DIGITAL COMPENSATION OF DYNAMIC ACQUISITION ERRORS AT THE FRONT-END OF ADCS

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

Parastoo Nikaeen

September 2008

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) Principal 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.

_____

(Robert W. Dutton)

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 A. Wooley)

Approved for the University Committee on Graduate Studies

Abstract

In Analog-to-Digital Converter (ADC) applications such as wireless base stations, sub-sampling at an Intermediate Frequency (IF) is an attractive method for minimizing component count and system cost. By applying this method, one or more steps of down-conversion are removed from the receiver path and some of the analog front-end signal processing functions can be moved to the digital domain. In such a solution, the ADC's linearity at high input frequencies becomes a critical issue. Despite the use of a dedicated track-and-hold amplifier (THA), nonlinearities in the circuit's input network often introduce dynamic errors that limit the performance of the ADC at high input frequencies.

A number of analog techniques have been proposed in the past to improve the linearity performance of the ADC's front-end. However, most of these techniques suffer from bandwidth limitations in the active analog circuitry and lose their performance at high input frequencies. In recent years, the continuous scaling of CMOS technology has resulted in low cost and high performance digital circuits. This has provided the feasibility to integrate complicated digital signal processing on chip and therefore use digital correction methods to compensate circuit nonlinearities in ADCs. However, these techniques mainly address static errors in the converter core and are not effective at removing dynamic nonlinearities at the front-end of the ADC.

This dissertation introduces a digital enhancement scheme that is specifically tailored to remove high frequency distortion caused by the dynamic nonlinearities at the sampling front-end of ADCs. The basic concept of digital compensation here is to

apply the inverse nonlinear function to the digital output of the ADC in order to minimize its error over the desired frequency range. Conceptually, a nonlinear system with memory can be modeled with a Volterra series. However, the inverse Volterra series becomes very complex as the order of nonlinearity and memory in the system increases and it requires intensive computational power that is impractical even in today's fine-line technology. Our proposed algorithm uses information about the sources of nonlinearity and judicious modeling to simplify the digital post processing scheme.

The main sources of dynamic error at the front-end of ADCs are investigated and analyzed in order to find a compact model for frequency-dependent nonlinearities in this stage. This model is then used to build a nonlinear filter in the digital domain for compensating the nonlinearities. The coefficients of the filter are calibrated using training signals in the desired frequency region. The correction scheme can be effectively used for canceling nonlinearities across the whole input bandwidth of the ADC.

Measurement results from applying the proposed method to a 14-bit commercially available ADC are presented at different frequency ranges. The experimental results show an improvement in SFDR of the ADC to more than 83 dB up to input frequencies of 470 MHz. Measurement results show that the algorithm is not very sensitive to temperature variations and is also applicable to any sub-sampling ADC that suffers from linearity degradation at high input frequencies.

Acknowledgments

I wish to acknowledge the many people who made my life at Stanford an enriching and enjoyable experience.

First, I would like to thank Professor Boris Murmann for supervising my doctoral research and for his generous support and guidance during my graduate career at Stanford. His great technical vision and continuous encouragement helped me a lot in conducting my research. Every single meeting I had with him motivated me to try new ideas and go one step forward in my research.

I am very grateful to Professor Robert Dutton for accepting me into his research group in my first year at Stanford, and for being very understanding and supportive through the problems I had during the past few years at Stanford. I appreciate his patience and guidance during the time I was looking for an interesting Ph.D. project. He has always been a great mentor for me and I enjoyed all the conversations I had with him.

I would like to extend my gratitude to Professor Bruce Wooley for serving as a member of my orals and reading committee and for his help and advice throughout my graduate studies at Stanford. I would also like to thank Professor Piero Pianetta who served as the chair of my orals committee.

The financial support of Center for Integrated Systems, National Semiconductor and Texas Instruments is greatly appreciated. I am grateful to David Boisvert and Bumha Lee at NSC, and Martin Izzard and Marco Corsi at TI for providing the test chips for our measurements.

I would like to thank current and previous members of Professor Murmann's and Professor Dutton's group for all the useful discussions I had with them and for their help and support in various aspects of my project. In particular, I would like to thank

Dr. Echere Iroaga, Dr. Yangjin Oh, Dr. JungHoon Chun, Jason Hu, Clay Daigle, Justin Kim, Pedram Lajevardi, Wei Xiong, Noam Dolev, Manar El-Chammas, Alireza Dastgheib, Dr. Yang Liu, Dr. Hai Lan, Dr. Yi-Chang Lu, JaeWook Kim and Dr. Reza Navid. I am especially grateful to Ann Guerra and Felly Barrera for their friendly help in administrative matters.

This work would not have been possible without the many technical discussions and great advice from several of my friends and Colleagues at Stanford. I especially like to thank Dr. Mehdi Mohseni who was my DSP expert friend and helped me a lot in understanding several signal processing related problems I had during my Ph.D. work. I would also like to thank Dr. Sam Kavusi, Dr. Hossein Kakavand and Valentin Abramzon for many useful discussions, and Dr. Mona Jarrahi and Bita Nezamfar not only for the technical discussions but also for being available to listen to my complaints whenever not everything was going smoothly.

I am grateful to the help and support of my many friends in PSA (Persian Student Association) during my Stanford years. They have all been like a big family for me and made the life much easier and more enjoyable while being at Stanford and away from my own family. In particular, I would like to thank Mona Jarrahi, Neda Beheshti, Negin Nejati, Mehdi Mohseni, Shahram Taghavi, Nafiseh Rahimzadeh, Sina Sohangir, Azin Sahabi, Amin Saberi, Sara Rabiey, Mohammad Shoeybi, Bita Nezamfar, Amir Amirkhany, Leili Baghaei, Pooya Sarabandi, Ava Rahbar, Hedi Razavi, Golnaz Alipour, Shirin Jalali, Vahideh Hosseinikhah, Narges Baniasadi, Nazanin Davani, Pendar Ardalan, Shadi Oveisgharan, Vahbod Pourahmad and Abtin Keshavarzian for their kindness and friendship. I especially like to express my sincere appreciation to my friends in the PSA music group, Mohammad Hekmat, Naiem Semsarilar and AmirAli Kia for all the great memories I share with them during our weekly practices in the past two years.

I would also like to thank my dear friends from undergraduate years, Sara Bavarian and Samira Vosoughi for their invaluable friendship and continuous love and encouragements in the past few years. My undergraduate studies at Sharif University of Technology helped me to build the required background in engineering and inspired

me to pursue my graduate studies abroad. I sincerely thank Professor Mehrdad Sharif Bakhtiar and Professor Masood Jahanbeglo for their encouragements and remarkable teachings.

I am very grateful to my mother and my only brother, Pedram, for all their continuing love, sacrifice and support in the past few years, especially after the tragic loss of my father five years ago. There is no word that can express my gratitude toward them; I just wish there was a way to pay back a little of what they have done for me.

Finally, I would like to express my deepest gratitude to the man who was not only my father but also one of my best friends and teachers in life. His unconditional love and belief in me was my main encouragement in all the achievements I had in life. If it was not because of his love and support, I would have never been anywhere close to where I am today. This dissertation, for what it is worth, is dedicated to the memory of my father.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Analog-to-digital converters (ADCs) are essential components in the receiver path of wireless communication systems. Figure 1.1(a) shows the block diagram of a conventional receiver architecture used in wireless base stations. Due to the limitations in sampling rate and bandwidth of the ADCs, the received RF signal can not be digitized at the front-end of the system. The input RF signal is down-converted in frequency using several stages of mixers and bandpass filters and is then digitized at baseband in order to do the required post processing.

With the recent progress in the design of analog-to-digital converters, intermediate frequency (IF) sub-sampling has become an attractive method in these receiver architectures. By doing IF sampling, we can take away one or more steps of down conversion and move some of the analog functions to the digital domain (see Figure 1.1(b)). This method reduces the number of components and therefore the size and cost of the analog part of the system. Also, some functions, such as I and Q demodulation, can be done more accurately in the digital domain.

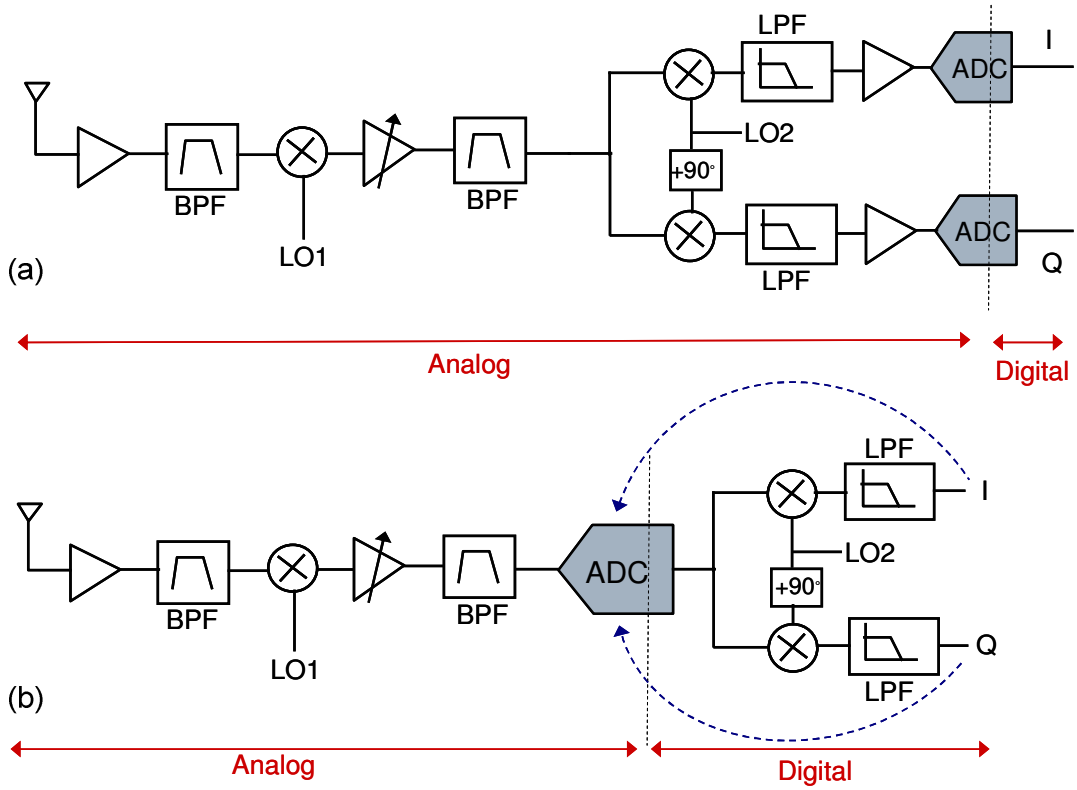

Figure 1.1: Block diagram of the receiver architecture in wireless base stations. (a) Conventional receiver using ADCs at baseband. (b) Using IF subsampling in the receiver architecture.

In order to take full advantage of this method, we need to have ADCs with higher acquisition bandwidth and larger dynamic range. Therefore, it is critical to have very high linearity in ADCs at high input frequencies. Despite the use of a dedicated track-and-hold amplifier (THA) at the front-end of ADC, nonlinearities in the circuit's input network often introduce dynamic errors that limit the performance of the ADC at high input frequencies. Therefore, it is essential to find methods to cancel these errors and improve ADC's linearity at IF input frequencies.

Previous efforts at achieving high frequency linearity have mainly focused on analog approaches. In [1] it was shown that integrated BJT input buffers can reduce the distortion across a wide bandwidth. This improvement, however, can not be realized in CMOS technology. In [2], a modified bootstrapping circuit was used to

cancel the backgate effect of the converter's input switch to improve linearity. This technique shows good performance up to moderate frequencies, but ultimately suffers from bandwidth limitations in the active bootstrap circuitry.

In recent years, the continuous scaling of CMOS technology has resulted in low cost and high performance digital circuits. This has provided the feasibility to integrate complicated digital signal processing on chip and therefore use digital correction methods to compensate circuit nonlinearities in ADCs, see e.g., [3]-[5]. However, these techniques mainly address static errors in the converter core and are not effective at removing dynamic nonlinearities at the front-end of the ADC.

In this thesis, we present a digital enhancement scheme that is specifically tailored to remove high frequency distortion caused by the dynamic nonlinearities at the sampling front-end of ADCs. The basic concept of digital compensation here is to apply the inverse nonlinear function to the digital output of the ADC in order to minimize its error over the desired frequency range (see Figure 1.2). Conceptually, a nonlinear system with memory can be modeled with a Volterra series [6], [7]. There are several methods in the literature that can be used to find the inverse of a Volterra series and extract its coefficients [8]-[10]. However, the inverse Volterra filter typically becomes very complex as the order of nonlinearity and memory in the system increase, and it therefore requires intensive computational power that is usually impractical even in today's fine-line technology.

The approach studied in this dissertation uses a simplified model for the dynamic errors generated at the acquisition stage of the ADC and applies an appropriate inverse nonlinear function to the digital output. We show in our analysis that the proposed correction scheme works with much less complexity than the inverse of the Volterra series. The algorithm is also investigated in the presence of other circuit non-idealities at the front-end of ADC and shows good agreement with the general distortion model of the front-end. In order to evaluate the proposed approach, we used an existing commercially available ADC [11] and applied the algorithm to its digital outputs. We show through our measurement results that we have been able to achieve very high

linearity performance (SFDR>83 dB) in the ADC at high input frequencies (f_{in}=470 MHz).

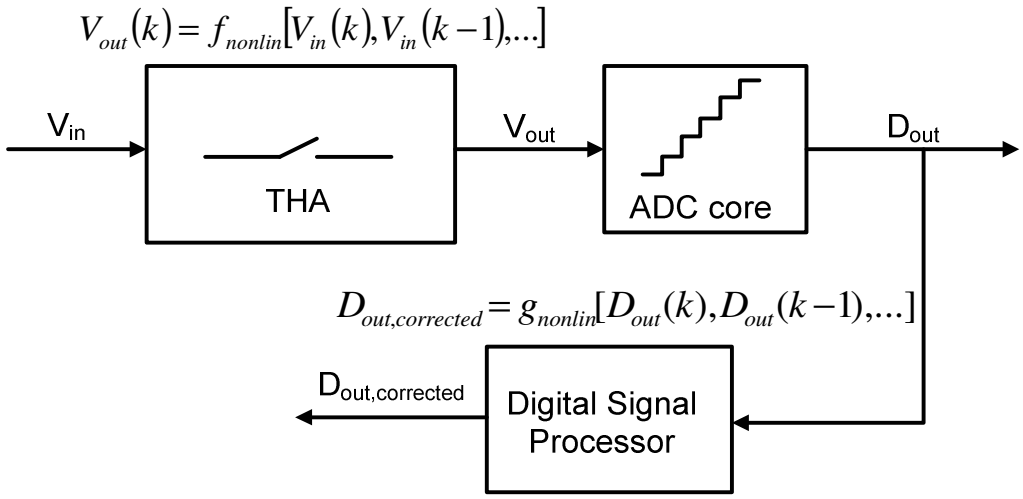

Figure 1.2: Applying the inverse nonlinear function to the ADC output for compensation of dynamic nonlinearities at its front-end.

1.2 Organization

This thesis is organized into six chapters of which this is the first. Chapter 2 introduces the sources of nonlinearity at the front-end of ADCs and shows how judicious modeling of these imperfections helps us simplify the correction process. The analysis in this chapter shows where the memory and nonlinearities at the acquisition stage of the ADC are coming from and how they affect the linearity performance of the system. A compact nonlinear model is developed and it is expanded to take nonidealities of the ADC's input driving circuit into account.

Our proposed digital correction algorithm is presented in Chapter 3. Several approaches for calibrating the filter coefficients and also simplifying the correction process are discussed and results of applying them to the nonlinear model of a track-and-hold circuit are presented. The simulation results show that the algorithm is capable of significantly increasing the linearity of the ADC's front-end. It can be used

to correct nonlinearities in any bandlimited signal and remains effective in presence of noise on the output signal.

Chapter 4 includes the experimental results from applying the algorithm to real ADC measurement data. It is shown that the digital correction scheme can greatly improve the linearity of the tested ADC up to very high input frequencies (470 MHz). The robustness and general applicability of the algorithm is analyzed and its performance on different kinds of input driving circuits is demonstrated.

Applying digital correction methods requires integrating the digital blocks on the same die as the analog components. This can cause switching noise in the substrate that can degrade the performance of the system. Chapter 5 presents an investigation on the impact of substrate noise on high speed ADCs. A flash ADC is used as a test block in this experiment and it is shown how the noise can couple to the comparators and generate wrong codes at their outputs. Measurement results are presented from testing the ADC with digital noise emulators on the chip.

Finally, Chapter 6 summarizes the contributions of this research and suggests areas for future investigations.

Chapter 2

Nonlinearity at the ADC's Front-End

Dynamic errors generated at the front-end of ADCs are a limiting factor in their linearity performance at high input frequencies. In this chapter we investigate sources of errors at the front-end of ADC and show how they can affect the performance of the converter. Analyzing the sources of errors helps us find a compact model for dynamic nonlinearities that can be used in the next chapter for developing the correction algorithm.

2.1 Analog-to-Digital Converters

ADCs are the link between the analog world of real signals and the digital world of signal processing. They are used for converting a continuously varying analog signal from instruments, sensors, etc. to discrete digital numbers [12], [13]. This process consists of sampling and quantization, as shown in the block diagram of Figure 2.1. Sampling converts the continuous signal into discrete-time sample points and the quantization process maps the continuous range of signal values into a finite set of discrete levels.

When analog-to-digital converters are used as part of a larger system, it is important to know the limitations of the converters and how they affect the performance of the entire system. Therefore, it is important to have metrics to

characterize the performance of the converter [12], [13]. Several static and dynamic metrics have been defined for this purpose, and depending on the particular application, some of these metrics become more or less relevant.

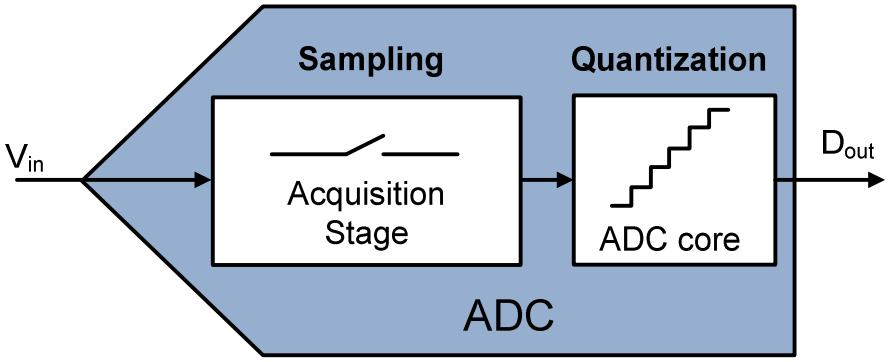

Figure 2.1: Block diagram of an analog-to-digital converter, including sampling and quantization.

One of the main metrics defining the performance of an ADC is its linearity, which is an important factor in high-resolution ADCs used in the receiver architectures discussed in Chapter 1. Any nonlinearity in the ADC path of these receiver architectures distorts the desired signal and spreads it in frequency as shown in Figure 2.2. This distortion corrupts the information in the received signal and affects the accuracy of the system. It is therefore important to achieve sufficient linearity in the ADC not to exceed the error tolerance of the system.

In order to characterize the linearity of the ADC, several metrics are defined such as: Integral Nonlinearity (INL), Differential Nonlinearity (DNL) and Spurious Free Dynamic Range (SFDR). INL and DNL are measured from the transfer function of the ADC and define its static linearity (or the ADCs linearity at low input frequencies). DNL is defined as the difference between the actual step width and the ideal step size of 1 LSB in the transfer function of the ADC. INL is defined as the deviation of the actual transfer function from the straight line passed through the end points of the transfer function. Figure 2.3 illustrates the definition for these terms on the static

transfer function of an ADC [13]. SFDR, on the other hand, is a performance metric used for characterizing the linearity of the ADC, using a dynamic measurement, over its entire input bandwidth. SFDR is measured by applying a sine-wave as the input to the converter and is defined as the ratio of the RMS value of the amplitude of the main signal component to the RMS value of the next largest noise or harmonic distortion component at the output (see Figure 2.4).

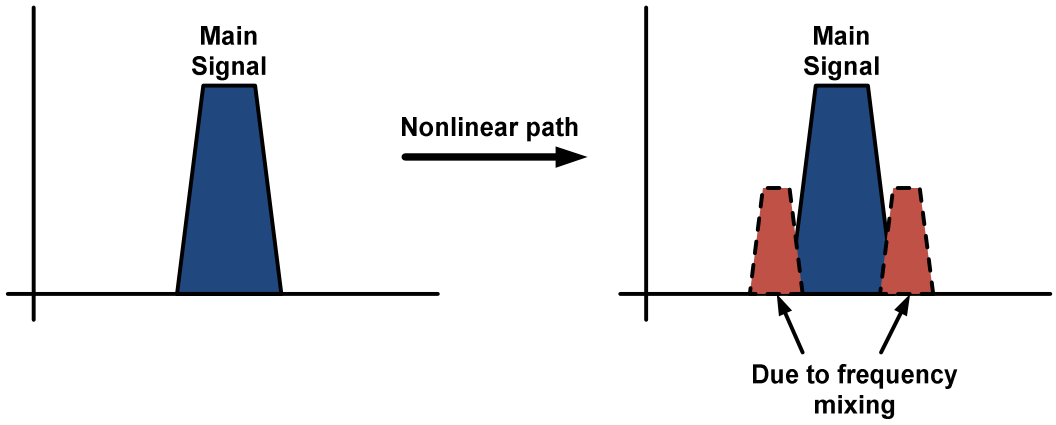

Figure 2.2: Impact of nonlinearity in distorting the frequency spectrum of the received signal.

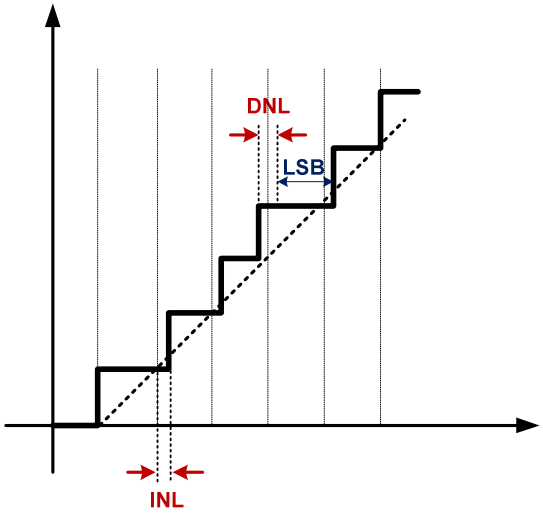

Figure 2.3: Definition of INL and DNL using the ADC's transfer function.

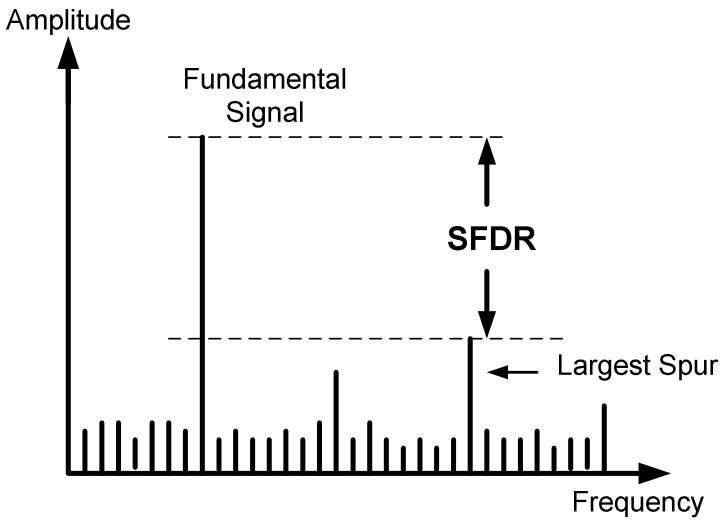

Figure 2.4: SFDR definition using the output frequency spectrum of an ADC.

2.2 Overview of ADC linearity limitations

ADCs used in communication applications usually need to sample signals at high frequencies. As explained in the previous section, it is also very important in these applications to add the least amount of distortion to the received signal during the digitization process. Therefore, the SFDR of these ADCs over their input bandwidth is an important factor defining the performance of the system.

The linearity of ADCs usually drops significantly with increasing input frequency. This can be observed as the degradation in their SFDR performance over the input frequency bandwidth as shown in Figure 2.5. This figure shows an SFDR plot of a state-of-the art 14-bit ADC [11] over its input bandwidth. The drop in linearity performance is due to the frequency dependent nonlinear errors generated in the transfer function of the ADC. Frequency dependent nonlinearities are caused by nonlinear functions with memory.

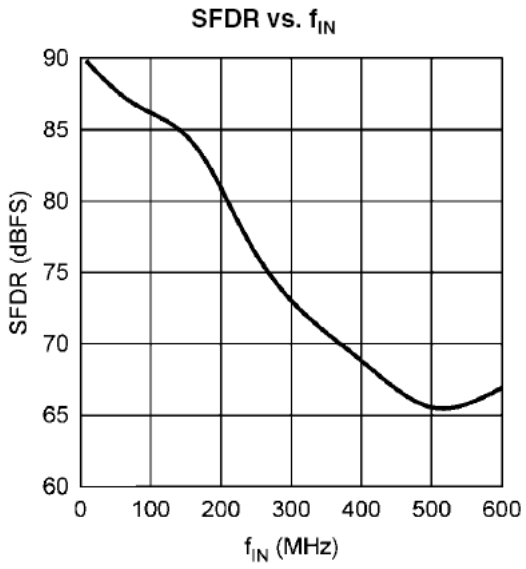

Figure 2.5: SFDR performance vs. input frequency, from National ADC14155 [11].

The nonlinear error generated in the ADC core (see Figure 2.1) is mainly the result of non-ideal quantization and is a static nonlinearity. This error is reflected in the INL/DNL performance of the ADC and also in its SFDR performance at low input frequencies [14]. These static nonlinear errors have been investigated extensively in the literature and several methods exist for minimizing and compensating them [15]-[18].

The linearity at high input frequencies is mainly limited by dynamic errors generated at the ADC's front-end, including the input driving circuit and the acquisition stage (see Figure 2.6). The acquisition stage is usually the most significant source of dynamic nonlinear error at the front-end of ADC. This stage includes a track-and-hold circuit consisting of a sampling switch and a capacitor, as shown in Figure 2.7, and is in charge of sampling the continuous-time input signal and converting it to discrete-time sample points.

The input driving circuit can be another source of dynamic errors at the ADC's front-end when used together with the acquisition stage. The driver can include a transformer for converting single-ended to differential signals (see Figure 2.8(a)), or it

can include buffers or differential amplifiers depending on the application (see Figure 2.8(b)) [19], [20]. Transformers are used to effectively drive the ADC input with minimal noise and distortion degradation of the wanted signal. These are passive elements that do not add power dissipation to the system and are usually the best option at high input frequencies. However, in some applications, buffers or differential amplifiers should be used in the ADC's driver circuit in order increase the dynamic range of the system or to keep the input impedance constant during the tracking and hold modes .

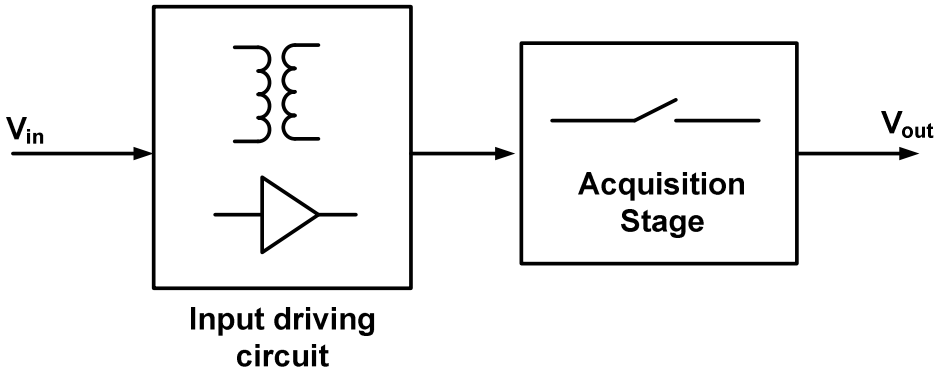

Figure 2.6: Block diagram of the ADC's front-end including input driving circuit and the acquisition stage.

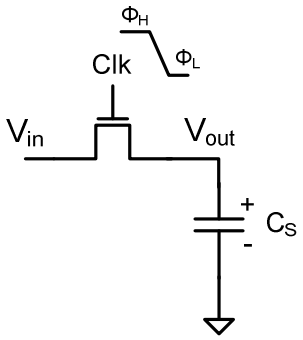

Figure 2.7: A simple track-and-hold stage at the acquisition stage of the ADC.

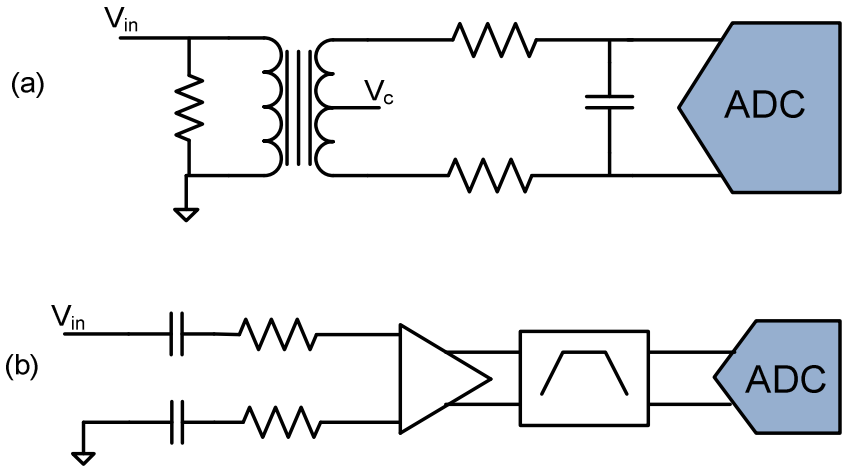

Figure 2.8: Block diagram of the ADC together with the input driving circuit. (a) Input driving circuit using a transformer. (b) Input driving circuit using a buffer or differential amplifiers.

The above-described circuits at the front-end of the ADC add nonlinear errors to the system which are dynamic, meaning that they depend on the input frequency, and static digital compensation methods can not be used to cancel them. Therefore, these stages are usually a limiting factor in the high-frequency linearity performance of ADCs. In order to compensate the errors at these stages it is essential to find a compact model for the nonlinearity that can be used for efficient post-processing in the digital domain. In the following section, we will first discuss the track-and-hold circuit, which is the main source of dynamic errors in ADCs, and derive a compact model for its dominant source of frequency dependent nonlinearity. We will then discuss nonlinearities caused by second order effects in the input driving circuit in the following section and generalize the model to take those errors into account.

2.3 Nonlinearity at the acquisition stage

The track-and-hold stage used for sampling at the front-end is a significant source of nonlinearity in the ADC's transfer function. The two main sources of errors at this stage are input-dependent charge injection and tracking nonlinearity due to the impedance modulation of the sampling switches. These two sources of nonlinearity cause the real output voltage to differ from the ideal V_{out} with some error during both tracking and the hold mode, as shown in Figure 2.9 [22]. These effects will be further explained in the following sections.

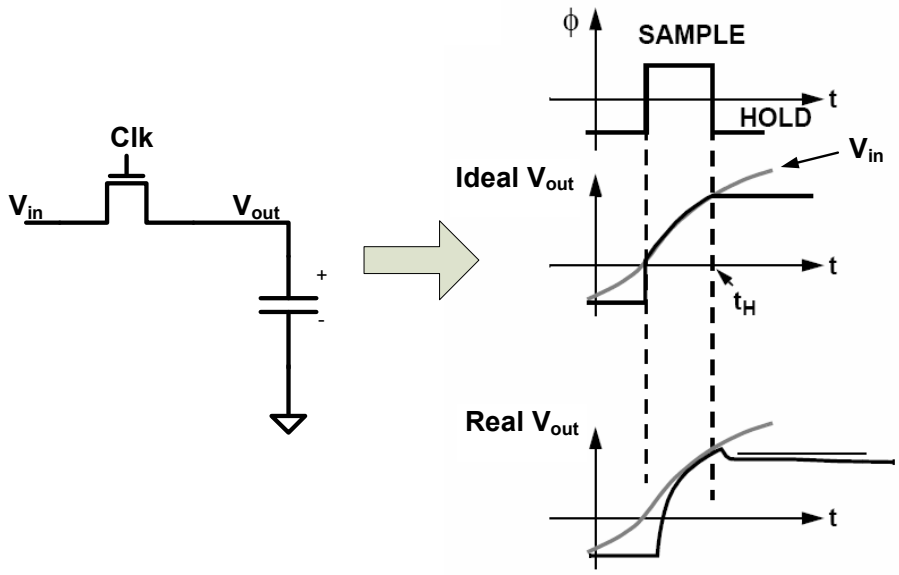

Figure 2.9: Ideal and real output of a track-and-hold stage during tracking and hold phases.

2.3.1 Input dependent charge injection

In the simple track-and-hold circuit shown in Figure 2.10, the sampling switch is on during tracking mode and V_{out} is tracking the input voltage. When the switch turns off to transition to the hold mode, the output voltage should ideally sample the final

value of the input during tracking mode and hold it constant as shown in Figure 2.9. However, due to the charge injected to the sampling capacitor during the transition, the real output voltage adds some error to the sampled value of the input voltage (see Figure 2.9). This charge injection comes from the MOS switch and is related to the charge on the overlap capacitance in the switch and also the charge stored in the transistor channel.

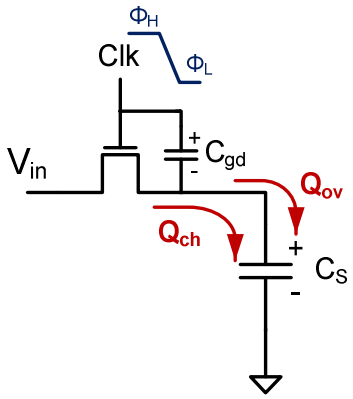

Figure 2.10: Charge injection in a track-and-hold circuit.

During transition from track mode to hold mode, the clock signal changes from Φ_h to Φ_L (see Figure 2.10). This voltage drop changes the charge on the overlap capacitor and injects Q_{ov} defined with the following equation to the sampling capacitor

$$Q_{ov} = \frac{C_{ov}C_s}{C_{ov}+C_s}(\Phi_H - \Phi_L) \qquad (2.1)$$

This error mechanism is called the clock feedthrough and affects the sampled value by injecting the above charge to the sampling capacitor. However, this charge is independent of the input voltage and hence only causes a constant offset on the sampled voltage which can be cancelled in a differential architecture.

The other source of charge injection during transition from tracking to hold mode is the charge in the transistor channel. Using square law device models, this charge can approximately be modeled as

$$Q_{ch} = C_{ch}[\Phi_H - (V_{gs} - V_{th})] \tag{2.2}$$

When the switch turns off, a fraction of the transistor channel charge goes toward the source and drain terminals depending on the impedance seen at these nodes and also the clock fall time. This charge strongly depends on the input and can cause significant distortion to the sampled value.

Several studies have been done in the past years to analyze what defines the fraction of charge going to each terminal and how it can be calculated [23]-[28]. For a slow clock fall time, the channel conductance follows the gate voltage and the circuit can be modeled as shown in Figure 2.11 [24]. The charge injection in this block can be analyzed by writing the differential equation and finding its solution through numerical methods. If the clock fall time is faster than the carrier transit time in the channel, the two ends of the channel pinch off at the beginning of the clock fall time and the terminal voltages do not have any effect on the charge distribution. This causes the charge to split equally between the source and drain terminal of the device. Figure 2.12 shows the ratio of the charge going to source and drain of the transistor versus their clock fall time and the ratio of the capacitor sizes at the two terminals. This figure was obtained by simulating the circuit of Figure 2.11 in HSpice (using a TSMC 130-nm process) for different clock fall time and capacitor sizes (similar to the plot in [24]). As can be observed from this figure, most of the channel charge flows to the terminal with larger capacitance for slow clock fall times; the charge splits equally between the two terminals at very fast clock fall times.

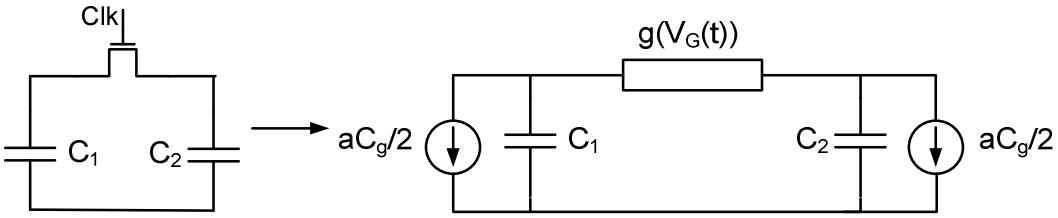

Figure 2.11: Simplified model for charge injection [24].

In a track-and-hold circuit, the fraction of transistor channel charge that is injected onto the sampling capacitor strongly depends on the input voltage and causes significant distortion on the sampled value. Several methods have been used in the past to minimize this input dependent charge injection on the sampling capacitor. These methods include using CMOS switches to achieve partial cancellation or use a dummy switch for providing the opposite charge [29]. However, none of these methods are able to cancel input dependent charge injection completely and do not provide sufficient accuracy for high-resolution ADC applications.

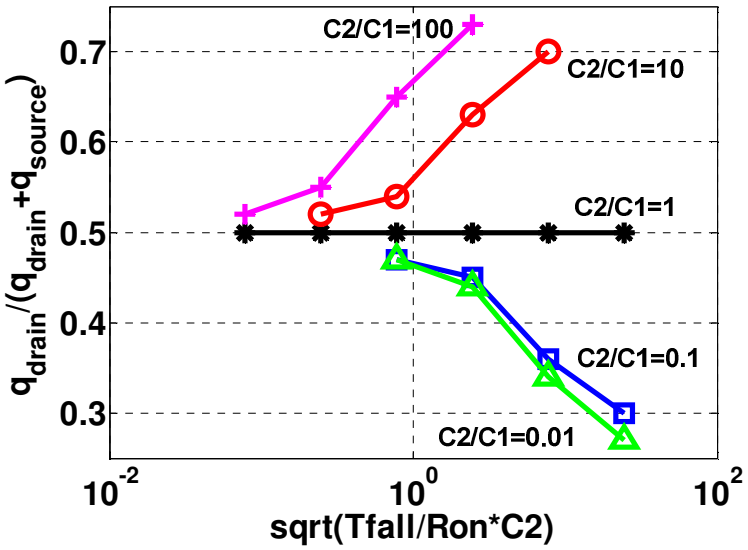

Figure 2.12: Ratio of charge injection into source and drain of transistor vs. clock fall time.

One common method that is usually used in the acquisition stage of high-performance ADCs is to use a bottom-plate track-and-hold for minimizing the input dependent charge injection [30], [31]. Figure 2.13 shows the structure of a bottom-plate track-and-hold circuit. The way this circuit works is that the bottom switch (M_2) is turned off first to freeze the charge at the bottom plate of the sampling capacitor. Therefore, when M_1 turns off node V_{bot} is a floating node and its charge can not change with the charge injection from M_1. The charge at V_{bot} is then redistributed using a feedback circuit, as the sampled value of the input voltage.

In this architecture we only have charge injection from M_2 affecting the sampled value. The charge stored in M_2 channel is defined by (2.1) and is independent of the input voltage. However, as we discussed above, the fraction of transistor charge going to the capacitor depends on the impedance seen at the drain of M_2 and therefore depends on the resistance of M_1. Due to the dependency of M_1 resistance to the input voltage (as will be discussed in detail in next section), there is still some small nonlinear input dependency in the charge injection from the bottom switch to the capacitor that can distort the sampled value.

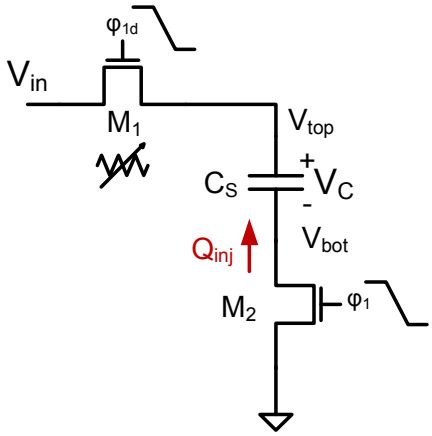

Figure 2.13: A bottom-plate track-and-hold circuit.

The switch resistance of M_1 can be kept nearly independent of the input voltage by using the bootstrap circuit discussed in the next section [32]. A bootstrap structure

keeps V_{gs} of transistor constant during the tracking mode and therefore makes its on resistance constant. However there are still some small input dependencies in the switch resistance due to the backgate effect in the transistor and also parasitic capacitances in the nonideal bootstrap circuitry. As a result, the charge injection to the sampling capacitor can still cause small, but significant distortion on the sampled value.

We did several device and mixed-mode simulations, using TCAD simulation tools [33], [34], of the circuit shown in Figure 2.13 (with a bootstrapped M_1 switch) to find the impact of different circuit parameters on the nonlinearity caused by charge injection. Device models were used for NMOS transistors in the circuit together with ideal capacitor and voltage sources. Clock was applied between source and gate terminal of M_1 to model an ideal bootstrapped switch. A channel length of 0.13μm was used for transistors and their doping profiles were optimized to achieve nominal performance for a 0.13μm process. The advantage of simulating the circuit using a TCAD device model for transistors is that all the charge flow and transistor channel variations can be observed and also the simulation results does not rely on the accuracy of the circuit models used for the devices. Appendix A includes an example netlist for modeling the transistor and the netlist of the circuit in Figure 2.13 implemented in the Taurus simulation tool.

In a bottom-plate track-and-hold, the input voltage is sampled as the charge on the bottom plate of the capacitor. Therefore, linearity of this charge defines the linearity performance of this block. In order to measure the nonlinearity caused by charge injection, a DC input voltage was used and its value was swept from 0 to 1 V. For each input voltage, the charge injected to the capacitor during M_2 turn-off was measured and was plotted versus the input voltage. A straight line is fitted to this plot and the linearity performance is defined as the ratio of the maximum deviation from the straight line to the total charge on the capacitor at full-scale (see Figure 2.14). This charge linearity is defined in the following equation

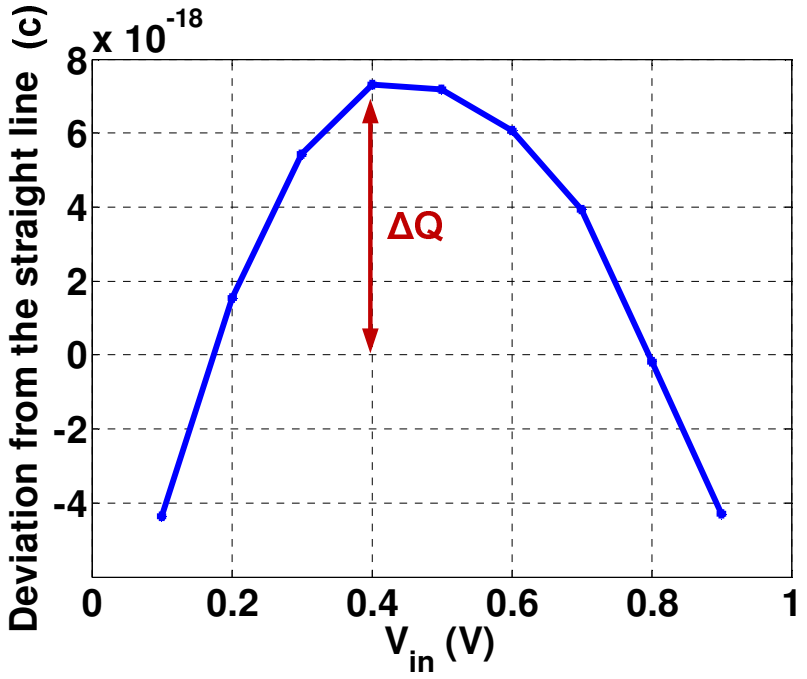

Figure 2.14: Deviation from a straight line in the plot of charge injection vs. input voltage. Total charge is defined as the charge on the capacitor at full scale, $Q_{total}=2pF\times1V=2e^{-12}$ C.

$$\text{Charge Linearity} = -20\times\log\left(\frac{Q_{error}}{Q_{total}}\right) \tag{2.3}$$

where Q_{error} is the deviation of the charge injection plot from the straight line and Q_{total} is the total charge on the capacitor (see Figure 2.14).

Figure 2.15 includes several plots showing the linearity performance of (2.3), defined for the bottom-plate track-and-hold circuit, versus different circuit parameters. Figure 2.15(a) shows the charge linearity performance vs. size of the bottom transistor (W_2). It can be observed from this figure that nonlinearity due to charge injection is reduced by reducing size of the bottom transistor. This is due to the fact that a smaller switch provides less charge injection. Figure 2.15(b) shows that linearity is improving with increasing size of the top switch, M_1. Using a larger switch size reduces the nonlinear section of the impedance seen at the drain of M_2 and hence reduces the nonlinear part of the charge injection. Figure 2.15(c) also shows that linearity

improves with smaller sampling capacitor size. This is again because the nonlinear switch resistance becomes a smaller portion of the total impedance at node V_{bot}. It is also observed from Figure 2.15(d) that nonlinearity due to charge injection is reduced by using slower clock fall times. This is explained by the fact that most of transistor channel charge is redirected to the source (ground connection) when having a slower clock fall time and that reduces the nonlinear charge injection to the capacitor. These results all show how different parameters in the circuits need to be chosen for minimizing the nonlinearity in the charge injection; though other factors such as the input load, thermal noise and the bandwidth requirement in the front-end need to be considered at the same time for choosing different circuit parameters in this stage.

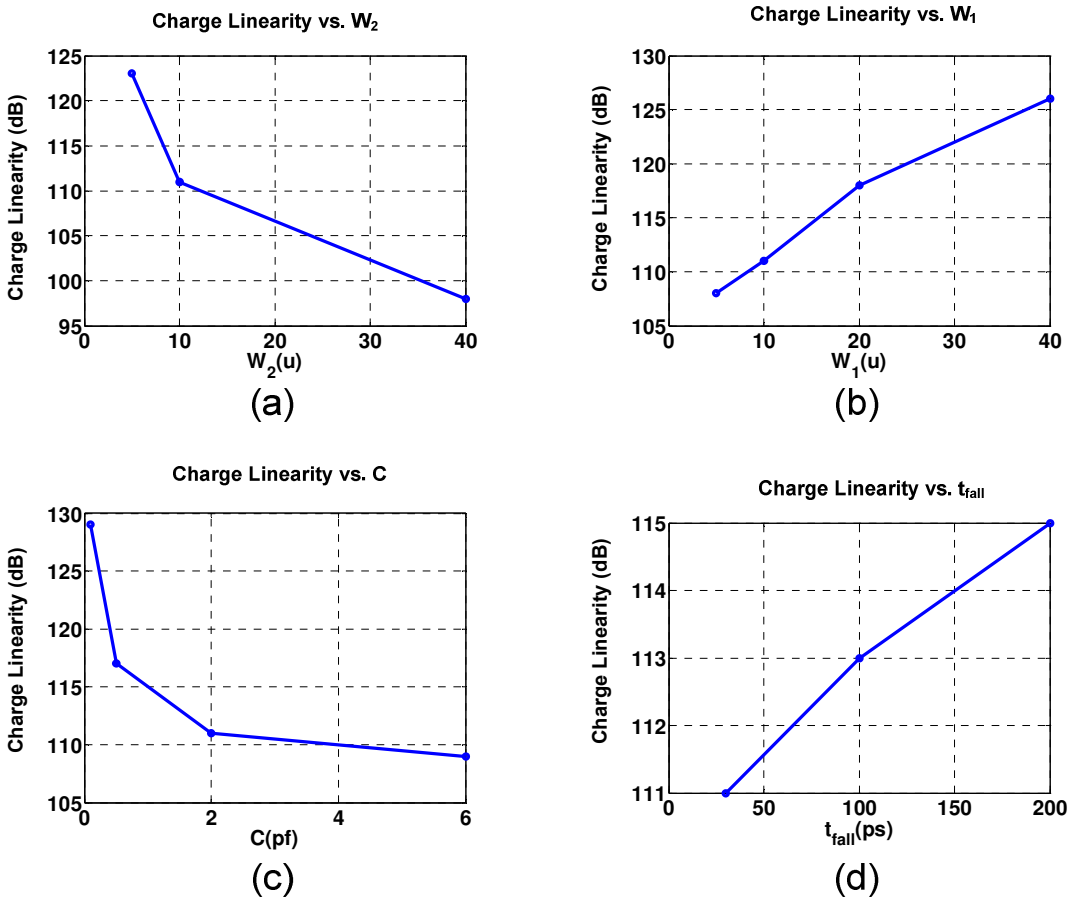

Figure 2.15: Charge injection nonlinearity in a bottom-plate track-and-hold vs. different circuit parameters (equation (2.3)). (a) Linearity vs. size of M_2 (W_2). (b) Linearity vs. size of M_1 (W_1). (c) Linearity vs. size of capacitor. (d) Linearity vs. Clock fall time.

2.3.2 **Nonlinearity in the tracking phase**

During tracking mode, the track-and-hold circuit of Figure 2.13 can be modeled as an RC circuit. Ideally the capacitor voltage V_C should track the input voltage; however the finite switch resistance causes this voltage to differ from the ideal V_{out} as shown in Figure 2.9. The relation between V_{in} and V_c during tracking mode can be modeled with the following equation

$$V_C = V_{in} - (R_1 + R_2)C\frac{dV_C}{dt} \tag{2.4}$$

where R_1 and R_2 are switch resistances of M_1 and M_2, respectively. R_2 is constant but R_1 is a nonlinear function of the input voltage defined with the following equation

$$R_1 = R_1(V_{in}) = \frac{1}{\mu_n C_{ox}\frac{W}{L}[V_{DD} - V_{in} - V_{th}]} \tag{2.5}$$

In the above equation, R_1 is a strong nonlinear function of the input voltage and causes significant distortion in the sampled voltage on the capacitor. Therefore, it is important to modify the circuit and make this resistance independent of the input voltage. One common method for achieving a constant switch resistance is to use a bootstrap switch structure [32] as shown in Figure 2.16. In this circuit, C_{boot} is charged first to V_{DD} and is then connected across the gate and source terminal of the transistor. This makes the V_{gs} of the transistor constant during the tracking mode and makes the impedance of the switch nearly independent of the input voltage.

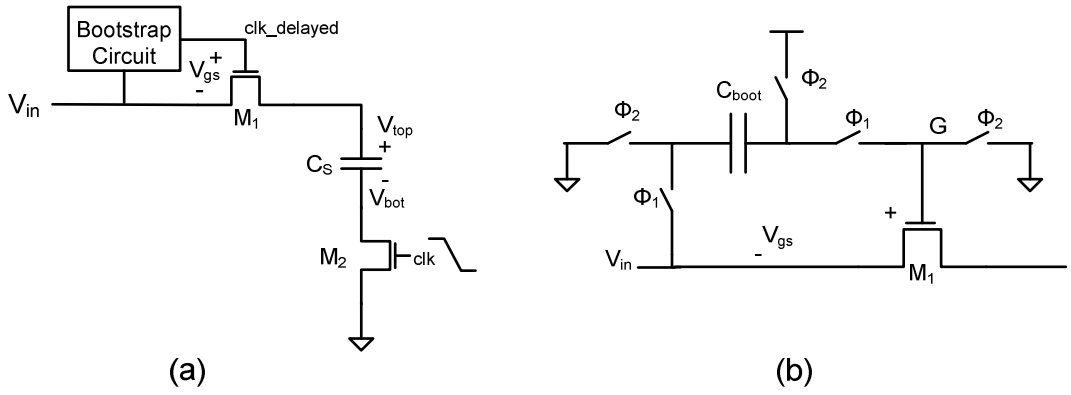

Figure 2.16: Bootstrap circuit for the sampling switch. (a) Bottom-plate track-and-hold with a bootstrap switch. (b) Bootstrap switch structure [32].

The switch resistance using the bootstrap architecture can be written as [36]

$$R_1 = \frac{1}{\mu_n C_{ox} \frac{W}{L}\left[\frac{C_{boot}}{C_{boot}+C_p}V_{DD} - \frac{C_p}{C_{boot}+C_p}V_{in} - V_{th}(V_{in},V_{top})\right]} \tag{2.6}$$

$$V_{th}(V_{in},V_{top}) = V_{th0} + \gamma(\sqrt{\varphi_0 + (V_{in},V_{top)}} - \sqrt{\varphi_0}) \tag{2.7}$$

where C_p is the parasitic capacitance at the gate of switch M_1, φ_0 is the surface potential and γ is the backgate effect parameter of the transistor.

As evident from (2.6), there is still a small input dependency in the bootstrapped switch resistance due to the backgate effect of the transistor and also the parasitic capacitance in the active bootstrap circuitry. The distortion caused by this nonlinear resistance is small and the circuit, if designed properly, is sufficiently linear for many ADC applications. However, due to the dynamic errors caused by this nonlinear resistance, SFDR can degrade significantly over input frequency and the circuit may not be linear enough for applications that need high SFDR performance at high input frequencies, such as the receiver in wireless base stations.

Figure 2.17 shows SFDR of the bottom-plate track-and-hold circuit vs. input frequency obtained from AC TCAD simulations (see Appendix A). In this figure, SFDR degradations from the two main sources of nonlinearity in the above circuit have been separated. In order to differentiate the two sources of nonlinearities in simulation, capacitor charge was measured at the end of the tracking phase (to model tracking error) and after M_2 turn off (to model charge injection). It can be observed from this figure that SFDR is mainly dominated by charge injection errors at low input frequencies and the tracking error becomes the main source of SFDR degradation at high input frequencies. Although both of these errors have dynamic effects, the charge injection error has mostly a static nature and does not change considerably with input frequency. Since the linearity of this stage is most important at high input frequencies, we will mainly focus on modeling and compensation of the tracking nonlinearity in the remainder of this dissertation. In the next section we investigate tracking nonlinearity in more detail in order to find a compact model that can be used for digital correction.

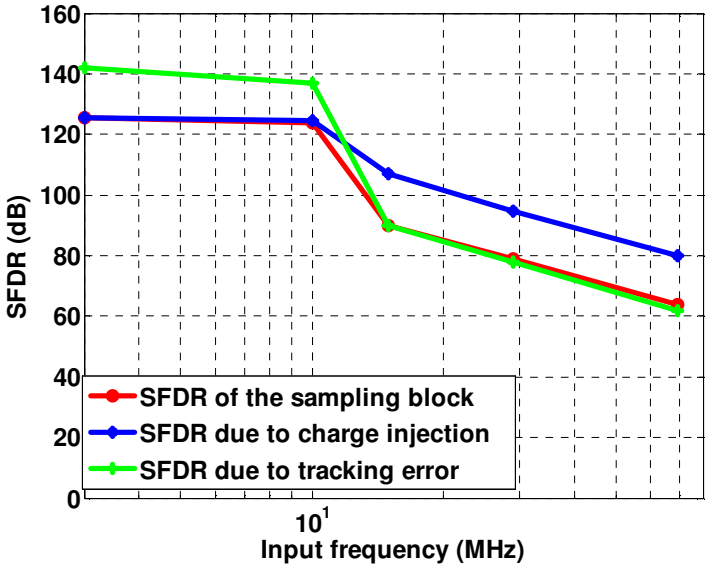

Figure 2.17: SFDR of the bottom-plate track-and-hold vs. input frequency obtained from AC TCAD simulations.

2.3.3 **Modeling tracking nonlinearity**

We start by looking at tracking nonlinearity in a flip-around track-and-hold amplifier which is commonly used at the front-end of high-speed and high-resolution ADCs. Figure 2.18 shows a single-ended schematic of this stage. The main part of this circuit is the same as the bottom plate track-and-hold that we discussed before; it only adds another switch and a transconductance amplifier in order to transfer the sampled signal to the ADC core for quantization [21]. This circuit works as follows: During the tracking mode (φ_1), M_1 and M_2 are closed and the voltage on the sampling capacitor (C) tracks the input signal. At the end of the tracking phase M_2 is turned off to freeze the charge at the bottom plate of the capacitor. M_1 is turned off with a small delay (φ_{1d}) to disconnect the input from the capacitor. During the hold mode (φ_2), M_3 turns on and transfers the sampled voltage to the input of the ADC core. Because the same capacitor is used for sampling and feedback, this circuit has an improved feedback factor compared to other track-and-hold amplifiers and therefore achieves better noise and speed performance [21].

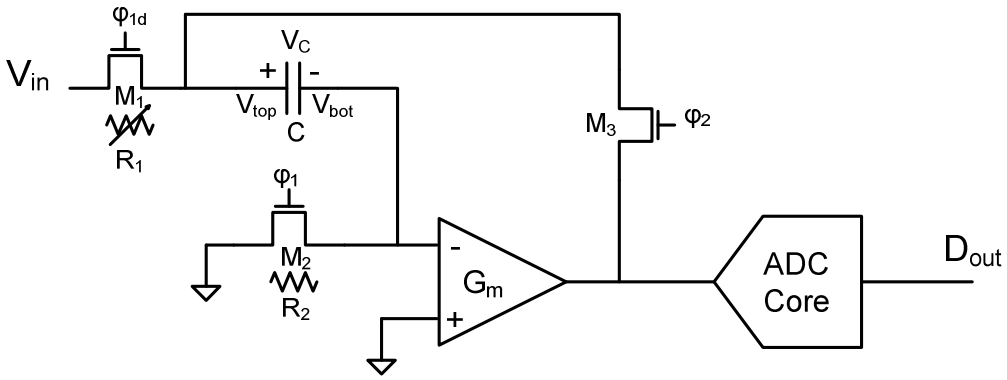

Figure 2.18: Schematic of a flip-around track-and-hold amplifier with bootstrapped switch.

During the tracking mode, the relation between V_{in} and V_C can be modeled with equation (2.4), where R_1 is a nonlinear function of input voltage defined by equation (2.6). During the hold mode, the charge at the bottom plate of the capacitor is

transferred to the output using switch M_3. Therefore, V_{out} is equal to the capacitor voltage at the end of the tracking phase. Using samples of the input and output voltage at the end of the tracking phase we can write the relation between discrete-time samples of V_{in} and V_{out} with the following equation

$$V_{in}(k) = V_{out}(k) + (a_0 + a_1 V_{out}(k) + a_2 V_{out}{}^2(k) + ...) \times \frac{dV_{out}(k)}{dt} \tag{2.8}$$

This equation is written based on the assumption that R_1 can be approximated as a static nonlinear function of V_{out}. From equation (2.6), we know that R_1 is a nonlinear function of V_{in} and V_{top}. In a sampling network with sufficiently large bandwidth, V_{top} and V_C track the input closely and differ by a weakly nonlinear term that is determined by the respective RC product and the signal derivative. Therefore, the resistance of M_1 can be approximated by a nonlinear static function of V_C and therefore $V_{out}(k)$.

In the above equation V_{in} is modeled as a nonlinear function of V_{out} with memory, where nonlinearity only comes from R_1 and is a static function and memory is all included in the derivative and is a linear function. Separating these two effects in our distortion function helps us simplify the correction filter as will be discussed in detail in Chapter 3.

2.4 Nonlinearity in the input circuit

As mentioned in Section 2.2, second-order non-idealities at the input driving circuit of the ADC can add more dynamic nonlinear errors to the system. Figure 2.19 shows a complete schematic of the front-end of the ADC including flip-around track-and-hold amplifier, package parasitics and the input driving circuit including a transformer for converting single-ended to differential signal. The input driving circuit can alternatively include buffers, differential amplifier or other kind of drivers, but the general form of nonlinearity in all of them is the same and will not change the analysis in this section.

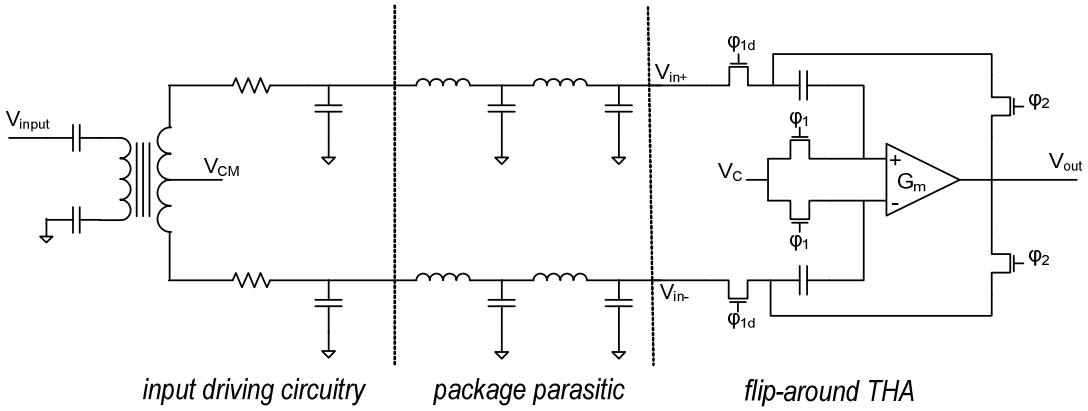

Figure 2.19: Schematic of the front-end of the ADC including flip-around track-and-hold amplifier, package parasitic and the input driving circuit.

The input driving circuit and package parasitics in this circuit can impact the distortion in the system. In a flip-around track-and-hold amplifier, the capacitor is not reset between the two sampling cycles and hence causes charge glitches to go to the input circuitry due to the difference between two successive samples. These charge glitches cause spikes on the input current as shown in Figure 2.20. These abrupt changes in the input current passing through the parasitic inductances can cause ringing and other transient response on the input voltage as shown in Figure 2.21. If the input circuitry is not fast enough to recover from this transient response by the time of the sampling, it can increase the nonlinear sampling error as will be discussed in detail in the following paragraphs.

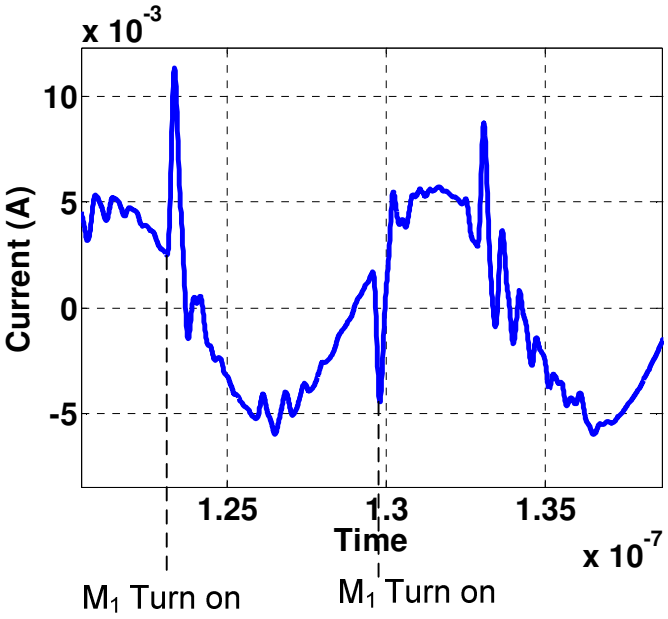

Figure 2.20: Current spikes at the input driving circuit caused by charge glitches coming from the sampling capacitor.

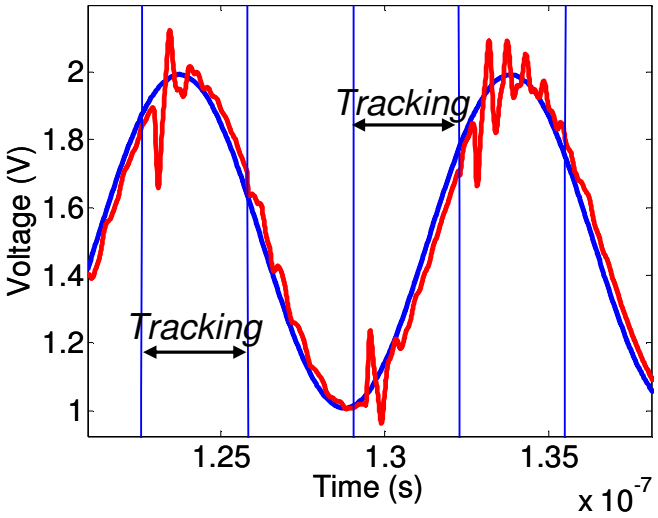

Figure 2.21: Input signal distorted by the ringing caused by parasitic inductances (L_{par}=3 nH, C_S=3 pF, C_{par}=6 pF).

2.4.1 **Input circuit including transformer**

In order to analyze the distortion generated in the circuit of Figure 2.19, we can first assume that we have an ideal transformer and no package parasitics; therefore the

input driving circuit together with the track-and-hold can be approximately modeled as shown in Figure 2.22(a). During tracking mode, the sampling switch is closed and the circuit can be modeled as a second order RC circuit as shown in Figure 2.22(b). The new distortion function during tracking mode can be modeled with the following differential equation

$$V_{in} = V_{out2} + (R_2C_2 + R_1C_2 + R_1C_1)\frac{dV_{out2}}{dt} + R_1R_2C_1C_2\frac{d^2V_{out2}}{dt^2} \qquad (2.9)$$
$$V_{out2}(t=0) = V_{20} \;,\; V_{out1}(t=0) = V_{10}$$

where V_{10} and V_{20} are the initial voltages on the two capacitors before starting the next tracking phase.

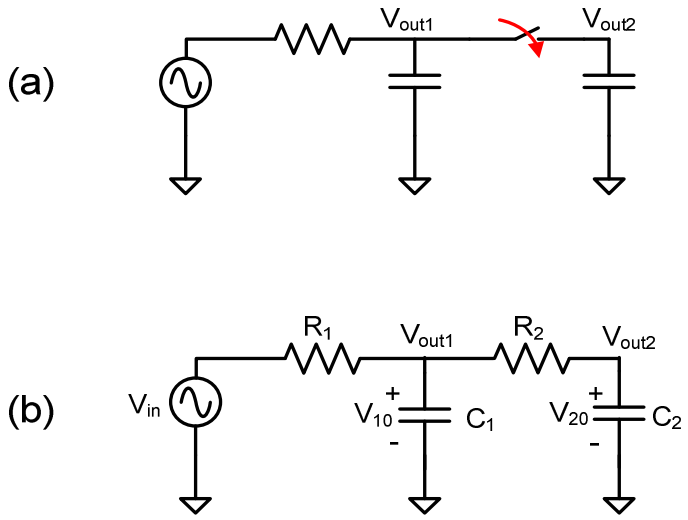

Figure 2.22: Simple model for the input circuit and the track-and-hold.

Due to the existence of a second capacitor from the input driving circuit, the differential equation is second order and includes dependency to the second derivative of the signal. In this model, again, V_{in} can be written as a nonlinear function of V_{out} with memory. Nonlinearity comes from R_1 and is a static function; however, memory effect is all included in the first and second signal derivative and is a linear function. Therefore, we can still model V_{in} as a product of a memoryless nonlinear function and

a linear function with memory which is still compatible with our initial model (equation (2.8)).

$$V_{in}(k) = f_{nonlin}(V_{out}(k)) \times f_{lin}(V_{out}(k), V_{out}(k-1), V_{out}(k+1), ...) \tag{2.10}$$

where f_{nonlin} is the result of RC product and f_{lin} is generated from different derivative orders.

Depending on the initial voltages on the capacitors the response of this system to a sinusoidal input voltage can be shown as

$$V_{out2} = k_1 e^{-\frac{t}{\tau_1}} + k_2 e^{-\frac{t}{\tau 2}} + A\sin(\omega t + \varphi) \tag{2.11}$$

where τ_1 and τ_2 are time constants of the circuit and k_1 and k_2 are defined based on initial voltages and the input sinewave. This shows that the output voltage has a transient response first, before reaching the steady-state sinewave response. If this transient has not settled completely by the time of the sampling, it causes dependency to $V_{out}(n-1)$ in the nonlinear part of the output voltage.

Now we can also add package parasitics to our model. The simplified circuit model including wire-bond inductance and pad capacitances is shown in Figure 2.23. The relation between V_{in} and V_{out} in this circuit can be modeled with a 3rd order differential equation as shown below

$$V_{in} = V_{out2} + (R_2C_2 + R_1C_2 + R_1C_1)\frac{dV_{out2}}{dt} + (R_1R_2C_1C_2 + L_1C_1 + L_1C_2)\frac{d^2V_{out2}}{dt^2} + (L_1C_1R_2C_2)\frac{d^3V_{out2}}{dt^3} \tag{2.12}$$

$$V_{out2}(t=0) = V_{20}\ ,\ V_{out1}(t=0) = V_{10}\ ,\ I_{l1}(t=0) = 0$$

Again, in this equation memory comes from different orders of signal derivative and is a separate factor from static nonlinearity. This can be used for simplifying the distortion function into equation (2.10).

In the above equation, depending on the roots of the characteristic function, there might be ringing in the transient response, with an amplitude defined by the previous stored voltage on the capacitor. The time constants of this circuit define how fast the ringing decays. If it has not completely vanished by the time of the sampling, the ringing causes dependency to the previous sampled value in the nonlinear output voltage.

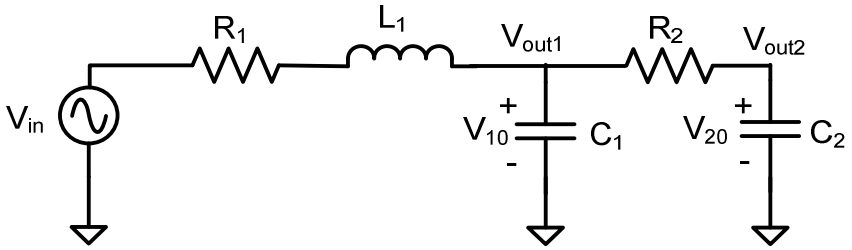

Figure 2.23: Model of the track-and-hold, input driving circuit and package parasitics.

A more realistic transformer model also includes parasitics as shown in Figure 2.24 [37], [38]. In this circuit, R_p and R_s represent the resistive loss and the loss in the ferrite core at low frequencies. L_p and L_s represent the leakage in the inductance and are caused by the incomplete magnetic coupling between the two windings. L_m represents the magnetizing inductance and is related to the permeability and cross sectional area of the magnetic core and the number of turns in the windings. R_m represents the hysteresis loss and eddy-current loss in the core, C_p and C_s represent coupling between turns of each winding and C_{ps} represents coupling between the primary and the secondary windings.

If all the elements in the transformer parasitic model are constant, the transformer is a linear block and only adds poles to the system transfer function but do not change the nonlinear model from the general form shown in equation (2.10).

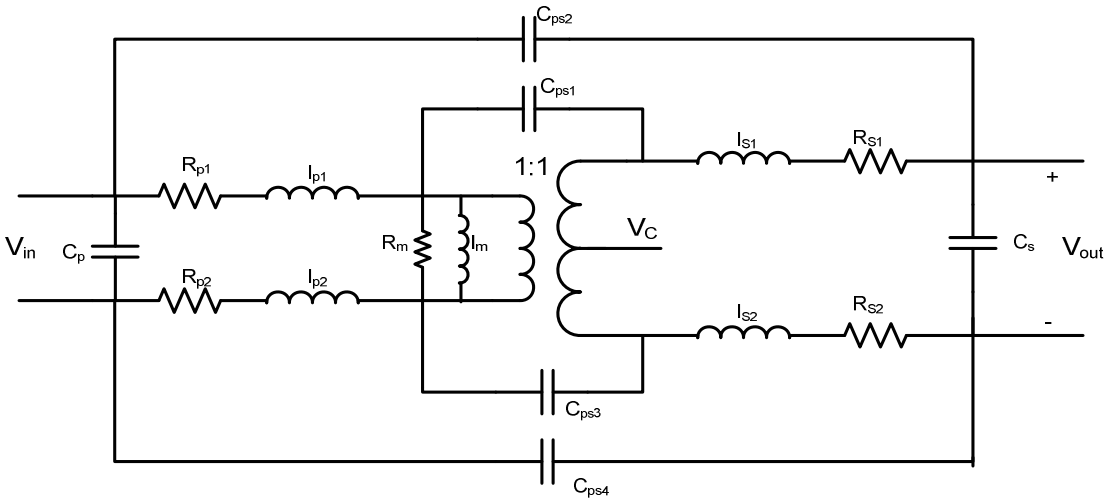

Figure 2.24: Schematic of a real transformer model with its parasitics.

The core of the transformer can sometimes be a source of distortion due to the nonlinear relation between magnetic flux density (B) and the intensity of magnetic field (H) [39], [40]. However, this is usually not a significant problem in RF transformers and at the voltage levels (< 3V) we are working with. Even if it is a significant nonlinear factor, it can be modeled with some nonlinear parasitic components and does not affect the general form of our nonlinear function (the effect amounts to a static nonlinearity that can be absorbed in the memoryless nonlinear portion of our model).

Some of the parasitic components in a transformer such as R_m, R_p and R_s are frequency dependent due to the increased skin effect in wires at high frequencies [37], [38]. This frequency dependence adds another source of memory to the system and generates terms such as $(dV_{out}/dt)^2$ in our nonlinear model, which are not compatible with the general form of the distortion function shown in equation (2.10). However, these are small second order effects and are usually negligible compared to other distortion sources in the front-end transfer function.

The main nonlinearity problem at the front-end caused by the transformer is due to the mismatch in its parasitic components for example the coupling capacitances. This mismatch can cause a large phase and magnitude imbalance between the two output ports of the transformer at high input frequencies and can lead to even order

harmonics [41], [42]. These even order harmonics have the same form as what is already included in our model, i.e. a product of a static nonlinear function and a linear function with memory. The harmonics can therefore be cancelled using our correction filter developed in the next chapter.

2.4.2 **Input circuit including amplifier**

In some applications, it is necessary to use a buffer or amplifier at the front-end of the ADC in order to keep input impedance constant during tracking and hold mode, help driving the input load or to increase the dynamic range of the system. These amplifier stages usually add significant amounts of noise and distortion to the system and make it harder to achieve high SFDR values.

Figure 2.25 shows a simple model including amplifier stage and the track-and-hold circuit. If the nonlinearity caused by the amplifier is static, it can be modeled with the following equation

$$V_{out1} = f_{nonlin}(V_{in}) \Rightarrow V_{in} = g_{nonlin}(V_{out1}) = a_0 + a_1 V_{out1} + a_2 {V_{out1}}^2 + ... \qquad (2.13)$$

The relation between V_{out} and V_{out1} is therefore the same as track-and-hold nonlinear model shown in (2.4). Therefore, the relation between V_{in} and V_{out} can be expressed by the following equation

$$V_{out1} = V_{out} + RC\frac{dV_{out}}{dt} \Rightarrow V_{in} = a_0 + a_1(V_{out} + RC\frac{dV_{out}}{dt}) + a_2(V_{out} + RC\frac{dV_{out}}{dt})^2 + ... \qquad (2.14)$$

$$V_{in} = g_{nonlin}(V_{out1}) = a_0 + a_1 V_{out1} + a_2 V_{out1}^{\ 2} + \ldots$$

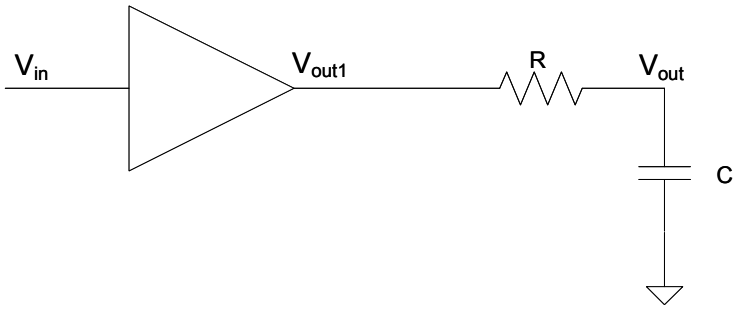

Figure 2.25: Basic model for the input circuit including buffer/amplifier.

The new distortion model shown in (2.14) is different from our initial distortion function (equation (2.8)) and can not be completely simplified into the form shown in equation (2.10). This model includes terms such as $(dV_{out}/dt)^2$, $(dV_{out}/dt)^3$, … which are not included in the general form of equation (2.10); however, the main part of distortion function can be fit into that equation and can be corrected with our compensation filter discussed in next chapter.

All the circuit nonidealities discussed in this section can be a source of dynamic nonlinearity in the system when used together with a track-and-hold stage. Although, it is important to find a general compact model that can include these errors into account, these are second order effects that add to the dynamic nonlinearities mainly generated at the track-and-hold stage.

2.5 Summary

In this chapter, we investigated the dynamic nonlinearity problem at the front-end of ADCs and introduced different sources causing this problem. The frequency-dependent nonlinear error at the front-end of the ADC mainly comes from the track-and-hold stage and is divided between the two sources of charge injection and tracking error. We showed in our simulation results that charge injection is a main source of linearity degradation at low input frequencies and tracking error becomes dominant at high input frequencies.

The rest of the chapter was focused on tracking nonlinearity and how it can be modeled in presence of other circuit components such as package parasitics, nonideal transformers or amplifiers. It was shown that the main part of the nonlinear function in these circuits can be modeled as a product of a memory-less nonlinear function and a linear function with memory. This is an important property that will be used in the next chapter for simplifying the correction process.

Chapter 3

Digital Correction of Dynamic Nonlinearities

It was shown in the previous chapter that the front-end of an ADC generates dynamic nonlinear errors which are a limiting factor in its linearity performance at high input frequencies. As mentioned in Chapter 1, previous efforts on achieving high frequency linearity have mainly focused on analog approaches such as using emitter followers in the sampling stage or modifying the switch bootstrap structure [1], [2]. These methods are either based on using expensive technologies or suffer from bandwidth limitations in the analog circuitry and do not keep good linearity up to very high input frequencies. In recent years, digital correction methods have been introduced to compensate circuit nonlinearities in ADCs, see e.g. [3]-[5]. However, these techniques mainly address static errors in the converter core and are not effective at removing dynamic nonlinearities in the THA.

With the recent scaling in CMOS technologies and the possibility of integrating complex digital processing functions into the system, digital compensation methods can be extended to correct for dynamic nonlinearities in ADCs and to achieve good linearity performance at high input frequencies. Several studies have been done on using phase-plane tables or Volterra series for compensating memory-dependent nonlinearities [10], [43]-[45]. However, these methods tend to be impractical and too complex for systems with a high order of memory or nonlinearity. In this chapter we will briefly look at previous approaches for canceling nonlinearity in the digital

domain and then introduce the proposed correction algorithm, which uses judicious modeling of the nonlinearity to simplify the correction process.

3.1 Dynamic error correction using look-up tables

Digital post correction using look-up tables (LUT) has been studied extensively in literature [43]-[48]. The general structure of the error correction using look-up tables is shown in Figure 3.1. The basic idea of this method is that the output samples from the ADC are used as an index (or address) to a table. The index points at a specific entry in the table and the corresponding value is either added to or used to replace the current ADC output sample [43].

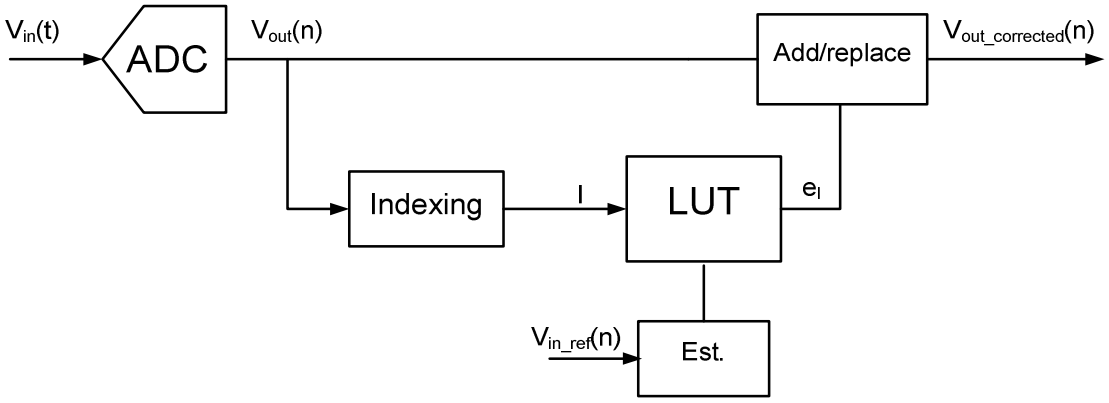

Figure 3.1: Block diagram of ADC error correction using LUT.

Different methods of LUT correction are classified based on the scheme used for indexing the table, which can be either static, state-space or phase-plane [46], [48]-[50]. A static look-up table correction scheme maps the present sample x(n) into an index I which depends neither on past nor future samples. As obvious from the name, this method can only be used for canceling static errors and is not useful for dynamic nonlinearities. One way to introduce dynamic effects into this correction method is to use a state-space structure. In this scheme the current (x(n)) and previous samples (x(n-1)) are used to build the index. This can be described as a two dimensional LUT

where x(n) and x(n-1) are used to index the two dimensions. Although this method can be used for correcting dynamic errors, it can only model errors with dependency between adjacent samples. However, most dynamic errors have a higher memory order and can not be corrected with only a two dimensional table. This method can be generalized to using K delayed samples together with the current sample; however the table size grows very fast with this dimension extension, and the memory required to store that would become prohibitively large.

As an alternative to state-space indexing, phase plane indexing can be used as shown in Figure 3.2 [45], [48]. In this method, the table index is constructed from the present sample x(n) and an estimate of the derivative of the input signal dx(t)/dt at the sampling point. The slope of the signal can be either measured with extra hardware [51] or estimated from the output of a digital filter [47]. Phase-plane tables produce a better estimate of the memory-dependent error compared to two dimensional state-space tables; however, they are still not accurate enough for modeling some dynamic errors. Also the size of the table is too large to be used for high-speed and high-resolution ADCs.

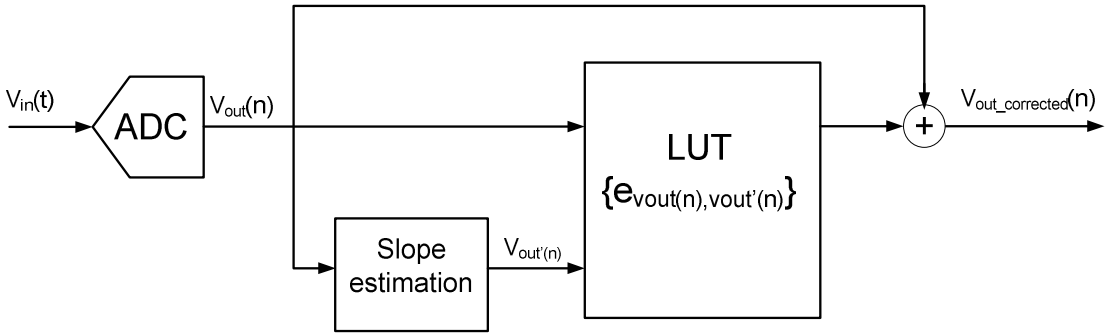

Figure 3.2: ADC dynamic error correction using phase-plane look-up table.

3.2 Modeling nonlinear systems with memory using Volterra series

A wide class of nonlinear systems with memory can be modeled with a Volterra series [6], [7]. Therefore, a possible approach for compensating dynamic ADC nonlinearities is to obtain a model by estimating the Volterra kernels and using an approximate Volterra inverse to recover the distortion [10].

As we saw in the previous chapter, in the track-and-hold stage at the front-end of ADC, there is a nonlinear relation with memory between the input and output voltage that can be modeled as the Volterra series of the form

$$V_{out}(k) = f_{nonlin}\left[V_{in}(k), V_{in}(k-1), V_{in}(k-2), \ldots, V_{in}(k-n)\right] \tag{3.1}$$

In a discrete-time system, the Volterra filter can be expressed by the following equation [6]

$$\begin{aligned} y(k) = h_0 &+ \sum_{m_1=0}^{M-1} h_1(m_1)x(k-m_1) + \sum_{m_1=0}^{M-1}\sum_{m_2=0}^{M-1} h_2(m_1,m_2)x(k-m_1)x(k-m_2) + \ldots \\ &+ \sum_{m_1=0}^{M-1}\ldots\sum_{m_n=0}^{M-1} h_n(m_1,m_2\ldots m_n)x(k-m_1)\ldots x(k-m_n) + \ldots \\ &+ \sum_{m_1=0}^{M-1}\ldots\sum_{m_N=0}^{M-1} h_N(m_1,m_2\ldots m_n)x(k-m_1)\ldots x(k-m_N) \end{aligned} \tag{3.2}$$

where M is the memory order of the system. Unfortunately, a well-known issue with Volterra models is that even for relatively simple systems, the corresponding inverse function (for error correction) can be prohibitively complex and needs a lot of computation [8]-[10]. In order to address this issue and to minimize the hardware requirement for the digital correction, we simplify the distortion model using circuit-specific insight and judicious approximations as discussed in the previous chapter.

3.3 Proposed digital algorithm

In the previous chapter, we found that an appropriate distortion model for ADC's front-end is given by the product of a memory-less nonlinear function and a linear function with memory (see (2.8)). This model has significantly fewer terms compared to the general form of the Volterra series shown in (3.2). Also, we note that in contrast to (3.1), the distortion model of (2.8) is already in its inverse form, i.e. $V_{in} = f(V_{out})$. Therefore, this expression is directly applicable for correction and no inversion is needed.

Figure 3.3 shows a block diagram of the track-and-hold together with the ADC and the digital post-processing. As shown in this figure, V_{out} of the THA goes to the ADC core, is quantized with ADC's resolution and generates the digital signal D_{out}. Provided that V_{out} is processed by the ADC core with sufficient resolution and linearity, (2.8) can be applied in the digital domain, i.e. $V_{out} \cong V_{ref} \cdot D_{out}$, where V_{ref} is the reference voltage of the converter.

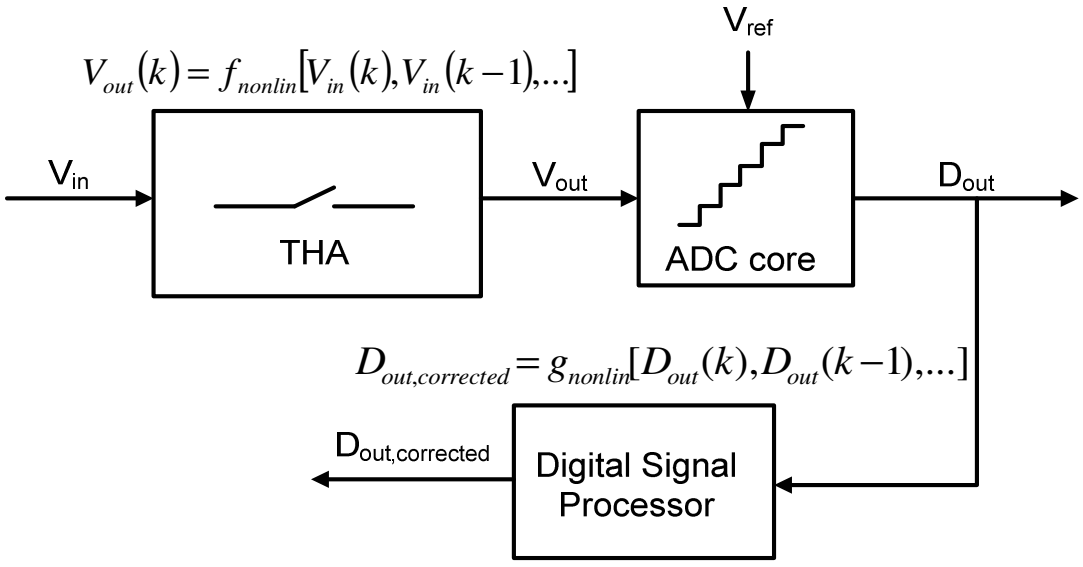

Figure 3.3: Block diagram of the track-and-hold together with the ADC and digital post-processing.

Having the signal derivative in the distortion model introduces memory effect into our nonlinear system. In order to find the correction filter using the distortion model of equation (2.8), we need to find a model for the derivative based on the output samples. Having a linear model for derivative and a nonlinear model for tracking resistance with unknown coefficients, we need to use training signals for calibrating the coefficients and using them in our nonlinearity correction filter. In the following sections, we first show how derivative can be modeled as a linear combination of previous and post samples and use that in the compensation filter and the calibration process. We will then discuss the issues with this approach and present a more efficient method which uses interpolation for modeling the signal derivative.

3.3.1 **Digital Correction using linear modeling of the derivative**

As shown in Appendix B, in each Nyquist zone[1], the slope of the signal at the sampling instants can be modeled as a linear function of previous and post samples. Therefore, the distortion model shown in (2.8) can be written as

$$V_{in}(k) = V_{out}(k) + \sum_{i=0}^{n} a_i {V_{out}}^i(k) \times \sum_{j=-M}^{M} b_j V_{out}(k-j) \tag{3.3}$$

and hence the following set of equations for each sample is generated

$$V_{in}(k) = \sum_{i=0}^{n} \sum_{j=-M}^{M} h_{ij} {V_{out}}^i(k) V_{out}(k-j) \tag{3.4}$$

This is a linear equation with respect to the coefficients h_{ij}; it can therefore be solved using Least Square (LS) methods [52] in order to find the coefficients that produce the best fit to a set of input and output samples. In order to calibrate the

[1] According to Nyquist theory of sampling, bandwidth of a signal should be smaller than half of the sampling frequency to be able to reconstruct the signal from its samples. This theory is true regardless of which Nyquist zone the signal is located in. Nyquist zones on the frequency axis are defined as regions with frequency span of $f_{clk}/2$ distributed in the entire input bandwidth of the system.

coefficients, training signals can be used to generate sets of points required for solving the equations (see Figure 3.4). As shown in Appendix B, the linear derivative model depends on band-pass filter coefficients and hence is different in each Nyquist zone. Due to this Nyquist zone dependence and also the approximations made in deriving the model in the previous section, the coefficients h_{ij} must be found for each desired Nyquist zone of operation and training signals within this frequency range must be used. After applying the training signals, the following matrix equation is formed and h_{ij} are extracted using LS solutions [52].

$$\begin{bmatrix} V_{in}(k) \\ V_{in}(k-1) \\ \cdots \\ V_{in}(1) \end{bmatrix} = \begin{bmatrix} V_{out}(k) & V_{out}(k-1) & V_{out}(k+1) \ldots & V_{out}^{\,2}(k) & V_{out}(k)V_{out}(k-1) & V_{out}(k)V_{out}(k+1) \ldots \\ V_{out}(k-1) & V_{out}(k-1) & V_{out}(k) \ldots & V_{out}^{\,2}(k-1) & V_{out}(k-1)V_{out}(k-2) & V_{out}(k-1)V_{out}(k) \ldots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ V_{out}(1) & V_{out}(0) & V_{out}(2) \ldots & V_{out}^{\,2}(1) & V_{out}(1)V_{out}(0) & V_{out}(1)V_{out}(2) \ldots \end{bmatrix} \times \begin{bmatrix} h_{00} \\ h_{01} \\ \cdots \\ h_{nm} \end{bmatrix} \tag{3.5}$$

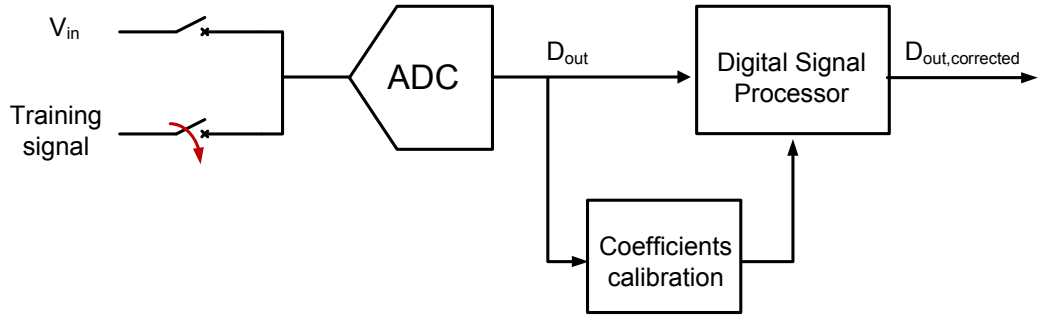

Figure 3.4: Calibrating coefficients required for the correction filter using training signals.

Assuming that the output of the track-and-hold circuit is quantized with the ADC core with sufficient resolution and accuracy, coefficients extracted in the above equation can be used to form the correction filter for nonlinearity compensation. Since

our distortion function is already in its inverse form we do not need to find its inverse and the correction filter can be formed according to equation (3.4) to be applied to the digital output of the ADC as shown in Figure 3.3 and the following equation

$$D_{out,corrected}(k) = \sum_{i=1}^{n} \sum_{j=-M}^{M} h_{ij} D_{out}{}^{i}(k) D_{out}(k-j) \tag{3.6}$$

Note that the linear terms are removed from the correction filter (i=0), since they do not contribute to the nonlinearity of the system.

By comparing our distortion model (equation (3.4)) with the general form of the Volterra series shown in equation (3.2), we observe that the number of coefficients has significantly reduced in our model, which is a special case of the Volterra series. For example, for a memory order of M and nonlinearity order of 3, the number of coefficients required for each of the two models is shown in the following equations

$$\begin{aligned} &Number\ of\ coefficients\ in\ Volterra\ series(11) \rightarrow M + M^2 + M^3 \\ &Number\ of\ coefficients\ in\ our\ filter(8) \rightarrow 2\times(M+M+M) = 6M \end{aligned} \tag{3.7}$$

Therefore, e.g. for M=40 (which is the memory order required in our system as will be discussed later), the number of required coefficients will be 65640 in (3.2) and 240 in (3.4), which makes a significant difference.

3.3.1.1 **Evaluation of the model: Random noise input**

As we saw earlier, nonlinearity in a track-and-hold circuit is a function of input voltage and its slope at the sampling instant. Therefore, in order to achieve proper calibration, we need to cover sufficient information on all possible combinations of sampled voltages and slopes in our training sequence. As a result, random noise would be a good choice. On the other hand, due to the bandwidth limitation of the track-and-hold circuit, the training signal has to be bandlimited to $f_s/2$ and should also be located in the desired frequency range for which we want to find the correction model.

Therefore, a bandpass filtered version of white noise in the corresponding Nyquist zone can be used as the training sequence.

The track-and-hold model discussed in Chapter 2 was emulated in Matlab Simulink as shown in Figure 3.5. In this model C was chosen constant (C=2pF) and R was modeled with the following equation

$$\begin{aligned} R &= R_1 + R_2 \\ R_1 &= R_1(V_{in}) = \frac{1}{\mu_n C_{ox} \frac{W_1}{L_1}\left[V_{DD} - V_{th}(V_{in})\right]} \\ R_2 &= \text{Constant} = \frac{1}{\mu_n C_{ox} \frac{W_2}{L_2}\left[V_{DD} - V_{th0}\right]} \end{aligned} \tag{3.8}$$

where V_{DD}=3.3 V, $\mu_n C_{ox}$=50 μA/V^2 , W_1/L_1=100, W_2/L_2=500 and V_{th0}=0.4 V. V_{th} of switch M_1 was defined with the following equation

$$V_{th}(V_{in}) = V_{th0} + \gamma(\sqrt{\varphi_0 + V_{in}} - \sqrt{\varphi_0}) \tag{3.9}$$

where γ= 0.4 and φ_0= 0.7 V. The training signal was created using a pseudo random sequence passed through a bandpass filter corresponding to each Nyquist zone. This colored noise was applied to the input of the model and a set of input and output samples were used in the calibration model to find its coefficients. Since 2nd and 3rd harmonics are usually significantly larger than higher order harmonics in our system, the calibration model was formed with nonlinearity order of 3 (n=2 in (3.6)). In order to find the required number of previous and post samples ("M" in equation (3.6)) for accurate derivative modeling and for proper correction of the system nonlinearity, we plotted the error in the corrected signal versus this parameter (see Figure 3.6). As can be observed from this figure, the error decreases monotonically until M=40 and stays almost constant after that. Therefore, we used a memory order of 40, which brings the total number of coefficients in the filter to 3 × (2×40+1) = 243. This memory order in

the system is due to the number of terms required for modeling derivative accurately (see Appendix B).

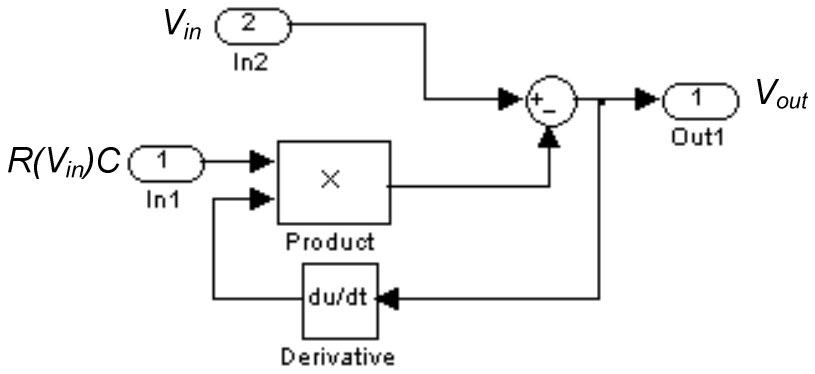

Figure 3.5: Block diagram of the tracking nonlinearity modeled in Matlab Simulink.

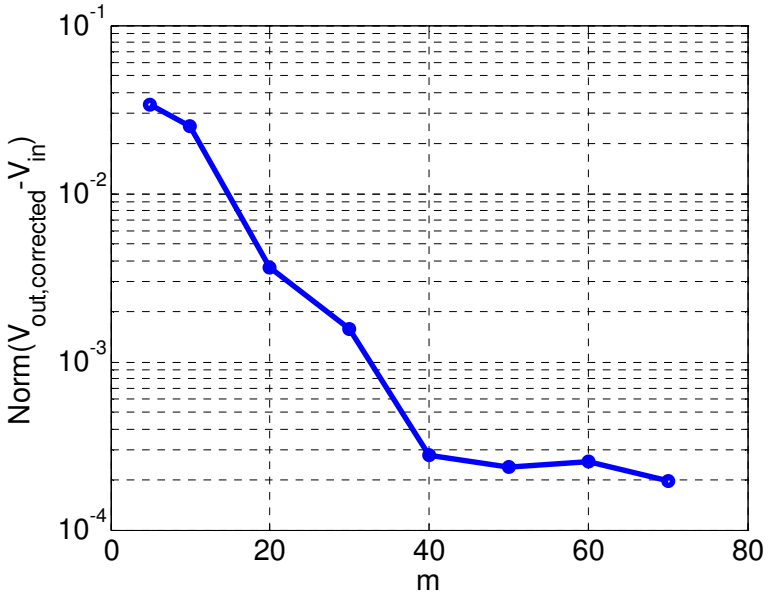

Figure 3.6: Error plot for corrected signal vs. number of previous and post samples (m).

Figure 3.7 shows SFDR plots versus input frequency in the first four Nyquist zones of this system with a sample rate of 100 MS/s. Each Nyquist zone is defined as a region with frequency span of $f_{clk}/2$ distributed in the entire input bandwidth of the system. The plots show SFDR before and after the digital enhancement. As we can see in this figure, SFDR has improved by more than 20 dB in all these frequency zones.

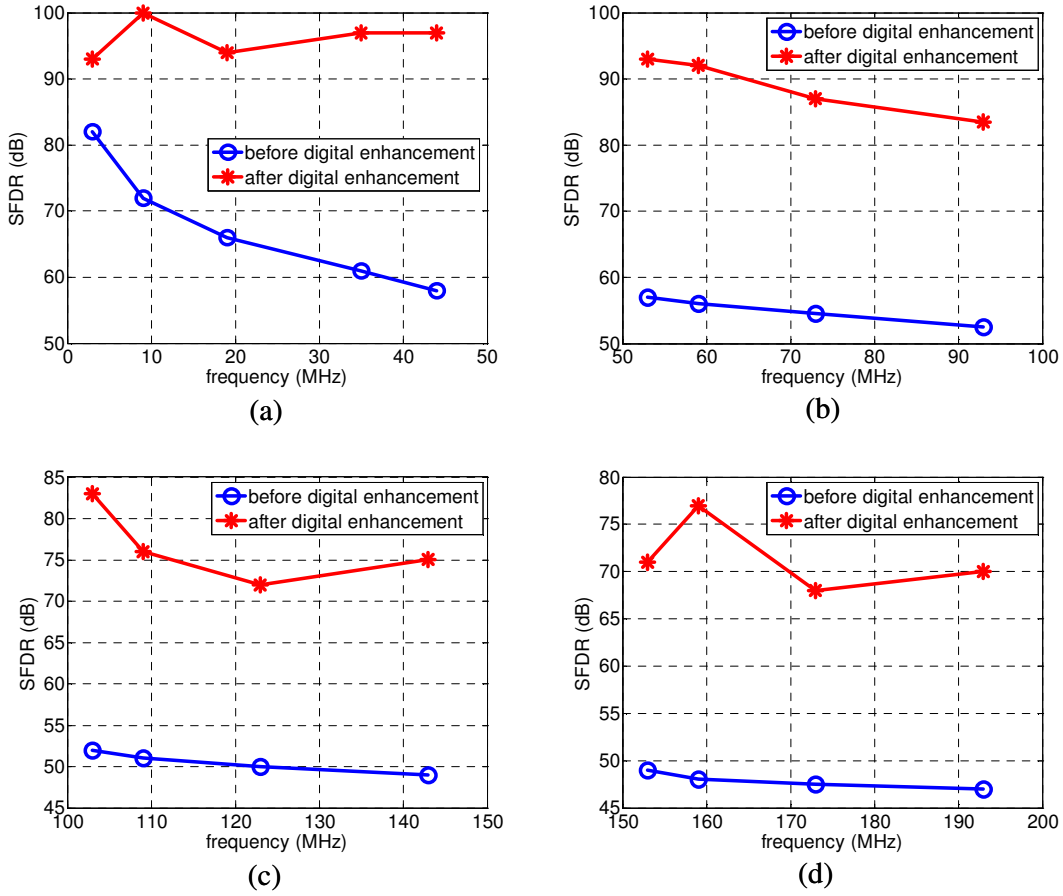

Figure 3.7: SFDR vs. input frequency in the first 4 Nyquist zone (f_{clk}=100 MHz) before and after digital enhancement, using noise signal for calibration. (a) SFDR in the 1st Nyquist zone. (b) SFDR in the 2nd Nyquist zone. (c) SFDR in the 3rd Nyquist zone. (d) SFDR in the 4th Nyquist zone.

Although the above calibration method works well in simulation, it has some practical drawbacks. As we saw in the previous section, in order to find the correction filter coefficients, a set of input and output samples must be known for the LS method (equation (3.5)). Assuming sufficient accuracy in the ADC core, output samples of the

track-and-hold are representative of the digital outputs of the ADC; but how can we find the input samples?

The input signal is an analog voltage; therefore, in order to know its exact sampled value, we should generate it in the digital domain and convert it into an analog voltage using a DAC as shown in Figure 3.8(a). However, the DAC itself adds errors and nonlinearity to the signal. Therefore, in order to measure nonlinear errors of the ADC and to compensate them, a very accurate DAC with greater linearity than that desired in the ADC is required. This is not easy to achieve in practice. Furthermore, the output of the DAC is in the first Nyquist zone and must be upconverted to the desired calibration zone, using a mixer (see Figure 3.8(a)). The mixer also introduces nonlinearities to the system; therefore, for proper calibration, the linearity of the mixer must also be better than the desired performance in the ADC.

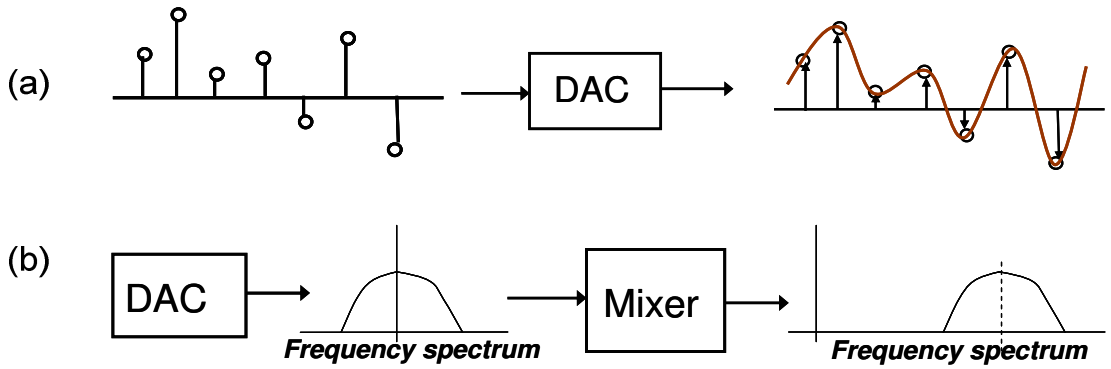

Figure 3.8: Block diagram of the calibration using random input samples. (a) Generating random samples in the digital domain and converting them into an analog voltage using a DAC. (b) Using a Mixer to shift the DAC output to the desired Nyquist zone.

3.3.1.2 **Evaluation of the model: Sinewave input**

Due to the practical limitations in the above calibration scheme, we consider using a different kind of signal as our training sequence. We need to use a bandlimited signal whose input samples can be estimated from the digital output. Therefore, single-tone, clean sinewaves are a good choice for the training signal. In order to have

different sampled values and slopes for the calibration, sinewaves with several different frequencies in the desired Nyquist zone need to be used. After applying the sinewave to the input, digital outputs are read and input samples are estimated by fitting a sine function with known frequency to the output samples. Although this estimated signal has some amplitude and phase difference from the actual input samples, it does not contain any harmonics. Hence, calibrating the filter coefficients using this signal will minimize the output nonlinearity.

The track-and-hold circuit model shown in Figure 3.5 was emulated in Matlab and calibrated using sinewaves in each Nyquist zone. A memory order of M=40 was used as in the case for noise calibration input. Due to the large number of previous and post samples used in the calibration model, a large number of test tones need to be used in each Nyquist zone in order to be able to correct any bandlimited signal in that zone. The number of required calibration tones is directly related to the memory order of the filter. Our simulation results show that for best correction with memory order of M=40, sinewaves with frequency steps of 1 MHz need to be applied as calibration signals.

Figure 3.9 shows the SFDR result of the system before and after digital enhancement in the first four Nyquist zones. We can see from these plots that SFDR has improved by more than 30 dB in all these frequency ranges.

As shown in Appendix B and discussed in previous sections, in a sub-sampled system a large number of previous and post samples are required to have an accurate estimate for the derivative. Using this model in the correction filter structure has the drawback of introducing a large number of unknown coefficients in the system which need to be extracted in the calibration process. Also, as was seen in the previous section, this large memory order increases the number of test signals required for proper calibration. Therefore, in order to reduce the number of model coefficients and the number of training signals, it is desirable to find a model for the derivative with fewer coefficients. This can be done using interpolation.

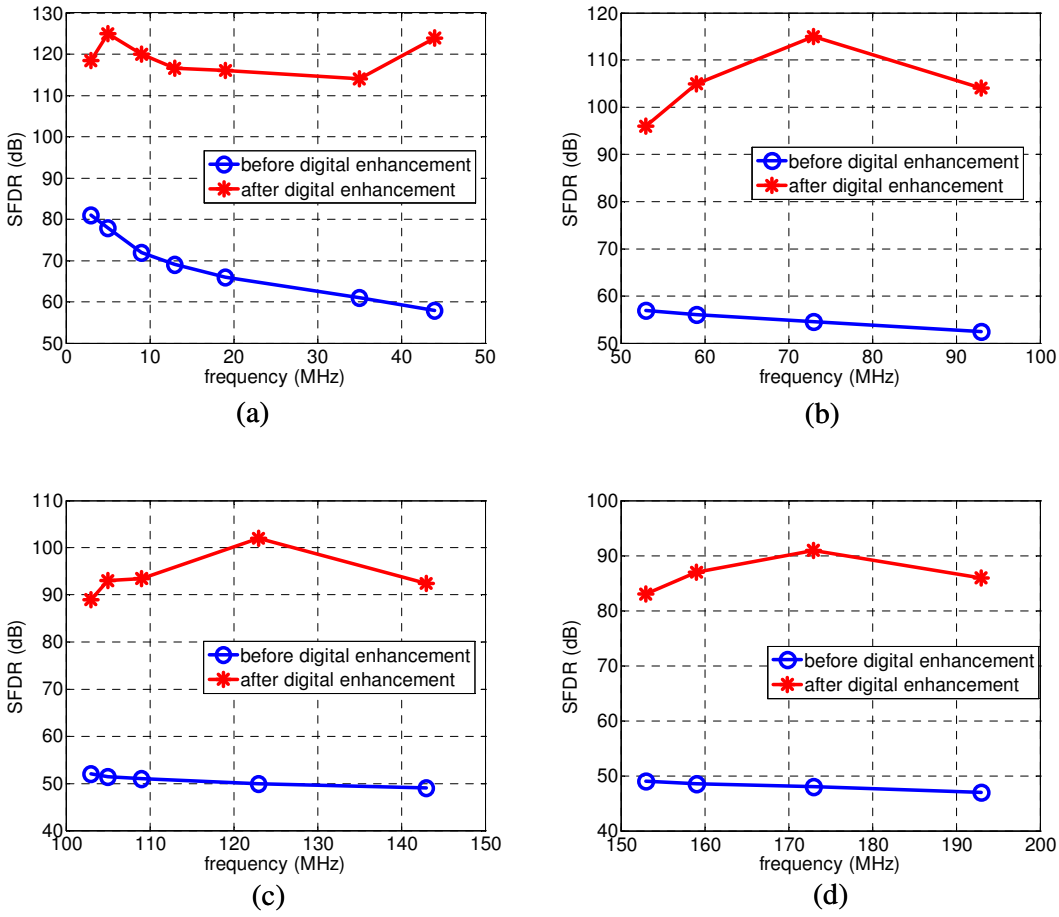

Figure 3.9: SFDR vs. input frequency in the first 4 Nyquist zone (fclk=100 MHz) before and after digital enhancement, using sine waves for calibration. (a) SFDR in the 1st Nyquist zone. (b) SFDR in the 2nd Nyquist zone. (c) SFDR in the 3rd Nyquist zone. (d) SFDR in the 4th Nyquist zone.

3.3.2 **Digital correction using interpolation**

Interpolation can be used to reconstruct samples of a desired bandlimited signal with higher sampling rate. Figure 3.10 shows a sub-sampled sinewave and its reconstructed samples. After doing the interpolation, only a few of the reconstructed samples are required in order to model the derivative and this will considerably simplify the distortion model and hence the correction filter.

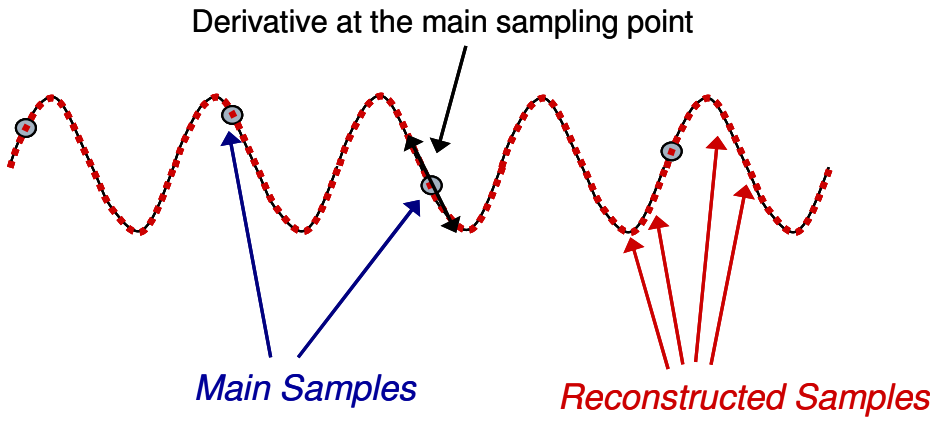

Figure 3.10: Subsampled signal with interpolation.

3.3.2.1 **How to do the interpolation?**

In order to reconstruct K samples between each two output points in a sub-sampled system, we must first upsample the digital output with a factor K (by adding K zeros between each two points). Afterwards, as shown in Figure 3.11, we pass the upsampled signal through a bandpass filter located at the original Nyquist zone of the input signal. This will recover the input signal with a sampling rate of $K \times f_s$.

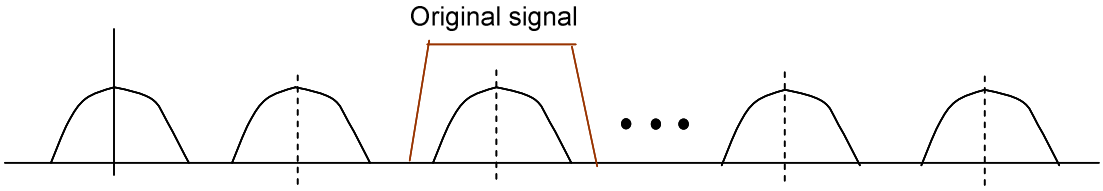

Figure 3.11: Frequency spectrum of the signal with K×fs sampling rate. The upsampled signal is passed through a bandpass filter to extract the interpolated samples.

If the upsampling factor is large enough, we should conceptually be able to estimate the signal derivative using only one previous reconstructed sample; however, due to aliasing in the interpolation process, the interpolated samples are not accurate

and using only one of them does not generate an accurate estimate of the derivative. The aliasing in the interpolation process is a result of nonlinearity in the system. Even though the input signal is bandlimited in one Nyquist zone, due to the nonlinearities in the ADC, the output signal will not be bandlimited. This causes aliasing in the upsampling process and hence the bandpass filter can not recover the main signal accurately. However, if the nonlinearities in the ADC are small, aliasing is small and a few of the interpolated samples are enough to model the derivative with sufficient accuracy. In order to compensate for errors in the interpolated samples and other approximations made in our model, we use a few of the interpolated samples with independent coefficients as shown in the following equation

$$V_{in}(k) = V_{out}(k) + \sum_{i=0}^{n} a_i {V_{out}}^{i}(k) \times \sum_{j=-m}^{m} b_j V_{out}\left(k - \frac{j}{K}\right) \tag{3.10}$$

This equation can be used with a set of input and output samples and the LS method to find all the coefficients. Due to the practical limitations with random noise input, which was discussed in previous section, sinewaves are used as the training sequence.

3.3.2.2 **Evaluation of the model**

The above correction filter was tested on the track-and-hold model (Figure 3.5) emulated in Matlab Simulink. The bandpass filter is implemented in the digital domain using an ideal rectangular filter multiplied by a Kaiser Window in the time domain. The number of taps in the filter is chosen based on the accuracy we need in the interpolated samples. Based on our simulation results, we chose an upsampling factor of 100 (K=100) and 2100 taps for the filter which resulted in 21 multiplications for finding each interpolated sample. Figure 3.12 shows the frequency response of the bandpass filter used for reconstructing the signal in the 3rd Nyquist zone with an upsampling factor of K=100. Having 2100 taps in the filter requires 21 multiplications including the current sample, 10 previous and 10 post samples for generating each of the interpolated samples.

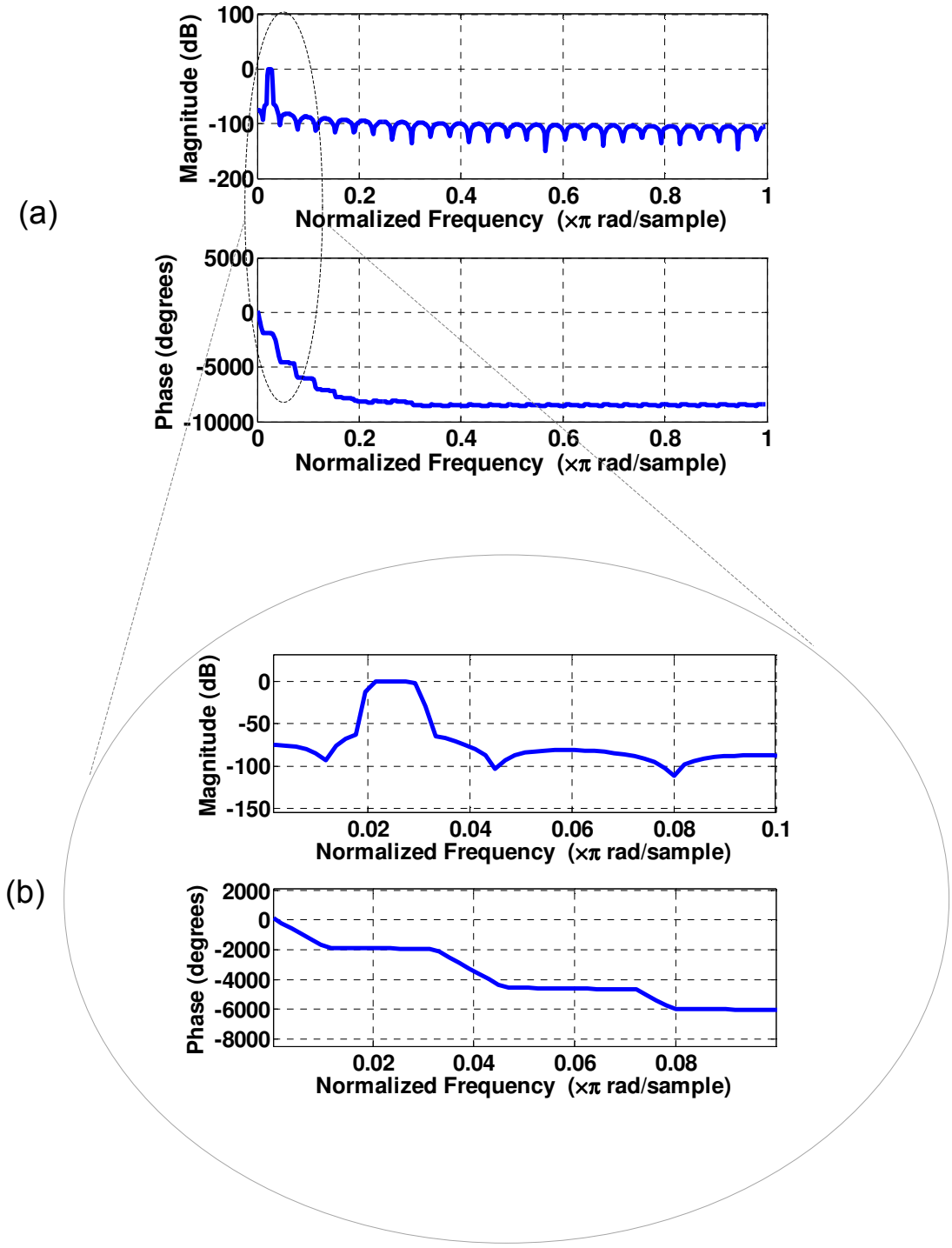

Figure 3.12: (a) Frequency response of the bandpass filter used for reconstructing interpolated samples with upsampling factor of K=100 located in the 3rd Nyquist zone. (b) Bandpass filter from the 1st to the 10th Nyquist zone

In order to find the required number of interpolated samples for proper nonlinearity compensation, the error in the corrected signal was plotted versus the number of interpolated samples used (see Figure 3.13). It is observed from this plot that the error in the corrected output reduces monotonically as we increase the number of interpolated samples; however, it does not change considerably for using more than 4 interpolated samples. Therefore, we use 4 interpolated samples together with the current sample (m=2 in (3.10)) to model the derivative and compensate the nonlinearities. The total number of coefficients in this model is therefore 15, which is significantly less than the previous model discussed in Section 3.3.1.

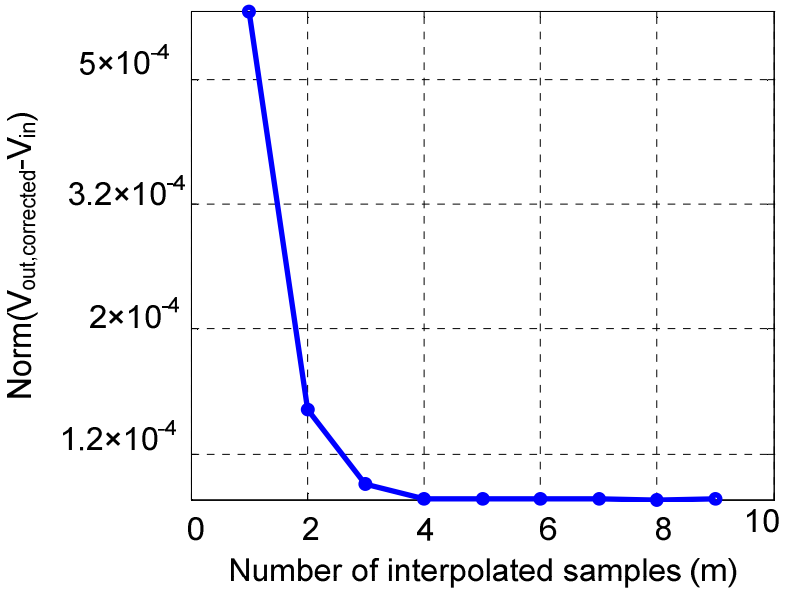

Figure 3.13: Error in the corrected output vs. number of interpolated samples used in the model.

Having fewer coefficients in the model also eases the requirement on the number of sinewaves used for coefficient calibration. We observed from our simulation results that using only three single tone sinewaves spread in each Nyquist zone is sufficient

for correction of all the bandlimited signals in that zone. Therefore, by using interpolation, the number of required calibration signals also significantly reduces compared to the previous method.

Figure 3.14 shows the SFDR plots resulted from applying this algorithm to the track-and-hold model in the first four Nyquist zones. As we can see in this figure, SFDR has improved by more than 40 dB in all frequency regions, which is better than the previous calibration approach. This improvement in the linearity is due to the more accurate model for the derivative, which helps in this system where the nonlinearity is only derivative-dependent.

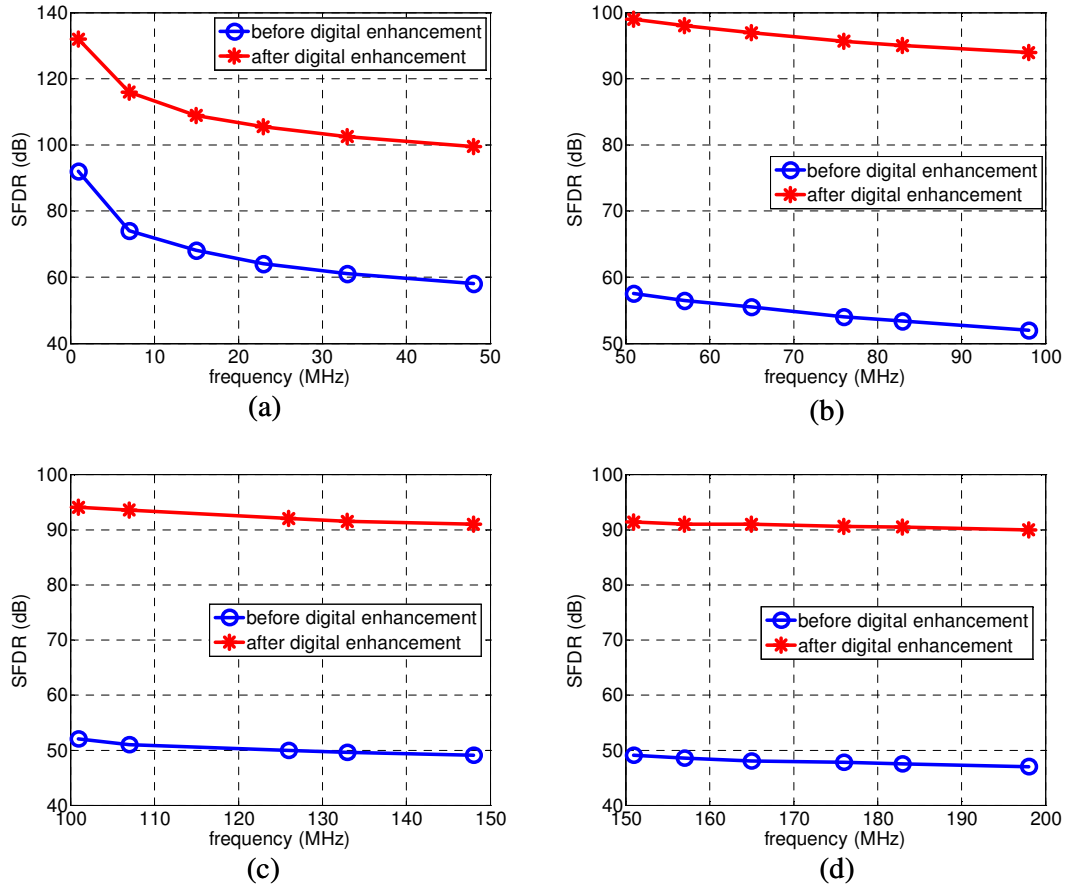

Figure 3.14: SFDR vs. input frequency in the first 4 Nyquist zones (f_{clk}=100 MHz) before and after digital enhancement, using interpolated samples in the model. (a) SFDR in the1st Nyquist zone. (b) SFDR in the 2nd Nyquist zone. (c) SFDR in the 3rd Nyquist zone. (d) SFDR in the 4th Nyquist zone.

3.3.3 Nonlinearity correction on a multi-tone input signal

As discussed in the previous sections, after calibrating the coefficients of the filter using a few single tone sinewaves in each Nyquist zone, the correction filter should be able to compensate the nonlinearities caused on any band-limited signal located in that specific frequency zone. In order to test this in simulation we applied two-tone and three-tone sinewaves as our input signal to the nonlinear system shown in Figure 3.5.

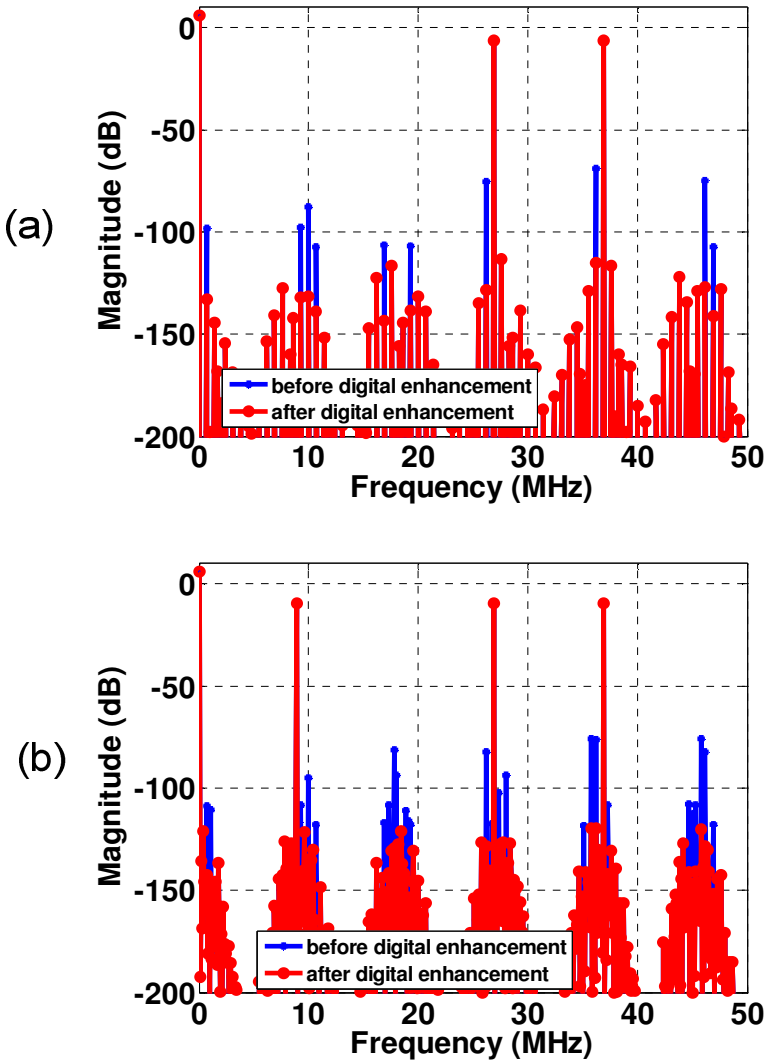

Figure 3.15: Results of applying the algorithm in simulation to the nonlinear model of the track-and-hold with two-tone or three-tone signals. (a) Input signal located at 263 MHz and 273 MHz. (b) Input signals located at 263 MHz, 273 MHz and 291 MHz.

Figure 3.15 shows the result of applying the correction filter on two-tone and three-tone signals located in the 5th Nyquist zone of the sampling system with a clock speed of 100 MHz. In order to find the filter coefficients, single tones sinewaves were used as training signals in the desired frequency range. Calibration tones were chosen at 255 MHz, 274 MHZ and 294 MHz. Figure 3.15(a) shows the output frequency spectrum of a two-tone signal located at 263 MHz and 273 MHz and Figure 3.15(b) shows the frequency spectrum on a three-tone signal located at 263 MHz, 273 MHz and 291 MHz. As we can see in these figures, all 2nd and 3rd order harmonics and inter-modulation terms caused by 2nd and 3rd order nonlinearities are significantly reduced after applying the correction filter.

Alternatively, the compensation filter can be applied on any arbitrary bandlimited signal located in this Nyquist zone. Figure 3.16 shows the result of applying the correction filter on the nonlinear output including nine single-tone sinewaves spread in the 5th Nyquist zone. As we can see in this figure all the tones caused by nonlinearity in this system are cancelled after applying the correction filter.

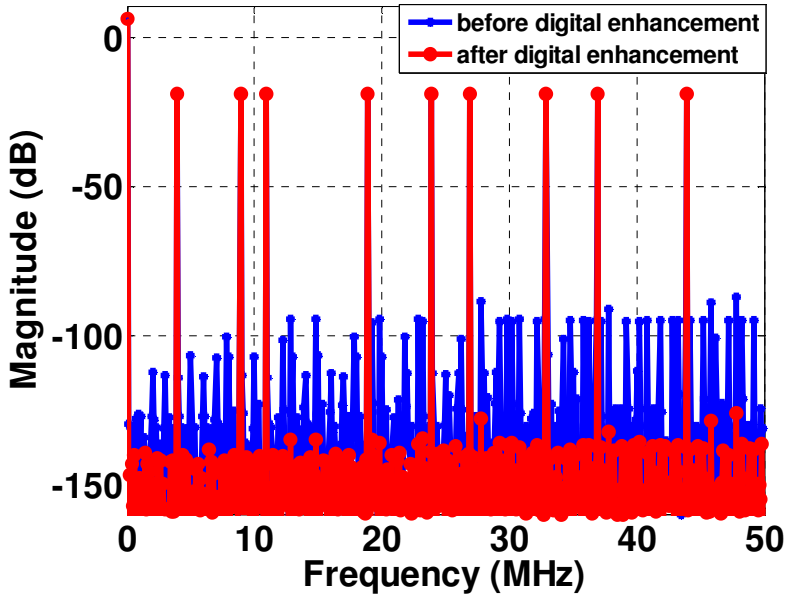

Figure 3.16: Applying correction algorithm on an arbitrary signal including nine single tone sinewaves located at the 5th Nyquist zone.

3.3.4 **Algorithm performance in presence of noise**

In order to evaluate the calibration algorithm in the presence of noise and other ADC non-idealities, we simulated the track-and hold model by adding thermal and quantization noise to its output. Assuming the ADC core quantizes the THA output with a resolution of 14 bits and generates outputs with 11 ENOB (Effective Number Of Bits), we passed the output samples through a 14-bit quantizer and added 2^3 LSB of random noise to its output. This causes the last three bits of the ADC's output to be corrupted by noise. Figure 3.17 shows the frequency spectrum of the output before and after digital enhancement at the input frequency of f_{in}=263 MHz and f_{clk}=100 MHz. As we can see in these plots, the 2nd and 3rd harmonics of the nonlinear system are significantly reduced after applying the digital correction algorithm.

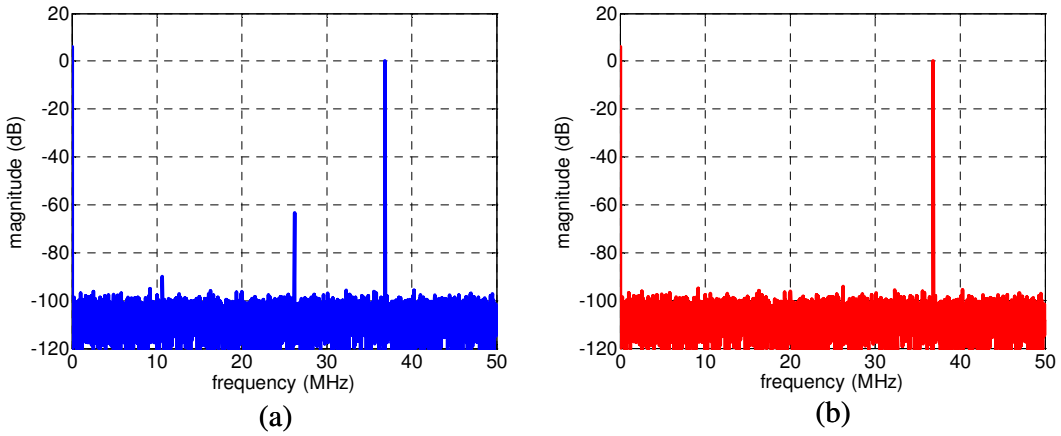

Figure 3.17: Frequency spectrum at the track-and-hold output with quantization noise and 8 LSB of random noise before and after digital correction (f_{in}=263 MHz, f_{clk}=100 MHz). (a) Before digital correction. (b) After digital correction.

The results of this experiment shows that the algorithm can still be used to correct nonlinearities on an output signal that has been corrupted by thermal and quantization noise. The noise on the samples can cause large errors in derivative estimation in each sample point; but, if a large number of sample points are used for filter coefficients calibration, the effect of noise is averaged out and correct coefficients are extracted during calibration process.

3.3.5 Algorithm performance in presence of other circuit non-idealities

It was shown in the previous chapter that other parts of the ADC front-end including input driving circuit and package parasitics add to the nonlinearities in the system. These circuit nonidealities basically change the distortion model from our initial nonlinear model shown in equation (2.8) and introduce dependency to second and third order derivatives but do not change the model from the general form shown in equation (2.10). Now, considering the fact that our correction filter is also based on equation (2.10) and separation of nonlinear effects from memory effects, we can still

use our correction filter (equation (3.6)) and apply it to the output of the ADC front-end circuit shown in Figure 2.19.

The general form of the nonlinear model at the front-end of ADC can be expressed by the following equation

$$V_{in}(k) = V_{out}(k) + f_{1\_nonlin}(V_{out}(k))\frac{dV_{out}(k)}{dt} + f_{2\_nonlin}(V_{out}(k))\frac{d^2V_{out}(k)}{dt^2} + \ldots \qquad (3.11)$$

It is shown in Appendix B that all derivatives of a sampled signal can be modeled as a linear combination of previous and post samples. Therefore, the above distortion function can still be modeled as a product of a static nonlinear function and a linear function with memory (as shown in equation (2.10)) and hence is still compatible with the initial form of our distortion model.

As we saw in the previous chapter, the above differential equation causes transient response at the output depending on the initial stored voltages and the input signal. If this transient response has vanished by the time of the sampling, the distortion function in the above equation is compatible with the general form of our nonlinear function and can be corrected with the correction filter shown in equation (3.6). However, if the transient response is not completely gone by the time of the sampling it will add errors to the system depending on the previous sampled voltage. This error is mainly caused by the residual charge on the sampling capacitor and is a problem in flip-around track-and-hold amplifiers where the capacitor is not reset between consecutive samples. In order to take this error into account, we modify the correction filter and add terms including the previous sampled value. Therefore, the new correction filter for these systems can be expressed by the following equation

$$D_{out,corrected} = D_{out}(k) + \sum_{i=1}^{K} {D_{out}}^{i}(k) \times \left[\sum_{j=0}^{2L} h_{ij} D_{out}(k - \frac{j-L}{M}) + g_i D_{out}(k-1) \right] \qquad (3.12)$$

This new correction filter helps correct for the errors caused by the charge glitches generated from the residual capacitor charge going back to the input circuitry.

We simulated the complete ADC front-end (Figure 3.18) in Spice and then applied the correction algorithm to its output in order to study how the algorithm performs in the presence of other circuit nonidealities. Figure 3.19 shows the frequency spectrum at the output of this stage before and after applying the digital enhancement. The results are shown at two input frequencies of 326 MHz and 898 MHz. It can be observed in these figures that the 2nd and 3rd order harmonics are significantly reduced after applying the correction algorithm. This clearly shows that the algorithm works in presence of second-order circuit nonidealities. Also Figure 3.18(b) shows that the algorithm works even at very high input frequencies and does not have any frequency limitation. This is a great advantage of digital post processing approaches when compared to analog linearity improvement techniques. The digital correction algorithm can correct nonlinearities as long as the sampled voltage can completely track the input signal and hence, the method works for the whole input frequency bandwidth of the ADC's front-end.

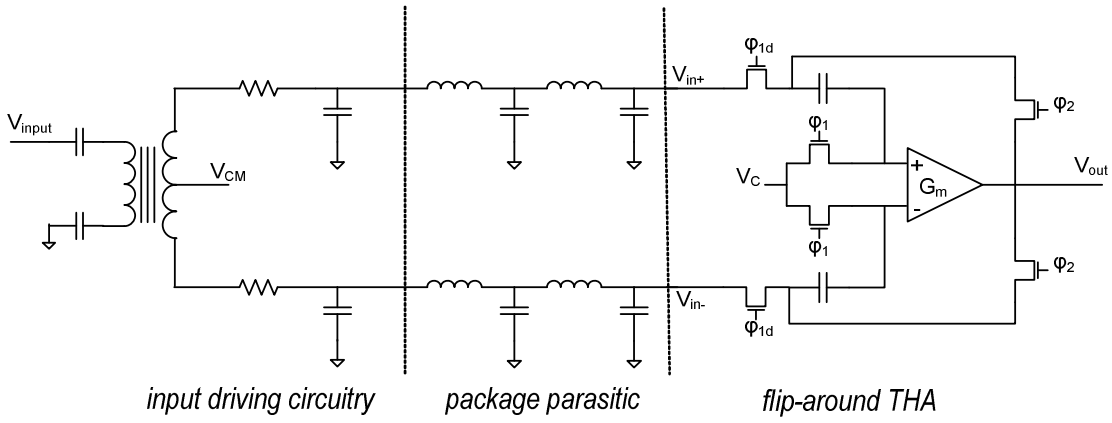

Figure 3.18: Circuit diagram of the ADC front-end including flip-around track-and-hold amplifier, package parasitics and the input driving circuit including transformer.

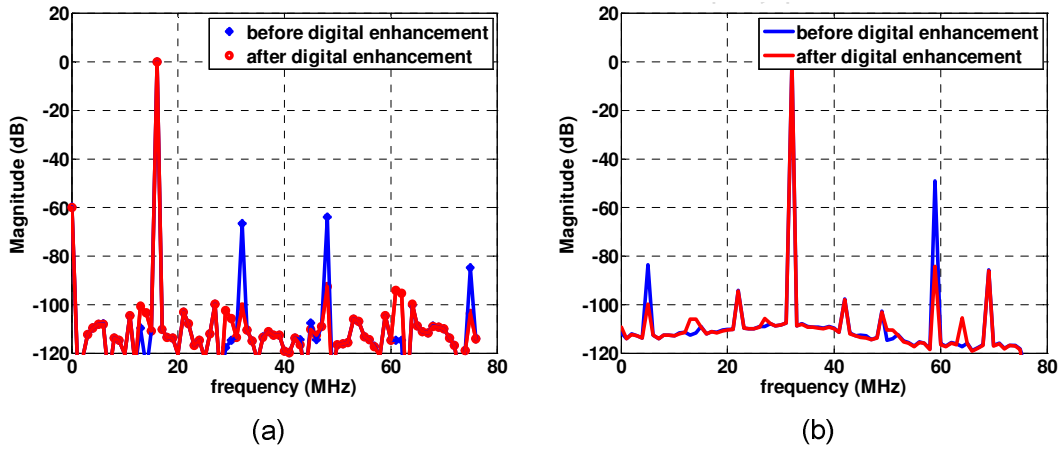

Figure 3.19: Frequency spectrum at the output of the ADC front-end simulation before and after digital enhancement. (a) f_{in}=326 MHz. (b)f_{in}=898 MHz.

3.4 Summary

It was shown in Chapter 2 that linearity of ADCs at high input frequencies are limited by the dynamic errors generated at their front-end. Several studies have been done to minimize those errors or compensate them in the digital domain. Analog approaches usually need using of expensive processes and also suffer from the bandwidth limitations in the active circuitry and degrade the performance at high frequencies. Digital methods to date are mostly based on Volterra series or using look-up tables which both get very complicated for high resolution ADCs and will not be practical even in today's fine line digital technologies.

In this chapter we proposed a digital correction algorithm for compensating dynamic nonlinear errors generated at the front-end of ADCs and to achieve good linearity performance at high input frequencies. The method uses circuit level information based on the models developed in Chapter 2 to simplify the correction process. Several approaches for coefficient calibration and correction were investigated here and some of their practical issues were discussed. In the final

proposed approach, three single-tone sinewaves are used as test tones for calibrating the coefficients and interpolation is used for estimating the derivative of the signal.

It was shown in this chapter that the proposed algorithm can be used to correct nonlinearities in any bandlimited signal located at a specific Nyquist zone. The algorithm was also tested in simulation on an output signal which was corrupted by thermal and quantization noise; but still showed good performance. It was also shown that the algorithm can work in presence of other circuit nonidealities by applying the correction scheme to the Spice simulation result of the complete ADC front-end circuit.

Chapter 4

Experimental Results

In the previous chapter we saw in simulation how the proposed correction algorithm operates effectively on the nonlinear model of a track-and-hold system and also functions in presence of other circuit nonidealities. In order to evaluate the performance of the algorithm on real ADCs we applied the correction scheme to a state-of-the-art, commercially available ADC. For this experiment we used the National ADC14155, which is a 14-bit ADC with a sample rate of 155 Msample/s and 1.1 GHz input bandwidth [11]. Figure 4.1 shows a photo of the ADC test board together with the data capture board.

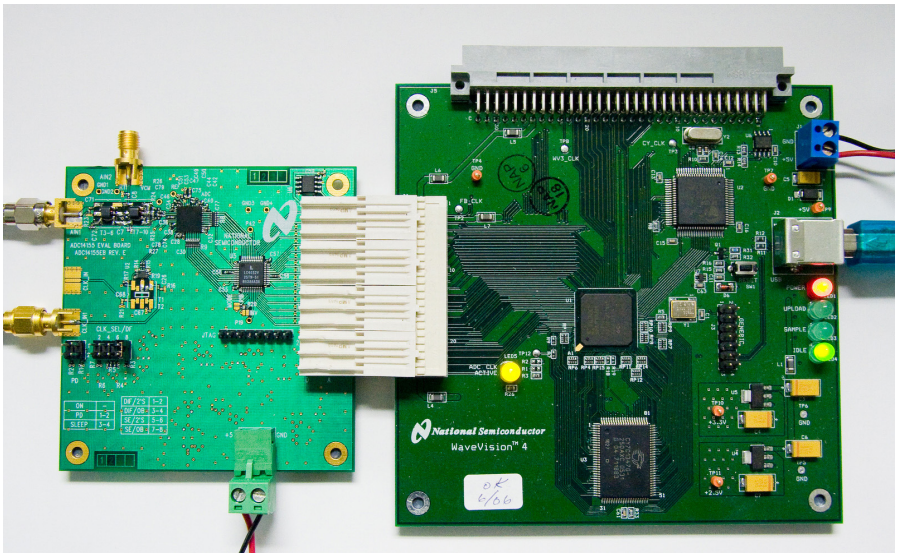

Figure 4.1: Photo of the ADC test setup with the data capture board.

4.1 ADC test setup

A schematic of the ADC together with its input driving circuit and the rest of our test setup is shown in Figure 4.2. HP 8644B signal generators are used for generating the sinewaves required for input and clock signals. The generated sinewave for the input goes to a tunable bandpass filter to remove all the harmonics and make the input signal linear enough for testing the linearity of the ADC. The input driving circuit of the ADC uses two cascaded balun type transformers for converting single-ended to differential signals. Balun transformers are more linear than regular flux coupled transformers at high input frequencies and using two of them in series minimizes the phase and magnitude imbalance at the two output ports at high frequencies [41].

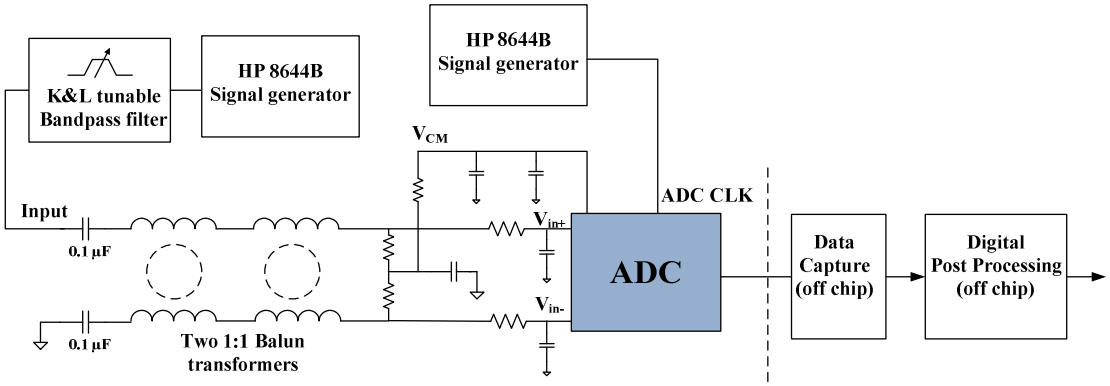

Figure 4.2: Block diagram of the test setup.

The ADC output goes to a data capture board which collects blocks of data that are transferred to a computer that emulates the digital post-processing. Alternatively the digital scheme could be implemented in real time for example using an FPGA. However, since there is no feedback from the digital post processing to the ADC, a real-time implementation would not alter the presented results. In Section 4.2, we look at the block diagram and implementation of the digital post-processing using Verilog and synthesis tools. Measurement results from this test setup are discussed in Section 4.3.

4.2 Implementation of the digital post processing

4.2.1 Digital correction scheme

It is crucial for our system to know how complex the digital post processing is and how much power it would add to the system if implemented in hardware. Figure 4.3 shows a block diagram of our digital correction scheme. As discussed in Chapter 3, we use 10 previous and 10 post samples for reconstructing the interpolated samples using the bandpass filter. Filter coefficients are stored in registers to be used in the interpolation process. We chose an upsampling factor of 100 for accurate modeling of the derivative. However, we only need to reconstruct four of these samples for the nonlinearity correction process. These four reconstructed samples, together with one previous sample, are transferred to the nonlinearity correction section and are multiplied with the powers of the current sample and calibrated coefficients in order to find the final corrected output.

In this structure, the number of bits required for storing each sample is the same as number of bits in the ADC (14 in our experiment). We used 16 bits for storing the calibrated coefficients to get enough accuracy in our correction. After each operation, the result is rounded to a 14-bit number. In order to find the number of operations in the proposed correction scheme, we note that the interpolation process with a 2100-tap filter, requires 21 multiplications and additions for generating each of the reconstructed samples. Since we need to reconstruct four of the interpolated samples for each main sample, we need 21×4=84 multiplications and 21×4=84 additions in the interpolation section. After these samples are generated, they are transferred to the next section, which performs the nonlinearity correction. Due to having large harmonics until 3rd order in the system, nonlinearity was modeled until 3rd order in our correction filter. The total number of multiplications in this section is 6+6×2+6×2=30 and the total number of additions is 6+6+6=18. Therefore, in total we need to perform 114 multiplications and 102 additions for finding each corrected sample. These results show that the major part of the digital processing power is consumed in the interpolation process. This power can be reduced by using fewer taps in the filter and

hence fewer operations during sample reconstruction. However, this will lead to losing some accuracy and therefore linearity improvement in the system.

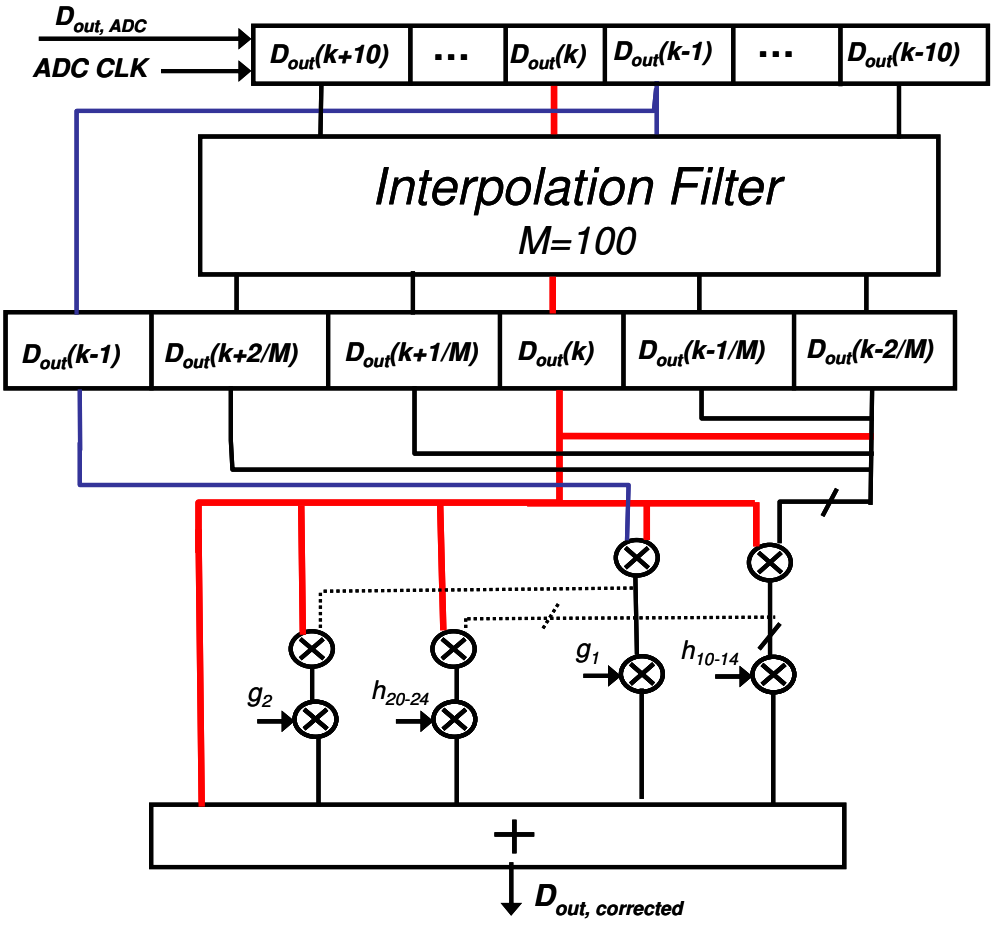

Figure 4.3: Block diagram of the digital post processor.

We implemented this digital algorithm in Verilog and synthesized it using standard CMOS cells in a 90-nm process. Table 4.1 shows a summary of the numbers resulting from the synthesis. As we can see in the table, the digital post-processor works at the same speed as the ADC with a clock rate of 155 MSample/s. The correction scheme adds 33 cycles of latency to the output in order to perform the post processing. The digital post-processing is implemented using 61,339 logic cells which occupy an area of 0.54 mm^2. The power consumption of this block was estimated from

the synthesis tool to be about 52 mW, which is a small overhead compared to the ADC power of 967 mW.

Table 4.1: Results of synthesizing the digital post processing scheme.

Technology	90-nm CMOS
Clock speed	155 MHz
Latency	33 clock cycles
Number of logic cells	61,339
Area	0.54 mm^2
Power	52 mW
ADC power (ADC14155)	967 mW

4.2.2 **Coefficient calibration**

An important part of the digital post processing is the filter coefficient calibration. The proposed calibration method here is a foreground digital calibration, meaning that the coefficients are calibrated once and are kept constant during the ADC operation. This coefficient calibration can be done at the power-up state of the ADC when the normal operation has not started yet. Since this process is only applied once, it does not add to the power of the system and it can also reuse some of the digital blocks. Therefore, its power and area consumption is not a major concern in our system. Although it is still important to find out how the calibration can be done using the digital hardware.

Coefficients of the correction filter are extracted using least square (LS) solutions on a set of test input and output points. These solutions are found through finding matrix inverses in Matlab. But, for hardware implementation we need to find an adaptive algorithm to extract the coefficients iteratively by updating them in each step of sampling. On the other hand, because of having several frequency tones in our test

signals, the input samples are not stationary and Least Mean Square (LMS) methods can not be used for finding the coefficients. Instead, Recursive Least Square (RLS) algorithms can be used to find the best solution for coefficients [53], [54]. The advantage of RLS algorithm is that it uses some information from previous samples in updating coefficients at each step. The RLS algorithm for finding the filter coefficients is explained in detail in Appendix C.

4.3 Measurement results

As discussed above, coefficients of the post-processor are extracted in a foreground calibration approach using three single tone sinewaves, distributed across each Nyquist zone. It was shown in Chapter 3 that different sets of coefficients must be extracted for each Nyquist zone; however with each set, any band-limited signal in the corresponding frequency range can be corrected.

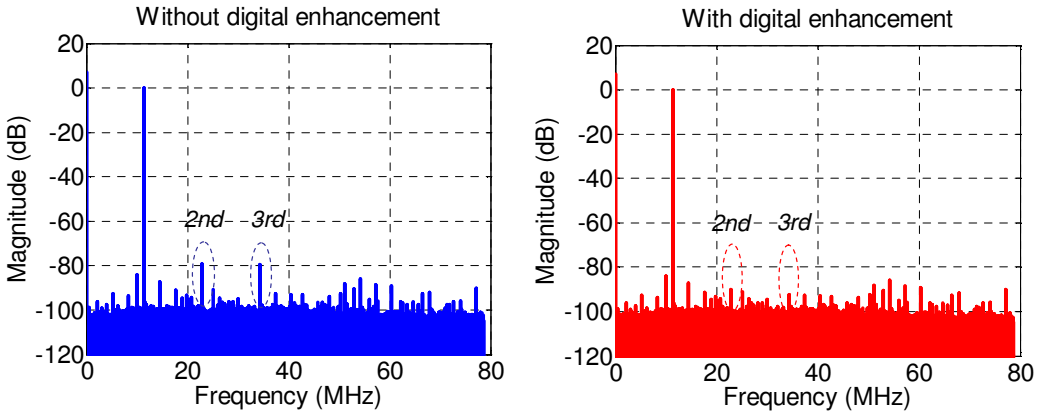

Figure 4.4: Frequency spectrum of the ADC output at f_{in}=326 MHz (f_{clk}=155 MHz).

Figure 4.4 shows the frequency spectrum at the output of the ADC at f_{in}=326 MHz. The results are shown before and after applying the digital enhancement scheme. The input frequency is located in the 5th Nyquist zone of the ADC and in order to find the filter coefficients, three single tone sinewaves were used at the same Nyquist zone located at frequencies of 322 MHz, 340 MHz and 390 MHz. Having a

3rd order nonlinear correction filter in our system, it can be observed from this figure that 2nd and 3rd order harmonics are significantly reduced after applying the correction scheme.

Figure 4.5 shows SFDR of the ADC versus input frequency in the 5th and 6th Nyquist zone before and after applying the digital enhancement. The correction filter was extracted individually for each of these Nyquist zones by using training signals in that specific frequency region. It is observed from these figures that SFDR has improved to more than 83 dB after applying the digital correction algorithm up to input frequencies of 470 MHz. The maximum achievable SFDR in this system is partly limited by noise on the samples and also the static nonlinearity in the ADC core after canceling dynamic errors at its front-end. In principle, some part of the static nonlinearity caused in the ADC core can be cancelled by our correction filter. However, since nonlinearities in the ADC core can be as high as 11th or 13th order, it would make our nonlinear filter prohibitively complex. If higher SFDR values are desired, there are simpler methods for canceling static nonlinearities in the ADC core [4], [5].

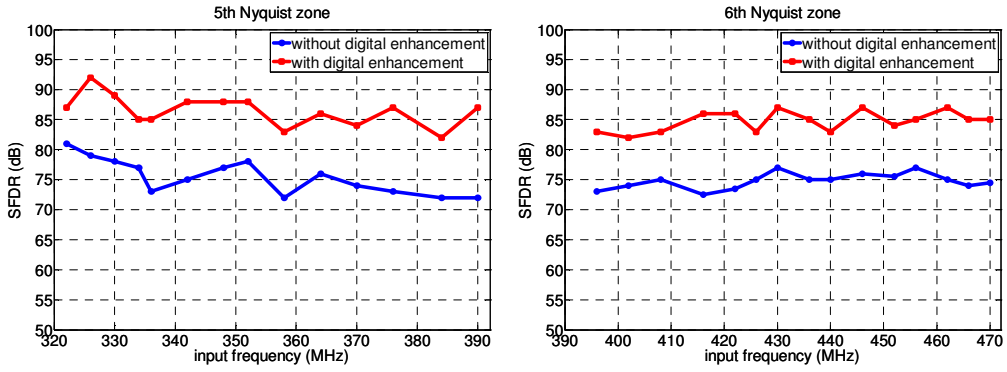

Figure 4.5: SFDR vs. input frequency in the 5th and 6th Nyquist zone (f_{clk}=155 MHz).

4.3.1 **Robustness of the algorithm with temperature variations**

Since the proposed correction algorithm performs a foreground calibration, it is important to test its robustness versus variations in environmental conditions. In order to test the robustness of the algorithm versus temperature, we calibrated the ADC once while it was working in its normal condition. After that, we subjected the ADC to temperature variations of ±30° C relative to room temperature and applied the same correction filter to its output.

For the temperature variation test, the ADC was heated with a heat gun and cooled down with a cooling spray. In order to be able to measure temperature variations, we used one of the ESD diodes on the chip and forward biased it according to the diagram shown in Figure 4.6. In this setup, pin PD (which is an unused pin in our experiment) is used for connecting the ESD diode to a negative voltage $-V_c$ through a 10K resistor. This makes the ESD diode to be forward biased with a voltage equal to V_{be} across its two terminals. V_{be} of the diode changes directly with temperature and has a slope of -1.8 mV per degree of temperature increase. Therefore, after each step of heating or cooling down process, temperature can be measured by measuring the voltage at node PD of the ADC package.

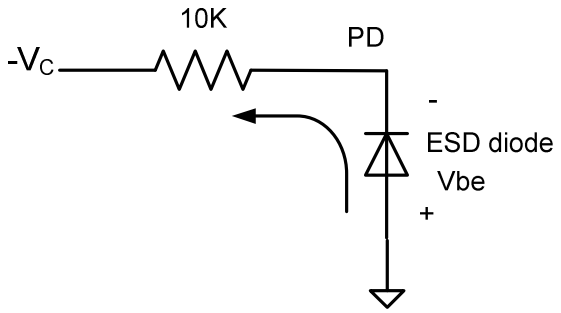

Figure 4.6: Block diagram of the test setup used for temperature variation measurement. PD is an unused pin on the chip and is connected to a negative voltage through a resistor in order to forward bias the ESD diode.

Figure 4.7 shows the SFDR of the tested ADC versus temperature variation of ±30°C relative to room temperature, before and after applying digital enhancement. As evident from this plot, the distortion compensation is only weakly affected by temperature and the SFDR of the post-processed converter remains above 80 dB. This result indicates that a simple foreground calibration approach is feasible in some applications that do not experience large temperature variations without providing a time window for re-calibration. Similar to the scheme implemented in [55], a foreground re-calibration could be triggered by a temperature sensor upon detection of a large change in operating temperature.

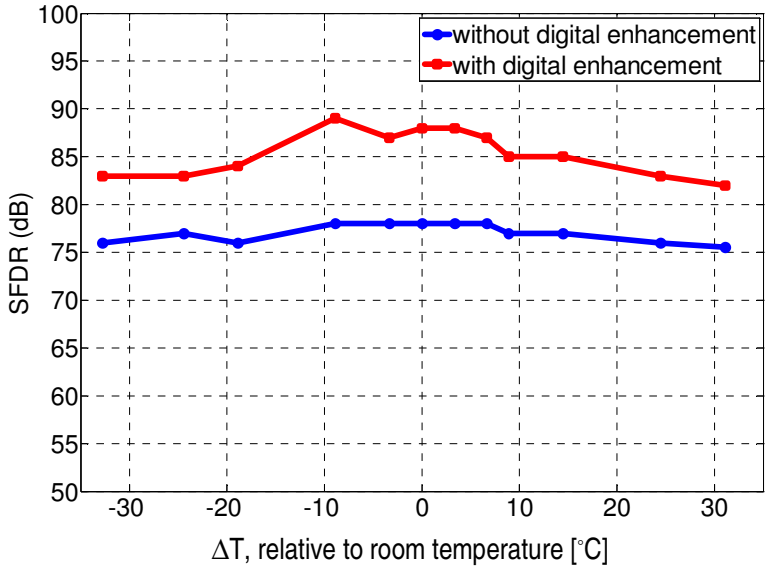

Figure 4.7: SFDR of the ADC vs. temperature variations (f_{in}=326 MHz, f_{clk}=155 MHz).

4.3.1.1 **Analysis of the impact of temperature variation on dynamic nonlinearity**

We saw in Chapter 2 that dynamic nonlinearity at the ADC's front-end is mainly generated by nonlinear switch resistance, and in the case of a bootstrapped switch,

nonlinearity comes from the backgate effect in the threshold voltage of the transistor. Both the threshold voltage and carrier mobility in a device change considerably with temperature variations. These effects have been widely investigated in the literature in order to make voltage references independent of temperature [56]-[58]. The variations in these two parameters change the switch resistance and its nonlinear factor and hence can change the nonlinearity in the system.

Threshold voltage mainly changes linearly with temperature; however, mobility has small variations with temperature which are nonlinear. These two effects can be approximately modeled with the following equations.

$$V_{th}(T) = V_{th}(T_0) + \alpha_{V_{th}}(T - T_0) \tag{4.1}$$

$$\mu_n(T) = \mu_n(T_0)\left(\frac{T}{T_0}\right)^{\alpha_\mu} \tag{4.2}$$

If we model $R(V_{in})$ as a polynomial function of the input voltage shown below, then each of the coefficients in this model (R_1, R_2, R_3, …) are a function of the temperature due to the variation of threshold voltage and mobility shown above.

$$R(V_{in}) = R_0 + R_1 V_{in} + R_2 V_{in}^{\ 2} + R_3 V_{in}^{\ 3} + \ldots \tag{4.3}$$

We included the temperature variation of V_{th} and μ_n into our simulation model of the nonlinear switch resistance and looked at the variation of the coefficients in the above polynomial vs. temperature. Figure 4.8 shows the result of this simulation. In this figure, the constant term in R and its first order and second order input dependency are plotted versus temperature variations.

Due to the differential architecture of the input track-and-hold circuit, the main nonlinearity caused by this stage is the third order nonlinearity. By looking at (2.8) in Chapter 2, we note that the third order nonlinearity in the system comes from the second order input dependency in the switch resistance. The variations of different

resistance terms versus temperature in Figure 4.8 show that the second order nonlinear term in R (R_2) does not change significantly in the temperature range of our experiment. Therefore, the dominant harmonic at the output (which is the third-order harmonic), and hence SFDR, show small variations with temperature.

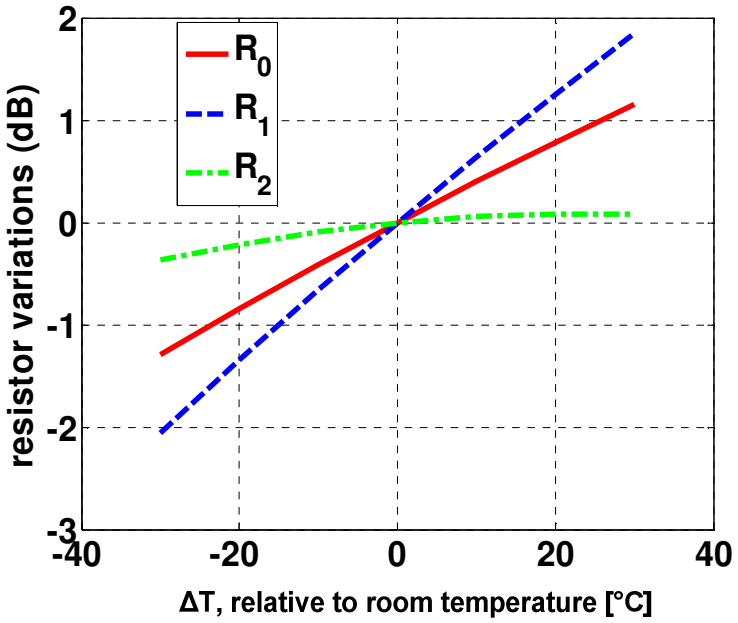

Figure 4.8: Variations in switch resistance nonlinear terms vs. temperature.

4.3.2 General applicability of the algorithm

In the above sub-sections, we showed the experimental results obtained by applying the correction method to the output of National ADC14155 [11]. As a further test of robustness and general applicability of our algorithm, we applied the post-processing scheme to a second ADC provided by a different vendor. The device chosen for this experiment was TI ADS6143 which is a 14-bit, 80-MS/s CMOS ADC with an input bandwidth of 450 MHz [59].

The correction filter was again extracted by using three single-tone sinewaves as training signals in each Nyquist zone. As in the previous experiment, four of the reconstructed samples together with one previous main sample were used in the correction filter. Figure 4.9 shows the SFDR of this ADC in its 11th Nyquist zone which extends to input frequencies of 440 MHz. The SFDR results are shown before and after digital enhancement. As can be observed in this figure, SFDR has been improved to more than 82 dB after applying the correction.

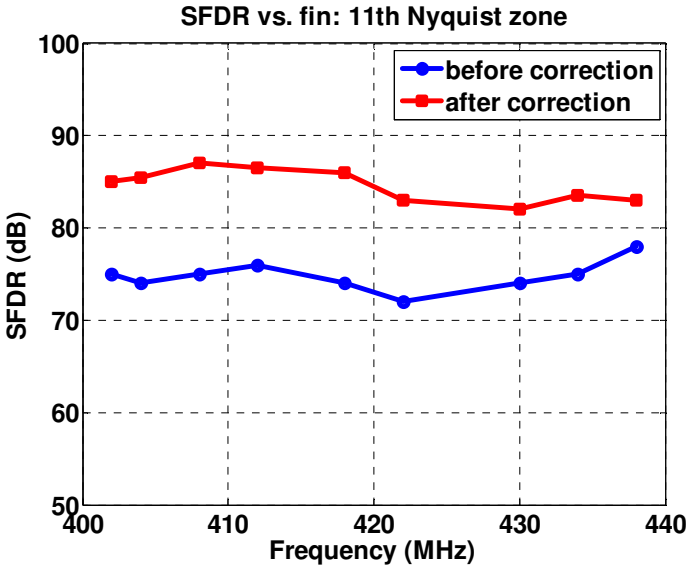

Figure 4.9: SFDR vs. input frequency in the 11th Nyquist zone for the ADS6143.

4.3.3 **Algorithm performance on different input driving circuits**

It was discussed in Chapter 2 that the transformer and other part of the input driving circuit can add more distortion to the system. The input driving circuit of the ADC is usually optimized for the frequency range of converter operation in order to introduce the least distortion into the signal [60].

Our measurement experiments shown for the National ADC14155 were done using its high frequency setup for the input circuit (see Figure 4.5). In order to test the performance of the algorithm when more distortion is caused at the input driving

circuit, we made another test setup for the ADC using its recommended input circuit for low frequency tests as shown in Figure 4.10 [60] and applied the digital correction scheme to its output. The result of this experiment is shown in Figure 4.11. This plot shows SFDR of the ADC in its 5th Nyquist zone before and after applying the digital enhancement. By comparing this figure with Figure 4.5, we note that the SFDR numbers before distortion have significantly degraded due to the different input driving circuit. However, we are still able to correct it to around 80 dB after applying the digital enhancement scheme.

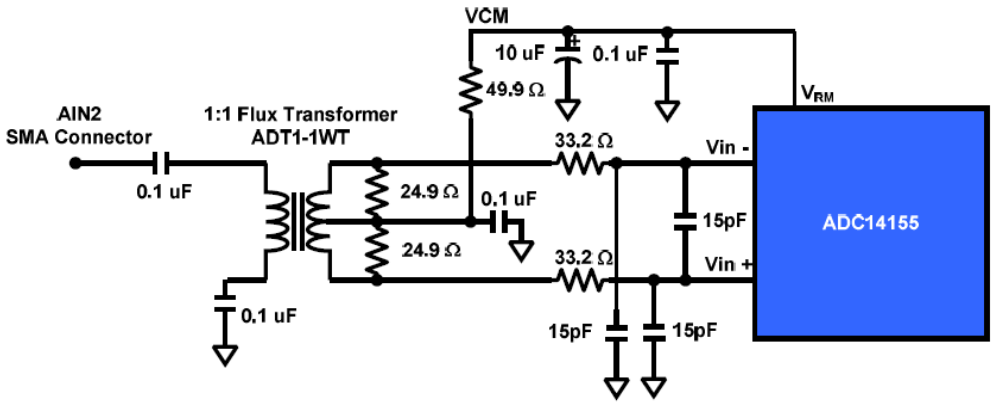

Figure 4.10: Block diagram of the ADC with the input driving circuit optimized for low frequency tests [60].

In order to test the performance of the algorithm on a different input driving circuit including amplifiers, we tried the correction scheme on the National ADC14155 together with a Digitally Variable Gain Amplifier (DVGA) at its front-end [61]. DVGAs are used in some communication applications in order to increase the dynamic range of the system (see Figure 4.12). These stages usually add distortion to the system; but their nonlinearity is mostly static and as discussed in Chapter 2, most of it can be corrected with our algorithm.

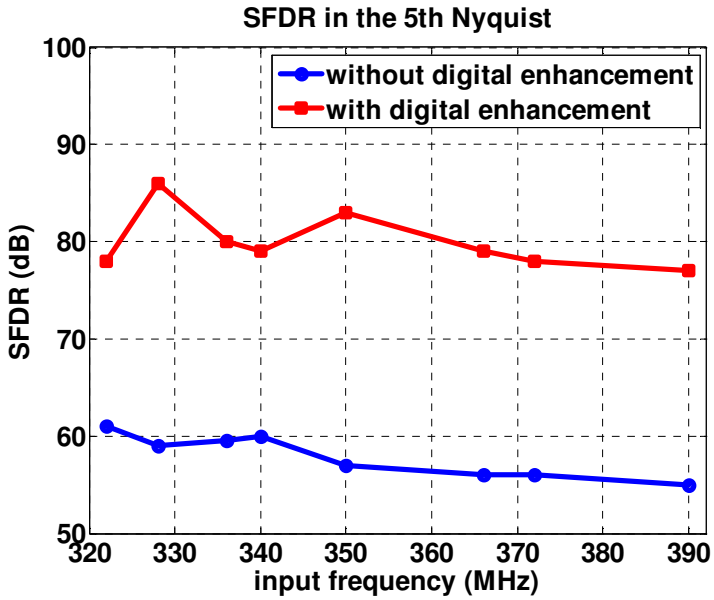

Figure 4.11: SFDR of the National ADC14155 vs. input frequency in the 5th Nyquist zone. The input circuit has been modified to cause more distortion.

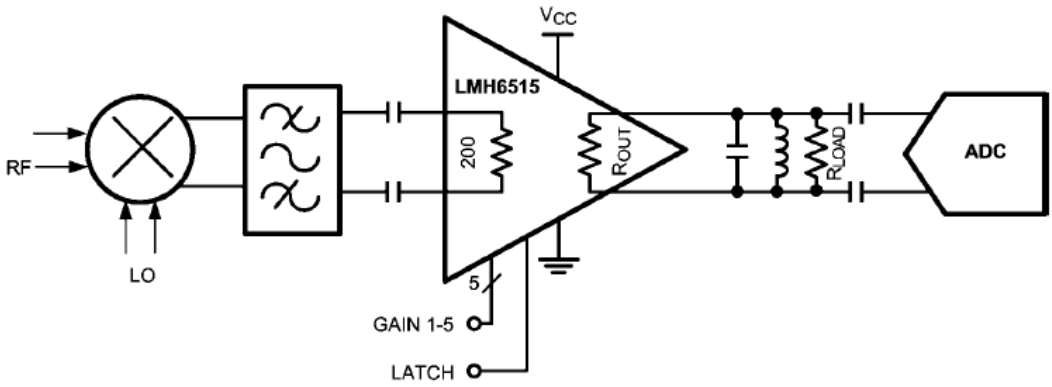

Figure 4.12: Block diagram of the ADC together with a DVGA [61].

Figure 4.13 shows the result of applying our correction algorithm to the output of the ADC together with DVGA. This plot shows SFDR of the ADC versus input frequency in the 5th Nyquist zone. It can be observed from this figure that SFDR of the system has been degraded significantly because of the DVGA. However, the digital

correction algorithm is still able to compensate most of the nonlinearities caused by that stage and improve the SFDR to around 80 dB. As discussed in Chapter 2 the static nonlinearities in the amplifier generate some terms in the overall distortion model of the system which are not included in our correction filter. However, these terms are a small portion of system nonlinearity and still most of the distortion elements can be cancelled with our compensation algorithm.

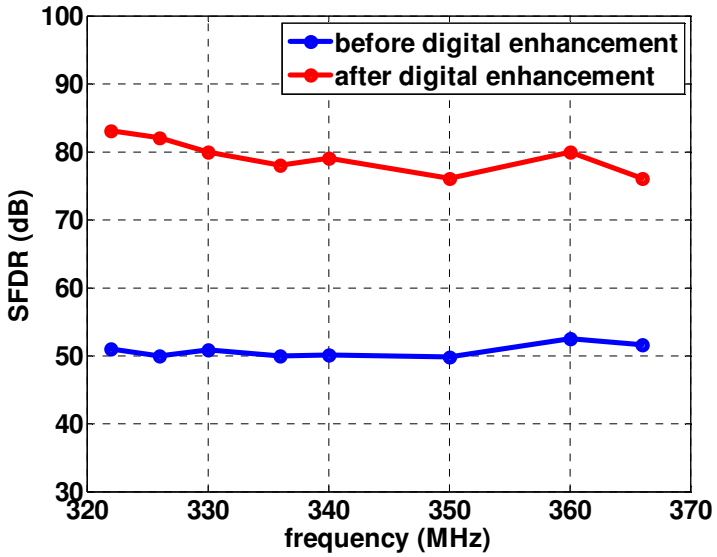

Figure 4.13: SFDR of the ADC together with DVGA in the 5th Nyquist zone.

4.4 Summary

Experimental results from applying the proposed algorithm to commercially available ADCs were shown in this chapter. It was shown that the correction scheme can significantly improve SFDR performance of the ADC up to very high input frequencies. The algorithm can be applied to any subsampling ADC that suffers from nonlinearity at high input frequencies and it does not have any frequency limitations. We also tested the robustness of the algorithm versus temperature and observed that

the correction is relatively insensitive to temperature changes. The correction method was also tested on different input driving circuits and showed good performance in presence of other sources of nonlinearities coming from the input network.

Figure 4.14 shows SFDR data of state-of-the-art high resolution ADCs (published or commercially available converters) reported at different frequencies together with the result of this work at the highest frequency in our measurement (470 MHz). As we can see in this figure the result of our work exceeds the performance of all the ADCs included in the survey. It is also observed in this figure that all the ADCs with good linearity performance use BiCMOS structures. This clearly shows that digital post-processing approaches can improve performance of pure CMOS ADCs beyond BiCMOS structures.

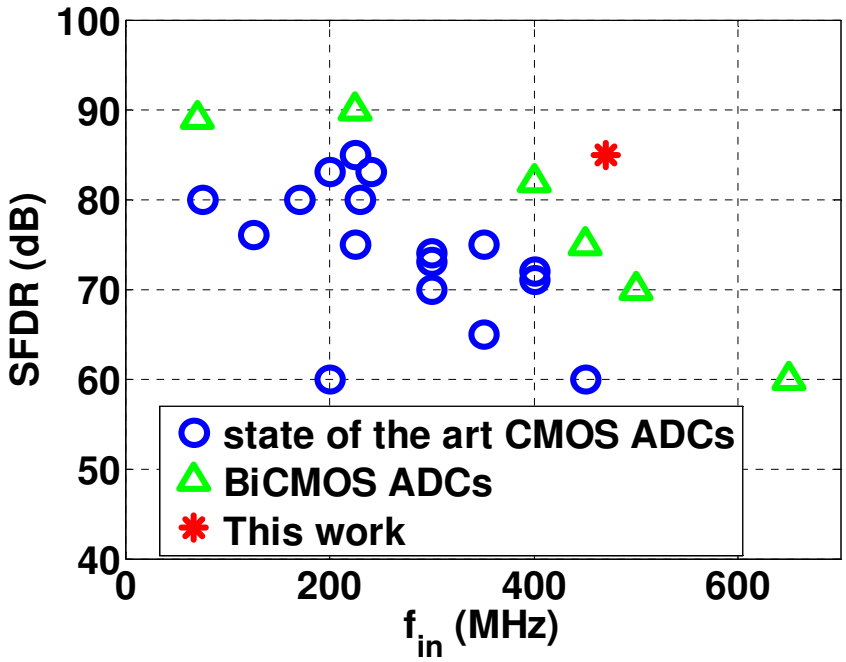

Figure 4.14: A comparison between SFDR performance of the state-of-the-art high resolution ADCs and the result of this work.

Chapter 5

Impact of Substrate Noise on the Performance of High-Speed ADCs

In the previous chapters we saw that digital post-processing can be used to compensate dynamic nonlinearities at the front-end of ADCs and improve their linearity at high input frequencies. In the experimental results shown in Chapter 4, the digital data from the ADC output were transferred to a computer which emulated the digital post processing. In a real application though, this algorithm needs to be implemented real time by either using an FPGA (which may already be a part of the system) or using digital blocks integrated on the same chip as the ADC for stand-alone or system-on-chip applications of the converter.

Integration of both analog and digital blocks on the same die can introduce a significant source of noise for analog components through the substrate [62]. This noise, which is generated due to the switching activities in the digital components can couple to the analog blocks and degrade their performance. Figure 5.1 shows a block diagram of noise injection and propagation in the substrate. It is important to investigate how this noise can impact the performance of sensitive analog parts of the system and how it can be minimized. Although substrate noise has been considered and analyzed widely in single devices or circuit blocks [63]-[65], there is no comprehensive work about its impact on a complex structure like an ADC.

Flash ADCs are among the highest speed data converters that can be used either standalone in high-speed, low-resolution applications or as a sub-circuit in higher resolution architectures [66], [67]. Therefore, we chose this architecture to investigate

its performance degradation in the presence of noise and analyze its different sub-blocks to find out how each of them can be affected by coupling noise in the substrate. In this chapter, we first describe the architecture of the designed flash ADC, then analyze different effects of substrate noise on its sub-blocks and discuss the impact of noise on the overall ADC performance. Experimental results from the tested ADC are presented at the end.

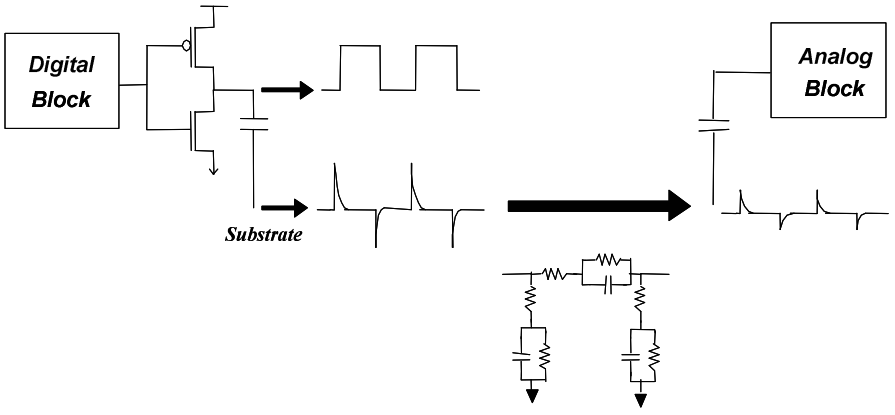

Figure 5.1: Noise injection and propagation in the substrate.

5.1 ADC architecture

In order to investigate the impact of substrate noise on high-speed ADCs, we chose a 4-bit flash ADC as our test prototype. The ADC uses 15 comparators and works at a clock rate of 1 GHz. Figure 5.2 shows a block diagram of this ADC. The input full-scale range is 0.5 V and a differential resistive ladder is used to generate the reference voltages for the comparators. Due to the low resolution of the ADC, no separate track-and-hold is used and sampling of the input voltage is done within each comparator.

Figure 5.3 shows a schematic of the comparator used in our test setup. During clock phase φ_1, switches S_3-S_7 are closed and the comparator is in the reset mode. Reference voltages are sampled on the capacitors in this phase. In clock phase φ_2

switches S_3-S_6 open and S_1, S_2 close to sample the input voltage. Switch S_7 is opened with a short delay to start the regeneration mode and amplify the difference between input and reference voltages [68].

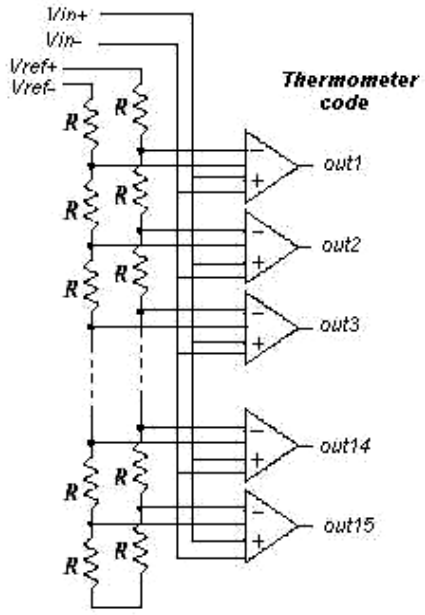

Figure 5.2: Block diagram of the flash ADC.

Bipolar transistors at the input are used together with input offset cancellation to achieve a small input offset of 4 mV. Having a full range of 0.5 V, the LSB of the ADC is 31.3 mV and hence the offset is small enough for normal ADC operation. An SR latch is used at the output stage to sample the comparator output before the reset phase and to hold the result until the next clock cycle.

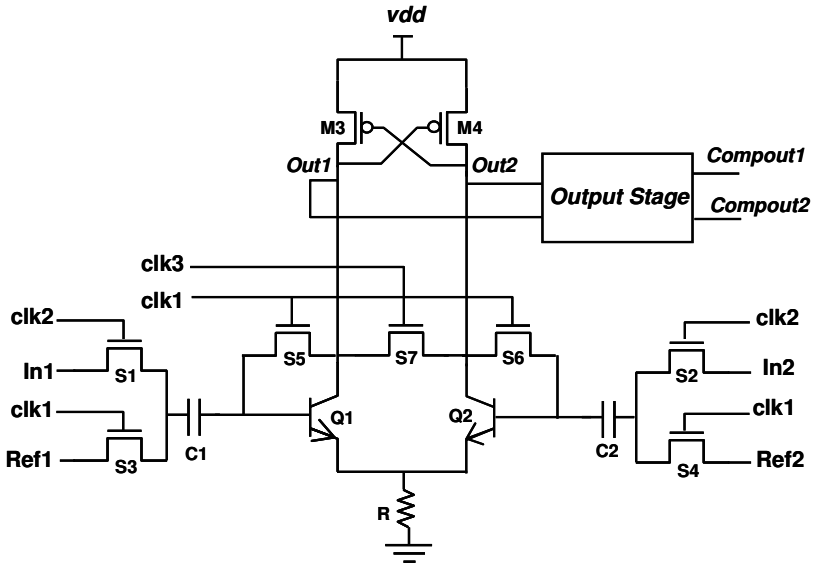

Figure 5.3: Block diagram of the comparator architecture.

5.2 Analysis of the impact of substrate noise on flash ADCs

Switching activity of the digital components, which are located on the same die as analog blocks, inject current spikes into the substrate. These current spikes are injected into the substrate through three mechanisms: 1) capacitive coupling between the digital switching nodes and the substrate, 2) the impact ionization in MOS transistors channels, and 3) supply noise at power and ground contacts, mainly due to the package and bond-wire inductances [63]. As shown in Figure 5.1, this noise propagates through the substrate network and couples to the analog components which are located on the same chip. The generated noise can affect the analog blocks via capacitive coupling to source and drain junctions of MOS transistors or collector-bulk junctions of bipolar transistors. In the MOS transistors, substrate noise can also affect the analog block via the back-gate effect in transistors; i.e. the threshold voltage is modulated by changes in V_{SB} through the following equation

$$V_{th} = V_{th0} + \gamma(\sqrt{V_{SB} + \phi_0} - \sqrt{\phi_0}) \tag{5.1}$$

In a flash ADC, the coupling of substrate noise can degrade the performance of the ADC by impacting its sampling or quantization stage. Noise in the substrate can change the threshold voltage of the sampling switches through back-gate effect and hence change their sampling instant. This acts like jitter in the sampling time and becomes important at high input frequencies or high resolution. If the error caused by jitter is the dominant source of noise, it can affect the SNR of the ADC by the following equation [69]

$$\begin{aligned} SNR &= -20\log(2\pi f \Delta t) \\ \Delta t &= \frac{\Delta V_{th}}{V_{dd}} t_{fall} \end{aligned} \tag{5.2}$$

where f is the input frequency and Δt is the standard deviation of the jitter. However, this is not a major problem in our designed ADC due to its low resolution and large quantization noise.

Noise can also couple to nodes of the comparator core through junction-to-substrate capacitance and change the output or the regeneration time of the latch. The most common impact of noise on a comparator is changing its decision time. The regeneration time of the latch is defined by the following equation [70]

$$t_d = \frac{C_{out}}{g_{m3,4}} \ln \frac{V_{final}}{V_{initial}} \tag{5.3}$$

where C_{out} is the total capacitance at nodes Out_1 or Out_2 and $g_{m3,4}$ is the transconductance of M_3 or M_4. Noise that couples symmetrically or asymmetrically to the output nodes of the latch can affect its regeneration time by changing the voltage levels and therefore the range the outputs must swing (V_{final}-$V_{initial}$). The regeneration time of the latch can also change due to variations in the transconductance of the switches ($g_{m3,4}$) caused by the impact of noise on the threshold voltage. The small

variations in the decision time of the comparator, caused by substrate noise, do not usually affect the performance of the converter. In a flash ADC, each comparator makes its decision at a different time depending on its respective input level; therefore, the latch output is always sampled right before the end of the clock cycle to make sure the comparators have completely resolved their outputs. As a result, when the comparator is in its normal operation region, small variations in its regeneration time do not transfer to the digital output code of the ADC. Nevertheless, if the comparator is close to the meta-stability region [12], variations in its regeneration time due to large noise spikes in the substrate can lead to an unresolved decision by the end of the clock cycle and therefore a wrong code at the output of the ADC.

Another impact of the substrate noise on the comparator block is generating an incorrect decision at its output. This happens when noise spikes couple asymmetrically to the input or output of the comparator latch at the beginning of the regeneration phase. Asymmetric noise coupling is the result of different distances from the noise source to the input or output nodes of the latch. If the differential input of the comparator is small, large noise spikes can reverse the input polarity and change the decision of the comparator (see Figure 5.4). This is the most important influence of noise on the flash ADC performance which can lead to an incorrect output digital code in the ADC and degrades its SNDR. With increasing noise frequency, there is a higher chance that the noise spikes happen during the regeneration phase of the latch; therefore SNDR degradation increases with noise frequency. This disturbed decision can also occur when noise spikes happen during the offset cancellation mode and corrupt the input offset of the comparator. With a large variation in the input offset, the polarity of the input changes and the comparator can make a wrong decision.

In general, the main source of performance degradation in a flash ADC is asymmetric noise coupling to the comparator blocks. Therefore, in order to have less sensitivity to substrate noise, complete symmetry should be considered in the layout of analog blocks both in terms of their distance to digital noise sources and the availability of substrate to ground contacts around sensitive nodes.

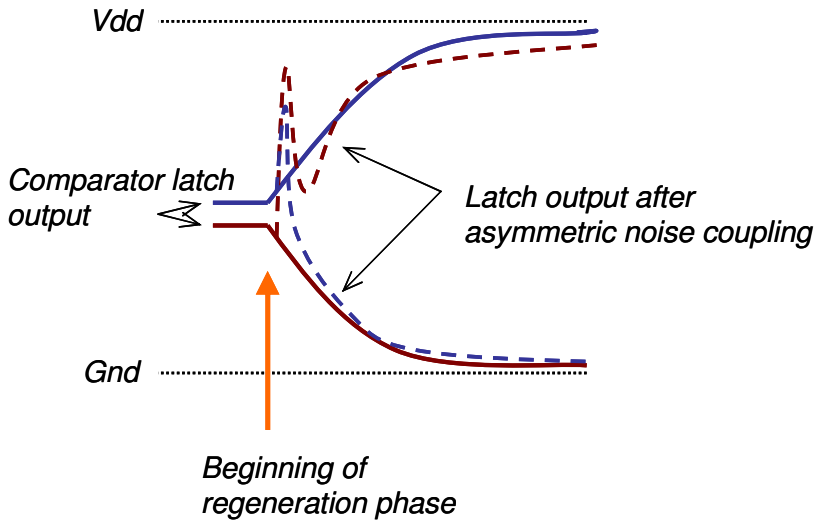

Figure 5.4: Latch output before and after noise coupling.

5.3 Experimental setup

In order to investigate the impact of substrate noise on flash ADC performance, noise emulators and ADC building blocks are integrated on a test chip fabricated in a 0.18-µm BiCMOS process. The tehcnology used in this chip is a non-epi process; therefore, the distance from the noise generator is an important factor of performance degradation in different blocks. Figure 5.5 includes the die photo and layout of the test chip which shows the location of different blocks on the chip. Two Digital Noise Emulators (DNEs) are used on the two corners of the chip to inject noise into the substrate at different locations. The DNE includes a chain of inverters that drive a large capacitance connected to the substrate. The capacitor is made out of a reversed biased P-N junction and is used for coupling a maximum amount of noise from the output of the inverter chain to the substrate. The noise generated by these DNEs well approximates switching noise of real digital blocks. Separate supply and ground networks are used for the DNEs and ADC blocks to exclude the effect of supply noise on the analog components. Figure 5.6 shows a block diagram of the DNE.

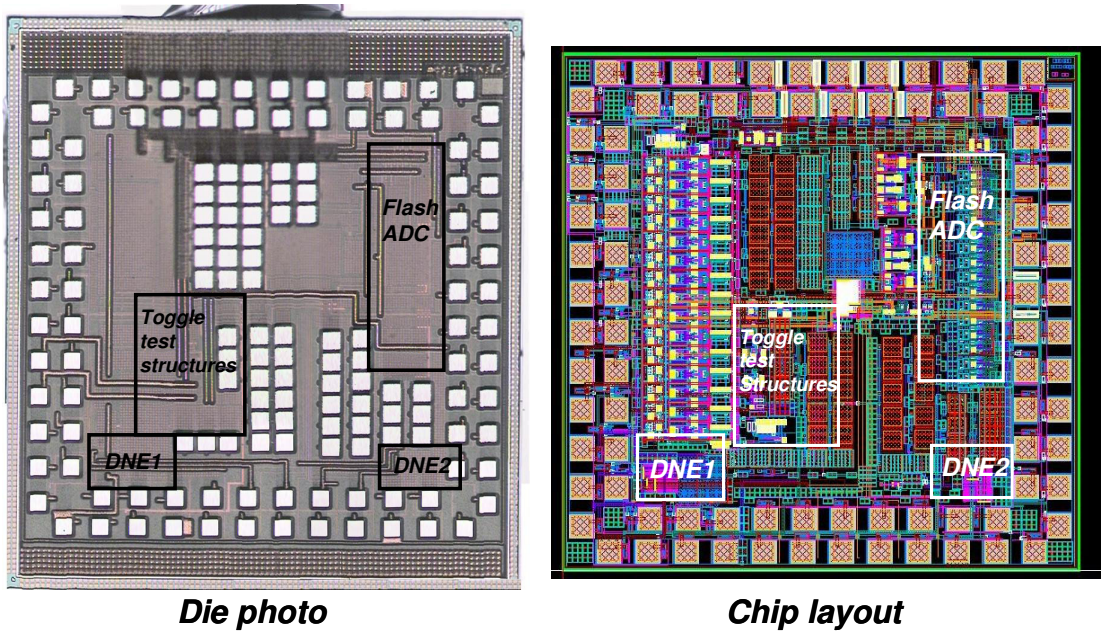

Figure 5.5: Die photo and layout of the test chip.

A 4-bit flash ADC is located on the right side of the chip to measure its overall performance degradation in presence of substrate noise. The ADC is made out of 15 comparators and occupies an active area of 0.2mm×0.9mm. The comparator outputs generate a thermometer digital code which is sent directly off chip as a digital representation of input samples.

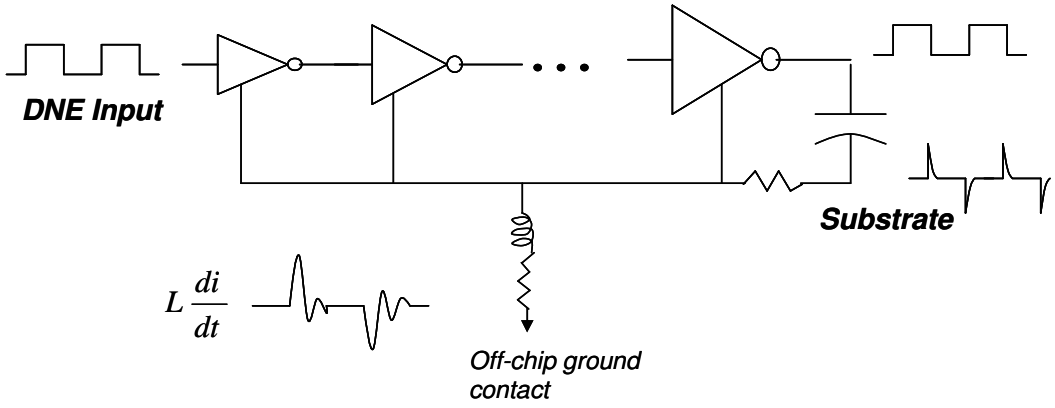

Figure 5.6: Block diagram of the digital noise emulator (DNE).

In order to test the impact of noise on the regeneration time or decision of the individual comparator blocks, comparators are used in a toggling test structure in other parts of the chip. In this test structure, which is shown in Figure 5.7, the comparator is used in a positive feedback loop and hence oscillates with half of the clock frequency. The comparator's outputs are attenuated to the minimum resolution of the ADC before connecting them back to the inputs to simulate the performance of the most sensitive blocks in the ADC. In this structure the variation in the regeneration time of the latch can be observed as jitter in zero-crossing points of the toggling output. This can be measured both with a jitter histogram in the time domain or by observing modulated tones generated by the noise signal in the frequency spectrum of the output. Furthermore, if the input polarity is reversed by noise coupling, it can be observed by having a cycle without toggling at the output.

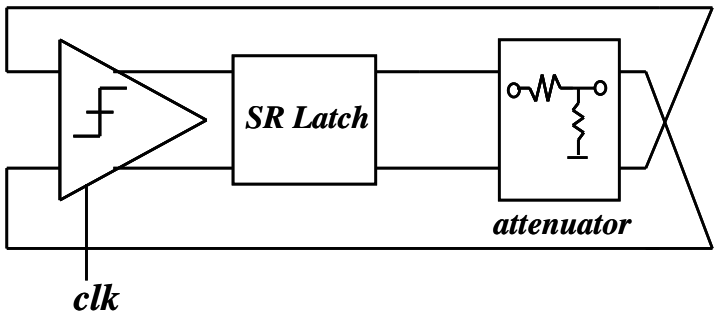

Figure 5.7: Block diagram of the toggling test structure.

5.4 Measurement results

The flash ADC on the chip was first tested without noise sources to measure its nominal performance characteristics. Figure 5.8 shows plots of INL and DNL of this ADC together with its SNDR vs. input and clock frequencies. The maximum INL and DNL of the ADC are 0.4 LSB and 0.35 LSB, respectively. SNDR of the ADC is 24 dB corresponding to an ENOB of 3.7 (at f_{in}=100 MHz and f_s=262 MHz).

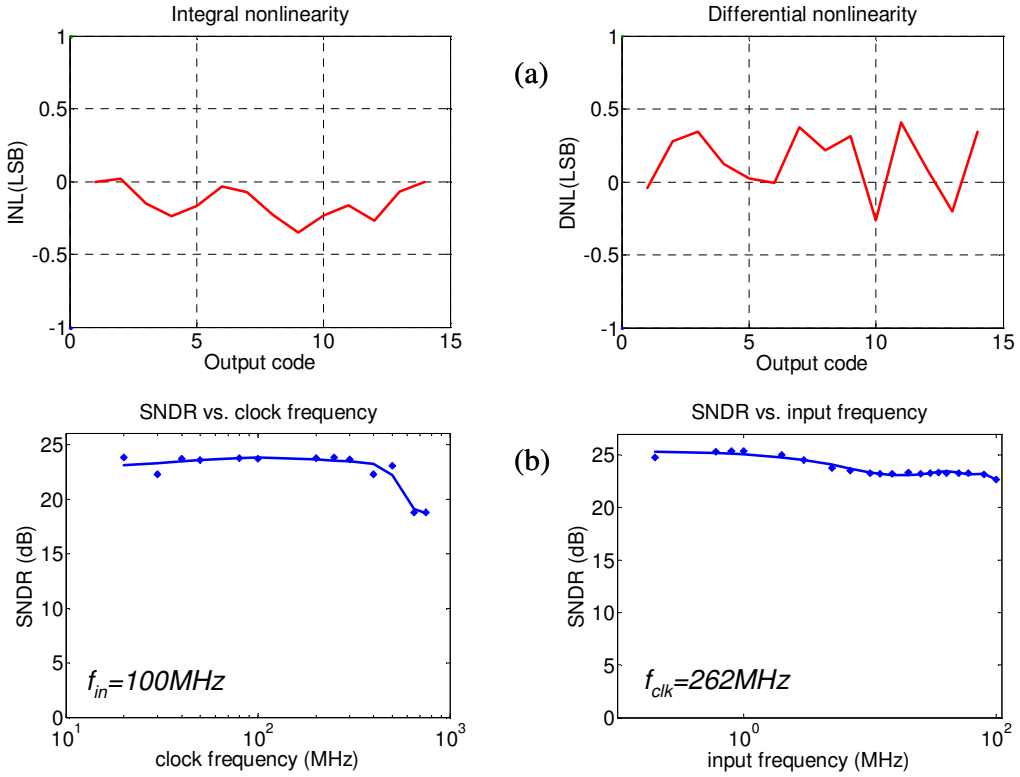

Figure 5.8: a) INL and DNL of the ADC. b) SNDR of the ADC vs. clock and input frequencies.

By turning on the Digital Noise Emulators, incorrect codes were observed at the output of the ADC that degrades its SNDR. The ADC performance degrades more with increasing digital clock frequency, which generates higher frequency noise spikes. Figure 5.9 shows the SNDR plot of the ADC vs. noise frequency for sampling speed of 262 MHz and input frequency of 5 MHz. An SNDR degradation of 2 dB can be observed in this plot for noise frequencies of above 200 MHz. This is equivalent to a 0.33-bit reduction in the effective number of bits.

The toggle test structure shown in Figure 5.7 was tested in presence of substrate noise to investigate the impact of noise on the comparator decision time. Jitter is measured in the time domain using the horizontal histogram in a digitizing oscilloscope. Figure 5.10(a) shows comparator jitter vs. noise frequency for two different clock frequencies. It is observed from this plot that jitter increases

significantly with increasing noise frequency and is worse for higher clock frequencies. In order to investigate the effect of distance on the sensitivity of a comparator block to substrate noise, the toggle test structure is tested separately with two DNEs on the chip. DNE1 and DNE2 are located 400µm and 1200µm away from the target block, respectively. Figure 5.10(b) shows comparator jitter vs. noise frequency for the two DNEs. It is observed from the plots that the closer DNE has a significantly larger impact on the comparator which is expected in a non-epi process.

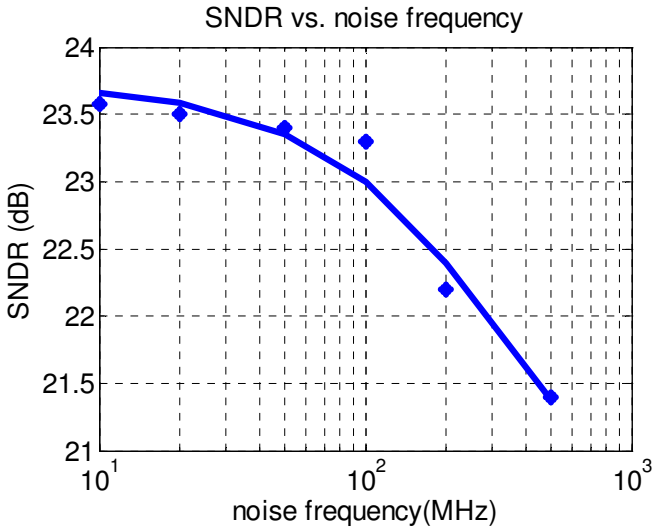

Figure 5.9: SNDR of the ADC vs. noise frequency.

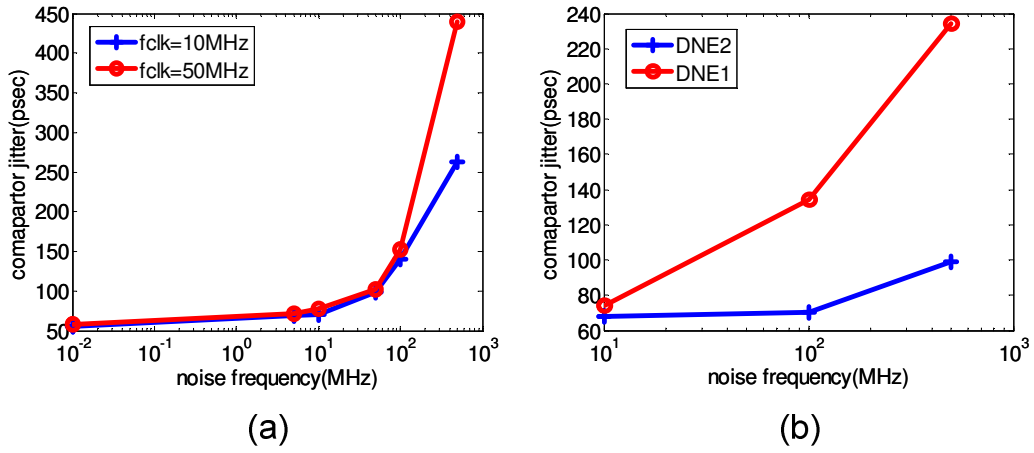

Figure 5.10: (a) Comparator output jitter vs. noise frequency for two clk frequencies, (b) Comparator jitter vs. noise frequency for DNEs at two different locations.

5.5 Summary

In this chapter, we investigated a 4-bit flash ADC and its individual building blocks in presence of substrate noise. Experimental results were shown for this ADC and a toggling test structure made by a comparator. Analysis and measurement results showed that substrate noise can cause major degradation in the performance of the ADC if the digital switching occurs at high frequencies and gets worse as the digital noise source is located closer to the sensitive analog blocks. The main reason for degradation of the ADC is generation of wrong codes at the output of comparators caused by asymmetric noise coupling to its differential nodes. Therefore, it is important to keep the layout of the circuit as symmetric as possible with respect to the noise sources. Substrate noise may not cause a significant problem in low-resolution or low-speed ADCs, but as the resolution and speed of the ADCs increases, more care should be taken in terms of keeping the sensitive analog parts further away from the digital noise sources and protecting the sensitive parts using methods of noise reduction like guard rings.

Chapter 6

Conclusion

6.1 Summary

IF sampling has become an attractive approach in the receiver path of wireless base stations. This method takes away one or more steps of down-conversion in the system and moves some of the analog functions to the digital domain; hence reduces the number of components and size of the analog part of the system. However, the ADCs used in these applications need to have very good linearity over their entire input bandwidth. The linearity of ADCs at high input frequencies is mainly limited by the dynamic errors generated at their front-end. These memory dependent errors are generally modeled with Volterra series. However, the inverse Volterra series becomes very complex as the order of nonlinearity and memory in the system increases and it requires intensive computational power that is usually impractical even in today's fine-line technology.

In this dissertation, we investigated the sources of nonlinearity at the front-end of the ADC consisting of the acquisition stage and the input driving circuit. The two main sources of error at the acquisition stage are charge injection and tracking nonlinearity. It was shown in our analysis that the main source of dynamic nonlinear errors at high input frequencies is the tracking nonlinearity caused by the input dependent switch resistance charging the sampling capacitor. Modeling of this error showed that nonlinearity and memory come from two different sources and can be

separated in the distortion function to make it a simpler version of the Volterra series. The nonideal circuit components in the input driving circuit also add to this nonlinear error but do not change the general form of the distortion model.

Based on the circuit level information from the sources of nonlinearity in the system, we proposed a digital correction scheme that can be applied to the ADC output in order to improve its linearity performance over its entire input bandwidth. The correction filter uses unknown coefficients that need to be calibrated using a set of input and output points from training signals. We showed in our analysis that three single tone sinewaves in each Nyquist zone provides enough information for extracting the filter coefficients. After calibrating the coefficients, the correction filter can be used to compensate nonlinearities in any bandlimited signal located in that specific Nyquist zone.

The proposed algorithm was applied in simulation using the nonlinear model of a track-and-hold system and showed significant improvement in its SFDR over several Nyquist zones. It was also applied to the simulation results of the whole front-end of the ADC and showed good improvement in linearity performance even in presence of other circuit nonidealities at the input driving circuit.

In order to test the performance of the proposed correction method in experiment, a state-of-the-art commercially available ADC was used and the algorithm was applied to its output. Our measurement results showed that the proposed digital enhancement scheme can improve SFDR of the ADC to more than 83 dB up to input frequencies of 470 MHz. The method was also tested on different ADCs and ADCs with different input driving circuits and showed significant improvement in linearity performance in all of them.

In order to test robustness of the algorithm, the coefficients of the filter were calibrated once while the ADC was in its nominal operating condition. The chip was then subjected to temperature variation of $\pm 30^{\circ}$C relative to room temperature and was tested with the same correction filter. Our results showed that the correction is not

very sensitive to temperature variations and has good linearity improvement over the temperature range in our experiment.

For ADC applications that do not include an FPGA in their system, the digital correction scheme should be integrated on chip together with the ADC. The presence of both analog and digital components on the same chip generates noise in the substrate which can degrade the performance of the system. In this thesis, we investigated the impact of this noise on a high speed flash ADC and analyzed how its different blocks can be affected by the switching noise in the substrate. We implemented a test setup and showed our measurement results from performance degradation in the ADC versus noise distance and frequency. It was shown that SNDR of the ADC degraded significantly with increasing noise frequency and is affected more with a digital noise source located at a closer distance.

6.2 Suggestions for future work

The proposed calibration algorithm in our method is a foreground calibration, meaning that the coefficients are extracted once and kept constant during ADC operation. However, if the ADC gets subjected to large variations in the environmental condition, its nonlinear function might change and the coefficients in the correction filter need to be updated. Therefore, an interesting future step would be to investigate the possibility of implementing a background calibration scheme. A block diagram of one possible background calibration scheme is shown in Figure 6.1. In this structure, a replica of the ADC's front-end including track-and-hold and the input driving circuit is used in the second path for updating the coefficients. The training signal is continuously applied to this replica front-end while the ADC is performing its normal operation. The output of the replica circuit goes to a second ADC which needs to be very accurate but can be very slow and the ADC output is used for updating the coefficients in the filter through the calibration algorithm.

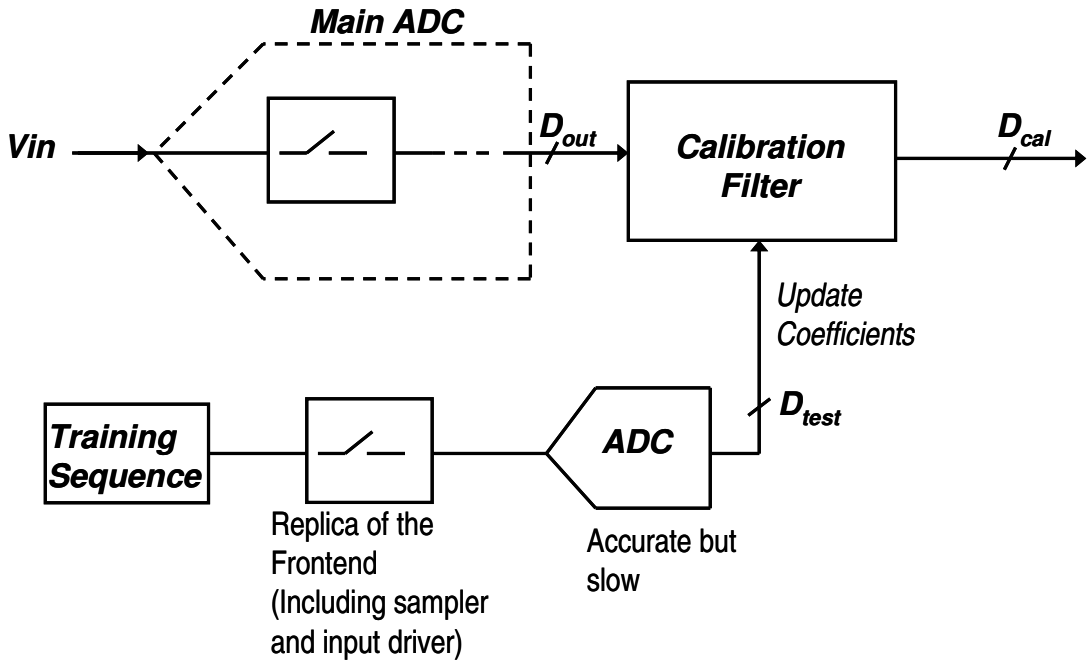

Figure 6.1: Block diagram of the background calibration scheme.

From the analysis in the previous chapters, we note that the main source of dynamic nonlinearity at the ADC's front-end does not depend on the sampling speed; therefore, the clock frequency of the second path can be chosen much slower than the main path to ease some of the requirements on the second ADC. However, some nonlinearity caused by the input driving circuit depends on the tracking time of the clock and hence that should stay constant in the two paths. Therefore, it is required to generate a clock with e.g. a duty cycle of 1/2000 if the frequency of the second clock is 1000 times smaller than the main clock. It is also important to choose the training signals in the desired Nyquist zone of ADC's operation, even though they will be subsampled with a larger factor in the second path.

The mismatch between the two paths can degrade the accuracy of the correction filter. It is therefore important to investigate the matching requirement between the ADC's front-end and its replica and make sure they generate sufficient accuracy in the correction process. The second ADC, which can be slow, needs to be more linear than

the desired linearity in the main ADC. This can be a high-resolution, low-speed ADC made e.g. with a sigma-delta modulator. The clock speed of the second path is chosen based on the time required for each step of calibration and the required update interval for the coefficients. It also needs to be optimized so that it will not add a considerable amount of power to the system.

Another useful future work is to find methods for generating the training signals required for calibrating the coefficients. In this work we used three single tone sinewaves in each Nyquist zone for extracting the filter coefficients in that zone. However, these sinewaves need to be clean and more linear than the requirement in the ADC. It would be interesting to investigate how these tones can be generated on chip or on the board with minimum added cost and power to the system. As an alternative, other kinds of training signals can be investigated to a find a more practical choice to be used in a background calibration scheme.

Some other suggestions for future work in this area include investigating how the proposed correction algorithm can perform in presence of a BJT acquisition stage or looking into the possibility of making some modification in the analog circuit at the front-end. For instance, bringing some feedback from the digital domain to correct some of the nonlinearities by analog means is one possibility. This can be done through e.g. adding a variable capacitor that can be adjusted based on measuring the nonlinearity of the resistance in the input track-and-hold system in order to compensate some of its nonlinearity.

Appendix A

TCAD Simulation of a Track-and-Hold Circuit

In order to perform simulations without relying on the accuracy of the BSIM model used for transistors and also to be able to observe charge flow and transistor channel variation, we simulated the bottom-plate track-and-hold circuit using mixed-mode TCAD tools (see Figure A.1). In this simulation the device structure is defined for NMOS transistors. These transistor models are then used inside the circuit to perform transient and AC simulations. At each step of simulation, voltages, currents and charges are defined by numerically solving the device equations including the boundary conditions.

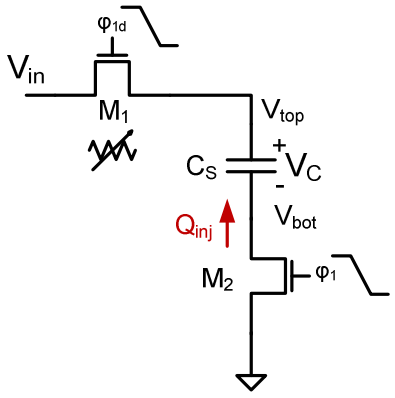

Figure A.1: Bottom-plate track-and-hold circuit.

In this appendix we include a netlist defining the device structure and also a netlist showing how the device is used inside a circuit simulation. These simulations have been done using Taurus simulation tools [35].

A.1 Defining device structure

The following netlist shows how an NMOS transistor with a channel length of 130 nm can be defined in TCAD simulation tools.

```
# Synopsys Taurus-Device Example 0.13um NMOSFET
# Structure generation

# Enable device mode
Taurus {device}

# Define the device size, list the regions, and specify fixed mesh
lines
DefineDevice (
  minX=-200nm, maxX=200nm,
  minY=-110nm, maxY=250nm,

  Region (name=silicon1, material=silicon),
  Region (name=oxide1,   material=oxide),
  Region (name=gate,     material=electrode),

  x=-100nm, x=-50nm, x=-10nm, x=0nm, x=10nm, x=50nm, x=100nm,
  y=-100nm, y=-2nm, y=-1nm, y=0nm, y=1nm, y=10nm, y=50nm, y=60nm
)

# Define the silicon substrate region
DefineBoundary (
  region=silicon1,
  Polygon2D (
    Point (x=-200nm, y=0nm), Point (x=-100nm, y=  0nm), Point (x=
100nm, y=  0nm),
    Point (x= 200nm, y=0nm), Point (x= 200nm, y=250nm), Point (x=-
200nm, y=250nm)
  )
)

# Define the oxide region
DefineBoundary (
  region=oxide1,
  Polygon2D (
    Point (x=-100nm, y=-100nm), Point (x= -50nm, y=-100nm),
    Point (x= -50nm, y=  -2nm), Point (x=  50nm, y=  -2nm),
    Point (x=  50nm, y=-100nm), Point (x= 100nm, y=-100nm),
    Point (x= 100nm, y=   0nm), Point (x=-100nm, y=   0nm)
  )
)

# Define the electrode gate region
DefineBoundary (
  region=gate,
  Polygon2D (
    Point (x=-50nm, y=-100nm), Point (x=   50nm, y=-100nm),
    Point (x= 50nm, y=  -2nm), Point (x=  -50nm, y=  -2nm)
```

```
  )
)
# Initial coarse regrid
#Regrid (gridProgram=pm, meshSpacingX=20nm, meshSpacingY=20nm)


# Flat Contacts
DefineContact (name=source,    X (min=-200nm, max=-99nm), Y (min= -
1nm, max=  1nm))
DefineContact (name=drain,     X (min= 99nm, max= 200nm), Y (min= -
1nm, max=  1nm))
DefineContact (name=substrate, X (min=-200nm, max= 200nm), Y
(min=249nm, max=251nm))
#DefineContact (name=gate, X (min=-50nm, max= 50nm), Y (min=-111nm,
max=-109nm))

# Substrate Doping: P-type Uniform
Profile (name=Ptype, region=silicon1, Uniform (value=5e17))
#Profile (name=Ntype, region=gate, Uniform (value=1e20))
Profile (name=Ptype, region=silicon1,
         Gauss ( peakvalue=4.2e18, sigma=25nm,
       Edge (point (x=-200nm, y=50nm), point(x=200nm, y=50nm) ) ) )
Profile (name=Ptype, region=silicon1,
         Gauss ( peakvalue=1e18, sigma=10nm,
      polygon (point (x=-50nm, y=10nm), point(x=-10nm, y=10nm),
                point (x=-10nm, y=50nm), point(x=-50nm, y=50nm) ) ) )

Profile (name=Ptype, region=silicon1,
        Gauss ( peakvalue=1e18, sigma=10nm,
      polygon (point (x=50nm, y=10nm), point(x=10nm, y=10nm),
                point (x=10nm, y=50nm), point(x=50nm, y=50nm) ) ) )


# Source extension doping: N-type Gaussian
Profile (
  name=Ntype, region=silicon1,
  Gauss (
    peakValue=7e19, sigma=3nm, lateralERFC=true, lateralRatio=.5,
    polygon (Point (x=-200nm, y=0nm), Point(x=-50nm, y=0nm)
             point (x=-50nm, y=40nm), point(x=-200nm, y=40nm))
  )
)

# Source doping: N-type Gaussian
Profile (
  name=Ntype, region=silicon1,
  Gauss (
    peakValue=2e20, sigma=15nm lateralERFC=true, lateralRatio=.5,
    polygon (Point (x=-200nm, y=0nm), Point(x=-100nm, y=0nm)
             point (x=-100nm, y=80nm), point(x=-200nm, y=80nm))

  )
)

# Drain extension doping: N-type Gaussian
```

```
Profile (
  name=Ntype, region=silicon1,
  Gauss (
    peakValue=7e19, sigma=3nm, lateralERFC=true, lateralRatio=.5,
    polygon (Point (x=200nm, y=0nm), Point(x=50nm, y=0nm)
             point (x=50nm, y=40nm), point(x=200nm, y=40nm))
  )
)

# Drain doping: N-type Gaussian
Profile (
  name=Ntype, region=silicon1,
  Gauss (
    peakValue=2e20, sigma=15nm lateralERFC=true, lateralRatio=.5,
    polygon (Point (x=200nm, y=0nm), Point(x=100nm, y=0nm)
             point (x=100nm, y=80nm), point(x=200nm, y=80nm))
  )
)


# Initial coarse regrid
Regrid (gridProgram=pm, meshSpacingX=20nm, meshSpacingY=20nm)

# Regrid on doping
Regrid (
  gridProgram=pm, meshSpacing=2nm,  region=silicon1
  Criterion (name=NetDoping, delta=.5, type=asinh)
)

# Regrid in channel
Regrid (
  gridProgram=pm,  meshspacingy=4A, region=silicon1
  minX=-50nm, maxX=50nm, maxY=6nm
)

# Zero-carrier solve at equilibrium
Symbolic (carriers=0)
Solve {}

# Regrid on potential
Regrid (
 gridProgram=pm, meshSpacing=2nm, region=silicon1
 Criterion (name=ElectricPotential, delta=.1, type=linear)
)

# Regrid to desired grading factor and maximum element angle
Regrid (
  gridProgram=pm, region=silicon1
  gradingFactor=2.01, MaximumAngle (value=90)
)

# Redo zero-carrier solve
Solve {}

# Save structure
```

```
Save (meshfile=mos_struc_130.tdf)
```

A.2 Using the transistor model in a circuit simulation

This section shows how the above device can be used for simulating the bottom-plate track-and-hold circuit of Figure A.1.

```
# Device Simulation Input File: mos_struc_ex.pdm
Taurus {device}

# Set up device
DefineDevice (
  name=pass1_1
  meshFile=mos_struc_ex.tdf
  areaFactor=10.0
)
DefineDevice (
  name=pass1_2
  meshFile=mos_struc_ex.tdf
  areaFactor=10.0
)
include (mos_mod_ex.pdm)
Contact (name=gate, polydoping=1e20)

DefineCircuit (name=sampler,
      netlist(
            CL1( n1=1, n0=24, value=2e-12),
            R1( n1=24, n0=4, value=5),
            Vg1( n1=2, n0=3, value=0),
            Vin1( n1=3, n0=15, pulse(initialValue=0, appliedvalue=.1,
            delaytime=8e-9,risetime=1e-12, falltime=1e-12,
            pulsewidth=2e-6, period=3e-6)),

            Vin2( n1=15, n0=16, pulse(initialValue=0,
appliedvalue=.1,
            delaytime=18e-9,risetime=1e-12, falltime=1e-12,
            pulsewidth=2e-6, period=3e-6)),

            Vin3( n1=16, n0=17, pulse(initialValue=0,
            appliedvalue=.1, delaytime=28e-9,risetime=1e-12,
            falltime=1e-12, pulsewidth=2e-6, period=3e-6)),

            Vin4( n1=17, n0=18, pulse(initialValue=0,
            appliedvalue=.1, delaytime=38e-9,risetime=1e-12,
            falltime=1e-12, pulsewidth=2e-6, period=3e-6)),

            Vin5( n1=18, n0=19, pulse(initialValue=0,
            appliedvalue=.1, delaytime=48e-9,risetime=1e-12,
            falltime=1e-12, pulsewidth=2e-6, period=3e-6)),
            Vin6( n1=19, n0=20, pulse(initialValue=0,
            appliedvalue=.1, delaytime=58e-9,risetime=1e-12,
```

```
            falltime=1e-12, pulsewidth=2e-6, period=3e-6)),

            Vin7( n1=20, n0=21, pulse(initialValue=0,
            appliedvalue=.1, delaytime=68e-9,risetime=1e-12,
            falltime=1e-12, pulsewidth=2e-6, period=3e-6)),

            Vin8( n1=21, n0=22, pulse(initialValue=0,
            appliedvalue=.1, delaytime=78e-9,risetime=1e-12,
            falltime=1e-12, pulsewidth=2e-6, period=3e-6)),

            Vin9( n1=22, n0=23, pulse(initialValue=0,
            appliedvalue=.1, delaytime=88e-9,risetime=1e-12,
            falltime=1e-12, pulsewidth=2e-6, period=3e-6)),

            Vin10( n1=23, n0=0, pulse(initialValue=0,
            appliedvalue=.1, delaytime=98e-9,risetime=1e-12,
            falltime=1e-12, pulsewidth=2e-6, period=3e-6)),



            vg1_2(n1=5, n0=0, Pulse(initialValue=0, appliedvalue=1.3,
            delaytime=0,risetime=3e-11, falltime=3e-11,
            pulsewidth=4.97e-9, period=1e-8))

            period=1e-5)),
            PNMOS1_1( source=3, drain=1, gate=2, substrate=0,
                  device=pass1_1, widthfactor=10)
            PNMOS1_2( source=0, drain=4, gate=5, substrate=0,
                  device=pass1_2, widthfactor=10)


      )
)
numerics (iterations=20)
symbolic (carriers=0)
solve {}
symbolic (carriers=2)
#Circuit_OP() { couple(iterations=10,LinearSolver=direct) {Poissons,
                ElectronContinuity,HoleContinuity}}

circuit_trans( time=11e-8, initialtimestep=.005e-11, conststep=false,
                  logfile=sampler_transboot4.data,
                  SignalSpecification(
                  Pulse( initialValue=0, appliedvalue=1.3,
                  delaytime=300e-12,
                  risetime=3e-11, falltime=3e-11, pulsewidth=4.97e-9,
                  period=10e-9) )
                  {BiasObject(name=vg1,type=VoltageSource) {pass1_1}
})
                    { couple(iterations=20,LinearSolver=direct)
                    {Poissons, ElectronContinuity,HoleContinuity} }
```

Appendix B

Modeling Signal Derivative

In this appendix we show how the slope of the signal at each sampling point can be related to a linear combination of previous and post samples. This can be used together with equation (2.8) to show that nonlinear model of a track-and-hold circuit can be separated into the product of a static nonlinear function and a linear function with memory.

B.1 Derivative as a linear combination of previous and post samples

For a sub-sampled signal (shown in Figure B.1(a)), the value of the signal at each sampling point can be written using the following equation

$$X(k) = \sum_{k=-\infty} x(t)\delta(t - kT_s) \tag{B.1}$$

where x(t) is the continuous signal, X(k) is the sampled value and T_s is the sampling period. By looking at the spectrum of this signal in the frequency domain (Figure B.1(b)), it is clear that the main signal can be recovered from the sampled values by bandpass filtering the discrete-time signal in the original frequency range. This process is shown in the following equation

$$x(t) = X(k) * h_{bp}(t) = \sum_{k=-\infty} x(kT_s)h_{bp}(t - kT_s) \tag{B.2}$$

Now, by taking derivative of this signal we find

$$x'(t) = \sum_{k=-\infty} x(kT_s) h'_{bp}(t - kT_s) \tag{B.3}$$

$$x'(nT_s) = X'(n) = \sum_{k=-\infty} x(kT_s) h'_{bp}((n-k)T_s) \tag{B.4}$$

where n defines the sample which its derivative is desired.

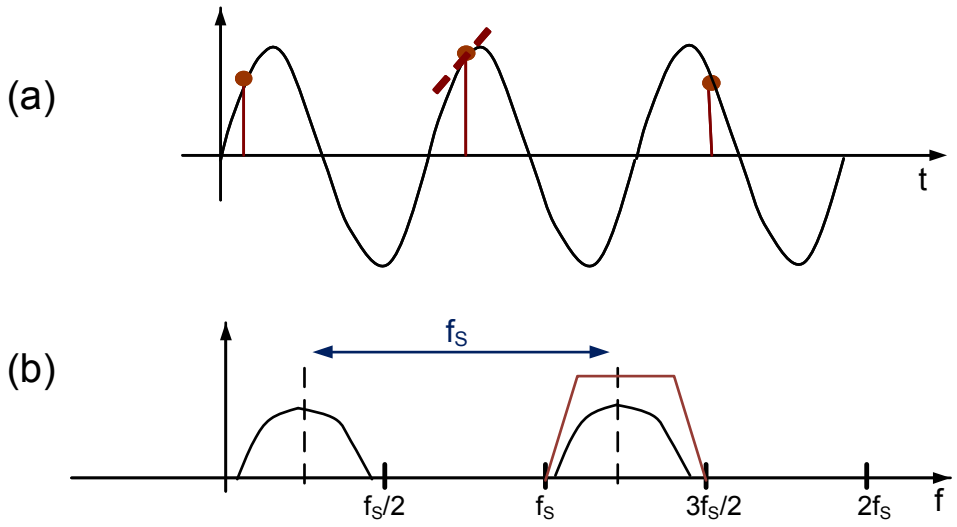

Figure B.1: a) Sub-sampled signal in the time domain. b) Frequency spectrum of a sub-sampled signal.

The above equation shows that the slope of the signal at each sampling point can be modeled as a linear combination of previous and post samples where the coefficients are slopes of the bandpass filter at the sampling points.

B.2 Second and third derivatives

In order to model the second and third order derivative of a sampled signal, we differentiate equation (B.3) again with respect to parameter t.

$$\frac{d^2x(t)}{dt^2} = \sum_{k=-\infty} x(kT_s)\frac{d^2h_{bp}(t-kT_s)}{dt^2} \Rightarrow x''(nT_s) = \sum_{k=-\infty} x(kT_s)h''_{bp}((n-k)T_s) \qquad \text{(B.5)}$$

$$\frac{d^3x(t)}{dt^3} = \sum_{k=-\infty} x(kT_s)\frac{d^3h_{bp}(t-kT_s)}{dt^3} \Rightarrow x'''(nT_s) = \sum_{k=-\infty} x(kT_s)h'''_{bp}((n-k)T_s) \qquad \text{(B.6)}$$

Therefore, from the above expressions, we can observe that all derivatives of a sampled signal can be modeled as a linear combination of previous and post samples where the coefficients are the corresponding derivatives of the bandpass filter at the sampling points.

Appendix C

Coefficient Calibration through RLS Algorithm

In this appendix we show how the Recursive Least Square (RLS) method can be used to calibrate the filter coefficients using three single tone sine waves as training signals. We discuss the general idea and different steps of RLS algorithm in the following sections and show its convergence in one simulation result.

C.1 RLS algorithm

In many adaptive filtering methods, it is common to use a gradient descent algorithm for finding the Least Mean Square (LMS) error

$$\varepsilon(n) = E\{|e(n)|^2\} \tag{C.1}$$

This method, however, requires knowledge of the autocorrelation of the input process and the cross-correlation between the input and the desired output. Therefore, in our calibration process, where the input consists of three sinewaves at three different frequencies and coefficients need to be optimized for all of them, this method is not applicable. LMS method converges to the best set of coefficients for each sinewave and switches to the new set of coefficients for a different sinewave with different statistical information; therefore, it can not find the optimum result for three sinewaves at the same time. An alternative is to consider error measures that can be computed directly from data. For example, a Least Square (LS) error defined in the following

equation requires no statistical information about the input or output and can be evaluated directly from the measured data.

$$\varepsilon(n) = \sum_{i=0}^{n} | e(i) |^2 \tag{C.2}$$

An efficient method for performing this minimization is the Recursive Least Square (RLS) method[52].

C.1.1 RLS method for coefficient calibration

As discussed above, the main goal in this process is to minimize

$$e(i) = V_{in}(i) - X(i)H_n \tag{C.3}$$

where $V_{in}(i)$ are the input samples estimated from the digital outputs, H_n is the matrix of coefficients and X(i) is each row of the matrix shown in equation (3.5), consisting of powers of the current output sample and previous output samples.

The different steps of RLS algorithm for our calibration algorithm can be summarized as shown below

- p= Number of coefficients (size of vector H_n)
- δ= Value used to initialize P(0) (a small positive constant)
- H_n=0
- P(0)= δ^{-1} I
- For n=1, 2, … (the number of samples) compute
    - z= P(n-1)×X(i)′
    - g(n)=[1/(1+X(i) ×z)] ×z
    - a= V_{in}(i)-X(i) ×H_{n-1}(i)
    - H_n=H_{n-1}+a×g(n)

- $P(n)=P(n-1)-g(n) \times z'$

Figure C.1 shows how three different coefficients in the system converge to their final value using RLS algorithm. In this experiment we used three sinewaves as our training signals and used 1000 sample points from each of the test sinewaves.

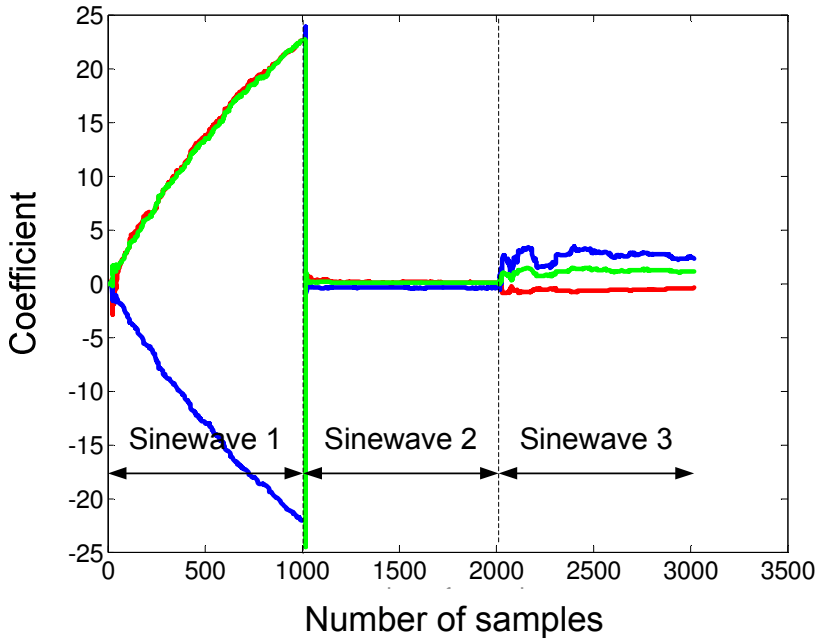

Figure C.1: Convergence of three of the coefficients in H_n using 1000 sample points from each sinewave.

Bibliography

[1] A.M.A. Ali et al., "A 14bit 125Ms/s IF/RF sampling pipelined A/D converter," *Proc. CICC*, pp. 391-94, Sept. 2005.

[2] H. Pan et al., "A 3.3V, 12b, 50MSample/s A/D converter in 0.6um CMOS with over 80dB SFDR," *IEEE J. Solid-State Circuits,* pp. 176-1780, Dec. 2000.

[3] B. Murmann and B. E. Boser, " A 12-bit 75-MS/s pipelined ADC using open-loop residue amplification," *IEEE J. Solid-State Circuits*, pp. 2040-2050, Dec. 2003.

[4] Y. Chiu et al., "Least Mean Square adaptive digital background calibration of pipelined analog-to-digital converters," *IEEE Trans. Ckts Syst. I*, pp. 38-46, Jan 2004.

[5] C. Grace, P.J. Hurst and S.H. Lewis, "A 12 b 80 MS/s pipelined ADC with bootstrapped digital calibration," *ISSCC Dig. Tech. Papers*, pp. 460-461, Feb. 2004.

[6] P. Wambacq and W.M. Sansen, *Distortion Analysis of Analog Integrated Circuits*. Boston, MA: Kluwer, 1998.

[7] M. Hasler and G. Kubin, "Mixed-domain system representation using Volterra series," *International Symposium on Circuits and Systems*, pp. 572-575, May 2008.

[8] M. Schetzen, *The Volterra and Wiener Theories of Nonlinear Systems*, Reprint Edition, Krieger, Malabar, Florida, 1989.

[9] M. Schetzen, ''Theory of *p*th–Order Inverses of Nonlinear Systems,'' *IEEE Trans. on Circuitsand Systems*, pp. 285–291, Vol. 23, No. 5, May 1976.

[10] John Tsimbinos, *Identification and compensation of nonlinear distortion*. Ph.D. dissertation, University of South Australia, Feb. 1995.

[11] National Semiconductor, "ADC14155 Datasheet" [Online]. http://www.national.com/pf/DC/ADC14155.html#Datasheet.

[12] Rudy van de Plassche, *CMOS Integrated Analog-to-Digital and Digital-to-Analog Converters*, Kluwer, 2nd edition, 2003.

[13] M. Gustavsson, J. J. Wikner and N. N. Tan, *CMOS Data Converters for Communications*, Kluwer, 2000.

[14] H. Pan and A. Abidi, " Spectral Spurs due to Quantization in Nyquist ADCs", *IEEE transaction on Circuits and Systems I*, Vol 51, pp. 1422-1439, Aug. 2004.

[15] B.M. Min et al., " A 69-mW 10-bit 80-MSample/s Pipelined CMOS ADC", *IEEE Journal of Solid-State Circuits,* Vol 38, pp. 2031-2039, Dec. 2003.

[16] H. Lee, D. Hodges and P. Gray, "A Self-Calibrating 15-bit CMOS A/D converter," *IEEE Journal of Solid-State Circuits*, pp. 813-819, Dec. 1984.

[17] G. Miller et al., "An 18b 10us Self-Calibrating ADC," in *ISSCC Digest of Technical Paper*, pp. 1238-1327, Dec. 1990.

[18] S. Lee and B. Song, "Digital-Domain Calibration of Multi-Step Analog-to-Digital Converters," *IEEE Journal of Solid-State Circuits,* pp.1679-1688, Dec. 1992.

[19] E. Newman and R. Reeder, "A Resonant Approach to Interfacing Amplifiers to Switched-capacitor ADCs", *Analog Devices*, Application Note AN-827.

[20] R. Reeder, "Frequency Domain Response of Switched-Capacitor ADCs", *Analog Devices*, Application Note AN-742.

[21] W. Yang, et al., "A 3-V 340-mW 14-b 75-MSample/s CMOS ADC With 85-dB SFDR at Nyquist Input," *IEEE J. Solid-State Circuits*, pp. 1931-1936, Dec. 2001.

[22] B. Murmann, "VLSI Data Conversion Circuits," Course Reader, Spring 2005.

[23] J. Kuo, R. Dutton and B. Wooley, "Turn-off Transients in Circular Geometry MOS Pass Transistors," *IEEE Journal of Solid-State Circuits*, pp. 837-844, Oct 1986.

[24] G. Wegmann, E. A. Vittoz and F. Rahali, "Charge Injection in Analog MOS Switches," *IEEE Journal of Solid-State Circuits,* pp. 1091-1097, Dec. 1987.

[25] B. Sheu and C.M. Hu, "Modeling the Swicth-Induced Error Voltage on a Switched-Capacitor," *IEEE Transaction on Circuits and Systems*, pp. 911-913, Dec. 1983.

[26] J. Shieh, M. Patil and B. Sheu, "Measurement and Analysis of Charge Injection in MOS Analog Switches," *IEEE Journal of Solid-State Circuits,* pp. 277-281, Apr. 1987.

[27] M. Chen et al., "Weak Inversion Charge Injection in Analog MOS Switches," *IEEE Journal of Solid-State Circuits,* pp. 604-606, May 1995.

[28] Y. Ding and R. Harjani, "A Universal Analytic Charge Injection Model," *IEEE International Symposium on Circuits and Systems*, May 2000.

[29] C. Eichenberger and W. Guggenbuhl, "Dummy Transistor Compensation of Analog MOS Switches," *IEEE Journal of Solid-State Circuits,* pp. 1143-1146, Aug. 1989.

[30] D.J. Allstot and W.C. Black, "Technological Design Considerations for Monolithic MOS Switched-capacitor Filtering Systems," *Proceeding of IEEE*, pp. 967-986, Aug. 1983.

[31] K.L. Lee and R.G. Meyer, "Low-Distortion Switched-Capacitor Filter Design Techniques," *IEEE Journal of Solid-State Circuits,* pp. 1103-1113, Dec. 1985.

[32] A.M. Abo and P.R. Gray, "A 1.5-V, 10-bit, 14.3-MS/s CMOS pipeline analog-to-digital converter," *IEEE J. Solid-State Circuits*, pp.599-606, May 1999.

[33] Medici User Guide, Version W-2004.09, Synopsis, Sept. 2004.

[34] Dessis Manual, ISE TCAD Release 10.0, Synopsis, 2004.

[35] Taurus Process and Device user's Manual, Version 2002.4, Synopsis, Feb. 2003.

[36] K. Vleugels, " Broadband Oversampling Analog-to-Digital Conversion for Digital Communication," Ph. D. dissertation Stanford University, 2002.

[37] J. Biernacki and D. Czarkowski, "High Frequency Transformer Modeling," *IEEE International Symposium on Circuits and System (ISCAS)*, pp. 676-679, May 2001.

[38] Mini-Circuits Application Note, "How RF Transformers Work," 1999 [online]. http://www.minicircuits.com/pages/pdfs/howxfmerwork.pdf

[39] P.R. Wilson, J.N. Ross and A.D. Brown, "Predicting Total Harmonic Distortion (THD) in ADSL Transformer using Behavioral Modeling," *Proceeding of the international workshop on behavioral model and simulation*, pp. 51-57, Oct. 2003.

[40] E.J. Cunningham et al., "Modeling of Transformer Operating in Saturation," *Electrical Insulation Conference*, pp. 22-25, Sept. 1997.

[41] R. Reeder and R. Ramachandran, "Wideband A/D Converter Front-End Design Considerations," *Analog Dialogue* 40-07, July 2006.

[42] R. Reeder, "Transformer-Coupled Front-End for Wideband A/D Converters," *Analog Dialogue* 39-04, April 2005.

[43] Henrik Lundin, "An Introduction to ADC Error Correction," *Royal Institute of Technology*, Stockholm, May 2005.

[44] J. Tsimbinos and K.V. Lever, " Application of higher order statistics to modeling, identification and cancellation of nonlinear distortion in high-speed samplers and analog-to-digital converters using Volterra and Wiener models," in *Proceedings of the IEEE Signal Processing Workshop on higher-order statistics*, pp. 379-383, June 1993.

[45] E. Balestrieri et al., "A State of the Art on ADC Error Compensation Methods," *IEEE Transaction on Instrumentation and Measurement*, pp. 1388-1394, Aug. 2005.

[46] F. H. Irons, "Dynamic Characterization and Compensation of Analog to Digital Converters," *in Proc. IEEE Int. Symp. Circuits Systems*, Vol. 3, pp. 1273-1277, 1986.

[47] D.M. Hummels, R.W. Cook and F.H. Irons, "Discrete-Time Dynamic Compensation of Analog-to-Digital Converters," *in Proc. IEEE Int. Symp. Circuits Systems*, pp. 1144-1147, 1993.

[48] T.A. Rebold and F.H. Irons, "A Phase-Plane Approach to the Compensation of High-Speed Analog-to-Digital Converters," *in Proc. IEEE Int. Symp. Circuits Systems,* pp. 455-458, 1987.

[49] A.C. Dent and C. Crown, "Linearization of Analog to Digital Converter," *IEEE Transaction on Circuits and systems*, pp. 729-737, June 1990.

[50] F.H. Irons, D.M. Hummels and S.P. Kennedy, "Improved Compensation for Analog-to-Digital Converters," *IEEE Transaction on Circuits and Systems*, pp. 958-961, Aug. 1991.

[51] B.J. Penny, "Analog-to-Digital Converters with Second Order Error Correction," *US. Patent 4985702*, Jan. 1991.

[52] M.H. Hayes, *Statistical Digital Signal Processing and Modeling*, New York, Wiley, 1996.

[53] Simon Haykin, *Adaptive Filter Theory*, 4th edition, Prentice Hall, N.J. 2002.

[54] Ali H. Sayed, *Fundamentals of Adaptive Filtering*, New Jersey, IEEE press, 2003.

[55] K. Poulton et al., "A 20-GSample/s 8b ADC with a 1-MByte Memory in 0.18-um CMOS," *ISSCC Dig. Tech. Papers*, pp 318-319, Feb. 2003.

[56] Y. Dai et al., "Threshold Voltage Based CMOS Voltage Reference," *IEE Proc. Circuits Devices Syst.*, pp. 58-62, Feb. 2004.

[57] F.M. Klaassen and W. Hes, "On the Temperature Coefficients of the MOSFET Threshold Voltage," *Solid-State Electronics*, Vol. 29, pp. 787-789, 1986.

[58] I.M. Filanovski and Ahmed Allam, "Mutual Compensation of Mobility and Threshold Voltage Temperature Effects with Applications in CMOS Circuits," *IEEE Transaction on Circuits and Systems-I*, Vol. 48, pp. 876-884, July 2001.

[59] Texas Instruments, "ADS6143 Datasheet" [Online]. http://focus.ti.com/lit/ds/symlink/ads6143.pdf

[60] National Semiconductor, "Evaluation board user's guide for ADC14155: 14-bit 155Msample/s ADC Converter" [online]. http://www.minicircuits.com/pages/pdfs/howxfmerwork.pdf

[61] National Semiconductor, "LMH6515 datasheet" [online]. http://cache.national.com/ds/LM/LMH6515.pdf

[62] D. K. SU, M. J. Loinaz, S. Masui, and B. A. Wooley, "Experimental results and modeling techniques for substrate noise in mixed-signal integrated circuits," *IEEE J. Solid-State Circuits*, vol. 28, no. 4, pp. 420-430, April 1993.

[63] E. Charbon, *Substrate noise: analysis and optimization for IC design*, Kluwer Academic Publishers, 2001.

[64] Y. Zinzius, G. Gielen, W. Sansen," Analyzing the impact of substrate noise on embedded analog-to-digital converters," *IEEE International Conference on Circuits and Systems for Communications*, June 2002.

[65] K. Makie-Fukuda, T. Anbo, T. Tsukada, T. Matsuura, M. Hotta, "Voltage-Comparator based measurement of equivalently sampled substrate noise waveforms in mixed-signal integrated circuits," *IEEE J. Solid-State Circuits*, vol. 31, no. 5, pp. 726-731, May 1996.

[66] K. Uyttenhove and M. S. J. Steyaert, "A 1.8-V 6-Bit 1.3-GHz Flash ADC in 0.25-mm CMOS," *IEEE J. Solid-State Circuits,* vol. 38, no. 7, pp. 1115-1122, July 2003.

[67] M. Choi and A. A. Abidi, "A 6-b 1.3-Gsample/s A/D converter in 0.35-mm CMOS," *IEEE J. Solid-State Circuits*, vol. 36, no. 12, pp. 1846-1858, Dec. 2001.

[68] B. Razavi, B.A. Wooley, “A 12-b 5-Msample/s Two-step CMOS A/D Converter,” *IEEE J. Solid-State Circuits*, vol. 27, no. 27, pp. 1667-1678, Dec. 1992.

[69] M. Shingawa, Y. Akazawa, T. Wakimoto, “Jitter analysis of High-Speed sampling systems,” *IEEE J. Solid-State Circuits*, vol. 25, no. 1, pp. 220-224, Feb.1990.

[70] B. Razavi, B.A Wooley, “Design Techniques for High-Speed, High-Resolution Comparators,” *IEEE J. Solid-State Circuits*, vol. 27, pp. 1916-1926, Dec. 1992.