FE8113 High Speed Data Converters: A Summary

Carsten Wulff

February 15, 2006

Preface

This document is a summary of the curriculum in *FE8113 High Speed Data Converters*. The goal of this document is to summarize the curriculum so that the results are presented. Proof of equations and statements can be found in the references. The focus of the summary is as follows:

1. Performance limitations of high-speed ADCs.
2. Suitable architectures for high-speed ADCs.
3. An overview of calibration algorithms for pipelined ADCs.
4. Description, analysis and discussion of recent state-of-the-art ADC publications.

Words About Writing

This document is written about a field that is highly specialized, therefore there are words and abbreviations not commonly used in the English language. Some abbreviations have become so prevalent that they are not even capitalized in some texts. One of those words is Operational Amplifier (OPAMP) which will in this document be written as opamp, without capitalization.

Acknowledgements

There are some people that deserve to be thanked.

- My son Villem for his smile that can light up the darkest hour[1].
- My fiance Anita for her support, excellent writing advice and taking care of Villem so I could concentrate on my writing.

[1] Even at 6'o clock in the morning after I've had 3.5 hours of sleep

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter will give an introduction to some of the background information needed to understand the rest of the report. For a deeper introduction to data converters we suggest reading Chapter 11 in [1] or Chapters 1 and 2 in [2]. We start this chapter by describing two phenomena that limit performance of data converters: noise and distortion. The two phenomena both serve to reduce the available dynamic range of a data converter.

1.1 Noise

Noise can be found in any analog, and to some extent, in digital systems. Noise sources are often divided into two categories: *intrinsic* and *extrinsic*. *Intrinsic* refers to some inherent property of the system, whether it be physical or a property of the signal processing. *Extrinsic* refers to an external influence, for example leakage of signals from one portion of the system to another. In this section we will describe the *Intrinsic* phenomena of noise.

Noise manifests itself as random fluctuation of a signal. In the output signal of a analog circuit block we will always have noise. These random fluctuations may have different power spectral densities, and thus affect the system in different ways. There are three main noise sources: thermal noise, shot noise and flicker noise. Thermal noise stem from the random fluctuation of charge carriers, shot noise from charge carriers moving across a potential barrier and flicker noise from the random trapping and release of charge carriers. Thermal noise and flicker noise are the dominating noise sources in MOSFET transistors. In high-speed converters, techniques are often employed to reduce flicker noise. Thus the dominating noise source in high-speed analog-to-digital converters (ADCs) is often thermal noise. For a detailed, and at times complex, explanation of the sources and properties of noise, we refer to [3].

The accumulated noise in a system place a lower limit on the resolution of the system. If the signal power drops below the noise power of a system the signal is difficult to detect. For your convenience, we have included a short document which gives an introduction to the mathematics of noise sources in Appendix B.

1.2 Distortion

When it comes to data converters, it is important to have sufficient linearity. Non-linearities will lead to distortion of the converted signal, and degrade the available dynamic range. To see how this occurs, we can define a simple model for the ADC. The output, y_{out}, of a ADC for a sinusoidal input can be written as

$$y_{out} = f(x), x = A\cos(\omega t) \tag{1.1}$$

where $f(x)$ is the system function, A is the amplitude, t is time and ω is the angular input frequency of the system. For a linear ADC $f(x)$ is often approximated by

$$f(x) = x + e_n \tag{1.2}$$

where e_n is a noise component. Thus the output would be

$$y_{out} = A\cos(\omega t) + e_n \tag{1.3}$$

A real converter is usually not linear, we will later discuss some of the sources that lead to non-linearity in high-speed ADCs. If the system function is weakly non-linear we can approximate $f(x)$ using a Taylor series expansion. In this example we will use a Taylor series expansion around zero. The system function $f(x)$ then becomes

$$f(x) = K_1x + K_2x^2 + K_3x^3 + ... + K_ix^i + e_n \tag{1.4}$$

where the coefficients K_i is given by

$$K_i = \frac{1}{i!}\frac{d^i f(0)}{dx} \tag{1.5}$$

We can calculate the output as a function of the input using (1.4); we will only include the first three terms.

$$y_{out} = K_1A\cos(\omega t) + K_2A^2\cos^2(\omega t) + K_3A^3\cos^3(\omega t) + e_n \tag{1.6}$$

By using the well know relation

$$\cos a\cos b = \frac{1}{2}[\cos(a-b) + \cos(a+b)] \tag{1.7}$$

we can rewrite (1.6) as

$$y_{out} = K_1A\cos(\omega t) + \frac{K_2A^2}{2}[1 + \cos(2\omega t)] + \frac{K_3A^3}{4}[3\cos(\omega t) + \cos(3\omega t)] \tag{1.8}$$

In general, we can say that for a weakly non-linear system with a single sinusoid excitation we will have harmonics in the output signal at $n\omega$ where n is an integer. If we have two or more sinusoidal input signals there will, in addition to harmonics, be inter-modulation products at $k\omega_1 \pm n\omega_2$, where ω_1 and ω_2 are the input signal frequencies and k and n are integers.

Since most analog and mixed signal integrated circuits use differential signaling it is useful to know how distortion behaves in a differential circuit. In addition to improve signal to noise ratio [1], a differential system suppress even order distortion. The output of a differential circuit can be defined as:

$$y_{out} = f_1(x) - f_2(-x) \tag{1.9}$$

where $f_k(x)$ are the individual non-linear transfer functions for the differential paths. We define $f_k(x)$ as

$$f_k(x) = K_{0k} + K_{1k}x + K_{2k}x^2 + K_{3k}x^3 \tag{1.10}$$

where K_{ik} are the distortion coefficients defined in (1.5). K_{0k} is the zero order distortion (DC) resulting from for example offsets. When we calculate the output y_{out} we get

$$y_{out} = K_{01} - K_{02} + [K_{11} + K_{12}]x + [K_{21} - K_{22}]x^2 + [K_{31} + K_{32}]x^3 \tag{1.11}$$

[1] Signals add linearly when combined after a differential system. For example a sinusoid with an amplitude of A in the differential paths the amplitude would be $2A$ after combination, as shown by (1.12). Assuming uncorrelated noise sources in the two differential paths with noise power of e_{n1}^2 and e_{n2}^2 the output noise power would be $e_{nout}^2 = e_{n1}^2 + e_{n2}^2$. If the noise sources have the same power the output root mean square value would be $e_{nout} = \sqrt{2}e_n$. Thus the signal to noise ratio improves with a factor of $\sqrt{2}$, since

$$S/N = \frac{2A}{\sqrt{2}e_n} = \sqrt{2}\frac{A}{e_n}$$

If $K_{i1} = K_{i2}$ the equation reduces to

$$y_{out} = 2K_{11}x + 2K_{31}x^3 \tag{1.12}$$

Equation (1.12) proves that even order distortion is removed if the distortion in the in differential paths are equal. However, there is usually some mismatch between the the two differential paths which result in even order distortion being suppressed, but not removed.

1.3 Digital Coding

There are many different ways to encode numbers with ones an zeros. If codes or coding schemes like offset binary, twos complement, thermometer code, gray code or circular code are unfamiliar we suggest reading Table 2.1 and Chapter 3.3 in [2]

1.4 Classification of Signals

It is useful to get the signal classifications more accurate than just "analog" and "digital". A pure analog signal is continuous in time and continuous in value. A pure digital signal is discrete in time and discrete in value. To convert from an analog signal to a digital signal two operations must be performed: quantization and sampling. Quantization converts a continuous value signal into a discrete value signal. Sampling converts a continuous time signal into discrete time signal. It does not matter which of the two operations comes first. Although, in the real world it is difficult to do quantization and then sampling. Sampling normally precedes quantization.

1.5 Quantization Errors

One of the more fundamental limitation of Nyquist converters, is the quantization error. Quantization of an continuous value signal is a non-linear operation. We define the output y_Q as

$$y_Q = Q(y_a) = y_a + q_e \tag{1.13}$$

where y_a is the input signal, $Q(x)$ is the quantization function and q_e is the error signal due to quantization.

1.5.1 The Exact Solution

The quantization operation distorts the input signal. We can write the quantization error, q_e, as

$$q_e = y_a - y_Q \tag{1.14}$$

If the input signal, y_a, is a ramp function, the quantization error will be a sawtooth function as shown in Figure 1.1.

Using a sinusoid as the input signal, the quantization error becomes more complex. If $y_a = A\sin\omega t$ then, according to [4], the quantized signal, y_Q, can be described as

$$y_Q = \sum_{p=1}^{\infty} A_p \sin p\omega t \tag{1.15}$$

where ω is the angular frequency, t is time and p is the harmonic index. The amplitude of the individual harmonics, A_p, is defined as

$$A_p = \delta_{p1}A + \sum_{m=1}^{\infty} \frac{2}{m\pi} J_p(2m\pi A), p = odd \tag{1.16}$$

$$A_p = 0, p = even \tag{1.17}$$

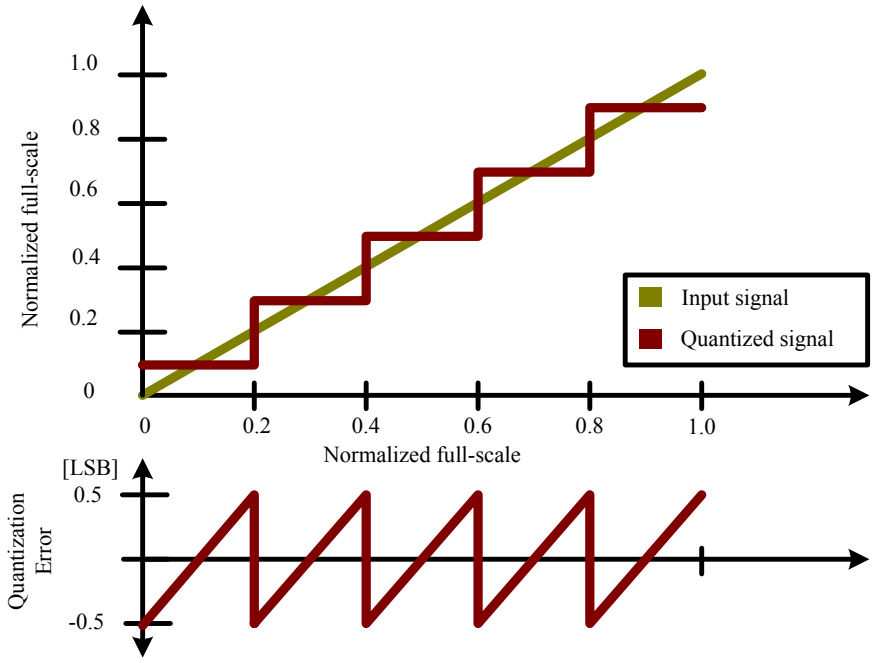

Figure 1.1: The input signal, quantized signal and the quantization error.

where

$$\delta_{p1} = 1, p = 1 \tag{1.18}$$
$$\delta_{p1} = 0, p \neq 1 \tag{1.19}$$

and $J_p(x)$ is a Bessel function of the first kind. If we approximate the amplitude of the input signal as

$$A = \frac{2^n - 1}{2} \approx 2^{n-1} \tag{1.20}$$

where n is the number of bits, we can rewrite (1.16) as

$$A_p = \delta_{p1} 2^{n-1} + \sum_{m=1}^{\infty} \frac{2}{m\pi} J_p(2m\pi 2^{n-1}), p = odd \tag{1.21}$$

Using (1.15) and (1.21), we can calculate the output signal y_Q. In Figure 1.2 a 10-bit quantizer spectrum is shown with 30.000 harmonics [2]. The spectrum shows how the amplitude of the harmonics decrease as frequency increases.

1.5.2 The Approximation

It is generally accepted that for sufficient quantization steps (enough bits) and an active input signal the quantization error, q_e, can be approximated by a white noise [5]. The quantization error varies between $-\frac{1}{2}LSB < q_e < \frac{1}{2}LSB$ and has an average power of $\overline{q_e^2} = \frac{1}{12}LSB^2$. As previously mentioned the quantization error places a fundamental limit of the resolution of a Nyquist converter with a finite number of bits. The general expression of signal to noise ratio is

$$SNR = 6.02n + 1.76dB \tag{1.22}$$

where n is the number of bits

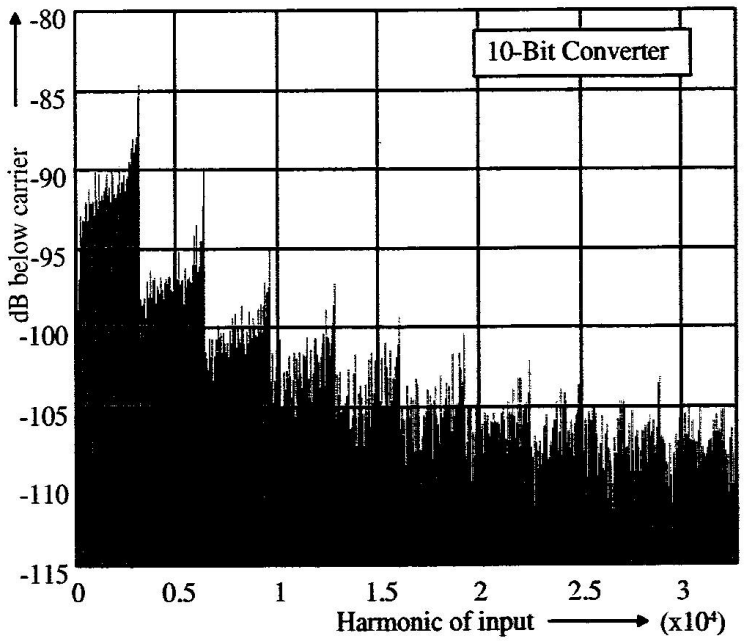

Figure 1.2: 10-bit quantizer spectrum with 30.000 components.

1.5.3 The Exact Solution Versus the Approximation

A more accurate expression for the dynamic range than (1.22) of a n-bit converter, derived from (1.15) and (1.21), is

$$S/N = \frac{2^{n-1} + \sum_{m=1}^{\infty} \frac{2}{m\pi} J_1(2m\pi 2^{n-1})}{\sqrt{\sum_{i=1}^{\infty} [\sum_{m=1}^{\infty} \frac{2}{m\pi} J_{2i+1}(2m\pi 2^{n-1})]^2}} \tag{1.23}$$

In Table 1.5.3 the S/N for 1 to 10 bits in the quantizer is shown [2]. The approximation (1.22) overestimates the signal-to-noise ratio. The overestimation is reduced with a higher number of bits.

Number of bits	Accurate S/N	Approximate S/N
1	6.31	7.78
2	13.30	13.80
3	19.52	19.82
4	25.59	25.84
5	31.65	31.86
6	37.70	37.88
7	43.76	43.90
8	49.82	49.92
9	55.87	55.94
10	61.93	61.96

Table 1.1: S/N as function of the number of bits.

1.6 Abbreviations and Measures

For a thorough definition of the different abbreviations and measures we refer to chapter 1 and 2 in [2]. This chapter summarizes some of the measures used in this report.

1.6.1 MSB and LSB

MSB is the Most Significant Bit and LSB is the Least Significant Bit. The LSB of a ADC is equal to the converter step.

1.6.2 INL

INL is the Integral Non-Linearity of a ADC. It is the deviation of the quantization steps from a straight line when linear errors (offset and gain errors) are removed.

1.6.3 DNL

DNL is the Differential Non-Linearity of a ADC. It describes the difference between two neighboring analog signals when compared to the LSB

1.6.4 SNR

SNR is the Signal-to-Noise Ratio of a system. It is usually defined as

$$SNR = 10 \log \frac{Signalpower}{NoisePower} \tag{1.24}$$

but authors have a tendency to use different definitions when they talk about SNR. For example some authors include distortion in SNR and some do not. The bottom line is that SNR is a measure of the dynamic range available in a signal. The maximum SNR of a Nyquist n-bit converter is usually defined as (1.22). SNR is sometimes written as S/N.

1.6.5 SFDR

SFDR is Spurious Free Dynamic Range. In a spectrum plot it is the difference between the power of the signal and the most powerful harmonic.

1.6.6 ENOB

ENOB is Effective Number Of Bits. If we have the measured SNR of an ADC we can use (1.22) to get effective number of bits:

$$ENOB = \frac{SNR - 1.76}{6.02} \tag{1.25}$$

It should be noted that in data sheets where SNR is given it is sometimes measured without the power of the first six harmonics. We would get a more accurate ENOB if we included distortion in the SNR. SNR with distortion is often named SNDR (Signal to Noise and Distortion Ratio) or SINAD (Signal to Noise And Distortion).

1.6.7 ERBW

ERBW is Effective Resolution BandWidth. It is defined as the bandwidth where the SNR (preferably with distortion) of the ADC stays within 3dB.

1.6.8 FOM

FOM is Figure Of Merit. A FOM is used to compare the performance of different systems, a collection of parameters are combined to get one number used for the comparison. The FOM commonly used with analog-to-digital converters, is

$$FOM = \frac{Power}{2^{ENOB} f_s} \tag{1.26}$$

A more accurate FOM would entail using $2ERBW$ instead of f_s, since the given sampling frequency in data sheets is not necessarily equal to $2ERBW$ [2]. FOM reduces with a factor of around 10 in 10 years [2]. The figure of merit reflects the three parameters that conflict in analog design: power dissipation, speed and accuracy.

1.7 Aliasing

Aliasing is an important phenomena that should be mentioned in any document about ADCs. Aliasing is the folding of input signal frequencies higher than the Nyquist frequency $f_s/2$ into the base-band. Here f_s is the sampling frequency of the ADC. Aliasing is mostly an unwanted phenomena. To avoid aliasing an anti-alias filter is used. This can be a pure analog filter in front of the ADC, or a combination of analog filtering and digital post filtering. Note that digital post filtering, in other words filtering after sampling, requires a certain oversampling of the base-band. If the base-band is limited to f_b the sampling frequency might be at $8f_b$ [2]. Design of alias filters will not be discussed in this report.

Chapter 2

Design Challenges in High-Speed ADCs

Two of the main challenges in design of high-speed ADCs are timing and distortion [2].

2.1 Timing Errors

Timing errors manifest through different sources. We will describe the most prominent in this section. The timing errors may lead to both noise and/or distortion, but we treat them under timing and not distortion, because of the source of the error.

2.1.1 Sampling Clock Jitter

Sampling is usually controlled by a clock signal at a certain sampling frequency, f_s. According to the Nyquist theorem, signal frequencies at, or below, $\frac{f_s}{2}$ can be accurately reproduced from the sampled data. How accurate a signal can be sampled depends on the sampling time uncertainty. One form of sampling time uncertainty is called jitter or clock phase noise. Jitter is a random fluctuation of the sampling instance. The source of jitter is often thermal noise in clock buffer-, amplifier- or generator-circuits [2]. To illustrate the effects of jitter we created a simple MATLAB example. The MATLAB code can be found in Appendix A.1. The results can be seen in Figure 2.1 and 2.2. In Figure 2.1 the sampled spectrum with and without jitter is shown. Note that the signal without jitter has finite resolution because we have added noise to emulate quantization noise. The jitter is simulated as a random fluctuation of the sampling instant, as can be seen from the code. From Figure 2.1 we can see that noise power is increased when jitter is added. In Figure 2.2, signals are shown in time domain. We can see that the signal with jitter samples an incorrect value from the input signal.

It is possible to derive equations for the maximum jitter that can be tolerated in an ADC. As we can see from Figure (2.2), at a specific time t we sample a value A without jitter, and with jitter we sample a value $A + \Delta A$. For the jitter not to have an adverse effect on the resolution of the converter, the factor ΔA must be less than the quantization step of the converter. An expression for the maximum jitter, Δt_{max}, can be written as (2.1) [2], where n is the number of bits and f_{in} is the maximum input frequency. For a 16-bit converter system with an input frequency of 20 kHz the maximum jitter must be less than $\frac{1}{4} nsec$ [2].

$$\Delta t_{max} = \frac{2^{-n}}{\pi f_{in}} \tag{2.1}$$

Since the amount of jitter depends on the input signal frequency, as shown in (2.1), it is imperative that clock amplifiers/buffers in high-speed ADCs are designed to have sufficiently low jitter.

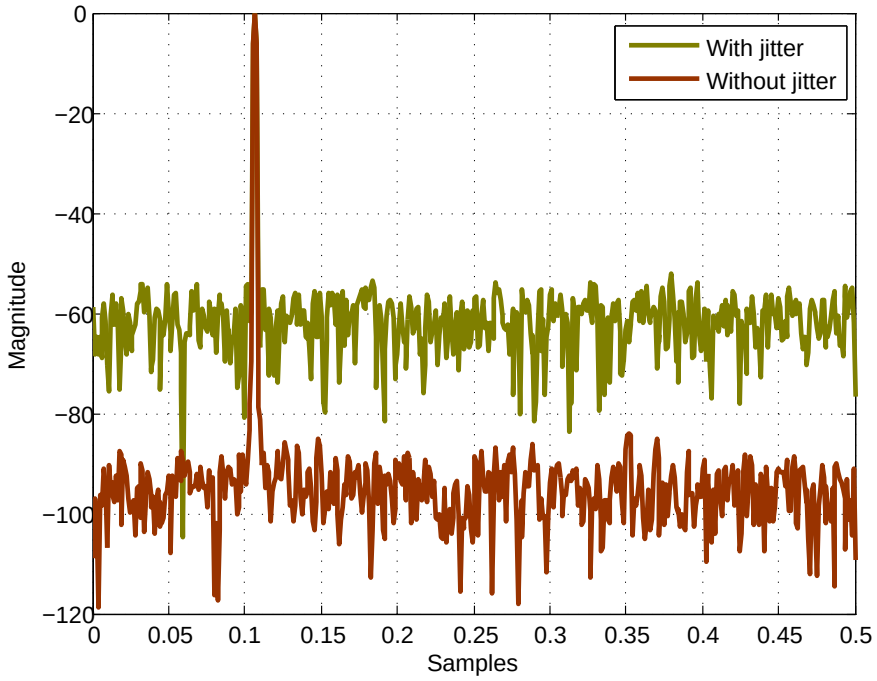

Figure 2.1: Spectrum with and without jitter in sampling of a sinusoid

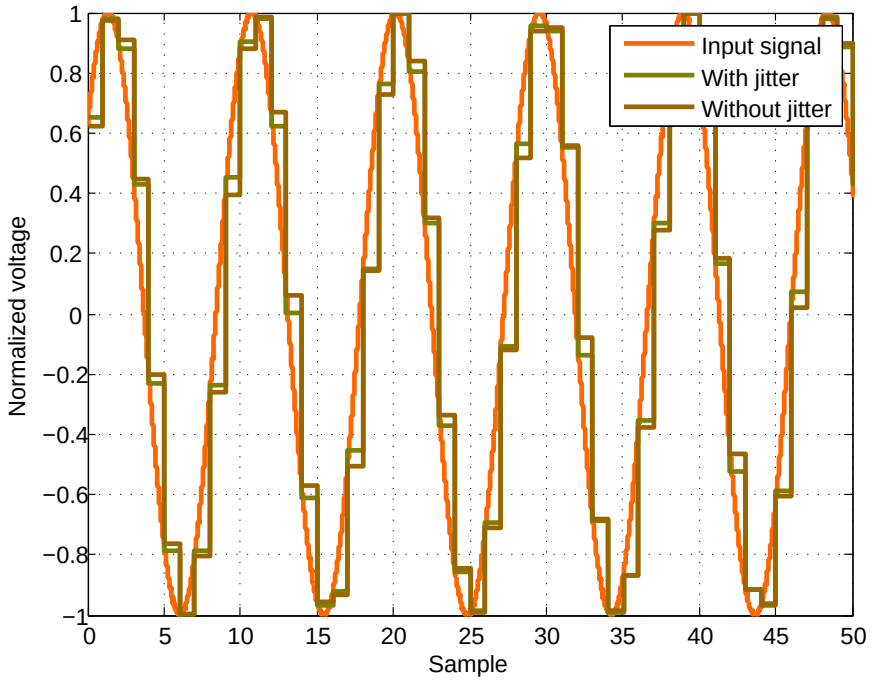

Figure 2.2: Time domain plot with and without jitter in sampling of a sinusoid

If we know how much jitter we have in a circuit, it is useful to have an expression for the reduction in ENOB. The reduction in the ENOB due to jitter can be written as [2]

$$ENOB_{reduction} = \frac{\log(1 + [2^n \pi \sqrt{6} \frac{T_{jit}}{T_{sig}}]^2)}{2 \log 2} \tag{2.2}$$

where n is the number of bits, T_{jit} is the root mean square of sampling clock phase noise and T_{sig} is the period of input signal. In [2] they have plotted (2.2) as a function of n and $\frac{T_{jit}}{T_{sig}}$, the result is duplicated in Figure 2.3.

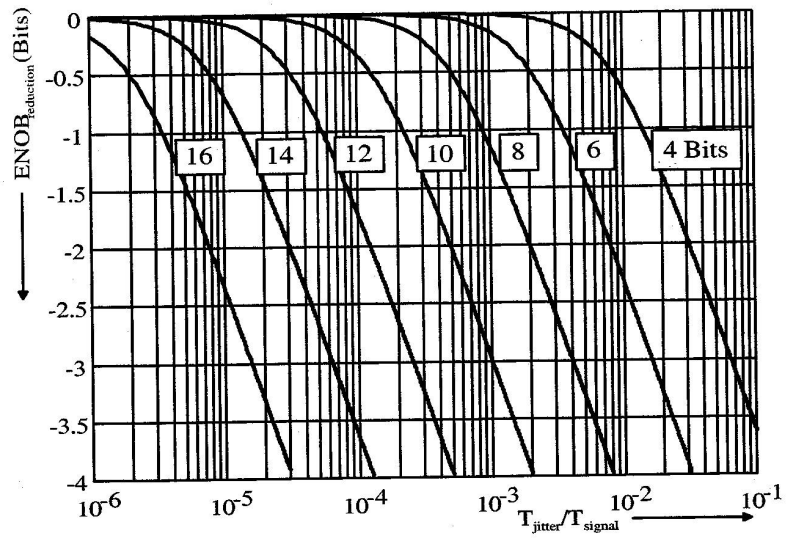

Figure 2.3: Reduction in ENOB versus $\frac{T_{jit}}{T_{sig}}$

2.1.2 Limited Rise and/or Fall Time of Sampling Clock

On-chip clock signals will have finite rise and fall times due to the finite resistance and non-zero capacitances of clock amplifiers/buffers, and interconnect. White noise in clock amplifiers/buffers might worsen the jitter if rise and/or fall times are to high. Clock amplifiers should be designed with low rise and fall times.

2.1.3 Skew of Clock or Input Signal

Assume that our converter looks like the converter in Figure 2.4. In this example distributed sampling is used, in other words the input signal is sampled at different locations on the chip, but preferably at the same time. Due to finite speed of signal propagation on-chip, which is between half and one-third of the speed of light [2], the input signal and sampling clock arrive at different locations at different times. A typical signal may travel around $100\mu m$ to $200\mu m$ in $1ps$, so that a converter like the one shown, may experience a difference of several pico seconds in sampling instant from one end of the converter to another. This time difference causes an error which results in non-linear distortion [2]. Skew of clock signal or analog signals is one of the major contributors to non-linearity, and thus distortion, in high-speed ADCs [2]

To avoid problems with clock or signal skew in high-speed ADCs it is common to implement tree like signal routing structure, like the one shown in Figure 2.5. With such routing the delays from $clk \Rightarrow clk_i$ are all equal.

2.1.4 Signal-Dependent Delay

If a signal-dependent delay is introduced, it can lead to non-linear distortion. A circuit with amplitude-limiting followed by bandwidth-limiting introduces a delay that is slope dependent, which can lead to distortion [2]. Circuits with amplitude-limiting and bandwidth-limiting are

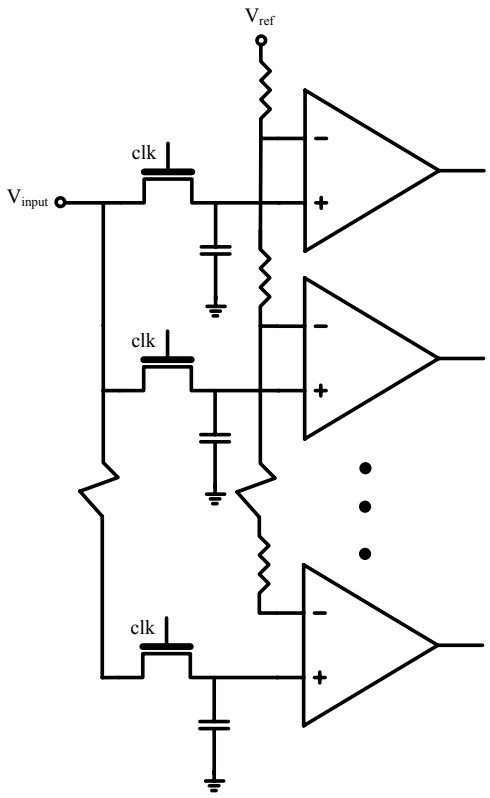

Figure 2.4: Distributed sampling in an ADC

almost always found in the first stages of a high-speed ADC in form of a sample-and-hold or input amplifiers.

2.1.5 Sampling Comparators Aperture Time

The aperture time is defined as the time between the command to sample occurs and the actual sample is taken. A large aperture time leads to high-frequency errors which cause an averaging in the time domain, leading to third-order distortion [2]. The architecture of the comparator may cause a large aperture time. Large rise or fall times in sampling clock, with respect too the clock period, will cause a large aperture time. Increasing the small signal bandwidth of the comparator and reducing the rise and/or fall times of sampling clock may reduce the effect. This is another of the major contributors to distortion in high-speed ADCs [2].

2.2 Distortion

Distortion of the quantized signal is a phenomena which have many sources. Some sources of distortion have their roots in timing errors, as explained in previous section. This section discussed some of the other sources of distortion in high-speed ADCs.

2.2.1 Distortion in Input Buffers and/or Amplifiers

A non-linear transfer function of input buffers or amplifiers will lead to distortion, as explained in Section 1.2. Most integrated analog-to-digital converters manufactured today use differential signaling. As previously explained, differential circuit suppress even order distortion; it is often third order distortion in input buffers/amplifiers that is the dominating source of non-linearities [2].

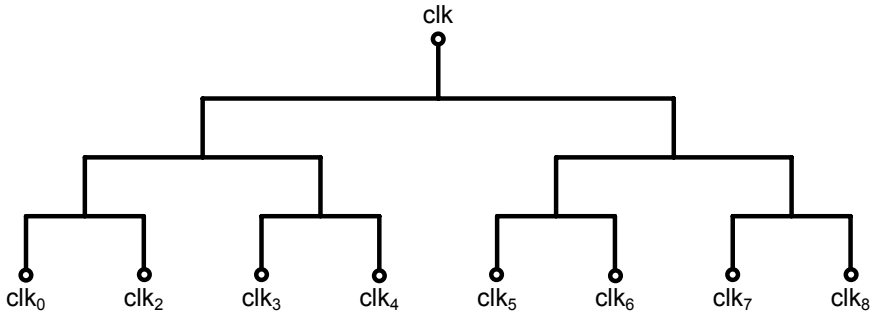

Figure 2.5: Tree signal routing with equal delay from $clk \Rightarrow clk_i$

To reduce distortion in amplifiers and/or buffers one can reduce internal signal swing, or avoid stacking of transistors in output stages. In a multistage opamp it is usually the output stage that dominates the non-linearities at lower frequencies; at higher frequencies the input stage dominates [6].

2.2.2 Offset in Input-Amplifiers/Comparators

Mismatch of devices is one of the more important sources of non-linearity in modern CMOS technologies [2]. Mismatch of transistor pairs in input-amplifiers/comparators leads to an offset, in other words the output currents of a differential pair are not equal when the differential input voltage is zero. Although offset is a linear error when referred to the individual input-amplifiers/comparators it will introduce non-linearities in ADCs. Assuming one comparator per quantization step, the offset of the individual quantization steps will vary, thereby degrading the INL of the ADC. In CMOS technology the offset of a differential pair can be written as

$$V_{offset} = \frac{a_{vth}}{\sqrt{WL}} \tag{2.3}$$

where a_{vth} is a process parameter, W is width of the transistor and L is the length of the transistor [2]. An increase of the gate area of the transistors will reduce the offset in a differential pair. Note that the reduction in offset voltage is proportional to $1/\sqrt{WL}$; the input capacitance of a MOSFET is proportional to WL. Thus input capacitance of the differential pair will increase faster than the offset voltage is reduced. There are techniques that can reduce offset without increasing input capacitance. These techniques will be covered in Chapter 3.

2.2.3 Changes in Reference Voltage(s)

The reference voltage or voltages of an ADC are essential. It can be said that it is next to impossible to make an ADC without a reference, which is usually true since it is difficult to quantify anything without having something to quantify it with respect to. Reference voltages can be defined through a resistive divider with unit resistors, for example one for each quantization step. Matching accuracy of unit resistors will affect the INL of the ADC in much the same way as offset in comparators. It is possible to estimate the required matching for a certain INL. Section 2.6.6 in [2] shows simulations of mismatch in unit resistors versus the resulting INL. Results are summarized in Table 2.2.3. We see that for a 14-bit ADC we need a matching accuracy of 0.4%

Number of bits	$INL \approx \frac{1}{2}LSB$	$INL \approx 1LSB$
8-bit	3.1%	6.2%
10-bit	1.6%	3.1%
12-bit	0.9%	1.7%
14-bit	0.4%	0.85%

Table 2.1: Resistor matching versus converter accuracy data.

to get an INL of $\frac{1}{2}LSB$. This source of distortion is usually not the dominating source since it is possible to achieve matching better than 0.1% for resistors and capacitors in modern processing technology [7]; of course this depends on the resolution of the converter. Another error that can introduce non-linearity through reference voltages is kickback from comparators [2]. During sampling of input signal a kickback from a comparator can temporarily change the reference voltage, thereby introducing an offset in the reference voltage.

Chapter 3

Architectures for High-Speed ADCs

In this chapter we will present some of the more common architectures for high-speed analog-to-digital converters.

3.1 Full-Flash Converters

In a flash ADC $2^N - 1$ comparators are used to extract N-bits. It is one of the more direct approaches to analog-to-digital conversion. The most common flash ADC is possibly the thermometer encoded flash. There have also been publications with other types of encoding, like gray encoding or circular encoding. These will not be discussed here and we refer to chapter 3.9 and 3.10 in [2] for further information.

In Figure 3.1, a 2-bit thermometer encoded flash converter is shown. The difference between the input signal voltage and a reference voltage is amplified using input amplifiers resulting in the outputs $A_0 - A_2$. The reference voltages are generated by voltage division in the string of unit resistors. After amplification $2^N - 1 = 3$ comparators are used to detect the zero crossing, which occurs when the input signal voltage is equal to a reference voltage[1]. Sampling is performed by clocking the comparators using the clock signal Clk. A decoder takes the thermometer encoded output, $T_1 - T_3$, from the comparators and converts it into the two binary outputs MSB and LSB.

For example, if the input signal voltage is larger than the reference voltage V_{ref1} but smaller than V_{ref2}, the amplifier outputs A_0 and A_1 will be larger than zero. Output A_2 will be less than zero. The output of the comparators would be $T_2 = 0, T_1 = 1, T_0 = 1$, and we get the binary code 10 from the decoder.

With modern processing technology, it is possible to get around 8-bit from a full-flash converter; anything more is impractical [2].

3.1.1 Comparators

Comparators in a full-flash ADCs are usually latched comparators. For implementation and architectures we refer to Chapter 7 in [1]. Note that when flash converters are described in the literature the input amplifier is often a part of the comparator. Therefore, when they describe problems with offset in comparators, it is often the offset in the input amplifier of the comparator they talk about. With high gain input amplifiers the offset of the latched comparator is less important. The offset error in the latched comparator is divided by the gain of the input amplifier when referred to the input V_{in}.

[1] For the moment we will ignore offset in the amplifiers and comparators

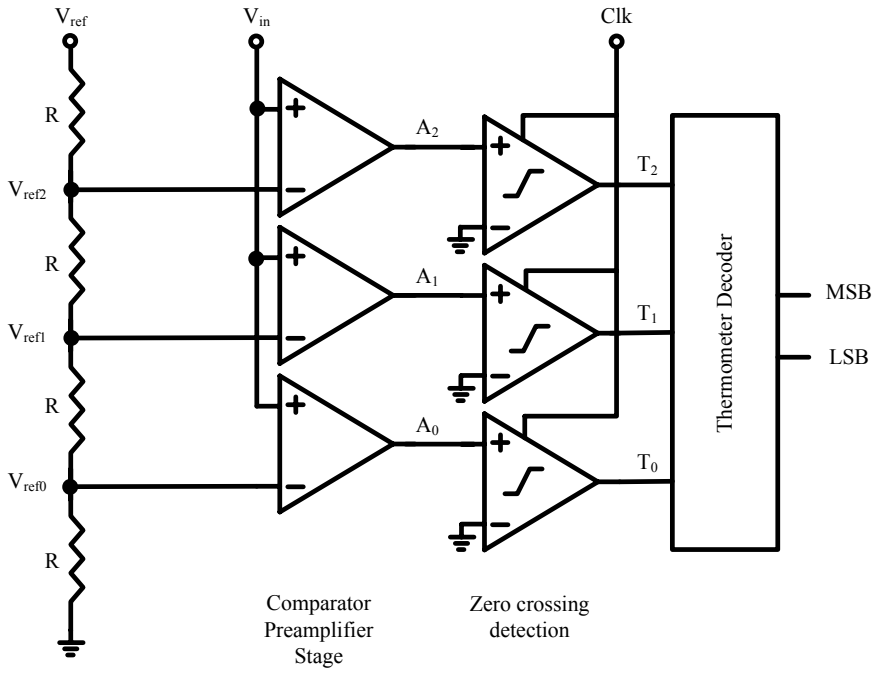

Figure 3.1: 2-bit full-flash converter

3.1.2 Input Amplifier

An example of an input amplifier is shown in Figure 3.2 a). It is a differential pair with resistive load and a current source that biases the transistors. The differential output voltage is ideally equal to

$$V_{out} = g_m R[V_{in} - V_{refi}] \tag{3.1}$$

where g_m is the transconductance of the transistors. However, due to mismatch between transistors the output exhibit an offset when the differential input is zero, $V_{in} = V_{refi}$, as seen in Figure (3.2) b). To have low offset in the amplifier the input transistors must have a large gate area, as previously explained. With a large gate area follows a large input capacitance. As all the input amplifiers are connected in parallel the accumulated capacitance load on the input can be quite large. A large input capacitance is a disadvantage in high speed circuits because input buffers will have high power dissipation [2].

The linearity of the input amplifiers is critical in a high-speed ADC, as previously explained. The input capacitance of the input amplifiers is often dominated by gate source capacitance, C_{gs}, of the transistor. This capacitance can be highly non-linear, especially if input transistors are biased in moderate inversion. Care must be taken by a designer to ensure that the non-linearities of the input capacitance are low enough.

3.1.3 Error Correction

Thermometer encoded ADCs usually implement some form of bubble error correction. A bubble error is a lone zero amidst a series of ones, for example for a 3-bit converter 0011011. For further reading about bubble correction we refer to chapter 3.18 in [2]

3.1.4 Interpolation

Interpolation can be used to reduce the number of input amplifiers in a full-flash ADC. An example of interpolation in a 3-bit full-flash is shown in Figure 3.3. Here a string of interpolating resistors

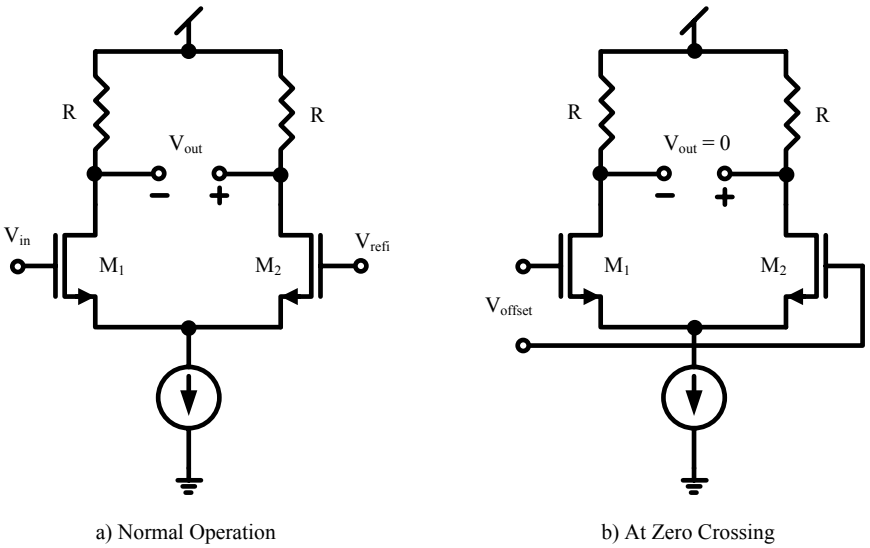

Figure 3.2: Comparator input amplifier

are inserted between the outputs of the input amplifiers. To avoid interaction between amplifiers one should insert some form of buffer stage at the amplifier output. An example of such buffering is shown in Figure 3.4.

Assume that the input signal lies between two reference levels, and that both amplifiers on either side are in linear operation. As the outputs will have a finite slope with respect to the input in linear operation (finite gain), and that the zero crossing occurs at different values of V_{in} for the two amplifiers, it is possible to achieve accurate interpolation of the zero crossing as shown in Figure 3.5. Let the reference levels of amplifier 0 and amplifier 1 be at V_{ref0} and V_{ref1} respectively. At a certain input voltage,

$$V_{in} = V_{ref0} + \frac{V_{ref1} - V_{ref0}}{2} \tag{3.2}$$

the amplifier outputs A_0 and A_1 would be as shown in Figure (3.5) where $-A_0 = A_1$. If the interpolation resistors are of equal size the interpolated voltage would be

$$V_{interpolated} = A_1 + \frac{A_0 - A_1}{2} = -A_0 + \frac{2A_0}{2} = 0 \tag{3.3}$$

Thus the zero crossing is correctly interpolated. The accuracy of interpolation depends on the linearity of the amplifiers and the mismatch between resistors. In addition to reduce the number of input amplifier interpolation improves the DNL of an ADC [2].

The 3-bit flash-ADC in Figure 3.3 used single interpolation where the number of input amplifiers is reduced by a factor of two. There are other types of interpolation possible.

Multiple Interpolation: The resistor string between amplifier outputs can consist of multiple resistors; we need less input amplifiers if multiple interpolation is used.

Active Interpolation: Instead of using resistor strings to interpolate between amplifier outputs, one can use active circuits. An example of active interpolation is shown in Figure 3.6, A_0 and A_1 is the output from the input amplifiers. This interpolation architecture performs well, even at high frequencies, if offset voltages and mismatch between devices are small compared to required interpolation accuracy. 3.6

Capacitive interpolation: In a discrete time systems it is possible to use capacitive interpolation. Example of capacitive interpolation is shown in Section 3.5.5 in [2].

It should be noted that interpolation is not exclusive to full-flash converters, it has also been used successfully in folding converters [8].

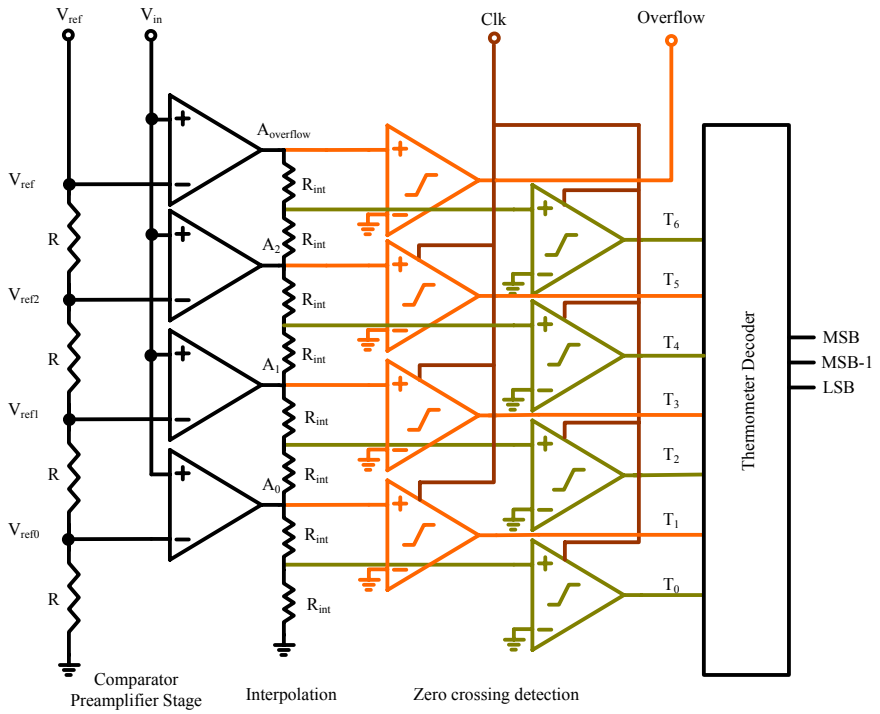

Figure 3.3: Example of Interpolation in a 3-bit flash converter.

3.1.5 Averaging

When implementing interpolation we inserted a buffer at the output of the input amplifiers to avoid interaction between the amplifiers. It turns out that allowing interaction can help reduce offset of the input amplifiers [2]. The technique is called averaging. Averaging can be performed by placing a resistor between the unbuffered outputs of the input amplifiers. The improvement in offset voltage is proportional to $\sqrt{N_{active}}$, where N_{active} is the number of linear active amplifiers. An example of an averaging circuit is shown in Figure 3.7. It is assumed that the amplifiers to the left and right of the zero crossing amplifier (M3/M4) is operating in linear region. Although averaging improves INL, it has non-linearities at the end of the averaging chain. The first and last amplifier only has active amplifiers to the left or right, not both. This reduces the usable range to around 75%. Some solutions have been proposed to deal with this non-linearity [2].

Adding extra amplifiers: By adding two extra amplifiers, one on each side of the chain, and scaling the last resistor one can achieve a usable range of 95%

Moebius band averaging: The resistors are connected in a Moebious band, thus the resistor chain has no end. This technique also achieves close to 95% usable range.

Active averaging system: Active averaging uses active transistors. An output signal is obtained as the sum of three neighboring differential pairs.

3.1.6 Full-Flash ADC Example

An example of a 6-bit flash ADC is shown in Figure 3.8 If we start from the left in the plot we have the reference generation with 15 taps. The input is fed to 15 amplifiers with averaging. Afterwards, a distributed track and hold is used to sample the signal. After sampling they have two stages with interpolation/averaging to achieve 63 zero crossings. These zero crossings are detected by the 63 clocked comparators. A thermometer code to binary code is performed at the

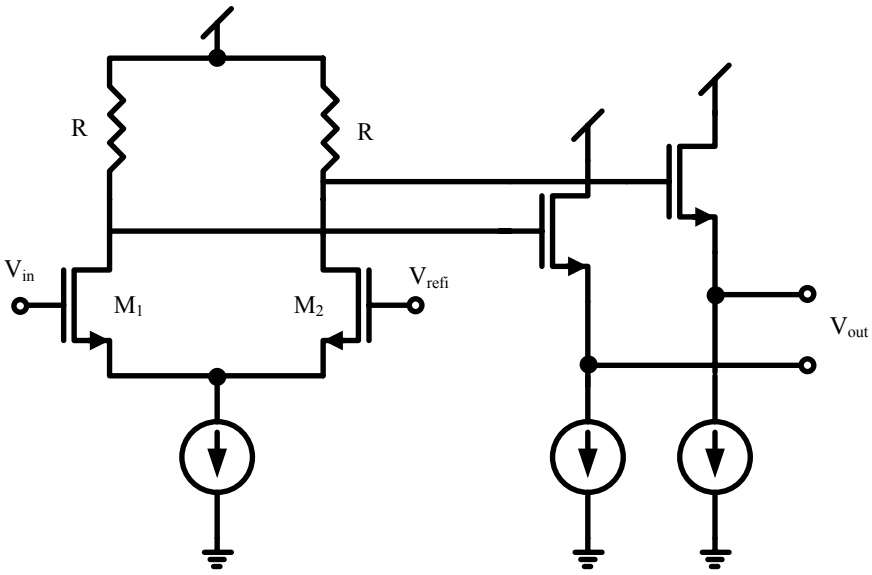

Figure 3.4: Input amplifier with output buffer.

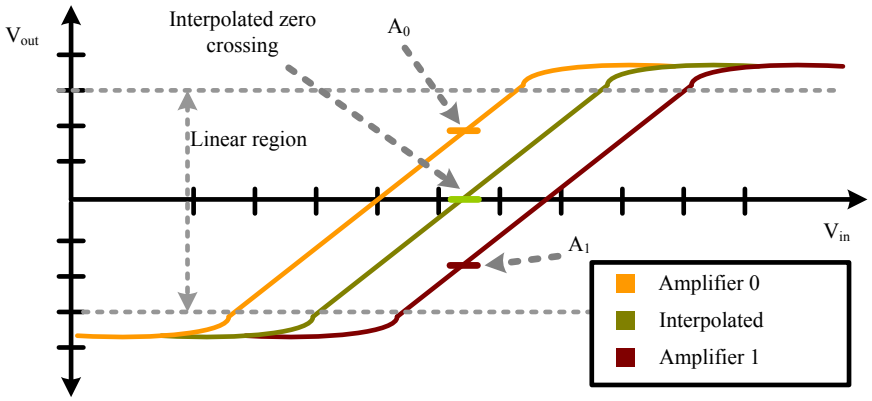

Figure 3.5: Output of nearby amplifiers and interpolated value

end. A performance summary of the converter is shown in Table 3.1.6. This converter has a FOM of 6.87 pJ per conversion step. It was published in 2001 [9].

3.1.7 Evolution Of The Flash Converter

We do not know which ADC architecture was the first, but when you look at them it looks like the flash ADC has evolved into several different species over the years. We have the thermometer encoded, gray encoded and circular encoded flash ADC. Often the difference is a small improvement within a certain area, for example gray encoded flash ADC uses analog preprocessing to reduce the number of comparators. Discrete time implementations, with a sample and hold in front of the converter, have been used to alleviate problems with clock skew.

To deal with the component count of full-flash ADCs the two-step flash ADC was developed. It has a drastically reduced component count compared to the full-flash ADC. The two step architecture is shown in Figure 3.9. The idea behind the two step flash is to do a coarse quantization first, subtract the quantized value from the input with the help of a DAC and then do quantization of the residue (fine quantization). With this architecture it is possible to achieve 8-bit resolution with only 40 comparators [2]. However, the accuracy of the DAC needs to be 8-bit for the ADC to

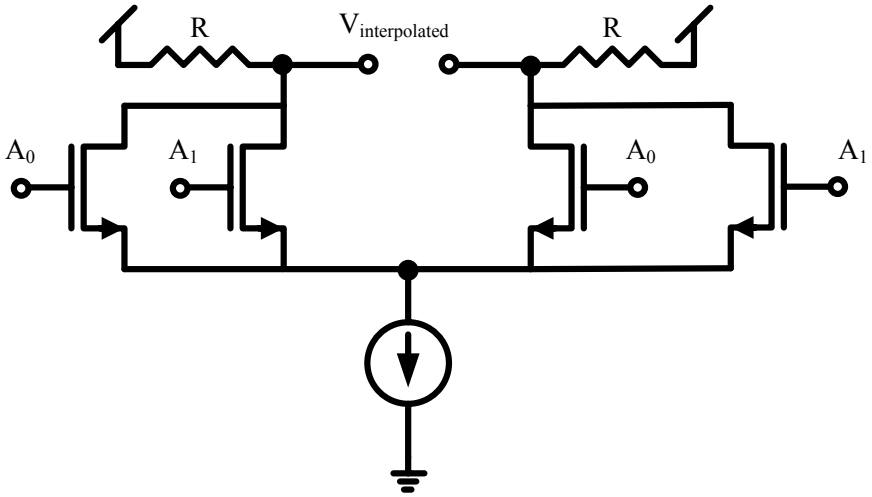

Figure 3.6: Active interpolation. Transistors are scaled by a factor of half compared to transistors in input amplifier 3.2

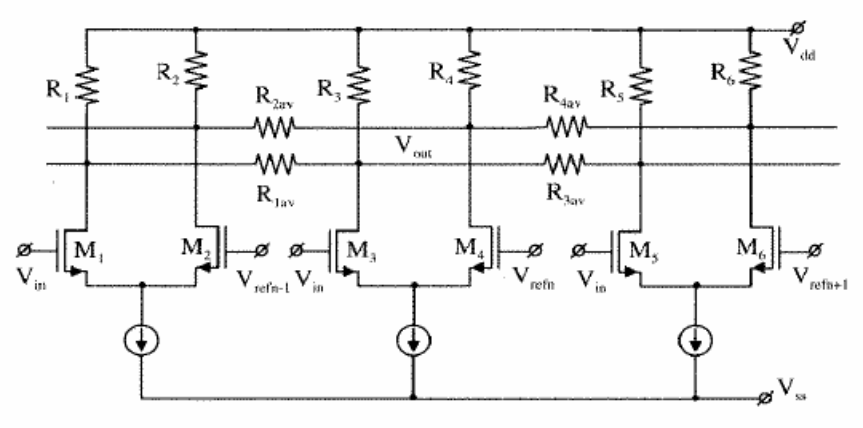

Figure 3.7: Resistive averaging scheme

have sufficient linearity for 8-bit operation. It is not uncommon to have a gain stage between the coarse and fine quantizations. The two-step flash ADC has evolved further into a prevalent species, the pipelined ADC. The pipelined ADC has become prevalent in the medium/high resolution and high speed segment. The pipelined ADC will be discussed in a separate section. Akin to the two-step flash is the sub-ranging converters. These have no gain stage between the coarse and fine stages. In the next section we will discuss another relative of two-step flash, the folding ADC.

3.2 Folding ADC

Folding ADC's combine the component savings of a two-step ADC with the digital sampling of a full-flash converter. A folding converter system can be seen in Figure 3.10. The folding ADC consists of a coarse quantization and a fine quantization, like the two-step converter. One of the differences is that the folding ADC does not need a sample-and-hold in front of the converter. The coarse quantizer determines the MSBs, the MSBs determine the number of times the input signal is folded. The output of the folding circuit is converted by the fine flash converter to get the LSB's.

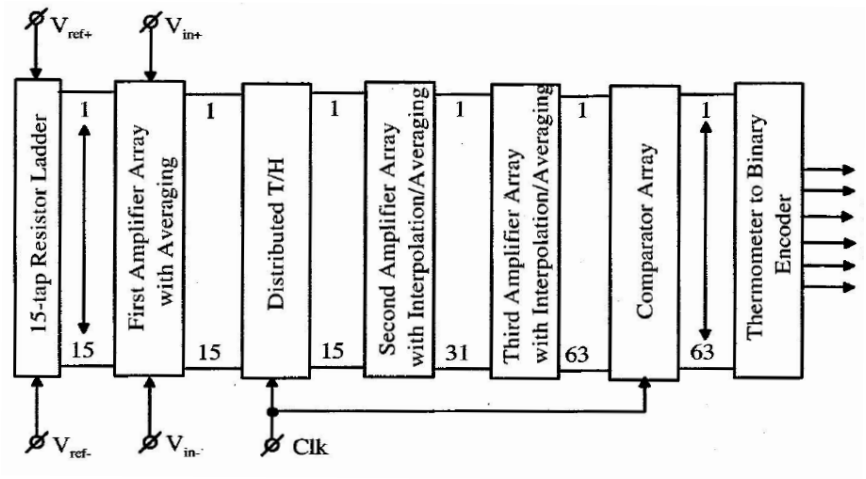

Figure 3.8: 6-bit converter architecture

Parameter	Value
Technology	0.35 μ Standard CMOS Single Poly, Five Metal
Resolution	6-bit
Effective Number of Bits	5.6 bit
Resolution Bandwidth	450 MHz
Maximum Clock Frequency	1.1GHz
Supply Voltage	3.3V
Power Dissipation	300mW

Table 3.1: Performance summary of 6-bit flash converter

To see how folding works, take a look at Figure 3.11. Here the input signal is a ramp from 0 to 1, the axis are normalized full-scale range. A 2 bit coarse quantization finds the MSBs. In a two-step flash ADC we would subtract the MSBs from the input signal to get a residue, the residue is shown in the Figure by the grey arrow. In a folding ADC the input signal is folded as shown by the folding signal in Figure 3.11, thereby limiting the input signal to 1 LSB of the coarse quantization. A fine quantizer can be used to convert the folded signal.

3.2.1 The Folding Block

An example of a folding block can be seen in Figure (3.12). The transistors M_2, M_4, M_5 are forward biased diodes. These will turn on, one at a time, when the input signal, I_{in}, increases. The reference currents I_0 are divided equally between the same size resistors R_1 and R_2 via the transistors M_1, M_3, M_5, M_7, thus the differential output voltage V_{out} is biased at zero. The circuit produces a folding signal similar to the one in (3.11). These triangular folding blocks suffer from high frequency problems. Band-limiting in the folding blocks will incur rounding of the folding signal [2]. As such the folding signal will not be shaped like a triangle, but rounded at the extremities. This will cause distortion of the fine quantized signal. One of the solutions to this problem is to use two parallel folding blocks where the folding thresholds are offset by a certain amount, as shown in Figure (3.13) [2]. Here the fine converter is used only during linear operation of the folding blocks. When folding block A deviates from a linear transfer function a switch is made, and the output signal from folding block B is used.

Another technique is to implement one folding block for each LSB in the residue with a small offset between the folding blocks, and use a single bit quantizer to detect a zero transitions. An example of a folding block that can be used in such a system can be seen in Figure (3.14). At

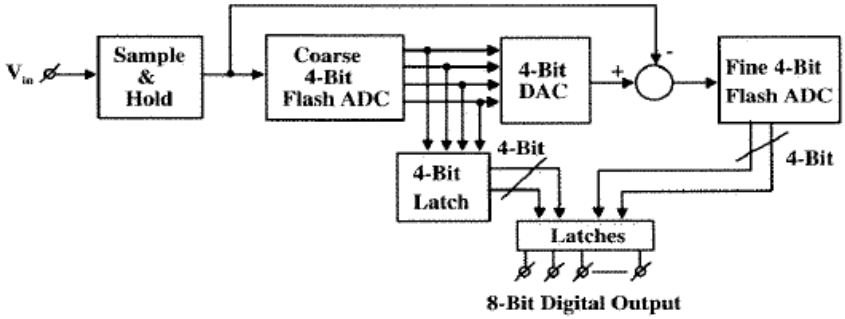

Figure 3.9: The Two-Step Flash Architecture

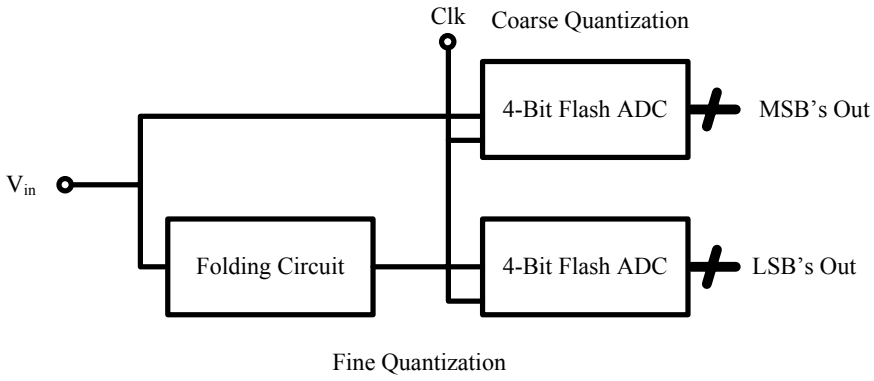

Figure 3.10: Folding converter system

zero input voltage $V_{in} = 0$ the differential output voltage V_{out} is zero. When the input voltage, V_{in}, is equal to V_{ref1}, the currents through M_1 and M_2 are equal, increasing the output voltage to roughly half the available swing. As the input voltage increases further the output voltage will probably saturate at maximum swing. When M_3 starts to draw current instead of M_4, the voltage will start to drop again and at $V_{in} = V_{ref2}$ we reach another half-way point. The circuit continues to fold the input signal depending on how many differential pairs there are in the folding block. If we have a 10-bit folding ADC with 3-bit coarse quantizer, we would need up to 128 folding blocks. Such a system would have a large input capacitance. To reduce the number of folding blocks we can use interpolation in much the same way as in the full-flash ADC.

3.2.2 Folding ADC Example

The block diagram of a 8-bit, 70MS/s, 110mW folding and interpolating converter is shown in Figure (3.15) [8]. A coarse quantization of 3-bits was used, thereby a folding rate of eight was used in the four folding amplifiers. Interpolation was used to bring the total up to 32 folded signals; 32 comparators detect the zero transitions and their outputs were combined with the coarse MSBs to form the 8-bit output. The converter achieves a FOM of around 56.7 pJ per conversion step. It was published in 1995.

3.2.3 Folding ADC Summary

To improve high frequency performance a sample-and-hold (S/H) can be inserted in front of the converter. It is also possible to do distributed track-and-hold (T/H) with analog pre-processing to reduce the required linear region, settling requirements and power dissipation compared to a single S/H [2]. The disadvantage of the distributed T/H is the need for multiple clock signals.

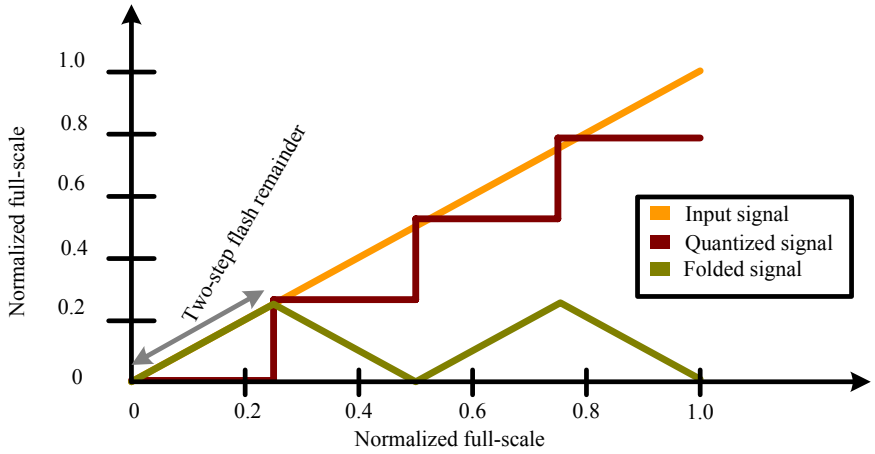

Figure 3.11: Signals in a folding converter

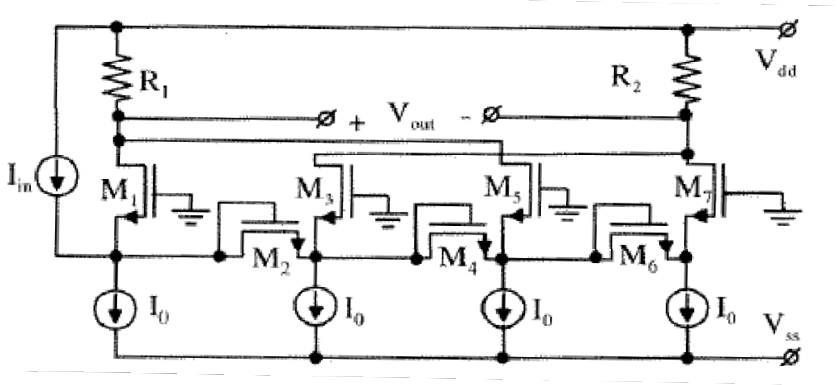

Figure 3.12: Example of a triangular folding block

It should be noted that the MSBs, the output from the coarse quantizer, is often determined using signals from the folding blocks of a converter.

One of the drawbacks of folding ADCs are the high frequencies in the folding circuit and loss of resolution due to limited bandwidth in folding circuit.

To increase the folding rate (number of times a signal is folded) but not the input capacitance, folding blocks can be connected in cascade. For more information see section 3.14.16 in [2].

3.3 Pipelined ADC

A block diagram of a pipelined ADC can be seen in Figure (3.16). The pipelined ADC consists of multiple stages, normally preceded by a sample-and-hold (S/H) circuit. In each stage p-bits are determined. A p-bit ADC quantizes the stage input signal, the quantized signal is subtracted from the stage input signal using a p-bit DAC. The residue after subtraction is amplified so the input swing of the pipelined stage is equal to the output swing. The last stage in a pipelined ADC is usually a flash ADC. An over-range is normally implemented in pipelined ADCs. In other words the sum of the p-bits bits from each stage is larger than the resolution of the overall ADC. This over-range is used to reduce the required accuracy of the ADC within the pipelined stage. In the first stage the DAC and amplifier (Gain) need to have full accuracy. In the subsequent stages the required accuracy is lower due to the accumulated gain. Therefore stages 2 through stage n are usually scaled in order to reduce the power dissipation of these stages. The number of bits

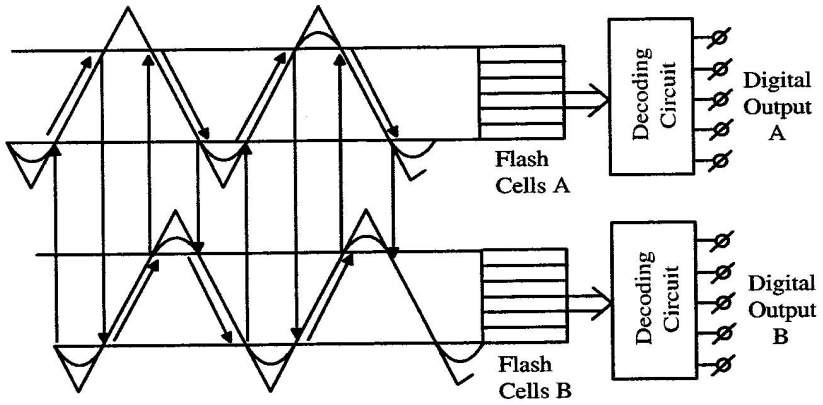

Figure 3.13: Double folding output signal

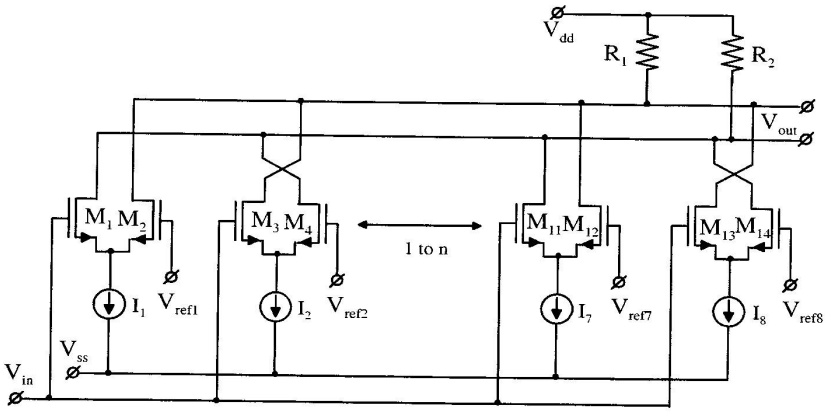

Figure 3.14: Example of zero crossing folding

selected for each stage, p, depends on the overall resolution of the ADC and what is possible to implement within the restrictions of the processing technology.

3.3.1 Speed Of Pipelined ADC

By pipelining stages, the speed of the converter can be equal to the maximum speed of each stage. Stages in a pipelined ADC normally have two phases: sampling and multiplication. Stages are normally clocked with opposite phases, in short stage 1 multiplies while stage 2 samples. If the clock period of the overall converter is T_s, each stage has $T_s/2$ for each phase. Sampling normally occurs at the end of a phase, thus stage 1 has $T_s/2$ to settle before stage 2 samples its output signal. A new sample is available at the output of the pipelined ADC at the end of each clock period. Although the pipelined ADC has a large throughput, the latency[2] depends on the number of stages. This excludes the pipelined ADC from some applications where latency is key, for example as a quantizer in a conventional $\Sigma\Delta$ ADC.

[2]Latency is the time it takes from the analog input signal is sampled to the digital word is available at the output of the ADC

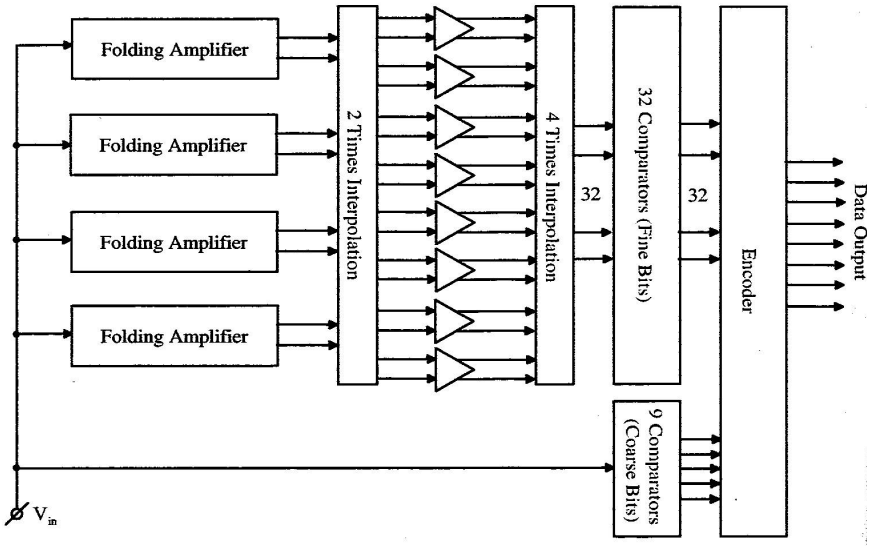

Figure 3.15: Block diagram of a 8-bit folding and interpolating converter

3.3.2 The Pipelined Stage

A common number for the bits in each stage is 1.5-bits. With 1.5-bits the stage ADC is often implemented as a flash ADC with two comparators. A common implementation of a 1.5-bit stage is shown in Figure (3.17) [2]. The two comparators have thresholds at $\pm V_r/4$, where V_r is the reference voltage. During sampling the comparators quantize the input signal. At the same time the input signal is sampled onto capacitors C_f and C_s with switches S_1, S_2 and S_3. Switch S_1 is often turned of before switches S_2 and S_3 to make the charge injection from the switches independent of the input signal. During the multiplication phase the quantized input signal is used to decide which of the voltages $+V_r$, 0 or $-V_r$ should be connected to the capacitor C_s, this is effectively the DAC. The capacitor C_f is connected to the opamp output during the multiplication phase. The opamp proceed to settle, and forces a virtual ground at the negative input of the opamp. When the opamp has settled the output signal is ready to be sampled by the next stage. The output of this stage can be written as

$$v_{residue} = \begin{cases} (1+\frac{C_s}{C_f})V_{input} - V_r, & V_{input} > \frac{V_r}{4} \quad (3.4) \\ (1+\frac{C_s}{C_f})V_{input}, & -\frac{V_r}{4} < V_{input} < \frac{V_r}{4} \quad (3.5) \\ (1+\frac{C_s}{C_f})V_{input} + V_r, & V_{input} < -\frac{V_r}{4} \quad (3.6) \end{cases}$$

The output codes from the stage ADC are usually

$$p_{out} = \begin{cases} 10, & V_{input} > \frac{V_r}{4} \quad (3.7) \\ 01, & -\frac{V_r}{4} < V_{input} < \frac{V_r}{4} \quad (3.8) \\ 00, & V_{input} < -\frac{V_r}{4} \quad (3.9) \end{cases}$$

3.3.3 Error Correction In Pipelined ADCs

Assume that one of the comparators in the 1.5-bit pipelined stage has an offset that makes the threshold larger than $V_r/4$, for example $1.5V_r/4$. If the input value of stage 1 is $1.2V_r/4$ the flash ADC will incorrectly decide that the quantized word should be “01” and not “10”, which would

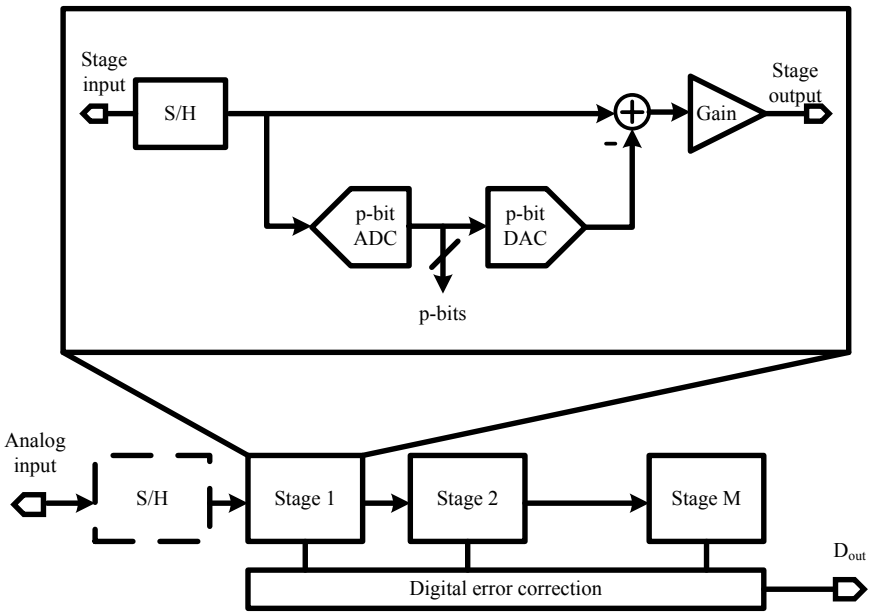

Figure 3.16: Block diagram of a pipelined ADC

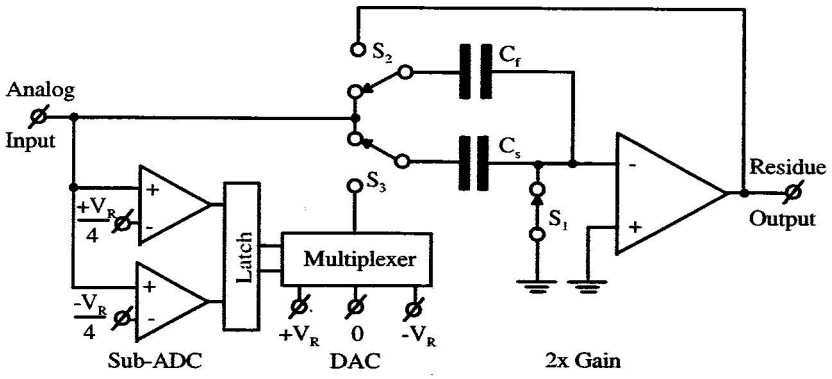

Figure 3.17: Implementation of a 1.5-bit stage

be correct. The output value of stage 1 would be $2 \times 1.2V_r/4 = 1.2V_r/2$. Stage 2 will then decide that the quantized word is "10", even if it has the same offset in comparator. To combine stage output words they are shifted and summed leading to

```
Stage 1:             01
Stage 2 MSB:          1
Corrected Stage 1:   10
```

Thus a 1.5-bit pipelined stage can tolerate an offset in comparators up to $\pm V_r/4$, greatly reducing the required accuracy of the stage ADC.

3.3.4 Reducing Power in Pipelined ADCs

The sample-and-hold in front of the converter and the opamps in each stage dissipates most of the power in a pipelined converter [7]. Note that the opamp in a stage is only used during half the clock period. Techniques have been developed that switch off the opamp during half the clock

period to reduce power dissipation, however this technique is normally not suitable for high-speed designs due to the slow turn on time of high-speed opamps [10] . Another technique is to share an opamp between two stages [2].

3.3.5 Pipelined ADC Summary

A switched capacitor amplifier, like the one used in 1.5-bit stage, requires a high-gain opamp. In the first stage the gain normally needs to be slightly higher than the resolution of the pipelined ADC. For example, for a 10-bit converter we may need 70dB gain in the first stage opamp [10]. High-speed, high-gain opamps have high power dissipation and may be difficult to implement in modern nanoscale processes. Too low gain in the opamp leads to incorrect settling of the switched-capacitor amplifier, in short the gain is less than 2 for a 1.5-bit stage. Due to the architecture of the pipelined converter this leads to non-linear distortion. Calibration techniques have been developed to deal with this problem [11]. Other effects that limit performance of pipelined converters will be presented in Chapter 4.

3.4 Time-Interleaved ADCs

Time-interleaved ADCs are used when very high bandwidths and sampling rates are required. Applications include digital oscilloscopes with bandwidths up to 6GHz and sampling speeds of 20GS/s [2]. In Figure (3.18) an example of time-interleaving is shown. Here three ADC's are interleaved. In front of each ADC is a sample and hold. For this example the sampling clock of each S/H runs at 1/3 of the sampling frequency f_s. A timing diagram is shown in Figure (3.19). Each of the clock phases $clk_1 - clk_3$ are out of phase. The output from the ADC, shown in Figure (3.19) as D_1, D_2, D_3, are fed to a digital multiplexer which combines the three interleaved signals into the output D_{out}.

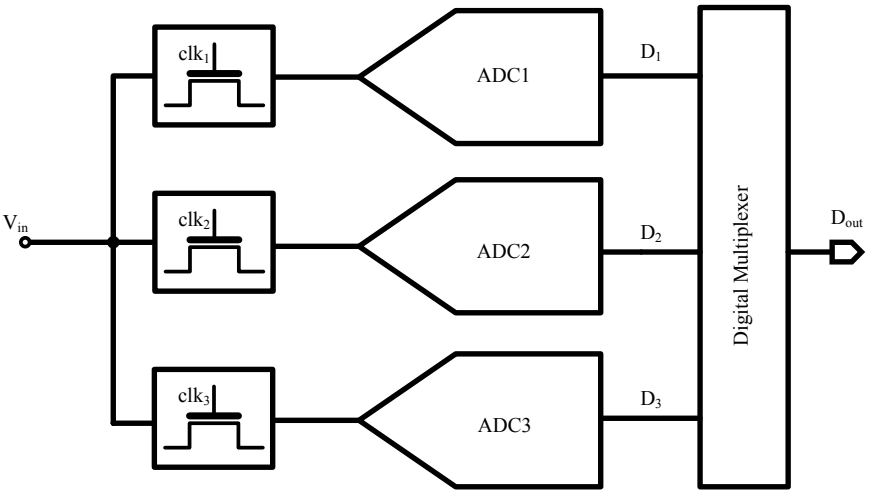

Figure 3.18: Time-Interleaving of three ADCs

Problems occur in time-interleaved converters because of the difference in offset and gain errors [12] between the ADCs. The difference in errors between the ADCs will, in this example, produce in-band signals at one-third of the sampling frequency f_s. Therefore time-interleaved converters may require trimming of offsets and gain.

Most ADC architectures can be used for interleaving. It does not necessarily have to be one of the high-speed architectures. In [13] eight successive-approximation (SAR) ADCs were interleaved. SAR ADCs are normally associated with slow but accurate ADCs. With this architecture they

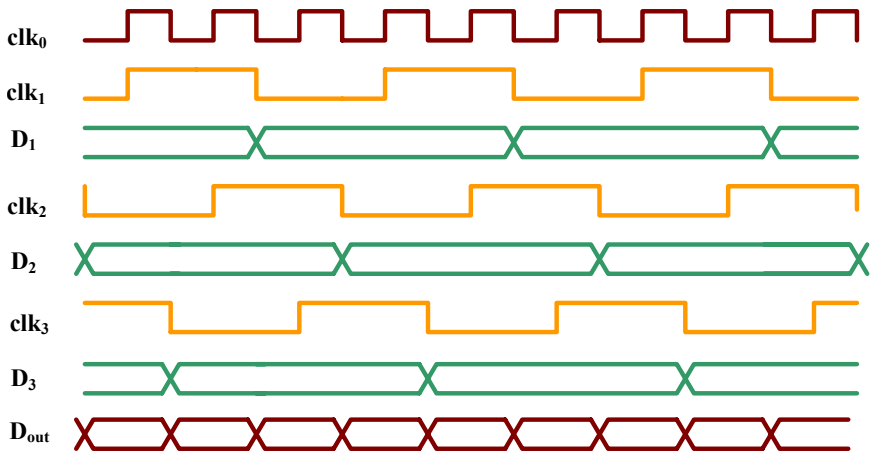

Figure 3.19: Timing of time-interleaved converter

achieved a performance of 6-bit, 600MHz maximum input frequency and a power dissipation of 10mW, resulting in a FOM of 0.4pJ per conversion step, which is a quite good result.

Chapter 4

Calibration Techniques For Pipelined ADCs

In this chapter we will discuss some of the calibration techniques that have been developed to deal with problems in pipelined ADCs. There are a multitude of papers on calibration available, and the techniques described here are only a sampling of the whole.

Each stage in a pipelined converter has a DAC. In high-resolution multi-bit stage pipelined converters these DACs, often implemented as switch capacitor circuits, suffer from non-linearities due to mismatch between unit capacitors in the DAC [11]. These non-linearities produce harmonics in the frequency spectrum of a converter. A technique called dynamic element matching (DEM) can be used to spread the power of these harmonics.

4.1 Dynamic Element Matching

An example of a simple current DAC can be seen in Figure (4.1). The output value of the DAC is determined by the sum of currents and the configuration of the output from the thermometer encoder. The DAC output, V_{out}, is given by

$$V_{out} = RI_{ref}[T_0(1+e_0) + T_1(1+e_1) + T_2(1+e_2) + T_3(1+e_3)] \tag{4.1}$$

where R is the resistance, I_{ref} is the reference current, T_i is the output from the thermometer encoder (T_i is 1 or 0) and e_i is the error introduced by the different paths. If the error is exactly the same for all paths, $e_0 = e_1 = e_2 = e_3$, the errors will introduce a gain error in the output, but the DAC will still be linear. This is only true if the summed currents are unit currents. Mismatch of devices makes it highly unlikely that all errors are exactly the same. The magnitude of the error is often randomly distributed around a typical value. The error is usually constant with respect to time. With different values for the errors, there will be a deterministic non-linear relationship between the output signal and the error, resulting in distortion.

A DAC with DEM can be seen in Figure (4.2). The number of current paths in this DAC has doubled to allow for DEM implementation,. It is not strictly necessary to have double the number of paths, but it is common to implement an over-range. After the thermometer encoder we have a scrambler that selects the correct number of currents to sum, which it does in a random fashion. This decorrelates the error from the output signal, which smears the power of the harmonics produced by non-linearities out in the frequency spectrum, effectivly reducing the distortion into noise. A pseudo random sequence is often used at the input of the scrambler.

An example of the output spectrum with and without DEM can be seen in Figure (4.3) [11].

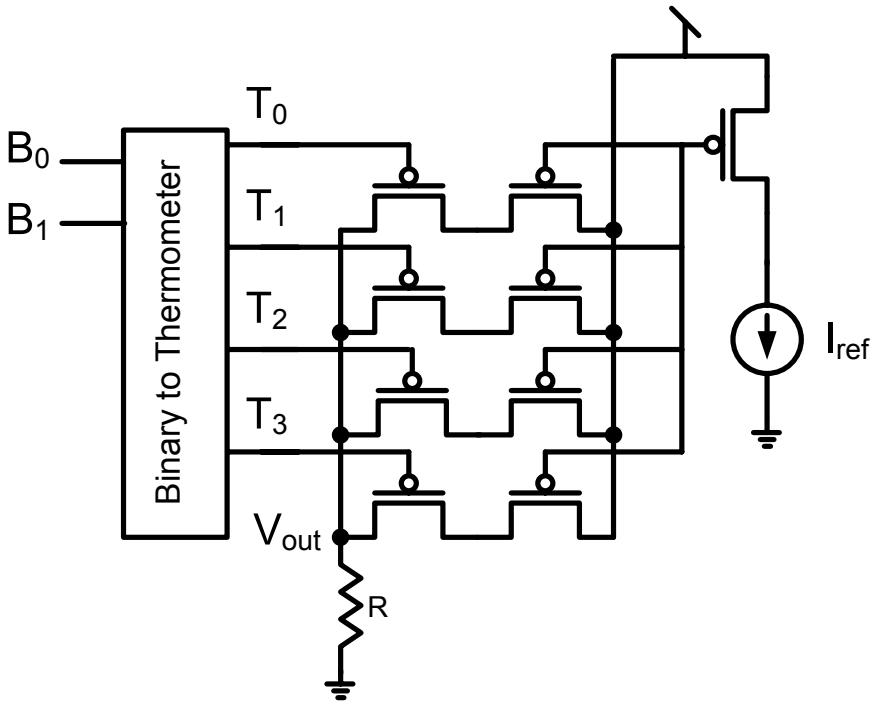

Figure 4.1: Simple Current DAC

4.2 DAC Noise Cancellation

DAC Noise Cancellation (DNC) is an extension of DEM encoding. It makes use of the fact that the DAC error can inherit some of the statistical properties of the pseudo-random signal used in the DEM encoding. In [11] a DEM encoder is implemented in such a way that the DAC error is given by

$$e_{DAC}[n] = \sum_{k} \sum_{r} \Delta_{k,r} s_{k,r}[n] + \Delta s_{seg} s_{seg}[n] \tag{4.2}$$

where $k \in 1, 2, 3$, $r \in 1, 2, 3, 4, 5$ and seg are indexes of switching blocks within the DEM encoder, $\Delta_{k,r}$ and Δ_{seg} are the constant errors due to mismatch and $s_{k,r}[n]$ and $s_{seg}[n]$ are the pseudo-random signals. For an explanation of how the DEM encoding is performed we refer to [11]. Sufficed to say, the error $\Delta_{k,r}$ or Δ_{seg} is the error of a specific path in the DAC. If we can find the magnitude of this constant error, it can be subtracted from the output code, thus reducing or removing the error introduced by that specific path in the DAC. In [11] a 9-level DAC was implemented, the DNC correction of the first stage is shown in (4.4). The residue signal from the pipeline is re-quantized to reduce the size of the DNC logic. The residue is then correlated with the each of the 9 pseudo-random sequences. After correlation the correlated signal is averaged, thereby estimating the value of $-\Delta_{k,r}$ or $-\Delta_{seg}$. The estimated error is then multiplied by the pseudo-random signals and the resulting value is subtracted from the residue signal of the ADC. The DNC technique effectively measures the individual error of each of the DAC paths and subtracts an equivalent error if the DAC path has been used in that particular residue code.

4.3 Gain Error Correction

Gain error correction (GEC) is used to correct for gain errors in pipelined stages due to for example, finite gain in stage opamps. An example of GEC is shown in (4.5). The first stage of a pipeline converter is shown, and the rest of the stages are shown as an ideal back-end ADC.

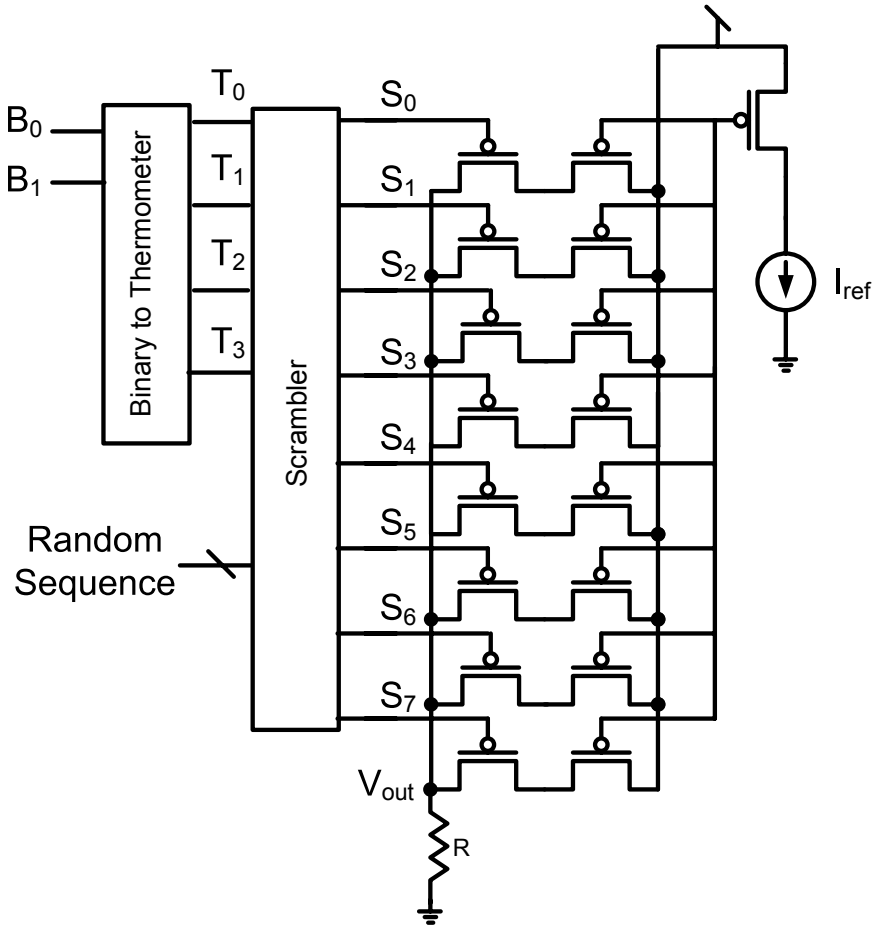

Figure 4.2: Current DAC with DEM encoder

The interstage gain is represented as $G(1+\varepsilon)$ where ε represents the gain error. The error in the interstage gain will introduce a non-linearity in the transfer function of the ADC, and thus reduce SFDR and SNDR [11].

A pseudo-random sequence, r_{gec}, is added at the input of the DAC; r_{gec} is white and has a mean of zero. This travels the same path as the residue[1], and see the same gain. The output from the back-end ADC can be written as

$$y_{backend} = G(1+\varepsilon)[y_{residue} - r_{gec}] \tag{4.3}$$

By calculating the covariance between $y_{backend}$ and r_{gec}, we get an estimate of the power due to r_{gec} in $y_{backend}$. We could write

$$P_{r_{gec}} = \frac{1}{N}\sum_{n=0}^{N} y_{backend} r_{gec} = -(1+\varepsilon)\frac{1}{N}\sum_{n=0}^{N} r_{gec}^2 \tag{4.4}$$

where N is a large number. Note that the power is proportional to $-(1+\varepsilon)$. This can be used to get the estimate $-(1+e)$ shown in Figure (4.5), where e is the estimate of ε. To get only the error estimate we add 1 to $-(1+e)$. For the error estimate to be accurate the path traveled must be linear and the residue, $y_{residue}$, should be uncorrelated with r_{gec}. If the residue is correlated with

[1]We use residue to mean both the signal before and after the gain. Note that the gain is there to expand the full-scale of the residue to the full-scale input of the stage. The gain does not alter the fact that it is a residue.

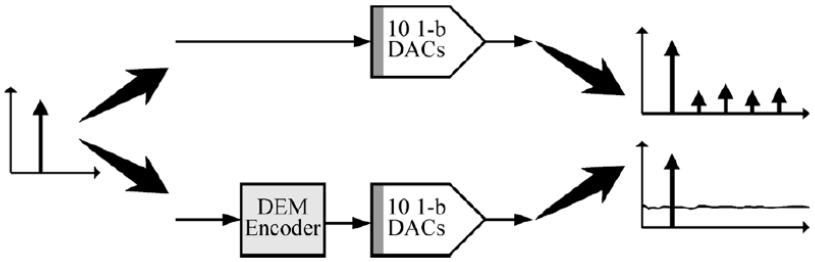

Figure 4.3: Effects of DEM on the output spectrum when mismatches are present.

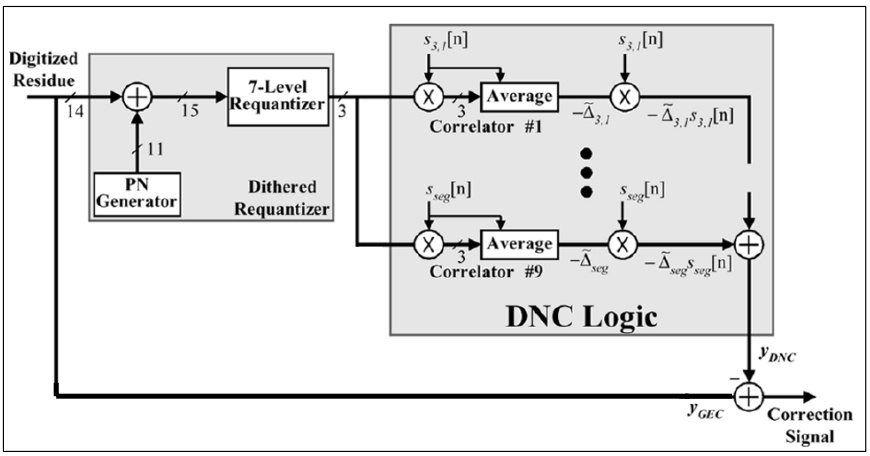

Figure 4.4: First-stage DNC correction logic

r_{gec} then (4.4) is incorrect. To correct the back-end signal we could multiply it by $-e/(1+e)$ and add it to the ADC output signal, but if we assume that the gain error is small then $1+e \approx 1$ [11]. As we see from the figure we multiply the residue by the gain error estimate and add it to the overall output.

4.4 Example of DNC and GEC

In [11] they present a 15-bit 40Msamples/s pipelined ADC with a peak SNR of 72 dB[2]. They achieve a FOM of 3.0pJ per conversion step. One of the problems with techniques like GEC and DNC is the time it takes for the converter converge. One adjustment of the estimate, e, might take up to $2^{20} - 2^{30}$ clock cycles [7]. To our knowledge this type of calibration has yet to be used in a commercial product.

4.5 Reference Scaling

In contrast to GEC where correction of interstage gain error is performed in the digital domain one can do the correction of the gain error in the analog domain. In [14] they used reference scaling, in other words changing the reference voltage of the DAC in the pipelined stage. They prove that scaling the reference voltages of the DAC is equal to correcting the gain error as was shown for GEC. The detection of a gain error used in [14] is similar to GEC. A pseudo-random test signal was injected into the residue path to measure the gain error. The error estimate was used

[2] An ENOB of 11.67-bit

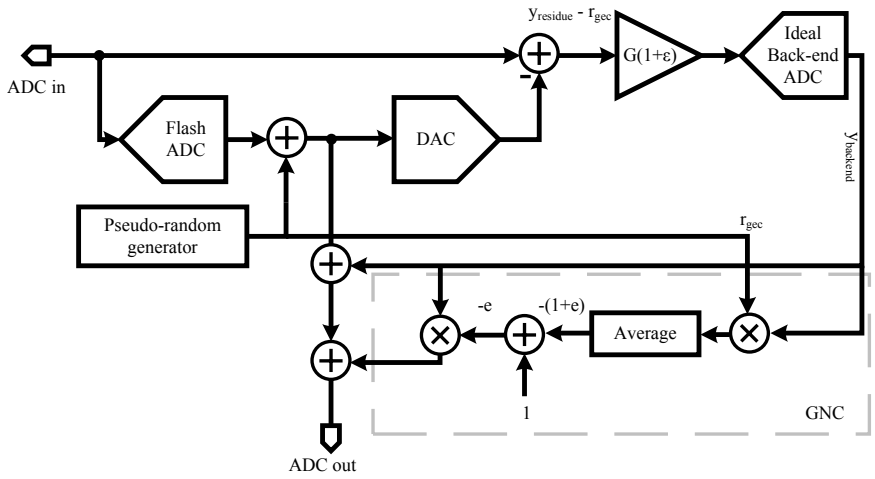

Figure 4.5: A pipelined ADC implementing gain error calibration (GEC)

to adjust the reference voltages of the DAC. A slow-but-accurate ADC was used to quantify the actual value of the adjusted reference value. In [14] the pipelined ADC achieved a performance of 8-bit, 80Msamples/s with a power dissipation 250mW if the calibration was frozen. This is equivalent to a FOM of 23.1 pJ per conversion step.

4.6 Calibration of Third-Order Distortion in Interstage Gain

The conventional GEC corrects for a linear error, a gain error, in the interstage gain. In [15] they present a method of detecting and calibrating gain error and third order distortion in the interstage gain. They show that by calculating the covariance of the square of the backend signal and the pseudo-random test signal, it is possible to get an estimate that quantifies the amount of the third-order distortion in interstage gain. This estimate is used to correct for third-order distortion.

Another approach to third-order distortion calibration was presented in [16]. Here they randomly changed the residue transfer function of the pipelined stages, giving two distinct transfer functions. In Figure (4.6) [16] a section of the the two transfer functions is shown. A statistics-based distance estimation was used to estimate the distances h_1 and h_2. If these two distances are unequal, there is a non-linearity present. They tweak the output from each stage so that the non-linearity no longer appears. In [16] they used an open loop amplifier as the gain in the pipelined stage. With the third-order distortion calibration in addition to more conventional gain calibration they achieved a performance of 12-bit 75Msamples/s with a power consumption of 290mW. This result in a FOM of only 2pJ per conversion step.

4.7 Skip & Fill

Skip and fill is a technique where one steals conversion cycles from the normal operation of the ADC to do calibration. This requires that the missing sample is replaced. In [17] they used a non-linear interpolation to estimate the missing sample, which is the same as fitting the output to a high-order polynomial. Both causal and non-causal taps were used, which means that samples from before and after the missing sample were used to estimate the value of the missing sample. Each sample before and after the missing sample was multiplied by a coefficient, and then summed as

$$x(0) = \sum_{k=-1}^{-n} x(k)C(k) + \sum_{k=1}^{n} x(k)C(k) \tag{4.5}$$

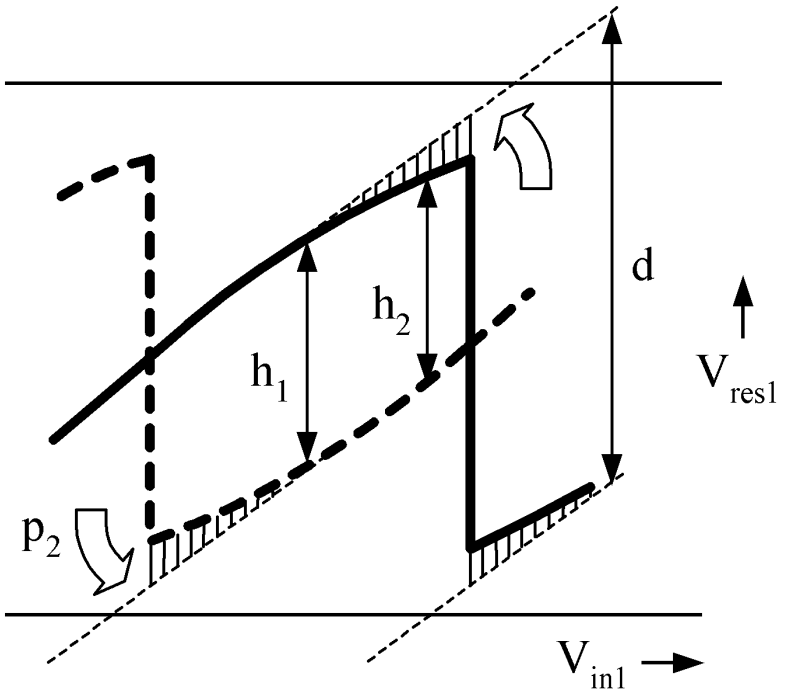

Figure 4.6: Section of stage transfer function

where n was the index of the sample, C the coefficient and x(0) the missing sample. C(k) was defined as

$$C(k) = \frac{n!n!}{(n+k)!(n-k)!}(-1)^{k+1} \tag{4.6}$$

To get 16-bit accuracy, they needed n = 22, resulting in 44 multiplications and summations of 16-bit numbers to get an estimate of the missing sample. This might lead to significant power dissipation in the non-linear interpolation filter.

4.8 Calibration of Memory Effects

Memory effects in pipelined converters can stem from

1. Capacitor dielectric absorption/relaxation effects
2. Incomplete stage reset effects
3. Opamp sharing

These memory effects can lead to non-linearities in a pipelined ADC [18]. In [18] they propose two methods to reduce memory effects. One is a Least Mean Square (LMS) algorithm with a slow but accurate ADC in parallel with the main ADC. The other uses DAC dithering akin to the GEC technique. Least Mean Square has faster settling and gives best results, but the tracking is dependent on the input signal statistics. In general one do not want a calibration algorithm which is dependent on the input signal statistics [10]. The dither method is independent of signal statistics, and does not require an accurate parallel ADC. On the other hand it has slow convergence and requires additional levels in the ADC and the DAC in the pipelined stages.

4.9 Calibration of Pipelined ADC Using Stage Alteration

Two multiplying DACs (MDACs) can be placed in parallel within the stage and ran at $f_s/2$. This effectivly produces a back-end ADC that has the same properties as a time-interleaved converter [19]. According to theory of time-interleaved converters gain mismatches between the two paths produce in-band images at $kf_s/2 \pm f_{in}$, k is an integer, f_s is the sampling frequency and f_{in} is the input frequency. By using previously published techniques to correct for gain mismatches between paths in time interleaved ADCs and alternating the configuration of the MDAC between a $radix = 2$ and $radix < 2$, it is possible to extract estimates for the gain errors and correct them [19].

4.10 Slow But Accurate ADC

A slow but accurate ADC can be connected in parallel to the high speed ADC. The slow ADC will produce an accurate value at f_s/M where M is the speed difference between the two converters. In [20] they connect the output of the fast ADC through a non-linear filter where the coefficients are adjusted by a least mean square algorithm that compare the output of the slow-but-accurate ADC with the output of the non-linear filter. This bear a resemblance to equalization of communication channels [20]. The system corrects for linear stage errors like capacitor mismatch, finite opamp-gain and input-referred offsets.

Chapter 5

State-of-the-art ADCs

This chapter presents some of the recent state-of-the-art ADC publications.

5.1 Power and Speed Programmable Pipelined ADC

One of the key elements in display systems is a low power ADC with an ENOB exceeding 9-bits [21]. The ERWB is proportional to the number of pixels and refresh rate. The ADC in the system may not be running at full capacity all the time, and it is an advantage to scale the power consumption. In [21] they present"A 90nm CMOS 1.2V 10-bit power and speed programmable pipelined ADC with 0.5pJ/Conversion-Step". A block diagram of the converter can be seen in Figure 5.1. The converter architecture is a pipelined ADC with stage scaling.

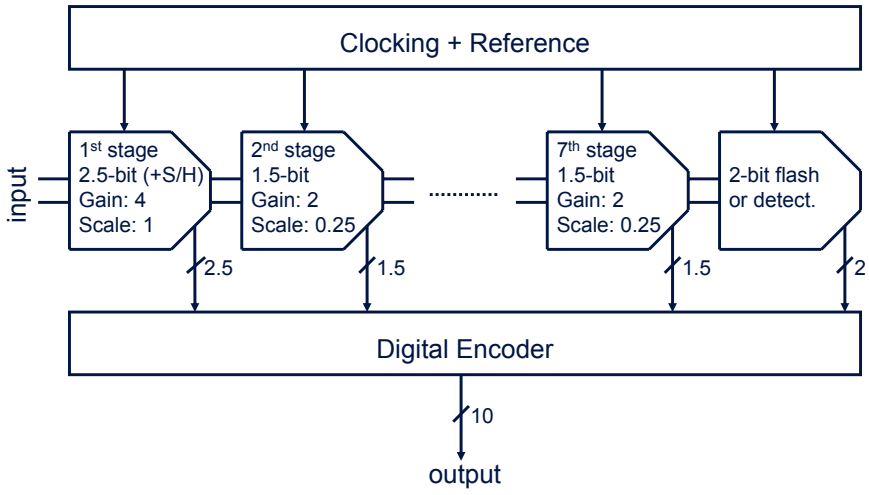

Figure 5.1: Block diagram of the ADC

The first stage has 2.5-bit resolution, the middle stages are 1.5-bits and the last stage is a 2-bit flash. In addition to an over-range in each stage, they have also used an over-range in the last flash stage. The stage is implemented using a conventional switched-capacitor multiplying digital to analog converter (MDAC).

One of the keys to the performance of [21] is the implemented opamp with a constant gain over a large bias range. It is a two-stage miller compensated opamp with a folded cascode as the first

stage, shown in 5.2. For this architecture the gain, A_0, is proportional to the transconductance, g_m, over the output conductance, g_o

$$A_0 \propto \frac{g_m}{g_o} \tag{5.1}$$

They achieved a gain of $A_0 > 65dB$ over a large bias range. The bandwidth of the opamp roughly determined by

$$\omega_b = \frac{g_m}{C_{miller}} \tag{5.2}$$

where ω_b is the angular frequency of the dominate pole, g_m is the transconductance and C_{miller} is the Miller capacitance. The bandwidth of the opamp is one of the determining factors for the sampling frequency, f_s. To achieve a certain resolution without calibration, the bandwidth of the opamp needs to be a certain multiple of f_s, otherwise the ENOB is degraded [1].

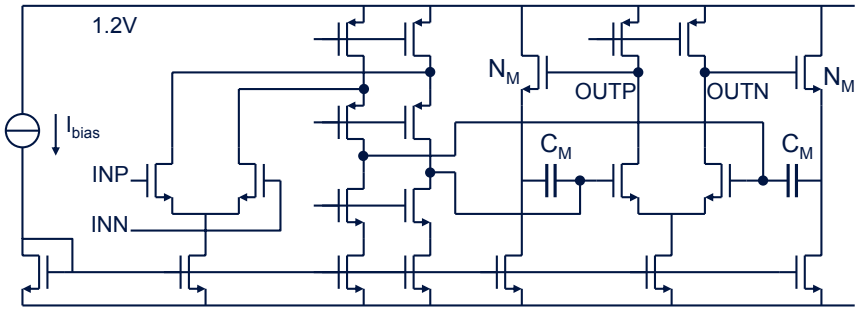

Figure 5.2: Schematic of the differential two-stage miller opamp

A comparison of this converter versus other previously published converters is shown in 5.3. We can see how the FOM of this converter is lower than the FOM of others. However, it has been commented that it might not be a fair comparison since [22] (ISSCC04 Hernes) was designed to work at almost double the sampling frequency of [21].

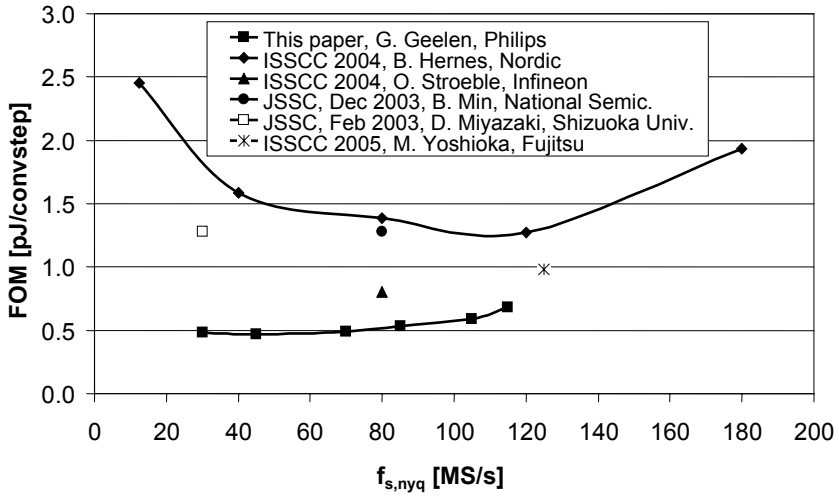

Figure 5.3: Measured FOM of previously published ADCs

The performance summary is shown in 5.4 [21]. Other interesting things about this converter is for example the input swing of $0.8V$ peak to peak differential. With this differential swing only $\frac{1}{3}$ of the supply voltage ($\frac{0.8V}{2} = 0.4V$) is used. One might speculate that this low swing is to get good linearity.

Architecture	Pipeline 1x2.5b, 7x1.5b stages
Technology	90nm digital CMOS, 1poly, 6metal
Supply voltage	1.2V
Input range	$0.8V_{pp,diff}$
Resolution	10b
DNL/INL	< 1.0LSB
ENOB	9.3b
SNR	58.5dB
THD	< -65dB
ERBW	> 100MHz
Fs,max	25 - 120MSample/s
Cin	1.0pF
Area	$0.3mm^2$
Power	0.3mW/MSample
FOM	0.5pJ/convstep

Figure 5.4: Performance summary of 10-bit pipelined power and speed programmable ADC

5.2 Pipelined ADC with Opamp Current Reuse

In [23] they present "A 10-bit 50MS/s Pipelined ADC with Opamp Current Reuse". Two techniques used to reduce power dissipation in pipelined ADC's are opamp sharing and switched-opamp. Both techniques take advantage of the fact that the opamp is only used during half the clock period. Opamp current reuse bear a resemblance to these techniques. A block diagram of the ADC is shown in Figure 5.5. It is a 10-bit pipelined ADC where two opamps are shared between the MDACs. Each stage has 3-bit resolution and a 2-bit flash is used to convert the LSB's. The opamp current reuse idea is based on the fact that a differential opamp is often designed with a NMOS differential input and a PMOS differential active load, or visa versa. The opamp current reuse stems from switching which of the transistor pairs serve as the input stage. For the MSBs the NMOS differential pair is used as input and input of the PMOS transistors are connect to a reference voltage, thus serving as an active load. For the LSB's the PMOS differential pair is used as input and the NMOS stage is used as active load. A schematic of the telescopic cascode opamp with gain boosting used is shown in Figure 5.6. Since the input transistors change, it is possible to reset the summing node in the MDAC of the individual stages. Inability to reset input is a problem in conventional opamp sharing [23]. The gain boosting cascode uses capacitive level shifting so that a NMOS cascode amplifier can be used for both the PMOS and NMOS gain boosting cascodes [23].

This converter has a FOM of 0.8pJ per conversion step, which is slightly worse than [21]. During discussion of this opamp current reuse architecture it was commented that "This is not something that I will be trying" [10]. Although the architecture is interesting, it is unsure if it is

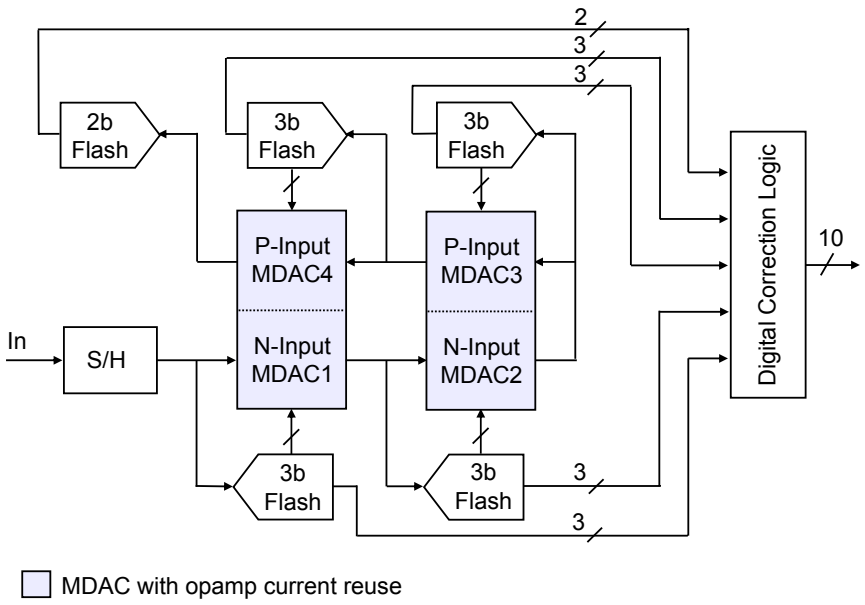

Figure 5.5: Block diagram of 10-bit pipelined ADC with opamp current reuse

possible to implement in nanoscale CMOS technology due to the transistor stacking in the opamp. Also a better result was achieved in [21] using more conventional techniques.

5.3 Comparator-Based Switched-Capacitor Circuits For Scaled CMOS Technologies

Switched-Capacitor circuits are prevalent in analog integrated circuit implementations. An example of a common switched-capacitor amplifier is shown in Figure 5.7 [24]. In the reset phase (not shown), the input voltage, V_{in} is sampled onto capacitor C_2. During the amplifying phase the opamp forces a virtual ground in the summing node V_x. This transfers the stored charge on C_2 to C_1 and the output voltage, V_o, becomes

$$V_o = \pm\frac{C_2}{C_1}V_{in} \tag{5.3}$$

Whether the amplifier is inverting or not, is determined by how sampling is performed in the reset phase. In addition to the capacitor matching, the gain of the opamp determines the accuracy of the switched-capacitor amplifier. In nanoscale CMOS technology, it has become more difficult to implement high gain opamps, among other things due to reduced supply voltage. In addition to this a high-gain, high-bandwidth opamp, which is needed in high-accuracy and high-speed switched-capacitor circuits, has a high power dissipation.

What is presented in [24] is a radical approach to switched-capacitor circuits. A MDAC in a pipelined ADC is often implemented as a switched capacitor amplifier. It is not important how the switched-capacitor circuit arrive at the final output voltage since sampling of the output voltage occur at the end of the clock period. Instead of forcing the summing node V_x to virtual ground with an opamp, a comparator is used to detect when virtual ground occurs at the summing node. A schematic of such a system is shown in Figure 5.8. In the amplifying phase a current source is connected to the output, charging the capacitors C_1 and C_L. The comparator compares the voltage V_x to the common mode V_{CM}, and when it detects virtual ground at V_x, it turns off the current source. The plots on the right side in Figure 5.7 and Figure 5.8 shows the voltage V_x

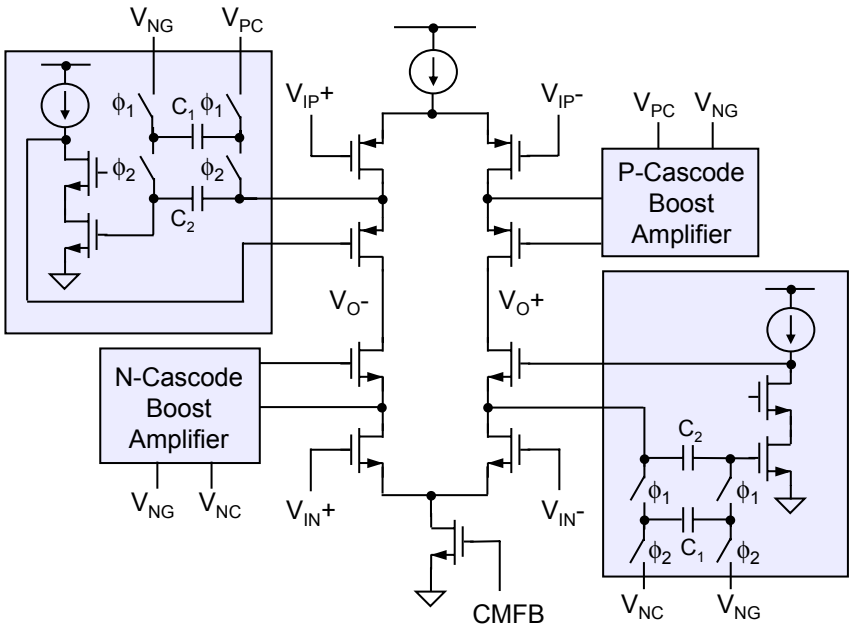

Figure 5.6: Telescopic cascode opamp with gain boosting

versus time in the amplifying phase. It might look like the comparator-based solution reaches its final value slower than the opamp based solution. This is a potentially erroneous conclusion. For the opamp based solution to reach the final value within a certain accuracy it normally slews for a while, then goes into linear settling when the opamp reaches its operating point. The comparator based solution only slews, and might therefore reach its final value faster. Another potential advantage of the comparator-based solution is that it is easier to implement a high accuracy (high gain) comparator than a high gain opamp since one do not need to consider the phase margin of the comparator.

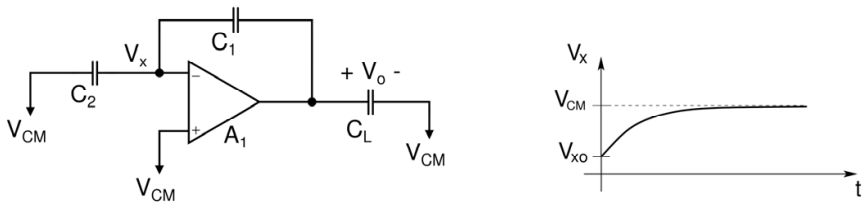

Figure 5.7: Switched-Capacitor Amplifier

In [24] the comparator-based switched capacitor MDAC was implemented using a dual ramp settling of the output as shown in Figure 5.9. A large current is connected to the output to charge the capacitors. After a while, virtual ground is detected, but due to finite speed of the comparator the output voltage will overshoot. A smaller current source is connected to sink current from the output. The virtual ground is detected again as shown plots on the right in 5.9. This architecture thus implements a fast settling with overshoot, followed by a slow-but-accurate settling. If the overshoot is small, the second settling need not take long.

A prototype 10-bit pipelined ADC was implemented using the comparator-based switched capacitor circuit technique and they achieved a FOM of 0.3pJ per conversion step. Although the

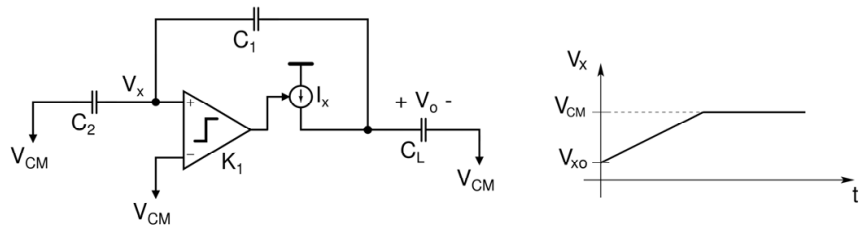

Figure 5.8: Comparator-Based Switched-Capacitor Amplifier

FOM is better than [21] it is to soon to declare the comparator-based switched-capacitor technique as superior to conventional switched-capacitor circuits. More research is needed on the limitations of this technique. However, it has the potential of rendering calibration techniques developed to compensate for insufficient gain in opamps a thing of the past.

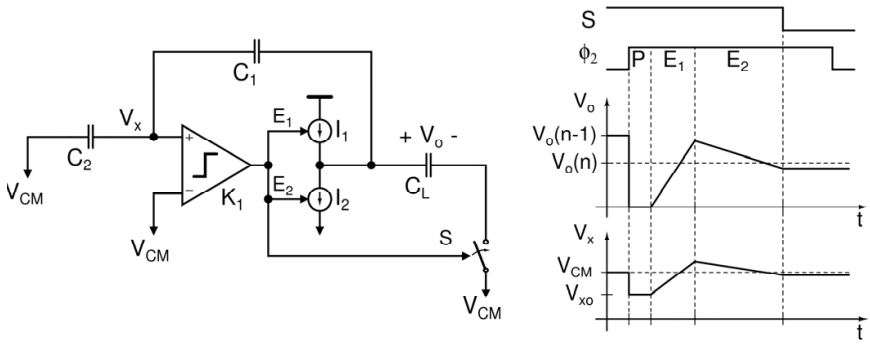

Figure 5.9: CBSC charge transfer phase using dual-ramp settling. Sampling phase not shown.

5.4 1GS/s 11b Time-Interleaved ADC

Time-Interleaved converters are presently the only method available to achieve ADCs with high-resolution and high conversion rates in the multi gigahertz range. Time-Interleaving converters have problems with offset, gain mismatch and timing mismatch error between the interleaved ADCs. In [12] they focus on the timing errors in time-interleaved ADCs. A block diagram of the ADC can be seen in Figure 5.10. The input is sampled by a bootstrapped sample and hold (SH switch in Figure) operating at the full sampling frequency f_s. The output from the first switch is sampled by N sub sample and holds (Sub-SH in figure). These Sub-SH are timed so that no more than one Sub-SH loads the SH switch for any appreciable time. Thus the bandwidth and linearity of the SH switch is not limited by Sub-SH [12]. The Sub-SH is turned off during off phase of the SH switch, thus the Sub-SH do not limit SNR. A pipelined ADC is used as the interleaved ADC. To reduce power dissipation, a double sampling technique is used in the interleaved ADCs. In effect double sampling entails that one have two parallel pipelined ADCs running out of phase. The opamps in the MDACs can be shared between the two pipelined ADCs since in a normal pipelined ADC the opamp is only used during half the clock period.

The converter has the best SNR and SNDR of any converter with a sampling speed above 500MHz. It has a FOM of below 0.5pJ per conversion step.

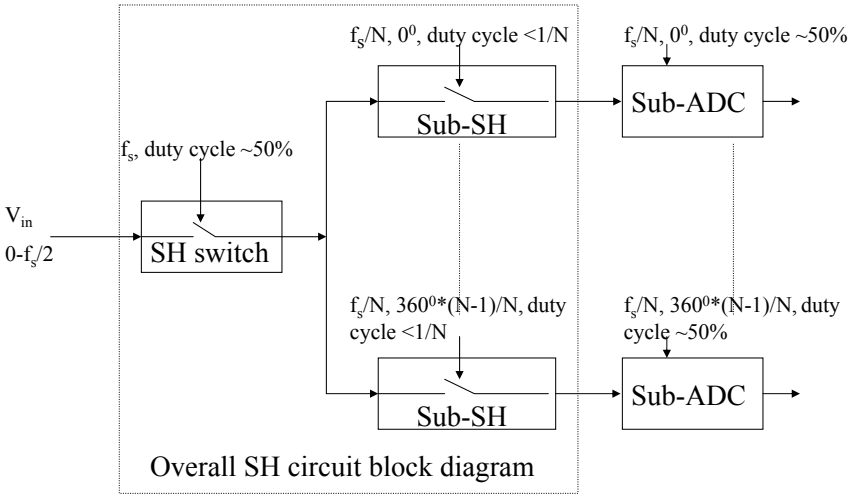

Figure 5.10: Proposed Time-Interleaved ADC architecture block diagram

Chapter 6

Concluding Remarks

This chapter contain some informal thoughts about the topics in this report. Thoughts like these might fly in the face of the accepted standard of report writing, but we believe they are appropriate.

6.1 Thoughts on ADC Architectures

To gauge the popularity of the different architectures we did an informal search on IEEE Xplore web site in the conference proceedings of ISSCC. Table 6.1 shows how many papers the different architectures have. This is by no means a proper sampling of all papers presented at ISSCC. For example, we would not say that folding is more popular than sub-ranging. But it gives an idea of where most research is being performed.

Architecture	Words searched	Latest reference	Hits
Full-flash	flash adc	2004	20
Two-Step Flash	two-step adc	2001	6
Sub-ranging	subranging adc	2004	5
Folding	folding adc	2004	6
Pipelined	pipelined adc	2005	25
Time Interleaved	time interleaved adc	2004	2

Table 6.1: Popularity of different architectures at International Solid State Circuit Conference.

6.2 Thoughts on Calibration

There has been several publications in the last years on calibration algorithms for pipelined ADCs. When reading calibration papers, it is easy to loose track of the point of calibration. Calibration is supposed to make the performance of ADCs better. The FOMs of the three calibration ADCs presented in this report are 3.0pJ [11], 23.1 pJ [14] and 2.0pJ [16]. Two of these figures are quite good, but in [16] it is unclear whether the calibration algorithm is included in the power calculations. The argument could be made that calibration is needed to achieve high ENOB above the 10-bit mark. But as previously mentioned, it is unknown whether any commercial products has been made using one of these calibration algorithms.

Bibliography

[1] D. Johns and K. Martin, *Analog Integrated Circuit Design*. John Wiley & Sons, Inc., 1997.

[2] R. van de Plassche, *CMOS Integrated Analog-to-Digital and Digital-to-Analog Converters*. Kluwer Academic Publishers, 2003.

[3] A. V. D. Ziel, *Noise in Solid State Devices and Circuits*. John Wiley & Sons Inc, 1986.

[4] N. M. Blachman, "The intermodulation and distortion due to quantization of sinusoids," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 33, no. 6, pp. 1417–1426, 1985.

[5] B. Widrow, "A study of rough amplitude quantization by means of nyquist sampling theory," *Circuit Theory, IRE Transactions on*, vol. 3, no. 4, pp. 266–276, 1956.

[6] B. Hernes and W. Sansen, "Distortion in signle-, two- and three-stage amplifiers," in *IEEE Trans. Circuits Syst. I*, vol. 52, no. 5, 2005, pp. 846–856.

[7] J. Bjørnsen, "Private conversations with."

[8] B. Nauta and A. Venes, "A 70-ms/s, 110mw, 8-bit cmos folding and interpolation a/d converter," *IEEE J. Solid-State Circuits*, vol. 30, no. 12, pp. 1302–1308, 1995.

[9] G.Geelen, "A 6b 1.1 gsample/s cmos a/d converter," in *ISSCC Digest of Technical Papers*, 2001.

[10] B. Hernes, "Private conversations with."

[11] E. Siragusa and I. Galton, "A digitally enhanced 1.8v 15-bit 40-msample/s cmos pipelined adc," *IEEE J. Solid-State Circuits*, vol. 39, no. 12, pp. 2126–2138, 2004.

[12] S. Gupta, M. Choi, M. Inerfield, and J. Wang, "A 1gs/s 11b time-interleaved adc in 0.13μm cmos," in *ISSCC Digest of Technical Papers*, 2006, pp. 576–577.

[13] D. Draxelmayr, "A 6b 600mhz 10mw adc array in digital 90nm cmos," in *Proc. ISSCC'04*, vol. 1, 2005.

[14] J. Ming and S. H. Lewis, "An 8-bit 80msample/s pipelined analog-to-digital converter with background calibration," *IEEE J. Solid-State Circuits*, vol. 56, no. 10, pp. 1489–1497, 2001.

[15] J. P. Keane, P. J. Hurst, and S. H. Lewis, "Background interstage gain calibration technique for pipelined adcs," *IEEE Trans. Circuits Syst. I*, vol. 52, no. 1, pp. 32–43, 2005.

[16] B. Murmann and B. E. Boser, "A 12-bit 75-ms/s pipelined adc using open-loop residue amplification," *IEEE J. Solid-State Circuits*, vol. 38, no. 12, pp. 2040–2050, 2003.

[17] U.-K. Moon and B.-S. Song, "Background digital calibration techniques for pipelined adc's," *IEEE Trans. Circuits Syst. II*, vol. 44, no. 2, pp. 102–109, 1997.

[18] J. P. Keane, P. J. Hurst, and S. H. Lewis, "Digital background calibration for memory effects in pipelined analog-to-digital converters," *IEEE Trans. Circuits Syst. I*, vol. 53, no. 3, pp. 511–525, 2006.

[19] K. El-Sankary and M. Sawan, "A new digital background calibration technique for pipelined adc," in *Proc. ISCAS 2004*, vol. 1, 2004, pp. I.5–I.8.

[20] Y. Chiu, C. W. Tsang, B. Nikolic, and P. R. Gray, "Least mean square adaptive digital background calibration of pipelined analog-to-digital converters," *IEEE J. Solid-State Circuits*, vol. 51, no. 1, pp. 38–46, 2004.

[21] G. Geelen, E. Paulus, D. Simanjuntak, H. Pastoor, and R. Verlinden, "A 90nm cmos 1.2v 10b power and speed programmable pipelined adc with 0.5pj/coversion-step," in *ISSCC Digest of Technical Papers*, 2006.

[22] B. Hernes and et al, "A 1.2v 220ms/s 10b pipeline adc implemented in 0.13μm digital cmos," in *ISSCC Digest of Technical Papers*, 2004, pp. 256–257.

[23] S.-T. Ryu, B.-S. Song, and K. Bacrania, "A 10b 50ms/s pipelined adc with opamp current reuse," in *ISSCC Digest of Technical Papers*, 2006, pp. 216–217.

[24] T. Sepke, J. K. Fiorenza, C. G. Sodini, P. Holloway, and H.-S. Lee, "Comparator-based switched-capacitor circuits for scaled cmos technologies," in *ISSCC Digest of Technical Papers*, 2006, pp. 220–221.

[25] B. Razavi, *Design of Analog CMOS Integrated Circuits*. McGraw-Hill, 2001.

Appendix A

Code

This appendix is a collection of code examples that have produced some of the graphics in this report.

A.1 Matlab Example On the Effects of Jitter

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Fjitter.m
%%% Description:   Example of effects of jitter
%%% Author:         <wulff@iet.ntnu.no>
%%% Created at:    Fri Apr 21 12:35:42 2006
%%% Modified at:   Fri Apr 21 12:36:56 2006
%%% Modified by:        <wulff@iet.ntnu.no>
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
samp = 2^10;
P = 2^10; %Samples that should be taken from input signal
N = P*samp; %Samples in input signal
R = 2^-10;
x = sin (2 *  pi * 109 * (1:N)/N) + R*(1- rand(1,N));;
out1 = zeros(1,P/2+1);
out2 = zeros(1,P/2+1);
y1 = zeros(1,P);
y2 = zeros(1,P);
r =round(samp/10*(1- rand(1,P)));
for i=1:P
    index = i*samp + r(i);
    if (index > N-1)
        break;
    end
    y1(i) = x(index);
    y2(i) = x(i*samp);
end

out1 = myfft(y1,P);
out2 = myfft(y2,P);

%-----------------------------------------------------------
%Create figure
%-----------------------------------------------------------
```

```
figure1 = figure(1);
axes1 = axes(...
  'XGrid','on',...
  'YGrid','on',...
  'Parent',figure1);
axis(axes1,[0 0.5 -120 0]);
xlabel(axes1,'Samples');
ylabel(axes1,'Magnitude');
box(axes1,'on');
hold(axes1,'all');
plot1 = plot(linspace(0,0.5,P/2+1),20*log10(out1),linspace(0,0.5,P/2+1),20*log10(out2));
set(plot1(1),...
  'Color',[0.5 0.5 0],...
  'LineWidth',2);
set(plot1(2),'LineWidth',2,'Color',[0.6 0.2 0]);
legend1 = legend(axes1,{'With jitter','Without jitter'});
figure2 = figure(2);
axes2 = axes(...
  'XGrid','on',...
  'YGrid','on',...
  'Parent',figure2);
xlabel(axes2,'Sample');
ylabel(axes2,'Normalized voltage');
box(axes2,'on');
axis(axes2,[0 50 -1 1]);
hold(axes2,'all');
x2 = x(P:N);
plot1 = plot(...
  linspace(0,P,length(x2)),x2,...
  'Color',[1 0.4 0],...
  'LineWidth',2);
stairs1 = stairs(...
  linspace(0,P,P),y1,...
  'DisplayName','With jitter',...
  'Color',[0.5 0.5 0],...
  'LineWidth',2);
stairs2 = stairs(...
  linspace(0,P,P),y2,...
  'DisplayName','Without jitter',...
  'Color',[0.6 0.4 0],...
  'LineWidth',2);
legend1 = legend(axes2,{'Input signal','With jitter','Without jitter'});



function y=myfft(x,N);
w = hanning(N);
x = x - mean(x);
y = fft(times(x,w'));
y = 2 * y(1:N/2+1);
y = y * 2/N;
y = abs(y);
```

Appendix B

Introduction to Mathematics of Noise Sources

This is a compilation of different books [1,3,25] and their introduction to noise analysis of electronic circuits.

B.1 Noise

Noise is a phenomena that occurs in all electronic circuits. It places a lower limit on the smallest signal we can use. Many now have super audio compact disc (SACD) players with 24bit converters, 24 bits is around $2^{24} = 16.78$ Million different levels. If 5V is the maximum voltage, the minimum would have to be $\frac{5V}{2^{24}} \approx 298nV$. That level is roughly equivalent to the noise in a 50 Ohm resistor with a bandwith of 96kHz. There exist an equation that relates number of bits to signal to noise ratio [1], the equation specifies that $SNR = 6.02 * Bits + 1.76 = 146.24dB$. As of 12.2005 the best digital to analog converter (DAC) that Analog Devices (a very big semiconductor company) has is a DAC with 120dB SNR, that equals around $Bits = (120 - 1.76)/6.02 = 19.64$. In other words, the last four bits of your SACD player is probably noise!

B.2 Statistics

The mean of a signal x(t) is defined as

$$\overline{x(t)} = \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{+T/2} x(t)dt \tag{B.1}$$

The mean square of x(t) defined as

$$\overline{x^2(t)} = \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{+T/2} x^2(t)dt \tag{B.2}$$

The variance of x(t) defined as

$$\sigma^2 = \overline{x^2(t)} - \overline{x(t)}^2 \tag{B.3}$$

For a signals with a mean of zero the variance is equal to the mean square. The auto-correlation of x(t) is defined as

$$\begin{aligned} R_x(\tau) &= \overline{x(t)x(t+\tau)} \\ &= \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{+T/2} x(t)x(t+\tau)dt \end{aligned} \tag{B.4}$$

B.3 Average Power

Average power is defined for a continuous system as (B.5) and for discrete samples it can be defined as (B.6). P_{av} usually has the unit A^2 or V^2, so we have to multiply/devide by the impedance to get the power in Watts. To get Volts and Amperes we use the root-mean-square (RMS) value which is defined as $\sqrt{P_{av}}$.

$$P_{av} = \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{+T/2} x^2(t)dt \tag{B.5}$$

$$P_{av} = \frac{1}{N} \sum_{i=0}^{N} x^2(i) \tag{B.6}$$

If x(t) has a mean of zero then, according to (B.3), P_{av} is equal to the variance of x(t).

Many different notations are used to denote average power and RMS value of voltage or current, some of them are listed in Table B.1 and Table B.2. Notation can be a confusing thing, it changes from book to book and makes expressions look different. It is important to realize that it does not matter how you write average power and RMS value. If you want you can invent your own notation for average power and RMS value. However, if you are presenting your calculations to other people it is convenient if they understand what you have written. In the remainder of this paper we will use $\overline{e_n^2}$ for average power when we talk about voltage noise source and $\overline{i_n^2}$ for average power when we talk about current noise source. The n subscript is used to identify different sources and can be whatever.

Table B.1: Notations for average power

Voltage	Current
V_{rms}^2	I_{rms}^2
$\overline{V_n^2}$	$\overline{I_n^2}$
$\overline{v_n^2}$	$\overline{i_n^2}$

Table B.2: Notations for RMS

Voltage	Current
V_{rms}	I_{rms}
$\sqrt{\overline{V_n^2}}$	$\sqrt{\overline{I_n^2}}$
$\sqrt{\overline{v_n^2}}$	$\sqrt{\overline{i_n^2}}$

B.4 Noise Spectrum

With random noise it is useful to relate the average power to frequency. We call this Power Spectral Density (PSD). A PSD plots how much power a signal carries at each frequency. In literature $S_x(f)$ is often used to denote the PSD. In the same way that we use V^2 as unit of average power, the unit of the PSD is $\frac{V^2}{Hz}$ for voltage and $\frac{A^2}{Hz}$ current. The root spectral density is defined as $\sqrt{S_x(f)}$ and has unit $\frac{V}{\sqrt{Hz}}$ for voltage and $\frac{I}{\sqrt{Hz}}$ for current.

The power spectral density is defined as two times the Fourier transform of the auto-correlation function [3]

$$S_x(f) = 2\int_{-\infty}^{\infty} R_x(\tau)e^{-j2\pi f\tau}d\tau \tag{B.7}$$

This can also be written as

$$\begin{aligned} S_x(f) &= 2\left[\int_{-\infty}^{\infty} R_x(\tau)\cos(\omega\tau)d\tau - \int_{-\infty}^{\infty} R_x(\tau)j\sin(\omega\tau)d\tau\right] \\ &= 2\left[\int_{-\infty}^{0} R_x(\tau)\cos(\omega\tau)d\tau + \int_{0}^{\infty} R_x(\tau)\cos(\omega\tau)d\tau\right] \\ &- 2j\left[\int_{-\infty}^{0} R_x(\tau)\sin(\omega\tau)d\tau + \int_{0}^{\infty} R_x(\tau)\sin(\omega\tau)d\tau\right] \\ &= 4\int_{0}^{\infty} R_x(\tau)\cos(\omega\tau)d\tau \\ &- 2j\left[-\int_{0}^{\infty} R_x(\tau)\sin(\omega\tau)d\tau + \int_{0}^{\infty} R_x(\tau)\sin(\omega\tau)d\tau\right] \\ &= 4\int_{0}^{\infty} R_x(\tau)\cos(\omega\tau)d\tau \end{aligned} \tag{B.8}$$

, since $e^{-j\omega\tau} = \cos(\omega\tau) - j\sin(\omega\tau)$, $R_x(\tau)$ and $\cos(\omega\tau)$ are symmetric around $\tau = 0$ while $\sin(\omega\tau)$ is asymmetric around $\tau = 0$.

The inverse of power spectral density is defined as

$$R_x(\tau) = \frac{1}{2}\int_{-\infty}^{\infty} S_x(f)e^{j2\pi f\tau}df = \int_{0}^{\infty} S_x(f)\cos(\omega\tau)df \tag{B.9}$$

If we set $\tau = 0$ we get

$$\overline{x^2(t)} = \int_{0}^{\infty} S_x(f)df \tag{B.10}$$

which means we can easily calculate the average power if we know the power spectral density. As we will see later it is common to express noise sources in PSD form.

Another very useful theorem when working with noise in the frequency domain is this

$$S_y(f) = S_x(f)|H(f)|^2 \tag{B.11}$$

, where $S_y(f)$ is the output power spectral density, $S_x(f)$ is the input power spectral density and $H(f)$ is the transfer function of a time-invariant linear system.

If we insert (B.11) into (B.10), with $S_x(f) = a\ constant = D_v$ we get

$$\overline{x^2(t)} = \int S_y(f)df = D_v\int |H(f)|^2df = D_v f_x \tag{B.12}$$

, where f_x is what we call the noise bandwidth. For a single time constant RC network the noise bandwidth is equal to

$$f_x = \frac{\pi f_0}{2} = \frac{1}{4RC} \tag{B.13}$$

where f_x is the noise bandwidth and f_0 is the 3dB frequency.

We haven't told you this yet, but thermal noise is white and white means that the power spectral density is flat (constant over all frequencies). If $S_x(f)$ is our thermal noise source and $H(f)$ is a standard low pass filter, then equation (B.11) tells us that the output spectral density will be shaped by $H(f)$. At frequencies above the f_x in $H(f)$ we expect the root power spectral density to fall by 20dB per decade.

B.5 Probability Distribution

Theorem 1 (Central limit theorem) *The sum of n independent random variables subjected to the same distribution will always approach a normal distribution curve as n increases.*

This is a neat theorem, it explains why many noise sources we encounter in the real world are white.[1] Take thermal noise for example, it is generated by random motion of carriers in materials. If we look at a single electron moving through the material the probability distribution might not be Gaussian. But summing probability distribution of the random movments with a large number of electrons will give us a Gaussian distribution, thus thermal noise is white.

B.6 PSD of a white noise source

If we have a true random process with Gaussian distribution we know that the autocorrelation function only has a value for $\tau = 0$. From equation (B.4) we have that

$$\begin{aligned} R_x(\tau) &= \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{+T/2} x(t)x(t-\tau)dt \\ &= \left[\lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{+T/2} x^2(t)dt \right] \delta(\tau) \\ &= \overline{x^2(t)}\delta(\tau) \end{aligned} \tag{B.14}$$

The reason being that in a true random process $x(t)$ is uncorrelated with $x(t+\tau)$ where τ is an integer. If we use equation (B.7) we see that

$$\begin{aligned} S_x(f) &= 2 \int_{-\infty}^{\infty} \overline{x^2(t)}\delta(\tau)e^{-j2\pi f\tau}d\tau \\ &= 2\overline{x^2(t)} \int_{-\infty}^{\infty} \delta(\tau)e^{-j2\pi f\tau}d\tau \\ &= 2\overline{x^2(t)} \end{aligned} \tag{B.15}$$

, since

$$\int \delta(\tau)e^{-j2\pi f\tau}d\tau = e^0 = 1 \tag{B.16}$$

This means that the power spectral density of a white noise source is flat, or in other words, the same for all frequencies.

B.7 Summing noise sources

Summing noise sources is usually trivial, but we need to know why and when it is not. We if we write the time dependant noise signals as

$$v_{tot}^2(t) = (v_1(t) + v_2(t))^2 = v_1^2(t) + 2v_1(t)v_2(t) + v_2^2(t) \tag{B.17}$$

[1] Gaussian distribution = normal distribution. Noise sources with Gaussian distribution are called white

The average power is defined as

$$\begin{aligned}\overline{e_{tot}^2} &= \lim_{T\to\infty}\frac{1}{T}\int_{-T/2}^{+T/2} v_{tot}^2(t)dt \\ &= \lim_{T\to\infty}\frac{1}{T}\int_{-T/2}^{+T/2} v_1^2(t)dt \\ &+ \lim_{T\to\infty}\frac{1}{T}\int_{-T/2}^{+T/2} v_2^2(t)dt \\ &+ \lim_{T\to\infty}\frac{1}{T}\int_{-T/2}^{+T/2} 2v_1(t)v_2(t)dt \\ &= \overline{e_1^2} + \overline{e_2^2} + \lim_{T\to\infty}\frac{1}{T}\int_{-T/2}^{+T/2} 2v_1(t)v_2(t)dt \end{aligned} \tag{B.18}$$

If $\overline{e_1^2}$ and $\overline{e_2^2}$ are uncorrelated noise sources we can skip the last term in (B.18) and just write

$$\overline{e_{tot}^2} = \overline{e_1^2} + \overline{e_2^2} \tag{B.19}$$

Most natural noise sources are uncorrelated.

B.8 Signal to Noise Ratios

Signal to Noise Ratio (SNR) is a common method to specify the relation between signal power and noise power in linear systems. It is defined as

$$\begin{aligned} SNR &= 10\log\left(\frac{Signal\ power}{Noise\ power}\right) \\ &= 10\log\left(\frac{\overline{v_{sig}^2}}{\overline{e_n^2}}\right) \\ &= 20\log\left(\frac{v_{rms}}{\sqrt{\overline{e_n^2}}}\right) \end{aligned} \tag{B.20}$$

Another useful ratio is Signal to Noise and Distortion (SNDR), since most real systems exibit non-linearities it is useful to include distortion in the ratio. One can calculate SNR and SNDR in many ways. If we don't know the expression for $\overline{e_n^2}$ we can do a FFT of our output signal. From this FFT we sum spectral components except at the signal frequency to get noise and distortion. SNR is normally calculated as

$$SNR = 10\log\left(\frac{Signal\ power}{Noise\ power\ -\ 6\ first\ harmonics}\right) \tag{B.21}$$

And SNDR is calculated as

$$SNDR = 10\log\left(\frac{Signal\ power}{Noise\ power}\right) \tag{B.22}$$

B.9 Noise figure and Friis formula

Noise factor is a measure on the noise performance of a system. It is defined as

$$F = \frac{\overline{v_o^2}}{source\ contribution\ to\ \overline{v_o^2}} \tag{B.23}$$

where $\overline{v_o^2}$ is the total output noise.

The noise figure is defined as (noise factor in dB)

$$NF = 10\log(F) \tag{B.24}$$

The noise factor can also be defined as

$$F = \frac{SNR_{input}}{SNR_{output}} \tag{B.25}$$

This brings us right into what is known as Friis formula. If we have a multistage system, for example several amplifiers in cascade, the total noise figure of the system is defined as

$$F = 1 + F_1 - 1 + \frac{F_2 - 1}{G_1} + \frac{F_3 - 1}{G_1 G_2} + ... \tag{B.26}$$

Here F_i is the noise figures of the individual stages and G_i is the available gain of each stage. This can be rewritten as

$$F = F_1 + \sum_{i=1}^{N} \frac{F_{i+1} - 1}{\prod_{k=1}^{i-1} G_i} \tag{B.27}$$

Friiss formula tells us that it is the noise in the first stage that is the most important if G_1 is large. We could say that in a system it is important to amplify the noise as early as possible!

B.10 Conclusion

We have looked at the properties of noise in time domain and frequency domain. The equations in this paper are useful tools when dealing with noise sources.