Generalized Noise Shaping in Delta Sigma Modulators

by

John Bell

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Electrical and Computer Engineering)
in the University of Michigan
2020

Doctoral Committee:

Professor Michael P. Flynn, Chair
Professor John P. Hayes
Associate Professor David Wentzloff
Associate Professor Zhengya Zhang

John Bell
johnlb@umich.edu
ORDCID iD: 0000-0001-8887-675X

For my dad—who helped me discover my love of electronics and the joy of solving problems.

And my mom—who has continuously supported me, through good times and... less than good times.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

LIST OF ABBREVIATIONS

ADC Analog to Digital Converter

AGWN Additive Gaussian White Noise

BB Baseband

BP Bandpass

BW Bandwidth

CA Carrier Aggregation

CCF Controllable Canonical Form

CMOS Complementary Metal Oxide Semiconductor

CRFF Cascade of Resonators with Feed-Forward

CRFB Cascade of Resonators with Feedback

CT Continuous-Time

CT-DT Continuous-Time to Discrete-Time

DAC Digital to Analog Converter

DR Dynamic Range

DT-CT Discrete-Time to Continuous-Time

DT Discrete-Time

ELD Excess Loop Delay

FF Feed-Forward

FoM Figure of Merit

IF Intermediate Frequency

LHP Left-Half-Plane

MB Multi-Band

MZT Matched-Z Transform

NRZ Non-Return-to-Zero

NTF Noise Transfer Function

OCF Observable Canonical Form

OSR Oversample Rate

RF Radio Frequency

RZ Return-to-Zero

SAR Successive Approximation Register

SFDR Spurious-Free Dynamic Range

SNDR Signal-to-Noise and Distortion Ratio

SNR Signal-to-Noise Ratio

SQNR Signal-to-Quantization Noise

STF Signal Transfer Function

TIA Trans-Impedance Amplifier

ABSTRACT

Ever increasing bandwidth demands in modern cellular networks are becoming difficult to meet, as there is no longer room to expand within the current commercial frequency allocation. As a result, an important part of current and future cellular protocols, such as LTE-A and 5G, is Carrier Aggregation—the ability to communicate on multiple channels at once. However, when those channels lie in separate bands, they can be separated by 100's of MHz. This complicates the analog front-end design, since it is difficult for a single ADC to convert such a wide bandwidth, to say nothing of the inefficiency in converting undesired spectrum along with desired. As a result, multiple power-hungry front-ends are typically required.

To reduce the need for separate analog front-ends, we introduce a new class of $\Delta\Sigma$ modulator—the Multi-Band $\Delta\Sigma$ modulator—that uses custom-designed noise shaping to digitize multiple bands simultaneously without wasting valuable noise shaping resources on undesired portions of the spectrum. The prototype Multi-Band $\Delta\Sigma$ Modulator (MB-$\Delta\Sigma$M) is fabricated in 40nm Complementary Metal Oxide Semiconductor (CMOS) technology and demonstrates two simultaneous bands: one at baseband and one at bandpass. These two bands are separated by 500MHz, have an aggregate bandwidth of 90MHz, with up to 55dB measured SNDR. In addition to reducing the number of ADCs, this new approach promises further system-level power savings by simplifying the RF front-end. The system-level power savings from requiring fewer analog mixers, LNAs, filters, and ADC drivers can be even more than the ADC power reduction.

We further develop two theoretical results that, when taken together, greatly simplify the design of this complex, high-order modulator. The first result introduces an easy-to-use, closed-form solution for Continuous-Time (CT) to Discrete-Time (DT) loop filter conversion which fully accounts for Excess Loop Delay (ELD) and can be used with all standard ELD compensation techniques. Our solution also allows for arbitrary Digital to Analog Converter (DAC) pulse shapes and is architecture agnostic.

The second result shows that any modulator using an ELD compensation path implemented in CT has limited noise shaping capacity. We also show the types of noise shaping that are available to such systems, making it possible for the designer to reliably implement more complex noise shaping.

CHAPTER 1

Introduction

$\Delta\Sigma$ modulation was introduced as a modification to Δ modulation, a then-common communication scheme, by Inose et al. in 1962 [5]. While Δ modulation's output is based on the differentiation of the input signal—which is later integrated to reconstruct the original signal—$\Delta\Sigma$ modulation brings both the difference and the integration into the modulator by applying feedback, hence the name $\Delta\Sigma$. The result is that the modulation error is "shaped" to high frequencies. A simple analog low-pass filter can then be used to filter out the shaped noise and reconstruct the original analog signal. It would be over a decade before James Candy would connect $\Delta\Sigma$ modulation to analog-to-digital conversion by implementing the low-pass filter digitally [6].

These early modulators were Continuous-Time (CT)—their loop filters were implemented with CT integrators [5] [7] [6] [8]. While Discrete-Time (DT) modulators were popular in the 1990s[1], CT $\Delta\Sigma$ modulators have stood the test of time in part due to their tolerance of the low headrooms common in advanced Complementary Metal Oxide Semiconductor (CMOS) processes, reduced anti-alias filter requirements, and improved energy efficiency [3] [9]. Bandpass (BP) modulators directly digitize an Intermediate Frequency (IF) or Radio Frequency (RF) signal and allow digital IQ mixing, removing the troublesome analog RF or IF mixer.

CT-$\Delta\Sigma$ Analog to Digital Converter (ADC)s are a popular choice for wireless systems since their resistive input and anti-aliasing properties require significantly less filtering and power-hungry buffering to drive compared to converters that must sample their input. To reduce noise, the input sampling capacitor in DT converters must be large, requiring a fast transient response and high instantaneous current draw from the driver.

[1] By the author's estimation, only 10 out of 55 papers in the Journal of Solid State Circuits (JSSC) between 1990-1999 were CT.

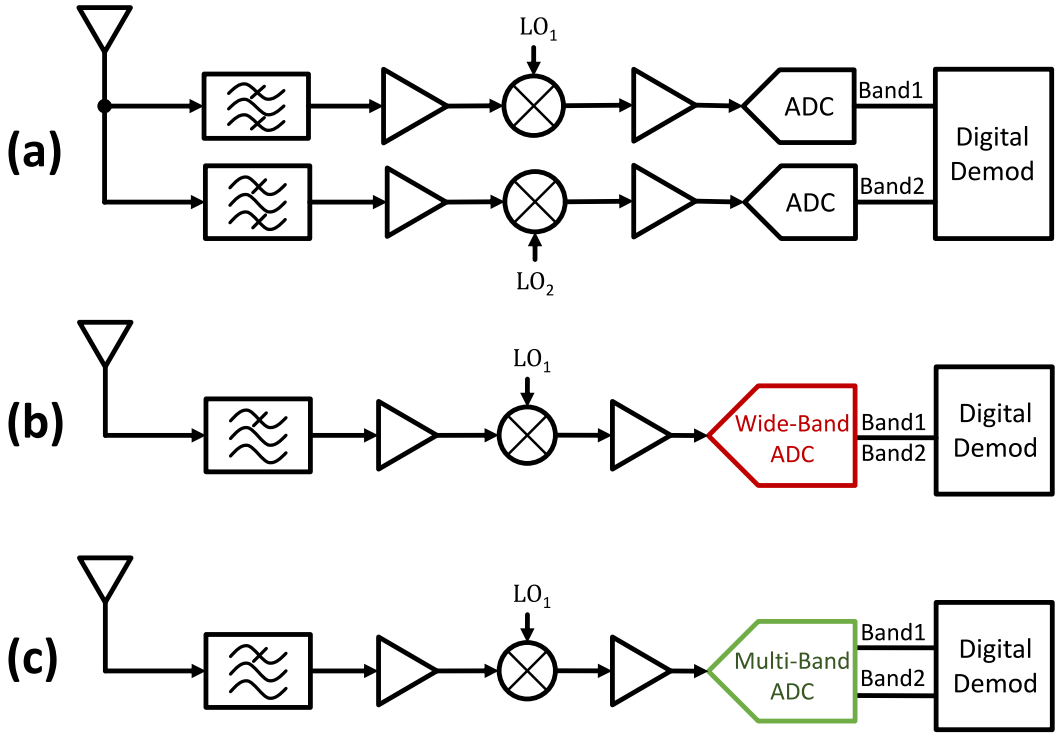


Figure 1.1: *(a) Conventional parallel receiver, (b) using power-hungry wide-band ADC, and (c) Multi-Band CT-$\Delta\Sigma$ ADC reduces ADC power and simplifies front-end.*

To keep up with increasing bandwidth demands, Carrier Aggregation (CA) has become an essential part of LTE-A and emerging 5G standards. While intra-band CA is approachable with current technology, inter-band CA is far more challenging to accomplish efficiently, due to the large band separations of 100's of MHz required. Conventional approaches to this problem use parallel receivers [10] (Fig. 1.1(a)) or digitize a wide range of spectrum encompassing all bands of interest [11] (Fig. 1.1(b)). However, both of these approaches are power-hungry and area-intensive. We introduce the Multi-Band (MB) CT-$\Delta\Sigma$ ADC to simultaneously and efficiently digitize only the bands of interest (Fig. 1.1(c)).

1.1 Single-Band Modulation

The idea of bandpass $\Delta\Sigma$ modulation was originally proposed by Richard Schreier and Martin Snelgrove in 1989 [12]. The earliest work was done in simulation, but the first physical implementation is due to Thurston, Pearce, and Hawksford in 1991 [13],

followed closely by a fully monolithic design from Jantzi, Snelgrove, and Ferguson in 1992 [14]. The implementation due to Thurston had an LC-based, CT loop filter, while the monolithic version was a switched-capacitor design. Switched capacitor modulators dominate the literature through the 90s. One notable exception is due to Tao and Khoury in 1997. Theirs was a hybrid modulator that uses a CT resonator in front, then mixes to baseband before passing through a switched-capacitor integrator and on to the quantizer. This allows for the benefits of a CT modulator (resistive input, anti-aliasing) while still doing direct-conversion at the quantizer.

Toward the end of the 90s, bandpass modulators in RF processes such as SiGe and InP began to appear [15] [16]. The 2000s also see an increased interest toward CT modulation, largely focused around direct-RF and IF-sampling wireless applications. The sub-sampled $\Delta\Sigma$ modulator was proposed in 2000 [17] and demonstrated in 2003 [18] by Hussein. In 2005, Latiri, Aboushady, and Beilleau proposed a sine-shaped Digital to Analog Converter (DAC) for use with high sample-rate ADCs [19] and 2007 brings an impressive 40GSPS, 2GHz center frequency ADC from Chalvatzis et al. By the early 2010s, most bandpass modulators in the literature seem to be CT and finally, in 2012, Chae et al. introduces a new single-amplifier biquad which uses positive feedback to boost its Q [20], reducing the power requirements for bandpass modulators (important, since they inherently require twice as many poles as a baseband modulator). This work builds on this rich history by using Chae's biquad to implement multiple simultaneous bands.

1.2 Noise Shaping

At their most basic, all $\Delta\Sigma$ modulators are simply variations on the theme of applying feedback to a quantizer. That quantizer can be as simple as a 1b flash ADC or as complex as an N-bit Successive Approximation Register (SAR) ADC, but the goal is always to reduce the error of the quantizer within some range of frequencies below the Nyquist frequency. The typical high-level model for this type of system is shown in Fig. 1.2(a). Roughly speaking, the system's feedback tends to randomize the quantizer's error [3], typically making it a rather good approximation to treat the quantizer as if it is linearly adding Gaussian white noise to its input, shown in Fig. 1.3. This is called the Additive Gaussian White Noise (AGWN) approximation, and it allows us to analyze an otherwise extremely nonlinear system using simple linear techniques.

The individual transfer functions—from the signal ($u[k]$) to the output ($y[k]$), and

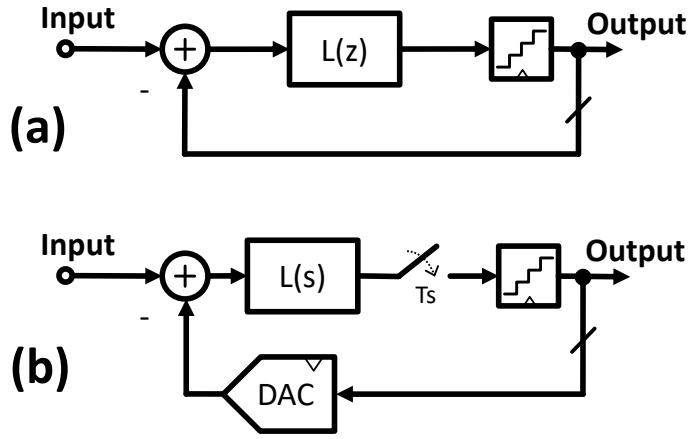


Figure 1.2: *Generic block diagram of* ***(a)*** *a Discrete-Time (DT) $\Delta\Sigma$ modulator, and* ***(b)*** *a Continuous-Time (CT) $\Delta\Sigma$ modulator.*

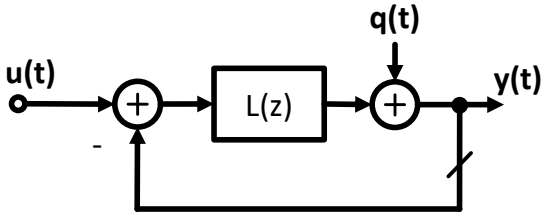


Figure 1.3: *The additive gaussian white noise model for a $\Delta\Sigma$ modulator.*

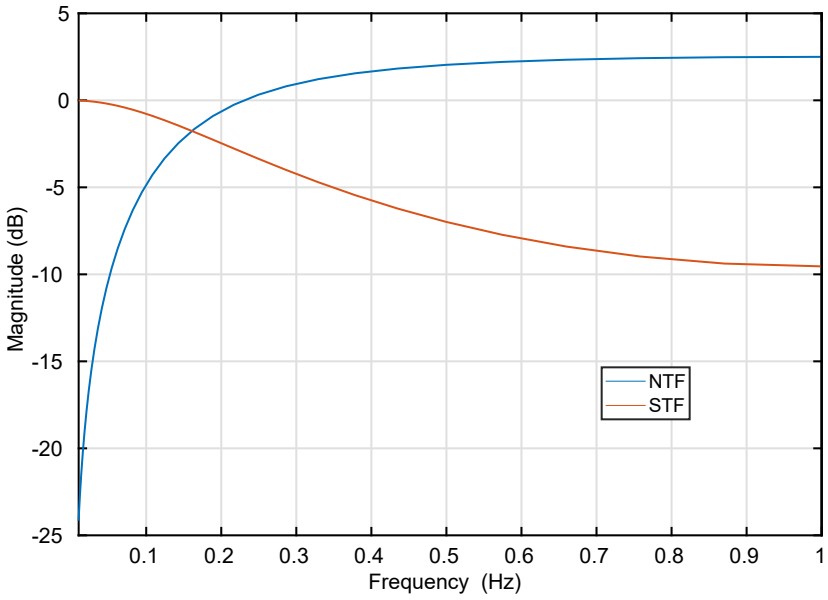


Figure 1.4: *An example of a Noise Transfer Function (NTF) and a Signal Transfer Function (STF).*

the noise ($q[k]$) to the output—are called the Signal Transfer Function (STF) and the Noise Transfer Function (NTF), respectively. For the system in Fig. 1.3, these are given, in the z-domain, by:

$$\begin{aligned} NTF(z) &= \frac{1}{1+L(z)} \\ STF(z) &= \frac{L(z)}{1+L(z)} \end{aligned} \tag{1.1}$$

When the magnitude of $L(z)$ is large, it is clear that $L(z) >> 1$ and so the denominator is: $1 + L(z) \approx L(z)$. As a result, the NTF becomes small ($1/L(z) \approx 0$) and the STF becomes 1 ($\approx L(z)/L(z) = 1$). This is the essence of noise shaping: because feedback provides for two distinct transfer functions, the signal can be passed, unaltered, while the quantization noise is filtered out. An example of these transfer functions can be seen in Fig. 1.4. In this case, $L(z)$ has a pole (infinite gain) at DC and so the error at DC is being filtered out completely.

Finally, it is common for the DT loop filter to be implemented in CT. This case is known as CT $\Delta\Sigma$ modulation and is shown in Fig. 1.2(b). In this case, a DAC drives the CT filter, which is then sampled at its output. The sampling action is typically an implicit part of the quantizer. This DAC-filter-sampler system has a DT

input and a DT output, so we can model it as a DT system, making the modulator equivalent to a fully-DT modulator. For this model, a DT filter is "equivalent" to the CT filter it is modeling if, for any given DT input, both DT outputs are identical for all time. Because the systems are linear, this is equivalent to the requirement that the DT impulse responses be identical. This is called Impulse Invariance and when this condition is met, the two modulators' responses to noise will be, by definition, identical cycle-for-cycle. The process of converting between CT and DT is discussed at length in Chapter 3.

1.3 Research Contributions

In order to improve the efficiency of inter-band carrier aggregation systems, a more efficient use of the noise spectrum can be employed: by only suppressing the noise in the bands-of-interest, a much lower clock rate can be used compared to similar wide-band $\Delta\Sigma$Ms, leading to significant overall power savings—this is a new class of modulators we call Multi-Band $\Delta\Sigma$ modulators. Having a single ADC interface also alleviates the need for complex, power-hungry front-ends. This work introduces the first example of a multi-band $\Delta\Sigma$ ADC, implementing two simultaneous bands: one at DC and one at 500MHz.

An additional goal of this work is to provide practical tools to make the design of multi-band ADCs significantly more approachable, given the level of complexity required to build such an ADC. While the theory used to create these tools may, itself, feel unapproachable, reference implementations are provided in a GitHub repository so that practicing engineers need not be familiar with the mathematical tools to use the results. See Section 6.1 for more details.

The primary tools developed in this work include: (1) a general solution to the Excess Loop Delay problem and Discrete-Time to Continuous-Time conversion which is compact & easy-to-use, and (2) a limitation of the noise shaping capacity in CT modulators due to the finite bandwidth inherent in CT-based Excess Loop Delay (ELD) compensation methods. (1) makes it possible to synthesize the CT loop filter directly from an arbitrarily complex NTF without the need for tedious, DAC- and architecture-specific hand calculations. This significantly improves design time and makes it possible to generate code to fully automate the system design process. (2) defines a critical restriction on the designer's choice of NTF, which allows the NTF to be designed precisely, with less worry that the finite-bandwidth of the CT filter will require significant changes to the final design. Taken together, they enable the

designer to use an NTF-first approach, which greatly simplifies otherwise complex loop filter calculations.

1.4 Organization

Chapter 2 gives a high-level overview of state-space systems. Next, Chapter 3 discusses the equivalence of CT and DT systems and the problem of ELD. Chapter 4 introduces an important property of CT-$\Delta\Sigma$ modulators which tends to limit the noise shaping capacity. Chapter 5 discusses a prototype multi-band $\Delta\Sigma$ modulator and presents measurement results. The prototype is designed with the theoretical results in Chapter 3 and Chapter 4 in mind. We finally conclude with Chapter 6.

CHAPTER 2

An Overview of State Space Systems

Beginning nearly a century ago with Harold Black's 1934 paper in Bell Labs' technical journal [21], the field of classical control theory has grown up primarily around the description of systems by the way they alter the frequency content of a signal; with Fourier, Laplace, and Z-transforms as its primary tools. This frequency-based approach has been immensely successful, giving electrical engineers alone such fundamental concepts as the Bode plot, the Miller Effect, and active filtering, just to name a few. Then, beginning in the 1960s [22], modern control theory began to introduce an alternate view of systems: by focusing on the time-domain, incredibly flexible and powerful tools can be developed. Modern and classical approaches have since complemented each other, greatly expanding on our ability to understand and design complex dynamical systems.

We do not claim that state-space methods should somehow replace classical methods, however. Both are merely tools that can be useful in complementary situations. In particular, we contend that the Impulse Invariance problem—a critical part of CT-$\Delta\Sigma$ design that is introduced in Chapter 3—is best formulated in state-space since it is, by definition, a time-domain problem. For this reason, this chapter aims to provide a high-level, intuitive understanding of the concepts from modern control theory used in this work. For a more in-depth introduction to state-space control systems, see [23].

Section 2.1 introduces the abstract fundamentals of linear algebra alongside the more practical application of these fundamentals in modeling real systems. Using this foundation, Section 2.2 builds the tools this work employs in Chapter 3 and Chapter 4.

2.1 Modern Systems Theory

In state-space, all systems—whether or not they are linear or time-invariant—are modeled as a set of coupled differential equations, which can be written as:

$$\begin{aligned}
\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \vdots \end{bmatrix} &= \begin{bmatrix} f_1(t, x_1(t), x_2(t), \ldots, u_1(t), u_2(t), \ldots) \\ f_2(t, x_1(t), x_2(t), \ldots, u_1(t), u_2(t), \ldots) \\ \vdots \end{bmatrix} \\
\begin{bmatrix} y_1(t) \\ y_2(t) \\ \vdots \end{bmatrix} &= \begin{bmatrix} h_1(t, x_1(t), x_2(t), \ldots, u_1(t), u_2(t), \ldots) \\ h_2(t, x_1(t), x_2(t), \ldots, u_1(t), u_2(t), \ldots) \\ \vdots \end{bmatrix}
\end{aligned} \tag{2.1}$$

where each $x_i(t)$ is an internal signal acting as memory for the system (known as its "state"), u_j is each input, and y_k is each output. The total number of state variables is the system's order. For linear, potentially time-varying systems, these equations take the form:

$$\begin{aligned}
\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \vdots \end{bmatrix} &= \begin{bmatrix} a_{11}(t)x_1(t) + a_{12}(t)x_2(t) + \ldots + b_{11}(t)u_1(t) + b_{12}u_2(t) + \ldots \\ a_{21}(t)x_1(t) + a_{22}(t)x_2(t) + \ldots + b_{21}(t)u_1(t) + b_{22}u_2(t) + \ldots \\ \vdots \end{bmatrix} \\
\begin{bmatrix} y_1(t) \\ y_2(t) \\ \vdots \end{bmatrix} &= \begin{bmatrix} c_{11}(t)x_1(t) + c_{12}(t)x_2(t) + \ldots + d_{11}(t)u_1(t) + d_{12}u_2(t) + \ldots \\ c_{21}(t)x_1(t) + c_{22}(t)x_2(t) + \ldots + d_{21}(t)u_1(t) + d_{22}u_2(t) + \ldots \\ \vdots \end{bmatrix}
\end{aligned} \tag{2.2}$$

Where each of the coefficients a_{ij} b_{kl}, c_{nm}, and d_{pq} are scalar functions of time. This is extremely cumbersome on its own, but by rewriting Eq. (2.2) as a matrix equation, we can apply very powerful tools from linear algebra:

$$\begin{bmatrix} \dot{x_1}(t) \\ \dot{x_2}(t) \\ \vdots \end{bmatrix} = \begin{bmatrix} a_{11}(t) & a_{12}(t) & \ldots \\ a_{21}(t) & a_{22}(t) & \ldots \\ \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \end{bmatrix} + \begin{bmatrix} b_{11}(t) & b_{11}(t) & \ldots \\ b_{11}(t) & b_{11}(t) & \ldots \\ \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} u_1(t) \\ u_2(t) \\ \vdots \end{bmatrix}$$
$$\dot{x}(t) = A(t)x(t) + B(t)u(t) \tag{2.3}$$

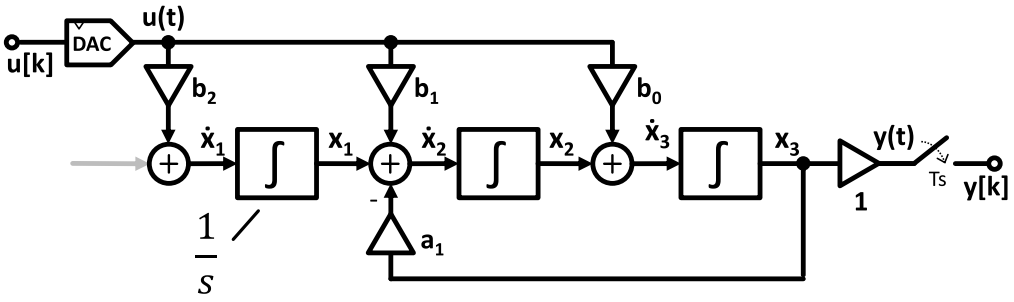


Figure 2.1: ***(top)*** *The block diagram of a CRFB modulator.* ***(bot)*** *The block diagram of the modulator's loop filter by itself.*

$$\begin{bmatrix} y_1(t) \\ y_2(t) \\ \vdots \end{bmatrix} = \begin{bmatrix} c_{11}(t) & c_{12}(t) & \dots \\ c_{21}(t) & c_{22}(t) & \dots \\ \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \end{bmatrix} + \begin{bmatrix} d_{11}(t) & d_{11}(t) & \dots \\ d_{11}(t) & d_{11}(t) & \dots \\ \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} u_1(t) \\ u_2(t) \\ \vdots \end{bmatrix}$$
$$y(t) = C(t)x(t) + D(t)u(t) \tag{2.4}$$

In this work, we only consider the time-invariant case and, for simplicity and readability, we omit the time variable for the input, state, and output signals. We are left with a very simple representation of a complex set of equations:

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du \end{aligned} \tag{2.5}$$

To illustrate these concepts, throughout this chapter we build on the filter example shown in figure Fig. 2.1, which shows a block diagram for a standard Cascade of Resonators with Feedback (CRFB) modulator [3]. The top of Fig. 2.1 is the classic depiction of the full modulator, while the bottom is a view of the loop filter on its own.

Considering the loop filter by itself, we can write down the equations at each node as:

$$\begin{aligned} \dot{x_1} &= b_0 u \\ \dot{x_2} &= x_1 - a_1 x_3 + b_1 u \\ \dot{x_3} &= x_2 + b_2 u \\ y &= x_3 \end{aligned} \tag{2.6}$$

Thus, writing this in matrix form:

$$\begin{aligned}\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\end{bmatrix} &= \begin{bmatrix}0 & 0 & 0\\ 1 & 0 & -a_1\\ 0 & 1 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix} + \begin{bmatrix}b_0\\ b_1\\ b_2\end{bmatrix} u\\ y &= \begin{bmatrix}0 & 0 & 1\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix}\end{aligned} \tag{2.7}$$

So that:

$$\begin{aligned}A &= \begin{bmatrix}0 & 0 & 0\\ 1 & 0 & -a_1\\ 0 & 1 & 0\end{bmatrix}\\ B &= \begin{bmatrix}b_0\\ b_1\\ b_2\end{bmatrix}\\ C &= \begin{bmatrix}0 & 0 & 1\end{bmatrix}\\ D &= 0\end{aligned} \tag{2.8}$$

where b_i and a_i are the i$^{\text{th}}$ numerator and denominator coefficients, respectively.

2.1.1 Making it Concrete

An abstract, compact way of representing something is not terribly useful without the ability to meaningfully interpret it. In Fig. 2.2, we show the general block-diagram equivalent of Eq. (2.5).

From the block-diagram view, a few things become clear. (1) the most basic unit of a system is an integrator; (2) the A matrix defines all of the feedback paths in the system; (3) the B, C, and D matrices define all feed-forward paths, going from the input to the state and going from the state to the output.

We can think about the classical transfer function of this system by replacing each integrator block with 1/s. This system would be made entirely of poles at the origin, but its feedback paths move these poles to different locations, depending on the coefficients in A. So, as a consequence of (2), the poles of this system are described entirely by the A matrix. The role of the B, C, and D matrices is discussed in more detail in Section 2.2.2.

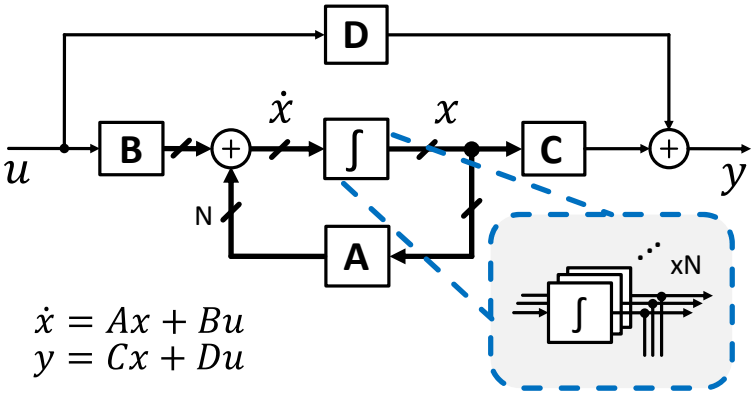


Figure 2.2: *A block diagram of the state-space representation of a system.*

Finally, we can take the Laplace Transform of the state equation as well, which becomes $sx = Ax + Bu$. Solving for x and substituting the result for x in the output equation, we find that:

$$\begin{aligned} Y(s) &= C\left[(sI - A)^{-1}BU(s)\right] + DU(s) \\ \frac{Y(s)}{U(s)} &= C(sI - A)^{-1}B + D \end{aligned} \tag{2.9}$$

2.1.2 Discrete-Time State-Space Modeling

Systems in DT are modeled in state-space in the same way as CT systems. The primary difference is the use of a delay—$x[k+1]$, with k as the time-variable—instead of the derivative, $\dot{x}$(t). This difference gives "difference equations" as the fundamental descriptor, rather than differential equations. All the analysis developed in this chapter is also valid for DT systems.

The system equations in this case are:

$$\begin{aligned} x[k+1] &= Ax[k] + Bu[k] \\ y &= Cx[k] + Du[k] \end{aligned} \tag{2.10}$$

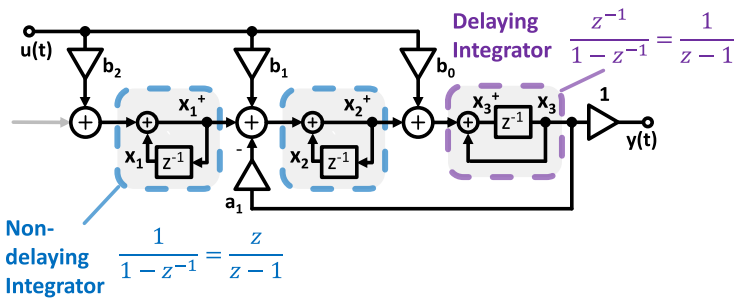


Figure 2.3: *Block diagram of the DT CRFB loop filter.*

Returning to the CRFB structure, the CT integrators are replaced with DT integrators, as shown in Fig. 2.3. Notice that, because the primary building block in DT systems are unit delays, rather than integrators, the equations for each state are significantly different from Eq. (2.7):

$$\begin{aligned} \begin{bmatrix} x_1^+ \\ x_2^+ \\ x_3^+ \end{bmatrix} &= \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & -a_1 \\ 1 & 1 & (1-a_1) \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} b_0 \\ b_1 \\ b_2 \end{bmatrix} u \\ y &= \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \end{aligned} \tag{2.11}$$

For simplicity, we use x^+ in place of $x[k+1]$ (as opposed to $x = x[k]$).

It is important to note that these two types of systems behave quite differently and cannot be easily compared in general, as Chapter 3 explores more deeply.

Finally, a note on Z-domain analysis. When looking at the Z-domain transfer function of a DT system, it is very common to write it using z^{-1} as the primary variable, as in: $a_N + \cdots + a_1 z^{-(N-1)} + a_0 z^{-N}$. This is extremely useful for engineers, since it directly relates to the filter's implementation in terms of unit delays—i.e., z^{-N} is a series of N delays. However, in order to analyze a Z-domain transfer function in terms of poles and zeros, the transfer function must be written in terms of the variable z, as in: $a_N z^N + \cdots + a_1 z + a_0$. We use this form throughout this work, since it is more useful for the type of analysis required here. However, moving between the two representations is simply a matter of factoring out the z term, as appropriate. For example, these two forms are equivalent: $a_2 + a_1 z^{-1} + a_0 z^{-2} = (a_2 z^2 + a_1 z^1 + a_0)/z^2$, which means this system has two poles at the origin and two zeros which depend on the coefficients a_0, a_1, and a_2.

2.2 Tools from Linear Algebra

2.2.1 Signals

One of the most fundamental concepts in linear algebra—one on which the entire discipline is built—is the concept of Linear Vector Spaces. A Linear Vector Space is a set of vectors—each of which consists of some set of numbers from a field (e.g., the real numbers, denoted $\mathbb{R}$, or the complex numbers, denoted $\mathbb{C}$)—in combination with the vector addition and scalar multiplication operators (vectors add element-wise and a scalar can be multiplied by each element in the vector). Think of a vector space as defining all possible vectors and the operations we can do on those vectors.

While these ideas are typically defined in a very abstract way, this chapter focuses more directly on their application in linear systems. In this context, vectors typically represent the values of a set of signals together at a particular time; most often, the overall state of the system itself. If we think of this vector as a point in a coordinate system, we can represent the values of all the signals in very complex systems with a single object: this same point in space.

Now consider unit-length vectors on each axis of a typical 2D coordinate system, which can be thought of as the vectors $\begin{bmatrix}1 & 0\end{bmatrix}^T$ and $\begin{bmatrix}0 & 1\end{bmatrix}^T$. We can construct a vector representing this point by scaling each of these unit vectors and adding them tip-to-tail. The unit vectors that lie on the euclidean axes are often written as $\hat{i}$ and $\hat{j}$. We can write this, for example, as:

$$\begin{aligned}\begin{bmatrix}2\\3\end{bmatrix} &= 2\hat{i} + 3\hat{j} \\ &= 2\begin{bmatrix}1\\0\end{bmatrix} + 3\begin{bmatrix}0\\1\end{bmatrix}\end{aligned} \tag{2.12}$$

This process, and much of linear algebra in general, is often best thought of geometrically, so the top of Fig. 2.4 illustrates this process.

This construction is, of course, the same operation we do every day when using coordinates in the Euclidean plane, with the scale factors equal to our usual coordinates. However, by starting with vectors, the axes are no longer fixed—we could just as easily choose any other set of initial vectors. The vector $\begin{bmatrix}2 & 3\end{bmatrix}^T$ describes a point in space in terms of the "basis" vectors $\hat{i}$ and $\hat{j}$, but there is no preferred starting point here. We happen to be using $\hat{i}$ and $\hat{j}$, but we could just as easily start with completely different basis vectors, say e_1 and e_2.

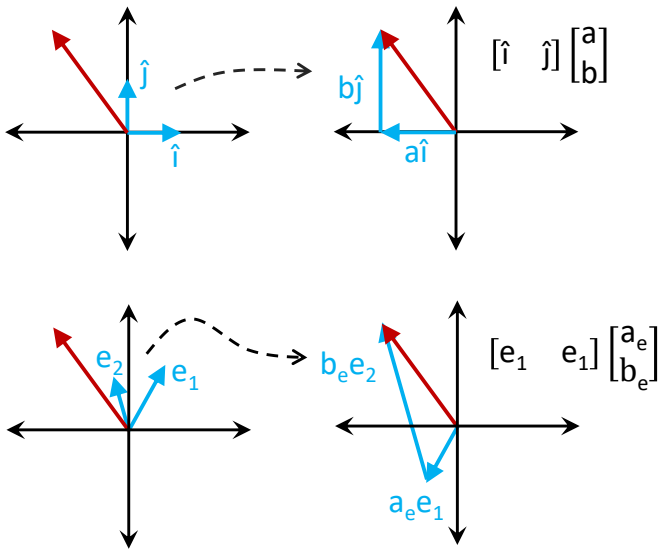


Figure 2.4: *A visual depiction of constructing vectors using a particular basis.* ***(top)****, the basis vectors* $(\hat{i}, \hat{j})$ *are multiplied by the coefficients* a *and* b *and added tip-to-tail, the result of which is the desired vector.* ***(bot)****, the same process is applied to a more arbitrary choice of basis vectors,* (e_1, e_2)*.*

Since the basis vectors point in different directions this time, we'll need to scale them differently to reproduce the same point. After some trial and error, we might find that:

$$\begin{aligned}\begin{bmatrix} 2 \\ 3 \end{bmatrix} &= 1 \begin{bmatrix} 4 \\ 2 \end{bmatrix} + -1 \begin{bmatrix} 2 \\ -1 \end{bmatrix} \\ &= 1e_1 + -1e_2\end{aligned} \tag{2.13}$$

Intuitively, we can think of writing the output vector—which is by necessity in some basis, in this case $(\hat{i}, \hat{j})$—in terms of some other basis, in this case (e_1, e_2). This is known as a change of basis.

For convenience, we can concatenate these basis vectors and write the operation as a matrix equation:

$$\begin{aligned}\begin{bmatrix} 2 \\ 3 \end{bmatrix} &= \begin{bmatrix} e_1 & e_2 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} \\ &= \begin{bmatrix} 4 & 2 \\ 2 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix}\end{aligned} \tag{2.14}$$

One way of viewing vector-matrix multiplication, then, is as a linear operation that changes our viewpoint of the same underlying system state (i.e., the same actual point in space) from one representation, or basis, to another. Note that the matrix for $(\hat{i}, \hat{j})$ is special, in that it maps any vector to itself. When the vectors are ordered so that all the 1's are on the diagonal, the resulting matrix is called the identity matrix and is denoted with an I. This matrix behaves very similarly to the number 1 in 1-dimensional algebra.

At this point, it is important to emphasize that, in general, a matrix, A and a vector, x, cannot be swapped when multiplying them; multiplication, in this case, is "non-commutative". After all, the operation xA doesn't even make sense, as we have defined it so far. Furthermore, multiplying two matrices is also non-commutative. That is, although the multiplications A_1A_2 and A_2A_1 can be calculated, where both A's have the same number of rows as columns (i.e., they are both "square" matrices), the result will, in general, be different. However, multiplication is distributive, as in: $(A_1 + A_2)x = A_1x + A_2x$.

With that said, the system of constructing arbitrary vectors based on known basis vectors breaks down in some cases: if the two basis vectors point in the same direction, we cannot represent any matrix other than a scalar multiple of the two. A

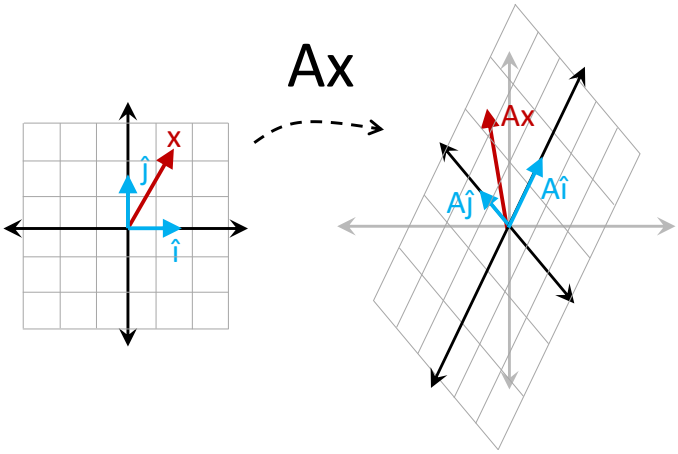


Figure 2.5: *An example of the linear operation of a matrix, A. It stretches and squishes both axes as well as rotating them both differently, distorting the original space. Any vector that undergoes this linear operation appears unchanged with respect to the new, squished coordinates, but is transformed with respect to the old coordinates.*

way of capturing this idea is called "rank." Formally, a matrix's rank is a count of the number of "linearly independent" basis vectors it contains. Two vectors are linearly independent if they cannot be scaled (with not all scale factors equal to zero) and added to become the zero vector, as in: $\alpha_1 e_1 + \alpha_2 e_2 \cdots \neq 0$ for all possible values of the coefficients α_i. For example, the vectors $\begin{bmatrix}2 & 4\end{bmatrix}^T$ and $\begin{bmatrix}5 & 10\end{bmatrix}^T$ are linearly dependent (with a scale factor of 2.5), whereas $\begin{bmatrix}2 & 4\end{bmatrix}^T$ and $\begin{bmatrix}4 & 3\end{bmatrix}^T$ are linearly independent.

When all of the vectors constituting a matrix are linearly independent, that is: when the rank is equal to the number of columns, we say it is "full rank" (or sometimes "full column rank" if there are more rows than there are columns). When the matrix's rank is less than the number of columns, we say that it "loses rank".

Finally, there is another important way to analyze matrix-vector multiplication: as a stretching and rotating of the entire vector space, leading to stretching and rotating the vectors inside it [24]. Fig. 2.5 depicts this process. Notice that, because the transform is linear, all parallel lines remain parallel. Though it is a central concept in linear algebra, it is not a viewpoint that is directly featured much in this work, so this chapter does not spend as much time on it. It is helpful to have this image, though, when understanding some of the concepts in linear algebra.

Solving y=Ax in General

It is often the case that, with the equation $y = Ax$, A and y are known and we would like to know X. For example, in Chapter 3, the equations defining the connection between CT and DT are naturally expressed as going from CT to DT with the equation: $B_d = \Gamma B_c$, where B_c and B_d are vectors and Γ is a non-square matrix, in general. By virtue of having designed the DT system, B_d & Γ are known and, in order to find the equivalent CT filter, we must find B_c.

Solving this equation requires a matrix inverse, which is defined as a matrix, A^{-1}, which has the property: $A^{-1}A = I$ (or $AA^{-1} = I$). This property is analogous to the scalar version, $a/a = 1$ for any scalar $a \neq 0$. Such a matrix only exists when: (1) there are an equal number of columns as rows (it is square), and (2) it is full rank.

When the basis vectors of a matrix (i.e., its "bases") cannot describe the entire vector space (say, all possible vectors in 3D space), it is known as a "subspace." A subspace is the collection of all vectors the basis can fully describe. Consider 3 basis vectors in 3D space that all lie on the same plane, but do not all lie on the same line. In this case, the rank is 2 and subspace would be the plane they are on.

Intuitively, the reason a matrix which loses rank cannot be inverted is because, in this case, some of the information about the original vector is lost after the transformation. If two basis vectors point in the same direction, it is impossible to represent a vector that doesn't also point in this direction. The closest we can get is to "project" the vector onto the bases we do have, that is: extend a line from the tip of the desired vector and intersect the subspace at a right angle. The vector defined by that intersection is the component of the desired vector that can be transformed. The subspace of the desired vector—after it is transformed—is called the range of the matrix, A. After the transformation, any information in the direction of the projection, which is orthogonal to the original subspace, becomes 0. In particular, an orthogonal subspace that describes this new orthogonal vector is called the "null space" of A.

Geometrically, it is clear that the original subspace and the null space, together, fully describe the vector space that contains the original vector. Because the transformation is linear (i.e., because of superposition), it must also be the case that what the transformation does to the components of a vector that lie in each of these subspaces fully describes the transformation itself. Algebraically, $y = Ax$ being rank-deficient means that, for x_R on the original subspace of A and x_N on the null-space of A, $y = A(x_R + x_N) = Ax_R + Ax_N = Ax_R$. So, trying to recover the component x_N from y is impossible.

When there are fewer equations than there are variables (that is, when A is wider

than it is tall), there exists infinitely many solutions for x. The previous discussion explains why: because the matrix is wide, it cannot have full column rank and, therefore, there must exist some part of the vector space that is lost in the transformation. We can, however, recover x_R, the component that does get transformed. Matrices that perform such transformations are known as "pseudo-inverses." The solution that gives $x_N = 0$ (i.e., returns a vector with no components in the null-space) is given by the Moore-Penrose pseudo-inverse and, if A has full row-rank, can be written as the "right pseudo-inverse": $x = A^T(AA^T)^{-1}y$. Note, however, that there are times when it is useful to include some extra component from the null-space, such as when there are additional restrictions in the problem statement. This sort of solution is required, in Chapter 3, to force the use of a particular DAC architecture.

When there are more equations than variables—that is, when A is taller than it is wide—there may not be any solution that solves all equations at once. The situation can be thought of as having more samples of a system's output than there are variables. If, for example, there is noise in the data or the linear model is slightly different from reality, the result cannot perfectly fit the data. It is somewhat more complicated than the wide case but, roughly speaking, the problem is that y may have components in the null-space of A's inverse, which makes it impossible for there to exist an x such that $Ax = y$ (because some components of y go to 0). A solution is to find the closest y vector that gives an exact solution and choose the value for x which gives that approximated solution. Again, the pseudo-inverse that chooses a y with no components on the null-space (i.e., the minimum square solution) is the Moore-Penrose pseudo-inverse and, if A has full column-rank, can be written as the "left pseudo-inverse": $x = (A^TA)^{-1}A^Ty$. Pseudo-inverses, in general, are a very complex topic, and more details can be found in [25].

Finally, for a more detailed explanation of the broader theory behind the important ideas of ranges and null-spaces, see [26].

2.2.2 Systems

With some basic tools in place, we now return to the discussion at the start of this section. The fact that we can represent a point in space in different ways, depending on our choice of basis, means that, as the system's state moves through its vector space, it too can be represented with different basis choices. Start with the state's trajectory through space represented as, $x(t)$ (the time variable included here for emphasis), which is in some arbitrary basis. If we know its relationship to another

basis, (call its state vector $\bar{x}$), we can change the basis by replacing x with $x = V\bar{x}$ for some V defined by relating the old and new bases, as in Section 2.2.1. Applying this transformation to the state equations, we see that:

$$\begin{aligned} V\dot{\bar{x}} &= AV\bar{x} + Bu \\ \dot{\bar{x}} &= V^{-1}AV\bar{x} + V^{-1}Bu \\ \dot{\bar{x}} &= \bar{A}\bar{x} + \bar{B}u \end{aligned} \tag{2.15}$$

The A and B matrices have become new matrices, $\bar{A}$ and $\bar{B}$, but the underlying system has not changed; we are only looking at it from a different point of view. A and $\bar{A}$ can be thought of as different implementations of the same filter—they are two different architectures, both implementing the same transfer function. In fact, substituting these new matrices into the state-space transfer function (Eq. (2.9)), and factoring out V, shows that the new transfer function is identical to the old transfer function (note: the 3rd step makes use of a property of inverses that says $(XYZ)^{-1} = Z^{-1}Y^{-1}X^{-1}$):

$$\begin{aligned} \bar{C}(sI - \bar{A})^{-1}\bar{B} &= CV(sI - V^{-1}AV)^{-1}V^{-1}B \\ &= CV\left(V^{-1}(sVIV^{-1} - A)V\right)^{-1}V^{-1}B \\ &= CVV^{-1}(sI - A)^{-1}VV^{-1}B \\ &= C(sI - A)^{-1}B \end{aligned} \tag{2.16}$$

To build on the CRFB example in Fig. 2.1, this same filter could instead be built using the Cascade of Resonators with Feed-Forward (CRFF) architecture instead [3]. This is essentially the same system written in a different basis. The CRFF architecture is shown in Fig. 2.6 and has the following state equations:

$$\begin{aligned} \begin{bmatrix} \dot{\bar{x}}_1 \\ \dot{\bar{x}}_2 \\ \dot{\bar{x}}_3 \end{bmatrix} &= \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & -a_1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \bar{x}_1 \\ \bar{x}_2 \\ \bar{x}_3 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} u \\ y &= \begin{bmatrix} b_2 & b_1 & (b_0 - b_2a_1) \end{bmatrix} \begin{bmatrix} \bar{x}_1 \\ \bar{x}_2 \\ \bar{x}_3 \end{bmatrix} \end{aligned} \tag{2.17}$$

where b_i and a_i are the i$^{\text{th}}$ numerator and denominator coefficients, respectively. Note that, although the B matrix in the CRFB case consists of only the transfer function's numerator coefficients, this is not true for CRFB. It can be shown that

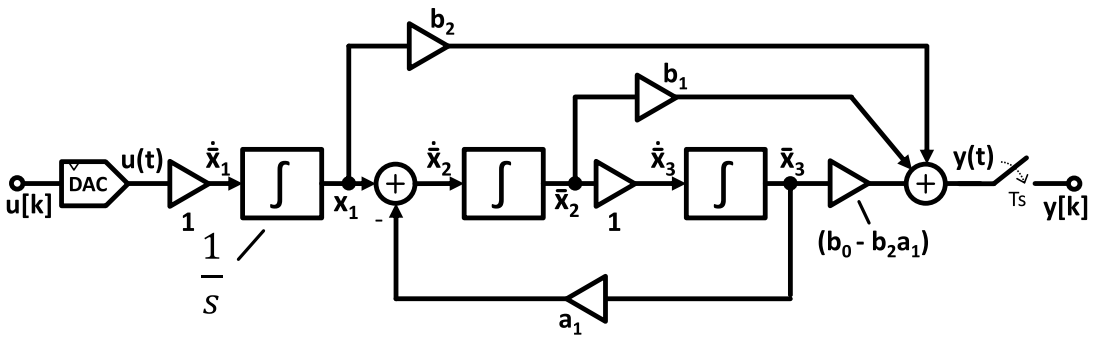


Figure 2.6: *A block diagram for the CRFF loop filter. This architecture has the same transfer function as the CRFB architecture in Fig. 2.1 and, therefore, is a different representation of the same system.*

these two systems have the same poles and zeros, and so there must exist some matrix V such that $x = V\bar{x}$, or:

$$\begin{aligned} \bar{A} &= V^{-1}AV \\ \bar{B} &= V^{-1}B \\ \bar{C} &= CV \\ \bar{D} &= D \end{aligned} \tag{2.18}$$

A method for finding this matrix is introduced at the end of this chapter.

The transformation $\bar{A} = V^{-1}AV$ is known as a similarity transform and is a powerful tool. Reading the function of each matrix from right to left, it first transforms a vector into a different basis, then applies some linear operation, A, in the new basis, and finally transforms back to the original basis. This process often makes it easier to apply a linear operation (i.e., $\bar{A}$) by choosing an intermediate basis in which the operation is easy to calculate.

Eigenvalues and System Poles

Some choices for basis vectors have unique properties. One such choice is called the "eigenbasis." For many linear transformations, there exist particular lines (through the origin) for which any vector starting on that line remains on that line after the transformation. An "eigenvector" is a vector that sits on this line. Its associated "eigenvalue" is the amount by which the transformation scales that vector. Algebraically, this is written as $Av = \lambda v$, for some scalar λ, NxN matrix A, and

N-dimensional vector v. Rearranging this equation, it can be said that:

$$(\lambda I - A)v = 0 \tag{2.19}$$

where I is the NxN identity matrix. The solution to this equation is to set the determinant of the matrix on the left-hand side to zero: $det(\lambda I - A) = 0$. This solution is called the characteristic equation, and an example is as follows:

$$\begin{aligned} det\left(\lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} 2 & -3 \\ 3 & 2 \end{bmatrix}\right) &= det\left(\begin{bmatrix} (\lambda - 2) & -3 \\ 3 & (\lambda - 2) \end{bmatrix}\right) \\ &= (\lambda - 2)(\lambda - 2) - (-3 \cdot 3) \\ &= \lambda^2 - 4\lambda + 13 \end{aligned} \tag{2.20}$$

Setting this equal to zero, we find that the eigenvalues are at $2 \pm 3j$.

In general, eigenvalues and eigenvectors can be complex, which generally correspond to vector rotations. There always exists N eigenvalues, where N is the dimension of the matrix, though there does not always exist N unique eigenvectors. A simple example of this is the identity matrix: its eigenvalues are all at 1, but none of the vectors in the vector space move at all, so the eigenvectors are effectively every possible vector. Another, less trivial, example is when there are multiple, repeated, eigenvalues. In cases like these, a small modification is possible to make a matrix block-diagonalizable. Repeated eigenvalues should be placed together with 1's on the off-diagonal between them, as follows:

$$\begin{bmatrix} 3 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 0 & 2 \end{bmatrix}$$

In this case, the eigenvalues are 3, 2, 2, and 2. The repeated eigenvalues are in a 3x3 block in the bottom-right corner. This form is called the Jordan Canonical Form (or Jordan Form) and is always diagonalizable.

As it turns out, all square matrices have a complete set of eigenvectors (or the analogous vectors in Jordan Form) which, together, form a matrix which can be used as a similarity transform. When this transform is applied to the matrix, the resulting matrix is diagonal with its eigenvalues as each of the entries on the diagonal.

Now, looking back on the characteristic equation, $det(\lambda I - A) = 0$ (Eq. (2.19)), it may seem reminiscent of the $(sI - A)$ term in Eq. (2.9) and this is not a coincidence.

In fact, writing the system in its diagonalized form (with Λ being the diagonalized form of A) and calculating the transfer function, we find that:

$$\begin{aligned} C(sI-\Lambda)B &= C\begin{bmatrix} (s-\lambda_1) & 0 & 0 & 0 \\ 0 & (s-\lambda_2) & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & (s-\lambda_N) \end{bmatrix}^{-1} B \\ &= C\begin{bmatrix} (s-\lambda_1)^{-1} & 0 & 0 & 0 \\ 0 & (s-\lambda_2)^{-1} & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & (s-\lambda_N)^{-1} \end{bmatrix} B \\ &= \frac{b_1c_1}{s-\lambda_1} + \frac{b_2c_2}{s-\lambda_2} + \cdots + \frac{b_Nc_N}{s-\lambda_N} \end{aligned} \tag{2.21}$$

In other words, the transfer function equation for a diagonalized system directly gives the partial fraction expansion. Looking at the system from this point of view, it is clear that the poles of a system are equal to the eigenvalues of its A matrix, and that the B and C matrices define the numerator of the transfer function (and so, its zeros). Furthermore, distributing the partial fraction expansion shows that, using only B and C matrices, there can only be N-1 zeros (since, for any term, the products in the numerator are always missing the associated pole from the denominator):

$$\begin{aligned} C(sI-\Lambda)B &= \frac{b_1c_1}{s-\lambda_1} + \frac{b_2c_2}{s-\lambda_2} + \cdots + \frac{b_Nc_N}{s-\lambda_N} \\ &= \frac{b_1c_1\prod_{i\neq 1}(s-\lambda_i) + b_2c_2\prod_{i\neq 2}(s-\lambda_i) + \cdots + b_Nc_N\prod_{i\neq N}(s-\lambda_i)}{(s-\lambda_1)(s-\lambda_2)\ldots(s-\lambda_N)} \end{aligned} \tag{2.22}$$

Thus, in order for the system's transfer function, $Y(s)/U(s)$, to have an equal number of poles and zeros, D must be nonzero. In that case, the full-order denominator enters the numerator (multiplied by D) and provides the extra high-order term. This makes some sense, since a system with an equal number of poles and zeros effectively has infinite bandwidth (at $s = \infty$, the numerator and denominator terms cancel), which would require a direct path from the input to the output.

The last observation we make on the topic is that a matrix is singular if and only if it has one or more eigenvalues at zero. After all, if we try to invert such a system in diagonalized form, we would have to divide by zero which, of course, is

undefined. In DT, a pole at the origin corresponds to a unit delay, z^{-1}, whereas in CT it corresponds to an integrator, $1/s$. This fact puts essential limitations on the process of equating CT and DT systems, discussed in Chapter 3.

Controllability and Observability

In addition to the eigenbasis, two other useful bases are called the Controllable Canonical Form (CCF) and Observable Canonical Form (OCF). They are based on the ideas of controllability and observability, which happens to construct a view of the system that is based entirely on the transfer function's numerator and denominator coefficients. Because the transfer function coefficients are directly available in the A, B, and C matrices, these bases can be instrumental in drawing connections between Laplace-domain analysis and state-space analysis.

At its core, observability formalizes and answers the question: can we determine the initial conditions of a system, x_0, given the input, u, and the output, y for all time $t > 0$? If this is possible, then it is also possible to know the entire evolution of the internal state of the system by working backward from the current time to a set of initial times that cover the time window in question. Restating the original question, then, observability asks: can we see the internal state of the system given only the external inputs and outputs?

Here, we study this question in DT, since it is significantly simpler and more instructive. However, the same results can be shown in CT as well. To begin, we consider the evolution of the DT system's output, with respect to its state (we can ignore the input since, if it is known, it can be moved to the left side of the equation and the form of the problem does not change):

$$\begin{aligned} y_1 &= Cx_1 \\ y_2 &= Cx_2 \\ &\vdots \\ y_N &= Cx_N \end{aligned} \tag{2.23}$$

where subscripts are used to denote the time-variable, for simplicity (e.g. $x_k = x[k]$). Noticing that $x_{k+1} = Ax_k$ (ignoring the input for now):

$$
\begin{aligned}
y_0 &= Cx_0 \\
y_1 &= Cx_1 = CAx_0 \\
y_2 &= CAx_2 = CA^2x_0 \\
&\vdots \\
y_{N-1} &= CAx_{N-1} = CA^{N-1}x_0
\end{aligned}
\tag{2.24}
$$

Factoring x_0, we can turn this into a matrix equation. This matrix is known as the Observability Matrix. Since it should only take N samples to take this matrix's inverse (making the matrix square), we can define the Observability Matrix, $\mathcal{O}$, as:

$$
\begin{aligned}
y &= \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{N-1} \end{bmatrix} x_0 \\
y &= \mathcal{O}x_0
\end{aligned}
\tag{2.25}
$$

where y, in this case, is a vector of the output samples at each point in time. It is clear from this equation that, in order to invert $\mathcal{O}$ (and therefore know x_0), $\mathcal{O}$ must be full-rank. When this is the case, we say that the system is "fully observable," or just "observable."

Continuing with the same logic, a related result can be obtained by considering the Controllability question: can we force the system's state to follow a desired trajectory using only its input and initial conditions? As it turns out, when this is possible, the given system's open-loop behavior (the set of poles and zeros) can be modified, using feedback, to make any system behavior we want. This is the foundation of state-variable feedback techniques (a subject which is, unfortunately, beyond the scope of this work).

We start by considering the state equations this time:

$$
\begin{aligned}
x_1 &= Ax_0 + Bu_1 \\
x_2 &= Ax_1 + Bu_2 \\
&\vdots \\
x_N &= Ax_{N-1} + Bu_N
\end{aligned}
\tag{2.26}
$$

Next, substituting the previous state equation for the current state, as before:

$$
\begin{aligned}
x_1 &= Ax_0 + Bu_0 \\
x_2 &= A(Ax_0 + Bu_0) + Bu_2 = A^2x_0 + ABu_0 + Bu_1 \\
x_2 &= A(Ax_1 + Bu_2) + Bu_2 = A^3x_0 + A^2Bu_0 + ABu_1 + Bu_2 \\
&\vdots \\
x_N &= A(x_{N-1} + Bu_N) = A^Nx_0 + A^{N-1}Bu_0 + A^{N-2}Bu_1 + \cdots + Bu_{N-1}
\end{aligned}
\tag{2.27}
$$

Subtracting the known contribution of x_0, we can define the Controllability matrix, $\mathcal{C}$ as:

$$
\begin{aligned}
x_N - A^Nx_0 &= \begin{bmatrix} B & AB & \dots & A^{N-1}B \end{bmatrix} u \\
x_N - A^Nx_0 &= \mathcal{C}u
\end{aligned}
\tag{2.28}
$$

Again, in order to solve this equation and find an input signal that generates the desired system response, $\mathcal{C}$ must be invertible, and therefore full-rank.

Since these matrices are invertible when the system is controllable or observable, it is possible to use them in a similarity transform. As it turns out, these transformations are related to the "Controllable Canonical Form" and the "Observable Canonical Form," respectively.

Though the following results are equally valid in DT, we now return to CT to discuss the Observable Canonical Form, for continuity with the rest of this chapter. The Observable Canonical Form has the following structure:

$$
\begin{aligned}
\dot{x} &= \begin{bmatrix} -a_{N-1} & 1 & 0 & \dots & 0 \\ -a_{N-2} & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -a_1 & 0 & 0 & \cdots & 1 \\ -a_0 & 0 & 0 & \cdots & 0 \end{bmatrix} x + \begin{bmatrix} b_{N-1} \\ b_{N-2} \\ \vdots \\ b_1 \\ b_0 \end{bmatrix} u \\
y &= \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \end{bmatrix} x
\end{aligned}
\tag{2.29}
$$

where the coefficients a_i are coefficients in the denominator of the system's transfer function and b_j are the coefficients in the transfer function's numerator. To confirm that the denominator coefficients are correct, writing out the determinant of this matrix (and, therefore, the denominator) shows that it takes the form: $\Delta(sI - A) = s^N + a_{N-1}s^{N-1} + a_{N-2}s^{N-2} + \cdots + a_0$.

There is a two-step process that can put any system into this form. Using the

transformation defined by $\mathcal{O}$ forces the A matrix to the correct form. However, an extra transform must be added to ensure that the B matrix contains the numerator coefficients. Let the subscript "o" denote a matrix written in this basis (e.g. A_o).

Let's first recognize that the observable form's observability matrix, $\mathcal{O}$, is equal to the identity matrix (since C is zero everywhere but its first index). Given this, we must find the similarity transform, T, that takes the state vector, x, from its original basis into the observable basis, $x_o = Tx$; which is to say that $\dot{x}_o = TAT^{-1}x_o + TBu$ and $y = \mathrm{CT}^{-1}x_o$. We can substitute this transform into the observable form's observability matrix:

$$\mathcal{O}_o = \begin{bmatrix} \mathrm{CT}^{-1} \\ \mathrm{CT}^{-1}(TAT^{-1}) \\ \mathrm{CT}^{-1}(TAT^{-1})^2 \\ \vdots \\ \mathrm{CT}^{-1}(TAT^{-1})^{N-1} \end{bmatrix} \tag{2.30}$$

The T terms inside the parentheses can be factored out, since $(TAT^{-1})^2 = TAT^{-1}TAT^{-1}\cdots = TA^2T^{-1}$. Canceling the inner T terms and factoring the right-most T, we see that:

$$\begin{aligned} &= \begin{bmatrix} \mathrm{CT}^{-1} \\ \mathrm{CT}^{-1}T(A)T^{-1} \\ \mathrm{CT}^{-1}T(A)^2T^{-1} \\ \vdots \\ \mathrm{CT}^{-1}T(A)^{N-1}T^{-1} \end{bmatrix} \\ &= \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{N-1} \end{bmatrix} T^{-1} \\ &= \mathcal{O}T^{-1} \end{aligned} \tag{2.31}$$

But we already said that $\mathcal{O}_o = \mathcal{O}T^{-1} = I$, so it must be the case that $\mathcal{O} = T$.

Using $\mathcal{O}$ as a transform ensures that the A matrix is in the correct form, but we still need to ensure the form for the B matrix. To do this, we need an additional transform:

$$W_o = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 & 0 \\ a_{N-1} & 1 & & \vdots & 0 & 0 \\ a_{N-2} & a_{N-1} & \ddots & 0 & \vdots & 0 \\ \vdots & a_{N-2} & \ddots & 1 & 0 & \vdots \\ a_2 & \vdots & \ddots & a_{N-1} & 1 & 0 \\ a_1 & a_2 & \cdots & a_{N-2} & a_{N-1} & 1 \end{bmatrix} \tag{2.32}$$

This new intermediate transform is lower-triangular and has the denominator coefficients on each of the off-diagonals.

Finally, the transform $T = W_o\mathcal{O}$ can be used to transform the system from the original basis into the Observable Canonical Form:

$$\begin{aligned} \dot{x}_o &= TAT^{-1}x_o + TBu \\ y &= \mathrm{CT}^{-1}x_o \end{aligned} \tag{2.33}$$

The same logic can be applied to the controllable form, which has the following structure:

$$\begin{aligned} \dot{x} &= \begin{bmatrix} -a_{N-1} & -a_{N-2} & \cdots & -a_1 & -a_0 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix} x + \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} u \\ y &= \begin{bmatrix} b_{N-1} & b_{N-2} & \cdots & b_1 & b_0 \end{bmatrix} x \end{aligned} \tag{2.34}$$

and with an intermediate transform of:

$$W_c = \begin{bmatrix} 1 & a_{N-1} & a_{N-2} & \cdots & a_2 & a_1 \\ 0 & 1 & a_{N-1} & \ddots & \ddots & a_2 \\ \vdots & 0 & 1 & \ddots & a_{N-2} & \vdots \\ 0 & \vdots & 0 & \ddots & a_{N-1} & a_{N-2} \\ 0 & 0 & \vdots & & 1 & a_{N-1} \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \tag{2.35}$$

we can construct the transform, $T = \mathcal{C}W_c$, used as follows (note that the inverses are reversed compared to Eq. (2.33)):

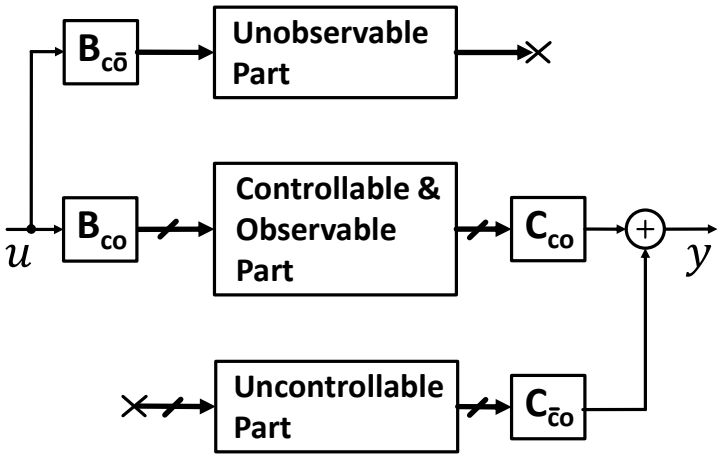


Figure 2.7: *The effective block diagram of a system with uncontrollable and unobservable modes. The uncontrollable/unobservable modes are cutoff from the input/output, respectively.*

$$\begin{aligned} \dot{x}_o &= T^{-1}ATx_o + T^{-1}Bu \\ y &= \mathrm{CT}x_o \end{aligned} \tag{2.36}$$

Finally, though we do not prove it in detail, uncontrollable/unobservable modes are akin to pole/zero cancellation in classical system analysis. The idea comes from a basis that is formed by the Kalman Decomposition, which separates unobservable/uncontrollable parts of the system. Though the details are beyond the scope of this work, the result is that uncontrollable poles are effectively cut off from the input and unobservable modes are cut off from the output. This has the same effect as pole/zero cancellation in classical analysis, except that this emphasizes that the internal dynamics still play a role in the short-term response, if there are initial conditions or disturbances (or the long-term response if those modes are unstable). A simplified block diagram depicting this situation is shown in Fig. 2.7.

Basis Transformations

Using the CCF, OCF, and eigenbases, it is finally possible to transform between two arbitrary bases. The main strategy is to use an intermediate basis whose transform is

known for any given matrix. If the two matrices are similar, then transforming them in this way gives the same result, which implies:

$$\begin{aligned} T^{-1}AT &= \bar{T}^{-1}\bar{A}\bar{T} \\ A &= T\bar{T}^{-1}\bar{A}\bar{T}T^{-1} \\ A &= (\bar{T}T^{-1})^{-1}\bar{A}\bar{T}T^{-1} \\ A &= \bar{T}_A^{-1}\bar{A}\bar{T}_A \end{aligned} \tag{2.37}$$

where T and $\bar{T}$ are transforms that take A and $\bar{A}$, respectively, to some shared basis and the new $\bar{T}_A = \bar{T}T^{-1}$ is the desired transformation. The shared basis used for this calculation may be any of the bases discussed here, such as the CCF, OCF, and eigen (or Jordan) bases.

The Matrix Exponential

An analytic function is one that can be described by a converging power series (such as its Taylor Series), shown in Eq. (2.38).

$$f(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)^2 + \dots \tag{2.38}$$

Since integer powers are simply the repeated multiplication of an argument ($A^3 = AAA$), this idea can be easily extended to matrices. Consider the fact that $(VAV^{-1})^k = VA^kV^{-1}$, since:

$$\begin{aligned} (VAV^{-1})^k &= (VAV^{-1})(VAV^{-1})(VAV^{-1})\dots \\ &= V(A)V^{-1}V(A)V^{-1}V(A)V^{-1}\dots \\ &= V(AAA\dots)V^{-1} \\ &= VA^kV^{-1} \end{aligned}$$

As a result, similarity transforms factor out of the function: $e^{VAV^{-1}} = Ve^AV^{-1}$! It is important to make clear, however, that a matrix does not generally stay in the same basis after applying a matrix function. For example, if A is in Observable Canonical Form, $log(A)$ is generally not. The main exception to this rule is diagonal matrices. Since diagonal matrices only have elements on the diagonal, raising the matrix to a power is the same as raising all its elements to the same power; thus, both input and output are diagonal. This gives a straightforward way to calculate functions: (1) transform into a diagonalized form, (2) calculate the function on the individual diagonal elements, (3) transform back.

One example that is used frequently in systems theory is the matrix exponential, e^A, which is used to solve for x in the state equations from Eq. (2.2). To show the response of the system to an input (its forced response), we can follow similar steps as in the 1-dimensional case. First, move the x term of the state equations to the left-hand side, then multiply both sides on the left by e^{-At} (to turn it into a known anti-derivative), and then integrate both sides:

$$\begin{aligned}
\dot{x}(t) &= Ax(t) + Bu(t) \\
\dot{x}(t) - Ax(t) &= Bu(t) \\
e^{-At}\dot{x}(t) - e^{-At}Ax(t) &= e^{-At}Bu(t) \\
\int_0^t \frac{d}{dt}\left(e^{-A\tau}x(\tau)\right)d\tau &= \int_0^t e^{-A\tau}Bu(\tau)d\tau \\
e^{-At}x(t) - e^{(-A0)}x_0 &= \int_0^t e^{-A\tau}Bu(\tau)d\tau
\end{aligned}$$

Solving for $x(t)$ gives the final solution:

$$x(t) = e^{At}x_0 + \int_0^t e^{A(t-\tau)}Bu(\tau)d\tau \tag{2.39}$$

As we discuss in Chapter 3, a direct result of this equation is that the matrix exponential (along with its inverse—the matrix form of the natural log) plays a key role in creating an equivalence between CT and DT filters, which is critical for the design of CT-$\Delta\Sigma$ modulators.

The DT State Solution

Finally, to round out our discussion, we note that the solution to the DT state equations (Eq. (2.10)) is simply the generalized version of Eq. (2.27):

$$\begin{aligned}
x[n] &= A^n x[0] + A^{n-1}Bu[0] + A^{n-2}Bu[1] + \cdots + Bu[n-1] \\
x[n] &= x[0]\,(A)^n + \sum_{k=0}^{n-1} A^{n-k-1}\,(B)\,u[k]
\end{aligned} \tag{2.40}$$

CHAPTER 3

Continuous-Time to Discrete-Time Conversion and the ELD Problem

In the late 1980s, it started to become clear that practical limitations in hardware implementation introduces extra, un-modeled delay in the feedback path which leads to performance degradation and instability [27]. This effect remains a critical, often difficult area of research; the problem is tightly coupled with the process of calculating the CT filter required to implement a given NTF, which has traditionally been a difficult problem in its own right.

In this chapter, we introduce an easy-to-use, closed-form solution for CT-DT conversion that fully accounts for Excess Loop Delay (ELD) and can be used with all standard ELD compensation techniques. Our solution allows for arbitrary DAC shapes spanning up to the first 2 sample periods, when loop delay is included.

We also introduce a related closed-form solution for the less-common compensation technique that use multiple DAC shapes, potentially allowing for DACs spanning >2 sample periods. Longer loop delays could, for example, enable much higher sampling rates, and therefore bandwidth, through the use of interleaved quantizers. The solution also includes the ability to control how each DAC connects to the filter, all without the need to tediously derive architecture-specific solutions.

3.1 Defining Impulse Invariance

The design problem introduced by Excess Loop Delay is intrinsically related to the process of converting between DT and CT loop filters. So, in order to discuss ELD, we must first elaborate on the ways in which this process can be carried out.

Fig. 3.2 shows the block diagrams of the DT- and CT-$\Delta\Sigma$ modulators. The DT modulator uses a DT loop filter, keeping the entire system in DT. In the CT

modulator, however, a DAC is used to drive a CT loop filter, which is finally sampled at the input to the quantizer. Because the quantizer's input and output are always in DT, the noise-shaping behavior of both $\Delta\Sigma$ modulators can be fully described in DT—any CT loop filter will have an exact DT analog. After all, any black-box filter with a DT input and DT output is, by definition, a DT system. [1]

The relationship between these two systems is described by the Impulse Invariant transform. It states that: a CT system, when driven by a DAC and sampled at its output, can be considered equivalent to a DT system when, for a given DT input, $u[n]$, the two systems have identical DT outputs at the sampling instant, $t = nT_s$. If y_d and y_c are the DT and CT outputs, we can say that: $y_d[n] = y_c(nT_s)$ for all n.

3.2 Impulse Invariance in Frequency Space

Traditional approaches to the Impulse Invariance problem are primarily rooted in the Laplace- and Z-transforms. At their core, they are interested in enforcing the equivalence between systems using the frequency domain. Let $\mathcal{L}$, $\mathcal{Z}$ be the Laplace- and Z-transforms, respectively, $\ell_c(t)$, $\ell_d[n]$ be the CT and DT loop filter impulse responses, and $L_c(s)$, $L_d(z)$ be the transformed versions of $\ell_c(t)$, $\ell_d[n]$. The relationship between DT and CT frequency domains, which must be solved, can be written as:

$$\begin{gathered} \ell_c(t)|_{t=nT} = \ell_d[n] \\ \text{or} \\ \mathcal{L}^{-1}\{L_c(s)\}|_{t=nT} = \mathcal{Z}^{-1}\{L_d(z)\} \end{gathered} \tag{3.1}$$

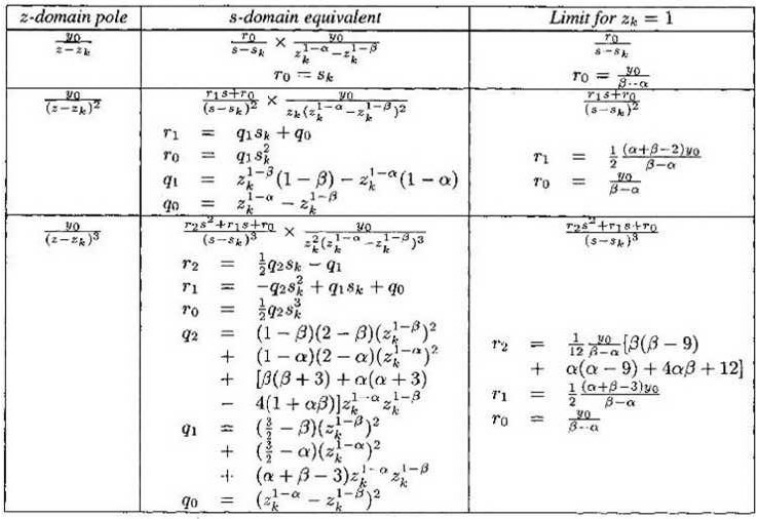
An example of the most common approach to this problem can be found in [1], which begins with the assumption that the loop filter is of the form:

$$L_d(z) = \frac{b_{N-1}z^{N-1} + b_{N-2}z^{N-2} \cdots + b_0}{(z-1)^N} \tag{3.2}$$

and that the DAC driving its input has a simple Return-to-Zero shape (the DAC's impulse response is 1 on the interval $t = [\alpha, \beta]$). This setup already limits the scope of the solution significantly, which is necessary due to the difficulty in applying Laplace- and Z-transforms directly.

From here, the partial fraction expansion is taken:

[1] However, as discussed in Section 3.4, the converse is not always true: not all DT systems have a CT equivalent. This happens when the matrices in Eq. (3.6) are not invertible.



z-domain pole	*s-domain equivalent*	*Limit for $z_k = 1$*
$\frac{y_0}{z-z_k}$	$\frac{r_0}{s-s_k} \times \frac{y_0}{z_k^{1-\alpha} - z_k^{1-\beta}}$ $r_0 = s_k$	$\frac{r_0}{s-s_k}$ $r_0 = \frac{y_0}{\beta-\alpha}$
$\frac{y_0}{(z-z_k)^2}$	$\frac{r_1 s + r_0}{(s-s_k)^2} \times \frac{y_0}{z_k(z_k^{1-\alpha} - z_k^{1-\beta})^2}$ $r_1 = q_1 s_k + q_0$ $r_0 = q_1 s_k^2$ $q_1 = z_k^{1-\beta}(1-\beta) - z_k^{1-\alpha}(1-\alpha)$ $q_0 = z_k^{1-\alpha} - z_k^{1-\beta}$	$\frac{r_1 s + r_0}{(s-s_k)^2}$ $r_1 = \frac{1}{2}\frac{(\alpha+\beta-2)y_0}{\beta-\alpha}$ $r_0 = \frac{y_0}{\beta-\alpha}$
$\frac{y_0}{(z-z_k)^3}$	$\frac{r_2 s^2 + r_1 s + r_0}{(s-s_k)^3} \times \frac{y_0}{z_k^2(z_k^{1-\alpha} - z_k^{1-\beta})^3}$ $r_2 = \frac{1}{2}q_2 s_k - q_1$ $r_1 = -q_2 s_k^2 + q_1 s_k + q_0$ $r_0 = \frac{1}{2}q_2 s_k^3$ $q_2 = (1-\beta)(2-\beta)(z_k^{1-\beta})^2$ $+ (1-\alpha)(2-\alpha)(z_k^{1-\alpha})^2$ $+ [\beta(\beta+3) + \alpha(\alpha+3)$ $- 4(1+\alpha\beta)]z_k^{1-\alpha} z_k^{1-\beta}$ $q_1 = (\frac{3}{2}-\beta)(z_k^{1-\beta})^2$ $+ (\frac{3}{2}-\alpha)(z_k^{1-\alpha})^2$ $+ (\alpha+\beta-3)z_k^{1-\alpha} z_k^{1-\beta}$ $q_0 = (z_k^{1-\alpha} - z_k^{1-\beta})^2$	$\frac{r_2 s^2 + r_1 s + r_0}{(s-s_k)^3}$ $r_2 = \frac{1}{12}\frac{y_0}{\beta-\alpha}[\beta(\beta-9)$ $+ \alpha(\alpha-9) + 4\alpha\beta + 12]$ $r_1 = \frac{1}{2}\frac{(\alpha+\beta-3)y_0}{\beta-\alpha}$ $r_0 = \frac{y_0}{\beta-\alpha}$

Figure 3.1: *The CT-DT conversion table in [1] used to equate Laplace- and Z-domain transfer functions. Using such a complex table is unwieldy and imposes significant limitations for its use, due to the assumptions required to generate it.*

$$L_d(z) = \frac{B_N}{(z-1)^N} + \frac{B_{N-1}}{(z-1)^{N-1}} \cdots + \frac{B_1}{(z-1)} \tag{3.3}$$

with B_i as scalars which depend on $L_d(z)$. The Z- to Laplace-domain conversion is somewhat easier to calculate on the individual terms in this form, allowing the transform to be applied to each term and the results added together in the Laplace-domain (thanks to the linearity of the transform). The transform for each term, in this case, was calculated by using the symbolic math software, Maple, and is shown in Fig. 3.1 for terms up to 3$^{\text{rd}}$ order. This table is unwieldy at best, even with the quite narrow limitations placed on the problem from the start.

[28] expands on this approach to allow for arbitrary DAC shapes by applying the definition of integration to integrate over the DAC shape itself. However, any method based on this approach is, at best, cumbersome; it especially does not scale well with order and still requires assumptions on the form of the loop filter. In [3], an approach using state-space methods is described, though it seems to be less common in the literature. We prefer this method, however, because it is scalable, easily implemented, and makes very few assumptions. Unfortunately, it doesn't provide any way to compensate ELD, perhaps partially explaining its minimal use.

This work builds on this state-space method by extending the standard matrix

equations to include the fast-path and a model for multi-cycle DAC shapes, making it possible to use impulse invariance with DAC shapes that span up to two cycles, which accounts for very nearly all examples in the literature. State-space impulse invariance is described in detail in Section 3.4.

3.3 Generality of Solution

The $\Delta\Sigma$ model investigated in this work is shown in Fig. 3.2. We do not consider any specific loop filter architectures, as keeping the loop filters abstract allows the result to be applied to any desired architecture. For that matter, it can quite naturally accommodate more complex filter models, such as those including excess poles due to parasitic capacitance or secondary amplifier poles.

The ELD correction method discussed is the classic fast-path around the quantizer, seen as the path c_0 in Fig. 3.3. As is discussed in detail in Section 3.5, this method works by making an extra degree of freedom available when equating the CT and DT filters, compensating for the complexity introduced when the DAC crosses into the second sample period.

Although we directly analyze only the fast-path approach, the result obtained can also be used for other conventional approaches, since all of the standard compensation techniques can be written as a simple manipulation of the c_0 path. Digital ELD correction [29], for example, is obtained by merely moving c_0's feedback point from the input of the quantizer to the output; the coefficient does not change. A summary of other approaches and their relation to the fast-path method can be found in [29].

3.4 Impulse Invariance in State-Space

As discussed in Section 3.1, calculating the loop filter in CT-$\Delta\Sigma$ modulators has typically been done through direct application of the Laplace- and Z-transforms. While an immense amount of work has been done on this technique, it can be tedious at best and downright confounding at worst. We contend that state-space tools can significantly simplify the solution and can contribute to a more complete understanding of the problem.

To restate the definition of impulse-invariance: a CT system, driven by a DAC and sampled at its output, is equivalent to a DT system when, for a given DT input, $u[n]$, the two systems have identical DT outputs. That is to say: $y_d[n] = y_c(nT_s)$ for all time n. As discussed in Section 3.1, directly applying the Laplace- and Z-transforms

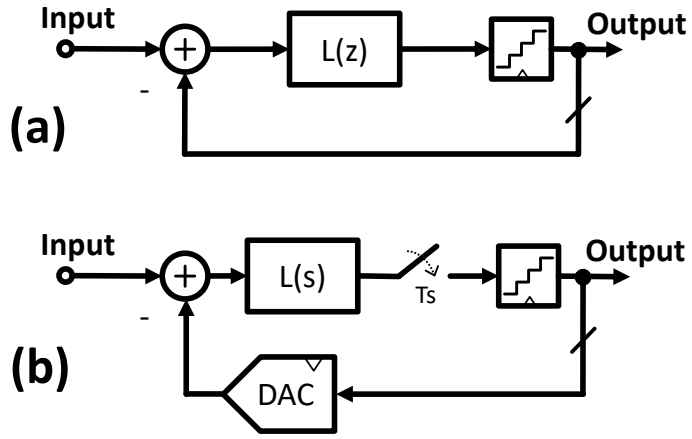


Figure 3.2: *Generic model of* ***(a)*** *a discrete-time $\Delta\Sigma$ modulator and* ***(b)*** *a continuous-time $\Delta\Sigma$ modulator used in this work.*

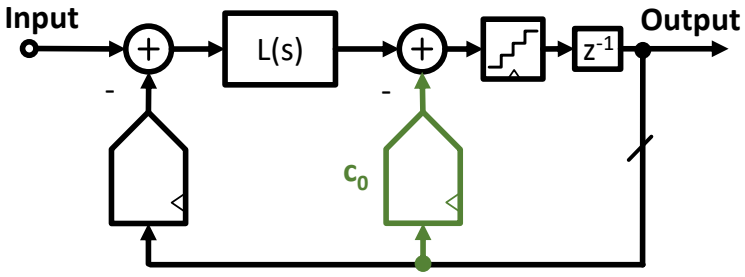


Figure 3.3: *$\Delta\Sigma$ modulator block diagram including the excess loop delay compensation path, c_0.*

to solve this problem results in large, unwieldy tables of equations that must, at minimum, be customized for the type of DAC being used. However, a compact solution in state-space is well-known and is described in [3]. First, let $\phi(t)$ be the impulse response of the DAC. An example of $\phi(t)$ for a Return-to-Zero (Return-to-Zero (RZ)) DAC is shown in Fig. 3.4.

The state equations for a DT filter are:

$$\begin{aligned} x_d[k+1] &= A_d x_d[k] + B_d u[k] \\ y_d[k] &= C_d x_d[k] + D_d u[k] \end{aligned} \tag{3.4}$$

and for a CT filter:

$$\begin{aligned} \dot{x}_c(t) &= A_c x_c(t) + B_c u_c(t) \\ y_c(t) &= C_c x_c(t) + D_c u_c(t) \end{aligned} \tag{3.5}$$

These equations describe the time evolution of any linear, time-invariant system (DT or CT, respectively) and, because impulse-invariance is defined in the time-domain, the impulse-invariant relationship between these two system models can be written as a simple relationship between the A,B,C and D matrices:

$$\begin{aligned} A_d &= e^{A_c T_s} \\ B_d &= \int_0^{T_s} e^{A_c(T_s-\tau)} B_c \phi(\tau) d\tau \\ C_d &= C_c \\ D_d &= D_c \end{aligned} \tag{3.6}$$

Derivations of these equations are readily available, as in [3]. To emphasize the assumptions being made, however, we give an example here. Consider that the solutions to the CT and DT systems, with CT input $u_c(t)$ and DT input $u_d[n]$, are (Eq. (2.39), and Eq. (2.40)):

$$\begin{aligned} x_c(t) &= x_c(0) e^{A_c t} + \int_0^t e^{A_c(t-\tau)} B_c u_c(\tau) d\tau \\ x_d[n] &= x_d[0] \left(A_d\right)^n + \sum_{k=0}^{n-1} A_d^{n-k-1} \left(B_d\right) u_d[k] \end{aligned} \tag{3.7}$$

The goal of the derivation is to manipulate the CT equation so that it is in the same form as the DT equation, then pick out the components that stand in for A_d and B_d in the new equation. Focusing on the CT response, the state of the system

at time $t = nT_s$ is:

$$x_c(nT_s) = x_c(0)e^{A_c(nT_s)} + \int_0^{nT_s} e^{A_c(nT_s-\tau)} B_c u_c(\tau) d\tau \tag{3.8}$$

Already, we can see that $A_d = e^{A_c T_s}$, since we can write the first term in the form $x_c(0)\left(e^{A_c T_s}\right)^n = x_d[0]\left(A_d\right)^n$. Now, assuming that $\phi(\tau) = 0$ outside the interval $[0, T_s]$, we can break up the integral by sample-period, allowing us to replace $u_c(\tau)$ with variables representing the DAC and DT input in the k^{th} sample period, $u_c(\tau) = \phi(\tau - kT_s)u_d[k]$:

$$\begin{aligned} x_c(nT_s) = x_c(0)e^{A_c(nT_s)} & \\ + \int_{(n-1)T_s}^{nT_s} e^{A_c(nT_s-\tau)} B_c \phi(\tau - nT_s) u_d[n] d\tau & \\ \vdots & \\ + \int_{T_s}^{2T_s} e^{A_c(nT_s-\tau)} B_c \phi(\tau - T_s) u_d[1] d\tau & \\ + \int_0^{T_s} e^{A_c(nT_s-\tau)} B_c \phi(\tau) u_d[0] d\tau & \end{aligned} \tag{3.9}$$

This allows us to factor out $u_d[k]$ from the integral:

$$x_c(nT_s) = x_c(0)e^{A_c(nT_s)} + \sum_{k=0}^{n-1} \int_{kT_s}^{(k+1)T_s} e^{A_c(nT_s)-\tau)} B_c \phi(\tau - kT_s) d\tau \; u_d[k] \tag{3.10}$$

After using $\tau = \lambda + kT_s$ as a change of variables and factoring out the constant $e^{A_c T_s(n-k-1)}$ term from the integral, the equation has the same form as the DT state-space solution and we can equate the terms in parentheses:

$$\begin{aligned} x_c(nT_s) &= x_c(0)\left(e^{A_c T_s}\right)^n + \sum_{k=0}^{n-1} \left(e^{A_c Ts}\right)^{(n-k-1)} \left(\int_0^{T_s} e^{A_c(T_s-\lambda)} B_c \phi(\lambda) d\lambda\right) u_d[k] \\ x_d[n] &= x_d[0]\left(A_d\right)^n + \sum_{k=0}^{n-1} \left(A_d\right)^{n-k-1} \left(B_d\right) u_d[k] \end{aligned} \tag{3.11}$$

Notice that, in order to split the integral to a summation of integrals in Eq. (3.10), $\phi(t)$ must be zero outside the first sample-period, $t = [0, T_s]$. As long as the DAC's

impulse response—with delay included—stays in that interval, Eq. (3.6) works perfectly well, regardless of the delay seen at the DAC. The ELD design problem, then, is just an effort to expand this limitation to include multiple sample periods, as is discussed in Section 3.5.

Finally, before moving on, it is useful to make a few simple observations of Eq. (3.6). These conclusions are not new, but can be helpful in orienting us to these equations.

1. The integral in Eq. (3.6) is not invertible when the DT system has poles at z=0 (since if $A_d = e^{A_c * T_s}$ has eigenvalues at 0, the integral does too). This implies that pure DT delays can't be converted to CT—it would require a pole at $-\infty$! This isn't typically a problem, however, since these can be easily implemented separately in DT as delays at the input to the DAC.

2. The poles of the DT system is always the exponent of the CT poles (since the matrix exponential of a diagonal matrix is the exponent of its elements and $e^{A_c T_s} = V e^{\Lambda_c T_s} V^{-1}$, where Λ_c is a diagonal matrix containing the eigenvalues of A_c—see Section 2.2.2).

3. The DAC's only effect is to modify the filter's zeros (and the overall gain) since it only appears in the equation relating the B matrices (see Section 2.2.2).

3.5 Extending Impulse Invariance: The ELD Design Problem

As discussed in Section 3.4, loop delay, in and of itself, does not present a problem as long as the DAC shape + delay (that is, $\phi(t)$) remains zero outside the first sample period, $[0, T_s]$. An RZ DAC shape, along with a delayed version of it, is shown in Fig. 3.4 as an example. The problem is introduced when $\phi(t)$ is allowed to extend into the second sample period, $[T_s, 2T_s]$ [1]. In this case, the assumptions made when deriving the equivalence in Eq. (3.6) are no longer upheld, and the transformation cannot be applied.

In order to apply impulse invariance in this case, [1] considers the system as a superposition of two different DACs: DAC_1 representing the interval $[0, T_s]$, DAC_2 representing $[T_s, 2T_s]$. DAC_2 is now 0 for the first cycle, which can be implemented as a unit delay added before DAC_2, as illustrated in Fig. 3.5. As a result, DAC_2 only

has to implement a DAC shape on the interval $[0, T_s]$, and impulse invariance can be applied as usual.

The problem is, this extra delay results in a filter with one less degree of freedom than is required to solve for its coefficients. Consider the result of this superposition: the DT equivalent of the CT filter + DAC system can be thought of as two parallel DT filters, shown in Fig. 3.6. One filter represents the DT equivalent of the CT filter using only DAC_1 ($L_{d1}(z)$); the other represents the conversion using only DAC_2, with the unit delay included before it ($z^{-1}L_{d2}(z)$). Algebraically, this means we must solve:

$$L_d(z) = L_{d1}(z) + z^{-1}L_{d2}(z) = \frac{N_{d1}(z)}{D_{d1}(z)} + \frac{N_{d2}(z)}{zD_{d2}(z)} \tag{3.12}$$

Since the only thing changing is the DAC, the poles of these systems are the same ($D_{d1}(z) = D_{d2}(z) = D_d(z)$). Therefore, the denominators cancel, and we are left with:

$$zN_d = zN_{d1} + N_{d2} \tag{3.13}$$

Eq. (3.13) now relates the numerator of the desired transfer function to the numerators of the impulse-invariant filters due to DAC_1 and DAC_2. Equating the coefficients on both sides of Eq. (3.13) shows that the constant term in N_{d2} must be 0, since both N_d and N_{d1} are being multiplied by a z term. This requirement cannot be guaranteed, however, since we cannot set the coefficients in N_{d2} independently from N_{d1}. Therefore, an extra degree of freedom is required so we can set this coefficient independently.

The solution suggested in [1] is to add an extra path around the quantizer (c_0 in Fig. 3.3), adding a new degree of freedom and making a solution possible—this is the now-standard fast-path method of ELD Compensation. Our aim in this section is to follow the method in [1], developing state-space equations in a way that mirrors their approach and, finally, produce an extended version of Eq. (3.6) that properly compensates systems with DACs that span 2 cycles.

In state-space, the DT systems can always be written in the Observable Canonical Form, as discussed in Chapter 2, making the B_{d1} and B_{d2} matrices (of the state-space representations of $L_{d1}(z)$ and $L_{d2}(z)$, respectively) equal to the coefficients of their respective numerators. In Eq. (3.13), the coefficients of N_d and N_{d1} shift up by one degree due to the extra z term from DAC_2's delay. Noting this, we can extend the B matrices—whose elements are these same coefficients—by shifting them up or zero-padding them, as necessary. Equating the coefficients in Eq. (3.13), we can rewrite

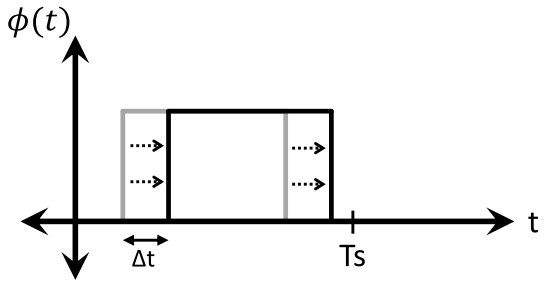


Figure 3.4: *A return-to-zero DAC shape and the same shape with small added delay, Δt.*

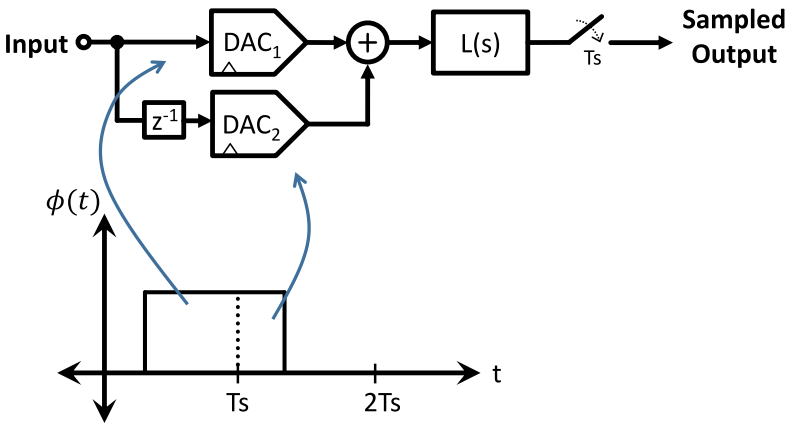


Figure 3.5: *The system equivalent to a DAC that spans two sample periods. DAC_1 implements the portion of the DAC shape that falls inside the first cycle, and DAC_2 implements the second cycle. Together, they are equivalent to the full DAC pulse shape.*

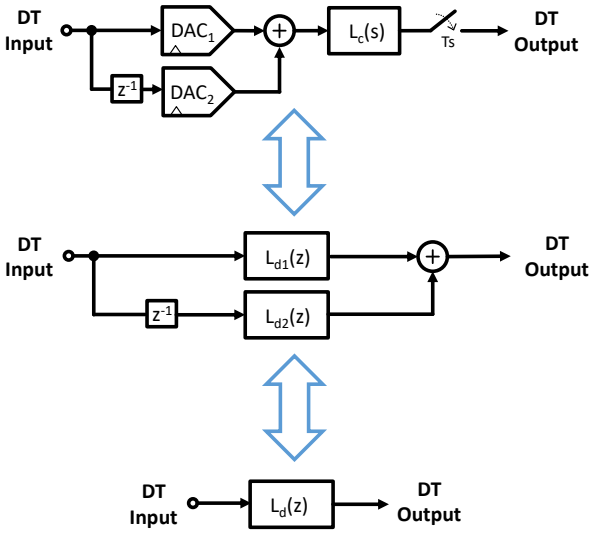


Figure 3.6: ***(top)*** *The model of the CT DAC + Filter.* ***(mid)*** *The DT equivalent of (top).* ***(bot)*** *The desired DT filter. All three should be made to be equivalent.*

the relationship as:

$$\begin{bmatrix} B_d \\ 0 \end{bmatrix} = \begin{bmatrix} B_{d1} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ B_{d2} \end{bmatrix} \tag{3.14}$$

Substituting for B_{d1}, B_{d2} using the state-space Impulse Invariant Transform (Eq. (3.6)) and noting that B_c can be factored out, we see that:

$$\begin{aligned} \begin{bmatrix} B_d \\ 0 \end{bmatrix} &= \left(\begin{bmatrix} \int_0^{T_s} e^{A_c T_s t} \phi(t) dt \\ 0 \quad 0 \quad \dots \quad 0 \end{bmatrix} + \begin{bmatrix} 0 \quad 0 \quad \dots \quad 0 \\ \int_0^{T_s} e^{A_c T_s t} \phi(t + T_s) dt \end{bmatrix} \right) B_c \\ &= \Gamma B_c \end{aligned} \tag{3.15}$$

Since Γ is (N+1)xN, an exact inverse doesn't exist. This is the central design problem for ELD restated: we must find a way to increase the degrees of freedom in the CT system so that Γ—the matrix relating the CT filter's unknown B_c matrix with the known DT B_d matrix—can be inverted.

The fast-path method in [1] solves this by adding an extra path—shown as the c_0 path in Fig. 3.3 and represented by adding a scalar c_0 term to the right-hand side of Eq. (3.12). After distributing the denominators in this modified equation, we see that: $zNum_d = zNum_{d1} + Num_{d2} + c_0 Den_d$, which can now be solved. In the same way as before, we can rewrite the coefficients of this equation in matrix form:

$$\begin{aligned}\begin{bmatrix} B_d \\ 0 \end{bmatrix} &= \begin{bmatrix} Den_d & \Gamma \end{bmatrix} \begin{bmatrix} c_0 \\ B_c \end{bmatrix} \\ &= \Gamma_{aug} \begin{bmatrix} c_0 \\ B_c \end{bmatrix}\end{aligned} \tag{3.16}$$

where:

$$\text{Den}_d \quad = \quad \begin{bmatrix} 1 & a_{N-1} & a_{N-2} & \cdots & a_0 \end{bmatrix}^T$$

Den_d here is a column vector containing the denominator coefficients from highest degree to lowest. The matrix is now square and, in most practical cases, it is invertible though it is beyond the scope of this work to prove precisely when it is invertible. In practice, if a system is found that is non-invertible or nearly non-invertible, making it invertible typically only requires some small change to the DAC shape or the filter's poles.

While this accounts for the most common modulators in the literature, it is not all-inclusive. It is possible, with other system configurations, for an otherwise convertible system (no poles on the negative real axis) to be non-convertible because Γ_{aug} is not invertible. If this problem is encountered in practice, choosing a different set of filter poles or a different DAC shape (which can be as simple as adding delay) is the only solution. More work should be done to determine the conditions under which this happens.

Finally, the matrix in Eq. (3.16) is square so, inverting it and combining the result with the original transformation of Eq. (3.6), we can write the complete transformation from DT to CT for any system with a DAC that spans two cycles as:

$$\begin{aligned} A_c &= \frac{1}{T_s} ln(A_d) \\ \begin{bmatrix} c_0 \\ B_c \end{bmatrix} &= \begin{bmatrix} Den_d & \Gamma \end{bmatrix}^{-1} \begin{bmatrix} B_d \\ 0 \end{bmatrix} \\ C_c &= C_d \\ D_c &= D_d \end{aligned} \tag{3.17}$$

where:

$$\Gamma \quad = \quad \begin{bmatrix} \int_0^{T_s} e^{A_c T_s t} \phi(t) dt \\ 0 \quad 0 \quad \ldots \quad 0 \end{bmatrix} + \begin{bmatrix} 0 \quad 0 \quad \ldots \quad 0 \\ \int_0^{T_s} e^{A_c T_s t} \phi(t + T_s) dt \end{bmatrix}$$

and that the discrete-time system is in the Observable Canonical Form.

Doing the conversion the other way—from CT to DT this time—the order of the DT system must be increased (to account for DAC_2's delay) and, for most choices of c_0, the higher-order B_d matrix does not have the form in Eq. (3.6). In fact, the correct choice of c_0 (given by Eq. (3.17)) makes the systems equivalent, even though they are not of the same order, by forcing one of $L_d(z)$'s zeros to the origin, canceling the pole created by DAC_2's delay. By augmenting A_d and B_d to include the effects of DAC_2, we see that the CT to DT transformation is:

$$\begin{aligned} A_d &= \begin{bmatrix} e^{A_c T_s} & B_{d2} \\ 0 & 0 \end{bmatrix} \\ B_d &= \begin{bmatrix} B_{d1} \\ 1 \end{bmatrix} \\ C_d &= \begin{bmatrix} C_c & c_0 \end{bmatrix} \\ D_d &= D_c \end{aligned} \tag{3.18}$$

where:

$$\begin{aligned} \mathrm{B}_{d1} &= \int_0^{T_s} e^{A_c T_s t} B_c \phi(t) dt \\ \mathrm{B}_{d2} &= \int_0^{T_s} e^{A_c T_s t} B_c \phi(t + T_s) dt \end{aligned}$$

The relatively compact equations given in Eq. (3.17) and Eq. (3.18) represent, finally, the general solution for classical excess loop delay compensation. Enforcing these equations guarantees that the response of the sampled CT and DT filter models are identical, cycle-for-cycle. Symbolic solutions can be found for some DAC shapes, if necessary. However, numerical solutions are simple to implement, easily automated, and significantly less cumbersome. A link to open-source examples of these solutions are given in Section 6.1.

3.6 An Aside: When ELD Correction is Unnecessary

In general, a designer may wish to use multiple DACs with distinct impulse responses [30][31][32]. This situation can be represented in state-space by making $\phi(t)$ a column vector of functions, each function representing a distinct DAC impulse response. If R is the number of DACs in $\phi(t)$ and N is the system's order, then B_c becomes NxR rather than Nx1. The i^{th} column of B_c represents the unique connection of the i^{th} DAC into the filter. A second-order example is shown in Fig. 3.7. When the two

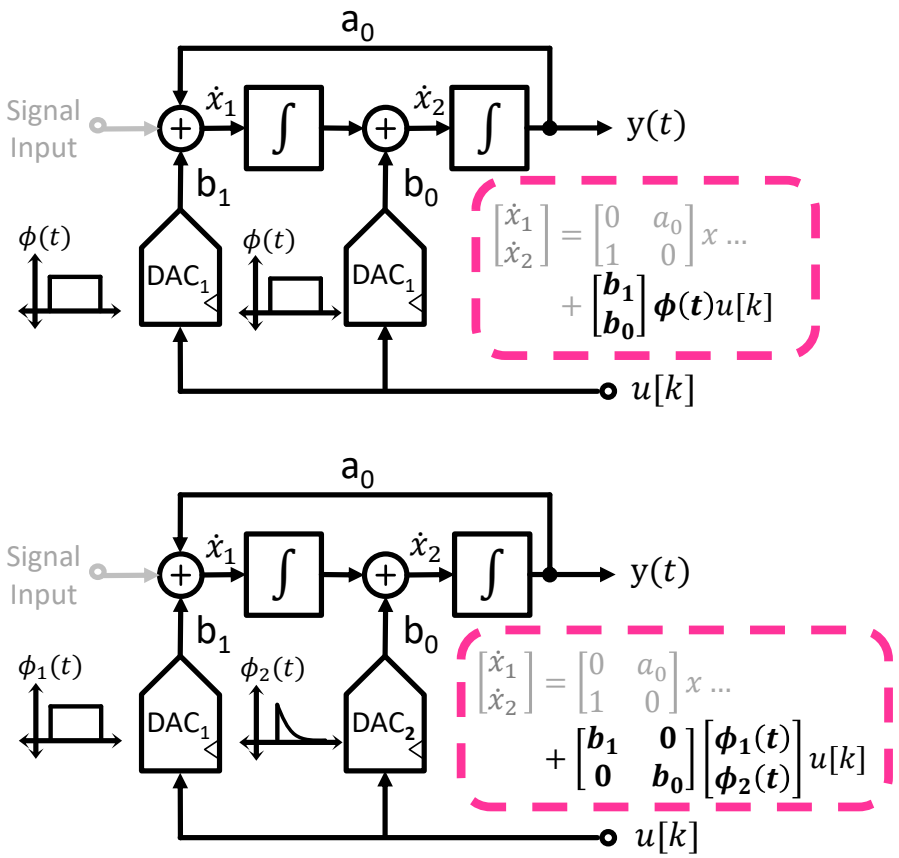


Figure 3.7: *An example of a 2^{nd} order loop filter with* ***(top)*** *a single RZ DAC shape and* ***(bot)*** *two distinct DAC shapes—an RZ and a switched capacitor DAC. The state equations for each case are given, illustrating the modifications required to account for multiple DACs.*

DACs have the same shape, $\phi(t)$ is a scalar function, as usual. However, when the two DAC shapes are distinct, $\phi(t)$ and B_c must be expanded.

When DT system is known, the calculation of B_c must begin by factoring it out of the integral in Eq. (3.6), allowing the inverse of the NxN matrix to be taken, as in the simpler single-DAC case:

$$B_c = \left[\int_0^{T_s} e^{A_c(T_s-t)}\phi(t)dt\right]^{-1} B_d \tag{3.19}$$

For systems using multiple DAC responses, however, this is not possible since $\phi(t)$ is no longer a scalar function and, therefore, cannot be commuted with B_c. To get around this, Eq. (3.19) can be re-written, using the linearity of integration, as:

$$
\begin{aligned}
B_d &= \int_0^{T_s} e^{A_c(T_s-t)} \begin{bmatrix} I\phi_1(t) & I\phi_2(t) & \dots & I\phi_k(t) \end{bmatrix} \begin{bmatrix} B_{c1} \\ B_{c2} \\ \vdots \\ B_{ck} \end{bmatrix} dt \\
&= \left[\int_0^{T_s} e^{A_c T_s t} \Phi(t) dt \right] B_c^*
\end{aligned}
\tag{3.20}
$$

where:

I	=	the NxN identity matrix
$\Phi(t)$	=	the expanded matrix of ϕ's elements
B_{ci}	=	the i$^{\text{th}}$ column of B_c
B_c^*	=	the vertical concatenation of the columns of B_c.

Finally, we can expand this equation to include the second sample period, as we did in Section 3.5. If the system is in Observable Canonical Form, we have:

$$
\begin{aligned}
\begin{bmatrix} B_d \\ 0 \end{bmatrix} &= \left(\begin{bmatrix} \int_0^{T_s} e^{A_c T_s t} \Phi(t) dt \\ 0 \quad 0 \quad \dots \quad 0 \end{bmatrix} + \begin{bmatrix} 0 \quad 0 \quad \dots \quad 0 \\ \int_0^{T_s} e^{A_c T_s t} \Phi(t+T_s) dt \end{bmatrix} \right) B_c^* \\
&= \Gamma_\Phi B_c^*
\end{aligned}
\tag{3.21}
$$

where Γ_Φ is defined by the expanded integral matrices in parentheses.

Since the matrix that must be inverted in this case is (N+1)x(R*N), there typically exists infinitely many choices of B_c that satisfy Eq. (3.21). In fact, if $R > 1$, an additional ELD path is not necessary since, as discussed in Section 3.5, the ELD path is only required to solve the problem of inverting a matrix with more rows than columns—this one has more columns than rows.

A naive approach to solving this problem might be to try the least-squares solution, which is given by the well-known Moore-Penrose Pseudoinverse [33]. While this can work for some designs, architecture choice usually adds restrictions to B_c's form (e.g., each DAC must only be connected to a particular part of the filter) that won't necessarily be met this way. In order to ensure that B_c is in the form that is desired, we have to "restrict" the solution space to some subspace defined by the matrix P_s, consisting of the subspace's basis-vectors. This is called a Restricted Generalized Inverse problem, and its solution can be found in [25]. Interpreted for our context, we see that:

$$
B_c^* = P_s \left(\Gamma_\Phi P_s \right)^\dagger B_d \tag{3.22}
$$

where † denotes a pseudoinverse such as Moore-Penrose. An exact solution is not always guaranteed, however, in which case the system cannot be compensated. The designer may have to try different configurations to find the best strategy. Future work should explore the requirements for an exact solution.

The simplest way to define P_s is as a diagonal matrix with 1's only on elements that correspond to elements in B_c^* that should be non-zero. One way to think about this is that P_s maps an arbitrary vector, x, into a vector with the desired form, B_c^*. For example, for B_c^* of the form $\begin{bmatrix} b_1 & 0 & b_3 & 0 \end{bmatrix}^T$, P_s we can construct P_s as follows:

$$\begin{aligned} \begin{bmatrix} b_1 \\ 0 \\ b_3 \\ 0 \end{bmatrix} &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} \\ &= P_s x \end{aligned} \tag{3.23}$$

One further modification must be added to make this solution practical. The meaning of the state signals change if we change the basis—connecting the DAC to the inputs of one particular integrator in one filter architecture is vastly different from connecting to that same integrator in a different architecture. So, specifying a particular way of connecting the DACs (i.e., restricting B to the subspace defined by P_s) requires first specifying that the state-space system must directly represent the desired filter architecture (i.e., it must be in the right basis). The problem is, the equations we have been using thus far (e.g., Eq. (3.17)) assume the system is written in Observable Canonical Form (OCF). Since the inversion happens in this basis, we would only be able to set the DAC connections for the filter architecture this basis describes, which is very limiting, to say the least. To fix this, we have to do a change-of-basis on B_c so that the matrix we invert starts in the correct basis.

The goal is to find a change-of-basis matrix, V_c that takes $A_c^{(des)}$ (that is, A_c written in the desired basis) to the basis that we arrive at after taking $\ln(A_d^{(ocf)})/T_s$ (A_d in OCF form). Fig. 3.8 shows the relationships between the relevant bases and their corresponding change-of-basis matrices.

Finding V_c is more straightforward than it seems, once you notice that any change of basis matrix in DT must also apply to the CT basis, since V factors out of the equation as in: $\ln(V_d A_d V_d^{-1})/T_s = V_d \ln(A_d) V_d^{-1}/T_s = V_d A_c V_d^{-1}$. Thus, $V_c = V_d$, which is simply the OCF transform (Section 2.2.2) for $A_d^{(des)} = \ln(A_c^{(des)})/T_s$, which is already known.

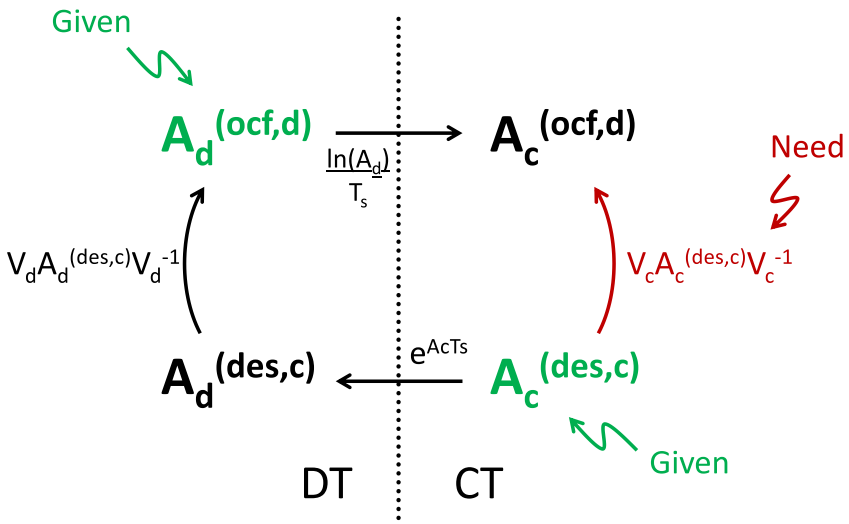


Figure 3.8: *The relationship between each basis used to reconstruct the desired basis (lower right). It is assumed that A_c in the desired basis can be calculated from $A_d^{(ocf)}$, which is a standard part of filter design (i.e., calculating circuit components from the transfer function). The arrows going left and right take each system from DT to CT (and vice versa), whereas vertical arrows move between different bases, remaining in either DT or CT.*

Finally, we repeat the conversion from B_c into B_c^* (Eq. (3.20)). Applying the similarity transform to B, we have: $B_c^{(ocf,d)} = V_c B_c^{(des,c)}$. V_c is simply multiplied by each column of $B_c^{(des)}$ so, concatenating the two resulting columns in the same way as Eq. (3.20) gives:

$$\begin{aligned} B_c^{*(ocf)} &= \begin{bmatrix} V B_{c,1}^{(des)} \\ V B_{c,2}^{(des)} \end{bmatrix} \\ &= \begin{bmatrix} V & 0 \\ 0 & V \end{bmatrix} B_c^{*(des)} \\ &= V^{*(des)} B_c^{*(des)} \end{aligned} \tag{3.24}$$

where $B_{c,i}$ is the i$^{\text{th}}$ column of B_c. With this, we can rewrite Eq. (3.21):

$$\begin{aligned} \begin{bmatrix} B_d \\ 0 \end{bmatrix} &= \Gamma B_c^{*(ocf)} \\ &= \Gamma \left(V^{*(des)} B_c^{*(des)} \right) \end{aligned} \tag{3.25}$$

Since $B_c^{*(des)}$ is exposed on the right, inverting everything else with the pseudo-inverse of Eq. (3.22) gives us an equation that (1) puts B_c into the desired basis and (2) imposes a restriction on the form of B_c, as defined by the matrix P_s:

$$B_c^{*(des)} = P_s \left(\Gamma V^{*(des)} P_s \right)^{\dagger} \begin{bmatrix} B_d \\ 0 \end{bmatrix} \tag{3.26}$$

This equation, finally, ensures the respective DACs are only connected to certain parts of the loop filter.

To conclude this discussion, we note that the extra ELD path required in the classical approach of Section 3.5 has been a considerable focus in the literature over the last decade, largely because adding a summing node at the input to the quantizer typically incurs an extra, undesirable amplifier [34] [35]. This approach is a potential way to bypass the need for that path altogether, avoiding the introduction of the extra power and area of an amplifier.

3.7 Simulations

In order to demonstrate the proposed impulse-invariance extension derived in Section 3.5, several example systems have been simulated using Simulink. A block diagram of the approach can be seen in Fig. 3.9. First, since the filters under test

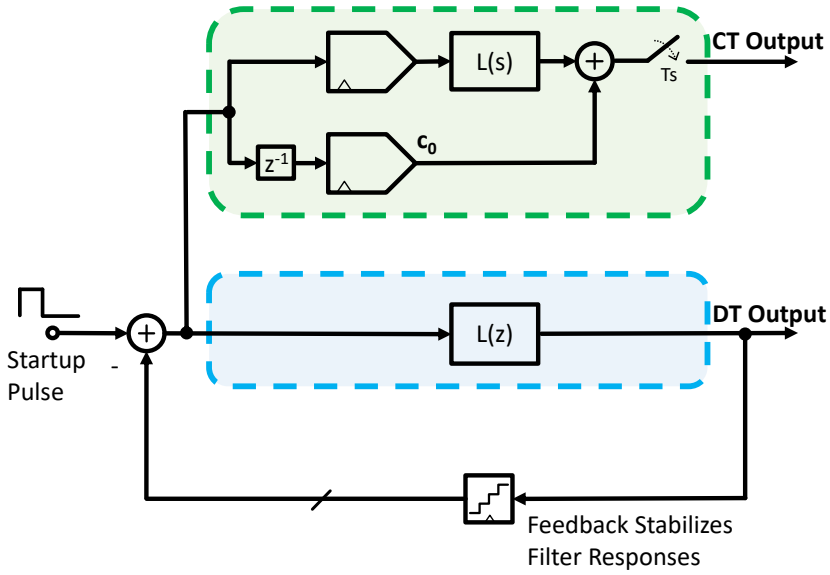


Figure 3.9: *The structure used to test for impulse-invariance. The DT filter and CT filter are both given the same DT input and their outputs are compared at the sampling instant. Since $\Delta\Sigma$ loop filters are not typically stable by design, feedback is applied to the DT filter in order to stabilize the output signals and an impulse is injected, in the first sample, to start the modulation.*

are not typically open-loop stable, the ideal DT filter is put in feedback. Next, the now-stabilized ideal filter's input signal is applied to the filter under test. Since all filters being compared are given the same DT input, the filters are equivalent, by the definition of impulse invariance, only if they have the same DT output for all time.

This equivalency condition is shown for three different loop filters in Fig. 3.10. In this test, one DT filter is compared with two CT filters, each of which calculated from the DT filter using Eq. (3.17). Each CT filter uses a different DAC with different amounts of delay applied, and each is shown to produce identical outputs at each sample. Fig. 3.10 demonstrates that the outputs of each filter are visually identical, and this is confirmed numerically (within the error tolerance of the simulation) up to the first million samples. We conclude that this is sufficient evidence that all three loop filters are identical, in the impulse invariant sense.

The same test was run to verify the multi-DAC solution from Section 3.6. The subspace matrix used is:

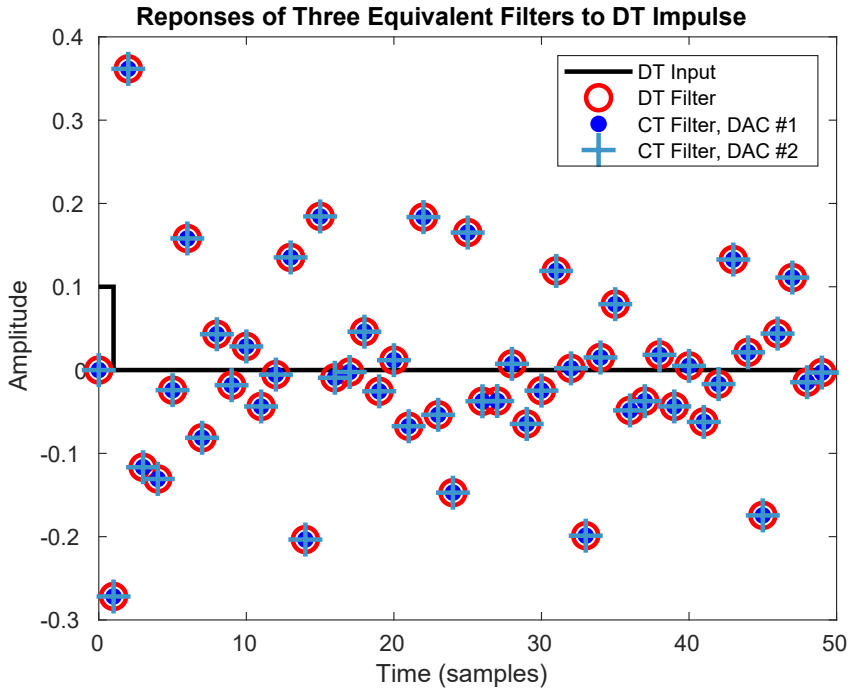


Figure 3.10: *The output of three equivalent loop filters, each being driven by the same DT input sequence, as in Fig. 3.9. DAC #1 is an NRZ DAC with no delay, whereas DAC #2 is an NRZ-DAC with a 1/2 cycle delay. The markers for each filter are chosen so they can be distinguished when overlapped, indicating that the filters' outputs are overlapped perfectly at every point in time.*

$$P_s = \begin{bmatrix} 1 & & & & & & \\ & 0 & & & & & 0 \\ & & 1 & & & & \\ & & & 1 & & & \\ & & & & 0 & & \\ & & & & & 1 & \\ & 0 & & & & & 0 \\ & & & & & & & 1 \end{bmatrix}$$

where the large zeros indicate that all other elements are filled with zeros. Applying this to Eq. (3.26), the B_c matrix (without any gain equalization) would be:

$$B_c = \begin{bmatrix} 17.9 & 0 \\ 0 & -154.3 \\ 216.6 & 0 \\ -133.3 & 55.9 \end{bmatrix}$$

meaning that DAC_1 is only connected to the 1^{st}, 3^{rd}, and 4^{th} integrators, whereas DAC_2 is only connected to the 2^{nd} and 4^{th} integrators. Fig. 3.11 shows the results of this test, the perfect overlap of both output signals indicating that the delay has been successfully compensated with no ELD path—only 2 distinct DACs. Again, this is confirmed numerically up to the first million samples.

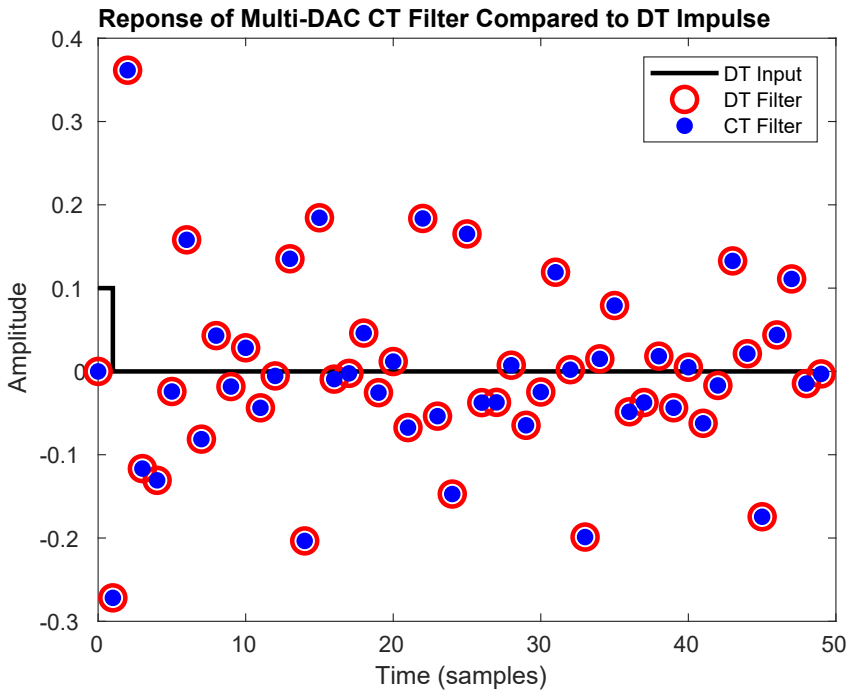


Figure 3.11: *The output of a loop filter with a 2-DAC compensation scheme compared with the ideal DT loop filter, setup as in Fig. 3.9. Both DACs are RZ DACs with pulse width of Ts/2. The delay is Ts/2 for DAC_1 and Ts for DAC_2. The markers for each filter are chosen so they can be distinguished when overlapped, indicating that the filters' outputs are overlapped perfectly at every point in time.*

CHAPTER 4

A Limitation on CT ΔΣ Modulation

In order to push the boundaries of the complexity realizable in a ΔΣ modulator, as multi-band ΔΣ modulators must, there is little room for approximation and cumbersome hand-calculation. In Chapter 3, we improve on the latter by introducing a compact, generalized, automatable way of making the connection between the continuous and discrete domains. However, while the performance of CT modulators in the literature has been quite impressive to date, the measured NTF shapes often have a lot of non-ideal features, as can be seen in [2] [36] and [37], for example. The spectrum from [2] is shown in Fig. 4.1. The modulator is incredibly well-designed, judging by its breakthrough Figure of Merit (FoM), and it is the first modulator to demonstrate the viability of a two-step quantization/feedback system. Even with such impressive performance, however, its NTF is significantly different from the type of NTF typically associated with an ideal DT modulator; in theory, they should be the same. In the ideal DT case, the out-of-band noise is flat, but in this case, the lower out-of-band frequencies are significantly boosted while the higher frequencies roll off. Without access to the original design, it is difficult to be completely certain of the cause, of course. Part of this deviation can likely be attributed to the reality of non-ideal amplifier parameters like extra, high-frequency amplifier poles. However, we contend that ELD introduces a surprising restriction on the shape of the NTF that can give strikingly similar results, even with perfectly ideal amplifiers. An example of such an NTF is shown in Section 4.4.3.

In this chapter, we begin by proposing a generalized definition for ideal NTFs, and provide a method of constructing them. We then use this construction method to show that CT-ΔΣ modulators with an ELD path implemented in CT can only be ideal if its NTF is symmetric about $f_s/4$. Furthermore, we show how to construct an NTF that, while non-ideal, can be used with any modulator with CT-based ELD compensation, without modification. The Multi-Band NTF in Chapter 5 is constructed with this

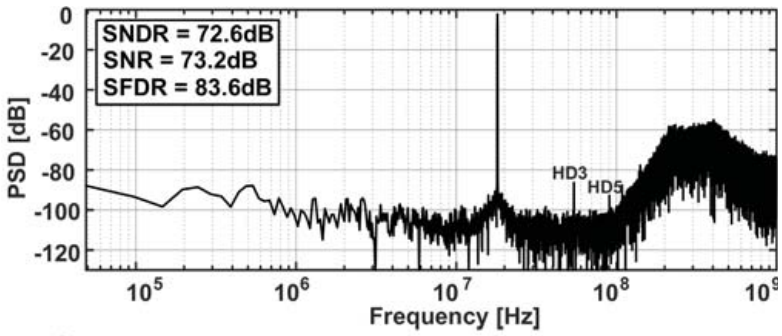


Figure 4.1: *The output spectrum from [2]. The hill in the noise shaping around 500MHz-1GHz and valley above 1GHz frequency noise shaping is non-ideal.*

limitation in mind.

4.1 Fundamental Requirements for NTFs

In order to use a given transfer function as an NTF, it must meet several fundamental requirements. [3] states that, for an NTF to be realizable, its transfer function must satisfy: $NTF(\infty) = ntf[0] = 1$ where $ntf[k]$ is the NTF's impulse response. This statement is equivalent to the requirement that: (1) $NTF(z)$ must have the same number of poles and zeros and (2) the gain of $H(z)$ must be 1, since this is the only set of conditions that makes $H(\infty) = 1$. This is to say that, if $H(z)$ is written in the following form:

$$\begin{aligned} H(z) &= \frac{H_{Num}(z)}{H_{Den}(z)} \\ &= \frac{b_M z^M + b_{M-1} z^{M-1} \cdots + b_0}{z^N + a_{M-1} z^{N-1} \cdots + a_0} \\ &= \frac{b_M (z - z_M)(z - z_{M-1}) \cdots (z - z_1)}{(z - p_N)(z - p_{N-1}) \cdots (z - p_1)} \end{aligned} \tag{4.1}$$

it must be the case that $M = N$ and $b_M = 1$. This requirement makes the NTF design process relatively straightforward: (1) choose a desired filter and (2) set the filter's gain to 1. However, there is one extra requirement when the NTF is used with fast-path ELD compensation.

As introduced in Chapter 3, ELD compensation makes it possible to equate a CT filter with any DT filter (in the impulse-invariant sense) when the DAC's impulse

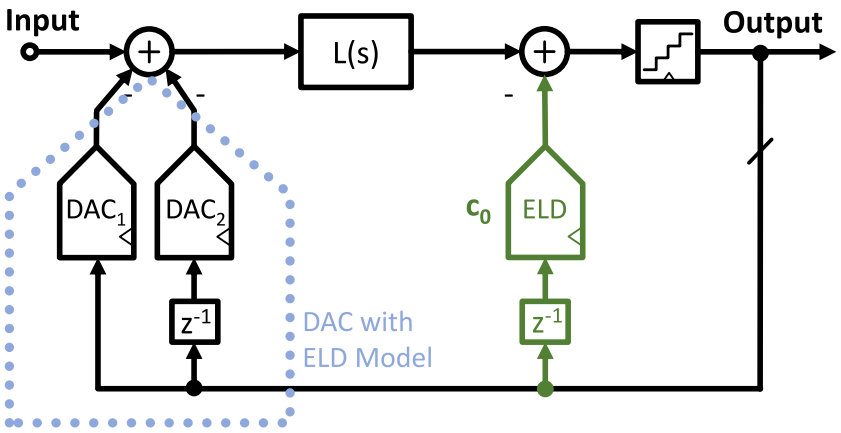


Figure 4.2: *Generic block diagram of a $\Delta\Sigma$-modulator, with fast-path and multi-cycle DAC model included.*

response extends into the next sample period. When this happens, we model the DAC as two separate DACs, shown in Fig. 4.2. DAC_1 represents the portion of the DAC's response that falls within the first sample period and DAC_2 represents its response in the second sample period.

The extra pole in the feedback path generated by DAC_2's delay adds a new zero at the origin, which can be seen by factoring the numerator and denominator of the NTF equation:

$$\begin{aligned} \text{NTF}(z) &= \frac{1}{1 + L_1(z) + z^{-1}L_2(z) + z^{-1}c_0} \\ &= \frac{zL_{Den}}{(z + c_0)L_{Den} + zL_{1,Num}(z) + L_{2,Num}(z)} \end{aligned} \tag{4.2}$$

where $L_{1,Num}(z)$ is the numerator of $L_1(z)$ and $L_{Den}(z)$ is the denominator of both $L_1(z)$ and $L_2(z)$.

The addition of ELD in the DAC adds an extra pole to the NTF, but without increasing the number of zeros that can be placed—there is effectively one extra pole that must be placed with no zero to counteract it. As will be shown in Section 4.4, ideal NTFs must have an equal number of poles and zeros, so this extra pole must always be placed at the origin, canceling the zero. However, as the remainder of this section shows, most NTFs with CT loop filters are unable to place this pole at the origin, meaning that any CT-$\Delta\Sigma$ with CT-based ELD compensation must have a

non-ideal NTF.

4.2 Requirements for Ideal NTFs

A lot of work has been done to optimize NTFs [3], primarily focused on finding robust numerical optimization methods.

In contrast to previous optimization efforts, this work focuses on NTFs with a 20N dB/decade slope in the transition band, where N is the order of the loop filter. "Optimal" NTFs often come with higher transition slopes, depending on how the optimizer's cost function is chosen. Higher slopes require placing NTF poles closer to the unit circle. Since, in this case, the poles don't have to travel as far to become unstable, these modulators tend to be more sensitive to small errors in the loop filter due to effects such as mismatch and passive component variation. Furthermore, it is typically considered a standard characteristic of an N^{th} order NTF to have a 20N dB/decade slope. For these reasons, we define an "ideal" NTF (as opposed to a properly "optimal" NTF) to be one with a transition slope of exactly 20N dB/decade.

This section demonstrates that, for an N^{th} order NTF with transition-band slope of 20N dB/decade, any ideal NTF must have a corresponding CT transfer function (its "prototype," discussed in Section 4.3) that can be written in the form $H(s) = H_0 - \hat{H}(s)$ where H_0 is a scalar term defining the maximum gain of the NTF and $\hat{H}(s)$ is an all-pole transfer function which defines the in-band behavior of the NTF. As a result, the NTF must have an equal number of independent poles and zeros.

4.2.1 Conditions for Ideality

In order to make comparisons between modulators, there should be a measure of their performance independent of any particular choice in specification. Bode's Sensitivity Integral provides a rather effective method—when applied to $\Delta\Sigma$ modulators, it states that the integral of the NTF's magnitude, in dB, must be zero:

$$\int_{\omega} \log |\mathrm{NTF}(z)| dz = 0 \tag{4.3}$$

where the integration path, ω, is the unit circle. This implies that the integral of the in-band (ib) section(s) of the NTF, in dB, must be equal to the integral of the out-of-band (oob) section(s):

$$\mathcal{N}_B = \int_{oob} log \left|\mathrm{NTF}(e^{j\omega})\right| d\omega = -\int_{ib} log \left|\mathrm{NTF}(e^{j\omega})\right| d\omega \tag{4.4}$$

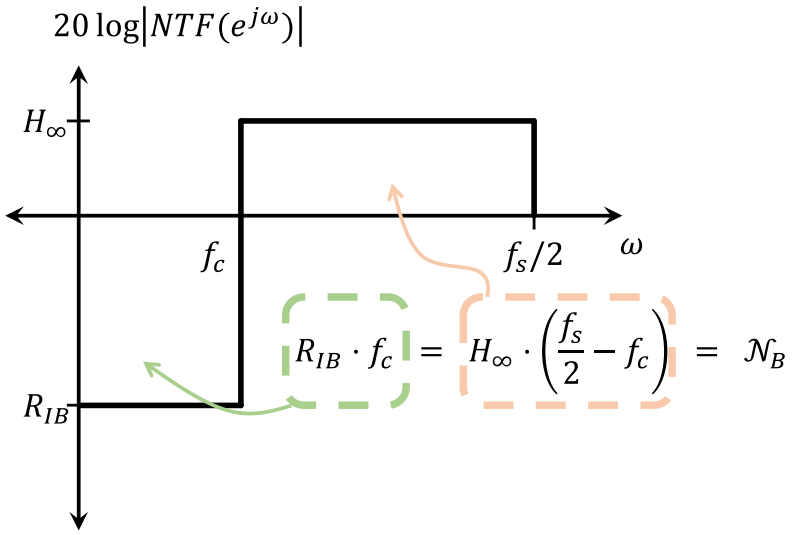


Figure 4.3: *An ideal, brick-wall NTF. The out-of-band gain is exactly equal to the Lee limit, H_∞, and the integral of the out-of-band region (in dB) is equal to the integral of the in-band region (in dB).*

By considering either side of this equation in isolation, we have a quantity that measures "how much" noise is being removed from the quantizer in question. In this work, we call this quantity the Noise Budget, and it is written as $\mathcal{N}_B$ in Eq. (4.4). An ideal NTF, therefore, maximizes $\mathcal{N}_B$ for a given set of in-band specifications.

Additionally, Lee's limit requires that [38], for the modulator to be stable, $||\text{NTF}(z)||_\infty$ (the maximum gain at any frequency) must be less than some maximum value, typically denoted as H_∞.

Taking these two requirements together, the maximum possible noise budget, for a given bandwidth, is given by the brick wall filter in Fig. 4.3. Because this NTF's out-of-band gain is equal to the Lee limit at all frequencies, its integral is maximized. Clearly, then, in order to maximize the noise budget for a finite-order NTF, that NTF should approach the Lee limit as quickly as is practical—20N dB/decade—and stay there.

4.3 Ideal NTFs by Construction

This section demostrates a way to create ideal NTFs. It does not represent a typical filter-first design approach, which would start by determining the filter architecture and calculating an NTF based on that architecture. Instead, this section focuses on creating the best possible NTF (according to our definition of 'ideal'), regardless of filter architecture.

The derivation of DT IIR filters typically begins in the CT domain and is then transformed into DT. Starting in the CT domain makes defining the filter shapes much more practical since the transfer function is evaluated along the imaginary axis, rather than the unit circle (as in: $s = j\omega$ vs. $z = e^{j\omega}$). The downside is that there does not exist a transform that perfectly maps the CT domain into DT since frequencies in CT can continue indefinitely, whereas frequencies in DT are limited by the sample-rate. Increasing the frequency beyond $f_s/2$ in DT causes the transfer function to repeat, since $e^{j\omega}$ continues around the unit circle and back to its starting point over and over again with increasing ω.

$$s = \frac{2(1 - z^{-1})}{T_s(1 + z^{-1})} \tag{4.5}$$

The Bilinear Transform is the result of replacing each s in the CT transfer function with a function of the DT z variable, shown in Eq. (4.5). It is the most common method used for generating DT filters from CT prototypes because it guarantees that the original filter specifications remain met after transformation. That is to say, if a stop-band of 60dB and Bandwidth (BW) of 1Hz is specified, those specifications will be unchanged after the transformation. However, it does so by warping the overall frequency response. In essence, it maps the entire $j\omega$ axis directly onto the unit circle by compressing the CT frequencies closer and closer together as frequency increases. This mapping results in the warped transfer function shown in Fig. 4.4(bot).In the figure, an example CT prototype with a low Oversample Rate (OSR) is shown along with the result of the Bilinear Transform of that prototype. It is clear that, although the original specifications are met for both (note that the stop-band specification is the difference between the pass-band gain and the stop-band gain), the DT transfer function has been considerably warped compared to its origin. This warping gets worse with low OSR NTFs and is unacceptable for our purposes since it does not result in an NTF with a consistent transition band. Note that the response is warped in high OSR NTFs as well, but the error is relatively small, as seen in Fig. 4.5(top).

The Matched-Z Transform (MZT), however, does not warp the frequency spec-

trum the way the bilinear transform does. To apply the MZT transform, all poles and zeros in the CT filter are mapped into DT with the complex exponential. That is:

$$\begin{aligned} H_c(s) &= k_c \frac{(s-z_1)(s-z_2)\cdots}{(s-p_1)(s-p_2)\cdots} \\ &\Downarrow \\ H_d(z) &= k_d \frac{(z-e^{z_1})(z-e^{z_2})\cdots}{(z-e^{p_1})(z-e^{p_2})\cdots} \end{aligned} \tag{4.6}$$

where k_d is chosen so that the gain at $z=-1$ is equal to the gain at $s=\infty$, k_c, which can be written as:

$$k_d = k_c \frac{(-1-e^{p_1})(-1-e^{p_2})\cdots}{(-1-e^{z_1})(-1-e^{z_2})\cdots} \tag{4.7}$$

Rather than warping the frequency response, as in the Bilinear case, the MZT transform introduces aliasing by mapping the $j\omega$ axis onto the unit circle linearly, with high-frequencies continuing to wrap around the unit circle as many times as necessary. As a result of this continuous wrapping, any behavior in the CT filter's transfer function beyond $f_s/2$ will get aliased down and added to the frequency response in the Nyquist band. This means that, if the NTF's out-of-band gain changes significantly above $f_s/2$, the in-band and transition-band characteristic may not meet the original specifications, shown in Fig. 4.4(bot). This is not surprising, though; it would be impossible to, for example, construct an NTF with a 20N dB/decade slope directly in DT if that slope does not reach its maximum before $f_s/2$.

If the CT transfer function is constant above $f_s/2$, however, the only thing that is aliased is a constant value overall frequencies, which gets removed when the overall gain of the transfer function is normalized to match that of the CT prototype. Therefore, applying the MZT transform when the NTF is constant above the Nyquist frequency gives a good match between the CT prototype and the final NTF, shown in Fig. 4.4(top). As a result, for the rest of this chapter, we will use the MZT transform with the assumption that the CT prototype is essentially constant above $f_s/2$.

Finally, notice that, in Fig. 4.4(bot), because the Bilinear Transform's original specification remains the same (the cutoff frequency is 1/OSR, and the difference between in-band and out-of-band gain is 80dB), it is tempting to think there is a very large in-band benefit to using it over MZT. However, this is an artifact of its much larger H_∞, which would likely cross the quantizer's Lee limit, making it unstable. If the stop-band rejection for the Bilinear Transform case is reduced so

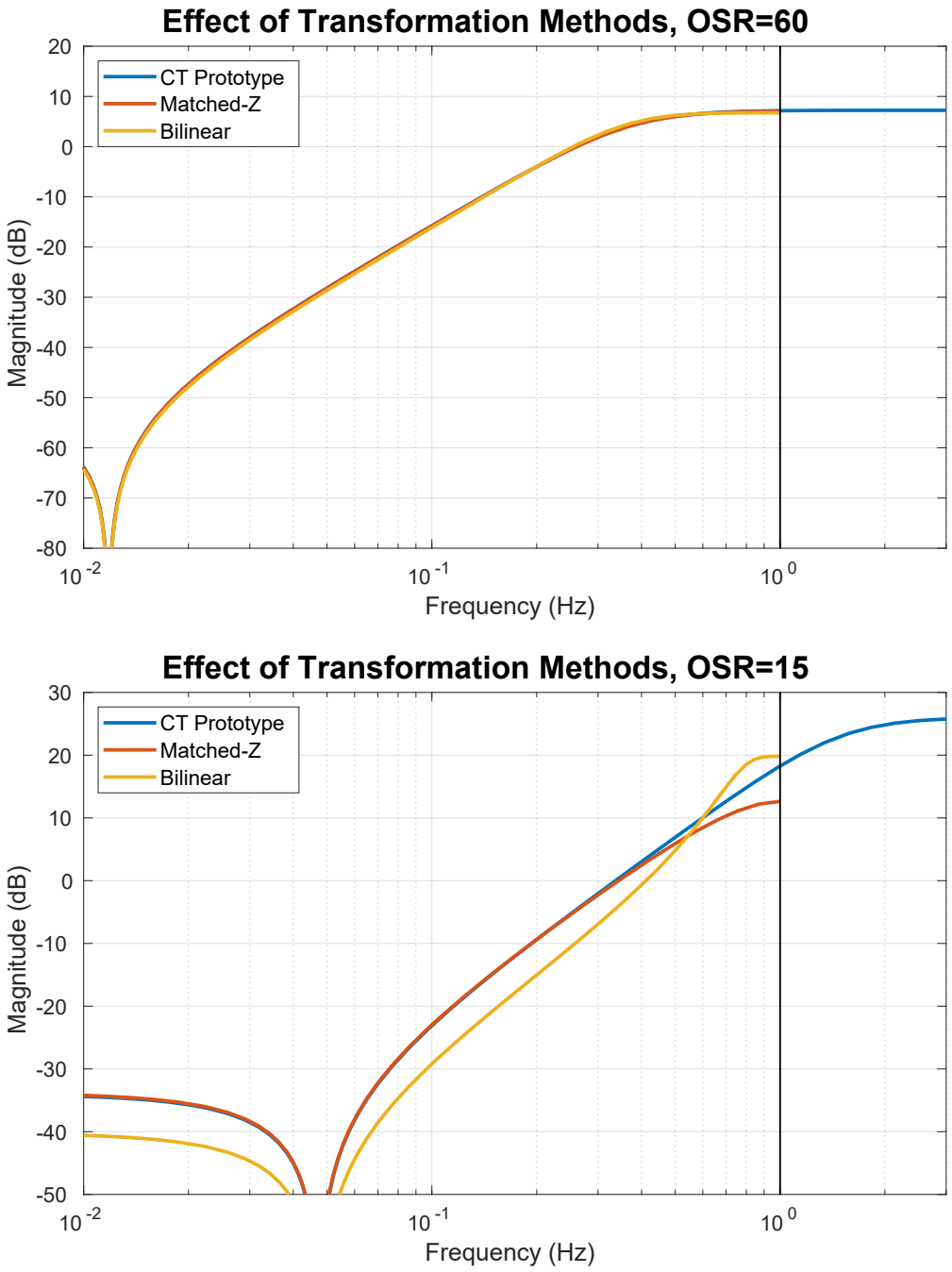


Figure 4.4: *The effects of applying different CT-DT transformations to the same CT prototype.* ***(top)*** *A case when the OSR is high enough that $f_s/2$ (black line) falls well within the out-of-band region of the CT prototype.* ***(bot)*** *A case when the OSR is low, causing $f_s/2$ to fall within the transition band of the CT prototype. The prototype shown is used for both MZT and Bilinear transforms. The sampling frequency is normalized so that the Nyquist frequency is at 1Hz (indicated by the vertical black line.)*

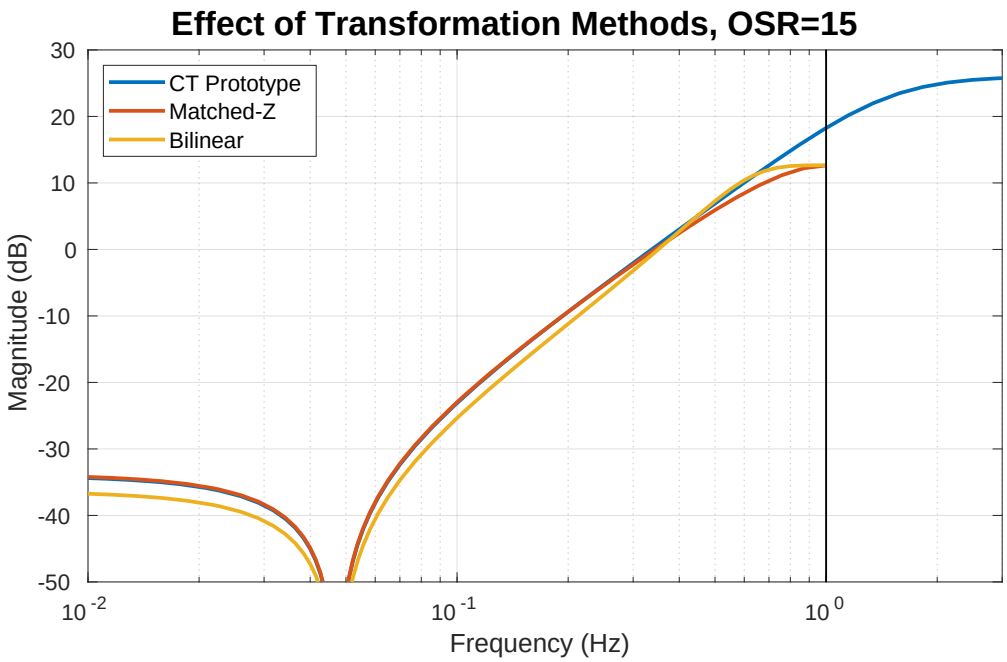


Figure 4.5: *Same conditions as in Fig. 4.4(bot), but with the stop-band-rejection of the Bilinear Transform reduced so that the max out-of-band gains match. The CT prototype shown is used for the MZT Transform, whereas the prototype for the Bilinear transform is the same shape with the stop-band decreased.*

that its H_∞ matches that of the MZT, shown in Fig. 4.5, the in-band benefit reduces to approximately 2.5dB. This benefit is due to the sharper rise in the out-of-band gain and does not seem to increase at lower OSRs.

4.3.1 Constructing the DT Inverse Chebyshev Filter

The filter construction method used in this work is generalized from the method used in [39] to derive the Inverse Chebyshev Filter (also known as Chebyshev Type-II). This is a common NTF choice since it results in the in-band region being as flat as possible with all its zeros on the unit circle. This section describes the relevant parts of the derivation and concludes with a brief investigation into its properties, which reveals that the zero locations are quite close to the optimal locations described in [3].

First note that, throughout this derivation, the magnitude-squared function is used as a stand-in for the magnitude. This is just an easier way of working around the otherwise difficult absolute value operator (since it is inside the square, we can remove it without changing the result). To find the final equation, we simply choose half of the poles and zeros, which will be discussed in more detail later. Secondly, we will work with the real parameter ω, rather than the usual s term to reflect the original derivation. Substituting $\omega = -js$ gives the final transfer function.

Now, the Inverse Chebyshev filter is built from the all-pole Chebyshev Filter (Type-I), which can be written as:

$$|\hat{H}|^2(\omega) = \frac{H_0^2}{1 + \epsilon^2 C_N^2(\omega)} \tag{4.8}$$

where ϵ is a scalar constant and $C_N(\omega)$ is the N^{th} order Chebyshev polynomial of the first kind. $C_N(\omega)$ is defined recursively as:

$$\begin{aligned} C_0(x) &= 1 \\ C_1(x) &= x \\ C_{N+1}(x) &= 2xC_N(x) - C_{N-1} \end{aligned} \tag{4.9}$$

For example, the first five Chebyshev polynomials are:

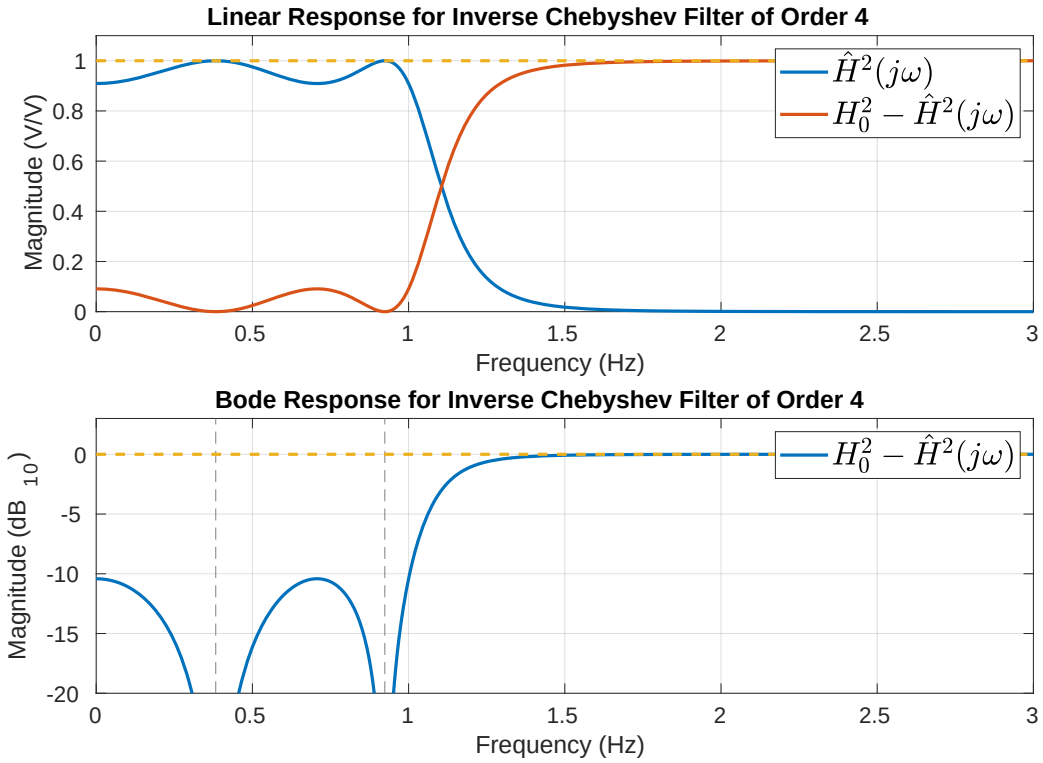


Figure 4.6: *The construction of an Inverse-Chebyshev filter.* ***(top)*** *A linear view of the magnitude squared for both the Chebyshev and Inverse Chebyshev; $H_0 = 1$ and $\epsilon = \sqrt{0.1}$.* ***(bot)*** *The Inverse Chebyshev magnitude response, in dB.*

$$\begin{aligned} C_1 &= x \\ C_2 &= 2x^2 - 1 \\ C_3 &= 4x^3 - 3x \\ C_4 &= 8x^4 - 8x^2 + 1 \\ C_5 &= 16x^5 - 20x^3 + 5x \end{aligned} \tag{4.10}$$

This filter uses the Chebyshev Polynomial to create a ripple in the pass-band, whose height is defined by ϵ, and which decays as frequency increases. An example of such a filter is shown in Fig. 4.6(top). Because $\hat{H}$ only contains poles, the high-frequency decay has a slope of 20N dB/decade. When viewed on a linear scale, rather than dB, it is clear that the ripple in the squared magnitude of this function touches H_0^2 N times, one for each ripple. Therefore, when the function is subtracted from H_0^2, those points become zeros, and the size of the ripples define the maximum stop-band

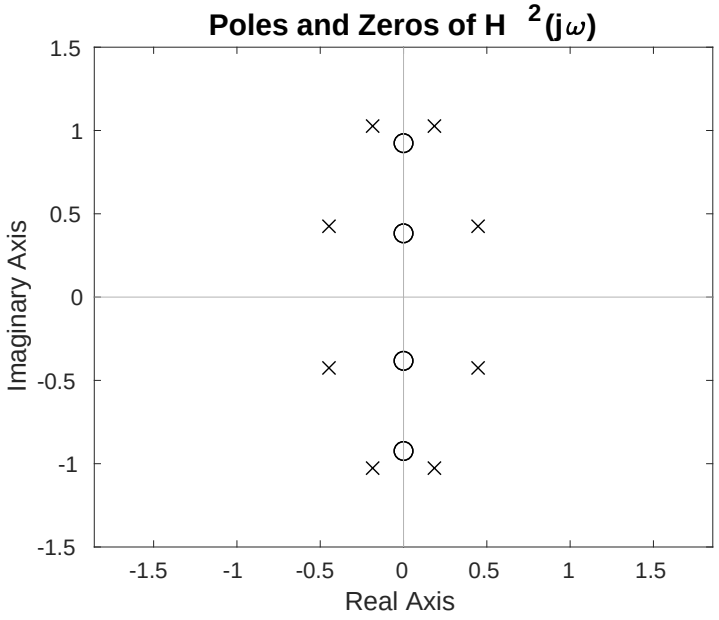


Figure 4.7: *The poles and zeros of* $H^2(\omega)$.

gain of the resulting function, $H^2(\omega)$:

$$\begin{aligned} |H(\omega)|^2 &= H_0^2 - \hat{H}(\omega) \\ &= H_0^2 - \frac{H_0^2}{1+\epsilon^2 C_N^2(\omega)} \\ &= \frac{H_0^2 \epsilon^2 C_N^2(s)}{1+\epsilon^2 C_N^2(\omega)} \end{aligned} \tag{4.11}$$

Furthermore, since $\hat{H}^2(\omega)$ approaches zero at a rate of 20N dB/decade at high-frequency, $H^2(\omega)$ must approach H_0^2 at the same rate. $H^2(\omega)$, therefore, has the well-known property of the Inverse Chebyshev Filter that its pass-band is maximally flat and is approached at 20N dB/decade [39].

To complete the construction, we first have to put the function in the s-domain by replacing ω with $-js$ (this just swaps the real and imaginary axes). Finally, we can take the square root of $|H(s)|^2$ by using only the poles and zeros on the Left-Half-Plane (LHP), to ensure stability. The poles and zeros of an example $H^2(s)$ is shown in Fig. 4.7. The zeros on the imaginary axis are repeated, so we take one of each, along with all of the poles to the left of the imaginary axis. The final transfer function, $H(s)$, is the combination of the chosen poles and zeros.

Since this filter prototype meets all the criteria for the ideal NTF defined in Section 4.2 and applying the Matched-Z Transform does not change the shape of the

Order	% Error (each pair)
2	22.5
3	0, 11.8
4	7.3, 12.6
5	0, 5.0, 7.0
6	3.6, 6.9, 8.5
7	0, 2.7, 5.4, 6.9
8	4.4, 2.1, 5.7, 6.4

Table 4.1: *The percent error between the optimal zero locations given by [3] and the zero locations given by the Inverse Chebyshev filer. Percent error is defined as* $\frac{|z_{optimal}-z_{cheby}|}{|z_{optimal}|}$.

filter (assuming the out-of-band gain settles before $f_s/2$), we conclude that applying the MZT transform (Eq. (4.6)) to this filter prototype yields an ideal NTF whose zeros are spread by the Chebyshev polynomials.

Finally, it is worth pointing out that the zeros given by the Chebyshev polynomials are similar to, but not quite the same as, the optimal zeros given in [3]. Chebyshev zeros guarantee that all the in-band lobes are at the same height, whereas the optimal zeros only guarantees minimal total in-band noise. However, comparing the optimal zero frequencies with the Chebyshev zero frequencies shows (Table 4.1) that the percent error decreases as order increases. In fact, with the exception of the 2_{nd} order case, the error between the two is never more than 12.6%.

Of course, the "optimal" zeros are only optimal when the Signal-to-Quantization Noise (SQNR) is significantly higher than the thermal noise of the implementation. If, as is more often the case, the SNDR is limited by circuit noise, the extra noise suppression at low frequencies wastes part of the Noise Budget since it increases the in-band integral. The preferred choice will depend on the application.

The closed-form equation for the CT Chebyshev zeros are given in [39] and can be written in DT as follows:

$$z_{z,k} = e^{j(\pi/\text{OSR})/sec(k\pi/2N)}, \qquad k = 1, 3, \ldots, 2N-1 \tag{4.12}$$

where ω_c is the CT prototype's stop-band width. Likewise, the pole locations are:

$$z_{p,k} = e^{(\sigma_k + j\omega_k)(\pi/\text{OSR})}, \qquad k = 1, 2, \ldots, N \tag{4.13}$$

where:

$$\sigma_k = -\sinh(a)\sin\left(\frac{(2k-1)\pi}{2N}\right)$$
$$\omega_k = \cosh(a)\cos\left(\frac{(2k-1)\pi}{2N}\right)$$
$$a = \frac{1}{N}\sinh^{-1}\frac{1}{\epsilon}$$

The NTF's H_∞ is:

$$||\text{NTF}(e^{j\omega})||_\infty = \frac{(-1-z_{z,1})(-1-z_{z,2})\cdots(-1-z_{z,N})}{(-1-z_{p,1})(-1-z_{p,2})\cdots(-1-z_{p,N})} \tag{4.14}$$

4.3.2 Generalizing an NTF Construction Method

Although, to the authors' knowledge, the previous method is not typically applied to other filter types (instead, frequency transformations are typically used to create high-pass filters, for example), there is no reason it can't be applied to any in-band shape we would like.

Rather than starting with the all-pole Chebyshev filter, we might choose, for example, to start with the Butterworth filter. This filter also only contains poles, so the result will be ideal in the sense described in Section 4.2, and is defined as:

$$\hat{H}^2(\omega) = \frac{H_0^2}{1+\omega^{2N}} \tag{4.15}$$

Applying the method from the previous section, subtracting H_0^2 gives the following:

$$H^2(\omega) = \frac{H_0^2\omega^{2N}}{1+\omega^{2N}} \tag{4.16}$$

Notice that all of this function's zeros are at the origin. That is, using the Butterworth filter as a prototype gives the usual ideal NTF for modulators without optimized zeros. In fact, by choosing a 2$^{\text{nd}}$ order prototype filter and setting the Nyquist frequency to 2.45x the Butterworth prototype's cutoff frequency, we get the same answer as the optimal NTF given on pg. 81 of [3]! The referenced NTF was arrived at after an exhaustive search of the possible NTFs for 2$^{\text{nd}}$ order modulators, maximizing SQNR in the process. The plot of this NTF is shown in Fig. 4.8. From our analysis so far, we have a potential explanation for this optimal point: it is likely the highest cutoff frequency for which the NTF still gets sufficiently close to its maximum value before $f_s/2$. If the cutoff frequency increased any more, the NTF's H_∞ would follow

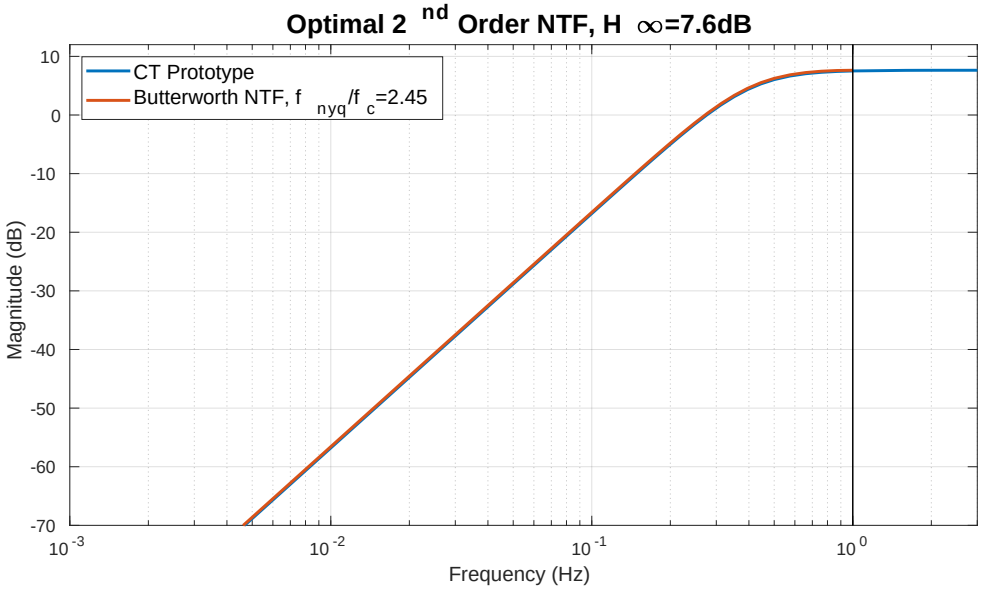


Figure 4.8: *The Butterworth NTF with cutoff frequency defined by the ratio $f_{nyq}/f_c = 2.45$, with $f_{nyq} = f_s/2$. This NTF has the same coefficients as the optimal NTF given in [3].*

the NTF's gain down, thereby decreasing its noise budget. This is only a conjecture, of course, but it seems to fit with the magnitude response in Fig. 4.8. Future work should investigate whether this can be extrapolated to higher-order NTFs.

In general, we can choose any all-pole prototype filter with a desired in-band shape and, because all-pole filters decay at high-frequency with 20N dB/decade, we can use this method to determine the ideal NTF for said in-band shape. For example, modifying the Chebyshev prototype slightly by multiplying H_0 by some factor, $0 < a^2 < 1$, gives an in-band shape that no longer fully extends to zero:

$$\hat{H}^2(\omega) = \frac{H_0^2(1-a^2) + H_0^2\epsilon^2 C_N^2(\omega)}{1 + \epsilon^2 C_N^2(\omega)} \tag{4.17}$$

Making the pass-band ripple small gives a relatively flat in-band shape, shown in Fig. 4.9. NTF's zeros, and therefore the CT loop filter's poles, are moved away from the imaginary axis, potentially making the CT loop filter less sensitive to small errors in the loop filter's transfer function due to process variation, since this sensitivity is usually proportional to the filter's Q [39]. Furthermore, if the in-band quantization noise floor is chosen to be below the thermal noise floor of the modulator, performance would not be sacrificed.

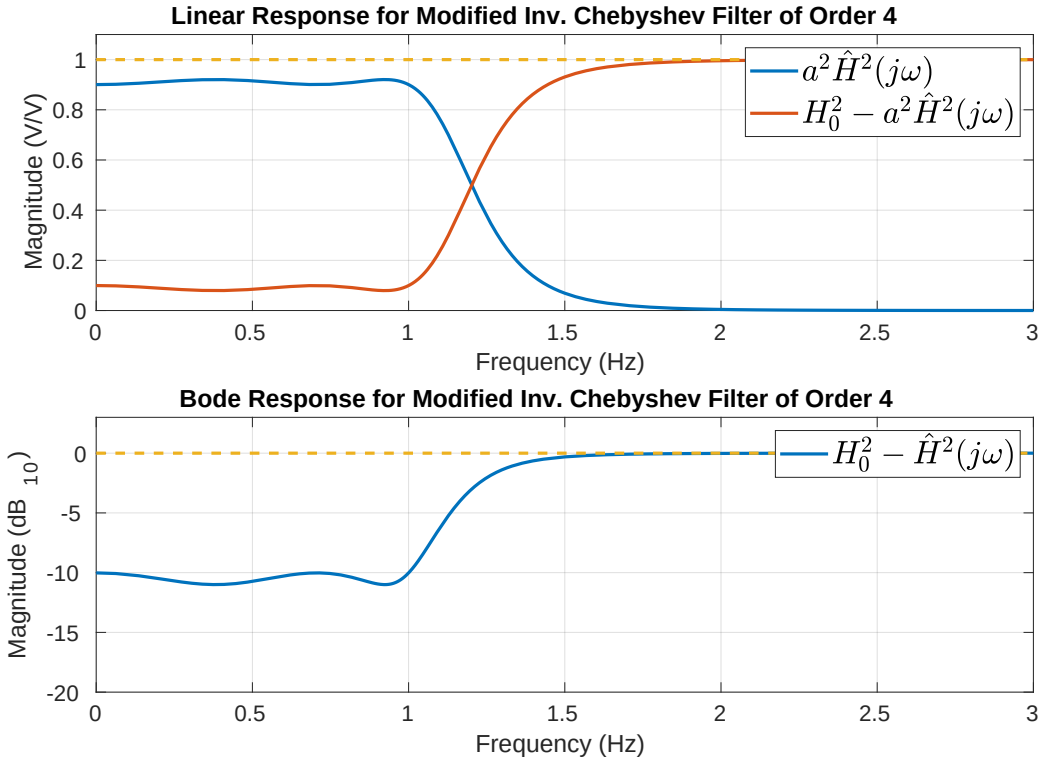


Figure 4.9: *Magnitude plots for a proposed alternative to standard Chebyshev NTFs. The NTF's zeros are pulled off the unit circle, giving a flatter in-band region and loop filter poles that are potentially more robust to component variation.*

4.4 A Limitation on CT-ΔΣ NTFs

Finally, with the ideal NTF well-defined, we illustrate that, a system with an ELD compensation path implemented in CT cannot have an ideal NTF, due to the finite BW of CT systems. In particular, we show that the bandwidth restriction requires that the poles' center of mass must be the same as the zeros' center of mass. That is: $\frac{1}{N}\sum p_i = \frac{1}{N}\sum z_i$, where p_i is the location of the i$^{\text{th}}$ pole, z_i is the location of the i$^{\text{th}}$ zero, and N the order of the NTF. So, if the zeros are not symmetric about the imaginary axis, then the poles cannot be placed so that the NTF is ideal.

Because of this restriction, the ideal NTFs of Section 4.3 can only be implemented by bandpass modulators centered on $f_s/4$. In fact, the performance degradation gets worse the more asymmetrical the NTF is about the imaginary axis, meaning that the performance of base-band modulators tends to suffer the most. The result is typically some form of peaking in the out-of-band gain.

4.4.1 Excess CT Poles

Real CT systems will always have limited bandwidth—at a minimum, parasitic capacitors cause the transfer function to roll-off at high-frequencies. This is the same as saying its transfer function must have more poles than zeros; after all, poles contribute -20dB/decade roll-off and zeros +20dB/decade, so there must be at least one extra pole to have a negative slope at frequencies higher than the highest pole or zero. Typically, it is enough to assume that, when these excess poles are at high enough frequency, they have no significant effect on the system being analyzed. However, the sampling operation moves all the poles of the CT system to the unit circle through the complex exponential function. The real part of a given CT pole becomes the magnitude of the DT pole, since the real and imaginary parts of the exponential can be separated into magnitude and phase components—$e^{\lambda} = e^{\sigma}e^{j\rho}$ for $\lambda = \sigma + j\rho$. As the CT pole's real part, σ, becomes more and more negative, the DT pole's magnitude, e^{σ}, becomes closer and closer to zero. Thus, even for a pole at infinitely high-frequency, its corresponding DT pole does not disappear, and instead remains very close to the origin. As a consequence, the root locus of such a loop filter will remain largely unchanged from one with a pole at a much lower frequency. Another way to think of it is that, as the real part of the high-frequency pole moves to higher and higher frequency (e.g., a secondary amplifier pole), it becomes closer and closer to a delay in DT—extremely destabilizing if not accounted for.

As discussed in Section 2.2.2, the transfer function of a state-space system with a

non-zero D matrix will have an equal number of poles and zeros. Likewise, a system with $D = 0$ is guaranteed to have at least one extra pole. The CT-DT conversion equations (Eq. (3.18) and Eq. (3.17) in Chapter 3) make it clear that, if $D_c = 0$, then $D_d = 0$ as well. So then, if the CT system has more poles than zeros (discussed in Section 2.2.2), its DT equivalent must as well.

Clearly, not all NTFs will produce a loop filter of this form. Solving the NTF's transfer function (Eq. (4.2)) for the loop filter gives:

$$zL_1(z) + L_2(z) + c_0 = \frac{z\mathrm{NTF}_{den}(z) - z\mathrm{NTF}_{num}(z)}{\mathrm{NTF}_{num}(z)} \tag{4.18}$$

For there to be one less zero on the right-hand side, the first two coefficients in the NTF's numerator and denominator must be equal, since the z term from DAC_2's delay increases the order of the numerator on the RHS and the order of the NTF's numerator is the same as the order of its denominator. The first coefficient is the same as in Section 4.1: $b_m = 1$. The second coefficient, however, can be related to the NTF's poles and zeros by using the first of Vieta's formulas, which can be written as:

$$\begin{aligned} -b_{M-1}/b_M &= \sum_{i=1}^{M} z_i \\ -a_{N-1}/a_N &= \sum_{i=1}^{N} p_i \end{aligned} \tag{4.19}$$

Since $b_N = a_N = 1$, the requirement that $b_{N-1} = a_{N-1}$ implies that:

$$\sum_{i=1}^{N} z_i = \sum_{j=1}^{N} p_j \tag{4.20}$$

Dividing both sides by N gives the relationship described in the beginning of this section: the zeros' center-of-mass must be equal to the poles' center-of-mass.

4.4.2 Symmetric NTFs

It can easily be seen that any NTF whose poles and zeros are symmetric about the imaginary axis solves this equation with both sides equal to 0. This is an important observation for bandpass modulators, as it implies that the optimal bandpass modulator is centered at fs/4 since, in this case, the NTF is symmetric about the imaginary axis.

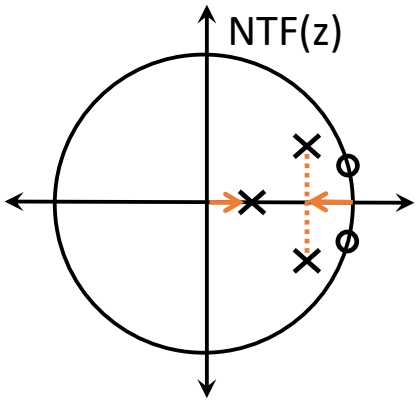


Figure 4.10: *Movement of extra pole at the origin to compensate for moving the primary poles away from the zeros in a purely CT NTF.*

4.4.3 Non-Symmetric NTFs

For non-symmetric NTFs, however, the story is different. Consider a lowpass NTF with all its zeros near $z = 1$. In order to satisfy this equation with an equal number of poles and zeros, all the poles would have to be near $z = 1$ as well (or else some would be made unstable to balance the center-of-mass), nearly canceling out the in-band rejection of the zeros. The only way to move them away from the zeros—and thus achieve some amount of noise shaping—is to move the pole at the origin to the right. This action is shown for a simple 2^{nd} order example in Fig. 4.10.

As discussed in Section 4.2, since an ideal NTF should approach a constant in its out-of-band section, its prototype must also approach a constant—thus, it must have an equal number of poles and zeros. However, in modulators with ELD, one zero is fixed at the origin, meaning that one pole should be placed at the origin to cancel it out and ensure there are an equal number of poles and zeros contributing to the NTF's magnitude. However, as we just discussed, when the loop filter is fully implemented in CT, this pole cannot, in general, be placed at the origin, meaning that its CT prototype must have one extra pole. Therefore, it will roll off at high-frequency instead of approaching a constant.

An example solution that attempts to make a flat out-of-band section for as wide a band as possible is shown in Fig. 4.11. This shape seems to be common in the literature [2][36][37]. Comparing it to Fig. 4.1, for example, the two are qualitatively similar.

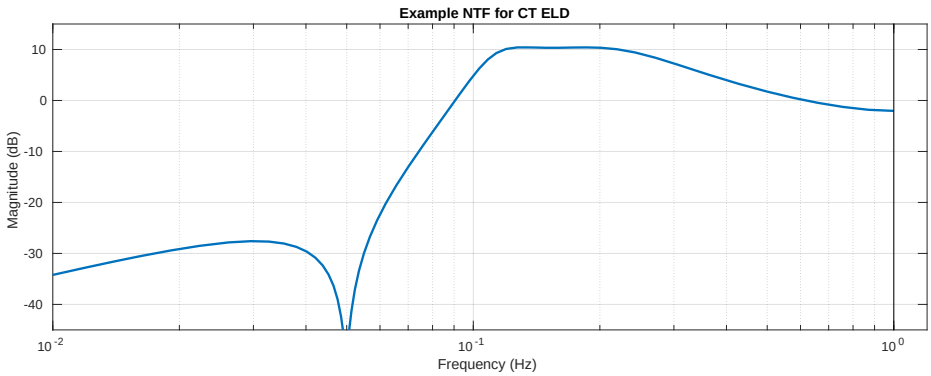


Figure 4.11: *Example NTF for CT-limited modulator.*

4.5 Recovering Ideal Performance

The performance limitation discussed here hinges on the bandwidth limitation of CT systems. The only way to get around this is to implement the infinite-bandwidth term, c_0, in DT—separately from the CT loop filter. One possible solution to this is the Digital ELD Compensation technique, where the ELD path is moved from the input of the quantizer to the output, allowing it to be implemented in DT [27]. This approach has its drawbacks, however. In particular, it is well-known that the amplitude at the quantizer's input increases, potentially reducing the maximum signal amplitude.

Another possible approach is to use a sample-and-hold at the input of the quantizer, with the sampling capacitor implemented as a CDAC. The feedback path can then be implemented in DT by connecting the top plates of the DAC's capacitors to +VREF or -VREF, according to the digital value of the feedback path. The output of the loop filter is then sampled on the bottom plate. By applying both signals at the same time, the subtraction required for the ELD path is created and applied in DT. The ELD path in [40] is an example of this approach. To improve on this, the method in [4] can be used to reduce the maximum amplitude of the loop filter, in contrast to digital ELD. A simple example of such an ELD method is shown in Fig. 4.12.

Other approaches are possible. [4] is an example of a fully hybrid CT-DT ADC by using a noise-shaping SAR for its quantizer. Since the input to the quantizer in this case is a sample-and-hold, the same method as before is applied to the ELD path, removing the restriction on its NTF. Additionally, [41] implements the ELD path by varying the voltage references of its flash ADC.

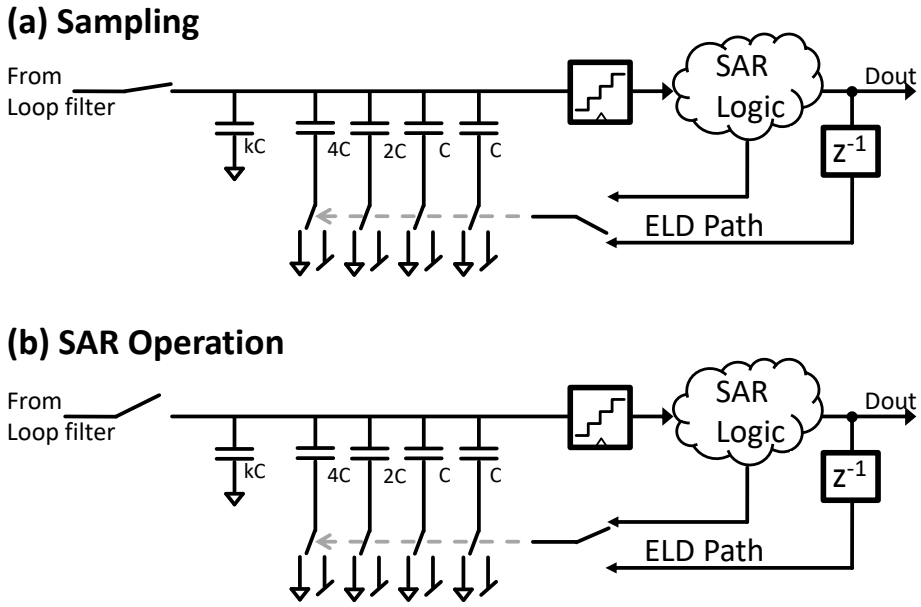


Figure 4.12: *A simple example of an ELD solution using a quantizer with a sample-and-hold. In this case, the quantizer is a SAR ADC. It also includes a method from [4] to reduce the maximum amplitude of the loop filter, implemented as an attenuation cap of size kC.*

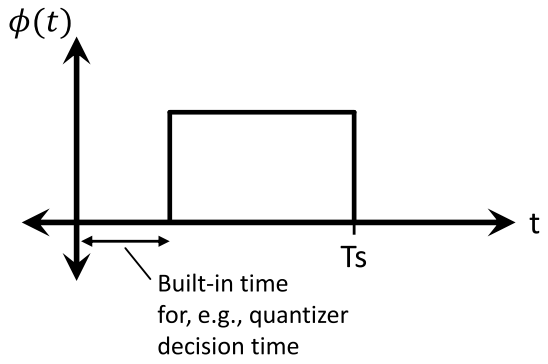


Figure 4.13: *An example of a DAC impulse response that includes time for delay in the loop, such as from non-zero quantizer decision time or delay in intermediate logic.*

4.6 Excess Loop Delay: An Aside

It is important to emphasize that this limitation only applies to modulators that use fast-path ELD compensation. As discussed in Chapter 3, this is only required when the DAC's impulse response extends into the second sample period. It is perfectly possible for the DAC to be designed to allow for any loop delay, such as the non-zero decision time of the quantizer. For example, an RZ DAC can be used where the start of the pulse is late enough in the sample period to allow the quantizer's signal to propagate to the DAC, and the end of the pulse occurs at or before T_s, shown in Fig. 4.13. In such a system, $c_0 = 0$ and so the NTF is not restricted which is especially useful for very low-speed ADCs.

The downside of using an RZ DAC in this way is that, as the amount of time the DAC is zero increases (as a percent of the sample period), the amplitude of the noise in the loop increases, which can reduce the maximum input amplitude. This noise increase happens because, when the DAC is zero, the amplitude of the error signal is equal to the full input signal, rather than being equal to the loop's estimation error. The more time the error signal is large, the more the loop filter will integrate this large error, producing larger internal signals.

Finally, the choice to use a DAC shape that accounts for loop delay without extending into the second sample period limits the quantizer choice to only the simplest ADCs such as a flash ADC, in order to minimize the decision time of the quantizer. When a higher-precision ADC is required, or the previously mentioned increase in

signal amplitude is unacceptable, implementing the relatively simple sample-and-hold discussed in Section 4.5 may be preferable.

CHAPTER 5

A Mutli-Band ADC

With the Continuous-Time to Discrete-Time (CT-DT) conversion tools of Chapter 3 and the NTF design tools of Chapter 4 in place, a widely applicable method of CT-$\Delta\Sigma$ ADC design can take shape. In this chapter, we discuss an approach to $\Delta\Sigma$ design that is flexible enough to make complex designs, such as the multi-band modulator presented here, more straightforward. By starting with the NTF and working linearly towards the loop filter, the process becomes automatable completely independently of parameters like loop order and filter type. We also discuss (1) a Feed-Forward (FF) synthesis method which simplifies the implementation of the high-order loop filter; (2) a modified single-amplifier biquad which simplifies the DAC design; (3) a modification to the input resistor network that reduces loop filter nonlinearity and inter-band distortion; and (4) a method for numerically compensating for non-ideal amplifier effects in the loop filter. Finally, we employ all these techniques to introduce the first Multi-Band $\Delta\Sigma$ modulator.

5.1 Multi-Band Architecture

We begin with an overview of the architecture. Fig. 5.1(a) shows a conceptual representation of a two-band $\Delta\Sigma$ modulator. In this work, the baseband (BB) is 3^{rd} order, and the bandpass (BP) is 6^{th} order, resulting in a 9^{th} order loop, overall. In the loop filter of Fig. 5.1(a), feed-forward (FF) paths bypass each band's sub-filter. This isolates the bands from one another, reducing the need for each band to pass the other's signals. Instead, the noise is passed through resistive FF paths, which do not have the linearity and bandwidth limitations of the sub-filters.

Fig. 5.2 shows the fully-realized prototype, in which each sub-filter is synthesized as a cascade of biquads using the method introduced in Section 5.3. Each filter block is a single biquad with 2 poles and 1 zero. Since summations are implemented

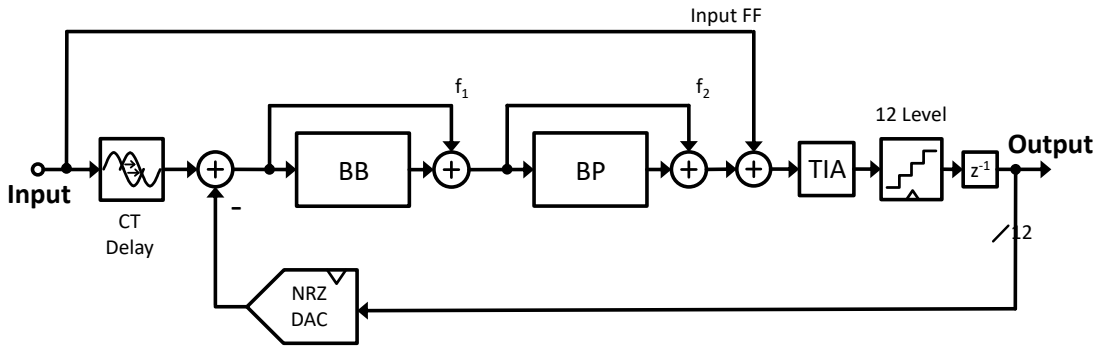


Figure 5.1: *Conceptual overview block diagram of the modulator. The BB and BP blocks are 3^{rd} and 6^{th} order filters, respectively.*

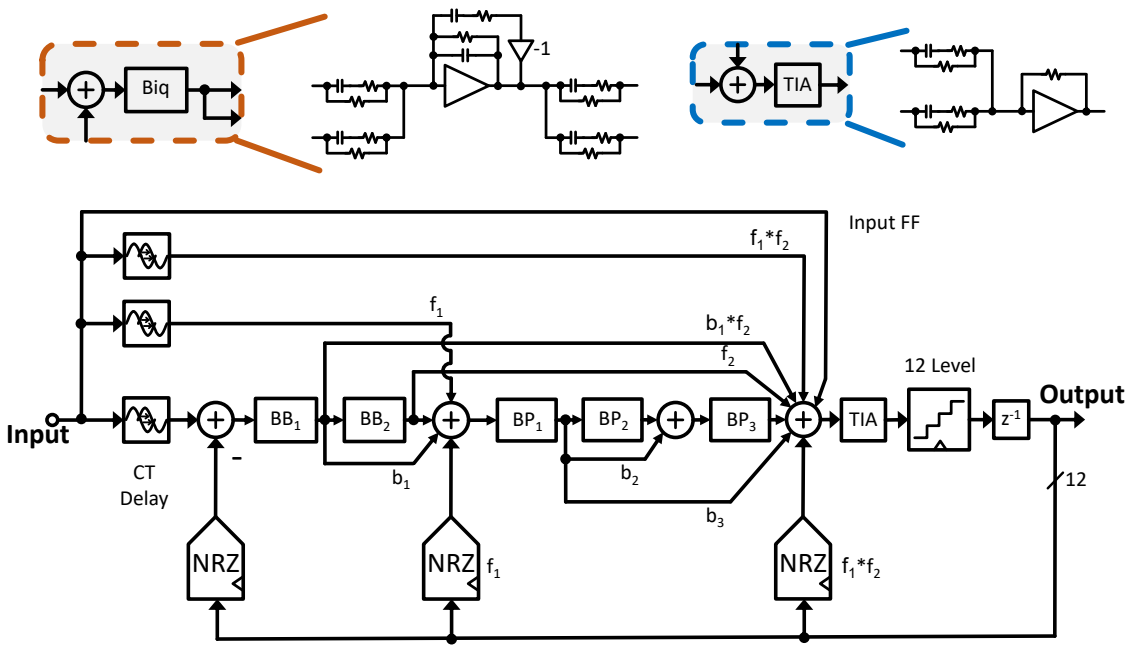


Figure 5.2: *Detailed block diagram of the modulator. The BB_1 block is an integrator, BB_2 and all BP blocks are single-amplifier biquads.*

in current-mode, with the FF resistors feeding into the following biquad's virtual ground, the output of these summations cannot be accessed directly to implement a FF. As a result, FF paths f1 and f2 must be split up into new FF paths to make an equivalent, implementable signal flow, shown with coefficients f2, f2*b1, and f1*f2 in Fig. 5.2. The NRZ DACs and quantizer are both 12-level, and the sampling rate is fixed at 2GHz. The final summation before the quantizer is implemented using a trans-impedance amplifier (Trans-Impedance Amplifier (TIA)). The CT delays are passive all-pass filters and are discussed in Section 5.5.

5.2 Multi-Band Modulator Design Process

There are three main principles underpinning the design of this multi-band modulator. (1) NTF-first design calculates the filter in terms of the NTF, as opposed to the more common filter-first approach in which the NTF is calculated in terms of the loop filter. This simplifies the process by allowing the DT-CT conversion to be applied directly to the desired NTF, whether algebraically or numerically, rather than requiring an algebraic solution linking the CT filter to the DT NTF. (2) The process should be as automated and NTF-/filter-agnostic as possible. This allows the same tools to be applied to any design, no matter the complexity or required architecture. (3) Loop filter non-idealities should be compensated for numerically, with the goal of making the final transfer function match the ideal transfer function as closely as possible. This allows more complex amplifier models to be used for compensation and provides guidance for final post-simulation optimization.

5.2.1 NTF Design

The design process begins by choosing the NTF to be implemented. The NTF used in this work is shown in Fig. 5.3(a) and (b). The three poles and zeros on top and bottom of (b) define the bandpass section, while the four poles and three zeros on the right define the baseband section.

Since there must be one extra pole, per the requirements for CT modulators in Chapter 4, it is natural to include the primary pole of the TIA in the NTF itself. The zero near the origin in Fig. 5.3(b) models this high-frequency pole, which is chosen to be 0.7 T_s. Note that, since it is being accounted for in this step, this pole does not have to be a large multiple of f_s; it is sufficient to place it far away from the baseband poles (in DT), to minimize any unwanted change to the NTF. Doing this also adds

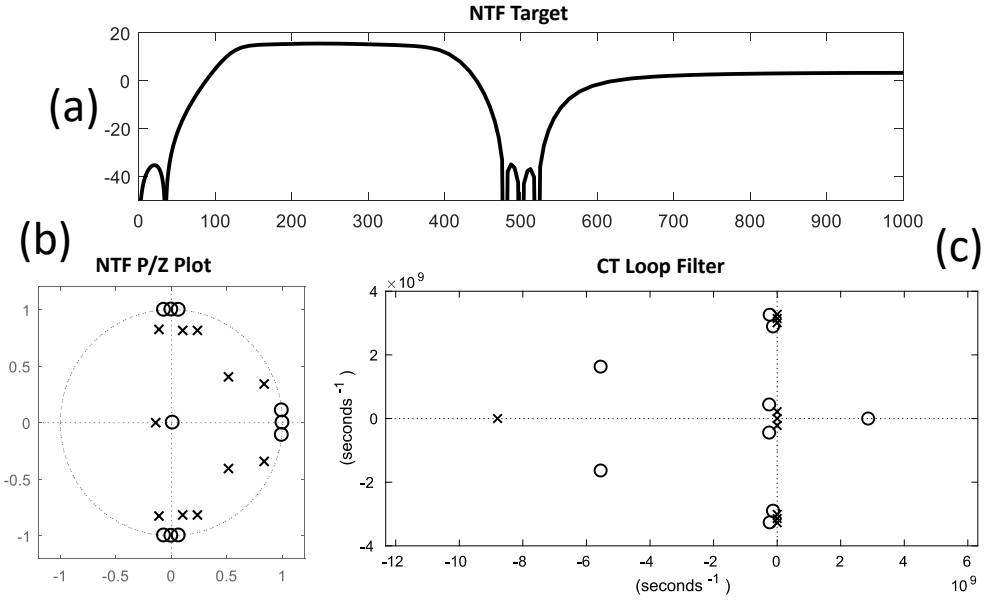


Figure 5.3: ***(a)*** *The transfer function of the target multi-band NTF.* ***(b)*** *The poles and zeros associated with the target NTF.* ***(c)*** *The poles and zeros of the loop filter, after conversion to continuous-time.*

an extra pole that can be used to shape the NTF.

The secondary TIA poles, however, are not accounted for and must be at high-frequency to avoid disturbing the desired NTF. In practice, a small amount of extra delay (typically 10-20% of a clock cycle) is added to the DAC to compensate for the extra phase added by these poles.

The NTF's pole/zero locations are chosen by hand with the goal of creating maximally flat out-of-band regions while keeping the H_∞ below 16dB, which is near the maximum for a 12-level quantizer (as determined with simulation[3]). Complicating this process is the fact that, since this modulator uses CT ELD compensation (i.e., a DAC-based fast-path), the NTF must follow the rules described in Chapter 3. In particular, the sum of the pole locations must be equal to the sum of the zero locations. The pole on the negative real axis (left of the origin in Fig. 5.3(b)) compensates for the imbalance in this equation caused by the baseband zeros being farther right than the poles, by necessity. Unfortunately, because of this limitation, the out-of-band gain in the inter-band region is significantly higher than the out-of-band gain at high-frequency, causing a reduction in the amount of noise that can be shaped (detailed discussion in Section 4.2.1). This is an unavoidable consequence of the choice

to implement the modulator as a fully-CT loop. This design choice was made to simplify this proof-of-concept design. Future work can avoid this problem by using hybrid loop architectures, as discussed in Chapter 4.

5.2.2 High-Level Loop Filter Design

Once the NTF is chosen, the DT loop filter can be calculated by solving for $L(z)$ in the NTF equation (including a unit delay in the quantizer):

$$\begin{aligned} \mathrm{NTF}(z) &= \frac{z^{-1}}{1 + z^{-1}L(z)} \\ L(z) &= \left(\frac{1}{\mathrm{NTF}(z)} - z \right) \end{aligned} \tag{5.1}$$

With the DT loop filter known, a direct application of the DT-CT conversion described in Chapter 3 gives the CT loop filter, $L_c(s)$, required to implement the target NTF. Note that the precise form of the state-space representations used for this conversion is not important—the state-space techniques in Chapter 3 can be primarily treated as an intermediate tool used to find the final pole/zero locations.

To begin implementing $L_c(s)$, we use techniques similar to that discussed in detail in Section 5.3. First, the TIA pole is removed from $L_c(s)$ and implemented as the TIA transfer function. This leaves a remainder, $L_{c1}(s)$, with nine poles and nine zeros. Next, the two bands are separated by removing the three poles and zeros closest to DC, which results in a 3-pole, 3-zero baseband section, $L_{\mathrm{BB}}(s)$ and a 6-pole, 6-zero bandpass section, $L_{\mathrm{BP}}(s)$. Removing the first zero from each section, finally, results in the high-level block diagram in Fig. 5.1. Each section is then separately synthesized by applying the synthesis method in Section 5.3.

From this, finally, we get everything we need to implement the loop filter: the transfer functions to be implemented by each biquad as well as the FF coefficients. The block diagram at this point is shown in Fig. 5.4. The final step is to rearrange the FF paths f_1 and f_2 so that they do not require forwarding the output of a summing block. This step is necessary because all summations are implemented in the current-domain, each current going into the virtual ground node of the biquad it is feeding into, making it impossible to directly forward the result of the summation. Instead, the inputs to the summation in question are duplicated and merged with the offending FF paths, resulting in the final, implementable diagram in Fig. 5.2.

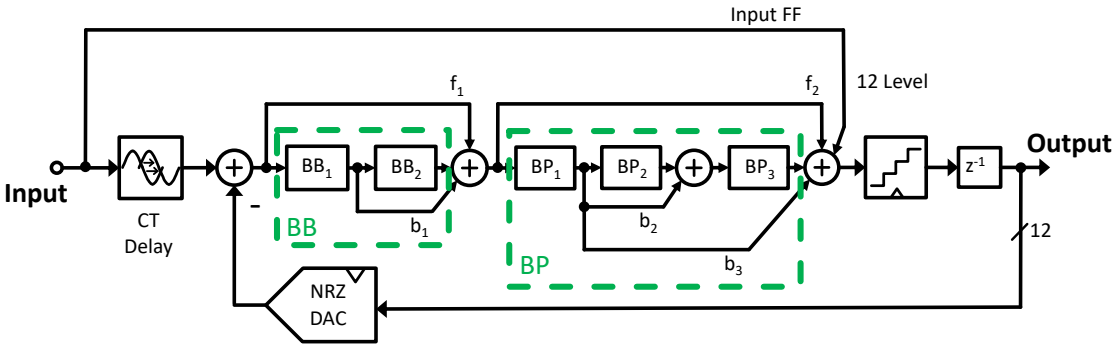


Figure 5.4: *The block diagram of the multi-band modulator before the last step of the synthesis process.*

5.2.3 Compensating for Non-Ideal Amplifiers in Biquads

The passives in each biquad are nominally chosen based on Eq. (5.4). However, non-ideal op-amp characteristics, such as finite gain and finite gain-bandwidth, inevitably make these simplistic equations of limited use on their own. Furthermore, deriving an equivalent equation that includes these non-idealities is very difficult and ultimately futile since including any model of the amplifier more complex than a single-pole increases the order of the biquad, making it impossible to equate with the ideal case.

Rather than focus on algebraic solutions, we use a simple machine-learning algorithm to find the filter parameters (R and C values) that give a compensated filter with poles and zeros at the correct locations. The algorithm works by observing that the inclusion of an amplifier is, in addition to adding extra poles/zeros, moving the desired poles/zeros to new locations. The algorithm attempts to reverse this process by moving the starting positions of the poles and zeros such that adding the amplifier moves them to the ideal locations. These new poles and zeros can then be used to calculate the filter parameters as usual. When those parameters are used in the real filter, the final transfer function will have poles and zeros in the correct place.

To find this new starting point, the algorithm chooses a starting point for each desired pole/zero equal to the ideal poles/zeros. It then numerically calculates the filter's transfer function, from this starting point, after adding the amplifier model (Fig. 5.5(1)), which is done by merely substituting $A(s)$ into the feedback equation:

$$\frac{I_{out}(s)}{I_{in}(s)} = \frac{A(s)Z^*(s)}{1 + \frac{A(s)Z^*(s)}{Z_{FB}(s)}} \frac{1}{Z_{\mathrm{FF}}(s)} \tag{5.2}$$

where Z_{FB} is the impedance of the feedback network in Fig. 5.8, Z_{FF} is the

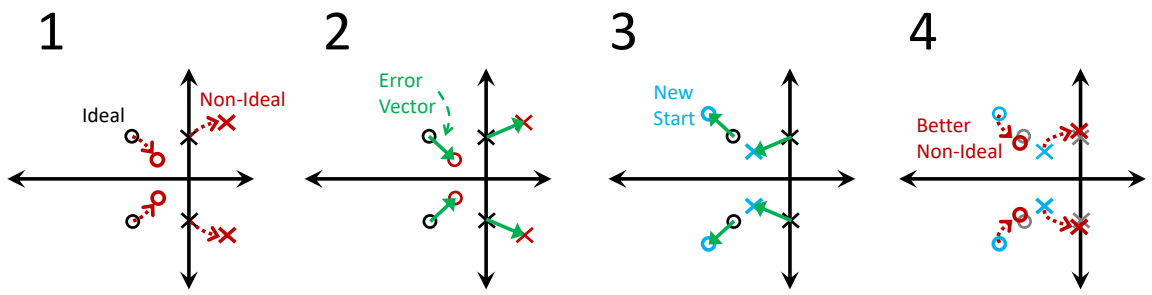


Figure 5.5: *The process of compensating a biquad for a non-ideal amplifier.* ***(1)*** *Ideal poles and zeros are moved to new locations by the inclusion of a non-ideal amplifier model.* ***(2)*** *The error vectors between ideal and non-ideal poles and zeros are calculated.* ***(3)*** *A new starting point is chosen by moving the poles and zeros in the opposite direction of their error vectors, with a small relative step size.* ***(4)*** *After adding the non-ideal amplifier model to the new starting poles and zeros, the resulting non-ideal poles and zeros are closer to ideal. The process is repeated until the magnitude of the errors are small, and the final filter parameters are calculated from the transfer function defined by the final starting poles and zeros.*

impedance of the feed-forward network after the amplifier, and Z^* is the input impedance in parallel with Z_{FB}.

Next, it then considers the nearest non-ideal poles/zeros to their ideal counterparts and calculates the difference between each ideal/non-ideal pair (Fig. 5.5(2)). This difference can be considered a vector that represents the distance and direction traveled by each of these poles/zeros after adding the amplifier. In reality, the path the output poles/zeros travel as the input poles/zeros change is extremely non-linear, but it can be linearized with sufficiently small step sizes. Thus, a new starting point is chosen by moving the poles and zeros of the original starting point in the direction opposite the distance vector, with a distance some small fraction of the actual distance vector (Fig. 5.5(3)). The filter is then re-calculated as if the new starting point was the ideal transfer function, repeating the process until the magnitude of the final pole/zero errors converge. By effectively undoing the movement of the poles and zeros caused by the non-ideal amplifier, the algorithm approaches a starting transfer function for which the non-ideal amplifier's effects create the desired final transfer function. The implemented filter parameters, then, are those calculated from the latest starting transfer function.

Using this method allows for arbitrarily complex amplifier models without any

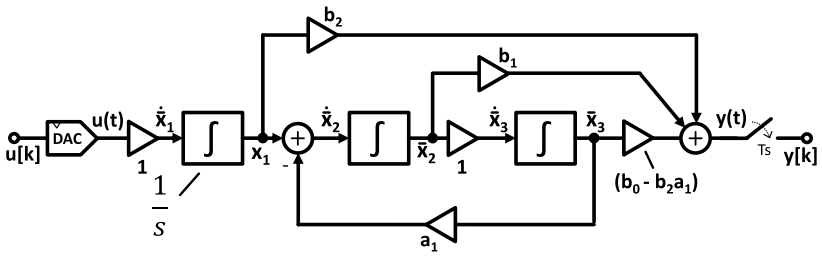


Figure 5.6: *The block diagram for a CRFF loop filter.*

increase in analysis difficulty. It should be noted, though, that often the location of high-frequency poles are more uncertain and can vary with process and layout, so care should be taken not to overly rely on the precision of the amplifier models used. For robust design, the addition of high-frequency poles should have a minimal effect on the final design.

5.3 Loop Filter Synthesis for Multi-Band Modulator

A challenge of the MB-$\Delta\Sigma$ Modulator is that the overall filter order is large because it combines lowpass and bandpass filter responses. High-order modulators are notoriously challenging to design; to address this, we adopt a cascaded form and introduce a new modulator synthesis technique.

The difficulty with high-order modulators is that most practical structures are not suited for high order modulation due to a high sensitivity to small errors in the feedback and feed-forward gains. Consider a CT version of the classic CRFF filter structure from [3], shown in Fig. 5.6. When implementing a 4^{th}-order transfer function, the open-loop denominator is: $s^4 + (g_1 + g_2)s^2 + g_1 g_2$, with local feedback paths g_1 and g_2. While there are many benefits to this architecture, it does not scale well with order. In essence, the roots of some high-order polynomials become more sensitive to small perturbations in their coefficients [42], meaning the open-loop pole locations become equally sensitive to the gain of the feedback paths g_1 and g_2. To alleviate this problem, we implement the filter in a cascaded form by constructing each pair of poles and zeros using biquads. While the use of biquads in a CT-$\Delta\Sigma$ modulator is not unusual, it is critical to successfully designing very high-order modulators.

In order to design the complex loop filter required for a multi-band modulator, we

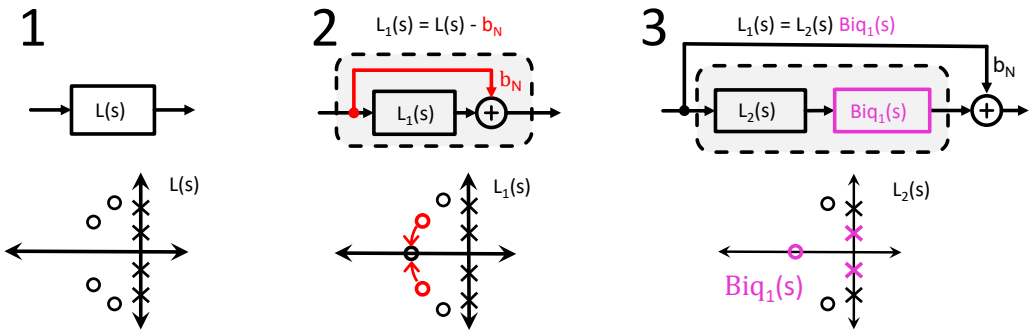


Figure 5.7: *Feed-forward synthesis method.* ***(1)*** *Required filter.* ***(2)*** *FF removed from L(s).* ***(3)*** *Biquad (with 1 zero) removed from $L_1(s)$.* ***(4)*** *Repeat (2) with L_2 (s).*

develop a synthesis method to iteratively produce the filter from the desired response. Our process (Fig. 5.7) is inspired by the passive filter synthesis technique in [39] and facilitates efficient single-opamp biquads [31], which are limited to two poles and a single zero. Consider the desired transfer function, L(s), shown as (1) in the figure. If the transfer function has an equal number of poles and zeros, then subtracting a constant removes the leading term in the numerator by canceling its coefficient. It can be said that we have "removed" a zero, since the numerator of the remainder, $L_1(s)$, is now one degree less. This subtraction of a constant can be implemented as an FF path around $L_1(s)$, (2) in the figure. We can now implement (i.e., "remove") two of $L_1(s)$'s poles and the (real) zero that has been created using a biquad and we are left, again, with a transfer function having an equal number of poles and zeros, $L_2(s)$, (3) in the figure. The process is repeated until the last biquad is implemented, and nothing remains. This filter design process can be easily applied to any desired loop filter, regardless of its complexity.

5.4 Modified Single-Amplifier Biquad

We implement the poles and zeros for each section using a modified version of the single-amplifier biquad in [31], shown in Fig. 5.8. A drawback of [31] is that it requires a complex system of DACs to stabilize the modulator due to insufficient freedom in the placement of zeros. We add an extra resistive path (R_{f2}) to the forward network, which adds a constant term in the numerator of the passive forward path's transfer function. This constant term creates the extra degree of freedom needed for the

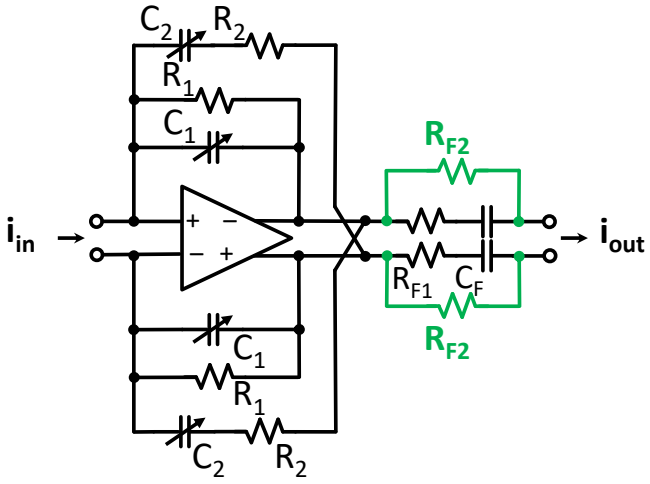


Figure 5.8: *Modified Biquad; R_{f2} added.*

biquad to implement any real zero. It alleviates the need for DACs with complex shapes, reducing power consumption, and simplifying the design process.

For an ideal amplifier, the transfer function can be calculated in the same way as in [31]:

$$\frac{I_{out}(s)}{I_{in}(s)} = \frac{R_1}{R_{f1}} \frac{1+\tau_2 s}{(1+\tau_{f1}s)} \frac{1+(\tau_{f1}+\tau_{f2})s}{(1+(\tau_1+\tau_2-R_1C_2)s+\tau_1\tau_2 s^2)} \tag{5.3}$$

where $\tau_i = R_i C_i$ and $\tau_{fi} = R_{fi} C_f$. Choosing $\tau_{f1} = \tau_2$ and factoring out the τ's, we see that:

$$\frac{I_{out}(s)}{I_{in}(s)} = \frac{R_1}{R_{f1}} \frac{\omega_1\omega_2}{\omega_z} \frac{s+\omega_z}{s^2+(\omega_1+\omega_2-\frac{1}{R_2C_1})s+\omega_1\omega_2} \tag{5.4}$$

where $\omega_i = 1/\tau_i$ and $\omega_z = 1/(\tau_2+\tau_{f2})$. The biquad transfer function's coefficients can finally be equated with Eq. (5.4) and the filter parameters solved. For an ideal transfer function, $TF(s)$, we can write two intermediate variables:

$$\begin{aligned} TF(s) &= \frac{b_1 s + b_0}{s^2 + a_1 s + a_0} \\ f_0 &= b_0/a_0 \\ f_1 &= b_1/\omega_1 - f_0 \end{aligned} \tag{5.5}$$

and use them to solve for each of the filter parameters:

$$
\begin{aligned}
R_1 &= \text{given} \\
\omega_1 &= \sqrt{a_0}/\beta \\
\omega_2 &= \sqrt{a_0} \cdot \beta \\
\\
C_1 &= R_1/\omega_1 \\
R_2 &= (\omega_1 + \omega_2 - a_1)/C_1 \\
R_{f1} &= R_1/f_0 \\
R_{f2} &= R_1/f_1 \\
C_f &= R_{f1}/\omega_2
\end{aligned}
\tag{5.6}
$$

There are now two free parameters: β, which chooses the ratio of ω_1 and ω_2, and R_1. In this prototype, we use R_1 =1kΩ. Note that this analysis differs from [31] with the introduction of the β term. Typically, β is chosen to be 1, as in [31], but in some cases R_2 can become small, requiring more power from the amplifier to drive. In this case, increasing β increases R_2 since $\omega_1 + \omega_2$ can be written as $\sqrt{a_0}(1/\beta + \beta) = \sqrt{a_0}(\beta^2 + 1)/\beta$. In the prototype presented here, we use both $\beta = 1$ and $\beta = 2$.

5.5 Improved Linearity Through Loop-Delay Matching

Digitizing multiple bands with the same ADC has the potential to cause significant inter-band distortion products. We apply and improve the conventional linearity-improving input FF technique [3], which allows the input signal to bypass the amplifiers through an input FF, as shown in Fig. 5.2(a). Since the FF path is injected just before the quantizer, any signal distortion from the FF path due to the TIA is noise-shaped, keeping the two bands well-isolated from each other. However, for CT modulators, the conventional resistive FF cannot fully cancel the input at high-frequencies due to extra phase and gain induced by the TIA, loop delay, and DAC, limiting the linearity benefits. Shown in Fig. 5.9, fully canceling the input requires that the input signal and its replica, after traveling around the loop, be identical in both magnitude and phase. To approximate this, we add a first-order model of the loop delay to the input network by changing the input resistors into passive all-pass

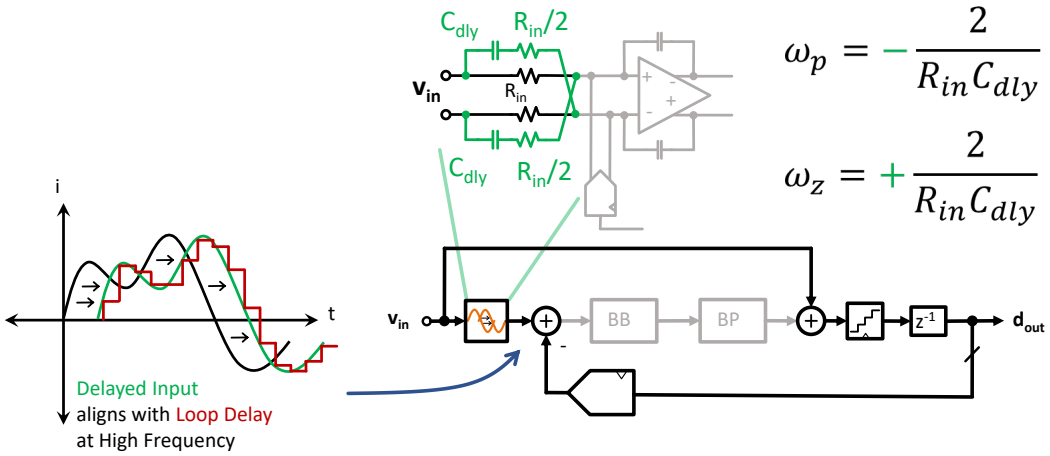


Figure 5.9: *Delay matching all-pass filter in the input network.*

filters. This cross-coupled path adds a left-half-plane pole and right-half-plane zero at the same frequency, set by C_{dly}. The pole and zero cancel each other's magnitude but add 180 degrees of phase shift. The transition of this phase shift is chosen so that the phase of the input path within the BP region approximates the phase of the FF path as it returns to the input summing node. This new approach requires only simple modifications to the existing resistive input and reduces the input signal present in the high-frequency biquads by >10dB over the bandpass bandwidth.

5.6 Amplifier Design

The amplifiers used in this work use a feed-forward architecture with three gm stages in the high-gain, low-frequency path and one gm stage in the high-frequency, low gain path. In the biquad amplifiers, shown in Fig. 5.10(a), cascoded gm-stages are preferred, especially in the FF path, due to their improved common-mode rejection. For the TIA, shown in Fig. 5.10(b), inverter-based gm stages are preferred due to their higher swing, which ultimately limits the modulator's maximum input amplitude (since this is the only amplifier that sees the full input signal + quantization noise).

Each amplifier also has a unity-gain output buffer, which uses feedback to isolate the gain stages from the load, shown on the right side of Fig. 5.10. Feedback in this stage boosts the frequency of the output pole and provides the previous stages with the small capacitive load, keeping secondary poles at high-frequencies without sophisticated compensation techniques. It also makes the transfer function of the

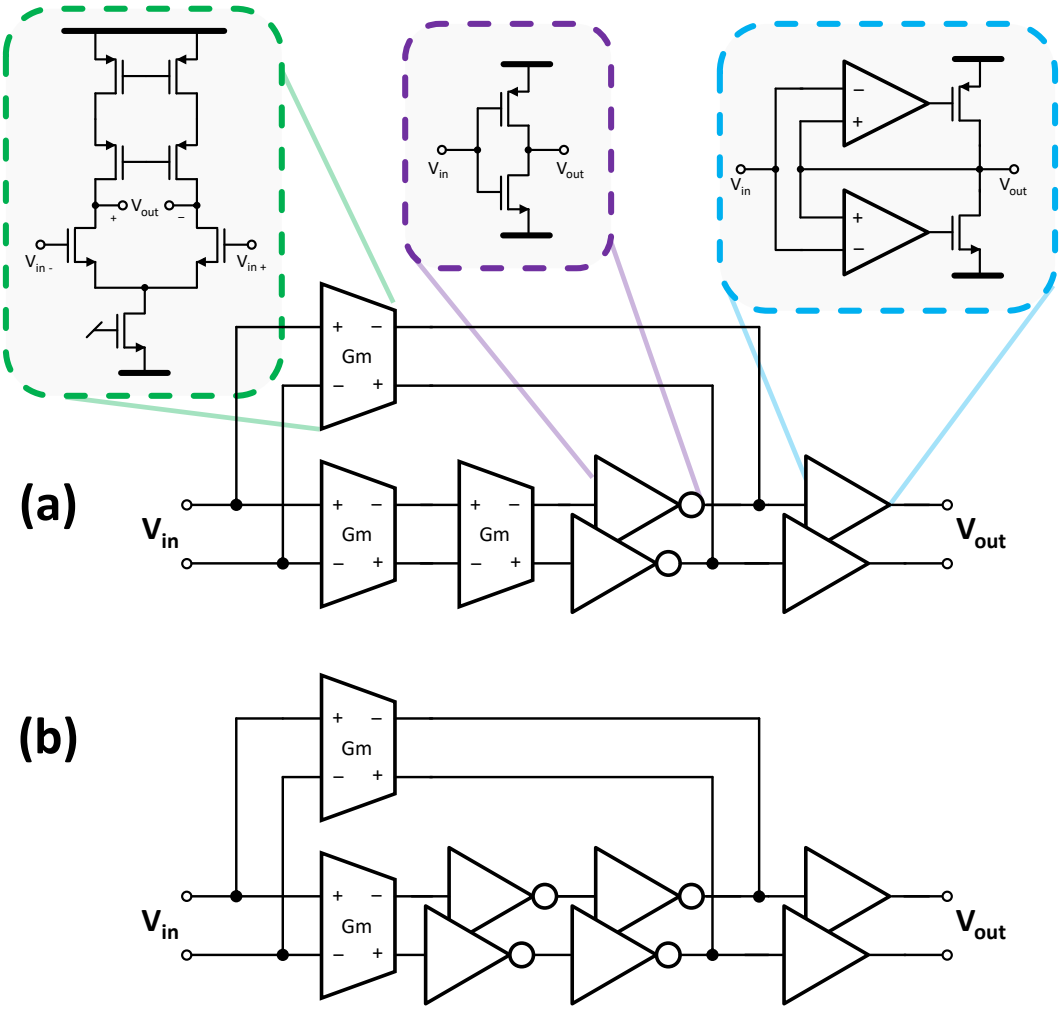


Figure 5.10: *A schematic of* ***(a)*** *the biquad amplifier and* ***(b)*** *the TIA amplifier.*

amplifier relatively unaffected by the load, allowing for a more straightforward calculation of the non-ideal amplifier's affects on the biquad transfer function. Finally, it simplifies the design process by allowing the compensation (and, therefore, the amplifier design) to be consistent across all biquads in a given band. Lastly, it keeps the load resistors—some of which are 100's of ohms—from impacting the amplifier's DC gain. The output buffers are based on the Common-Source with Error Amplifier configuration in [43].

With a 1kΩ, 3pF parallel load, the BB amplifiers have a unity-gain frequency of approximately $1.3f_s$ and a DC gain of 50dB. With a 1kΩ, 600fF load, the BP amplifiers have a unity-gain frequency of $1.75f_s$ and a DC gain of 52dB. Finally, with a 2kΩ, 60fF load, the TIA amplifier has a unity-gain frequency of $2.5f_s$ and a DC gain of 41dB.

5.7 Results

The prototype is fabricated in 40nm CMOS, occupies 0.22mm^2 and consumes 45mW from a 1.2V supply. Shown in Fig. 5.11, the measured Dynamic Range (DR) is 62dB in the BB and 43dB in the BP. Fig. 5.12, and Fig. 5.13 show the measured spectra for a single tone in each band, demonstrating a peak Signal-to-Noise and Distortion Ratio (SNDR) of 54.3dB and 41.4dB for 10MHz and 504MHz inputs, respectively. Fig. 5.14 shows the spectrum for simultaneous tones in both bands, demonstrating a peak SNDR of 55dB and 39dB or, at most, 2.4dB less than with single-band operation. In this test, the tones are at equal power—after compensating for slight differences in measured STF—and are increased together until instability, using a worst-case choice of BP frequency: where the 4$^{\text{th}}$ and 8$^{\text{th}}$ BP harmonics fall in the BB. Note that the inter-band distortion tones do not contribute significantly to the BB SNDR. The Spurious-Free Dynamic Range (SFDR) in each band—including inter-band distortion components—is 64dB and 42dB, respectively.

Fig. 5.15 shows the power distribution and a die photo. The quantizer and digital buffers in the feedback path are included in the "Digital" category.

Each spectrum is overlaid with a plot of the target NTF. Clearly, the measured NTF is a good match for the target NTF, demonstrating the accuracy of this design approach—even with a design as complex as a multi-band modulator. The most significant exception to the prediction is the in-band noise in the BP section. This noise is not due to the NTF; it is caused by unintended thermal noise in the BP biquads. Also, the drop off in the measured NTF at high-frequency is due to error in

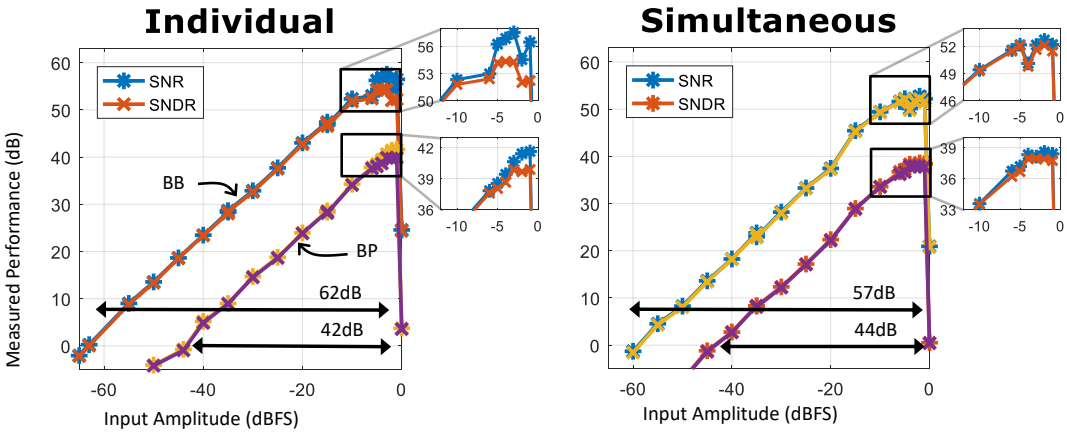


Figure 5.11: *Measured SNDR vs. input amplitude for single-band and multi-band operation. In* ***(left)****, blue and orange represent SNR and SNDR for the BB; purple and gold for BP. In* ***(right)****, yellow and teal are for the BB; purple and orange for the BP. In the multi-band case, all measurements at each input amplitude are taken simultaneously.*

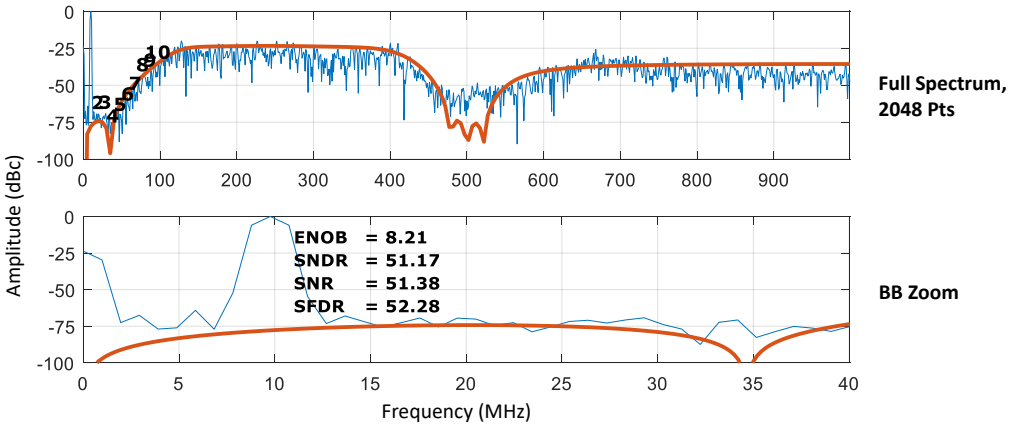


Figure 5.12: *Measured spectrum for baseband operation alone.*

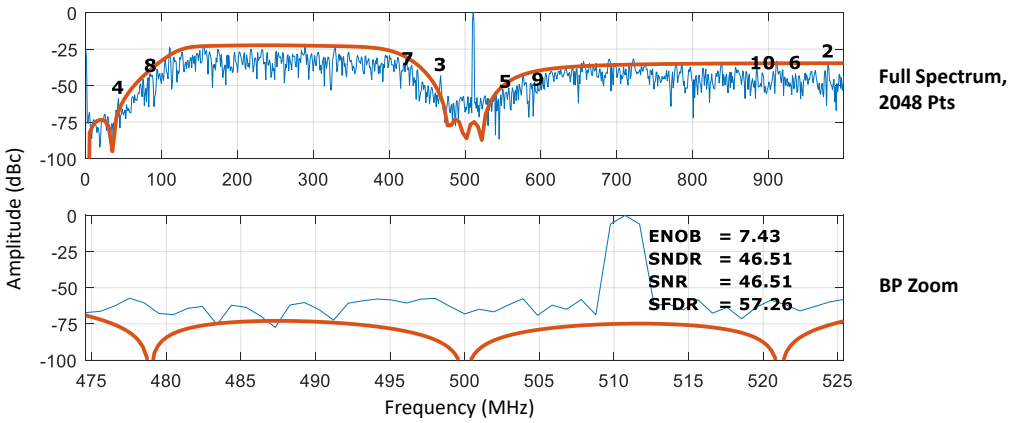


Figure 5.13: *Measured spectra for bandpass operation alone.*

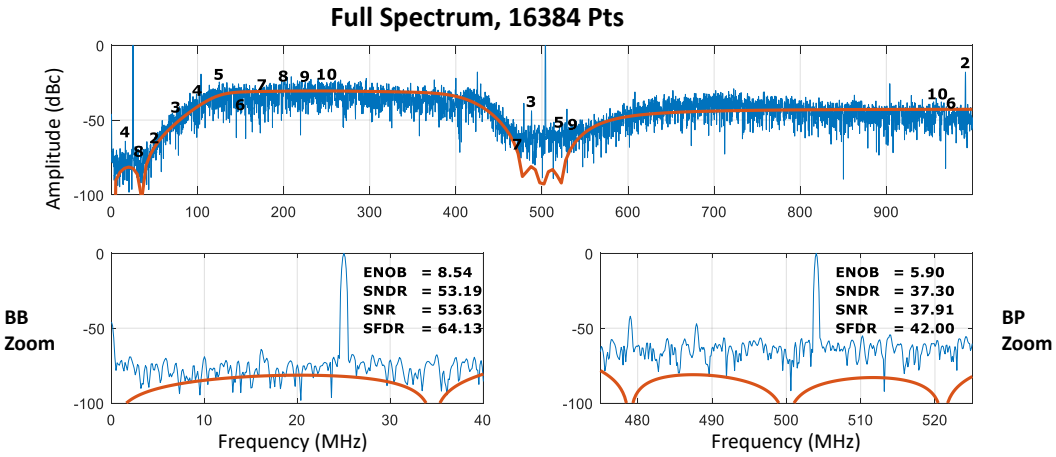


Figure 5.14: *Measured spectrum for multi-band operation,* ***(top)*** *baseband only,* ***(mid)*** *bandpass only.* ***(bot)*** *full spectrum.*

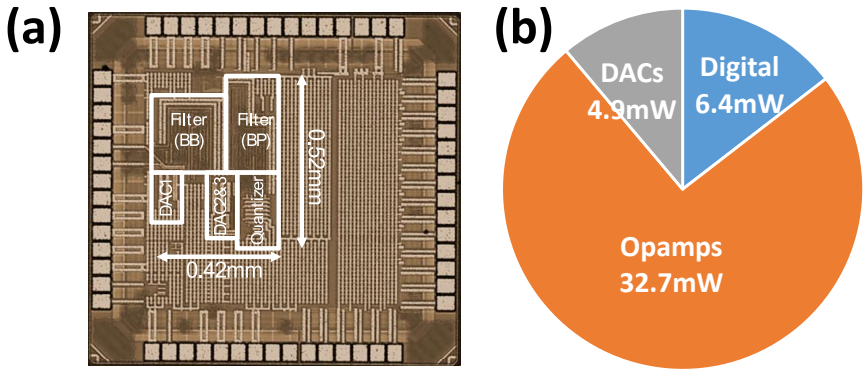


Figure 5.15: *Power breakdown and die photo of the Multi-Band $\Delta\Sigma$ Modulator.*

the loop filter's transfer function caused by secondary poles in the TIA.

5.7.1 Modified Figure of Merit

The Schreier Figure of Merit (FoM) is the standard performance measure for $\Delta\Sigma$ ADCs and can be written as:

$$\begin{aligned} \text{FoM}_S &= 10\log_{10}\frac{\text{DR}\times BW}{P} \\ &= DR_{dB} + 10\log_{10}\frac{BW}{P} \end{aligned} \tag{5.7}$$

However, this equation assumes a single band of interest. In order to compare this work with the state-of-the-art, the Schreier FoM, must be modified slightly. To preserve the relationships between dynamic range, bandwidth, and power, each band should be considered independently and summed linearly:

$$\text{FoM}_{MS} = 10\log_{10}\frac{\sum \text{DR}_i BW_i}{P} \tag{5.8}$$

Consider, for example, a modulator with two bands: the second with 2x the BW but 1/2 the DR (linearly, so 3dB less) of the first. The power of such a modulator would be twice the single-band case, since the power requirements of each band are equal. With this figure of merit, the power in the denominator is correctly offset by the change in the numerator. If instead, each component were summed before multiplying, the numerator would increase by 4.5x creating an imbalance with the 2x increase in power.

			Similar Aggregate BW		Similar Frequency Range		Similar Individual Bands	
	This Work		Bolatkale ISSCC 2011	Huang ISSCC 2017	Dong ISSCC 2016	Lee ISSCC 2014	Breems ISSCC 2016	Chae ISSCC 2012
Architecture	**CT-MB-ΔΣM**		CT-ΔΣM	CT-ΔΣM	1-2 CT-MASH	TI-SAR	CT-ΔΣM	BP-ΔΣM
Process (nm)	**40**		45	16	28	65	65	65
Fs (GHz)	**2**		4	2.15	8	1	2.2	0.8
Aggregate OSR	**11**		16	8.6	8.6	1	44	16.7
Power (mW)	**44**		256	54	930	19.8	41.4	12
Band Type	**BB**	**500MHz BP**	BB	BB	BB	BB	BB	200MHz BP
BW (MHz)	**40**	**50**	125	125	465	500	25	24
DR (dB)	**62**	**43**	70	74.8	69	51.4	77	60
Peak SNR (dB)	**57.6**	**41.6**	65	71.9	63	--	--	--
Peak SNDR (dB)	**54.3**	**41.4**	65	72.6	64	51.4	77	58
Area (mm^2)	**0.22**		0.9	0.217	1.4	0.78	0.25	0.2
FOM1 (dB)	**N/A**		156.9	168.4	156.0	155.4	164.8	153.0
FOM2 (dB)	**151.7**		156.9	168.4	156.0	155.4	164.8	153.0

Table 5.1: *Performance Comparison Table*

Unfortunately, this observation makes the dB form somewhat more unwieldy than the original—the dynamic range must first be converted to its linear value, computed, then converted back again. Since this is primarily a problem for intuitive calculation, a shorthand form can be used if the ratios of the DR and BW are easily known. The following is the result of simply factoring $\mathrm{DR}_1 BW_1$ from the summation in the numerator:

$$\begin{aligned} \mathrm{FoM}_{MS}(dB) &= \mathrm{DR}_1(dB) + 10log\left(\frac{BW_1}{P}\right) + 10log\left(1 + \sum_{i\neq 1} \frac{\mathrm{DR}_i}{\mathrm{DR}_1}\frac{BW_i}{BW_1}\right) \\ &= FoM_{S1} + 10\log_{10}\left(1 + \sum_{i\neq 1} \frac{\mathrm{DR}_i}{\mathrm{DR}_1}\frac{BW_i}{BW_1}\right) \end{aligned} \tag{5.9}$$

This equation illustrates that the modified FoM can be thought of as simply the Schreier FoM for one band with a correction factor to account for each additional band. For a second band with the same FoM as the first, the correction is +3dB; three bands gives +4.7dB, and four bands gives +6dB.

Using this metric, the prototype has an Aggregate Schreier FoM of 151.7dB. In Table 5.1, its performance is compared to other ADCs with similar aggregate specifications. Not only is this ADC capable of digitizing signals further apart than any conventional single-band $\Delta\Sigma$, its performance is comparable to other state-of-the-art

converters with similar aggregate specifications. Furthermore, even with multi-band operation, the area of the prototype is among the smallest.

The prototype's area is comparable to even the smallest single-band modulator. Furthermore, using the two single-band ADCs listed here to provide a similar base-band + bandpass interface would result in higher power consumption, even though it would only have half the aggregate BW and half the bandpass center frequency. Finally, the only $\Delta\Sigma$ ADC that is able to convert the full 500 MHz spectrum is [11] ("Dong ISSCC 2016" in Table 5.1) requires more than 20x the power consumption as this multi-band prototype.

CHAPTER 6

Conclusion

We present the first multi-band $\Delta\Sigma$ modulator. Our multi-band CT-$\Delta\Sigma$ prototype simultaneously digitizes two bands, one at baseband and one at 500MHz, with an aggregate bandwidth of 90MHz [44] [45].

In addition to reducing the number of ADCs, our multi-band ADC approach alleviates the need for multiple power-hungry front-ends. By only driving one ADC, we remove mixers, buffers, and LNAs, since each of the bands-of-interest does not need to be separated before the ADC. Instead, we efficiently separate the bands as a natural part of the ADC's existing digital decimation filtering.

Our multi-band ADC also improves on a single wide-band ADC by making much more efficient use of the $\Delta\Sigma$ noise-shaping spectrum. By only shaping noise away from the regions of interest, as opposed to the full spectrum, we can do more with the same overall OSR, allowing for a much lower clock rate and leading to significant overall power savings—nearly 900mW, in the case of [11]. Lastly, even though [11] has the widest BW of any CMOS $\Delta\Sigma$ to-date, it is still not wide enough to capture the 500MHz band separation used in this work.

To enable this new architecture, we also introduce: (1) a FF synthesis method which simplifies the implementation of the high-order loop filter; (2) a modified single-amplifier biquad which simplifies the DAC design; (3) a modification to the input resistor network that reduces loop filter nonlinearity and inter-band distortion; and (4) a method for numerically compensating for non-ideal amplifier effects in the loop filter

This work also introduces several theoretical insights which greatly simplify the design of highly complex modulators, such as the multi-band modulator. These insights include: (1) a general solution to the Excess Loop Delay problem and Discrete-Time to Continuous-Time conversion which is compact & easy-to-use, and (2) a limitation of the noise shaping capacity in CT modulators due to the finite bandwidth inherent

in CT-based ELD compensation methods.

Finally, by demonstrating that arbitrary noise-shaping is possible, this work is a step toward custom, application-driven noise-shaping solutions that provide new tools to solve challenging design problems. While this prototype is an essential first step, future work must demonstrate the addition of more than two bands, tunability of band separations, and must more precisely address the needs of a specific application areas such as LTE and 5G.

6.1 Open Source Code

One goal of this work is to provide tools that can make replicating the multi-band ADC presented here more approachable, even with its high degree of complexity. To that end, MATLAB reference implementations of the CT-DT and Discrete-Time to Continuous-Time (DT-CT) tools developed in Chapter 3 have been written and are available in the following GitHub repository:

https://github.com/johnlb-delta-sigma/matlab

6.2 Future Work

The most straightforward improvement would be to use a DT-based ELD compensation scheme, such as those described in Section 4.5. This would allow the noise budget to increase and provide the flexibility required to implement tunable bandpass bands, using the tunability of the biquads along with a few tunable resistors for FF paths. Demonstrating an ADC with more than two bands would also be useful.

A critical step toward a multi-band ADC that is practical for modern wireless systems is to refine the STF as well. While this work's primary focus was on the NTF, reducing STF peaking, while certainly achievable, is likely non-trivial. Furthermore, future designs should be made with an eye to specific communication standards, such as LTE-A or 5G.

Finally, we believe there is the opportunity for as-yet unconsidered, application-specific noise shaping. $\Delta\Sigma$ modulation has a history of finding surprising applications and enabling incredible technologies; application-specific noise shaping has the potential to continue that history into the future.

BIBLIOGRAPHY

[1] J. Cherry and W. Snellgrove, *Continuous-Time Delta-Sigma Modulators for High-Speed A/D Conversion*. IEEE, 2002.

[2] C. H. Wei and N. C. Chen, "A 72.6dB-SNDR 100MHz-BW 16.36mW CTDSM with preliminary sampling and quantization scheme in backend subranging QTZ," in *2019 IEEE International Solid-State Circuits Conference (ISSCC)*, 2019.

[3] S. Pavan, R. Schreier, and G. C. Temes, *Understanding Delta-Sigma Data Converters*. IEEE, 2017.

[4] J. Liu, S. Li, W. Guo, G. Wen, and N. Sun, "A 0.029-mm2 17-fJ/conversion-step third-order CT $\Delta\Sigma$ ADC with a single OTA and second-order noise-shaping SAR quantizer," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 2, pp. 428–440, Feb 2019.

[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. SET-8, no. 3, pp. 204–209, Sept 1962.

[6] J. 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.

[7] S. AMBATI, "Studies on a data communication system using digital matched-filter techniques part i. theoretical investigations," *International Journal of Electronics*, vol. 36, no. 1, pp. 49–71, 1974.

[8] R. 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.

[9] S. D. Kulchycki, "Continuous-time sigma-delta adcs," *Application Note SNAA098*, 2008. [Online]. Available: http://www.ti.com/lit/an/snaa098/snaa098.pdf

[10] R. Chen and H. Hashemi, "Reconfigurable SDR receiver with enhanced front-end frequency selectivity suitable for intra-band and inter-band carrier aggregation," in *ISSCC*, Feb 2015, pp. 1–3.

[11] Y. Dong, J. Zhao, W. Yang, T. Caldwell, H. Shibata, R. Schreier, Q. Meng, J. Silva, D. Paterson, and J. Gealow, "A 930mW 69dB-DR 465MHz-BW CT 1-2 MASH ADC in 28nm CMOS," in *ISSCC*, Jan 2016, pp. 278–279.

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

[13] A. M. Thurston, T. H. Pearce, and M. J. Hawksford, "Bandpass implementation of the sigma-delta A-D conversion technique," in *1991 International Conference on Analogue to Digital and Digital to Analogue Conversion*, 1991, pp. 81–86.

[14] S. A. Jantzi, M. Snelgrove, and P. F. Ferguson, "A 4th-order bandpass sigma-delta modulator," in *1992 Proceedings of the IEEE Custom Integrated Circuits Conference*, 1992, pp. 16.5.1–16.5.4.

[15] Weinan Gao, J. A. Cherry, and W. M. Snelgrove, "A 4 GHz fourth-order SiGe HBT band pass $\Delta\Sigma$ modulator," in *1998 Symposium on VLSI Circuits. Digest of Technical Papers (Cat. No.98CH36215)*, 1998, pp. 174–175.

[16] L. E. Pellon, "RF-to-digital receivers employing bandpass multibit sigma delta ADC architectures," in *GaAs IC Symposium. IEEE Gallium Arsenide Integrated Circuit Symposium. 20th Annual. Technical Digest 1998 (Cat. No.98CH36260)*, 1998, pp. 11–14.

[17] A. I. Hussein and W. B. Kuhn, "Bandpass sigma delta modulator employing undersampling of RF signals for wireless communication," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 47, no. 7, pp. 614–620, 2000.

[18] A. I. Hussein, "A 2.4-GHz bluetooth CMOS transceiver for wireless LANs," in *2003 46th Midwest Symposium on Circuits and Systems*, vol. 3, 2003, pp. 1251–1254 Vol. 3.

[19] A. Latiri, H. Aboushady, and N. Beilleau, "Design of continuous-time sigma delta modulators with sine-shaped feedback DACs," in *2005 IEEE International Symposium on Circuits and Systems*, 2005, pp. 3672–3675 Vol. 4.

[20] H. Chae, J. Jeong, G. Manganaro, and M. Flynn, "A 12mW low-power continuous-time bandpass $\Delta\Sigma$ modulator with 58dB [sndr] and 24MHz bandwidth at 200MHz If," in *2012 IEEE International Solid-State Circuits Conference*, 2012, pp. 148–150.

[21] H. Black, "Stabilized feedback amplifiers," *The Bell System Technical Journal*, vol. 13, no. 1, pp. 1–18, Jan 1934.

[22] E. Fernandez-Car and E. Zuazua, "Control theory: History, mathematical achievements and perspectives," 01 2003.

[23] *Linear State-Space Control Systems*. John Wiley Sons, Ltd, 2007.

[24] G. Sanderson, *The Essence of Linear Algebra*. 3blue1brown, 2016. [Online]. Available: https://www.youtube.com/watch?v=fNk_zzaMoSs&list=PLZHQObOWTQDPD3MizzM2xVFitgF8hE_ab

[25] A. Ben-Israel and T. N. Greville, *Generalized Inverses: Theory and Applications*. Springer, 2003.

[26] G. Strang, "The fundamental theorem of linear algebra," *The American Mathematical Monthly*, vol. 100, no. 9, pp. 848–855, 1993. [Online]. Available: http://www.jstor.org/stable/2324660

[27] M. Keller, A. Buhmann, J. Sauerbrey, M. Ortmanns, and Y. Manoli, "A comparative study on excess-loop-delay compensation techniques for continuous-time sigma–delta modulators," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 55, no. 11, pp. 3480–3487, Dec 2008.

[28] H. Shamsi and O. Shoaei, "Continuous time delta-sigma modulators with arbitrary dac waveforms," in *APCCAS 2006 - 2006 IEEE Asia Pacific Conference on Circuits and Systems*, Dec 2006, pp. 187–190.

[29] M. Keller, A. Buhmann, J. Sauerbrey, M. Ortmanns, and Y. Manoli, "A comparative study on excess-loop-delay compensation techniques for continuous-time sigma–delta modulators," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 55, no. 11, pp. 3480–3487, Dec 2008.

[30] O. Shoaei and W. M. Snelgrove, "A multi-feedback design for LC bandpass delta-sigma modulators," in *Proceedings of ISCAS'95 - International Symposium on Circuits and Systems*, vol. 1, April 1995, pp. 171–174 vol.1.

[31] J. Jeong, N. Collins, and M. P. Flynn, "A 260 MHz IF sampling bit-stream processing digital beamformer with an integrated array of continuous-time bandpass delta sigma modulators," *Jssc*, vol. 51, no. 5, pp. 1168–1176, May 2016.

[32] M. Schmidt, M. Grozing, S. Heck, I. Dettmann, M. Berroth, D. Wiegner, W. Templ, and A. Pascht, "A 1.55 GHz to 2.45 GHz center frequency continuous-time bandpass delta-sigma modulator for frequency agile transmitters," in *2009 IEEE Radio Frequency Integrated Circuits Symposium*, June 2009, pp. 153–156.

[33] J. S. Golan, *Foundations of Linear Algebra*. Springer, 1995.

[34] C. Weng, T. Wei, E. Alpman, C. Fu, and T. Lin, "A continuous-time delta-sigma modulator using ELD-compensation-embedded SAB and DWA-inherent time-domain quantizer," *IEEE Journal of Solid-State Circuits*, vol. 51, no. 5, pp. 1235–1245, 2016.

[35] C. Ho, C. Liu, C. Lo, H. Tsai, T. Wang, and Y. Lin, "A 4.5 mw CT self-coupled $\Delta\Sigma$ modulator with 2.2 MHz BW and 90.4 dB SNDR using residual ELD compensation," *IEEE Journal of Solid-State Circuits*, vol. 50, no. 12, pp. 2870–2879, 2015.

[36] T. He, M. Ashburn, S. Ho, Y. Zhang, and G. Temes, "A 50MHz-BW continuous-time $\Delta\Sigma$ ADC with dynamic error correction achieving 79.8dB SNDR and 95.2dB SFDR," in *2018 IEEE International Solid - State Circuits Conference - (ISSCC)*, Feb 2018, pp. 230–232.

[37] S. Wu, T. Kao, Z. Lee, P. Chen, and J. Tsai, "A 160MHz-BW 72dB-DR 40mW continuous-time $\Delta\Sigma$ modulator in 16nm CMOS with analog ISI-reduction technique," in *2016 IEEE International Solid-State Circuits Conference (ISSCC)*, 2016, pp. 280–281.

[38] W. Lee, "A novel higher order interpolative modulator topology for high resolution oversampling A/D converters." Ph.D. dissertation, 1987.

[39] W.-K. Chen, *Passive, active, and digital filters*. Taylor & Francis, 2006.

[40] B. Wu, S. Zhu, B. Xu, and Y. Chiu, "A 24.7mW 45MHz-BW 75.3dB-SNDR SAR-assisted CT $\Delta\Sigma$ modulator with 2nd-order noise coupling in 65nm CMOS," in *2016 IEEE International Solid-State Circuits Conference (ISSCC)*, Jan 2016, pp. 270–271.

[41] S. Ho, C. Lo, J. Ru, and J. Zhao, "A 23 mW, 73 dB dynamic range, 80 MHz BW continuous-time delta-sigma modulator in 20 nm CMOS," *IEEE Journal of Solid-State Circuits*, vol. 50, no. 4, pp. 908–919, April 2015.

[42] J. H. Wilkinson, "The perfidious polynomial," *Studies in Numerical Analysis*, no. 24, pp. 1–28, 1984.

[43] P. R. Gray, P. J. Hurst, S. H. Lewis, and R. G. Meyer, *Analysis And Design Of Analog Integrated Circuits*. Wiley, 2009.

[44] J. Bell and M. P. Flynn, "A simultaneous multiband continuous-time $\Delta\Sigma$ ADC with 90-MHz aggregate bandwidth in 40-nm CMOS," in *ESSCIRC 2019 - IEEE 45th European Solid State Circuits Conference (ESSCIRC)*, 2019, pp. 1–4.

[45] J. Bell and M. P. Flynn, "A simultaneous multiband continuous-time $\Delta\Sigma$ ADC with 90-MHz aggregate bandwidth in 40-nm CMOS," *IEEE Solid-State Circuits Letters*, vol. 2, no. 9, pp. 91–94, 2019.