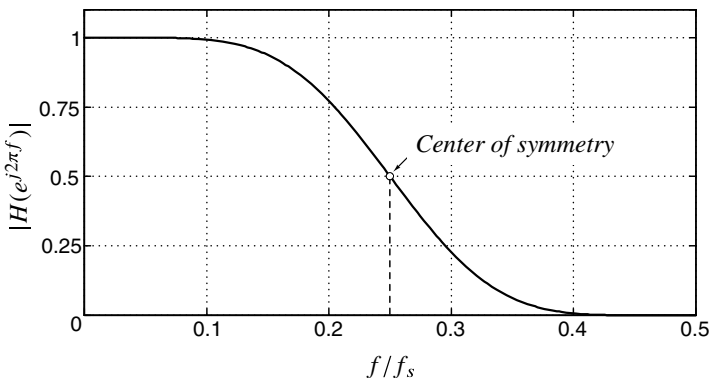
Figure 13.20   Frequency response of a half-band filter.

limit frequencies must be symmetrically located, and the ripples must be the same in the two bands. These restrictions are usually acceptable in the interpolation-by-2 filtering task performed here.)

The second stage of the IF has a clock frequency of $4f_s$. Its task is to remove the images between $3f_B$ and $5f_B$, $5f_B$ and $7f_B$, and so on, as shown in Figure 13.19(c). Its cutoff needs to be much less abrupt than that of the first stage. In the system described in [3], this task required a 24-tap half-band FIR filter. The third stage, operated at $4f_s$,

is a 4-tap half-band FIR filter. It reduces the first, third, etc. of the remaining images (Figure 13.19(d)).

Finally, a digital sample and hold operation was implemented by simply raising the sampling rate to $64f_s$, and repeating each output sample of the third IF stage 8 times. This S/H operation introduced a sinc function that had zeros at $8f_s$, $16f_s$, $24f_s$, and so on, thus contributing slightly to the filtering at no added cost, as illustrated in Figure 13.19(e). The final OSR is thus 64.

Note that the IF is designed to provide most of its noise suppression just above the signal band, where the analog filter following the DAC has the most difficulty in removing noise. In the noise-shaping loop NL following the IF, some truncation noise is added to the residual out-of-band noise (not shown in Figure 13.19). Ideally, the resulting spectrum is then accurately reproduced at the DAC output, and finally all noise is removed by the analog LPF, giving the output spectrum illustrated in Figure 13.19(f). The word length used for data in the IF is 18 bits. 19-bit accuracy is used for the constant coefficients. The overall truncation noise is 107 dB below the full-scale sine-wave signal power, consistent with an approximately 18-bit performance.

Note that FIR filters are commonly used in $\Delta\Sigma$ systems because these filters can have perfectly flat group delay, and also because the required hardware can be clocked at the lower of the input and output data rates. IIR filters are less common, but they have the advantage of being able to provide greater stopband attenuation with a given hardware complexity.

13.5   Analog Post-Filters for Delta-Sigma DACs

As discussed earlier, difficult analog circuit problems may arise in the design of the post-filter of the $\Delta\Sigma$ DAC. This filter, as illustrated in Figure 13.19, needs to remove all out-of-band portions of the output signal of the internal DAC, and it must not introduce appreciable nonlinear distortion into the signal while doing so. This task is particularly difficult for single-bit DACs, where a large two-level analog signal enters the post-filter.

Depending on the application, it may also be necessary for the post-filter to provide exactly or approximately linear phase characteristics. Alternatively, it may be designed with mildly nonlinear phase, and the phase error compensated in the digital interpolation filter.

In this section, the post-filter design issues arising for single-bit and for multi-bit DACs will be discussed separately, and illustrated with examples from commercial chips.

13.5.1   Analog Post-Filtering in Single-Bit Delta-Sigma DACs
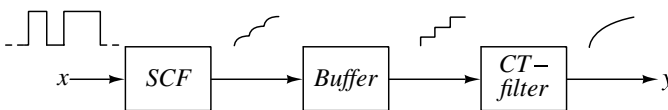
Figure 13.21   Post-filter for a 1-bit $\Delta\Sigma$ DAC and associated signals.

The block diagram of a typical post-filter for a single-bit DAC is shown in Figure 13.21. The functions of the various blocks will be explained next. As mentioned above, the input signal $x(t)$ of the post-filter in a single-bit DAC is a large two-level signal. The minimum swing of $x(t)$ is restricted by the condition that the in-band component (i.e., the useful signal) needs to be much larger than the thermal and other noises introduced by the filter itself. This necessitates a large amplitude, since most of the power in the DAC output signal is out of band. Hence, if this signal were to be entered into a conventional active filter, the active component (opamp or transconductance) would need an impractically high slew rate to avoid slew-rate-limited operation that generates harmonic distortion.

A more subtle linearity problem is due to the slew-rate limited slopes and imperfect symmetry of the waveform of $x(t)$ itself, since it is generated by an imperfect internal DAC. Also the exact shape of the waveform may depend on its previous values. Thus, while the periodic samples $x(nT)$ of $x(t)$ may correctly and linearly reproduce the useful signal, the Fourier transform of the continuous-time $x(t)$ will usually contain harmonics.

To alleviate both problems, it is customary to use switched-capacitor filter (SCF) stages as the input stages of the post-filter. An SCF with sampled-data input and output needs only the samples $x(nT)$ of $x(t)$ as its input signal, and it can remove most of the high-frequency power from the signal (and thus reduce its step-size) without requiring a high opamp slew-rate. Once the step-size of the waveform is small enough, such that the slew-rate required for its linear continuous-time (CT) processing is acceptably low, it can then be filtered by a CT active filter.
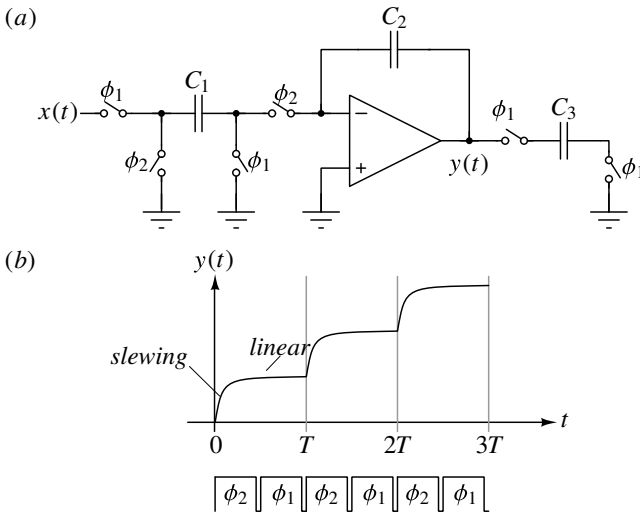

Figure 13.22   An SC integrator.

To understand the basic difference between the slew-rate requirements of CT and SC filters, consider the SC integrator shown in Figure 13.22. During phase ϕ_1, the input voltage $x(t)$ charges C_1; at the end of the nth clock phase ϕ_1, for properly designed switches, the charge will very accurately equal $C_1 \cdot x[nT]$. At the same time, C_3 samples the output voltage $y(t)$ of the opamp. This voltage has abruptly changed when ϕ_2 went high; as illustrated (in an exaggerated fashion) in Figure 13.22(b), it underwent a slewing and settling process. The settling continues during ϕ_1. For properly designed opamp and switches, the

final value $y(nT)$ will be very close to the theoretical value $y(nT-T)+(C_1/C_2)x(nT-T)$, and the charge in C_3 will be very close to $C_3 \cdot y(nT)$. Thus, the sampled signal processing is essentially unaffected by the nonlinear effects introduced by the slewing and (possibly nonlinear) settling of the opamp. Hence, its slew rate does not need to be very high, just high enough to allow accurate settling of $y(t)$ in the allocated time period. This makes the SCF particularly well suited for the task at hand.
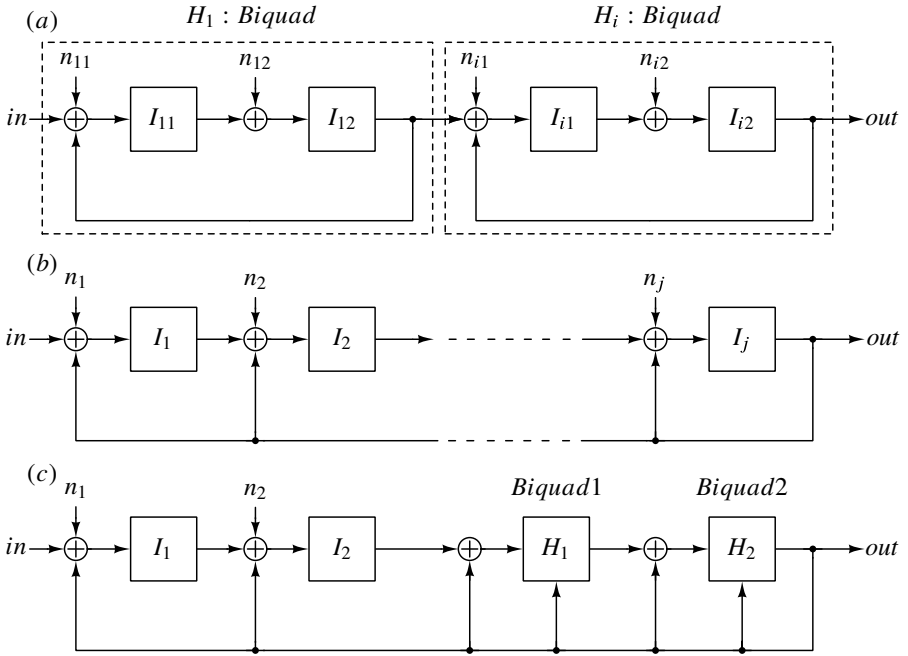

**Figure 13.23**   Reconstruction filter architectures.

Since the signal/noise ratio of the SC filter in a $\Delta\Sigma$ DAC must often be extremely high, its susceptibility to internal noise sources is an important design factor. This may result in the choice of unconventional architectures for the SCF. The commonly used configuration, a cascade of biquads, has inferior noise-gain properties, as will be demonstrated next. Consider the block diagram of such a filter, shown along with its noise sources n_{ij} in Figure 13.23(a). Clearly, n_{11} has the same gain to the output as the input. When n_{12} is referred back to the input, its power is divided by $|I_{11}|^2$, where I_{11} is the transfer function of the first integrator. This division is equivalent to differentiating (highpass filtering) the noise, and it thus reduces the in-band noise introduced by n_{12}.

Consider now the first noise source n_{i1} of the ith biquad. When it is referred back to the input, its power is divided by the factor $|H_1 H_2 \dots H_{i-1}|^2$, where H_k is the transfer function of the kth biquad. Dynamic range scaling causes $|H_1 H_2 \dots H_{i-1}| \leq 1$, and hence when n_{i1} is referred back to the input, its in-band power is not reduced. The power gain of n_{i2} is $1/|I_{11}|$ times that of n_{i1}, and hence it is first-order noise-shaped. In conclusion, for high oversampling ratios, the input-referred noise power is the weighted sum (with weight factors larger than or equal to 1) of the unshaped input noise powers of all biquads in the SCF, a decidedly unhappy situation.

Consider, by contrast, the structures shown in Figures 7.23(b) and (c). Simple analysis of the first one (sometimes called 'inverse follow-the-leader' structure) shows that referring noise source n_j to the input is equivalent to multiplying its noise power by $1/|I_1 I_2 \dots I_{j-1}|^2$. Thus, all noise sources (except n_1) are shaped, and for high OSR the noise is dominated by n_1, with all other sources contributing negligible in-band noise power.

For the structure of Figure 13.23(c), analysis shows that n_1 remains unshaped, n_2 first-order shaped, and all other noise sources are suppressed by a second or third-order shaping. Hence, again n_1 dominates the overall noise, for a very good noise performance. (Notice that – like the structure of Figure 13.23(a), but unlike that of Figure 13.23(b), Figure 13.23(c) allows the realization of finite transmission zeros, which improves its selectivity capabilities.)

In conclusion, in high-accuracy DACs, the architectures of Figures 13.23(b) and (c) may be preferable to that of Figure 13.23(a), or other commonly used SCF structures. (Note that our discussions ignored the different sensitivities of the various configurations to element value variations. This is usually less important in the present context, since the variations tend to be small, and the exact response not too critical.)
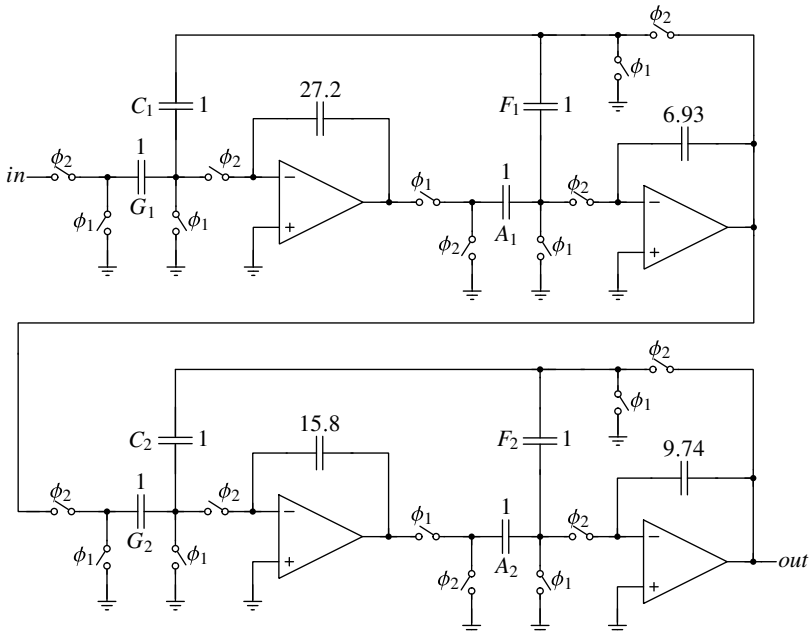
Figure 13.24   A fourth-order Bessel filter implemented with a cascade of biquads.

Figure 13.24–Figure 13.27 compare the realizations and noise-shaping properties of two Bessel SCFs realizing the same transfer function [12]. Figure 13.24 shows a biquad realization, and Figure 13.25 its noise transfer functions from source to output.   Figures 13.26 and 13.27 show the same for the inverse follow-the-leader structure. The curves demonstrate the superior noise-shaping properties of the latter architecture.

After sufficient filtering, the step-size of the SCF waveforms can be greatly reduced. However, the waveforms will still exhibit opamp-induced transients representing nonlinear
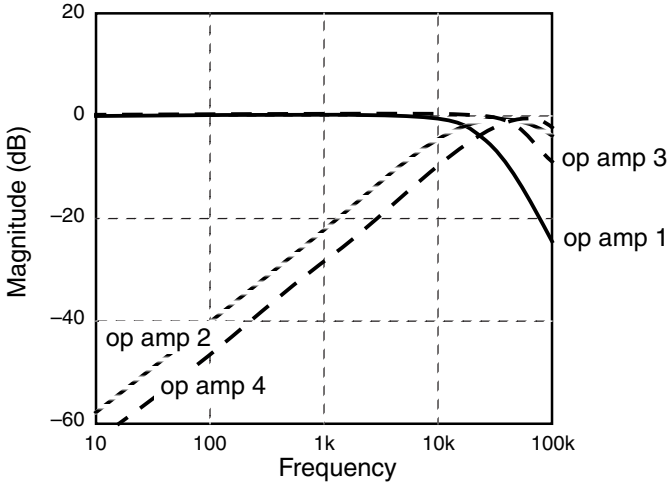
Figure 13.25   Noise gains from each opamp input to the output for the circuit of Figure 13.24.
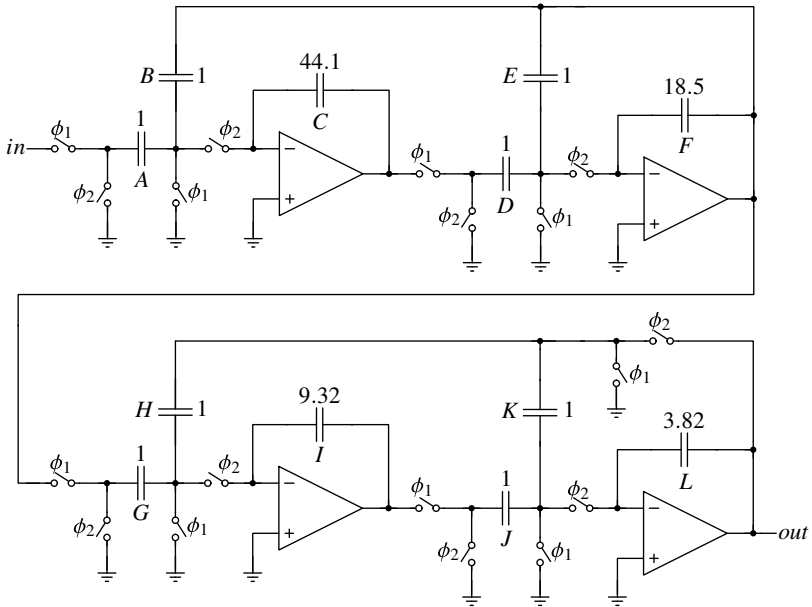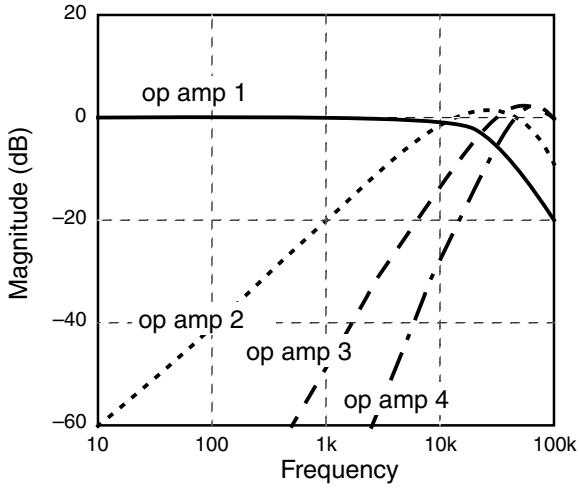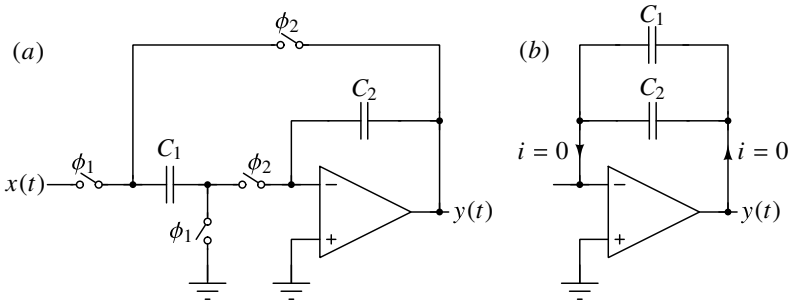

Figure 13.26   A fourth-order Bessel filter implemented with the inverse follow-the-leader topology.

Figure 13.27   Noise gains from each opamp input to the output for the circuit of Figure 13.26.


Figure 13.28   A direct-charge-transfer (DCT) stage.

distortion. Hence, it is necessary to use a buffer stage that is driven by the samples $y(nT)$ of the SCF output, and that provides a waveform free of such transients. This can be achieved by using a direct-charge-transfer (DCT) stage [13]. A lowpass DCT stage is shown in Figure 13.28(a). It samples the input signal $x(t)$ at the end of phase ϕ_1, storing it on C_1. As ϕ_2 goes high, C_1 is switched across C_2, and the two capacitors share charges (Figure 13.28(b)). Since the left terminal of the parallel combination is floating at this time, no external charge enters the branch during the charge transfer; in particular, the opamp does not need to contribute to the high impulsive current flowing. Thus, this transient is governed by a simple first-order differential equation, with only the switch on-resistances and the capacitance $C_1 + C_2$ determining the time constant. This way, a fast and clean transient, that does not exhibit the slewing and nonlinear settling behavior that the opamp would normally exhibit, can be obtained.

The output of the buffer stage can now be fed to the CT filter. This filter needs to eliminate the remaining noise above f_B. Typically, it is a second- or third-order active-RC circuit, often using the Sallen–Key configuration [14].

13.5.2   Analog Post-Filtering in Multi-Bit Delta-Sigma DACs

For multi-bit $\Delta\Sigma$ DACs, the tasks, and hence the design, of the post-filter is much easier to execute. The out-of-band noise power is reduced due to the smaller step-size; the remaining power decreases exponentially with the number of bits N retained after truncation. The corresponding simplification in the SCF is hence also greatly dependent on N.
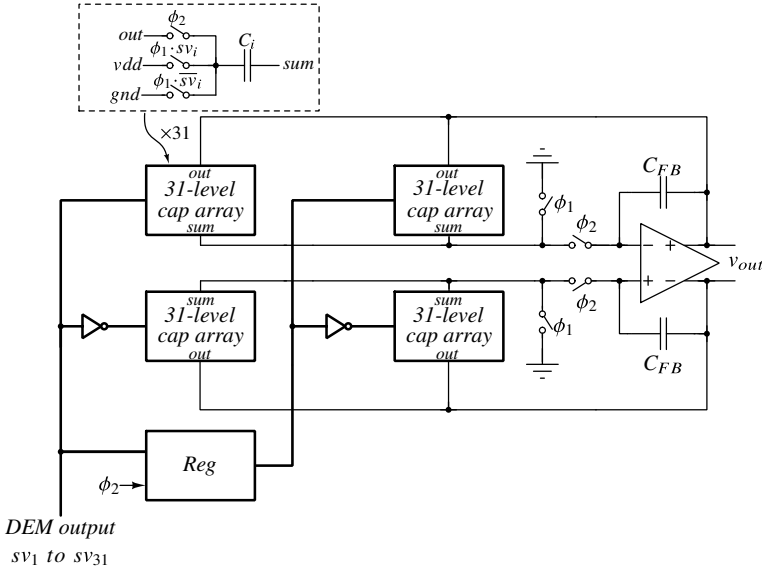
Figure 13.29   Combined DAC, DCT, and filter for a multi-bit $\Delta\Sigma$ DAC [8].

Two examples will be shown to illustrate the design of post-filters for multi-bit $\Delta\Sigma$ DACs. For the first one [8], $N \approx 5$ (31 levels) is used in the truncation. A single SC stage (Figure 13.29) is used to perform the functions of the internal DAC and the SCF.

Since it is a direct-charge-transfer circuit, the extra SC-to-CT buffer stage is unnecessary. Thus the DAC, SCF, and buffer functions, which would have required 5–6 opamps for a single-bit DAC, are all performed by a single opamp for the 5-bit system. In the circuit of Figure 13.29, the DAC operation is performed during ϕ_1, by pre-charging some of the 4×31 small capacitors in the four capacitor arrays to V_{DD} and discharging others. By duplicating the capacitor arrays, and driving the secondary arrays with a delayed digital signal, a first-order (2-tap) FIR filter function is realized. During ϕ_2, all capacitors are connected in parallel with the feedback capacitors C_{FB} in a DCT operation. The overall transfer function between the digital input and the samples of the output signal, normalized to the full-scale output, is hence

$$H(z) = \frac{1}{2}\left(\frac{1 + z^{-1}}{1 + r - rz^{-1}}\right),$$ (13.3)

where

$$r = \frac{C_{FB}}{2(C_1 + C_2 + \ldots + C_{31})}.$$ (13.4)

This simple first-order IIR filter was adequate to prepare the signal for CT filtering, performed off-chip by a Sallen–Key filter.
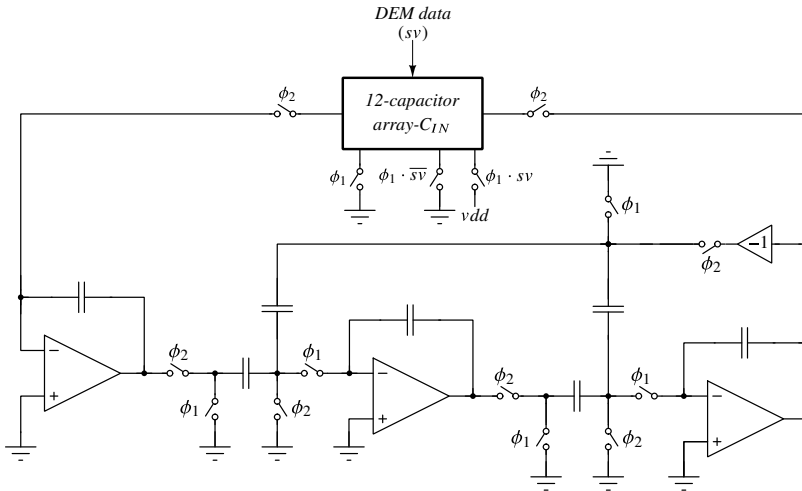
Figure 13.30   Another $\Delta\Sigma$ DAC with merged DAC, DCT and SCF filter functions [15].

Another $\Delta\Sigma$ DAC, described in [15], uses a 13-level ($N \approx 3.7$ bits) truncation. Its SCF is a third-order Chebyshev filter. It is shown in Figure 13.30 as a single-ended circuit (the actual implementation is fully differential). The DAC action is performed during ϕ_1, by charging to V_{DD} or discharging the 12 capacitors in the input array C_{IN}. During ϕ_2, the circuit is configured as a third-order inverse follow-the-leader SC filter, with the charge acquired by C_{IN} during ϕ_1 acting as its input signal. A simple first-order active-RC stage is used to perform both the CT filtering and the differential-to-single-ended conversion.

13.6   Conclusions

In this chapter on the design of $\Delta\Sigma$ DACs, the general principles and basic DAC architectures were discussed, and then various structures available for realizing their noise-shaping loops were described. Due to the high accuracy made possible by the all-digital loop, some novel loop architectures (not practical for ADC loops) exist in delta-sigma DACs. These were introduced, along with some variants of the conventional MASH configurations specific to DACs.

As was the case for $\Delta\Sigma$ ADCs, either single-bit or multi-bit internal quantization may be used in $\Delta\Sigma$ DACs. The relative merits of these two options were compared, and various methods were discussed for the filtering or compensation of the error signals introduced by the unavoidable nonlinearity of a multi-bit internal DAC. Again, some of these schemes are similar to those applicable to multi-bit ADCs, as were previously discussed in Chapter 6; others are specifically aimed at $\Delta\Sigma$ DACs, and were described in this chapter.

Next, the design issues of the digital interpolation filter were discussed, and illustrated by an example. The example chosen was an efficient multi-stage filter used in a commercial 18-bit audio delta-sigma DAC. Finally, the design of the analog post-filter used in $\Delta\Sigma$ DACs was discussed. The two different situations arising for single-bit and for multi-bit truncations were contrasted, and filter design techniques were described for both systems, along with some typical examples.

References

[1] P. J. Naus, E. C. Dijkmans, E. F. Stikvoort, A. J. McKnight, D. J. Holland, and W. Brandinal, "A CMOS stereo 16-bit D/A converter for digital audio," *IEEE Journal of Solid-State Circuits*, vol. 22, pp. 390–395, June 1987.

[2] J. C. Candy and A. Huynh, "Double integration for digital-to-analog conversion," *IEEE Transactions on Communications*, vol. 34, no. 1, pp. 77–81, Jan. 1986.

[3] N. S. Sooch, J. W. Scott, T. Tanaka, T. Sugimoto, and C. Kubomura, "18-bit stereo D/A converter with integrated digital and analog filters," presented at the 91st convention of the Audio Engineering Society, New York, Oct. 1991, preprint 3113.

[4] X. F. Xu and G. C. Temes, "The implementation of dual-truncation $\Sigma\Delta$ D/A converters," *Proceedings of the IEEE International Symposium on Circuits and Systems*, pp. 597–600, May 1992.

[5] A. Hairapetian, G. C. Temes, and Z. X. Zhang, "A multi-bit sigma-delta modulator with reduced sensitivity to DAC nonlinearity," *Electronics Letters*, vol. 27, no. 11, pp. 990–991, May 23 1991.

[6] R. Adams, K. Nguyen, and K. Sweetland, "A 113 dB SNR oversampling DAC with segmented noise-shaped scrambling," *IEEE Journal of Solid-State Circuits*, vol. 33, no. 12, pp. 1871–1878, Dec. 1998.

[7] S. R. Norsworthy, D. A. Rich, and T. R. Viswanathan, "A minimal multi-bit digital noise-shaping architecture," *Proceedings of the IEEE International Symposium on Circuits and Systems*, pp. I-5–I-8, May 1996.

[8] I. Fujimori, A. Nogi, and T. Sugimoto, "A multi-bit $\Delta\Sigma$ audio DAC with 120 dB dynamic range," *IEEE Journal of Solid-State Circuits*, vol. 35, pp. 1066–1073, August 2000.

[9] D. Groeneveld, H. J. Schouwenaars, H. A. Termeer, and C. A. Bastiaansen, "A self-calibration technique for monolithic high-resolution D/A converters," *IEEE Journal of Solid-State Circuits*, vol. 24, pp. 1517–1522, Dec. 1989.

[10] A. R. Bugeja and B.-S. Song, "A self-trimming 14-b 100 MS/s CMOS DAC," *IEEE Journal of Solid-State Circuits*, vol. 35, pp. 1841–1852, Dec. 2000.

[11] U. K. Moon, J. Silva, J. Steensgaard, and G. C. Temes, "Switched-capacitor DAC with analogue mismatch correction," *Electronics Letters*, vol. 35, pp. 1903–1904, Oct. 1999.

[12] M. Rebeschini and P. F. Ferguson, Jr., "Analog Circuit Design for ΔΣ DACs," *in S. Norsworthy, R. Schreier and G.C. Temes, Delta-Sigma Data Converters*, Sec. 12.2.3, IEEE Press, 1997.

[13] J.A.C. Bingham, "Applications of a direct-transfer SC integrator," *IEEE Transactions on Circuits and Systems*, vol. 31, pp. 419–420, April 1984.

[14] R. Schaumann and M. E. Van Valkenburg, *Design of Analog Filters*, pp. 161–163, Oxford University Press, 2001.

[15] M. Annovazzi, V. Colonna, G. Gandolfi, F. Stefani, and A. Baschirotto, "A low-power 98-dB multi-bit audio DAC in a standard 3.3-V 0.35-μm CMOS technology," *IEEE Journal of Solid-State Circuits*, vol. 37, pp. 825–834, July 2002.

CHAPTER 14

INTERPOLATION AND DECIMATION FILTERS

Most of this book deals with the $\Delta\Sigma$ modulator within a $\Delta\Sigma$ ADC or $\Delta\Sigma$ DAC system. This chapter shifts the emphasis to the design of the companion digital interpolation and decimation filters. An interpolation filter is used within a $\Delta\Sigma$ DAC system to transform low-rate data into oversampled data for the digital $\Delta\Sigma$ modulator. Conversely, a decimation filter is used within a $\Delta\Sigma$ ADC system to transform the high-speed and coarsely quantized output of the analog $\Delta\Sigma$ modulator into high-resolution low-speed data. The combination of an analog or digital $\Delta\Sigma$ modulator with its companion digital decimation or interpolation functions constitutes a full $\Delta\Sigma$ ADC or DAC system.
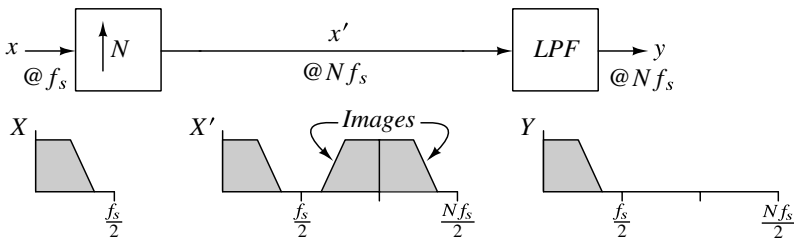
Figure 14.1   Interpolation.

As depicted in Figure 14.1, an interpolation filter effectively up-samples its low-rate input and lowpass-filters the resulting high-rate data to produce a high-rate output devoid of images. A decimation filter, illustrated in Figure 14.2, effectively does the reverse: high-rate input data is lowpass-filtered and then down-sampled to produce low-rate data containing the low-frequency content of the input signal with minimal aliasing of quantization noise or unwanted out-of-band signals. As we will see shortly, this duality extends deeper than just operational inversion: the transfer function of a decimation filter can be used in an interpolation filter, and block diagrams can be turned around to transform a decimation filter into an interpolation filter, and vice versa. For this reason, a reader only interested in one of interpolation or decimation will benefit from an understanding of the other topic. We start with interpolation because many of the concepts are more accessible with interpolation, and we strongly recommend that readers solely interested in decimation study the interpolation material as well.
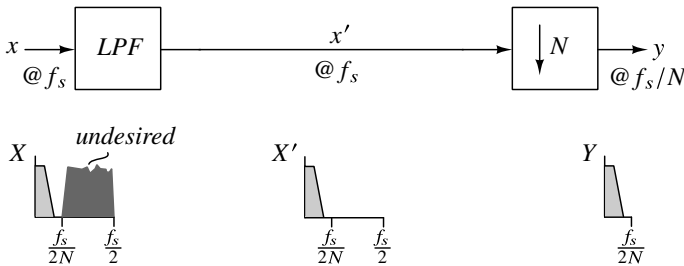

Figure 14.2   Decimation.
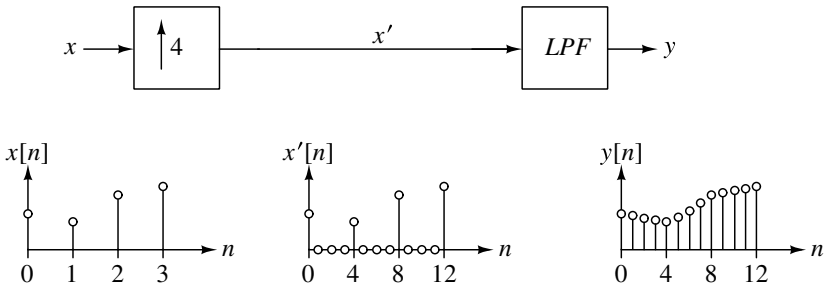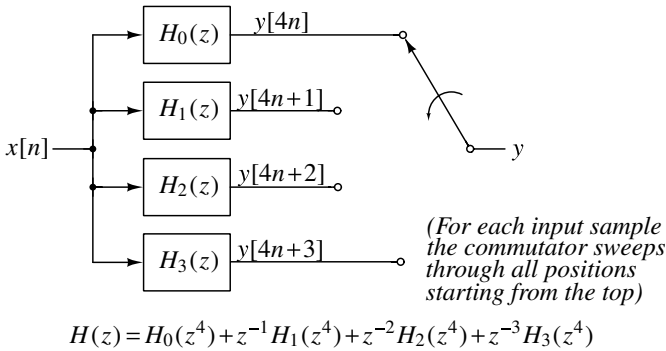
14.1   Interpolation Filtering

Figure 14.3   Interpolation by a factor of 4.

Figure 14.3 illustrates the time-domain signals associated with interpolation by a factor of $N = 4$. The incoming data is zero-stuffed with $N - 1$ zero samples to increase the sample rate by a factor of N, and the resulting high-rate data is lowpass-filtered to remove the spectral images produced by zero-stuffing. In practice, a direct implementation of these operations is inefficient because the lowpass filter LPF operates at the high output rate and must process many zero samples.

To exploit the zeros in the data, we first express N consecutive samples of the LPF output as shown below for $N = 4$:

$$
\begin{aligned}
y[4n] &= h[0]x[n] + h[4]x[n-1] + h[8]x[n-2]\ldots, \qquad (14.1)\\
y[4n+1] &= h[1]x[n] + h[5]x[n-1] + h[9]x[n-2]\ldots,\\
y[4n+2] &= h[2]x[n] + h[6]x[n-1] + h[10]x[n-2]\ldots,\\
y[4n+3] &= h[3]x[n] + h[7]x[n-1] + h[11]x[n-2]\ldots.
\end{aligned}
$$

In these expressions, $h[n]$ is the impulse response of LPF, $x[n]$ is the low-rate input

$$H(z) = H_0(z^4) + z^{-1}H_1(z^4) + z^{-2}H_2(z^4) + z^{-3}H_3(z^4)$$

Figure 14.4   Polyphase decomposition of an interpolation filter $H(z)$.

data, and the zeros associated with the zero-stuffed data $x'[n]$ have been dropped. This set of N equations is essentially a mathematical description of the *polyphase decomposition* illustrated in Figure 14.4. Instead of operating an M-tap FIR filter at the 4× rate, this arrangement operates four $(M/4)$-tap filters at the 1× sample rate, and thereby reduces the computation rate by a factor of four.
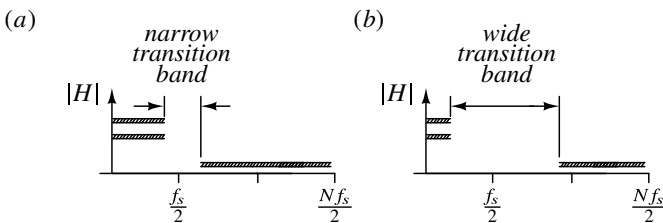

Figure 14.5   LPF specifications for (a) $OSR_{low} \approx 1$ (b) $OSR_{low} \gg 1$.

As illustrated in Figure 14.5, the LPF transition band is narrow when the input is lightly oversampled, but the transition band is much wider when OSR_{low}, the oversampling ratio of the low-rate input, is large. The LPF therefore needs to be complex when $OSR_{low} \approx 1$ whereas with $OSR_{low} \gg 1$ the LPF can be much simpler.

The simplest interpolation filter is the *zero-order hold* (ZOH), which instead of zero-stuffing the low-rate input data to produce high-rate data simply holds each sample of the low-rate data for N high-rate periods. In signal processing terms, the zero-order hold is

equivalent to zero-stuffing and filtering with the rectangular impulse response

$$h[n] = \begin{cases} 1, & 0 \le n \le (N-1), \\ 0, & \text{otherwise.} \end{cases} \tag{14.2}$$
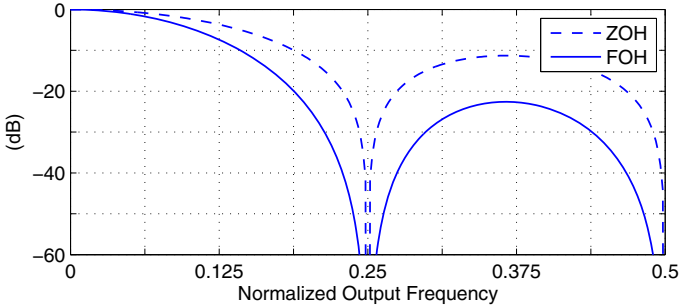
Figure 14.6   Frequency response (normalized to the dc gain) of a zero-order hold and a first-order hold ($N = 4$).

The frequency response of this filter (Figure 14.6) shows that the image attenuation is modest unless OSR_{low} is quite high. For example, a zero-order hold provides less than 40 dB of image attenuation if OSR_{low} is less than about 50.
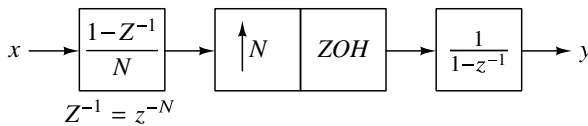
The z-transform of (14.2)

$$H(z) = \frac{1 - z^{-N}}{1 - z^{-1}} \tag{14.3}$$

suggests that using the filter

$$H(z) = \left(\frac{1 - z^{-N}}{1 - z^{-1}} \right)^2 \tag{14.4}$$

would double the alias suppression. This filter, scaled by $1/N$ to give unity dc gain, is a *first-order hold* (FOH) and is equivalent to linearly interpolating between input samples. Figure 14.6 plots the frequency response of a FOH for $N = 4$.

Figure 14.7   An efficient implementation of a first-order hold.

Figure 14.7 shows a FOH implementation in which the low-rate data is first differentiated and scaled by $1/N$, then held and integrated for N high-rate clock periods. That this arrangement implements a FOH can be understood by multiplying the transfer functions of the blocks in Figure 14.7:

$$\begin{aligned} H(z) &= \left(\frac{1 - Z^{-1}}{N} \right) \left(\frac{1 - z^{-N}}{1 - z^{-1}} \right) \left(\frac{1}{1 - z^{-1}} \right) \\ &= \frac{1}{N} \left(\frac{1 - z^{-N}}{1 - z^{-1}} \right)^2. \end{aligned} \tag{14.5}$$

Note that in both the equation above, and in Figure 14.7, a capital Z is used as the z-transform variable for low-rate data. Thus, Z^{-1} represents a delay of one low-rate clock period, and, because one low-rate clock period is equivalent to N high-rate clock periods, $Z^{-1} = z^{-N}$.

This implementation is very efficient but requires careful initialization. Specifically, the initial states of the memory elements in the differentiator and the integrator need to be the same (the usual choice is to initialize both to zero); otherwise, there will be a dc offset between the input and output. Alternatively, the state of the integrator can be set to the input every N cycles. The latter choice is recommended because the interpolator will then be robust in the face of arithmetic or round-off errors, which would otherwise persist indefinitely.

At this point, we owe the reader a discussion of scaling for interpolation filters. Since applying a constant value to the input of a ZOH produces the same constant value at the output, a ZOH clearly has a dc gain of unity. However, the dc gain of the transfer function $H(z)$ given in (14.3) is N. The missing factor of $1/N$ is from zero-stuffing (which reduces the dc content by a factor of N) but this factor is not apparent when an impulse is applied to the input of the interpolation filter. In order to convert an impulse response measured in this way into the impulse response of the effective transfer function of the interpolation filter, the response therefore needs to be scaled by $1/N$.
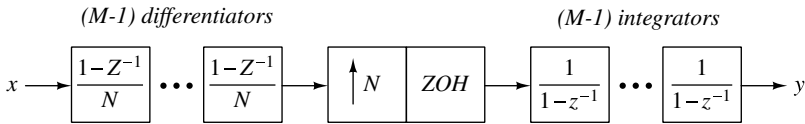
Figure 14.8   Hogenauer implementation of a sinc_N^M interpolator [1].

Further improvement in the image suppression can be obtained by using higher-order filters. Figure 14.8 shows that an Mth-order sinc_N filter

$$SINC_N^M(z) = \frac{1}{N^M} \left(\frac{1 - z^{-N}}{1 - z^{-1}} \right)^M \tag{14.6}$$

can be implemented by sandwiching a zero-order hold between $M - 1$ differentiators operating at the low rate and an equal number of integrators operating at the high rate [1]. The scaling by $1/N$ in each differentiator is often either omitted or replaced by scaling the low-rate input by $1/N^{M-1}$. The reader is reminded that the system must be initialized properly and that it is vulnerable to arithmetic errors unless further precautions are taken.

The structure of Figure 14.8 shows that if we omit the scaling by $1/N^{M-1}$, then a sinc_N^M interpolator can be implemented with $M - 1$ additions at the low rate and $M - 1$ additions at the high rate. This structure is very useful when variable N factors need to be supported, but when only a fixed value of N is needed, it is often more efficient to use other arrangements.

For $N = 2$, the direct implementation obtained by cascading a ZOH with $M - 1$ blocks implementing $(1 + z^{-1})$ accomplishes sinc_2^M interpolation with only $M - 1$ additions at the $2\times$ rate. A polyphase implementation can be even more efficient. For example, Figure 14.9 shows that a polyphase implementation of a sinc_2^5 interpolator only requires five additions at the $1\times$ rate, which is less than the four additions at the $2\times$ rate needed
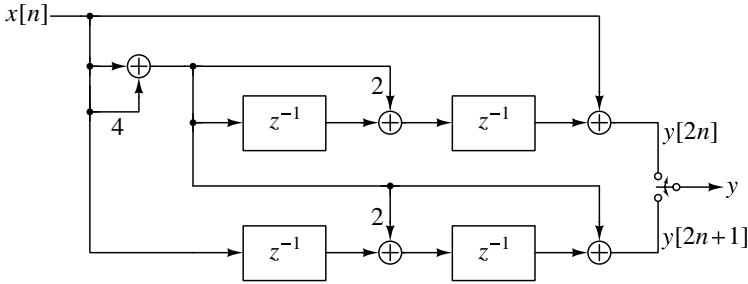
Figure 14.9   Polyphase implementation of a sinc_2^5 interpolator.

by a direct implementation. The reader is encouraged to verify that the impulse response of the filter shown in Figure 14.9 is equal to that of $(1 + z^{-1})^5$, namely $\{1, 5, 10, 10, 5, 1\}$. Since the implementation of sinc_2^M stages can be done quite economically, a cascade of two interpolate-by-two stages is usually more efficient than a single interpolate-by-four stage. For this reason, interpolators usually decompose the interpolation factor as finely as possible.

14.2   Example Interpolation Filter

Let's design an interpolation filter that interpolates a low-rate signal having $OSR_{low} = 2$ by a factor of 64 and provides $> 60\,\text{dB}$ image suppression with $< 0.5\,\text{dB}$ passband gain variation. To minimize computation, the interpolation filter will consist of a cascade of six interpolate-by-2 stages (I1–I6) and will use sinc filters wherever possible.

The minimum image attenuation provided by a first-order sinc_N filter is given by

$$|H_1(e^{j2\pi f_i/N})| = \left| \frac{1 - e^{-j2\pi f_i}}{N(1 - e^{-j2\pi f_i/N})} \right| = \left| \frac{\sin(\pi f_i)}{N \sin(\pi f_i/N)} \right|, \tag{14.7}$$

where $f_i = 1 - 1/(2OSR_i)$ and OSR_i is the oversampling ratio of the low-rate input to filter stage i. For the last stage (I6), $N = 2$ and $OSR_i = 64$. The minimum alias attenuation provided by a first-order sinc_2 filter with this value of OSR_i is

$$|H_1(e^{j2\pi f_i/N})| = -38\,\text{dB}. \tag{14.8}$$

I6 therefore needs to use a second-order sinc_2 filter to provide more than $60\,\text{dB}$ of image attenuation.

For I6, the droop of a sinc_2^2 filter at the passband edge $f_p = 1/(2OSR_i)$ is a mere $0.001\,\text{dB}$ and is therefore negligible. In general, however, the required sinc filter order must account for the passband droop. We therefore need to compute A_1, the minimum attenuation provided by a first-order sinc filter relative to the passband droop, as indicated below and tabulated in Table 14.1:

$$A_1 = |H_1(e^{j2\pi f_i/N})|/|H_1(e^{j2\pi f_p/N})|. \tag{14.9}$$

Table 14.1 lists the A_1 values and the resulting filter orders for all six stages. The required filter order is given by $\lceil 60/20 \log_{10}(A_1) \rceil$.

Table 14.1   Sinc filter order for example interpolator.

Stage	OSR_i	A_1 (dB)	Sinc$_2$ order	Droop (dB)
I6	64	38.2	2	0.001
I5	32	32.2	2	0.005
I4	16	26.2	3	0.03
I3	8	20.1	3	0.1
I2	4	14.0	5	0.8
I1	2	7.7	8	5.5

Table 14.1 shows that sinc filters of order 2 or 3 can be used for stages I3–I6, and that these stages do not violate the 0.5-dB passband droop specification. For I1 and I2, however, the sinc filter orders are much higher, and the filters have more droop than is tolerable. Two solutions to this problem will be compared: adding a compensation filter COMP before I1, and implementing I1 with an FIR filter that also provides droop compensation for I2–I6.

The code fragment below uses MATLAB$^{\text{TM}}$'s `firpm` Parks–McClellan filter design[1] function to design the COMP compensation filter.

```
%% Design FIR compensator
order = [8 5 3 3 2 2];
fp = 0.25;                    % passband edge
% Evaluate freq response of IF for npb passband bands
npb = 20;
f = linspace(0,fp,npb*2);
H = ones(size(f));
for i=1:6
    H = H .* zinc(f/2^i,2,order(i));
end
% Assemble arguments for firpm
comp_order = 4;       % Determined by trial and error
wt = abs(H(1:2:end));
[b err] = firpm(comp_order,2*f, 1./abs(H),wt)
% Want 1+err/1-err < 0.5dB, i.e. err < 0.029
```

and yields

```
b =   0.1633   -0.8711    2.4243   -0.8711    0.1633
err =  0.0087
```

The order of the compensation filter, `comp_order = 4`, is chosen to make `err` less than the target value of 0.029, as indicated in the code.
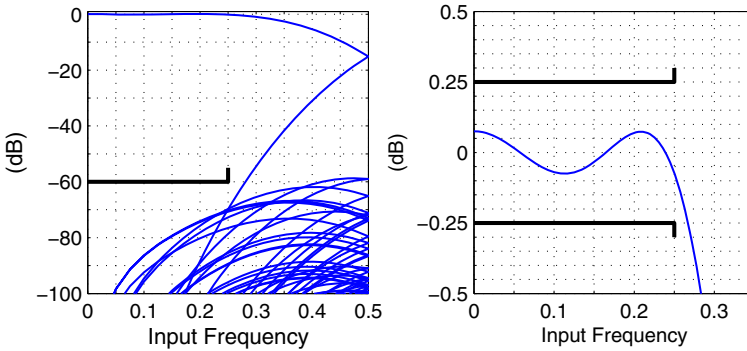
Figure 14.10 shows the full frequency response of the complete filter along with the frequency responses of stages I3–I6. (To prevent clutter, the responses of stages I1 and I2 are not shown.) As this figure shows, I6, whose output data rate is 64×, attenuates the

---

[1]The four arguments to `firpm` as used in the code fragment are: i) the order of the FIR filter, ii) the endpoints of the `npb` frequency bands in the passband, iii) the desired magnitude response at each point, and iv) the weight associated with each band. Since `firpm` is intended to fit a frequency response over a set of frequency bands spaced by "don't-care" regions, the code fragment above abuses `firpm` somewhat. For large values of `npb`, the resulting FIR filter yields an equiripple response when cascaded with the sinc filters.
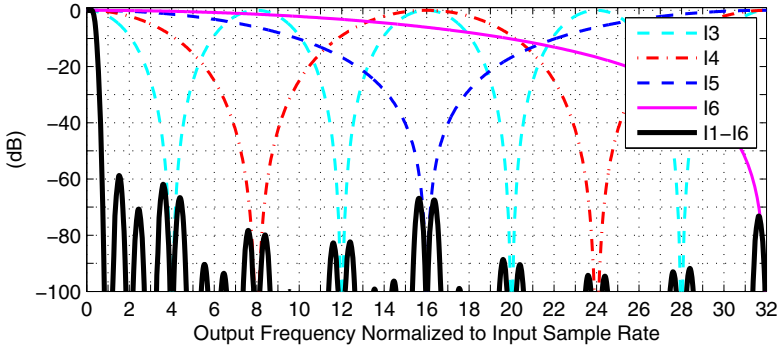
Figure 14.10   Frequency response of the complete interpolation filter.

image near $f = 32$. Similarly, I5 notches out the images near $f = 16$, while I4 and I3 take care of the images at $f = 8, 24$ and $f = 4, 12, 20, 28$, respectively. I2 and I1 quash the rest.

Figure 14.11   Folded and passband frequency responses.

Although Figure 14.10 does a good job of showing the big picture, the folded frequency response of Figure 14.11(a) does a better job of showing the transfer function magnitude in terms of the input frequency. In this plot, the horizontal axis is the input frequency, and the numerous curves represent the transfer functions to all the image terms. This plot shows that frequencies in the declared passband $[0, 0.25]$ appear at the output with very little attenuation and that the magnitudes of the various image terms are less than -70 dB. (The fact that the magnitude of most image terms is less than our -60 dB spec at all frequencies is accidental, and since $OSR_{low} = 2$ has been pre-specified, this property is also irrelevant.) The zoomed-in frequency response shown in Figure 14.11(b) confirms the passband ripple, at 0.2 dB, is also well within spec.

Next, we investigate doing compensation within I1 itself. We can again use `firpm`, but we need to add specifications to meet the requirement for 60 dB of attenuation in $[0.75, 1.0]$:

```
%% Design FIR interpolator with compensation
order = [8 5 3 3 2 2];
fp = 0.25;                    % passband edge
```

```
% Evaluate freq response of I2-I6 for npb passband bands
% and nsb stopband bands
npb = 20;    nsb = 10;
f = [linspace(0,fp,npb*2) linspace(1-fp,1,nsb*2)];
H = ones(size(f));
for i=2:6
    H = H .* zinc(f/2^i,2,order(i));
end
% Assemble remaining arguments for firpm
I1_order = 8;
a = [1./abs(H(1:2*npb)) zeros(1,2*nsb)];
rwt = (undbv(0.25)-1)/undbv(-60);
wt = [ abs(H(1:2:2*npb)) abs(H(2*npb+1:2:end))*rwt ];
[b err] = firpm(I1_order,2*f/2,a,wt)  % Want err < 0.029
```
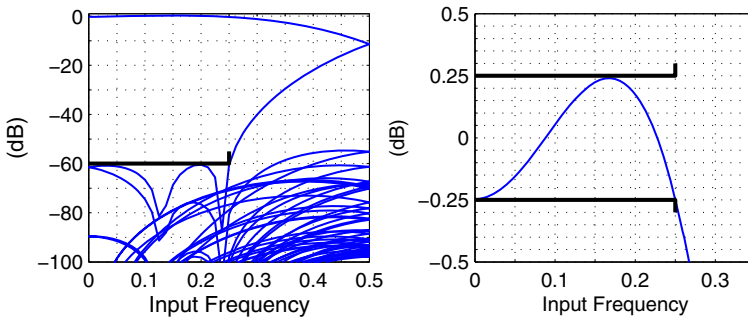
yielding

```
b(1:5) =  -0.0215   -0.0718    0.0141    0.3131    0.5041  ...
err = 0.0280
```


Figure 14.12   Folded and passband frequency responses of the alternative design.

As before, the order of the filter, `I1_order = 8`, is determined iteratively. Figure 14.12 demonstrates that this design also meets the specifications, but just barely. In practice, it is unwise to leave no margin for coefficient quantization, but we will ignore this concern and proceed with a comparison of the computational complexity of the two designs.

A polyphase implementation of this design's 9-tap I1 requires 5-tap and 4-tap FIR filters operating at the 1× rate. In contrast, the first design uses a 5-tap FIR filter operating at the 1× rate plus a sinc_2^5 interpolator. As illustrated in Figure 14.9, a polyphase implementation of a sinc_2^5 interpolator only requires 5 additions at the 1× rate and therefore requires less computation than the 4-tap FIR filter, which requires 2 multiplications and 3 additions. Additional savings accrue due to the fact that coefficients in COMP can be quantized more severely than coefficients in I1, since accurate coefficients are primarily needed to achieve stopband attenuation specifications but COMP has no such requirements. In fact, quantiz-

ing COMP's coefficients to two or three canonical signed digit (CSD) terms as follows:
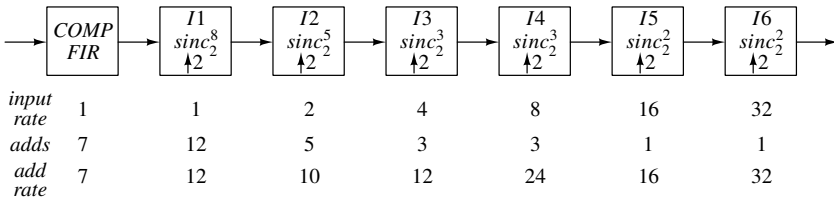
$$
\begin{aligned}
b_0 &= 2^{-3} + 2^{-5}, \\
b_1 &= -2^0 + 2^{-3}, \\
b_2 &= 2^1 + 2^{-1} - 2^{-4}, \\
b_3 &= b_1, \\
b_4 &= b_0, \tag{14.10}
\end{aligned}
$$

provides sufficient accuracy to meet the passband ripple requirements (see Figure 14.13)
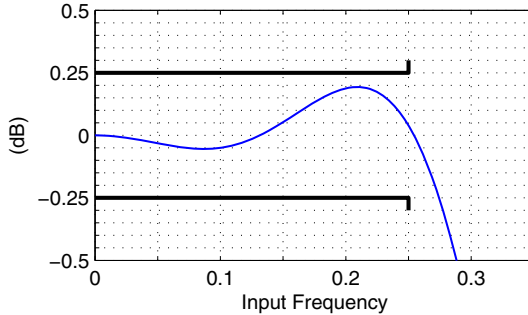
Figure 14.13   Passband response with quantized COMP coefficients.

and thus the compensation filter only needs to perform 7 additions per input sample. Figure 14.14 shows the architecture of the chosen design and lists the number of additions required per input sample for each stage. A total of 113 additions are required at the $1\times$ rate with no multiplications. The computational burden is therefore less than two additions per output sample.

|	COMP FIR	$I1$ $sinc_2^8$ $\uparrow 2$	$I2$ $sinc_2^5$ $\uparrow 2$	$I3$ $sinc_2^3$ $\uparrow 2$	$I4$ $sinc_2^3$ $\uparrow 2$	$I5$ $sinc_2^2$ $\uparrow 2$	$I6$ $sinc_2^2$ $\uparrow 2$
input rate	1	1	2	4	8	16	32
adds	7	12	5	3	3	1	1
add rate	7	12	10	12	24	16	32

Figure 14.14   Interpolation filter structure and associated computational requirements.

Having chosen the filter architecture and coefficients, our next step is to select word widths based on the noise requirements. This step is performed by choosing the places where truncation is to be performed and then computing the transfer functions from the truncators to the output of the filter. The word widths follow from dividing the noise budget among the truncators. Since this is a fairly straightforward but not an especially instructive exercise, we stop here.

14.3   Decimation Filtering

As described in the introduction, decimation filtering can be performed by filtering the high-rate input data with a lowpass filter whose cutoff is at $f_s/(2N)$ and then selecting one out of every N samples from the output of the filter. As was the case with interpolation, such a direct implementation is inefficient because the lowpass filter operates at the high (input) rate and performs unnecessary computations.
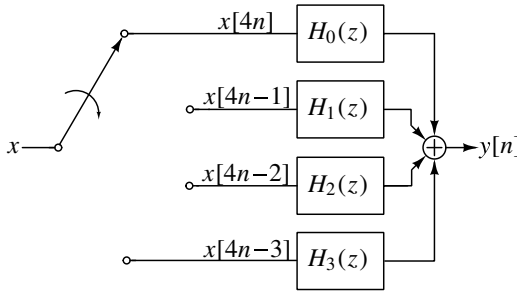
$$H(z) = H_0(z^4) + z^{-1}H_1(z^4) + z^{-2}H_2(z^4) + z^{-3}H_3(z^4)$$

Figure 14.15   Polyphase implementation of a decimation filter ($N = 4$).

The polyphase decomposition can reduce the computational burden by employing the structure illustrated in Figure 14.15 for a decimation factor $N = 4$. The response of this filter to an impulse at $n = 0$ is

$$h_0 \quad = \quad \{ h[0], \ h[4], \ h[8], \ \dots \}, \tag{14.11}$$

while the response to impulses at $n = -1, -2$ and -3 are

$$h_1 \quad = \quad \{ h[1], \ h[5], \ h[9], \ \dots \}, \tag{14.12}$$
$$h_2 \quad = \quad \{ h[2], \ h[6], \ h[10], \ \dots \}, \tag{14.13}$$
$$h_3 \quad = \quad \{ h[3], \ h[7], \ h[11], \ \dots \}. \tag{14.14}$$

The system therefore does the same job as filtering with a filter whose impulse response is

$$h = \{ h[0], \ h[1], \ h[2], \ \dots\} \tag{14.15}$$

and down-sampling the output of that filter by a factor of four, but does so with approximately one quarter of the computations.

Equations (14.11) through (14.14) demonstrate a somewhat counterintuitive property of a decimation filter. Even though we describe such a system with a transfer function $H(z)$, which is a property of a linear *time-invariant* system, decimation filters are, strictly speaking, time-varying. The periodic nature of the time variation is responsible for the mixing products that we otherwise know as aliases.

As with interpolation filtering, it is efficient to perform decimation filtering in stages. The example interpolation filter of Section 14.2 implemented a filter

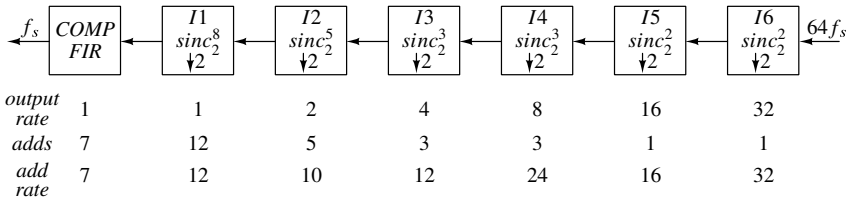$$H(z) = COMP(z^{64})I_1(z^{32})I_2(z^{16})I_3(z^8)I_4(z^4)I_5(z^2)I_6(z), \tag{14.16}$$

f_s ← COMP FIR ← I1 $sinc_2^8$ ↓2 ← I2 $sinc_2^5$ ↓2 ← I3 $sinc_2^3$ ↓2 ← I4 $sinc_2^3$ ↓2 ← I5 $sinc_2^2$ ↓2 ← I6 $sinc_2^2$ ↓2 ← $64 f_s$

output rate	1	1	2	4	8	16	32
adds	7	12	5	3	3	1	1
add rate	7	12	10	12	24	16	32

Figure 14.16   Decimation filter corresponding to Figure 14.14.

which we can turn around as shown in Figure 14.16 to accomplish decimation by a factor of 64. The alias attenuation and passband performance of this decimation filter are equivalent to the image attenuation and passband performance of the interpolation filter. Since the input to the interpolation filter was assumed to be oversampled by a factor of two, the output of the decimation filter is also oversampled by a factor of two, and thus in the context of a $\Delta\Sigma$ ADC system, further filtering would probably be necessary.
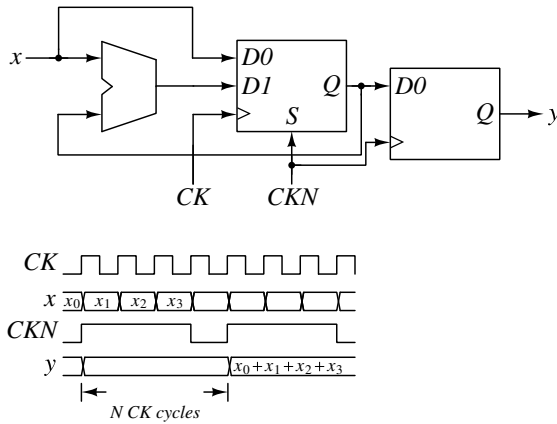
Figure 14.17   Accumulate-and-dump (AAD) decimator.

Thus far, we have emphasized the similarities between interpolation and decimation filtering. Important differences that deserve mention relate to sinc filters and decimation for $\Delta\Sigma$ ADCs. The decimator corresponding to a zero-order hold interpolator is the accumulate-and-dump (AAD) decimator stage depicted in Figure 14.17. This block implements the first-order sinc response, scaled by N,

$$H(z) = \frac{1 - z^{-N}}{1 - z^{-1}}, \tag{14.17}$$

by accumulating the input for N cycles and then latching the result and resetting the integrator. The first difference is therefore that an AAD stage requires computation whereas a ZOH does not. For the $sinc_N^M$ decimator illustrated in Figure 14.18, two more distinctions emerge. The reader should be immediately alarmed by the open-loop integrators connected to the input, since there is nothing to prevent a dc input from causing the integrators to ramp to infinity. However, because we know that the combination of integrators and differentiators has a gain N^M, and if we assume that x consists of nonnegative integers less than or equal to K, then implementing all the arithmetic operations shown in Figure 14.18 modulo
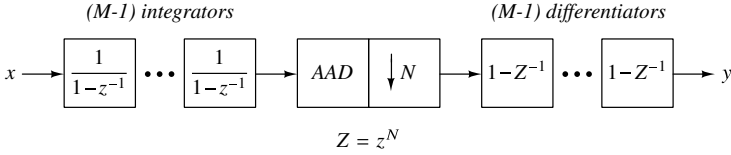
Figure 14.18   Sinc$_N^M$ decimator.

any integer larger than KN^M will yield the correct result. Thus, the integrators are not problematic, provided that they are allowed to wrap and enough bits are used. The second distinction is that arithmetic errors within this structure result in finite-length transients, and thus improper initialization of the memory elements in the integrators and differentiators is not catastrophic.

The final differences between decimation and interpolation that we wish to highlight relate to their use in $\Delta\Sigma$ systems. First, the word width of a $\Delta\Sigma$ ADC output is typically quite narrow, 1–4 bits, and this fact can favor architectures that decimate by a large factor up front. Second, the attenuation requirements of a decimation filter are typically much more stringent than those of an interpolation filter. To see why, recall that the decimation filter is responsible for rejecting out-of-band signals and, in systems such as wireless receivers, out-of-band signals typically must be attenuated by roughly 100 dB to provide adequate selectivity. Even in the absence of large interferers, the need to make the sum of the noise from $N - 1$ alias bands smaller than the in-band quantization noise typically requires more than 80 dB of alias attenuation, whereas for interpolators 60 dB of image suppression is often sufficient.

14.4   Example Decimation Filter

Let's design a decimation filter for a $\Delta\Sigma$ ADC operating at an oversampling ratio of 64. Our specifications for the decimation filter are >100 dB alias attenuation and <0.1 dB passband variation. We start by designing a filter that decimates by 32 using five decimate-by-2 stages. The procedure is similar to that used in the interpolation example. Specifically, for each stage we calculate the minimum alias protection provided by a first-order sinc$_N$ filter

$$A_1 = \frac{|H_1(e^{j2\pi f_p/N})|}{|H_1(e^{j2\pi f_i/N})|} = \left| \frac{\sin(\pi f_p)\sin(\pi f_i/N)}{\sin(\pi f_p/N)\sin(\pi f_i)} \right|, \tag{14.18}$$

where $f_p = 1/(2OSR_i)$ is the upper edge of the passband, $f_i = 1 - 1/(2OSR_i)$ is the lower edge of the stopband, OSR_i is the oversampling ratio at the *output* of stage i, and $N = 2$ because each stage decimates by 2. The required sinc$_2$ order is then given by $\lceil 100/(20\log_{10} A_1) \rceil$, where $\lceil x \rceil$ means the smallest integer greater than or equal to x. See Table 14.2.

This table shows that the passband droop of stage D5 is quite large. Nonetheless, using the same method as in the interpolation example, we find that a symmetric eighth-order FIR compensation filter with coefficients

```
b = [ 0.0709 −0.4964 1.8350 −4.495 7.1722  −4.495 ...]
```

Table 14.2   Sinc order for first five stages of the example decimator.

stage	OSR_i	A_1 (dB)	$sinc_2$ order	droop (dB)
D1	32	32.2	4	0.010
D2	16	26.2	4	0.042
D3	8	20.1	5	0.210
D4	4	14.0	8	1.348
D5	2	7.7	14	9.628

satisfies the 0.1-dB passband variation specification with sufficient margin that the quantized coefficients

$$
\begin{aligned}
b_0 &= 2^{-4} + 2^{-7}, \\
b_1 &= -2^{-1} + 2^{-8}, \\
b_2 &= 2^1 - 2^{-3} - 2^{-5} - 2^{-7}, \\
b_3 &= -2^2 - 2^{-1} + 2^{-8}, \\
b_4 &= 2^3 - 2^0 + 2^{-2} - 2^{-4} - 2^{-6},
\end{aligned}
\tag{14.19}
$$

with b_5–b_8 equal to b_3–b_0, also meet the passband ripple spec.
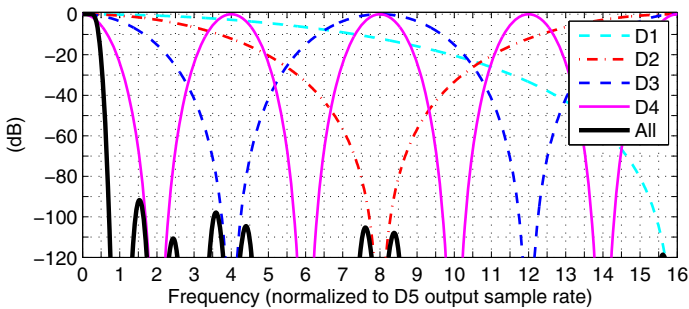
Figure 14.19   Frequency response of D1-COMP.
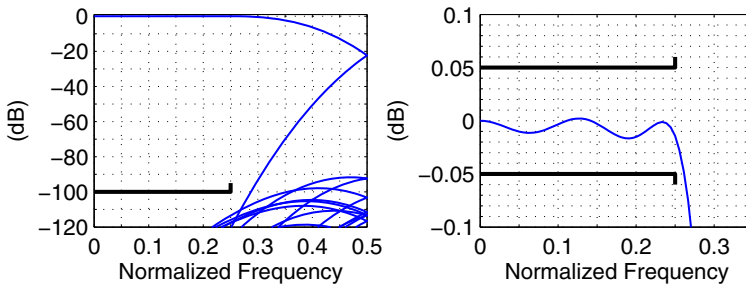
Figure 14.20   Folded frequency response of D1-COMP.

Figure 14.19 and Figure 14.20 plot the full and folded frequency response of D1 to D5 plus the compensator. Figure 14.19 includes the individual responses of D1–D4 to illustrate

how these stages contribute to the overall frequency response. (To maintain legibility, the response of D5 is not shown.) These figures demonstrate that sinc decimation with post-compensation is able to accomplish decimation down to an output OSR of 2.

Let's take a moment to tally the computational complexity of the decimation filter thus far. A direct implementation of a sinc_2^M decimator requires M additions at the $2\times$ rate, minus one, because the last $(1 + z^{-1})$ block is subsampled, or $2M - 1$ additions at the $1\times$ rate. Similarly, a direct implementation of a sinc_2^M interpolator requires $M - 1$ additions at the $2\times$ rate, or $2M - 2$ additions at the $1\times$ rate. Table 14.3 lists these numbers along with the number of $1\times$-rate additions for optimized polyphase implementations up to $M = 10$.

Table 14.3   Number of low-rate additions for optimized sinc_2^M interpolation/decimation.

M	1	2	3	4	5	6	7	8	9	10
Interpolator– direct	0	2	4	6	8	10	12	14	16	18
Interpolator– polyphase	0	1	3	4	5	8	9	9	12	14
Decimator– direct	1	3	5	7	9	11	13	15	17	19
Decimator– polyphase	1	2	5	5	7	9	13	11	17	16

This table shows that for interpolation a polyphase implementation is generally more efficient than a direct implementation, with particularly large savings for $M = 5$ and 8. For decimation, a polyphase implementation is somewhat less advantageous, but fortunately, for the design at hand, the $M = 4$, 5, and 8 cases provide roughly 20% savings. A polyphase implementation is therefore advised for stages D1–D4. For D5, a polyphase implementation of sinc_2^{14} requires 26 adds per output sample whereas a cascade of a non-decimating sinc_2^6 with a polyphase implementation of a decimating sinc_2^8 (the same block used in D4) requires $12 + 11 = 23$ adds and is thus slightly more efficient. Last, if no partial sums are reused, COMP requires 19 additions per output sample. However, by identifying terms that are related by a power of 2, namely the $-2^{-1} + 2^{-8}$ term in b_1 and b_3, and the $-2^{-5} - 2^{-7}$ and $-2^{-4} - 2^{-6}$ terms in b_2 and b_4, the adder count can be reduced to 17.

With these optimizations, the total computational burden is 210 additions per output sample, nearly double that of our interpolation filter example. The primary reason for the increase is the more stringent stopband specification. The one mitigating factor is that the input word width is typically only a few bits, and thus the additions in the first (highest-rate) stage have a lower power penalty than those in subsequent stages. In our design, the first stage accounts for 38% of the total addition rate, and thus the savings associated with a low word width are significant. In the extreme case of a single-bit modulator, implementing the first few decimation stages with a lookup table can be especially efficient.

For a general-purpose ADC, it often suffices to decimate the data to within a factor of two of the Nyquist rate. The reasons are twofold. First, because the channel filtering requirements are dependent on the application, the user may need to filter the data further. Second, implementing a steep transition band for frequencies that the user may reprocess anyway wastes power and adds unnecessary latency. The decimation filter consisting of D1–D5 and COMP is therefore a practical system as it stands. We will nonetheless add a halfband filter (HBF) in the next section to complete the design, but at this point, we take that stage as a given and show both the full architecture of the filter (Figure 14.21) and example spectra at each stage (Figure 14.22).
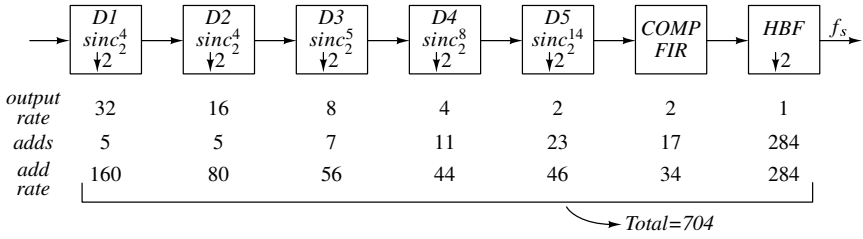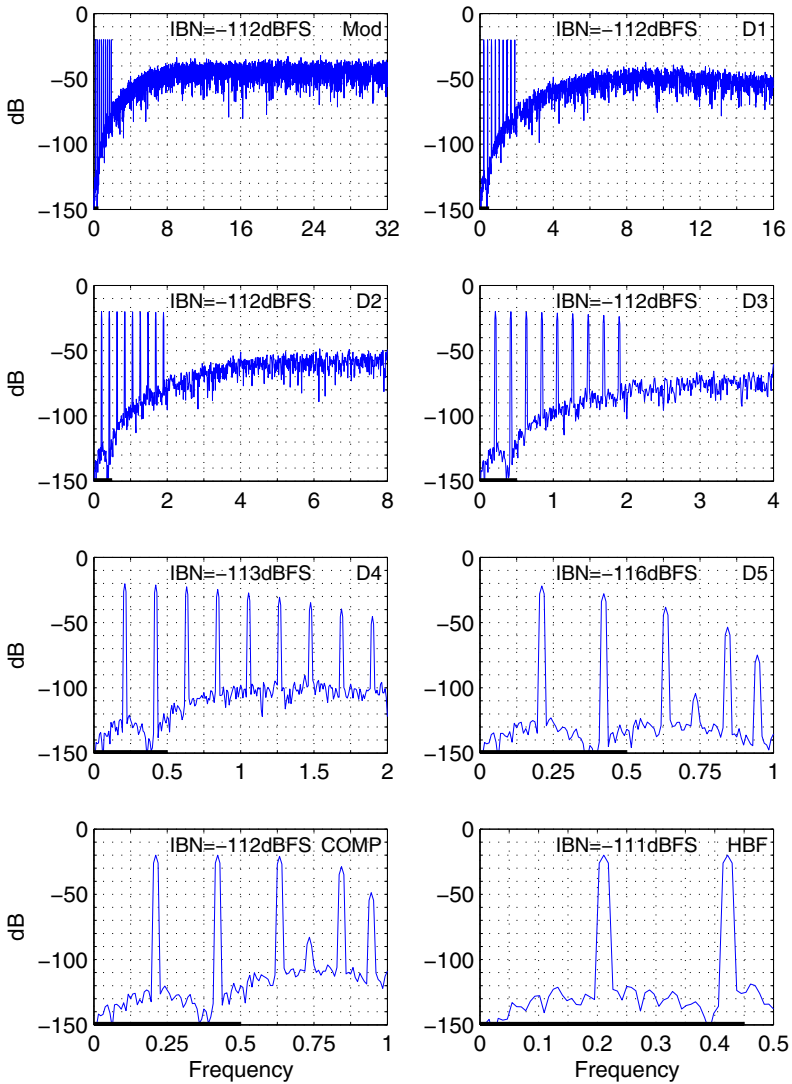
Figure 14.21   Complete decimation filter.

Figure 14.22   Simulated spectra after each decimator stage.

The decimation filter requires ~700 additions per output sample. (Multiplications are converted into additions based on the number of CSD terms.) Roughly 40% of these operations are associated with the last stage of decimation. Since this stage has a very high order (more than 200) and operates on wide words, the filter area is also dominated by the last stage. The reasons for wanting to omit this stage should now be quite clear.

Figure 14.22 starts with the output spectrum of a third-order, five-level $\Delta\Sigma$ modulator given 9 equally spaced -20-dBFS low-frequency input tones that extend beyond the signal band. As the sample rate is decreased by each stage of decimation, the block of tones occupies a wider portion of the frequency axis until the output of D5, where the decimation filter has essentially eliminated three of the input signal's highest-frequency components. Looking at the $[0, 0.5]$ region of this spectrum, we see some attenuation of the second in-band tone but no evidence of aliased tones. In $[0.5, 1.0]$ the alias tones at $f = 0.93$ and 0.7 are apparent, but these get eliminated along with the other out-of-band tones by the halfband filter. Looking at the second last plot, we can tell that COMP is doing its job by the fact that the amplitude of the second in-band tone has been restored. The fact that only the in-band tones remain in the last plot shows that HBF is also doing its job.

In addition to observing the behavior of the decimation filter with test signals at various frequencies, it is important to verify that the decimation filter provides sufficient attenuation of the modulator's quantization noise. The plots in Figure 14.22 report the in-band noise (IBN) for this reason. As these plots indicate, the in-band noise is not affected until after D5 where IBN actually *decreases* by 4 dB. This IBN decrease is a result of the attenuation of D5 in the upper region of the passband and does not correspond to a true SQNR increase. This fact explains why IBN returns to its original value after the equalization provided by COMP. Since IBN is degraded by only 1 dB after the final stage of decimation, this simulation also indicates that the decimation filter provides adequate attenuation of the modulator's quantization noise.

14.5 Halfband Filters

Halfband filters are a special class of filters suitable for decimation or interpolation by a factor of 2. The frequency response of such filters satisfies the symmetry condition

$$H(e^{j2\pi(0.25-f)}) = 1 - H(e^{j2\pi(0.25+f)}), \tag{14.20}$$

which causes nearly half of the impulse response samples to be zero. Figure 14.23 illustrates the impulse and frequency responses of an example 15-tap halfband filter that provides 32 dB of stopband attenuation with an 80% passband and stopband. All odd taps are zero except for the middle tap, which is 0.5, and thus no computation is required for 7 of the 15 taps. The symmetry property implies that the passband ripple is `dbv(1+undbv(-32))` $= 0.2$ dB.

Let's compare the computational requirements of a halfband filter designed for an 80% passband and >80 dB of stopband attenuation, which implies a passband ripple of less than 0.001 dB, with those of a general FIR filter. Using Matlab's `firhalfband` function, we find that a 47-tap halfband filter meets the specifications and that this filter has 24 non-trivial taps. A general FIR filter designed for similar stopband specifications but with the passband ripple relaxed to 0.1 dB has 34 nonzero taps. In this case, the microscopic passband deviation of the halfband filter comes at a lower computational cost than a general
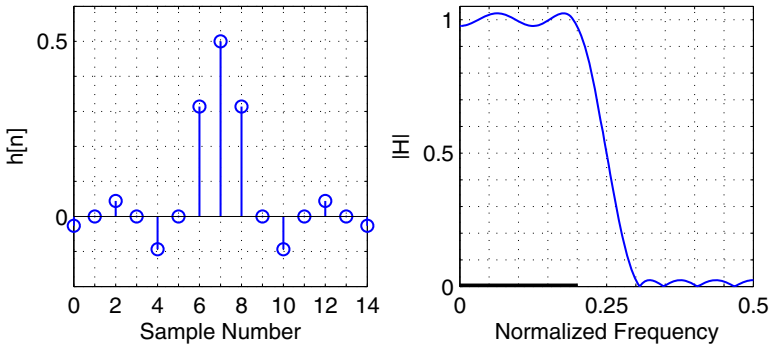
Figure 14.23   Example halfband filter.

FIR filter having a much larger passband variation. The halfband filter does require more registers and has higher latency, however.
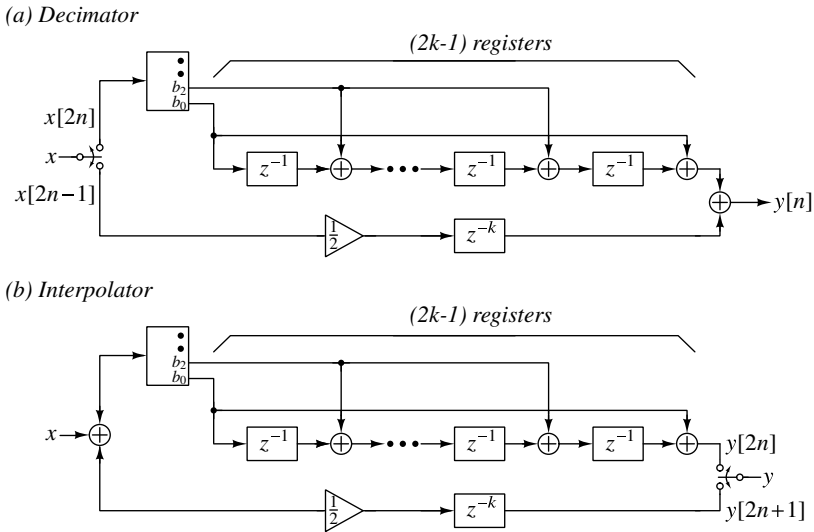
(a) Decimator

(b) Interpolator

**Figure 14.24**   Decimating and interpolating halfband filter implementations.

Since we are interested in using halfband filters for decimation or interpolation by a factor of two, Figure 14.24 depicts polyphase implementations of decimating and interpolating halfband filters. The alternating zeros of the impulse response make one branch of each polyphase implementation collapse to a pure delay, which saves both computation and registers. To facilitate reuse of partial products, each diagram uses the transpose filter topology and includes a block that performs all coefficient multiplications on each incoming data sample at one time.

14.5.1 Saramäki Halfband Filter

Despite the significant computational savings provided by a halfband filter relative to a regular FIR filter, the amount of computation can still be quite high, especially when the transition band is narrow and the stopband attenuation is large. Normally, large stopband attenuation requires accurate coefficients and consequently many CSD terms for each coefficient, but the remarkably efficient halfband filter structure proposed by Saramäki [2] achieves high attenuation using just a few CSD terms. The design process is somewhat complex, but fortunately, the $\Delta\Sigma$ designer the $\Delta\Sigma$ toolbox function `designHBF` (page 525) encapsulates the procedure.

Let's use this function to design the final stage of our decimation filter. We will continue to require 100 dB of stopband attenuation but will need to reduce the fraction of protected frequencies below 100%. If we choose a passband fraction of 90%, then frequencies between 0 and $0.9 \times f_{s,out}/2$ will experience minimal attenuation and corruption by aliases but frequencies between $0.9 \times f_{s,out}/2$ and $f_{s,out}/2$ may be attenuated and corrupted by aliases. The latter frequency range may need to be filtered out by the user according to the channel-filtering requirements. The code below creates halfband filters having a 90% passband and a -100-dB stopband using the standard and Saramäki methods.

```
hbf1 = firhalfband('minorder',0.9*0.5,undbv(-100));
[f1,f2,info] = designHBF(0.9*0.25,undbv(-100),0);
```
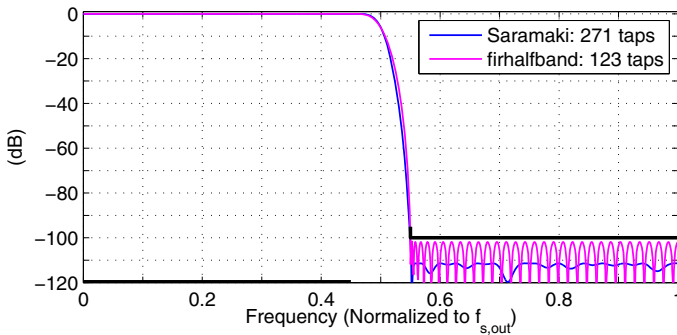

Figure 14.25   Frequency response of the example decimator's halfband stage.

The `firhalfband` function returns a 123-tap filter requiring a minimum of 31 multiplications plus 62 registers and additions while `designHBF` returns a filter that requires about 200 registers but only 284 additions and no multiplications. The frequency response curves shown in Figure 14.25 show that the Saramäki system does not have the numerous notches in the stopband exhibited by the standard filter but nonetheless provides more than 110 dB of attenuation over much of the stopband. A realization of this filter is depicted in Figure 14.26, and the coefficients and their CSD expansions are listed in Table 14.4.

F2 subfilter:
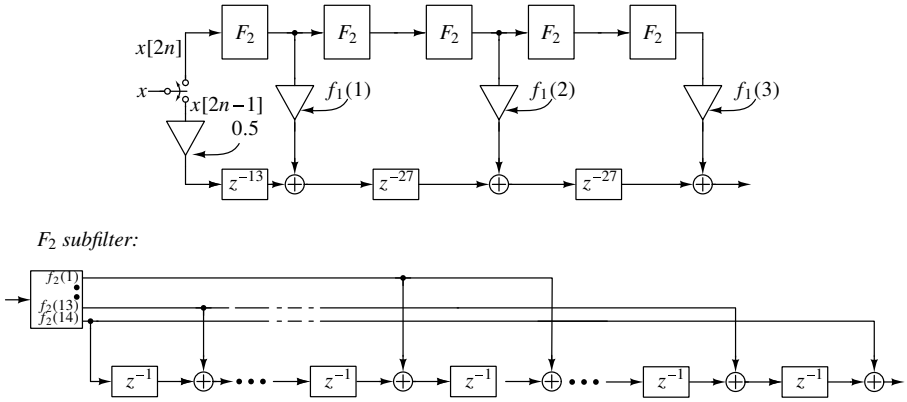
Figure 14.26   Structure of the Saramäki halfband decimation filter.

Table 14.4 F_1 and F_2 coefficients.

n	$f_1(n)$	CSD	$f_2(n)$	CSD
1	0.9453	$2^0 - 2^{-4} + 2^{-7}$	0.6249	$2^{-1} + 2^{-3} - 2^{-13}$
2	−0.6406	$-2^{-1} - 2^{-3} - 2^{-6}$	-0.2031	$-2^{-2} + 2^{-5} + 2^{-6}$
3	0.1953	$2^{-2} - 2^{-4} + 2^{-7}$	0.1177	$2^{-3} - 2^{-7} + 2^{-11}$
4			−0.0791	$-2^{-4} - 2^{-6} - 2^{-10}$
5			0.0566	$2^{-4} - 2^{-8} - 2^{-9}$
6			−0.0410	$-2^{-5} - 2^{-7} - 2^{-9}$
7			0.0311	$2^{-5} - 2^{-13} - 2^{-15}$
8			−0.0232	$-2^{-6} - 2^{-7} + 2^{-12}$
9			0.0168	$2^{-6} + 2^{-10} + 2^{-12}$
10			−0.0122	$-2^{-6} + 2^{-8} - 2^{-11}$
11			0.0085	$2^{-7} + 2^{-10} - 2^{-12}$
12			−0.0058	$-2^{-8} - 2^{-9} + 2^{-14}$
13			0.0037	$2^{-8} - 2^{-12} + 2^{-15}$
14			−0.0032	$-2^{-8} + 2^{-10} - 2^{-12}$

14.6   Decimation for Bandpass Delta-Sigma ADCs
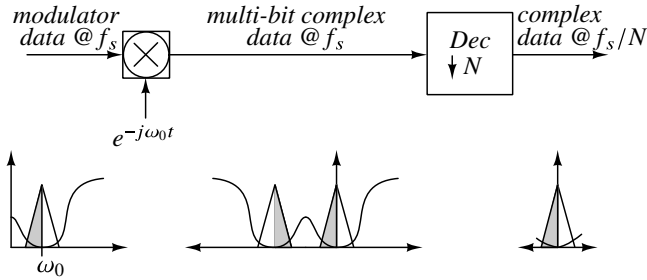
Figure 14.27   Bandpass decimation.

Decimation for a bandpass ADC can be implemented as shown in Figure 14.27. In this arrangement, the coarsely-quantized output of the modulator is multiplied by a complex exponential $e^{-j\omega_0 t}$ to mix the desired band to dc. The resulting complex data is then processed by a pair of real decimation filters.
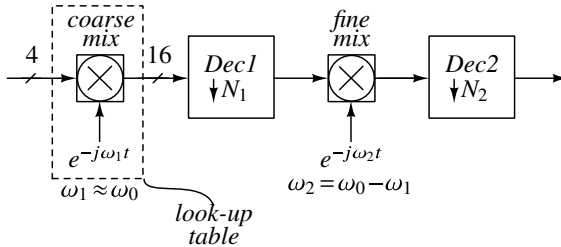
Figure 14.28   Coarse/fine mixing.

Since a complex mix operating at the fast modulator rate can be computationally expensive, it is helpful to restrict the mixing frequencies to, say, multiples of $f_s/64$ so that the mixer only needs to multiply by 16 possible values of sine and cosine, and thereby allows us to replace multiplication with a table lookup operation. After partial decimation, a fine mixing operation can center the passband precisely at dc without incurring a large power penalty because the data rate has been reduced. Figure 14.28 illustrates the concept. The granularity of the coarse mix limits the amount of decimation that can be performed before the fine mix, and thus there is a trade-off between the power consumed in the coarse and fine mixing operations. For example, if the coarse mix has a resolution of $f_s/64$, then decimation by no more than a factor of 16 or so can be performed without requiring steep filter characteristics. If the power consumption of a multiplier operating at $f_s/16$ is reasonable, then the resolution of the coarse mix is sufficient. If not, then the coarse mix needs to have finer resolution.

If the center frequency is fixed at $f_s/4$, the mixing operation is trivial, because sine and cosine are period-4 sequences: $\{0, 1, 0, -1\}$ and $\{1, 0, -1, 0\}$. In this case, the structure of Figure 14.29 can be used to perform both downconversion by $f_s/4$ and filtering by a transfer function $H(z)$ that has been decomposed into $H(z) = H_0(z^2) + z^{-1}H_1(z^2)$. Similar structures can be used for other simple rational fractions of f_s.
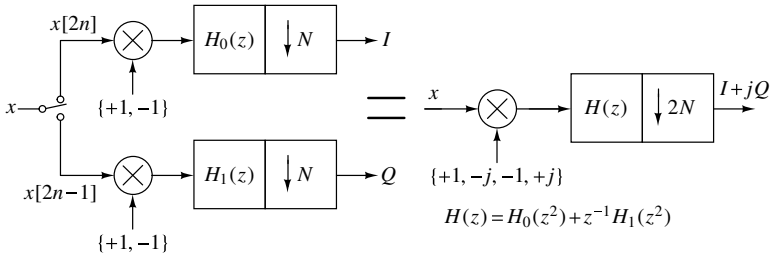
$$H(z) = H_0(z^2) + z^{-1} H_1(z^2)$$

Figure 14.29   Bandpass decimation for $f_0 = f_s/4$.

14.7   Fractional Rate Conversion

Thus far, we have only considered interpolation or decimation by integer factors. Decimation by a rational factor M/N can, in principle, be performed by interpolating by M and then decimating by N [3]. This approach is workable if M is small, but is prohibitive in the general case. We start with an example that falls into the first category and then consider decimation by arbitrary fractional factors.

14.7.1   Decimation by 1.5
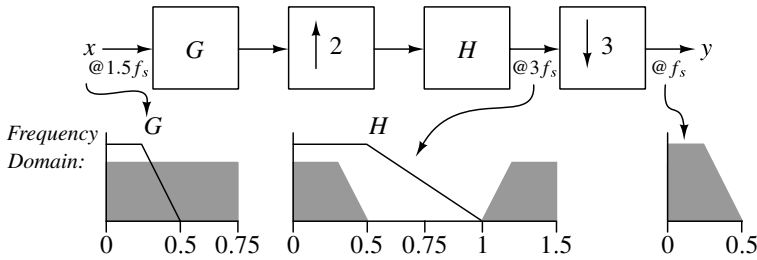
Figure 14.30   Architecture of a decimate-by-1.5 system.

Figure 14.30 depicts a candidate architecture for a decimate-by-1.5 system. As illustrated in the figure, filtering performed at the input rate widens the transition band of the high-rate filter H, and thus we expect this arrangement to be more efficient than omitting G and putting the full burden on the H filter. The reader should recognize that H can be merged with the up-sampler by employing a 2-phase interpolator topology or merged with the down-sampler by employing a 3-phase decimator topology. However, if H is merged with the up-sampler, then 2/3 of the computed samples are simply discarded and thus this arrangement is very inefficient. Similarly, if H is merged with the down-sampler, then half of the data on which the phase filters operate is zero. To eliminate these inefficiencies, the polyphase arrangement shown in Figure 14.31 can be used. The reader is encouraged to

verify that this system mimics the up-sampling, filtering, and downsampling operations of Figure 14.30.[2]
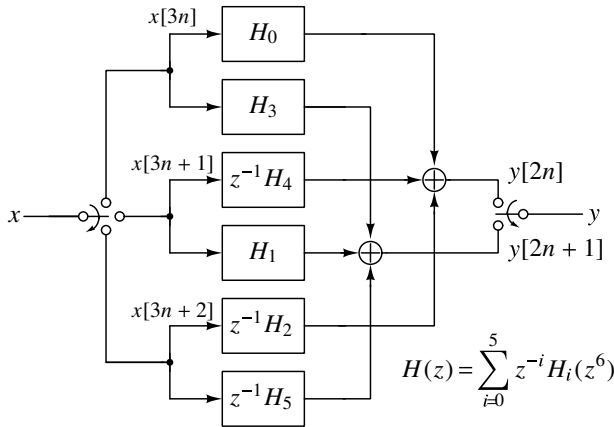

Figure 14.31   A polyphase decimate-by-1.5 arrangement.

Let's compare decimation with and without the G filter in the context of an example. Our design targets are a 90% passband with a ripple of 0.1 dB and at least 100-dB attenuation for all aliases and image terms. Code for performing the two designs is given below, and the resulting frequency responses are shown in Figure 14.32 and Figure 14.33.
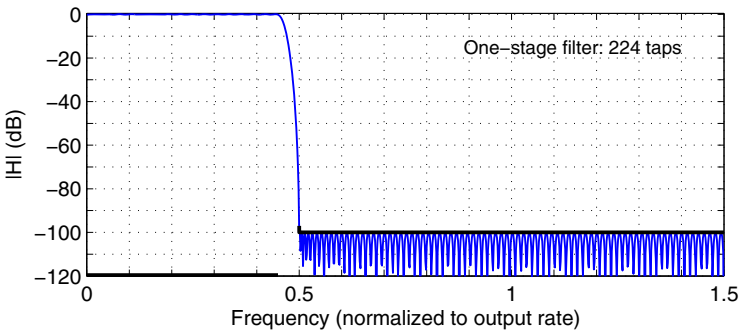
Figure 14.32   Frequency response of the single-stage decimate-by-1.5 filter.

```
%% Single-stage filter
A_min = 100;    % dB
pbr = 0.1;      % passband ripple
pbf = 0.9;      % passband fraction
f = [[0 0.5*pbf 0.5]/3 0.5];
a = [1 1 0 0];
```

[2]Hint: Confirm that applying impulses at times 0, 1, and 2 to the input of the upsampler in Figure 14.30 produces the sequences $\{h[0], h[3], h[6], h[9], \ldots\}$, $\{0, h[1], h[4], h[7], \ldots\}$ and $\{0, 0, h[2], h[5], \ldots\}$, respectively, at the y output. Then verify that the diagram in Figure 14.31 does likewise.
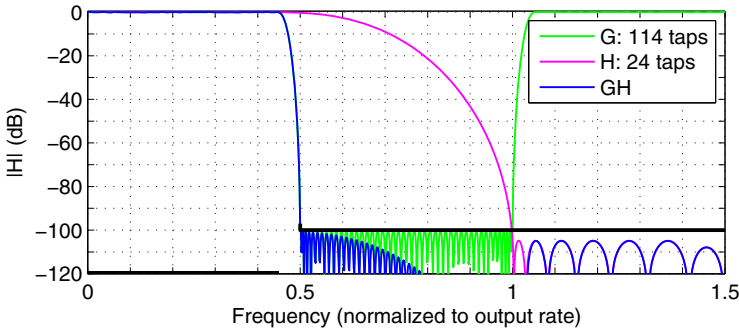
Figure 14.33   Frequency response of the cascade decimate-by-1.5 filter.

```
w = [1/(undbv(pbr)-1) undbv(A_min)];
[b0 err] = firpm(223,f*2,a,w);
%% G prefilter
k = 0.75; % fraction of pbr allocated to prefilter
f = [0 0.5*pbf 0.5 0.75]/1.5;
a = [1 1 0 0];
w = [1/(undbv(k*pbr)-1) undbv(A_min)];
[g err] = firpm(113,f*2,a,w);
%% H
f = [[0 0.5*pbf 1]/3 0.5];
a = [1 1 0 0];
w = [1/(undbv((1-k)*pbr)-1) undbv(A_min)];
[h err] = firpm(23,f*2,a,w);
```

As with previous examples, the filter orders (113 and 23) given in the listing above were determined by trial and error.

This example shows that the H filter is considerably simpler when the prefilter G is present (24 vs. 224 taps). Since the number of taps (114) in G is approximately half of that of H in the single-stage design, it would appear at first glance that the cascade design requires much less computation than the single-stage design. A closer look shows that G operates on data at the 1.5× rate, whereas H operates at effective rate of 0.5× (H is broken into six pieces, three of which are used to produce an output sample), and thus the balance is shifted in favor of the single-stage design. Looking even closer, we find that the coefficient symmetry in G halves the number of multiplications, whereas coefficient symmetry in H is less helpful because only coefficients which are shared by H_0 and H_3, or H_1 and H_4, or H_2 and H_5, offer savings. Taking all these factors into account, we find that the single-stage design requires 94 multiplications per output sample, whereas the cascade arrangement needs 95.5 and therefore conclude that including a prefilter is not helpful after all.

The preceding conclusion is valid if G is an FIR filter. We prefer FIR filters because symmetric FIR filters have linear phase and because the polyphase decomposition only works for FIR filters. However, because G is not after an up-sampler or before a down-sampler, it is unable to make use of the polyphase decomposition, and thus one of our usual reasons for preferring an FIR filter has evaporated. Designing an IIR filter as follows

```
%% IIR prefilter
[gn wp] = ellipord(pbf*0.5/1.5*2, 0.5/1.5*2, 2*k*pbr, A_min);
[gb ga] = ellip(gn, 2*k*pbr, A_min ,wp);
```

we find that an 11th-order IIR filter meets the specifications while only requiring a total of 23.5 multiplications per output sample. Therefore, if we allow G to be an IIR filter, the savings associated with prefiltering can be significant. Figure 14.34 shows the frequency response of this design.
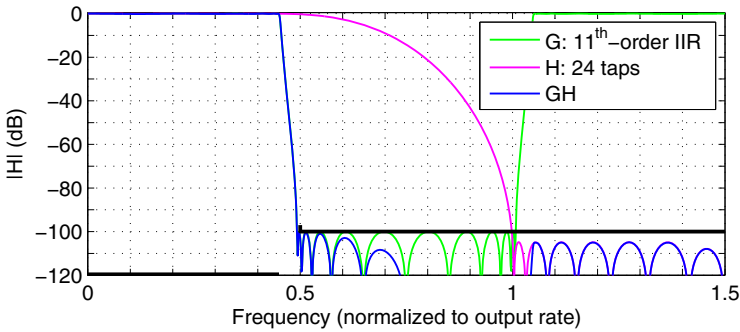

Figure 14.34   Frequency response of the decimate-by-1.5 with an IIR prefilter.

14.7.2   Sample-Rate Conversion
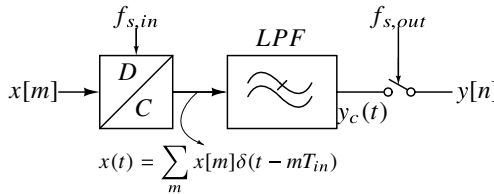
Figure 14.35   Conceptual model of a sample-rate converter.

The last topic we consider is decimation or interpolation by arbitrary factors. This operation is essential for interfacing a data converter operating at a one rate to a DSP operating at an unrelated rate, or for bridging digital systems operating asynchronously. The signal-processing model we use is depicted in Figure 14.35. In this conceptual system, the incoming sequence $x[m]$ is first converted into a continuous-time signal consisting of Dirac impulses, given by

$$x(t) = \sum_m x[m]\delta(t - mT_{in}).\tag{14.21}$$

This is then filtered by a lowpass filter whose role is to eliminate the images of $x(t)$ around multiples of $f_{s,in}$. The impulse response of the LPF is denoted by $h_c(t)$. The resulting continuous-time signal at the filter's output $(y_c(t))$ is then resampled at the output sampling rate, yielding the sequence $y[n]$. The bandwidth of the lowpass filter must be less than both
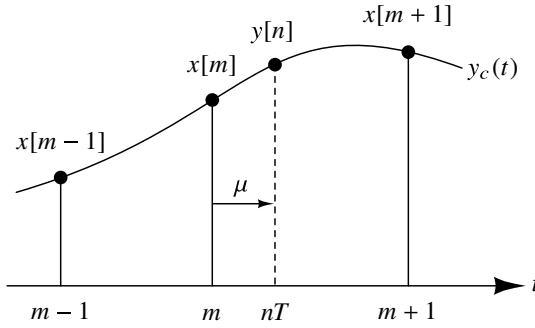
Figure 14.36   Input and output samples of the sample-rate converter.

$f_{s,in}/2$ and $f_{s,out}/2$ to prevent aliasing, and the stop band attenuation of the filter must be large enough to keep image and alias artifacts at acceptable levels.

Our goal is to mimic the operation of the system in Figure 14.35 using only digital processing. If we assume that the input sample rate is 1 Hz, then $y_c(t)$ is given by

$$y_c(t) = \sum_{m=-\infty}^{\infty} x[m]h_c(t - m), \tag{14.22}$$

and our job is to compute $y_c(t)$ at points $t = nT$, where T is the output sample period. As depicted in Figure 14.36, we define $m = \lfloor nT \rfloor$ and $\mu = nT - \lfloor nT \rfloor$, where $\lfloor nT \rfloor$ means the largest integer less than or equal nT, and thus

$$y[n] = y_c(nT) = \sum_{k=-\infty}^{\infty} x[m - k]h_c(k + \mu). \tag{14.23}$$

Now, if $h_c(k + \mu)$ and all its derivatives are continuous over the interval $[k, k + \mu]$, it can be expanded in a Taylor series around k as

$$h_c(k + \mu) = h_c(k) + \frac{dh_c(t)}{dt}\bigg|_{t=k}\mu + \frac{1}{2!}\frac{d^2 h_c(t)}{dt^2}\bigg|_{t=k}\mu^2 + \frac{1}{3!}\frac{d^3 h_c(t)}{dt^3}\bigg|_{t=k}\mu^3 + \cdots. \tag{14.24}$$

Denoting

$$h_c(k) = c_0[k],$$
$$\frac{1}{n!}\frac{d^n h_c(t)}{dt^n}\bigg|_{t=k} = c_n[k],$$

we can rewrite (14.23) as

$$y[n] = \sum_{k=-\infty}^{\infty} x[m - k]\left\{c_0(k) + \mu c_1[k] + \mu^2 c_2[k] + \cdots\right\}. \tag{14.25}$$

The resulting block diagram is shown in Figure 14.37. Thus, $y[n]$ can be determined by filtering $x[m]$ with a bank of filters with impulse responses given by $c_0[k], c_1[k], \cdots$, and weighted addition of the bank outputs. As seen earlier, the impulse response of the Mth filter in the bank is $1/M!$ times the sampled version of the Mth derivative of $h_c(t)$.
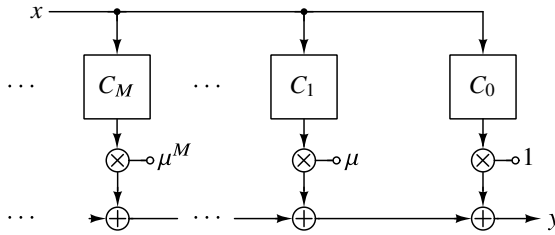
Figure 14.37 $y[n]$ can be determined by appropriately weighting the outputs of a bank of filters driven by $x[m]$.

The Taylor expansion in (14.24), in general, has infinite terms, and, it will therefore necessitate an infinite number of filters in the bank. If, however, $h_c(t)$ was chosen to be an Mth-order polynomial in t, the RHS of (14.24) would have only $(M + 1)$ terms. Further, because we are only interested in evaluating $h_c(k + \mu)$ for $\mu \in [0, 1)$, it is sufficient if $h_c(t)$ is a polynomial piecewise. For example, $h_c(t)$ could be expressed as an Mth-order polynomial for $0 \leq t < 1$, and a *different* Mth-order polynomial for $1 \leq t < 2$. This increased freedom can only be expected to help with the design of $h_c(t)$.
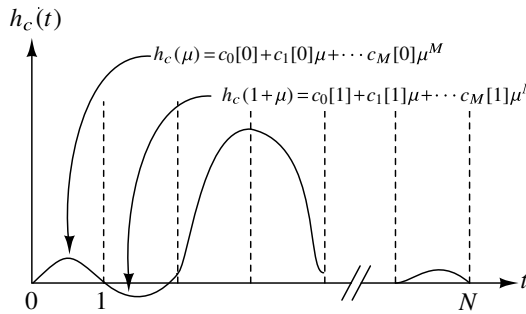
Figure 14.38 $h_c(t)$ constructed with polynomial fragments.

Further, to simplify computation, we restrict $h_c(t)$ to last for N periods of the input clock. Since $h_c(t)$ is also polynomial piecewise, it follows that all its derivatives must also last for N input clock cycles. Since the taps of all the filters in the bank of Figure 14.37 depend on the sampled derivatives of $h(t)$, it follows that these filters will have N taps.

Inspection of Figure 14.37 shows that $h_c(t)$, which consists of N unit-width Mth-order polynomial fragments spanning $[0, N]$, can be expressed as

$$h_c(k + \mu) = \begin{cases} c_0[k] + c_1[k]\mu + \ldots c_M[k]\mu^M, & 0 \leq k < N \\ 0, & k \geq N. \end{cases} \tag{14.26}$$

A sample $h_c(t)$ is illustrated in Figure 14.38. Restrictions on the $c_i[k]$ coefficients can result from such requirements as symmetry (for flat group delay), threading the $x[m]$ sample points, continuity, and smoothness. However, in the examples that follow, we only require symmetry and continuity.

The block diagram of Figure 14.37 can be equivalently recast as that shown in Figure 14.39. This structure, called the Farrow filter [7], provides an efficient implementation
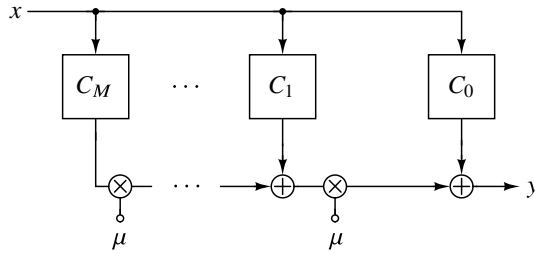
Figure 14.39   The Farrow structure.

of (14.23). Recall that, as in Figure 14.37, the $(M + 1)$ blocks $C_0 \ldots C_M$ are N-tap FIR filters. The impulse response of filter C_i is $\{c_i[0], c_i[1] \ldots c_i[N-1]\}$. Note that these $(M + 1)$ filters operate on the same input sequence x and thus can use the same sample memory. In a processor-based implementation, it is efficient to write the input samples to a circular buffer in a RAM and then read the samples out and perform the filter computations at the lesser of the input and output rates. Of course, the multiplications by μ need to occur at the output rate. In a hardware implementation, the multiplications in the FIR filters can be hardwired, because the coefficients are known, but the multiplications by μ require true multipliers.
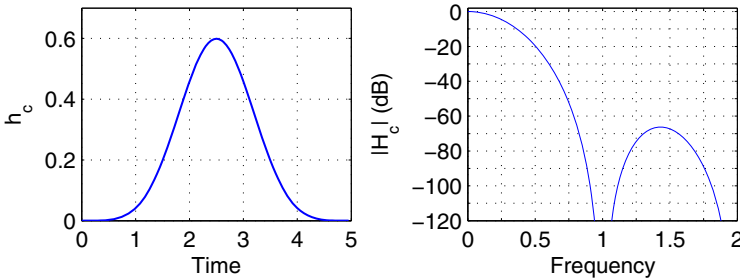
Figure 14.40   (a) $h_c(t)$ obtained by repeated convolution of a unit rectangle. (b) Fourier transform.

Figure 14.40 shows an example $h_c(t)$ obtained by convolving the unit rectangle with itself four times. The coefficients of the resulting five fourth-order segments can be conveniently be assembled into a 5×5 matrix

$$C = \frac{1}{24} \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 1 & 4 & 6 & 4 & -4 \\ 11 & 12 & -6 & -12 & 6 \\ 11 & -12 & -6 & 12 & -4 \\ 1 & -4 & 6 & -4 & 1 \end{bmatrix}. \tag{14.27}$$

The rows of this matrix are the polynomial coefficients for each segment and the columns are the coefficients of the FIR filters. For example, the polynomial associated with the second segment is

$$h_c(1 + \mu) = \frac{1}{24} + \frac{\mu}{6} + \frac{\mu^2}{4} + \frac{\mu^3}{6} - \frac{\mu^4}{6}, \tag{14.28}$$

and the impulse response of the C_1 filter (which implements the μ^1 polynomial term for all five segments) is

$$\{c_1[k]; \; k = 0, 1, 2, 3, 4\} = \left\{0, \frac{1}{6}, \frac{1}{2}, -\frac{1}{2}, -\frac{1}{6}\right\}. \tag{14.29}$$

Since scaling the coefficients by a factor of 24 yields simple integers, the multiplications in the FIR filters can be implemented with a small number of additions. (Scaling by 1/24 can be performed at either the input or output.) Unfortunately, as the Fourier transform plotted in Figure 14.40 shows, the image/alias attenuation provided by this filter is modest and the droop is large unless the input is oversampled. Droop can be compensated with a pre-filter, but if 100 dB of alias protection is needed, the input must be oversampled by at least a factor of 5.5.
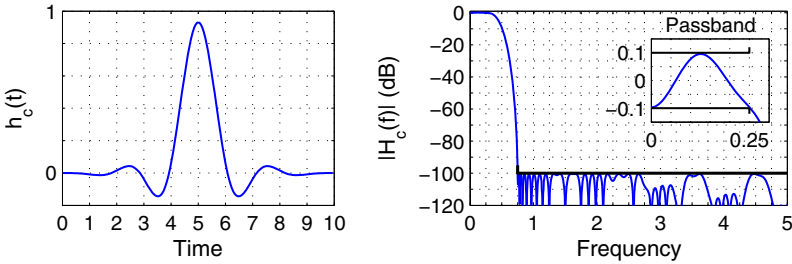

Figure 14.41   (a) $h_C(t)$ obtained by `designPBF`. (b) Fourier transform.

Coefficients yielding more aggressive filtering can be found via the $\Delta\Sigma$ toolbox function `designPBF`, which follows Hunter's method [8]. Figure 14.41 shows the impulse response of a 10-segment fifth-order polynomial-based filter and its associated Fourier transform. This filter provides 100 dB of image attenuation with 0.1 dB of passband ripple if the input oversampling ratio is only two. The C matrix for this filter is given below.

```
%C matrix obtained by designPBF.
% Use mu-0.5 as the polynomial argument.
C = [
-0.001345 -0.007276 -0.013868 -0.007952  0.008608  0.011467
-0.012460  0.016669  0.074003  0.038627 -0.042437 -0.039262
 0.042131 -0.025342 -0.246761 -0.140367  0.134082  0.118809
-0.144527 -0.015873  0.677698  0.517909 -0.219522 -0.254117
 0.610687  1.112924 -0.491399 -1.063063  0.120475  0.375982
 0.610687 -1.112924 -0.491399  1.063063  0.120475 -0.375982
-0.144527  0.015873  0.677698 -0.517909 -0.219522  0.254117
 0.042131  0.025342 -0.246761  0.140367  0.134082 -0.118809
-0.012460 -0.016669  0.074003 -0.038627 -0.042437  0.039262
-0.001345  0.007276 -0.013868  0.007952  0.008608 -0.011467
];
```

If the ratio T of the output sample rate to the input sample rate is known exactly, then the Farrow structure only needs to be augmented with a block that determines m and μ for each output sample. If the ratio is not known exactly, or if the input and output clocks are asynchronous, then a block that estimates T is needed, as indicated in Figure 14.42. In this system, the incoming data are written into the sample memory using a write port on
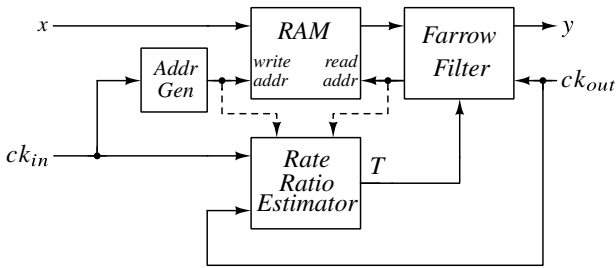
Figure 14.42   The asynchronous sample-rate converter.

the RAM while a Farrow filter reads samples out of the read port. The rate-ratio estimator supplies the Farrow filter with an estimate of T in order to determine m and μ for each output sample. T must be updated sufficiently often to track changes in the input and output frequencies but otherwise should be heavily filtered. To prevent the read address from overrunning the write address, the rate-ratio estimator can use the write and base read addresses as part of the estimation process.

14.8 Summary

In this chapter, we examined interpolation for $\Delta\Sigma$ DAC systems and decimation for $\Delta\Sigma$ ADC systems. We observed numerous parallels between the two operations, and noted that a decimator can be converted into an interpolator and vice versa. We found that when the oversampling ratio before interpolation or after decimation is 2 or more, a cascade of sinc filters combined with equalization yields an efficient system. Decimating down to, or interpolating up from, an oversampling ratio close to 1 is a computationally intensive operation that is most efficiently performed by a halfband filter. The Saramäki topology implements a halfband filter and allows the coefficients to be quantized to a few CSD terms. We closed with abbreviated discussions of decimation and interpolation by fractional factors and asynchronous sample-rate conversion using the Farrow filter structure.

References

[1] E. Hogenauer, "An economical class of digital filters for decimation and interpolation," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 29, no. 2, pp.155–162, Apr. 1981.

[2] T. Saramäki, "Design of FIR filters as a tapped cascaded interconnection of identical subfilters," *IEEE Transactions on Circuits and Systems*, vol. 34, no. 9, pp. 1011–1029, Sep. 1987.

[3] R. E. Crochiere and L. Rabiner, "Interpolation and decimation of digital signals– A tutorial review," *Proceedings of the IEEE*, vol. 69, no. 3, pp. 300-331, Mar. 1981.

[4] R. E. Crochiere and L. R. Rabiner, *Multirate Digital Signal Processing*. Prentice-Hall, Englewood Cliffs, 1983.

[5] A. Y. Kwentus, Z. Jiang, and A. N. Wilson, "Application of filter sharpening to cascaded integrator-comb decimation filters," *IEEE Transactions on Signal Processing*, vol. 45, no. 2, pp. 457–467, 1997.

[6] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing.* Prentice-Hall, Engle-wood Cliffs, 1989.

[7] C. W. Farrow, "A continuously variable digital delay element," *IEEE International Symposium on Circuits and Systems*, pp. 2641-2645, June 1988.

[8] M. T. Hunter, "Design of polynomial-based filters for continuously variable sample rate conversion with applications in synthetic instrumentation and software defined radio," *Ph.D. thesis*, University of Florida, 2008.

APPENDIX A

SPECTRAL ESTIMATION

The purpose of this appendix is to demystify the procedure of analyzing $\Delta\Sigma$ data using the fast Fourier transform (FFT) [1]. The FFT is widely used to estimate the power spectral density of $\Delta\Sigma$ data, but, it is also sometimes abused in the process. When using the FFT to analyze $\Delta\Sigma$ data, the $\Delta\Sigma$ designer needs to be familiar with several important concepts, namely windowing, scaling, noise bandwidth, and averaging. This appendix deals with each of these subjects in turn, applies them to an example, and concludes with a brief discussion of the mathematical background.

The FFT is a fast algorithm for computing the Fourier transform

$$X[f] = \sum_{n=0}^{N-1} x[n] \exp\left(-j2\pi f n\right) \tag{A.1}$$

of a length-N discrete-time sequence $x[n]$ at the N frequency points, the *FFT bins:* $0, 1/N, 2/N, \cdots, (N-1)/N$.[1] A discrete-time signal with period N consists of a dc term and harmonics of the fundamental frequency $f_1 = 1/N$. $|X[f]|$ of a sinusoidal sequence

$$x[n] = A \cos\left(2\pi \frac{i}{N} n + \phi\right), \tag{A.2}$$

---

[1] Here, as earlier, the sampling rate is assumed to be 1 Hz. Further, $X[f]$ is a sequence, since f takes on discrete values.

where $i \neq 0$ and $i \neq N/2$ is given by

$$X[k] = \begin{cases} \frac{A}{2}N, & k = i, \\ 0, & \text{otherwise.} \end{cases}$$

The amplitude of the ith harmonic of a period-N sequence is therefore given by $2|X[f_i]|/N$; so the FFT can easily be used to compute the power spectrum of a periodic signal. Unfortunately, since $\Delta\Sigma$ data is typically not periodic, a direct application of the FFT to $\Delta\Sigma$ data is unwise at best.

We will consider the "noise" associated with $\Delta\Sigma$ data to be like a random signal, for which a more technical term is *stochastic process* [2]. If the data comes from measurements, then it is bound to contain components that are true noise and our viewpoint is justified. For data obtained from simulation, the noise is the result of a *deterministic* process, and so it is not strictly proper to describe this process as random. However, since the process is complex, nonlinear, and often chaotic, the fact that the process is actually deterministic has little practical impact.

A.1   Windowing

Windowing is the act of multiplying the signal to be analyzed by a *window function* $w[n]$ before subjecting it to an FFT. At first glance, it would appear that this operation would alter the spectral content of the signal and therefore be undesirable. Although it *is* true that windowing alters a signal's spectrum, some windowing is inevitable because we can never obtain an infinite-length record of modulator data. The best we can do is operate on a finite record of length N. Since a finite-length record can be thought of as the product of the infinite-duration modulator output and a rectangular window

$$w_{rect}[n] = \begin{cases} 1, & 0 \leq n \leq (N-1), \\ 0, & \text{otherwise,} \end{cases} \tag{A.3}$$

the damage caused by windowing the data has already been done. Thus the question is not "Should I window my data?" but rather "*How* should I window my data?"

The answer to this question lies in the relationship between the spectrum of the original data and that of the windowed data. Since multiplication in the time domain corresponds to convolution in the frequency domain, the spectrum of a windowed signal is, loosely speaking, the spectrum of the unwindowed signal convolved with the window's spectrum. In order to obtain an accurate spectrum, the designer must choose a window that introduces sufficiently low errors through spectral convolution.

Consider the three windowing functions illustrated in Figure A.1. Table. A.1 lists their definitions and summarizes various parameters discussed in this Appendix. Since a rectangular window has discontinuities at its endpoints, whereas the Hann and Hann^2 windows are continuous (up to the second and fourth derivatives, respectively), we would expect the rectangular window to have a great deal more high-frequency content than the other two windows. This suspicion is confirmed in Figure A.2, which plots the magnitudes of the Fourier transform

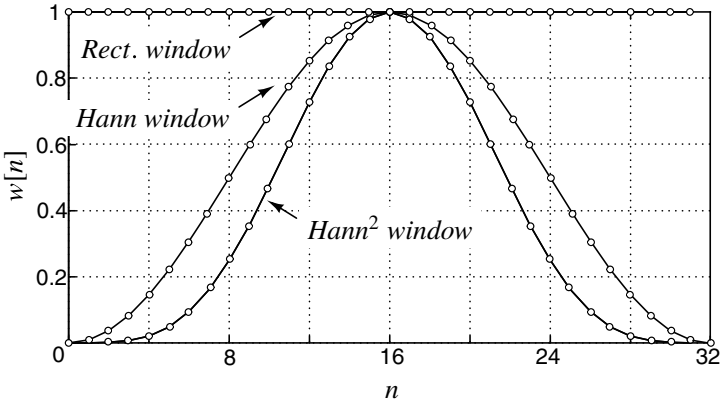$$W(f) = \sum_{n=0}^{N-1} w[n] \exp(-j2\pi f n) \tag{A.4}$$

Figure A.1   The rectangular, Hann and Hann^2 windows.

Window	Rectangular	Hann	Hann^2
$w[n], n = 0, 1, \cdots, N-1$	1	$\frac{1}{2}\left[1 - \cos\left(\frac{2\pi n}{N}\right)\right]$	$\frac{1}{4}\left[1 - \cos\left(\frac{2\pi n}{N}\right)\right]^2$
$\|w_2\|^2$	N	$\frac{3}{8}N$	$\frac{35}{128}N$
No. of nonzero FFT bins	1	3	5
$W[0]$	N	$\frac{1}{2}N$	$\frac{3}{8}N$
NBW	$\frac{1}{N}$	$\frac{3}{2N}$	$\frac{35}{18N}$

Table A.1   Properties of the three windows illustrated in Figure A.1.

normalized by the dc gain $W(0)$, for each of these windows. $N = 32$ points were used.

As Figure A.2 shows, the peaks of the high-frequency lobes in the spectrum of the rectangular window approach a constant value (which is proportional to $1/N$), whereas the peaks of the high-frequency lobes of the Hann and Hann^2 windows go to zero with -60 dB/decade and -100 dB/decade slopes, respectively. The high-frequency behavior of a window's spectrum is of critical importance in determining the magnitude of the error resulting from convolution.
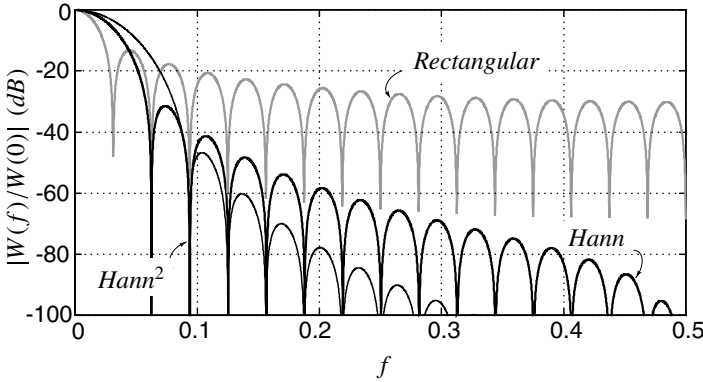
Figure A.2   Fourier transform magnitudes of the rectangular, Hann and Hann^2 windows.

As a demonstration of the convolution problem, Figure A.3 shows the spectrum of some noise-shaped data, the Fourier transform of a 256-point rectangular window, and the Fourier transform of the windowed data. As indicated in Figure A.3, the skirts of the window convolve with the out-of-band noise, thereby filling in the noise notch and dramatically reducing the apparent SNR.
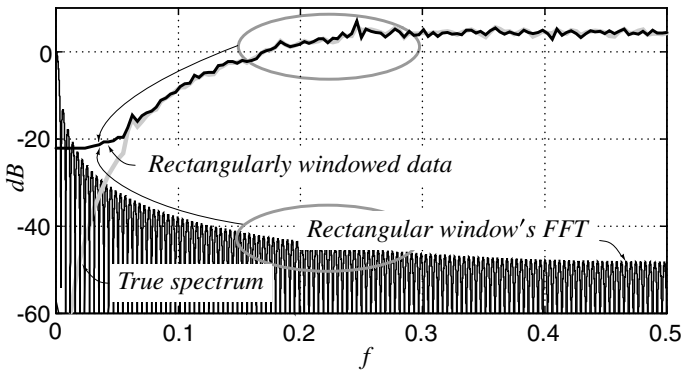
Figure A.3   A rectangular window obscures a noise null.

The $\Delta\Sigma$ designer must ensure that this *noise leakage* is small compared to the actual in-band noise density. In the context of a high-accuracy $\Delta\Sigma$ modulator, the difference

between the out-of-band noise density and the in-band noise density can be 80 dB or more. Figure A.3 indicates that with $N = 256$, the observable difference between the out-of-band and in-band noise densities is only about 23 dB. Increasing N improves the situation, but only at the rate of 3 dB per octave. According to this trend, a rectangular window would need to use more than 10^8 points to reliably observe an 80-dB difference in noise densities!

The $\Delta\Sigma$ designer is therefore compelled to use something other than a simple rectangular window. Many different windows exist (e.g., see [3] or [4]), but the feature of greatest importance to the $\Delta\Sigma$ designer is the amount of high-frequency attenuation provided by the window. Windows that have finite high-frequency attenuation, such as the Hamming window, are less desirable than windows whose high-frequency attenuation increases without bound. In particular, a Hann window with $N = 512$ is able to resolve an 80-dB difference in noise density, while a Hann^2 window does similarly well with $N = 256$. Since the number of data points needed to provide sufficient frequency resolution is usually on the order of several thousand, a simple Hann window usually provides sufficient protection against noise leakage.
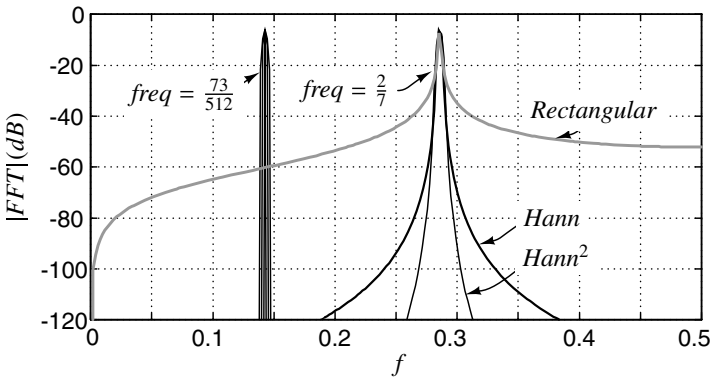
Figure A.4   FFTs of cohorent and incohorent sine waves ($N = 512$).

Another important consideration in the analysis of $\Delta\Sigma$ data is *signal leakage*. It is convenient, both in the lab and in simulation, to use sine-wave excitation. However, the frequency of that sine wave must be located precisely in an FFT bin; otherwise, signal power will bleed into all bins. Figure A.4 illustrates this phenomenon with several length-512 FFTs of two sine waves. The first has a frequency that is located precisely in an FFT bin (specifically, bin 73, which is close to a frequency of 1/7), while the second has a frequency of exactly 2/7 and so is not in an FFT bin. In the first case (the coherent case), the sine wave's power is concentrated in a small number of FFT bins (1 for a rectangular windowing, 3 for Hann, and 5 for Hann^2). In the *incoherent* case, the sine wave's power is smeared over all FFT bins. The severity of the spreading is determined by how far away the sine wave is from the nearest bin frequency, and by the shape of the window's skirts. As was the case with noise leakage, the rectangular window exhibits the greatest signal leakage because its skirts are the broadest.

In simulation, it is a simple matter to place the signal frequency in an FFT bin and thereby eliminate signal leakage entirely. The reader is cautioned to use an accurate value of π when computing the samples of a signal $x[n] = \cos(2\pi f n/N)$. With a Hann window,

rounding π to 4 decimal creates skirts on the signal whose total power is -84 dBc; with a rectangular window, the skirt power is 18 dB higher. In the lab, signal leakage can be minimized by phase-locking the generators, and by setting the signal frequency accurately. If it is not possible to place the signal frequency in an FFT bin, windowing can be used to reduce spectral pollution. Alternatively, the signal's frequency, amplitude and phase can be estimated, and the estimated signal can be subtracted from the data record to leave only the noise.

The final question regarding windowing that we will address is the length of the window required to obtain an accurate estimate of a modulator's SNR. The simplest way to estimate SNR is to compute the ratio of the power in the signal bins to the power in the in-band noise bins. Since the signal is usually placed in-band, it therefore occupies a few of the in-band bins. The number of bins occupied by the signal should be a relatively small fraction (less than 20%) of the in-band bins in order to have a small effect (less than 1 dB) on the SNR estimate. If we are using a Hann window, the signal will occupy 3 bins, and thus we should have at least 15 in-band bins, which in turn requires $N \geq (30 \cdot OSR)$ bins in all, where OSR is the oversampling ratio.

If the noise is assumed to be flat in-band, the missing noise power can be accounted for by multiplying the power in the noise bins by $1/(1 - a)$, where a is the fraction of in-band bins occupied by the signal. Unfortunately, at least in simulation, the in-band quantization noise follows the *NTF* and so tends not to be flat. Alternatively, the noise in the signal bins can be estimated by subtracting out the estimated signal component before or after performing the FFT.

Another consideration regarding the required length of the data record relates to the repeatability of the SNR measurement. Since the FFT of a random signal is itself a random quantity, the in-band noise power computed from the FFT is also a random quantity. Numerical experiments indicate that using $N = 30 \cdot OSR$ results in SNR estimates that have a standard deviation of about 1.4 dB. Using $N = 64 \cdot OSR$ results in a standard deviation of about 1.0 dB, while $N = 256 \cdot OSR$ is needed to reduce the standard deviation to 0.5 dB. This degree of repeatability is usually not required on an individual measurement, since it is common for many measurements to be carried out, as would be the case during an input amplitude sweep. The authors recommend using $N = 64 \cdot OSR$.

A tertiary consideration regarding the required length of the data record relates to the observable spurious-free dynamic range (SFDR). Using $N = 64 \cdot OSR$ is usually sufficient to reliably observe an SFDR that is about 10 dB greater than the SNR. In order to detect tones that are more than 10 dB below the total in-band noise, N needs to be increased. Specifically, doubling N increases the observable SFDR by 3 dB.

A.2   Scaling and Noise Bandwidth

The spectral spike associated with a sine-wave component in the data record has a height that is dependent on both the window type and the window length. Most windows have $\max|W(f)| = W(0)$ (i.e., the peak in the window's spectrum occurs at dc), and so the height of the peak corresponding to a sine wave is $(A/2)W(0)$, where A is the amplitude of the sine wave.

When displaying a spectral plot, it is customary to scale the FFT such that a full-scale sine wave yields a 0-dB spectral peak. If we denote the full-scale range by FS, the amplitude of a full-scale sine wave is $A = FS/2$ and thus the scaled version of the FFT that we would present is

$$\hat{S}'_x(f) = \left| \frac{1}{(FS/4)W(0)} \sum_{n=0}^{N-1} w[n] \exp(-j2\pi f n) \right|^2. \tag{A.5}$$

Note that we have squared the magnitude in order to allow us to interpret $\hat{S}'_x(f)$ as a power spectral density (PSD). The power of a sine-wave signal relative to the power of a full-scale sine-wave signal is therefore given by $10 \log \hat{S}'_x(f)$, where f is the frequency of the signal. The units of this quantity are often given as dBFS (dB relative to full-scale) in order to emphasize that the reference power is the power of a full-scale sine wave, but we will see shortly that these units omit an important detail. The caret on the symbol $\hat{S}'_x(f)$ indicates that the expression above is an estimate of the PSD, while the prime indicates that we have scaled the estimate so that sine-wave signals yield calibrated spike heights.
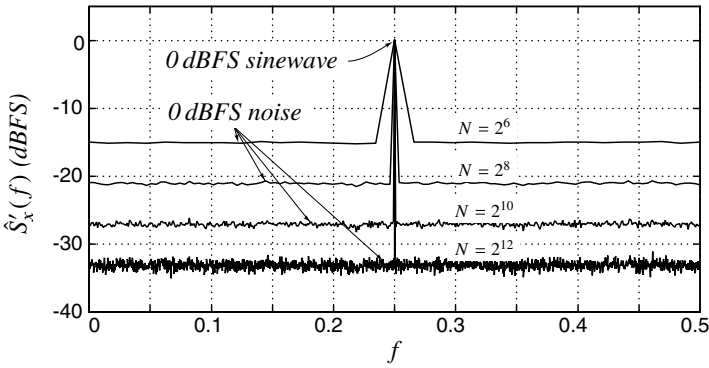
Figure A.5   FFTs of a sine wave plus white noise; sine-wave scaling.

Although the scaling above is convenient for analyzing a signal that consists of sine waves, it is less convenient for analyzing a signal that contains noise. Figure A.5 illustrates the problem by plotting $\hat{S}'_x(f)$ for a signal consisting of a 0-dBFS sine wave plus white noise having the same power as the sine wave, when different-length rectangular windows are used. (A rectangular window is safe here, since we are not interested in observing notches in $\hat{S}'_x(f)$.) Although the signal spike has the same height in each case, the average "noise floor" of $\hat{S}'_x(f)$ drops by 3 dB every time N doubles. Since the noise power is actually 0 dBFS in each case, the location of the "noise floor" is not the only piece of information that the reader needs.

The problem with sine-wave scaling is that the noise power is, on average, evenly distributed over all FFT bins, whereas the sine-wave power is concentrated in only a few bins. With sine-wave scaling, the power of individual sine-wave components can be read directly from the spectral plot, but in order to determine the noise power, the powers of all the noise bins must be added together.

An alternative method of scaling, that is common in signal-processing texts, is to scale the FFT such that it provides a calibrated noise density. The appropriate scale factor in this

case is $1/\|w\|_2^2$, where

$$\|w\|_2^2 = \sum_{n=0}^{N-1} |w[n]|^2 \tag{A.6}$$

is the energy of the window.[2] When this scaling is used, the PSD estimate is

$$\hat{S}_x'(f) = \left| \frac{1}{\|w\|_2} \sum_{n=0}^{N-1} w[n] \exp(-j2\pi f n) \right|^2 \tag{A.7}$$

and as a result unit-power white noise yields a unit (0 dB) density, regardless of the window type or length. Unfortunately, scaling for noise in this way makes the height of a sine-wave spike dependent on both the window type and length.

Our resolution of the scaling dilemma follows the solution adopted in laboratory instruments, specifically in spectrum analyzers. A spectrum analyzer must contend with the problem of representing the *spectrum* of a periodic signal such as a sine wave on the same display as is used to represent the *spectral density* of a broadband signal such as noise. In a spectrum analyzer, the signal can be thought of as being processed by a bank of filters possessing identical filtering characteristics (gain, bandwidth, etc.), albeit with different center frequencies. The instrument measures the powers at the outputs of the filters and constructs a plot of power versus center frequency.

For a sine-wave input, the display shows a peak at the input frequency, the height of which equals the input power. Thus, the spectrum displayed by a spectrum analyzer is like an FFT with sine-wave scaling. For noise, the display indicates the power of the noise that lies in each filter's bandwidth. In other words, for a noise-like signal, the display gives the product of the noise density and the *noise bandwidth* of the filters.

For a filter with infinitely steep roll-off, the noise bandwidth (NBW) is equal to the filter's bandwidth, while for a filter with a single-pole roll-off, NBW is $\frac{\pi}{2}$ times the 3-dB bandwidth. In general, a filter's NBW is the bandwidth of an ideal brick-wall filter that has the same output power given a white noise input and that has the same mid-band gain as the filter under consideration. A spectrum analyzer solves the scaling problem by providing NBW along with each spectrum, thereby providing the designer with the information needed to convert the power shown on the display into a power density. NBW depends on various analyzer settings, including those that determine the frequency range and resolution for the spectral plot.

The solution to the scaling problem in the case of a PSD obtained from a sine-wave scaled FFT is similarly simple. All we need do is provide the value of NBW.

The value of NBW is listed in Table. A.1 for the three windows considered in this appendix. For each window, NBW is inversely proportional to N, and thus doubling N reduces the apparent level of the noise in a sine-wave scaled PSD by 3 dB, as indicated in Figure A.5. Let us now illustrate the use of NBW in the calculation of the total noise power. From the top curve in Figure A.5, we observe a "noise floor" of approximately -15 dBFS, which is the amount of power in bandwidth $NBW = 1/N = 2^{-6}$. The total

---

[2]The notation $\|w\|_2$ means 2-*norm*, a special case of the *p-norm* $\|w\|_p = \left[\sum_{n=0}^{N-1} |w[n]|^p \right]^{\frac{1}{p}}$.

power in the entire Nyquist band $[0, 0.5]$ is therefore $0.5/NBW = 2^5$ times (15 dB more than) the observed $-15\,\text{dBFS}$ value, or $0\,\text{dBFS}$, which is exactly the correct value.

The NBW, or at least sufficient information for calculating it (i.e., N and the window type), should always be given for a sine-wave-scaled FFT. Furthermore, in order to emphasize that such an FFT represents a power spectral density, the units on the vertical axis should be shown as power per unit bandwidth. Since we report power in dBFS and since the unit of bandwidth is NBW, the vertical axis is usually labeled "dBFS/NBW."[3]

A.3 Averaging

Our final point of discussion regarding the use of FFTs for spectral analysis is averaging. We noted earlier that the FFT of a random waveform is itself a random quantity. This quantity is random in both magnitude and phase, but since we are concerned with power we will only consider the magnitude. The magnitude of a particular frequency bin in an FFT of a random signal is a random value that has both a mean and a standard deviation. It turns out that the expected value[4] of the FFT magnitude equals the actual PSD (convolved with the window), as we would hope, but that the standard deviation of the magnitude is large. In fact, the standard deviation is equal to the expected value!
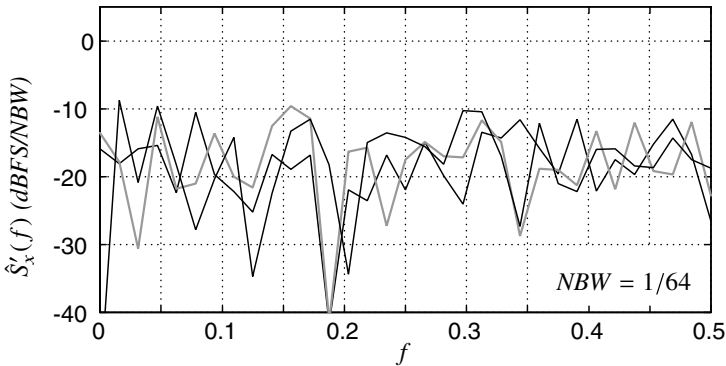
Figure A.6   Three length-64 FFTs of 0-dBFS white noise.

The upshot of this property is that a single FFT results in a "noisy" spectral estimate, as demonstrated in Figure A.6, where three length-64 FFTs of 0-dBFS white noise are shown. Each curve is expected to be a flat line at $-15\,\text{dBFS}$, but this is not apparent in the figure due to the large degree of variability in the individual bin magnitudes.

If one is computing a noise power (by summing the powers over a range of FFT bins), the variability of an individual bin magnitude is not problematic, provided that a sufficient number of bins are used. However, if one is trying to construct a clear graph of a spectral

---

[3]Although in common usage, units such as dBFS/NBW or dBm/Hz are deceptive: a density of 1 dBm/Hz does yield 1 dBm of power in a 1 Hz bandwidth but not 2 dBm of power in a 2 Hz bandwidth! Doubling the bandwidth doubles the power (a 3-dB increase), yielding a 4 dBm power in a 2 Hz bandwidth. The way out of this notational morass is to interpret dBm/Hz to mean dB with respect to a 1 mW per Hz density. Similarly, dBFS/NBW means dB with respect to a density equivalent to the power of a full-scale sine wave spread over a bandwidth NBW.

[4]*Expected value* is another term for mean, or average, value.
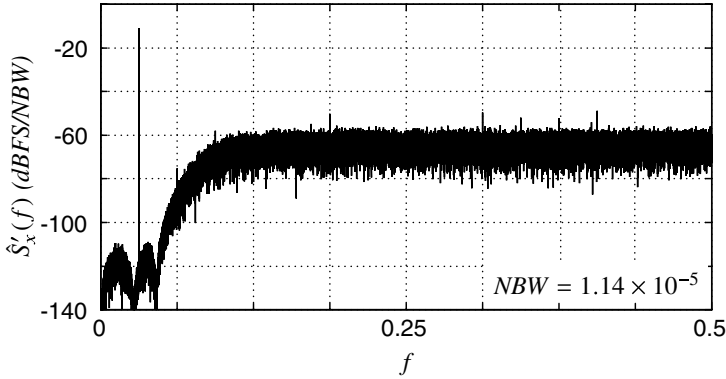
Figure A.7 A length 2^{17} FFT without averaging.

density, the erratic nature of the bin magnitudes results in a fuzzy plot such as that shown in Figure A.7. Here, instead of a smooth curve, we see a broad black band spanning nearly 20 dB that obscures the true noise density. There are two solutions to this problem, namely averaging and integration. Averaging can be performed by either averaging many FFTs, or by averaging nearby bins in a single FFT.

Averaging multiple FFTs requires the use of multiple data records. When a single large data record is available, it can be partitioned into many (possibly overlapping) records that are individually windowed and transformed with an FFT. Averaging the squared magnitudes of these FFTs reduces the standard deviation and so improves the legibility of the spectral plot. (At this point, we owe the reader a confession: the spectra shown in Figures A.3 and A.7 were averaged in this way for the sake of clarity.) Similarly, averaging the powers in adjacent bins will also smooth the FFT, effectively by "filtering" it. We present an example of this form of averaging in the next section.
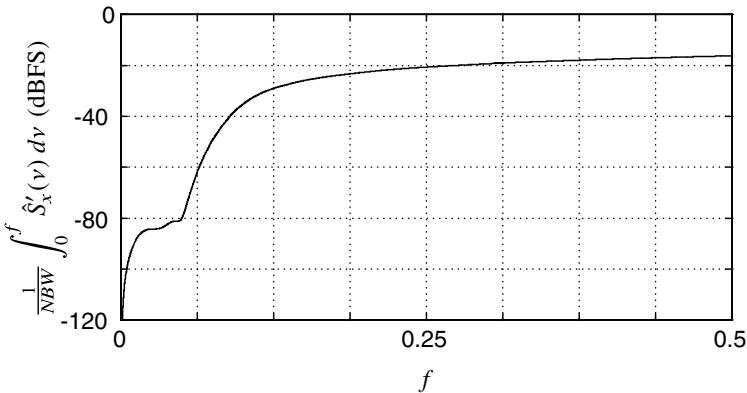
Figure A.8 Integrated version of the noise spectrum in Figure A.7.

The second method for increasing the legibility of a spectrum is to plot its accumulated value, scaled by $1/(N \cdot NBW)$. The resulting plot shows the amount of power contained in the band from dc up to the current frequency, and so saves the reader the effort

of performing an integration to obtain a noise power. Of course, the designer is obliged to empty the signal bins before performing the integration, and the method must be changed when dealing with bandpass noise-shaping. Figure A.8 illustrates the effectiveness of the technique in smoothing out the FFT of Figure A.7. (Some might argue that the technique is too effective, since it hides high-frequency tones that are readily apparent in the original FFT.)

A.4   An Example

We have now covered the essential topics regarding the use of the FFT to calculate a sine-wave scaled PSD, and so are ready to demonstrate them with an example. An example MATLAB code for producing a record of $\Delta\Sigma$ data, and analyzing it according to the procedures above is given below.

```
% Compute modulator output and actual NTF
%
OSR = 32;
ntf0 = synthesizeNTF(5,OSR,1);
N = 64*OSR;
fbin = 11;
u = 1/2*sin(2*pi*fbin/N*[0:N-1]);
[v tmp1 tmp2 y] = simulateDSM(u,ntf0);
k = mean(abs(y)/mean(y.^2))
ntf = ntf0 / (k + (1-k)*ntf0);
%
% Compute windowed FFT and NBW
%
w = hann(N); % or ones(1,N) or hann(N).^2
nb = 3; % 1 for Rect; 5 for Hann^2
w1 = norm(w,1);
w2 = norm(w,2);
NBW = (w2/w1)^2
V = fft(w.*v)/(w1/2);
%
% Compute SNR
%
signal_bins = fbin + [-(nb-1)/2:(nb-1)/2];
inband_bins = 0:N/(2*OSR);
noise_bins = setdiff(inband_bins,signal_bins);
snr = dbp(sum(abs(V(signal_bins+1)).^2)/sum(abs(V(noise_bins+1)).^2
%
% Make plots
%
figure(1); clf;
semilogx([1:N/2]/N,dbv(V(2:N/2+1)),'b','Linewidth',1);
hold on;
[f p] = logsmooth(V,fbin,2,nb);
```

```
plot(f,p,'m','Linewidth',1.5)
Sq = 4/3 * evalTF(ntf,exp(2i*pi*f)).^2;
plot(f,dbp(Sq*NBW),'k--','Linewidth',1)
figureMagic([1/N 0.5],[],[], [-140 0],10,2);
```

The first block of code synthesizes a fifth-order NTF, creates the binary $\Delta\Sigma$ data, estimates the quantizer gain and computes the actual NTF. The second block of code computes the scaled and windowed FFT, as well as NBW. The last two code blocks compute the SNR and create the plots shown in Figure A.9.

The NTF has zeros optimized for an oversampling ratio $OSR = 32$. The number of FFT points is set to $N = 64 \cdot OSR = 2048$, as advocated in Section A.1. Note that the half-scale input signal is placed precisely in an FFT bin (specifically, bin 11). Many practitioners use an odd FFT bin to ensure that the input data contains no repeated segments, but this is not strictly necessary. In a Nyquist converter, it is wise to use an input that contains no repeated segments in order to exercise as many codes as possible, but in a $\Delta\Sigma$ converter the internal state of the converter typically ensures that the output data is not periodic.

As indicated in the second code block, different windows can be tried. The Hann window is used here. The number of nonzero signal bins for the Hann window is $nb = 3$. NBW is calculated directly for the window, using an expression that will be justified in the next section. As indicated in Table A.1, the result for the Hann window is $NBW = 1.5/N = 7.3 \times 10^{-4}$. The last line of the second code block calculates the FFT, and scales it by half the dc gain of the window to perform sine-wave scaling.

In the third code block, the SNR is calculated as the ratio of the total power in the signal bins to the total power in the noise bins, and the result for this simulation is SNR=81 dB. If a rectangular window were used instead, the poor high-frequency roll-off of the window would corrupt the in-band portion of the spectrum and yield a lower SNR. For this particular example, the shortfall was a disastrous 23 dB.
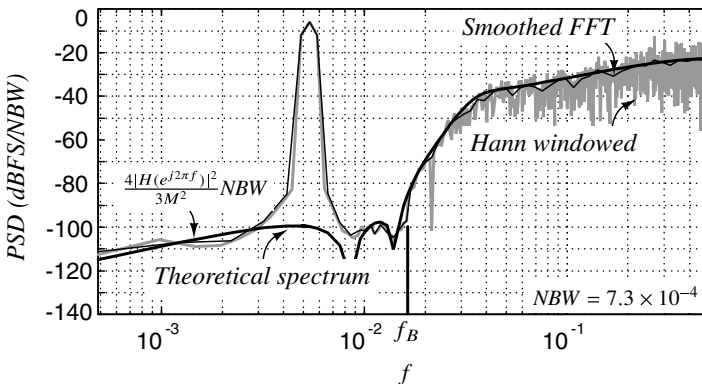
Figure A.9 Example PSDs produced by the MATLAB code fragment.

In the fourth and final code block, the raw and smoothed PSD estimates are compared graphically with the theoretical PSD. As expected, the raw FFT produces an erratic graph from which it would be difficult to tell if the observed PSD follows the expected PSD or not. Smoothing is performed with the $\Delta\Sigma$ toolbox function `logsmooth`. This function

averages the power across a number of bins in order to reduce the variance of the PSD, and also subsamples the result to yield points that are spaced fairly evenly on a logarithmic axis. This nearly even spacing is achieved by using a number of bins that increases geometrically beyond a user-specified frequency (that defaults to the third harmonic of the input). See the "help" information associated with `logsmooth` for further details. With this function, it is feasible to present the results of a multi-million point simulation without having to plot millions of points in the spectrum. As Figure A.9 shows, the agreement between the expected and observed PSDs is quite good. The justification of the formula used to calculate the expected PSD is also relegated to the next section.

As a final demonstration, we will use the PSD plot of Figure A.9 to manually estimate the SNR. The signal power is read directly from the graph: -6 dBFS. The noise power is computed by multiplying the noise density by the bandwidth, or in logarithmic terms, by adding $10\log(\frac{BW}{NBW})$ to the noise density in dB. For $OSR = 32$, $BW = 0.5/OSR = 1.6 \times 10^{-2}$, and since $NBW = 7.3 \times 10^{-4}$, the conversion factor from average noise density to total noise power is $10\log(\frac{BW}{NBW}) = 13$ dB. Since the average noise density in the passband is -100 dBFS/NBW, the estimated SNR is $-6 - (-100 + 13) = 81$ dB.

A.5   Mathematical Background

Until now, the emphasis in this Appendix has been on using the FFT to perform spectral estimation for the signals in a $\Delta\Sigma$ modulator, while keeping the mathematics to a minimum. The mathematical theory associated with spectral estimation involves a number of concepts from stochastic processes that are worthy of chapters in themselves. We cannot do justice to such concepts in a subsection of an Appendix, so we content ourselves to list the key results and to use these as justification for a number of formulas that appeared earlier in the Appendix without justification. See [2] for background.

The autocorrelation function of a discrete-time stationary[5] random process x is defined as

$$r_x[k] = E\{x[n]x[n+k]\}, \tag{A.8}$$

where E denotes expectation ("average"). The z-transform of r_x is

$$R_x(z) = \sum_{n=-\infty}^{\infty} r_x[n]z^{-n}, \tag{A.9}$$

and the PSD $S_x(f)$ of x is defined as

$$S_x(f) = R_x(\exp(j2\pi f)). \tag{A.10}$$

In other words, *the PSD is defined as the Fourier transform of the autocorrelation function.* The link between this definition and the more intuitive definition that $S_x(f)$ is the amount of power between frequencies f and $f + df$ divided by df is established by the following two properties of S_x:

---

[5]A *stationary* random process is a random process whose statistical properties (mean, variance, etc.) do not vary with time.

a. $P_x = \int_0^1 S_x(f)df$, where $P_x = E[|x[n]|^2]$ is the power of x.

b. If the output of a linear system having a transfer function $H(z)$ is y when the input x, then $S_y(f) = |H(e^{j2\pi f})|^2 S_x(f)$.

The first property says that integrating the power spectrum over all frequencies yields the power in the signal. The second says that filtering the signal multiplies its power spectrum by the squared magnitude of the filter's transfer function. The intuitive definition of $S_x(f)$ results from considering an ideal filter H having a bandwidth df about a frequency f.

Note that the first property required S_x to be integrated over the range $[0, 1]$. Since real signals have symmetric spectra, it is common in practice to use the range $[0, 0.5]$ and double the density

$$P_x = \int_0^{0.5} 2S_x(f)\, df. \tag{A.11}$$

Since the range $[0.5, 1]$ is equivalent to the range $[-0.5, 0]$, this convention is similar to the conventional use of single-sided spectral densities when dealing with continuous-time signals.

Next, we consider the estimate of S_x (repeated from (A.7))

$$\hat{S}'_x(f) = \left| \frac{1}{\|w\|_2} \sum_{n=0}^{N-1} w[n] \exp(-j2\pi fn) \right|^2. \tag{A.12}$$

(We can obtain $\hat{S}_x(f_i)$ for $f_i = i/N$ from a length-N FFT of x windowed by w.)

The properties of this estimate are

a. $E\left[\hat{S}_x(f)\right] = S_x(f) * \frac{S_w(f)}{\|w\|_2^2}$, where $S_w(f) = |W(f)|^2$ and $*$ denotes circular convolution.

b.
$$E\left\{ \sum_{i=0}^{N-1} \frac{\hat{S}_x(i/N)}{N} \right\} = P_x.$$

c. $Var[\hat{S}_x(f)] \approx [S_x(f)]^2$, where $Var[y]$ denotes the variance of a statistical quantity y.

The first property states that $\hat{S}'_x(f)$ is a biased estimator of $S_x(f)$, namely, that the expected value of $\hat{S}'_x(f)$ is not equal to the true value of $S_x(f)$. For example, if a rectangular window is used, then the expected value of the estimated PSD is equal to the actual PSD convolved with

$$\frac{S_w(f)}{\|w\|_2^2} = \frac{1}{N} \left(\frac{\sin(N\pi f)}{\sin(\pi f)} \right)^2. \tag{A.13}$$

The second property effectively states that summing the values of $\hat{S}'_x(f_i)$ contained in the FFT, and dividing by N (effectively, integrating $\hat{S}'_x(f)$ over [0, 1]) yields an unbiased estimate of the power in x. This property justifies summing the powers in the in-band bins of an FFT to produce an estimate of the in-band noise power.

The third property states that the PSD estimate is very "noisy," in that the standard deviation of the estimate is as large as the quantity being estimated. This property necessitates the use of averaging, as discussed in Section A.3.

We are now in the position to calculate the NBW of a sine-wave-scaled FFT employing a window w. We assume that $|W(f)|$ has a peak at $f = 0$ and that the full-scale range is $[-1, 1]$. The sine-wave-scaled PSD estimate is, from (A.5),

$$\hat{S}'_x(f) = \left| \frac{1}{W(0)/2} \sum_{n=0}^{N-1} w[n] \exp\left(-j2\pi f n\right) \right|^2. \tag{A.14}$$

Thus, $\hat{S}'_x(f)$ is related to $\hat{S}_x(f)$ (as given in (A.12)) by

$$\hat{S}'_x(f) = \frac{\hat{S}_x(f) \|w_2\|^2}{|W(0)/2|^2}. \tag{A.15}$$

Since we want the integral of $\hat{S}'_x(f)$ to yield the power in x relative to the power in a full-scale sine wave, which is 0.5, we need to find the value of NBW that makes the following equation true:

$$E\left[\int_0^{0.5} \frac{\hat{S}'_x(f)}{NBW} df \right] = \frac{P_x}{0.5}. \tag{A.16}$$

Since $E\left[\int_0^{0.5} 2\hat{S}'_x(f) df \right] = P_x$,

$$NBW = \frac{\|w\|_2^2}{|W(0)|^2}. \tag{A.17}$$

This expression was used to tabulate NBW for the three windows in Table. A.1. If we assume that $w[n] \geq 0$, then $|W(0)| = \|w\|_1$ and we arrive at the compact result

$$NBW = \frac{\|w\|_2^2}{\|w\|_1^2}, \tag{A.18}$$

that was used in the code that generates Figure A.9 . Since the full-scale range affects both the definition of $\hat{S}'_x(f)$ and the reference power of 0.5 proportionally, NBW is independent of the full-scale range.

The final formula that requires explanation is the formula for the expected PSD of the shaped quantization noise of a $\Delta\Sigma$ modulator, that was also used in the code that generates Figure A.9. Since the power of quantization noise is $\Delta^2/12$, where Δ is the step size, the power of the quantization noise for an M-step quantizer with a step size $\Delta = 2$, relative to the power $M^2/2$ of a full-scale sine wave is $2/(3M^2)$. Assuming the quantization noise is white, its one-sided PSD is twice this amount, or $4/(3M^2)$. Thus, the PSD of the shaped quantization noise is

$$S_q(f) = \frac{4|H(e^{j2\pi f})|^2}{3M^2}. \tag{A.19}$$

For consistency with a plot of $\hat{S}'_x(f)$, $S_q(f)$ must be multiplied by NBW.

References

[1] E. O. Brigham, *The Fast Fourier Transform and its Applications*. Prentice Hall, 1988.

[2] A. Papoulis and S. U. Pillai, *Probability, Random Variables, and Stochastic Processes*. Tata McGraw-Hill Education, 2002.

[3] A. V. Oppenheim, R. W. Schafer, and J. R. Buck, *Discrete-Time Signal Processing*. Prentice-Hall, 1989.

[4] f. j. harris, "On the use of windows for harmonic analysis with the discrete Fourier transform," *Proceedings of the IEEE*, vol. 66, no. 1, pp. 51–83, 1978.

APPENDIX B

THE DELTA-SIGMA TOOLBOX

Getting Started

Go to `http://www.mathworks.com/matlabcentral/fileexchange/` and search for `delsig`. Download and unzip the `delsig.zip` file. Add the `delsig` directory to the MATLAB path. To improve simulation speed, compile the `simulateDSM.c` file by typing `mex simulateDSM.c` at the MATLAB prompt. Do the same for `simulateMS.c`.

The Delta-Sigma toolbox requires the Signal Processing toolbox and the Control Systems toolbox; the `clans` and `designPBF` functions also require the Optimization toolbox.

The following conventions are used throughout the Delta-Sigma toolbox:

- Frequencies are normalized; $f = 1$ corresponds to the sampling frequency, f_s.

- Default values for function arguments are shown following an equals sign in the parameter list. To use the default value for an argument, omit the argument if it is at the end of the list, otherwise use `NaN` (not-a-number) or `[]` (the empty matrix) as a place-holder.

- The loop-filter of a general delta-sigma modulator is described with an *ABCD* matrix. See "Modulator Model Details" on page 533 for a description of this matrix.

Demonstrations and Examples

dsdemo1	Demonstration of the `synthesizeNTF` function. Noise transfer function synthesis for a 5th-order lowpass modulator, both with and without optimized zeros, plus an 8th-order bandpass modulator with optimized zeros.
dsdemo2	Demonstration of the `simulateDSM`, `predictSNR` and `simulateSNR` functions: time-domain simulation, SNR prediction using the describing function method of Ardalan and Paulos, spectral analysis and signal-to-noise ratio calculation. Lowpass, bandpass, multi-bit lowpass examples are given.
dsdemo3	Demonstration of the `realizeNTF`, `stuffABCD`, `scaleABCD` and `mapABCD` functions: coefficient calculation and dynamic range scaling.
dsdemo4	Audio demonstration of MOD1 and MOD2 with sinc^n decimation.
dsdemo5	Demonstration of the `simulateMS` function: simulation of the element selection logic of a mismatch-shaping DAC.
dsdemo6	Demonstration of the `designHBF` function. Hardware-efficient halfband filter design and simulation.
dsdemo7	Demonstration of the `findPIS` function: positively-invariant set computation.
dsexample1	Discrete-time modulator design example.
dsexample2	Continuous-time lowpass modulator design example.
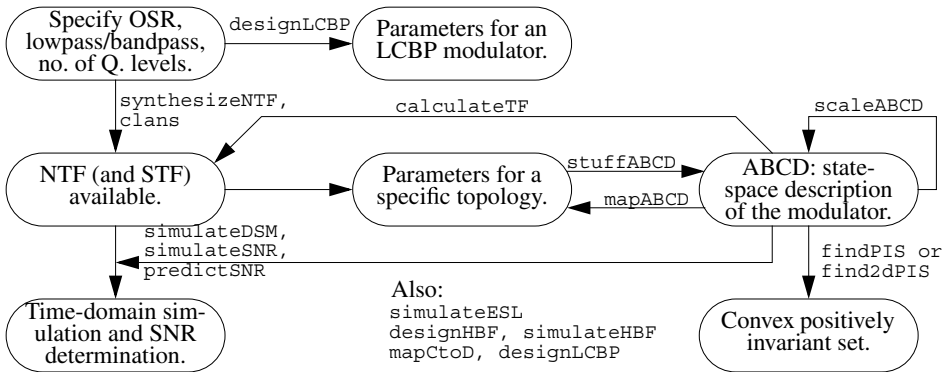
Figure B.1   Flowchart of key $\Delta\Sigma$ toolbox functions.

Key Functions

```
ntf = synthesizeNTF(order=3,R=64,opt=0,H_inf=1.5,f0=0)
ntf = clans(order=4,R=64,Q=5,rmax=0.95,opt=0)
ntf = synthesizeChebyshevNTF(order=3,R=64,opt=0,H_inf=1.5,f0=0)
```
Synthesize a noise transfer function.

```
[v,xn,xmax,y] = simulateDSM(u,ABCD,nlev=2,x0=0)
[v,xn,xmax,y] = simulateDSM(u,ntf,nlev=2,x0=0)
```
Simulate a delta-sigma modulator with a given input.

```
[snr,amp] = simulateSNR(ntf,OSR,amp=...,
                        f0=0,nlev=2,f=1/(4*R),k=13)
```
Determine the SNR versus input amplitude curve by simulation.

```
[a,g,b,c] = realizeNTF(ntf,form='CRFB',stf=1)
```
Convert a noise transfer function into coefficients for the specified topology.

```
ABCD = stuffABCD(a,g,b,c,form='CRFB')
```
Calculate the ABCD matrix given the parameters of the specified topology.

```
[a,g,b,c] = mapABCD(ABCD,form='CRFB')
```
Convert the ABCD matrix into the parameters of the specified topology.

```
[ABCDs, umax] = scaleABCD(ABCD,nlev=2,f=0,xlim=1,ymax=nlev+2)
```
Perform dynamic range scaling on a delta-sigma modulator described by ABCD.

```
[ntf,stf] = calculateTF(ABCD,k=1)
```
Calculate the NTF and STF of a delta-sigma modulator described by the given ABCD matrix, assuming a quantizer gain of k.

```
[sv,sx,sigma_se,max_sx,max_sy] =
    simulateMS(v,mtf,M=16,d=0,dw=[1-],sx0=[0-])
```
Simulate the element-selection logic of a mismatch-shaping DAC.

Functions for Continuous-Time Systems

```
[ABCDc,tdac2]= realizeNTF_ct(ntf,form='FB',tdac,ordering=[1:n],
                             bp=zeros(-),ABCDc)
```
Realize an NTF with a continuous-time loop-filter.

```
[sys, Gp] = mapCtoD(sys_c,t=[0 1],f0=0)
```
Map a continuous-time system to a discrete-time system whose impulse response matches the sampled pulse response of the original continuous-time system. See `dsexample2`.

```
H = evalTFP(Hs,Hz,f)
```
Compute the value of the product of the continuous-time transfer function H_s and the discrete-time transfer function H_z at frequencies f. Use this function to evaluate the signal transfer function of a CT $\Delta\Sigma$ ADC system.

Functions for Quadrature Systems

```
ntf = synthesizeQNTF(order=3,OSR=64,f0=0,NG=-60,ING=-20)
```
Synthesize a noise transfer function for a quadrature delta-sigma modulator.

```
[v,xn,xmax,y] = simulateQDSM(u,ABCD|ntf,nlev=2,x0=0)
```
Simulate a quadrature delta-sigma modulator with the given input.

```
ABCD = realizeQNTF(ntf,form='FB',rot=0,bn)
```
Convert a quadrature noise transfer function into a complex ABCD matrix for the specified structure.

```
ABCDr = mapQtoR(ABCD) and [ABCDq ABCDp] = mapR2Q(ABCDr)
```
Convert a complex matrix into its real equivalent and vice versa.

```
[ntf stf intf istf] = calculateQTF(ABCDr)
```
Calculate the noise and signal transfer functions of a quadrature modulator.

```
[sv,sx,sigma_se,max_sx,max_sy]=
    simulateQESL(v,mtf,M=16,sx0=[0-])
```
Simulate the Element Selection Logic of a quadrature differential DAC.

Note: `simulateSNR` works for a quadrature modulator if given a complex NTF or ABCD matrix; `simulateDSM` can also be used for a quadrature modulator if given an `ABCDr` matrix and a 2-element `nlev` vector.

Specialty Functions

```
[f1,f2,info] = designHBF(fp=0.2,delta=1e-5,debug=0)
```
Design a Saramäki half-band filter for use in a decimation or interpolation filter.

```
y = simulateHBF(x,f1,f2,mode=0)
```
Simulate a Saramäki half-band filter in the time domain.

```
[C, e, x0] = designPBF(N,M,pb,pbr,sbr,ncd,np,ns,fmax)
```
Design a symmetric polynomial-based filter (PBF) according to Hunter's method.

```
[snr,amp,k0,k1,sigma_e2 = predictSNR(ntf,OSR=64,amp=...,f0=0)
```
Predict the SNR versus input amplitude curve using the describing function method.

```
[s,e,n,o,Sc] = findPIS(u,ABCD,nlev=2,options)
```
Find a convex positively-invariant set for a delta-sigma modulator.

```
[data, snr] = findPattern(N=1024,OSR=64,ntf,ftest,Atest,
                          f0=0,nlev=2,quadrature=0,dbg=0)
```
Create a length-N data record which has good spectral properties when repeated.

Utility Funtions

Delta-Sigma Utility

```
mod1, mod2
```
Set the ABCD matrix, NTF and STF of the standard 1st- and 2nd-order modulators.

```
snr = calculateSNR(hwfft,f,nsig=1)
```
Estimate the SNR given the in-band bins of a windowed FFT and the location of the input.

```
[A B C D] = partitionABCD(ABCD, m)
```
Partition ABCD into A, B, C, D for an m-input state-space system.

```
H_inf = infnorm(H)
```
Compute the infinity norm (maximum absolute value) of a z-domain transfer function.

```
y = impL1(ntf,n=10)
```
Compute n points of the impulse response from the comparator output back to the comparator input for the given NTF.

```
y = pulse(S,tp=[0 1],dt=1,tfinal=10,nosum=0)
```
Compute the sampled pulse response of a continuous-time system.

```
sigma_H = rmsGain(H,f1,f2)
```

Compute the root mean-square gain of the discrete-time transfer function H in the frequency band $[f_1, f_2]$.

General Utility

`dbv(), dbp(), undbv(), undbp(), dbm(), undbm()`
The dB equivalent of voltage/power quantities, and their inverse functions.

`window = ds_hann(N)`
A Hann window of length N. Unlike MATLAB's original `hanning` function, `ds_hann` does not smear tones which are located exactly in an FFT bin (i.e. tones having an integral number of cycles in the given block of data). MATLAB 6's `hanning(N,'periodic')` function and MATLAB 7's `hann(N,'periodic')` function are the same as `ds_hann(N)`.

`mag = zinc(f,n=64,m=1)`
Calculate the magnitude response of a cascade of m sinc_n filters at frequencies f.

Graphing Utility

`plotPZ(H,color='b',markersize=5,list=0)`
Plot the poles and zeros of a transfer function.

`plotSpectrum(X,fin,fmt)`
Plot a smoothed spectrum.

`figureMagic(xRange,dx,xLab, yRange,dy,yLab, size)`
Performs a number of formatting operations for the current figure, including axis limits, ticks and labelling.

`printmif(file,size,font,fig)`
Print a figure to an Adobe Illustrator file and then use `ai2mif` to convert it to FrameMaker MIF format. `ai2mif` is an improved version of the function of the same name originally written by Deron Jackson <djackson@mit.edu>.

`[f,p] = logsmooth(X,inBin,nbin)`
Smooth the FFT X, and convert it to dB. See also `bplogsmooth` and `bilogplot`.

synthesizeNTF

Synopsis: `ntf = synthesizeNTF(order=3,OSR=64,opt=0,H_inf=1.5,f0=0)`
Synthesize a noise transfer function (NTF) for a delta-sigma modulator.

Input

`order`	The order of the NTF. `order` must be even for bandpass modulators.
`OSR`	The oversampling ratio. `OSR` is only needed when optimized NTF zeros are requested.
`opt`	A flag used to request optimized NTF zeros.
	`opt=0` puts all NTF zeros at band-center.
	`opt=1` optimizes the NTF zeros according to the high-OSR limit.
	`opt=2` puts at least one zero at band-center, but optimizes the rest.
	`opt=3` uses the Optimization toolbox to optimize the zeros.
`H_inf`	The maximum out-of-band gain of the NTF. Lee's rule states that `H_inf<2` should yield a stable modulator with a binary quantizer. Reducing `H_inf` increases the likelihood of success, but reduces the attenuation provided by the NTF and thus the theoretical resolution of the modulator.
`f0`	The center frequency of the modulator. $f_0 \neq 0$ yields a bandpass modulator; `f0=0.25` puts the center frequency at $f_s/4$.

Output

`ntf`	The modulator NTF, given as an LTI object in zero-pole form.

Bugs

If `OSR` or `H_inf` are low, the NTF is not optimal. Use `synthesizeChebyshevNTF` instead.

Example

Fifth-order lowpass modulator; zeros optimized for an oversampling ratio of 32.

```
» H = synthesizeNTF(5,32,1)
Zero/pole/gain:
(z−1) (ẑ2 − 1.997z + 1) (ẑ2 − 1.992z + 1)
-------------------------------------
(z−0.7778) (ẑ2 − 1.613z + 0.6649) (ẑ2 − 1.796z + 0.8549)
Sampling time:  1
```
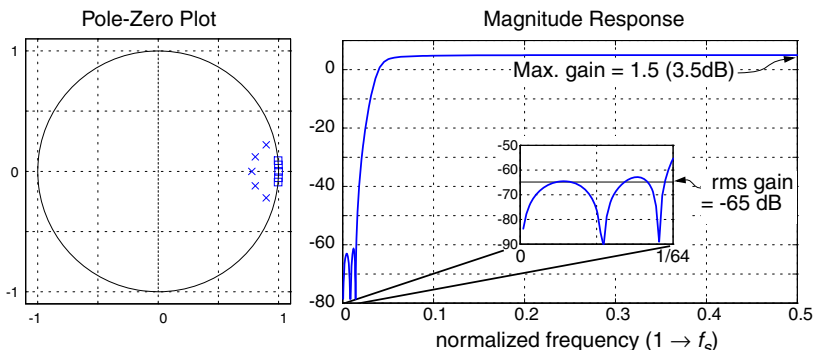
clans

Synopsis: `ntf = clans(order=4,OSR=64,Q=5,rmax=0.95,opt=0)`
Synthesize a lowpass NTF using the CLANS (Closed-loop analysis of noise-shaper) methodology [1]. This function requires the Optimization toolbox.

[1] J. G. Kenney and L. R. Carley, "Design of multibit noise-shaping data converters," *Analog Integrated Circuits Signal Processing Journal*, vol. 3, pp. 259-272, 1993.

Input

order	The order of the NTF.
OSR	The oversampling ratio.
Q	The maximum number of quantization levels used by the fed-back quantization noise. (Mathematically, $Q = \|h\|_1 - 1$, i.e. the sum of the absolute values of the impulse response samples minus 1.) The maximum stable input of a $\Delta\Sigma$ modulator is guaranteed to be at least $(n_{lev} - Q)$.
rmax	The maximum radius for the NTF poles.
opt	A flag used to request optimized NTF zeros.

Output

ntf	The modulator NTF, given as an LTI object in zero-pole form.

Example

5th-order lowpass modulator; time-domain noise gain of 5, zeros optimized for $OSR = 32$.
» `H= clans(5,32,5,.95,1)`
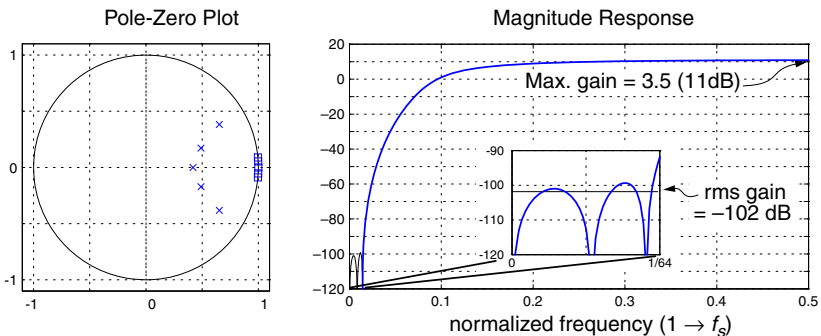
synthesizeChebyshevNTF

Synopsis: `ntf = synthesizeChebyshevNTF(order,OSR,opt,H_inf,f0)`
Obtain a noise transfer function (NTF) in which has equiripple magnitude in the passband.
`synthesizeChebyshevNTF` creates NTFs which are no better than `synthesizeNTF`,
except when `OSR` or `H_inf` are low.

Input and Output
Same as `ssynthesizeNTF`, except that the `opt` argument is not supported yet.

Examples
Compare the NTFs created by `synthesizeNTF` and `synthesizeChebyshevNTF` when
OSR is low:
```
» OSR = 4; order = 8; H_inf = 3;
» H1 = synthesizeNTF(order,OSR,1,H_inf);
» H3 = synthesizeChebyshevNTF(order,OSR,1,H_inf);
```
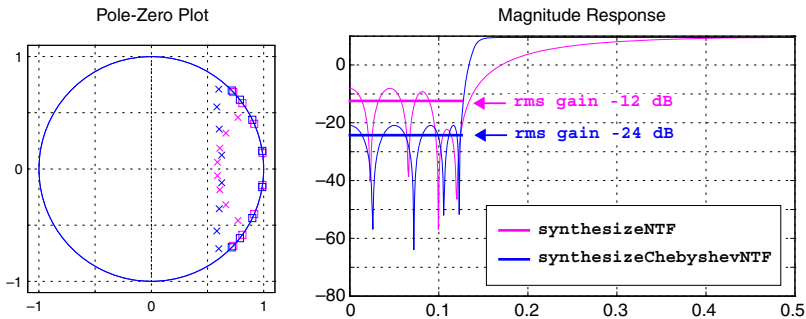

Repeat for H_inf low:
```
» OSR = 32; order = 5; H_inf = 1.2;
» H1 = synthesizeNTF(order,OSR,1,H_inf);
» H3 = synthesizeChebyshevNTF(order,OSR,1,H_inf);
```
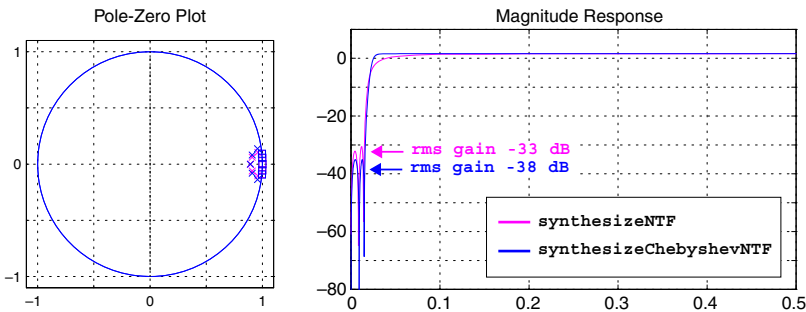
simulateDSM

Synopsis: `[v,xn,xmax,y] = simulateDSM(u,ABCD|ntf,nlev=2,x0=0)`
Simulate a delta-sigma modulator with a given input. For maximum speed, make sure that the compiled mex file is on your search path by typing `which simulateDSM` at the MATLAB$^{\text{TM}}$ prompt.

Input

u	The input sequence to the modulator, given as a $m \times N$ matrix, where m is the number of inputs (usually 1). Note that full-scale corresponds to an input of magnitude `nlev`−1.
ABCD	A state-space description of the modulator loop-filter.
ntf	The modulator NTF, given in zero-pole form. The modulator STF is assumed to be unity.
nlev	The number of levels in the quantizer. Multiple quantizers are indicated by making `nlev` a column vector.
x0	The initial state of the modulator.

Output

v	The samples of the output of the modulator, one for each input sample.
xn	The internal states of the modulator, one for each input sample, given as an $n \times N$ matrix.
xmax	The maximum absolute values of each state variable.
y	The samples of the quantizer input, one per input sample.

Example

Simulate a 5$^{\text{th}}$-order binary lowpass modulator with a half-scale sine-wave input and plot its output in the time and frequency domains.

```
>> OSR = 32; H = synthesizeNTF(5,OSR,1)}
>> N = 8192; fB = ceil(N/(2*OSR));}
>> f=85; u = 0.5*sin(2*pi*f/N*[0:N-1]);}
>> v = simulateDSM(u,H);
```
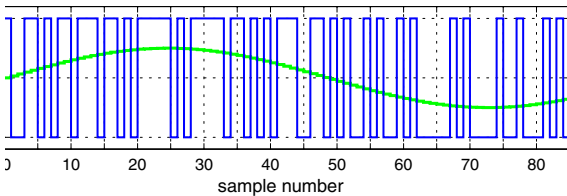


```
t = 0:85;
stairs(t, u(t+1),'g');
hold on;
stairs(t,v(t+1),'b');
axis([0 85 -1.2 1.2]);
ylabel('u, v');
```
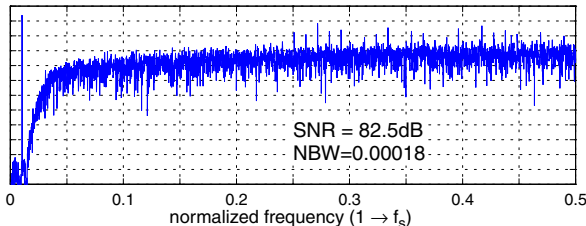


```
spec=fft(v.*ds_hann(N))/(N/4)
plot(linspace(0,0.5,N/2+1), .
   dbv(spec(1:N/2+1)));
axis([0 0.5 -120 0]);
grid on;
ylabel('dBFS/NBW')
snr=calculateSNR(spec(1:fB),f
s=sprintf('SNR = %4.1fdB\n',s
text(0.25,-90,s);
s=sprintf('NBW=%7.5f',1.5/N);
text(0.25, -110, s);
```

simulateSNR

Synopsis: `[snr,amp] = simulateSNR(ntf|ABCD|function,osr,amp,f0=0,`
`         nlev=2,f=1/(4*OSR),k=13,quadrature=0)`

Simulate a delta-sigma modulator with sine wave inputs of various amplitudes and calculate the signal-to-noise ratio (SNR) in dB for each input.

Input

`ntf`	The modulator NTF, given in zero-pole form.
`ABCD`	A state-space description of the modulator loop-filter, or the name of a function taking the input signal as its sole argument.
`osr`	The oversampling ratio.
`amp`	A row vector listing the amplitudes to use. Defaults to [-120 -110...-20 -15 -10 -9 -8 ... 0] dB, where 0 dB means a full-scale (peak value = `n_lev`−1) sine wave.
`f0`	The center frequency of the modulator.
`nlev`	The number of levels in the quantizer. Multiple quantizers are indicated by making `nlev` a vector.
`f`	The test frequency, adjusted to be an FFT bin.
`k`	The number of time points used for the FFT is 2^k.
`quadrature`	A flag indicating that the system being simulated is quadrature. This flag is set automatically if either `ntf` or `ABCD` are complex.

Output

`snr`	A row vector containing the SNR values calculated from the simulations.
`amp`	A row vector listing the amplitudes used.

Example

Compare the SNR versus input amplitude curve determined by the describing function method of Ardalan and Paulos with that determined by simulation for a 5th-order modulator.

```
» OSR = 32; H = synthesizeNTF(5,OSR,1)
» [snr_pred,amp] = predictSNR(H,OSR);
» [snr,amp] = simulateSNR(H,OSR);
```
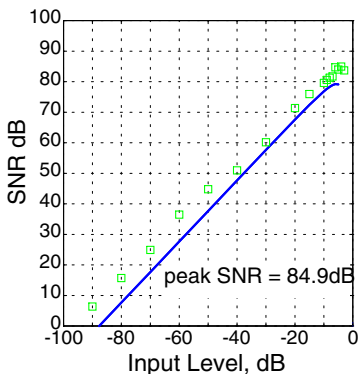


```
plot(amp,snr_pred,'b',amp,snr,'gs');
grid on;
figureMagic([-100 0], 10, 2, ...
    [0 100], 10, 1);
xlabel('Input Level, dB');
ylabel('SNR dB');
s=sprintf('peak SNR = %4.1fdB\n',...
    max(snr));
text(-65,15,s);
```

realizeNTF

Synopsis: `[a,g,b,c] = realizeNTF(ntf,form='CRFB',stf=1)`
Convert an NTF into a set of coefficients for a particular modulator topology.

Input

`ntf`	The modulator NTF, given in zero-pole form (i.e. a zpk object).
`form`	A string specifying the modulator topology.
	`CRFB` Cascade-of-resonators, feedback form.
	`CRFF` Cascade-of-resonators, feedforward form.
	`CIFB` Cascade-of-integrators, feedback form.
	`CIFF` Cascade-of-integrators, feedforward form.
	`---D` Any of the above, but the quantizer is delaying.
	Structures are described in "Modulator Model Details" on page 533.
`stf`	The modulator STF, specified as a zpk object. Note that the poles of the STF must match those of the NTF in order to guarantee that the STF can be realized without the addition of extra state variables.

Output

`a`	Feedback/feedforward coefficients from/to the quantizer ($1 \times n$).
`g`	Resonator coefficients ($1 \times \lfloor n/2 \rfloor$).
`b`	Feed-in coefficients from the modulator input to each integrator ($1 \times (n+1)$).
`c`	Integrator inter-stage coefficients. ($1 \times n$). In unscaled modulators, c is all ones.

Example

Determine the coefficients for a 5th-order modulator with the cascade-of-resonators structure, feedback (CRFB) form.

```
>> H = synthesizeNTF(5,32,1);}
>> [a,g,b,c] = realizeNTF(H,'CRFB')}
 a = 0.0007    0.0084    0.0550    0.2443    0.5579}
 g = 0.0028    0.0079}
 b = 0.0007    0.0084    0.0550    0.2443    0.5579    1.0000}
 c = 1     1     1     1     1
```

See Also

Use `realizeNTF_ct` (page 516) to realize an NTF with a continuous-time loop-filter.

stuffABCD

Synopsis: `ABCD = stuffABCD(a,g,b,c,form='CRFB')`
Calculate the ABCD matrix given the parameters of a specified modulator topology.

Input

`a`	Feedback/feedforward coefficients from/to the quantizer.
`g`	Resonator coefficients.
`b`	Feed-in coefficients from the modulator input to each integrator.
`c`	Integrator inter-stage coefficients.
`form`	See `realizeNTF` on page 510 for a list of supported forms and "Supported Modulator Topologies" on page 534 for block diagrams of them.

Output

`ABCD`	A state-space description of the loop-filter.

mapABCD

Synopsis: `[a,g,b,c] = mapABCD(ABCD,form='CRFB')`
Calculate the parameters for a specified modulator topology, assuming ABCD fits that topology.

Input

`ABCD`	A state-space description of the modulator loop-filter.
`form`	See `realizeNTF` on page 510 for a list of supported structures.

Output

`a`	Feedback/feedforward coefficients from/to the quantizer.
`g`	Resonator coefficients.
`b`	Feed-in coefficients from the modulator input to each integrator.
`c`	Integrator inter-stage coefficients.

scaleABCD

Synopsis: `[ABCDs,umax]=scaleABCD(ABCD,nlev=2,f=0,xlim=1,ymax=nlev+5,`
`umax,N=1e5)`

Scale the ABCD matrix so that the state maxima are less than a specified limit. The maximum stable input is determined as a side-effect of this process.

Input

ABCD	A state-space description of the modulator loop-filter.
nlev	The number of levels in the quantizer.
f	The normalized frequency of the test sinusoid.
xlim	The limit on the states. May be given as a vector.
ymax	The threshold for judging modulator stability. If the quantizer input exceeds $ymax$, the modulator is considered to be unstable.

Output

ABCDs	The scaled state-space description of the modulator loop-filter.
umax	The maximum stable input. Input sinusoids with amplitudes below this value should not cause the modulator states to exceed their specified limits.

calculateTF

Synopsis: `[ntf,stf] = calculateTF(ABCD,k=1)`
Calculate the NTF and STF of a delta-sigma modulator.

Input

ABCD	A state-space description of the modulator's loop-filter.
k	The quantizer gain to assume.

Output

ntf	The modulator NTF, given as an LTI system in zero-pole form.
stf	The modulator STF, given as an LTI system in zero-pole form.

Example

Realize a 5th-order modulator with the cascade-of-resonators structure, feedback form. Calculate the ABCD matrix of the loop-filter and verify that the NTF and STF are correct.

```
>> H = synthesizeNTF(5,32,1)
Zero/pole/gain:
(z-1) (z^2 - 1.997z + 1) (z^2 - 1.992z + 1)
-----------------------------------------------------------
(z-0.7778) (z^2 - 1.613z + 0.6649) (z^2 - 1.796z + 0.8549)
Sampling time: 1

>> [a,g,b,c] = realizeNTF(H)
a = 0.0007    0.0084    0.0550    0.2443    0.5579
g = 0.0028    0.0079
b = 0.0007    0.0084    0.0550    0.2443    0.5579    1.0000
c = 1     1     1     1     1

>> ABCD = stuffABCD(a,g,b,c)
ABCD =
1.0000         0         0         0         0    0.0007   -0.0007
1.0000    1.0000   -0.0028         0         0    0.0084   -0.0084
1.0000    1.0000    0.9972         0         0    0.0633   -0.0633
     0         0    1.0000    1.0000   -0.0079    0.2443   -0.2443
     0         0    1.0000    1.0000    0.9921    0.8023   -0.8023
     0         0         0         0    1.0000    1.0000         0

>> [ntf,stf] = calculateTF(ABCD)
Zero/pole/gain:
(z-1) (z^2 - 1.997z + 1) (z^2 - 1.992z + 1)
-----------------------------------------------------------
(z-0.7778) (z^2 - 1.613z + 0.6649) (z^2 - 1.796z + 0.8549)
Sampling time: 1

Zero/pole/gain:
1
Static gain.
```

simulateMS

Synopsis: `[sv,sx,sigma_se,max_sx,max_sy]`
`= simulateMS(v,M=16,mtf,d=0,dw=[1,1,...],sx0=[0-])`
Simulate the element selection logic of a mismatch-shaping DAC.

Input

v	The DAC input. v must be in $-M:2:M$ if $dw = [1, 1, \ldots]$. For other dw, v must be in the range $[-\sum_i^M dw(i), \sum_i^M dw(i)]$.
M	The number of DAC elements.
mtf	The mismatch-shaping transfer function, given in zero-pole form.
d	Dither uniformly distributed in $[-d, d]$ is added to the sy input of the vector quantizer.
dw	A vector containing the nominal weight associated with each element.
sx0	An $n \times M$ matrix containing the initial state of the element selection logic. n is the order of mtf.

Output

sv	The selection vector: a vector of zeros and ones indicating which elements to enable.
sx	An $n \times M$ matrix containing the final state of the element selection logic.
sigma_se	The rms value of the selection error, $se = sv - sy$. sigma_se may be used to analytically estimate the power of in-band noise caused by element mismatch.
max_sx	The maximum value attained by any state in the ESL.
max_sy	The maximum value attained by any component of the (un-normalized) "desired usage" vector sy.

See Also

simulateTSMS, simulateBiDWA, simulateXS and simulateMXS.
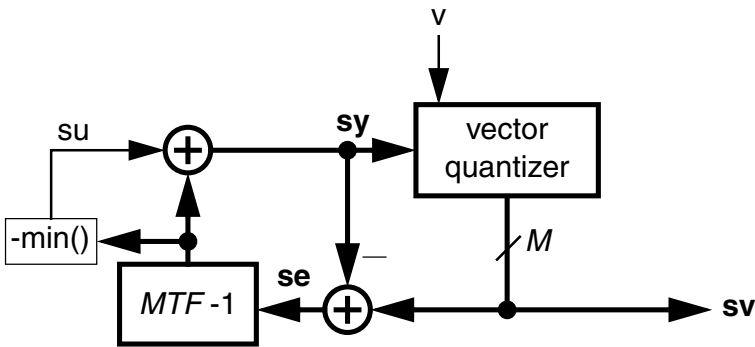
Block diagram of the Element Selection Logic

Compare the usage patterns and example spectra for a 16-element DAC driven with thermometer-coded, 1st-order and 2nd-order mismatch-shaped data generated by a 3rd-order modulator.

```
ntf = synthesizeNTF(3,[],[],4);
```

```
M = 16;
N = 2^14;
fin = round(0.33*N/(2*12));
u = M/sqrt(2)*sin((2*pi/N)*fin*[0:N-1]);
v = simulateDSM(u,ntf,M+1);
sv0 = ds_therm(v,M);
mtf1 = zpk(1,0,1,1);                  % First-order shaping
sv1 = simulateMS(v,mtf1,M);
mtf2 = zpk([ 1 1 ], [ 0 0 ], 1, 1);   % Second-order shaping
sv2 = simulateMS(v,mtf2,M);
ue = 1 + 0.01*randn(M,1);             % 1% mismatch
dv0 = ue' * sv0;
spec0 = fft(dv0.*ds_hann(N))/(M*N/8);
plotSpectrum(spec0,fin,'g');
```
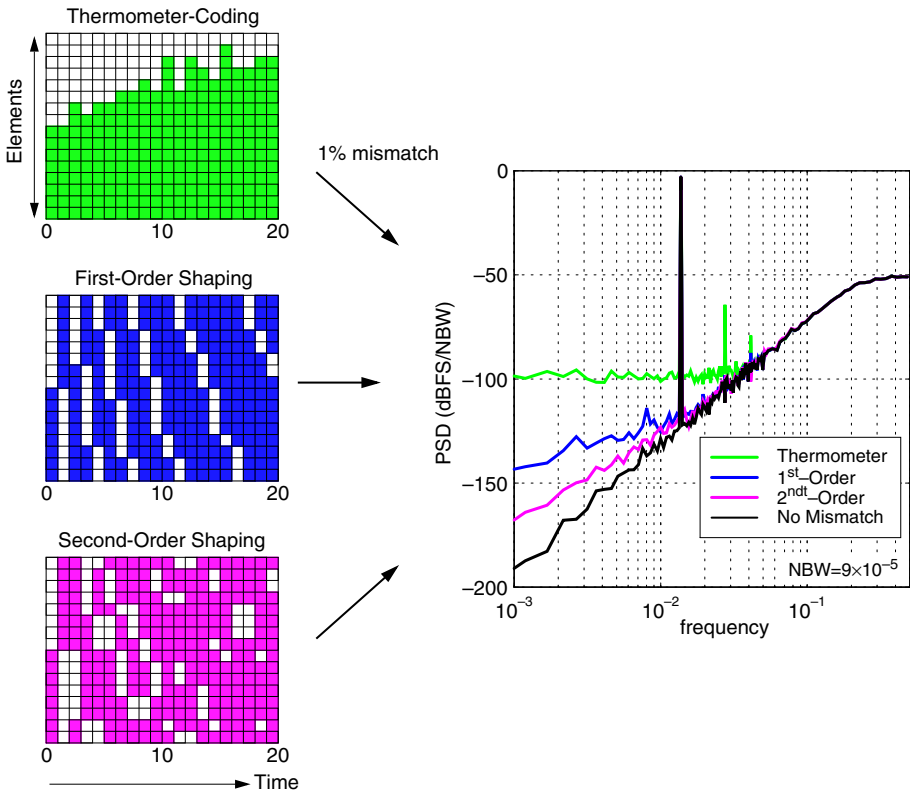
realizeNTF_ct

Synopsis: `[ABCDc,tdac2] = realizeNTF_ct(ntf,form='FB' ,tdac=[0 1],`
`ordering=[1:n],bp=zeros(-),ABCDc)`
Realize a noise transfer function (NTF) with a continuous-time loop-filter.

Input

`ntf`	The modulator NTF, specified as an LTI object in zero-pole form.
`form`	A string specifying the modulator topology.
	`FB` Feedback form.
	`FF` Feedforward form.
`tdac`	The timing for the feedback DAC(s). If `tdac(1)` ≥ 1, direct feedback terms are added to the quantizer. Multiple timings (one or more per integrator) for the FB topology can be specified by making tdac a cell array, e.g.
	`tdac = {[1,2]; [1 2]; [0.5 1],[1 1.5]; [];}`
`ordering`	A vector specifying which NTF zero-pair to use in each resonator. Default is for the zero-pairs to be used in the order specified in the NTF.
`bp`	A vector specifying which resonator sections are bandpass. The default (zeros(...)) is for all sections to be lowpass.
`ABCDc`	The loop-filter structure, in state-space form. If this argument is omitted, ABCDc is constructed according to `form`.

Output

`ABCDc`	A state-space description of the CT loop-filter.
`tdac2`	A matrix with the DAC timings, one for each input, including ones that were automatically added.

Example

Realize the NTF with a CT system (cf. the example on page 517).

```
>> ntf = zpk([1 1],[0 0],1,1);
>> [ABCDc,tdac2] = realizeNTF_ct(ntf,'FB')

ABCDc =
         0         0    1.0000   -1.0000
    1.0000         0         0   -1.5000
         0    1.0000         0    0.0000

tdac2 =
    -1        -1
     0         1
```

mapCtoD

Synopsis: `[sys, Gp] = mapCtoD(sys_c,t=[0 1],f0=0)`
Map a MIMO continuous-time system to a SIMO discrete-time equivalent. The criterion
for equivalence is that the sampled pulse response of the CT system must be identical to
the impulse response of the DT system. I.e. if y_c is the output of the CT system with an
input v_c taken from a set of DACs fed with a single DT input v, then y, the output of the
equivalent DT system with input v satisfies $y[n] = y_c(n^-)$ for integer n. The DACs are
characterized by rectangular impulse responses with edge times specified in the `t` matrix.

Input
`sys_c`	The LTI description of the CT system.
`t`	The edge times of the DAC pulse used to make CT waveforms from DT inputs. Each row corresponds to one of the system inputs; `[-1 -1]` denotes a CT input. The default is `[0 1]` for all inputs except the first, which is assumed to be a CT input.
`f0`	The frequency for which the `Gp` filters' gains are to be set to unity. Default 0 (DC).

Output
`sys`	The LTI description for the DT equivalent.
`Gp`	The mixed CT/DT prefilters which form the samples fed to each state for the CT inputs.

Reference
R. Schreier and B. Zhang, "Delta-sigma modulators employing continuous-time circuitry," *IEEE Transactions on Circuits and Systems I*, vol. 43, no. 4, pp. 324-332, April 1996.

Example
Map the standard second-order CT modulator shown below to its DT equivalent and verify
that the NTF is $(1 - z^{-1})^2$.

$$\begin{bmatrix} \dot{x}_{1c} \\ \dot{x}_{2c} \\ y_c \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & -1 \\ 1 & 0 & 0 & -1.5 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{x}_{1c} \\ \dot{x}_{2c} \\ u_c \\ v_c \end{bmatrix}$$
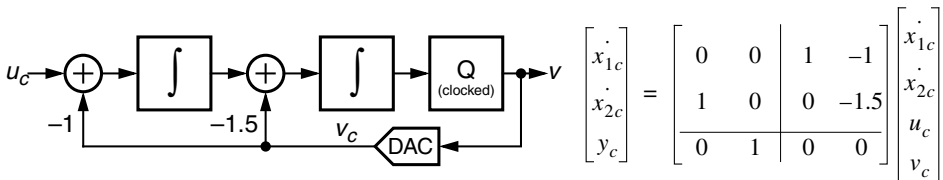
```
>> LFc = ss([0 0;1 0], [1 -1;0 -1.5], [0 1], [0 0]);
>> tdac = [0 1];
>> [LF,Gp] = mapCtoD(LFc,tdac);
>> ABCD = [LF.a LF.b; LF.c LF.d];
>> H = calculateTF(ABCD)

Zero/pole/gain:
(z-1)^2
-------
  z^2
Sampling time: 1
```

evalTFP

Synopsis: `H = evalTFP(Hs,Hz,f)`
Use this function to evaluate the signal transfer function of a continuous-time (CT) system. In this context `Hs` is the open-loop response of the loop-filter from the u input and `Hz` is the closed-loop noise transfer function.

Input

`Hs`	A continuous-time transfer function in zpk form.
`Hz`	A discrete-time transfer function in zpk form.
`f`	A vector of frequencies.

Output

`H`	The value of $H_s(j2\pi f)H_z(e^{j2\pi f})$.

See Also
`evalMixedTF` is a more advanced version of this function which is used to evaluate the individual feed-in transfer functions of a CT modulator.

Example
Plot the STF of the 2nd-order CT system depicted on page 517.

```
Ac = [0 0; 1 0];
Bc = [1 -1; 0 -1.5];
Cc = [0 1];
Dc = [0 0];
LFc = ss(Ac, Bc, Cc, Dc);
L0c = zpk(ss(Ac,Bc(:,1),Cc,Dc(1)));
tdac = [0 1];
[LF,Gp] = mapCtoD(LFc,tdac);
ABCD = [LF.a LF.b; LF.c LF.d];
H = calculateTF(ABCD);
% Yields H=(1-z^-1)^2
f = linspace(0,2,300);
STF = evalTFP(L0c,H,f);
plot(f,dbv(STF));
```
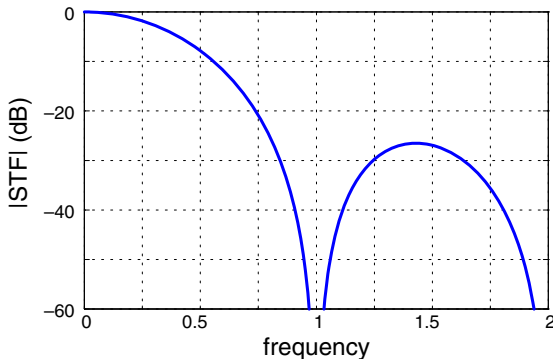
synthesizeQNTF

Synopsis: `ntf = synthesizeQNTF(order=3,OSR=64,f0=0,f0=-60,ING=-20,`
                    `n_im=order/3)`

Synthesize a noise transfer function (NTF) for a quadrature delta-sigma modulator.

Input

`order`	The order of the NTF.
`OSR`	The oversampling ratio.
`f0`	The center frequency of the modulator.
`NG`	The rms in-band noise gain (dB).
`ING`	The rms image-band noise gain (dB).
`n_im`	Number of image-band zeros.

Output

`ntf`	The modulator NTF, given as an LTI object in zero-pole form.

Bugs

ALPHA VERSION. This function uses an experimental ad hoc method that is neither optimal nor robust.

Example

Fourth-order, $OSR = 32$, $f_0 = 1/16$, bandpass NTF with an rms in-band noise gain of -50 dB and an image-band noise gain of -10 dB.

```
>> ntf = synthesizeQNTF(4,32,1/16,-50,-10);

Zero/pole/gain:

     (z-(0.953+0.303i)) (z^2 - 1.85z + 1) (z-(0.888+0.460i))
-----------------------------------------------------------------------
(z-(0.809+0.003i)) (z-(0.591+0.245i)) (z-(0.673-0.279i)) (z-(0.574+0.570i))

Sampling time: 1
```
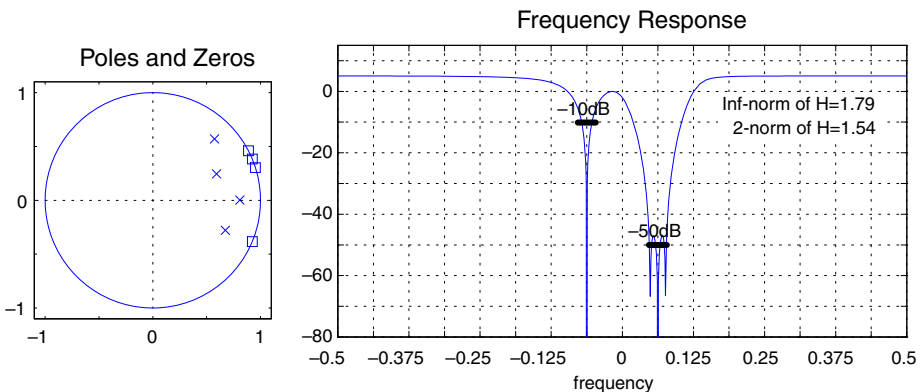
simulateQDSM

Synopsis: `[v,xn,xmax,y] = simulateQDSM(u,ABCD|ntf,nlev=2,x0=0)`
Simulate a quadrature delta-sigma modulator with a given input. For improved simulation speed, use `simulateDSM` with a 2-input/2-output `ABCDr` argument as indicated in the example in `mapQtoR` on page 522.

Input

u	The input sequence to the modulator, given as a $1 \times N$ row vector. Full-scale corresponds to an input of magnitude $nlev - 1$.
ABCD	A state-space description of the modulator's loop-filter.
ntf	The modulator NTF, given in zero-pole form.
nlev	The number of levels in the quantizer. Multiple quantizers are indicated by making `nlev` a column vector.
x0	The initial state of the modulator.

Output

v	The samples of the output of the modulator, one for each input sample.
xn	The internal states of the modulator, one for each input sample, given as an $n \times N$ matrix.
xmax	The maximum absolute values of each state variable.
y	The samples of the quantizer input, one per input sample.

Example

Simulate a 4th-order 9-level quadrature modulator with a half-scale sine-wave input and plot its output in the time and frequency domains.

```
nlev = 9; f0 = 1/16; osr = 32; M = nlev-1;
ntf = synthesizeQNTF(4,osr,f0,-50,-10);
N = 64*osr; f = round((f0+0.3*0.5/osr)*N)/N;
u = 0.5*M*exp(2i*pi*f*[0:N-1]);
v = simulateQDSM(u,ntf,nlev);
```

```
t = 0:25;
subplot(211)
plot(t, real(u(t+1)),'g');
hold on;
stairs(t,real(v(t+1)),'b');
figureMagic(…)
ylabel('real');
```
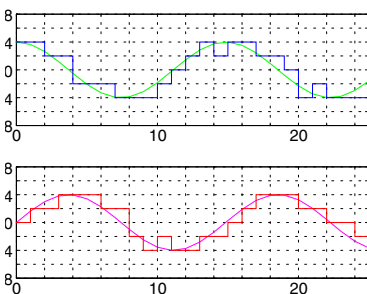
```
spec = fft(v.*ds_hann(N))/(M*N/2);
spec = [fftshift(spec) spec(N/2+1)];
plot(linspace(-0.5,0.5,N+1), dbv(spec))
figureMagic([-0.5 0.5],1/16,2, [-120 0],10
ylabel('dBFS/NBW')
[f1 f2] = ds_f1f2(osr,f0,1);
fb1 = round(f1*N); fb2 = round(f2*N);
fb = round(f*N)-fb1;
snr = calculateSNR(spec(N/2+1+[fb1:fb2]),f
text(f,-10,sprintf(' SNR = %4.1fdB\n',snr)
text(0.25, -105, sprintf('NBW=%0.1e',1.5/N
```
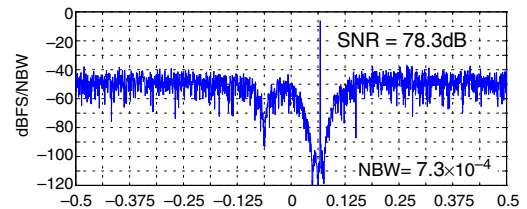
realizeQNTF

Synopsis: `ABCD = realizeQNTF(ntf,form='FB',rot=0,bn)`
Convert a quadrature NTF into an ABCD matrix for the specified structure.

Input

`ntf`	A zpk object specifying the modulator's NTF.
`form`	A string specifying the modulator topology.
	`FB` Feedback
	`PFB` Parallel feedback
	`FF` Feedforward
	`PFF` Parallel feedforward
`rot`	`rot=1` means rotate states to make as many coefficients as possible real.
`bn`	The coefficient of the auxiliary DAC for `form = 'FF'`.

Output

`ABCD`	State-space description of the loop-filter.

Example

Determine coefficients for the parallel feedback (PFB) structure.

```
>> ntf = synthesizeQNTF(5,32,1/16,-50,-10);
>> ABCD = realizeQNTF(ntf,'PFB',1)
ABCD =
Columns 1 through 4
 0.8854+0.4648i       0                 0                  0
 0.0065+1.0000i   0.9547+0.2974i        0                  0
     0            0.9715+0.2370i    0.9088+0.4171i         0
     0                0             0.8797+0.4755i    0.9376+0.3477i
     0                0                 0             -0.9916-0.1294i
Columns 5 through 7
     0            0.0025            0.0025+0.0000i
     0                0             0.0262+0.0000i
     0                0             0.1791+0.0000i
     0                0             0.6341+0.0000i
 0.9239-0.3827i       0             0.1743+0.0000i
-0.9312-0.3645i       0                 0
```

mapQtoR

Synopsis: `ABCDr = mapQtoR(ABCD)`
Convert a quadrature matrix into its real (IQ) equivalent.

Input
`ABCD`   A complex matrix describing a quadrature system.

Output
`ABCDr`   A real matrix corresponding to ABCD. Each element z in `ABCD` is replaced by a 2×2 matrix to make `ABCDr`. Specifically

$$z \rightarrow \begin{bmatrix} x & -y \\ y & x \end{bmatrix} \text{ where } x = Re(z) \text{ and } y = Im(z).$$

Example
Replace a call to `simulateQDSM` with a faster code block using `simulateDSM`.

```
% v = simulateQDSM(u,ntf,nlev);
ABCD = realizeQNTF(ntf,'FF');
ABCDr = mapQtoR(ABCD);
ur = [real(u); imag(u)];
vr=simulateDSM(ur,ABCDr,nlev*[1;1]);
v = vr(1,:) + 1i*vr(2,:);
```

mapRtoQ

Synopsis: `[ABCDq ABCDp] = mapR2Q(ABCDr)`
Map a real `ABCDr` to a quadrature `ABCD`. `ABCDr` has its states paired (real, imaginary) as indicated above in `mapQtoR`.

Input
`ABCDr`   A real matrix describing a quadrature system.

Output
`ABCDq`   The quadrature (complex) version of ABCDr.
`ABCDp`   The mirror-image system matrix. `ABCDp` is zero if `ABCDr` has no quadrature errors.

calculateQTF

Synopsis: `[ntf stf intf istf] = calculateQTF(ABCDr)`
Calculate the noise and signal transfer functions for a quadrature modulator.

Input
ABCDr   A real state-space description of the modulator's loop-filter. I/Q asym-
                 metries may be included in the description. These asymmetries result in
                 non-zero image transfer functions.

Output
ntf, stf   The noise and signal transfer functions.
intf, istf   The image noise and image signal transfer functions.
All transfer functions are returned as LTI systems in zero-pole form.

Example
Examine the effect of mismatch in the first feedback.

```
>> ABCDr = mapQtoR(ABCD);
>> ABCDr(2,end) = 1.01*ABCDr(2,end);   % 0.1% mismatch in first feedback
>> [H G HI GI] = calculateQTF(ABCDr);
```
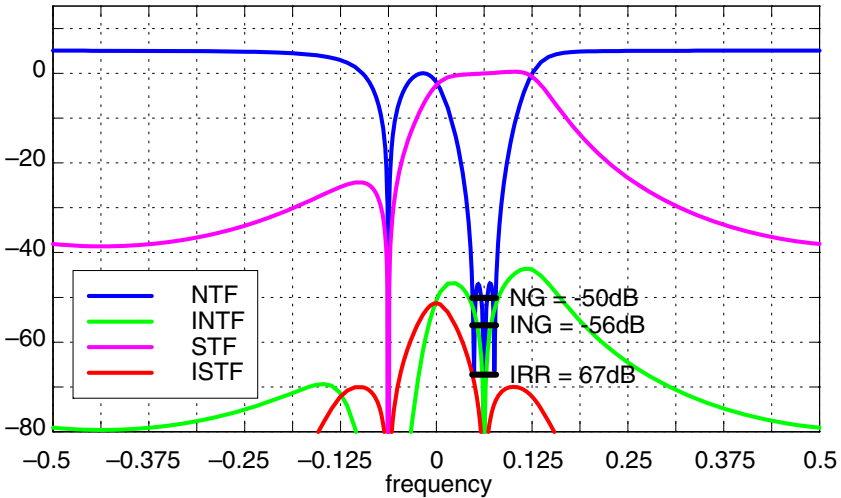
simulateQESL

Synopsis: `[sv,sx,sigma_se,max_sx,max_sy]`
`= simulateQESL(v,mtf,M=16,sx0=[0-])`
Simulate the element selection logic (ESL) of a quadrature differential DAC.

Input

`v`	A vector the digital input values.
`mtf`	The mismatch-shaping transfer function, given in zero-pole form.
`M`	The number of elements. There is a total $2M$ elements.
`sx0`	An $n \times M$ matrix whose columns are the initial state of the ESL.

Output

`sv`	The selection vector: a vector of zeros and ones indicating which elements to enable.
`sx`	An $n \times M$ matrix containing the final state of the ESL.
`sigma_se`	The rms value of the selection error, $se = sv = sy$. `sigma_se` may be used to estimate the power of in-band noise caused by element mismatch.
`max_sx`	The maximum absolute value attained by any state in the ESL.
`max_sy`	The maximum absolute value attained by any input to the VQ.

Example

```
>> mtf1 = zpk(exp(2i*pi*f0),0,1,1);
% First-order complex shaping
>> sv1 = simulateQESL(v,mtf1,M);
```
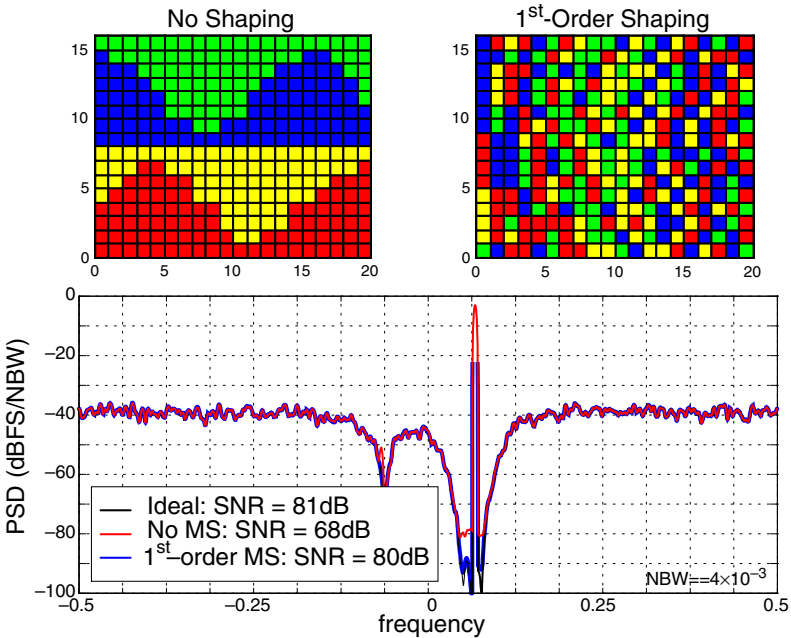
designHBF

Synopsis: `[f1,f2,info]=designHBF(fp=0.2,delta=1e-5,debug=0)`
Design a hardware-efficient linear-phase half-band filter for use in the decimation or inter-polation filter associated with a delta-sigma modulator. This function is based on the pro-cedure described by Saramäki [1]. Note that since the algorithm uses a non-deterministic search procedure, successive calls may yield different designs.

[1] T. Saramäki, "Design of FIR filters as a tapped cascaded interconnection of identical subfilters," *IEEE Transactions on Circuits and Systems*, vol. 34, pp. 1011-1029, 1987.

Input
fp	Normalized passband cutoff frequency.
delta	Passband and stopband ripple in absolute value.

Output
f1,f2	Prototype filter and subfilter coefficients and their canonical-signed digit (csd) representation.
info	A vector containing the following information data (only set when debug=1):

complexity	The number of additions per output sample.
n1,n2	The length of the f1 and f2 vectors.
sbr	The achieved stop-band attenuation in dB.
phi	The scaling factor for the F2 filter.

Example
Design of a lowpass half-band filter with a cut-off frequency of $0.2f_s$, a passband ripple of less than 10^{-5} and a stopband gain less than 10^{-5} (-100 dB).

```
>> [f1,f2] = designHBF(0.2,1e-5);
>> f = linspace(0,0.5,1024);
>> plot(f, dbv(frespHBF(f,f1,f2)))
```

A plot of the filter response is shown below. The filter achieves 109 dB of attenuation in the stopband and uses only 124 additions (no true multiplications) to produce each output sample.
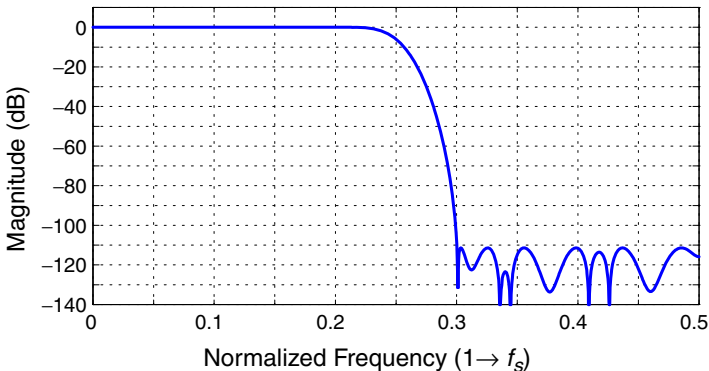
The structure of this filter as a decimation or interpolation filter is shown below. The coefficients and their canonical signed-digit (csd) decompositions are

```
[f1.val]' =     [f2.val]' =     >> f1.csd           >> f2.csd
    0.9453          0.6211       ans =               ans =
   -0.6406         -0.1895            0 -4 -7             -1 -3 -8
    0.1953          0.0957            1 -1  1              1  1 -1
                   -0.0508       ans =               ans =
                    0.0269            -1 -3 -6            -2 -4 -9
                   -0.0142            -1 -1 -1            -1  1 -1
                                ans =               ans =
                                    -2 -4 -7            -3 -5 -9
                                     1 -1  1             1 -1  1
                                                    ans =
                                                        -4 -7 -8
                                                        -1  1  1
                                                    ans =
                                                        -5 -8 -11
                                                         1 -1 -1
                                                    ans =
                                                        -6 -9 -11
                                                        -1  1 -1
```

In the csd expansions, the first row contains the powers of two while the second row gives their signs. For example, $f_1(1) = 0.9453 = 2^0 - 2^{-4} + 2^{-7}$. Since the filter coefficients use only 3 csd terms, each multiply-accumulate operation shown in the diagram below needs only 3 additions. An implementation of this 110th-order FIR filter therefore needs only $3 \times 3 + 5 \times (3 \times 6 + 6 - 1) = 124$ additions at the low ($f_s/2$) rate.

Decimation structure.

Interpolation structure.
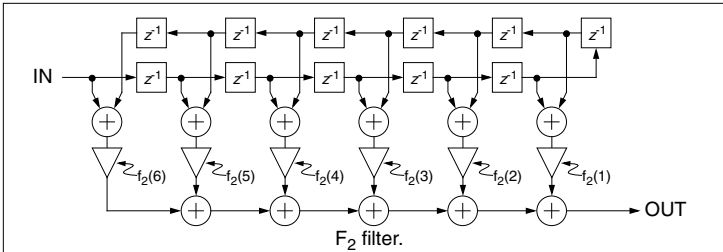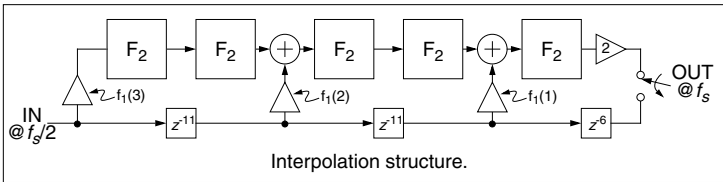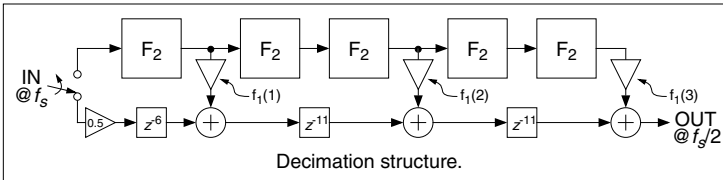
F_2 filter.

simulateHBF

Synopsis: `y = simulateHBF(x,f1,f2,mode=0)`
Simulate a Saramäki half-band filter (see `designHBF` on page 525) in the time domain.

Input

x	The input data.
f1,f2	Filter coefficients. `f1` and `f2` can be vectors of values or struct arrays like those returned from `designHBF`.
mode	This flag determines whether the input is filtered, interpolated, or decimated according to the following:

0	Plain filtering, no interpolation or decimation.
1	The input is interpolated.
2	The output is decimated, even samples are taken.
3	The output is decimated, odd samples are taken.

Output

y	The output data.

Example

Plot the impulse response of the HBF designed on the previous page.

```
>> N = (2*length(f1)-1)*2*(2*length(f2)-1)+1;
>> y = simulateHBF([1 zeros(1,N-1)],f1,f2);
>> stem([0:N-1],y);
>> figureMagic([0 N-1],5,2, [-0.2 0.5],0.1,1)
```
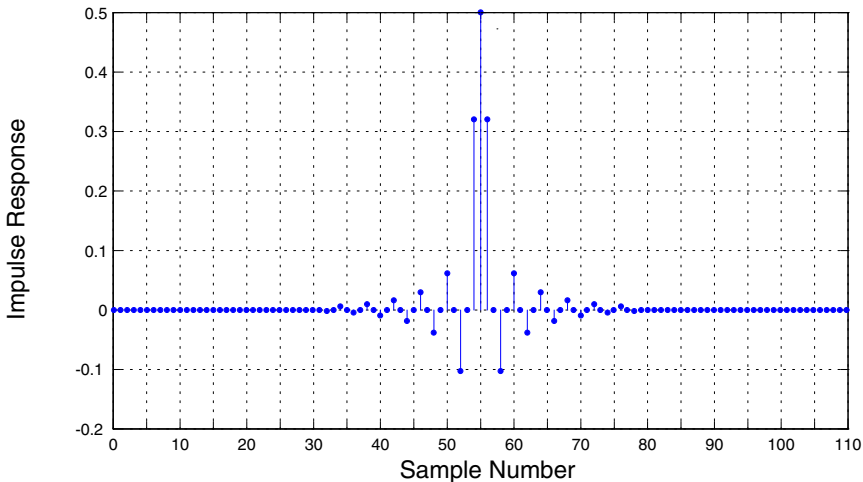
designPBF

Synopsis: `[C, e, x0] = designPBF(N,M,pb,pbr,sbr,ncd,np,ns,fmax)`
Design a symmetric polynomial-based filter (PBF) according to Hunter's method [1]. `designPBF` requires the Optimization toolbox.

[1] M. T. Hunter, "Design of polynomial-based filters for continuously variable sample rate conversion with applications in synthetic instrumentation and software defined radio," Ph.D. thesis, University of Florida, 2008.

Input
`N=10`	Number of polynomial pieces.
`M=5`	Order of the polynomial pieces.
`pb=0.25`	Passband width. Relative to the input sample rate, the passband is $[0, \text{pb}]$ and the stopband is $[1 - \text{pb}, \infty)$. Use `pb = 0.5/OSR` where `OSR` is the oversampling ratio of the input.
`pbr=0.1`	Passband ripple in dB.
`sbr=-100`	Stobpand ripple in dB.
`ncd=0`	Number of continuous derivatives. To allow the impulse response itself to be discontinuous, use `ncd = -1`.
`np=100`	Number of points in the passband.
`ns=1000`	Number of points in the stopband.
`fmax=5`	Maximum frequency checked in the stopband.

Output
`C`	$N \times (M + 1)$ matrix containing the coefficients of the polynomial pieces. Piece i is $p_i(x) = C(i, 1) + C(i, 2)x + C(i, 3)x^2 + ... + C(i, M + 1)x^M$.
`e`	The maximum weighted error. $e \le 1$ indicates the specs were met.
`x0=-0.5`	Offset on the polynomial argument, i.e. $x = \mu + \text{x0}$, where $\mu \in [0, 1]$.

Example
Construct a 10-segment PBF using polynomials of order 5 for interpolating signals with an input OSR of 2. Aim for a passband ripple of 0.1 dB and a stopband ripple of -100 dB.

```
[C, e, x0] = designPBF(10, 5, 0.5/2, 0.1, -100);
[hc, t] = impulsePBF(C,20,x0);
subplot(121); plot(t, hc, 'Linewidth', 1);
f = linspace(0,5,1000);
Hc = frespPBF(f,C,x0);
subplot(122); plot(f, dbv(Hc), 'Linewidth', 1);
```
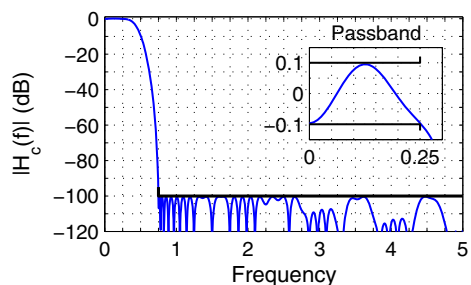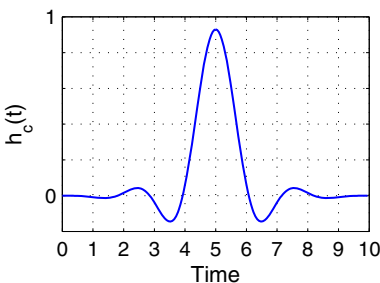
predictSNR

Synopsis: `[snr,amp,k0,k1,sigma_e2] = predictSNR(ntf,OSR=64,amp=...,f0=0)`
Use the describing function method of Ardalan and Paulos [1] to predict the signal-to-noise ratio (SNR) in dB for various input amplitudes. This method is only applicable to binary modulators.

[1] S. H. Ardalan and J. J. Paulos, "Analysis of nonlinear behavior in delta-sigma modulators," *IEEE Transactions on Circuits and Systems*, vol. 34, pp. 593-603, June 1987.

Input

`ntf`	The modulator NTF, given in zero-pole form.
`OSR`	The oversampling ratio.
`amp`	A row vector listing the amplitudes to use. `amp` defaults to $[-120 - 110... - 20 - 15 - 10 - 9 - 8...0]$ dB, where 0 dB means a full-scale (peak value = 1) sine wave.
`f0`	The center frequency of the modulator.

Output

`snr`	A row vector containing the predicted SNRs
`amp`	A row vector listing the amplitudes used.
`k0`	A row vector containing the signal gain of the quantizer model.
`k1`	A row vector containing the noise gain of the quantizer model.
`sigma_e2`	A row vector containing the mean square value of the noise in the quantizer model.

Example
See the example on page 509.

The Quantizer Model
The binary quantizer is modeled as a pair of linear gains and a noise source, as shown in the figure below. The input to the quantizer is divided into signal and noise components which are processed by signal-dependent gains k_0 and k_1. These components are added to a noise source, which is assumed to be white and to have a Gaussian distribution to produce the quantizer output. The variance σ_e^2 of the noise source is also signal-dependent.
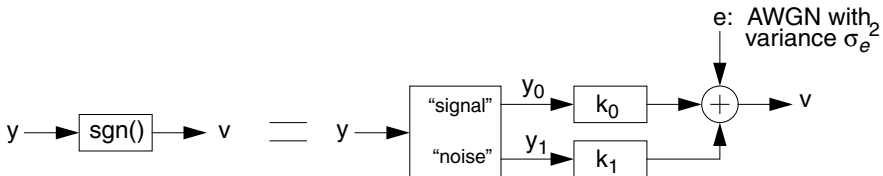
findPIS, find2dPIS (in the PosInvSet subdirectory)

Synopsis: `[s,e,n,o,Sc] = findPIS(u,ABCD,nlev=2,options)`
         `[s,e,n,o,Sc] = findPIS(u,ABCD,nlev=2,options)`
  `options = [dbg=0 itnLimit=2000 expFactor=0.005 N=1000 skip=100]`
Find a convex positively-invariant set for a delta-sigma modulator. `findPIS` requires compilation of the `qhull` mex file; `find2dPIS` does not but is limited to second-order systems.

This function is an implementation of the method described in [1][1]

Input

u	The input to the modulator. If `u` is a scalar, the input to the modulator is constant. If `u` is a 2×1 vector, the input to the modulator may be any sequence whose samples lie in the range $[u(1), u(2)]$.
ABCD	A state-space description of the modulator loop-filter.
nlev	The number of quantizer levels.
dbg	Set `dbg=1` to see a graphical display of the iterations.
itnLimit	The maximum number of iterations.
expFactor	The expansion factor applied to the hull before every mapping operation. Increasing `expFactor` decreases the number of iterations but results in sets which are inflated.
N	The number of points to use when constructing the initial guess.
skip	The number of time steps to run the modulator before observing the state. This handles the possibility of transients in the modulator.
qhullArgA	The 'A' argument to the `qhull` program. Adjacent facets are merged if the cosine of the angle between their normals is greater than the absolute value of this parameter. Negative values imply that the merge operation is performed during hull construction, rather than as a post-processing step.
qhullArgC	The 'C' argument to the `qhull` program. A facet is merged into its neighbor if the distance between the facet's centrum (the average of the facet's vertices) and the neighboring hyperplane is less than the absolute value of this parameter. As with the above argument, negative values imply pre-merging while positive values imply post-merging.

Output

s	The vertices of the set ($dim \times n_v$).
e	The edges of the set, listed as pairs of vertex indices ($2 \times n_e$).
n	The normals for the facets of the set ($dim \times n_f$).
o	The offsets for the facets of the set ($1 \times n_f$).
Sc	The scaling matrix which was used internally to round out the set.

Find a positively-invariant set for the second-order modulator with an input of $1/\sqrt{7}$.

```
>> ABCD = [
1      0      1     -1
1      1      1     -2
0      1      0      0];
>> s = find2dPIS(sqrt(1/7),ABCD,1)
```

[1][1] R. Schreier, M. Goodson and B. Zhang "An algorithm for computing convex positively invariant sets for delta-sigma modulators," *IEEE Transactions on Circuits and Systems I*, vol. 44, no. 1, pp. 38-44, January 1997.

```
s =
Columns 1 through 7
-1.5954    -0.2150     1.1700     2.3324     1.7129     1.0904     0.4672
-2.6019    -1.8209     0.3498     3.3359     4.0550     4.1511     3.6277
Columns 8 through 11
-0.1582    -0.7865    -1.4205    -1.5954
2.4785     0.6954    -1.7462    -2.6019
```


Iteration 29: 0 image vertices outside
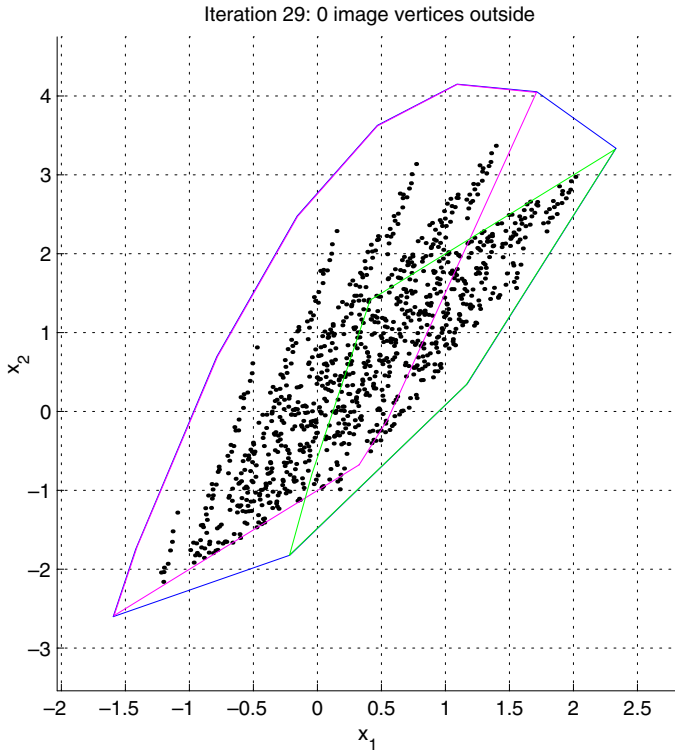
findPattern

Synopsis: `[data, snr] = findPattern(N=1024,OSR=64,ntf,ftest,Atest,`
`                    f0=0,nlev=2,quadrature=0,dbg=0)`

Use delta-sigma modulation to create a length-*N* data-stream that has good spectral properties when repeated.

Input

`N`	The length of the data record.
`OSR`	The oversampling ratio.
`NTF`	The modulator NTF.
`ftest`	The signal frequency. `ftest` may be a vector.
`Atest`	The target output level as a fraction of full-scale.
`f0`	The center frequency.
`nlev`	The number of levels in the output data.
`quadrature`	A flag which indicates to use quadrature modulation.
`dbg`	A flag which enables showing the progress of the iterations.

Output

`data`	$1 \times N$ data record.
`snr`	The in-band signal-to-noise ratio, in dB.

Example

Length-1024 data record containing a -3-dBFS, 5-cycle sine wave with low in-band noise for an oversampling ratio of 32.

```
N = 1024;
osr = 32;
ntf = synthesizeNTF(5,osr,1,1.5);
ftest = 5/N;
Atest = undbv(-3);
[data snr] = findPattern(N,osr,ntf,ftest,Atest);
spec = fft(data)/(N/2);
inband = 0:ceil(N/(2*osr));
lollipop(inband,dbv(spec(inband+1)),'b',2,-120);
```
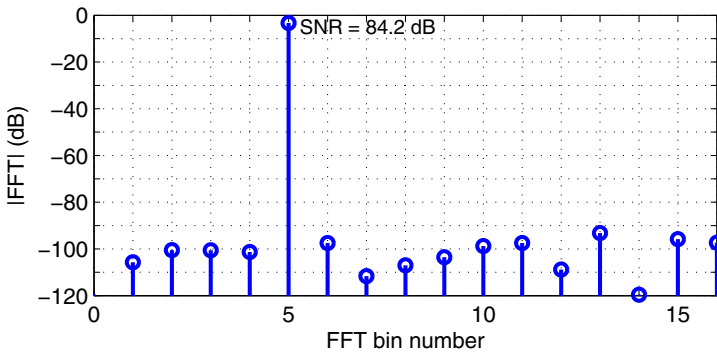
Modulator Model

A delta-sigma modulator with a single quantizer is assumed to consist of quantizer connected to a loop-filter as shown in the diagram below.
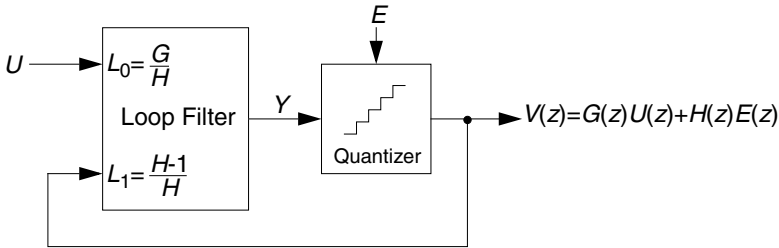
The Loop Filter

The loop-filter is described by an *ABCD matrix*. For single-quantizer systems, the loop-filter is a two-input, one-output linear system and ABCD is an $(n + 1) \times (n + 2)$ matrix, partitioned into A $(n \times n)$, B $(n \times 2)$, C $(1 \times n)$ and D (1×2) sub-matrices as shown below:

$$ABCD = \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array} \right] \tag{B.1}$$

The equations for updating the state and computing the output of the loop-filter are

$$x[n + 1] = Ax[n] + B \left[\begin{array}{c} u[n] \\ v[n] \end{array} \right]$$

$$y[n] = Cx[n] + D \left[\begin{array}{c} u[n] \\ v[n] \end{array} \right] \tag{B.2}$$

This formulation is sufficiently general to encompass all single-quantizer modulators that employ linear loop-filters. The toolbox currently supports translation to/from an ABCD description and coefficients for the following topologies:

CIFB	Cascade-of-integrators, feedback form.
CIFF	Cascade-of-integrators, feedforward form.
CRFB	Cascade-of-resonators, feedback form.
CRFF	Cascade-of-resonators, feedforward form.
CRFBD	Cascade-of-resonators, feedback form, delaying quantizer.
CRFFD	Cascade-of-resonators, feedforward form, delaying quantizer
Stratos	A CIFF-like structure supporting NTF zeros on the unit circle (Jeff Gealow)
DSFB	Double-sampled, feedback (Dan Senderowicz)

Multi-input and multi-quantizer systems can also be described with an ABCD matrix and Eq. (B.2) will still apply. For an n_i-input, n_o-output modulator, the dimensions of the sub-matrices are $A : n \times n$, $B : n \times (n_i + n_o)$, $C : n_o \times n$ and $D : n_o \times (n_i + n_o)$.

The Quantizer

The quantizer is ideal, producing integer outputs centered about zero. Quantizers with an even number of levels are of the mid-rise type and produce outputs which are odd integers. Quantizers with an odd number of levels are of the mid-tread type and produce outputs which are even integers.
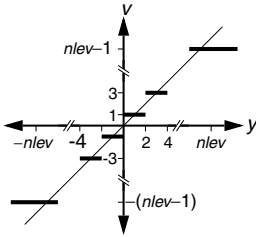
Transfer curve of a quantizer with an even number of levels.
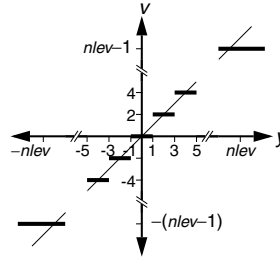
Transfer curve of a quantizer with an odd number of levels.

Supported Modulator Topologies

CIFB Structure

Even Order

Odd Order

CIFF Structure

Even Order

Odd Order

CRFB Structure

Even Order

Note that with NTFs designed using synthesizeNTF, omission of the $b_{2\ldots}$ coefficients in the CRFB structure will yield a maximally-flat STF.

Odd Order

CRFF Structure

Even Order

Odd Order

CRFBD Structure

Even Order
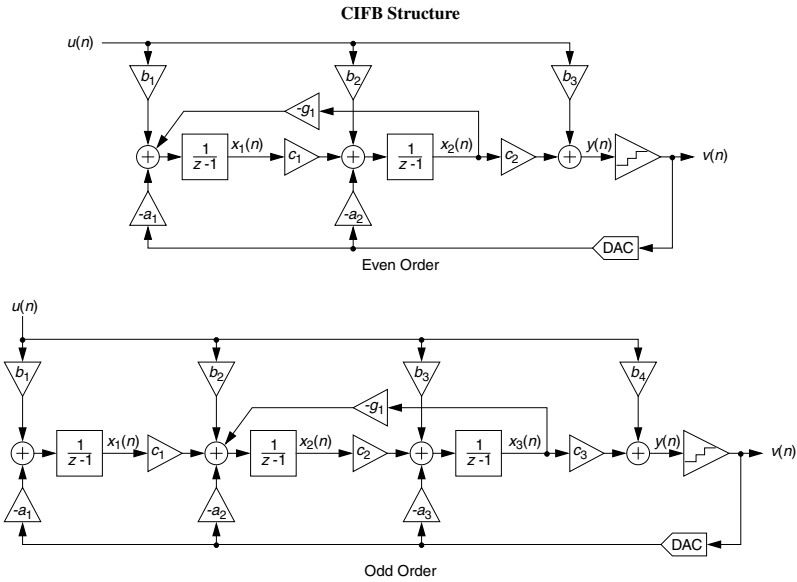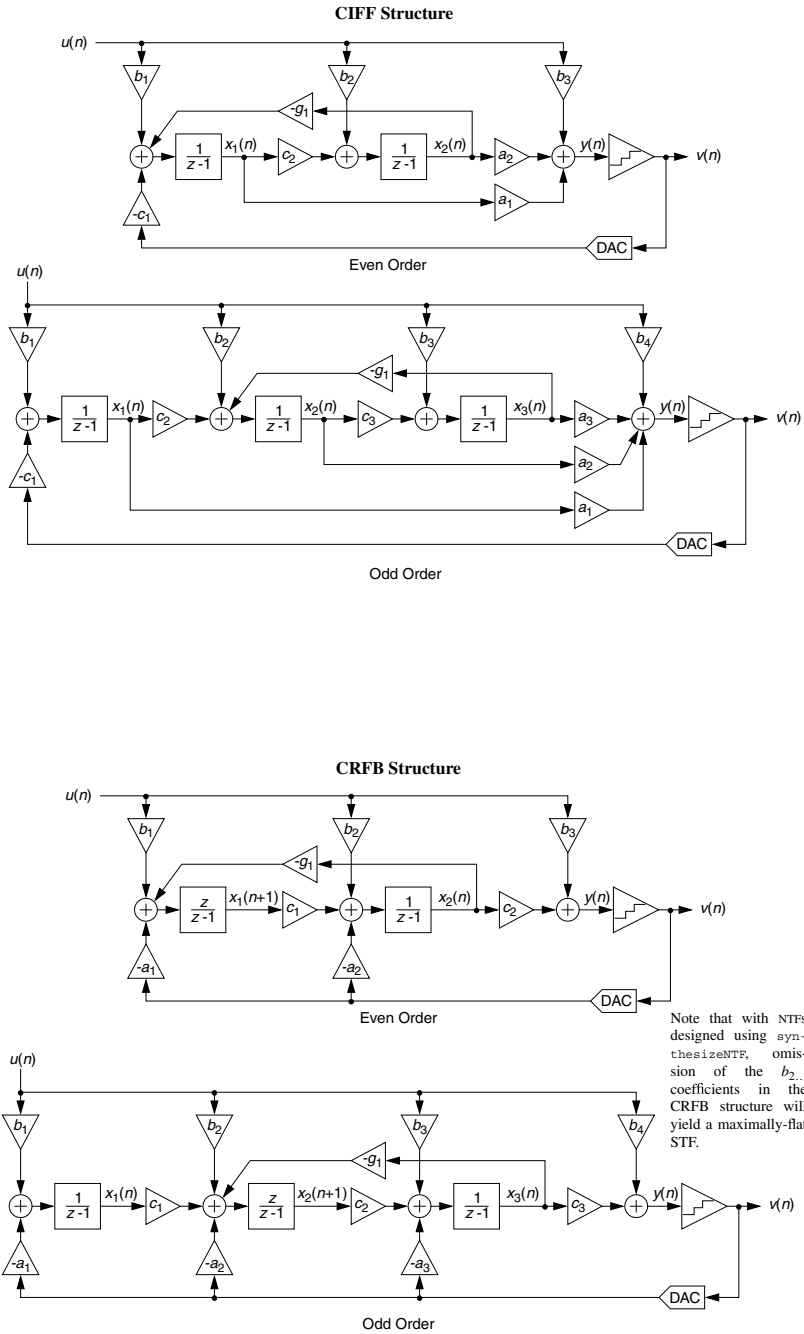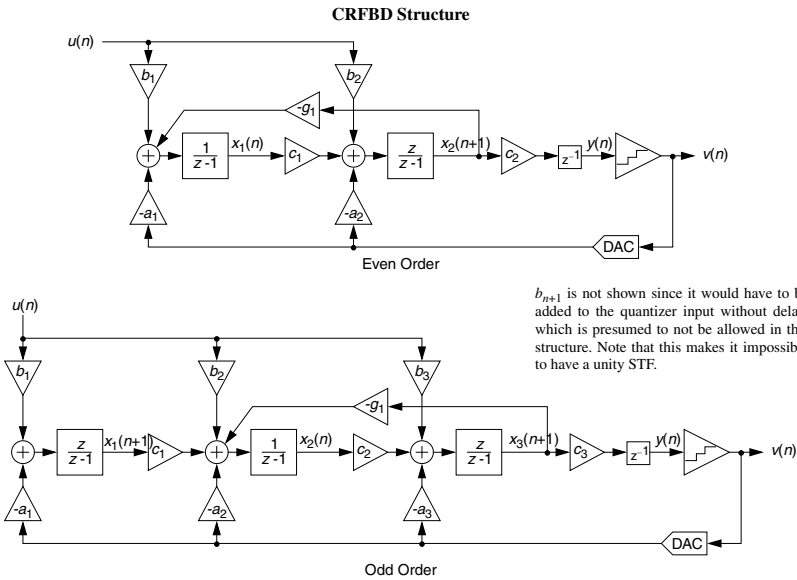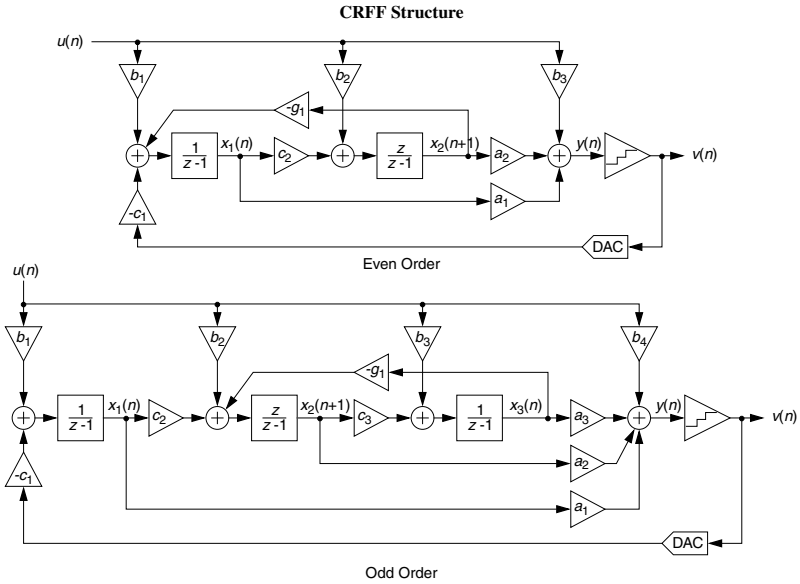
b_{n+1} is not shown since it would have to be added to the quantizer input without delay, which is presumed to not be allowed in this structure. Note that this makes it impossible to have a unity STF.

Odd Order

CRFFD Structure

Even Order

Odd Order

DSFB Structure (Developed with D. Senderowicz 2014-03)

3rd Order

4th Order
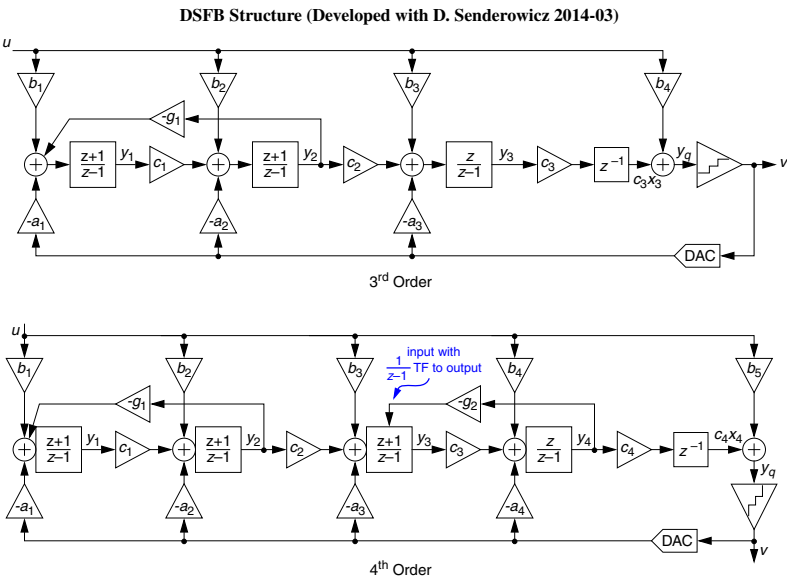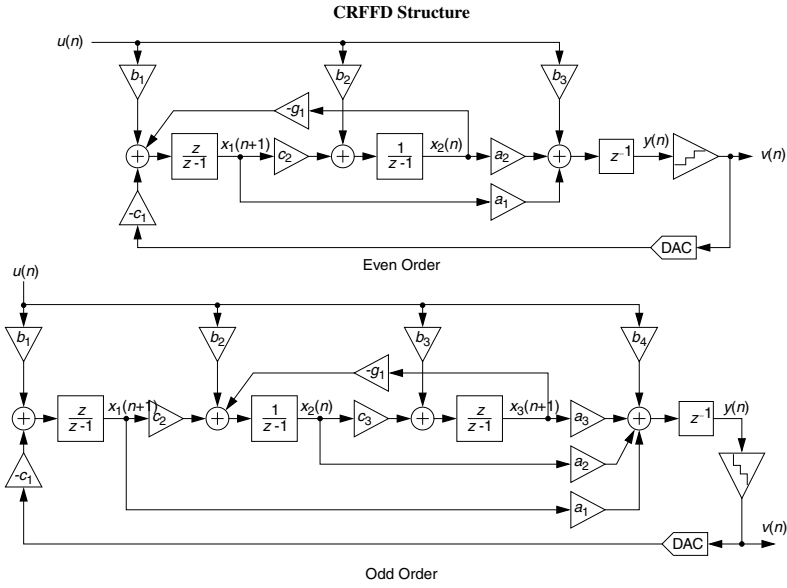
APPENDIX C

LINEAR PERIODICALLY TIME-VARYING SYSTEMS

We first review notions of linearity and time (in)variance. We then discuss an important class of systems called linear periodically time-varying (LPTV) systems, and some of their properties. It turns out that LPTV systems are very relevant in our journey to the $\Delta\Sigma$ world.

C.1 Linearity and Time (In)variance

Assume that an initially relaxed system (all initial conditions are zero), yields outputs $y_1(t)$ and $y_2(t)$ when excited by $x_1(t)$ and $x_2(t)$, respectively. The system is linear if it obeys superposition; namely, an input $\alpha x_1(t) + \beta x_2(t)$ yields the output $\alpha y_1(t) + \beta y_2(t)$.

$$
\begin{aligned}
x_1(t) &\rightarrow y_1(t), \\
x_2(t) &\rightarrow y_2(t), \\
\alpha x_1(t) + \beta x_2(t) &\rightarrow \alpha y_1(t) + \beta y_2(t).
\end{aligned}
$$

A system is linear *and* time-invariant (LTI) if, in addition to the constraints above, delaying the input by τ results in an output that is also delayed by τ. That is,

$$x_1(t - \tau) \rightarrow y_1(t - \tau). \tag{C.1}$$

An LTI system is characterized by its impulse response $h(t)$, that is the output of the initially relaxed system for an input $\delta(t)$. Due to time invariance, $h(t)$ can also be interpreted as the response observed at an arbitrary time t_1 due to an impulse applied at a time t before the time of observation, namely, at time $(t_1 - t)$.
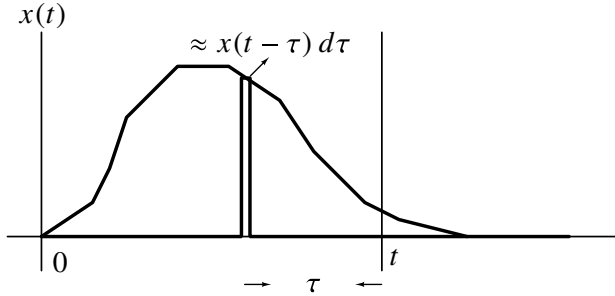
Figure C.1   Response of an LTI system at time t to an input $x(t)$.

How does the system respond to an arbitrary input $x(t)$? To determine this, we express the input as a sum of slivers with width $d\tau$ and height $x(t - \tau)$, as shown in Figure C.1. The response at time t due to the sliver is given by $x(t - \tau) \, d\tau \cdot h(\tau)$. The output due to $x(t)$ can thus be obtained by summing the response due to all the slivers, and this is given by the familiar convolution integral

$$y(t) = \int_0^\infty \underbrace{x(t - \tau) \, d\tau}_{\substack{\text{size of impulse} \\ \text{applied at } (t-\tau)}} \underbrace{h(\tau)}_{\substack{\text{response to impulse} \\ \text{applied } \tau \text{ earlier}}} = \int_0^\infty h(\tau)x(t - \tau) \, d\tau. \tag{C.2}$$

A complex exponential is especially significant in the study of linear systems. When the LTI system is excited by $x(t) = e^{j2\pi ft}$, using the convolution integral above yields

$$y(t) = \int_0^\infty h(\tau)e^{j2\pi f(t-\tau)} \, d\tau = e^{j2\pi ft} \underbrace{\int_0^\infty h(\tau)e^{-j2\pi f\tau} \, d\tau}_{H(f)}. \tag{C.3}$$

It is thus seen that the output of an LTI system, when excited by a complex exponential, is simply a scaled version of the input. The "gain", which depends on the frequency of the input, is the complex number $H(f)$.

As seen above, $H(f)$ is the Fourier transform of the impulse response $h(t)$, and it can be thought of in the following way [1]:

$$H(f) = \frac{\text{Response to } e^{j2\pi ft}}{e^{j2\pi ft}}. \tag{C.4}$$

Thus, a complex exponential at frequency f results in a *steady-state* output that is the input scaled by a complex number $H(f)$. Conversely, if the output of an LTI system happens to

be a complex sinusoid at frequency f, it must follow that the input is also a sinusoid at f. In a circuit simulator, the frequency response $H(f)$ is obtained by running a .AC analysis.

Many networks used in practice are nonlinear. Further, they are often used around a time-invariant *operating point*, with the inputs themselves being called *small signals*. When linearized about a time-invariant operating point, the nonlinear network results in a time-invariant linear network. In a circuit simulator, the small-signal frequency response is obtained, therefore, by first running a .OP analysis, that yields the operating point and the small-signal LTI network, after that a .AC analysis is performed.

C.2   Linear Time-Varying Systems

$$x(t) \longrightarrow \bigotimes \longrightarrow y(t)$$
$$g(t)$$
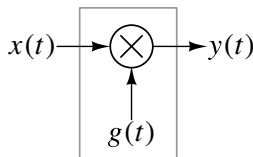
Figure C.2   An example of a linear system that is time varying.

Figure C.2 shows an example of a system whose gain varies with time. It is linear since

$$x_1(t) \quad \rightarrow \quad g(t)x_1(t),$$
$$x_2(t) \quad \rightarrow \quad g(t)x_2(t),$$
$$\alpha x_1(t) + \beta x_2(t) \quad \rightarrow \quad g(t)(\alpha x_1(t) + \beta x_2(t)).$$

However, it is not time-invariant:

$$x_1(t) \quad \rightarrow \quad g(t)x_1(t),$$
$$x_1(t - t_1) \quad \rightarrow \quad g(t)x_1(t - t_1) \neq g(t - t_1)x_1(t - t_1). \qquad (C.5)$$

Such systems are best described as being linear time-varying (LTV). As in the LTI case, an LTV system is characterized by an impulse response – however, as the reader may well know, this impulse response depends on the time at that the input impulse is applied. The response of an LTV system at a time t, due to an impulse applied at a time $t - \tau$, is denoted by $h(t, \tau)$:[1].

$$\text{Impulse response} = h(\underbrace{t}_{\substack{\text{time of} \\ \text{observation}}}, \quad \underbrace{\tau}_{\substack{\text{impulse applied } \tau \text{ before} \\ \text{time of observation}}}). \qquad (C.6)$$

To reiterate, the first variable in the impulse response denotes the time of observation. The second variable indicates that the system was excited by an impulse launched at a time τ *before* the time of observation.

---

[1]Recall that in an LTI system, the output of the system at t when it is excited by an impulse at $t - \tau$ is simply $h(\tau)$.

Thus, the response of an LTV system not only depends on how long before the observation time it was excited by the impulse but also on the time of observation. Contrast this to an LTI system, whose response depends *only* on how long before the observation time it was excited by the impulse. An LTI system, therefore, can be thought of as a special case of an LTV system whose impulse response satisfies $h(t, \tau) = h(\tau)$.
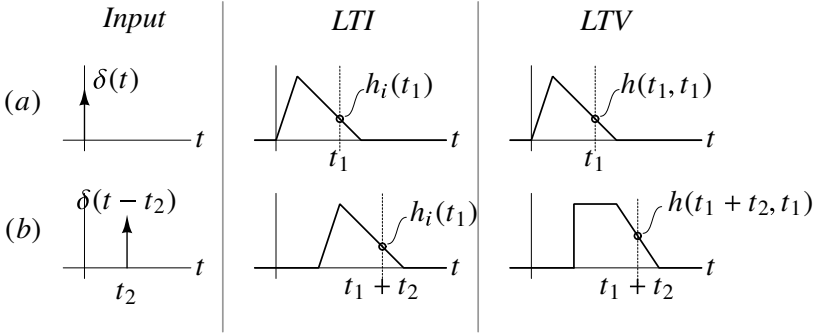
Figure C.3   Impulse responses of LTI and LTV systems.

The responses of LTI and LTV systems to impulse excitations are illustrated in Figure C.3. Part (a) of the figure shows the responses of example LTI and LTV systems to an impulse applied at $t = 0$. The output of the former is $h_i(t)$, where h_i is the impulse response. At an observation time t_1, the output is $h_i(t_1)$.

The output of the LTV system is given by $h(t, t)$, where h denotes the impulse response. When observed at a time t_1, the output is $h(t_1, t_1)$. This is to be interpreted as the output at t_1 due to an impulse applied t_1 prior to the time of observation – that is, the input is $\delta(t - (t_1 - t_1)) = \delta(t)$.

When the input is delayed by t_2, as shown in Figure C.3(b), the output of the LTI system is given by $h_i(t - t_2)$. Thus, if the time of observation moves by t_2 to $t_1 + t_2$, the output remains $h_i(t_1 + t_2 - t_2) = h_i(t_1)$. What happens in the LTV case? When observed at $t_1 + t_2$, the output is seen to be $h(t_1 + t_2, t_1)$. This is not (necessarily) equal to $h(t_1, t_1)$, since the output of an LTV system not only depends on the difference between the times of observation and excitation but also on the absolute time at that the output is observed.

How does an LTV system respond to the complex sinusoid $x(t) = e^{j2\pi ft}$? Proceeding as we did in the LTI case, we obtain

$$y(t) = \int_0^\infty h(t, \tau)e^{j2\pi f(t-\tau)}\, d\tau = e^{j2\pi ft} \underbrace{\int_0^\infty h(t, \tau)e^{-j2\pi f\tau}\, d\tau}_{H(f,t)}. \tag{C.7}$$

We see that, as in the time-invariant case, the input sinusoid is scaled by a complex number $H(f, t)$. However the "gain" experienced by the sinusoid is not only a function of its frequency (as in the LTI case) but also a function of time. This makes sense, since the system is varying with time. Further, we note that the (time-varying) frequency response can be interpreted, as in the time-invariant case, as

$$H(f, t) = \frac{\text{Response to } e^{j2\pi ft}}{e^{j2\pi ft}}. \tag{C.8}$$

C.3   Linear Periodically Time-Varying (LPTV) Systems
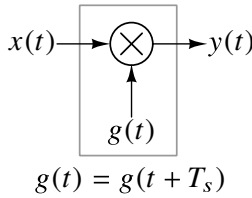
$$g(t) = g(t + T_s)$$

Figure C.4   An example of a linear system that is periodically time varying.

Consider now the system shown in Figure C.4, where the gain $g(t)$ varies periodically with time, meaning $g(t) = g(t + T_s)$. This is an example of a linear periodically time-varying (LPTV) system. It is a special case of an LTV system whose impulse response satisfies

$$h(t, \tau) = h(t + T_s, \tau). \tag{C.9}$$

In other words, the shape of the response remains unchanged if the time of observation of the output (t) and the time at that the system is excited (t_1) are *both* shifted by T_s. T_s, that is a characteristic of the system, is called the period of the LPTV system.
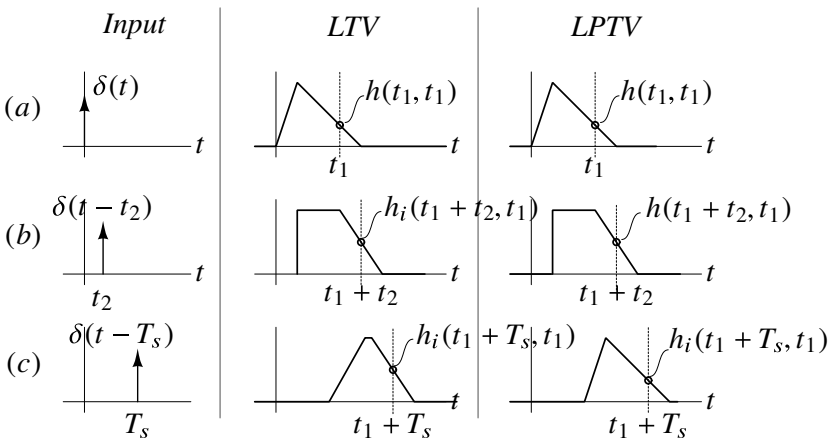
Figure C.5   Impulse responses of LTV and LTPV systems.

Figure C.5 compares the responses of LTV and LPTV systems when excited by impulses. In the LPTV case, we see that the output, when the system is excited by an impulse delayed in time by "this special" T_s, is simply a time-delayed version of the response obtained when the impulsive input occurs at $t = 0$. In fact, if one only considers the first and third rows of Figure C.5, the LPTV system could be mistaken to be a time-invariant one! In that sense, an LPTV system can be considered to be "closer" to an LTI system than to a time-varying one.

How does an LPTV system respond to an an input $x(t) = e^{j2\pi ft}$? From (C.7), we see that the output is given by

$$y(t) = H(f, t)e^{j2\pi ft}.$$

Since (C.9) holds, we observe that

$$
\begin{aligned}
H(f,t) &= \int_0^\infty h(t,\tau)e^{-j2\pi f\tau}\,d\tau \\
&= \int_0^\infty h(t+T_s,\tau)e^{-j2\pi f\tau}\,d\tau \\
&= H(f,t+T_s).
\end{aligned}
\tag{C.10}
$$

Thus, the frequency response $H(f,t)$ of an LPTV system is periodic with T_s. What this means is that the output of an LPTV system excited by $e^{j2\pi ft}$ can be thought of as $e^{j2\pi ft}$ that is scaled by a gain that changes periodically with time. Again, this is intuitively satisfying, since the system is varying periodically.
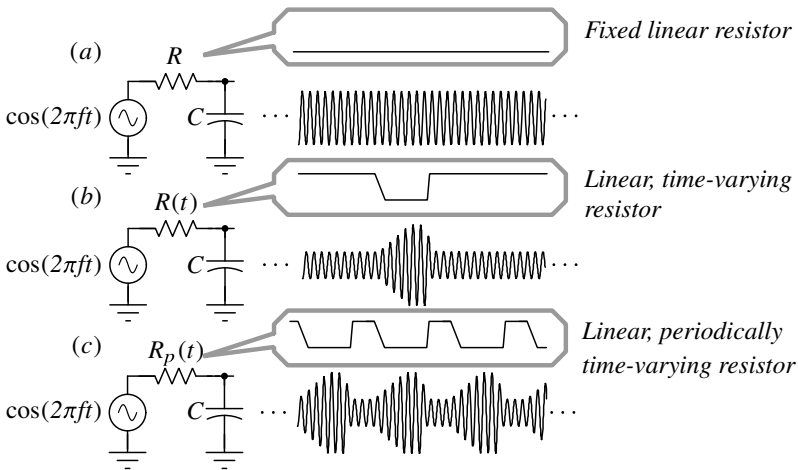
Figure C.6   Response of RC networks to a sinusoidal input. (a) Linear, time-invariant resistor; (b) linear, time-varying resistor; and (c) linear, periodically time-varying resistor.

Figure C.6 illustrates example LTI, LTV, and LPTV systems excited by a sinusoidal input. In part (a) of the figure, R and C are fixed. The voltage across the capacitor is a sinusoid with a constant envelope. In Figure C.6(b), the resistor is linear, but varies with time. The envelope of the output changes with time, becoming larger for smaller values of resistance. The resistor in Figure C.6(c) varies periodically with time. As a result, the gain of the input tone also varies periodically with time, as is evident from the shape of the envelope.

Since $H(f,t)$ of an LPTV system is periodic with time-period T_s, it can be expanded as a Fourier series in t, resulting in

$$
H(f,t) = \sum_{k=-\infty}^{\infty} H_k(f)e^{j2\pi f_s kt} \quad , \quad f_s = 1/T_s.
\tag{C.11}
$$

The coefficients of the Fourier series $H_k(f)$, called the harmonic transfer functions, are easily obtained using

$$
H_k(f) = \frac{1}{T_s}\int_0^{T_s} H(f,t)e^{-j2\pi f_s kt}\,dt.
\tag{C.12}
$$

The response of an LPTV system to $e^{j2\pi ft}$ is therefore given by

$$H(f,t)e^{j2\pi ft} = \sum_{k=-\infty}^{\infty} \underbrace{H_k(f)}_{\substack{Harmonic \\ Transfer\ Function}} \underbrace{e^{j2\pi(f+kf_s)t}}_{\substack{frequencies \\ f+kf_s}}. \tag{C.13}$$

In circuit simulators capable of analyzing LPTV systems in the frequency domain, the $H_k(f)$ can be obtained by what is often referred to as the *periodic* AC analysis (.PAC analysis). In simulator parlance, $H_k(f)$ are often referred to as the kth side-band response.

We make the following observations.

a. The output of an LPTV system (varying periodically at f_s) driven by a sinusoid at f consists of frequency components at f, $f \pm f_s$, $f \pm 2f_s$, and so on. $H_k(f)$ represents the gain from the input (at frequency f) to the output (at frequency $f + kf_s$), as shown in Figure C.7.
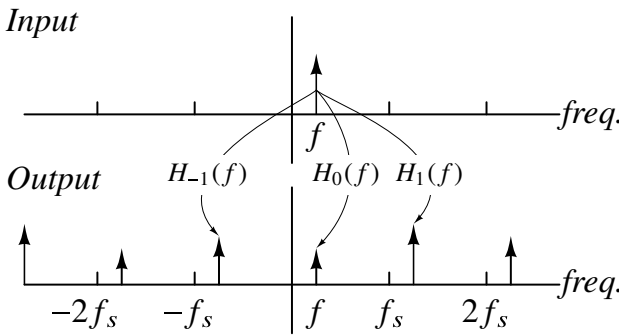
Figure C.7   In an LPTV network varying at f_s, and excited at a frequency f, the output consists of tones at frequencies $f + kf_s$, where k is an integer. $H_k(f)$ represents the gain from the input at frequency f and output at frequency $f + kf_s$.

b. By the same token, if the output of an LPTV system (varying periodically at f_s) consists of a sinusoid with frequency f, this is, in general, due to contributions from inputs at frequencies f, $f \pm f_s$, $f \pm 2f_s$, and so on.

Let us apply the techniques we have learned to the LPTV system of Figure C.4. The time-varying impulse response of the system is given by

$$h(t, \tau) = g(t - \tau)\delta(t - (t - \tau)) = g(t - \tau)\delta(\tau). \tag{C.14}$$

The time-varying frequency response of the system is, therefore,

$$H(f,t) = \int_0^\infty h(t, \tau)e^{-j2\pi f\tau}\, d\tau = \int_0^\infty g(t - \tau)\delta(\tau)e^{-j2\pi f\tau}\, d\tau = g(t). \tag{C.15}$$

Since $g(t)$ is periodic, it can be expanded as a Fourier series according to

$$g(t) = \sum_k g_k e^{j2\pi kf_s t}. \tag{C.16}$$

The harmonic transfer functions are thus given by $H_k(f) = g_k$.

<div align="center">

Example 1

</div>

Consider the system of Figure C.4, with $g(t) = A_{LO} \cos(2\pi f_s t)$ and $x(t) = A_{rf} \cos(2\pi f t)$.

$$y(t) = \frac{1}{2} A_{LO} A_{rf} \cos(2\pi(f - f_s)t) + \frac{1}{2} A_{LO} A_{rf} \cos(2\pi(f + f_s)t). \quad \text{(C.17)}$$

We see that an input tone with frequency f produces output tones at frequencies $f \pm f_s$. The harmonic transfer functions are given by $H_{\pm 1}(f) = (A_{LO}/2)$, and $H_{\pm k}(f) = 0$ for $k \neq \pm 1$.

We now ask the inverse question: say we observe that the output y is a tone at frequency f. What could x be? Since $H_k(f)$ is nonzero only when $k = \pm 1$, it follows that the input must consist of tones at $(f \pm f_s)$.

How does one determine the harmonic transfer functions of an LPTV in a circuit simulator? Akin to the `.AC` analysis used in an LTI network, simulators equipped to analyze LPTV systems perform what is often referred to as the `.PAC` analysis, where `PAC` refers to *periodic AC*.

Many practically useful networks are nonlinear, and often used around a periodically time-varying *operating point*, with the inputs themselves being called *small signals*. The nonlinear system, when linearized about a periodically time-varying operating point, yields an LPTV network. In a circuit simulator, the small-signal frequency response is obtained, therefore, by first running a `.PSS` analysis[2], that yields the periodic operating point and the small-signal LPTV network, after that a `.PAC` analysis is performed.

The reader might wonder what all the theory above has to do with our study of CTΔΣMs. To see this, consider a purist's model of a CTΔΣM, where quantization er-
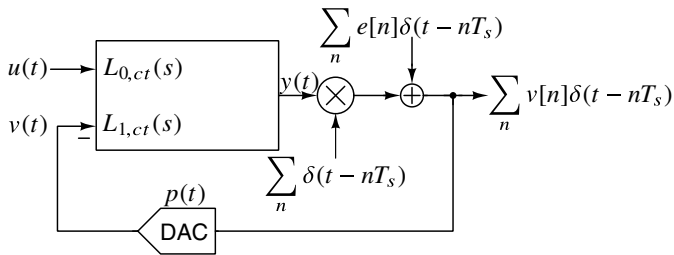
Figure C.8   A CTΔΣM as an LPTV system.

ror is assumed to be additive (Figure C.8). Sampling the loop-filter output is equivalent to multiplying $y(t)$ with a Dirac delta train. Quantization noise is modeled as an impulse train with random amplitudes. The output is an impulse train that is fed back after being filtered

---

[2]`PSS` stands for *periodic steady state*.

by the DAC pulse $p(t)$. The CTΔΣM is, therefore, an LPTV system with two inputs (u and e) and one output v, varying with frequency f_s (time-period T_s).

Apart from being periodically time-varying, a CTΔΣM has another distinguishing feature. It is an LPTV system where the output of interest is sampled (in this case y, corrupted by e). Further, y is sampled at a frequency f_s, that is also the frequency at that the LPTV system varies. It turns out that this has fundamental and important consequences, as shown next.

C.4 LPTV Systems with Sampled Outputs

$$x(t) = e^{j2\pi ft} \longrightarrow \boxed{\begin{array}{c} LPTV\ System \\ Period\ T_s \end{array}} \xrightarrow{y(t)} \otimes \longrightarrow \sum_n y(nT_s)\delta(t - nT_s)$$

$$\sum_n \delta(t - nT_s)$$
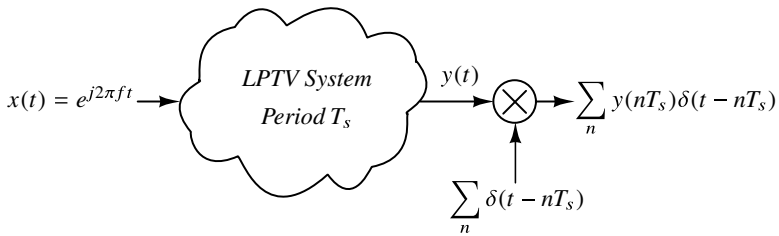
Figure C.9   An LPTV system (varying at a rate f_s) whose output is sampled at f_s.

Consider the LPTV system, varying with period T_s, shown in Figure C.9. It is excited by a complex exponential $x(t) = e^{j2\pi ft}$. The output $y(t)$ is sampled with the same period T_s as shown in the figure. From the discussion in the previous section, a CTΔΣM is a system of this type.

Since the system is LPTV, we have

$$y(t) = \sum_{k=-\infty}^{\infty} H_k(f)e^{j2\pi(f+kf_s)t}. \tag{C.18}$$

The samples of $y(t)$ are given by

$$y(nT_s) = \sum_{k=-\infty}^{\infty} H_k(f)e^{j2\pi(f+kf_s)nT_s} = e^{j2\pi fnT_s} \sum_{k=-\infty}^{\infty} H_k(f). \tag{C.19}$$

Consider now the system of Figure C.10. It is a linear *time-invariant* system, chosen so that its frequency response $H_{eq}(f)$ is *chosen* to be $\sum_k H_k(f)$, where $H_k(f)$ are the harmonic transfer functions of the LPTV system of Figure C.9. If this LTI system is excited by $e^{j2\pi ft}$, its output is

$$\hat{y}(t) = e^{j2\pi ft} \sum_{k=-\infty}^{\infty} H_k(f). \tag{C.20}$$

When sampled at a rate $f_s = 1/T_s$, we obtain

$$\hat{y}(nT_s) = e^{j2\pi nfT_s} \sum_{k=-\infty}^{\infty} H_k(f). \tag{C.21}$$
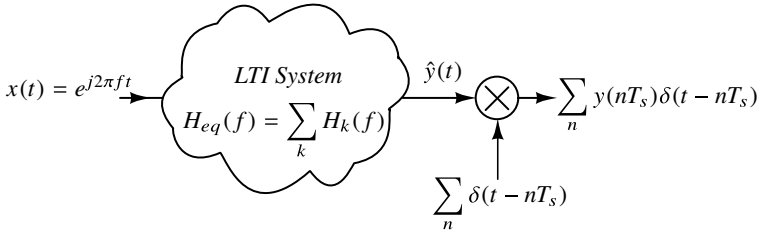
Figure C.10   An LTI system whose output sampled at f_s yields the *same* sequence as the system of Figure C.9.

From (C.19) and (C.21), we see that as far as *output samples* are concerned, an LPTV system whose output is sampled at f_s is equivalent to an LTI system with output sampled at f_s. The equivalent LTI filter has a frequency response [2]

$$H_{eq}(f) = \sum_{k=-\infty}^{\infty} H_k(f). \tag{C.22}$$

Since any arbitrary input $x(t)$ can be represented as a sum of complex exponentials via the Fourier transform, it follows that the output samples of an LPTV system (when the sampling rate is the same as that at that the system is varying) can be thought of as being obtained by exciting an LTI filter by $x(t)$ and sampling its output at a rate f_s.
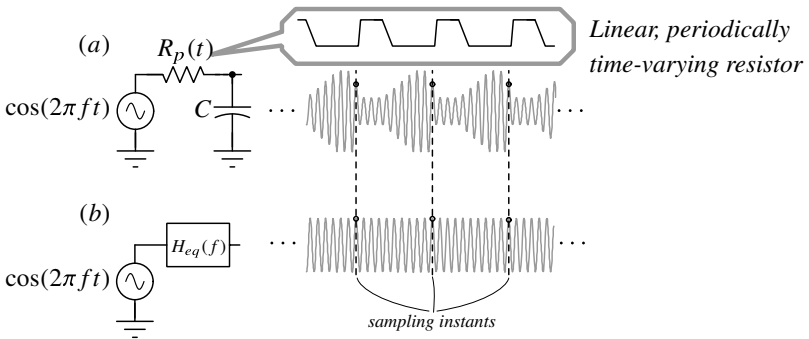
Figure C.11   (a) An LPTV system varying at f_s excited by a sinusoid, with output sampled at f_s, and (b) the output of an equivalent LTI system sampled at f_s.

The result derived above makes intuitive sense due to the following. When an LPTV system is excited by a tone at f, the output comprises of tones at frequencies $f + k f_s$, where k is an integer. When sampled at f_s, frequency components higher than f_s are aliased to f. Thus, if one is only interested in the samples of the system's output, they could as well be produced by a properly chosen LTI filter acting on an input tone at a frequency f. We also emphasize that the equivalence holds only for samples, and not for the waveforms. Referring to Figs. C.9 and C.10, we note that $y(nT_s) = \hat{y}(nT_s)$, but $y(t)$ need not equal $\hat{y}(t)$.

How does one interpret the result and discussion above in the context of a CT$\Delta\Sigma$M? Recall that in Chapter 8, we concluded (neglecting shaped quantization noise) that the
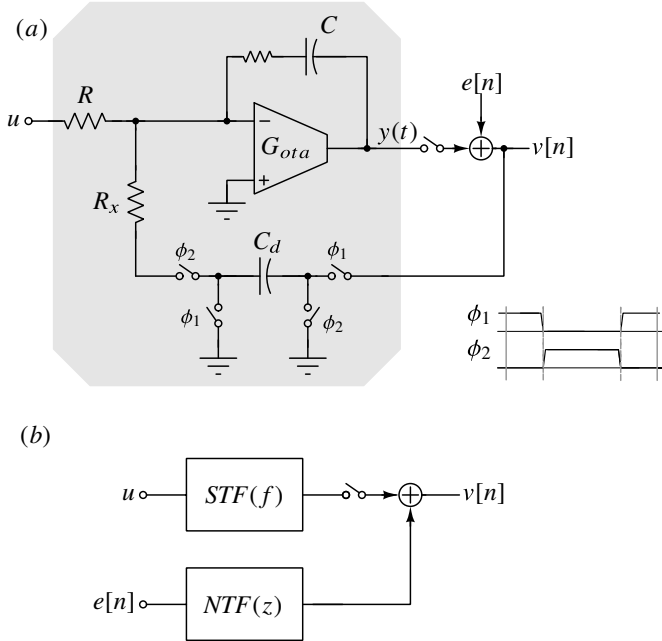
Figure C.12   A first-order CTΔΣM with a switched-capacitor feedback DAC. $Y(s)$ cannot be expressed as $L_{0,ct}U(s) - L_{1,ct}V(s)$. (b) Model for the CTΔΣM.

output sequence of a CTΔΣM (of the kind depicted in Figure C.8) can be thought of as being obtained by first filtering the input by a time-invariant continuous-time filter with transfer function

$$STF(f) = L_{0,ct}(j2\pi f)NTF(e^{j2\pi f}) \tag{C.23}$$

and then sampling the resulting waveform at a rate f_s. We showed this by manipulating the signal flow graph of the modulator, by separating it into "continuous-time" and "discrete-time" portions in an effort to make analysis tractable. However, it is not always possible to separate the modulator in this fashion. A case in point is a CTΔΣM with a switched-capacitor feedback DAC, shown in Figure C.12(a), that we encountered in Chapters 9 and 10. Due to the switching nature of the DAC, it is not possible to express $Y(s)$ as $L_{0,ct}U(s) - L_{1,ct}V(s)$.

The result of (C.22), however, provides a fundamental basis for our being able to model the input-output path of the CTΔΣM by means of an LTI filter whose output is sampled. In the context of the modulator of Figure C.12(a), it means that even though the loop-filter output cannot be expressed as $L_{0,ct}U(s) - L_{1,ct}V(s)$, the model for the modulator remains that shown in Figure C.12(b), namely a CT filter (the STF) operating on the signal whose output is sampled and added to the output of a DT filter (the NTF) operating on the quantization error. We thus realize that our model for a CTΔΣM result is far more fundamental.

How do we determine the transfer function of the equivalent LTI system $H_{eq}(f)$, given an LPTV system? One way of doing this is to determine $H_k(f)$ of the latter, and then use (C.22). In many cases, however, it is easier to proceed in the time domain, as illustrated using Figure C.13.
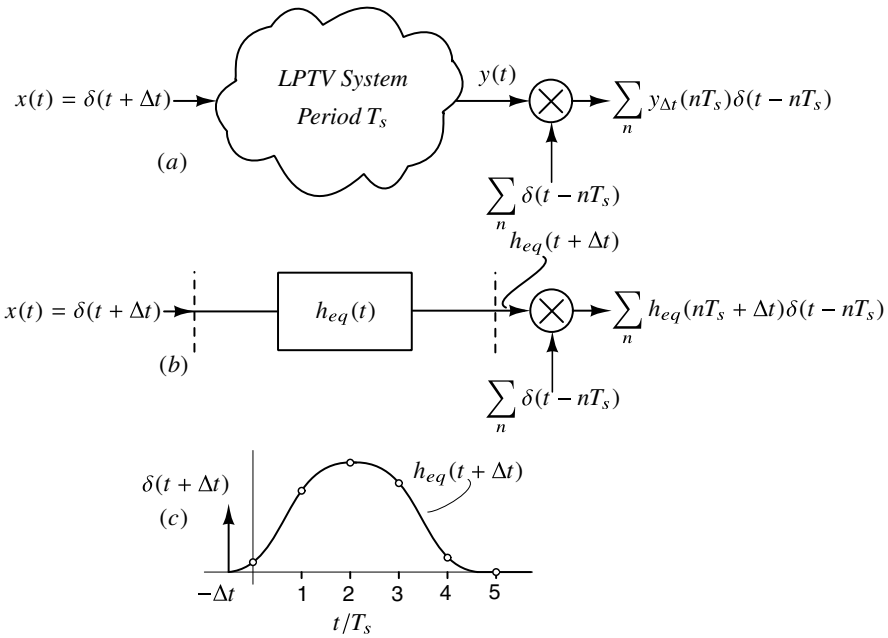
Figure C.13   Determining the impulse response of the equivalent LTI filter corresponding to an LPTV system with sampled outputs. (a) Excite the LPTV system with an impulse at $-\Delta t$. The sampled output sequence obtained is denoted by $y_{\Delta t}(nT_s)$. (b) Exciting the equivalent LTI filter with $\delta(t + \Delta t)$ yields $h_{eq}(nT_s + \Delta t)$. (c) The conceptual output of the equivalent LTI filter and its samples.

The impulse response of the equivalent LTI filter is denoted by $h_{eq}(t)$. Exciting the LPTV system by an impulse $\delta(t + \Delta t)$ and sampling the output $y(t)$ yields a sequence that we denote by $y_{\Delta t}(nT_s)$. The output of the LTI filter (with input $\delta(t + \Delta t)$), when sampled, yields the sequence $h_{eq}(nT_s + \Delta t)$. By the principle of equivalence we have been discussing, it must follow that

$$h_{eq}(nT_s + \Delta t) = y_{\Delta t}(nT_s), \qquad (C.24)$$

as seen in Figure C.13(c). Thus, the output sequence of the LPTV system, when the input is $\delta(t + \Delta t)$, yields the samples $h_{eq}(nT_s + \Delta t)$ of the equivalent LTI filter. By sweeping Δt from 0 to T_s in sufficiently fine increments, we should be able to construct $h_{eq}(t)$ in its entirety.

We demonstrate the technique by determining the STF of a first-order CT$\Delta\Sigma$M with an NRZ DAC, as shown in Figure C.14. The sampling rate of the modulator is 1 Hz, and
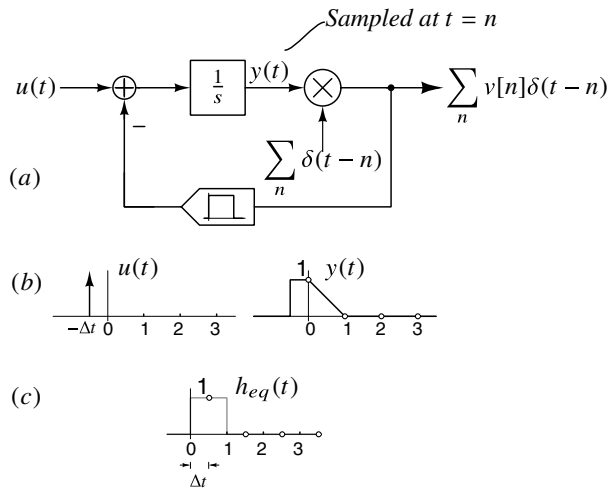
Figure C.14   (a) CT-MOD1. (b) $u(t) = \delta(t + \Delta t)$ and the resulting $y(t)$ and (c) constructing $h_{eq}(t)$.

the output sequence of interest is $y(t)$ sampled at multiples of 1 s. As discussed above, to construct the impulse response corresponding to the STF, we excite the modulator with $u(t) = \delta(t + \Delta t)$ and "measure" the sequence $y[n]$. $y(t)$ is a step that goes to 1 at $t = -\Delta t$. It is sampled at $t = 0$ and fed back through a filter whose impulse response is the NRZ pulse. $y(t)$, therefore, goes to zero in a linear fashion, as seen in Figure C.14(b), and remains 0 after $t = 1$. The samples of $y_{\Delta t}[n]$ are given by $1, 0, 0, \cdots$. It is easy to see that $y_{\Delta t}[n]$ remains $1, 0, 0, \cdots$ for $0 < \Delta t < 1$. $h_{eq}(t)$ is therefore a rectangular pulse, as shown in part (c) of the figure. Thus, $STF(f) = e^{-j\pi f} sinc(f)$, that is in agreement with the results obtained in Chapter 8.

Determining the impulse response of the equivalent LTI filter by applying successively advanced impulses, as shown in Figure C.13, is useful but time-consuming. Fortunately, by using the concept of inter-reciprocity, $h_{eq}(t)$ can be found by a one-shot process. The key result that enables this is explained with the aid of Figure C.15. Part (a) of the figure shows an LPTV network $\mathcal{N}$ being excited by a current impulse at $t = t_i$. The output of interest is the voltage $v_2(t)$, sampled at $f_s = 1/T_s$ with a timing offset of t_o, namely, the sequence $v_2[nT_s + t_o]$. It is possible to find another LPTV network with the same time
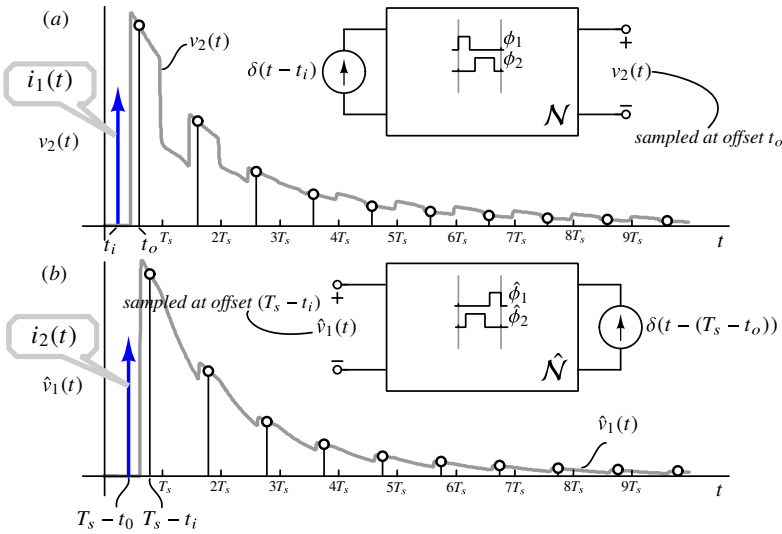
Figure C.15   Determining the impulse response of the equivalent LTI filter corresponding to an LPTV system with sampled outputs using the inter-reciprocal (or adjoint) network.

period T_s, called the inter-reciprocal (or adjoint) network, denoted by $\hat{\mathcal{N}}$ (Figure C.15(b)) with the following interesting property.

Let the adjoint be excited by an impulse current at its *output* port at time $(T_s - t_o)$. The voltage at the input port $\hat{v}_1(t)$, when sampled at a timing offset $(T_s - t_i)$, yields exactly the same sequence $v_2[nT_s + t_o]$. That is,

$$\hat{v}_1[nT_s + T_s - t_i] = v_2[nT_s + t_o]. \tag{C.25}$$

Since $\hat{\mathcal{N}}$ is an LPTV network with period T_s, exciting it at $T_s - t_o$ and observing the response at $(nT_s + T_s - t_i)$ is equivalent to exciting it at $-t_o$ and observing the response at $(nT_s - t_i)$. Since t_i and t_o in the experiment of Figure C.15(a) has turned into $-t_i$ and $-t_o$, and since the signals controlling the time-varying elements are flipped in time, the term "time-reversed" is often used in connection with the adjoint network of Figure C.15(b).

The adjoint network $\hat{\mathcal{N}}$ has the same graph as $\mathcal{N}$, and can be derived from $\mathcal{N}$ by applying the following element-by-element substitution rules shown in Table C.1:

a. A branch in $\mathcal{N}$ that is a linear resistor, capacitor, or inductor remains unchanged in $\hat{\mathcal{N}}$.

b. A periodically operated switch in $\mathcal{N}$ controlled by a waveform $\phi(t)$ is replaced in $\hat{\mathcal{N}}$ by a switch that is controlled by $\phi(-t)$.

c. Linear controlled sources in $\mathcal{N}$ are replaced by appropriate linear controlled sources in $\hat{\mathcal{N}}$. For instance, a CCCS in $\mathcal{N}$ is replaced by a VCVS in $\hat{\mathcal{N}}$, with the controlling and controlled ports interchanged, as seen in Table C.1.

d. If $\mathcal{N}$ is expressed as a signal flow graph, summing junctions, and pick-off points of $\mathcal{N}$ are replaced by pick-off points and summing junctions, respectively, in $\hat{\mathcal{N}}$.

Table C.1   Transformations of linear controlled sources, summing junctions, pick-off points, multipliers, and periodically operated switches from $\mathcal{N}$ to $\hat{\mathcal{N}}$.

$\mathcal{N}$	$\hat{\mathcal{N}}$
v_1 μv_1	μi_2 i_2
i_1 μi_1	μv_2 v_2
v_1 $g_m v_1$	$g_m v_2$ v_2
i_1 $R i_1$	$R i_2$ i_2
$x \longrightarrow \oplus \longrightarrow z$ y	$x \longleftarrow \bullet \longrightarrow z$ y
$z \longrightarrow \bullet \longrightarrow x$ y	$z \longleftarrow \oplus \longleftarrow x$ y
$x \longrightarrow \otimes \longrightarrow z$ $g(t)$	$x \longleftarrow \otimes \longleftarrow z$ $g(-t)$
$\phi(t)$	$\phi(-t)$

e. A time-varying gain $g(t)$ in $\mathcal{N}$ is replaced by another time-varying gain $g(-t)$ in $\hat{\mathcal{N}}$.

The adjoint network greatly simplifies the process of determining $h_{eq}(t)$ of $\mathcal{N}$, as discussed below. In what follows, the output sequence of interest is assumed to be $v_2(nT_s)$ (in other words, $t_o = 0$). We denote the impulse response of the equivalent LTI filter by $h_{eq}(t)$. To obtain $h_{eq}(nT_s + \Delta t)$, as we discussed earlier, $\mathcal{N}$ should be excited by $\delta(t + \Delta t)$ and $v_2(t)$ should be sampled at $t = nT_s$, as shown in Figure C.16(a). In the inter-reciprocal network, this corresponds to exciting the adjoint $\hat{\mathcal{N}}$ with $\delta(t)$ at its "output" port, but sampling $\hat{v}_1(t)$ with a timing offset of $-(-\Delta t) = \Delta t$. In other words, $h_{eq}(nT_s + \Delta t) = v_2(nT_s) = \hat{v}_1(nT_s + \Delta t)$.
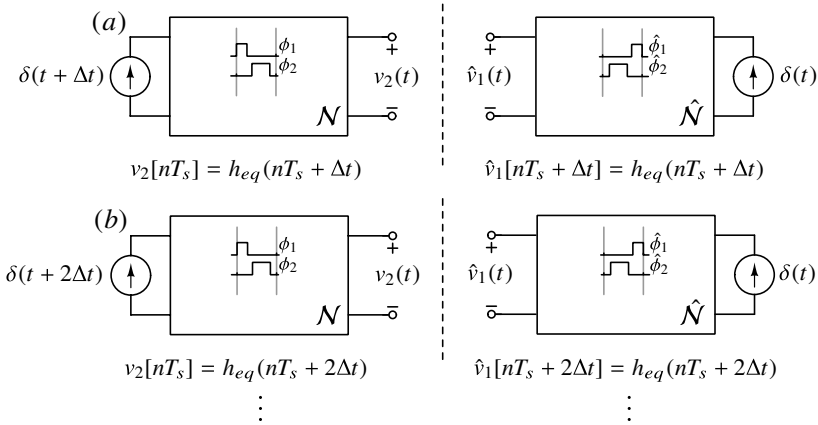
Figure C.16   (a) Obtaining $h_{eq}(nT_s + \Delta t)$ using the original and adjoint networks and (b) obtaining $h_{eq}(nT_s + 2\Delta t)$.

The next step is to obtain $h_{eq}(nT_s + 2\Delta t)$. This is accomplished by driving $\mathcal{N}$ with a current $\delta(t + 2\Delta t)$ and sampling $v_2(t)$ at $t = nT_s$, as shown in Figure C.16(b). In the adjoint, the output port should be driven by a current $\delta(t)$ (as in Figure C.16(a)), but $\hat{v}_1(t)$ should now be sampled with a timing offset of $-(-2\Delta t) = 2\Delta t$. Repeating this for $0 \leq \Delta t < T_s$, we see that $h_{eq}(t)$ is the *waveform* $\hat{v}_1(t)$ that manifests at the "input" port of $\hat{\mathcal{N}}$ when its "output" port is excited by an impulse current at $t = 0$. Thus, multiple experiments, each involving excitation with an impulse, that are needed to determine $h_{eq}(t)$ from $\mathcal{N}$ are not necessary – simply exciting the adjoint *once* is sufficient.

This is best illustrated with an example. We (again) attempt to find $h_{eq}(t)$ for CT-MOD1, shown in Figure C.17(a). The output is the sampled version of $y(t)$, sampled at $t = n$. The first step in the process is to draw the adjoint signal flow graph (Figure C.17(b)). The input of CT-MOD1 is the output of relevance in the adjoint. The directions of the integrator, and DAC (that acts like a filter with a rectangular impulse response) are reversed. The Dirac delta train is time-reversed; since the impulse appear at multiples of 1 s, the reversal has no impact on the waveform. The "output" port in this adjoint signal flow graph should be excited by an impulse at $t = 0$. The output of the integrator in the adjoint, denoted by $\hat{u}$, has a step at $t = 0$, as seen in Figure C.17(c). This is fed back after being convolved by the NRZ DAC pulse, resulting in a feedback waveform that is initially a unit ramp. At $t = 1$, the ramp is sampled and fed into the integrator. Since $\hat{v}(1) = 1$, $\hat{u}(t)$ becomes zero beyond $t = 1$. As a consequence, $\hat{v}(t)$ ramps linearly downwards, attaining a
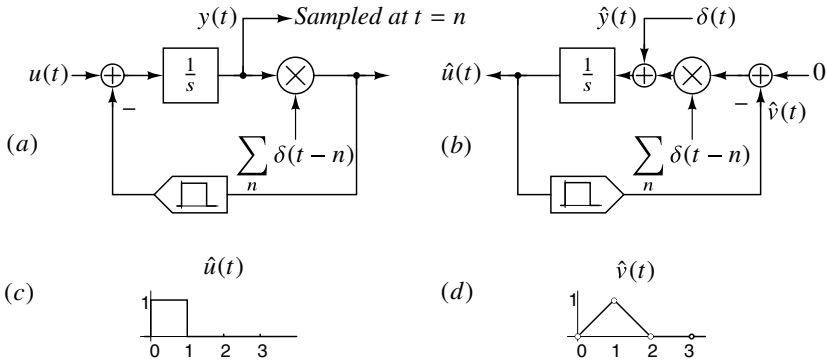
Figure C.17   Determining $h_{eq}(t)$ for CT-MOD1 using the adjoint signal flow graph: (a) CT-MOD1 (b) The adjoint corresponding to CT-MOD1, excited by an impulse at its "output" port. (c) $\hat{u}(t)$, that is the impulse response corresponding to the STF and (d) $\hat{v}(t)$.

value of 0 at $t = 2$, and it remains zero thereafter, as shown in Figure C.17(d). The impulse response corresponding to the STF, therefore, is the unit rectangular function, as we would expect.

C.4.1   Multiple Inputs
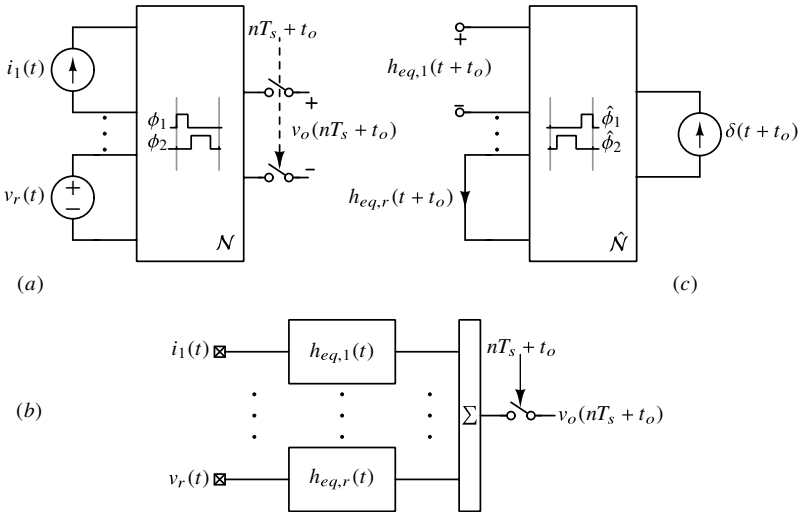
Figure C.18   (a) Original network with multiple inputs and one output, sampled at a timing offset t_o. (b) Equivalent model with LTI filters and (c) determination of $h_{eq,1\cdots r}(t)$ using the adjoint network.

Figure C.18 shows the extension of our results to an LPTV network with multiple input sources. The output of interest is assumed (without loss of generality) to be $v_o(t)$, sampled at a timing offset of t_o. The r inputs can be voltage or current sources. The output

of the system can be represented as shown in Figure C.18(b), where $h_{eq,1,\cdots,r}(t)$ denote the impulse responses of LTI filters. Thanks to inter-reciprocity, *all* the impulse responses can be determined using just *one* time domain analysis as shown in Figure C.18(c), by exciting the adjoint network $\hat{N}$ with an impulsive current at time $-t_o$.

C.4.1.1 *Noise* If the multiple inputs in Figure C.18(a) were noise sources, the model of Figure C.18(b) simplifies the evaluation of the contributions of the individual noise sources and the total noise spectral density of the output sequence. In practice, individual noise sources are usually independent. Denoting the autocorrelation function of the lth noise process by $R_{n,l}(\tau)$, the corresponding function at the output of the equivalent LTI filter is given by

$$R_l(\tau) = R_{n,l}(\tau) * h_{eq,l}(\tau) * h_{eq,l}(-\tau), \tag{C.26}$$

where $*$ denotes convolution. The autocorrelation function of the noise sequence after sampling is given by

$$R_l[m] = R_l(mT_s). \tag{C.27}$$

The power spectral density of the output sequence due to the lth noise source is simply the Fourier transform of the sequence $R_l[m]$ in (C.27). Since the noise sources are independent, the autocorrelation function of the sampled output is given by

$$R[m] = \sum_l R_l(mT_s). \tag{C.28}$$

C.4.2 Alias Rejection in Continuous-Time Delta-Sigma Modulators Revisited

Having armed ourselves with the arsenal needed to tackle sampled LPTV systems, we are now in a position to deal with CTΔΣMs whose loop-filters are time-varying. Examples of such modulators are those that use switched-capacitor, or return-to-open DACs. As we discussed in reference to the modulator of Figure C.8, the relation $STF(f) = L_{0,ct}(j2\pi f)NTF(e^{j2\pi f})$ only applies when the loop-filter is time-invariant. How does one find the STF when the loop-filter is time-varying, as in Figure C.19(a)?

The quantizer samples the loop-filter's output $y(t)$, and the rate at that $y(t)$ is sampled is the same as that at which the loop-filter is varying. Under these circumstances, as we discussed earlier in this section, one can find an equivalent time-invariant transfer function $L_{eq,ct}(s)$, that when excited by $u(t)$ yields the same $y[n]$ that would result, had $u(t)$ been the input of a time-varying loop-filter. Let $l_{eq}(t)$ denote the impulse response corresponding to $L_{eq,ct}(s)$. To determine $l_{eq}(t)$, we would have to break the ΔΣ loop (Figure C.19(b)), set v to zero, and use the adjoint of the resulting LPTV system.

The procedure is best illustrated with an example. Consider the loop-filter of a CIFF CTΔΣM with a switched-capacitor DAC. When the loop is broken, and v is set to zero, the resulting network is as shown in Figure C.20(a). The input integrator is assumed to be of the active-RC type, and uses a single-stage OTA. The rest of the loop-filter, assumed to be time-invariant, is represented by $L_2(s)$. The output of L_2 is sampled on the edge of ϕ_s.

The adjoint network is shown in Figure C.20(b). The OTA (that is a VCCS) in the original network is replaced by an OTA whose input and output ports are interchanged. The switch control signals are time reversed – in other words, if $\phi(t)$ denotes a switch
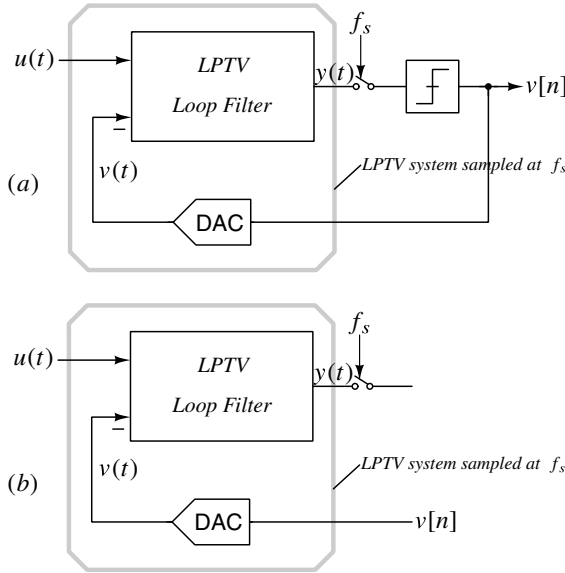
Figure C.19   (a) CTΔΣM with a time-varying loop-filter. (b) The LPTV loop-filter, whose output is sampled at f_s, can be treated as an equivalent LTI system, that can be determined using the adjoint method.

control signal in the original network, it is be replaced by $\phi(-t)$ in the adjoint. $l_{eq}(t)$ is obtained by exciting the adjoint network with a current impulse $\delta(t)$ at the "output" port and observing the resulting current waveform through R, that is $v_x(t)/R$.

To find $v_x(t)$ in the adjoint, we proceed as follows. The OTA and input resistor R are replaced by their Thevenin equivalent, as seen by the SC DAC (Figure C.21(a)). $v_{th}(t)$ is a step function with magnitude $1/C$. Without the DAC, $v_x(t)$ and $v_{th}(t)$ would be equal – and $l_{eq}(t)$, that is $v_x(t)/R$, would then be the impulse response of a time-invariant loop-filter. We thus denote $(1/R)v_{th}(t)$ by $l_{ideal}(t)$.

The DAC capacitor C_d is connected to x (through R_d) during $\hat{\phi}_2$. This causes v_x to dip momentarily. The OTA then charges C_d to $v_{th}(t)$ with a time-constant $C(R_d + 1/G_{ota})$. At the end of $\hat{\phi}_2$, v_x reaches v_{th}. The same sequence of events repeats in subsequent clock cycles. The resulting v_x is shown in Figure C.21(b), that also depicts $v_{th}(t)$ for convenience. To achieve good jitter immunity, $0.5C/g_{ota} \ll T_s$, that means that the DAC capacitor is almost instantly charged to $v_{th}(t)$. The DAC current can therefore be approximated by an impulsive sequence as follows.

$$i(t) \approx f_s C \sum_{n=0}^{\infty} v_{th}((n + 0.5)T_s) \cdot \delta\left(t - (n + 0.5)T_s\right). \tag{C.29}$$

Since $l_{eq}(t) = v_x(t)/R$ and $v_x(t) = v_{th}(t) - (i(t)/G_{ota})$, we have

$$l_{eq}(t) \approx \underbrace{\frac{v_{th}(t)}{R}}_{l_{ideal}(t)} - \frac{C}{RG_{ota}} v_{th}(t) \sum_{n=0}^{\infty} \delta\left(t - (n + 0.5)T_s\right). \tag{C.30}$$

The first term on the right-hand side (RHS) of the equation above is $l_{ideal}(t)$. The second term, arising due to the SC DAC, is the product of v_{th} and a Dirac delta train with frequency
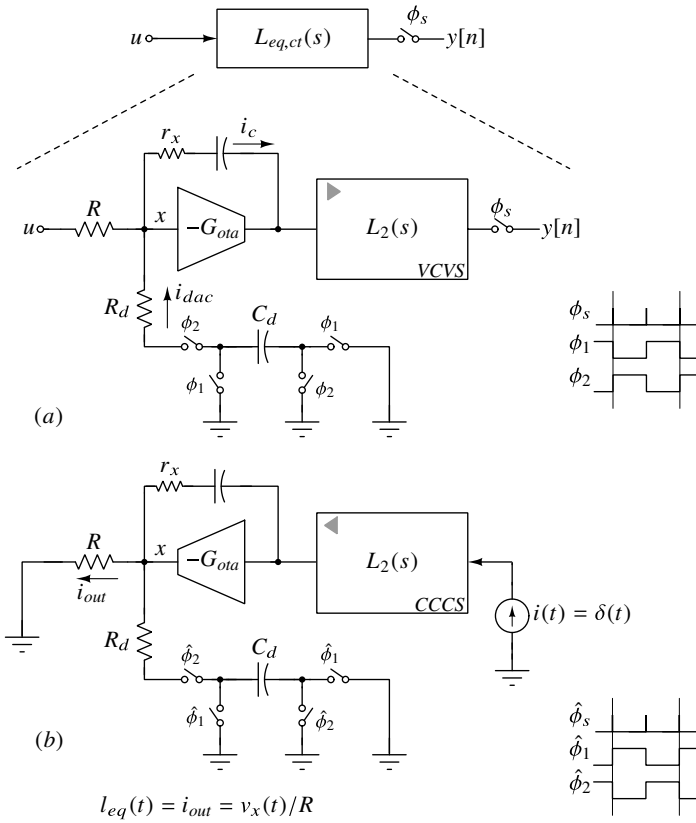
Figure C.20   (a) Loop filter of a CIFF CT$\Delta\Sigma$M with a switched-capacitor DAC. (b) Determining the impulse response of the equivalent time-invariant filter using the adjoint network.
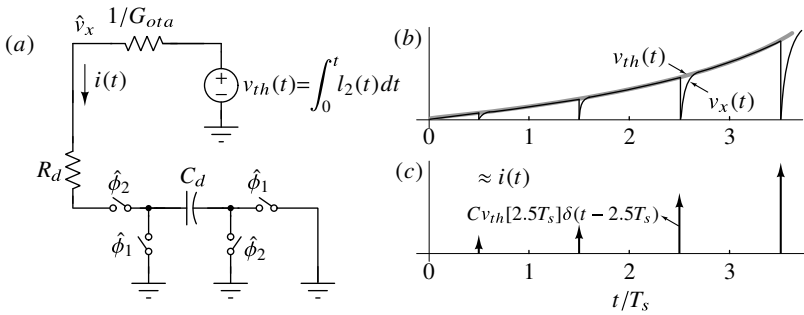
Figure C.21   (a) The OTA replaced by its Thevenin equivalent. (b) $\hat{v}_x$ and $v_{th}(t)$, and (c) impulsive approximation to $i(t)$.

f_s. This indicates that sampling of v_x occurs at this rate. This makes sense, since the virtual ground node is being sampled every clock period. The Fourier transform of $l_{eq}(t)$ is thus seen to be

$$L_{eq}(j2\pi f) \approx L_{ideal}(j2\pi f) - \underbrace{\frac{f_s C}{G_{ota}} \sum_k L_{ideal}(j2\pi (f - kf_s))}_{E(j2\pi f)}. \qquad (C.31)$$
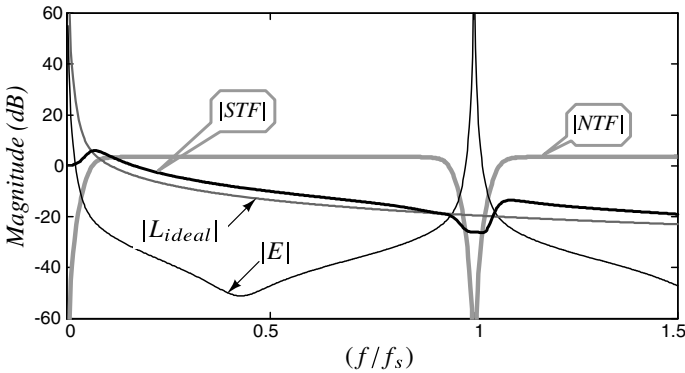
Figure C.22 Magnitude plots of L_{ideal}, $E(j2\pi f)$, NTF, and STF for a third-order CIFF CT$\Delta\Sigma$M with a switched-capacitor feedback DAC.

Figure C.22 shows representative magnitude responses of $L_{ideal}(j2\pi f)$ and $E(j2\pi f)$. E is periodic with f_s. To obtain the STF, $L_{eq}(= L_{ideal} - E)$ is multiplied by the NTF. As seen from (C.31), for frequencies of the form $(\Delta f + kf_s)$, where $\Delta f \ll f_s$, $L_{eq}(j2\pi f) \approx (f_s C/G_{ota}) L_{ideal}(j2\pi\Delta f)$. Further, $NTF(e^{j2\pi(f+\Delta f)}) = NTF(e^{j2\pi\Delta f}) \approx 1/L_{ideal}(j2\pi\Delta f)$. We thus have

$$|STF(f + \Delta f)| \approx \frac{f_s C}{G_{ota}} = \frac{1}{G_{ota}R}. \qquad (C.32)$$

Using an SC DAC has, therefore, resulted in a severe degradation of the modulator's alias rejection, as seen in Figure C.22. This is consistent with our first-order analysis in Chapter 10, that was based on "average" arguments.

References

[1] L. A. Zadeh, "Frequency analysis of variable networks," *Proceedings of the IRE*, vol. 38, no. 3, pp. 291–299, 1950.

[2] S. Pavan and R. S. Rajan, "Interreciprocity in linear periodically time-varying networks with sampled outputs," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 61, no. 9, pp. 686–690, 2014.

Index

IEEE PRESS SERIES ON MICROELECTRONIC SYSTEMS

The focus of the series is on all aspects of solid-state circuits and systems including the design, testing, and application of circuits and subsystems, as well as closely related topics in device technology and circuit theory. The series also focuses on scientific, technical and industrial applications, in addition to other activities that contribute to the moving the area of microelectronics forward.

R. Jacob Baker, *Series Editor*

<table>
<tr><td>1.</td><td>Nonvolatile Semiconductor Memory Technology: A Comprehensive Guide to Understanding and Using NVSM Devices
William D. Brown, Joe Brewer</td></tr>
<tr><td>2.</td><td>High-Performance System Design: Circuits and Logic
Vojin G Oklobdzija</td></tr>
<tr><td>3.</td><td>Low-Voltage/Low-Power Integrated Circuits and Systems: Low-Voltage Mixed-Signal Circuits
Sanchez-Sinencio</td></tr>
<tr><td>4.</td><td>Advanced Electronic Packaging, Second Edition
Richard K. Ulrich and William D. Brown</td></tr>
<tr><td>5.</td><td>DRAM Circuit Design: Fundamental and High-Speed Topics
Brent Keith, R. Jacob Baker, Brian Johnson, Feng Lin</td></tr>
<tr><td>6.</td><td>CMOS: Mixed-Signal Circuit Design, Second Edition
R. Jacob Baker</td></tr>
<tr><td>7.</td><td>Nonvolatile Memory Technologies with Emphasis on Flash:A Comprehensive Guide to Understanding and Using NVM Devices
Joseph E. Brewer, Manzur Gill</td></tr>
<tr><td>8.</td><td>Reliability Wearout Mechanisms in Advanced CMOS Technologies
Alvin W. Strong, Ernest Y. Wu, Rolf-Peter Vollertsen, Jordi Sune, Giuseppe La Rosa, Timothy D. Sullivan, Stewart Rauch</td></tr>
<tr><td>9.</td><td>CMOS: Circuit Design, Layout, and Simulation, Third Edition
R. Jacob Baker</td></tr>
<tr><td>10.</td><td>Quantum Mechanics for Electrical Engineers
Dennis M. Sullivan</td></tr>
<tr><td>11.</td><td>Nanometer Frequency Synthesis Beyond Phase-Locked Loop
Liming Xiu</td></tr>
<tr><td>12.</td><td>Electrical, Electronics, and Digital Hardware Essentials for Scientists and Engineers
Ed Lipiansky</td></tr>
<tr><td>13.</td><td>Enhanced Phase-Locked Loop Structures for Power and Energy Applications
Masoud Karimi-Ghartemani</td></tr>
</table>