A Highly Digital VCO-Based ADC With Lookup-Table-Based Background Calibration

by

Sulin Li

A Dissertation

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Doctor of Philosophy

in

Electrical and Computer Engineering

by

July 30th 2019

APPROVED:

Dr. John A. McNeill, Dean of Engineering, Ph.D. Advisor

Dr. Michael C. Coln, Fellow, Analog Devices, Inc.

Dr. Ulkuhan Guler, Assistant Professor of Electrical and Computer Engineering

Abstract

CMOS technology scaling has enabled dramatic improvement for digital circuits both in terms of digital speed and power efficiency. However, most traditional analog-to-digital converter (ADC) architectures are challenged by ever-decreasing supply voltage. The improvement in time resolution enabled by increased digital speeds drives design towards time-domain architectures such as voltage-controlled-oscillator (VCO) based ADCs. The main challenge in VCO-based ADC design is mitigating the nonlinearity of VCO voltage-to-frequency (v-to-f) characteristics. Achieving signal-to-noise ratio (SNR) performance better than 40dB requires some form of calibration, which can be realized by analog or digital techniques, or some combination. This dissertation proposes a highly digital, reconfigurable VCO-based ADC with lookup-table (LUT) based background calibration based on "split ADC" architecture [1–3]. Each of the two split channels, ADC "A" and "B", contains two VCOs in a differential configuration. This helps alleviate even-order distortions as well as increase the input dynamic range. A digital controller on chip can reconfigure the ADCs' sampling rates and resolutions to adapt to the various application scenarios. Different types of input signals can be used to train the ADCs LUT parameters through the simple, anti-aliasing continuous-time input to achieve target resolution. The chip is fabricated in a 180 nm CMOS process, and the active area of analog and digital circuits is 0.09 and 0.16 mm^2, respectively. Power consumption of the core ADC function is 25 mW. Measured results for this prototype design with 12-b resolution show ENOB improves from uncorrected 5-b to 11.5-b with calibration time within 200 ms (780K conversions at 5 MSps sample rate).

Acknowledgements

First of all, I would like to express my deepest gratitude to my Ph.D. advisor, Professor John McNeill. It has been a great experience for me to work with him at Worcester Polytechnic Institute (WPI) for the past four years. His vision and understanding of integrated circuit (IC) design has significantly influenced me during my Ph.D. life. I really learned a lot from him, not only about circuits, but also the mentality to be a good engineer. His patience, support and guidance in my research work and life, have helped me to become a strong independent researcher. I sincerely thank him for giving me the opportunity to work on this VCO-based ADCs. Without him, this dissertation would not have been possible.

I would also like to thank Dr. Michael Coln at Analog Devices. He is always very kind and patient and I really appreciated his advice on my research. I highly value his guidance to me when I had my internship with him. He opened a door for me in the world of complex ADC system design. Many of my ideas came from discussions with him.

I also want to thank Professor Ulkuhan Guler for her help, support, and guidance. I learned a lot when I was a teaching assistant of her advanced analog IC design class.

Many thanks to Long Pham for his help when I started this project. Many thanks to Jianping Gong for his advice and help in layout. I would also like to thank Ian Costanzo and Devdip Sen for their help and advice in PCB design. Thanks for William Appleyard for his help in soldering. I also want to thank Bob Brown and Edward Burnham for their assistant in Cadence maintenance. I also want to give special thanks to Ian Costanzo for his help and suggestions in grammar, syntax and style of my dissertation.

I am thankful for my friends Junqing Qiao and Yuteng Zhou for their help in

digital circuit design and chip test debugging. I really appreciate them for their encouragement which gave me a lot of power during my Ph.D. journey.

Last but not the least, I would like to thank my family, my wife Han, and my parents for their love, support, and encouragement.

Contents

List of Figures

List of Tables

6

Chapter 1

Introduction

1.1 Motivation

Analog-to-digital converter (ADC) is one of the popular and recurring themes in integrated circuits (ICs) these years. Different kinds of ADCs are used in various scenarios, building a bridge between analog world and digital processing. With the rapid development of microelectronic systems and scaling of CMOS technology [7–9], ultra-low-power ADCs are becoming one of the hottest topics in IC design. They are needed in many analog mixed-signal applications such as implanted biomedical devices, wireless communication, low power sensors, and even high speed wirelines, that demand high power efficiency.

Scaling of CMOS to nanometer dimensions has enabled large improvement in digital power efficiency, with lower power supply voltage and decreased power consumption for logic functions. However, most traditionally prevalent ADC architectures are not well suited to the lower voltage environment. Moreover, another advantage that come with the nanometer CMOS is its dramatically increased speed, which drives the traditional analog design to highly digital assisted mixed-signal system

design.

Thus, it will be a meaningful and valuable work to develop a novel ADC that takes advantage of nanometer CMOS technology. The large improvement in time resolution enabled by increased digital speeds drives the design towards time-domain architectures. Voltage-controlled-oscillator (VCO) based ADCs are one of the most favored ADC architecture nowadays. In contrast to classical ADCs, VCO-based ADC, which is a kind of time-based converter, transfers the continuous analog signal to time-doman signals by taking the advantage of the increased digital speed in nanometer CMOS.

A main obstacle in the VCO-based technique is linearizing the VCO voltage-to-frequency (v-to-f) characteristic. Achieving SNR performance better than 40dB requires some form of calibration, which can be realized by analog or digital techniques, or some combination. A further challenge is implementing calibration without degrading energy efficiency performance.

1.2 Goals

The goal of this Ph.D. work is to develop a highly digital VCO-based ADC with LUT based background calibration. Based on the calibratable "split ADC" architecture [1–3], a differential analog circuit configuration with real-time digital calibration system to linearize the VCO's v-to-f characteristics is proposed. The main challenge to design a good VCO-based ADC is to deal with the nonlinearity of VCO v-to-f characteristics. As described in Chapter 2, most other VCO-based ADC works usually use balanced analog and digital circuitry combination (e.g., $\Delta\Sigma$ loop, replica VCO, etc.) to achieve a good SNR. However, this work chooses a special method, using only simple analog circuitry with highly digital assisted calibration system

to alleviate the VCO's v-to-f nonlinearity. This dissertation not only introduce the background calibration algorithm but also provides the design flow of highly digital self-calibrating VCO-based ADCs. Because of the fast scaling of CMOS technology, the highly digital VCO-based ADC must be one of best ADCs in the future.

1.3 Contribution Overview

To summarize the contribution of the Ph.D. work, this section gives the overview of the related contribution in detail with corresponding publications.

In paper [10], the fundamental limits on energy efficiency performance of VCO-based ADCs are introduced, which include the limits from quantization and jitter. Analysis of a simplified model of the VCO-based ADC approach shows that there is an opportunity for order-of-magnitude improvement in efficiency F.O.M., with a quantization limit floor due to process energy per gate transition. Limitation due to oscillator jitter is expected to be appreciable only for ADC resolutions of order 12-13 bits, depending on process. Simulation results for a reconfigurable VCO-based ADC show conformance with theoretical predictions.

In paper [11], the specific circuit level design of the proposed highly digital VCO-based ADC is completed in 65 nm CMOS process. The proposed ADC is constructed according to split ADC calibratable architecture, and its novel and straightforward design by combining differential arrangement of VCOs and digital calibration engine provide another applicable option for VCO-based ADC creation. The variations of the VCOs v-to-f characteristics are analyzed through Monte Carlo simulation. The detailed calibration algorithm implementation in the digital circuit is also introduced. Simulation results in a 65 nm CMOS process show the propsed VCO-based ADC targeting 13-b resolution can achieve 12.5-b ENOB.

In paper [12], the related digital calibration system is developed and implemented for a delay lock loop (DLL) in 28 nm CMOS process. The two-step coarse and fine digital control circuit is used to mitigate the effects of jitter accumulation iteratively. The simulation results show the proposed DLL output can achieve $\leq$ 0.1psrms jitter clock. The DLL can operate with input clock frequency from 2GHz to 10GHz, enabling interleaved ADC sampling with a low-jitter sample clock over a 20GHz to 100GHz frequency range.

1.4 Organization

This dissertation is organized as follows: Chapter 2 starts with the background of ADC characteristics and ADC nonideal behaviors. Then, the VCO-based ADC is investigated including its advantages and disadvantages. After that, the general "split ADC" concept is introduced briefly. Chapter 3 demonstrates a system architecture of the VCO-based ADC, providing a whole picture of the design. Then, the analog front-end of the proposed work is illustrated block by block. It consists of VCO choice, phase measurement circuits and input ports design implementation. Chapter 4 shows the background calibration and correction technique in detail, as well as introduces the design techniques when implementing the algorithm into real circuits. Chapter 5 introduces the measurement setup including test board design and test plan. Then, the measurement results are demonstrated in detail with quantitative analysis. Finally, Chapter 6 summarizes the research work presented and provides possible paths for future investigation in VCO-based ADC area.

Chapter 2

Background

2.1 ADC Characterization

Analog-to-digital converter (ADC) translates continuous-time signals into digital numbers to represent their discrete amplitude. This section introduces fundamental knowledge of the ADC as well as basic ideals of how an ADC works.

An ADC performs the conversion by sampling the real world analog signal periodically. The number of conversions that the ADC needs to convert analog input

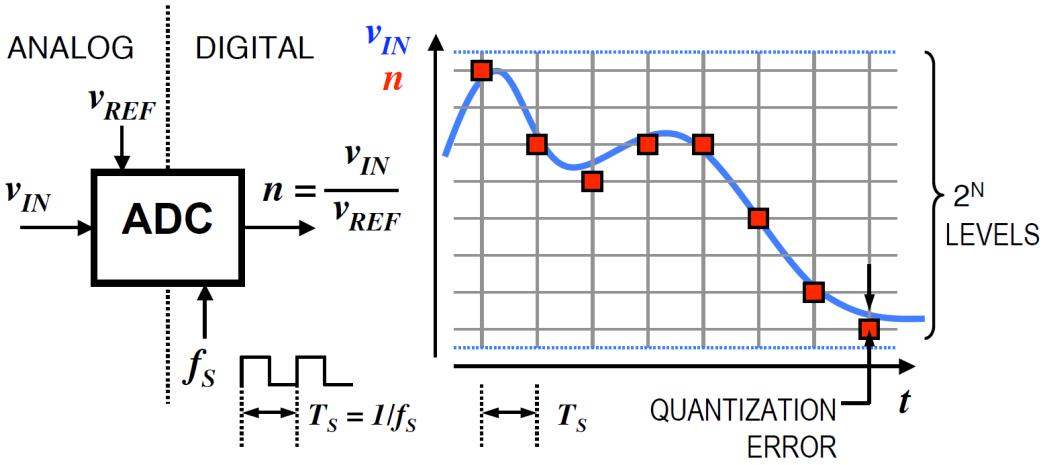


Figure 2.1: ADC overview

to a digital code within a specific time is called the sampling rate, f_s. For example, a 1kS/sec ADC collects one thousand samples in a second of time. As we can see in Figure 2.1, the blue line represents the analog input signal, and the red rectangular points represent the periodically sampled signal data. The time T_s between two adjacent data points is the conversion time. The sample rate is inversely proportional to conversion time, $f_s = 1/T_s$.

The resolution of an ADC is the number of output levels of quantizing an analog signal and it is given in powers of 2 as the output of a ADC comes in binary format. An N-bit ADC represents the analog input using 2^N quantization levels. Right part of Figure 2.1 shows that the input signal range is divided equally into 2^N levels. An ADC needs a reference voltage to digitize the analog signal and divides the reference voltage into small quantization levels. The smallest quantization level that ADC can resolve called the least significant bit (LSB) and it is defined as [13],

$$LSB = \frac{V_{FSR}}{2^N}, \tag{2.1}$$

where V_{FSR} is full scale range (FSR) of input voltage. An ideal ADC transfer function is illustrated in the above plot in Figure 2.2. The Y-axis represents the ADC digital output and X-axis is the analog input. The quantized value of the analog input is represented by the diagonal staircase. The distance between two successive transition points is defined as 1 LSB.

2.2 ADC Errors

The source of ADC errors which affect the accuracy of ADC when it is transferring analog signals to the digital domain, can be described by quantization error, gain error, offset error and nonlinearity error including differential nonlinearity (DNL)

and integral nonlinearity (INL) [13,14]. They all can be expressed in least significant bit (LSB) units or sometimes as a percentage of the full scale range.

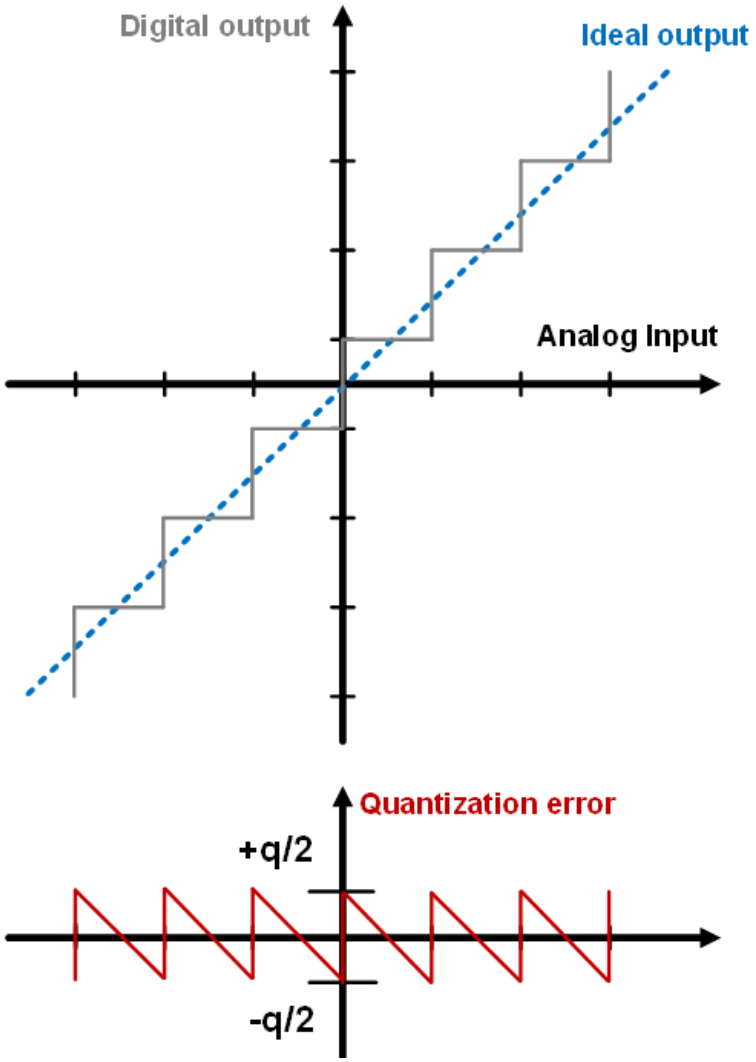


Figure 2.2: Quantization error

2.2.1 Quantization Error

In an ideal ADC system, when the analog input is quantized to a limited resolution of digital codes, error can appear between the analog input and the digital output. For example, a change in the input signal is smaller than the size of the ADC's

LSB, that change can not be detected by the converter and then quantization error occurs.

Figure 2.2 illustrates the quantization error of an ideal ADC. In this example, the quantization level space is assumed as q. The sawtooth as error signal $e(t)$ is the difference between the analog input signal and the quantized digital output signal. The maximum error an ideal ADC makes during the conversion is one half of LSB.

When we treat the quantization error e as having equal probability of lying anywhere in the range $\pm q/2$, its mean square value is given by,

$$e_{rms}^2 = \frac{1}{q}\int_{-q/2}^{q/2} e(t)^2 dq = \frac{q^2}{12}. \tag{2.2}$$

For a N-bit ADC, the fullscale analog input sinewave V_{FS} can be expressed as,

$$V_{FS} = q \cdot 2^{N-1} \dot{s}in(2\pi f t). \tag{2.3}$$

Thus, the signal-to-noise ratio (SNR) for an ideal N-bit ADC is,

$$SNR = 20log\frac{V_{FS,rms}}{e_{rms}} = 6.02N + 1.76dB. \tag{2.4}$$

It is important to know that in equation (2.2), the RMS quantization error is estimated over the full Nyquist bandwidth, DC to $f_s/2$, where f_s is the sampling frequency. If the signal of interest has a smaller bandwidth (BW) or any noise shaping method is used to filter out the noise components, then a correction factor (called process gain) needs to be added to RMS quantization error equation. Therefore this correction factor results in increased SNR and we have,

$$SNR = 6.02N + 1.76 + 10log(\frac{f_s}{2 \times BW}). \tag{2.5}$$

where BW is the bandwidth of interest [13]. The equation (2.5) looks reasonable, because if sampling rate is larger than twice of input signal bandwidth, more unrelated quantization errors are accumulated in equation (2.4). Then a compensation term is added in equation (2.5).

2.2.2 Gain and Offset Error

The gain error shown in Figure 2.3 is defined as the difference between the nominal and actual gain points on the transfer function after the offset error has been corrected to zero. For an ADC, the gain point is the midstep value when the digital output is full scale. The error represents a difference in the slope of the actual and ideal transfer functions and as such corresponds to the same percentage error in each step.

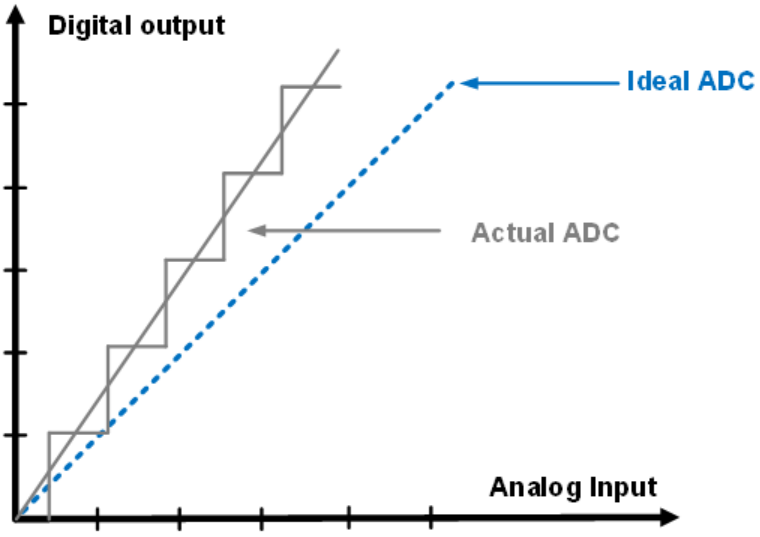


Figure 2.3: Gain error

The offset error as shown in Figure 2.4 is defined as the difference between the nominal and actual offset points. In an ADC, the offset point is the midstep value when the digital output is zero. This error affects all codes by the same amount and

can usually be compensated for by a trimming process [14].

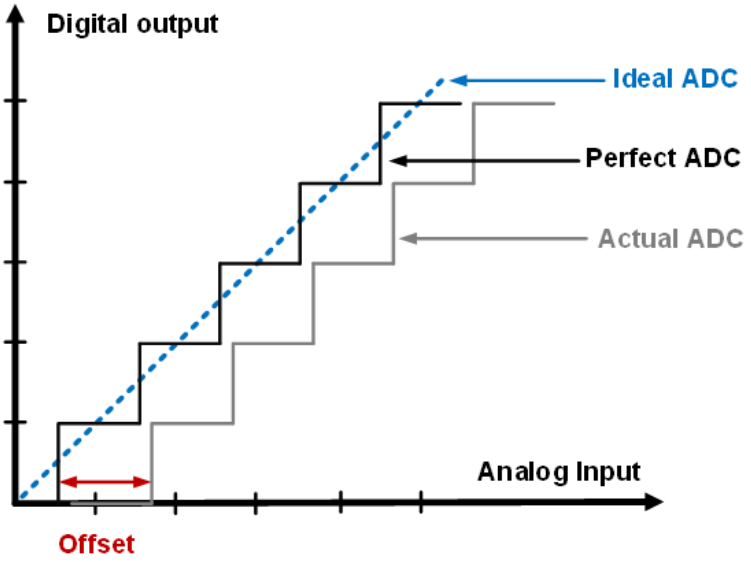


Figure 2.4: Offset error

2.2.3 Differential Nonlinearity (DNL) & Integral Nonlinearity (INL)

The differential nonlinearity error (DNL) shown in Figure 2.5 is the difference between an actual step width and the ideal value of 1 LSB [13]. Therefore if the step width is exactly 1 LSB, then the differential nonlinearity error is zero. If the DNL exceeds 1 LSB, there is a possibility that the converter can become nonmonotonic. This means that the magnitude of the output gets smaller for an increase in the magnitude of the input. In an ADC there is also a possibility that there can be missing codes, for example, one or more of the possible 2^N binary codes are never output.

The integral nonlinearity error (INL) shown in Figure 2.6 is the deviation of the values on the actual transfer function from a straight line [13]. This straight line can be either a best fit straight line which is drawn so as to minimize these deviations or

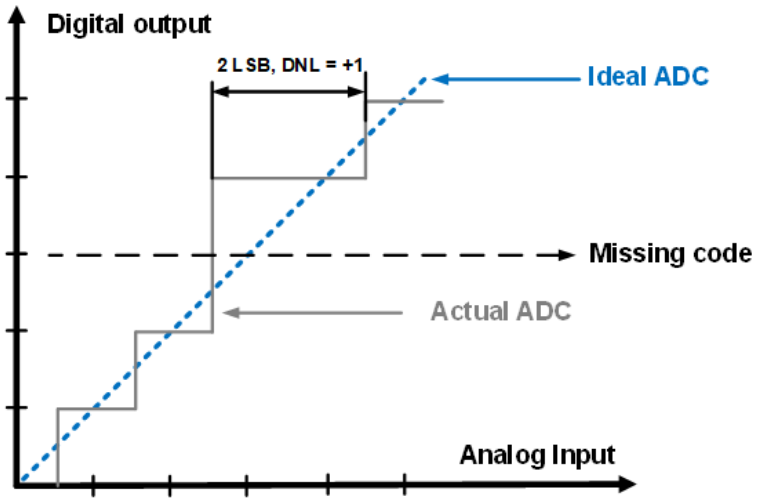


Figure 2.5: DNL

it can be a line drawn between the end points of the transfer function once the gain and offset errors have been nullified. The name integral nonlinearity derives from the fact that the summation of the differential nonlinearities from the bottom up to a particular step, determines the value of the integral nonlinearity at that step.

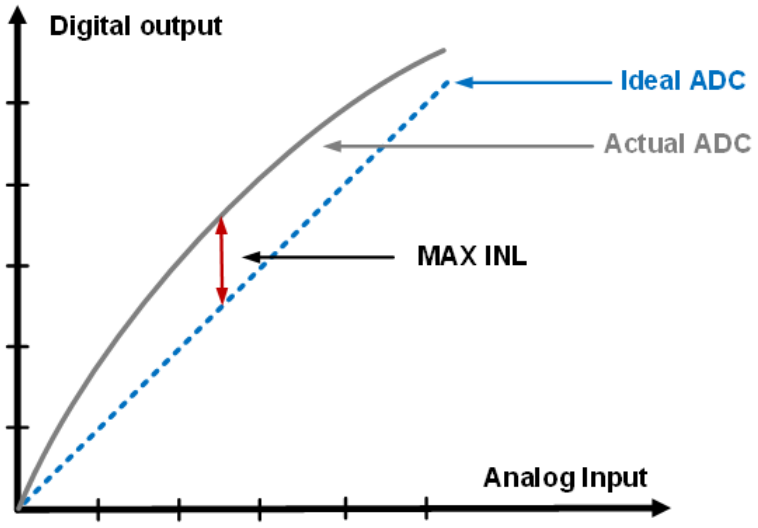


Figure 2.6: INL

2.3 VCO-Based ADC

Many parameters characterize ADC design and performance, ADCs can be broadly described by:

Resolution Number of bits N in the ADC digital output, resolving the input into 2^N quantization levels,

Speed Rate f_S at which ADC samples its analog input and updates digital output,

Power Dissipation Power P_{DISS} consumed by the ADC from supply and any necessary voltage reference,

Architecture The organization of the internal ADC functional blocks.

Design of ADCs is subject to challenging tradeoffs among resolution, speed, power consumption, and complexity. As a result several different ADC architectures have evolved, each suited to a different range of performance. The Walden figure-of-merit (F.O.M.) is commonly used to compare different ADCs, both within and across architectures, and it is defined by

$$FOM = \frac{P_{DISS}}{f_S \cdot 2^{ENOB}} \left[\frac{J}{step}\right]. \tag{2.6}$$

In equation (2.6), resolution is expressed as effective number of bits (ENOB) defined by

$$ENOB = \frac{SNDR - 1.76dB}{6.02dB}. \tag{2.7}$$

in which SNDR is the signal-to-(noise+distortion) ratio and is derived from equation (2.4) .

With the rapid development of ADCs, there are already large variety of different ADC architectures available for different requirements and applications. For example, flash ADCs are usually used in high-speed, low resolution scenarios, while $\Delta\Sigma$ ADCs are the main choice for high-resolution, low-speed applications. As for the

theme of this dissertation, power efficiency is the most critical parameter that is being considered.

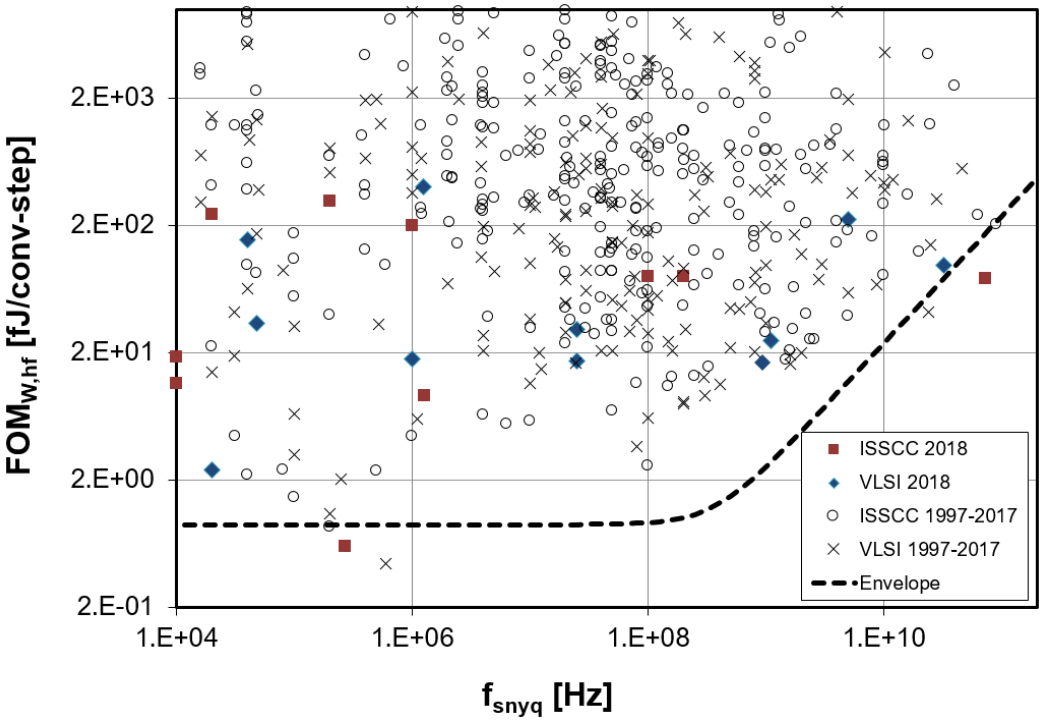


Figure 2.7: Walden FOM vs. Speed [4]

As we can see in the Figure 2.7, the Walden FOM vs. Speed of ADC survey from [4], due to the scaling of CMOS to nanometer dimensions, performance of ADCs are approaching to the lower value of Walden FOM. This is because the nanometer CMOS has enabled dramatic improvement in digital power efficiency, with lower supply voltage and decreased power consumption for logic functions. Thus, a novel ADC architecture called VCO-based ADC is proposed, which can take advantage of the high speed performance and low power consumption of nanometer CMOS technology.

This section consists of following materials: (1) simplified VCO-based ADC structure and its operating principle, (2) properties of VCO-based ADCs, (3) previous works of VCO-based ADCs, (4) fundamental limits of VCO-based ADCs, (5)

nonlinearity of VCO-based ADCs, (6) nonlinearity mitigation techniques.

2.3.1 VCO-Based ADCs Properties

The VCO-based ADC has several important properties making it as a popular candidate among other ADCs, especially in current nanometer CMOS technology. A detailed analysis of VCO-based ADCs' properties was published in paper [5, 15].

One property of VCO-based ADC is that the continuous time input behaves as a low-pass filter, which could be considered as an anti-aliasing filter [5]. Compared with traditional ADCs with discrete time input (sample and hold), it largely improves the ADC's performance. Ideally, the instantaneous frequency of the VCO can be expressed as,

$$f_{VCO}(t) = \frac{K_{VCO}}{2\pi} \cdot v(t), \tag{2.8}$$

where K_{VCO} is the VCO gain in radians per second per volt. The phase-to-digital converter quantizes the VCO phase (i.e., the time integral of the instantaneous frequency) and generates output samples of the result at times nT_s for n = 0, 1, 2,..., where T_s is the counting period.

As we can see in equation (2.9), the nth output sample of the phase to digital converter in radian is a quantized version of

$$\phi[n] = \int_0^{nT_s} K_{VCO} \cdot v(\tau) d\tau. \tag{2.9}$$

Equivalently, equation (2.9) can be written as

$$\phi[n] = \sum_{k=1}^{n} \omega[k] \tag{2.10}$$

where ω can be expressed as (by integrating input signal in a constant period of time)

$$\omega[n] = \int_{(n-1)T_s}^{nT_s} K_{VCO} \cdot v(\tau) d\tau. \tag{2.11}$$

If the input signal is a sinusoidal function $cos(2\pi ft)$, where f is the input frequency. K_{VCO} is constant in the input signal range. If we consider a simple example when the phase is integrated from 0 to T_s,

$$\omega[1] = K_{VCO} \cdot \int_0^{T_s} cos(2\pi f\tau) d\tau = K_{VCO} \cdot \frac{sin(2\pi fT_s)}{2\pi f}. \tag{2.12}$$

Thus, $\omega[n]$ can be considered by passing through a low-pass filter as shown in equation (2.12). The system of Figure 2.8(b) is, therefore, equivalent to that of Figure 2.8(a). It obtains by sampling a filtered version of the input signal as described above and implements (2.10) as a discrete time integrator [5].

Another property that makes ring VCO-based ADC superior to other architectures is that the output of a ring VCO is digital in nature. The clock signal toggles

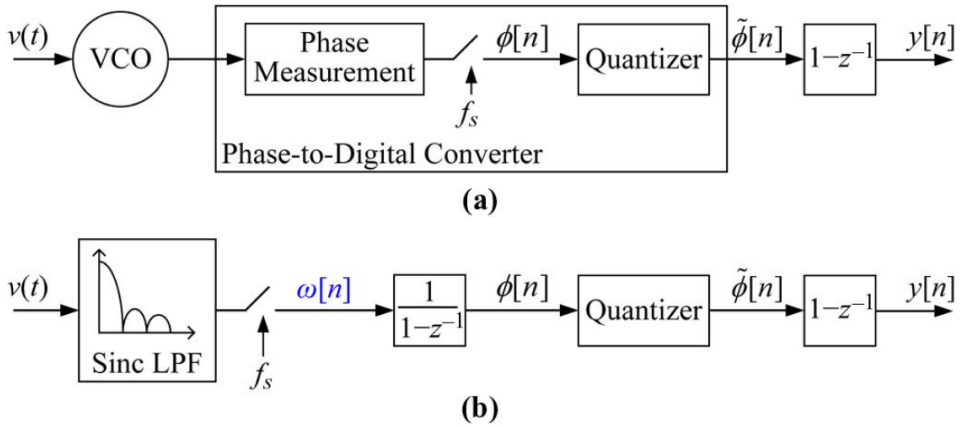


Figure 2.8: Equivalent systems (a) a generic VCO-based $\Delta\Sigma$ modulator, (b) the cascade of a continuous-time low-pass filter, sampler, quantizer, and digital differentiator [5].

between two discrete levels, either VDD or GND. This property makes the VCO a great converter building block which takes advantage of the high performance nanometer CMOS technology without worrying about the decreasing power supply. In addition, since the amplitude of the VCO output does not play an essential role in the quantization process, the architecture greatly reduces the need of additional analog circuitry such as buffers and amplifiers, reducing chip area and power consumption.

What's more, the ring VCO-based ADC can be more reconfigurable than other structures. From equation (2.13) we see that resolution can be increased simply by allowing more time T_{CONV} for the conversion process. As long as the analog circuitry (e.g. Sample & Hold circuit, signal chain bandwidth) are compatible with desired sample and hold time, no reconfiguration of analog circuitry is required. To realize performance commensurate with the increased resolution, the digital back end must be designed with the capability for the maximum resolution required, but pushing the complexity of this trade-off into the digital domain is preferred in nanometer scaled CMOS.

In the end, the VCO-based ADC is inherently monotonic, and exhibits good DNL performance since it does not rely on any matching considerations to maintain local linearity of the ADC transfer characteristic. But INL performance does depend on the linearity of the VCO V-to-f characteristic, and will require some kind of correction.

2.3.2 Other Works of VCO-Based ADCs

There are many different architectures of VCO-based ADC. The main focus of those works is trying to improve VCO-based ADC linearity.

Table 2.1 summarizes the reported performance for previously and recently pub-

lished work [16–29]. One technique is to employ the VCO as an integrator in a $\Delta\Sigma$ modulator [17, 19, 20, 22, 24, 25, 27, 30–33]. Since the VCO is inside a feedback loop, the effects of VCO nonlinearity are reduced. However the technique requires a digital-to-analog converter (DAC) in the $\Delta\Sigma$ feedback path and/or additional analog amplifiers; these go against the design philosophy of avoiding analog complexity in nanometer scaled CMOS. Another technique to improve linearity is to reduce input signal amplitude to avoid nonlinear portions of the transfer characteristic [34, 35], which imposes an SNDR penalty by reducing input signal range. Similarly, circuit techniques such as degeneration to linearize the voltage-to-current characteristic in a current-starved VCO [36] rely on open-loop analog behavior which is not well controlled in nanometer scaled CMOS.

Calibration can also be performed, but techniques reported to date use offline calibration [5, 15, 33, 37, 38] which requires additional precise circuitry and/or the ADC to be removed from the signal path, or use blind calibration techniques [31] which place restrictions on the content of the ADC input signal.

Table 2.1: Summary of other works of VCO-based ADCs

Ref.	Year	Tech [nm]	f_S [MHz]	Bandwidth [MHz]	SNDR [dB]	Power [mW]	Wal F.O.M [fJ/step]	Area [mm^2]
[16]	2008	130	950	10	72	40	614	0.42
[17]	2009	130	900	20	78.1	87	330	0.45
[18]	2011	90	640	8	59.1	4.3	366	0.1
[19]	2012	90	600	10	78	16	120	0.41
[20]	2013	65	2400	37.5	70	39	196	0.08
[21]	2014	90	640	5	73.9	4.1	101	0.16
[22]	2014	65	1280	50	64	38	294	0.49
[23]	2014	40	1600	40	66.8	4.98	35	0.16
[24]	2014	180	35	3.5	70	5	272	0.4
[25]	2015	40	1600	40	59.5	2.57	42	0.017
[26]	2015	65	1000	30	65	8.2	94	0.62
[27]	2015	65	1200	50	71.5	54	176	0.5
[28]	2017	180	51.2	2.5	73.4	4.8	244	0.22
[29]	2019	28	5000	1780	45.2	22.7	30.5	0.023

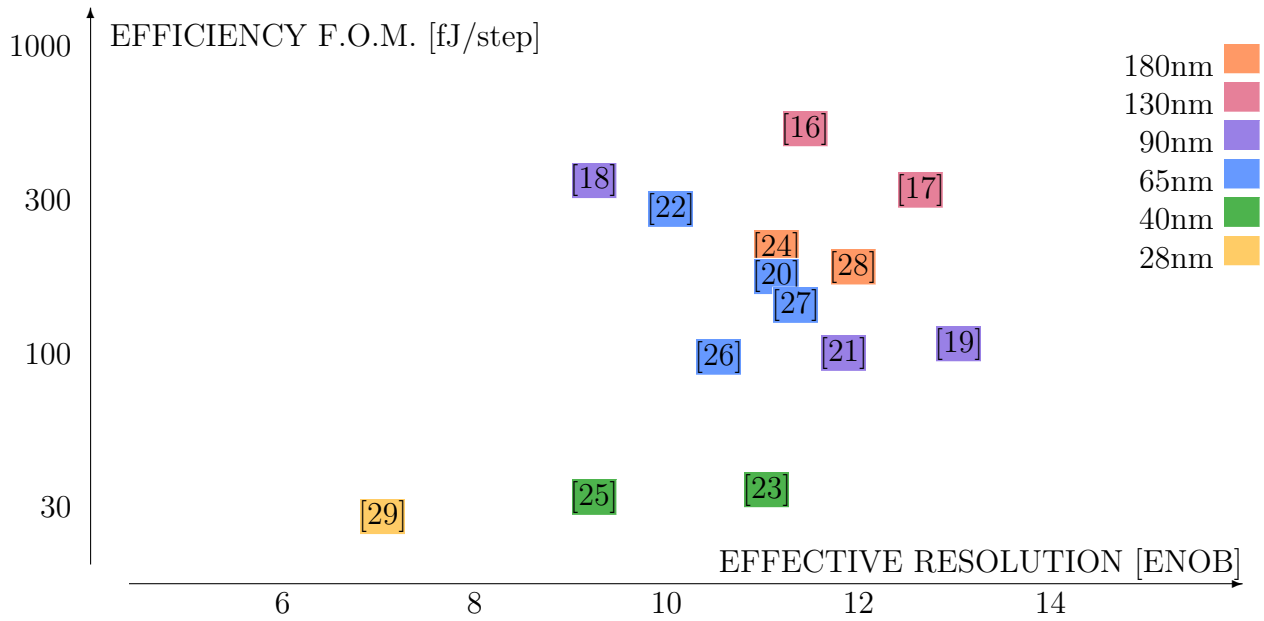


Figure 2.9: Survey of VCO-based ADCs.

2.3.3 Operating Principle of VCO-Based ADC

A VCO-based ADC is a time-based architecture which uses a VCO as an integrator converting continuous analog input to VCO phases which are then quantized by digital measurement circuitry [15].

Figure 2.10 shows a basic and simple VCO-based ADC. A ring VCO is usually used as the integrator to accumulate the phases since it is monotonic and has much lower power consumption comparing with other VCO structures like LC oscillator. The number stages of the VCO is considered according to specific conditions. For example, for single-ended ring VCO, there should be an odd number of stages. If a differential ring VCO is used, we can choose even number. A large number of ring VCO stages could reduce VCO frequency to the following counters, which could reduce some of the design difficulties such as static timing analysis. However, a

large number of ring VCO stages will cost more chip area and it might also induce overtone problems. The detailed discussion will be covered in Chapter 3.

The input v_{IN} of ADC is actually the controlling signal v_{CTL} to the VCO. When the input analog signal changes its value, it will consequently influence the frequency

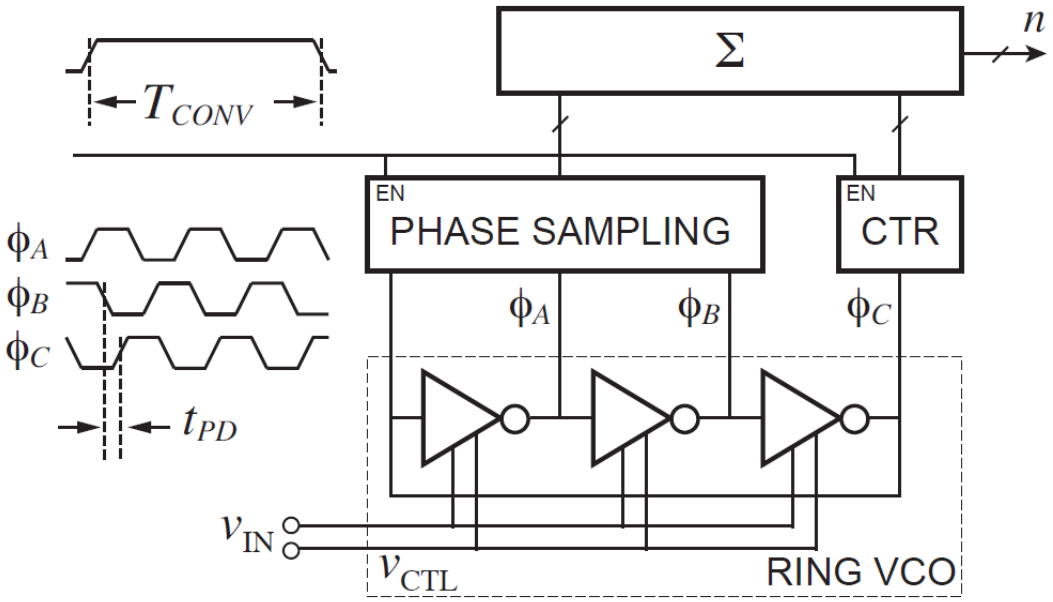


Figure 2.10: Simplified VCO-based ADC structure

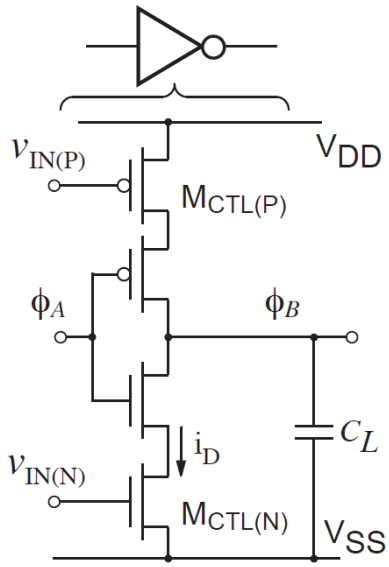


Figure 2.11: Delay stage of VCO-based ADC

of VCO. In other words, the control voltage changes the gate propagation delay t_{PD} or transition speed of phases in the VCO. The frequency-to-digital converter is essentially a phase measurement circuit: The ring phases ϕ_A, ϕ_B, ϕ_C are sampled and the number of oscillator cycles is counted. The result at the end of the conversion time is the ADC output n, the total number of phase transitions observed in the time T_{CONV} . From the timing diagram in Figure 2.10, the output n is given by

$$n = \frac{T_{CONV}}{t_{PD}}. \tag{2.13}$$

in which t_{PD} is the ring oscillator gate delay time.

To develop an expression involving v_{IN}, we need the v-to-t_{PD} characteristic of the delay stage, which depends on the particular circuit architecture being used. Figure 2.11 shows a current-starved gate architecture, which we will use in this example. We can approximate the MOSFET drain current i_D during the gate delay time t_{PD} as

$$i_D = G_m \cdot v_{IN}, \tag{2.14}$$

where G_m is the slope of the V_{GS} to I_D relationship or transconductance for the M_{CTL} MOSFETs.

To develop the simplified operating principle of the VCO-based ADC, we temporarily make the assumption that this relationship is linear. Techniques for mitigating the effects of nonlinearity will be described in the following chapters. For a conservative approximation of the gate propagation delay, we apply charge conservation as the gate output drives the total load capacitance C_L over a peak-to-peak voltage swing of V_{DD} in time t_{PD} with drain current i_D as

$$i_D = \frac{C_L \cdot V_{DD}}{t_{PD}} \tag{2.15}$$

Combining equations (2.13), (2.14) and (2.15) gives for the output n,

$$n = \frac{T_{CONV}}{C_L/G_m} \cdot \frac{v_{IN}}{V_{DD}}. \tag{2.16}$$

From (2.16), it indicates that the output n is proportional to the input v_{IN}, which is the desired relationship for an ADC, building a bridge from continuous analog input to discrete digital codes.

2.3.4 VCO-Based ADCs Nonideality

Although VCO-based ADCs have attractive advantages and properties, there still exist a critical challenge to be solved before it can be used as a good ADC. Ideally, to build a workable ADC, we need to have a linear relation between analog input and digital output. In VCO-based ADC, the bridge from analog to digital is between

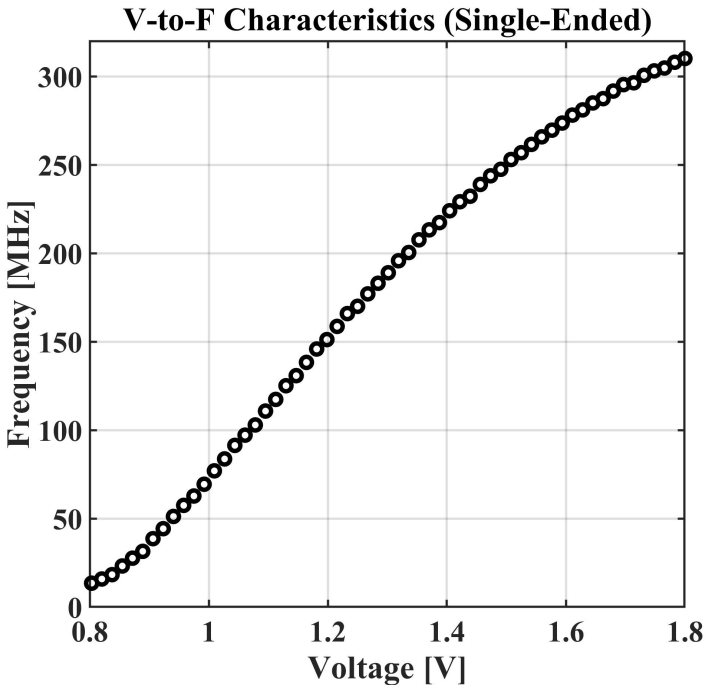


Figure 2.12: V-to-f characteristics of 17-stage single-ended VCO in 180nm CMOS

VCO controlling input and VCO ouput phases, which is called voltage-to-frequency characteristics. However, the v-to-f relationship is not linear as we expected.

Figure 2.12 shows the voltage-to-frequency characteristic of a 17-stage single-ended VCO, the structure of this VCO is demonstrated in section 3.2.2. When VCO input voltage is tuning from 0.8 to 1.8 V, the VCO's output frequency is increasing in a nonlinear form. This phenomenon can also be expressed in a mathematical way, the VCO gain K_{VCO} is no longer a constant, and it should be considered as value which will change according to the tuning region,

$$K_{VCO}(f) = \frac{\partial f_{VCO}}{\partial V_{in}}. \tag{2.17}$$

This directly induces the nonlinearity into the ADC system, which highly degrades both static and AC performance of VCO-based ADC. Thus, mitigating the nonlinearity of v-to-f characteristics becomes the most important and challenging work in designing a good VCO-based ADC. In the later chapters, detailed design consideration of how to solve this problem will be demonstrated including both analog and digital techniques.

2.3.5 Fundamental Limits on VCO-Based ADCs

This section investigates the fundamental limit achievable for efficiency of the VCO-based approach [10]. In the following subsections, the quantitative analysis of the VCO-based ADC will be demonstrated.

Quantization Limited

Before we start to analyze the VCO-based approach, we adopt the following assumptions: (1) VCO jitter is negligible over conversion time T_{CONV}. (2) Linearity

has been corrected such that ENOB is limited by unshaped quantization noise. (3) Power consumption is dominated by VCO. (4) Time resolution is limited by the gate delay t_{PD} (no phase interpolation used)

An advantage of analyzing the simplified VCO-based ADC of Figure 2.10 with these assumptions is that we can develop a simple expression for the fundamental limit on the efficiency figure-of-merit, defined as

$$FOM = \frac{Power}{f_s \cdot 2^{ENOB}}. \tag{2.18}$$

To find this limit we need the power dissipation, sampling frequency, and effective resolution.

The worst-case maximum power dissipation will occur for the maximum current i_D, which from (2.15) will occur for the minimum gate delay $t_{PD(MIN)}$:

$$i_{D(MAX)} = \frac{C_L \cdot V_{DD}}{t_{PD(MIN)}} \tag{2.19}$$

which gives for $Power$,

$$Power = i_{D(MAX)} \cdot V_{DD} = \frac{C_L \cdot V_{DD}^2}{t_{PD(MIN)}}. \tag{2.20}$$

To find the limit of best achievable performance for the effective-number-of-bits expression 2^{ENOB}, we will use the ADC resolution n_{MAX}, the maximum number of counts n. Using $t_{PD(MIN)}$ in equation (2.13) we have,

$$n_{MAX} = \frac{T_{CONV}}{t_{PD(MIN)}}. \tag{2.21}$$

Combining equation (2.18), (2.20), (2.21),

$$\frac{C_L \cdot V_{DD}^2}{t_{PD(MIN)}} \cdot T_{CONV} \cdot \frac{t_{PD(MIN)}}{T_{CONV}} \Rightarrow FOM = C_L \cdot V_{DD}^2 \tag{2.22}$$

The result in equation (2.22) also indicates benefits of the VCO-based ADC approach:

(1) Scaling friendly: the ADC efficiency F.O.M. from (2.22) is the same as the energy per gate transition figure of merit, the reduction of which is the goal of digital scaling. Also, from equation (2.21), we see that achievable resolution in a given conversion time T_{CONV} will improve as $t_{PD(MIN)}$ decreases at smaller geometry nodes.

(2) Reconfigurable resolution: equation (2.21) we see that resolution can be increased simply by allowing more time T_{CONV} for the conversion process. No reconfiguration of analog circuitry is required.

(3) Efficiency F.O.M. is independent of resolution: from (2.22) we see that the efficiency figure-of-merit depends only on the supply voltage V_{DD} and the gate load capacitance C_L. So if the ADC is reconfigured for a different point in the speed-resolution trade-off space, the efficiency F.O.M. should be unchanged.

VCO Jitter Limited

In general, the quantization noise portion of SNDR can be improved by increasing ADC resolution. The ability to trade an increase in conversion time for improved resolution without reconfiguring hardware is an advantage of this technique. As resolution increases, however, at some point noise performance will be limited by some fundamental aspect of the analog to digital conversion process. In this case, since we have moved A/D conversion into the time domain, a possible limit on ADC noise performance is oscillator jitter. Jitter-limits the ability of the VCO to measure time accurately. In this section we drop the assumption of negligible VCO jitter;

this can be considered a time domain approach complementary to the frequency domain approach of [15].

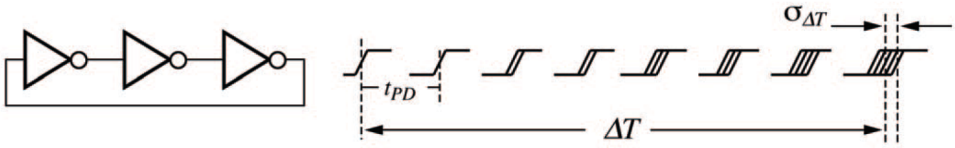


Figure 2.13: Jitter accumulation in ring oscillaor

Figure 2.13 shows how jitter accumulates over a time interval ΔT. For a VCO-based ADC, the time delay ΔT in equation (2.23) is the conversion time T_{CONV}. Since the jitter in separate delay stages is caused by noise in different MOS devices, we can assume the added noise in each stage to be independent, so the RMS jitter increases proportional to the square root of delay [39, 40] and can be characterized by a figure of merit κ,

$$\sigma_{\Delta T} = \kappa\sqrt{\Delta T}. \tag{2.23}$$

The difficulty with noise occurs as we attempt to increase resolution n_{MAX} by lengthening T_{CONV} as indicated in equation (2.21). From equation (2.21) and Figure 2.13 we can see that the rms jitter of the VCO will also keep increasing. To find the contribution of jitter to ADC noise as T_{CONV} increases, we express the time error of in ADC counts n by dividing by the gate delay t_{PD},

$$\sigma_n = \frac{\sigma_{\Delta T}}{t_{PD}} = \frac{\kappa\sqrt{T_{CONV}}}{t_{PD}}. \tag{2.24}$$

For a worst-case analysis, the maximum noise occurs with the minimum t_{PD} in (2.24); using equation (2.21) gives,

$$\sigma_n = \frac{\kappa\sqrt{T_{CONV}}}{\sqrt{t_{PD}}} \cdot \sqrt{n_{MAX}}. \tag{2.25}$$

In reference [40], κ for a full swing CMOS delay is approximated as,

$$\kappa = 2\sqrt{\frac{kT}{i_D V_{DD}}} \tag{2.26}$$

in which k is Boltzmanns constant and T is absolute temperature. Combining equations (2.26), (2.25), (2.15),

$$\sigma_n = 2\sqrt{\frac{kT}{C_L V_{DD}^2}}\sqrt{n_{MAX}}, \tag{2.27}$$

showing that the noise contribution due to jitter will increase as the square root of the target resolution n_{MAX}.

To find the resolution at which the noise effect of jitter becomes appreciable, define n^*_{MAX} as the resolution for which the rms noise due to jitter is just equal to the ADC quantization noise of $1/\sqrt{12}$ LSB,

$$1/\sqrt{12} = 2\sqrt{\frac{kT}{C_L V_{DD}^2}}\sqrt{n^*_{MAX}} \Rightarrow n^*_{MAX} \approx \frac{C_L V_{DD}^2}{48kT} \tag{2.28}$$

Evaluating (2.28) with C_L and V_{DD} suggesting that that jitter will limit noise performancefor resolution above log2 (6590) = 12.7 bits.

Further interpreting the result in (2.28) shows:

(1) The dimensionless quantity n^*_{MAX} in (2.28) is the result of a ratio of two energies: the energy of the switching event $C_L V_{DD}^2$ relative to the random thermal energy kT. This makes intuitive sense since the noise effect of jitter will be of less concern as the random thermal energy is a small fraction of the switching energy which determines the time domain behavior of the delay stage.

(2) As switching energy per transition decreases with scaling, we expect the resolution n^*_{MAX} to decrease, meaning that jitter will affect performance at lower ENOB.

The limit in equation (2.28) was developed assuming a thermal noise jitter model. For channel lengths less than 100nm, it is likely that 1/f effects [41] and/or excess noise [42, 43] will contribute additional noise beyond the prediction of equation (2.28).

2.4 Nonlinearity Mitigation Techniques

To design a good VCO-based ADC, the major work is to solve the nonlinearity of v-to-f characteristics of VCO. As shown in Figure 2.14, we can see the organization of architecture choices for VCO-based ADC.

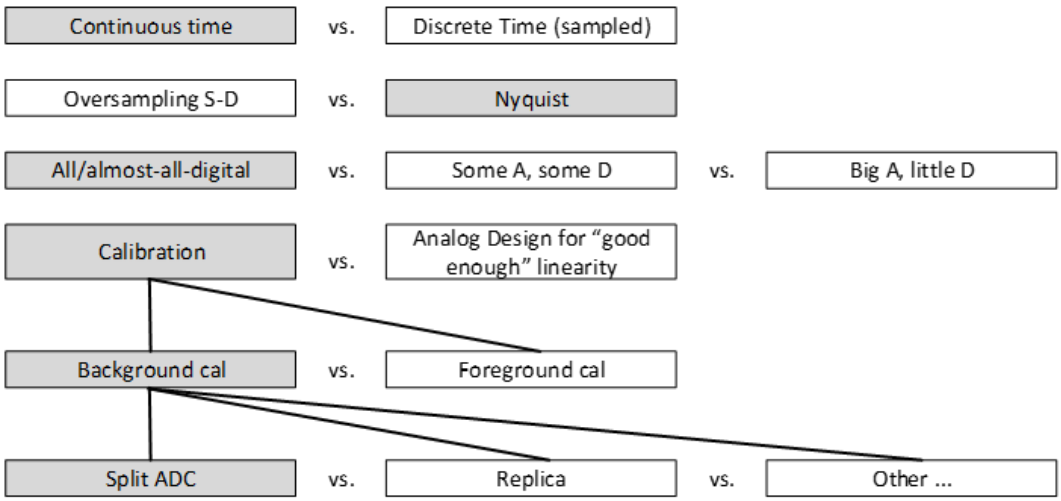


Figure 2.14: Organization of architecture choices for VCO-based ADC

Firstly, there are two choices for input ports: continuous or discrete time input. For the continuous time input, the average value of signal during every conversion time is used as the input voltage to ADC. While for the discrete time input, a fixed value of signal at every conversion time is sampled as the input voltage.

According to the sample rate of the ADC, the ADC architecture can be divided into oversampling ADC or Nyquist ADC. Then, the detailed structure of VCO-based

ADC can be further decided. In the third row of Figure 2.14, "All/almost-all-digital" block represents that the VCO-based ADC has only very simple analog circuitry, such as single-ended ring VCO, and the most other parts are digital circuits. "Some A, some D" and "Big A, little D" blocks represent the ADC structure has some complex analog circuits, while digital circuits are simplified. For example, the most popular VCO-based ADC architecture nowadays is the oversampling ADC by using a $\Sigma\Delta$ loop to mitigate the nonlinearity of VCO v-to-f characteristics, which locates in the "Some A, some D" or "Big A, little D" section.

Furthermore, to deal with the VCO v-to-f nonlinearity, designing a "good enough" linear analog circuit might not be a smart and visionary option in the future sub-nanometer CMOS era. Calibration methods including background or foreground calibration are a relatively hot topic. Although the foreground cal is much easier, it does not provide real-time adaption to the temperature changing or other unpredictable process influence. Therefore, the background calibration technique is more attractive.

Under the background calibration category, there are also various kinds of calibration algorithms. For example, the "split ADC" based calibratable system which is used in this work is one of the most achievable and efficient algorithms [1]. Additionally, the algorithm using replica VCOs published in [5] is another very interesting and excellent work.

In this work, in order to hit the highly digital, low power consumption target, the following architecture is investigated: continuous time input, Nyquist sampling, almost all digital background calibration assistant system based on "split ADC" structure.

2.5 Split ADC Concept

In this section, the general idea of how "split ADC" based calibratable system is discussed. Figure 2.15 shows the concept of the general "split ADC architecture [1–3]: the ADC is split into two channels, each converting the same input v_{IN} and producing individual output codes x_A and x_B. The average of the two outputs is the ADC output code x. The background calibration signal is developed from the difference Δx between codes x_A and x_B and is completely transparent to converter operation in the output code signal path.

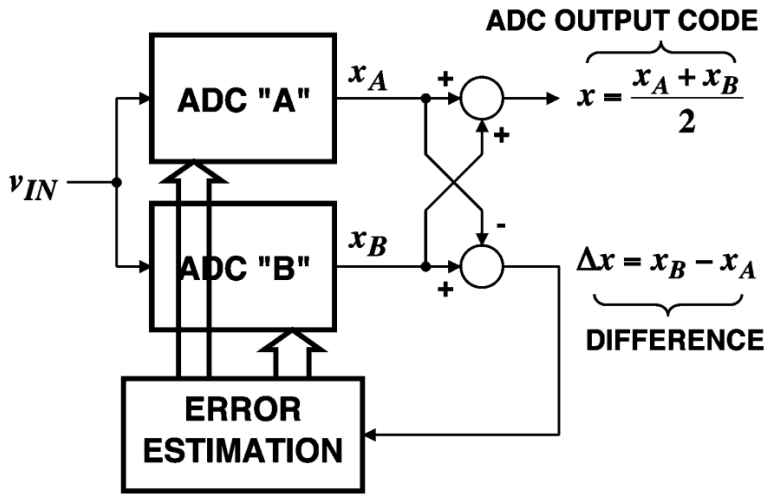


Figure 2.15: Split ADC architecture

If both ADCs are correctly calibrated, the two outputs will agree and the difference Δx will be zero. In the presence of nonzero differences, the pattern of disagreements in Δx can be examined in an error estimation process to adjust calibration parameters in each ADC, driving the difference and the ADC errors to zero. In the split ADC approach calibration and correction are performed entirely in the digital domain, without interrupting normal ADC operation.

The "split ADC" was originally developed by professor John McNeill and his

collaborators Michael Coln and Brian Larrivee from Analog Devices [1–3]; a similar technique was developed independently by Moon and Li at Oregon State [44]. An indication of quality and impact of this technique is the broad range of published work using the "split ADC" approach developed by other investigators.

While the original work [1–3] was for an algorithmic (cyclic) ADC architecture, work has also been published applying the "split ADC" concept to architectures such as pipeline [44–64], oversampling [65–69], flash [70], folding [71], interleaved [72, 73] and SAR [74–77] ADCs. Over a broad range of the speed-resolution-architecture ADC trade-off space, there are examples in the literature in which the "split ADC" approach enables fast digital background self-calibration. The proposed work would be the first to extend the advantages of the "split ADC" approach to VCO-based ADCs. As will be shown later, this will enable drastically improved power efficiency by moving all calibration and correction into the digital domain, allowing VCO-based ADCs to fully realize the promise of nanometer scaled CMOS while avoiding the performance trade-offs and disadvantages of the calibration techniques.

Chapter 3

Analog Front-End Design

This chapter describes the analog front-end design of the proposed VCO-based ADC. Firstly, the system architecture of the proposed VCO-based ADC is introduced. Secondly, the analysis and choice of the VCO architecture is discussed. After that, the customized phase measurement circuits including ripple counters, uniform phase detector and phase decoder are demonstrated in detail for this specific VCO-based ADC application. The differential output buffers which are used to improve the driving ability of VCO outputs are also shown. In the end, the overall architecture of the whole analog front end will be demonstrated

3.1 System Architecture of the proposed VCO-Based ADC

First of all, this section gives a whole picture of the overall VCO-based ADC system architecture of the proposed VCO-based ADC. The system block diagram is shown in Figure 3.1. For the calibratable "split ADC" architecture [1–3], there are two identical ADC channels in the systems, called ADC "A" and ADC "B". They are

absolutely the same. In the each ADC "A" or ADC "B", there are two identical VCOs in a differential configuration. This structure is chosen because the differential signal path can suppress even order harmonics (as shown in Figure 5.11). In other words, it can improve the linearity of v-to-f characteristics and reduces the workload for the digital calibration system.

Under the single-ended condition, the analog input range is from 0.8 to 1.8V and the corresponding VCO frequency is roughly from 10 to 310 MHz (To have a large input signal swing, the maximum input is equal to the supply voltage. The minimum input is the one that can just make the ring start to oscillate). Thus, for the differential configuration, the differential input is from -1 to +1 V, and the differential VCO frequency is from -300 to $+300$ MHz. The input dynamic range increases as well as the v-to-f linearity. However, the power consumption is doubled. In fact, there is a trade-off between power and ENOB. This depends on the specific application, but in general, to make Walden F.O.M remain the same or smaller, the ENOB should increase at least 1 bit if the power is doubled. Thus, the question is if the differential configuration for v-to-f linearity improvement could bring another 1 bit ENOB, or if it can improve calibration speed, chip area and other system performance.

In fact, one of the important reasons to choose the differential configuration in this work, is to improve the input signal bandwidth. As discussed in the last chapter, it will be faster for calibration to reach convergence if the input signal is slow, as it means the LUT can built in a more continuous way without skipping lots of LUT locations. Thus, if the differential signal path enables a relatively linear relationship between voltage and VCO output frequency, under the same calibration time requirement as a single-ended system, the input signal bandwidth can be larger.

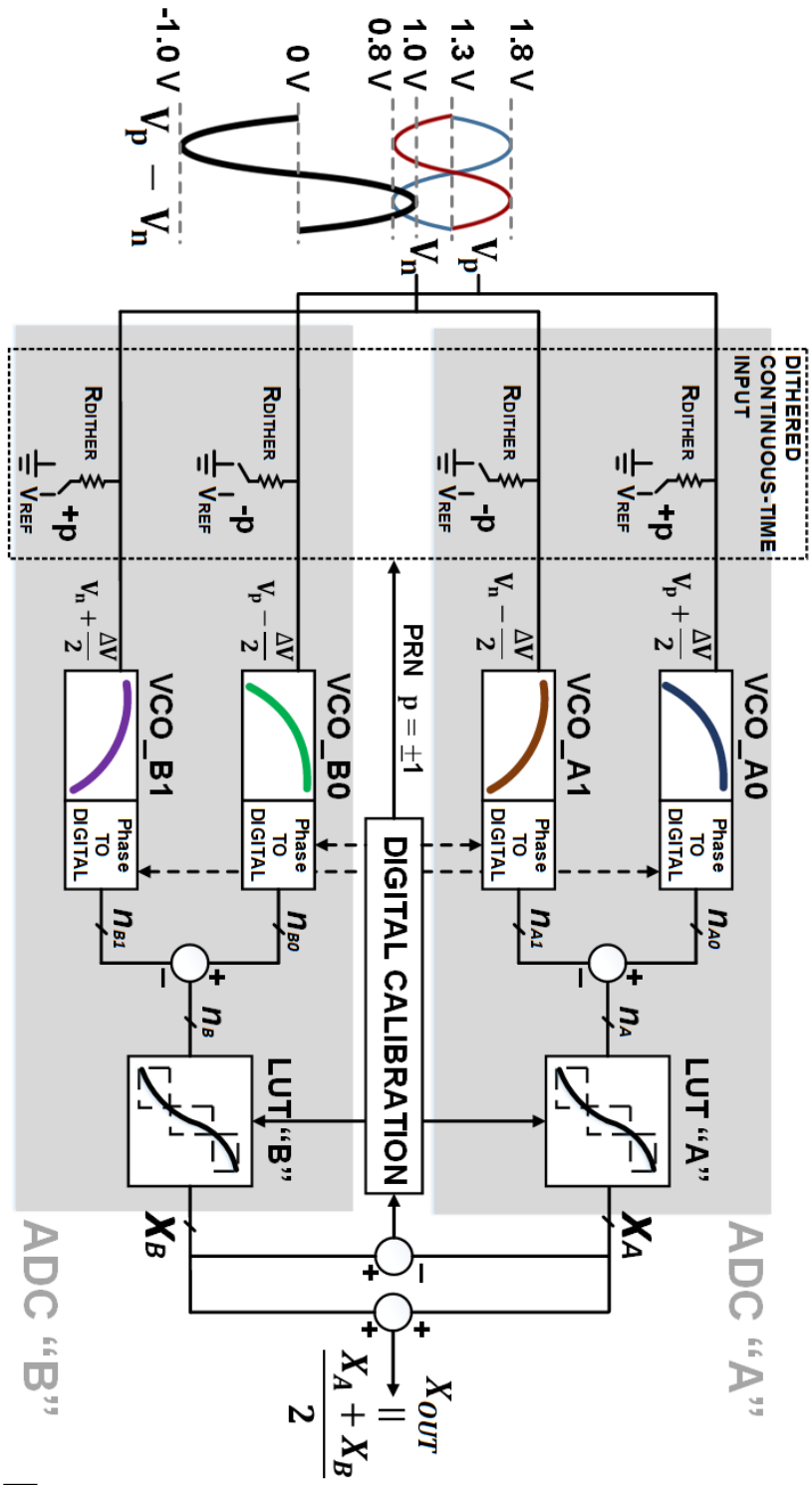


Figure 3.1: System Architecture of the VCO-based ADC

3.2 VCO Choice for VCO-Based ADC

This subsection mainly discusses the choice of VCO architecture for the usage in VCO-based ADC. It will start from introduction of different types of VCOs with analysis of their advantages and disadvantages. Then, according to this proposed VCO-based ADC work's requirement and application scenarios, the main reason of choosing a ring VCO will be given. In the end, the implementation of the VCO on chip will be shown in detail.

3.2.1 LC Oscillator

LC oscillator is a well-known VCO topology that is famous for its performance in phase noise [78]. The oscillation frequency is determined primarily by its resonant LC tank. And the LC oscillator could be tuned by varing an element value, such as a voltage controlled capacitance like varactor. However, in this work, the LC oscillator is not chosen for the following reasons,

(1) This VCO-based ADC work is focusing on the low power consumption target. Though the LC oscillator could provide very good jitter performance, it is not suitable for this application condition because the large bias current required by LC oscillator (which cost lots of power) goes against the original goal of this project. The ring oscillator's moderate jitter performance is good enough for this VCO-based ADC design requirement.

(2) LC oscillator requires much larger chip area than a ring oscillator because of its resonant LC tank. In this highly digital system, the die area is supposed as small as possible and the large integrated inductors are to be avoided.

(3) The tuning in LC oscillator by varying capacitance of varactors is not as straightforward as in ring VCO, which need only control its input gate voltage.

The indirect tuning method of the LC oscilator will induce some unexpected design issues. For example, due to this work's calibration algorithm requirement, small dithers need to be added at the input ports, which can be a big problem if using an LC VCO.

(4) Another advantage of this novel highly digital VCO-based ADC is its scalling friendly merit. As CMOS technology is moving to nanometer very fast from 65nm to 28nm or even 7nm these days, simple analog circuitry with digital systems supporting will be the main trend. It is a big requirement that the design could be re-usable in different CMOS technologies, however, LC oscillator due to its resonance reason, it usually needs to be re-designed when moving to new technology.

3.2.2 Ring Oscillator

Another common type of oscillator is the ring oscillator. Figure 3.2 shows the simplest 3-stage, single-ended ring oscillator. The delay stage is usually created by a CMOS inverter as shown in Figure 3.3. Its oscillation frequency is determined by the propagation delay t_{PD} of each delay stage. Thus ring VCO output frequency can be expressed as,

$$f_{vco} = \frac{1}{2 \cdot N \cdot t_{PD}} \tag{3.1}$$

where N is the number of ring VCO stages.

However, due to different characteristics of NMOS and PMOS (for example, difference of carrier mobility of electron and holes, mismatches in two transistors) in the inverter as shown in Figure 3.3, the gate delay actually could be separated into two different delays, t_{rise} and t_{fall}. The paths through PMOS and NMOS are not ideally the same, therefore, the equivalent path resistance of PMOS and NMOS will

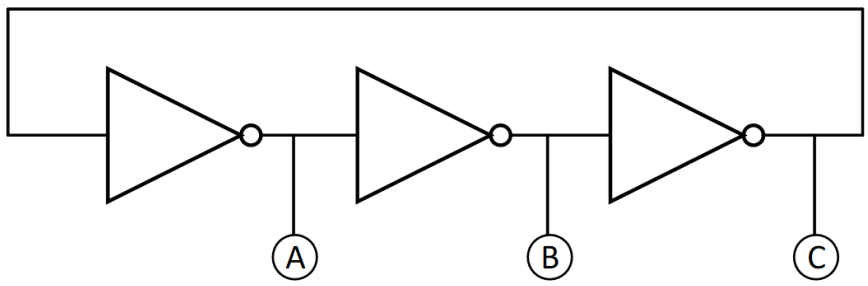


Figure 3.2: 3-stage single-ended ring oscillator

be slightly different. t_{rise} represents the delay time when the inverter transitions from a low voltage level to a high voltage level. t_{fall} represents the delay time when the inverter transitions from a high voltage level to a low voltage level. Though it seems not a big problem as seen in the VCO's output frequency, it is a critical challenge if the VCO-based ADC requires a high resolution. The specific design of uniform phase sampling will be explained in the later sections.

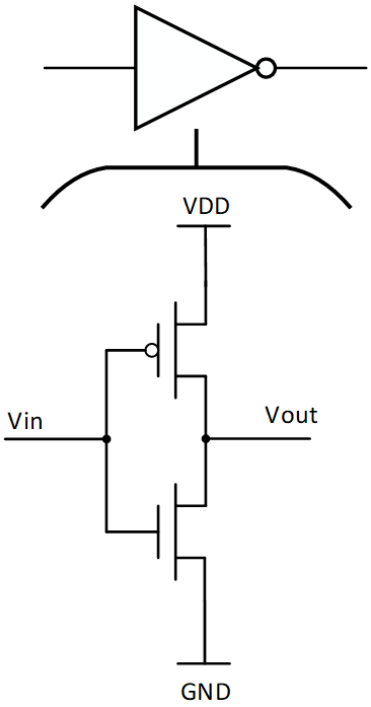


Figure 3.3: Delay stage of 3-stage single-ended ring oscillator

Now, if we differentiate gate delay t_{PD} by using t_{rise} and t_{fall}, the output fre-

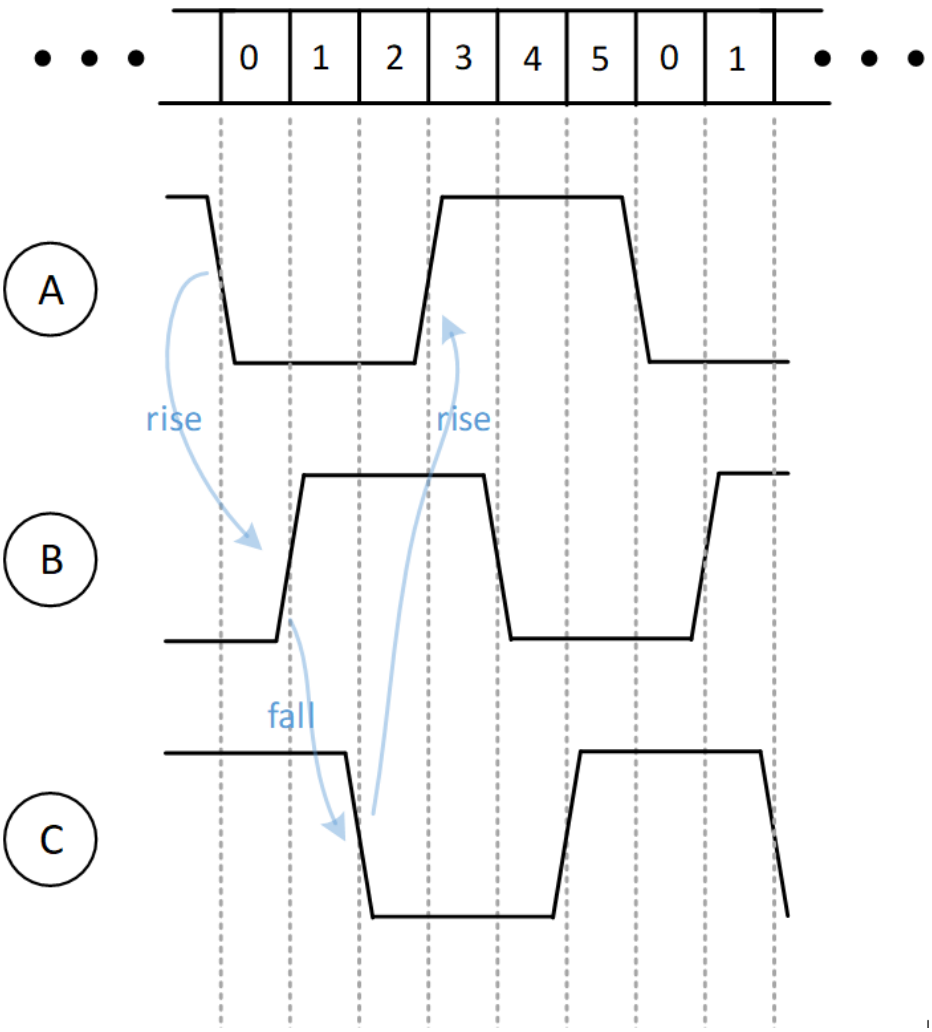


Figure 3.4: Waveform of 3-stage single-ended ring oscillator

quency of ring VCO can be expressed as,

$$f_{vco} = \frac{1}{N \cdot (t_{rise} + t_{fall})}. \tag{3.2}$$

To explain the transition condition more carefully, an example of a 3-stage, single-ended ring VCO output waveform is displayed in Figure 3.4. The A, B, C nodes represent a short period of the output waveform of the 3-stage, single-ended ring VCO output as shown in Figure 3.2. The numbers 0, 1, 2, 3, 4, 5 represent

the total transition phase conditions in one clock period. As we can see in Figure 3.4, when the clock waveform transfers from a low voltage level in node A to a high voltage level in node B, there is a rising time delay between node A and node B, thus a phase condition 0 is created. Similarly, when the clock waveform transfers from a high voltage level in node B to a low voltage level in node C, there is a falling time delay between node B and node C, thus, a phase condition 1 is created. Thes rising and falling time gate delays will appear alternatively and finally create a full clock period.

In this specific 3-stage ring oscillator case, a full clock period T_{clock} can be expressed as,

$$T_{clock} = t_{rise} + t_{fall} + t_{rise} + t_{fall} + t_{rise} + t_{fall} \tag{3.3}$$

and it could be summarized as,

$$T_{clock} = 3 \cdot t_{rise} + 3 \cdot t_{fall} = n \cdot (t_{rise} + t_{fall}) \tag{3.4}$$

where N = 3, and $T_{clock} = 1 \ / \ f_{VCO}$

There may provoke a question: why a full clock is built by $2 \cdot N$ gate delays? The most straightforward answer is the clock period represents a pattern. However, this is actually related to an interesting overtone issue which will be discussed in the next subsection about how to choose number of stages in ring oscillator.

Compared with the LC oscillator, ring oscillator may not have better jitter performance [40], however, the advantages of ring oscillator are matching the proposed VCO-based ADC design:

(1) Low power: because ring oscillator only consumes power when a transition happens, there are no other passive components or large static currents burning power.

(2) Small area: ring oscillator is consist of only simple inverters which are built by PMOS and NMOS, no large integrated inductors or capacitors are needed. This significantly reduces die area requirements.

(3) Easy Tuning: there are quite a lot of ways to tune the frequency of a ring oscillator. For example, by controlling the transition current or supply voltage, which are both direct and simple.

(4) Scaling friendly: ring oscillators are much simpler to be re-designed than LC oscillators when moving from one CMOS technology to another. It typically involves changing the ratio of widths and lengths in PMOS and NMOS to make them stay balanced. What's more, for the case in the ultra small nanometer CMOS technology, ring oscillator is also very easy to implement by creating a positive closed loop of inverters without consideration of other complex questions in LC oscillator.

Thus, according to the requirements of VCO-based ADC design, a ring voltage controlled oscillator is chosen over an LC oscillator. In the following subsection, design considerations and real chip implementation is explained.

Number of Stages in a Ring VCO

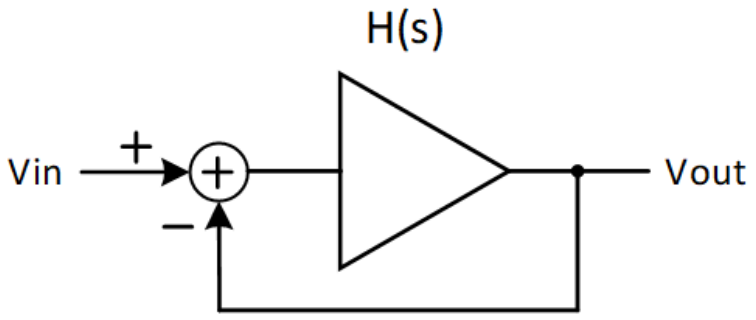


Figure 3.5: Feedback system

An oscillator is actually a unity gain negative feedback system, as displayed in Figure 3.5. The relation of output and input can be given as,

$$\frac{V_{out}}{V_{in}}(s) = \frac{H(s)}{1 + H(s)}. \tag{3.5}$$

However, the difference between an oscillator and a normal negative feedback system is that its loop gain satisfies two conditions,

$$|H(s)| \geq 1 \tag{3.6a}$$

$$\angle H(s) = 180^\circ \tag{3.6b}$$

It can be explained that oscillators could start to oscillate when the forward gain is large enough and the the total phase shift is actually 360°. Thus, the first thing for considering to design a simple ring oscillator is the number of stages, which directly decides the gain and phase shift in oscillator.

For single-ended ring VCOs, the number of delay stages should be odd numbers like 3, 5, 7, 9, 11, etc. An even number of delay stages could make a single-ended VCO output locked, which means at every output of the ring VCO, the voltage level is stable. On the other hand, for differential ring VCOs, even and odd number of delay stages could both be used because the differential output of two VCOs can be swapped to avoid locking problem.

There are some other practical considerations beyond theoretical reasons. For example, the ring VCO will usually be followed by a digital counter to collect the number of clock periods. However, a small number of delay stages will make the VCO output frequency very large, as defined in equation (3.2): when N is small, f_{VCO} will be a large value. It could cause a very big problem since the digital circuits may not pass static timing analysis (STA). A very high output frequency may unecessarily increase the difficulties in the following digital circuit design.

However, to design a good high speed and high resolution VCO-based ADC, counting the number of clock period is not enough; all of the phase transitions should be collected. This will be a challenge; the standard cell of D flip flops may not meet the requirement, and customized fast flip flops are needed to handle tiny setup and hold time window. One important point to remember is, if we are counting transition numbers in a constant time at one same output node of a ring oscillator, result is constant,

$$n = \frac{T_{CONV}}{t_{PD}} \tag{3.7}$$

where T_{CONV} is the constant counting time (or, the conversion time), and t_{PD} is the gate delay.

Another significant and interesting design issues related to number of delay stages in ring VCO is called "overtone". For the single-ended ring oscillator case, the issue usually happens for an odd, composite number of delay stages. Recall the formula which be used to calculate VCO output frequency in (3.2), when the VCO architecture and CMOS technology are the same (gate delay of VCO is the same), then, if a 3-stage ring VCO's clock period is T_clk_3, another 9-stage ring VCO's clock period should be $T\_clk\_9 = 3 \cdot T\_clk\_3$. In Figure 3.6, the waveforms of 9-stage and 3-stage ring oscillator are shown, the period of 9-stage ring oscillator is 3 times of that in 3-stage.

According to this special case analysis, the general case could be derived. Assuming there is a odd composite number $k = p \cdot q$, then the output clock period of two ring oscillators (everything is the same but number of ring stages), with k stages and p stages, will have the relationship as follows,

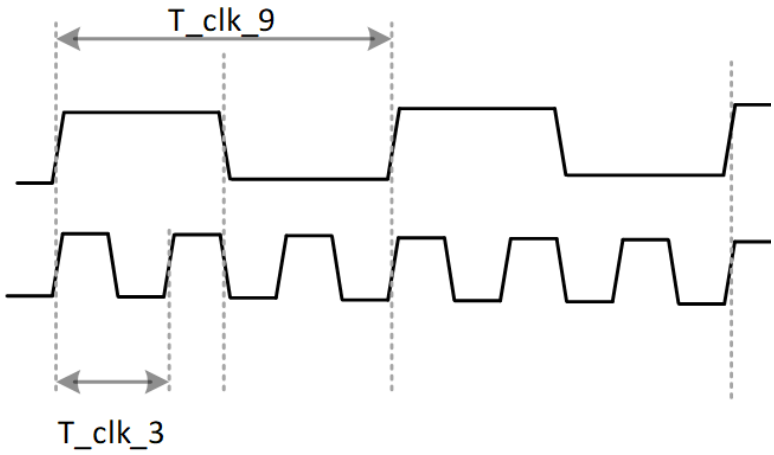


Figure 3.6: Waveform of 3-stage and 9-stage single-ended ring osc

$$T\_clk\_k = \frac{k}{p} \cdot T\_clk\_p = q \cdot T\_clk\_p. \tag{3.8}$$

According to the work in [79] and the above analysis, a ring oscillator with composite number of delay stages may have higher harmonic components besides its own fundamental frequency. These unexpected harmonics will introduce noise into the whole ADC system. Therefore prime numbers of delay stages are preferred. Thus, in this chip implementation, the number of delay stages in ring oscillator is 17. Because 17 is a relatively large stage number which do not require very high speed counter to collect the output transitions and it is also a prime number to avoid overtone issues.

Tuning Method in Ring VCO

To use a ring oscillator in VCO-based ADC, we need to add some tuning techniques to change the simple oscillator to a voltage controlled oscillator, whose gate delay transition time could be tuned with an input signal. As shown in equation (3.2), for a constant conversion period, to change the digital output counts n the only choice

in ring oscillator is to change the gate delay, t_{PD}.

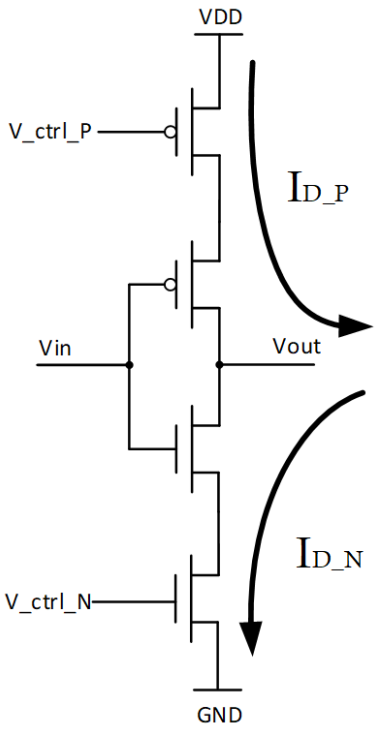


Figure 3.7: Delay stage of current starved VCO

There are several ways to control the gate delay of a ring oscillator. The first one is to control the current strength which charges and discharges the load of each inverter. This topology is referred as the current-starved VCO. Figure 3.7 shows a simplified view of a delay stage of a current-starved VCO. The two additional MOSFETs work as current sinks and sources, and are controlled by the input voltage. Thus, the transient drain current when the transition happens could be limited. In other words, the inverter is starved for current as it could not get enough current from supply voltage. The total sum of the output capacitor of the first stage and the input capacitor of the second stage can be modeled as a load capacitor C_L. The time it takes to charge and discharge C_L depends on the drain current I_{D_P} and I_{D_N}, respectively. These time delays are actually the same as t_{rise} and t_{fall}.

Another similar option is to change the gate delay by controlling the supply

voltage. The supply voltage not only influences the charging drain current but also the output swing of the oscillator. However, these kinds of two combination tuning will introduce nonlinearity into the ADC system and is not usually used.

Instead of changing the charging and discharging current strength, the time delay is also dependent on the load capacitance. Thus, another method is to vary the propagation delay by changing the output capacitive load C_L. This variation in load capacitance can be realized by one or more voltage dependent capacitors called a varactor. A reverse-bias PN junction can serve as a varactor which has capacitance [80],

$$C_{var} = \frac{C_0}{(1 - V/\Phi_B)^m} \tag{3.9}$$

where C_0 is a zero bias capacitance, V is the applied voltage, Φ_B is the built in voltage of the junction, and m is a value typically between 0.3 and 0.5 [80]. Adding a varactor diode increases the load capacitance, directly affecting the tuning range of the VCO. Additionally, the nonlinear relationship between controlled voltage and the varactor diode capacitance translates into the nonlinearity of the VCO.

Ring VCO Architecture

Until now, the details design of VCO's gate delay, tuning method are discussed. The next step is to analyze the applied ring VCO architecture while considering the benefit of the whole ADC system performance. There are many ways to implement a ring VCO such as single-ended, true differential, source-coupled and other architecture. By upholding the design target of low-power, small area and scaling friendly, the pseudo differential ring VCO architecture is used on chip as we can see in Figure 3.8.

In Figure 3.9, the delay stage of is displayed in detail. The two middle gray

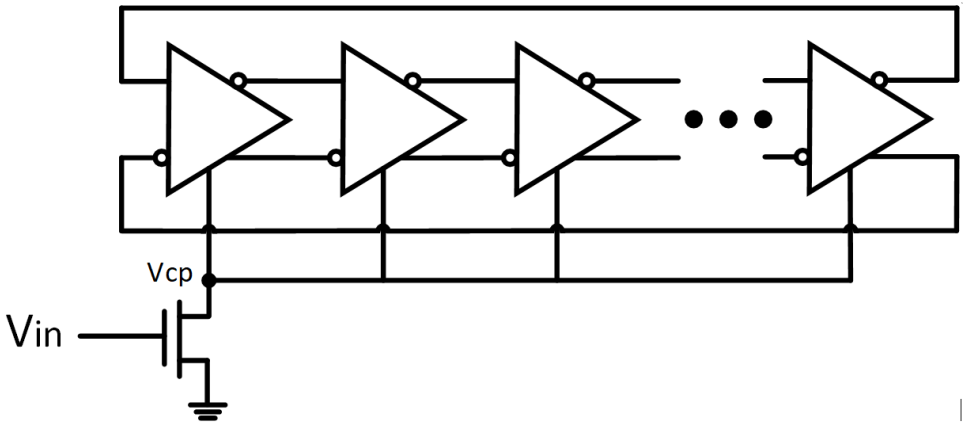


Figure 3.8: Pseudo differential ring VCO

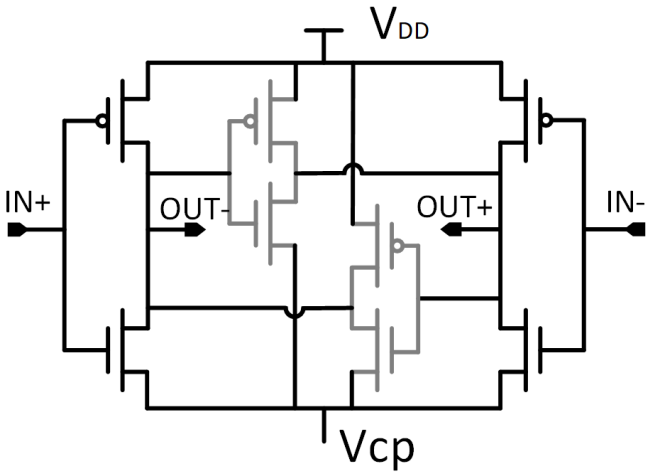


Figure 3.9: Delay stage of pseudo differential ring VCO

inverters are used to lock the pseudo differential ring oscillator in a 180° phase difference, and the size of these two inverters should be the small. Because if their size is comparable with the two outside large inverters, they will influence the VCO's output frequency. Actually, the middle two inverters are usually designed in the smallest size and the main inverters for the ring oscillation are in a proper larger size. Thus, the total power consumption of the pseudo differential ring VCO will be small. A large tail transistor is used to limit the current of the whole oscillator. It could improve the noise performance of the ring oscillator by blocking it from

supply noises.

The most important reason to use the pseudo differential ring VCO is the need to have uniform phase quantization by checking only rising edge or only falling edge of the VCO transition. Recall the equation that represents the VCO's frequency in (3.2), because of the different characteristics of NMOS and PMOS, and the other mismatches in the charging and discharging paths, the rising delay time t_{rise} and the falling delay time t_{fall} are different. If we use single-ended ring VCO, we need to check the rising and falling time alternatively, which can induce large noises because of non-uniformed phase quantization. However, since the two output signals of the pseudo differential VCO are 180° out of phase, we can check only the falling time or only the rising time by sampling the two ring oscillator outputs alternatively. This method will be explained more carefully in section 3.3.2.

3.3 Phase Measurement Circuit

To build a VCO-based ADC, besides the design of VCO as an integrator, a phase measurement circuit is also required to transfer the VCO frequency or phase into discrete digital codes. In this section, the detailed design of the phase measurement circuit is described. The phase measurement circuit is consist of digital counter,

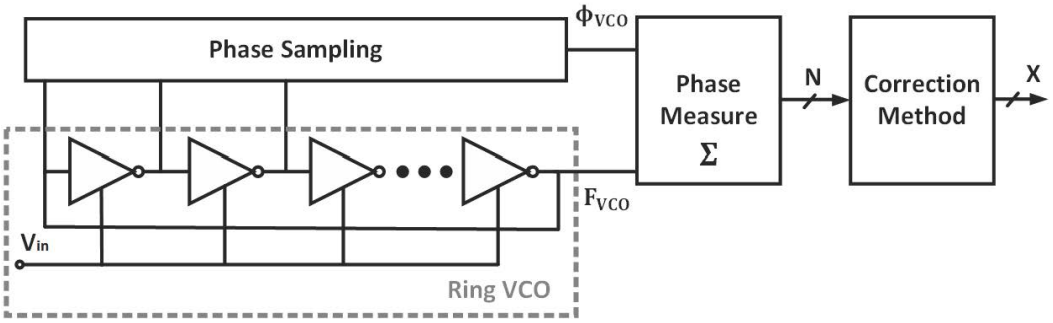


Figure 3.10: Simplified VCO-based ADC structure

phase detector, phase decoder and output buffer.

Figure 3.10 shows a basic VCO-based ADC structure. The phase measurement circuits measures the ring VCO's phase through the combination of VCO's output frequency and VCO's phase. There are other measurement choices available depends on the application requirement. For example, for the requirement of large input bandwidth, the phases of VCO are sampled within one VCO clock cycle and there is no digital counter needed to count the VCO's output frequency. [28]

In this work, the VCO's output frequency and VCO's phase are measured at the same time. This combination method allows the phase measurement circuits to have greater range and precision.

3.3.1 Digital Up Counter

The digital counter is used to transfer the phase to digital codes by measuring the VCO's output frequency. Though it seems that the design could be as simple as a digital counter with several lines of Verilog HDL code, there are actually some very important design techniques for this low power consumption VCO-based ADC application. There are two main counter architectures to discuss, synchronous and asynchronous counters

Synchronous Counter

Synchronous counters are synchronous because the clock input of the flip-flops are all clocked together at the same time with the same clock signal. Due to this common clock pulse all output states change simultaneously. With all clock inputs wired together there is no inherent propagation delay (as we can see in Figure 3.12). Synchronous counters are sometimes called parallel counters as the clock is fed in parallel to all flip-flops. The inherent memory circuit keeps track of the counters

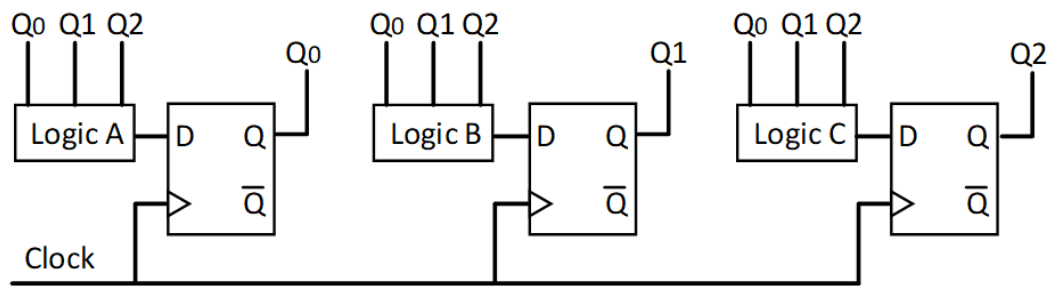


Figure 3.11: Synchronous 3 bit counter using D Flip Flops

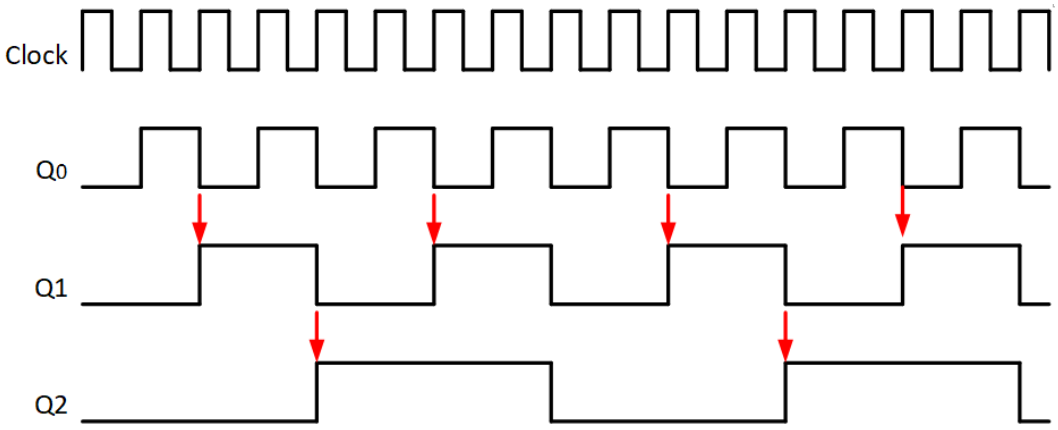


Figure 3.12: Waveform of synchronous 3 bit counter

present state. The count sequence is controlled using logic gates. As shown in Figure 3.11, there are some additional blocks "logic A", "logic B" and "logic C". These blocks add complexity and power consumption.

Asynchronous Counter

Asynchronous counters can easily be made by solely using D-type flip-flops. They are called asynchronous counters because the clock input of the flip-flops are not all driven by the same clock signal. Each output in the chain depends on a change in state from the previous flip-flops output. Asynchronous counters are sometimes called ripple counters because the data appears to "ripple" from the output of one flip-flop to the input of the next. A drawback of an asynchronous counter is when

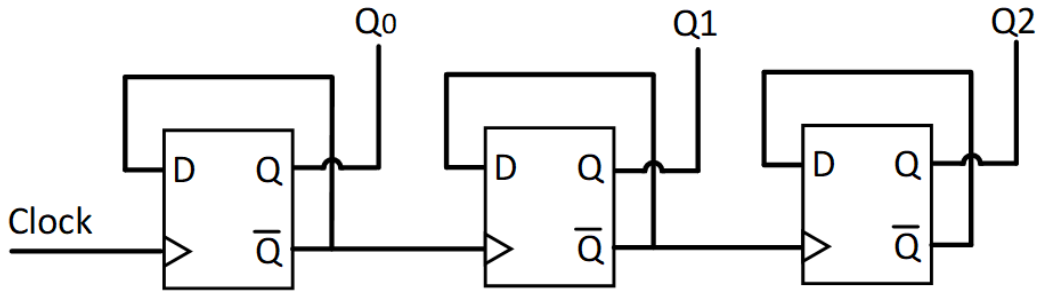


Figure 3.13: Asynchronous 3 bit counter using D Flip Flops

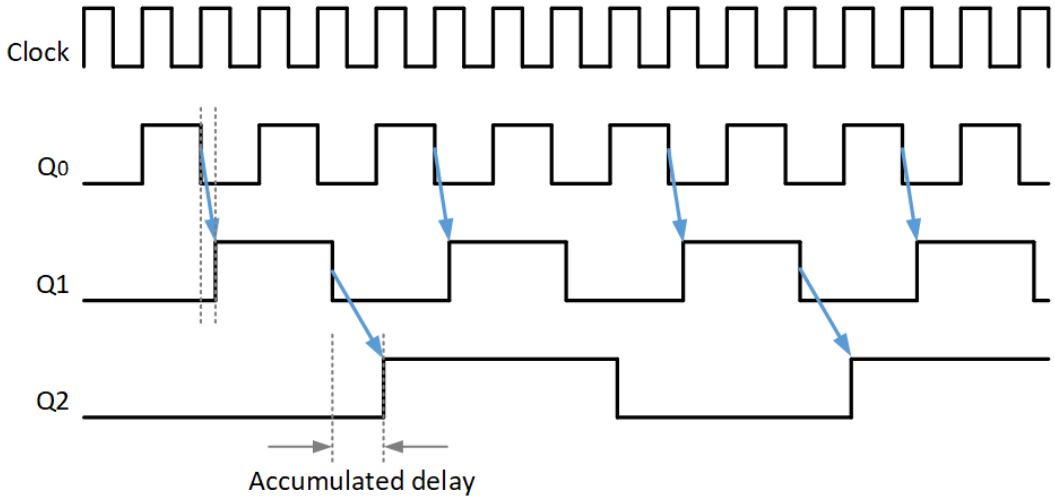


Figure 3.14: Waveform of asynchronous 3 bit counter

counting a large number of bits, propagation delay by successive stages may become undesirably large, as we can see in Figure 3.14.

In this proposed VCO-based ADC, the asynchronous ripple counter is chosen, because it can provide super low power consumption comparing with the synchronous counter. Even though the synchronous counter could provide the synchronous clock edge toggling, we do not actually care about this advantage in our VCO-based ADC application. We only care if the counter could provide the VCO phase transition number correctly with low power consumption. Therefore, asynchronous counter is a better candidate here. However, we do need to account for some time to let the ripple counter settle down before we try to sample the transition number.

3.3.2 Phase Sampler & Decoder

The phase sampler block is used to capture the phase state of VCO. In combination with the VCO frequency, the phase state within one VCO clock period can further increase the ADC resolution. In other words, instead of just counting the clock edges of VCO output, we can now also collect the transition phases of VCO.

The detailed design of phase sampler should be carefully considered. In Figure 3.15 shows an ideal case of pseudo differential VCO output swing, it looks symmetric and beautiful. However, in Figure 3.16, the realistic swing represents lots of undesirable features. For example, the maximum swing is not as high as VDD, there might even be a case that the output swing is lower than the digital threshold. What's more, since the rising and falling time of VCO clock edges are varying, if we keep checking rising and falling edges like in a single-ended cases, the phase quantization will be nonuniform.

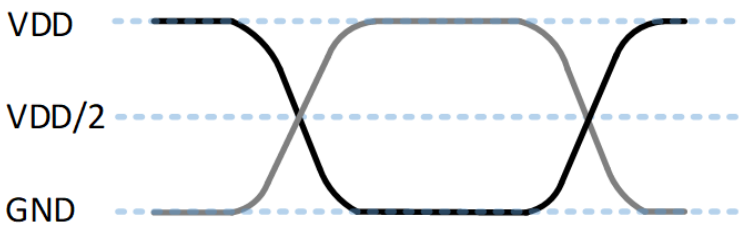


Figure 3.15: Ideal pseudo differential swing

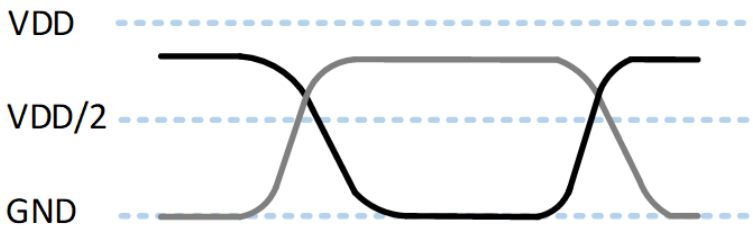


Figure 3.16: Practical pseudo differential swing

Firstly we will add a output buffer after the VCO to solve the low swing issues,

the output buffer will be discussed in a later section 3.3.3. For the nonuniform quantization problem, we adopt a method from [5], as we can see an simplified 5-stage pseudo differential ring VCO case in Figure 3.21. By checking only falling edges of this 5-stage VCO (in fact, we check the two VCOs alternatively in this differential architecture), the 10 states is uniformly quantized. The digital implementating of 17-stage pseudo differential VCO is shown in Figure 3.22.

In addition, for high speed applications, there is another serious issue called "Metastability". The VCO may oscillate at very high speed, however, the standard cell D flip flop in digital library can not operate fast enough for sampling. In other words, the setup and hold time window of the standard D flip flop is longer than the gate delay, thus the standard D flip flop can not detect the exact correct phase states. We need to design a customized fast flip flop to sample the phases, and at the same time, we may need design other digital decision circuits for metastability recovery.

The phase decoder block is used to combine the VCO's output frequency and VCO's phase information together. In one conversion period, phase sampler samples twice, at the beginning and the end respectively (to collect the difference of phase transferring in one conversion). An asynchronous counter (ripple counter) is used to count the clock cycles. The total number of transitioned phases N_{total_phase} can be expressed as

$$N_{total\_phase} = 2 \cdot N_{stage} \cdot C_{clock} + D_{phase\_difference}, \tag{3.10}$$

where N_{stage} is number of VCO stages, C_{clock} is the counted number of VCO clock cycles and $D_{differential\_phase}$ is the phase code difference between initial state and final state in one conversion period.

3.3.3 Output Buffer

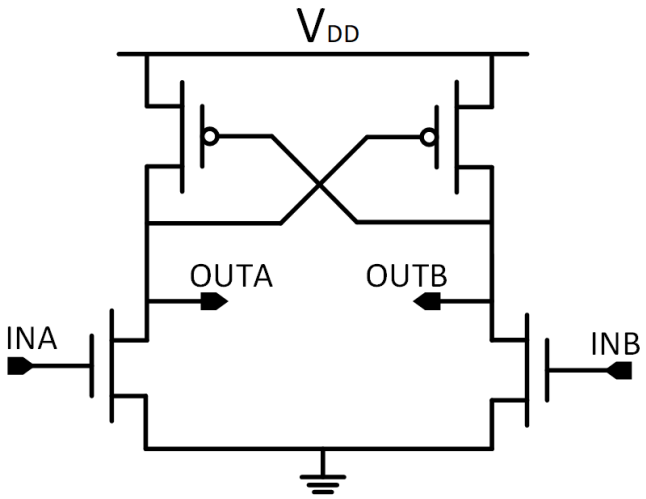


Figure 3.17: Output buffer of pseudo differential ring VCO

Before integrating the ring VCO and the phase measurement circuits, we need to consider their influence to each other. Directly connecting VCOs with phase measurement circuits may induce unpredictable problems. For instance, the driving ability of the output of VCO may be too weak to drive the ripple counter, or sometimes VCO's output swing is smaller than digital threshold voltage. To avoid these issues, differential analog buffers are applied between the two blocks to enhance the driving strength of the VCO output and stretch the small output swing to full scale. In Figure 3.17 shows a simple schematic of differential buffer.

However, there is no free lunch. The output buffer will also increase the power consumption. Thus, we my want to use switches to open and close buffers to save power. Also, the buffers take some more area on chip.

3.4 Input Ports

Input ports are another important design block in this VCO-based ADC design. As we know from Chapter 2, this work is enabled by "Split ADC" architecture, the first part of the background calibration is to add small dither to each channel, allowing the two identical ADC "A" and ADC "B" to be split. Here comes the critical problem: because the proposed background calibration is started with the split two ADC channels, any nonlinearity in the added dither will not be "seen" by the calibration algorithm. In other words, the errors that induced from the input ports can not be reduced or mitigated by calibration algorithm. However, the differential configuration adopted in each channel can alleviate this problem.

3.4.1 Discrete-Time Input

Sample and hold (S&H) circuits are traditional analog circuits for ADC input signal sampling. However, due to the requirements of this proposed VCO-based ADC limitation, classical S&H circuits are not very suitable. For this specific application,

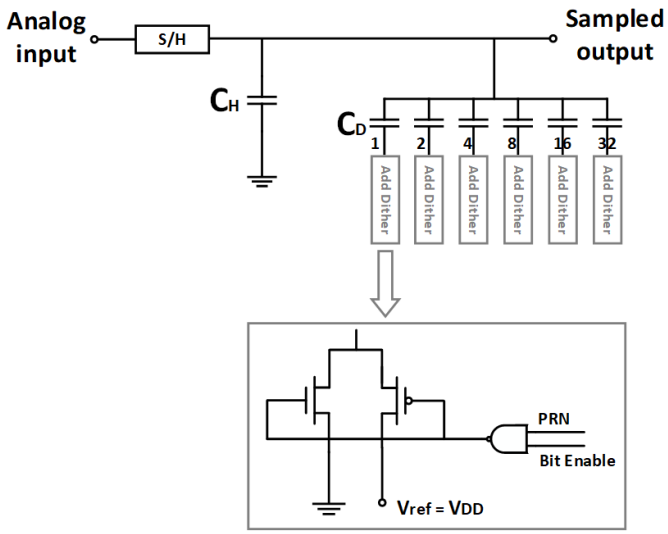


Figure 3.18: Sample & Hold with dither implementation

the background calibration needs input signal to train the LUT. Therefore, a wider input range could give better calibration performance. Additionally a wider input range can also increase the ADC's dynamic range. Thus, a bootstrapped sample and hold circuit is used in this work [81]. It can not only provide a full scale of input swing but also give a reliable environment for adding dither technique which is required for the digital background calibration.

In Figure 3.18, the block diagram of the S&H circuit is presented. C_H is the large hold capacitor and C_D are small capacitors for adding dither. The C_D capacitors are built in a binary form for fast capacitance varation. As we can see in the figure, capacitors in C_D is controlled by a switch. "PRN" is the pseudo random number created by digital circuits and "Bit Enable" is the digital control signal.

The S&H circuit operates as follows. Firstly, the switch of S&H is closed during the "sample" period, and at the same time, the C_D is connecting to ground,

$$V_{out,sample} = V_{in}, Q_{total} = (C_H + C_D) \cdot V \tag{3.11}$$

After some time, the switch is open in "hold" period, and at the same time, C_D is connecting to $V_(ref)$. According to "charge reservation", we can get,

$$V_{out,hold} \cdot C_H + (V_{out,hold} - V_{ref}) \cdot C_D = (C_H + C_D) \cdot V_{out,sample}. \tag{3.12}$$

Therefore, from equation (3.11) and (3.12), the final output voltage is,

$$V_{out,hold} = \frac{C_D}{C_D + C_H} \cdot V_{ref} + V_{out,sample} = \frac{C_D}{C_D + C_H} \cdot V_{DD} + V_{in}. \tag{3.13}$$

In the equation (3.13), the dither is successfully added into input signal. To avoid the integrated noise caused by small capacitor, we usually prefer a larger

capacitance, about 1pF. A larger capacitor can not only suppress $\frac{kT}{C}$ noise, but also reduce the droop voltage in the hold period. The smallest capacitance of hold capacitor C_H is limited by the ADC resolution,

$$\frac{kT}{C} \leq \frac{1}{2} \cdot \frac{V_{FS}}{2^N} \tag{3.14}$$

where k is the Boltzmann constant, T is the absolute temperature, V_{FS} is the full scale range of input voltage and N is the ADC resolution

3.4.2 Continuous-Time Input

Comparing with the discrete-time input, the continuous-time input is much simpler. Because it is just built with resisters as shown in Figure 3.19,

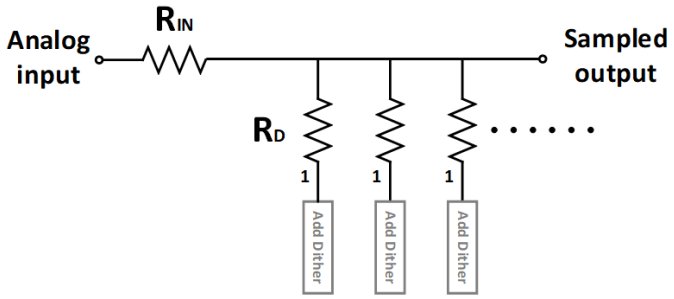


Figure 3.19: CT input with dither implementation

The "Add Dither" block is the same as the one in S&H circuit. The resistors R_D for adding dither is different from S&H. Instead of binary form, R_D is built by several parallel resistors. And R_{in} here is actually not added in real circuit, because it is the same as the output impedance of input source, which is normally 50 Ω.

3.5 System Architecture of Analog Front-End

Ideally, an excellent VCO design providing high linear v-to-f characteristic can directly build a good performance VCO-based ADC. However, traditionally extra analog circuits are introduced to achieve high linearity. This design direction increasing analog circuit complexity is not suitable in the nanometer CMOS technology. Instead of focusing on designing a highly linear VCO, the whole systems linearity is improved by applying a differential circuit structure. By arranging VCOs and input voltage direction, the system's equivalent v-to-f characteristic can be in a linear form [11].

To alleviate the v-to-f nonlinearity shown in Figure 2.12, two equivalent half circuits are constructed. The system architecture of analog front end is shown in Figure 3.20. V_p and V_n are driving these two VCOs in a differential mode. When V_p is rising from low to high voltage and V_n is falling from high to low (in 180-degree phase difference), the positive half circuit provides digital codes from small to large while the negative half circuit provides digital codes from large to small. Then these two inverted digital codes are subtracted from each other to get a new differential digital output code. The equivalent system's v-to-f characteristic is greatly improved since the differential structure removes the even order distortions and improves the noise performance. As we can see in Figure 5.11, the linearity is improved a lot from single-ended to differential configuration.

The VCO is designed as a pseudo-differential ring oscillator which can provide gate delay's rising or falling time alternatively. Moreover, by using the uniform quantization technique [5], the transition time to the phase decoder is uniformly spaced for any VCO frequency, which can mostly mitigate the nonlinear distortions.

As previously discussed in section 3.2.2 , short, odd number of stages like 5 or

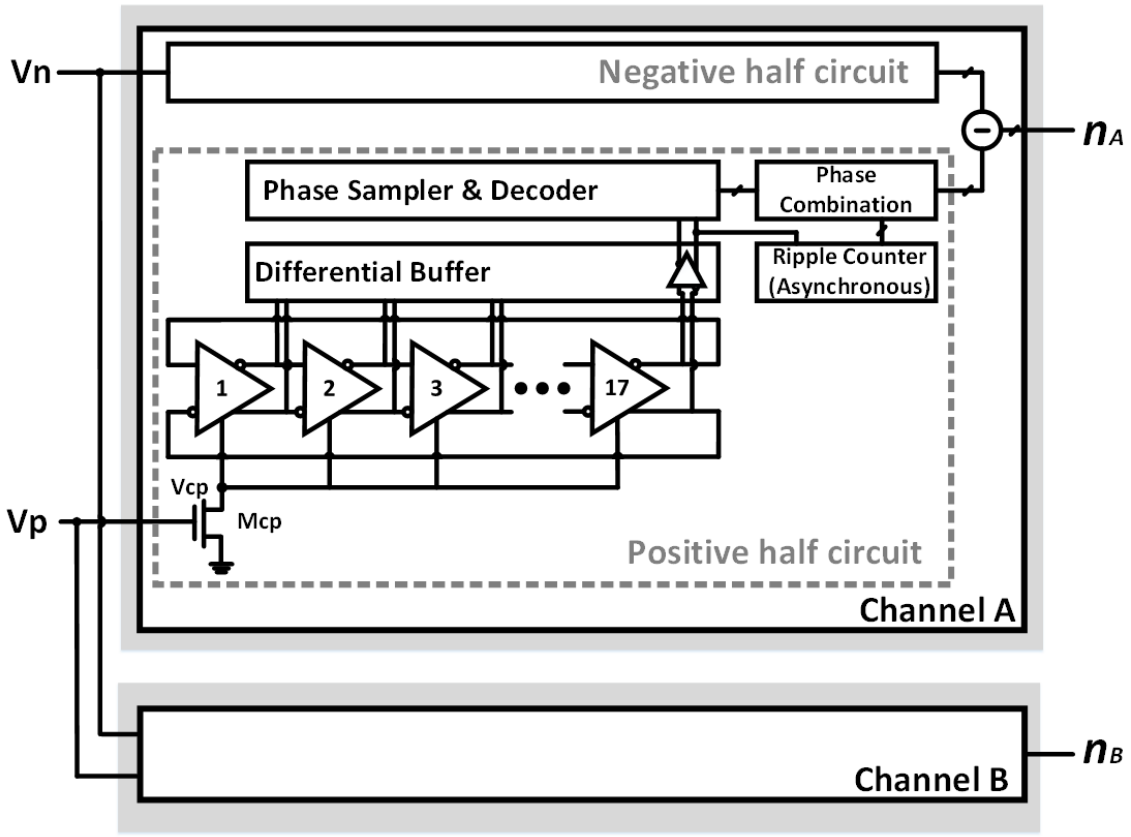


Figure 3.20: System architecture of analog front end

7 are not used, because the extreme high-speed VCO output cannot be collected correctly by the phase collecting circuit due to the technology timing constraints. A larger number of stages is also not recommended since it will waste area. Any non-prime odd number like 9 or 15 is not preferred since it sometimes induces overtone issues [79]. 17 is chosen as a compromise between speed, power and area.

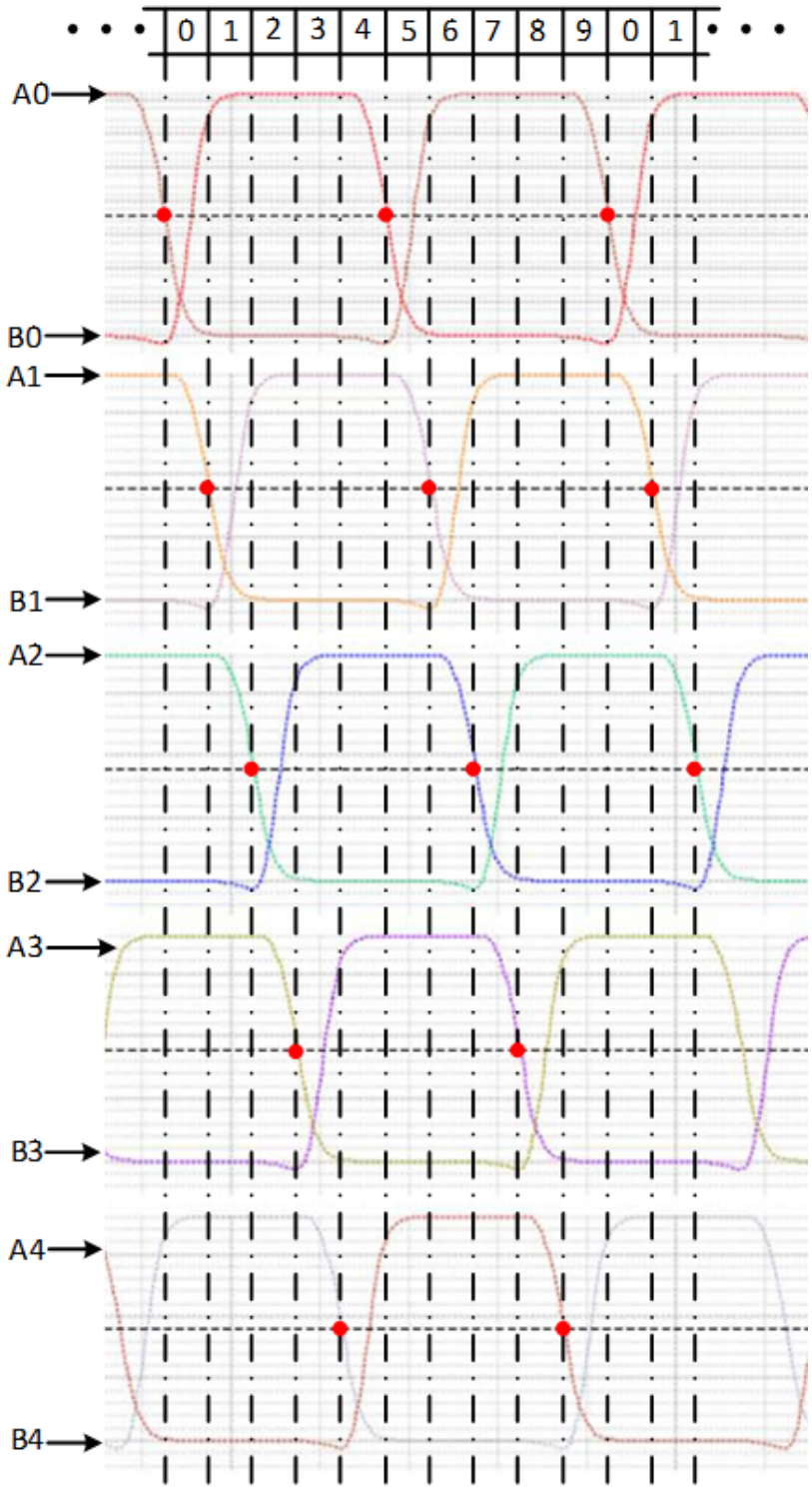
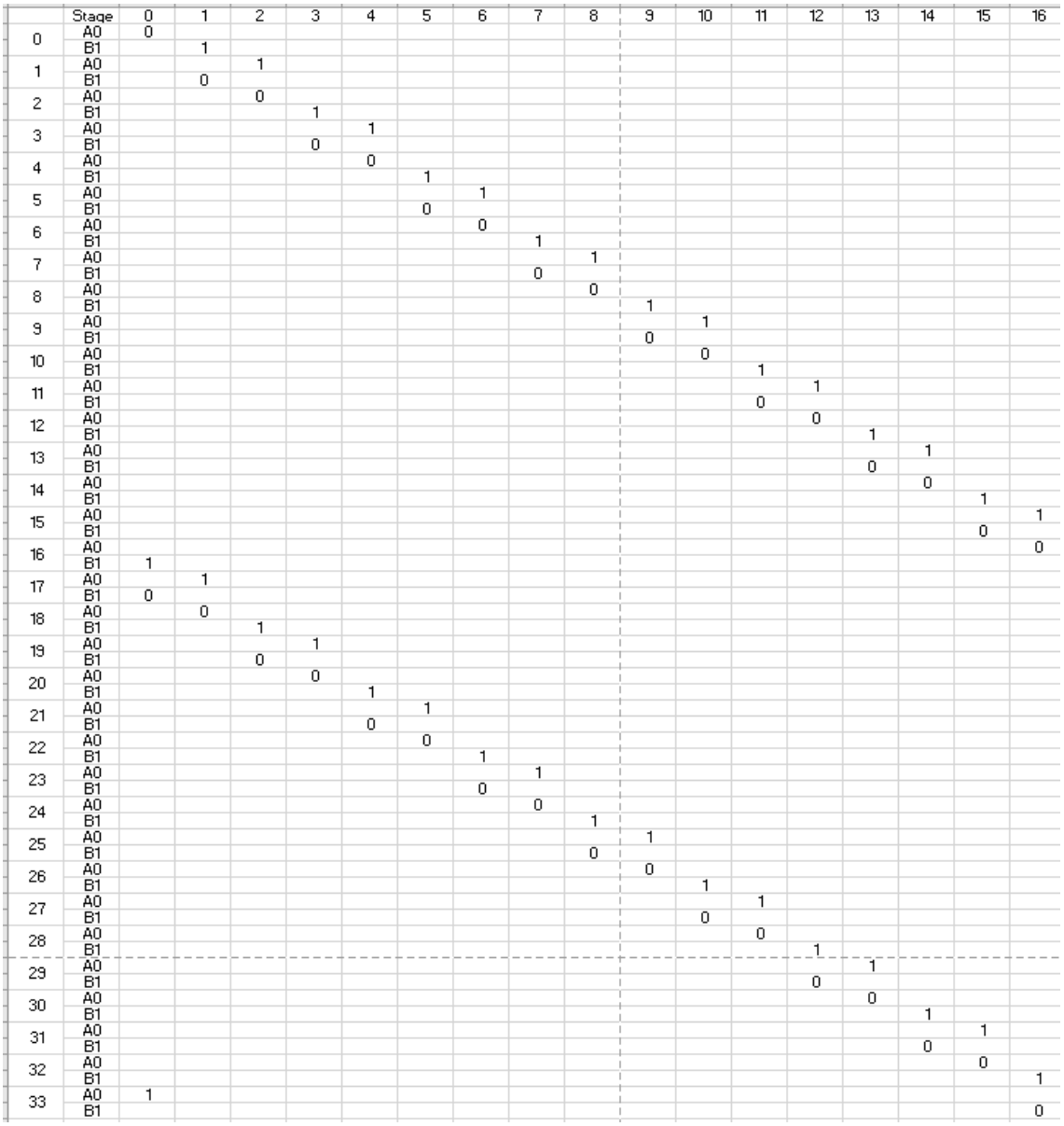


Figure 3.21: Uniform quantization of 5-stage pseudo differential ring VCO



	Stage	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	A0	0																
	B1		1															
1	A0			1														
	B1		0															
2	A0			0														
	B1				1													
3	A0					1												
	B1				0													
4	A0					0												
	B1						1											
5	A0							1										
	B1						0											
6	A0							0										
	B1								1									
7	A0									1								
	B1								0									
8	A0									0								
	B1										1							
9	A0											1						
	B1										0							
10	A0											0						
	B1												1					
11	A0													1				
	B1												0					
12	A0													0				
	B1														1			
13	A0															1		
	B1														0			
14	A0															0		
	B1																1	
15	A0																	1
	B1																0	
16	A0																	0
	B1	1																
17	A0		1															
	B1	0																
18	A0		0															
	B1			1														
19	A0				1													
	B1			0														
20	A0				0													
	B1					1												
21	A0						1											
	B1					0												
22	A0						0											
	B1							1										
23	A0								1									
	B1							0										
24	A0								0									
	B1									1								
25	A0										1							
	B1									0								
26	A0										0							
	B1											1						
27	A0												1					
	B1											0						
28	A0												0					
	B1													1				
29	A0														1			
	B1													0				
30	A0														0			
	B1															1		
31	A0																1	
	B1															0		
32	A0																0	
	B1																	1
33	A0	1																
	B1																	0

Figure 3.22: Phase decoder for 17-stage pseudo differential ring VCO

Chapter 4

Digital Background Calibration

A significant challenge for the VCO-based ADC architecture is to mitigate the effect of the nonlinear v-to-f characteristic of VCO. This section discusses a calibration method to improve the linearity of the VCO-based ADC. First, the lookup-table with linear interpolation method is discussed. The next section investigates the "split ADC" background digital calibration approach. Finally, the calibration and correction technique is summarized and the functional block diagram, as how the technique is implemented, is presented.

4.1 Clock Signal Generator

As a highly digital VCO-based ADC, before we could go to the lookup-table-based correction and background calibration step, it will be better to introduce the whole clock signal distribution of this ADC system, which could give a deeper understanding of how the ADC works by looking at the system timing diagram. The reason to create a signal generator on chip instead of off chip is to avoid the unexpected clock skew. If all the controlling clock signals are coming from off-chip, there will be a large clock skew which may degrade the ADC system's timing, especially in

the high speed condition.

The master clock of this VCO-based ADC generated by a crystal oscillator on a DE2-115 FPGA board. All of the other on-chip clock signals are created from this master clock [82,83]. It is very important rule to design the other clocks by only one master. If two or more master clocks are used, there will be a big design challenge to build a system with these asynchronous clocks.

The sub-clocks are generating based on the a counter. The main idea is different clock signals will have a positive pulse at some specific number of cycles of the master clock. Different clocks' timing difference can be handled by counting the number of cycles between pulses.

The simplified timing diagram is shown in Figure 4.1. The number of clock cycles are reduced to the whole timing flow. In the real system, the number of clock cycles are much larger. The master clock is 50 MHz, and the other clock signals can be adjusted according to different scenarios since this VCO-based ADC is reconfigurable. All of the digital clocks are described as follows:

- $MCLK$: Master clock (50 MHz) is from a crystal oscillator on the DE2-115 FPGA board.
- $S\&H\_CLK$: In sample period, ADC samples analog input. In hold period, ADC counts VCO transition gate delays.
- VCO_GATE: Gate window to block the VCO output signal, freezing the counter outputs while the ADC is the sampling analog input.
- VCO_1: Dummy VCO output oscillation for demonstration.
- OUT_1: VCO output covered by VCO_GATE signal.
- $INITIAL\_CLK$: Sample phase state in the beginning of every conversion.

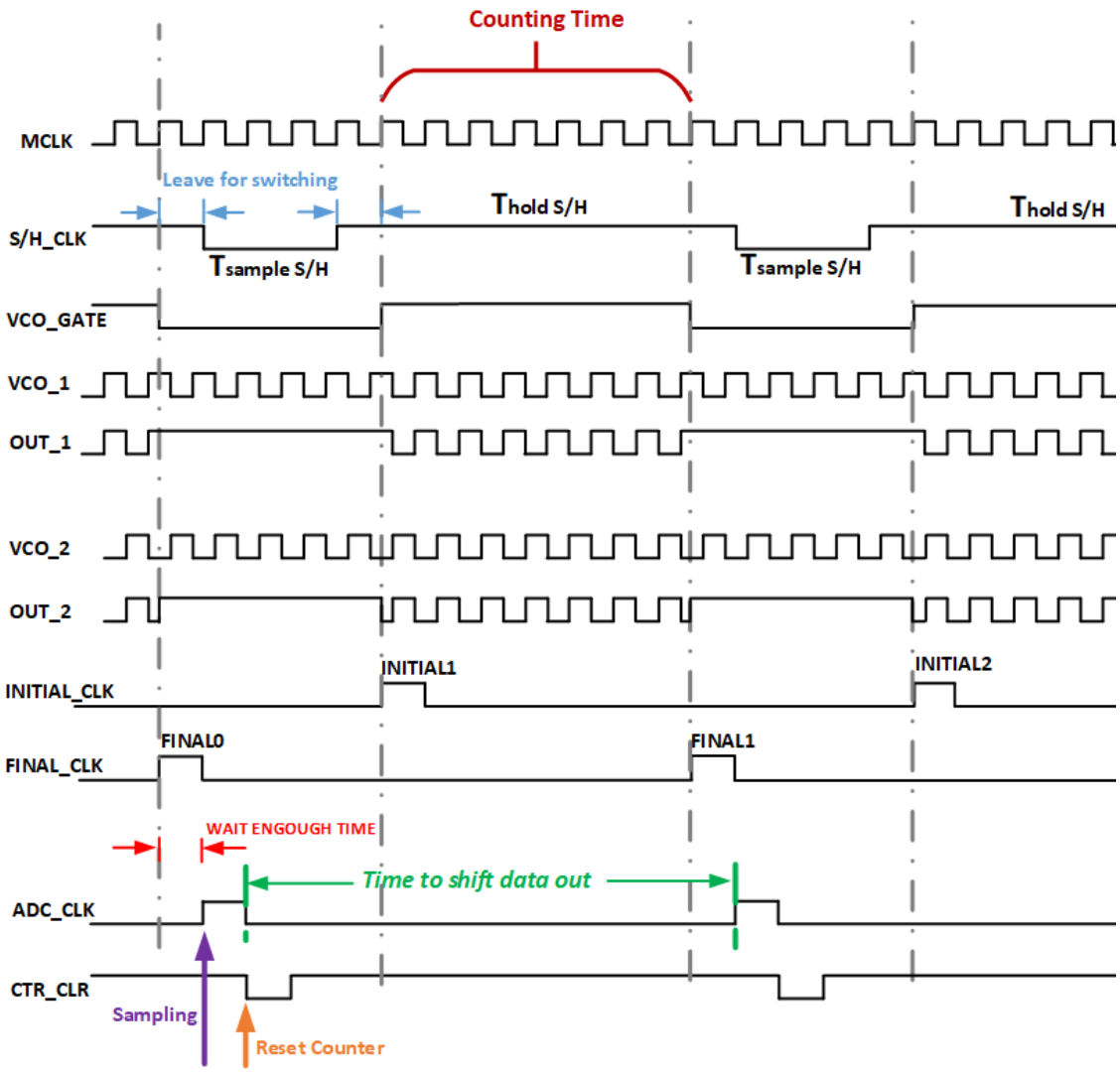


Figure 4.1: Timing diagram of the ADC system

- $FINAL\_CLK$: Sample phase state in the end of every conversion.
- ADC_CLK: Sample digital output code from counter in every conversion.
- CTR_CLR: Reset ripple counter after every conversion.

There are some significant design techniques:

- Before ADC_CLK triggers sampling the ADC data, there must be a wait period for ripple counters to settle.

- After $INITIAL\_CLK$ and $FINAL\_CLK$, there should be a copy version of these two clocks for the temporary registers to hold the phase state value.
- If the time between two adjacent conversion is too small to shift all the ADC data, we may try downsampling to take the data out.
- According to experience, the sample time is about one-fourth of holding time [14].

4.2 Lookup-Table-Based Correction

An ideal VCO-based ADC has the digital output code n proportional to the analog input voltage v_{IN}. However, as discussed in Chapter 2, the real relationship between the VCO frequency (or digital counts n) and the input v_{IN} is nonlinear. In order to correct the nonideal raw (means not corrected by digital circuits) output code n, a digital correction technique using a lookup-table (LUT) is proposed.

This LUT provides an additional transfer function between the uncorrected count and the final desired digital output code. Mathematically, this transfer function is inversely symmetric to the VCO v-to-f characteristic; thus if we could combine these

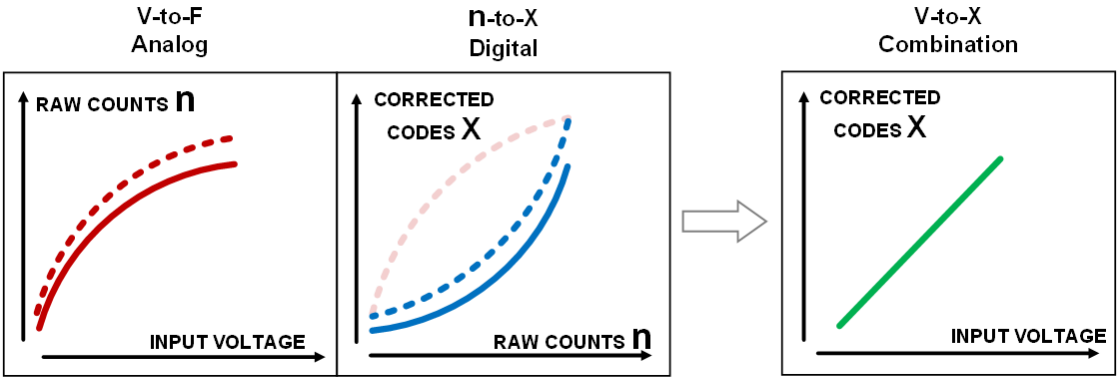


Figure 4.2: Transfer functions of nonlinear VCO-based ADC with LUT correction

two inverse mappings together, the final relationship between analog input voltage v_{IN} and the digital corrected code X will be in a linear form. Figure 4.2 illustrates this concept.

Since LUT is a discrete point by point mapping implementation of a transfer characteristic, it requires at least 2^N entries to fully cover the whole range of the N bit ADC uncorrected output. In order to reduce complexity of the digital circuit implementation, a combination of the LUT approach and linear interpolation is used [6, 84].

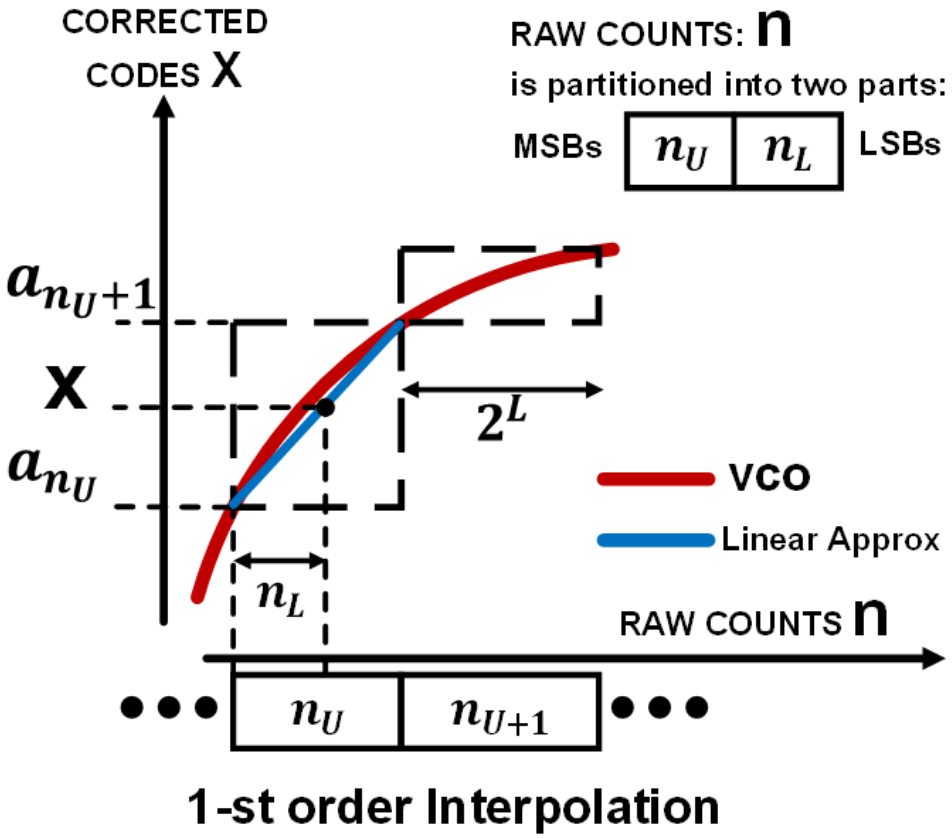


Figure 4.3: Lookup Table with 1-st order interpolation corretion

Figure 4.3 shows the basic idea and operation of the LUT. The uncorrected raw output code n is separated into two parts: a upper group of bits, MSBs and a lower group of bits, LSBs. The upper MSBs n_U determines the maximum number of

points M in the LUT: $M \leq 2^U + 1$. The MSBs n_U (U is length of n_U.) are used as index in LUT. The LSBs n_L (L is length of n_L.) are used to linearly interpolate between adjacent a_{n_U} and a_{n_U+1} coefficients in the LUT. Since the two adjacent LUT entries are separated by 2^L on the x-axis, the corrected output code x can be calculated as,

$$x_k = a_{n_U} + (\frac{n_L}{2^L})[a_{n_U+1} - a_{n_U}] \tag{4.1}$$

where x_k presents any corresponding corrected code.

The LUT can be implemented using digital registers and multiplexers. One multiplication is required for the linear interpolation. While for the division, instead of directly dividing 2^L, it can be simply realized by an L bit shift in radix point. Since the lengths of the MSB word and LSB word determine the LUT length and spacing, they also affect the linearity of the final digital output. The number of points in the LUT needed for adequate correction is determined by the desired ADC accuracy and the nonlinearity of the VCO v-to-f characteristic [6,84]. Furthermore, the resolution of the LUT coefficients should also be carefully considered. It is impractical and redundant to have a very large resolution register for one LUT coefficient, which will induce exponential increment in chip area if the LUT size is large. Therefore, one of the good choices will be to choose the bits of LUT coefficients as about twice of the ADC output bits. Because it can provide enough bits for a large value of the multiplication of two outputs. Therefore, though Larger LUT size and higher resolution of LUT coefficients can provide a better linearity of digital output, we should also do a trade-off by considering the complexity of the circuitry, power consumption and chip area.

Another aspect that must be considered when implementing the LUT is the redundancy factor. If there is a region where the slope of the LUT transfer function dx/dn is greater than 1, if the input count increases by 1, the corresponding output

code might increase by more than 1, this behaviour will lead to missing codes in the ADC. To ensure every output x can be reached by at least one input count n, the slope dx/dn must always be less than unity [85].

For an N-bit ADC, the output code x will have 2^N possible values, 0 to $2^N - 1$. In order to ensure the slope to be less than unity, the total possible output counts of the VCO must be increased to $R \cdot 2^N - 1$, where R is a redundancy factor. For an ideal ADC, the slope is always 1, therefore, no redundancy would be required and R = 1.

There are several ways to implement the redundancy factor in VCO-based ADC. The first one is to initialize the LUT so that total output range is reduced by a factor R compared to the input range. For example, as discussed previously, the distance between two adjacent LUT entries in the n domain is 2^L where L is the LSB word length. The redundancy factor R could be realized by the LUT as the difference between two values in the adjacent LUT locations is $2^L/R$. Another simpler approach is to shift the radix point of the digital output code. For example, the redundancy factor of 4 can be implemented by a left shift in the radix point by 2 bits. This method only works if 2^L is a power of 2, it is easy to be implemented in digital domain.

4.3 Digital Background Calibration Flow

The task of calibration is to determine the correct coefficients in LUT. The word "correct" means with the proper value of coefficients a_i as expressed in Figure 4.3 and in equation (4.1), which could build a inversely symmetric mapping (as shown in Figure 4.2) between raw counts n and corrected codes x.

One option is to take ADC offline, sweep the input linearity over the entire signal

range and determine the proper coefficients a_i [15]. Though it is a straight forward method, the ADC has to be offline, which means the calibration is not real-time. This offline calibration can not handle the VCO v-to-f characteristics changing if the work environment or condition varies. The offline calibration has to take ADC offline frequently to find new coefficients a_i for a new VCO v-to-f relation, which is not practical or useful in a real application. What's more, the offline calibration also depends on a known input signal. It can not work with unknown signals.

A real-time background calibration procedure which, is briefly introduced in Chapter 2 is now proposed in this section. The "split ADC" approach [1, 2] is a real-time calibration algorithm with no need for an accurately known input signal. In Figure 4.4, the dithered "split ADC" calibration system block diagram is shown.

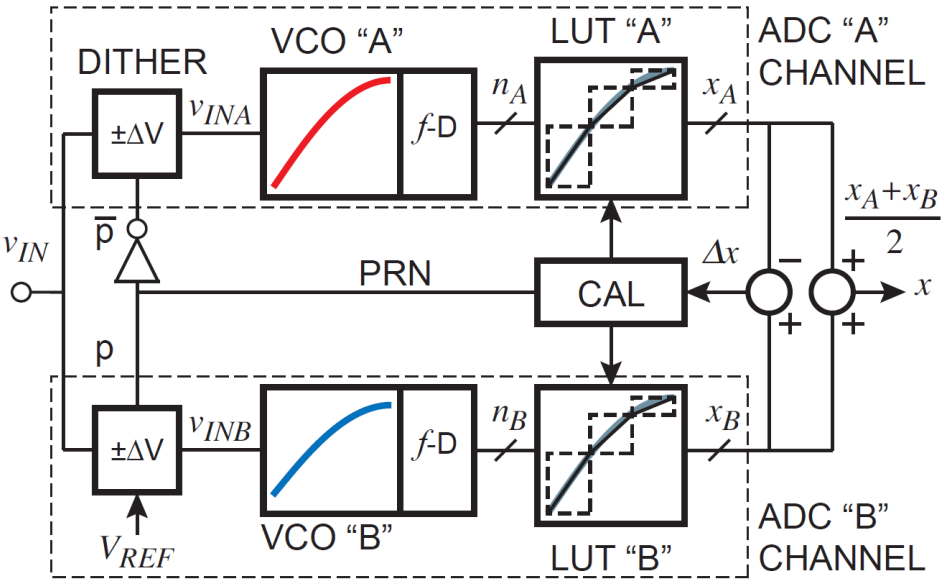


Figure 4.4: Dithered "Split ADC" system block diagram [6]

In the "split ADC" concept, unlike a tradition ADC, there are two identical channels called ADC "A" and ADC "B". Each can produce individual output codes x_A and x_B. Then, a small dither $\pm\Delta V$ is added to the input voltage so that the inputs of each channels are

$$v_{INA} = v_{IN} - p\Delta V \tag{4.2a}$$

$$v_{INB} = v_{IN} + p\Delta V \tag{4.2b}$$

where $p = \pm 1$ is chosen on a pseudo-random basis for each conversion.

If ADC "A" and "B" are both linear, there will be a linear mapping between the analog input v_{IN} and the digital output code x, as shown below,

$$v_{INA} = v_{IN} - p\Delta V \Rightarrow x_A = x - pD \tag{4.3a}$$

$$v_{INB} = v_{IN} + p\Delta V \Rightarrow x_B = x - pD. \tag{4.3b}$$

D is the corresponding digital code of analog dither ΔV, thus, D is digital dither and can be derived as,

$$\frac{\Delta V}{V_{FSR}} = \frac{D}{2^N} \Rightarrow D = \frac{\Delta V}{V_{FSR}} \cdot 2^N \tag{4.4}$$

where V_{FSR} is the full scale range of the analog input signal, and N is the targeting resolution of the ADC.

From equation (4.4), to keep a reasonable relation between the ADC's input and output, the choice of digital excursion D is not independent of analog voltage dither ΔV. For example, if D is large, ΔV must be a large value as well, which will reduce the input signal range.

In the "split ADC" architecture, the final digital output is from the average of the two outputs of ADC "A" and ADC "B",

$$x_{OUT} = \frac{x_A + x_B}{2} = \frac{(x - pD) + (x + pD)}{2} = x \tag{4.5}$$

indicating that the dither operation self-cancels at the ADC output when both ADCs are correctly calibrated. At the same time, the background calibration signal is developed from the difference of ADC "A" and ADC "B",

$$\Delta x = x_B - x_A. \tag{4.6}$$

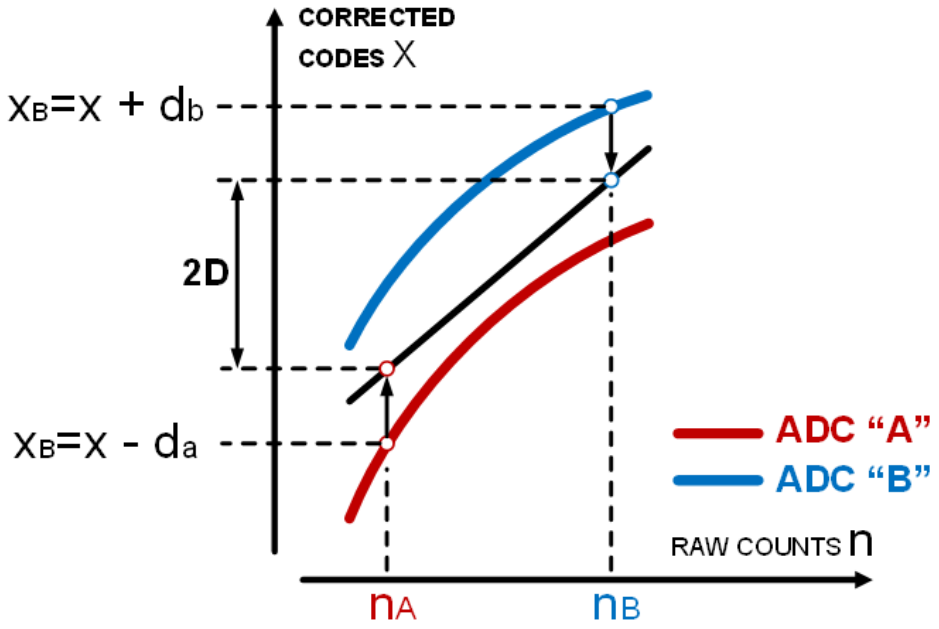


Figure 4.5: Split ADC calibration with dither

Ideally or if both ADC channels are correctly calibrated, the difference is,

$$x_B - x_A = (x + pD) - (x - pD) = 2pD = \pm 2D \tag{4.7}$$

Thus, equation (4.7) can be used as a reference in the calibration flow. If the two ADC channels' digital output codes can reach this target or get close enough (within the ADC's resolution), then the ADC is correctly calibrated.

Normally, the two ADC channels have different characteristics (mismatch), thus, there will be two LUTs (LUTA and LUTB). If the LUTs are not calibrated correctly,

then the mapping of input voltages to output codes will not follow the ideal case of (4.3a). In general, as shown in Figure 4.5, the ADC "A" and "B" outputs are,

$$x_A = x - d_a \tag{4.8a}$$

$$x_B = x + d_b \tag{4.8b}$$

In this practical case,

$$x_B - x_A = (x + d_b) - (x - d_a) = d_b + d_a. \tag{4.9}$$

According to the target value $\pm 2D$, the next step is to find a way to adjust the "wrong" difference in equation (4.9) to $\pm 2D$. As explained in LUT-based correction section, the digital output codes are all created from the first order interpolation. The calibration algorithm then will try to remove the error in the LUTs coefficients to correct the output codes x.

4.3.1 Error Estimation

In the previous sections, we have discussed that to get a corrected output code x, the LUTs' coefficients have to be updated by removing the errors caused by ADC nonlinearity.

First, recall the equation (4.1), re-writte it here as,

$$x = a_{n_U} + \underbrace{(\frac{n_L}{2^L})}_{y}[a_{n_U+1} - a_{n_U}]. \tag{4.10}$$

The fraction term $\frac{n_L}{2^L}$ is temporarily expressed as y for easier mathematical arrange-

ment. Then, equation in (4.10) can be rearrange as,

$$x = (1 - y)a_{n_U} + ya_{n_U+1}. \tag{4.11}$$

Since ADC "A" and ADC "B" have different characteristics and we assume there are uncorrected errors in the coefficients of LUTA and LUTB, then the lower corner subscript and "hat" indicators are added into equation 4.11 to represent a uncorrected output codes, as shown below,

$$\hat{x}_A = (1 - y_A)\hat{a}_{n_{UA}} + y_A\hat{a}_{n_{UA}+1} \tag{4.12a}$$

$$\hat{x}_B = (1 - y_B)\hat{b}_{n_{UB}} + y_B\hat{b}_{n_{UB}+1}. \tag{4.12b}$$

Now define each of coefficients in LUTA and LUTB to be the sum of a correct "true" value and "error",

$$\hat{a}_{n_{UA}} = \underbrace{a_{n_{UA}}}_{TRUE} + \underbrace{\varepsilon_{n_{UA}}}_{ERROR} \tag{4.13a}$$

$$\hat{b}_{n_{UB}} = \underbrace{b_{n_{UB}}}_{TRUE} + \underbrace{\varepsilon_{n_{UB}}}_{ERROR}. \tag{4.13b}$$

The goal is to remove the error terms.

Using equation (4.13a) and equation (4.12a), the output codes of ADC "A" and ADC "B" can be also expressed as,

$$\hat{x}_A = (1 - y_A)(a_{n_{UA}} + \varepsilon_{n_{UA}}) + y_A(a_{n_{UA}+1} + \varepsilon_{n_{UA}+1}) \tag{4.14a}$$

$$\hat{x}_B = (1 - y_B)(b_{n_{UB}} + \varepsilon_{n_{UB}}) + y_B(b_{n_{UB}+1} + \varepsilon_{n_{UB}+1}). \tag{4.14b}$$

Rearrange the equation in 4.14a to take the "True" value of x out,

$$\hat{x}_A = x_A + (1 - y_A)\varepsilon_{n_{UA}} + y_A\varepsilon_{n_{UA}+1} \tag{4.15a}$$

$$\hat{x}_B = x_B + (1 - y_B)\varepsilon_{n_{UB}} + y_B\varepsilon_{n_{UB}+1}, \tag{4.15b}$$

Recall the target, the difference of VCO output ADC "A" and ADC "B",

$$\hat{x}_B - \hat{x}_A = x_B - x_A + (1 - y_B)\varepsilon_{n_{UB}} + y_B\varepsilon_{n_{UB}+1} - (1 - y_A)\varepsilon_{n_{UA}} - y_A\varepsilon_{n_{UA}+1}. \tag{4.16}$$

Substituting equation (4.7) into (4.16),

$$\hat{x}_B - \hat{x}_A - 2pD = (1 - y_B)\varepsilon_{n_{UB}} + y_B\varepsilon_{n_{UB}+1} - (1 - y_A)\varepsilon_{n_{UA}} - y_A\varepsilon_{n_{UA}+1}. \tag{4.17}$$

Equation (4.17) captures the contribution of each LUT entry error to the variation of $\Delta x = x_B - x_A$ from its ideal value $2pD$. If all the error terms ε in equation (4.17) are zero, then the left hand side must be equal to zero, indicating the correct offset and slope calibration of the ADC characteristics.

Four conversions are needed to solve for the four unknown errors in a specific LUT entry in equation (4.17). Since there are many LUTs errors to determine, an ensemble of K conversions is accumulated (In this example, we use 1024 conversions per ensemble). A matrix representation of these results are described by:

$$\begin{bmatrix} \mathbf{Y_A} & \mathbf{Y_B} \end{bmatrix}_{K\times 2M} \cdot \begin{bmatrix} -\mathbf{e_A} \\ \mathbf{e_B} \end{bmatrix}_{2M\times 1} = (\mathbf{\Delta x} - 2D\mathbf{p})_{K\times 1} \tag{4.18}$$

$\mathbf{Y_A}$ and $\mathbf{Y_B}$ are $K \times M$ matrices containing coefficients y_A and y_B; K is the

ensemble size and M is the length of the LUT. An example of $\mathbf{Y_A}$ is shown in equation (4.19),

$$\mathbf{Y_A} = \begin{bmatrix} (1-y_{A1}) & y_{A1} & 0 & 0 & \dots \\ 0 & 0 & (1-y_{A2}) & y_{A2} & \dots \\ \vdots & \vdots & \vdots & \vdots & \dots \\ 0 & 0 & (1-y_{Ak}) & y_{Ak} & \dots \\ \vdots & \vdots & \vdots & \vdots & \dots \\ 0 & (1-y_{AK}) & y_{AK} & 0 & \dots \end{bmatrix}_{K\times M} \tag{4.19}$$

$\mathbf{Y_B}$ is similar to $\mathbf{Y_A}$, because ADC "A" and ADC "B" are identical but split by small dither. The entries in $\mathbf{Y_A}$ or $\mathbf{Y_B}$ matrix are decided by the input signal. Since in each conversion there is only one input signal, in each row of $\mathbf{Y_A}$ or $\mathbf{Y_B}$, only two adjacent entries will be used.

$\mathbf{e_A}$ and $\mathbf{e_B}$ in equation (4.18) are $M \times 1$ vectors, they represent the error terms in the LUT coefficients,

$$\mathbf{e_A} = \begin{bmatrix} \varepsilon_{0A} \\ \varepsilon_{1A} \\ \vdots \\ \varepsilon_{MA} \end{bmatrix}_{M\times 1}, \mathbf{e_B} = \begin{bmatrix} \varepsilon_{0B} \\ \varepsilon_{1B} \\ \vdots \\ \varepsilon_{MB} \end{bmatrix}_{M\times 1} \tag{4.20}$$

On the right side of equation (4.18), Δx and $2D\mathbf{p}$ are $K \times 1$ vectors, which are the calibration target difference,

$$\mathbf{\Delta x} - 2D\mathbf{p} = \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \vdots \\ \Delta x_k \\ \vdots \\ \Delta x_K \end{bmatrix}_{K\times 1} - 2D \cdot \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_k \\ \vdots \\ p_K \end{bmatrix}_{K\times 1} \tag{4.21}$$

Ideally, the LUT error vectors $\mathbf{e_A}$ and $\mathbf{e_B}$ can be determined by solving equation 4.18. After that, according to equation 4.17, we can remove the "solved" errors from LUT coefficients to get "true" corrected output codes.

However, there are difficulties that limit this approach. If an LUT location is not used by any of the ensemble conversions, then the corresponding column of will consist entirely of zeros. This will occur when the input does not cover the entire ADC signal range. This case is rank deficient and a unique solution of 4.18 does not exist. Intuitively this makes sense, since there can be no information in the matrix about unused LUT locations. Even if all LUT locations are used and there are no zero columns, the relationship between the coefficients in equation (4.12a) causes the matrix to be rank deficient, and there is no unique solution for this case.

4.3.2 Least Mean Square Loop for Iterative Solution

The main challenge to the calibration process proposed in previous sections is the ability to solve for an exact solution to equation (4.18). Thus, instead of trying to get a unique value, we adopt a least mean square (LMS) estimation method [2,3,86] to give a numerical answer by an iterative procedure. The procedure begins by first

multiplying both sides of (4.18) with the transpose,

$$(\mathbf{Y^T} \cdot \mathbf{Y}) = \mathbf{Y^T} \cdot (\mathbf{\Delta x} - \mathbf{d}) \tag{4.22}$$

For simplicity, assume (unrealistically) that the matrix $\mathbf{Y^T} \cdot \mathbf{Y}$ is equal to the identity matrix,

$$\mathbf{Y^T} \cdot \mathbf{Y} = \mathbf{I} \tag{4.23}$$

Substituting equation (4.23) into (4.22) gives a solution to the error matrix,

$$\mathbf{e} = \begin{bmatrix} -\mathbf{e_A} \\ \mathbf{e_B} \end{bmatrix}_{2M \times 1} = \mathbf{Y^T} \cdot (\mathbf{\Delta x} - \mathbf{d}). \tag{4.24}$$

The corrected LUT entries would be calculated by subtracting the error terms from the incorrect LUT coefficients. A LMS method is adopted by subtracting a small portion μ of the estimated errors ε as follows,

$$a_i{}^{new} = a_i{}^{old} - \mu \varepsilon_{iA} \tag{4.25a}$$

$$b_i{}^{new} = b_i{}^{old} - \mu \varepsilon_{iB}. \tag{4.25b}$$

Thus, large $\mathbf{Y}$ matrix need not be stored, since the information required for the $\mathbf{e_A}$ and $\mathbf{e_A}$ estimation can be accumulated on a conversion-by-conversion basis. Every conversion, if an LUT location is hit, the error for that entry will be accumulated and then finally subtracted from the actual LUT coefficients at the end of the ensemble.

A key advantage of the LMS approach is that the error estimates need not be accurate; all that is required is that they be zero-bias and (on average) steer the convergence of each LUT entry in the correct direction [3]. Secondly, development

of the "split ADC" approach relies on the A and B inputs differing by a known ΔV dither. In practice, noise will cause an additional difference, leading to inaccuracy in error estimation even if the error terms had been solved exactly. By averaging information over many conversions, the LMS approach averages out the effect of noise in determining calibration parameters [6, 84].

4.3.3 "Stitching" Estimation and LUT Adjustment

This proposed calibration algorithm is based on input signal training (the LUT entries which are not hit will not have enough information to remove error) as described in the previous sections, thus, the calibration efficiency highly depends on the how large the regions that input signals can cover in every ensemble.

In each conversion, only two adjacent LUT entries are hit. If the signal activity histogram changes to access previously unused LUT entries, after one ensemble, the coefficients in LUTs can be efficiently updated. In other words, if the regions of LUTs are not covered by signals, the update is meaningless.

"Stitching" Estimation

Consider the example shown in Figure 4.6. The histogram of input signal distribution in one ensemble of K conversions is shown. The blue portion is the high probability region, while the white portion is low probability or never hit region in this ensemble case.

Suppose the error estimation process begins with an initial distribution of linearly spaced LUT entries a_i as shown in the plot of the figure. Since the LUT entries are incorrect the error estimation process would result in nonzero values, which will be used in the LMS loop to drive the errors toward zero. Due to the limited signal range in this ensemble, only LUT segments 1-4 are used; the input voltage never

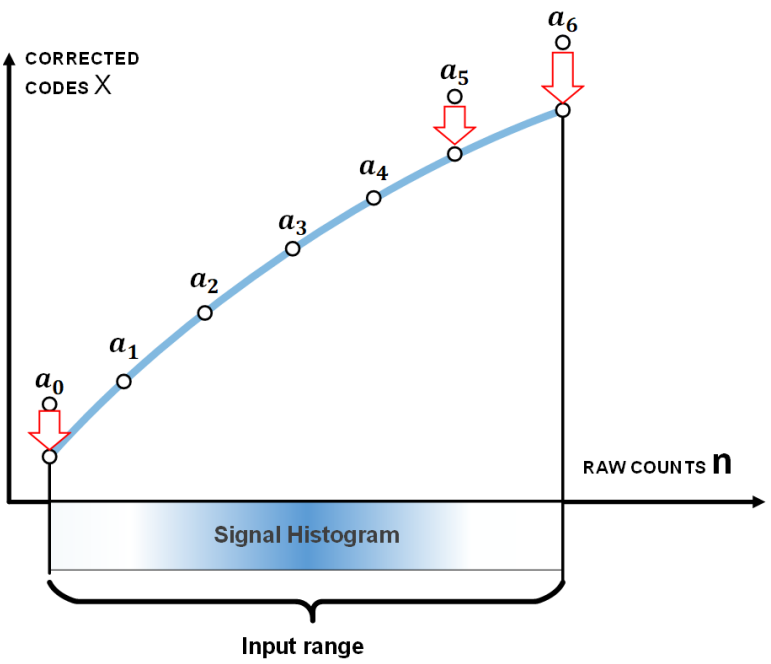


Figure 4.6: "Stitching" estimation for signal continuity in one ensemble

reaches the range corresponding to segments 0, 5, and 6. Nonzero error estimates are correctly returned for locations a_1 to a_4. But since LUT locations a_0, a_5, a_6 are not hit, they are not estimated. This induces a large problem of LUT continuity. In this example, there exits gaps between a_0, a_5, a_6 with other hit regions (a_1 to a_4). If in the future ensembles, those kinds of gaps are accumulated, the calibration efficiency will be poor.

The solution implemented in this work is based on the ability to distinguish between the cases of $\varepsilon = 0$ due to a true zero error, and $\varepsilon = 0$ due to an LUT location not being used. During each ensemble of conversions, the calibration algorithm keeps track of whether an LUT location has been used or not. After calculation of the ε_i values, but before LMS updating, the algorithm checks for unused LUT locations. When the algorithm reaches an unused LUT location, $\varepsilon = 0$ in that location is replaced with the nearest valid estimate from an LUT location that was used. This substitution is represented with the red big arrows in Figure 4.6. We can see a_0,

a_5, a_6 are roughly estimated to preserve continuity of the LUT and reduces ADC errors when the input signal range histogram changes. This stitching of the LUT does not completely eliminate ADC errors. Since there is no signal in the unused LUT locations no information as to the correctness of those values is determined. However this technique does preserve continuity of the lookup table at the boundary between the used and unused portions of the signal range in one ensemble.

4.3.4 LUT Adjustment

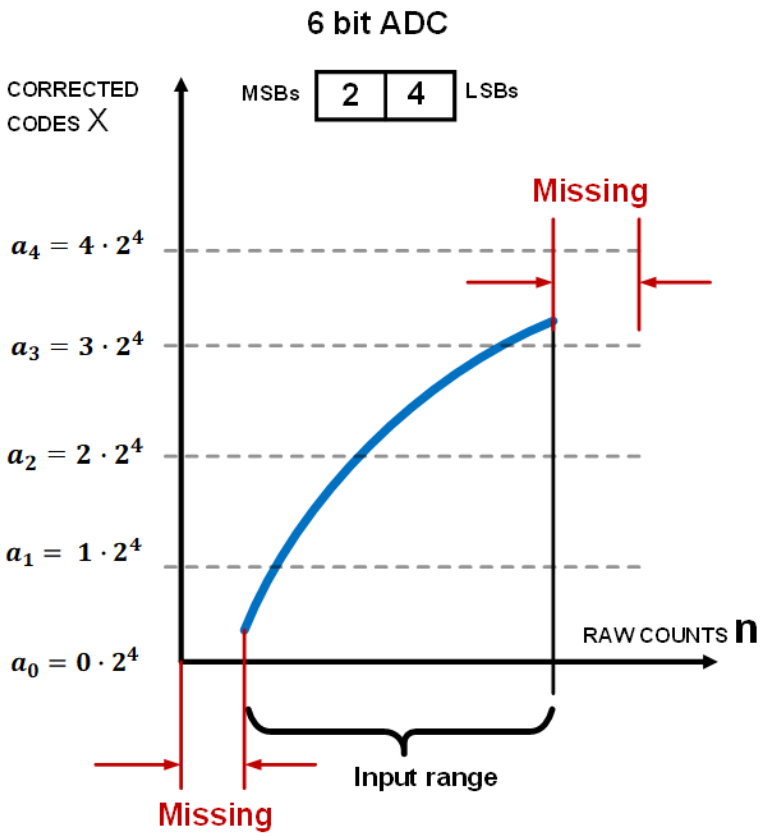


Figure 4.7: Lookup-table adjustment

Besides the problem induced by less hit region of LUT, there is another critical case to pay attention to. Figure 4.7 shows one end of the LUT and the entire LUT

can cover the input signal range. However, even though all of the LUT entries are hit extensively, there is still a problem that the two ends of LUT entries are not corrected well because they are only updated from one side. For example, an LUT entry that is not in the end of the LUT, can be updated by two sides of signals, as shown in 4.7. But the LUT entries at the ends, or close to the end of LUT are not corrected as efficiently as those in the middle of LUT. To solve this issue, we could pick the middle portion of the LUT as your output codes range, and the choice of LUT range could depend on different applications. For example, we could choose to reduce the input range to gain more linearity. The regions that are not hit in current ensemble must be hit in next ensembles. However, the two ends of LUT entries are always less calibrated since the input signal can only hit them from one side.

In the end, the difference of "stitching" estimation and LUT adjustment is,

- "Stitching" estimation keeps the signal continuity in ONE ENSEMBLE, to improve the calibration efficiency and accuracy.
- LUT adjustment solves the problem of poor calibration of entries at the ends of the LUT.

4.3.5 Offset Consideration

As noted earlier, this calibration approach provides no information on offset. The error estimation block only sees the difference between the two channels, thus has no vision on the absolute offset of the ADC. Since the LMS loop is a perfect numerical integrator, any systematic offset errors in the estimation process would accumulate indefinitely, causing numerical overflow. To prevent this numerical problem, the value of one location in the LUT is fixed and all other error estimates are referenced

to that location to prevent a global drift in offset of the lookup-table entries.

4.3.6 Background Calibration Flow

Figure 4.8 summarizes the lookup-table-based background calibration technique. The right block of the figure enclosed by a dotted line represents the conversion path of this VCO-based ADC. The left block of the figure enclosed by a dotted line represents the calibration flow which is enabled every K conversions.

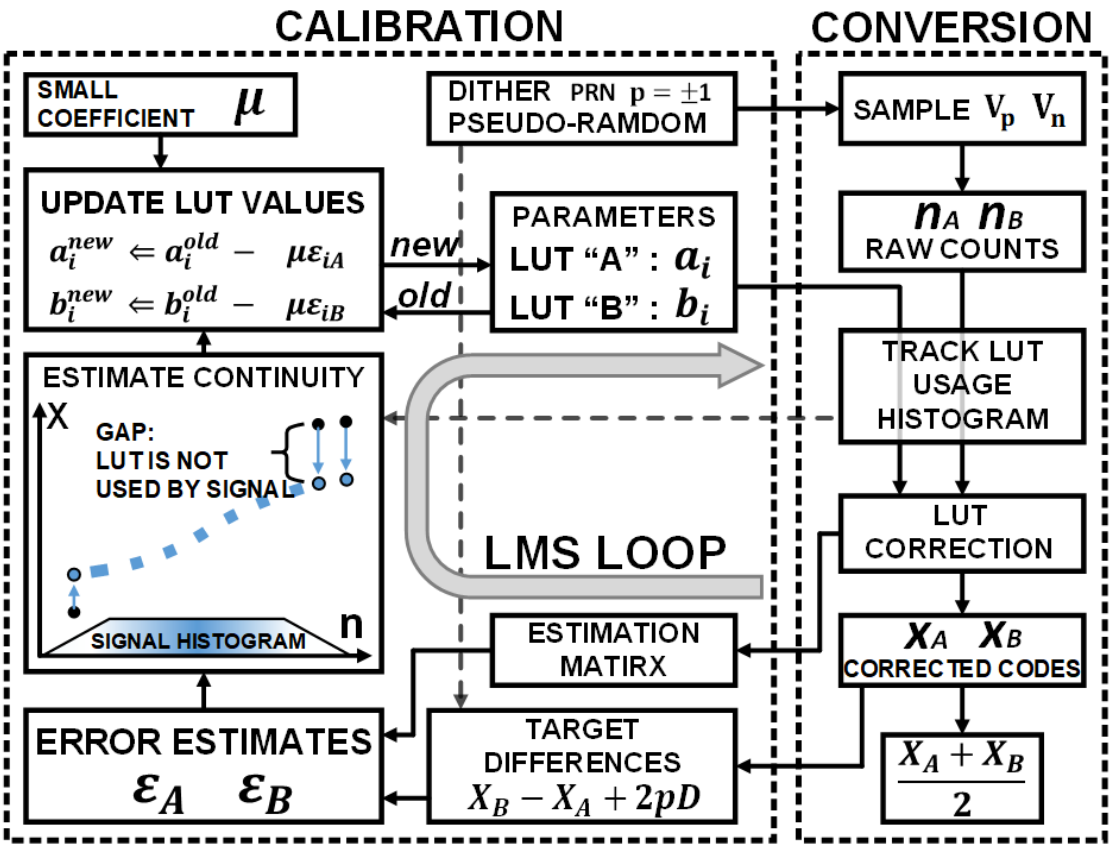


Figure 4.8: Background calibration flow

Initially, the analog input voltage is sampled and added with a known dither voltage ΔV to one channel and subtracting ΔV from the other. The sign of the dither is determined by a pseudo-random sequence $p = \pm 1$. The analog voltage

is converted in to uncorrected digital outputs (raw counts) n_A, n_B through the nonlinear VCO's v-to-f relationship in one constant conversion time. To deal with the nonlinearity in the analog domain, LUTs are used to build another inversely symmetric mapping from raw counts to corrected output codes X_A, X_B. Therefore, the relationship between input voltages and output codes is linear. The final digital code is obtained and the effect of dither is eliminated by averaging the two outputs of the two channels, x_A and x_B.

In each conversion, there are two adjacent LUT coefficients will be hit. These LUT location information are stored in TRACK USAGE HISTOGRAM block as shown in Figure 4.8. Additionally, all parameters needed to estimate the error using equation (4.25a) are also calculated, including y_A, y_B and ΔX. The error terms ε_A and ε_B of all the used LUT locations are then accumulated over K conversions. Finally, before the LUTs are updated by subtracting a small amount $\mu\varepsilon$, the error of the unused LUT coefficients are managed by "stitching" algorithm to preserve the continuity of the LUT.

Chapter 5

Chip Measurement and Results Analysis

This chapter starts with the chip measurement plan and setup including how the chip will be tested, how to design the chip evaluation PCB and how to control and collect chip data using FPGA board. Then, the measurement results are represented with detailed analysis.

5.1 Measurement Setup

The prototype of the VCO-based ADC is fabricated in TSMC 180nm CMOS technology through MOSIS on a $2 \times 2\ mm^2$ die. A PCB evaluation board and DE2-115 FPGA board are both used to test the VCO-based ADC performance. A photo of the completed PCB and FPGA board is taken as shown in Figure 5.1. To illustrate the details of the measurement setup, a functional diagram of the whole test setup is presented in Figure 5.2. A signal generator is used for creating differential inputs through an ADC driver for the ADC input requirement. A 8 GHz 80 GSa/s

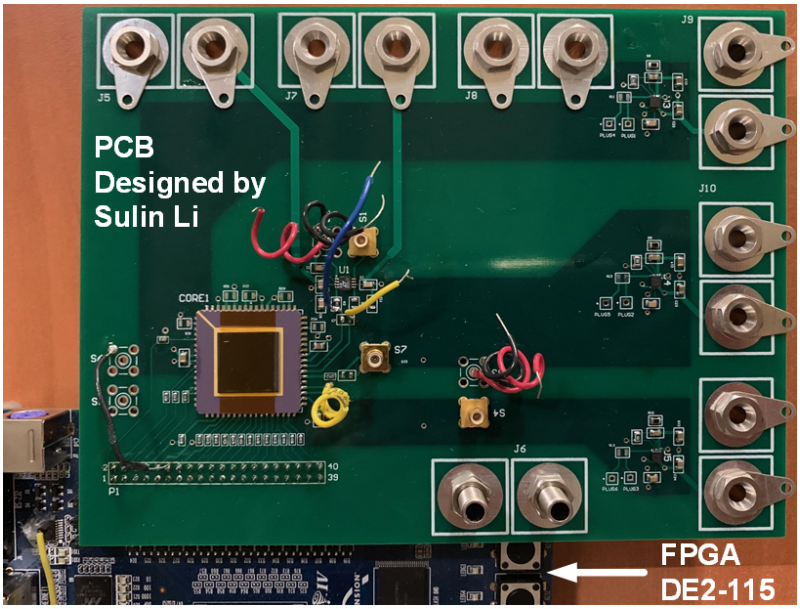


Figure 5.1: Test PCB and FPGA

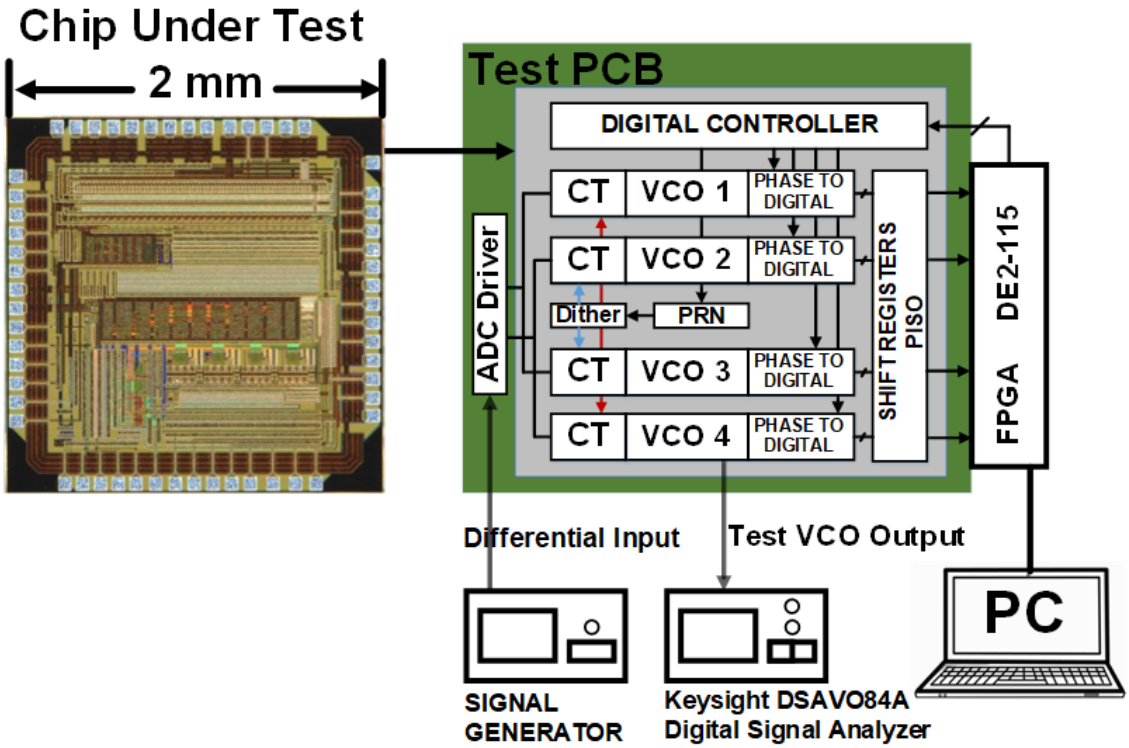


Figure 5.2: Block diagram of measurement setup

Keysight DSAVO84A digital analyser is used to test the VCO outputs. A DE2-115 FPGA board is used to collect the ADC digital outputs and load digital control parameters into chip. A UART protocol and SRAM are both used for transferring data from the FPGA to a PC.

5.1.1 Chip Introduction

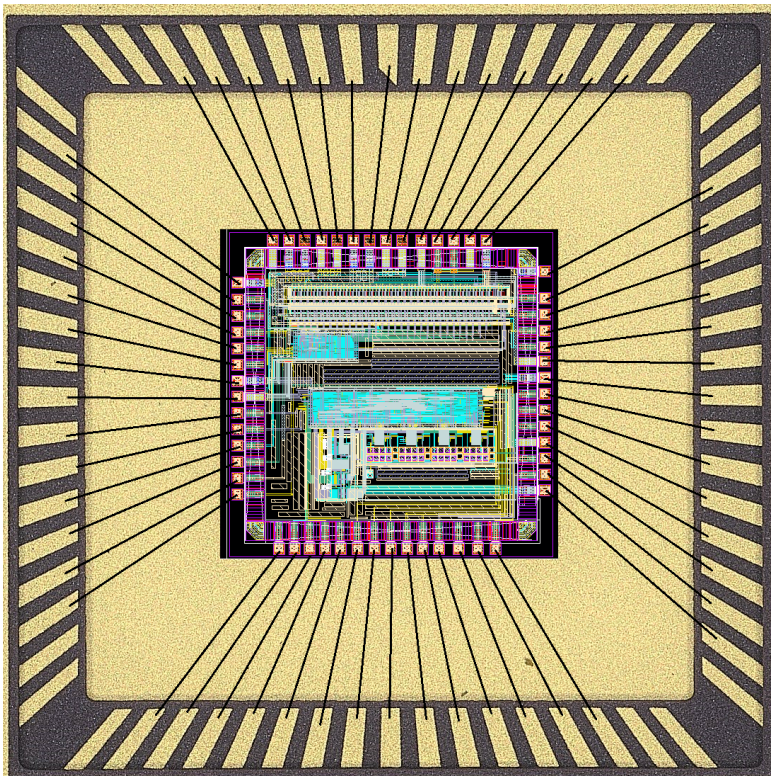


Figure 5.3: Chip connection with package

The prototype of the VCO-based ADC is fabricated in TSMC 180nm CMOS technology through MOSIS on a $2 \times 2\ mm^2$ die. Since the entire chip is shared with a seperate delay-lock-loop work [12], the VCO-based ADC takes less than half of the chip area. The active area of the analog and digital circuits are 0.09 mm^2 and 0.16 mm^2 separately. The bare dies are wire bonded by ADVOTECH into a 68-pin leadless chip carrier (LCC) package. The connection is shown in Figure 5.3. The

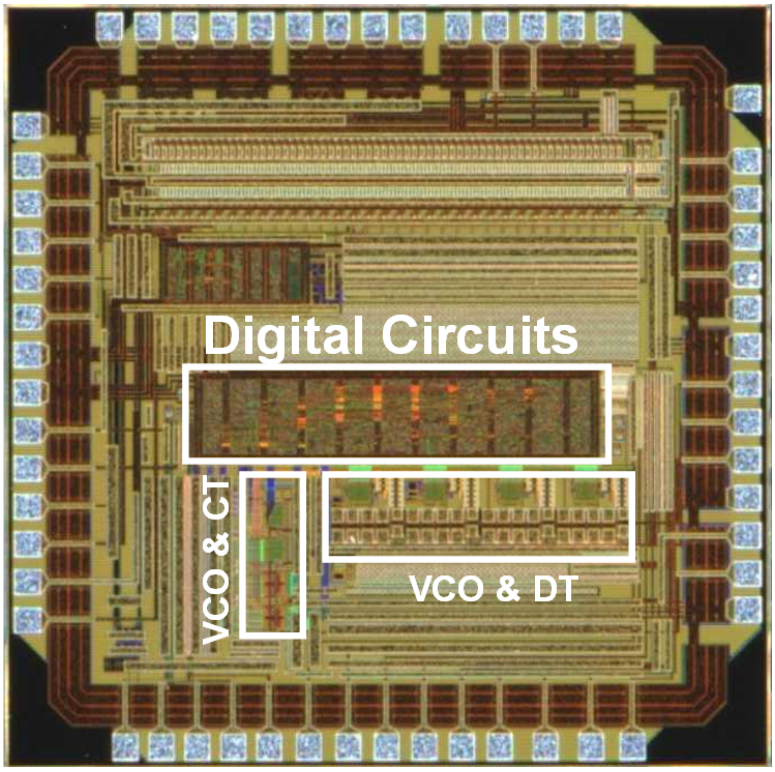
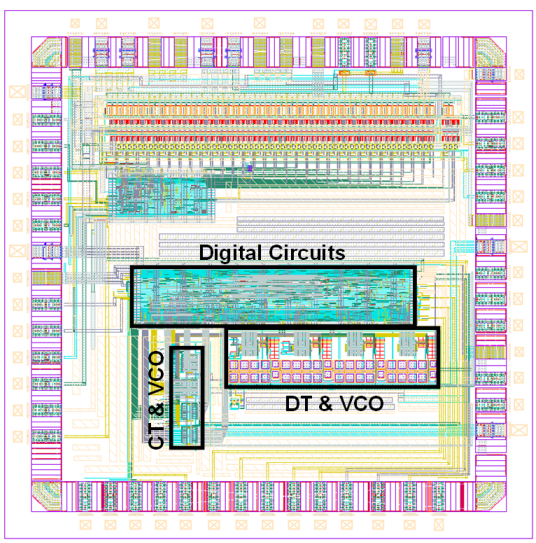
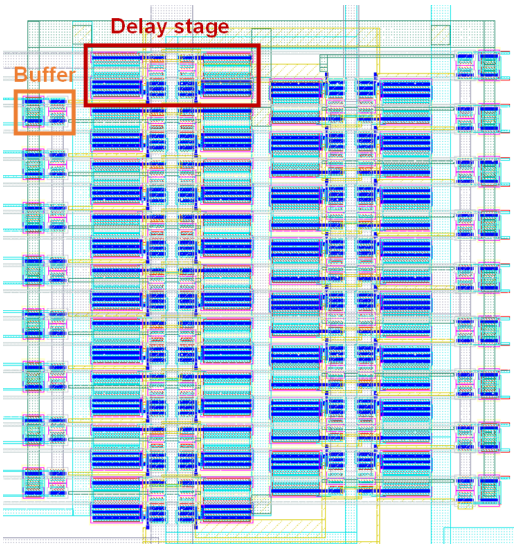


Figure 5.4: Die photo of the VCO-based ADC prototype



(a) Chip layout



(b) VCO layout

Figure 5.5: Layouts

die photo of the VCO-based ADC prototype is shown in Figure 5.4.

The analog VCOs, discrete-time input ports and continuous-time input ports are custom laid out by hand. The layout of digital circuits including ripple counters, phase measurement circuits, clock generations and other digital supporting circuits are automatically synthesized and laid out by Synopsys Design Compiler and Cadence Enounter. Details of the layout are shown in Figure 5.5.

5.1.2 Test PCB Design

A PCB as an interface between the chip and supporting circuits for chip evaluation is designed in a four-layer structure. The inner two layers are split V_{DD} planes and ground planes. The layout of the whole PCB for chip evaluation is presented in Figure 5.6. The PCB consists of the ADC chip, low dropout regulators (LDOs), ADC drivers, GPIO ports and other supporting circuits. The LDO and ADC driver

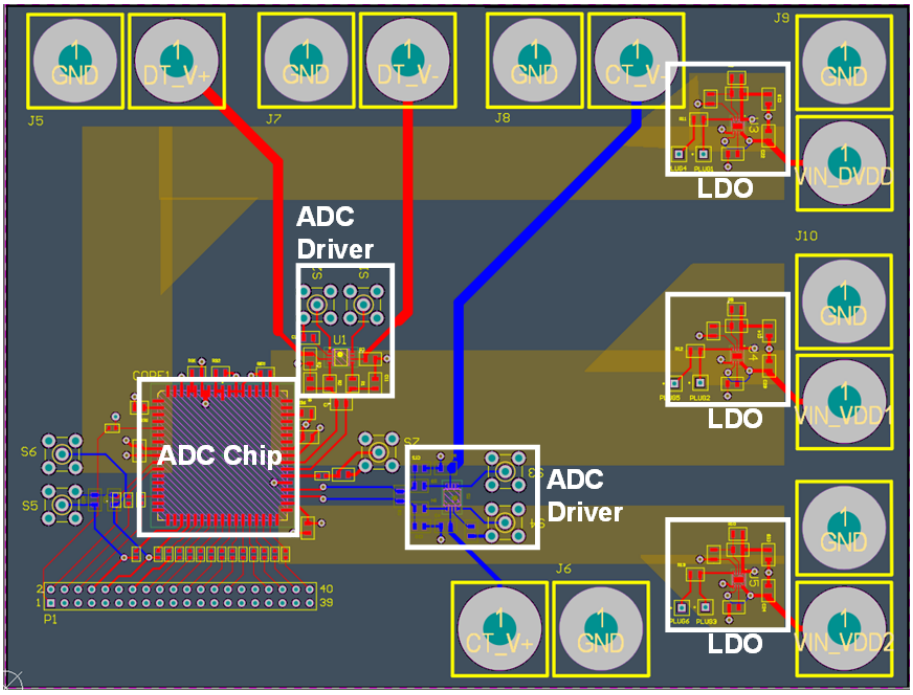


Figure 5.6: PCB layout for VCO-based ADC evaluation

are illustrated in the following subsections.

Low Dropout Regulator (LDO)

To create a clean and low noise power supply voltage, linear regulators with low dropout voltage, the TPS7A92 from Texas Instruments are used. Figure 5.7 shows the schematic of low dropout voltage regulator circuit. The supporting resistors and capacitors are chosen according to the datasheet. To separate the influence of

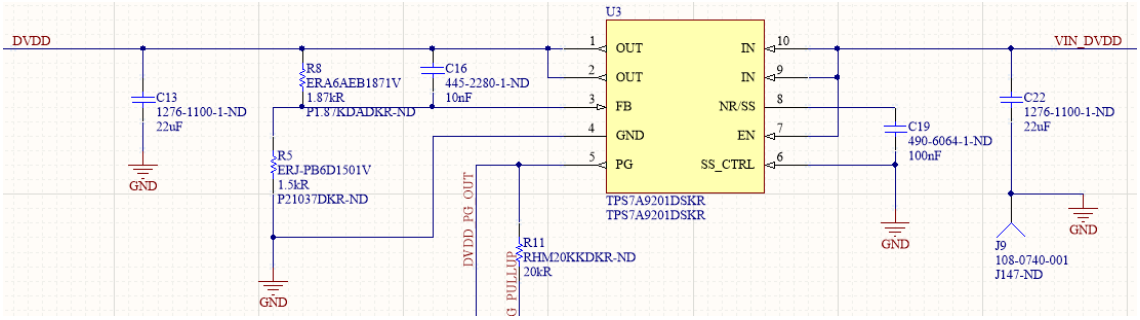


Figure 5.7: Schematic of LDO

supply voltage on the analog and digital circuits, there are three voltage regulators on the board. One is used for the digital circuit, the other two are used for the analog circuits. In the analog blocks, the continuous-time input with VCO and discrete-time input with VCO both have one voltage regulator respectively. Thus, all the circuit blocks under test have their own supply voltage, which prevents them influencing from each other.

ADC Driver

In order to provide a differential input for the VCO-based ADC, an ADC driver is needed. A fully-differential amplifier LT6363 from Analog Devices, is used. Figure 5.8 shows the schematic of a gain of 2 ADC driver made with the LT6363. The supporting resistors and capacitors are chosen according to the datasheet. The

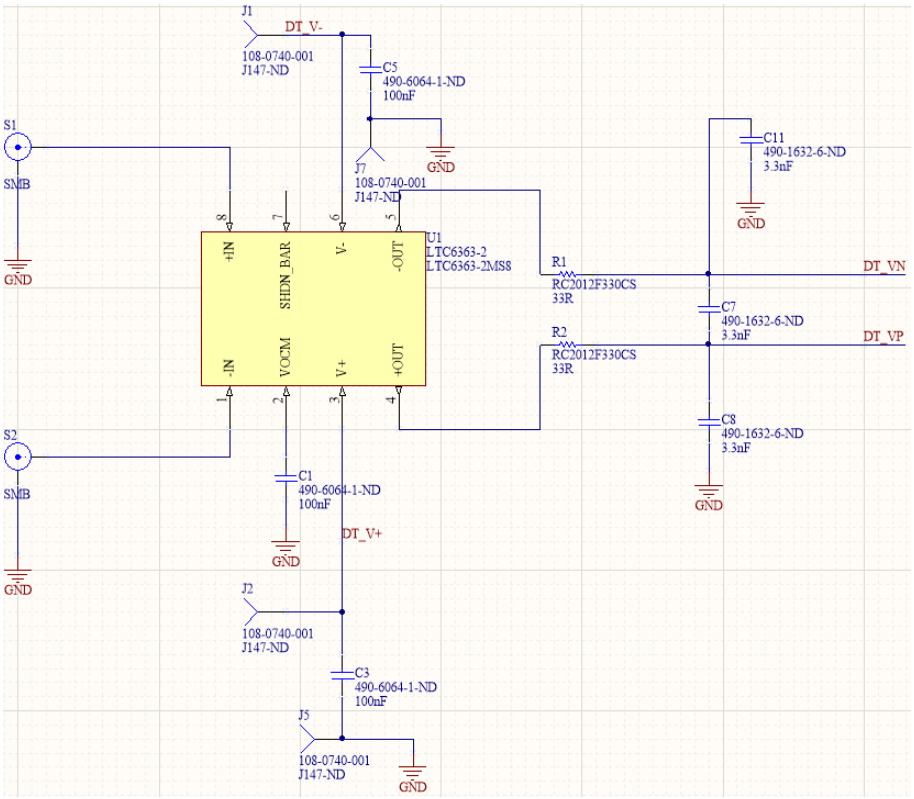


Figure 5.8: Schematic of ADC driver

differential input "0.8 to 1.8 V" required by the VCO-based ADC can be created by the ADC driver with one differential input and one DC input. Two ADC drivers are used; one for continuous-time input and one for discrete-time input respectively. To preserve the differential configuration, the two input signal paths are placed close to the chip under test and are routed symmetrically for noise suppression.

5.1.3 FPGA Board DE2-115 Implementation

The DE2-115 FPGA board is used to (1) collect the ADC digital output and (2) push parameters into the chip. The functional block diagram of the chip and FPGA is displayed in Figure 5.9. Due to the limitation of pins, customized shift registers were designed for shifting data in and out. The three blue paths are used for transferring

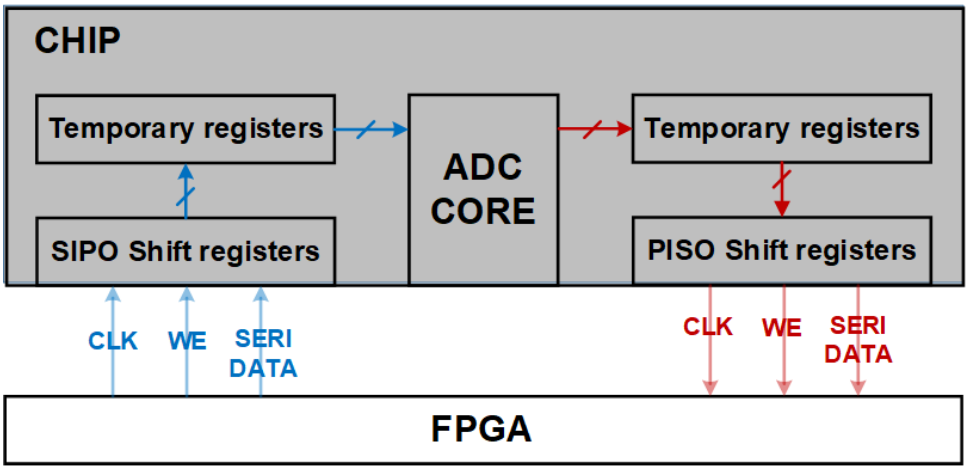


Figure 5.9: Block diagram of the chip and FPGA connection

"input reference clock", "input write enable" and "input serial data". There is a serial-in parallel output registers on chip. The serial data will be shifted into the on-chip registers through temporary holding registers (for stability and timing consideration), and finally into the ADC core. The dither size and all of the clock timing are decided by the input shifted data. Similarly, the three red paths are used for transferring "output reference clock", "output write enable" and "Output serial data". The ADC output digital codes will be stored in the temporary registers first, and then held in a parallel-in serial out shift registers. Finally the data are all slowly shifted out in the FPGA board.

5.2 Measurement Results

This prototype is designed with two different input ports: discrete-time input and continuous time input. The discrete-time input is designed as a bootstrapped S&H circuit as described in Chapter 3. The continuous-time input is realized by simply connecting the input with resisters to the gate of the control MOSFETS, also described in Chapter 3. However, the VCO-based ADC with discrete-time input ports are failed (the corresponding ADC digital outputs are showing random patterns). Thus, all the measurement results presented here are based on the continuous-time input ports.

5.2.1 VCO Measurement

The VCO designed for this highly digital VCO-based ADC is a 17-stage pseudo differential ring oscillator. The single-ended VCO input signal range is from 0.8 V to 1.8 V, thus, the differential input is -1 to +1 V. In Figure 5.10, the measurement results of the VCO is represented. The maximum frequency of VCO is about 320 MHz.

Furthermore, the v-to-f measurement results are displayed in Figure 5.11. Both of the single-ended and differential VCO v-to-f relation have been plotted. The two VCOs in a differential configuration shows a much better linearity comparing with a single-ended one. The input and output ranges are also doubled.

5.2.2 A 12-bit Resolution ADC Analysis

The measurement results of the VCO-based ADC, targeting 12-bit resolution, is demonstrated in detailed. Since the sampling frequency is limited by the crystal oscillator (50 MHz) on DE2-115 FPGA board, the sampling frequency was chosen

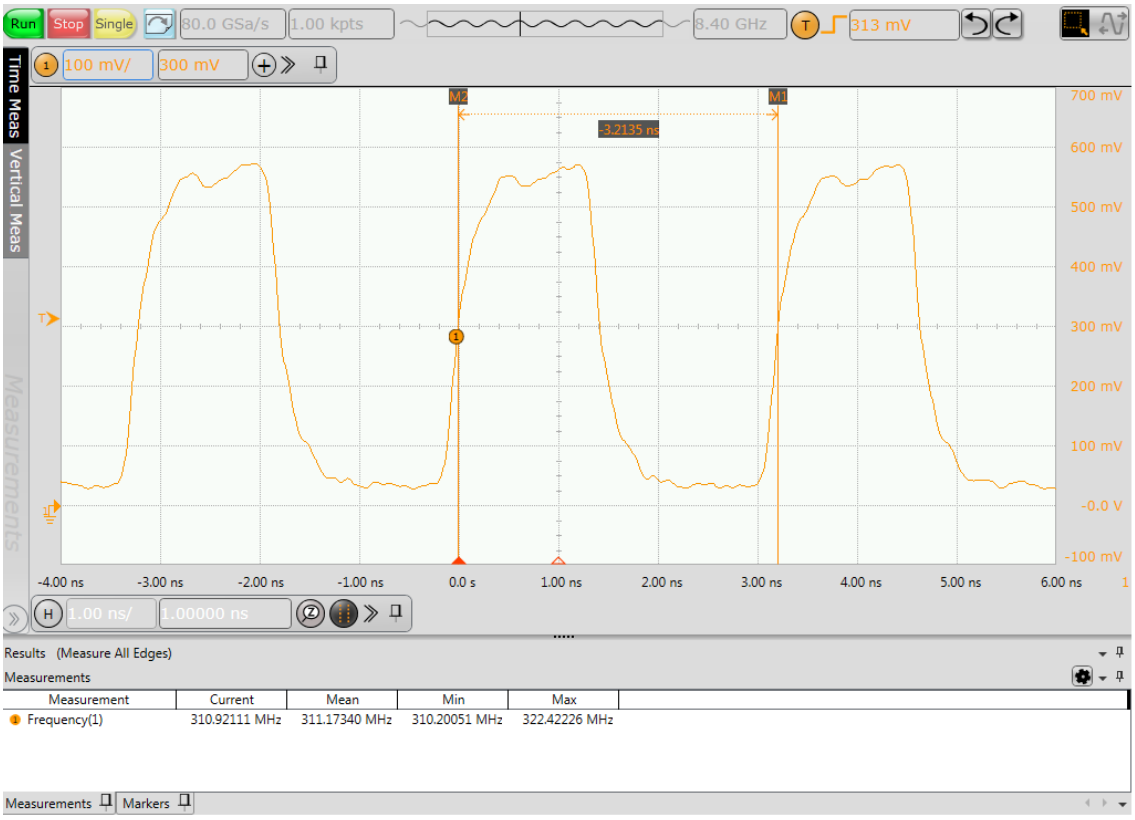
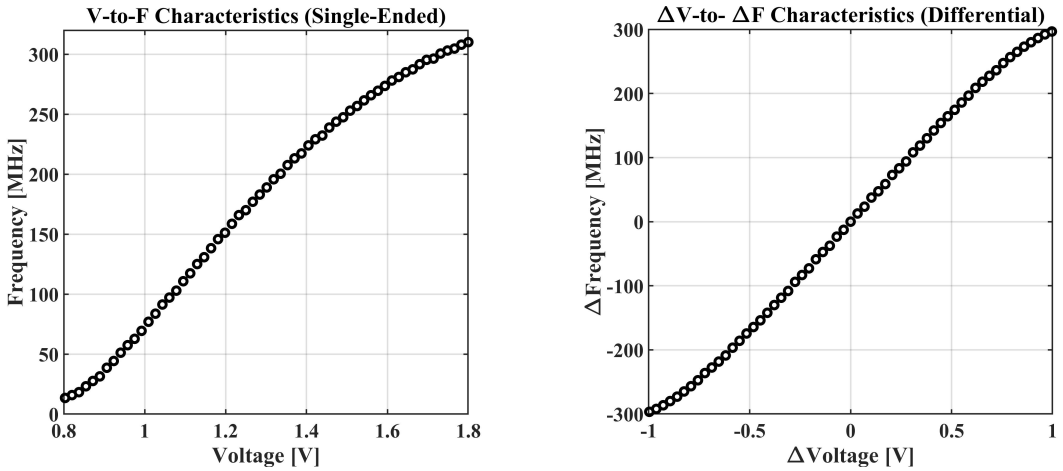


Figure 5.10: Waveform of VCO at $V_{in} = 1.8V$ (Maximum)



(a) Single-ended (b) Differential

Figure 5.11: V-to-f characteristics

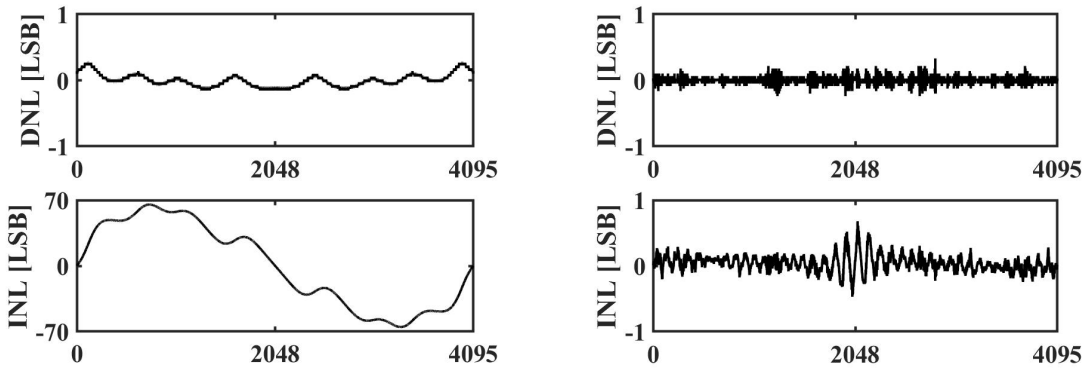
as 5 MHz. As we know from the previous section, the frequency of a single-ended 17-stage pseudo differential ring oscillator is about 300 MHz frequency range, thus,

a 12-b resolution ADC can be designed as (differential configuration introduces the number 2),

$$Digital \quad Code \quad Range = \frac{300MHz \cdot (2 \cdot 17) \cdot 2}{5MHz} \approx 2^{12} \tag{5.1}$$

DC Linearity

Figure 5.12 shows shows DC linearity results for the VCO-based ADC with and without background calibration. When there is no calibration used, the peak-to-peak INL error is -70 to +70 LSB and the DNL error is within -/+ 1 LSB (both at the at 12 b level). When the calibration is in use, the peak-to-peak INL error is within -/+1 LSB and the DNL error is also within +/- 1 LSB (both at the at 12 b level). The results indicate that the calibration can improve the INL about ×70.



(a) Without Calibration (b) With Calibration

Figure 5.12: DNL & INL

In Figure 5.13 and 5.14 demonstrates the output waveform of triangle input with/without calibration. The differential input range is -1 to +1 V. The output code is from -2048 to +2048, 12-bit resolution. (The uncorrected output codes range are a little larger than 4096 since it is a raw data)

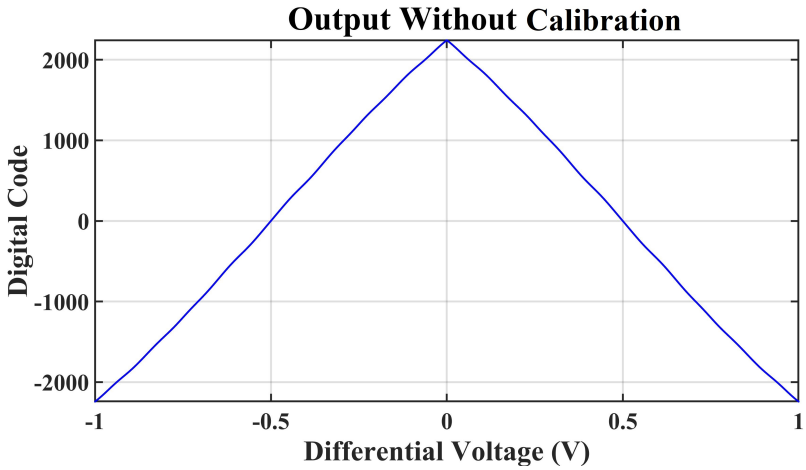


Figure 5.13: Output (Without calibration) of triangle input

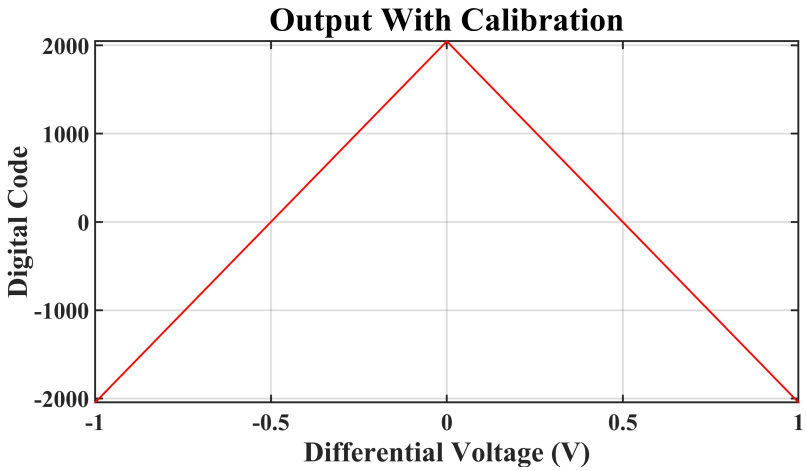


Figure 5.14: Output (With calibration) of triangle input

AC Analysis

In Figure 5.15, an 8192 point FFT analysis is presented. The sampling frequency is 5 MHz and input signal frequency is 40 kHz. With the help of calibration, the odd-order harmonics are mostly suppressed, and the ENOB is improved from 5-b to 11.5-b. The SFDR is improved from 30 dB to 78 dB. As we can see in Figure 5.16, the SNDR is linear increased with input amplitude up to full-scale swing.

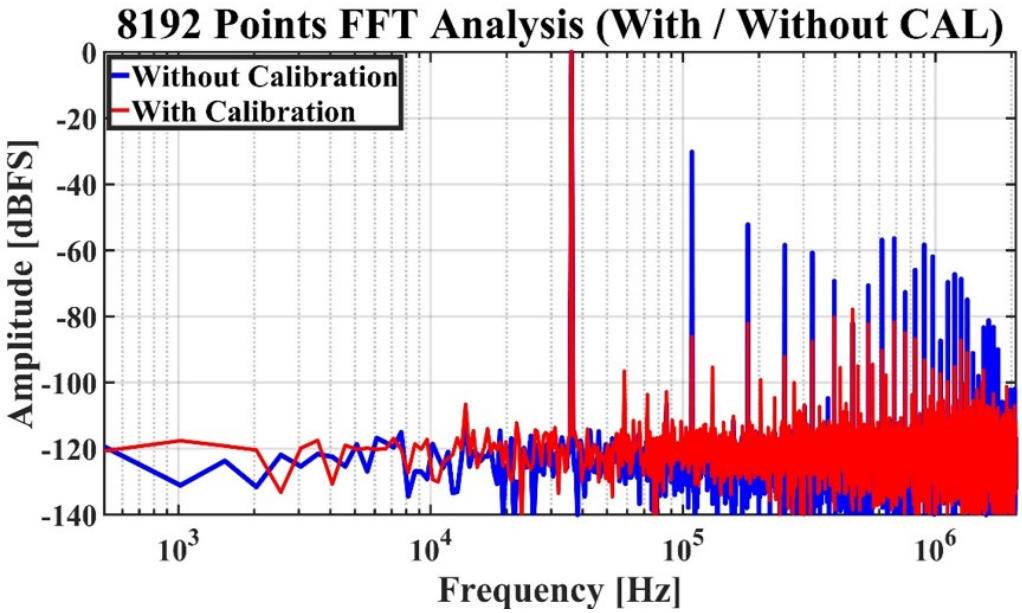


Figure 5.15: 8192 Points FFT analysis

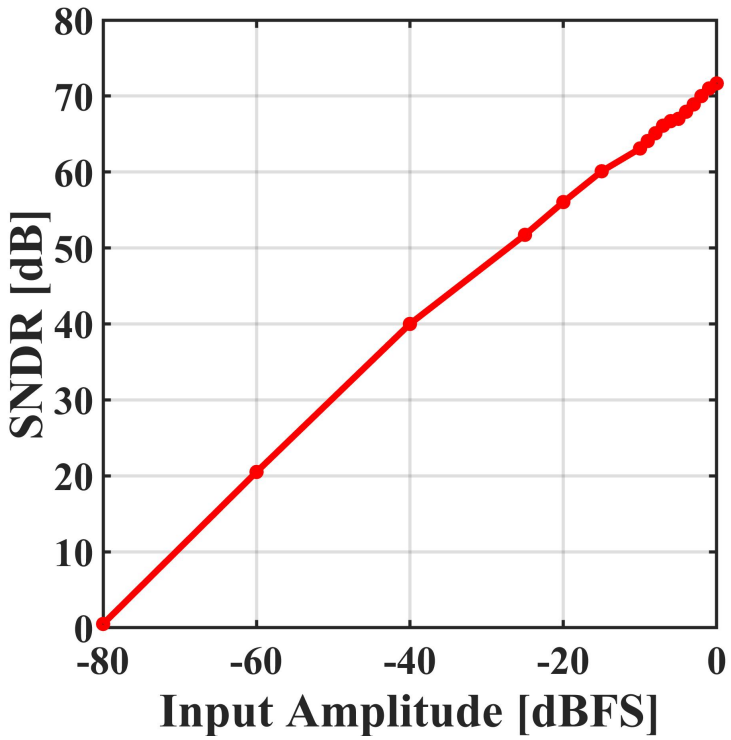


Figure 5.16: SNDR versus Input amplitude

Lookup Table Size

The LUT size is an important parameter for this VCO-based ADC design. More LUT locations will cost more chip area and increase digital circuits complexity. Figure 5.17 shows the relationship between ENOB of corrected ADC and different of LUT size. We can see that there is a limitation for LUT size. With a small LUT size, it means the distance between two adjacent LUT locations are very large, when implementing 1st-order interpolation, the resolution is too coarse and the ENOB performance is degraded. If we increases the LUT size, the distance between two adjacent LUT locations becomes smaller, which indicates a fine resolution for interpolation. The dashed line asymptote in Figure 5.17 shows that ENOB initially improves by 2 bits for each doubling of the LUT size. The 2-bits-per-doubling factor is plausible, as the interpolation involves a linear approximation to a curved V-to-f characteristic that is roughly quadratic within each LUT segment. Eventually ENOB performance is limited by quantization noise rather than LUT resolution, and increasing LUT size provides no improvement in ENOB.

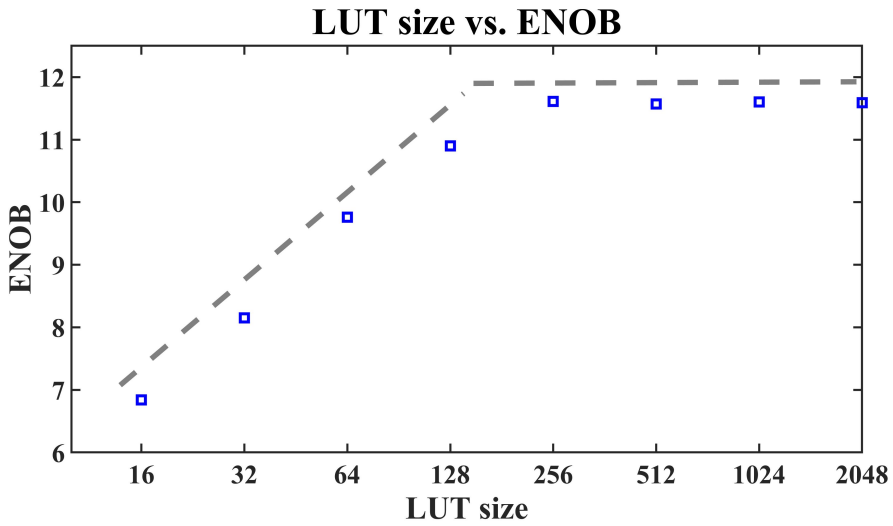


Figure 5.17: LUT size versus ENOB

Ensemble Size

The ensemble size indicates the number of conversions required before calibration system updates the value in LUT entries once. There is no specific formula to calculate a "true" value for it. The ensemble size is determined empirically.

In Figure 5.18, we can see that the ensemble size will also influence the ADC performance. Before the discussion, we need to know the figure is plotted based on the other fixed parameters (μ, LUT size and conversion time). From a large set experiments results, the ensemble size that is approximately two adjacent LUT entries is suitable for the best ADC performance. If the ensemble size is too large, the ENOB will decrease. The results are plausible. If such large ensemble size is covering a large input range or several periods of the input sigal, the calibration system can not handle so much information at one time. However, a large ensemble size could be used effectively according to some specific applications with large LUT segments.

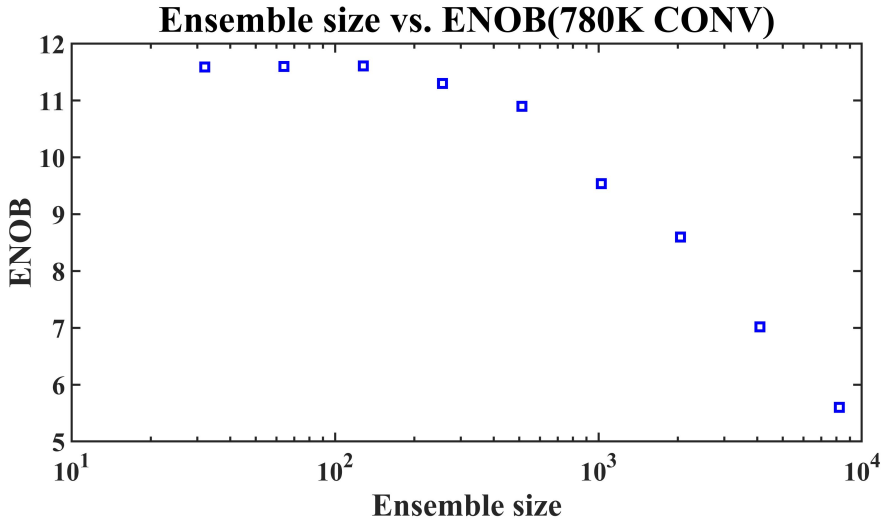


Figure 5.18: Ensemble size versus ENOB (Other parameters are fixed)

LMS Loop

The LMS loop convergence is key component to make the ADC calibration work. However, there is no specific way to calculate a value of μ. A small portion (μ value) is actually a experimental and relative result, there is no definition which illustrates it is a small or big value. The value of μ depends on different scenarios. For example, the μ for a 10-bit resolution is usually larger than a 12-bit ADC since a 10-bit ADC's error is 4 times larger than a 12-bit one. A small μ will perform well in both of these two cases, however a small μ will induce longer convergence time. When we decide the value of μ, we need to consider the trade-off between resolution and calibration time simultaneously. For this 12-bit resolution VCO-based ADC case, μ is chosen as around 2^{-10}. μ values larger than 2^{-10} can not provide 12-bit resolution and can cause divergence problems. Smaller μ values will induce longer convergence time.

5.2.3 Analysis of the Proposed VCO-Based ADC at Different CMOS Process Nodes

Due to some unexpected and unpredictable reasons, the proposed highly digital VCO-based ADC had to be fabricated in 180nm CMOS process. However, compared with other smaller CMOS nodes, the 180nm process can not take advantages of the scaling of CMOS, and the highly digital architecture can not show its compelling

Table 5.1: Analysis of Chip Area Comparison

Analog & Digital Area Report in Different CMOS Nodes			
Process node [nm]	180	65	28
Total area [mm^2]	1.5954	0.1993	0.0729
Analog area [mm^2]	0.0408	0.0136	0.0065
Digital area [mm^2]	1.5546	0.1857	0.0664
Percentage of analog area [%]	3	7	9
Percentage of digital area [%]	97	93	91

performance, such as the chip area improvement. Since the proposed highly digital VCO-based ADC moves the design complexity as much as possible into the digital domain, there will require a large LUT and calibration circuits on chip to deal with the VCO's v-to-f non-linearity. But for the first prototype, the LUTs are implemented on the FPGA board. Thus, it will be valuable to give a estimated analysis to show the proposed VCO-based ADC will have attractive characteristics: extremely area reduction, when it is applied in the smaller CMOS nodes.

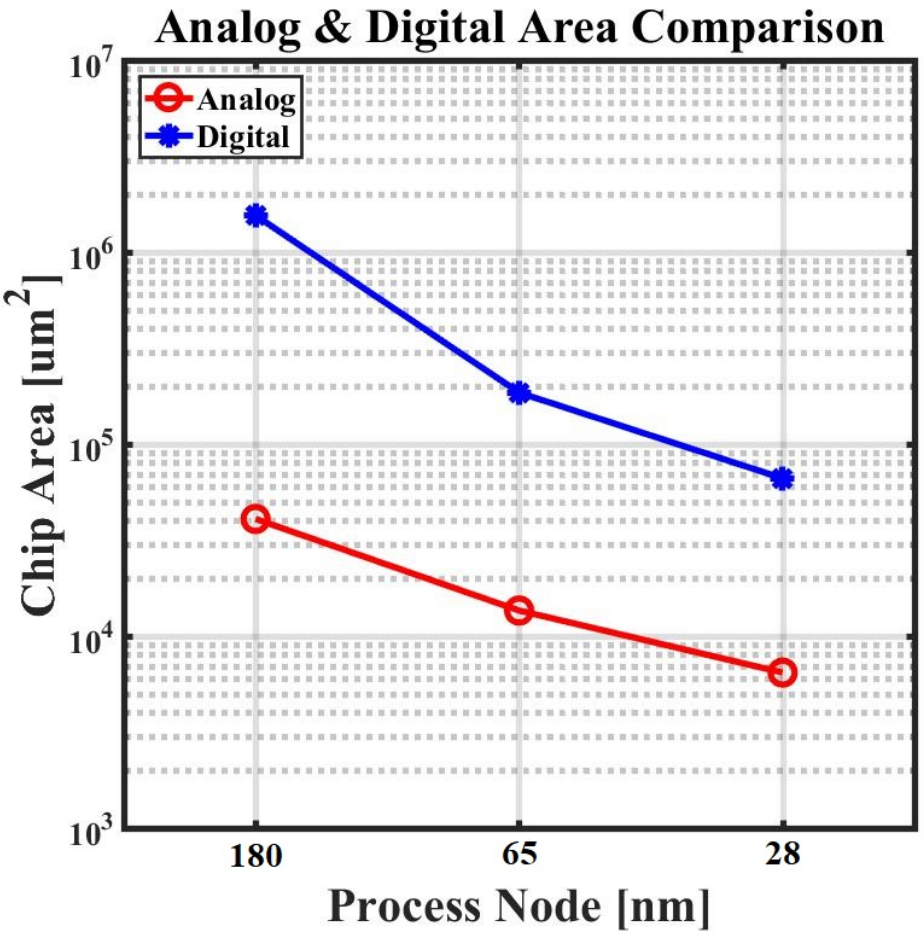


Figure 5.19: Analog and digital area comparison in different CMOS process nodes

Table 5.1 shows the estimated analog and digital chip area in different CMOS process nodes (due to the limitation to the access to more advanced technologies,

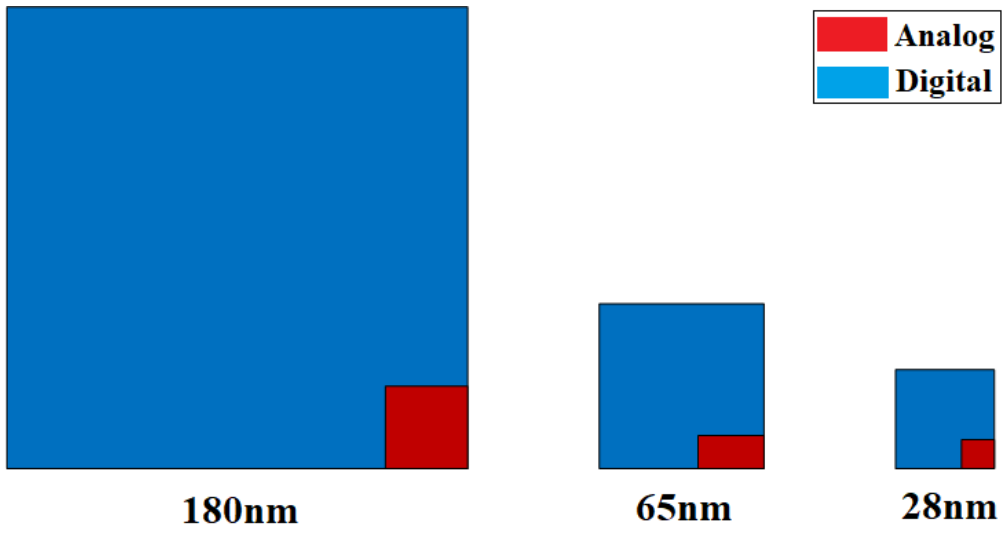


Figure 5.20: Percentage of analog and digital area in different CMOS process nodes

we can only show the comparison in 28nm, 65nm and 180nm). The analog circuit is estimated from the area of simple ring VCOs with the same transistors' size and size ratio in the prototype. The digital circuit is consist of all of the counters, phase samplers, LUTs and calibration elements. The estimated digital area is from the report of the synthesis results of Design Compiler (DC). To keep the fairness of the comparison, the same Verilog code (keep the same LUT size, same LUT coefficients' resolution and same ADC resolution, etc.) and same compile effort in DC is used in the three cases. In this particular area analysis report, LUT size is 64, LUT coefficients resolution is 24 bit and DC area compile effort is medium.

As we can see in Figure 5.19, the reduction of analog circuit area is in a linear form. The analog area is about one times less from 65nm to 28nm and two times less from 180nm to 65nm. However, the area of digital circuit is largely reduced in a exponential form. For example, the estimated digital circuit area in 180 nm is about $1.55mm^2$ but it only costs about $0.19mm^2$ in 65nm and $0.07mm^2$ in 28nm. Additionally, Figure 5.20 shows that the fraction of digital circuit area in the total chip area decreases when it uses smaller CMOS process nodes, from 97% in 180nm to

91% in 28nm. It also indicates that the reduction of digital circuit area is much larger than that in the analog circuit area. Therefore, the considerable area reduction (with the same ADC performance) when the design technique is applied in deeper CMOS nodes allows the proposed highly digital architecture become one of best candidates among other VCO-based ADCs.

Chapter 6

Conclusions

A highly digital VCO-based ADC targeting 12-bit resolution is presented in this project. A LUT based background calibration technique is used to linearize the nonlinear VCO. This VCO-based ADC is enabled by "split ADC" architecture. Each of the two split channels ADC "A" and "B" contains two VCOs in a differential

Table 6.1: PARAMETER / RESULTS SUMMARY

PROTOTYPE SYSTEM PARAMETERS/RESULTS		VALUE	UNITS
VCO	CMOS Technology	180	nm
	Input Range Used	-1 - +1	V
	VCO Frequency Range	0.5 - 320	MHz
	VCO stages	17	
ADC	Resolution	12	bits
	Sample Rate f_s	5	MSps
	Dither ΔV	1 %	FS
	DNL	-0.16 / +0.25	LSB
	INL	-0.55 / +0.48	LSB
	ENOB	11.5	bits
	Area	0.25	mm^2
	Supply Voltage	1.8	V
	Power Consumption	25	mW
	Walden Figure-of-Merit (FOM)	1.78	pJ/conv-step
LUT	Lookup-Table Size	256	points
	Counts / Segment	128	counts
LMS Loop	Parameter μ	2^{-10}	
	Convergence Time Constant	780,000	convs
	Ensemble Size	128	convs

configuration, which helps alleviate even-order distortions as well as increase the dynamic range. A digital controller on chip can reconfigure the ADCs sampling rate and resolutions to adapt to various application scenarios. Different types of input signals can be used through the simple, anti-aliasing continuous-time input to train the ADC's LUT parameters to achieve the target resolution. The chip is fabricated in 180 nm CMOS process, and the active area of analog and digital circuits is 0.09 and 0.16 mm^2, respectively. Power consumption of core ADC function is 25 mW. Measured results for a prototype design with 12-b resolution show ENOB improved from uncorrected 5-b to 11.5-b with calibration time within 200 ms (780K conversions at 5 MSps sample rate). Table 6.1 summarizes the design specifications of the 12-bit VCO-based ADC in detail.

6.1 Future Work

As shown in Table 6.1, the Walden Figure-of-Merit is 1.78 pJ/conv-step, meaning that the power consumption is not competitive with the state of the art. The most important factor that induces such large F.O.M is the 180 nm technology. Compared with smaller nanometer CMOS technologies, the speed of VCO is slow, which limits the ADC resolution. The power consumed by analog circuits is also high due to larger device size and higher voltage power supply. If we could move to a 28 nm or even smaller technology, the performance of this highly digital VCO-based ADC would improve significantly. Another critical issue of this design is the usage of output buffers. Though they solve the driving ability of the VCO output, they also consume a lot of power. Thus, instead of using output buffers, we may try to design a customized low input threshold flip flops to remove the requirement of analog buffers.

Since this is the first prototype of this style of VCO-based ADC, some parts of the background calibration circuits are not implemented on chip. There are still lots of space left for further consideration to implement all of the digital circuits on chip. Also, the customized shift registers for ADC outputs collecting are a little slow and unstable. Down-sampling is used to solve this problem. However, in the future, for the higher speed implementation, a standard protocol like SPI may be a better choice.

Appendix A

MATLAB Code

A.1 LUT-Based Calibration Algorithm

```
1000 Na = na_periods;
     Nb = nb_periods;
1002 PRN = prn_periods;


1004

     MSB = 8;
1006 LSB = 7;


1008 nspacing = 2^LSB;


1010 % Initialize Look-up-table
     luta = (-2^(MSB+LSB-1) : 2^LSB : 2^(MSB+LSB-1))';
1012 lutb = (-2^(MSB+LSB-1) : 2^LSB : 2^(MSB+LSB-1))';


1014 % Preset size just for speed
     axout = zeros(size(Na));
1016 bxout = zeros(size(Nb));
```

```
deltax = zeros(size(Na));

alutlocation = zeros(en_size, 1);
blutlocation = zeros(en_size, 1);

rawea = zeros(2^MSB + 1, 1);
raweb = zeros(2^MSB + 1, 1);

% K is number of conversions
K = length(Na);
for i = 1 : 1 : K

    e0 = mod(i,en_size);
    e = e0 + 1;
  % A
    acorrected = lookuptablecorrection(Na(i), luta, MSB, LSB);
    axout(i) = acorrected(1);
    alutlocation(e) = acorrected(2);
    axlsb = acorrected(3);
    ya = axlsb/nspacing;

    % B
    bcorrected = lookuptablecorrection(Nb(i), lutb, MSB, LSB);
    bxout(i)= bcorrected(1);
    blutlocation(e) = bcorrected(2);
    bxlsb = bcorrected(3);
    yb = bxlsb/nspacing;

    % deltax = xb - xa -/+ 2pD
    D = 200;
```

```
1048    if (PRN(i) == 1)
            deltax(i) = bxout(i) - axout(i) + 2*D;
1050    elseif (PRN(i) == 0)
            deltax(i) = bxout(i) - axout(i) - 2*D;
1052    end

1054    % raw error in A
        rawea(alutlocation(e))     = rawea(alutlocation(e))     - deltax(i)
        * (1-ya);
1056    rawea(alutlocation(e) + 1) = rawea(alutlocation(e) + 1) - deltax(i)
        * ya;

1058    % raw error in B
        raweb(blutlocation(e))     = raweb(blutlocation(e))     + deltax(i)
        * (1-yb);
1060    raweb(blutlocation(e) + 1) = raweb(blutlocation(e) + 1) + deltax(i)
        * yb;


1062
        %% Calibration Loop
1064    if (e == 1)
            % Stitching Algorithm (Continuous)
1066        for s = 1: length(rawea)
                if s < (min(alutlocation))
1068                rawea(s) = rawea(min(alutlocation));
                elseif s > (max(alutlocation)+1)
1070                rawea(s) = rawea(max(alutlocation)+1);
                end
1072        end
            for s = 1: length(raweb)
1074            if s < (min(blutlocation))
```

```
                raweb(s) = raweb(min(blutlocation));
1076            elseif s > (max(blutlocation)+1)
                raweb(s) = raweb(max(blutlocation)+1);
1078            end
        end
1080
        % Store error
1082    ea = rawea;
        eb = raweb;
1084
        % Update LUT coefficients in every ensemble conversions
1086    u = 2^-10;

1088    luta = luta - u*rawea;
        lutb = lutb - u*raweb;
1090
        % Reset raw error
1092    rawea = zeros(2^MSB + 1, 1);
        raweb = zeros(2^MSB + 1, 1);
1094    end

1096 end
```

cal.m

A.2 DNL & INL Test

```
function [dnl, inl] = dnlinl(data)
%Calculate DNL, INL
code_edges=(min(data)-0.5:1:max(data)+0.5);
dnl_count=histc(data,code_edges);
dnl=dnl_count(2:end-2);
dnl=dnl/mean(dnl)-1;
% inl as cumsum
inl=cumsum(dnl);
```

dnlinl.m

A.3 LUT Correction

```
%Look-up-table correction function


% nin: input
% lut: look up table
function outputs = lookuptablecorrection(nin,lut,MSB,LSB)


nspacing = 2^LSB;


% slope betweetn elements of lookuptable entries
slope = diff(lut)/nspacing;
loc_offset = 2^(MSB-1);
lutloc = floor(nin/nspacing) + 1 + loc_offset;

% nL
nL = nin - nspacing*(lutloc - 1 - loc_offset);
```

```
% x = au + (nL)/(2^L)*(au+1 - au)
xcorrected = zeros(size(nin));
for i = 1:length(nin)
xcorrected(i) = lut(lutloc(i)) + slope(lutloc(i))*nL(i);
end


outputs = [xcorrected;lutloc;nL];
```

lookuptablecorrection.m

Appendix B

Verilog Code

B.1 Clock Generator

```
module CLKGEN(

input CLK,
input rst,



input [8:0] x,        // divide clock parameter
input [8:0] y, // decide where Sample and Hold

input [9:0] z, // decide where to sub-sample



output reg [8:0] x_reg,
output reg [8:0] y_reg,
```

```
output reg [9:0] z_reg,



output reg [8:0] cnt,
output reg [9:0] cntz,


output SH_CLK,
output VCO_GATE,
output ADC_CLK,
output CTR_RST,

output initial_clk,
output final_clk,

output phase_sample_clk,

output sub_sample_clk

);

// just for checking -----------
always @(posedge CLK) begin
        if (rst) begin
                x_reg <= 9'd0;
```

```
                y_reg <= 9'd0;
                z_reg <= 10'd0;
        end

        else begin                  //
                x_reg <= x;
                y_reg <= y;
                z_reg <= z;
        end
end
//------------------------------

always@(posedge CLK) begin
        if (rst) begin
                cnt <= 9'd0;
        end

        else begin
                if(cnt == x_reg) begin
                        cnt <= 9'd0;
                end

                else begin
                        cnt <= cnt + 1'b1;
                end
        end
```

```
end


always@(posedge CLK) begin
        if (rst) begin
                cntz <= 9'd0;
        end

        else begin
                if (cntz == z_reg) begin
                        cntz <= 9'd0;
                end

                else if (ADC_CLK) begin
                        cntz <= cntz + 1'b1;
                end
        end
end



assign SH_CLK = !( (cnt >= 1) && (cnt <= (y_reg - 1)) );
assign VCO_GATE = !( cnt <= y_reg );
assign ADC_CLK = (cnt == (y_reg - 2));
assign CTR_RST =        !(cnt == (y_reg - 1) );
```

```
assign initial_clk = (cnt == y_reg + 2);
assign final_clk = (cnt == 1);
assign phase_sample_clk = (cnt == 0) || (cnt == y_reg + 1);

assign sub_sample_clk = (cntz == z_reg);

endmodule
```

B.2 Phase Measurement Circuits

```
module DFLIPFLOP(

clk,
reset,
d,
q,
qb

);

input clk, reset, d;
output reg q;
output qb;

always@(posedge clk or negedge reset) begin
```

```
        if(~reset) begin
                q <= 1'b0;
        end

        else begin
                q <= d;
        end
end

assign qb = ~q;

endmodule

module RPCNT17(

ripplecnt_clk,
ripplecnt_rst,
ripplecnt_bit

);

input ripplecnt_clk, ripplecnt_rst;
output [9:0] ripplecnt_bit;


wire qb0, qb1, qb2, qb3, qb4, qb5, qb6, qb7, qb8, qb9;
```

```
DFLIPFLOP ripplecnt_dff0(.clk(ripplecnt_clk), .reset(ripplecnt_rst),
.d(qb0), .q(ripplecnt_bit[0]), .qb(qb0));

DFLIPFLOP ripplecnt_dff1(.clk(qb0), .reset(ripplecnt_rst),
.d(qb1), .q(ripplecnt_bit[1]), .qb(qb1));

DFLIPFLOP ripplecnt_dff2(.clk(qb1), .reset(ripplecnt_rst),
.d(qb2), .q(ripplecnt_bit[2]), .qb(qb2));

DFLIPFLOP ripplecnt_dff3(.clk(qb2), .reset(ripplecnt_rst),
.d(qb3), .q(ripplecnt_bit[3]), .qb(qb3));

DFLIPFLOP ripplecnt_dff4(.clk(qb3), .reset(ripplecnt_rst),
.d(qb4), .q(ripplecnt_bit[4]), .qb(qb4));

DFLIPFLOP ripplecnt_dff5(.clk(qb4), .reset(ripplecnt_rst),
.d(qb5), .q(ripplecnt_bit[5]), .qb(qb5));

DFLIPFLOP ripplecnt_dff6(.clk(qb5), .reset(ripplecnt_rst),
.d(qb6), .q(ripplecnt_bit[6]), .qb(qb6));

DFLIPFLOP ripplecnt_dff7(.clk(qb6), .reset(ripplecnt_rst),
.d(qb7), .q(ripplecnt_bit[7]), .qb(qb7));
```

```
DFLIPFLOP ripplecnt_dff8(.clk(qb7), .reset(ripplecnt_rst),
.d(qb8), .q(ripplecnt_bit[8]), .qb(qb8));

DFLIPFLOP ripplecnt_dff9(.clk(qb8), .reset(ripplecnt_rst),
.d(qb9), .q(ripplecnt_bit[9]), .qb(qb9));

endmodule

module PD17(

input pd_clk,

input initial_clk,
input final_clk,

//input [33:0] pd_in,
input [16:0] pd_in_A,
input [16:0] pd_in_B,

output reg [5:0] pd_out_initial,
output reg [5:0] pd_out_final

);


always@(posedge pd_clk) begin
```

```
if (initial_clk) begin
        if ( (pd_in_A[0] == 0) && (pd_in_B[1] == 1) ) begin
                pd_out_initial <= 6'd0;
        end

        else if ( (pd_in_A[2] == 1) && (pd_in_B[1] == 0) ) begin
                pd_out_initial <= 6'd1;
        end

        else if ( (pd_in_A[2] == 0) && (pd_in_B[3] == 1) ) begin
                pd_out_initial <= 6'd2;
        end

        else if ( (pd_in_A[4] == 1) && (pd_in_B[3] == 0) ) begin
                pd_out_initial <= 6'd3;
        end

        else if ( (pd_in_A[4] == 0) && (pd_in_B[5] == 1) ) begin
                pd_out_initial <= 6'd4;
        end

        else if ( (pd_in_A[6] == 1) && (pd_in_B[5] == 0) ) begin
                pd_out_initial <= 6'd5;
        end

        else if ( (pd_in_A[6] == 0) && (pd_in_B[7] == 1) ) begin
```

```
        pd_out_initial <= 6'd6;
end

else if ( (pd_in_A[8] == 1) && (pd_in_B[7] == 0) ) begin
        pd_out_initial <= 6'd7;
end

else if ( (pd_in_A[8] == 0) && (pd_in_B[9] == 1) ) begin
        pd_out_initial <= 6'd8;
end

else if ( (pd_in_A[10] == 1) && (pd_in_B[9] == 0) ) begin
        pd_out_initial <= 6'd9;
end

else if ( (pd_in_A[10] == 0) && (pd_in_B[11] == 1) ) begin
        pd_out_initial <= 6'd10;
end

else if ( (pd_in_A[12] == 1) && (pd_in_B[11] == 0) ) begin
        pd_out_initial <= 6'd11;
end

else if ( (pd_in_A[12] == 0) && (pd_in_B[13] == 1) ) begin
        pd_out_initial <= 6'd12;
end
```

```
else if ( (pd_in_A[14] == 1) && (pd_in_B[13] == 0) ) begin
        pd_out_initial <= 6'd13;
end

else if ( (pd_in_A[14] == 0) && (pd_in_B[15] == 1) ) begin
        pd_out_initial <= 6'd14;
end

else if ( (pd_in_A[16] == 1) && (pd_in_B[15] == 0) ) begin
        pd_out_initial <= 6'd15;
end

else if ( (pd_in_A[16] == 0) && (pd_in_B[0] == 1) ) begin
        pd_out_initial <= 6'd16;
end

else if ( (pd_in_A[1] == 1) && (pd_in_B[0] == 0) ) begin
        pd_out_initial <= 6'd17;
end

else if ( (pd_in_A[1] == 0) && (pd_in_B[2] == 1) ) begin
        pd_out_initial <= 6'd18;
end

else if ( (pd_in_A[3] == 1) && (pd_in_B[2] == 0) ) begin
```

```
        pd_out_initial <= 6'd19;
end

else if ( (pd_in_A[3] == 0) && (pd_in_B[4] == 1) ) begin
        pd_out_initial <= 6'd20;
end

else if ( (pd_in_A[5] == 1) && (pd_in_B[4] == 0) ) begin
        pd_out_initial <= 6'd21;
end

else if ( (pd_in_A[5] == 0) && (pd_in_B[6] == 1) ) begin
        pd_out_initial <= 6'd22;
end

else if ( (pd_in_A[7] == 1) && (pd_in_B[6] == 0) ) begin
        pd_out_initial <= 6'd23;
end

else if ( (pd_in_A[7] == 0) && (pd_in_B[8] == 1) ) begin
        pd_out_initial <= 6'd24;
end

else if ( (pd_in_A[9] == 1) && (pd_in_B[8] == 0) ) begin
        pd_out_initial <= 6'd25;
end
```

```
else if ( (pd_in_A[9] == 0) && (pd_in_B[10] == 1) ) begin
        pd_out_initial <= 6'd26;
end

else if ( (pd_in_A[11] == 1) && (pd_in_B[10] == 0) ) begin
        pd_out_initial <= 6'd27;
end

else if ( (pd_in_A[11] == 0) && (pd_in_B[12] == 1) ) begin
        pd_out_initial <= 6'd28;
end

else if ( (pd_in_A[13] == 1) && (pd_in_B[12] == 0) ) begin
        pd_out_initial <= 6'd29;
end

else if ( (pd_in_A[13] == 0) && (pd_in_B[14] == 1) ) begin
        pd_out_initial <= 6'd30;
end

else if ( (pd_in_A[15] == 1) && (pd_in_B[14] == 0) ) begin
        pd_out_initial <= 6'd31;
end

else if ( (pd_in_A[15] == 0) && (pd_in_B[16] == 1) ) begin
```

```
                pd_out_initial <= 6'd32;
            end

            else if ( (pd_in_A[0] == 1) && (pd_in_B[16] == 0) ) begin
                pd_out_initial <= 6'd33;
            end

            else begin
                pd_out_initial <= 6'd0;
            end

        end

////////////////////////////////////////////////////////////////////////////

        else if (final_clk) begin

            if ( (pd_in_A[0] == 0) && (pd_in_B[1] == 1) ) begin
                pd_out_final <= 6'd0;
            end

            else if ( (pd_in_A[2] == 1) && (pd_in_B[1] == 0) ) begin
                pd_out_final <= 6'd1;
            end

            else if ( (pd_in_A[2] == 0) && (pd_in_B[3] == 1) ) begin
```

```
        pd_out_final <= 6'd2;
end

else if ( (pd_in_A[4] == 1) && (pd_in_B[3] == 0) ) begin
        pd_out_final <= 6'd3;
end

else if ( (pd_in_A[4] == 0) && (pd_in_B[5] == 1) ) begin
        pd_out_final <= 6'd4;
end

else if ( (pd_in_A[6] == 1) && (pd_in_B[5] == 0) ) begin
        pd_out_final <= 6'd5;
end

else if ( (pd_in_A[6] == 0) && (pd_in_B[7] == 1) ) begin
        pd_out_final <= 6'd6;
end

else if ( (pd_in_A[8] == 1) && (pd_in_B[7] == 0) ) begin
        pd_out_final <= 6'd7;
end

else if ( (pd_in_A[8] == 0) && (pd_in_B[9] == 1) ) begin
        pd_out_final <= 6'd8;
end
```

```
else if ( (pd_in_A[10] == 1) && (pd_in_B[9] == 0) ) begin
        pd_out_final <= 6'd9;
end

else if ( (pd_in_A[10] == 0) && (pd_in_B[11] == 1) ) begin
        pd_out_final <= 6'd10;
end

else if ( (pd_in_A[12] == 1) && (pd_in_B[11] == 0) ) begin
        pd_out_final <= 6'd11;
end

else if ( (pd_in_A[12] == 0) && (pd_in_B[13] == 1) ) begin
        pd_out_final <= 6'd12;
end

else if ( (pd_in_A[14] == 1) && (pd_in_B[13] == 0) ) begin
        pd_out_final <= 6'd13;
end

else if ( (pd_in_A[14] == 0) && (pd_in_B[15] == 1) ) begin
        pd_out_final <= 6'd14;
end

else if ( (pd_in_A[16] == 1) && (pd_in_B[15] == 0) ) begin
```

```
        pd_out_final <= 6'd15;
end

else if ( (pd_in_A[16] == 0) && (pd_in_B[0] == 1) ) begin
        pd_out_final <= 6'd16;
end

else if ( (pd_in_A[1] == 1) && (pd_in_B[0] == 0) ) begin
        pd_out_final <= 6'd17;
end

else if ( (pd_in_A[1] == 0) && (pd_in_B[2] == 1) ) begin
        pd_out_final <= 6'd18;
end

else if ( (pd_in_A[3] == 1) && (pd_in_B[2] == 0) ) begin
        pd_out_final <= 6'd19;
end

else if ( (pd_in_A[3] == 0) && (pd_in_B[4] == 1) ) begin
        pd_out_final <= 6'd20;
end

else if ( (pd_in_A[5] == 1) && (pd_in_B[4] == 0) ) begin
        pd_out_final <= 6'd21;
end
```

```
else if ( (pd_in_A[5] == 0) && (pd_in_B[6] == 1) ) begin
        pd_out_final <= 6'd22;
end

else if ( (pd_in_A[7] == 1) && (pd_in_B[6] == 0) ) begin
        pd_out_final <= 6'd23;
end

else if ( (pd_in_A[7] == 0) && (pd_in_B[8] == 1) ) begin
        pd_out_final <= 6'd24;
end

else if ( (pd_in_A[9] == 1) && (pd_in_B[8] == 0) ) begin
        pd_out_final <= 6'd25;
end

else if ( (pd_in_A[9] == 0) && (pd_in_B[10] == 1) ) begin
        pd_out_final <= 6'd26;
end

else if ( (pd_in_A[11] == 1) && (pd_in_B[10] == 0) ) begin
        pd_out_final <= 6'd27;
end

else if ( (pd_in_A[11] == 0) && (pd_in_B[12] == 1) ) begin
```

```
        pd_out_final <= 6'd28;
end

else if ( (pd_in_A[13] == 1) && (pd_in_B[12] == 0) ) begin
        pd_out_final <= 6'd29;
end

else if ( (pd_in_A[13] == 0) && (pd_in_B[14] == 1) ) begin
        pd_out_final <= 6'd30;
end

else if ( (pd_in_A[15] == 1) && (pd_in_B[14] == 0) ) begin
        pd_out_final <= 6'd31;
end

else if ( (pd_in_A[15] == 0) && (pd_in_B[16] == 1) ) begin
        pd_out_initial <= 6'd32;
end

else if ( (pd_in_A[0] == 1) && (pd_in_B[16] == 0) ) begin
        pd_out_final <= 6'd33;
end

else begin
        pd_out_final <= 6'd0;
end
```

```
        end

end


endmodule

module RP_PD_COMB(

input mclk,
input rst,

input phase_sample_clk,

input initial_clk,
input final_clk,

input vcogate_clk, ctr_rst,
input adc_sample_clk,

input vcofreq,

input [16:0] pd_in_A,
input [16:0] pd_in_B,


output reg [16:0] pd_A_buffer,
```

```
output reg [16:0] pd_B_buffer,

output rpcnt_inputclk,
output [9:0]rpcnt_output,

output [5:0] pd_out_initial,
output [5:0] pd_out_final,


output reg [15:0] adc_out

);



assign rpcnt_inputclk = ~(vcofreq & vcogate_clk);

RPCNT17 RPCNT17_INST1(.ripplecnt_clk(rpcnt_inputclk),
.ripplecnt_rst(ctr_rst), .ripplecnt_bit(rpcnt_output));

PD17 PD17_INST1(.pd_clk(mclk), .initial_clk(initial_clk),
 .final_clk(final_clk), .pd_in_A(pd_A_buffer_wire),
 .pd_in_B(pd_B_buffer_wire), .pd_out_initial(pd_out_initial),
 .pd_out_final(pd_out_final));
```

```
wire [15:0] rpcntX16;
wire [15:0] rpcntX2;

wire [16:0] pd_A_buffer_wire;
wire [16:0] pd_B_buffer_wire;


assign rpcntX16[15:0] = {2'b00, rpcnt_output[9:0], 4'b0000};
assign rpcntX2[15:0] = {5'b00000, rpcnt_output[9:0], 1'b0};

assign pd_A_buffer_wire = pd_A_buffer;
assign pd_B_buffer_wire = pd_B_buffer;


// Phase Sampler
always@(posedge mclk) begin
        if (rst) begin
                pd_A_buffer <= 17'd0;
                pd_B_buffer <= 17'd0;
        end

        else if (phase_sample_clk) begin
                pd_A_buffer <= pd_in_A;
                pd_B_buffer <= pd_in_B;
        end
```

```
end



// Phase Combination
always@(posedge mclk) begin
        if (rst) begin
                adc_out <= 16'd0;
        end

        else if (adc_sample_clk) begin
                adc_out <= rpcntX16 + rpcntX16 + rpcntX2
                        + pd_out_final - pd_out_initial;
        end

end

endmodule
```

Appendix C

PCB Schematics

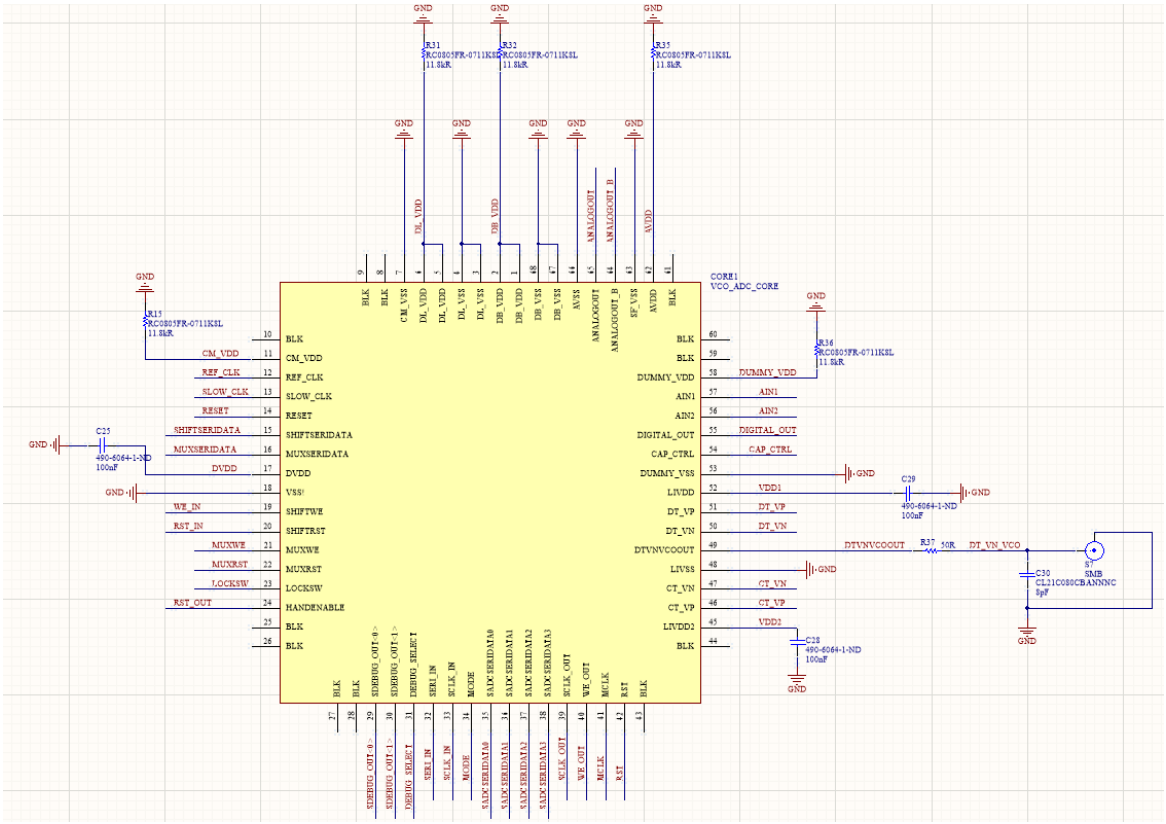


Figure C.1: Test circuits around the chip

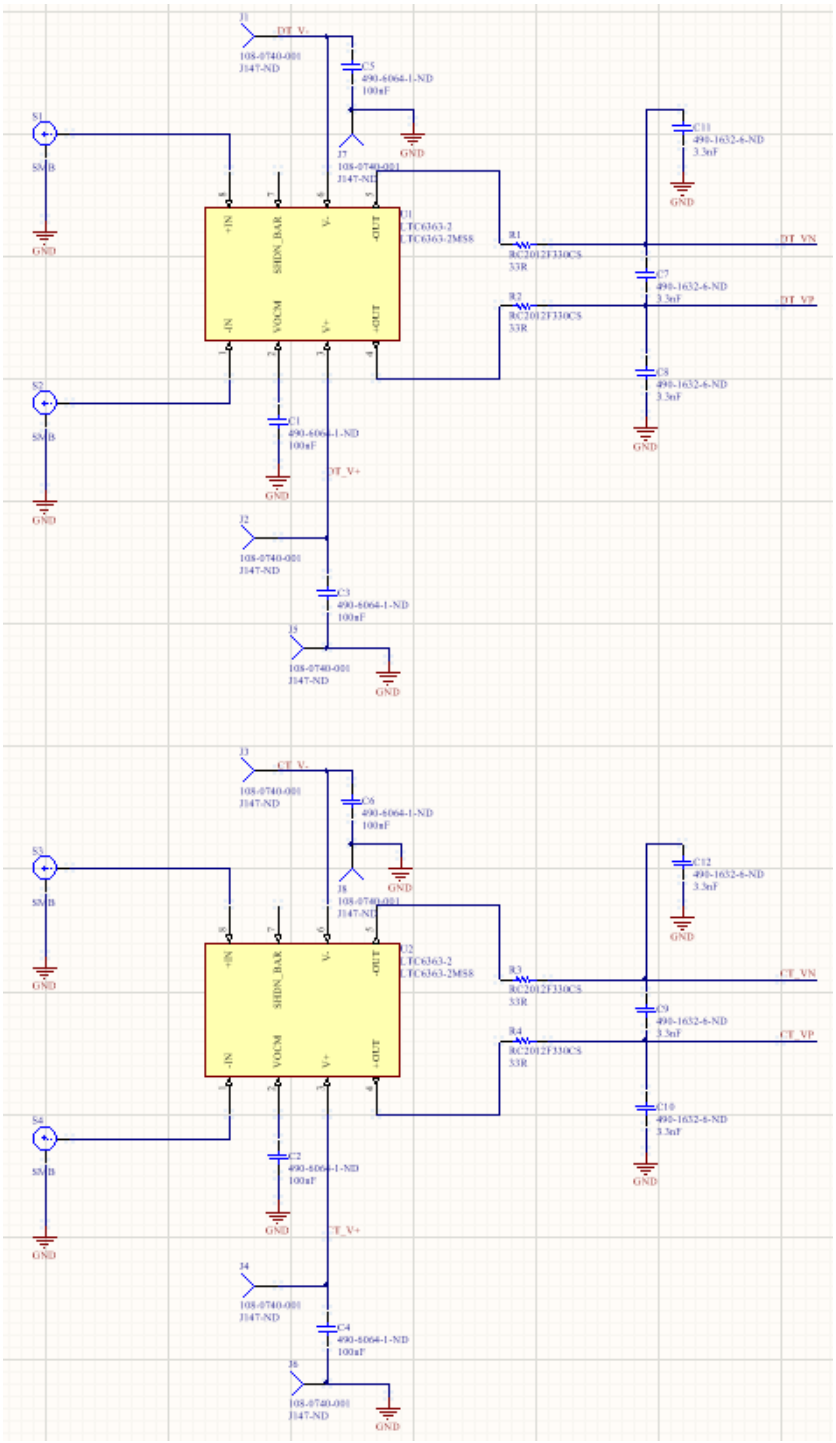


Figure C.2: ADC drivers

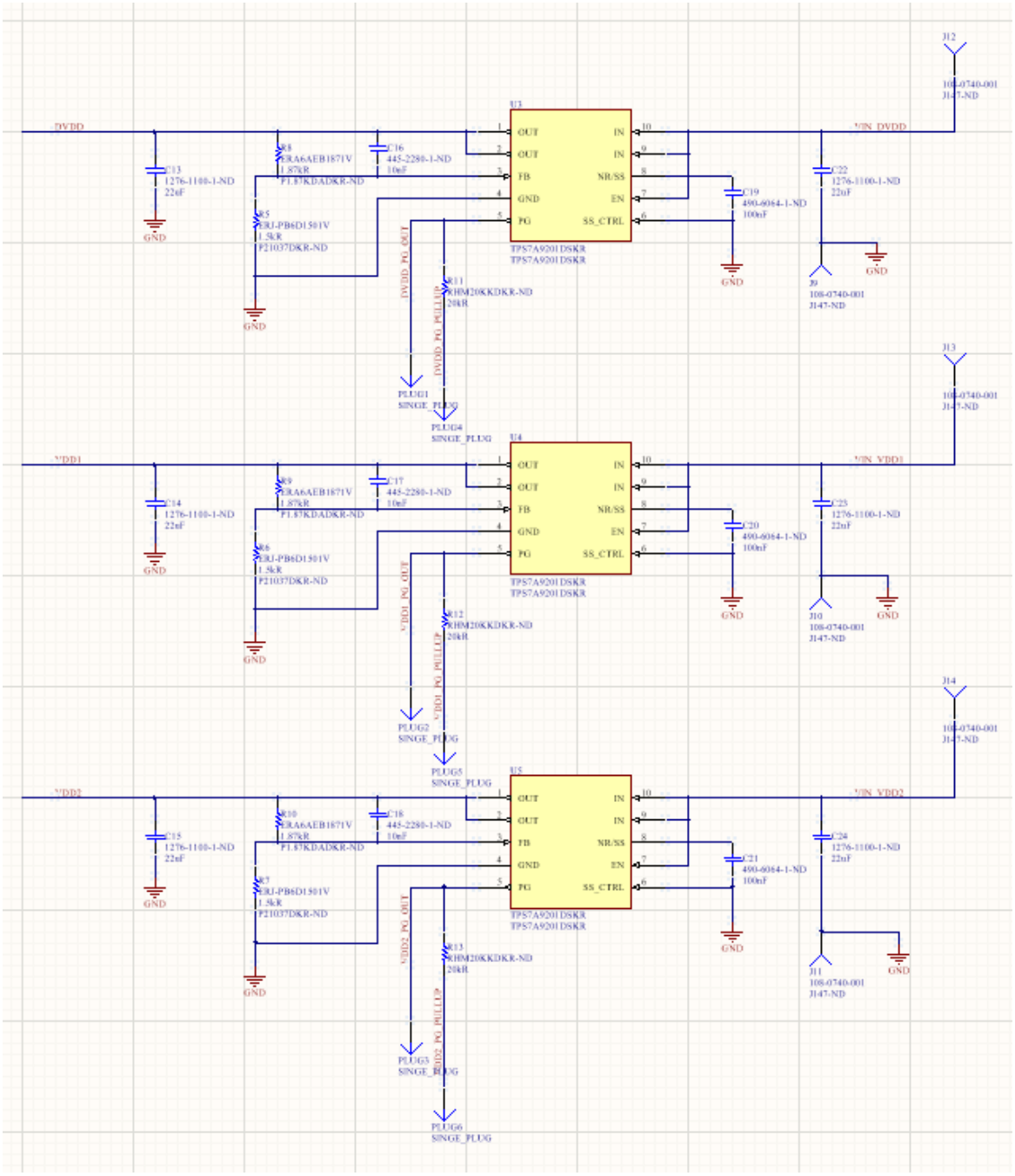


Figure C.3: LDOs

Bibliography

[1] J. McNeill, M. C. Coln, and B. J. Larivee, "A 'Split ADC' Architecture for Deterministic Digital Background Calibration of a 16-bit 1-MS/s ADC," in *International Solid-State Circuits Conference*, pp. 276–277, 2005.

[2] J. McNeill, M. C. Coln, and B. J. Larivee, "A 'Split ADC' Architecture for Deterministic Digital Background Calibration of a 16-bit 1-MS/s ADC," *IEEE Journal of Solid-State Circuits*, vol. 40, no. 12, pp. 2437–2445, 2005.

[3] J. A. McNeill, M. C. Coln, D. R. Brown, and B. J. Larivee, "Digital Background Calibration Algorithm for 'Split ADC' Architecture," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 56, no. 2, pp. 294–306, 2009.

[4] B. Murmann, "ADC Performance Survey 1997-2018," in *[Online]. Available: http://web.stanford.edu/ murmann/adcsurvey.html.*, 2018.

[5] G. Taylor and I. Galton, "A Mostly-Digital Variable-Rate Continuous-Time Delta-Sigma Modulator ADC," *IEEE Journal of Solid-State Circuits*, vol. 45, no. 12, pp. 2634–2646, 2010.

[6] J. A. McNeill, R. Majidi, and J. Gong, "'Split ADC' Background Linearization of VCO-Based ADCs," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 62, no. 1, pp. 49–58, 2015.

[7] G. E. Moore, "No Exponential is Forever: but" Forever" Can Be Delayed!," in *2003 IEEE International Solid-State Circuits Conference, 2003. Digest of Technical Papers. ISSCC.*, pp. 20–23, IEEE, 2003.

[8] P. R. Kinget, "Device Mismatch and Tradeoffs in the Design of Analog Circuits," *IEEE Journal of Solid-State Circuits*, vol. 40, no. 6, pp. 1212–1224, 2005.

[9] B. Razavi, "CMOS Technology Characterization for Analog and RF Design," *IEEE Journal of Solid-State Circuits*, vol. 34, no. 3, pp. 268–276, 1999.

[10] J. McNeill, S. Li, J. Gong, and L. Pham, "Fundamental Limits on Energy Efficiency Performance of VCO-based ADCs," in *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–4, IEEE, 2017.

[11] S. Li, J. Gong, and J. A. McNeill, "VCO-Based ADC With Digital Background Calibration in 65nm CMOS," in *2018 16th IEEE International New Circuits and Systems Conference (NEWCAS)*, pp. 195–199, IEEE, 2018.

[12] J. Gong, S. Li, and J. A. McNeill, "Sub-picosecond-jitter Clock Generation for Interleaved ADC with Delay-Locked-Loop in 28nm CMOS," in *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 2763–2766, IEEE, 2016.

[13] W. Kester and A. D. I. Engineeri, *Data Conversion Handbook*. Newnes, 2005.

[14] D. A. Johns and K. Martin, *Analog Integrated Circuit Design*. John Wiley & Sons, 2008.

[15] J. Kim, T.-K. Jang, Y.-G. Yoon, and S. Cho, "Analysis and Design of Voltage-controlled oscillator Based Analog-to-digital Converter," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 57, no. 1, pp. 18–30, 2010.

[16] M. Z. Straayer and M. H. Perrott, "A 12-bit, 10-MHz Bandwidth, Continuous-Time $\Delta\Sigma$ ADC With a 5-bit, 950-MS/s VCO-Based Quantizer," *IEEE J. Solid-State Circuits*, vol. 43, no. 4, pp. 805–814, 2008.

[17] M. Park and M. H. Perrott, "A 78 dB SNDR 87 mW 20 MHz Bandwidth Continuous-Time $\Delta\Sigma$ ADC With VCO-Based Integrator and Quantizer Implemented in 0.13 *mu*m CMOS," *IEEE J. Solid-State Circuits*, vol. 44, no. 12, pp. 3344–3358, 2009.

[18] S. Rao, B. Young, A. Elshazly, W. Yin, N. Sasidhar, and P. K. Hanumolu, "A 71dB SFDR Open Loop VCO-Based ADC Using 2-level PWM Modulation," in *2011 Symposium on VLSI Circuits-Digest of Technical Papers*, pp. 270–271, IEEE, 2011.

[19] K. Reddy, S. Rao, R. Inti, B. Young, A. Elshazly, M. Talegaonkar, and P. K. Hanumolu, "A 16-mW 78-dB SNDR 10-MHz BW CT $\Delta\Sigma$ ADC Using Residue-Cancelling VCO-Based Quantizer," *IEEE journal of solid-state circuits*, vol. 47, no. 12, pp. 2916–2927, 2012.

[20] G. Taylor and I. Galton, "A Reconfigurable Mostly-Digital Delta-Sigma ADC With a Worst-Case FOM of 160 dB," *IEEE Journal of Solid-State Circuits*, vol. 48, no. 4, pp. 983–995, 2013.

[21] S. Rao, K. Reddy, B. Young, and P. K. Hanumolu, "A Deterministic Digital Background Calibration Technique for VCO-based ADCs," *IEEE Journal of Solid-State Circuits*, vol. 49, no. 4, pp. 950–960, 2014.

[22] B. Young, K. Reddy, S. Rao, A. Elshazly, T. Anand, and P. K. Hanumolu, "A 75dB DR 50MHz BW 3rd Order CT-$\Delta\Sigma$ Modulator Using VCO-Based Integrators," in *2014 Symposium on VLSI Circuits Digest of Technical Papers*, pp. 1–2, IEEE, 2014.

[23] P. Zhu, X. Xing, and G. Gielen, "A 40MHz-BW 35fJ/step-FoM Nonlinearity-Cancelling Two-Step ADC with Dual-Input VCO-Based Quantizer," in *ESSCIRC 2014-40th European Solid State Circuits Conference (ESSCIRC)*, pp. 63–66, IEEE, 2014.

[24] A. Sanyal, K. Ragab, L. Chen, T. Viswanathan, S. Yan, and N. Sun, "A Hybrid SAR-VCO $\Delta\Sigma$ ADC with First-Order Noise Shaping," in *Proceedings of the IEEE 2014 Custom Integrated Circuits Conference*, pp. 1–4, IEEE, 2014.

[25] X. Xing and G. G. Gielen, "A 42 fJ/Step-FoM Two-Step VCO-Based Delta-Sigma ADC in 40 nm CMOS," *IEEE Journal of Solid-State Circuits*, vol. 50, no. 3, pp. 714–723, 2015.

[26] A. Ghosh and S. Pamarti, "Linearization Through Dithering: A 50 MHz Bandwidth, 10-b ENOB, 8.2 mW VCO-Based ADC," *IEEE Journal of Solid-State Circuits*, vol. 50, no. 9, pp. 2012–2024, 2015.

[27] K. Reddy, S. Dey, S. Rao, B. Young, P. Prabha, and P. K. Hanumolu, "A 54mW 1.2 GS/s 71.5 dB SNDR 50MHz BW VCO-Based CT $\Delta\Sigma$ ADC Using Dual Phase/Frequency Feedback in 65nm cmos," in *2015 Symposium on VLSI Circuits (VLSI Circuits)*, pp. C256–C257, IEEE, 2015.

[28] K. Ragab and N. Sun, "A 12-b ENOB 2.5-MHz BW VCO-Based 0-1 MASH ADC With Direct Digital Background Calibration," *IEEE Journal of Solid-State Circuits*, vol. 52, no. 2, pp. 433–447, 2017.

[29] M. Baert and W. Dehaene, "A 5GS/s 7.2 ENOB Time-Interleaved VCO-Based ADC Achieving 30.5 fJ/conv-step," in *2019 IEEE International Solid-State Circuits Conference-(ISSCC)*, pp. 328–330, IEEE, 2019.

[30] S. Y. J. Hamilton and T. R. Viswanathan, "An Uncalibrated 2MHz, 6mW, 63.5dB SNDR Discrete Time Input VCO-based ADC," in *IEEE Custom Integrated Circuits Conference (CICC2012)*, 2012.

[31] H. V. J. G. M. Gande, H. Lee and U.-K. Moon, "Blind Background Calibration of Harmonic Distortion Based on Selective Sampling," in *IEEE Custom Integrated Circuits Conference (CICC2013)*, 2013.

[32] Y. Y. K. Lee and N. Sun, "1.8mW 2MHz-BW 66.5dB-SNDR ADC Using VCO-Based Integrators with Intrinsic CLA," in *IEEE Custom Integrated Circuits Conference (CICC2013)*, 2013.

[33] G. I. Taylor. G, "A Reconfigurable Mostly-Digital ADC With a Worst-case FOM of 160dB," in *Symposium on VLSI Circuits (VLSIC)*, 2012.

[34] P. Sharma and M. S.-W. Chen, "A 6b 800MS/s 3.62mW Nyquist AC-Coupled VCO-Based ADC in 65nm CMOS," in *IEEE Custom Integrated Circuits Conference (CICC2013)*, 2013.

[35] W. X. B. Kim and C. H. Kim, "Fully-Digital Beat-Frequency Based ADC Achieving 39dB SNDR for a 1.6mVpp Input Signal," in *IEEE Custom Integrated Circuits Conference (CICC2013)*, 2013.

[36] K. N. A. Gupta and T. Viswanathan, "A Two-stage ADC Architecture with VCO-based Second Stage," in *IEEE Trans. Circuits Syst. II*, 2011.

[37] M. S. A. W. J. Daniels, W. Dehaene, "A $0.02mm^2$ 65nm CMOS 30MHz BW All-Digital Differential VCO-Based ADC with 64dB SNDR," in *IEEE Symposium on VLSI Circuits(VLSIC)*, 2010.

[38] A. Ghosh and S. Pamarti, "A 50MHz Bandwidth, 10-b ENOB, 8.2mW VCO-Based ADC Enabled by Filtered-dithering Based Linearization," in *IEEE Custom Integrated Circuits Conference (CICC2013)*, 2013.

[39] J. A. McNeill and D. Ricketts, *The Designer's Guide to Jitter in Ring Oscillators*. Springer Science & Business Media, 2009.

[40] J. A. McNeill, "Jitter in Ring Oscillators," *IEEE Journal of Solid-State Circuits*, vol. 32, no. 6, pp. 870–879, 1997.

[41] C. Liu and J. A. McNeill, "Jitter in Oscillators with 1/f Noise Sources," in *2004 IEEE International Symposium on Circuits and Systems (IEEE Cat. No. 04CH37512)*, vol. 1, pp. I–773, IEEE, 2004.

[42] J. A. McNeill, "Noise in Short Channel MOSFETs," in *2009 IEEE Custom Integrated Circuits Conference*, pp. 567–572, IEEE, 2009.

[43] C.-H. Chen, D. Chen, R. Lee, P. Lei, and D. Wan, "Thermal Noise modeling of Nano-scale MOSFETs for Mixed-signal and RF applications," in *Proceedings of the IEEE 2013 Custom Integrated Circuits Conference*, pp. 1–8, IEEE, 2013.

[44] J. Li and U.-K. Moon, "Background Calibration Techniques for Multistage Pipelined ADCs with Digital Redundancy," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 50, no. 9, pp. 531–538, 2003.

[45] A. Baschirotto, R. Castello, F. Campi, G. Cesura, M. Toma, R. Guerrieri, R. Lodi, L. Lavagno, and P. Malcovati, "Baseband Analog Front-End and Digital Back-End for Reconfigurable Multi-standard Terminals," *IEEE Circuits and Systems Magazine*, vol. 6, no. 1, pp. 8–28, 2006.

[46] B. Peng, H. Li, S.-C. Lee, P. Lin, and Y. Chiu, "A Virtual-ADC Digital Background Calibration Technique for Multistage A/D Conversion," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 57, no. 11, pp. 853–857, 2010.

[47] H. Wang, X. Wang, P. J. Hurst, and S. H. Lewis, "Nested Digital Background Calibration of a 12-bit Pipelined ADC Without an Input SHA," *IEEE Journal of Solid-State Circuits*, vol. 44, no. 10, pp. 2780–2789, 2009.

[48] K. Iizuka, H. Matsui, M. Ueda, and M. Daito, "A 14-bit Digitally Self-calibrated Pipelined ADC with Adaptive Bias Optimization for Arbitrary Speeds up to 40 MS/s," *IEEE Journal of Solid-State Circuits*, vol. 41, no. 4, pp. 883–890, 2006.

[49] B. R. Gregoire and U.-K. Moon, "An Over-60 dB True Rail-to-Rail Performance Using Correlated Level Shifting and an Opamp With only 30 dB Loop Gain," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 12, pp. 2620–2630, 2008.

[50] M. Taherzadeh-Sani and A. A. Hamoui, "Digital Background Calibration of a 0.4-pJ/step 10-bit Pipelined ADC Without PN Generator in 90nm Digital CMOS," in *IEEE Asian Solid-State Circuits Conference*, pp. 53–56, 2008.

[51] I. Ahmed and D. A. Johns, "An 11-bit 45 MS/s Pipelined ADC With Rapid Calibration of DAC Errors in a Multibit Pipeline Stage," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 12, pp. 1626–1637, 2008.

[52] L. H. Hung *et al.*, "A 10-bit Pipelined A/D Converter with Split Calibration and Opamp-sharing Technique," in *International Symposium on VLSI Design Automation and Test*, pp. 190–193, 2010.

[53] C. Tsang *et al.*, "Background ADC Calibration in Digital Domain," in *IEEE Custom Integrated Circuits Conference*, pp. 301–1304, 2008.

[54] B. Murmann and B. E. Boser, "Digital Domain Measurement and Cancellation of Residue Amplifier Nonlinearity in Pipelined ADCs," *IEEE Transactions on Instrumentation and Measurement*, vol. 56, pp. 2504–2514, 2007.

[55] K. W. Hsueh *et al.*, "A 1V 11b 200MS/s Pipelined ADC with Digital Background Calibration in 65nm CMOS," in *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, pp. 546–634, 2008.

[56] N. Petrellis *et al.*, "An Ultra Low Area Asynchronous Combo 4/8/12-bit/quaternary A/D Converter Based on Integer Division," *Microelectronics Journal*, vol. 41, pp. 291–307, 2010.

[57] F. Centurelli *et al.*, "A Simple Technique for Fast Digital Background Calibration of A/D Converters," *EURASIP Journal on Advances in Signal Processing*, 2008.

[58] T. Moosazadeh and M. Yavari, "A Novel Digital Background Calibration Technique for Pipelined ADCs," in *2010 5th European Conference on Circuits and Systems for Communications (ECCSC)*, pp. 127–130, 2010.

[59] T. Moosazadeh and M. Yavari, "A Simple Digital Background Gain Error Calibration Technique for Pipelined ADCs," in *18th Iranian Conference on Electrical Engineering (ICEE)*, pp. 437–441, 2010.

[60] A. Jalili *et al.*, "A Frequency Based Digital Background Calibration Technique for Pipelined ADCs," in *15th IEEE International Conference on Electronics*, pp. 1209–1212, 2008.

[61] L. H. Hung and T. C. Lee, "A Split-Based Digital Background Calibration Technique in Pipelined ADCs," *IEEE Transactions on Circuits and Systems II*, vol. 56, pp. 855–859, 2009.

[62] C. Francesco *et al.*, "A Simple Technique for Fast Digital Background Calibration of A/D Converters," *EURASIP Journal on Advances in Signal Processing*, 2008.

[63] B. Peng *et al.*, "An Offset Double Conversion Technique for Digital Calibration of Pipelined ADCs," *IEEE Transactions on Circuits and Systems Ii-Express Briefs*, vol. 57, pp. 961–965, 2010.

[64] J. L. Fan *et al.*, "A Robust and Fast Digital Background Calibration Technique for Pipelined ADCs," *IEEE Transactions on Circuits and Systems I*, vol. 54, pp. 1213–1223, 2007.

[65] K. Lee *et al.*, "A Noise-Coupled Time-Interleaved Delta-Sigma ADC With 4.2 MHz Bandwidth, -98 dB THD, and 79 dB SNDR," *IEEE Journal of Solid-State Circuits*, vol. 43, pp. 2601–2612, 2008.

[66] K. Lee and G. C. Temes, "Enhanced Split-Architecture Delta-Sigma ADCs," *Analog Integrated Circuits and Signal Processing*, vol. 56, pp. 251–257, 2008.

[67] K. Lee and G. C. Temes, "Enhanced Split-Architecture Delta-Sigma ADC," *Electronics Letters*, vol. 42, pp. 737–739, 2006.

[68] K. Lee, G. C. Temes, and F. Maloberti, "Noise-Coupled Multi-Cell Delta-Sigma ADCs," in *2007 IEEE International Symposium on Circuits and Systems*, pp. 249–252, IEEE, 2007.

[69] S. Pamarti, "A Theoretical Study of the Quantization Noise in Split Delta-Sigma ADCs," *IEEE Transactions on Circuits and Systems I-Regular Papers*, vol. 55, pp. 1267–1278, 2008.

[70] A. C. R. Majidi and J. McNeill, "Digital Background Calibration of Redundant Split-Flash ADC," in *IEEE International Symposium on Circuits and Systems*, 2012.

[71] Y. Nakajima *et al.*, "A Background Self-Calibrated 6b 2.7 GS/s ADCWith Cascade-Calibrated Folding-Interpolating Architecture," *IEEE Journal of Solid-State Circuits*, vol. 45, pp. 707–718, 2010.

[72] M. C. R. C. J. McNeill, C. David, "'Split ADC' Calibration for All-Digital Correction of Time Interleaved ADC Errors," *IEEE Transactions on Circuits and Systems II*, vol. 56, pp. 344–348, 2009.

[73] T. Strohmer and J. Xu, "Fast Algorithms for Blind Calibration in Time-Interleaved Analog-to-Digital Converters," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2007.

[74] M. C. C. D. J. A. McNeill, K. Chan and C. Brenneman, "All-Digital Background Calibration of a Successive Approximation ADC using the Split ADC Architecture," *IEEE Transactions on Circuits and Systems II*, vol. 58, pp. 2355–2365, 2011.

[75] W. Liu *et al.*, "A 12b 22.5/45MS/s 3.0 mW 0.059 mm2 CMOS SAR ADC Achieving Over 90dB SFDR," in *IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, 2010.

[76] N. Petrellis *et al.*, "Calibration Method for a CMOS 0.06 mm^2 150MS/s 8-bit ADC," in *2th Euromicro Conference on Digital System Design, Architectures, Methods and Tools DSD09*, 20091.

[77] M. C. K. C. J. A. McNeill, C. David and C. Brenneman, "'Split ADC' Background Self-Calibration of a 16-b Successive Approximation ADC in 180nm CMOS," in *Proceedings of the 2013 International Instrumentation and Technology Conference*, 2013.

[78] B. Razavi and R. Behzad, *RF Microelectronics*, vol. 1. Prentice Hall New Jersey, 1998.

[79] N. Sasaki, "Higher Harmonic Generation in CMOS/SOS Ring Oscillators," *IEEE transactions on Electron Devices*, vol. 29, no. 2, pp. 280–283, 1982.

[80] R. F. Pierret, *Semiconductor device fundamentals*. Pearson Education India, 1996.

[81] B. Razavi, "The Bootstrapped Switch [a circuit for all seasons]," *IEEE Solid-State Circuits Magazine*, vol. 7, no. 3, pp. 12–15, 2015.

[82] N. H. Weste and K. Eshraghian, "Principles of CMOS VLSI Design: A Systems Perspective," *NASA STI/Recon Technical Report A*, vol. 85, 1985.

[83] J. M. Rabaey, A. P. Chandrakasan, and B. Nikolic, *Digital Integrated Circuits*, vol. 2. Prentice hall Englewood Cliffs, 2002.

[84] J. McNeill, R. Majidi, J. Gong, and C. Liu, "Lookup-Table-Based Background Linearization for VCO-Based ADCs," in *Proceedings of the IEEE 2014 Custom Integrated Circuits Conference*, pp. 1–4, IEEE, 2014.

[85] R. Hamming, *Numerical Methods for Scientists and Engineers*. Courier Corporation, 2012.

[86] J. McNeill, C. David, M. C. Coln, K. Y. Chan, and C. Brenneman, "Split ADC Background Self-Calibration of a 16-b Successive Approximation ADC in 180nm CMOS," in *2013 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, pp. 310–313, 2013.