HIGH-RESOLUTION SAR A/D CONVERTERS

WITH LOOP-EMBEDDED INPUT BUFFER

A DISSERTATION

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING

AND THE COMMITTEE ON GRADUATE STUDIES

OF STANFORD UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

Martin Krämer

August 2015

This dissertation is online at: http://purl.stanford.edu/fc450zc8031

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Boris Murmann, Primary Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Thomas Lee

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Bruce Wooley

Approved for the Stanford University Committee on Graduate Studies.

Patricia J. Gumport, Vice Provost for Graduate Education

This signature page was generated electronically upon submission of this dissertation in electronic format. An original signed hard copy of the signature page is on file in University Archives.

Abstract

Successive-approximation-register (SAR) analog-to-digital converters (ADCs) have generated a significant amount of interest in the past several years. While most of the recent literature focuses on low-to-moderate resolution designs (8-10 bits), we are now beginning to see significant advancements in the high-resolution space, targeting $\geqslant$14 bits at 10-100 MS/s. Traditionally, this performance range has been dominated by pipelined and delta-sigma architectures. However, several applications in high-speed control and interference cancellation require low latency and this is where the SAR topology becomes attractive. While several SAR ADCs have demonstrated efficient digitization at high speed and resolution, the difficulty of driving a large sampling capacitor with high accuracy in a short sampling window is an important challenge in the System-on-a-Chip (SoC) integration of such ADCs that is often ignored in the literature.

The work of this dissertation led to the design of a 14-bit 35 MS/s SAR ADC in 40 nm CMOS with a loop-embedded input buffer that consumes only 23% of the total ADC power. The buffer uses a source follower topology whose nonlinearities are cancelled by the SAR algorithm, allowing us to achieve 99 dB spurious-free dynamic range (SFDR) despite the small amount of invested power. That approach made it possible to reduce the input capacitance of the ADC by a factor of eighteen, easing the drive requirements substantially and resulting in the highest reported SFDR for SAR ADCs with a sampling speed greater than 20 MS/s.

Additionally, a second SAR ADC design (30 MS/s) based on a noise filter gear-shifting concept is introduced. Using a comparator with time varying noise performance, this ADC achieves a state-of-the-art figure of merit (FOM) of 161.6 dB and a peak signal to noise-plus-distortion ratio (SNDR) of 77 dB, which is the highest reported value for SAR ADCs with a sampling frequency above 20 MHz.

Acknowledgements

The PhD endeavor is often described as a journey; this description fits well in hindsight. Most of our journeys today are actually quite predictable. We often know where we want to go, most of the time we have an idea how we going to get there, and most journeys do not typically take five years or more. Think about it: how do you plan a trip, which has only a vaguely defined destination and takes that long? Simply put: "This is more than a journey, it is an expedition! And normally we don't do these kind of projects alone!" and that's why we write this section, to give credit to all the people that helped along the way and made this expedition possible!

First, I want to thank my advisor Prof. Boris Murmann. It goes without saying that without him I would not have had the opportunity to start this expedition nor of finishing it. To use a figure of speech, he entrusted me with a well-equipped ship and allowed me to set a course on a deep and wide-open ocean, free to discover. Sometimes I went off course, which is to be expected when you are free to sail, but he consistently encouraged and supported me with his knowledge and experience ensuring that I would find my way. You cannot ask for more. Furthermore, Prof. Murmann is an incredible teacher, and that helped tremendously with my research project. He teaches you how to actually use a map and compass, allowing you to plot your own course, instead of handing you a map with a predefined course to follow. I could go on but I stop here, nevertheless I encourage the reader to read more acknowledgments from former PhD students of Prof. Murmann, one will understand that we owe him big time.

I also want to deeply thank Erwin Janssen and Kostas Doris from NXP semiconductors. They supported my PhD expedition during many turning points and critical moments. Kostas thank you for taking care of all the business side and make me feel that I don't have to worry about it. I worried enough about my chip, I'm not sure if I could have dealt with that as well. Erwin you have been a great

resource of wisdom and technical expertise, and for that I am so thankful. I really enjoyed our discussions during long “sailing days” with a cup of tea or coffee. I want to thank NXP Semiconductors Inc., for fabrication support, as well as the NXP team for countless little hints and tips. Thank you Athon Zanikopoulos, Marcello Ganzerli, Maarten Vertregt, Hans Tuinhout. A special thanks goes to Marcel Pelgrom who has made this industry cooperation possible.

Once you traveled the oceans and survived the expedition, you have to write your findings down. This brings me to my committee and co-advisers. Prof. Bruce A. Woolly and Prof. Tom H. Lee, thank you for your valuable input during my research and while I wrote this thesis. It was an honor to have such accomplished and respected “sailors” helping me. Furthermore I want to thank the remaining members of my oral committee, Prof. R. Lanier Anderson and Prof. Julius O. Smith. Not only have they been part of the committee, they also thought my great things that were a little of course, Prof. Smith thank you for introducing me to the world of audio signal processing and Prof. Anderson thank you for giving me the chance to gain some insight into philosophy.

Thank you to the crew of the “Murmann Mixed-Signal Group”, my partners in crime. Even though everybody has its own ship, we still traveled a long way side by side and helped each other. Thanks Bill Chen, Kevin Zheng, Nikolaus Hammler, Stephen Weinreich, Man-Chia Chen, Jonathon Spaulding, Alex Omid-Zohoor, Nishit Harshad Shah, Sean Fischer, Lita Yang, Danny Bankman, Elaina Chai, Chris Young, Mahmoud Saadat, Po-Hsuan Wei, Alex Guo, Chenxin Zhu,Hussein Ali, Aldo Peña Perez, Vaibhav Tripathi, Ross Walker, Noam Dolev, Siddharth Seth, Alireza Dastgheib, Donghyun Kim, Pedram Lajevardi, Ray Nguyen, Drew Hall, Justin Kyungryun Kim, Wei Xiong, Clay Daigle, Manar El-Chammas. Special thanks goes to Ryan Boesch, Douglas Adams and Jonathon Spaulding for being awesome friends. Jon was always the first person who had to read my writings, and he helped my make that actually readable. Jon you most have suffered from time to time, I owe you. Thank you Ann Guerra (our admin, she should be admiral by now), you are simply the best!

Another great experience have been my several industry connections, I want to thank BOSCH R.T.C. and in particular Christoph Lang. He brought me to California and paved the way for me to join Stanford. Christoph thank you so much. Furthermore, I want to thank the people from ARDA Technology, letting me experience the start-up spirit. Here I want to thank Clemenz Portmann, you have been a great friend and help. You gave my career/expedition several import beacons that helped me avoiding dangerous cliffs.

Stanford and its surroundings is a very special place, mainly because of the incredible people you can find. It is big ocean and I was fortune enough to meet some really great people out there. They brought a lot of color into my life and to my journey. Thank you: Aaron Adcock (lunch buddy) and Sam Emaminejad (best project partner ever), the non-engineering folks: Carter Hunt, Dennis Pterezelka, Silvia De Toffoli, Mandy Mclean. Thanks to all the people from the German Student Association, in particular Andreas and Alexander Zoellner (Die Zoellner Brüder), and thanks to the Silicon Valley Folks: Matthias Hanel, Angela & Reinhard, Laura & Thomas, all at the "Financial Aid Dorm", Ellen & Jared, and Alex Reimers.

Here I want to thank my long time friends, some of them I know since Kindergarten. You need good friends in live and you guys are most certainly are, you keep me grounded. Thank you Jan Schumann, Tim Schumann, Patrick Eich, Nils Schumann, Julian Schneider and Söeren Kemmann.

Last but certainly not least, I want to thank my family. It is hard to find words that can describe my gratitude and appreciation. All that I can say is that they gave and still give me so much joy, love and support. All the best to my Aunts and Uncles and their families: Hildegard & Ulrich, Judith & Marc, Stephan, Genie, Sophie and Anton. Hugs for my brothers Mathias, Christoph and their families: Sabine, Hanna and Tim, as well as Christina and Paul. Finally, I want to send all my love to my parents Adele and Franz Krämer.

Danke Mama und Papa, für alles.

"Voltaire sagte, der Himmel habe uns zum Gegengewicht gegen die vielen Mühseligkeiten des Lebens zwei Dinge gegeben: die Hoffnung und den Schlaf. Er hätte noch das Lachen dazu rechnen können."

"Voltaire said that Heaven has given us two things to compensate us for the many miseries of life: hope and sleep. He could also have added laughter."

Immanuel Kant, aus der Kritik der Urteilskraft

(Critique of Judgment)

Content

List of Figures

List of Tables

1 Introduction

1.1 Motivation

Successive-approximation-register (SAR) analog-to-digital converters (ADCs) have received a great deal of interest in the past several years. While the majority of recent work has pushed low-to-moderate resolution designs (8-10 bits or 50-65 dB SNDR), we are now also seeing significant advancements in the high-resolution space, targeting ≥14 bits of resolution at 10-100 MS/s [1], [2]. This is visualized in Figure 1.1, showing the signal to noise plus distortion ratio (SNDR) versus the corresponding input frequency for published SAR ADCs at the two major circuit conferences ISSCC and VLSI [3]. Traditionally, this range has been dominated by pipelined and delta-sigma architectures. However, several applications in instrumentation, high-speed control, and interference cancellation require low latency, making the SAR topology attractive.

In this research, we consider the specific problem of input drive for SAR ADCs in the above-stated performance regime, where the converter's input capacitance must be large (typically 3-10 pF) and the available input acquisition time is short. This is further illustrated in Figure 1.2. Due to the sequential operation of the converter, a relatively short time (10-20% of the cycle or a few nanoseconds) is available for input signal tracking. During this window, the ADC driver must fully charge the large sampling capacitor (C_S) with low noise and distortion. To meet these stringent requirement, the vendors of high-speed, high-resolution standalone SAR ADCs offer custom-designed driver ICs (often bipolar), which typically consume more power than the ADC itself.

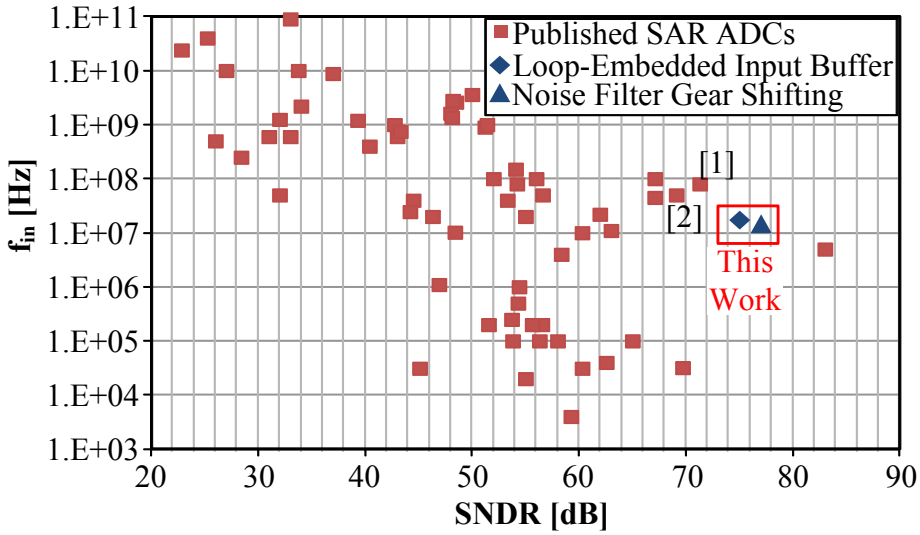

Figure 1.1: SNDR versus corresponding input frequency (f_{in}) for published SAR ADCs (data extracted from [3]), as well as for the two presented SAR ADCs of this work

Figure 1.3 illustrates this fact by showing an overview of commercial available SAR ADC driver chips. The power ranges from 10 mW to about 250 mW, while the spurious-free dynamic range (SFDR) ranges from 75-92 dB (see Figure 1.3(a)). Figure 1.3(b) plots the input frequency versus the SFDR specified at that frequency. We note that faster drivers together with higher SFDR tend to con-

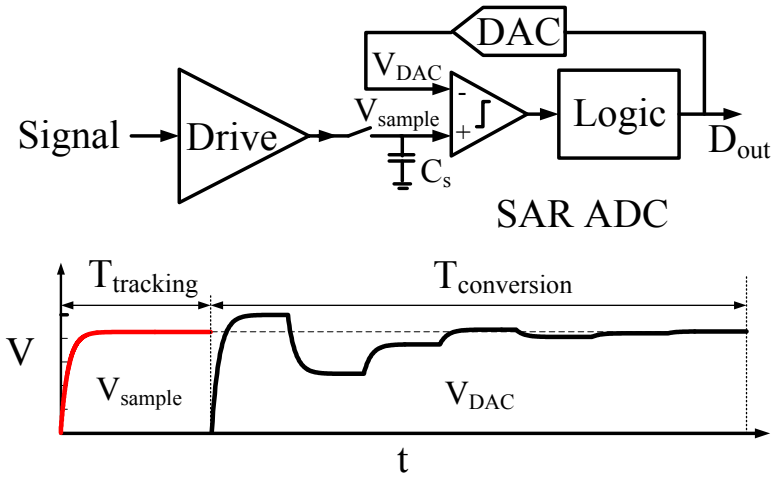

Figure 1.2: SAR ADC input interface and signal waveforms

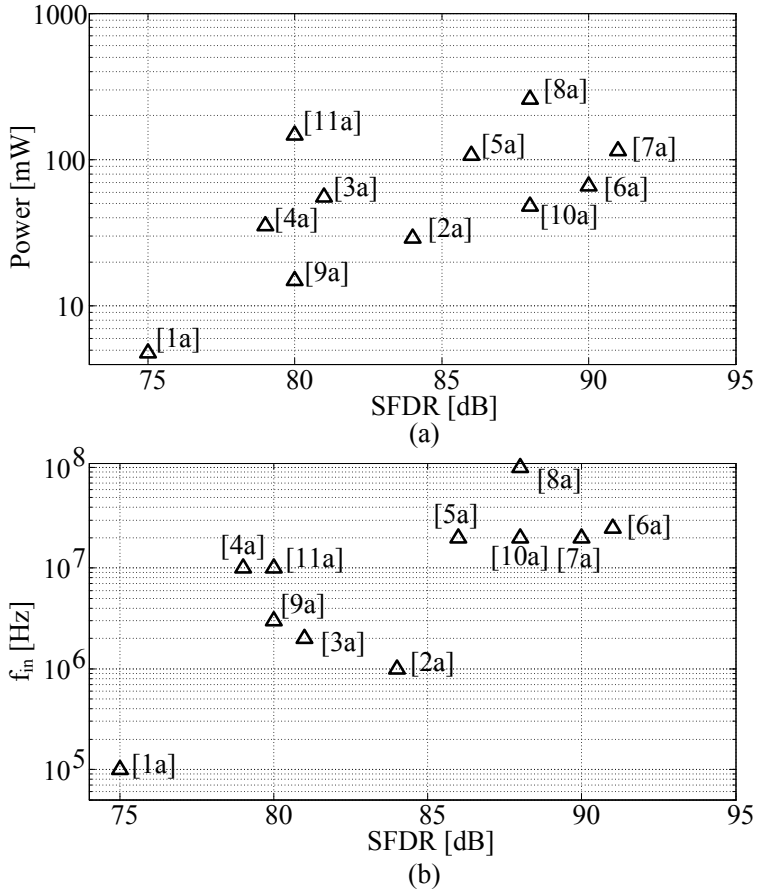

Figure 1.3: (a) Power versus SFDR overview of commercially available buffer ICs (data taken from datasheets available on vendors' webpage). (b) f_{in} versus SFDR[1]

sume more power (Power > 50 mW for SFDR > 85 dB). The goal of this work was to identify a CMOS SoC-compatible solution that minimizes the drive power overhead.

The solution pursued in this work was inspired by the 10-bit, time-interleaved design of Doris et al. [4], which introduced the idea of placing a signal buffer inside the SAR loop to reduce the converter's input capacitance. Since the same buffer is used during signal acquisition and conversion, its nonlinearities are suppressed by the SAR loop and therefore a simple source follower can be employed to drive the sampling capacitance.

[1] [1a] LTC6362, [2a] LT6350, [3a] LT1994, [4a] LTC6403, [5a] LTC6404, [6a] LTC6405, [7a] LTC6406, [8a] LTC6409, [9a] AD8476, [10a] ADA4932, [11a] ADA4899

Another challenge of high-speed, high-resolution SAR ADCs is the thermal noise during the ADC's conversion phase. SAR ADCs need high bandwidth, during that phase, to resolve 14 or more decisions in the given time window, which increases the total integrated noise. This can lead to a substantial power overhead to lower the noise spectral density. We address this issue by introducing noise filter gear shifting, which allows us to run the ADC 22% faster while keeping the power nearly constant. This technique is based on the concepts reported in [5], which exploits redundancy. However, the work of [5] needs two comparators to implement this technique leading to an increase in hardware complexity. This work, on the other hand, takes advantage of the noise properties of a g_m-C integrator (used as pre-amp), allowing us to use only one comparator instead of two.

1.2 Organization

The remainder of this thesis is organized as follows. Chapter 2 presents a brief introduction on SAR ADCs and surveys different ways of incorporating redundancy in the conversion algorithm. Chapter 3 introduces a block level description of the proposed SAR ADC using a loop-embedded input buffer. It also presents an overview of the ADC operation and a first-order noise calculation. Chapter 4 then describes circuit-level details and analyzes relevant nonidealities. Chapter 5 describes the test setup and measurements of a 35 MS/s, 75 dB SNDR, 99 dB SFDR prototype implementation [6] in a 40 nm LP CMOS process that serves as a proof-of-concept (see Figure 1.1). The concept of noise filter gear shifting along with implementation details and measurements is covered in Chapter 6. The experiments are based on a modified version of the design from Chapter 5. The described design achieves 77 dB SNDR at a conversion rate of 30 MS/s. Finally, Chapter 7 summarizes this work and provides suggestions for future work.

2 The SAR Algorithm

In this chapter, we briefly discuss the SAR algorithm, including how redundancy can benefit algorithm performance. We conclude this chapter by showing how redundancy can be used to speed up the SAR conversion considering a single pole settling behavior for the feedback DAC. The presented concepts are not new contributions; however, they provide important background information for the presented work.

2.1 An Introduction to the SAR Algorithm

The SAR algorithm is well known and has been used for many years in analog to digital converters. A nice summary of the operation and history of SAR ADCs can be found in [7]. The SAR algorithm is introduced using a scale example, similar to [7]. We want to measure an unknown weight between 0 to 16 kg with a maximum error of 1 kg. One question to ask is: What is the minimum number of weights and measurements required to perform the measurement? Using the SAR algorithm, the minimum number of weights and measurements (N_M) is achieved when binary scaled weights are used in a specific measurement sequence. We call the largest weight that can be measured a full-scale (FS) input. The largest measurement weight is identified as most significant bit (MSB) and the smallest is the least significant bit (LSB). We can calculate N_M using the following equation:

$$N_M = \log_2\left(\frac{FS}{LSB}\right) = \log_2\left(\frac{16\ kg}{1\ kg}\right) = 4 \tag{2.1}$$

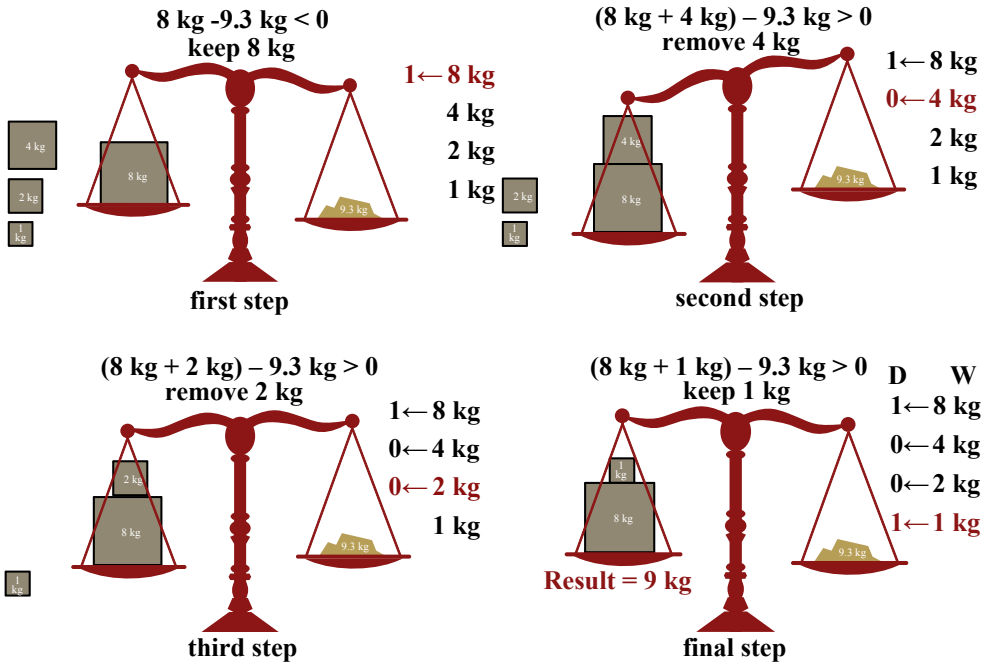

Figure 2.1: Binary weighted SAR algorithm, illustrated using a scale

Figure 2.1 shows an example measurement sequence (SAR algorithm) that gives the minimum number of measurements. As an example, consider a 9.3 kg weight. We measure 9 kg after four measurements, giving us a 0.3 kg quantization error, which is within our defined error margin. We could get closer by using an additional 0.5 kg weight (resulting in a reduced LSB); notice that to add another weight we require another measurement step, per (2.1). Figure 2.2(a) shows the

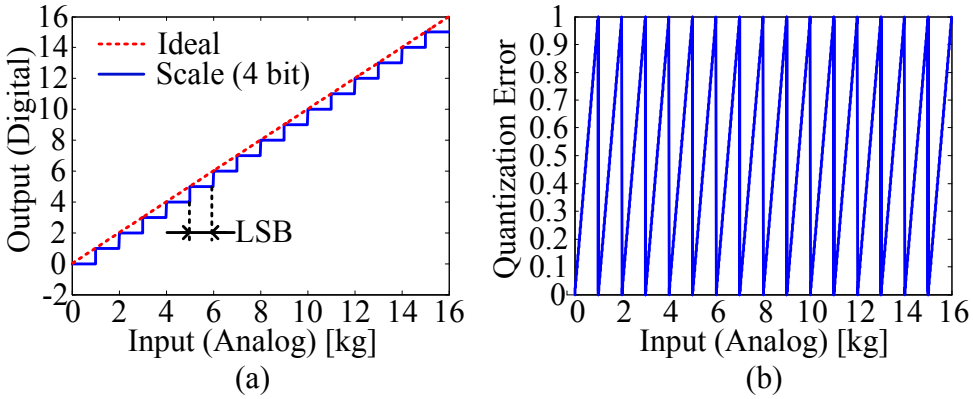

Figure 2.2: Scale transfer function (a) and quantization error versus scale input (b)

scale's "analog-to-digital" transfer function and its corresponding quantization error (Figure 2.2(b)). Notice that the described operation results in a quantization error that lies within 0 to 1 LSB. This is somewhat uncommon, since the quantization error is typically taken between -1/2 to 1/2 LSB, in particular for differential signals. However, the scale is single ended (no negative weights) and we use this convention to avoid dealing with (irrelevant) details on the definition of quantizer offset. Further details and a flow diagram of the SAR algorithm are presented in [8].

The use of a scale to implement the SAR algorithm makes sense intuitively, but we want to use this algorithm for an analog-to-digital converter, designed in a CMOS process. We can convert the scale into a block diagram (Figure 2.3), which can then be used to define circuit blocks that perform each operation. The right side of the scale, where we place the unknown weight, becomes a track-and-hold block. This block samples the input signal, typically voltage, and freezes it in time so that we can run the SAR algorithm. The weights and the left side of the scale are realized using a digital-to-analog converter (DAC). Notice that the algorithm is performed by a digital circuit (SAR algorithm block) and hence we have to interface to the analog world via a DAC to create the binary-weighted references. The main body of the scale becomes a summing node and comparator, which subtracts the input value from the output of the DAC and evaluates the sign of this result. In other words, the comparator determines if the input side is heavi-

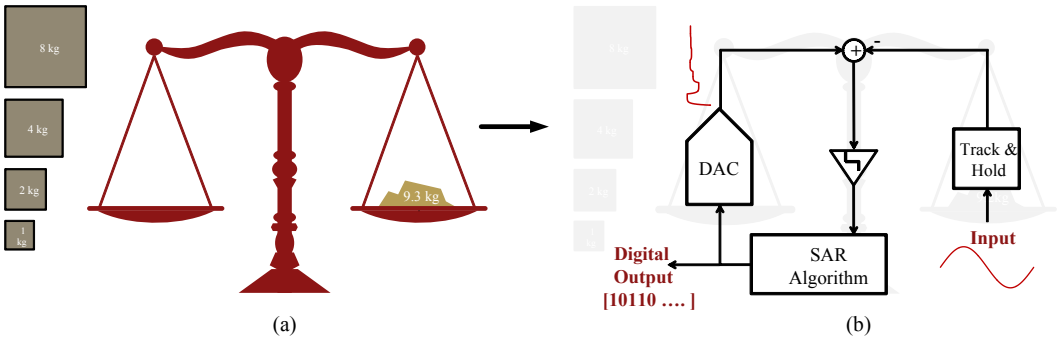

Figure 2.3: From the scale (a) to the SAR ADC block diagram (b)

er (scale example) or larger (voltage example) than the reference side. The result of this decision is fed into the digital SAR block, which runs the described algorithm and sets the new comparison voltage via the DAC.

Figure 2.4(a) shows the SAR ADC block diagram, which will be used in the remainder of this work. Furthermore, Figure 2.4(b) shows the DAC output waveform versus time, performing the same example as in Figure 2.1, but this time the signal and weights are voltages (V_{in} is 9.3 V, FS = 16 V and LSB = 1 V). Furthermore, the residue voltage V_{res} versus time is plotted in Figure 2.4(c), showing that the comparator only has to find the sign of that voltage. The actual value of V_{res} is not needed to perform the SAR algorithm. Notice that the comparator waits a certain amount of time until the next decision is made. This ensures that the DAC can reach its desired value and is stable during the time the comparator needs to make a decision.

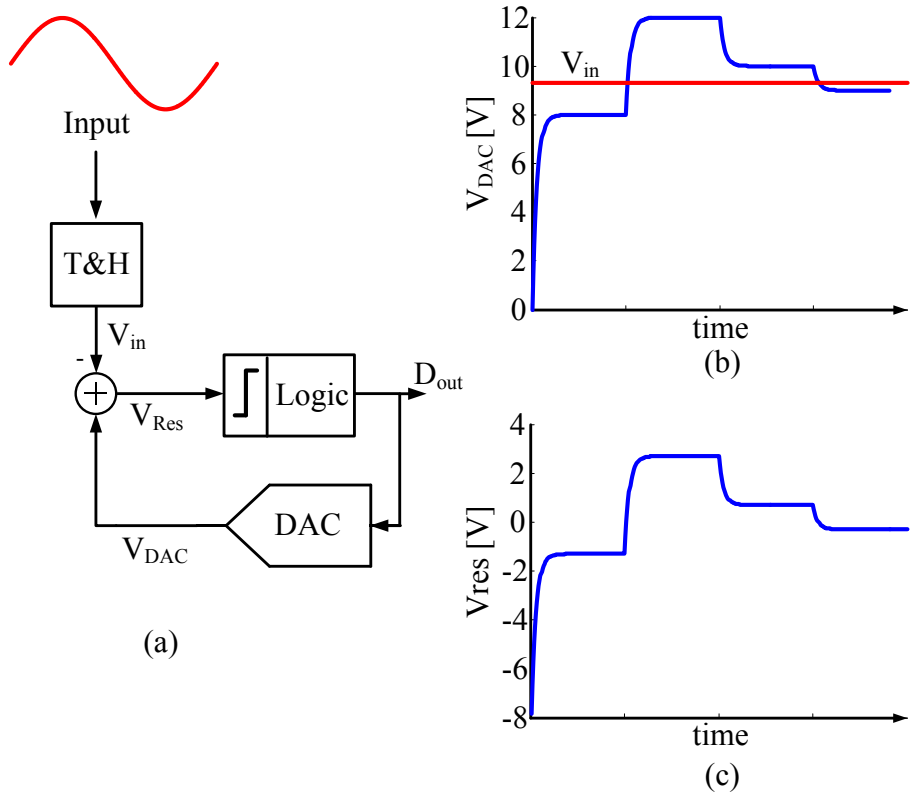

Figure 2.4: (a) SAR ADC block diagram, (b) V_{DAC} and (c) V_{res} versus time

Finally, we introduce some notation. Under the same example presented in Figure 2.1, we can write the weights as a vector W_b = [8 4 2 1] kg. We additionally store the results of each scale measurement in another vector D = [1 0 0 1] (zero for removing a weight and one for keeping the weight), which becomes the digital output. These two vectors are used to calculate the result of our measurement:

$$W_b \cdot D^T = (8 \cdot 1 + 4 \cdot 0 + 2 \cdot 0 + 1 \cdot 1)\, kg = 9\, kg \qquad (2.2)$$

Many different circuit implementations exist for the SAR ADC. We will not go into details of these architectures here; the interested reader can start with reference [3], which lists all ADCs papers of the last several years that were published at the ISSCC and the VLSI Circuit Symposium (including SAR ADCs).

2.2 Redundancy in SAR ADCs

Redundancy is a well-established concept that has been used in SAR ADCs for a long time. A comprehensive review of redundancy in SAR ADCs is given in [9]. We will briefly explain the concept and use it afterwards to show how it can help speed up the SAR conversion. This has been demonstrated for example in [10], [11].

One important aspect is that the SAR algorithm does not have to use a binary weight reference vector W. For example we could use the following vector W_{16} = [1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1] and still run the SAR algorithm. Figure 2.5(a) shows an example for a mid range input using W_{16}. It can be seen that the SAR algorithm converges within the defined accuracy. Unfortunately, using this weight vector we need 16 measurements (instead of 4) to ensure that we can reach values close to FS (Figure 2.5b)). In other words, using binary weights only minimizes the number of measurements, but the SAR algorithm can still converge for non-binary weights (given certain conditions that will be explained below). Using additional weight elements to make extra (redundant) decisions during the conversion process is termed redundancy.

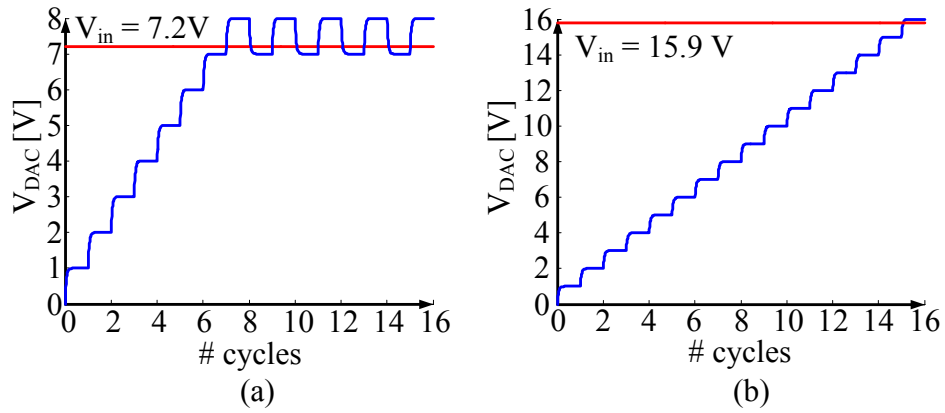

Figure 2.5: SAR ADC waveform versus # cycles with a non-binary weighted DAC W_{16} (a) V_{in} = 7.2 V, (b) V_{in} = 15.9 V (close to FS)

Redundancy allows us to correct for decision errors that happened during early decisions. We will show this by investigating two cases, one without redundancy (binary weighted W_b) and the other with redundancy (W_R). Both cases are assumed to make a decision error during the first cycle. For the purpose of this example, we do not care where this error comes from. However, the error only appears during the first step, all remaining decisions are correct. The redundancy is implemented by adding an additional binary weighted step, giving us W_R = [8 4 4 2 1]. The outcome is shown in Figure 2.6. While (a) cannot recover from the decision error, the case with redundancy (b) can. We have used the same example as

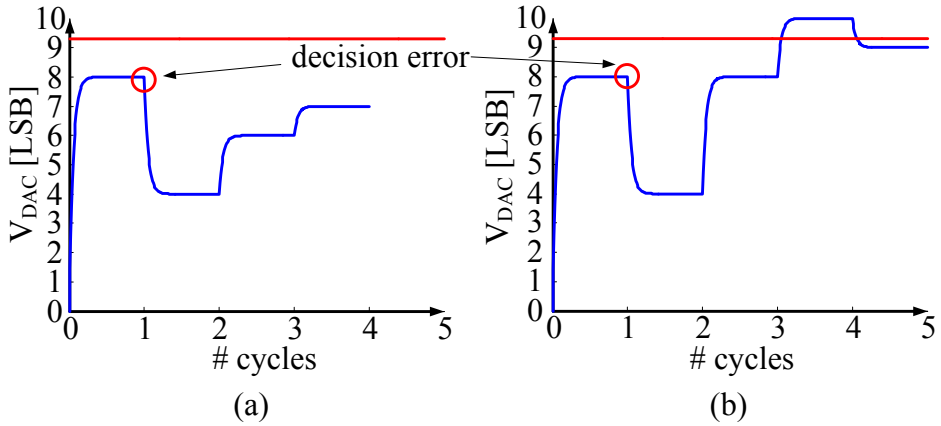

Figure 2.6: Example with decision error during the first cycle, (a) without redundancy (b) with redundancy

in Figure 2.4, however with an error during the first cycle. Notice that we pay a price for the ability to recover, which is an additional measurement.

We can calculate the maximum error from which we can recover in any given step when redundancy is present. For the k^{th} step of the SAR algorithm, we call this the max recoverable error of the k^{th} step q(k). This calculation will lead to a general formula, derived in [12]. Figure 2.7(a) shows a graphical solution that was constructed in the following way. The introduced error is the maximum allowed value, q(1). However due to the wrong decision in the 1^{st} step, we increased that error by the following test voltage $W_R(2)$. To recover from this error, all remaining W_R values ($W_R(3)$ to $W_R(5)$) have to sum up to $q(1)+W_R(2)$, to compensate. We add one more LSB to that, which is the allowed quantization error. Notice that we use here again a quantization error range from 0 to 1 LSB, to be consistent with [12]. This gives us:

$$q(1) + W_R(2) = W_R(3) + W_R(4) + W_R(5) + LSB \tag{2.3}$$

$$q(1) = -4 + 4 + 2 + 1 + 1 = 4 \tag{2.4}$$

We can generalize this in the following way [12]:

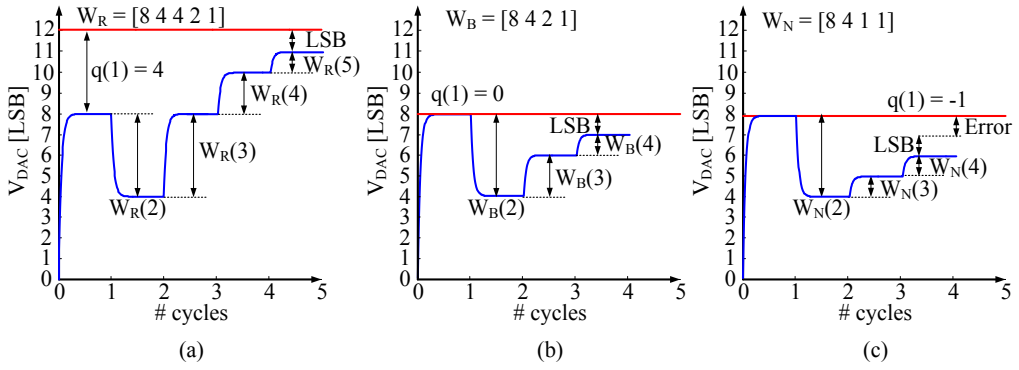

Figure 2.7: Graphical solution to find q(1): (a) case with redundancy, (b) binary weighted case, (c) non-binary weighted case resulting in negative q(1)

$$q(k) = -W(k+1) + LSB + \sum_{n=k+2}^{M} W(n) \quad (2.5)$$

where M is the length of the vector W. Figure 2.7(b) shows that the binary weighted case (W_B) has q(1) equal to zero, meaning that we cannot allow any error during the first decision, and in fact this is also true for all other decisions. Every decision in a binary weighted SAR has to have LSB-level accuracy. Notice that we said that the SAR algorithm will converge within an LSB for non-binary weighted Ws; this is only true as long q(k) is equal or greater than zero. The SAR algorithm will not converge with in an LSB for all codes if any of the q(k) are negative (Figure 2.7 (c)).

The error seen in our example, which caused the wrong decision, needs a little more attention. It is important to see that this error disappears after the first cycle, because using (2.5) for k = 2 and W_R yields q(2) = 0. This implies that the redundancy is now used up and if the error is still present during the second cycle, we cannot recover. This limits the approach to error sources that behave in this way. However, the DAC settling behavior falls within this category, as will be discussed next.

2.3 SAR Conversion Speed-up of a Single-Pole DAC using Redundancy

We mentioned that the DAC needs a certain amount of time to reach its final value; this is caused by limited DAC bandwidth. For high-resolution converters, this is typically a large part of the time that is needed to run the SAR algorithm. Leveraging redundancy can help to improve this [12]. We can model the settling behavior of the DAC with a single pole. The code-dependent output waveform as a function of time can be written as:

$$V_{DAC}(t) = \sum_{p=0}^{M-1} W(p+1)D(p+1)\left[1 - e^{\left(-\frac{t-pT_C}{\tau}\right)}\right] s(t - pT_C) \quad (2.6)$$

where W and D are the previously defined vectors for the reference values and DAC code, T_C is the time allowed for a single cycle, s(t) is the step function, τ is the time constant related to the single pole of the DAC, and M is the length of the vectors W and D. Furthermore, we can write the time-dependent settling error as:

$$V_{set\ error}(t) = \sum_{p=0}^{M-1} W(p+1)D(p+1)e^{\left(-\frac{t-pT_C}{\tau}\right)} s(t-pT_C) \qquad (2.7)$$

Figure 2.8(a) shows the plot of (2.6) for two different DAC time constants, resulting in two different numbers of settling time constants ($N_S = T_C/\tau$). Notice that we set D to one for all entries in this example, meaning that we settle to full scale. This is the worst case regarding settling. Figure 2.8 indicates that we introduce errors by not allowing complete settling for the DAC. Figure 2.8(b) shows the settling error versus time for $N_S = 2.85$, demonstrating that in fact the first few decisions are not settling within an LSB. This implies that if the DAC doesn't have enough bandwidth, we can cause incorrect comparator decisions, thus compromising the SAR algorithm. To ensure proper operation for a binary weighted SAR ADC, we can either lower τ of the DAC or we have to increase T_C, so that the DAC can settle to within an LSB. Decreasing τ leads to higher DAC power and increasing T_C slows down the SAR conversion; both are undesirable conse-

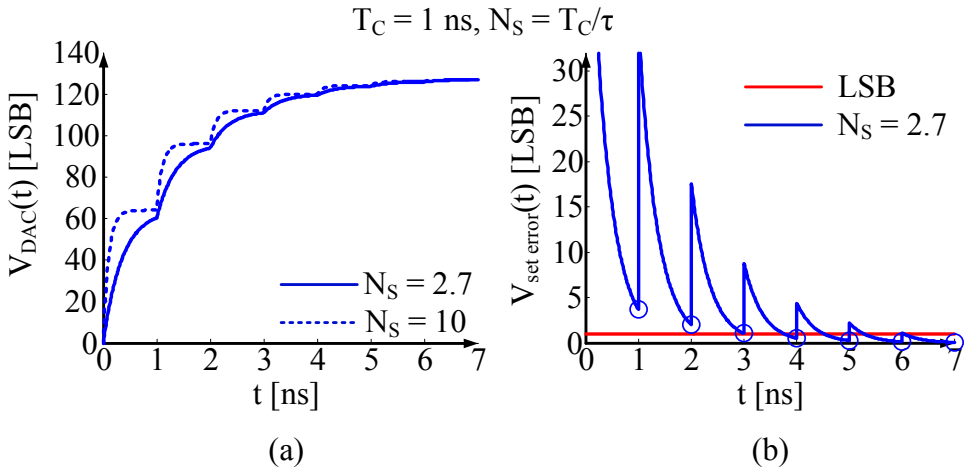

Figure 2.8: (a) DAC voltage (normalized to LSB) versus time for two different numbers of settling time constants (b) Settling error (normalized to LSB) versus time

quences.

However, a closer look at Figure 2.8(b) reveals that only the first few error steps are not settling within an LSB, meaning the settling error vanishes over time. This fact opens the door for the redundancy approach; we can allow settling errors during the early stage of the SAR conversion, because we can recover from small dynamic errors. Extra conversion cycles are required to implement redundancy, and therefore one must be careful to ensure that a real net speed up is achieved.

We will demonstrate the basic idea using a 14 bit example: We need $N_S \cong 10$ to achieve 14-bit settling, furthermore we set $T_C = 1$ ns giving us a conversion time $T_{conv} = T_C \cdot 14 = 14$ ns, also resulting in $T_C/NS = \tau = 100$ ps. Let us now assume that we can tolerate $N_S = 5$ by investing one additional cycle, which would lead to a new cycle time $T_{CR} = 500$ ps (using the same DAC), giving us $T_{conv} = T_{CR} \cdot 15 = 7.5$ ns! We could therefore run the conversion nearly twice as fast, or save power in the DAC. This technique was analyzed in [12]. Notice that redundancy tends to be more helpful for higher resolutions, and it is not as powerful for lower resolution SAR ADCs [13]. The reason is the overhead due to the comparator and digital delay time, which are more or less constant for a given technology. We assumed in the above example that this time is much smaller than the DAC settling time and it was therefore ignored. However, for lower resolutions this is not the case anymore and hence we only get small or no speed improvement by adding additional cycles.

We will end the review of redundancy here. Appendix (A1) gives more details on how we optimize and implement the redundancy (how many additional steps are needed, and how the redundancy range should be distributed in our design). The take-away from this chapter is that redundancy is quite powerful for high resolution SAR ADCs and is used in this design space quite often.

2.4 Summary

This chapter started with a quick review of the binary weighted SAR algorithm and introduced a general block diagram. We continued our discussion by introducing the non-binary weighted SAR algorithm and demonstrated that it can be used to incorporate redundancy into the SAR algorithm. Finally, we explained that redundancy can be used to speed up the SAR conversion time by recovering from settling errors in the early SAR cycles.

3 SAR ADC Using a Loop-Embedded Input Buffer

In this chapter, we introduce the loop-embedded input buffer to address the drive issue discussed in Chapter 1. First, we will review the drive problem in more detail and discuss previously published solutions, followed by an introduction to our approach. After that, we develop a behavioral model that allows us to extract design equations and insight for several building blocks. Thereafter, we focus on the operation of the ADC and its noise performance.

3.1 The Drive Problem and Previous Work

In Chapter 1 we highlighted the ADC drive challenge for high-resolution and high-speed SAR ADCs. In general, the problem arises from the fact that a large sampling capacitor needs to acquire a sample in a short time window (often a few nanoseconds). Furthermore, since both conversion and tracking must occur within the conversion period T_s, this window decreases as we increase the sampling frequency $f_s = 1/T_s$. The large sampling capacitance is primarily dictated by noise requirements, and resolutions of 14 bits and higher lead to values on the order of 3...10 pF. The combination of these two requirements puts significant pressure on the driving circuit. This challenge is typically addressed either by time interleaving (to allow for extra time) or spending power to implement an adequately strong buffer that drives the ADC.

Kapusta et al. [1] used a time-interleaved architecture using two ADC channels to increase the tracking time substantially. While one sub-ADC performs its conversion, the other converter is used to sample the input signal, hence giving nearly a full ADC cycle to track the signal. However, time interleaving artifacts (skews, offset) now have to be dealt with, and this can be very difficult for SFDR values higher than 90 dB.

The alternative option is to buffer the input, as commonly done in commercial standalone SAR ADCs [14]. The buffer is then typically housed in a separate IC (often in a bipolar process) and tends to consume more power than the ADC itself. In CMOS SoC applications, this issue is exacerbated, since the buffer must be integrated on the same chip, and it therefore cannot benefit from process options available for standalone parts

The source follower (SF) topology is one candidate for a wideband ADC buffer that can be integrated on the same die. However, one big challenge of the SF is its relatively poor linearity, which is due to the limited intrinsic gain ($g_m r_o$) in modern processes. Doris et al. [4] addressed this issue and showed that it is possible to linearize a buffer by placing it inside the SAR ADC's feedback loop. In this scheme, the buffer first appears in the forward path (during the tracking phase) and is then used in the feedback path during conversion (Figure 3.1(a)). This leads to a cancellation of the nonlinearity, which will be explained in more detail below.

The design presented in [4] uses the buffer to re-sample the signal into a sub-ADC, since it is a time-interleaved architecture. This means that there is a track and hold (T&H) in front of the buffer and hence this ADC still presents a large switched capacitance at its input. We extended the idea from [4] and removed the input T&H, allowing us to sample after the buffer. This leads to a buffered ADC with a well-behaved input impedance during tracking (Figure 3.1(b)), which is easy to drive. The nonlinearity of the SF is cancelled by the feedback action of the SAR loop during the conversion, similar to [4]. As a next step, we describe the ADC behavior at the block level and investigate various nonidealities.

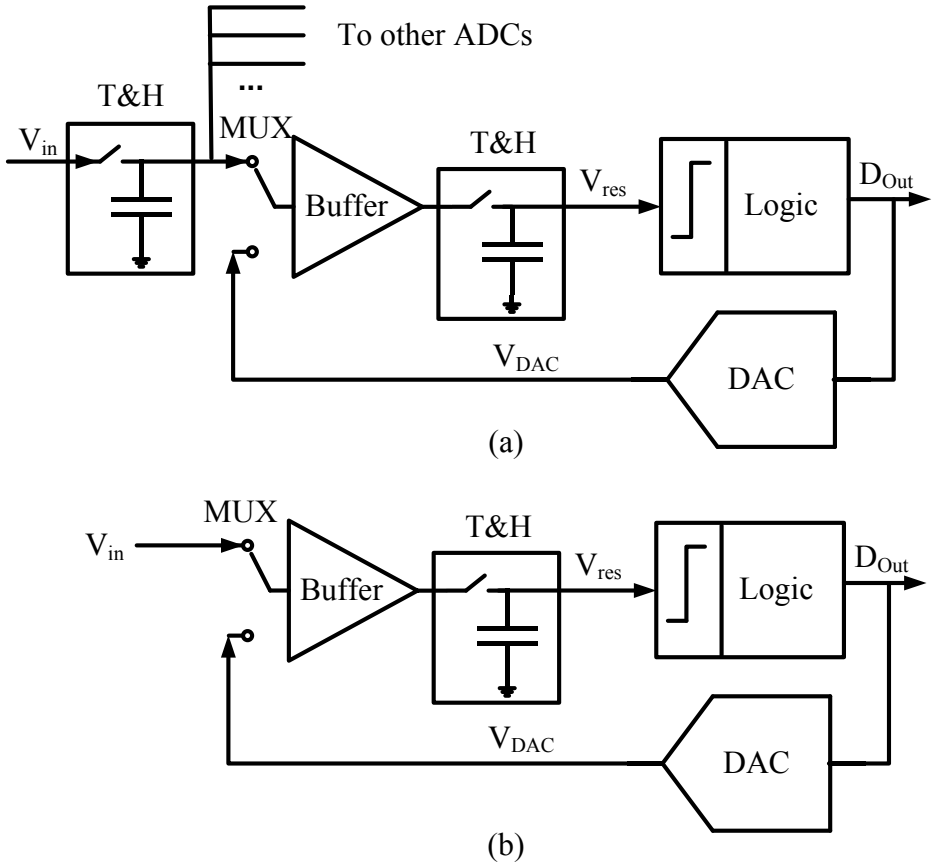

Figure 3.1: Top-level ADC block diagram of [4] (a) and,

(b) This work with removed input T&H

3.2 ADC Modeling and Design Equations

We will now develop several behavioral models of the proposed ADC architecture. This will help us understand the nonlinearity cancellation and gain a deeper insight into possible problems that must be addressed at the circuit level.

Figure 3.2 shows the basic model of the loop-embedded input buffer ADC. The input signal (V_{in}) passes through the nonlinearity A(V) first, modeling the buffer during tracking, and is then compared with the DAC signal that was passed through another nonlinearity B(V), modeling the buffer during the conversion phase. Ideally, A(V) = B(V), since we use the same buffer during tracking and conversion phases, but, as we shall see, this condition cannot be guaranteed and

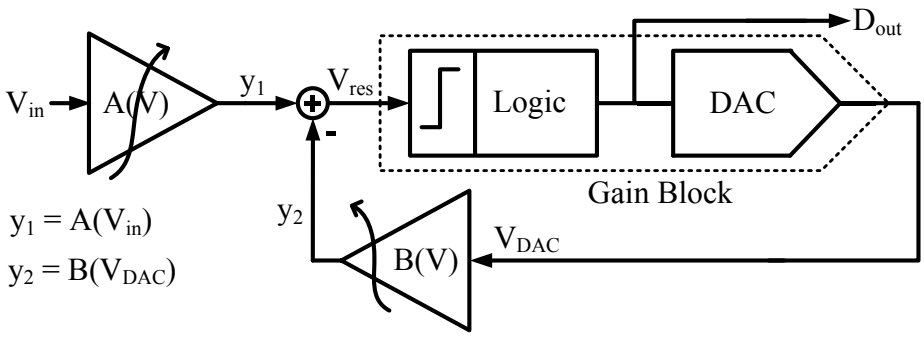

Figure 3.2: Basic ADC model

we will therefore investigate the general case A(V) ≠ B(V) as well. Furthermore, the SAR logic will drive the V_{res} voltage within ±1 LSB at the end of the conversion, and since we are designing for a 14-bit ADC it is reasonable to assume that V_{res} approaches zero, which leads to the following approximation:

$$y_1 - y_2 = V_{res} \approx 0 \rightarrow y_1 = y_2 \tag{3.1}$$

where y_1 and y_2 are the distorted input and DAC voltages given in Figure 3.2. This equation essentially describes a feedback system with infinite gain. Note that in this case the logic and DAC can be modeled as a gain block as indicted in Figure 3.2.

Before the mathematical solution is given, we will spend some time on building intuition. In order to do this, we will use a graphical approach to show the operation of the nonlinearity cancellation. In Figure 3.3(a) we plot the DC transfer characteristic of the two buffer nonlinearities A(V) and B(V) and assume here that they are equal. Furthermore, we show the situation after the DAC is set to produce the MSB reference voltage. The y-axis shows the output voltages y_1 and y_2 of the buffer models during tracking and conversion $A(V_{in})$ and $B(V_{DAC})$, which are used to create the residue voltage V_{res}. Using (3.1) and letting V_{res} approach zero, we see that V_{DAC} approaches V_{in}. Thus, we see that the nonlinearity of the buffer does not influence the ADC output, assuming that the digital output code produc-

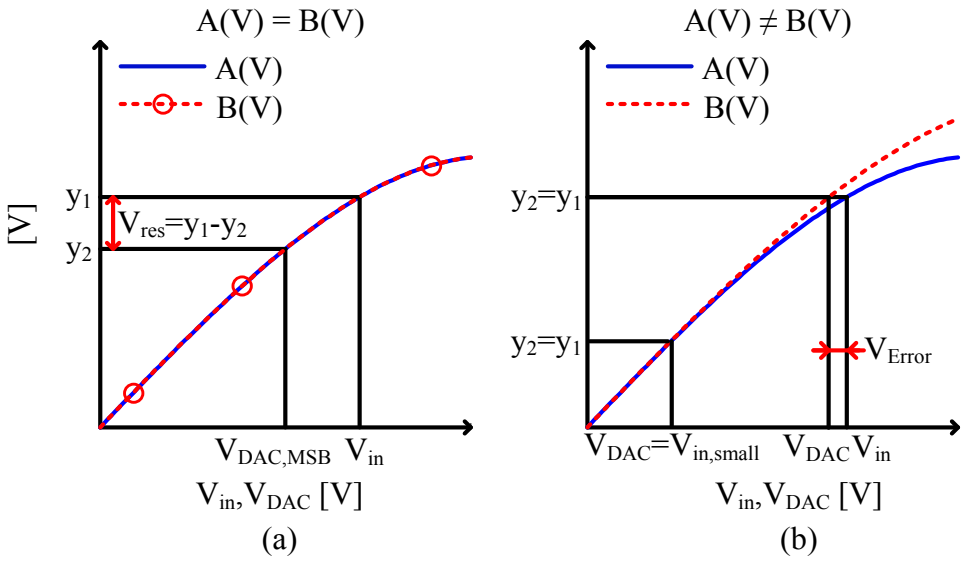

Figure 3.3: (a) V_{in} and V_{DAC} versus y_1 and y_2 for A(V)=B(V). (b) V_{in} and V_{DAC} versus y_1 and y_2 for A(V)≠B(V)

es a linear DAC voltage. This approach works perfectly only if the buffer has the same nonlinearity behavior during the two ADC phases.

Figure 3.3(b) illustrates the case when this condition is not met (A(V)≠B(V)). Notice that we already show here the situation when $y_1=y_2$. We can clearly identify an error voltage (V_{Error}) between the input and the DAC voltage. Furthermore, this error voltage is signal dependent. For low input voltages ($V_{in,small}$), the two nonlinearities are similar and the error voltage is smaller. This results in a nonlinear input to output relationship. The buffer should have the same nonlinearity behavior during tracking and conversion.

We can analyze the situation in Figure 3.3 using the block diagram of Figure 3.2. First, we define the two nonlinear functions A(V) and B(V):

$$y_1 = A(V_{in}) = a_1 V_{in} + a_3 V_{in}^3 \tag{3.2}$$

$$y_2 = B(V_{DAC}) = b_1 V_{DAC} + b_3 V_{DAC}^3 \tag{3.3}$$

A Taylor series approach is used to describe the nonlinear behavior of the buffer to keep the approach general. Furthermore, we only consider a third order distor-

tion, since the final circuit is operated fully differentially, leading to sufficient suppression of even-order nonlinearities. We are interested in the input to output relationship of the block diagram shown in Figure 3.2, where we define the output voltage as V_{DAC}. This can be calculated using a series reversion on (3.3) and using (3.1 and 3.2):

$$V_{DAC} = \sum_{n=1}^{\infty} B_n y_2^n \xrightarrow{y_2=y_1} = \sum_{n=1}^{\infty} B_n {y_1}^n = \sum_{n=1}^{\infty} B_n (a_1 V_{in} + a_3 V_{in}^3)^n$$
$$B_1 = \frac{1}{b_1}, B_2 = 0, B_3 = \frac{-b_3}{b_1^4}, B_4 = 0, \ldots \tag{3.4}$$

Equation (3.4) is now expanded, and we collect the significant terms. Higher order terms are ignored, arguing that these are small. This gives us the simplified transfer characteristic of the ADC:

$$V_{DAC} \approx \frac{a_1}{b_1} V_{in} + \left(\frac{a_3}{b_1} - \frac{a_1^3 b_3}{b_1^4} \right) V_{in}^3 \tag{3.5}$$

We can use (3.5) to analyze the situation in Figure 3.3(a), setting the coefficients a_k=b_k , which implies A(V)=B(V). This results in the known linear relation V_{DAC} =V_{in}.

We are more interested in the non-ideal case were the buffer sees different behavior, meaning the coefficients of A and B do not match. We define the third harmonic distortion (HD_3) of the ADC using (3.5) with $\hat{v}_{in}$ as signal amplitude:

$$HD_{3\,ADC} = \frac{1}{4} \frac{\left(\frac{a_3}{b_1} - \frac{a_1^3 b_3}{b_1^4} \right)}{\frac{a_1}{b_1}} {\hat{v}_{in}}^2 = \frac{1}{4} \left(\frac{a_3}{a_1} - \frac{a_1^2 b_3}{b_1^3} \right) {\hat{v}_{in}}^2 \tag{3.6}$$

Furthermore, we denote the HD_3 of the buffer during tracking (using nonlinearity A(V)) as HD_{3A} and use HD_{3B} for the buffer during conversion (with nonlinearity B(V)). This gives two cases:

- Case I: b_1=a_1 and b_3=a_3+Δ

$$HD_{3A} = \frac{1}{4}\frac{a_3}{a_1}\hat{v}_{in}{}^2 \tag{3.7}$$

$$HD_{3B} = \frac{1}{4}\frac{a_3 + \Delta}{a_1}\hat{v}_{in}{}^2 \tag{3.8}$$

$$HD_{3\,ADC} = \frac{1}{4}(-\Delta)\hat{v}_{in}{}^2 \tag{3.9}$$

In this case, we assume that the coefficients a_1 and b_1 are equal, but the third-order term of B differs from that of A by the error factor Δ. Figure 3.4(a) shows the third harmonic distortion of the ADC versus the difference between A(V) and B(V) in dB. Here, HD_{3A} was parameterized with two different third order coefficients, as we can see that we can tolerate greater errors between the two buffer states when the initial buffer is already more linear (smaller a_3 term). From here, we can get a first indication how linear the buffer has to be in order to achieve a desired level of cancelation.

The second case introduces an error between the coefficients a_1 and b_1:

- Case II: $b_1=a_1+\Delta$ and $b_3=a_3$

Case II is plotted in Figure 3.4(b), showing that the error margin is much tighter

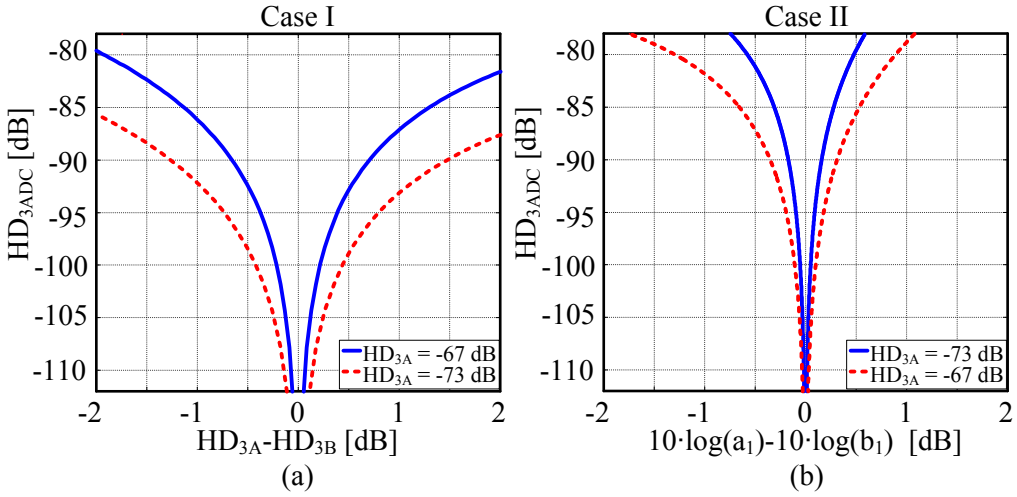

Figure 3.4: (a) Case I: HD_{3ADC} versus third order distortion tone difference between A and B (a_1=0.9, $a_{3\,-67dB}$=0.002 1/V^2, $a_{3\,-73dB}$=0.001 1/V^2). (b) Case II: HD_{3ADC} versus difference of the first order coefficients of A and B (a_1=0.9, $b_1=a_1+\Delta$, $a_{3\,-67dB}$=0.002 1/V^2, $a_{3\,-73dB}$=0.001 1/V^2)

than for Case I. We can only allow an error of less than 0.2-0.5 dB to get a decent amount of cancellation.

We arrived at a first piece of important insight: if we cannot guarantee matching buffer behavior between the two ADC phases, we have to design a more linear buffer. The design plots shown in Figure 3.4 quantify the needed linearity of the buffer. One important design task is to identify possible mechanisms that may alter the matching between the two buffer states (see Section 4.1.1).

3.2.1 Comparator Offset and the Effects on the Nonlinearity Cancelation

To make the results presented in the previous section more realistic, we must add comparator offset to our model. The comparator offset is not present during the tracking phase, however it will be visible during conversion, when the comparator block is used. We modify Figure 3.2 and add an offset to the summing node (see Figure 3.5).

We now have to update (3.1) as well:

$$y_1 - y_2 + V_{off} = V_{res} \xrightarrow{V_{res}=0} y_1 + V_{off} = y_2 \tag{3.10}$$

Two options are available for absorbing the offset, y_1 or y_2. Physically, the offset is added to y_2 and we use this fact also for the graphical illustration shown in Fig-

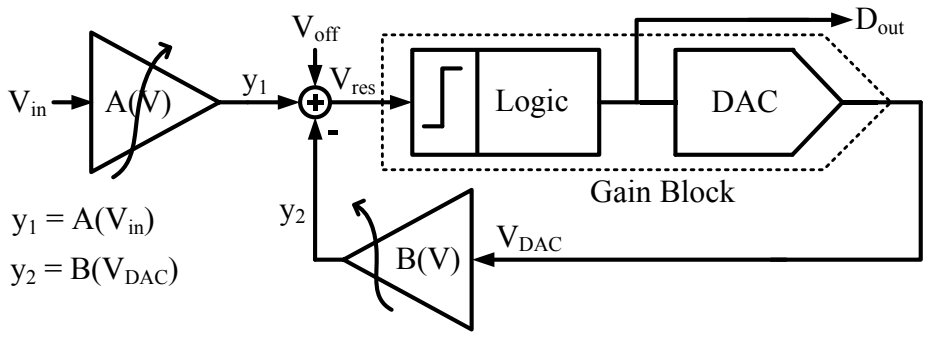

Figure 3.5: ADC block diagram with comparator offset V_{off}

ure 3.6. However, later we will add the offset to y_1, which is more convenient for analysis.

As before with the matching buffer condition, we can use a graphical illustration to show the impact of the offset on the linearity. Figure 3.6 shows this situation. As can be seen, for small inputs, the ADC output offset ($V_{off,out\ small}$) is equal to the input offset (we assume a buffer gain of 1), but for larger input values the output offset ($V_{off,out}$) gets larger due to the nonlinear behavior. It is now clear that the comparator offset can impact the linearity of the ADC and hence we will now analyze this effect in more detail.

We drive the analysis using (3.4), however, this time we use $y_2=y_1+V_{off}$ instead of $y_2=y_1$. This yields the following equation:

$$V_{DAC} = \sum_{n=1}^{\infty} B_n y_2^n \xrightarrow{y_2=y_1+V_{off}} \sum_{n=1}^{\infty} B_n\left(a_1 V_{in} + a_3 V_{in}^3 + V_{off}\right)^n \tag{3.11}$$

$$B_1 = \frac{1}{b_1}, B_2 = 0, B_3 = \frac{-b_3}{b_1^4}, B_4 = 0, \ldots$$

We assume $a_k=b_k$ to focus on the effect of the offset, and as before we expand

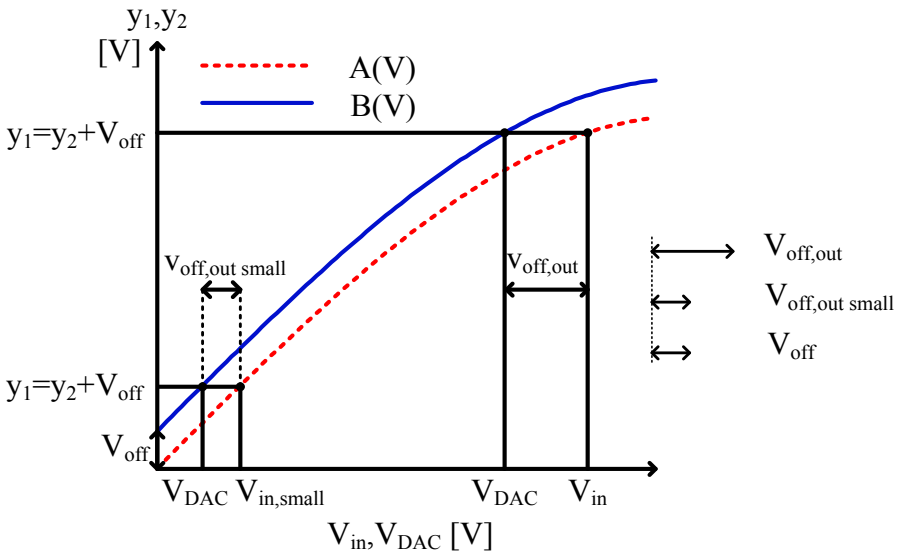

Figure 3.6: Impact of comparator offset on nonlinearity cancelation

(3.11) and ignore higher order terms:

$$V_{DAC} \approx -\frac{3a_3^2V_{off}^2}{a_1^4}V_{in}^3 - \frac{3a_3V_{off}}{a_1^2}V_{in}^2 + \left(1 - \frac{3a_3V_{off}^2}{a_1^3}\right)V_{in} + \frac{V_{off}}{a_1} - \frac{a_3V_{off}^3}{a_1^4} \tag{3.12}$$

We now apply one more simplification, by arguing that that $\frac{3a_3V_{off}}{a_1^2}$ and $\frac{V_{off}}{a_1}$ are large compared to the other coefficients in (3.12)[2], which gives:

$$V_{DAC} \approx V_{in} - \frac{3\left(a_3V_{off}\right)}{a_1^2}V_{in}^2 + \frac{V_{off}}{a_1} \tag{3.13}$$

A second-order term appears, caused by the offset. Notice that no second order terms are originally present in either A(V) or B(V). From here, it is feasible to use (3.13) and calculate the second-order fractional harmonic distortion of the ADC:

$$HD_{2ADC} = \frac{1}{2}\frac{3a_3V_{off}}{a_1^2}\hat{v}_{in} \tag{3.14}$$

Figure 3.7 shows the plot of (3.14) versus the comparator offset voltage, again parameterized with different HD_3 values for the buffer. The more linear the buffer is, the more offset voltage can be tolerated. For example, an offset voltage smaller than 2-3 mV is sufficient to ensure an SFDR > 90 dB for the assumed linearity range.

[2] Assuming $a_1 = 1$ (notice that a_1 is actually about 0.9, however it is more convenient to approximate it as 1 here), $a_3 < 0.01\frac{1}{V^2}$ and $V_{off} < 10\,mV$ we get $3a_3V_{off}^2 < 3 \cdot 10^{-6}$, $3a_3^2V_{off}^2 < 3 \cdot 10^{-8}\frac{1}{V^2}$ and $a_3V_{off}^3 < 1 \cdot 10^{-8}$ V, which are all smaller compared to $3a_3V_{off} = 10^{-4}\frac{1}{V}$

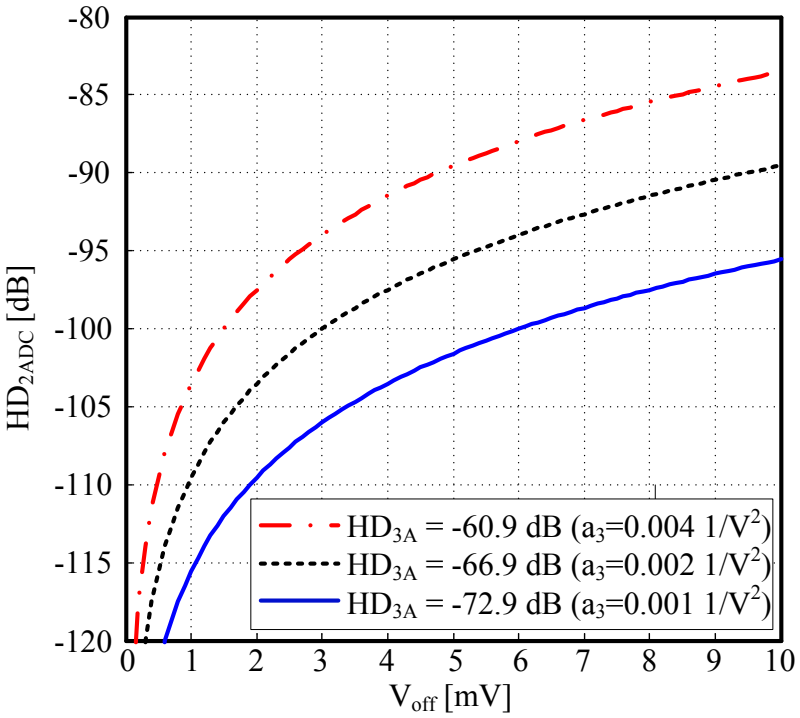

Figure 3.7: HD_{2ADC} versus comparator offset voltage V_{off} with HD_{3A} as parameter and a_1=0.9

3.2.2 Phase Imbalance Considerations and the Effect of Second-Order Suppression

The effects of imbalance in the ADC's differential input path are now analyzed. This effect is of importance, since the conversion from a single ended signal to a differential signal is not perfect. We use a transformer to create a differential signal to characterize our ADC, and hence we have to account for its imperfections. Notice that in a SoC application this effect might not be as important as in the transformer case, however it is still beneficial to understand how a nonideal differential signal influences the ADC.

Figure 3.8 shows a model of the single-ended to differential conversion [15]. $A_s(V)$ is the nonlinear model of the single-ended buffer and has the following form:

$$A_s(V) = a_1 V + a_2 V^2 + a_3 V^3 \tag{3.15}$$

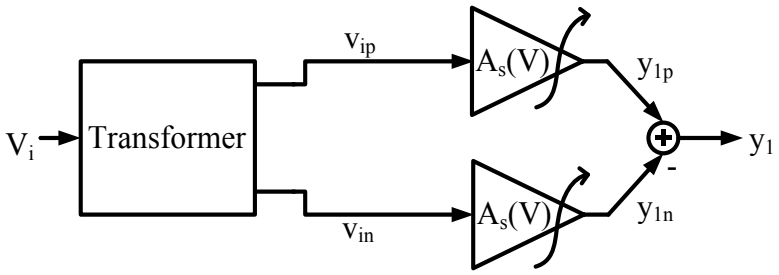

Figure 3.8: Differential model of the ADC input path using a transformer to convert a single-ended signal to a differential input.

We stated before that a perfectly differential signal will suppresses the even-order nonlinearities of A_s and hence y_1 would not have an a_2 coefficient or other even order terms. This assumption often leads to the misconception that the second-order term of a differential circuit does not matter. It is important to understand that the second-order term only cancels if the differential signal is perfectly balanced. This condition implies that the two signals v_{in} and v_{ip} have equal amplitude and are 180° out of phase (v_{ip}=f(t) and v_{in}=-f(t)). However, this constraint is not easily satisfied in a real-world single ended to differential conversion, since the transformer is not an ideal building block and it can suffer from frequency-dependent amplitude and phase imbalance [15]. The conclusion is that a second order term will always be present in a real world system, and hence this deserves further analysis.

We will only consider the phase imbalance problem, because it turns out to be the dominant error source [15]. Hence, we model v_{ip} and v_{in} as follows:

$$v_{ip} = \hat{v}_{in} \sin(\omega t) \tag{3.16}$$

$$v_{in} = -\hat{v}_{in} \sin(\omega t + \varphi) \tag{3.17}$$

We can derive $y_1(t,\varphi)$ and the corresponding fundamental amplitude ($Fund_{y1(\varphi)}$), $HD_{2y1(\varphi)}$, and $HD_{3y1(\varphi)}$ by using (3.15)-(3.17) and Figure 3.8 (see Appendix A2):

$$y_1(t,\varphi) = \left(2a_1\hat{v}_{in} + \frac{3a_3\hat{v}_{in}^3}{2}\right)\cos(\frac{\varphi}{2})\sin(\omega t + \varphi_1) - a_2\hat{v}_{in}^2\sin(\varphi)\sin(2\omega t + \varphi_2) - \left(\frac{a_3\hat{v}_{in}^3}{2}\right)\cos\left(\frac{3\varphi}{2}\right)\sin(3\omega t + \varphi_3) \quad (3.18)$$

$$Fund_{y_1(\varphi)} \approx 2a_1\cos(\frac{\varphi}{2})\hat{v}_{in} \quad (3.18a)$$

$$HD_{2y_{1(\varphi)}} \approx \frac{a_2\sin(\varphi)}{2a_1\cos(\frac{\varphi}{2})}\hat{v}_{in} \approx \frac{a_2}{2a_1}\sin(\varphi)\,\hat{v}_{in} \quad (3.18b)$$

$$HD_{3y_{1(\varphi)}} \approx \frac{a_3\cos\left(\frac{3\varphi}{2}\right)}{4a_1\cos\left(\frac{\varphi}{2}\right)}\hat{v}_{in}^2 \quad (3.18c)$$

Figure 3.9(a) plots HD_2 versus the phase imbalance of (3.18). As expected, for a zero phase error, HD_2 approaches zero. However, this suppression quickly goes away for larger phase errors. For example, if we have a buffer A_s with an HD_{2_As}=

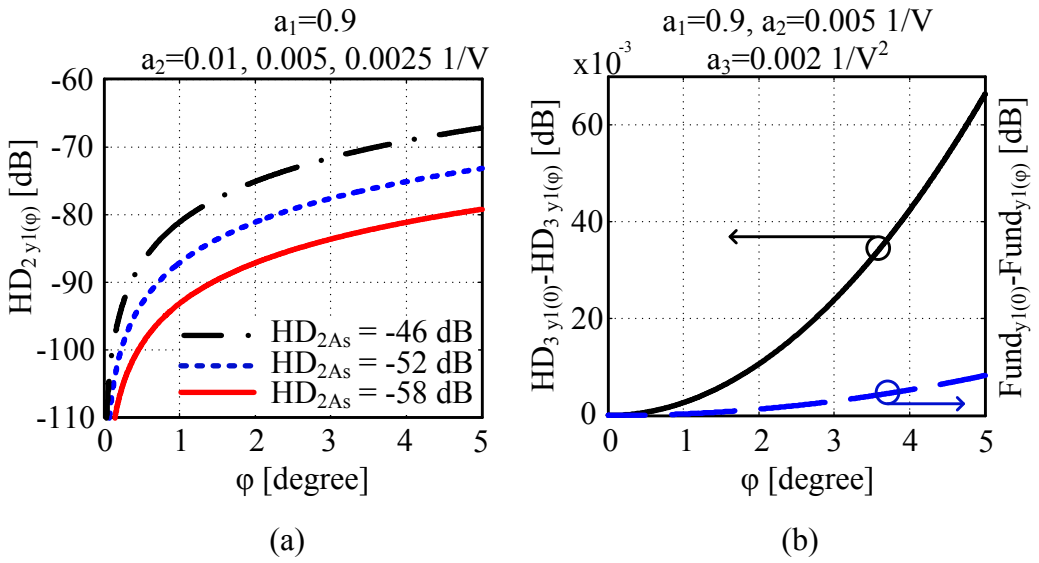

Figure 3.9: (a) HD_2 of $y_1(t,\varphi)$ versus phase imbalance φ and (b) change of HD_3 and Fundamental of $y_1(t,\varphi)$ versus phase imbalance referenced to φ=0

-58 dB, $y_1(t,\varphi)$ has an $HD_{2y1(\varphi)}$ of -90 dB for a phase imbalance of 1.5°. Interestingly, the third harmonic and fundamental are not altered by much, as shown in Figure 3.9(b). We only see a change of 0.06 dB in $HD_{3y1(\varphi)}$ for a phase error of 5°. We will use this and previous results to calculate the HD_2 of the complete ADC.

It is important to realize that the errors caused by the phase imbalance are present during the tracking phase of the ADC only, and are acquired on the sampling capacitor. The feedback path cannot reproduce this error, since the feedback DAC does not suffer from phase imbalance. This is equivalent to saying that the DAC has a stable common mode voltage. We are therefore confronted with the situation that the buffer has different nonlinearity behavior between the two ADC phases. We can use the results from Section 3.2.1 to understand how the phase error affects the buffer nonlinearity cancelation. First, the third-order cancelation is not affected, since the fundamental and third-order change between the two ADC phases are only about 0.06 dB and much less than the 0.5 dB (see Section 3.2.1, Figure 3.4), respectively. However, the second-order term is not cancelled. We can use the results derived in Section 3.2.2, noting that neither the offset nor the second-order nonlinearity is cancelled. Hence, similar analysis steps can be used, but instead of $y_2 = y_1 + V_{off}$ we use $y_2 = y_1 - a_2\hat{v}_{in}^2 \sin(\varphi)\sin(2\omega t + \varphi_2)$. By arguing that the nonlinear cancelation of a_3 is not affected and using (3.13), we can write:

$$V_{DAC} \approx V_{in} - \frac{3(a_3 a_2 \hat{v}_{in}^2 \sin(\varphi)\sin(2\omega t + \varphi_2))}{a_1^2} V_{in}^2 + \frac{a_2 \hat{v}_{in}^2 \sin(\varphi)\sin(2\omega t + \varphi_2)}{a_1} \tag{3.19}$$

where V_{in} is given by

$$V_{in} \approx 2\hat{v}_{in}\sin(\omega t + \varphi_1) \tag{3.20}$$

We can simplify (3.19) and argue again that the $a_3 a_2 \hat{v}_{in}^2 \sin(\varphi)$ term is very small, eliminating the fourth-order term. Now the HD_{2ADC} can be derived as a function of the buffer nonlinearity coefficients and the phase error:

$$HD_{2ADC} \approx \frac{1}{2}\frac{a_2}{a_1}\sin(\varphi)\,\hat{v}_{in} \tag{3.21}$$

Notice that HD_{2ADC} is equal to the $HD_{2y1(t,\varphi)}$ and hence Figure 3.9(a) corresponds to the HD_2 of the complete ADC as well. Equation (3.21) is the main design equation capturing the linearity requirements of the buffer for a given phase imbalance specification.

3.2.3 The Residue Voltage of the ADC

As a final step, we derive the residue voltage V_{res} (see Figure 3.2). We do this merely for completeness, since V_{res} has no influence on the final linearity of the ADC. However, it is important to understand that the residue voltage is distorted and that can become relevant if one tries to pipeline this SAR ADC.

Figure 3.10 shows the simulated residue plot of an example ADC (solid line), we used the condition A(V)=B(V) and we exaggerated the a_3 term to make the V_{res} distortion visible (a_1=0.9, a_3=-0.05). We can calculate the maximum error voltages by stepping through the LSBs:

$$\begin{aligned} V_{res} &= \frac{A(n\cdot LSB + LSB) - A(n\cdot LSB)}{LSB} \\ &\xrightarrow{V_{in}=n\cdot LSB} = \frac{A(V_{in}+LSB) - A(V_{in})}{LSB} \end{aligned} \tag{3.22}$$

Equation (3.22) gives us the circles plotted in Figure 3.10. It is interesting to realize that letting the LSB size approach zero gives in the limit:

$$V_{res} = \lim_{LSB\to 0}\frac{A(V_{in}+LSB) - A(V_{in})}{LSB} = \frac{dA(V_{in})}{dV_{in}} \tag{3.23}$$

This expression is shown as the dashed line in Figure 3.10. It gives us a quick and handy tool to calculate the residue distortion.

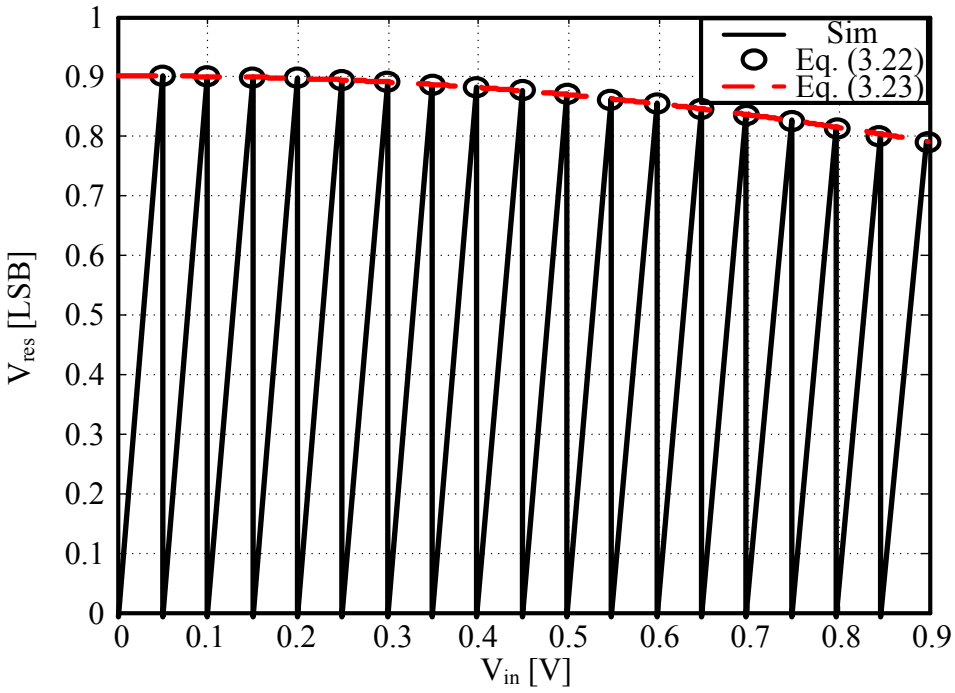

Figure 3.10: Simulated residue plot of an example ADC

This concludes the basic modeling aspects of the ADC. We will use the derived results in the following sections to justify some of the key design choices.

3.3 ADC Operation

In this section, we will introduce the actual ADC architecture with its main building blocks and describe the operation and timing of the ADC by going through the two main phases, tracking and conversion. This description still based on a block diagram; circuit design details follow later.

Figure 3.11 shows the ADC block diagram. The input signal is acquired on the sampling capacitors (C_S) via bottom-plate sampling [16]. The SAR conversion and DAC feedback are self-timed [17], while the low-jitter sampling pulse (ϕ_{1e}) is generated from a main clock input (see Figure 3.11). The feedback DAC uses a current-steering topology with non-binary weights and its reference current is derived from an on-chip bandgap (BG). Compared to the traditional charge redistribution architecture, this DAC doesn't require a low impedance reference volt-

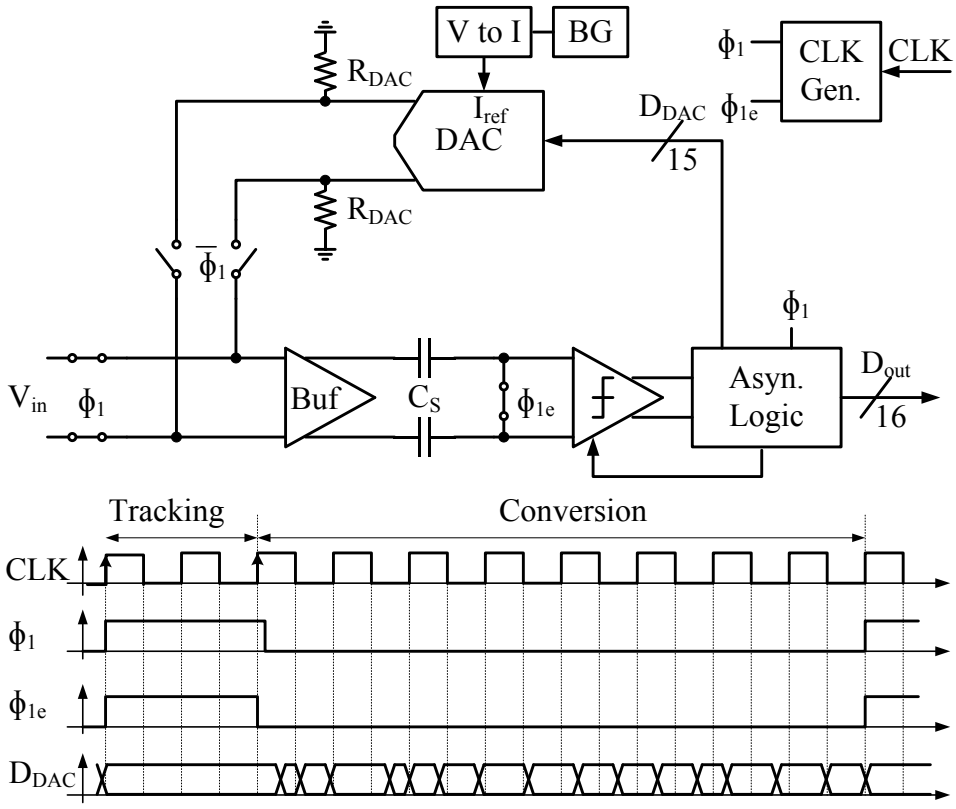

Figure 3.11: ADC Block diagram and timing

age and helps mitigate cross-talk issues due to its class-A nature; this is a critical aspect in SoC environments. The DAC is segmented into 8b MSB and 7b LSB sections. This creates two redundant SAR cycles, which helps alleviate the DAC settling requirements [1] [2]. The DAC MSB weights are measured at start-up using the LSB section similar to [2] [18]; this will be explained later (Section 4.2.2). This calibration allows us to match only the LSB section to 14 bits, while the MSB section is sized to stay within the redundancy range, ensuring a compact design. The measured weights are used to assemble the final binary output from the 16b raw data (D_{out}).

The pseudo-differential buffer consists of two source followers that are designed to settle to within a fraction of the 14-bit LSB during tracking. This leads to a (single-sided) input capacitance of approximately 200 fF, which is substan-

tially smaller than the 3.5 pF sampling capacitance and hence makes this ADC easy to drive.

3.3.1 The ADC Tracking and Sampling Phase

Figure 3.12 shows the differential input circuitry of the ADC during the tracking phase. The input signals (V_{ip} and V_{in}) are sampled after the buffer onto the sampling capacitors. The size of the sampling capacitors is purely based on noise considerations. It is of great interest to sample the signal after the buffer as linearly as possible, because each nonlinearity introduced by the tracking and sampling operation (and not the buffer itself) will lead to a mismatch between the tracking and conversion phase, and hence will limit the nonlinearity cancelation (see Section 3.2.1). It is counterintuitive, but we sample a distorted signal as linearly as possible in order to preserve the buffer nonlinearity. It should also be highlighted that the input capacitance of the comparator C_P (Figure 3.12) is voltage dependent and hence nonlinear. However, this is not relevant for the ADC's linearity performance, since it is always charged to the same common-mode voltage during tracking and we will show later that the capacitor nonlinearity is also irrelevant during the conversion phase.

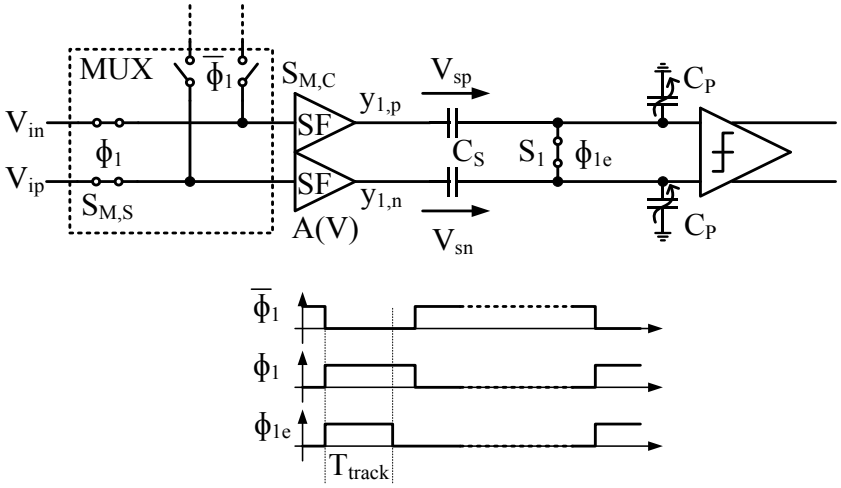

Figure 3.12: ADC during the tracking phase

The input signal has to pass through a MUX, since the input must be disconnected from the buffer allowing it to be used during the conversion phase. Notice that the signal fidelity right in front of the buffer (after the MUX switch $S_{M,S}$) plays a major part in the achievable linearity of the ADC. Any distortion introduced by the MUX switch is directly visible at the ADC output. This leads to the use of a bootstrapped switch to mitigate the nonlinear switch resistance [19]. Note that the small buffer input capacitance also helps to keep the distortion of the MUX switch small, since the MUX current is mainly defined by that capacitor.

3.3.2 The ADC Conversion Phase

The ADC is switched to the conversion phase after the input signal has been sampled. Figure 3.13 shows the ADC block diagram during that state. We define here all important timing signals and aspects of the ADC during that phase. First, we define the different time delays using Figure 3.13. After that, we will inspect the waveforms and timing of the first two cycles (see Figure 3.14). (It is straightforward to extend the operation to all remaining cycles because they will just repeat

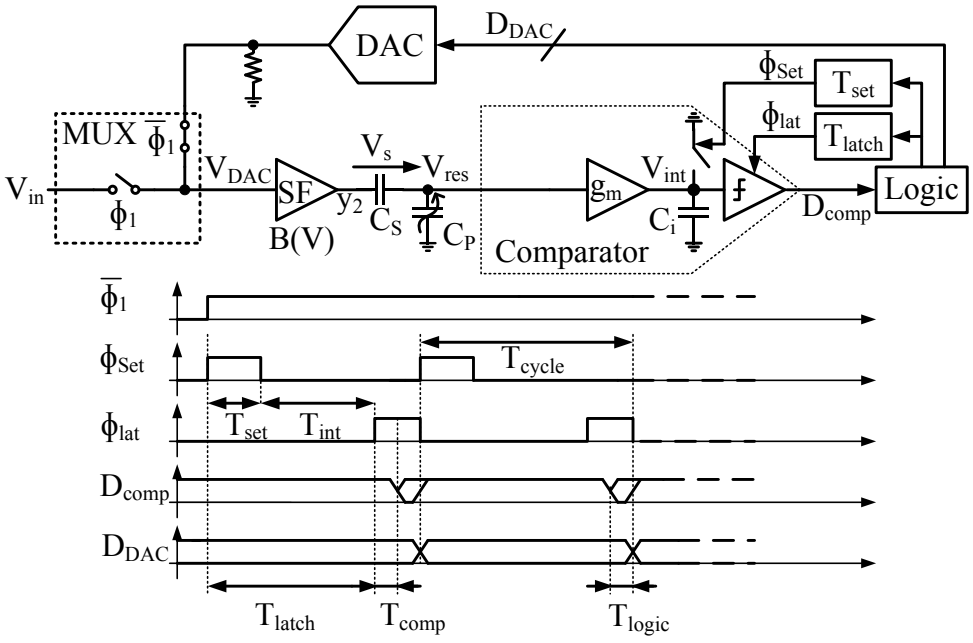

Figure 3.13: Block and timing diagram of the ADC during the conversion phase

in the same way.)

We use a self-timed scheme during conversion, as mentioned already. The cycle time (T_{cycle}) of this self-timed architecture is defined by a fixed but controlled delay time (T_{latch}), plus the time the comparator needs to make a decision (T_{comp}) and the logic delay (T_{logic}), giving us $T_{cycle}=T_{latch}+T_{comp}+T_{logic}$. The employed circuitry is nearly identical to that used in [20] and a more detailed description can be found in [21].

We show in Figure 3.13 a more detailed block diagram of the comparator, since it is an important building block and some critical timing signals must be explained. The comparator is made up of a pre-amplifier using an integrator and a high-speed latch. We will discuss the comparator in more detail in Chapter 4. The main difference of the self-timed logic compared to [20] [21], is that we have two delays, T_{set} and T_{latch}. These delays allow us to generate two important timing signals, the settling time (T_{set}) for the high-speed nodes V_{DAC} and V_{res}, and the integration time ($T_{int}=T_{latch}-T_{set}$), which controls the noise filter of the comparator (see Section 3.4.2). Both delays are programmable, allowing us to change the time allocations for DAC settling and comparator integration.

The nonlinear input capacitance of the comparator has no influence on the linearity of the ADC because we only detect the sign of V_{res} to implement the SAR algorithm (see details in Section 3.24). However, the capacitive divider formed by C_S and C_P attenuates V_{res} and this impacts the noise performance of the ADC. The attenuation is calculated as follows:

$$V_{res} = (y_2 - V_s)\frac{C_s}{C_s + C_P(V)} = (y_2 - V_s)G_C \tag{3.24}$$

$$\begin{aligned} V_s &= y_1 = A(V_{in}) \\ y_2 &= B(V_{DAC}) \end{aligned} \tag{3.25}$$

Here, the same convention as in Section 3.2 is used to describe the two buffer states.

Figure 3.14 shows the transient waveforms of the relevant nodes during the first two conversion cycles and for a given sampled input voltage. The current steering DAC is a tri-state DAC, which allows zero, positive and negative outputs. During the first cycle, we set the DAC to zero and V_{DAC} settles from V_{in} to zero. V_{res} tracks this voltage right away, since this is a high bandwidth node. The residue voltage has T_{set} to settle and after that, we start to integrate that voltage, resulting in V_{int}. This integration phase is used to filter the noise of the conversion phase and plays a major role in achieving the targeted the noise performance (see Section 3.4.2). The latch is fired after $T_{latch}=T_{set}+T_{int}$ and makes a decision on whether the residue voltage is positive or negative. The logic uses the comparator output to decide the sign of the next DAC value, and thereby performs the SAR algorithm. This operation takes $T_{comp}+T_{logic}$[3] completing one conversion cycle. We then move to the next cycle and steer the MSB current source of the DAC

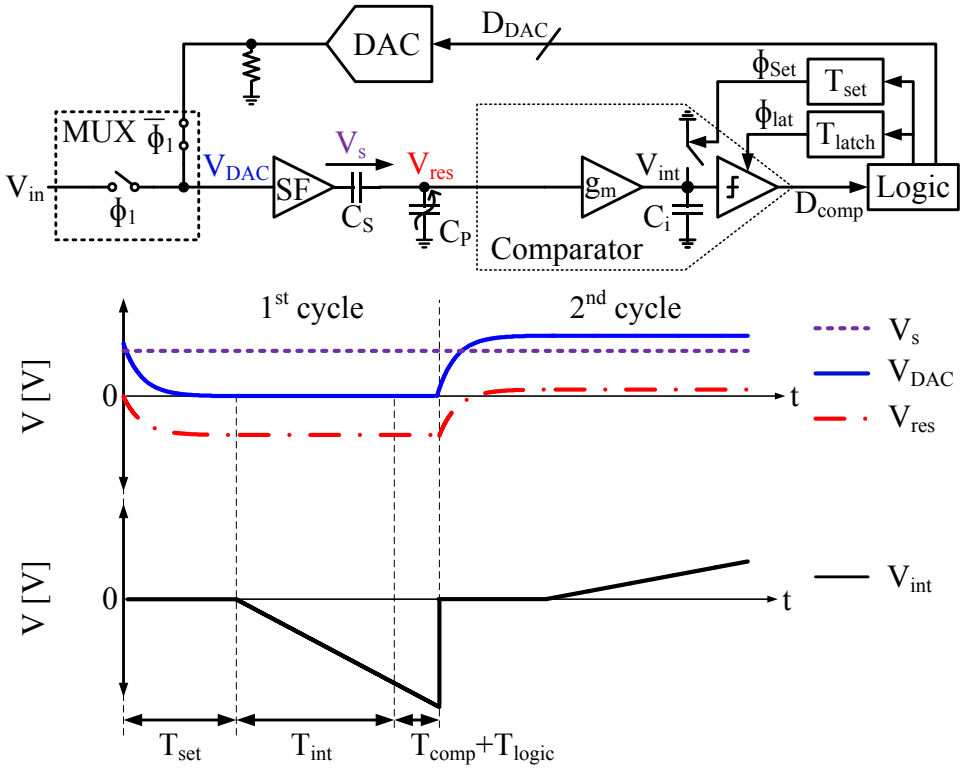

Figure 3.14: Waveforms for the first two cycles of the ADC conversion

[3] Note that the logic waits until the comparator makes a decision [21], hence T_{comp} depends on the input voltage of the comparator V_{res} (smaller V_{res}, longer T_{comp}).

according the previous comparator decision to a positive or negative value (see Figure 3.14), and repeat all the steps as in the first cycle. By repeating the cycles, we successively approximate the input voltage with the DAC voltage and drive the residue voltage to zero (see Section 2.1).

We can now calculate the conversion time of the ADC, which is $T_{ADC}=T_{track}+\#_{cycles}\cdot T_{cycle}$ and $f_s=1/T_{ADC}$. We use 16 cycles ($\#_{cycles}$) during conversion, as mentioned above, and two cycles for tracking, giving us $T_{ADC}=18\cdot T_{cycle}$. This completes the description of the basic ADC operation. Next, we will describe a first-order noise model for the ADC, which gives further insight on important block level decisions (like the comparator topology).

3.4 First-Order ADC Noise Model

We will present and derive in this section the first-order noise equation of the ADC, which can be used to calculate the peak signal-to-noise ratio (SNR):

$$SNR_{ADC}=\frac{\frac{1}{2}\hat{v}_{in}^2}{\overline{V_{ADC}^2}}=\frac{\frac{1}{2}\left(\frac{V_{FS}}{2}\right)^2}{\overline{V_{ADC}^2}} \tag{3.26}$$

where V_{FS} is the ADC's full scale voltage and $\overline{V_{ADC}^2}$ is the total input-referred noise of the ADC and can be written as:

$$\overline{V_{ADC}^2}=\overline{e_q^2}+\overline{V_{SN}^2}+\overline{V_{CN}^2} \tag{3.27}$$

Here, we consider three noise terms, the quantization noise $\overline{e_q^2}$, the sampling noise $\overline{V_{SN}^2}$, and the noise of the conversion phase $\overline{V_{CN}^2}$. We will analyze the noise in a similar way as we described the ADC operation. First, we will investigate the tracking and sampling phase and then derive the sampling noise, followed by the noise analysis for the conversion phase of the ADC. Even though (3.27) uses the input-referred noise (see Figure 3.15), we will later move the reference point to the V_{res} voltage node for convenience. As long we refer the input signal to the same point, the computed peak SNR stays the same.

The quantization noise term in (3.27) follows from the known relationship:

$$\overline{e_q^2} = \frac{LSB^2}{12} \tag{3.28}$$

where LSB is the voltage of the least significant bit ($LSB=V_{FS}/2^B$).

For the remaining noise terms, we consider a first-order noise model at the block level. The following sub-sections analyze the appropriate model during sampling and conversion.

3.4.1 Noise Analysis of the Tracking and Sampling Phase

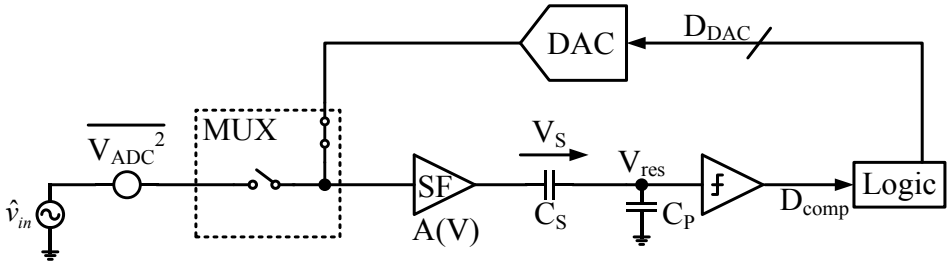

Figure 3.15: ADC noise model showing the input referred ADC noise

Figure 3.16 shows the noise model of the ADC during the sampling phase, and we will calculate the noise voltage ($\overline{V_{SN}^2}$) that is sampled onto the sampling capacitor C_S. This is a simplified, single-ended version of Figure 3.12 including the various noise generators. We identify three noise sources, the noise of the MUX resistor R_{MUX}, the noise of the sampling switch resistor R_{SW}, and the noise of the buffer. G_{SF} is the DC gain of the buffer.

We break the noise analysis into two parts, first we calculate the noise of the buffer and sampling switch that is sampled onto the sampling capacitor ($\overline{V_{SN_{SW}}^2}$). This is the well-known kT/C relation (Appendix A3):

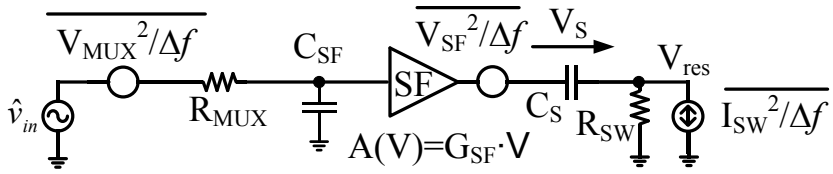

Figure 3.16: Half circuit noise model of the tracking and sampling phase (G_{SF} is the DC gain of the SF buffer)

$$\overline{V_{SN_{SW}}^2} = \frac{kT}{C_S} nf \tag{3.29}$$

where *nf* accounts for any excess noise from the buffer implementation.

Next, the noise due to the MUX switch can be calculated by realizing that the equivalent noise bandwidth f_N is mainly defined by the sampling network $f_N = \frac{\pi}{2}\frac{1}{2\pi R_{SW} C_S}$. Notice that the buffer input capacitance C_{SF} should be much smaller than the sampling capacitor C_S (C_{SF}«C_S), which is the fundamental idea of this work: minimizing the ADC input capacitor. Hence, for our first order calculation, we assume that C_{SF} can be neglected and we can write:

$$\overline{V_{SN_{MUX}}^2} = \frac{4kTR_{MUX}G_{SF}^2}{4R_{SW}C_S} = \frac{kT}{C_S}\frac{R_{MUX}}{R_{SW}}G_{SF}^2 \tag{3.30}$$

The MUX-related noise sees the gain of the input buffer G_{SF}, hence the squared DC gain term in (3.30)

Finally, we can add the two noise expressions (3.29) and (3.30) and calculate the total noise of the sampling phase:

$$\overline{V_{SN}^2} = 2\left(\overline{V_{SN_{MUX}}^2} + \overline{V_{SN_{SW}}^2}\right) = 2\frac{kT}{C_S}\left(\frac{R_{MUX}}{R_{SW}}G_{SF}^2 + nf\right) \tag{3.31}$$

Here, the factor of two accounts for the differential signal path of the ADC. This result shows that the value of the MUX resistance is quite important and that its proper sizing depends on the sampling switch resistance. The noise increases

dramatically if R_{MUX} is much larger than R_{SW}. We have to keep this important result in mind when designing the MUX switches (Chapter 4.1.2).

Finally, we refer the noise to the V_{res} node, as already mentioned above. We can use (3.24) with y_2 set to zero to obtain:

$$\overline{V_{SN_{res}}^2} = \overline{V_{SN}^2} G_C^2 = 2\frac{kT}{C_S}\left(\frac{R_{MUX}}{R_{SW}} G_{\mathrm{SF}}^2 + nf\right) G_C^2 \tag{3.32}$$

Next, the input signal is referred to V_{res} as well, again using (3.24):

$$V_{S_{res}} = V_{in} G_{SF} G_C \tag{3.33}$$

3.4.2 Noise Analysis for the ADC Conversion Phase

The noise of the conversion phase requires careful analysis, since it has multiple high bandwidth nodes, like V_{DAC} and V_{res} (see Figure 3.14), which can easily cause a significant noise increase. It is important to design the conversion loop such that it has one node that limits the bandwidth of all important noise sources. This is why we employ a g_m-C integrator as part of the pre amplifier. In [22] [23] it was shown that the g_m-C integrator acts as a noise filter whose input-referred equivalent noise bandwidth is proportional to $f_{NBW}=1/(2T_{int})$, there T_{int} is the integration time. Figure 3.17 shows the basic setup of the g_m-C noise filter. We use a resistor to model the input noise source in front of the integrator. The output-referred noise can then be calculated and referred back to the input (see Appendix A4), which results in:

$$\overline{V_{in}^2} = 4kTR\Delta f \xrightarrow{\Delta f = f_{NBW}} = 4kTRf_{NBW} = \frac{4kTR}{2T_{int}} \tag{3.34}$$

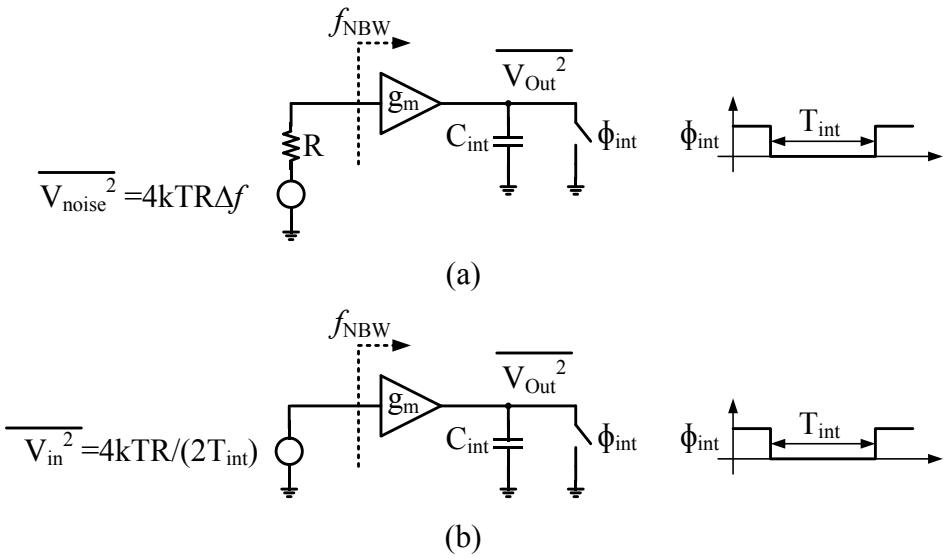

Figure 3.17: Basic g_m-C integrator noise filter operation. (a) Noise model with input noise resistor and (b) with equivalent input noise generator

To calculate the total noise of the conversion phase, we can therefore simply refer all noise sources to the V_{res} node and multiply them with $f_{NBW}{}^4$, as indicated above.

Figure 3.18 shows the ADC during conversion, with three noise sources corresponding to the DAC, buffer and the comparator. The differential DAC noise can be expressed as:

$$\frac{\overline{V_{DAC}^2}}{\Delta f} = 8kTR_{DAC}\left(1+\gamma_{DAC}\frac{g_{m_{DAC}}}{I_D}\frac{V_{FS}}{4}\right) \tag{3.35}$$

and is composed of the noise of the DAC load resistors and the DAC current source noise. A detailed derivation of (3.35) can be found in Appendix A5. $V_{FS}/4$ corresponds to the common mode voltage of the DAC.

[4] Note that the noise of the sampling phase is not filtered by the g_m-C integrator, because it is already sampled.

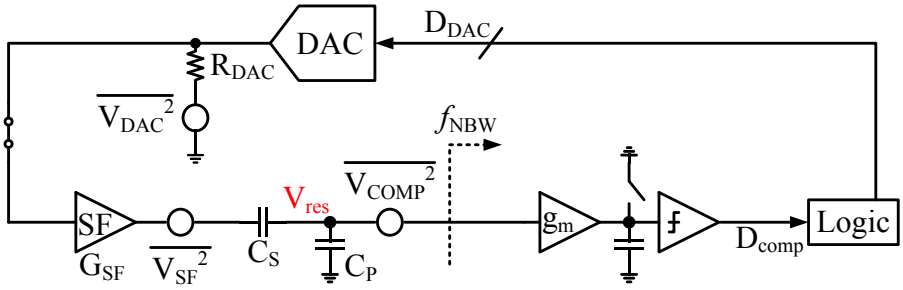

Figure 3.18: Noise model of the conversion phase

Next, the buffer noise is described by:

$$\frac{\overline{V_{SF}^2}}{\Delta f} = \frac{8kT}{g_{m_{SF}}} n f_{SF} \tag{3.36}$$

where nf_{SF} accounts for the additional noise of the SF current source. Finally, the input-referred comparator noise can be estimated using:

$$\frac{\overline{V_{COMP}^2}}{\Delta f} = \frac{8kT}{g_{m_{comp}}} n f_{comp} \tag{3.37}$$

We assume for the comparator that the preamplifier (Figure 3.18) is implemented using a cascade of differential pairs gain stages and that the first stage is the dominant noise source.

Now we can refer all noise sources to V_{res} to quantify the total noise of the conversion phase:

$$\begin{aligned} \overline{V_{CN_{res}}^2} &= \left(G_{SF}^2 G_C^2 \frac{\overline{V_{DAC}^2}}{\Delta f} + G_C^2 \frac{\overline{V_{SF}^2}}{\Delta f} + \frac{\overline{V_{COMP}^2}}{\Delta f} \right) f_{NBW} \\ &= 8kT \left(G_{SF}^2 G_C^2 R_{DAC} \left(1 + \gamma_{DAC} \frac{g_{m_{DAC}}}{I_d} \frac{V_{FS}}{4} \right) \right. \\ &\quad \left. + G_C^2 \frac{n f_{SF}}{g_{m_{SF}}} + \frac{n f_{comp}}{g_{m_{comp}}} \right) \frac{1}{2T_{int}} \end{aligned} \tag{3.38}$$

Together with the sampling and quantization noise, we can now also calculate the SNR_{ADC}:

$$SNR_{ADC} = \frac{\frac{1}{2}\left(\frac{V_{FS}}{2} G_{SF} G_C\right)^2}{\overline{e_q^2} G_{SF}^2 G_C^2 + \overline{V_{SN_{res}}^2} + \overline{V_{CN_{res}}^2}} \tag{3.39}$$

We will now discuss some important insights that can be gained from (3.39) and (3.38). First, notice that both the signal and the quantization noise scale with the buffer gain, and hence there is no quantization noise penalty for a buffer gain below unity. This is a benefit of our architecture that does not apply to a standard converter where the buffer is placed outside of the SAR loop. The price paid for this feature is the extra thermal noise introduced by the required input MUX.

Equation (3.38) also shows that the conversion noise strongly depends on the integration time T_{int}. If we want to run the ADC faster, we must lower T_{int} (see Section 3.3.2), which will in turn increase the noise. This noise increase can be compensated by lowering R_{DAC} and increasing g_{mSF} and g_{mcomp}, at the expense of extra power consumption. In Chapter 6, we will discuss a solution that partly breaks this tradeoff between speed, noise and power dissipation.

Appendix A6 uses (3.39) to calculate the SNR of the ADC for parameter values taken from the ADC prototype design (schematic values). We obtain an SNR estimate of 74.92 dB, which is very close to the measured SNR of 75 dB.

3.5 Summary

We introduced in this chapter the loop-embedded input buffer as well as its behavioral model, from which important design insights can be deduced. Furthermore, we highlighted the basic SAR ADC operation followed by a first order noise calculation.

4 ADC Implementation

In this chapter, we will discuss the circuit level implementation and details of the SAR ADC using a loop-embedded input buffer (Figure 4.1). First, we will discuss the front-end of the ADC, since it is the critical part regarding linearity. This includes the buffer design and architecture, a modified bottom plate sampling scheme and the PCB to ADC interface. After that, we move to the current steering DAC and detail the different design choices that were made, as well as the self-calibration strategy. Finally, the comparator architecture is introduced and various circuit details are highlighted.

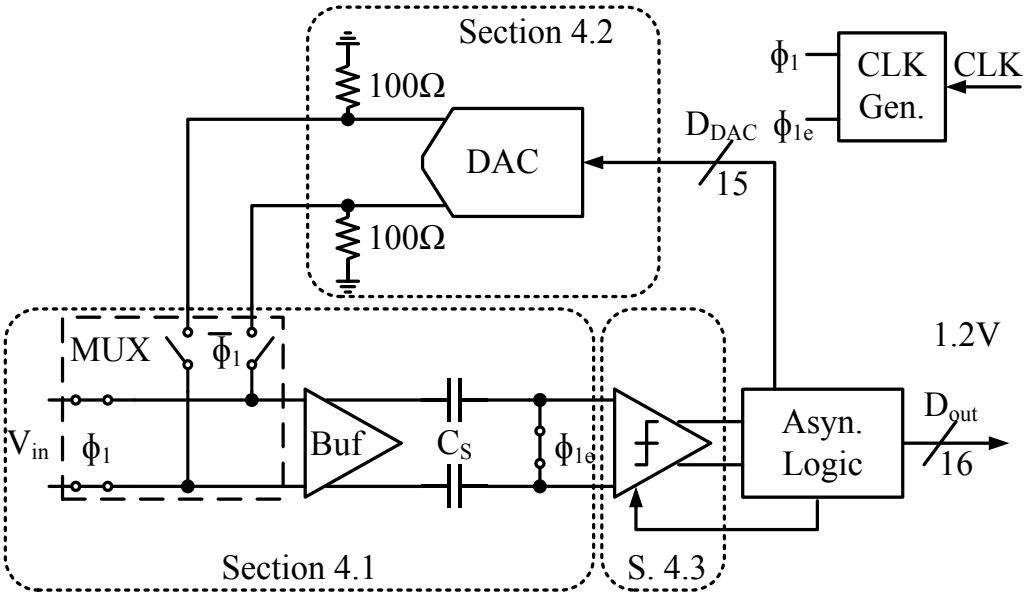

Figure 4.1: ADC block diagram with section numbers for the different ADC blocks

4.1 The SAR ADC Front-End

Figure 4.2 shows the ADC front-end including the source follower buffer circuit and bottom plate sampling switches. In this section, we will discuss several important aspects of the front-end. We will start with the buffer, justifying our architecture choice as well the sizing of the buffer.

4.1.1 The Input Buffer

We mentioned already in Chapter 3 that source followers (SF) are used to implement the buffer. We want to give some additional justification here on why we chose this architecture. First, the SF is a single branch circuit; all the invested power is used to drive the sampling capacitor, with no additional current being wasted. This results in an efficient buffer, because we do not have to invest any extra power to establish a large loop gain, like in a closed-loop buffer implementation. However, we have to compensate for the weak nonlinearity of the SF, and we do this by putting it into the SAR feedback loop (see Chapter 3).

Furthermore, we want to lower the ADC input capacitance, which is one of the main motivations of this work. The SF allows us to do that. Figure 4.3 shows a simplified circuit model of the SF. It is assumed that C_{in} is on the order of C_{gg},

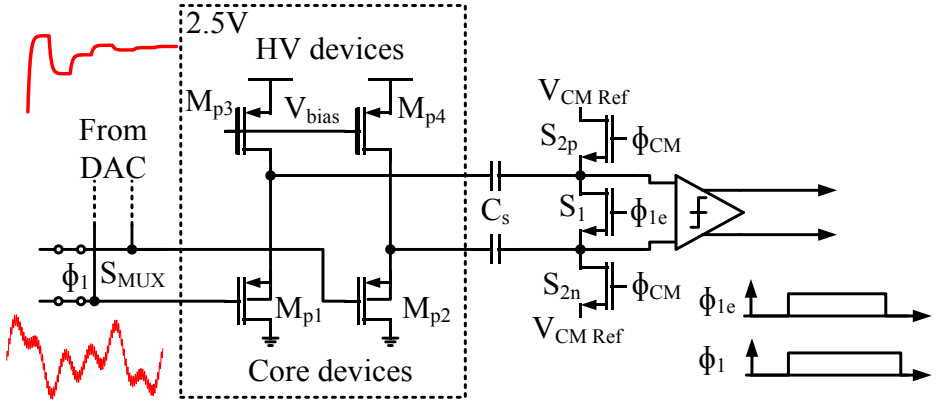

Figure 4.2: SAR ADC Front-End

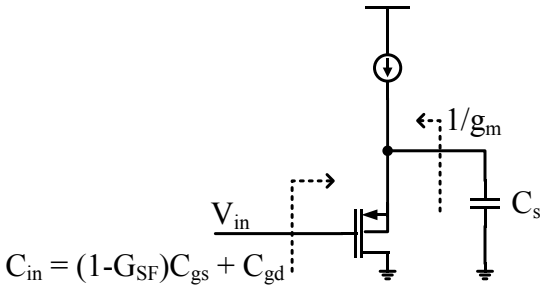

Figure 4.3: Simplified source follower circuit, showing C_{in} and the output resistance (ignoring gds)

ignoring the bootstrapping effect for C_{gs}. This allows us to simplify the algebra at the expense of pessimistic results. However, it should be noted that during large signal transients, we can see the full input capacitance C_{gg}, for example when the SF is slewing.

To proceed we estimate the output pole frequency f_p using the load capacitor C_S and the buffer output resistance (neglecting the transistor's finite output conductance):

$$f_p \approx \frac{g_m}{2\pi C_s} \tag{4.1}$$

The transit frequency f_t of the transistor is given by:

$$f_t \approx \frac{g_m}{2\pi C_{gg}} \tag{4.2}$$

combining (4.1) and (4.2) results in:

$$\frac{f_t}{f_p} \approx \frac{C_s}{C_{gg}} \approx \frac{C_s}{C_{in}} \tag{4.3}$$

We can quickly see from (4.3) that a high f_t and a lower f_p are desired to minimize C_{in} for a given C_s. We cannot arbitrarily lower f_p due to the 14-bit settling requirement, which requires a minimum of about 10 time constants (τ) and gives us a lower bound on f_p for a given tracking time. Furthermore, f_t is bounded by the process and design choices such as the channel length and the bias conditions. Let

us assume an f_t of about 50 GHz (possible to achieve in a 40 nm process) and a f_p of about 1 GHz. Using (4.3) then shows that the C_{in} of the SF device can be 50 times lower than C_s.

Figure 4.2 shows that PMOS transistors are used to implement the buffer. This gives us an input voltage range from 0 V to 1.2 V (see Figure 4.4(a)), which is compatible with the allowed core device voltage range. However, we use a more practical input voltage range from 0 V to 0.9 V, resulting in a 1.8 V differential full scale voltage V_{FS}. Notice that we would have to level-shift the input voltage or sacrifice useable input voltage range, if an NMOS-based SF was used (see Figure 4.4(b)). However, the PMOS devices have a lower f_t and hence we do not get the smallest possible C_{in}.

It is important to note that the maximum output voltage of the SF is about 1.4 V (0.9 V + V_{gs}), which is higher than the gate oxide breakdown voltage (1.2 V). However, both the gate-source and gate-drain voltages V_{gs} and V_{gd} are smaller than 1.2 V at all times[5] (see Figure 4.4(a)) and hence the oxide will not break down. Only the V_{ds} voltage is higher than 1.2 V, which is safe because the channel and the PN diodes can withstand these voltages.

We chose for this prototype ADC a sampling speed of 35 MS/s, giving us

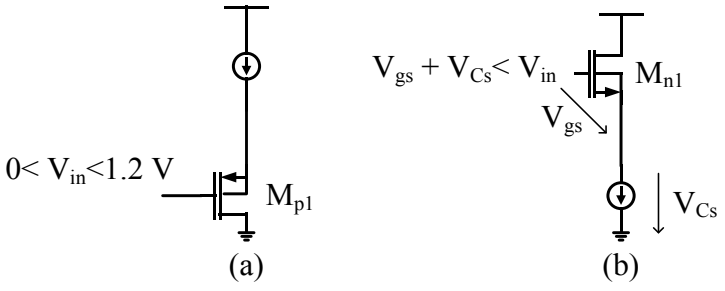

Figure 4.4: (a) PMOS source follower (b) NMOS source follower

[5] We employ proper power sequencing at the board level to protect the device during start-up. We first apply the input common mode to the buffer and then turn on its supply, which keeps V_{gs} and V_{gd} of the source follower devices always below 1.2 V.

roughly 28.6 ns for tracking and conversion. 20% of this time is used for tracking, i.e., T_{track} = 5.7 ns. Furthermore, we use a 3.5 pF sampling capacitor, sized for kT/C noise (see Appendix A6). From these numbers, we can calculate the g_m (30 mS) of the M_{p1} and M_{p2} transistors (see Appendix A7). Finally, we have to determine two additional parameters, the channel length L_p of M_{p1}-M_{p2} and their g_m/I_D ratio (i.e. the inversion level), to complete the buffer design. However, these two parameters influence the ADC performance on multiple levels, such as power consumption, noise, input capacitance and linearity. We will use the insight from Chapter 3 in the following discussion to choose L_p and g_m/I_D.

In Section 3.2.3, we showed that phase imbalance is a very critical parameter that has a major impact on the SFDR performance of the ADC (limited by HD_2). Furthermore, the proposed nonlinearity cancelation will not reduce this error term. Two options are available to address this issue: minimize the phase imbalance and/or improve the linearity of the buffer to provide better HD_2. The phase imbalance is dictated by the available PCB circuit blocks, such as the transformer. In our case, we must tolerate an error up to 1 degree for the frequency range of interest [24], which can lead to a large second harmonic. Figure 3.9(a) shows that we need an HD_2 of about -54 dB (1° phase error) to achieve an ADC SFDR > 90 dB, dictating a lower bound for the linearity performance of the SF. This leaves us only with the option to improve the buffer linearity. We use a bulk–to-source connection to eliminate the backgate effect. Furthermore, we use a 2.5 V supply, helping us improve linearity by giving the transistors more headroom for full-scale signals.

Another design option is to increase the intrinsic gain of the transistors M_{p1} M_{p2} (see Appendix A8). This is done by increasing the channel length and/or the g_m/I_D ratio (see Figure 4.5). However, by doing this we also lower f_t at the same time, which in turn gives us a larger input capacitance according to (4.3). In our design, we compromised with a channel length of L_p = 80 nm and a g_m/I_D = 12 1/V (moderate inversion), resulting in an f_t of about 14 GHz and HD_2 = -55 dB for a full-scale input. It should be noted that a high g_m/I_D is of advantage since it leads

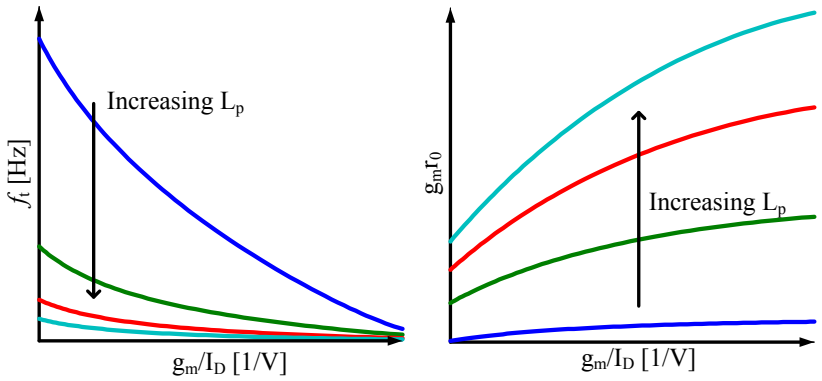

Figure 4.5: Transit frequency f_t and intrinsic gain $g_m r_0$ versus the g_m/I_D ratio (with transistor length L_p as parameter)

to a lower bias current. A g_m of about 30 mS with g_m/I_D = 12 1/V results in a drain current of about 2.5 mA. This gives us a total buffer power of 12.5 mW. Furthermore, with the resulting buffer pole (f_p = 670 MHz) and f_t (14 GHz), the input capacitance is 1/18th of C_S.

Next, we will consider the mismatch mechanisms between the two ADC phases, which can lead to SFDR degeneration (see analysis in Section 3.2.1). The first significant issue related to buffer nonlinearity mismatch stems from the fact that the circuit is driven with a high-frequency continuous-time input during tracking (ϕ_1), but sees voltage steps from the DAC that settle to "DC" during conversion (see Figure 4.2). The buffer nonlinearity between these two phases can only match if the distortion is nearly frequency independent, i.e. it follows a memoryless power series model. Fortunately, we found that this condition is met due to the high settling speed required from the buffer. The buffer was designed for a 3-dB bandwidth that is a large multiple of the input bandwidth (first Nyquist zone) so that the transients settle to within a fraction of the 14-bit LSB during tracking. Figure 4.6(a) shows the simulated distortion of a single-ended follower vs. input frequency. The harmonic distortion terms stay almost constant from DC to Nyquist. As shown in Figure 4.6(b), the phase of the third harmonic is shifted by about 3 degrees near Nyquist. Further analysis shows that the HD_3 of the ADC can be estimated using the following equation (see Appendix A9):

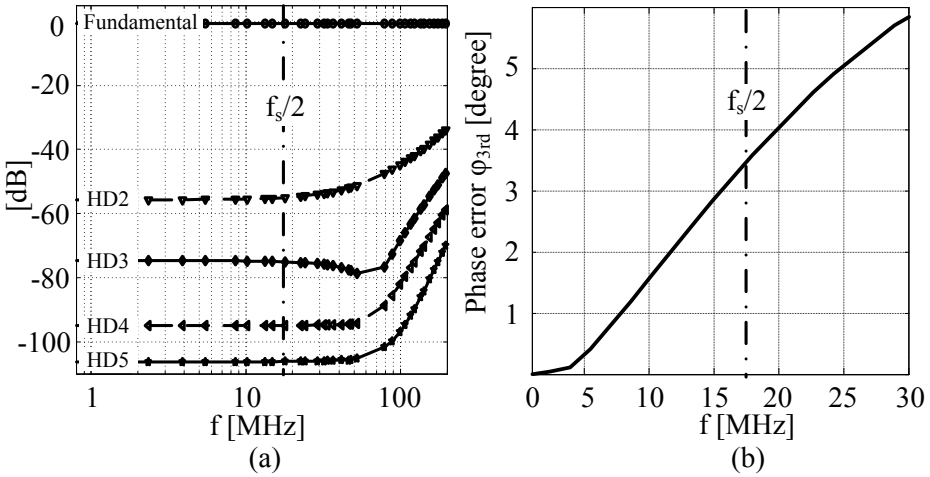

Figure 4.6: (a) Simulated source follower distortion vs. frequency (single ended). (b) Phase error of the 3rd harmonic vs. frequency

$$HD_3(\varphi_{3rd}) = \frac{a_3 \hat{v}_{in}^2}{2a_1} \sin\left(\frac{\varphi_{3rd}}{2}\right) \tag{4.4}$$

giving us a HD_{3ADC} < -95 dB for a 3 degree phase error (Figure 4.7). The second harmonic is cancelled by the pseudo-differential nature of the circuit, provided that the input has proper phase balance, as discussed above.

A second potential nonlinearity mismatch issue is related to the buffer's input common mode. Since the output common mode of the DAC and the common-mode of the input won't match perfectly, there will be a change in the buffer's

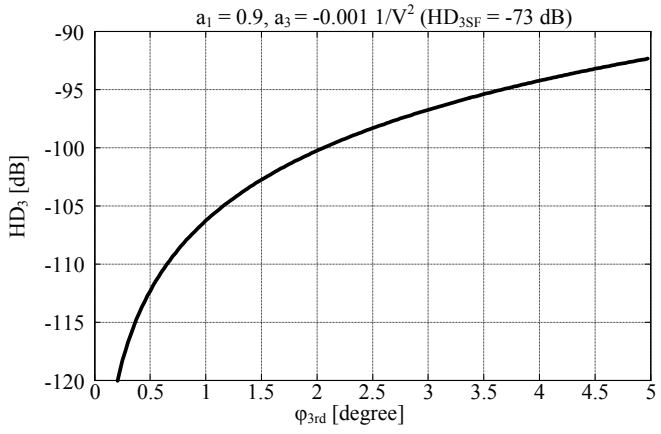

Figure 4.7: HD_3 of the ADC versus 3rd harmonic phase error φ_{3rd} (calculated using (4.4))

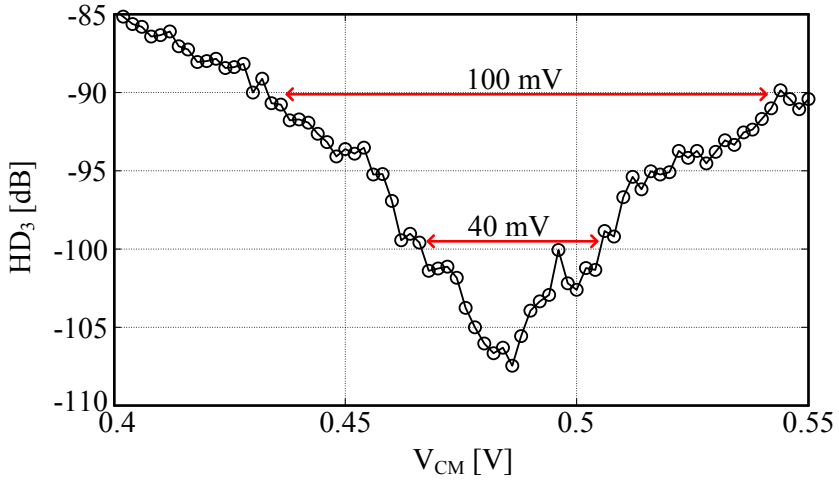

Figure 4.8: Measured HD_3 versus input signal common mode voltage (V_{CM})

operating point as it transitions from track mode to conversion. We studied this issue carefully in simulation and also during the measurement phase of the ADC. Figure 4.8 shows the measured HD_3 of the converter as the input common mode is swept. We find that there is a window of approximately 40 mV for which the HD_3 stays below -100 dB. To maintain this level of performance in a practical SoC implementation, the common-mode feedback loop of the driving circuit can be referenced to a replica of the DAC common-mode voltage.

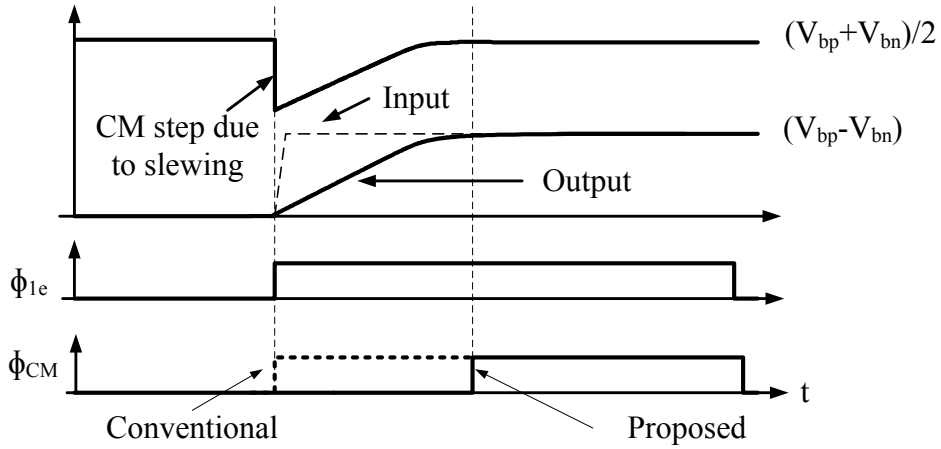

Figure 4.9: Timing of the common-mode switches

A final (and more general) design consideration for the buffer is related to its slewing behavior. Conventionally, the common mode (CM) switches ($S_{2p,n}$ in Figure 4.2) are turned on together with the main sampling switch (S_1) at the beginning of the tracking phase. However, this leads to a large current draw from V_{CM}, since the buffer slews initially and its outputs have a CM that deviates from the nominal value. To address this issue without investing extra power and large switches, the turn-on of $S_{2p,n}$ is delayed by 1 ns to protect V_{CM} from the transient (see Figure 4.9). At this time, the slewing of the buffer has finished and the buffer's CM is much closer to its nominal value. Both options (conventional and modified timing) were implemented on chip. Figure 4.10 shows the spectrum for a 20 MHz input signal for both cases, indicating an HD_3 improvement of 27 dB when

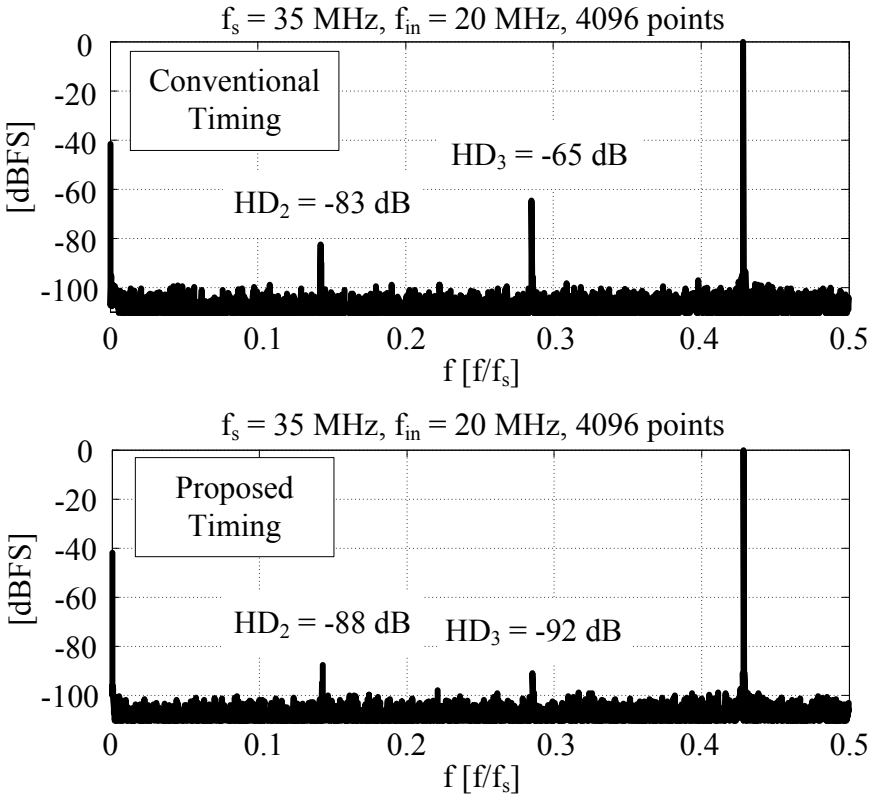

Figure 4.10: Measured ADC output spectrum (conventional vs. proposed timing)

the modified timing is enabled. Due to the modified switching, we were able to generate V_{CM} using only 100 μA.

4.1.2 The MUX Switches

The MUX switches in Figure 4.1 serve to switch the buffer from the ADC input to the DAC feedback signal as the converter transitions from the tracking to the conversion phase. One important aspect of this MUX is its noise performance. During conversion, it is clear that the switch resistance must be smaller than the 100 Ω DAC resistors to maintain low noise. The situation during tracking is analyzed in Section 3.4.1 and shows that the differential noise acquired on the sampling capacitors is given by:

$$\overline{V_{SN}^2} = 2\frac{kT}{C_S}\left(\frac{R_{MUX}}{R_{SW}}G_{SF}^2 + nf\right) \tag{4.5}$$

Here, R_{MUX} is the MUX resistance, R_{SW} is the total resistance of the remaining sampling network (buffer output resistance plus bottom plate switch (ϕ_{1e}) resistance), *nf* the excess noise factor of the source follower, and G_{SF} is the follower's voltage gain. This result indicates that the ratio R_{MUX}/R_{SW} must be kept small for low noise. To achieve this, we employ bootstrapped switches [19], sized to obtain R_{MUX} = 12 Ω and R_{MUX}/R_{SW} = 0.25. The resulting noise contribution from the tracking phase is about 31% of the total ADC noise (see Appendix A6). Lastly, note that the switch bootstrapping is not only essential for low noise, but also helps minimize nonlinear tracking errors.

4.1.3 Input Network

Unfortunately, and as we will discuss below, the small MUX resistance causes complications with the overall input network of the converter, which includes a bondwire inductance when the input is supplied through a wire-bonded package. This problem does not exist in the end application of this converter (within an SoC that drives the ADC inputs from within the chip), but it was important to understand the issue for prototype evaluation.

Consider the model of Figure 4.11(a). The input capacitance of the buffer (C_{SF}) and the bondwire inductance (L) form a series resonance tank through R_{MUX} with quality factor $Q \sim \frac{1}{R_{MUX}}\sqrt{\frac{L}{C_{SF}}}$. Since R_{MUX} and C_{SF} are small, the quality factor is large and this leads to excessive ringing during the MUX switching events. For our prototype, we addressed this issue by splitting up the PCB filter capacitance (originally 10 pF) and moving 3.5 pF of it onto the chip along with an addi-

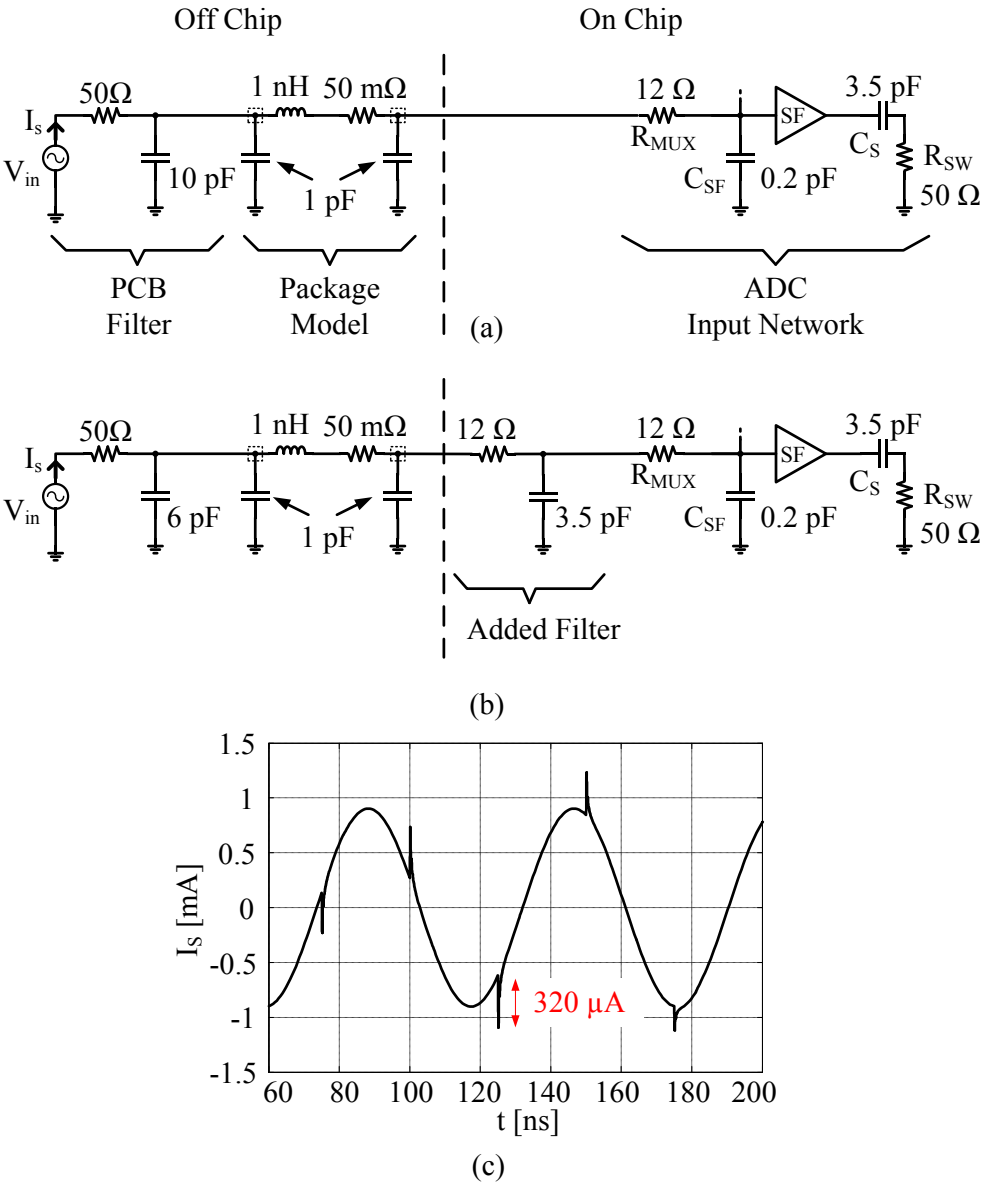

Figure 4.11: Buffered ADC interface: (a) without on-chip filter, (b) with on-chip filter. (c) Simulated source current for the circuit with input filter

tional 12 Ω resistor Figure 4.11(b). Aside from reducing Q, this brings several additional benefits. First, at the start of the tracking phase, the on-chip filter capacitance provides nearly 94% of the signal to the buffer input via charge sharing. This means that both the bondwire inductor and the input source see only small current glitches during switching (see Figure 4.11(c)). Second, the network sees only a small capacitance change between the two ADC phases, since only the relatively small C_{SF} is switched in and out. Both of these properties, enabled by the on-chip buffer with low-input capacitance, make our ADC easy to drive.

For comparison, Figure 4.12(a) illustrates the model for a standard, unbuffered ADC interface in which the large sampling capacitance is directly driven by the external circuitry. As seen from Figure 4.12(b), the un-buffered interface demands dynamic currents of up to 7 mA during switching events, which

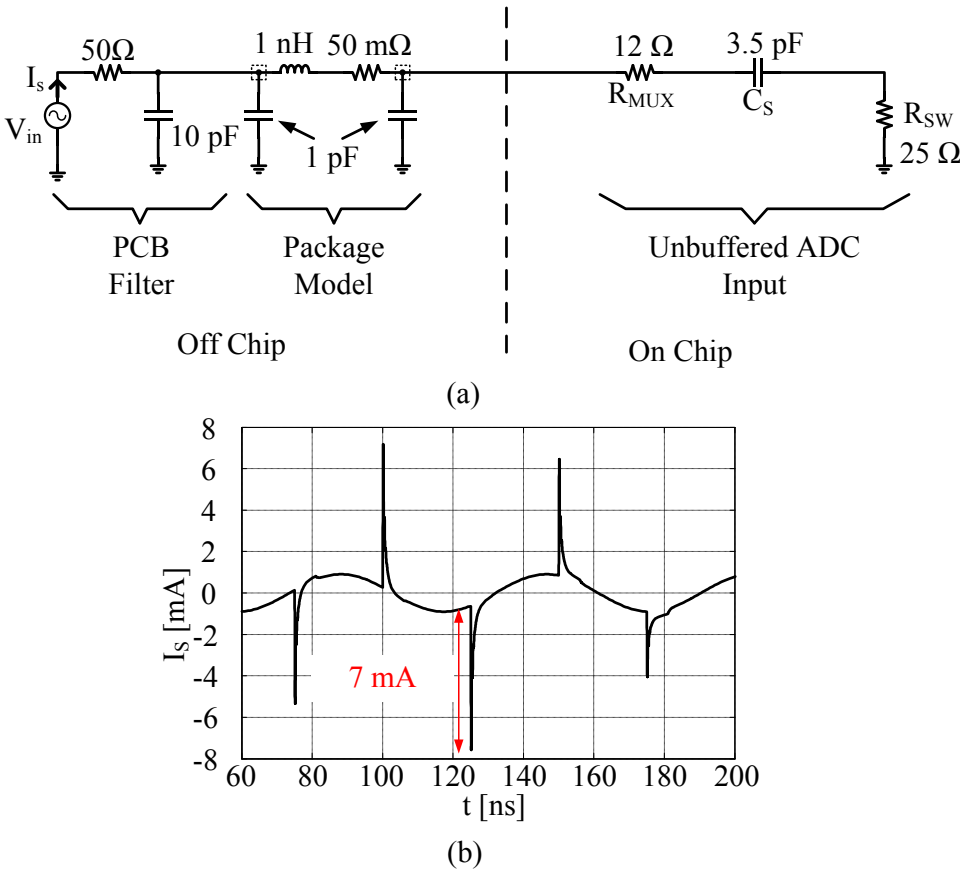

Figure 4.12: (a) Unbuffered ADC interface. (b) Simulated source current versus time

is more than an order of magnitude larger than the spikes observed in Figure 4.11(c).

4.2 Feedback DAC

The feedback DAC uses a current-steering topology (see Figure 4.13) with non-binary weights and its reference current is derived from an on-chip bandgap reference. Compared to the traditional charge redistribution architecture, this DAC doesn't require a low impedance reference voltage and helps mitigate cross-talk issues due to its class-A nature; this can be a critical aspect in SoC environments. This DAC can also be used for other functions on the same SoC[6] as it has good drive capability. However, all of these advantages come at the price of higher power consumption, so future work could consider the replacement with a more efficient switched-capacitor topology.

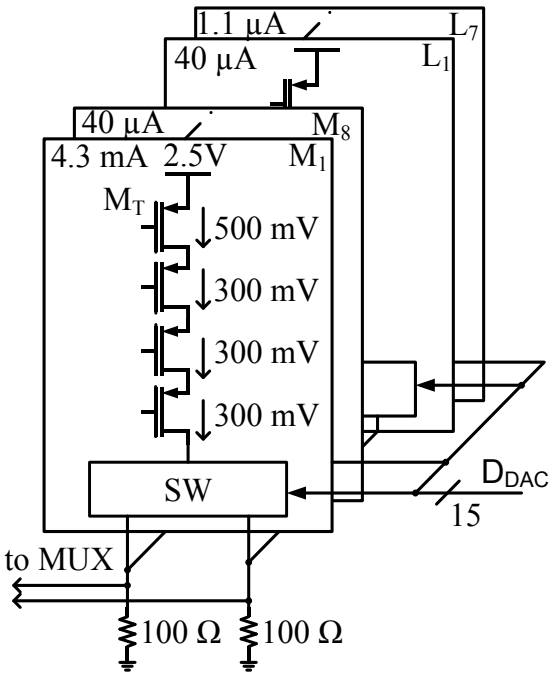

Figure 4.13: Current steering DAC

[6] For example, one could think about a half-duplex system were data either is received or transmitted but not at the same time, and hence the ADC and DAC operation could be timed shared

The DAC is segmented into 8-bit MSB and 7-bit LSB sections with the following non-binary weight vectors: [3820 1960 1080 600 320 160 80 40] and [40 20 11 6 4 2 1]). This creates two redundant SAR cycles, which helps alleviate the DAC settling requirements [1], [2]. The DAC MSB weights are measured at start-up using the LSB section similar to [2], [18]. This calibration allows us to match only the LSB section to 14 bits, while the MSB section is sized to stay within the redundancy range, ensuring a compact design (Section 4.2.2). The measured weights are used to assemble the final output from the 16 bit raw data (D_{out}) off chip.

Note that although the DAC switches run at very high speed due to the asynchronous SAR loop, its transient glitches do not affect the conversion result as long as the DAC settles to within the redundancy range. This means that the dynamic linearity performance of the DAC is not important, and we mainly care about its DC linearity. With DC calibration in place, residual inaccuracies are mainly due to the signal-dependent output resistance of the DAC cells, an effect that we mitigate using triple-cascoded current cells, as shown in Figure 4.13. More information about the DAC biasing can be found in Appendix A10.

Another important design parameter of the DAC is its noise power spectral density (PSD), which can be computed using (Appendix A5):

$$\frac{\overline{V_{DAC}^2}}{\Delta f} = 8kTR_{DAC}\left(1+\gamma\frac{g_m}{I_D}\frac{V_{FS}}{4}\right) \tag{4.6}$$

In this expression R_{DAC} is the DAC load resistor, g_m/I_D is the transconductance efficiency of the tail current devices (M_T in Figure 4.13), γ is their thermal noise parameter (about 0.7 in our process) and V_{FS} is the converter's differential full-scale range (1.8 V_{pp} in our design). With 500 mV across the tail current devices, their g_m/I_D (which is approximately equal to $2/V_{Dsat}$) can be no smaller than 4 1/V. Combining all of these parameters gives a value of 2.3 for the bracketed term in (4.6). With R_{DAC} = 100 Ω (and a resulting DC current of 9 mA), the contribution

from the DAC to the overall ADC noise is about 38 % (see detailed analysis in the Appendix A6).

4.2.1 Current Steering DAC Switches

Another critical DAC component is the switch architecture that performs the current steering. Traditionally, a structure like in Figure 4.14(a) is implemented for a single current source [25], which uses two PMOS switches to steer the current I_0 to the left or right branch. However, we use the DAC in a tri-state configuration, meaning that we can activate both switches simultaneously to create a zero output signal without changing the DAC common mode voltage. It is important for that operation that the current I_0 splits evenly, creating the same voltage drop across both resistors. However, the structure of Figure 4.14(a) achieves this only when the drain voltages of the transistors are equal, which won't be the case when other unit elements in the array are steered to create a differential output. To understand the issue, consider Figure 4.14(b), which shows a single current cell when all codes are set to the zero output state. Here, the voltage sources model the rest of the current steering DAC, producing the 450 mV common mode voltage. In this case, the current I_0 splits evenly, because both branches have equal voltages. On the other hand, Figure 4.14(c) shows the case when other DAC current sources

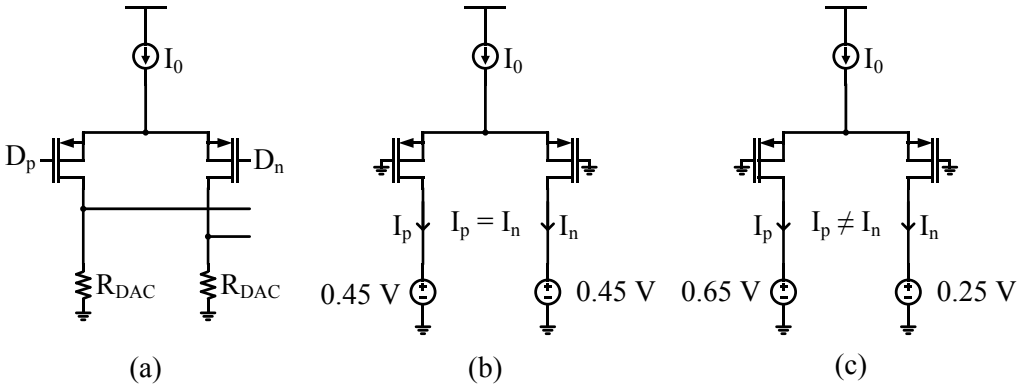

Figure 4.14: (a) Standard DAC cell switch implementation, (b) Tri-stated DAC cell with no differential output signal. (c) Tri-stated DAC cell with non-zero differential output signal

are steered, producing a differential output voltage. In this case, the current I_0 does not split evenly anymore due to V_{DS} modulation, creating an additional and unwanted signal from the tri-stated cell.

For this reason, we use the switch implementation shown in Figure 4.15, which splits the current source and switches in half, enforcing an even current split during tri-stating. We will use the following notation to indicate the state of the DAC: D = 0 split current (zero output), D = -1 current to one side (negative output) and D = 1 current to the other side (positive Output). Notice that we do not get a perfect current split due to mismatches between the current sources. However, this offset is only present when the DAC is in the zero state (D = 0) and disap-

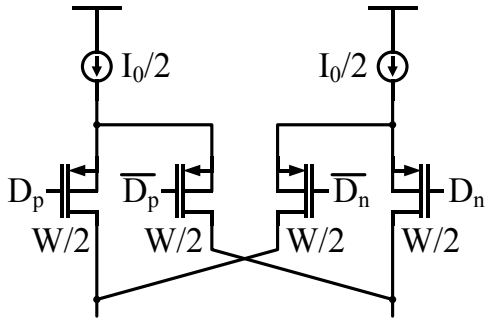

Figure 4.15: Split current source and switch implementation of a single DAC cell

pears when the current is steered to one or the other side. Hence, this offset is coverd by redundancy, since the offset disappears at the end of the conversion phase (no current source in D = 0).

The ability to generate zero DAC output has additional advantages in the context of using non-binary weight vectors. In a binary weighted DAC, we can create a zero output (in order to measure the sign) by switching the MSB to one side and switching all other current sources to the other. However, in a non-binary weighted DAC this operation is much more challenging, since the MSB weight is smaller than the sum of the remaining current sources. Furthermore, the tri-state also eases the self-calibration scheme, which will be explained next.

4.2.2 DAC Self-Calibration

The weights of the MSB section are measured using the LSB section [2], [18]. We want to give a brief overview on how this is implemented in our design. Figure 4.16(a) shows the simplified DAC waveform during two calibration cycles, we are using the same notation as in Figure 1.2. We first sample a zero input (V_{sample}). However, we will sample an offset that is introduced by the buffer (SF) (Figure 4.16). We next enter a modified conversion phase. The MSB current

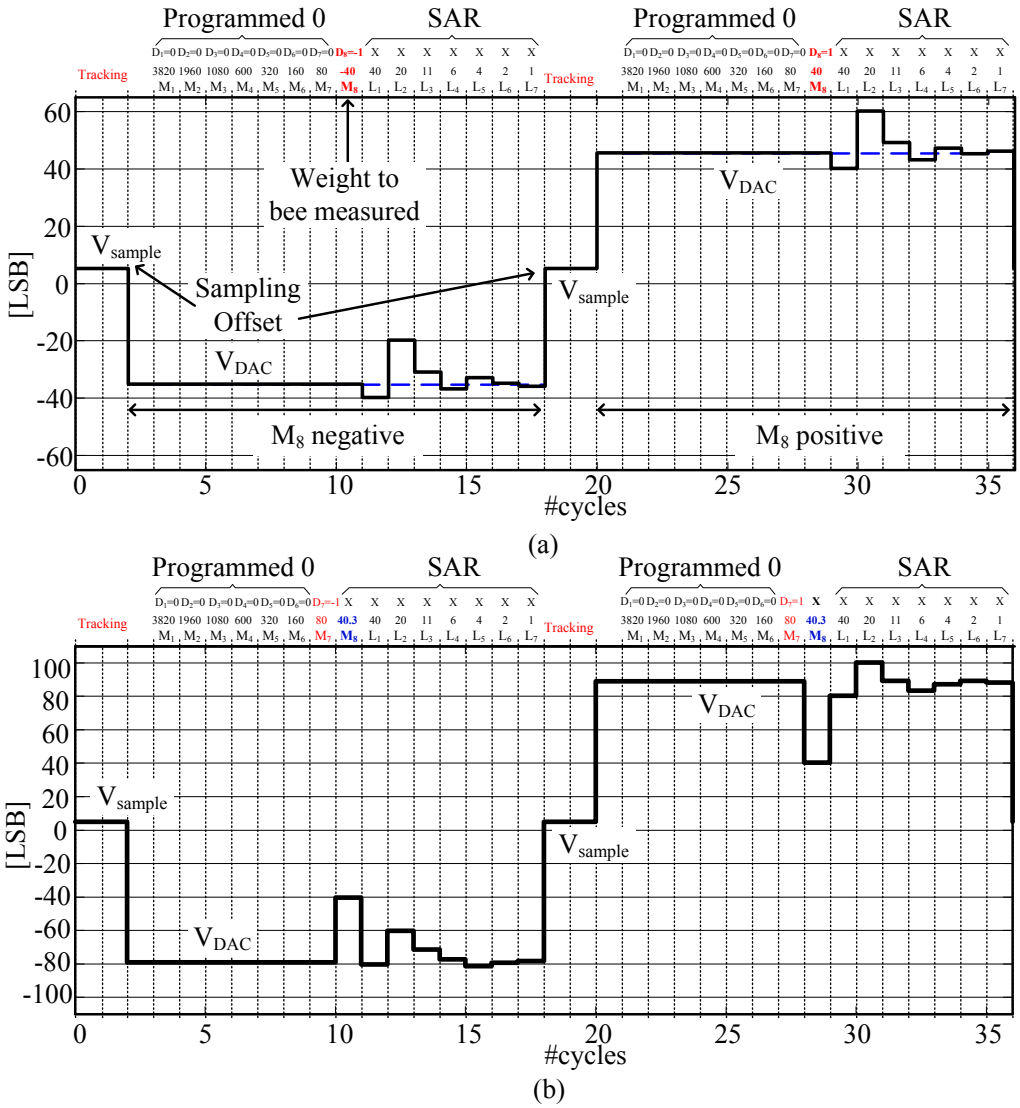

Figure 4.16: DAC self-calibration scheme. (a) Cell M_8 is measured using the LSB section, (b) cell M_7 is measured using the LSB+M_8 section

sources M_1-M_7 are programmed to a zero output (tri-state). M_8 is the current source that we want to measure and is hard wired to produce a negative output (D_8 = -1). The LSB section then runs the SAR algorithm, measuring the hard – wired M_8 reference voltage plus the offset. This is indicated by the X values in Figure 4.16, resulting from the SAR algorithm. The next cycle repeats this, but this time the M_8 source is switched to the positive output side (D_8 = 1). This approach implements a chopping sequence that allows us to remove the offset. The sequence is repeated many times, averaging the result to measure the M_8 current source. Notice that the ADC is thermal noise-limited, resulting in sufficient LSB movement to average to an accurate measurement.

After measuring M_8, the ADC is reconfigured and M_8 can now be used in subsequent SAR iterations, since its value is now known. We are now activating M_7 and leave M_1-M_6 programmed to zero, allowing us to measure M_7. This procedure is repeated for all the remaining current sources. Figure 4.17 shows the measured histogram for all MSB current sources, as well as their mean and standard devia-

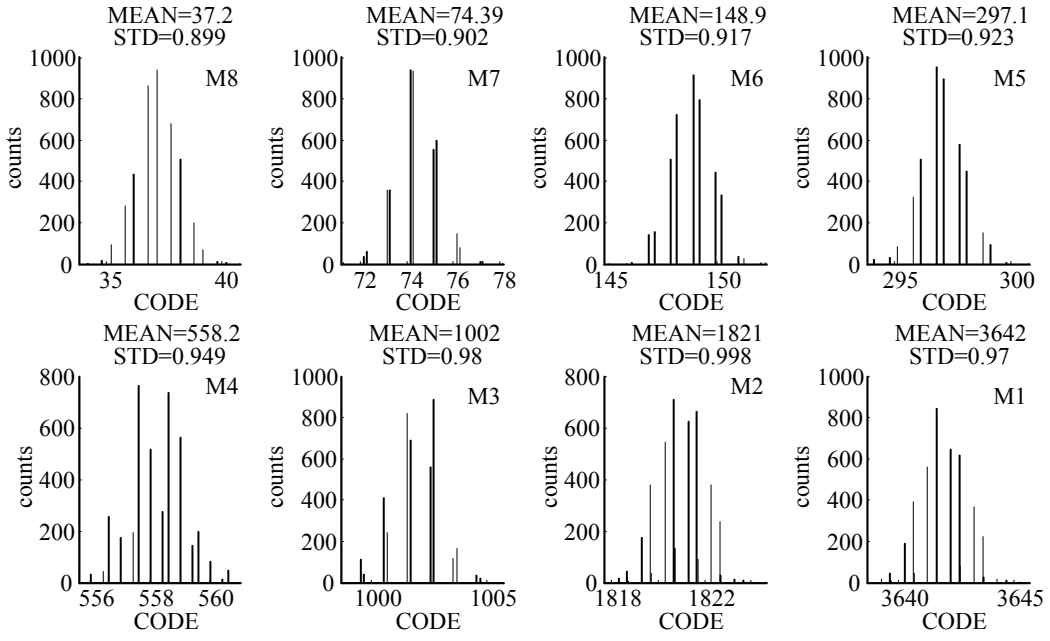

Figure 4.17: Measured histogram of the ADC MSB current sources obtained via self calibration (4096 measurements for every source)

tion (STD). The mean values are used to form the weight vector (W_C = [3642 1821 1002 558.2 297.1 148.9 74.39 37.2 40 20 11 6 4 2 1]) that is used to assemble the ADC output. Notice that the STD lies in the range between 0.9-0.98 LSB, and provides a measure of the ADC noise. We can use this to estimate the peak SNR of the ADC using (3.16):

$$SNR = 10\log\frac{\frac{(\sum W_C)^2}{2}}{STD^2} = 10\log\frac{\frac{(7665.1)^2}{2}LSB^2}{0.98^2\,LSB^2} = 74.85dB \qquad (4.7)$$

where $\sum W_C$ is the sum of the weights and is equal to half scale. We will show in the measurement Section 4.4 that the low frequency SNDR is in fact 75 dB.

4.2.3 DAC Load Resistor Considerations

A final consideration that went into the DAC design is the choice of load resistor. We preferred using polysilicon resistors for their small voltage dependence. However, for these devices, self-heating must be carefully considered due to the relatively poor thermal coupling to the substrate [26], [27]. Detailed analysis shows (see Appendix A11) that the HD_3 induced by self-heating can be estimated using:

$$HD_3 = \frac{1}{4}\,T_{C1}R_{th}R_{DAC}I_{DAC}^2 \qquad (4.8)$$

Here, T_{C1} is the temperature coefficient of the poly resistor and R_{th} is the thermal resistance to bulk. R_{th} is inversely proportional to the resistor area, which is then the only free available for controlling thermally-induced HD_3 in our design. For the process parameters of our prototype, a resistor area of 53 μm x 53 μm was necessary to mitigate potential self-heating issues. We verified through post-layout simulations that the parasitic capacitance of this large resistor did not increase the DAC's settling time. In our design, the speed-limiting nodes are in the cascode stack of the MSB current source.

4.3 Comparator

Figure 4.18 shows the high-level architecture of the implemented comparator. There are two main design parameters of concern for a comparator used in a 14-bit converter: noise and metastability. Reference [28] showed that the probability of metastability is an exponentially function of the comparator latch's time constant $\tau_{latch.}$. Thus, to achieve a low metastability rate, the latch needs a small time-constant and thus a large bandwidth ($\frac{1}{\tau_{latch}}$), which adversely affects its noise. A common way to break this tradeoff is to use pre-amplification in front of the high speed latch [29]. The preamplifier defines the noise specification, while the latch sets the metastability rate. We therefore designed the latch first, and then added a pre-amplifier to set the noise.

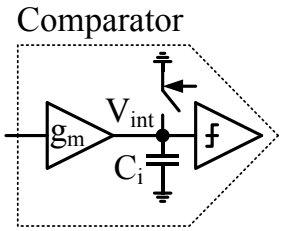

Figure 4.18: Comparator architecture

Figure 4.19 shows the schematic of the comparator latch, which uses the same architecture as described in [30]. The idea behind this implementation is to minimize the capacitive loading (C_L) of the latch nodes, hence lowering the latch time constant τ_{latch}. This is done by only using a resistive load and an NMOS latch structure. Notice that M_{C1} and M_{C2} act more as switches than cascodes; their function is to isolate the output nodes from the large parasitic capacitors (C_p) of the big driving transistors (M_{D1}, M_{D2}) during the latch phase. The time constant was estimated using extracted layout simulation and is for the typical case $\tau_{latch} = 7.5\ ps$.

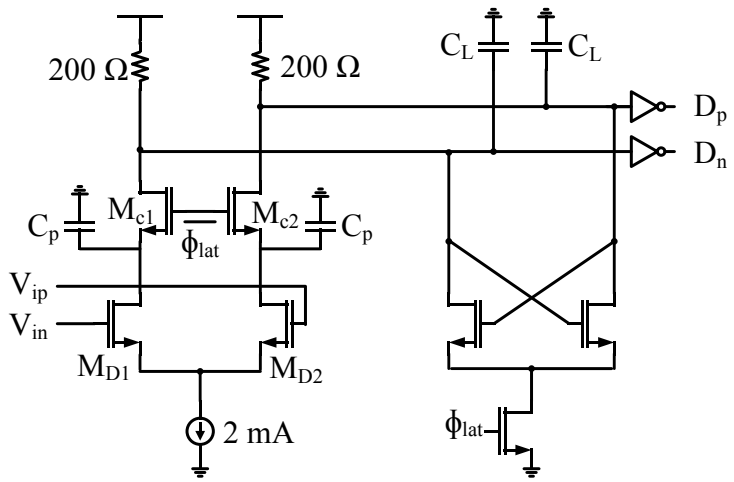

Figure 4.19: Schematic of the comparator's latch

The maximal decision time (T_{MAX}) for the comparator is needed, in order to calculate the probability of a metastable event [28]. We can take advantage of the asynchronous conversion [17], meaning that we only have a small number of hard comparator decisions during the conversion[7]. The other decisions will be much faster. From simulation, we found that the fast decisions are in the order of 130 ps. From here we can calculate T_{MAX}, by assuming that we have 14 fast and two slow decisions, which must equate to the given average time for the comparator ($T_{comp\ A}$ =150 ps):

$$T_{comp\ A} = \frac{14 \cdot 130\ ps + 2 \cdot T_{MAX}}{16} = 150\ ps \ \rightarrow T_{MAX} \approx 250\ ps \tag{4.9}$$

Now we can calculate the probability of a metastable event:

$$P(Error) = \frac{1}{A}\frac{V_{DD}}{\frac{V_{FS}}{2^B - 1}} e^{-\frac{T_{MAX}}{\tau_{latch}}} = \frac{1}{80}\frac{1.2\ V}{\frac{1.8\ V}{2^{14} - 1}} e^{-\frac{250\ ps}{7.5\ ps}} \approx 4.5 \cdot 10^{-13} \tag{4.10}$$

[7] In a binary-weigthed SAR ADC we only have one hard decision, however in a non-binary SAR there is the possibility to get more than one hard decision due to the redundancy.

where A is the preamplifier gain and is about 80 in our design.

A P(Error) = $4.5 \cdot 10^{-13}$ is already low; however, if we drop the sampling speed of 35 MS/s to 34.4 MS/s, adding 500 ps to the conversion, we can push P(Error) to 10^{-20}. Notice that we only used first-order hand calculation to estimate P(Error). We did not set up the capability to measure this probability, because the mean time to failure (MTF) is about $1/(4.5 \cdot 10^{-13} \cdot 35\ \text{MHz}) \approx 1200$ minutes.

C_L is extracted from post layout extraction and is used to estimate the noise. It has been shown that the noise of the latch is proportional to kT/C_L [31], [32], however additional terms are needed, which describe the large signal dynamics. We will use a first order hand calculation that assumes that the latch noise is just kT/C_L. This assumption treads the latch as a linear positive feedback circuit ignoring any large signal behavior. Hence, the latch output referred noise is approximated as:

$$\overline{V_{LatchO}^2} \approx \frac{2kT}{C_L} \tag{4.11}$$

Figure 4.20 shows the latch with a preamplifier block in front. We use this block diagram to refer the output noise to the input:

$$\overline{V_{comp\ Latch}^2} \approx \frac{2kT}{A^2 C_L} \tag{4.12}$$

We chose the gain A to be large enough to make the noise of the latch insignificant. With C_L of 40 fF and a gain of 40 we get a total integrated noise of 0.12 nV^2, which is small compared to the total noise 9.7 nV^2 (see Appendix A6). We increased the gain by another factor of two (A=80) to account for the inaccuracy of the first-order noise model. Furthermore, the preamplifier gain reduces the latch offset as well, which is useful, since we need a small comparator offset to avoid compromising the nonlinearity cancelation (see Section 3.2.2).

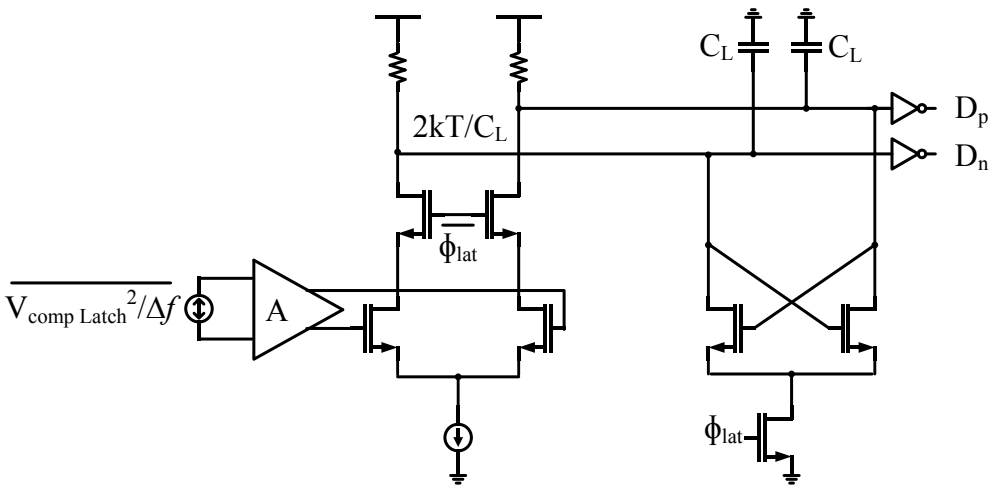

Figure 4.20: Input referred noise of the latch

We already mentioned that we use a g_m-C integrator as a preamplifier (see Section 3.4), since we use noise filtering during the conversion. In addition to that, the g_m-C integrator has to provide signal gain as well, as mentioned above. It is not easy to get a gain of 80 out of a g_m-C integrator in a 40 nm process, since its gain is fundamentally bounded by the $g_m r_o$ product of the transistors. There are two design options for achieving a large $g_m r_o$ product. One is to maximize r_o using cascoding and/or gain boosting, but this is difficult with short channel devices and low V_{DD}. Alternatively, we decided to boost g_m using four wideband stages with an aggregate voltage gain of approximately 80 (Figure 4.21). The fifth stage of the g_m cell merely serves to present a large resistance to the integration capacitor C_i.

For noise analysis, the design can be reduced to the first gain stage, arguing that the following stages are only adding a fraction to the noise due to the first stage's gain. Furthermore, we use stage scaling to keep the power low. The input-referred noise of a differential pair was calculated using (3.37), and from here it is possible to calculate the required g_m and design the preamplifier. We invest 2.5 mA in the first differential pair and 1 mA in the second stage. The last four stages, including the actual integrator stage, are sized the same and use 500 µA. The gain

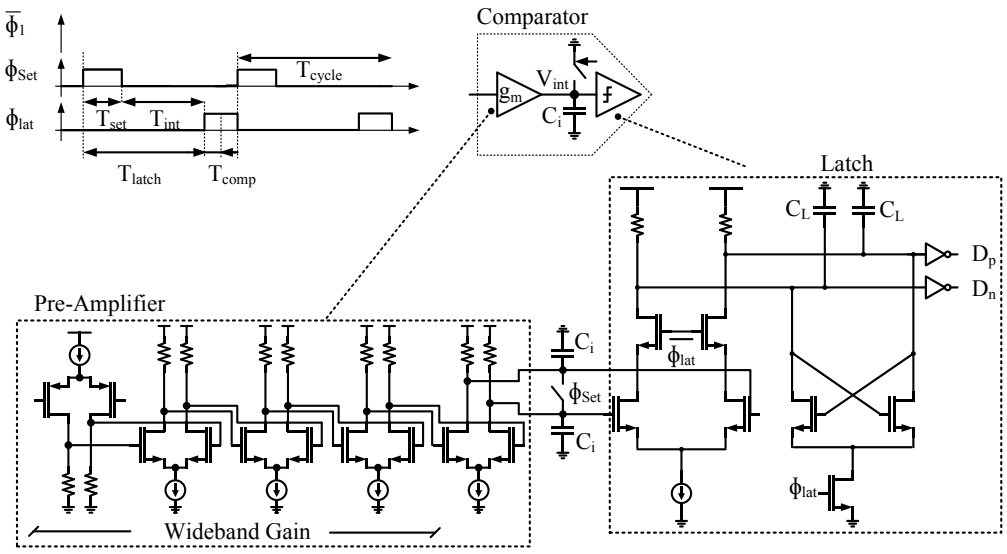

Figure 4.21: Schematic of the complete Comparator

of a single stage is about 3, giving A=3·3·3·3=81. The total preamplifier power is about 6 mW, while the latch adds another 1.8 mW, resulting in a total comparator power of 7.8 mW.

4.4 Summary

We highlighted several implementation details of the SAR ADC using a loop-embedded input buffer. We started with the buffer and how it is integrated in the ADC front-end, followed by a discussion about the current steering DAC, including the DAC self-calibration scheme and self-heating issues of the DAC load resistors. Finally, the comparator architecture and design strategy were highlighted. This concludes the discussion on ADC implementation, and we will next turn our attention to measurement results.

5 Test Setup and Measurement Results

In this chapter, we will show the test setup and present measurement results for the SAR ADC using a loop-embedded input buffer. The measurement results confirm the feasibility of the described buffer nonlinearity cancelation, and highlight the advantage of having an input buffer.

5.1 Test Setup

Figure 5.1 shows the main part of the prototype PCB used to test the ADC. Beginning from the left, we have a signal generator, which is filtered by a bandpass filter to generate an input with a SFDR better than 100 dB. Next, the signal passes through a transformer to generate a differential signal, which then sees a first order RC low pass filter. The transformer has a center tap on the secondary side that sets the input common mode voltage, which is adjustable to match it with the DAC's common mode voltage. After the RC filter, the differential signal enters the ADC. A socket is used for flexibility in testing, allowing us to switch quickly between different chips. All measurements were made using the socket.

The main clock is routed orthogonal to the signal on the PCB minimize coupling between these two paths. An Agilent 8644B source generates the low jitter clock signal, which is converted to a differential signal via a transformer, similar to the input signal path. However, we use a different transformer part (Figure 5.1) that is more suitable for high-speed signals, since the clock signal runs at 350 MHz, while the ADC input signal goes only up to about 25 MHz.

The ADCs digital outputs use a differential LVDS signaling. The bits are serialized to minimize the pin count and to minimize voltage spikes on the pad ring caused by fast digital switching. The chip sends four 4-bit packets during each ADC cycle, giving us 16 raw output bits. A sync bit marks the first package and a clock output signal (CLK) is provided to the data acquisition system (in our case a Logic Analyzer).

The test setup is completed with a serial interface that can load control bits into the ADC's shift register. This register is used to configure some ADC parameters, such as different bias currents or different delays for the asynchronous SAR logic. Furthermore, we can address every current cell and hard wire them to a zero, positive or negative output. This feature is used to implement the self-calibration. It is also possible to route currents from every DAC current source individually to an output pin to perform the DAC calibration off chip.

Finally the data is collected and sent to MATLAB©, were we use the 16 raw bits and the DAC weights (W_C) to assemble the final ADC output. The ADC control bits can be programmed via MATLAB© as well, allowing us to conveniently implement the self-calibration scheme or other automated tests cases.

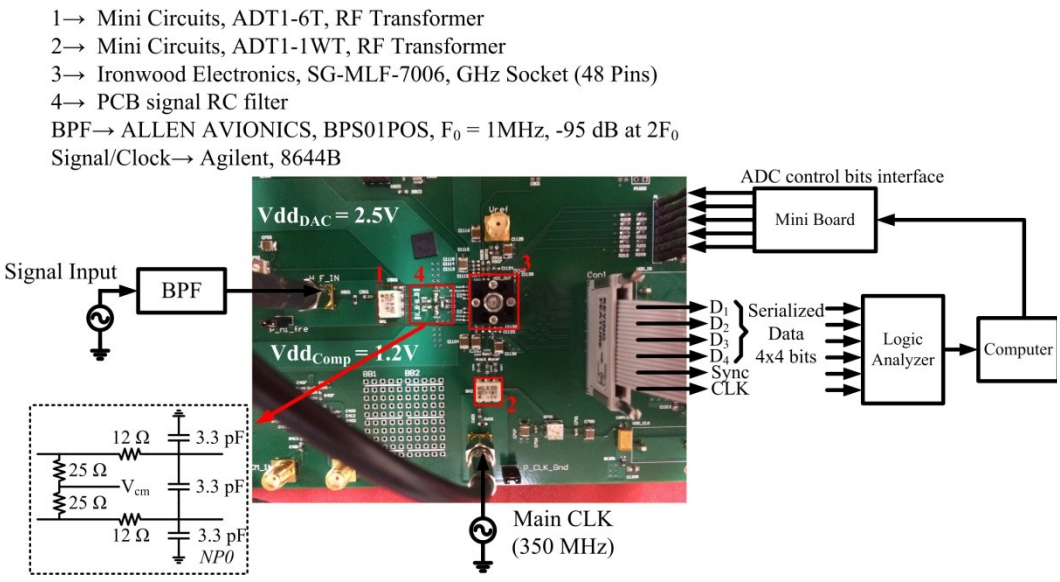

Figure 5.1: PCB test setup and parts list

5.2 Prototype Measurements

In this section, we will first highlight some general properties of the fabricated prototype, like chip area and power break down. Next we present the static differential nonlinearity (DNL) and integral nonlinearity (INL) measurements. From here, we then show the ADC performance versus frequency and highlight the difference between the buffered versus unbuffered version of the ADC. Finally, we compare our ADC to the state of the art.

The ADC was fabricated in a 40 nm LP CMOS process and measures 0.43 mm x 0.55 mm (0.236 mm^2) including the supply and DAC decoupling capacitors (see Figure 5.2). The buffer and DAC consume 12.5 mW (23% of the total) and 30 mW (55% of the total), respectively, from the 2.5 V supply. The clock network, comparator, and SAR logic operate from 1.2 V and consume a total of 12 mW (22% of the overall power) (including the CML clock buffer). This gives a total power dissipation of 54.5 mW including the DAC and its internal reference, but

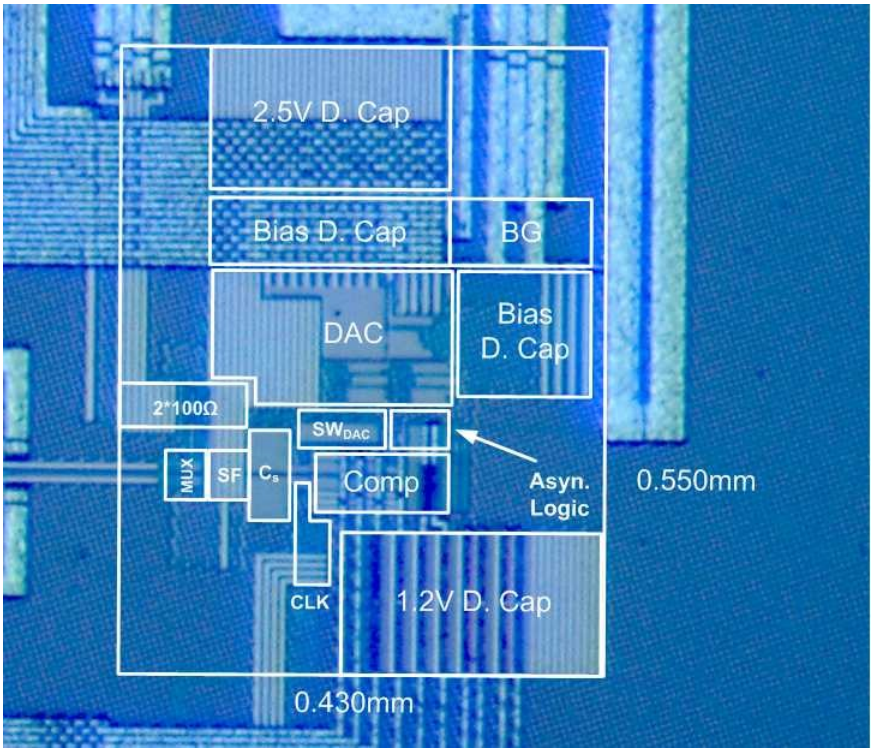

Figure 5.2: Die photograph

excluding the bandgap reference (1.2 mW) and I/Os (90 mW).

The measured DNL and INL characteristics are summarized in Figure 5.3. After DAC weight calibration (performed using off-chip software), the peak DNL and INL lie within 0.73/0.67 with respect to the 14-bit LSB, respectively. Notice that two interesting artifacts can be identified. First, there is a large step in the uncalibrated INL (see Figure 5.3I) and second, there are fewer codes than the expected $2^{14} = 16384$. These two artifacts are related. Recall that we matched only the LSB section to 14 bits, and hence the MSB section uses differently sized transistors and biases. That leads to a large change from the LSB section to the MSB section. For example, M_8 should be 40 LSB, but is actually 37.2 LSB, and M_1 is 3642 LSB instead of 3820 LSB. This mismatch can be seen as a 200 LSB step in the INL. Furthermore this 5-7 % error between MSB and LSB section leads to the

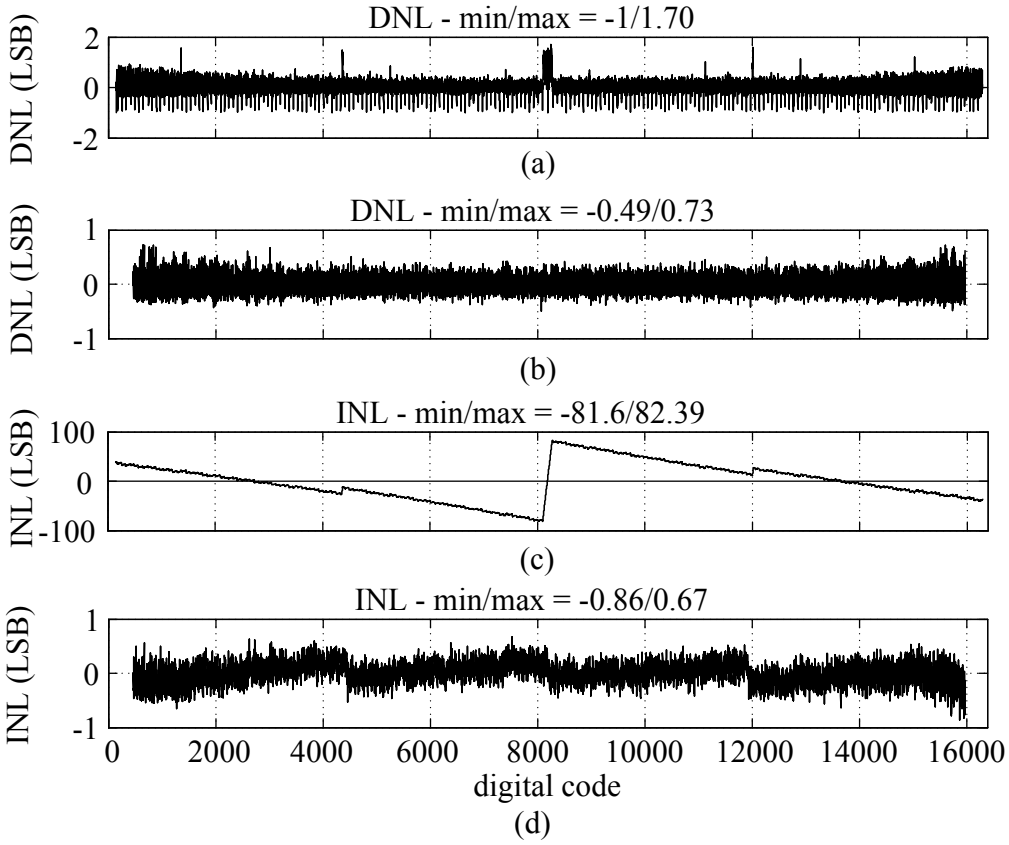

Figure 5.3: Measured ADC DNL (a) before DAC calibration and (b) after DAC calibration. ADC INL (c) before DAC calibration and (d) after DAC calibration

loss of codes. In (4.7) we calculated $V_{FS}/2$ = 7665.1 LSB and hence V_{FS}=15330.2 LSB instead of 16384 LSB.

Figure 5.4(a) shows the output spectrum for a 1 MHz input signal. The measured SFDR of 99 dB confirms that the buffer's nonlinearity is strongly suppressed. Furthermore, we obtained an SNDR of 75 dB, which is consistent with the noise estimation using (4.7) and the presented noise calculation in Section 3. Figure 5.4(b) compares the SFDR versus f_{in} with integrated buffer to that of a non-buffered version of the ADC that was fabricated on the same run. A clear

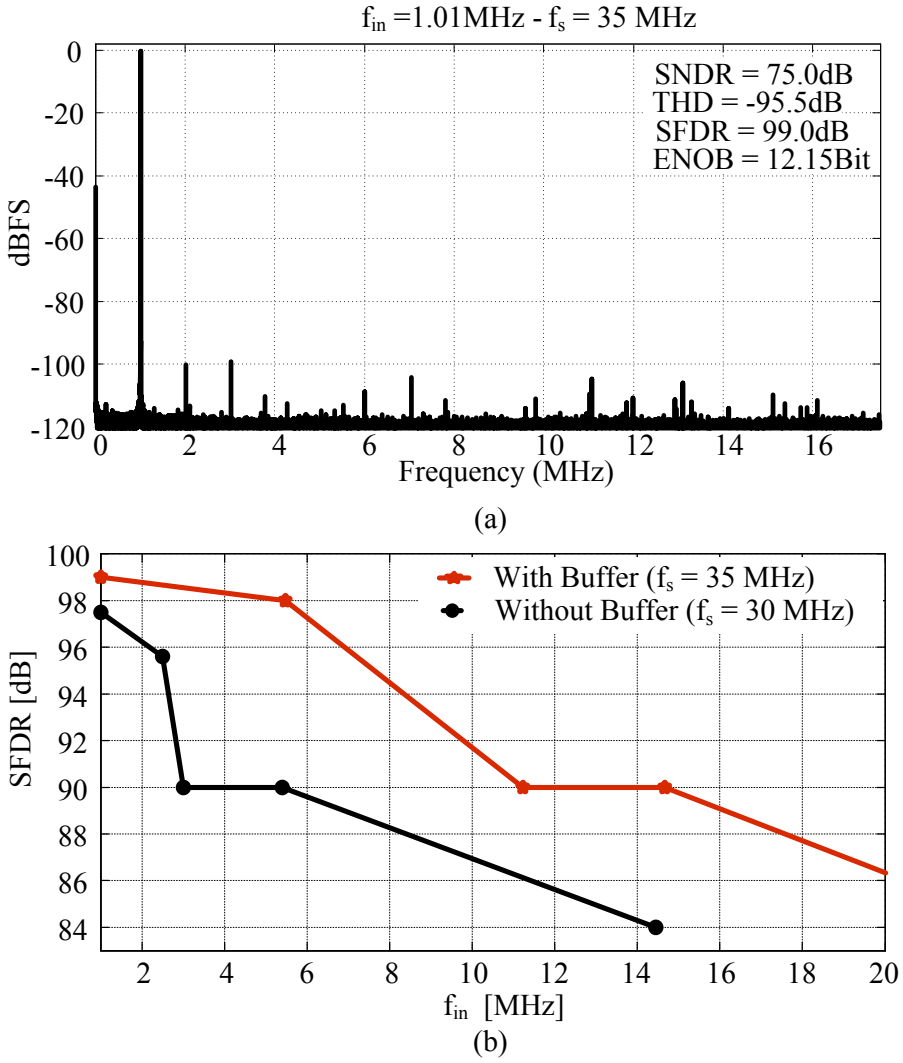

Figure 5.4: (a) Low frequency spectrum. (b) Comparison of buffered vs. unbuffered ADC

advantage of the ADC with buffer can be observed, having an SFDR greater than 90 dB with an input ranging from DC to 15 MHz. In comparison, the SFDR of the ADC without the buffer drops below 90 dB for an input frequency of about 14 MHz. We can also identify a fast SFDR drop to 90 dB for $f_{in} > 2$ MHz. This test was performed without changing the measurement setup between the two chips. The chips are nearly identical, the only difference being that one chip has the buffer removed from the signal path.

Figure 5.5(a) shows the SNDR, HD_2, HD_3 and SFDR vs. f_{in}, indicating peak SNDR = 75 dB, SFDR = 99 dB at low f_{in} and 74.4 dB and 90 dB near Nyquist. These results demonstrate that the nonlinearity cancelation is effective across the entire first Nyquist zone. The high frequency SFDR is limited by HD_2 rather than HD_3 (see Figure 5.5(a)). We conjecture that this is due to the phase imbalance of the balun that performs the single-ended to differential conversion in our test set-up, as described previously. In the design phase of the ADC, we assumed phase errors of up to one degree, which gives an HD_2 of 90 dB for our design. The measured results confirm that we are operating close to this condition. Finally, Figure 5.5(b) and (c) compare the performance with the state-of-the-art designs of [1], [2] to show that this ADC achieves higher SFDR and SNDR within a bandwidth of 18 MHz.

For further comparisons with the state of the art (see Table 5.1), we employ the SNDR-based Schreier FOM [13], evaluated at Nyquist:

$$FOM_{S,nyq} = SNDR(dB)|_{f_{in} \cong f_s/2} + 10\log\left(\frac{f_s/2}{P}\right) \tag{5.1}$$

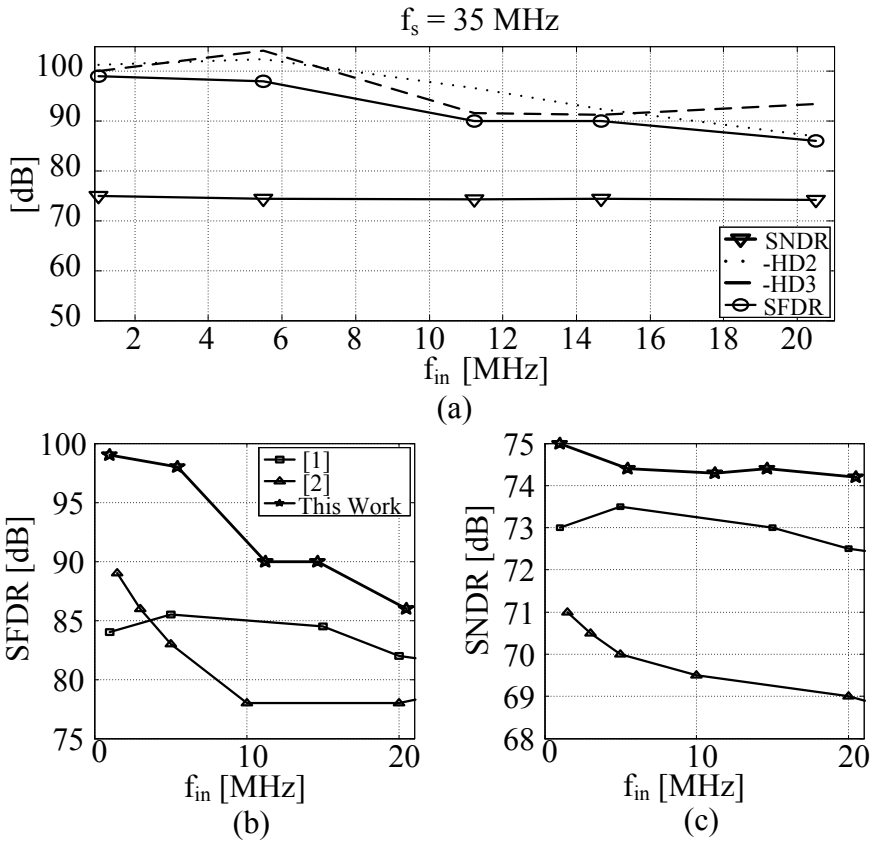

Figure 5.5: (a) Measured performance vs. input frequency, (b) SFDR and (c) SNDR
74onductson to state-of-the-art SAR ADCs

For our design, we find $FOM_{S,nyq}$ = 159.5 dB (including buffer). This number is close to the efficiency of [1] (161.8 dB), which excludes the buffer power. Reference [2] has a higher $FOM_{S,nyq}$ (165.1 dB) and is buffered on chip, but the buffer power is not reported/included. In addition, [2] has ~14 dB lower SFDR and 7 dB lower SNDR at Nyquist, which makes a fair FOM comparison difficult. The design of [33] also has an integrated buffer and its power is included in the reported numbers. The figure of merit for this design is within 2 dB of our work, but the sampling rate is seven times lower.

Table 5.1: Performance summary and comparison

	This Work	[1]	[2]	[34]	[33]
Architecture	SAR	TI-SAR	SAR	SAR	TI-SAR
CMOS technology (nm)	40	65	28	130	130
Resolution (bits)	14	14	15	12	14
Sampling rate (MS/s)	35	80	100	22.5 / 45	5
SNDR at low freq. (dB)	75	73.6	71	71.1 / 68	83
SNDR at Nyq. (dB)	74.4	71.3	67.1	70.1 / 67	82
SFDR at low freq. (dB)	99	85	89.2	94.6 / 90	-88.7 (THD)
min SFDR thru Nyq. (dB)	90	80	76	90.3 / 84.7	-86 (THD)
Integrated buffer	yes	no	yes	no	yes
Power without ref. (mW)	-	-	-	2.81 / 3.02	-
Power with ref. (mW)	42.5	35.1	8.0	-	-
Power of input buffer (mW)	12	-	Not reported	-	-
Total power (mW)	54.5	-	-	-	66
$FOM_{S,Nyq}$ w/ input buf. And ref. (dB)	159.5	-	-	-	157.8
$FOM_{S,Nyq}$ w/o input buf. w/ ref. (dB)	160.6	161.8	165.1	-	-

5.3 Summary

This chapter presented the measurement results of the SAR ADC using a loop-embedded input buffer, as well as the test setup that was used to perform these measurements. By putting the buffer in the feedback loop, we were able to relax the buffer requirements and we are only investing 22% of the total ADC power in the buffer. Furthermore, the buffer eased the input drive requirements, by lowering the ADC input capacitance by a factor of 18, resulting in a highly linear signal acquisition. This ADC achieves the highest SFDR (up till $f_s/2$) among state-of-the-art SAR ADCs with a sampling speed higher than 25 MS/s.

6 A SAR ADC Using Noise Filter Gear Shifting

In this chapter, we describe another concept studied in this research. We call this idea "noise filter gear shifting," and it allows us to run the ADC 22% faster without investing additional power and without sacrificing SNR. First, we will highlight how redundancy can be used to improve the noise speed trade-off. After that, we show how this work leverages this technique, and finally conclude the chapter with measurement results that validate the concept.

6.1 Noise Filter Gear Shifting

We analyzed the noise contributed during the ADC conversion phase in Chapter 3. The result is repeated here for convenience:

$$\overline{V_{CN_{res}}^2} = 8kT\left(G_C^2 R_{DAC}\left(1+\gamma_{DAC}\frac{g_{m_{DAC}}}{I_d}\frac{V_{FS}}{4}\right)+\frac{nf_{comp}}{g_{m_{comp}}}\right)\frac{1}{2T_{int}} \quad (6.1)$$

Notice that we removed the buffer noise and gain in (6.1), which will be justified in the next section. The conversion phase noise is inversely proportional to the comparator integration time T_{int}. To lower the noise, T_{int} has to be increased, resulting in a slower ADC. To maintain a certain speed and to improve the SNR, R_{DAC} must be lowered and g_{mcomp} must be increased. Both actions lead to an undesired increase in power consumption. The same is true for the case where one wants to improve the ADC speed and keep the SNR constant.

Figure 6.1 illustrates this tradeoff, considering only the DAC noise in this simplified example. We start with a baseline ADC (Figure 6.1 (a)) that has a T_{int} = 1.6 ns to achieve SNR = 77 dB, resulting in a conversion speed of 24.5 MS/s. Figure 6.1(b) reduces T_{int} to 0.8 ns making the ADC faster. Notice that the power is not changed between these two cases. However, errors can be identified in the first and fourth decision, due to noise, resulting in a lower ADC SNR, which is in line with (6.1).

As explained in [5] and [35], low-noise comparisons are only needed in the last few decisions (where little or no redundancy is present). In order to benefit from this feature, one must find a way to dynamically adjust the noise to gain improvements in energy and/or speed. One possibility is to use two comparators, one with relaxed noise for the MSBs, and a low-noise version for the LSBs [5]. This primarily helps reduce energy, but comes with overhead in complexity, additional

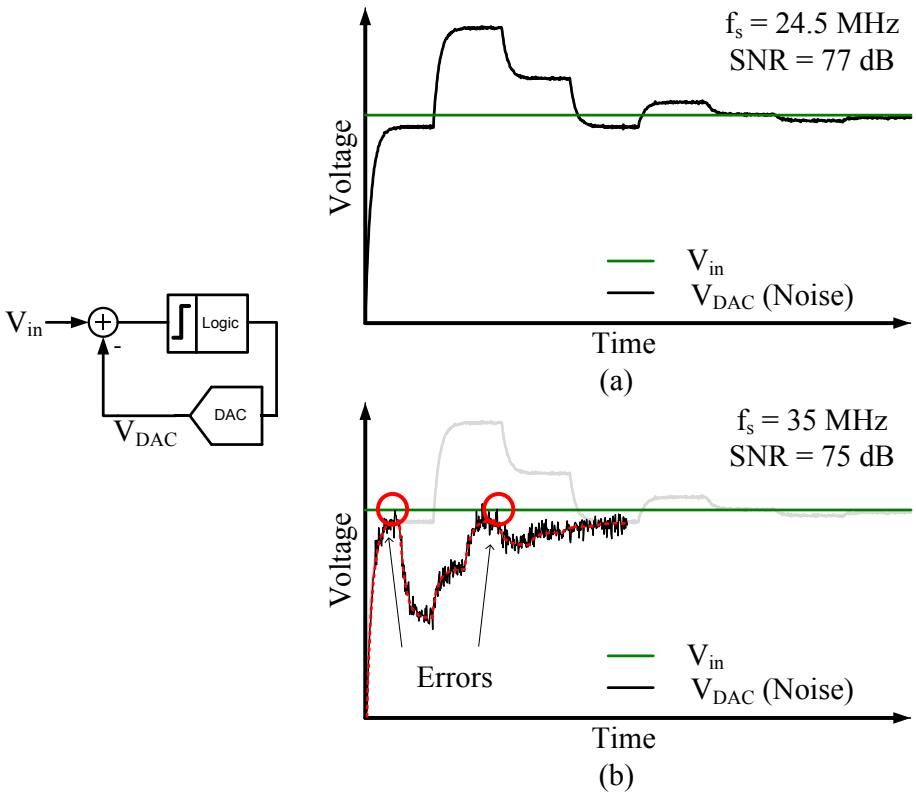

Figure 6.1: Example ADC conversion phase (a) with a large T_{int} (b) a short T_{int}

capacitance at the charge conservation node (resulting in a noise penalty) and the need for extra redundancy to absorb the offset mismatch between the two comparators. Furthermore, this solution would not help to filter the DAC noise. Another option is presented in [36], the 40 kS/s SAR ADC overcomes these issues by using a single comparator that is fired multiple times during critical decisions, thereby achieving low noise on demand via majority voting. Unfortunately, this technique is incompatible with high-speed design, since the critical decisions are detected by measuring the latch decision time. In a high-speed converter with latch time constants of a few picoseconds, the time resolution needed to distinguishing between critical/non-critical decisions becomes impractically small.

The option explored here uses an alternative solution that is similar to [5] in that the comparisons are done with two noise levels, but we employ only one comparator whose noise is varied via the integration time (Figure 6.2) of its G_m-C preamplifier (noise filter gear shifting). In contrast to [5] the comparator power stays constant, but we gain in conversion speed (22%) due to the shorter decisions in the high-noise setting. This scheme is a good fit for designs that combine high-speed and high resolution, since a high-fidelity preamplifier is already required.

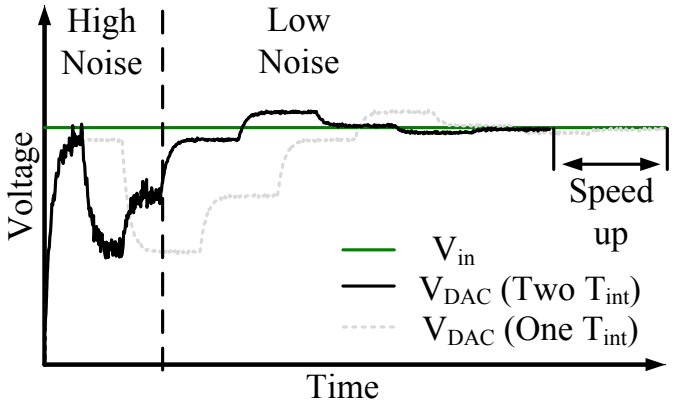

Figure 6.2: ADC conversion phase with two noise states realized with two different integration times T_{int} (solid line) compared to an ADC using a single integration time (dashed line)

6.2 ADC Implementation

Figure 6.3(a) shows the ADC block diagram including the variable integration time. The overall design is nearly the same as the ADC described in Chapters 4 and 5. The main differences between the present design and the on presented earlier are: (1) The input buffer was removed and (2) the gear shifting was added to reduce the DAC and comparator noise. In essence, this means that the present design was optimized for higher SNR and lower power, whereas the design of

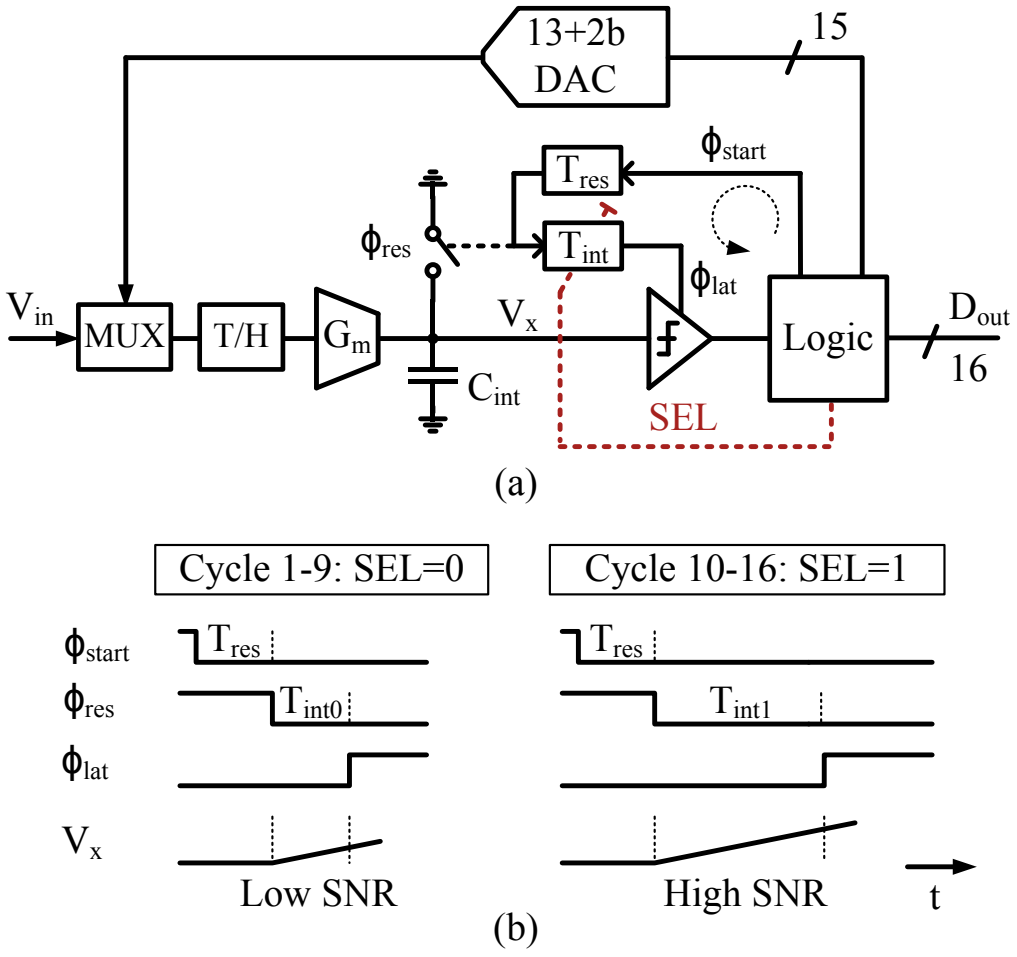

Figure 6.3: (a) ADC block diagram. (b) Timing diagram for the low-SNR MSB decisions (left) and high-SNR LSB decisions (right)

chapter 4 was aimed at easing the input drive requirements (which also helped maximize SFDR) and maximizing speed.

The key innovation of this design lies in the timing of the comparator and its G_m-C pre-amplifier, illustrated in more detail in Figure 6.3(b). At the start of each conversion cycle, the output of the G_m-C pre-amplifier is reset via ϕ_{res} and the DAC input bits are updated. The duration of this reset phase is fixed and defined via a delay line (T_{res} in Fig. 1(a), ~600 ps). After reset, the G_m-C stage integrates its input for a time that depends on the state of the SAR algorithm (programmable delay line T_{int}). During the MSB cycles 1-9, the integration time is short (T_{int0} = 0.85 ns), yielding a low SNR at the output of the G_m-C stage. Any decision errors caused by the extra noise are absorbed by the converter's redundancy.

For the critical LSB decisions in cycles 10-16, the integration time is increased (T_{int1} = 1.6 ns) to improve the SNR. The input-referred noise bandwidth of the switched G_m-C stage is inversely proportional to the integration time (6.1), and thus the input–referred noise power is approximately halved. At the end of the above-described integration phase, the latch is fired to detect the polarity of V_x, which determines the output bit for the present cycle. The conversion then proceeds to the next cycle as usual.

6.3 ADC Measurements

The test setup is the same as described in Chapter 5, since the chips are nearly identical. We only powered down the buffer and wired the MUX directly to the top plate of the sampling capacitor. The DAC uses the same self-calibration scheme as described in Chapter 4.

To evaluate the noise filter gear shifting, we measured the ADC's SNR as a function of integration time (see Figure 6.4(a)). In this measurement, the same T_{int} is used for all cycles and the ADC therefore runs at a reduced speed (20 MS/s). As expected, we see an SNR change as Tint is swept between the two extreme values (T_{in0} = 0.85 ns and T_{int1} = 1.6 ns). The observed change is consistent with

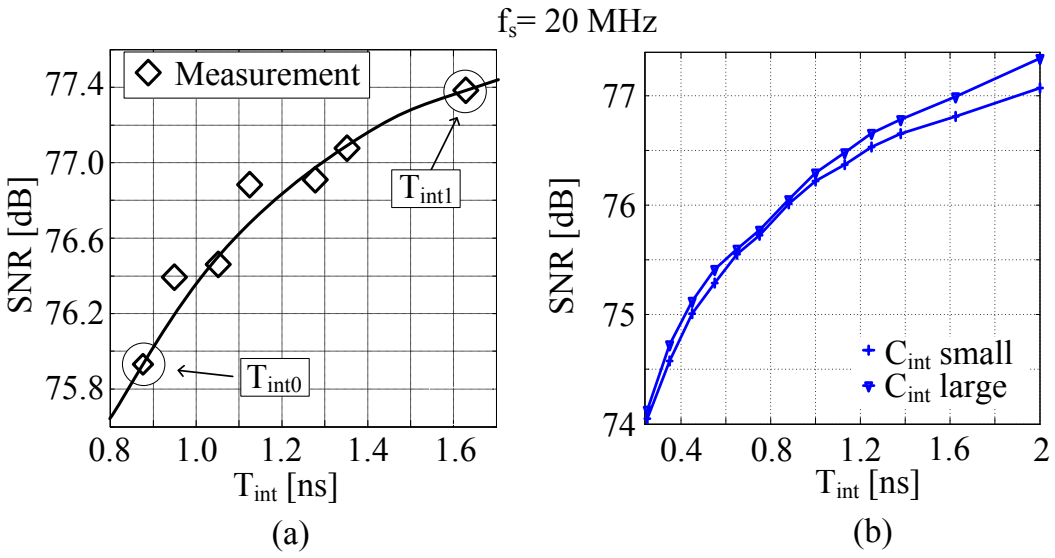

Figure 6.4: (a) measured SNR vs. integration time (T_{int}) (b) measured SNR vs. integration time (T_{int}) for two different integration capacitors C_{int}

halving of the DAC and comparator noise for the large integration time setting (T_{int1}). Note that the measured overall ADC SNR changes by less than 3 dB due to the other noise contributors (kT/C sampling noise, etc.). Furthermore, we are starting to approach the kT/C limit of the gm-C integrator (Appendix A4) for higher integration times. Figure 6.4(b) illustrates this effect even more by plotting the SNR versus integration time for two different integration capacitors, which was designed to be programmable. Notice that the two curves match for short integration time, which is to be expected, since the SNR independent of C_{int} in this regime (6.1). For longer integration times, however, we can see that the two curves start to diverge, showing the beginning of the integrator's kT/C_{int} limit (Appendix A4, equation (A4.2)). These curves will flatten out for longer integration time to the point were no SNR improvement can be gained by increasing the integration time. Figure 6.4(b) indicates that we were able to measure integration times smaller than T_{int0}, however this was accomplished by setting the integration time to its lowest value and then increasing the settling time T_{set} (Figure 3.13).

The SNDR versus f_s plot of Figure 6.5 illustrates the speed advantage gained by the gear shifting. Using T_{int1} for all cycles (dashed line) limits the maximum

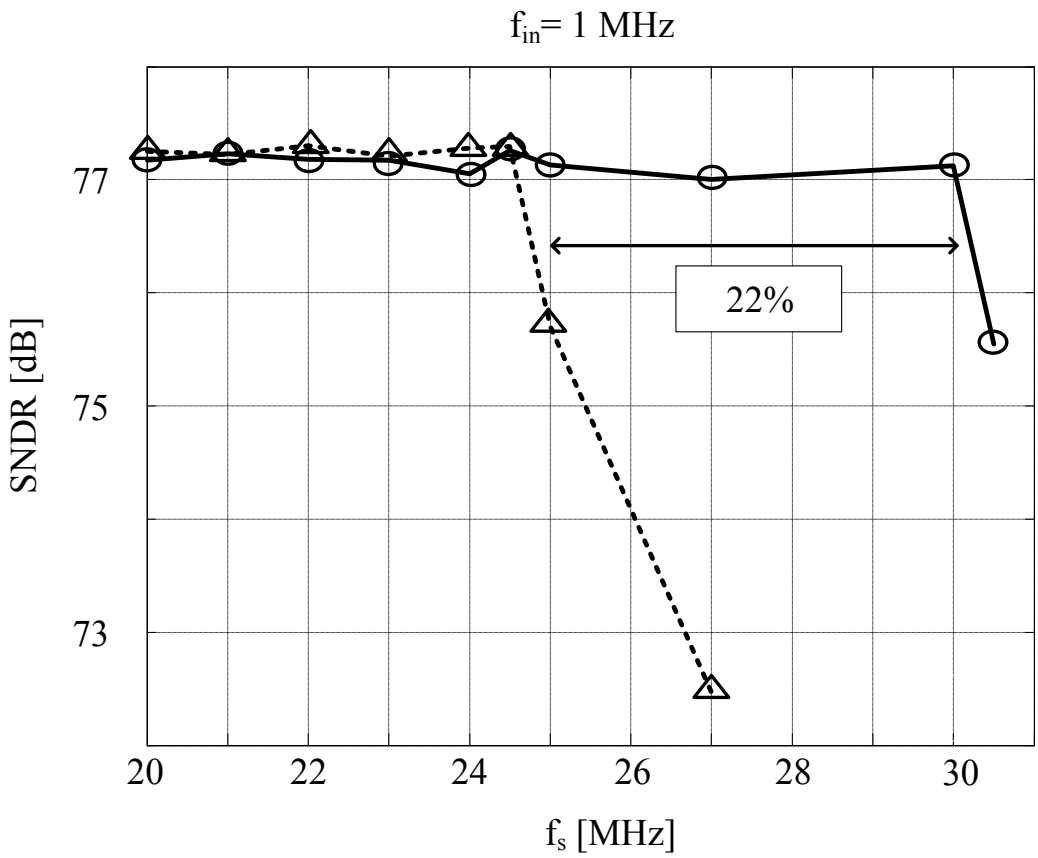

Figure 6.5: SNDR vs. sampling frequency (f_s) using fixed $T_{int} = T_{int1}$ (dashed) and with gear shifting between T_{int0}/T_{int1} (solid)

speed to 24.5 MS/s, while gear shifting between T_{int0}/T_{int1} (solid line) yields 30 MS/s (22 % higher). Note that there is no notable difference in the SNDR. The dramatic SNDR losses are due to the fact that the last several bits are no longer resolved in the given time, hence the 14-bit converter becomes a 13-bit or even a 12-bit converter.

Figure 6.6 summarizes measurements with the gear shifting activated (at 30 MS/s) and shows the output FFT for a 1 MHz full-scale input. The ADC achieves a low frequency SNDR of 77.2 dB, which degrades by 1.5 dB at Nyquist due to distortion. This is to be expected, since we sacrificed the input buffer showing again that it helps reduce the high frequency distortion. The SNR loss over input frequency is about 0.5 dB.

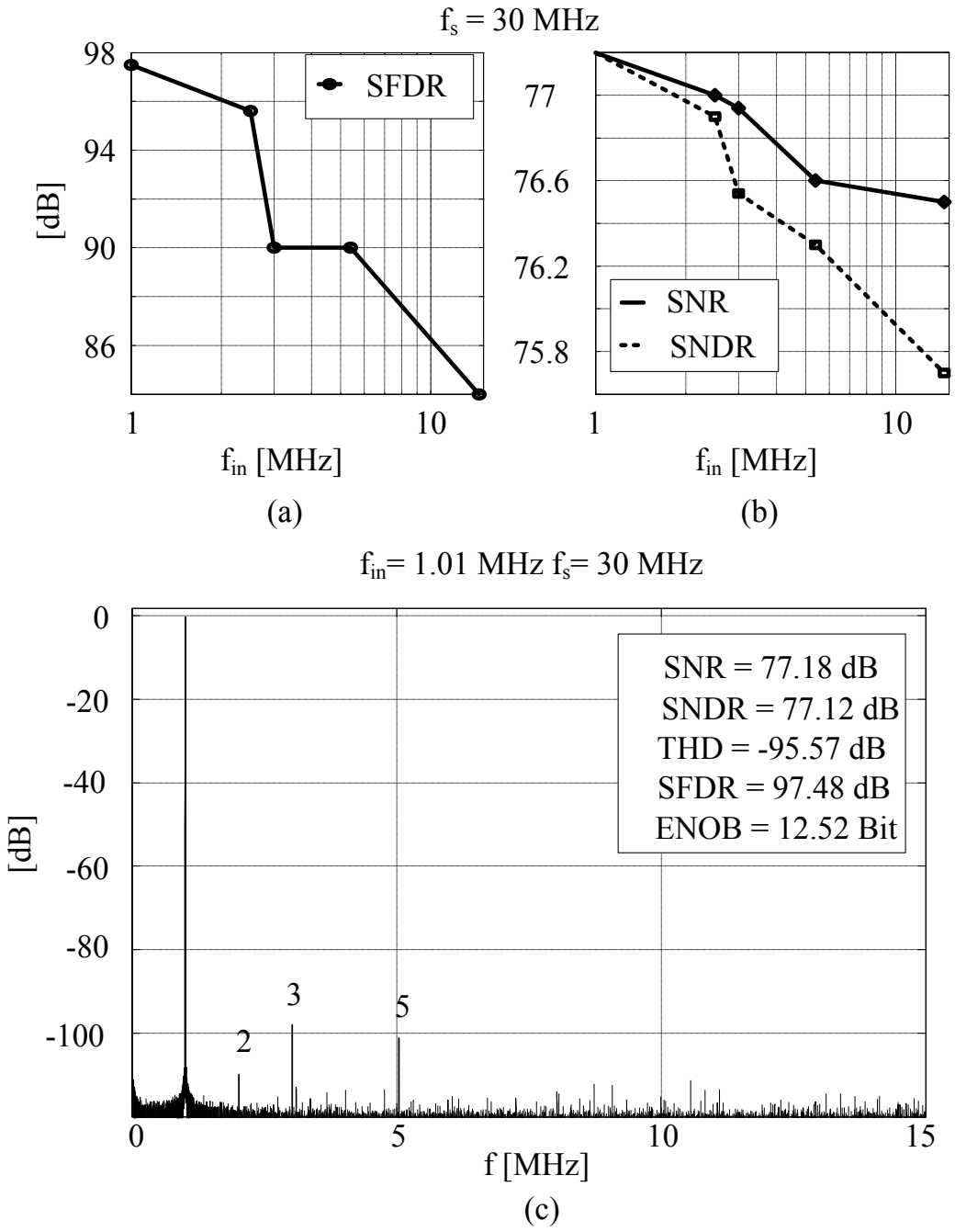

Figure 6.6: (a) Measured SFDR, (b) SNR and SNDR vs. f_{in}.(c) Measured output FFT

The ADC consumes a total power of 38 mW excluding the bandgap and digital I/Os. This number includes 27.3 mW from the DAC and the remainder is due to the comparator, SAR logic, CML clock buffer and clock distribution. The design achieves an SNDR-based Schreier FOM of 163.1 dB for low frequencies and

161.6 dB at Nyquist. Table 6.1 compares our design with the state of the art. Compared to the ADC presented in Chapter 5, the present design achieves higher SNR and lower power, at the expense of a small reduction in speed and SFDR. The loss in speed is due to the large integration time used for the critical decisions, which is key for achieving the reported SNDR. The measured low-frequency SNDR is the highest reported for Nyquist SAR ADCs with sampling rates above 20 MS/s.

The tuning range for the comparator integration time in this proof-of-concept demonstration was limited to a factor of two. Future designs will likely be able to

Table 6.1: Performance comparison

	This Work	[1]	[2]	[6]
Process (nm)	40	65	28	40
Resolution (bits)	14	14	15	14
Sampling rate (MS/s)	30	80	100	35
SNDR at LF (dB)	77.2	73.6	71	75
FOM_S^* at LF (dB)	163.1	164.7	169	160.1
FOM_S^* at Nyq. (dB)	161.6	161.8	165.1	159.5
SFDR up to Nyq. (dB)	84	80	76	90
Total power (mW)	38	35.1	8	54.5

$^*FOM_S = SNDR(dB) + 10\log\left(P/(f_s/2)\right)$

benefit from wider range and could also consider more than two integration time settings. Notice that going to 100-200 ps integration times will yield in an SNR of about 70-74dB, indicating that we could improve the speed even further since we have enough redundancy to recover from noise-induced errors.

6.4 Summary

We introduced the noise filter gear-shifting concept in this chapter and described a prototype implementation. This technique allows us to improve the ADC speed by 22% without increasing the comparator or DAC power significant-

ly. Furthermore, we presented measurements of this ADC showing the highest peak SNDR for SAR ADCs with a sampling speed greater than 25 MS/s.

7 Conclusions and Future Work

7.1 Summary

This research focused on high-resolution, high-speed SAR ADCs. These ADCs suffer from two major challenges: The input drive, which is a significant issue in achieving high linearity, as well as the noise requirements for the comparator and DAC. These blocks require high speed, which leads to a large noise bandwidth and this is typically compensated by increasing the power consumption.

The main part of this work described the analysis, design and measurement of a proof-of-concept SAR ADC with loop-embedded input buffer to address the input drive challenge. The buffer shields the converter's large switched sampling capacitance from the input, thereby easing the input drive requirements and enabling a highly linear signal acquisition. We were able to lower the 3.5 pF input capacitance to 200 fF, an 18-fold reduction. Since the buffer resides inside the SAR conversion loop, its nonlinearities are largely suppressed. We analyzed second-order effects that limit this suppression and found good agreement between analysis, simulation and measurement. The presented ADC achieves SNDR = 75 dB and SFDR = 99 dB at low f_{in} and 74.4 dB and 90 dB at Nyquist, respectively. The design consumes a total power of 54.5 mW and runs at 35 MS/s. Comparison with the state of the art shows that this ADC achieves the highest SFDR through the first Nyquist zone for SAR ADCs with a sampling rate above 25 MS/s. However, the figure of merit ($FOM_{S,nyq}$ = 159.5 dB) for this design is less competitive, mainly owing due to the power overhead from the integrated current steering DAC, as well as the expended buffer power.

Additionally, this work described a second ADC with a design technique that helps alleviate the stringent tradeoff between comparator/DAC noise and speed in high-speed, high-resolution SAR ADCs. By shortening the integration time of the G_m-C preamplifier during non-critical decisions (with redundancy), we demonstrated a 22% speed-up compared to the standard design configuration with the same SNDR while keeping the power nearly constant. This technique, called noise filter gear shifting, was implemented in a proof-of-concept SAR ADC achieving the highest peak SNDR = 77.2 dB for SAR ADCs with a sampling rate above 25 MS/s. The converter runs with a sampling speed of 30 MS/s and consumes 38 mW, resulting in a low frequency FOM_S = 163.1 dB and at Nyquist $FOM_{S,nyq}$ = 161.6 dB, which is in line with the state of the art.

7.2 Future Work

7.2.1 DAC

Both of the presented ADCs suffer from the inefficiency of the class-A operation of the current steering DAC. It should be emphasized that none of the presented design advancements (loop-embedded input buffering or noise filter gear shifting) critically rely on the current steering DAC architecture.

The loop-embedded input buffer architecture could be modified to use a switched capacitor DAC instead of the current steering DAC. This is viable since the buffer presents a high-impedance input and can be used as a charge-conserving node. Additional research would be needed to investigate the details of this interface, specifically in terms of implementation details and the impact of the (weakly) nonlinear input capacitance of the buffer.

The FOM of the noise filter gear shifting ADC would improve significantly if a traditional switched capacitor DAC was used. Notice that 27.3 mW of the 38 mW of total power is consumed by the DAC alone; this is almost 72 percent. We

can quickly calculate that a SF reference buffer capable of driving the capacitive DAC would only need 1 mA of current to achieve a drive resistance of 100 Ω (g_m = 10 mS and g_m/I_D=10 1/V) . Hence, it is fair to assume that the DAC plus reference buffer could be built with about 2-3 mW of power, giving us a total ADC power of 13 mW. These calculations lead to FOM_S = 167.8 dB at low f_{in} and $FOM_{S,nyq}$ = 166.2 dB at Nyquist.. Furthermore, we would also improve the noise performance. Recall that the current steering DAC noise is given by (3.35):

$$\frac{\overline{V_{DAC}^2}}{\Delta f} = 8kTR_{DAC}\left(1+\gamma_{DAC}\frac{g_{m_{DAC}}}{I_d}\frac{V_{FS}}{4}\right) \approx 8kTR_{DAC}\cdot 3 \qquad (7.1)$$

where the factor three comes from the approximation $V_{FS}/4$ = 0.45 V and g_m/I_D = 4 1/V. The noise from a SF reference buffer would give us:

$$\frac{\overline{V_{SF}^2}}{\Delta f} = 8kT\frac{1}{g_{m_{SF}}}\left(1+\frac{g_{m_{CS}}}{g_{m_{SF}}}\right) \approx 8kT\frac{1}{g_{m_{SF}}}\cdot 1.3 \qquad (7.2)$$

Equation (7.2) shows that the SF DAC reference buffer would only contribute half the noise compared to the current steering DAC, improving the FOM even further. This quick overview is meant to re-emphasize how compelling the noise filter gear shifting is; even with the major disadvantages of the current steering DAC it was possible to achieve state-of-the-art performance.

Another interesting research direction would be to re-use the current steering DAC instead of removing it. A switched-capacitor DAC has very limited drive capability, but the current steering DAC could be used to drive other circuits in an SoC environment. We could think about using this in a system, were both functions, ADC and DAC, are needed but do not run at the same time. For example, in a half-duplex communication system, where one only receives or transmits data.

7.2.2 Advanced Front-Ends

We have shown that it is possible to improve the linearity of a SF buffer by putting it into the SAR feedback loop. Another direction could be to extend this idea to other interface circuits. For example, the transimpedance amplifier looks like a good candidate. It is possible to place the amplifier into the SAR loop (Figure 7.1), and using the current steering DAC directly without a current to voltage conversion via the resistor, hence canceling the nonlinearities of the am-

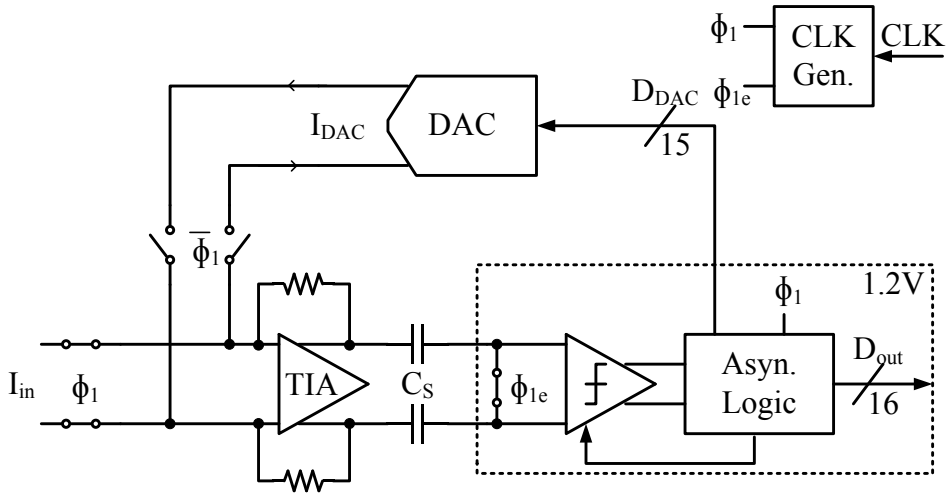

Figure 7.1: Advanced ADC front-end using a TIA in the SAR loop

plifier. However, more research is needed to show if this can lead to a more efficient design.

Another limiting factor is the phase imbalance, which affects standard ADC interfaces as well. Further research could try to sense this error and adjust for it. The sensing mechanism would be the CM voltage, since a phase imbalance results in a differential to common-mode conversion. However, it is not obvious how to control the phase of the single input path.

The source follower nonlinearity can be effectively suppressed by the SAR loop, because the nonlinearity can be modeled by a static model. One further avenue of research would be to correct for the known nonlinearity in the digital domain. This would remove the input MUX, giving us a traditional buffered ADC. However, it is not clear how one could obtain the model parameters of the SF buffer; this is another interesting path that could further improve the ADC interface.

7.2.3 Noise Filter Gear Shifting Improvements

Although we demonstrated the effectiveness of the noise filter gear shifting, we know it can be further improved. One possible direction could be to couple the integration time T_{int} to the available redundancy. This idea is inspired by [1], which couples the settling error to the redundancy. Notice that our measurements show that we could have gone to 300-400 ps of integration time instead of 750 ps, improving the speed by 35 % instead of 22 %. However, our implementation was limited to a minimum delay time between 700 and 800 ps.

This concludes this thesis and I want to end with a quote from Professor Hubert J. Farnsworth, which was used during some grim times that every researcher encounters: “Buddha! Zeus! God! One of you guys, do something! . . . Help! Satan! You owe me!!![8]”

[8] It is quite common for scholars to make all kinds of deals with the devil, a prominent example would be the agreement between Dr. Faust and Mephisto.

Appendix

A1 Calculating the Number of Steps and Weights for the SAR Algorithm

We will show here how we arrived at the weights W for the SAR algorithm with redundancy. The beginning of the derivation is independent of the ADC architecture used in this work. However, the last part is somewhat connected to the implemented ADC. This is a common problem with redundancy, many optimization strategies exist and they usually depend on the used architecture. We are not claiming here that our strategy is the best; however, it fits our needs very well. In order to understand this appendix, it can be useful to read this thesis up until the end of Chapter 4 and then come back to this section.

We use the radix < 2 approach, also known as "beta-expansion"[37]. However, we take it a bit further by customizing the weights. The binary weight vector can be calculated using the following equation:

$$W_B(k) = 2^{N_B-k} \ for\ 1 \le k \le N_B \tag{A1.1}$$

where N_B is the number of ADC bits.. For example, for $N_B = 4$ (A1.1) gives:

$$W_B(k) = 2^{5-k} = [16\ 8\ 4\ 2\ 1] \tag{A1.2}$$

We can generalize this approach using the "beta-expansion":

$$W_R(k) = \alpha \cdot \beta^{N_R-k}\ for\ 1 \le k \le N_R \tag{A1.3}$$

defining β and α as follows:

$$\beta = 2^{\frac{N_B}{N_R}} \quad and \quad \alpha = (\beta - 1)\, for\, N_R \geq N_B \tag{A1.4}$$

N_R is the number of bits including redundancy, for example with N_B=5 and N_R=6, we get $\beta = 2^{\frac{5}{6}} \approx 1.782$ and $\alpha = (\beta - 1) \approx 0.782$:

$$W_R(k) = [14.04 \;\; 7.88 \;\; 4.42 \;\; 2.48 \;\; 1.39 \;\; 0.78]$$

Notice that α normalizes the full-scale value of W_R to the W_B full scale value, hence not wasting any redundancy in an over range [9].

Only one question is not answered so far, what is the optimum number of additional steps or what N_R should we use? We will answer this question numerically, by implementing W_R for a number of N_R's and calculating the minimal conversion time not violating the redundancy requirement. We will demonstrate this by running through one implementation and presenting a Matlab© script that can run several implementations.

The start design has the following parameters: N_B=14 and N_R=15 ($\beta = 2^{\frac{14}{15}} \approx 1.9098$ and $\alpha \approx 0.9098$) giving us W_R:

$$W_R = [7805, 4087, 2140, 1121, 587, 307, 161, 84, 44, 23, 12, 6, 3, 2, 1]$$

Notice that we rounded the weights in the above vector.

These weights give redundancy, which can be calculated using (2.5) and is plotted in Figure A1.1(a). Notice that step 10 to 12 only have one LSB redundancy, while the last two of steps do not have any redundancy. Next we can calculate the tolerable settling error using (2.7). Notice that we only need the settling error at the end of the cycle and all Ds are set to 1. Furthermore, to keep the approach general we use a normalization and let $T_C/\tau = N_S$. N_S is the number of time constants, which defines how long we allow the DAC voltage to settle. Figure A1.1(B) shows the redundancy q(k) as well as the settling error $V_{set\ error}(k)$ for two different N_S. The settling errors increase for smaller N_S values and moving to-

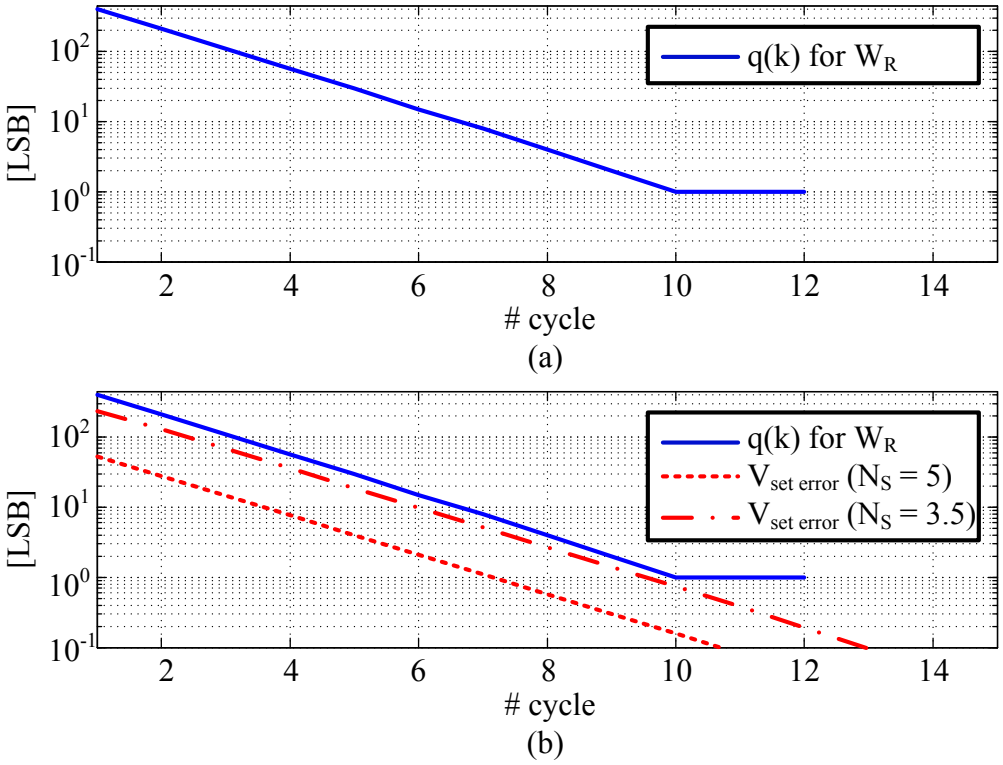

Figure A1.1: (a) max redundancy q(k) versus number of cycle, (b) q(k) and settling error (for two different N_S) versus number of cycle

wards q(k), which also could be called tolerable settling error. Notice that we cannot have any settling error above the solid line (q(k)), otherwise we introduce non-recoverable errors.

We use a numerical approach and minimize N_S, while Q-$V_{set\ error}$ is still positive. The number of settling time constant can be used to calculate the actual conversion time using the timing information from Chapter 4:

$$T_{conv} = \left(N_s\tau + T_{comp} + T_{int}\right) \cdot (N_R - 1) \qquad \text{(A1.5)}$$

$$\tau = 250\ ps, \qquad T_{int} = 0.75\ ns, \qquad T_{comp} = 150\ ps$$

The baseline ADC would need 10 time constants to settle (N_s = 10) and has N_R = N_B = 14, evaluating (A1.5) therefore gives T_{conv} = 44.2 ns. We use 5.7 ns for settling and hence we can get a T_{total} = 49.9 ns or approximately f_s = 20 MHz. We find a minimum N_S = 3.06 for N_R = 15, giving us T_{conv} = 23.3 ns or f_s = 34.5

MHz, improving the ADC speed by 72 %, and nearly halving T_{conv}. Figure A1.2 shows an example Matlab© code that repeats the procedure for several different N_{RS}.

```
clear;
NB=14; % Number of Bits of original ADC
NSOs=1:0.01:5; % Sweep vector of number of time constants NS
tau = 250e-12; % DAC single pole time constant
comp = 150e-12; % comparator and logic delay
Tin = 0.75e-9; % gm-c integration time
Tsamp = 5.7e-9; % Sampling time
Time_base=(10*tau+comp+Tin)*(NB-1); % Conversion time of the
base line ADC, no redundancy
% cycle through additional redundancy cycles 1 till 10
for ii=1:1:10
    %%%% Generating weights Vector %%%%%
    N1=0:1:NB+ii;
    NN1=sort(N1,'descend');
    % calculating beta using (A1.4)
    beta=2^(NB/(ii+NB));
    % calculating W with no over rang
    W_G=(beta-1).*(beta.^NN1(2:end));
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
         %%%%%% finding the number of time constant NS to stay
        %%%%%% within the Redundancy
        for jj=1:1:length(NSOs)
            % getting redundancy using (2.5) for each step
            [ Q ] = Get_R( W_G );
            % calculating setting error in each step using (2.7)
            [ Vr ] = Verr_s(W_G,NSOs(jj));
            % calculating the error between Q and settling error
            error=Q-Vr(1:end-1);
            R(jj)=min(error);
        end
        % finding the first R that is greater than 0, that means
        % that error is greater then 0
        k1=find(R>0);
        R(k1(1));
        % getting minimum number Ns
        NS(ii)=NSOs(k1(1));
        % calculating the conversion time of new ADC
        Time(ii)=(NS(ii)*tau+comp+Tin)*(NB+ii-1);
end
Cy=NB:1:NB+10;
fs=[1./(Time_base+Tsamp) 1./(Time+Tsamp)];
```

Figure A1.2: Matlab© code to find the ADC sampling speed (f_s) for different numbers of Bits (N_{RS})

The code from Figure A1.2 is used to plot f_s versus N_R (see Figure A1.3). We have a maximum for a N_R between 16 and 17 of about 35 MS/s. However, the optimum is shallow and hence we have some freedom to implement the weights.

We start with the weights using N_R=15 and modify them by adding an additional step, which is not created using the beta expansion approach. Instead, we add an additional step at the point where we segmented the DAC in order to give enough range to allow the self-calibration. That leads us to W_{RDAC} as given below. Furthermore, the MSB weights have been modified so that they can be divided by 4. This is convenient for layout, since we divide the weights by two already, to create the split current source and hence the split sources still have an even number. Notice that we divided W_R by two and let the input be in the range from –FS/2 to FS/2:

$$W_R = [3902, 2043, 1070, 560, 293, 154, 80, 42, 22, 12, 6, 3, 2, 1]$$

$$W_{RDAC} = [3820, 1960, 1080, 600, 320, 160, 80, 40, 40, 20, 11, 6, 4, 2, 1]$$

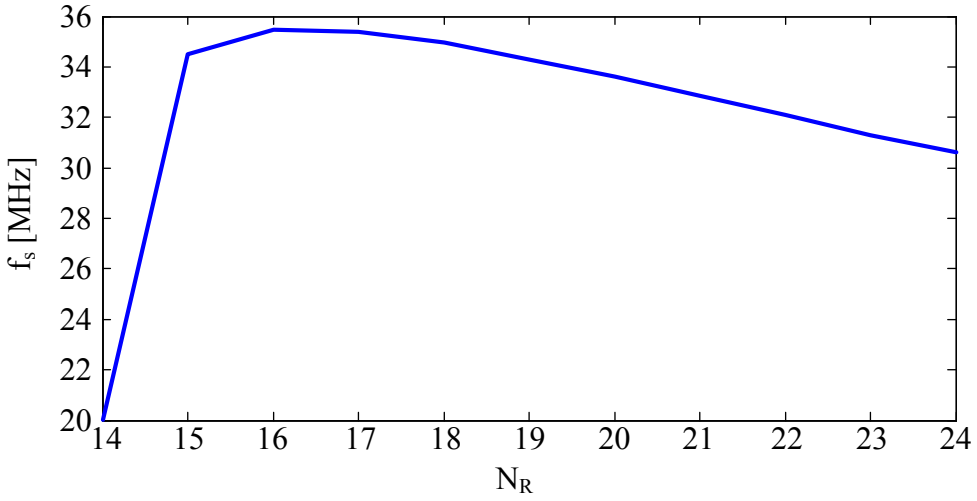

Figure A1.3: Sampling speed (f_s) versus number of bits (N_R)

Figure A1.4 shows the redundancy and the settling error for both implementations. The large redundancy bump at step 7 in Figure A1.4(b) is by design, this is where we switch between the two DAC segments. Notice that we can divide the MSB section W_{MSB} = [3820 1960 1080 600 320 160 80 40] by 40, hence giving us also a unit cell for the MSB section. Figure A1.4(b) shows that we could use N_S = 2.8, if only settling errors are considered. It should be noted that the single pole DAC settling is somewhat oversimplified. We use τ = 250 ps, but the DAC resistor is 100 Ω and the buffer input capacitance is about 200fF giving us τ = 20 ps. This is a factor 10 lower than what was used in the calculation. We use this factor of ten to account for other error source. For example, the current sources of the DAC can see disturbances during the switching events, and these have long settling times. Furthermore, due to the split current source approach, an offset is introduced, which disappears once the current cell is switched to one side. Hence,

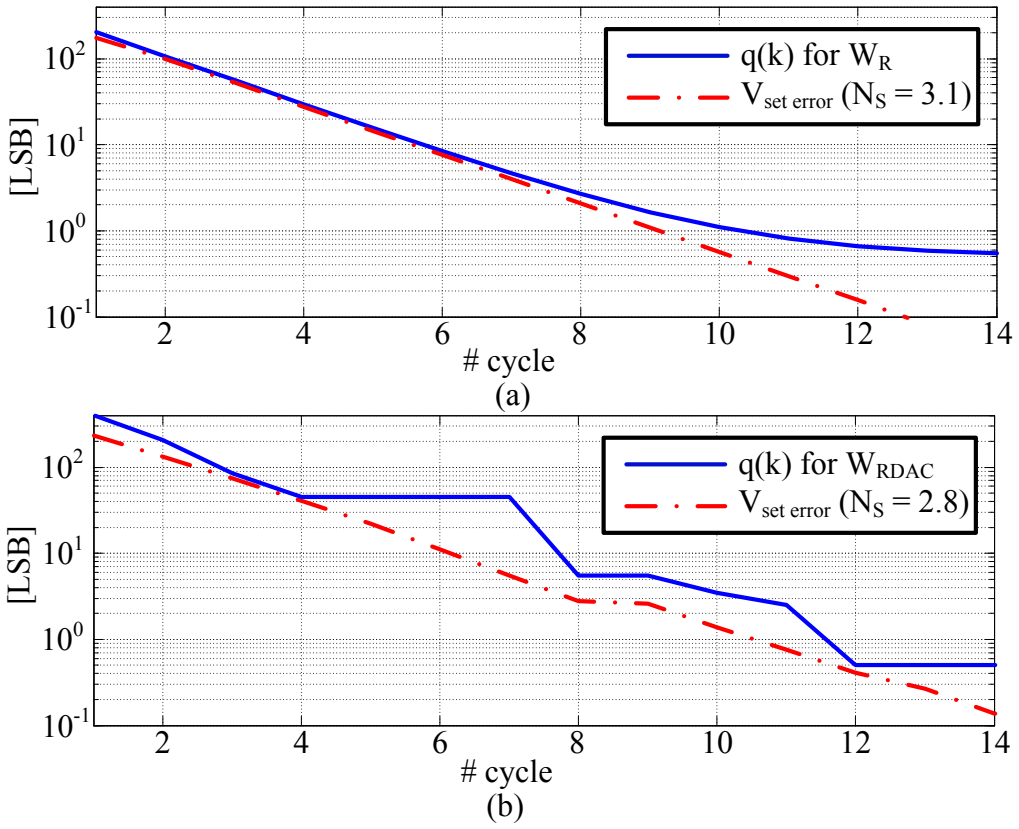

Figure A1.4: (a) redundancy and settling error versus number of cycle for W_R using N_R = 15, (b) redundancy and settling error versus number of cycle for W_D and N_R = 16

these offsets should be covered by the redundancy. In the end, we implemented a programmable DAC settling time in order to address these uncertainties.

A2 Fully Differential Signals and the Effects of Phase Imbalance

We will derive y_1 for the input buffer path including phase imbalance. Figure 3.8 is used to set up the analysis framework. We start as in [15] and using (3.15)-(3.17) to write:

$$y_1 = a_1(v_{ip} - v_{in}) + a_2(v_{ip}^2 - v_{in}^2) + a_3(v_{ip}^3 - v_{in}^3) \tag{A2.1}$$

From here, we obtain:

$$\begin{aligned} y_1 = &\left(a_1\hat{v}_{in} + \frac{3a_3\hat{v}_{in}^3}{4}\right)([1+\cos(\varphi)]\sin(\omega t) + \cos(\omega t)\sin(\varphi)) \\ &-\left(\frac{a_2\hat{v}_{in}^2}{2}\right)([1-\cos(2\varphi)]\cos(2\omega t) + \cos(2\varphi)\sin(2\omega t)) \\ &-\left(\frac{a_3\hat{v}_{in}^3}{4}\right)([1+\cos(3\varphi)]\sin(3\omega t) + \cos(3\omega t)\sin(3\varphi)) \end{aligned} \tag{A2.2}$$

[9]

We can use an in-phase and quadrature approach to simplify (A2.2) by using the known relation:

$$a\sin(x) + b\cos(x) = c\sin(x + \varphi_q) \tag{A2.3}$$

where

$$c = \sqrt{a^2 + b^2} \quad \text{and} \quad \varphi_q = \text{atan2}(b, a) \tag{A2.4}$$

The phase terms φ1, φ2 and φ3 can be calculated using (A2.4):

$$y_1 = \left(a_1\hat{v}_{in} + \frac{3a_3\hat{v}_{in}^3}{4}\right)\sqrt{(1+\cos(\varphi))^2 + \sin(\varphi)^2}\,\sin(\omega t + \varphi_1) \tag{A2.5}$$

[9] Notice that the 3rd order term in [15] has an error, they flipped a sign.

$$-\left(\frac{a_2\hat{v}_{in}^2}{2}\right)\sqrt{(1-\cos(2\varphi))^2+\sin(2\varphi)^2}\,\sin(2\omega t+\varphi_2)$$

$$-\left(\frac{a_3\hat{v}_{in}^3}{4}\right)\sqrt{(1+\cos(3\varphi))^2+\sin(3\varphi)^2}\,\sin(3\omega t+\varphi_3)$$

Finally, by employing the following identities:

$$\sqrt{(1+\cos(x))^2+\sin(x)^2}=\sqrt{2(1+\cos(x))}=2\cos\left(\frac{x}{2}\right) \quad \text{(A2.6)}$$

$$\sqrt{(1-\cos(x))^2+\sin(x)^2}=\sqrt{2(1-\cos(x))}=2\sin\left(\frac{x}{2}\right) \quad \text{(A2.7)}$$

we arrive at a more useful way of writing y_1 as function of time and phase:

$$\begin{aligned} y_1(t,\varphi)=&\left(2a_1\hat{v}_{in}+\frac{3a_3\hat{v}_{in}^3}{2}\right)\cos(\frac{\varphi}{2})\sin(\omega t+\varphi_1)\\ &-a_2\hat{v}_{in}^2\sin(\varphi)\sin(2\omega t+\varphi_2)\\ &-\left(\frac{a_3\hat{v}_{in}^3}{2}\right)\cos\left(\frac{3\varphi}{2}\right)\sin(3\omega t+\varphi_3) \end{aligned} \quad \text{(A2.8)}$$

A3 Noise Contribution of the Buffer and Sampling Switch (Sampling Phase)

Figure A3. (a) shows the schematic of the source follower and the sampling network. We have three noise sources in the circuit, the noise of the two transistors M1 and M2, and the noise of the sampling switch. Figure A3.1 (b) is used to analyze the noise due to the transistors. The current to V_{out} transfer function is:

$$\frac{V_{out}}{I_n} = \frac{1}{g_{m1}} || \left(\frac{1}{sC_S} + R_{SW}\right) \cdot \frac{\frac{1}{sC_S}}{\frac{1}{sC_s} + R_{SW}} = \frac{1}{g_{m1}} \frac{1}{1 + sC_S\left(\frac{1}{g_{m1}} + R_{SW}\right)} \tag{A3.1}$$

Notice that the noise current (Figure A3. (b)) simplifies to:

$$\frac{\overline{I_n^2}}{\Delta f} = 4kTg_{m1}\left(1 + \frac{g_{m2}}{g_{m1}}\right) = 4kTg_{m1}nf \tag{A3.2}$$

The total integrated noise due to the transistors M1 and M2 is then:

$$\overline{V_{M1M2}^2} = \int_0^\infty \frac{\overline{I_n^2}}{\Delta f}\left(\frac{V_{out}}{I_n}\right)^2 df = \frac{\frac{kT}{g_{m1}}nf}{C_S\left(\frac{1}{g_{m1}} + R_{SW}\right)} \tag{A3.3}$$

Next, we will calculate the noise contribution of the sampling switch. Again, we can calculate the transfer function from the noise source to the output voltage using Figure A3.1:

$$\frac{V_{out}}{V_{nSW}} = \frac{\frac{1}{sC_S}}{\frac{1}{g_{m1}} + R_{SW} + \frac{1}{sC_S}} = \frac{1}{1 + sC_S\left(\frac{1}{g_{m1}} + R_{SW}\right)} \tag{A3.4}$$

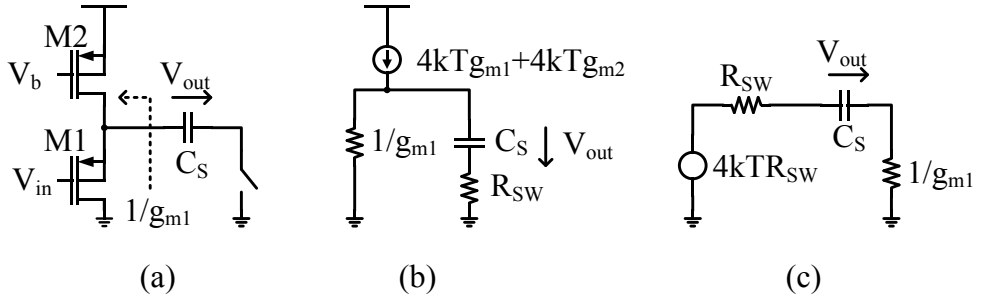

Figure A3.1: (a) Schematic of the buffer plus bottom plate sampling network. (b) Noise model including the transistor noise sources. (c) Noise model of the bottom switch.

This results in the total integrated noise due to the sampling switch given by:

$$\overline{V_{nSW}^2} = \int_0^\infty 4kTR_{SW}\left(\frac{V_{out}}{V_{nSW}}\right)^2 df = \frac{kTR_{SW}}{C_S\left(\frac{1}{g_{m1}} + R_{SW}\right)} \tag{A3.5}$$

Adding (A3.5) and (A3.3) now yields the total noise contributed by the buffer and the sampling switch:

$$\overline{V_{SN_{SW}}^2} = \overline{V_{nSW}^2} + \overline{V_{M1M2}^2} = \frac{kT}{C_S} nf \tag{A3.6}$$

A4 Noise Analysis of the g_m-C Integrator

Figure 3.17 introduced the g_m-C noise filter and this appendix gives some deeper insight into the noise filter operation. The analysis does not start from first principals, since this has been in done in [23]. We start from the equation that calculates the total integrated input-referred noise, which was derived in [23]:

$$\overline{V_{in}^2} = \frac{\overline{V_{out}^2}(t)}{G(t)^2} = \frac{\frac{kTR_{in}}{C_{int}} g_m^2 R_L\left[1 - e^{-\frac{2t}{\tau_0}}\right]}{\left(-g_m R_L\left[1 - e^{-\frac{t}{\tau_0}}\right]\right)^2} \tag{A4.1}$$

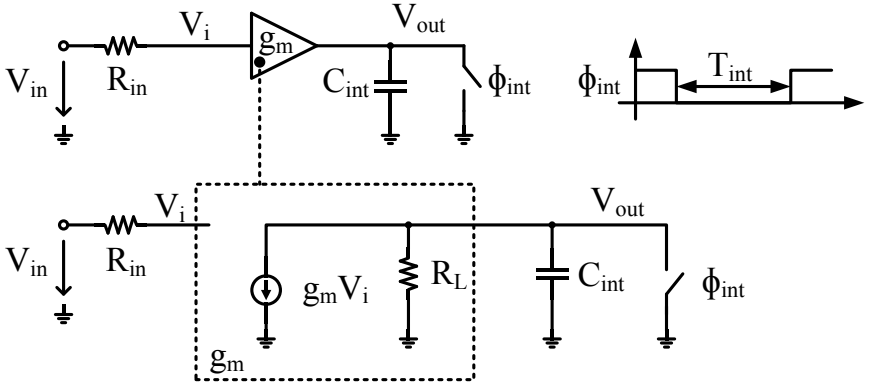

Figure A4.1: Small-signal model of the g_m-C integrator

where R_L is the output impedance of the g_m stage, C_{int} is the integration capacitor and $\tau_0 = R_L C_{int}$. Notice that the small signal model for the g_m-C integrator shown in Figure A4.1 was used to derive A4.1.

First, we consider (A4.1) for $t \gg \tau_0$, meaning that the exponential terms disappear and the circuit operates like an amplifier that fully settles. This yields the following input referred noise:

$$\overline{V_{in}^2} = \frac{\frac{kTR_{in}}{C_{int}} g_m^2 R_L}{(-g_m R_L)^2} = \frac{kTR_{in}}{C_{int} R_L} \qquad (t \gg \tau_0) \tag{A4.2}$$

Next, we move to the other extreme of $t = T_{int} \ll \tau_0$, which gives us integrator operation:

$$\overline{V_{in}^2} = \frac{\frac{kTR_{in}}{C_{in}} g_m^2 R_L \left[\frac{2T_{int}}{\tau_0}\right]}{\left(-g_m R_L \left[\frac{T_{int}}{\tau_0}\right]\right)^2} = \frac{2kTR_{in}}{T_{int}} = 4kTR_{in}\frac{1}{2T_{int}} \tag{A4.3}$$

$$= 4kTR_{in} f_{NBW} \qquad (T_{int} = t \ll \tau_0)$$

Equation (A4.3) shows that in order to calculate the input referred noise of the integrator, we just have to multiply the input white noise PSD (power spectral density) with the equivalent input referred noise bandwidth f_{NBW}.

Figure A4.2 shows the calculated input referred noise, using (A4.1)-(A4.3) and some given parameters (shown in the figure). We compare the result to a transient noise simulation as well, and come to a good agreement with the calculation.

Notice that we analyzed here the circuit with a single noise source, and assumed that the rest of the circuit is noiseless. It should be mentioned that the transistor and output load produce noise as well. This leads to an initial noise sample on C_{int}, which is ignored here [38]. However, later we will see that the g_m-C filter has several gain stages in front of it and hence this additional noise is suppressed by that gain.

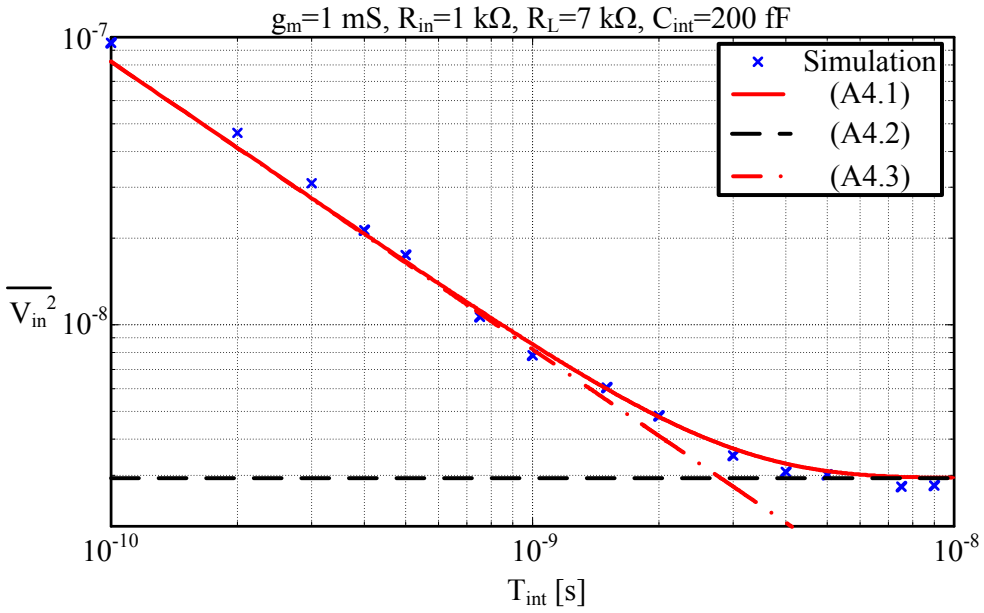

Figure A4.2: Input-referred noise using (A4.1), (A4.2), and (A4.3) of the gm-C integrator compared to transient noise simulation.

Some consequences arise from this analysis. First, the input-referred noise is independent of the integration capacitor if we operate the circuit in its integrator regime. This is somewhat counterintuitive, since we are used to kT/C-type noise contributions. Furthermore, we have to decrease the noise resistor R_{in} if we want to lower the input referred noise for a given integration fixed time. Ultimately, we will be stopped by (A4.2) since we cannot build an ideal integrator, which means that we are still bounded to a kT/C term for a non-ideal integrator.

A5 Current Steering DAC Noise Model and Analysis

Figure A5.1 illustrates the noise model of a fully differential current steering DAC, including the noise sources of the current source and the load resistors. We assume that the noise of the DAC current source is due to a transistor having a noise current of:

$$\frac{\overline{I_c^2}}{\Delta f} = \gamma_{DAC} 4kT g_{mDAC} \qquad \text{(A5.1)}$$

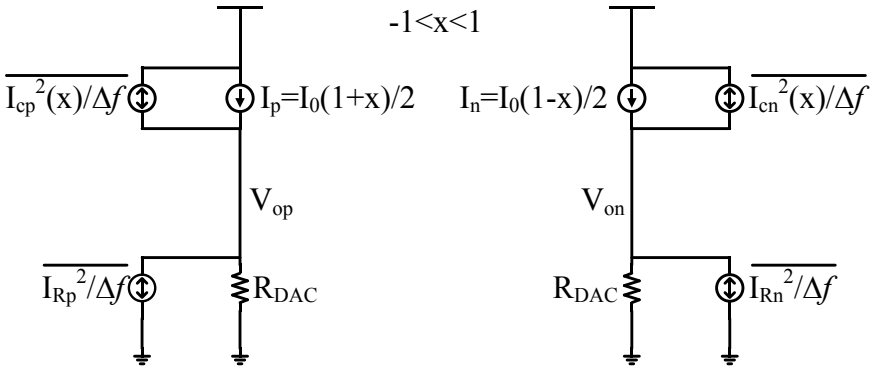

Figure A5.1: Noise model of the current steering DAC.

where γ_{DAC} is the thermal noise parameter of the current source transistor. Furthermore, we assume that the g_{mDAC} is signal dependent. When all the current is steered to one side, this side sees all the noise due to the current source, while the other side does not see any noise. We can describe this in the following way by using the g_m/I_D ratio:

$$\frac{\overline{I_{cp}^2}}{\Delta f}(x) = \gamma_{DAC} 4kT \frac{g_{mDAC}}{I_D} \frac{I_0}{2}(1+x) \tag{A5.2}$$

$$\frac{\overline{I_{cn}^2}}{\Delta f}(x) = \gamma_{DAC} 4kT \frac{g_{mDAC}}{I_D} \frac{I_0}{2}(1-x) \tag{A5.3}$$

The noise PSD of the load resistors is:

$$\frac{\overline{I_{Rn}^2}}{\Delta f} = \frac{4kT}{R_{DAC}} \quad \frac{\overline{I_{Rp}^2}}{\Delta f} = \frac{4kT}{R_{DAC}} \tag{A5.4}$$

The fully differential DAC output noise voltage $\left(\frac{\overline{V_{DAC}^2}}{\Delta f}\right)$ is the sum of (A5.2)-(A5.4) multiplied with the squared load resistor R_{DAC}:

$$\frac{\overline{V_{DAC}^2}}{\Delta f} = \left(\frac{4kT}{R_{DAC}} + \frac{4kT}{R_{DAC}} + \gamma_{DAC} 4kT \frac{g_{mDAC}}{I_d} \frac{I_0}{2}(1+x) + \gamma_{DAC} 4kT \frac{g_{mDAC}}{I_d} \frac{I_0}{2}(1-x) \right) R_{DAC}^2 \tag{A5.5}$$

$$\frac{\overline{V_{DAC}^2}}{\Delta f} = 8kTR_{DAC}\left(1 + \gamma_{DAC}\frac{g_{mDAC}}{I_d}\frac{I_0}{2}R_{DAC}\right)$$

Notice that $\frac{I_0}{2}R_{DAC}$ is equal to the DAC common mode voltage ($V_{DAC_{CM}}$), however we can also use the full-scale voltage by realizing that:

$$\frac{V_{FS}}{2} = I_0 R_{DAC} \tag{A5.6}$$

giving us:

$$\frac{\overline{V_{DAC}^2}}{\Delta f} = 8kTR_{DAC}\left(1 + \gamma_{DAC}\frac{g_{mDAC}}{I_D}\frac{V_{FS}}{4}\right) \tag{A5.7}$$

Interestingly, the differential noise is signal independent. It is convenient to write the DAC noise as shown in (A5.7), since we only have three design parameters: the full-scale input voltage, the g_m/I_D ratio of the current source and the DAC load resistor.

A6 ADC Noise Estimation

We will here calculate the noise based on the design parameters of the ADC. Notice that most of these parameters will be introduced in Chapter 4, however we will use them here to confirm the accuracy of the derived noise equations.

We start with the sampling noise using (3.31):

$$\overline{V_{SN_{RES}}^2} = 2\left(\overline{V_{SN_{MUX}}^2} + \overline{V_{SN_{SW}}^2}\right)G_C^2 = 2\frac{kT}{C_S}\left(\frac{R_{MUX}}{R_{SW}}G_{SF}^2 + nf\right)G_C^2 \tag{3.1}$$

We now use the following parameters, C_s = 3.5 pF, R_{MUX}/R_{SW} = 0.25, G_{SF} = 0.94, G_C = 0.92, nf = 1.3 to obtain:

$$\overline{V_{SN}^2} = 2\frac{kT}{C_S}\left(\frac{R_{MUX}}{R_{SW}}G_{SF}^2 + nf\right) = 3.02\ nV^2 \tag{A6.1}$$

Next, we go the conversion phase. Notice that we need the integration time to calculate the noise. The complete ADC cycle is 28.7 ns (35 MS/s), 5.7 ns is allocated for tracking, leaving us about 23 ns for the conversion. We have 16 SAR decision hence we have 23/16 ns = 1.4375 ns per conversion cycle. We use 550 ps for DAC settling and 150 ps for the comparator and logic giving us an integration time of about 740 ps. With this, we can look at the DAC noise referred to V_{res} (3.38):

$$\overline{V_{DAC_{res}}^2} = 8kTR_{DAC}\left(1+\gamma_{DAC}\frac{g_{m_{DAC}}}{I_d}\frac{V_{FS}}{4}\right)\frac{G_{SF}^2 G_C^2}{2T_{int}} \tag{A6.2}$$

where R_{DAC} = 100 Ω, $\gamma_{DAC} = 0.7$, V_{FS} = 1.8 V and $\frac{g_{m_{DAC}}}{I_D} = 4\frac{1}{V}$ resulting in:

$$\overline{V_{DAC_{res}}^2} = 3.75\ nV^2 \tag{A6.3}$$

Next, we calculate the noise of the comparator:

$$\overline{V_{COMP_{res}}^2} = 8kT\frac{nf_{comp}}{g_{m_{comp}}}\frac{1}{2T_{int}} \tag{A6.4}$$

where we use g_{mcomp} = 20 mS and nf_{comp} = 1.3 giving us:

$$\overline{V_{COMP_{res}}^2} = 1.44\ nV^2 \tag{A6.5}$$

The noise of the buffer during conversion is:

$$\overline{V_{SF_{res}}^2} = \frac{8kT}{g_{m_{SF}}} nf_{SF}\frac{1}{2T_{int}}G_C^2 \tag{A6.6}$$

where g_{mSF} = 30mS and n_{fSF} = 1.3

$$\overline{V_{SF_{res}}^2} = 0.8\ nV^2 \tag{A6.7}$$

The quantization noise is about $1nV^2$ ($LSB^2/12$) at 14 bits, which is referred to V_{res}:

$$\overline{e_{q_{res}}^2} = \frac{LSB^2}{12} G_{SF}^2 G_C^2 = 0.75\ nV^2 \tag{A6.8}$$

We can now use (3.39) to calculate the SNR of the ADC:

$$\begin{aligned} SNR_{ADC_{res}} &= \frac{\frac{1}{2}\left(\frac{V_{FS}}{2} G_{SF} G_C\right)^2}{\overline{e_q^2} G_B^2 G_C^2 + \overline{V_{SN_{res}}^2} + \overline{V_{CN_{res}}^2}} \\ &= \frac{0.303}{(0.75 + 0.8 + 1.44 + 3.75 + 3.02) nV^2} \\ &= 74.92\ dB \end{aligned} \tag{A6.9}$$

This is very close to the actual measured ADC SNR performance of 75 dB. Table A6.1 shows the noise summary.

Noise Source	Noise referred to V_{res} [nV^2]	%
Sampling	3.02	30.9
DAC	3.75	38.4
Comparator	1.44	14.8
Buffer	0.8	8.2
Quantization	0.75	7.7
Total	9.76	100

Table A6.1: Noise Summary

A7 Source Follower Settling Considerations

We will calculate here the buffer g_m based on the linear settling requirements, assuming a sampling capacitor of C_S = 3.5 pF, which was chosen based on kT/C requirements. Figure A7.1(a) shows the SF with a bottom plate sampling switch and (b) the small signal equivalent circuit. We assume here that the transistor has infinite output resistance. We can derive the input to output transfer function:

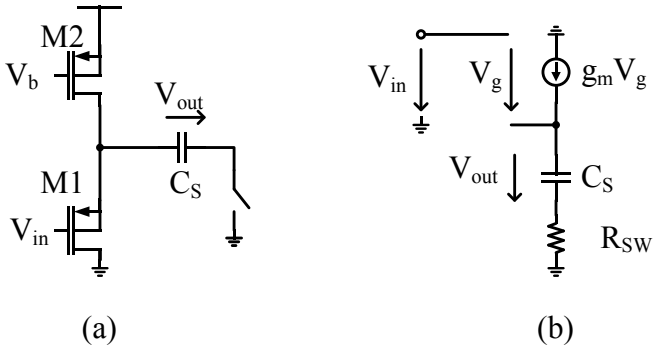

Figure A7.1: (a) Source follower with bottom plate sampling switch. (b) Small signal model

$$\frac{V_{out}}{V_{in}} = \frac{1}{1 + sC_s\left(R_{SW} + \frac{1}{g_m}\right)} \tag{A7.2}$$

where R_{SW} is the bottom plate sampling switch resistance. Furthermore we make a simplification and assuming that $R_{SW}=1/g_m$ giving us:

$$\tau_{SF} = \frac{C_S 2}{g_m} \tag{A7.2}$$

Next, we have a settling time of 5.7 ns, we will assume that half of that is used for slewing giving us 2.85 ns for linear settling. We need approximately 10 time constants for 14-bit settling, however we use here 12 instead. This over-design accounts for post layout parasitic capacitances. Furthermore, we make the bias of the SF programmable. Now we can calculate g_m:

$$g_m = \frac{C_s 2}{\tau_{SF}} = \frac{2 \cdot 3.5\ pF}{\frac{2.85\ ns}{12}} = 29.5\ mS\ \approx 30\ mS \tag{A7.3}$$

The buffer bandwidth is about 670 MHz ($f_p = \frac{1}{\tau_{SF} 2\pi}$).

A8 Buffer HD_2 Improvement due to Higher Intrinsic Gain

We will calculate here the nonlinear input to output voltage relation for a source follower at DC, showing that the intrinsic gain improves the HD_2 of the buffer. A nonlinear transconductance model is used for this analysis (see Figure A8.1). Using this model, we can write the input to output relation:

$$V_{out} = R\sum_{n=1}^{\infty} g_{m_n} V_g^n = R\sum_{n=1}^{\infty} g_{m_n}(V_{in} - V_{out})^n \tag{A8.1}$$

We use series reversion on (A8.1) and ignore all g_{m_n} for n > 2 to simplify the analysis, giving us:

$$V_g = \sum_{n=1}^{\infty} H_n V_{out}^n = V_{in} - V_{out} \tag{A8.2}$$

$$H_1 = \frac{1}{Rg_{m_1}}, H_2 = -\frac{Rg_{m_2}}{\left(Rg_{m_1}\right)^3}, H_3 = \frac{2\left(Rg_{m_2}\right)^2}{\left(Rg_{m_1}\right)^5}, H_4 = \frac{-5\left(Rg_{m_2}\right)^3}{\left(Rg_{m_1}\right)^7}, \ldots$$

We can solve (A8.2) for V_{in}, this only results in a modification of H_1:

$$V_g = \sum_{n=1}^{\infty} H_n^* V_{out}^n = V_{in} \tag{A8.3}$$

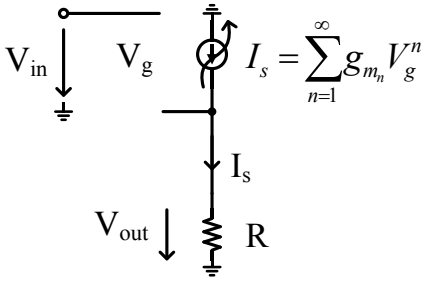

Figure A8.1: Nonlinear DC model of a source follower

$$H_1^* = 1 + \frac{1}{Rg_{m_1}}, H_2^* = -\frac{Rg_{m_2}}{\left(Rg_{m_1}\right)^3}, H_3^* = \frac{2\left(Rg_{m_2}\right)^2}{\left(Rg_{m_1}\right)^5}, H_4^* = \frac{-5\left(Rg_{m_2}\right)^3}{\left(Rg_{m_1}\right)^7}, \ldots$$

Another series reversion on (A8.3) is performed to arrive at the final input to output relation:

$$V_{out} = \sum_{n=1}^{\infty} Y_n V_{in}^n \tag{A8.3}$$

$$Y_1 = \frac{1}{1 + 1/Rg_{m_1}}, Y_2 = \frac{Rg_{m_2}}{\left(1 + 1/Rg_{m_1}\right)^3 \left(Rg_{m_1}\right)^3}, \ldots$$

We stop the series in (A8.4) after n=2 and assume that Rg_{m_1} is large compared to 1 simplifying (A8.4) to:

$$V_{out} = V_{in} + \frac{Rg_{m_2}}{R^3 g_{m_1}^3} V_{in}^2 \tag{A8.5}$$

We see that that the second-order term is divided by the third power of the intrinsic gain of the transistor, hence increasing that gain makes the source follower more linear.

A9 Analysis of the Phase Error of the 3rd Order Nonlinearity

We investigate here the influence of the frequency-dependent phase effect on the ADC nonlinearity cancelation. It should be noted that a nonlinear system with memory can change the amplitude and the phase of higher order distortion products. In order to describe such a system perfectly, one would have to derive a Volterra series. This can be done for our circuit, but is not necessary. We rely on the fact that the frequency dependent nonlinearity effects are relatively weak. We start by using the model of Chapter 3 to gain some insight:

$$y_1 = A(V_{in}) = a_1 V_{in} + a_3 V_{in}^3 \tag{3.4}$$

Now we apply a test tone $V_{in} = \hat{v}_{in} \cos(\omega_{in} t)$ and after some simplifications we arrive at:

$$y_1 = \left(a_1 \hat{v}_{in} + a_3 \frac{3}{4} \hat{v}_{in}^3\right) \cos(\omega_{in} t) + a_3 \frac{1}{4} \hat{v}_{in}^3 \cos(3\omega_{in} t) \tag{A9.1}$$

Now we can introduce a phase term to the third-order term that models the frequency dependent nonlinearity:

$$\begin{aligned} y_{1\varphi_{3rd}} = &\left(a_1 \hat{v}_{in} + a_3 \frac{3}{4} \hat{v}_{in}^3\right) \cos(\omega_{in} t) \\ &+ a_3 \frac{1}{4} \hat{v}_{in}^3 \cos(3\omega_{in} t + \varphi_{3rd}) \end{aligned} \tag{A9.2}$$

Furthermore, we will define an error term $err(t, \varphi_{3rd}) = y_{1\varphi_{3rd}} - y_1$ giving us:

$$\begin{aligned} err(t, \varphi_{3rd}) &= a_3 \frac{1}{4} \hat{v}_{in}^3 [\cos(3\omega_{in} t + \varphi_{3rd}) - \cos(3\omega_{in} t)] \\ &= -a_3 \frac{1}{2} \hat{v}_{in}^3 \sin\left(\frac{\varphi_{3rd}}{2}\right) \sin\left(3\omega_{in} t + \frac{\varphi_{3rd}}{2}\right) \end{aligned} \tag{A9.3}$$

where we used the trigonometric identity:

$$\cos(u) - \cos(v) = -2 \sin\left(\frac{u - v}{2}\right) \sin\left(\frac{u + v}{2}\right) \tag{A9.4}$$

Now we can write the following:

$$y_{1\varphi_{3rd}} = y_1 + err(t, \varphi_{3rd}) \tag{A9.5}$$

This result is very useful, since we have isolated the third-order error term from the original equation. Notice that this is exactly the form we used to model the comparator offset (Section 3.2.2, Figure 3.5) and again, we can use the result from (3.13), but this time replacing V_{off} with the calculated error term from (A9.5):

$$V_{DAC} \approx V_{in} - \frac{3\left(a_3 err(t, \varphi_{3rd})\right)}{a_1^2} V_{in}^2 + \frac{err(t, \varphi_{3rd})}{a_1} \tag{A9.6}$$

From here, we can calculate the HD_3 of the ADC based on the phase error of the third harmonic. We approximate this by only considering the last part of the sum of (A9.6):

$$HD_3(\varphi_{3rd}) = \frac{a_3 \hat{v}_{in}^2}{2a_1} \sin\left(\frac{\varphi_{3rd}}{2}\right) \tag{A9.7}$$

A10 DAC Current Source Biasing

This appendix describes the bias circuit that was used to bias the current steering DAC. We already mentioned that we used a triple cascode approach, which was chosen to provide a sufficiently high output resistance. Notice that only the DAC DC characteristic defines the linearity of the DAC, since we assume that the DAC completely settles and that only the settled value matters. Furthermore, source degeneration or regulated cascoding (also known as gain boosting) was not used, because of the more complicated routing during layout.

The bias circuit topology was mainly defined by noise considerations. The noise of the bias circuit can actually reach the DAC output when it is approaching the full scale voltage. In [39] the noise of an open-loop differential pair was analyzed and showed that for an unbalanced case the bias noise is moved from CM to the differential output. Notice that a current steering DAC is similar to a differential pair, and it can be seen as a "quantized" version of a differential pair. We will assume a worst-case scenario by investigating the DAC bias circuit for a full-scale output.

Figure A10.1 shows parts of the employed bias circuit generation. Only the bias for the current sources is shown, the cascode bias circuits will follow later. We start with a bandgap voltage V_{BG} that is forced via an amplifier on a resistor R_B, hence the bias current is $I_B = V_{BG}/R_B$. The total output current I_O (with FS output) is $I_O = nI_B$ and the output voltage becomes:

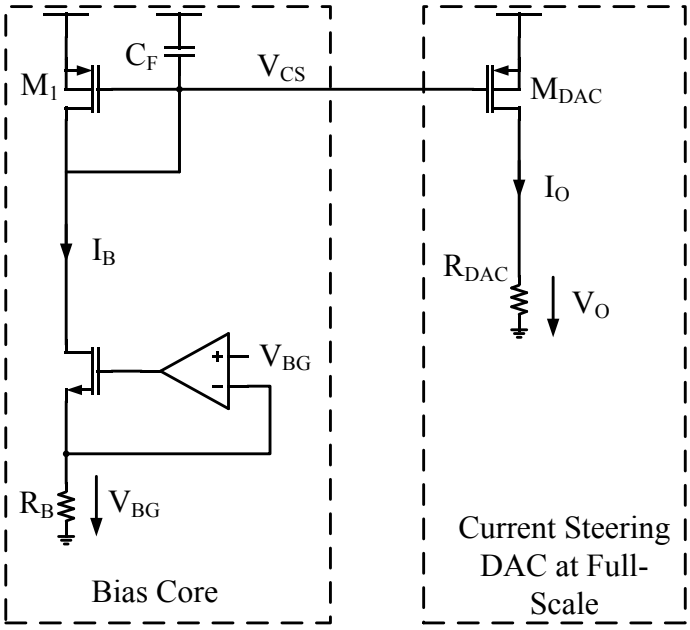

Figure A10.1: Simplified bias core with current steering DAC at full scale

$$V_O = I_O R_{DAC} = n\frac{R_{DAC}}{R_B}V_{BG} \tag{A10.1}$$

where n is the ratio between the bias current and the DAC FS current. Notice that R_B and R_{DAC} are made of the same unit elements and have the same temperature behavior; this makes the FS output voltage and DAC CM voltage first-order temperature independent.

Next, we calculate the noise due to the bias circuit. We have two major noise sources, the diode connected transistor M_1 and the resistor R_B. The noise due to the bandgap voltage and amplifier are ignored, since they have been heavily filtered already. We will calculate the DAC output noise in two steps, first we calculate the total integrated noise at the V_{CS} node and then we refer it to the output node V_O. We can do this because we limit the noise bandwidth at the V_{CS} node. The total integrate noise at V_{CS} is:

$$\overline{V_{CS}^2} = \left(4kTg_m + \frac{4kT}{R_B}\right)\frac{1}{g_m^2} f_{NBW} = \left(\frac{4kT}{g_m} + \frac{4kT}{g_m^2 R_B}\right)\frac{1}{4C_F \frac{1}{g_m}} = \frac{kT}{C_F}\left(1 + \frac{1}{g_m R_B}\right) \quad \text{(A10.2)}$$

We can simplify (A10.2) by realizing that $R_B = V_{BG}/I_B$, leading to:

$$\overline{V_{CS}^2} = \frac{kT}{C_F}\left(1 + \frac{1}{\frac{g_m}{I_B} V_{BG}}\right) \quad \text{(A10.3)}$$

Now we can refer that noise to the output of the current steering DAC. Notice that for the full scale code, the DAC looks like a CS stage looking from the V_{CS} node to the output and hence the gain from V_{CS} to V_O is $g_{mDAC}R_L$. This allows us to write the following relation (we use again $R_L=V_{FS}/I_O$):

$$\overline{V_{O_{bias}}^2} = \frac{kT}{C_F}\left(1 + \frac{1}{\frac{g_m}{I_B} V_{BG}}\right)\left(g_{m_{DAC}} R_L\right)^2 = \frac{kT}{C_F}\left(\left(\frac{g_{m_{DAC}}}{I_O}\right)^2 + \frac{g_{m_{DAC}}}{I_O}\frac{1}{V_{BG}}\right) \quad \text{(A10.4)}$$

It is correct to say that $g_{mDAC}/I_O = g_m/I_B$, since both transistors (M_1 and M_{DAC}) form a current mirror and hence that ratio is in fact the same. Again, we do not have top many options for lowering the noise. The g_m/I_D ratio is already set to its lowest value (4 1/V), we use C_F = 50pF giving us about $1nV^2$ at the DAC output from the bias circuit. That is about 10% of the total noise budget, but only at FS. Furthermore, the current ratio between I_O and I_B is about 20.

Figure A10.2 shows the complete bias circuit. Notice that we use resistors to generate the appropriate V_{DS} voltages. Again, these resistors are all made of the same unit elements like R_B and hence these voltages are to first order temperature independent as well as the DAC FS voltage. Furthermore, the core bias generator is only used to provide the DAC current source bias. A second mirror is used to derive additional bias currents that are used to generate the cascode bias voltages.

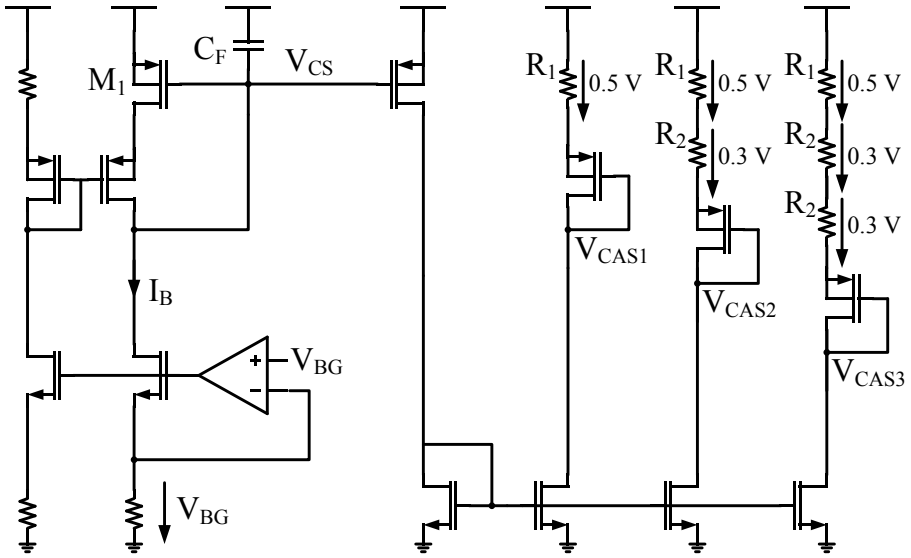

Figure A10.2: Full bias circuit

This decouples the current source bias from the remaining bias circuitry, giving us more isolation.

Finally, this bias current generator is used to create all the important bias currents of the chip, in particular for the SF and the comparator. We used resistively loaded differential pairs in the pre-amplifier for the comparator, and these resistors are of the same kind as R_B, giving us a well-defined CM.

A11 Self-Heating Effect

Polysilicon resistors are often used for their good linearity performance and low temperature coefficient. However, this is only true for low current densities. For high currents, self-heating can corrupt the desired performance. We show here a first-order hand calculation.

Poly resistors are covered by SiO_2 (Figure A11.1), which is a very poor thermal conductor (100 times worse than Si, for example). However, resistors produce heat due to the dissipated power (P=IV) and if this heat cannot to be transferred to the environment, the resistor will heat up. Unfortunately, resistors are temperature dependent, which usually is modeled with the following equation:

$$R(T) = R_{nom}(1 + T_{c1}(T - T_{nom}) + T_{c2}(T - T_{nom})^2) \\ = R_{nom}(1 + T_{c1}\Delta T + T_{c2}\Delta T^2) \quad \text{(A11.1)}$$

where T_{c1} and T_{c2} are fitting parameters to model first- and second-order behavior and R_{nom} is the nominal resistor at a reference temperature T_{nom}. In our design, the current of the DAC is converted into a voltage using a poly resistor, hence the consumed power is signal dependent ($P=RI_{sig}^2$) and due to the self-heating the resistor is changing its temperature resulting in a resistor value change (A11.1). We will first calculate the temperature rise using a first order thermal calculation. Assuming that the electrical power is completely converted to a heat flow (Q), we can write:

$$Q = P_{elec} = RI_{sig}^2 \quad \text{(A11.2)}$$

The temperature rise due that electrical power can be calculated by multiplying Q with the thermal resistance:

$$\Delta T = QR_{th} = RI_{sig}^2 = VI_{sig} \quad \text{(A11.3)}$$

R_{th} is a material property and follows following equation:

$$R_{th} = \frac{L}{\kappa A} \quad \text{(A11.4)}$$

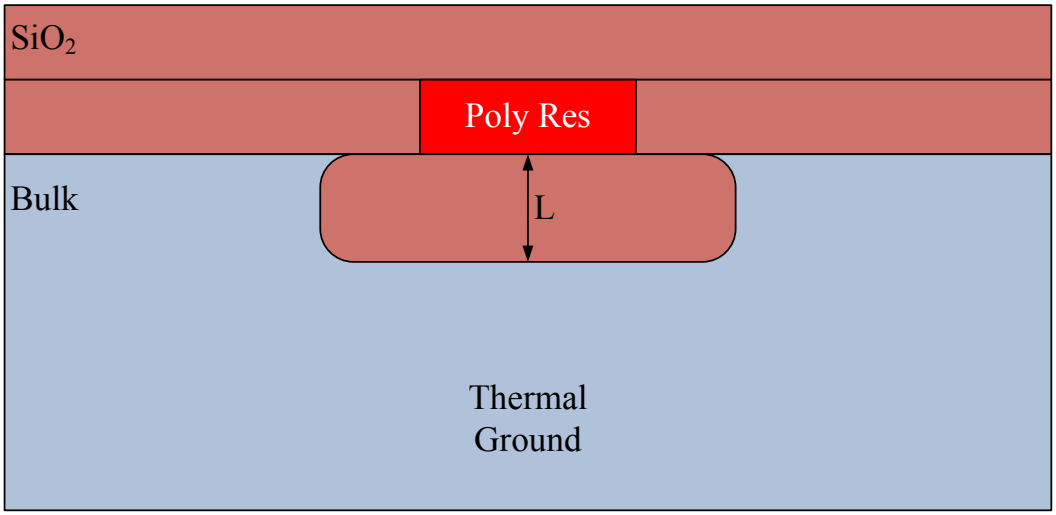

Figure A11.1: Cross section of a poly resistor for thermal modeling

where L is the distance in the heat flow direction, A is the area perpendicular to the heat flow, and κ is the thermal conductivity ($\kappa = 1.4\frac{W}{mK}$ for SiO_2). A is the drawn area of the resistor and L is the distance to the bulk (see Figure A11.1), which is filled with SiO_2. Since the bulk has a 100 times better thermal conductivity, we assume the bulk to be a thermal ground. L is a process parameter and has to be provided by the foundry, while *A* is a design parameter.

We can now calculate HD_3 as shown below. In this analysis, we dropped T_{c2} to simplify the algebra. For a given current, the voltage is:

$$V = R_{nom}(1 + T_{c1} \cdot \Delta T)I_{sig} \tag{A11.5}$$

from here we can calculate Q:

$$Q = VI_{sig} = R_{nom}(1 + T_{c1} \cdot \Delta T)I_{sig}^2 \tag{A11.6}$$

using (A11.3) and (A11.6) and solving for ΔT then yields:

$$\Delta T = \frac{R_{th}R_{nom}I_{sig}^2}{1 - T_{c1}R_{th}R_{nom}I_{sig}^2} \approx R_{th}R_{nom}I_{sig}^2 \tag{A11.7}$$

We assumed T_{c1} to be small, which is true for most poly resistors, to simplify (A11.7). Inserting (A11.7) into (A11.5) then gives:

$$V = R_{nom}I_{sig} + T_{c1}R_{th}R_{nom}^2I_{sig}^3 \tag{A11.8}$$

Finally, we can calculate HD_3:

$$HD_3 = \frac{\frac{1}{4}T_{c1}R_{th}R_{nom}^2}{R_{nom}}I_{sig}^2 = \frac{1}{4}T_{c1}R_{th}R_{nom}I_{sig}^2 \tag{A11.9}$$

Bibliography

[1] R. Kapusta, J. Shen, S. Decker, H. Li, and E. Ibaragi, "A 14b 80MS/s SAR ADC with 73.6dB SNDR in 65nm CMOS," in *ISSCC Dig. Tech. Papers*, Feb. 2013, pp. 472–473.

[2] M. Inerfield, A. Kamath, F. Su, J. Hu, Y. Xinyu, V. Fong, O. Alnaggar, F. Lin, and T. Kwan, "An 11.5-ENOB 100-MS/s 8mW dual-reference SAR ADC in 28nm CMOS," in *Symp. VLSI Circuits Dig.,* June 2014, pp. 1–2.

[3] B. Murmann, "ADC Performance Survey 1997-2015." [Online]. Available: http://web.stanford.edu/~murmann/adcsurvey.html.

[4] K. Doris, E. Janssen, C. Nani, A. Zanikopoulos, and G. van der Weide, "A 480mW 2.6GS/s 10b 65nm CMOS time-interleaved ADC with 48.5dB SNDR up to Nyquist," in *ISSCC Dig. Tech. Papers*, Feb. 2011, pp. 180–182.

[5] V. Giannini, P. Nuzzo, V. Chironi, A. Baschirotto, G. Van der Plas, and J. Craninckx, "An 820 μW 9b 40MS/s Noise-Tolerant Dynamic-SAR ADC in 90nm Digital CMOS," in *ISSCC Dig. Tech. Papers*, Feb. 2008, pp. 238–610.

[6] M. Kramer, E. Janssen, K. Doris, and B. Murmann, "14b 35MS/S SAR ADC achieving 75dB SNDR and 99dB SFDR with loop-embedded input buffer in 40nm CMOS," in *ISSCC Dig. Tech. Papers*, Feb. 2015, pp. 284–285.

[7] W. Kester, *The Data Conversion Handbook*, Newnes, 2005, [Online]. Available: http://www.analog.com/library/analogDialogue/archives/39-06/data_conversion_handbook.html.

[8] M. J. M. Pelgrom, *Analog-to-Digital Conversion*. Springer, New York, 2nd ed. 2013.

[9] B. Murmann, "On the use of redundancy in successive approximation A/D converters," in *International Conference on Sampling Theory and Applications (SampTA)*, Bremen, Germany, July 2013.

[10] Z. Boyacigiller, B. Weir, and P. Bradshaw, "An error-correcting 14b/20 μs CMOS A/D converter," in *ISSCC Dig. Tech. Papers*, Feb. 1981, pp. 62–63.

[11] F. Kuttner, "A 1.2V 10b 20MSample/s non-binary successive approximation ADC in 0.13/spl mu/m CMOS," in *ISSCC Dig. Tech. Papers*, Feb. 2002, pp. 176–177

[12] T. Ogawa, T. Matsuura, H. Kobayashi, N. Takai, M. Hotta, H. San, A. Abe, K. Yagi, and T. Mori, "Non-binary SAR ADC with digital error correction for low power applications," in *Asia Pacific Conference on Circuits and Systems (APCCAS)*, Dec. 2010, pp. 196–199.

[13] A. H. T. Chang, "Low-power high-performance SAR ADC with redundancy and digital background calibration," PhD Thesis, Massachusetts Institute of Technology, 2013.

[14] Linear Technology, "DC1826A - LTC238918-Bit/16-Bit, 2.5Msps Low Noise, SAR ADCs with Pin-Configurable Analog Input Range," 2004. [Online]. Available: http://www.linear.com/demo/DC1826a.

[15] R. Reeder and R. Ramachandran, "Wideband A/D Converter Front-End Design Considerations. When to Use a Double Transformer Configuration," *Analog Dialogue*, vol. 40, no. 7, Jul. 2006.

[16] D. G. Haigh and B. Singh, “A switching scheme for SC filters which reduces the effect of parasitic capacitances associated with switch control terminals,” in Proc. *IEEE International Symposium on Circuits and Systems*, May 1983, vol. 41, pp. 586–589.

[17] S.-W. M. Chen and R. W. Brodersen, “A 6-bit 600-MS/s 5.3-mW Asynchronous ADC in 0.13- CMOS,” *IEEE Journal of Solid-State Circuits*, vol. 41, no. 12, pp. 2669–2680, Dec. 2006.

[18] K. Doris, E. Janssen, C. Nani, A. Zanikopoulos, and G. van der Weide, “A 480 mW 2.6 GS/s 10b Time-Interleaved ADC With 48.5 dB SNDR up to Nyquist in 65 nm CMOS,” *IEEE Journal of Solid-State Circuits*, vol. 46, no. 12, pp. 2821–2833, Dec. 2011.

[19] A. M. Abo and P. R. Gray, “A 1.5-V, 10-bit, 14.3-MS/s CMOS pipeline analog-to-digital converter,” *IEEE Journal of Solid-State Circuits*, vol. 34, no. 5, pp. 599–606, May 1999.

[20] V. Tripathi and B. Murmann, “An 8-bit 450-MS/s single-bit/cycle SAR ADC in 65-nm CMOS,” in *Proc. ESSCIRC*, Sept. 2013, pp. 117–120.

[21] V. Tripathi, “Design of high-speed, high-resolution SAR A/D converters in nano-scale CMOS processes,” PhD Thesis, Stanford University, 2014.

[22] L. R. Carley and T. Mukherjee, “High-speed low-power integrating CMOS sample-and-hold amplifier architecture,” in *Proc. Custom Integrated Circuits Conference,* May 1995, pp. 543–546.

[23] T. Sepke, P. Holloway, C. G. Sodini, and H.-S. Lee, “Noise Analysis for Comparator-Based Circuits,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 56, no. 3, pp. 541–553, March 2009.

[24] Mini Circuits, “ADT1-6T (Datasheet).” [Online]. Available: http://194.75.38.69/pdfs/ADT1-6T.pdf.

[25] K. Doris, *Wide-Bandwidth High-Dynamic Range D/A Converters*. Springer, Dordrecht, 1st ed., 2006.

[26] C. C. McAndrew and T. Bettinger, "Robust Parameter Extraction for the R3 Nonlinear Resistor Model for Diffused and Poly Resistors," *IEEE Transactions on Semiconductor Manufacturing*, vol. 25, no. 4, pp. 555–563, Nov. 2012.

[27] R. C. McLachlan, A. Gillespie, M. C. W. Coln, D. Chisholm, and D. T. Lee, "A 20b Clockless DAC With Sub-ppm INL, 7.5 nV/ sqrt(Hz) Noise and 0.05 ppm/C Stability," *IEEE Journal of Solid-State Circuits*, vol. 48, no. 12, pp. 3028–3037, Dec. 2013.

[28] C. Portmann, "Characterization and Reduction of Metastability Errors in CMOS Interface Circuits," PhD Thesis, Stanford University, 1995.

[29] B. Razavi and B. A. Wooley, "Design techniques for high-speed, high-resolution comparators," *IEEE Journal of Solid-State Circuits*, vol. 27, no. 12, pp. 1916–1926, Dec. 1992.

[30] P. Shettigar and S. Pavan, "Design Techniques for Wideband Single-Bit Continuous-Time Modulators With FIR Feedback DACs," *IEEE Journal of Solid-State Circuits*, vol. 47, no. 12, pp. 2865–2879, Dec. 2012.

[31] I. E. Opris, "Noise estimation in strobed comparators," *Electronics Letters*, vol. 33, no. 15, pp. 1273–1274, July 1997.

[32] P. Nuzzo, F. De Bernardinis, P. Terreni, and G. Van der Plas, "Noise Analysis of Regenerative Comparators for Reconfigurable ADC Architectures," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 55, no. 6, pp. 1441–1454, July 2008.

[33] W. Liu, P. Huang, and Y. Chiu, "A 12b 22.5/45MS/s 3.0mW 0.059mm2 CMOS SAR ADC achieving over 90dB SFDR," in *ISSCC Dig. Tech. Papers*, Feb. 2010, pp. 380–381.

[34] M. Hesener, T. Eichler, A. Hanneberg, D. Herbison, F. Kuttner, and H. Wenske, "A 14b 40MS/s Redundant SAR ADC with 480MHz Clock in 0.13pm CMOS," in *ISSCC Dig. Tech. Papers*, Feb. 2007, pp. 248–249.

[35] P. Harpe, G. Dolmans, K. Philips, and H. de Groot, "A 0.7V 7-to-10bit 0-to-2MS/s flexible SAR ADC for ultra low-power wireless sensor nodes," in *Proc. ESSCIRC*, Sept. 2012, pp. 373-376.

[36] P. Harpe, E. Cantatore, A. van Roermund, "A 10b/12b 40 kS/s SAR ADC With Data-Driven Noise Reduction Achieving up to 10.1b ENOB at 2.2 fJ/Conversion-Step," *IEEE Journal of Solid-State Circuits*, vol.48, no.12, pp. 3011-3018, Dec. 2013.

[37] I. Daubechies, R. DeVore, C. S. Gunturk, and V. A. Vaishampayan, "Beta expansions: a new approach to digitally corrected A/D conversion," in *IEEE International Symposium on Circuits and Systems,* May 2002, pp.784–787

[38] E. Iroaga, "Pipelined analog-to-digital converters using incomplete settling," PhD Thesis, Stanford University, 2007.

[39] C. Daigle, "Switched-capacitor DACs using open loop output drivers and digital predistortion," PhD Thesis, Stanford University, 2010.