ANALOG-TO-DIGITAL CONVERTERS FOR HIGH-SPEED LINKS

A DISSERTATION

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING

AND THE COMMITTEE ON GRADUATE STUDIES

OF STANFORD UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

Valentin Abramzon

2008

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

_____

(Mark Horowitz) Principal Adviser

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

_____

(Boris Murmann)

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

_____

(Thomas H. Lee)

Approved for the University Committee on Graduate Studies

_____

ABSTRACT

In today's technology, high-speed links play an important role, enabling faster, cheaper, and more reliable data communications. Data converters, present in some form in almost all modern high-speed links, are a key to performing equalization - the process of compensating bandwidth limitations of the communication channel. In particular, baud-rate ADCs at the receiver front ends can enable easily-scalable digital implementations of various equalization schemes, such as feed-forward equalization (FFE), decision-feedback equalization (DFE), and even maximum-likelihood sequence estimation (MLSE). However, power limitations of on-chip high-speed link receivers make front-end ADC design very challenging. Therefore, in this work we pay special attention to power efficiency of the ADCs and not only to their performance. This leads us to the idea of heavily interleaving very simple and efficient ADCs to obtain high aggregate conversion rates.

We choose a single-slope ADC as a candidate for interleaving because of its simplicity, linearity, low-power operation, small area, and small input capacitance. This choice is nevertheless unusual because of single-slope's reputation for long conversion time, normally taking 2^{Nbits} time steps, where N_{bits} is the ADC resolution. However, because PLLs and/or DLLs in high-speed links normally generate very fine time steps, the conversion rates of single-slope ADCs for relatively low resolution can be pushed to Gsps range. We demonstrate the suitability of single-slope ADCs for high-speed low-power operation with a proof-of-concept design in the high-speed 45nm TI CMOS technology. In simulation, the ADC was capable of 4.5bit 1.6Gsps or 5.5bit 0.8Gsps operation while consuming 3mW of power from a 1V supply.

The prototype was, however, fabricated without redesign in a different, low-leakage, variant of 45nm technology. Due to differences in device characteristics, the chip operated at only 800MHz in a 4.5-bit mode and at 400MHz in a 5.5-bit mode while consuming 4mW from a 1.2V supply. Nevertheless, it lays groundwork for simple high-speed low-power ADCs based on a single-slope architecture.

ACKNOWLEDGMENTS

First and foremost, I would like to thank my adviser, Professor Mark Horowitz, for without his signature this document would have never seen the light. On a more serious note, Mark has been a great mentor for the past ... I cannot remember how many years. His technical advice was of huge value to me, and, probably just as importantly, his kind encouragement helped me get through many difficult times. Professor Boris Murmann, my co-adviser, has offered his deep expertise in the area of data converters, and also gave me great support during my time at Stanford. I am deeply thankful to Professor Thomas Lee for his great lectures on analog IC design, for his amazingly broad knowledge that he was always willing to share, and for his enthusiasm and optimism. I would like to thank Professor Fabian Pease and his group, as their e-beam ADC project gave me inspiration to work on high-speed ADCs.

I would like to thank Andy Joy, Ajith Amerasekera, Peter Hunt, Tom Leslie, and everyone at TI UK group for giving me the privilege of working with them, for supporting this project, and for making the single-slope ADC test chip possible. I would like to thank TI for fabricating the chip.

I would like to thank our wonderful administrative assistants, Teresa Lynn, Ann Guerra, and June Wang for all their help. I would also like to thank Electrical Engineering department for offering me a graduate fellowship that allowed me to come to Stanford in the first place.

I also want to take this opportunity to let my friends and co-workers know how much I appreciate their company and technical advice. Bita Nezamfar, Parastoo Nikaeen, Mona Jarrahi, Xiling Shen, Elad Alon, Amir Amirkhany, Dinesh

Patil, Mohammad Hekmat, Alireza Dastgheib, Farshid Moussavi, Alex Solomatnikov, Amin Firoozshahian, Pedram Lajevardi, Ilya Katsnelson, and many other friends – you are important to me.

Finally, my family – you are the most important thing in my life. Without you, all that Ph. D. research, no matter how interesting it is, would be meaningless.

The words of gratitude in my dissertation may not say much and may not be as eloquent as I would like them to be, but I really hope that even without these words, all of you feel very much appreciated. You truly are.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1. INTRODUCTION

Most modern digital systems consist of multiple integrated circuits (ICs), which need to communicate with each other. As the processing speed of each IC increases, it demands higher and higher input/output (I/O) bandwidth. The term **high-speed link** refers to both the physical channel and the I/O circuits that aim to support this ever-increasing need for bandwidth. To keep up, high-speed links are forced to both employ more parallel channels and increase the data rate in each channel.

Although communicating digital 0s and 1s, especially over wires, may seem trivial, at high frequencies, it becomes more complex. The resistive and dielectric losses in wires increase at high frequencies, resulting in distorted digital signals and presenting constant new challenges to link design as data rates increase. Significant effort goes into developing better communication channels, ranging from more advanced printed circuit boards, IC packages, and connectors to optical, capacitive, inductive, and radio-frequency (RF) interconnects. Meanwhile, most high-speed links today have to resort to multiple signal processing techniques to overcome the bandwidth limitations of existing channels.

Basic signal processing tasks, such as continuous-time equalization, finite-impulse response filtering, and decision-feedback equalization can be efficiently performed with analog circuits. However, as the complexity of filters increases to compensate for channel losses at higher and higher frequencies, exploiting the benefits of digital scaling by moving signal processing to digital domain becomes an interesting alternative. To accomplish this in a link receiver, an analog signal from the channel needs to be digitized first, requiring a high-speed analog-to-digital converter (ADC).

It is possible to use conventional ADC architectures, such as FLASH or interleaved pipeline or successive-approximation ADCs, for the link receiver front-ends. Unfortunately, neither of these ADCs is very efficient for the range of resolution and conversion rates typically needed for high-end link receivers. Therefore, in this work, we attempt to find a different ADC architecture that would directly exploit the specificity of a high-speed link environment to provide much better power- and area-efficiency. In particular, the availability of precise and very fine timing references present in high-speed links hints at better exploiting the time dimension. To this end, we propose an interleaved single-slope ADC that can be thought of as a time-domain equivalent of FLASH, only with smaller input capacitance, good intrinsic linearity, and low power consumption. With this example, we hope to demonstrate that the advantages of developing novel circuits, possibly very unconventional, but with their architecture driven by the specific needs of the overall system, can far outweigh additional difficulties and risks associated with the development.

1.1. ORGANIZATION

To understand the application space for ADCs, we first review the basic concepts of high-speed link systems in Chapter 2. In particular, we focus on equalizers – circuits that enable high-speed data transmission through a band-limited channel by compensating its non-uniform frequency response. Although equalizers have been traditionally implemented as analog or semi-digital circuits, digital scaling has motivated the exploration of fully-digital equalizers. However, to interface digital equalizers in a link receiver with an analog channel, a baud-rate analog-to-digital converter (ADC) is needed.

In Chapter 3 and Chapter 4, we address the task of designing ADCs for high-speed link receivers. In Chapter 3, we outline the strengths and limitations of conventional high-speed ADC architectures, such as FLASH and interleaved

successive-approximation and pipeline ADCs. We show that neither of these architectures completely addresses the needs of a front-end ADC in high-speed link receivers. Therefore, in Chapter 4, we propose an interleaved single-slope ADC topology that, for several reasons, promises to become a more suitable choice for this application. We focus on the development of an individual high-speed single-slope ADC and describe a test chip designed in a 45nm TI CMOS technology to verify this rather unusual concept.

In Chapter 5, we summarize our work and outline opportunities for future research in the area.

CHAPTER 2. HIGH-SPEED LINK SYSTEMS: AN OVERVIEW

A typical high-speed link system consists of 3 basic components: a serializing transmitter, a communication channel, and a deserializing receiver, as shown on the block diagram in Figure 1.

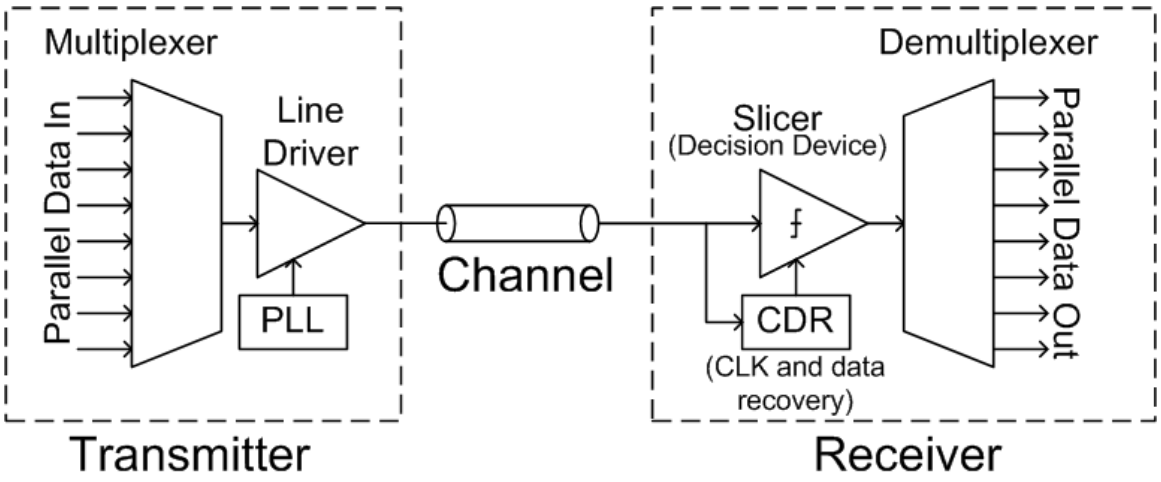


Figure 1. Block diagram of a high-speed link

The transmitter usually includes a multiplexer that converts parallel data into serial stream, a line driver that send data bits through the physical channel, and a clock source, usually implemented as a phase-locked loop (PLL).

The receiver decides which discrete digital value has most likely been transmitted. The decision device is usually referred to as **slicer** or **comparator**. Then the serial data stream is usually demultiplexed into parallel word, more naturally suited for data processing. If no explicit clock signal is sent along with the data, a

clock and data recovery (**CDR**) block is needed to derive the timing information from data transitions.

The communication channel, generally speaking, is the medium through which information is transmitted. In this work, we focus on chip-to-chip interconnects that normally use copper traces on a printed circuit board (PCB) as communication channels. The chips are often located on different PCBs, linked with each other through connectors. One very important example of such setup is an Internet router, where multiple daughter boards (**switch card** and **line cards**) are connected with each other through a long motherboard (**backplane**), as shown in Figure 2. Line cards accept external network connections, while the switch card provides the routing between these connections. High-speed links are necessary for high-throughput data exchanges between these line cards and the switch card.

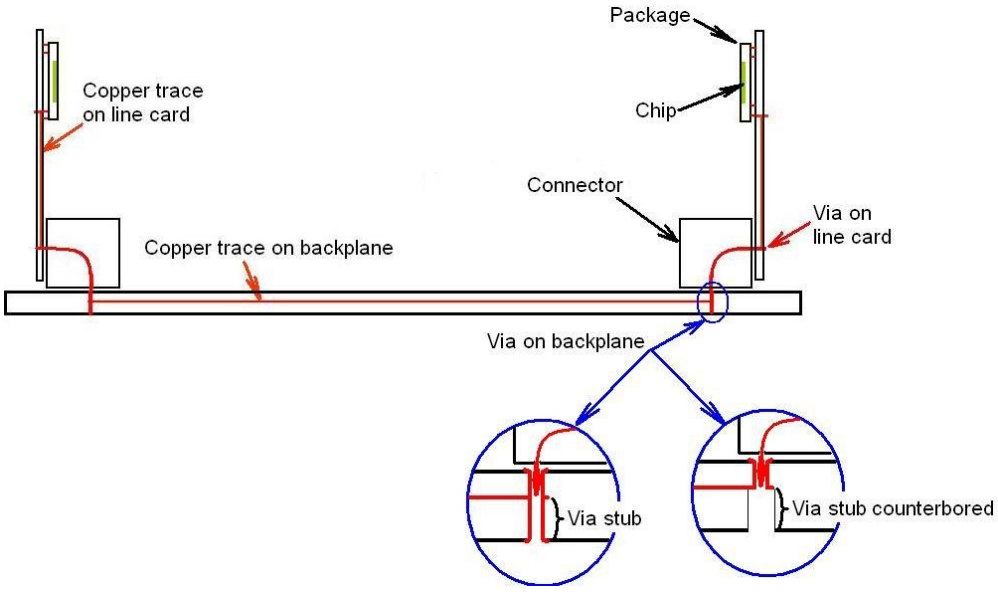


Figure 2. Signal path and channel components in a typical backplane system

Most issues in the design of high-speed links stem from the fact that, over a sufficiently large range of frequencies, even such relatively short wireline

channel presents a highly non-uniform frequency response. Apart from parasitic RLC elements in a chip package itself, the non-uniform frequency response is due primarily to three effects: skin effect, dielectric loss, and reflections. **Skin-effect** is the tendency of an AC current to crowd toward the surface of the conductor due to eddy currents. Since higher-frequency currents flow through a smaller cross-section than the lower-frequency currents, the effective resistance increases with frequency.

Dielectric loss is electric energy lost to heat due to movement or rotation of the atoms or molecules in a dielectric subjected to a varying electric field. Dielectric loss increases linearly with frequency, with a coefficient of proportionality depending on the dielectric material. The property of a material that specifies dielectric loss is called **loss tangent, tan δ**. The type of material most commonly used for making PCBs is FR4 (abbreviation for flame-retardant 4), which has a relatively poor (high) loss tangent. Although better materials, such as NELCO and Rogers, recently became available, FR4 still dominates the high-speed PCB scene, both because it is inexpensive, and because it is used in most legacy backplanes.

Reflections occur whenever a signal encounters a discontinuity in the channel. Apart from further attenuating a signal, multiple back-and-forth reflections produce resonances at certain frequencies (large dips in a frequency response). There are several potential sources of discontinuity: vias (metal-plated holes that connect layers of the board), board-to-board connectors, chip packages, and imperfect terminations. Vias produce especially unpleasant resonances if they have a "dead-end" - via stub, shown in zoomed-in portion of Figure 2. Although a via stub can be partially removed by counterboring, it adds to the price of the PCB. Similarly, the other sources of discontinuities can be minimized, but at an expense of redesign and price increase of the overall system.

This has forced link designers to advance signaling techniques that work even with legacy backplanes, whose representative frequency responses are

shown in Figure 3. The general high-frequency roll-off is due to skin effect and dielectric loss. Note that the roll-off is steeper for longer backplane traces. The dips in the frequency response are most likely caused by various discontinuities, with the largest dips caused by the via stub.

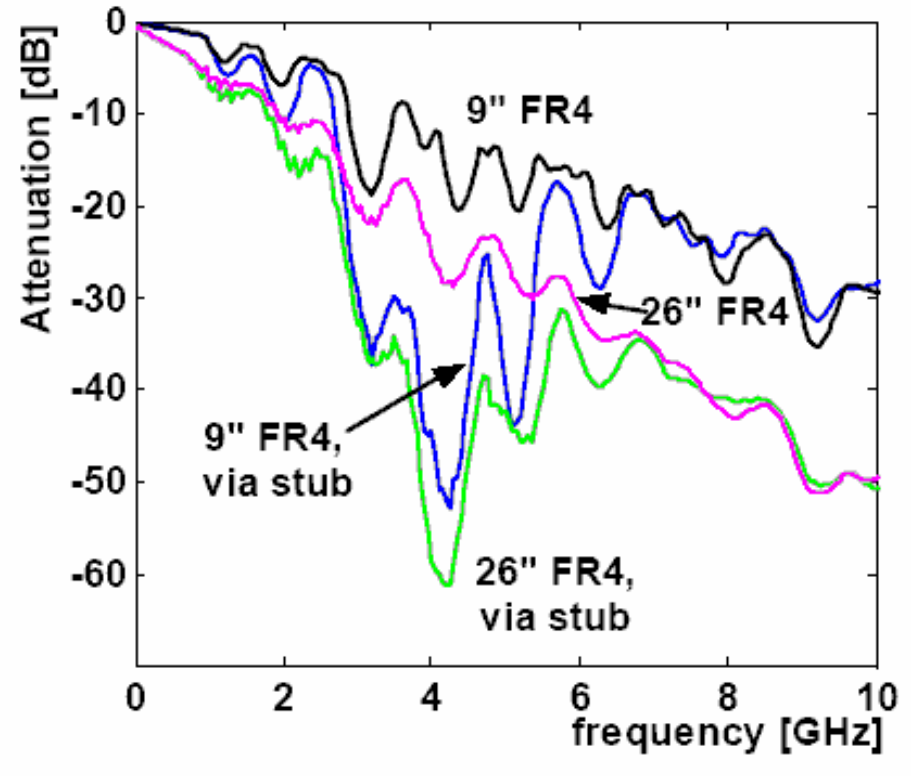


Figure 3. Frequency responses of several realistic backplane channels **[15]**

An alternative convenient way to describe a channel is by its **pulse response** in time domain. A pulse response of the channel is its output resulting from an isolated single-bit pulse (…0001000…) at its input. A general low-pass nature of most channels suggests that they cannot instantaneously respond to infinitely sharp edges of a pulse, delaying and dispersing it, as shown in Figure 4. When such pulse response is sampled at bit times, the largest sample is called the **cursor**; the samples before the cursor are called **pre-cursors**, and the samples after the cursors are called

post-cursors. Pre-cursors interfere with previously sent bits, while post-cursors interfere with the following bits. To cancel such **inter-symbol interference**, most modern links use **equalization**. An equalizer provides an inverse channel response such that the overall frequency response is flat over the bandwidth of interest.

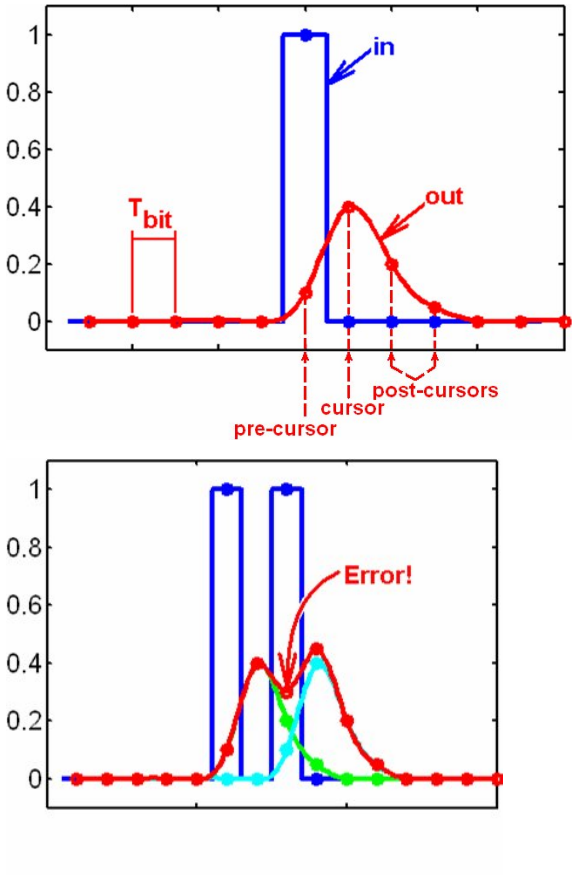


Figure 4. Pulse dispersion and inter-symbol interference

Equalization is an important application space for digital-to-analog and analog-to-digital converters. Equalizer implementations fall into 3 general categories: analog, semi-digital, and digital. As we shall see, this classification is somewhat oversimplified, because most "analog" equalizers incorporate some digital adjustment, while all "digital" equalizers require ADCs and DACs to interface digital processors

with analog channels. In fact, most equalizers operate on both analog and digital signals, so ADCs and DACs are needed to perform the conversion between the two domains. It is mainly equalizers that drive the development of data converters in high-speed links. To understand how ADCs can be optimized for this purpose, let us describe a few most common equalization techniques used in high-speed links.

2.1. LINEAR EQUALIZATION

The simplest equalizer type is a linear filter that provides a high-frequency boost to counteract the channel attenuation, as shown in Figure 5. Such a **linear equalizer** can be placed on the transmitter (**transmit pre-emphasis**) or receiver side, can be implemented in continuous or sampled time, and in analog or digital voltage domains. Table 1 summarizes some common choices.

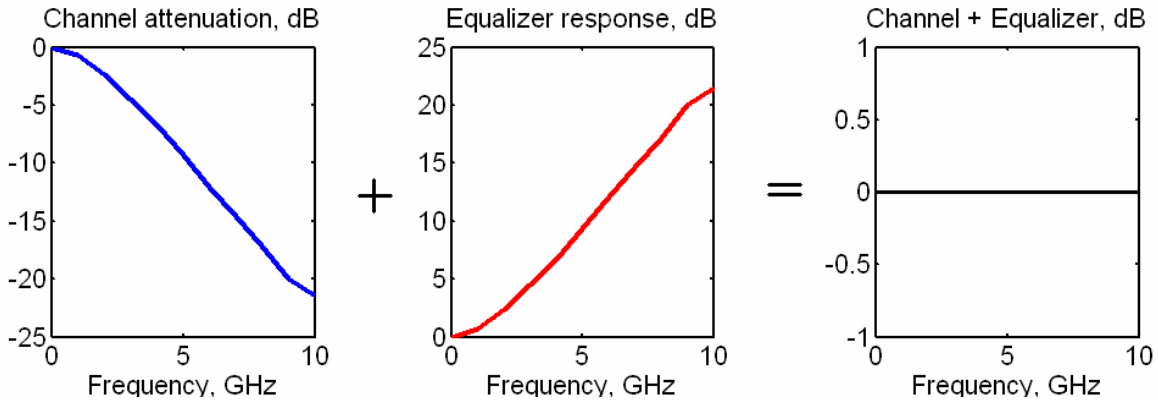


Figure 5. Linear equalization: frequency domain

	Transmit side • channel response unknown (need back-channel for adaptation) • digital data known		Receive side • channel response can be measured • digital data unknown	
	Analog voltage	Digital voltage	Analog voltage	Digital voltage
Continuous time			Analog filter with 1 or more diff. pairs	
Discrete time	Filter in digital domain (usually 1 or more LUTs) + DAC	Semi-digital FIR (finite impulse response) filter with digital delay elements, taps set by DACs, and addition in analog domain	Semi-digital filter with analog delay elements, taps set by DACs, and addition in analog domain	ADC + digital filter

Table 1. Summary of common linear equalizer options

In continuous time and voltage domains, high-speed equalizers are typically implemented using differential pairs with adjustable frequency response **[3]-[12]**. In the simplest case **[3]**, a source-degenerated differential amplifier with degeneration resistor shunted out at high frequencies provides a single-zero (+6dB/decade) boost at these frequencies, as shown in Figure 6. A more sophisticated design in **[9]** replaces a simple differential pair with a Cherry-Hooper amplifier for higher bandwidth. More degrees of freedom in terms of shaping of frequency response can be realized with multiple filter elements, either in series or in parallel.

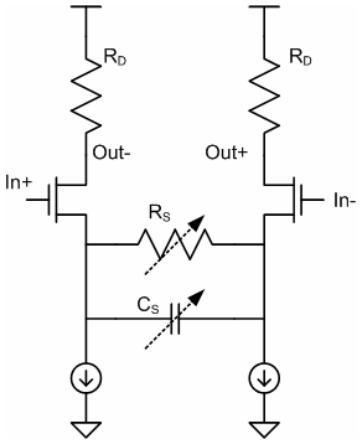


Figure 6. Zero-peaked equalizer

An equalizer is adapted to the characteristic of the channel either manually or, more often, automatically. The adjustment is carried out by changing R and/or C, often by means of simple digital-to-analog converters (DACs).

But the equalizers that make heavy use of data converters are the discrete-time equalizers. Figure 7 illustrates such topologies. Out of these topologies, the transmit FIR equalizer, shown in Figure 7(b), is most common, although it requires communications from the receiver (that can estimate channel characteristics) back to the transmitter for adapting the filter to the actual channel. Placing a FIR equalizer at the receiver does not require a back-channel for adaptation, but uses an analog delay

line that is much harder to implement than its digital equivalent. Finally, digital filters either at the transmitter or the receiver require a high-speed DAC or ADC as well digital processing hardware. Although, as we show later, these approaches might be advantageous for a large number of filter taps, they are less efficient than a semi-digital FIR filter when just a few taps are needed.

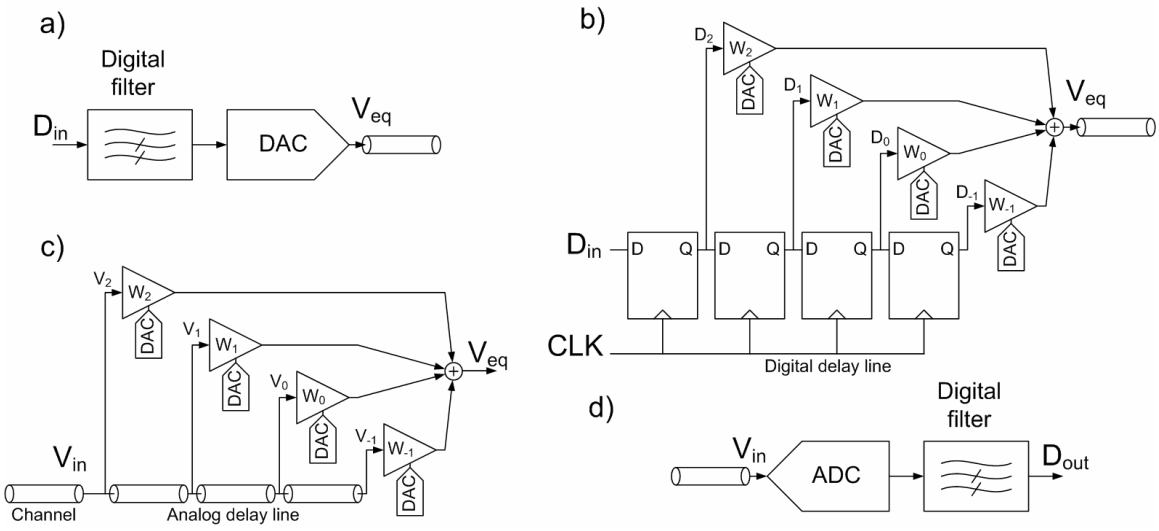


Figure 7. Discrete-time equalizers: (a) Digital transmit equalizer, (b) semi-digital FIR transmit equalizer, (c) semi-digital FIR receive equalizer, and (d) digital receive equalizer

2.2. DECISION-FEEDBACK EQUALIZATION

All linear equalizers suffer from another problem. Ideally, an equalizer placed at the transmitter would boost a signal at high frequencies leaving low-frequency content intact. However, due to limited voltage swing of the channel driver, the filter is forced to attenuate lower frequencies rather than amplify higher frequencies, reducing the total signal energy and, thus, the signal-to-noise ratio (SNR). On the other hand, an equalizer placed at the receiver amplifies high-frequency noise along with the signal, again degrading the SNR.

To circumvent such noise amplification problem, **non-linear** equalizers can be used. The simplest, efficient, and thus most common non-linear equalizer used in high-speed link receivers is a **decision-feedback equalizer (DFE)**, depicted in Figure 8 for a 4-tap example. Assuming that the 4 previous bits were detected correctly, DFE subtracts their influence from the currently received analog input voltage, thus cancelling their post-cursor ISI. Note that DFE cannot cancel pre-cursor ISI, because the cursor bit has not been sliced yet.

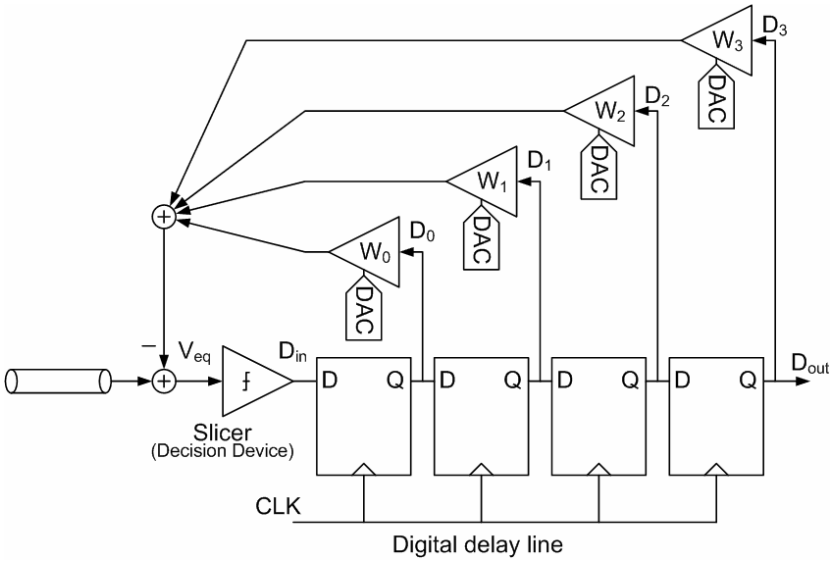


Figure 8. Decision-feedback equalizer on the receiver side

Nevertheless, DFE is very useful, because due to its non-linear nature (it contains a comparator), it does not amplify high-frequency noise from the channel. It is susceptible to error propagation, however, meaning that if a bit has been incorrectly detected, its ISI will not be corrected properly, leading to error propagating to the current bit. While this may cause chains of errors in low-SNR channels (like wireless), such error propagation is much less probable in higher-reliability short-reach copper links.

2.3. SEMI-DIGITAL FIR FILTER IMPLEMENTATION

The semi-digital linear equalizers as well as the DFE depend on the semi-digital FIR filter, shown in Figure 9. A shift register can be thought of as a window, running along the digital data stream, and supplying data to the filter. Each data bit is multiplied by its corresponding weight W_i, and the multiplication results for all the taps are added up in a summing node. Multiplication by a constant and addition can be efficiently implemented in analog domain using differential current-mode circuits. Multiplication $W_i \cdot D_i$ is carried out by a differential pair, whose tail current is set to W_i, normally using a current-mode DAC. Since the multiplication results are currents, they can be summed by simply connecting them together and dropping the resulting current across a resistor **[2]**.

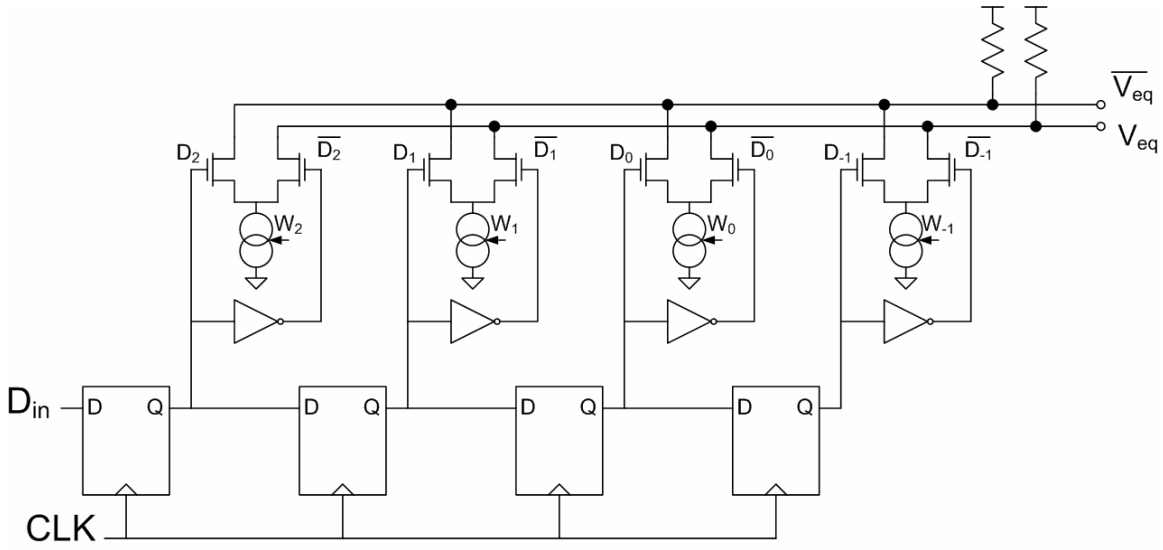


Figure 9. Circuit implementation of a semi-digital FIR filter

Two problems arise as number of taps increases (due to increasing signaling rate through the same channel, for example). First, overall quantization noise budget splits between an increasing number of taps, requiring higher and higher resolution of each weight-setting DAC. Second, the current summing nodes become excessively loaded by the increasing number of differential pairs that implement

individual taps. These two problems motivate migrating filters into the digital domain with signal-path ADCs and DACs inserted at the interface between the digital filter and an analog channel.

2.4. DIGITAL EQUALIZER IMPLEMENTATIONS

Implementing a filter in digital domain allows one to compute filtering results with higher resolution in digital domain and then round them off only once to the closest data converter level. This is in contrast with semi-digital implementation, where the value of each tap is rounded off, leading to accumulation of rounding errors. This problem is illustrated in Figure 10. Only three taps are shown for simplicity, but one can imagine the problem only getting worse for more taps.

To understand this issue more fully, we can view the operation of the filter as a function of data bits as moving along a tree. The "root" of the tree is output for no filter taps. For one single tap, depending whether the previous data bit was 0 or 1, the filter output branches to either ($-W_0$) or W_0. If the filter has two taps, depending on whether data bits were 00, 01, 10, or 11, there are four possibilities for the filter output: ($-W_1-W_0$), ($-W_1+W_0$), ($+W_1-W_0$), and ($+W_1+W_0$). For 3 taps, there are 8 possible outputs, and for N_{taps} taps $-2^{N_{taps}}$. Figure 10(a) shows the tree without any quantization. Figure 10(b) illustrates how quantization of each individual tap weight leads to error accumulation. Finally, Figure 10(c) shows that computing filter output with higher precision in digital domain and then rounding solves the error accumulation problem.

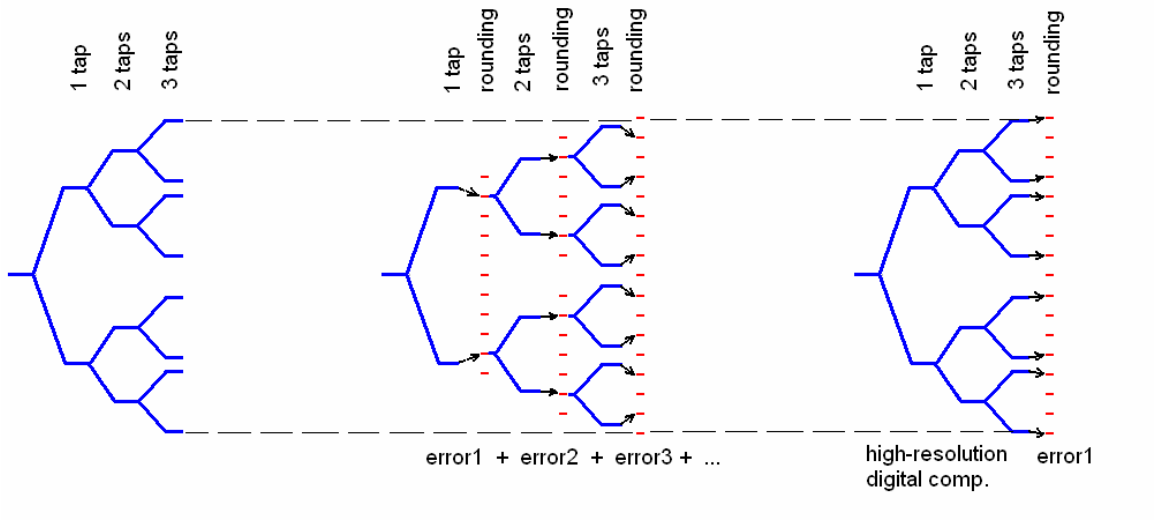


Figure 10. FIR filter output as a function of input bits (a) without any quantization, (b) with each tap weight W_i quantized separately, and (c) with filter output computed digitally with high resolution and then rounded off to the closest available data converter level

Unfortunately, implementing a digital filter at high speed and with high resolution is also problematic. Thus, instead of performing digital computations on-the-fly, look-up tables (LUTs) can be used, as demonstrated by B. Casper, *et. al.* in **[10]** for transmit equalization and by M. Harwood, *et. al.* in **[13]** for receive equalization. Both equalizers are shown in Figure 11.

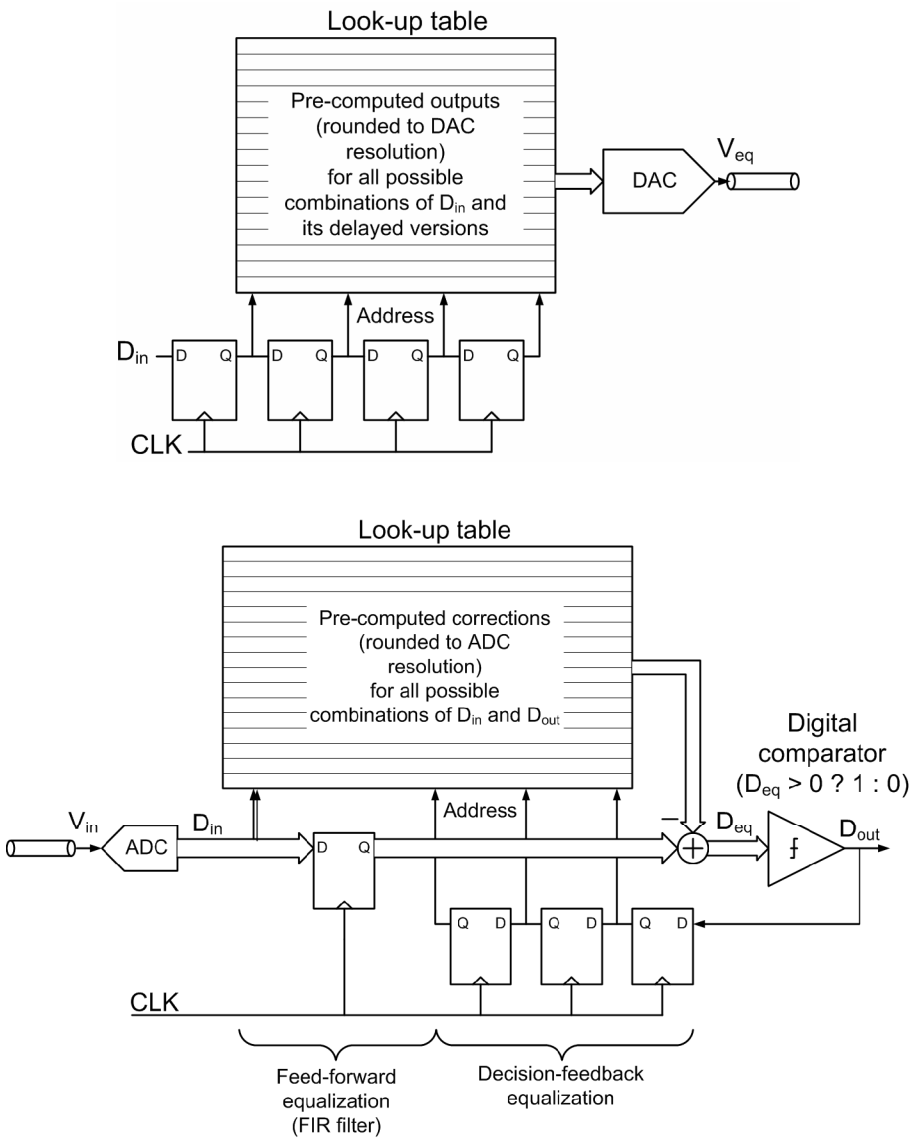


Figure 11. Digital filters using look-up tables (a) for transmit equalization, and (b) for receive equalization

Note that using an ADC at the receiver front-end permits the use of digital delay elements (flip-flops) to implement a FIR filter. Although only one-tap FIR filter is implied in Figure 11(b), more taps can be included at an expense of increased complexity. Typically, however, only one-tap FIR filter is needed to cancel pre-cursor ISI, while all post-cursor ISI can be more efficiently cancelled with DFE.

2.5. ADC FOR RECEIVER EQUALIZATION – SYSTEM REQUIREMENTS

Apart from digital filters, the digital transmit equalizer requires a baud-rate DAC, while the digital receive equalizer requires a baud-rate ADC. As baud rates of modern links extend well into Gbps range, both DACs and ADCs operating at such rates are large research topics in themselves. In this work, we focus only on designing high-speed ADCs for the link receivers. The complexity, and even feasibility, of this task as well as the choice of the ADC topology depend on system requirements for the ADC. While the baud rate of the link determines the sampling rate of the ADC, its resolution is defined by two considerations.

First, the LSB size must be small enough so that it does not noticeably degrade the noise margin of the link, defined by half a signal swing minus all bounded error sources. Equalization normally removes the precursor and postcursor ISI, leaving only the main cursor as a signal to be detected and reducing the useful signal swing by the sum of magnitudes of all pre- and post-cursors. On top of this attenuation, the bounded error sources include unequalized ISI, reflections, and crosstalk. The ADC quantization error becomes an additional bounded error source, and thus should be kept to a small fraction α of the main cursor. The value of α should be typically less than 10-20%.

Second, the ADC range must cover the entire swing of the receiver input, including the worst-case effects from the pre- and post-cursor ISI (unless they have been cancelled by the transmit equalizer). The worst-case signal swing is equal to the sum of magnitudes of all taps in a pulse response (pre- and post-cursors plus the cursor itself).

Given these two considerations, the number of ADC levels ($N_{levels} = 2^{N_{bits}}$) should be approximately

$$N_{levels} = 2^{N_{bits}} = \frac{\sum_{i=-\infty}^{+\infty} |p_i|}{\alpha \cdot p_0}, \tag{2.1}$$

where p_i is the i-th tap in a channel pulse response, with $i < 0$ for pre-cursors, $i = 0$ for the main cursor, and $i > 0$ for post-cursors. α is the fraction of equalized signal amplitude allocated to quantization error of the ADC. Taking α = 10% and pulse response shown in Figure 4 as an example, we obtain $N_{levels} \approx 20$ and $N_{bits} \approx 4.3$. As it turns out, for most typical backplane channels, N_{bits} of 4-5 bits is required. Thus, for this work, we will target a resolution of 4-5 bits.

The most important effect that limits ADC resolution at high frequencies is sampling jitter. To evaluate the effect of sampling jitter on general-purpose ADCs, it is usually translated into voltage noise assuming that an input sinusoidal signal is sampled at Nyquist rate at the point of its maximum slope, as shown in Figure 12(a). In this case, $\sigma_{\Delta V} \approx A\omega \cdot \sigma_{\Delta t}$, where $\sigma_{\Delta V}$ is RMS voltage error due to RMS clock jitter $\sigma_{\Delta t}$. However, the assumption of sinusoidal signal hardly describes an input to the link receiver. Instead, the receiver is normally designed to sample the data eye at its maximum opening, as shown in Figure 12(b). Because the input signal stays relatively flat during that time, the effect of sampling jitter is very small.

But this model may also be inaccurate when most of equalization is performed at the receiver, after ADC samples the input. In this case, the ADC samples an un-equalized (or under-equalized, if some equalization has been performed at the transmitter) data stream, as shown in Figure 12(c). Because equalization is incomplete, some samples take place on the slope rather than on the flat region of the eye.

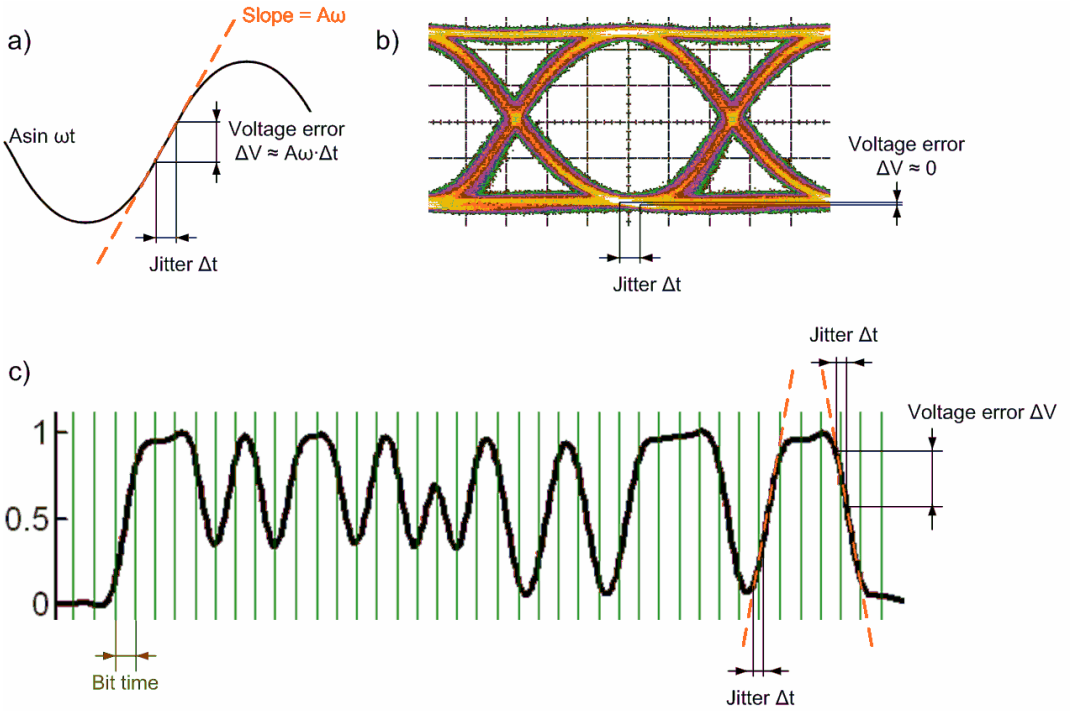


Figure 12. Result of sampling jitter: a) worst-case scenario when sinusoidal signal is sampled at Nyquist rate at zero-crossings, b) best-case scenario when a link receiver samples a signal in the middle of the equalized data eye, and c) realistic case when a link receiver samples output of an un-equalized or under-equalized channel

The worst-case slope of un-equalized signal can be estimated by taking a derivative of the channel step response. As shown in Figure 13, step response is a superposition of pulse responses shifted by bit time T_{bit} with respect to each other. Thus, the slope of the step is equal to the pulse response samples divided by bit time T_{bit}.

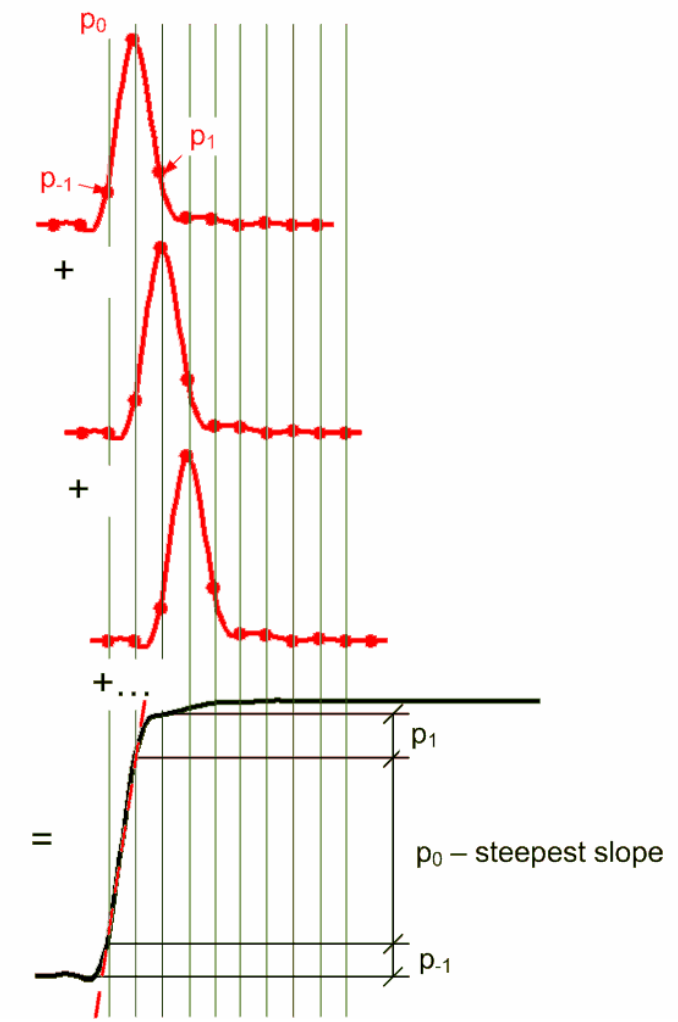


Figure 13. Worst-case slope in un-equalized channel

The maximum effect of jitter on sampled voltage is then

$$\sigma_{\Delta V} \approx \frac{dV}{dt}\sigma_{\Delta t} \approx \frac{p_0}{T_{bit}}\sigma_{\Delta t}, \tag{2.2}$$

where p_0 is the cursor size. Given our target of αp_0 for an LSB size of the ADC, $\frac{\sigma_{\Delta t}}{T_{bit}}$ should be much less than $\alpha \approx$ 10-20% in order for jitter not to limit the ADC

performance, which is generally well within a jitter specification of the timing reference of a typical link receiver. For example, in the link described in **[15]**, the peak-to-peak PLL jitter was 26ps for T_{bit}=200ps, which would translate into 0.65-1.3*LSB* peak-to-peak voltage noise. Therefore, the jitter performance of a typical link receiver would also satisfy the requirements for an ADC-based receiver.

2.6. SUMMARY

Many modern high-speed links signal over channels with very limited bandwidths. To compensate for the resulting channel distortion, they utilize various equalization techniques, most of which require data conversion between digital and analog domains. Depending on the overall system design and channel characteristics, such data converters range from low-speed DACs used to adapt equalizers to slowly-varying characteristics of the channel to very high speed signal-path ADCs and DACs for interfacing the channel to fully-digital equalizers. Since the main focus of this work is analog-to-digital conversion, in Chapter 3 and Chapter 4 we concentrate on the design of baud-rate ADCs for serial link receivers.

CHAPTER 3. ANALOG-TO-DIGITAL CONVERTERS

Fast and efficient analog-to-digital converters (ADCs) are a key to enabling digital equalization in a high-speed link receiver. The advantages of digital receive equalization as well as the requirements for the front-end ADC have already been described in Sections 2.4. and 2.5. , so in this and the next chapter, we focus on the ADC design itself.

Until recently, the need for multi-Gsps rates left designers one choice – to use FLASH ADCs. Unfortunately, these had a reputation of power and area hogs, hardly suitable for high-speed link receivers. However, aggressive scaling of CMOS technology brought changes to the landscape of high-speed ADCs. First, some recent work (for example, by G. Van der Plas, *et al.* **[36]** and by B. Verbruggen, *et al.* **[37]**) has demonstrated spectacular improvement in the power- and area-efficiency of FLASH ADCs (or their slight modifications), at least for low-Gsps conversion rates and 4-5 bits of resolution. This preempted or roughly coincided with the first commercial usage of a FLASH ADC in a high-speed link receiver **[13]**. On the other hand, the strategy of time-interleaving multiple slower but potentially more efficient ADCs also proved very successful, as demonstrated, for example, in **[38]**-**[42]** for successive-approximation (SAR) ADCs and in **[43]**-**[45]** for pipeline ADCs.

In this chapter, we first review the more proven, in the context of high-speed links, FLASH architecture, but then point out what difficulties are likely to arise as we attempt to push FLASH conversion rates into tens of Gsps, which will likely require interleaving. We briefly review the interleaving technique itself and the two most commonly interleaved ADC architectures (apart from FLASH): SAR and pipeline ADCs. However, for 4-5-bit resolution, neither of them has a clear advantage

over FLASH. Therefore, in Chapter 4, we choose to investigate what we think is a promising but unusual alternative: a single-slope ADC. Although single-slope ADCs are normally found only in very low-speed systems (such as imagers), we demonstrate their suitability for high-speed operation and their advantages for interleaving.

3.1. FLASH ADC

A FLASH ADC works by comparing an input signal with a set of reference voltages at the same time, as shown in Figure 14(a). Because the effective sampling instants of real-life comparators may differ slightly (due to mismatch, clock skew, etc.), a centralized track-and-hold circuit (T/H) is often inserted in front of a comparator array. The comparator array outputs a **thermometer code**, which, given a certain imagination, resembles the display of an alcohol thermometer. The thermometer code is then decoded into a binary code, usually more suitable for digital processing. For a resolution of N_{bits}, $2^{Nbits}-1$ comparators are required, leading to the exponential increase of power and area with resolution, as shown in Figure 14(b). Therefore, FLASH ADCs with $N_{bits}\geq 8$ are rarely, if at all, encountered, while for $N_{bits}<5$, FLASH is almost always the best choice.

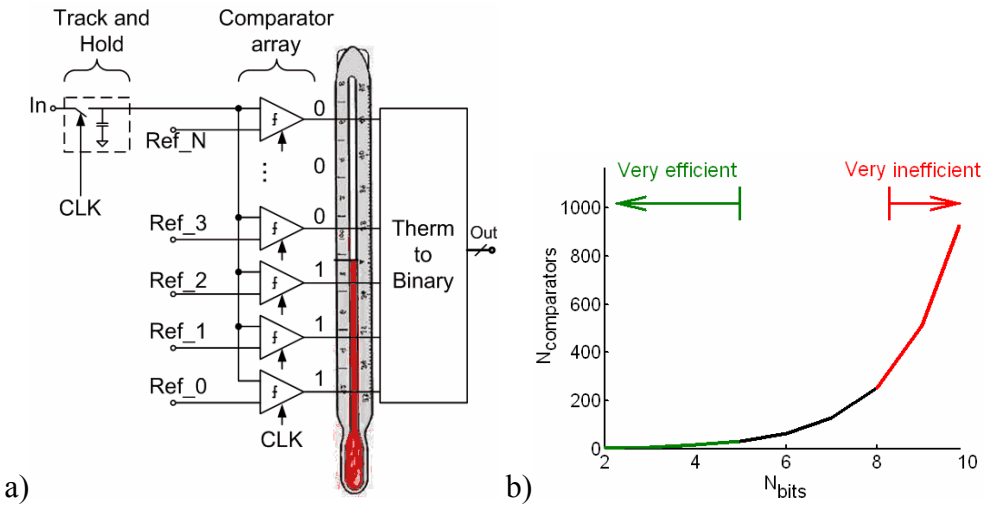


Figure 14. (a) FLASH ADC is an array of comparators that compare an input signal with multiple voltage reference levels at the same time; (b) FLASH power consumption and die area grow exponentially with N_{bits}

Nevertheless, even for our target resolution of 4-5 bits, FLASH architecture poses some challenging problems. Most of the problems arise because of imperfections in the basic and most important block of a FLASH ADC – the comparator. As it is replicated many times, the comparator design is subject to stringent power consumption and area constraints. The comparators should exhibit low enough input-referred thermal noise and high enough tolerance to supply and common-mode variations in order not to limit the system performance. Almost always, the supply and common-mode rejection requirements necessitate differential designs.

3.1.1. COMPARATORS

A **comparator** is a highly non-linear amplifier that amplifies the difference between its signal input *In* and reference input *Ref* and saturates in *Low* and *High* regions, which are unambiguously interpreted by the downstream digital logic as digital 0 and 1, as shown in Figure 15(a). When the comparator output falls in the

transition region between *Low* and *High*, it is considered **metastable** or invalid. Thus, the **comparator gain** A should be as large as possible to minimize the probability of invalid outputs. More formally, assuming uniform distribution of quantization noise in the ADC, gain A required to guarantee the total error probability of P_{error} can be estimated as

$$A = \frac{High - Low}{P_{error} \cdot LSB}, \tag{3.1}$$

where (*High* – *Low*) is the minimum swing of comparator output required for the digital logic to unambiguously discriminate logic 1 from logic 0; *LSB* is the ADC input step size, and P_{error} is the probability that any of the comparators in ADC produces invalid output (from **[30]**). Most short-reach high-speed links do not have any error correction capabilities, and thus require very low error rates $P_{error} < 10^{-12}$ and, therefore, extremely high comparator gain.

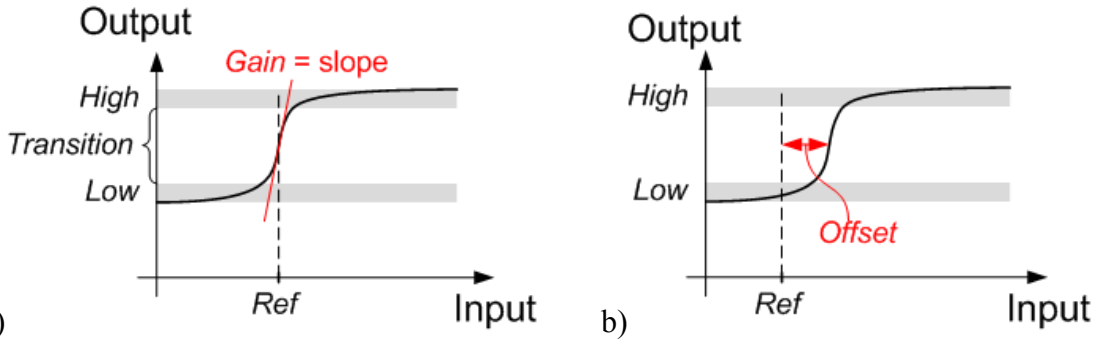


Figure 15. Transfer characteristics of a hypothetical comparator

Ideally, the output of a comparator transitions from *Low* to *High* when its signal input, *In*, crosses its reference, *Ref*. **Input-referred offset** is the horizontal shift in the transfer characteristic of the comparator from being centered around *Ref* to being centered around *Ref*+*Offset*, as shown in Figure 15(b). Random offset is usually

due to mismatch of transistors (or resistors) in the comparator. If not cancelled or taken into account, offset results in static non-linearity of the ADC.

3.1.1.1. Static Non-linearity

The term **static linearity** refers to the linearity of the ADC transfer characteristic at DC or low frequencies. The linear and hypothetical non-linear transfer characteristics are shown in Figure 16. There are two types of non-linearity – differential and integral – defined for ADCs. After performing gain and offset corrections, the differential non-linearity (DNL) is defined as the difference between an actual step width and the ideal value of 1*LSB*, while the integral non-linearity (INL) is defined as the deviation of an actual transfer function from a straight line.

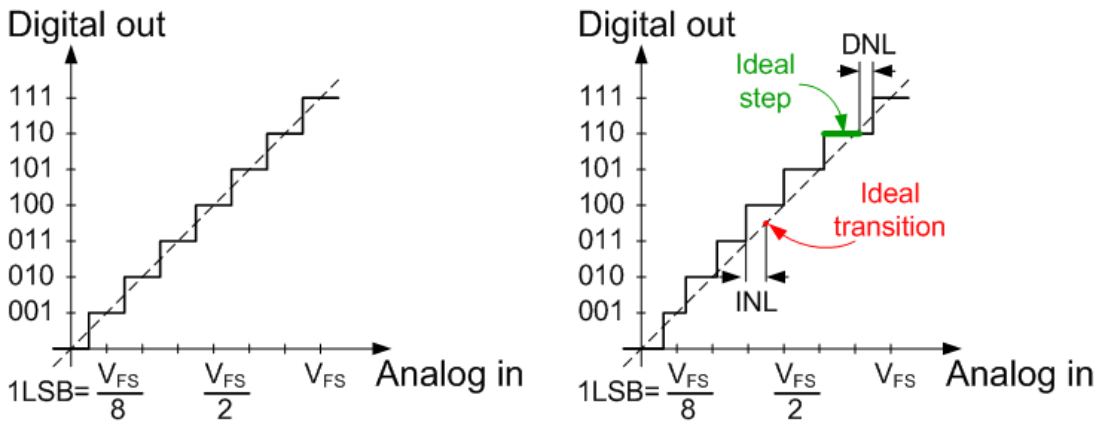


Figure 16. a) Linear and b) hypothetical non-linear transfer characteristics of a 3-bit ADC

The static non-linearity of a FLASH ADC is normally limited by the comparator mismatch. Since the reference levels are commonly generated with a relatively linear resistor-string DAC, their non-uniformity is usually only a second-order effect. There are several ways to guarantee matching of the individual comparators. First, making comparators physically large reduces mismatch, since random mismatch is inversely proportional to the device area. Apart from die area, this

approach also requires a proportional increase in power consumption needed to drive the increased parasitics.

The second possibility is to dynamically cancel the comparator offsets on every conversion cycle. This usually involves storing the offset on the capacitors in the signal path. Unfortunately, the additional switch and capacitor parasitics slow down the comparator. Furthermore, although comparator offset changes slowly (if at all), charge on the offset-storage capacitors needs to be refreshed continuously, significantly lengthening the conversion cycle.

Finally, it is possible to correct the offset "digitally" using a digital-to-analog converter (DAC) in each comparator to induce a current, tranconductance, or capacitive imbalance, counteracting a built-in offset. As long as the offset is not particularly large, either of these schemes works fairly well, but to determine the appropriate DAC setting, the offset needs to be measured first. Unfortunately, measuring comparator offset generally requires rearranging a signal path, again slowing down the comparator.

In the preceding discussion, we mentioned that the gain of comparators must be high enough for the required metastability error rate, and that the offsets must be low enough for the required linearity. However, so far we have only alluded to the fact that both high gain and low offset come at a price of comparator speed, while, in fact, this is a critical consideration for FLASH ADCs. More formally, this trade-off is described by the gain-bandwidth product.

3.1.1.2. Gain-Bandwidth Product

Simultaneously achieving high gain and high bandwidth presents a serious problem. In fact, someone used to a linear trade-off between gain and bandwidth must think it is impossible to design ADCs with conversion rates much higher than f_t / A, where f_t is the unity-gain frequency of transistors in a given

technology, and A is the comparator gain, required for a given metastability error rate. Furthermore, any offset cancellation schemes generally deduct from the available gain-bandwidth product less directly but quite measurably. Nevertheless, many such designs exist. What makes them possible is the fact that gain and bandwidth trade off linearly only in first-order systems (such as an operational amplifier with one dominant pole). More generally, the trade-off is between three parameters: the gain, the bandwidth, and the delay (or latency). There are several ways to exploit this, leading to different comparator circuits and ADC architectures.

From a fundamental perspective, a circuit that most efficiently trades off delay for gain and bandwidth is a distributed amplifier, shown in Figure 17(b). We can arrive at this topology, if we start with a regular common-source amplifier stage, Figure 17(a). Its bandwidth limitations arise from the pole associated with $R_L C_{out}$ and possibly from C_{in} presenting a capacitive load to the previous stage. However, if the common-source stage is split into N sections, and the sections are joined together with inductors, both C_{in} and C_{out} are absorbed into the artificial input and output transmission lines. To guarantee stability, the inductor values should be chosen so that the characteristic impedance of the output line $Z_{out} = \sqrt{\frac{L_{out}}{C_{out}}}$ equals $2R_L$. For maximum gain, the phase velocities in the input and output transmission lines should match:

$$v_{out} = \frac{1}{\sqrt{\frac{L_{out}}{l} \cdot \frac{C_{out}}{l}}} = \frac{l}{\sqrt{L_{out} C_{out}}} = \frac{l}{\sqrt{L_{in} C_{in}}}. \tag{3.2}$$

Under these conditions, the gain of a distributed amplifier is $|A_v| = g_m Z_{out}/2 = g_m R_L$, which is the same as that of the original common-source stage. As increasing R_L does not degrade the bandwidth, distributed amplifier is theoretically capable of achieving large gain-bandwidth product at arbitrary low power

consumption. Of course, there is an upper limit on Z_{out} for any practical transmission line.

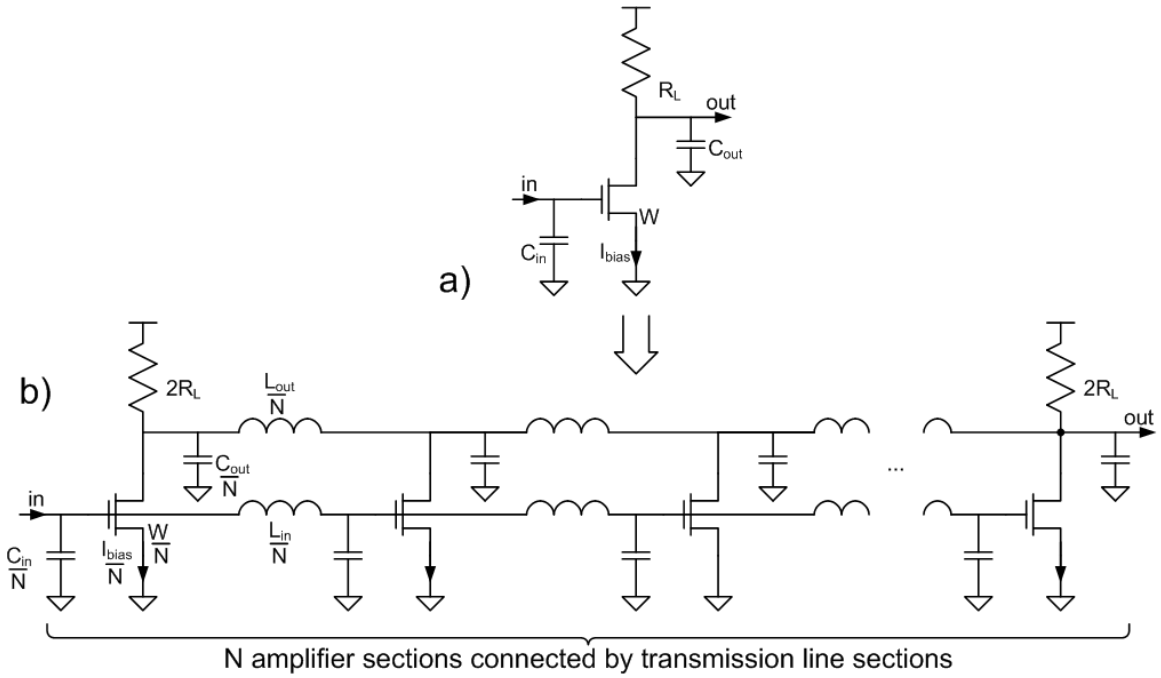


Figure 17. Evolution from a lumped common-source amplifier (a) to a distributed amplifier (b)

In addition to using distributed amplifiers as comparators, one might also envision building the whole comparator array as a distributed structure to absorb the ADC input capacitance into a transmission line, as shown in Figure 18.

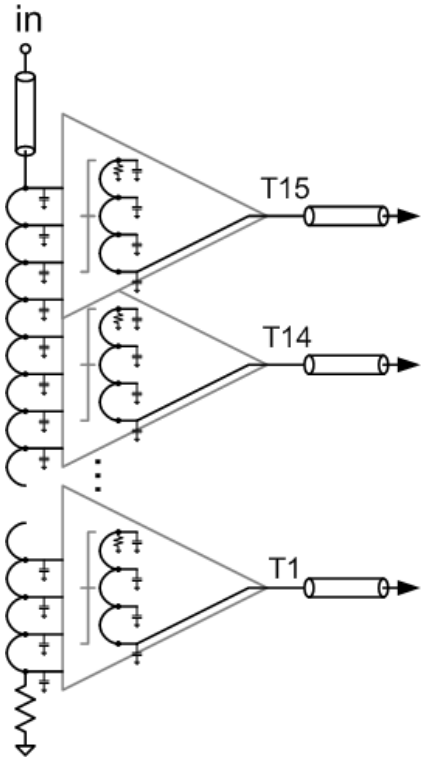


Figure 18. Distributed FLASH ADC

Theoretically, such an ADC consumes arbitrarily low power even for large bandwidth (throughput), provided that we are willing to wait arbitrarily long time for a conversion result (arbitrary long delay through comparators). The only power consumption is associated with an input and the outputs of the ADC. This power cannot be made arbitrarily small, or the useful signals would be overwhelmed by thermal noise.

In reality, many practical issues make a distributed amplifier a questionable, if not poor, choice for a comparator. Performance of distributed amplifiers is limited by deviations from ideal behaviors of the device and the transmission lines. These include non-quasi-stationary phenomena in the channel (or base) of transistors, hard-to-control parasitics, and loss in the transmission lines. In digital CMOS technologies, inductors and transmission lines are particularly lossy.

Further, the area budgets for most high-speed links are very tight, essentially prohibiting or severely limiting the use of inductors. The concept of a distributed ADC, however, may become more practical for stand-alone high-speed ADCs less constrained in die area and choice of technology. More importantly, the distributed ADC concept shows the full potential of making use of all available latency budget, either at a circuit level or at an architectural level or both.

At a circuit level, short of an ideal distributed amplifier, there are several practical ways to exploit latency. The first one is a cascade of multiple low-gain amplifier stages, such as shown in Figure 19. While overall gain A of the cascade is the product of gains of individual stages, the -3dB bandwidth is

$$f_{-3dB} = \frac{\sqrt{2^{1/n} - 1}}{2\pi \cdot R_L C_L}, \tag{3.3}$$

where n is the number of stages.

Thus, instead of staying constant, the gain-bandwidth product $f_{-3dB} \cdot A$ increases with the number of stages in a cascade.

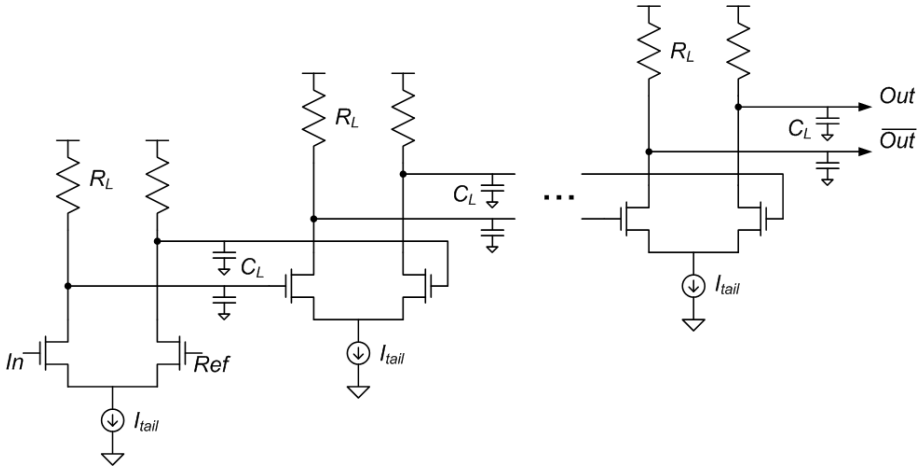


Figure 19. A cascade of multiple low-gain amplifier stages (assuming single-ended inputs for simplicity)

Unfortunately, there are problems associated with a cascade of low-gain stages. First, all stages continuously consume current resulting in high power consumption for large number of stages. Second, as the order of the system increases with increasing number of stages, the phase distortion near corner frequency becomes worse. Since the outputs of different comparators in an array are square waves of different duty cycles, they get distorted and delayed unevenly, producing dynamic distortion. In addition, continuous-time amplifiers have bandwidth mismatches, whose effect gets worse with delay. Therefore, most practical comparator designs in FLASH ADCs use a master clock to synchronize the comparator array, essentially operating in discrete time.

A very efficient amplifier that works well in discrete time is a regenerative amplifier, such as a CML (current-mode logic) latch shown in Figure 20. In the sampling phase (*Sample* = 1; *Regen* = 0), the differential pair stores the somewhat amplified input on a pair of load capacitors C_L. In the regeneration phase (*Sample* = 0; *Regen* = 1), the positive feedback loop exponentially amplifies the sampled voltage. This positive feedback loop around an amplifier can be thought of as

a signal propagating through a long chain of cascaded amplifiers, at a cost of power and area of only one amplifier stage.

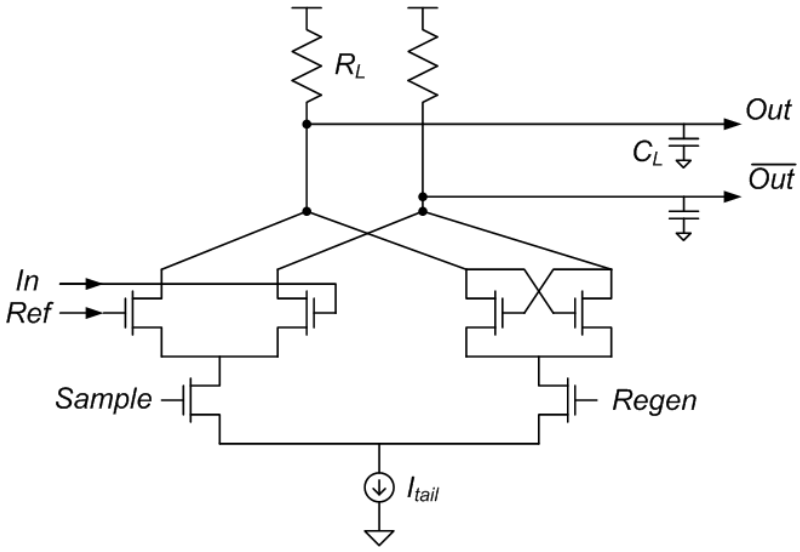


Figure 20. A regenerative amplifier (CML latch)

There are numerous variations of this basic concept, notably a number of topologies that consume no static current at all as, for example, different modifications of a StrongArm latch, shown in Figure 21 and first developed for digital circuits **[29]**. Unfortunately, these tend to have somewhat larger parasitic capacitances than the simple CML latch. For very short clock cycles, the outputs of a StrongArm latch also spend a large proportion of the regeneration phase in the middle of supply rails, negating the power savings that come from CMOS-like operation with no static power consumption. For more discussion of comparative merits of CML and StrongArm comparators, the reader is referred to **[66]**.

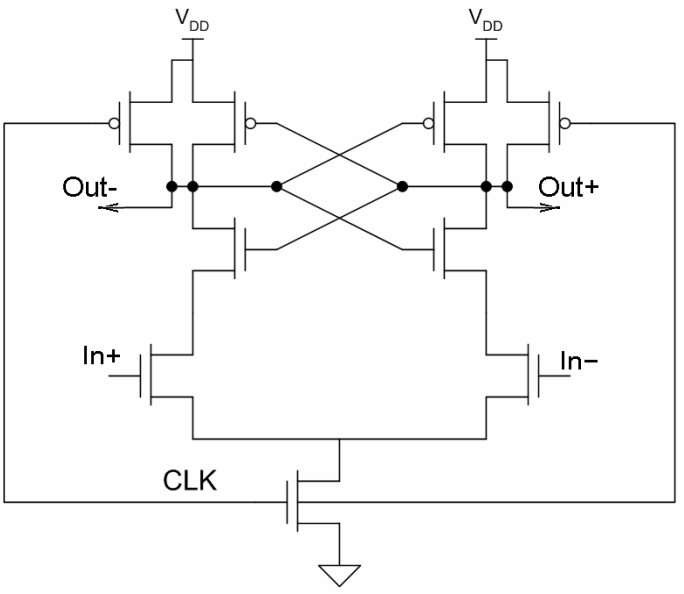


Figure 21. Simplified schematic of a StrongArm latch

It can be shown that the gain of a regenerative amplifier is

$$A \approx A_0 \exp\left(\frac{t_{reg}}{C_L / g_m}\right) = A_0 \exp\left(\frac{t_{reg}}{\tau_{reg}}\right), \tag{3.4}$$

where A_0 is the linear gain of an input stage; t_{reg} is the duration of regeneration phase; τ_{reg} is the regeneration time constant; C_L is the load capacitance at output nodes, and g_m is the transconductance of transistors. When the clock period leaves enough time t_{reg} for regeneration, the regenerative amplifiers perform so well that they can be used as comparators by themselves (for example, in **[36]** and **[37]**).

However, as the conversion rate increases, the gain A drops off sharply. To compensate, several regenerative amplifier stages have to be cascaded (pipelined), as shown in Figure 22(a). Note that pipelining is a convenient way to make use of the latency budget at a circuit level. The accuracy and bandwidth of the later stages is less important, so regular digital latches (**metastability latches**) are often used.

Unfortunately, the overhead of a reset phase and associated circuitry in a regenerative amplifier and metastability latches limit the practical pipeline depth.

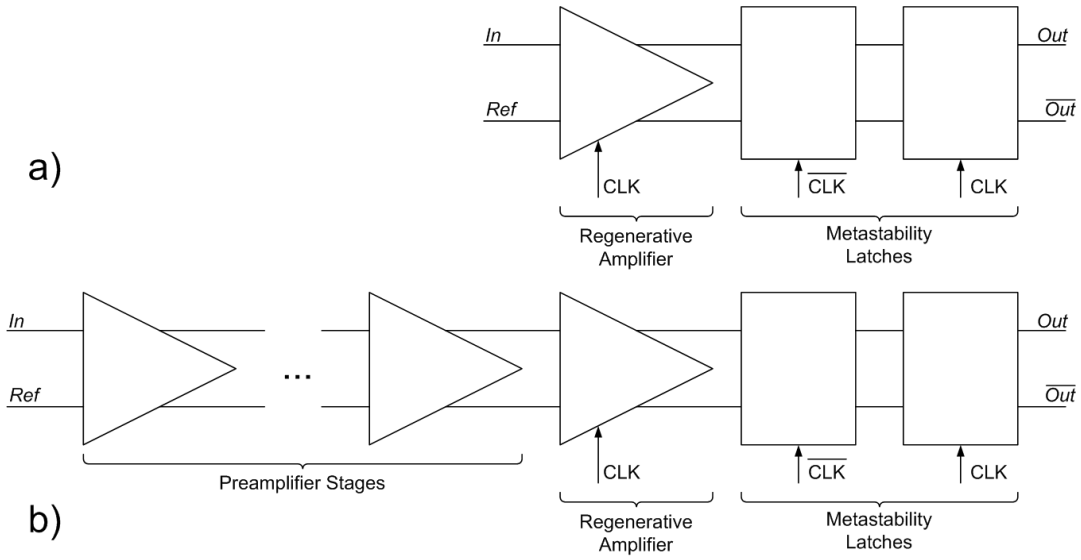


Figure 22. (a) Pipelining regenerative amplifier and metastability latches and (b) adding continuous-time preamplifier stages in front of regenerative amplifier

As shown in Figure 22(b), several continuous-time preamplifier stages are also often placed before a regenerative amplifier for several reasons. In addition to contributing to the overall comparator gain-bandwidth product, they reduce the feedthrough from a regenerative amplifier back into the inputs, have better offset and noise performance, and facilitate various offset cancellation and averaging schemes. However, preamplifiers consume static power, which makes them less attractive.

A natural alternative to pipelining and cascading stages is parallelism, which translates into time-interleaving multiple ADCs.

3.2. TIME-INTERLEAVING

The concept of time-interleaving is illustrated in Figure 23. The sampling times of n ADCs are staggered in time, so that each of the sub-ADCs is only required to complete one conversion every $n \cdot T_S$, where T_S is the sampling period of the entire ADC. But since there are n ADCs instead of one, interleaving makes sense only if backing off the conversion rate by a factor of n scales down the size and/or power consumption by a much larger factor, large enough to justify the overhead of interleaving. Most often, this is the case only if either the resolution is too high for a FLASH ADC, or T_S is so short that it makes comparators in FLASH very inefficient for reasons mentioned above. Interleaving is also necessary regardless of area and power considerations when the required T_S simply can not be attained with one single ADC.

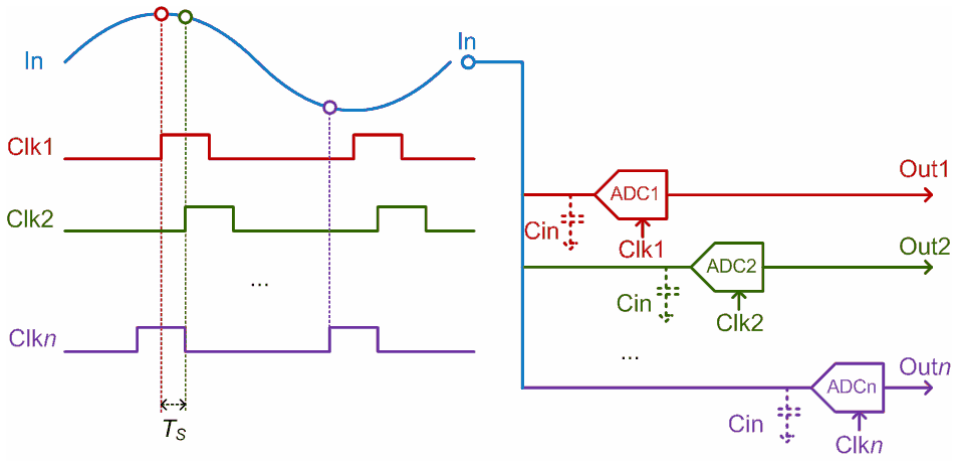


Figure 23. Time-interleaving n ADCs increases aggregate throughput by a factor of n

The structure in Figure 23 also requires the acquisition bandwidth of each sub-ADC must still be as high as the bandwidth of the overall ADC. At the same time, the input capacitances of all sub-ADCs add up, presenting a significant capacitive load to the input. Placing a high-speed track-and-hold (T/H) in front of each sub-ADC as shown in Figure 24, aims to alleviate both problems. First, the high-

bandwidth requirement is removed from the sub-ADCs and transferred to the T/Hs. Second, whenever a T/H goes into a hold mode (its Clk is LOW), Cin of its corresponding sub-ADC does not load the input. Ideally, this would reduce the total input capacitance by the duty cycle of the clock from n·Cin to DutyCycle·n·Cin. In practice, the parasitic capacitance of the track-and-holds significantly adds to the input capacitances of the sub-ADCs themselves, offsetting the improvement. Interestingly, the sampling scheme of Figure 24 resembles the distributed input network in Figure 18, with inductors replaced by phased switches, more amenable to CMOS implementation.

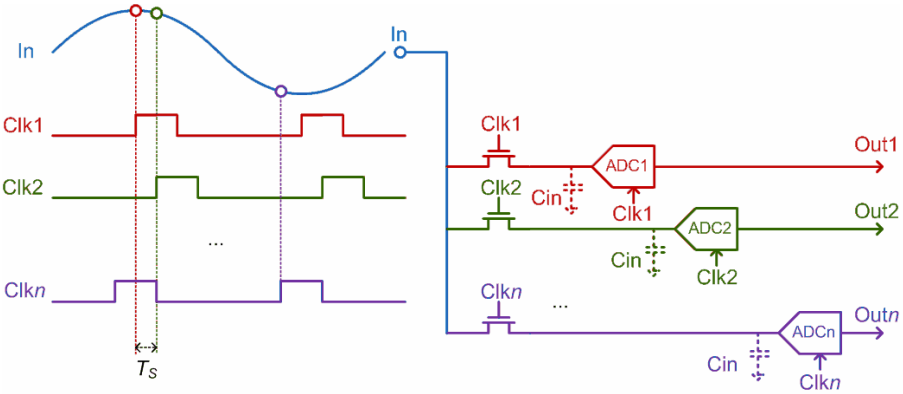


Figure 24. Time-interleaving with track-and-hold circuits in front of each sub-ADC

Apart from the bandwidth and load capacitance issues, any offset mismatch between sub-ADCs produces a periodic error sequence that shows up as spurs in the output spectrum of the ADC. The gain mismatch between sub-ADCs results in an amplitude modulation. In addition, the time-skew between sampling times of the sub-ADCs presents a completely new issue specific to interleaving. The impact of the offset and gain mismatches and time-skew on performance of interleaved ADCs is discussed in detail in **[26]**. While multiple solutions for minimizing, calibrating, and cancelling these errors have been proposed, interleaving must offer significant advantages in order for this additional complexity to be justified.

One of the reasons to consider interleaving is the fact that FLASH ADCs become rather inefficient for resolution higher than about 5-6 bits, while somewhat slower architectures, like successive-approximation (SAR) and pipeline ADCs, can be designed for nominal resolution of up to about 14 bits and interleaved for high aggregate throughput.

3.2.1. SUCCESSIVE APPROXIMATION (SAR) AND PIPELINE ADCS

The block diagram and operation of a SAR ADC is shown in Figure 25. A SAR ADC first compares the sampled voltage V_H with 0 (mid-range), determining the sign bit or MSB. Then the state machine changes the DAC output V_{DAC} to compare V_H with the midpoint of either the top half (if MSB was 1) or the bottom half (if MSB was 0). This determines the next most significant bit. The process of half-sectioning continues until the LSB is determined.

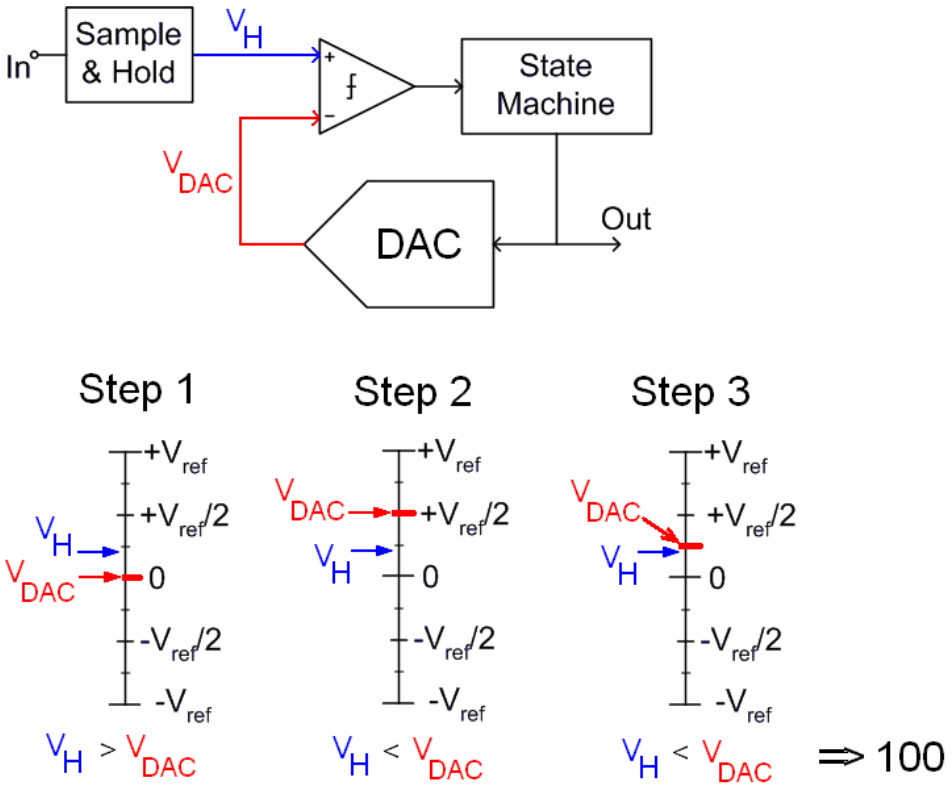


Figure 25. Successive-approximation register (SAR) ADC

The pipeline ADC, shown in Figure 26, resembles a SAR ADC, except that it pipelines the half-sectioning steps instead of performing them sequentially. Each pipeline stage compares its input with 0, zooms in to either the positive half-range (if input was positive) or the negative half-range (if input was negative), and passes the result to the next stage.

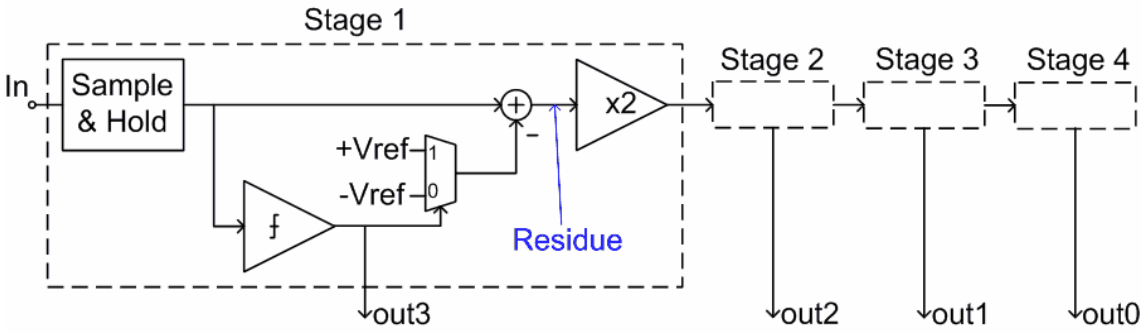


Figure 26. Basic pipeline ADC implementation

Both SAR and pipeline ADCs take N_{bits} steps to perform a conversion. A SAR ADC takes these steps sequentially, while pipeline ADC pipelines them. Nevertheless, both architectures require N_{bits} comparisons in contrast to a FLASH, which performs 2^{Nbits}-1 comparisons at each conversion. This is the main reason for replacing a FLASH with an interleaved SAR or pipeline ADC for resolution higher than 5-6 bits. This conventional cut-off between FLASH and SAR or pipeline ADC may seem strange: if we only count the number of comparisons, then bit-at-a-time converters should become more efficient than FLASH already for N_{bits}=2-3. But it is additional complexity – a sample and hold, a DAC (for SAR), and a linear amplifier (for pipeline) – that pushes the realistic cut-off to approximately 5-6 bits.

3.2.2. Interleaving for lower-resolution ADCs

The 4-5 bit resolution needed for high-speed link receivers does not justify switching to interleaved pipeline or SAR ADCs with their additional complexity. Still, there is another reason to consider interleaving. As we attempt to increase the conversion rate and bandwidth of a FLASH ADC, its power consumption grows faster than linearly.

There are several reasons for this. First, due to the fixed parasitic capacitance (associated with interconnect and certain device parasitics), active device

sizes need to be scaled up to increase the speed. When active device sizes become so large that their scalable parasitic capacitance dominates the fixed parasitic capacitance, any additional increase in speed requires a very large increase in device size and, thus, power consumption. This consideration is discussed more formally for continuous-time comparators in Section 4.2.7. .

The second reason is more applicable to CMOS-like regenerative amplifiers. The CV^2f-power of a single regenerative amplifier stage scales linearly with frequency. However, as time available for regeneration shortens, the gain per regenerative stage drops exponentially, requiring more cascaded stages for a given total gain target. This, again, results in power consumption scaling with frequency much faster than linearly.

As shown in a hypothetical power-versus-frequency curve in Figure 27, for conversion rates higher than a certain frequency, it becomes more power-efficient to interleave, even though there is a certain power overhead associated with interleaving.

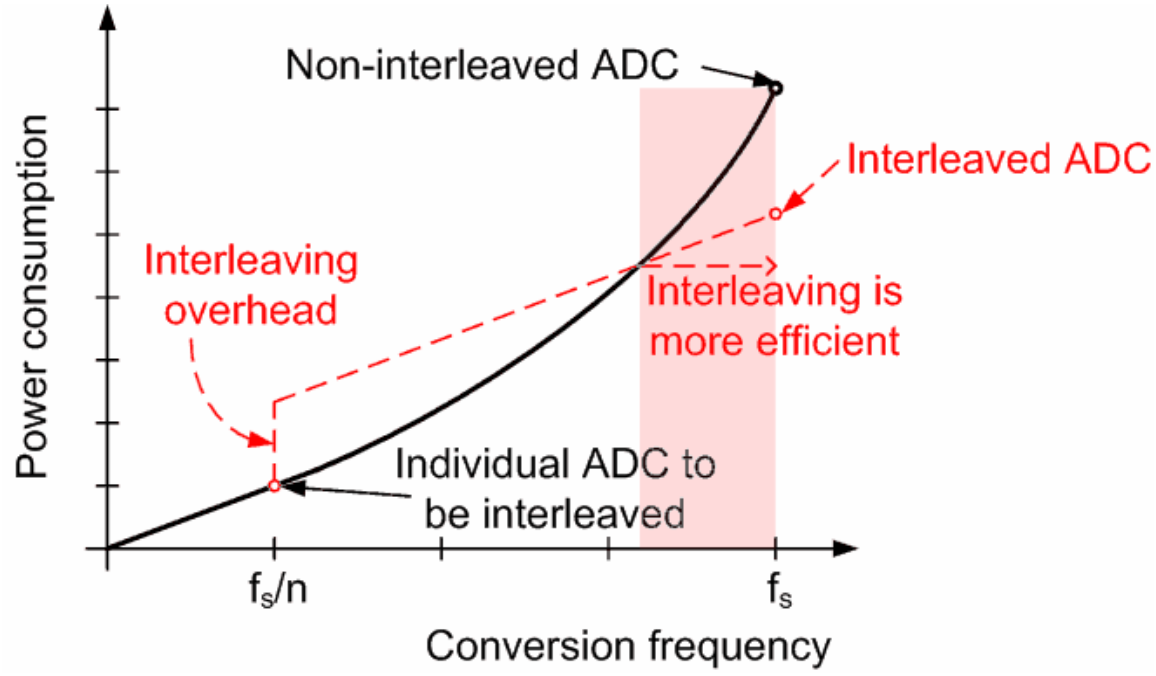


Figure 27. Power savings from interleaving

But what ADCs should be interleaved? As we pointed out, for resolution below 5-6 bits, switching from FLASH to either of the bit-at-a-time architectures does not pay off. At the same time, FLASH ADCs have a very large input capacitance, which makes them difficult to interleave. If no track-and-holds are used, *n* FLASH ADCs load the input, while T/Hs that have to drive large load capacitances consume a lot of power on their own. The solution that we propose in this work is to use the high-speed single-slope ADCs instead of FLASH, as described in detail in the next chapter.

CHAPTER 4. HIGH-SPEED SINGLE-SLOPE ADC

The single-slope ADC, a simple architecture frequently used in imagers, offers high linearity, low power consumption, and small input capacitance. Single-slope ADCs belong to a class of **level-at-a-time** converters that take an order of 2^{Nbits} steps to perform an N_{bits} conversion and, thus, are traditionally considered suitable only for low-speed applications. But, as we show in this chapter, carefully leveraging the high-speed link environment allows us to increase the conversion rate of single-slope ADCs to around 1Gsps in a 45nm CMOS technology. Furthermore, the simplicity and small input capacitance of single-slope ADCs make them better candidates for interleaving than either bit-at-a-time ADCs (relatively complex) or FLASH ADCs (large input capacitance). As a result, interleaved single-slope ADC becomes a promising solution for link receivers operating at tens of Gbps.

To evaluate this concept, we implement a high-speed single-slope ADC on a Texas Instruments 45nm test chip. Based on the performance of this implementation, we estimate within what resolution and conversion rate limits this single-slope ADC is efficient. We also investigate interleaving multiple single-slope ADCs and study the power efficiency of such a solution. Finally, we conclude and describe potential directions of future investigation. But first let us overview the basic single-slope architecture, its advantages and limitations.

4.1. OVERVIEW OF BASIC SINGLE-SLOPE ARCHITECTURE

A simplified block diagram of a single-slope ADC is shown in Figure 28. The ADC works as follows. At the beginning of a conversion cycle, a differential input voltage is sampled onto a pair of sampling capacitors C_{H+} and C_{H-}. Then the current source I_{drain} switches on and starts draining charge from C_{H+}. The voltage on

node *hold+* linearly decreases until it crosses the voltage on node *hold-* (assuming for now that $V_{hold+}>V_{hold-}$). At that point, the comparator that compares V_{hold+} with V_{hold-} trips, causing a set of latches to latch the current counter value *count*. Meanwhile, the counter keeps running, and its wraparound initiates a new conversion starting with sampling an input voltage again. In this way, initial counter state is always synchronized with start of a conversion, and the latched counter state is proportional to the input voltage.

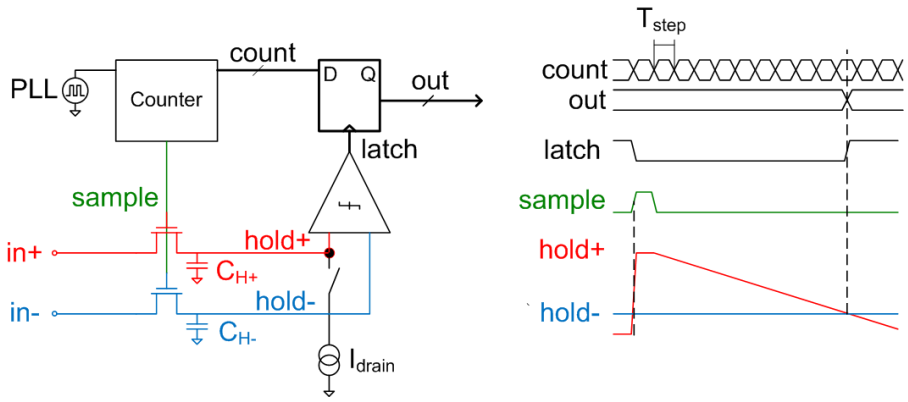


Figure 28. Simplified block diagram of a single-slope ADC

There are variations on this basic theme, but the main concept is the same. In fact, there is even a more general way to view single-slope converter – it is a voltage-to-time converter (consisting of a ramp generator and a comparator) followed by a time-to-digital converter (set of latches that latch the current counter value). This view will be useful later in the discussion of the efficiency of voltage-to-time converter.

As already mentioned, a single-slope ADC has the following features:

- Linear, because ramp generator can be made to have relatively good linearity, and the ADC produces no missing codes

- Has relatively easily measurable and correctable offset and gain errors
- Low complexity
- Power-efficient, because there is only one comparator
- Slow, because it takes 2^{Nbits} clock cycles per conversion

However, let us look at the speed limitation more closely. Single-slope ADCs are generally considered slow, because they take 2^{Nbits} steps per conversion. But as opposed to bit-at-a-time ADC topologies, the duration of a single-slope conversion step is not limited by the comparator delay. For example, a successive-approximation (SAR) ADC takes only N_{bits} conversion steps to complete a conversion, but at every step it must wait for a comparator to completely resolve. The throughput of a pipeline ADC is limited by the delay of one of its stages, a large proportion of which is again taken up by the comparator and residue amplifier delays. The advantage of a single-slope ADC is that the comparator does not need to resolve fully at each conversion step – assuming it is a continuous-time comparator, it only switches once per entire conversion cycle. Thus, the duration of a conversion step is not limited by the comparator delay but by the speed of mostly digital timing circuits, which improve rapidly with technology scaling.

Serial link receivers already incorporate accurate high-speed time references, usually even with multiple-phase clocks that can be used to further improve timing resolution. For example, on the test chip described below, a 6.25GHz clock with quadrature phases allows T_{step} = 40ps time resolution. Even with 2^{Nbits} steps per conversion, 5-bit conversion will only take 1.28ns (~780Msps conversion rate). This is already very far beyond the speeds single-slope converters normally run at, but even further improvement is possible if more clock phases or clock interpolation is used. For comparison, high-speed SAR and pipeline ADCs tend to run at somewhat

lower conversion rates, but with higher resolution, which is not needed for high-speed link applications.

Unfortunately, for higher conversion rates, the comparator speed does start limiting the performance of single-slope ADCs. Similarly to FLASH, the comparator gain-bandwidth product, not the delay, is of primary importance. While a fixed comparator delay simply results in an offset, a limited gain-bandwidth product causes some input voltage levels to stay undetected or to be measured incorrectly. As shown in Figure 29, if the crossover between *hold+* and *hold-* is not large enough and/or does not occur for a long enough time, the comparator does not produce a full-range transition. Instead, it produces a malformed pulse that either does not latch a count value at all or latches it at a somewhat later time, resulting in non-linearity.

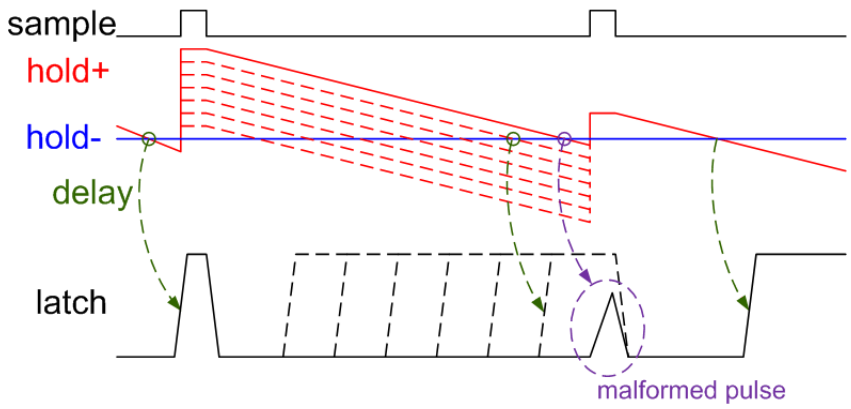


Figure 29. The result of finite gain-bandwidth product of the comparator: small crossover between *hold+* and *hold-* produces a malformed *latch* pulse

As a result, to guarantee a net resolution of N_{bits}, the conversion time T_{conv} has to be extended from the nominal $T_{conv} = 2^{Nbits} \cdot T_{step}$ to

$$T_{conv} = 2^{Nbits} \cdot T_{step} + T_{overhead}, \tag{4.1}$$

where $T_{overhead}$ is the overhead associated with the comparator. As we attempt to increase the conversion rate of a single-slope ADC (for example, by using finer T_{step}), the overhead $T_{overhead}$ also needs to shrink proportionally – at least assuming we would like it to take not longer than a fixed fraction of the conversion time.

However, at a certain rate, the power penalty of decreasing the overhead (increasing comparator gain-bandwidth product) becomes large enough that it is more efficient to interleave a larger number of slower ADCs. In the next section, we investigate this and other practical issues with a specific implementation of a high-speed single-slope ADC.

4.2. SINGLE-SLOPE CONVERTER IMPLEMENTATION IN 45NM CMOS

In order to assess the bounds of conversion rates and resolutions for which single-slope ADCs are practical and can compete with other ADC topologies, we taped out an ADC test chip in a 45nm CMOS technology. The ADC can be programmed to have either 5-bit or 6-bit resolution running at a nominal 1.5625Gsps or 781.25Msps conversion rate respectively. As its timing reference, the ADC uses an on-chip 6.25GHz PLL designed by Peter Hunt at TI to provide quadrature differential clocks for the TI's next generation serial link receiver.

4.2.1. TOP-LEVEL DESIGN

The majority of high-speed links use bipolar signaling, meaning that voltage at *in*- can be greater than *in*+. In such case, when *hold*+ voltage is ramped down, it never crosses over with the voltage at *hold*-. There are several ways to address this problem. One way is to offset the system in such a way that *in*+ is guaranteed to be bigger than *in*-. A roughly equivalent (but perhaps more elegant solution) is to use a purely differential comparator that compares sampled differential

input signal to the ramped reference voltage. However, both solutions boil down to introducing sufficiently large offset to the comparator, which is hard to do, especially for large input voltage range. Another possibility is to compare *in*+ with *in*- before the conversion and then ramp down whichever one is greater. Here, the disadvantage is that this extra comparison will add to the conversion time and complexity of the ADC. The solution we chose is to simply place two ADCs in parallel. One ADC ramps down *in*+, while the second one – *in*-. Although this requires double the hardware, it roughly doubles the conversion rate for a given resolution. Unfortunately, the conversion rate is not exactly doubled, because when the conversion time is shortened, the comparator overhead stays the same and occupies a larger proportion of a conversion time.

There are a few other issues as well. First, since two independent ADCs are used for positive and negative differential voltages, they have different offsets and gains that need to be matched. Second, to make sure that there is no "dead zone" between positive and negative halves, there needs to be a certain overlap of voltage ranges. Thus, for example, the ADC operating on negative half of the signal needs to have a slight offset that extends its range into the positive half. For these reasons, in the ADC implemented on this test chip, the offsets and gains of two half-ADCs are made independently programmable, as shown on the block diagram in Figure 30.

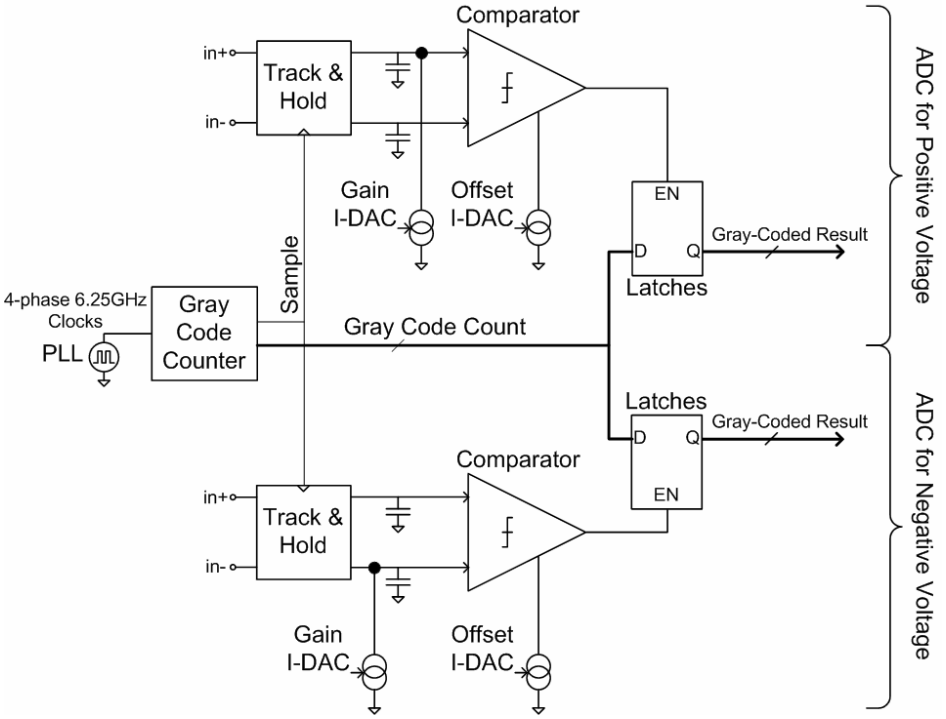


Figure 30. Block diagram of single-slope ADC implemented on a test chip

As evident from the block diagram, the two half-ADCs output their conversion results independently, leaving stitching them to digital post-processing. Each half-ADC is very similar to the concept drawing in Figure 28 with a few minor differences. First, current-mode digital-to-analog converters (I-DACs) are used for adjusting ADC gain and offset. Second, a Gray counter is used to avoid catastrophic errors when counter output is latched at or around a code transition.

4.2.2. TIMING REFERENCE

To generate a timing reference for the ADC, we start with a 4-phase 6.25GHz clock shown in Figure 31.

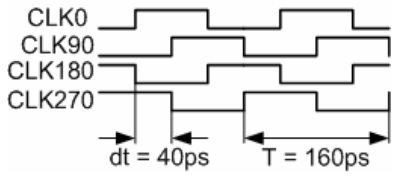


Figure 31. Reference clock from the PLL

Since there are 4 clock phases, we achieve a T_{step} = 1 / (4 · 6.25GHz) = 40ps. By itself, reference clock only goes through 4 distinct states per period, but for N_{bits} conversion, 2^{Nbits} distinct timing states are needed. Thus, some sort of a counter (clock divider) is required. However, a binary code counter can transiently go through invalid states when 2 or more bits change in its output at nominally the same time. Because latching of a counter output occurs asynchronously with the clock, an invalid state may be latched leading to a catastrophic error. This may occur, for example, when a binary counter transitions from 01 to 10 through transient invalid state 00, as shown in Figure 32.

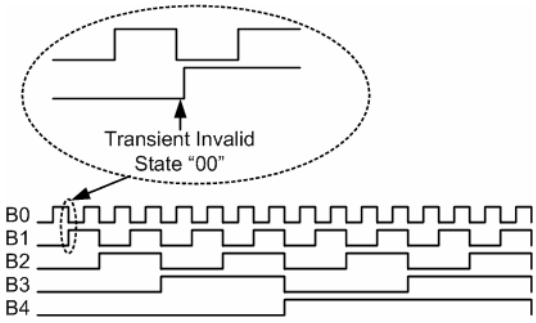


Figure 32. Problems with transient invalid states in a binary code counter

There are multiple ways to encode counter states to avoid this problem, but Gray code provides such encoding with the least number of bits (the same as for binary encoding). A 5-bit Gray code counter waveforms are shown in Figure 33. Since Gray code always has only single-bit difference between neighboring codes, there is no possibility for latching some intermediate invalid state.

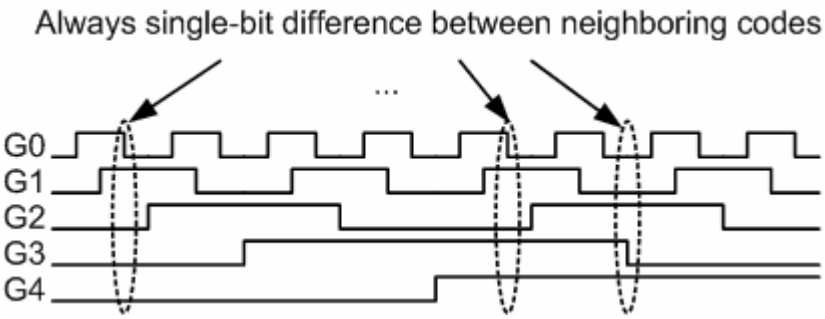


Figure 33. Gray counter waveforms – no possibility of latching an invalid transient state

Ideally (ignoring track and hold and comparator overhead), with a 5-bit Gray coded timing reference, each of the half-ADCs performs 5-bit conversion, yielding (again, ideally) 6-bit overall conversion result when the positive and negative halves are stitched together. To increase conversion rate at an expense of resolution, we can run the ADC in a 5-bit mode with each of the half-ADCs performing 4-bit conversions. To use the same 5-bit timing reference for 4-bit conversions, we perform two conversions per timing reference cycle as shown in Figure 34 – one "odd" conversion for codes 00000 through 01000 and one "even" conversion for codes 11000 through 10000. To obtain identical output encoding for "odd" and "even" conversions, we have to ignore G4 and invert G3 for "even" conversions. Also in the 6-bit mode, the track and hold samples the input signal once per timing reference cycle, while in the 5-bit mode, it samples it twice.

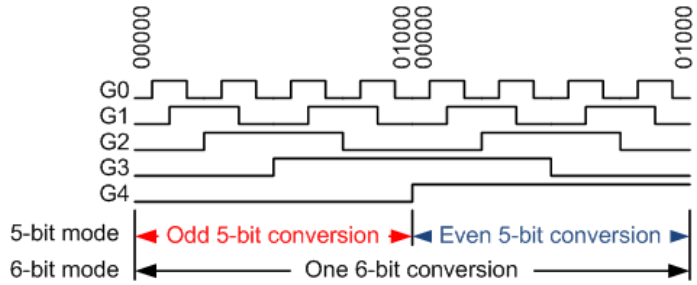


Figure 34. Using 5-bit Gray counter for 5-bit and for 6-bit conversions

To generate Gray-coded signals, we first divide down the input clock by 2, 4, and 8 using a ripple counter and then retime the outputs with appropriate phases of a reference 6.25GHz clock. To obtain divided down clocks with phases needed for the Gray code, not only true (Q) and complement (QB) but also quadrature (QQ and QQB) outputs of a divide-by-2 circuit are used, as shown in Figure 35.

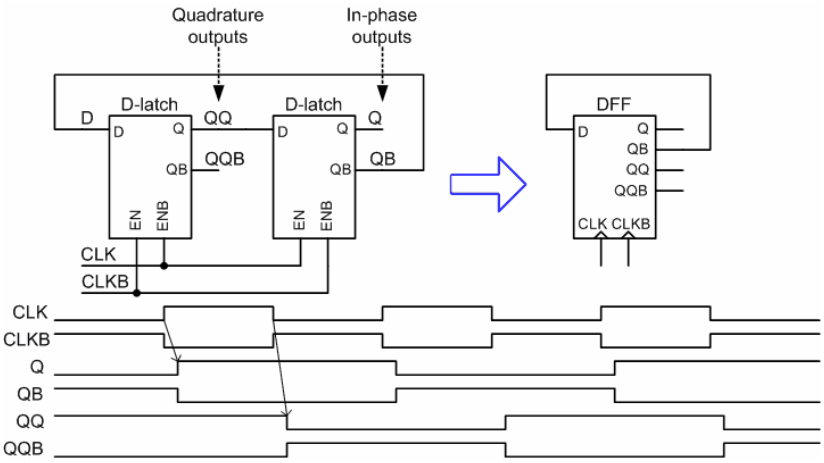


Figure 35. Divide-by-2 circuit used as a building block for Gray code timing generator

The divide-by-8 ripple counter is shown in Figure 36.

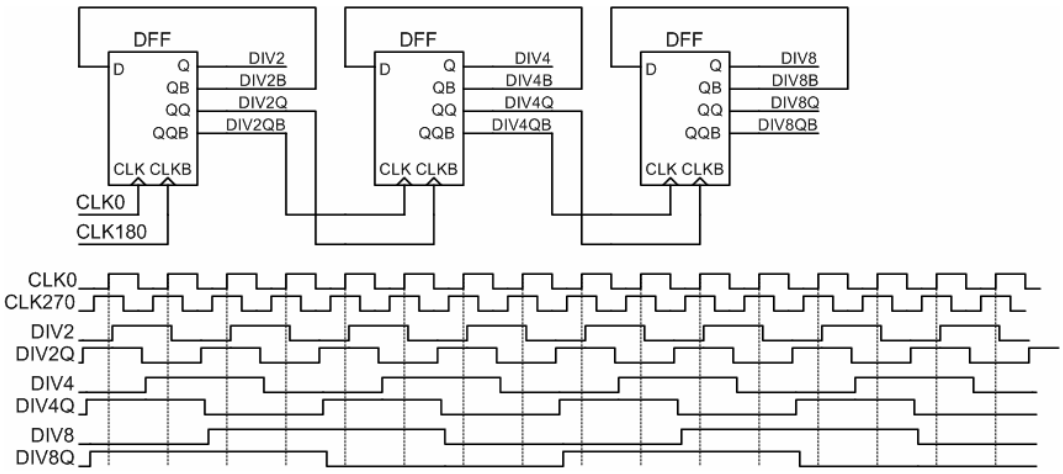


Figure 36. Ripple counter used as a base for generating Gray code

If we ignore slight delays between stages, signals CLK270, DIV2, DIV4, DIV8, and DIV8QB can be used as Gray code timing signals G0, G1, G2, G3, and G4 respectively. However, for accurate timing, we need to retime these signals. Since DIV2 is launched on the rising edge of CLK0, it is retimed with CLK0. Signals DIV4, DIV8, and DIV8QB are launched on the rising edge of DIV2QB, which is in

turn launched on the rising edge of CLK180. Thus these signals are retimed with CLK180. Although the timing requirements for DIV8QB are tight – three Clk→Q delays and the flip-flop setup time have to fit in a 160ps-long cycle, full-custom flip flops allowed us to meet the timing over process, voltage, and temperature corners.

The outputs of the first retime stage drive a decoder that generates pulses initiating new conversions, but to ensure completely balanced load on all final Gray code outputs, another retiming stage has been added, as shown in Figure 37. Clocks CLK270 and CLK90 were used for the second retiming stage instead of CLK0 and CLK180 to approximately balance the loading that timing generator presents to all 4 PLL clock phases.

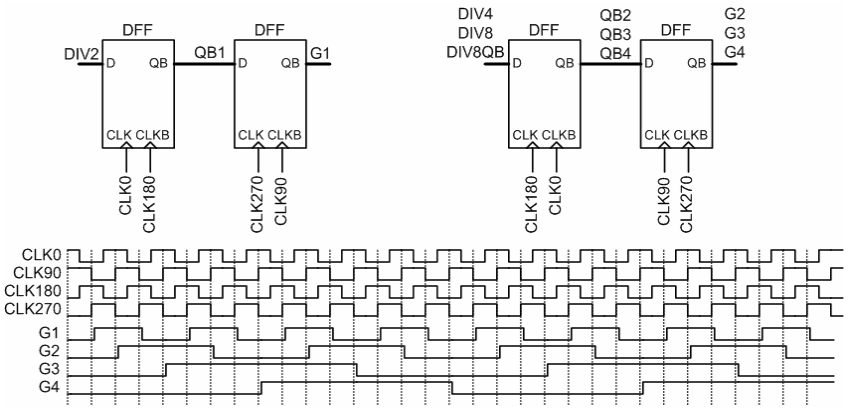


Figure 37. Retiming divided down clock to generate clean Gray-coded timing reference

While G4-G1 are generated by appropriately retiming the divided down clocks, G0 is one of the clock signals itself (specifically, CLK180 for retiming shown in Figure 37), only delayed to match the delay of retiming flip-flops. The CLK→Q delay of a master-slave flip-flop is the CLK→QB delay of the slave latch shown in Figure 38(a). It can be approximately matched with the delay of a "dummy latch" shown in Figure 38(b). The load on node Q in a real latch is slightly bigger, because it has to drive gates of transistors Mn2 and Mp2 in a small tri-state buffer. But its effect

on delay is offset by parasitic capacitance of additional transmission gate S2 in a dummy latch not present in a real latch.

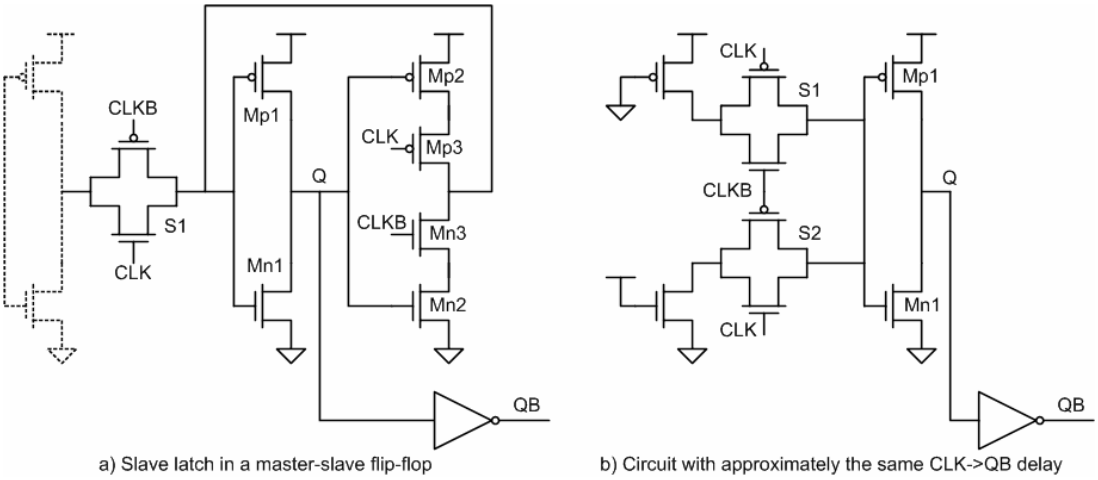


Figure 38. Matching the delay of a flip-flop with a "dummy latch"

The final part of a timing generator circuit generates control signals for the track and hold, comparator, and latches. The first signal is a *Sample* pulse that, when asserted, puts track and hold in track mode, turns off the ramp generator pull-down current, and also resets the comparator. The second signal is a *Reset* pulse that memorizes current conversion result and resets ADC latches to the known initial state in preparation for the new conversion. Since the comparator has significant delay, a previous conversion may not be over before the initiation of the new conversion. Thus, to extend the conversion time somewhat, *Reset* is delayed one time unit T_{step} (nominally 40ps) with respect to *Sample*. The delay of one T_{step} resulted in the least overhead due to comparator delay, as determined from simulations. Both *Sample* and *Reset* stay asserted for two time units $2T_{step}$ (nominally 80ps). To perform 6-bit conversions, *Sample* and *Reset* need to be pulsed every 32 time units, so they can be generated by decoding Gray counter state G<4:0> =x100x, for example, and then gating an appropriate clock phase with the resulting signal. To perform 5-bit conversions, *Sample* and *Reset* need to be pulsed every 16 time units, so they can be

generated by decoding Gray counter state G<4:0> =xx00x, and then again gating the clock with the resulting signal. Figure 39 shows the circuit used to generate *Sample* and *Reset* signals. In this circuit, MODE selects the number of bits and is 1 for 6-bit conversion and 0 for 5-bit conversion.

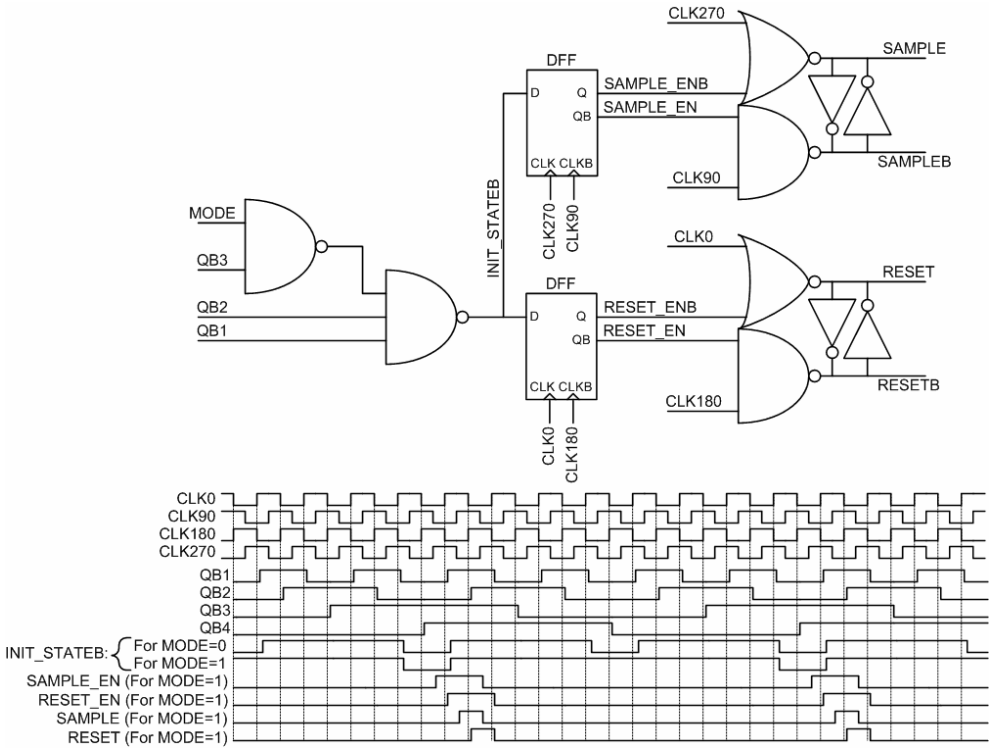


Figure 39. Sample and Reset signal generation

The cross-coupled inverters at the outputs are used to better synchronize the timing of SAMPLE with SAMPLEB and of RESET with RESETB and to minimize the effect of gate delay mismatches.

4.2.3. TRACK AND HOLD

To sample an input signal, we use a simple PMOS pass gate circuit without a buffer amplifier, since the input signals are expected to have a common-

mode voltage close to V_{dd} ($\geq V_{dd} - 200mV$), and ADC input capacitance is very small and can easily be driven directly by the 25Ω source with very high bandwidth. However, this track and hold circuit is tracking only for a short time (two time units, or 80ps), so the PMOS pass gate has to have a relatively low on resistance to ensure complete settling. Since leakage for such short conversion times is not an issue, minimum-length devices are used in the PMOS pass gates. The pass gates were then sized to guarantee settling to ¼LSB in worst-case corner and for worst-case scenario when input signal is at its lowest voltage ($V_{dd} - 400mV$), and sampling capacitor has been discharged by the current source in the previous conversion cycle. However, as we found out shortly before tapeout, for ESD reasons design rules do not allow connection of source/drain terminals of minimum-length transistors directly to the pads. To avoid making significant changes to the main circuits at the last moment, we simply inserted a voltage divider between pads and the track and hold circuit, as shown in the overall front-end circuit diagram (Figure 40). The unsilicided poly resistors (on top of n-well) in series with input pads provided some level of ESD protection for minimum-length transistors MP1 and MP2.

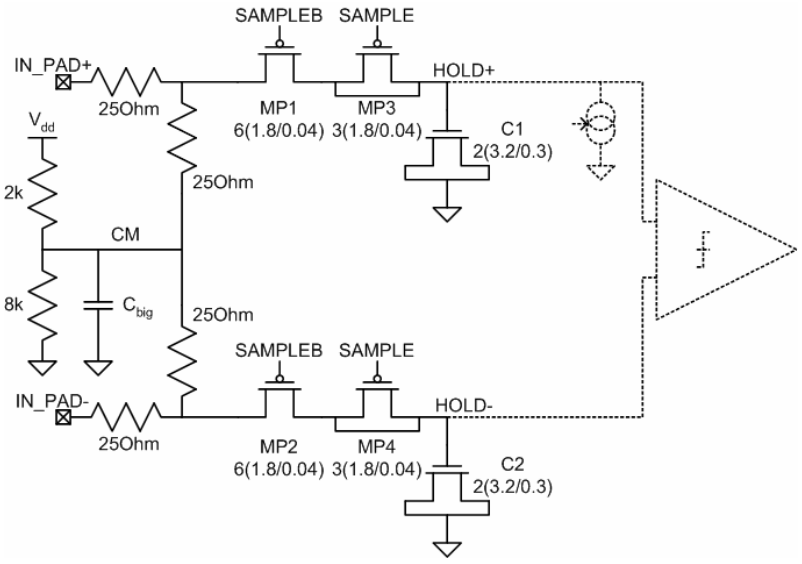


Figure 40. Track and hold front end with voltage divider termination

For testing the ADC functionality, this extra voltage divider attenuation has to be compensated by increasing the input signal amplitude. On the positive side, the voltage divider increases the bandwidth. In the track and hold circuit itself, note transistors MP3 and MP4 that perform clock feedthrough cancellation to avoid excessive shift down of common-mode voltage at the track and hold circuit output. Also note that MOS sampling capacitors were used, because for the expected range of voltage samples, they provide linear enough capacitance.

4.2.4. RAMP GENERATOR

After the input signal is sampled, either a capacitor on its positive side (for positive half-ADC) or a capacitor on its negative side (for negative half-ADC) is linearly discharged by a current source. This current source needs to be adjustable for two reasons – first, to provide coarse ADC gain control from 50mVp-p to 300mVp-p full-scale to match the input signal range, and second, to match the gains of the two half-ADCs (and possibly sub-ADCs in an interleaved structure, if interleaving is

used). Thus, the ramp generator current source is built of multiple individually switched current sources (an I-DAC), as shown in Figure 41.

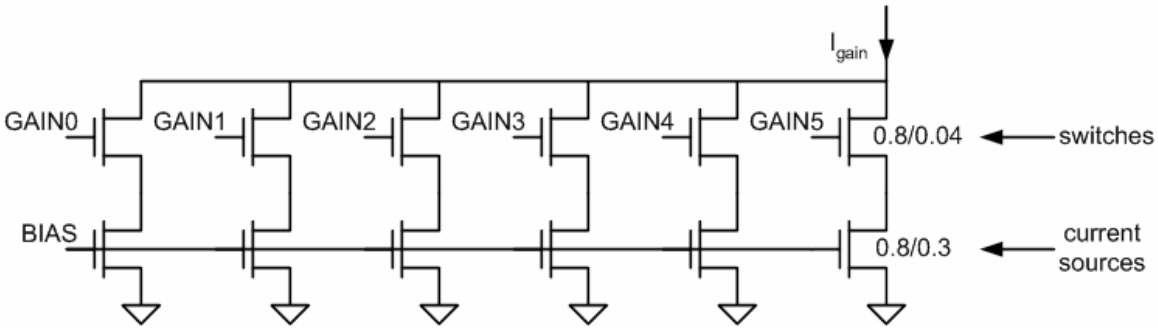


Figure 41. Current-mode digital-to-analog converter (I-DAC) for ADC gain control

While coarse gain adjustment is accomplished by choosing how many current sources are on, slight gain mismatch between the two half-ADCs is a lot harder to trim out, because there is a practical lower limit on the current source magnitude. One possibility is to build two separate I-DACs for coarse and fine adjustment and to merge their outputs with different weights. In this case, some fixed current would also have to be added to guarantee certain overdrive of current mirror transistors. However, such an elaborate structure would consume significant area and add unnecessary power consumption.

An interesting alternative is to use mismatch between current sources in the main I-DAC for fine adjustment. To trim out gain mismatch between the two half-ADCs, we keep constant the number N of current sources that are on, but change WHICH particular current sources are on. The number of such configurations for each half-ADC is $\frac{M!}{(M-N)!N!}$, where M is the total number of current sources (6 in this case). When only $N=1$ current source is on, the number of total configurations is 6. However, the number of possible configurations increases to 30 for $N=2$ current sources and to mind-boggling 120 for $N=3$. Thus, because there is always some

mismatch between individual current sources, we can choose a combination of current sources that matches the gains of two half-ADCs very accurately.

In a general-purpose ADC, the need for such gain adjustment would be problematic, as it would require a separate calibration procedure with known input voltages. However, in a high-speed link receiver, the ADC gain is adjusted adaptively anyway to match the swing of an input signal. Usually, the ADC gain is changed until a desired (small) percentage of input samples causes the ADC overrange. The same procedure carried out for both half-ADCs forces their gains to match. Furthermore, the gain mismatches between multiple interleaved ADCs can be trimmed out in exactly the same manner.

While the track and hold circuit is tracking, the ramp current source is turned off to avoid input voltage drop across the sampling switch. To keep current sources in saturation when SAMPLEB is 0, we use a differential current-steering switch (Figure 42).

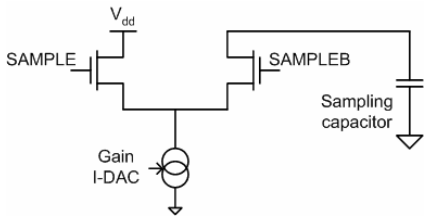


Figure 42. Differential switch for turning off ramp generator when track and hold is tracking

4.2.5. COMPARATOR

Once the input signal is sampled, and one of its sides (IN+ or IN-) is ramped down, it needs to be compared with the other side. Although regenerative clocked comparators are generally more efficient than continuous-time comparators, they cannot run with a 40ps clock period, required for the 40ps time resolution used in

this converter. A continuous-time comparator, on the other hand, continuously monitors its inputs with unlimited time resolution.

There are several considerations driving the design of the comparator. First, it needs to produce a sharp enough output transition to trigger all latches at the same time, regardless of their individual input-referred offsets. Second, variation of its delay with supply voltage should be minimized to prevent supply noise from translating into noise at the ADC output. Third, the comparator needs to have enough common-mode rejection to prevent variations in input common-mode from coupling to the ADC output. Fourth, the offset of the comparator needs to be digitally adjustable to match the offsets of the two half-ADCs (and multiple ADCs, if interleaving is employed). Finally, the comparator bandwidth needs to be maximized to minimize the comparator overhead added to the conversion time. In addition, the CMOS latches triggered by the comparator output require full-rail CMOS levels.

The need to reject the input common-mode voltage requires at least one differential stage at the input. At the same time, generating a full-rail CMOS output is hard to do with fully-differential circuits (at least using only one supply voltage), so the comparator output is essentially single-ended in nature. However, the comparator must amplify the differential signal enough first to guarantee that any shift in the threshold of the single-ended stage (likely from supply noise) does not cause significant change in the comparator delay. More formally, the rejection of ADC to input-referred noise of single-ended portion of the comparator is equal to the gain of differential portion of the comparator. Assuming about 20mVp-p supply noise, as measured in **[65]**, the differential gain must be at least 50 to guarantee that the input-referred noise is less than 0.4mV≈0.5LSB (for the smallest input range of 50mVp-p).

Although such gain can be implemented in a single amplifier stage, the bandwidth of such amplifier would be very limited. An approach that results in higher bandwidth is to cascade multiple low-gain stages instead. In fact, maximum bandwidth

is attained, as shown in **[64]**, when gain per stage is only $e^{0.5}\approx 1.6$, where e is the base of natural logarithm. However, since each differential stage continuously draws current, the total power consumption becomes large. Therefore, we take some bandwidth penalty and limit the number of differential gain stages (including differential to single-ended converter) to 3, as shown in Figure 43.

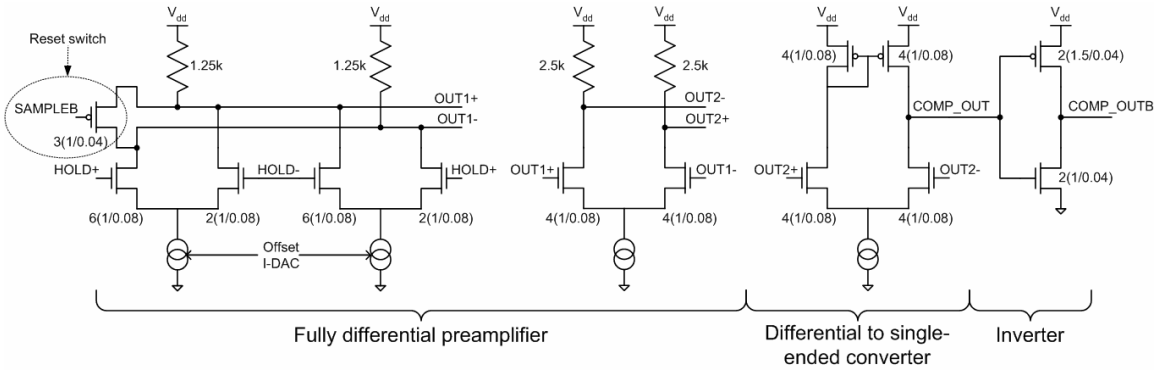


Figure 43. Comparator circuit

The first two stages are fully-differential amplifiers with resistive loads for lower parasitic capacitance. The third stage is a differential to single-ended converter with a current mirror load for best conversion efficiency - both differential halves are utilized to drive the output. The current mirror load has larger parasistics but also larger small-signal resistance than unsilicided poly resistors used in the first two stages. Therefore, although for optimum bandwidth the gain of all stages should be the same, in this case it makes sense to allocate more gain (about 8) to the differential to single-ended converter and less gain (about 2.5 per stage) to the fully-differential stages. The differential to single-ended converter is followed by an inverter that brings the output to full-scale CMOS levels and provides buffering for driving latches.

The first comparator stage incorporates a reset switch that equalizes nodes OUT1+ and OUT1+ to minimize history effect due to feedthrough from these

nodes back to the track and hold. While the first comparator stage is being reset, the later stages still have more time to finish processing the previous sample, effectively "borrowing" time from the next conversion cycle. A common practice of resetting all comparator stages at the same time would impose a hard boundary between the two conversion cycles, in this case increasing the comparator overhead. An interesting possibility would be to stagger the resets of the comparator stages in time to track the propagation of the signal through the comparator. However, this would be difficult to arrange across process, voltage, and temperature variations.

Note that relatively small transistors have poor matching, so the comparator has a large random offset. The first comparator stage is designed to digitally compensate this offset. This stage consists of two intentionally mismatched differential pairs, and the ratio of their tail currents controlled by a current-mode DAC varies the input-referred offset over wide enough range to counteract worst-case mismatch. Although this approach reduces the effective transconductance of the stage, its most common alternative, shown in Figure 44, pulls static current from the output nodes of the comparator, essentially also reducing the transconductance obtained for a given bias current budget.

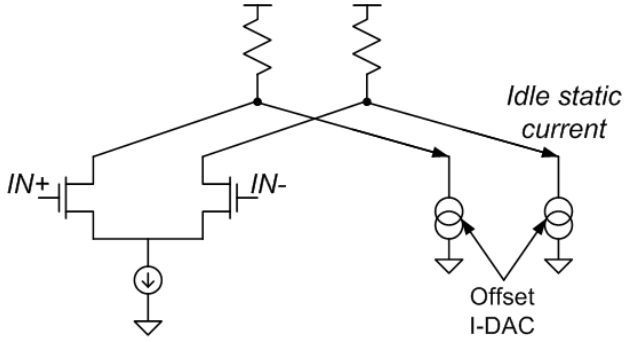


Figure 44. Varying comparator offset by pulling unequal currents from two output nodes

To reduce the impact of DAC element mismatch itself on resolution of offset cancellation, we incorporate certain redundancy in the DAC, as shown in Figure 45.

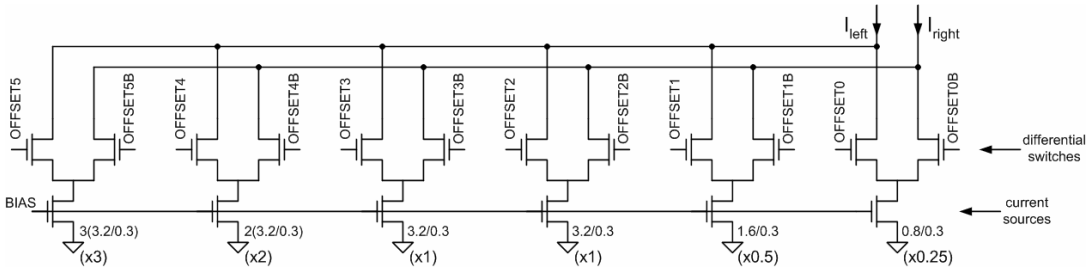


Figure 45. Current-mode DAC for offset compensation

While 3 LSBs have binary weights, 3 MSB weights are arranged so that even with mismatch, the precision of offset adjustment is never worse than LSB (x0.25 current source) + deviation of x1 current source from its nominal value. This is illustrated by the DAC transfer function for the 3 MSBs in Figure 46.

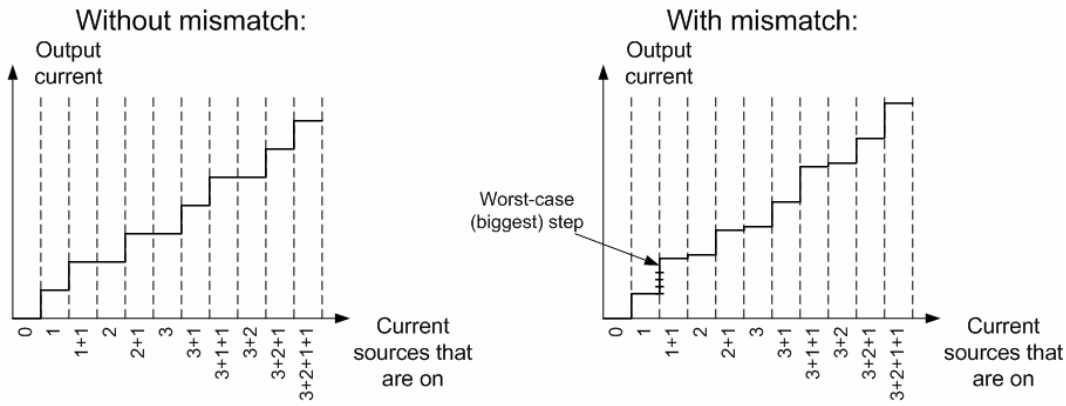


Figure 46. Transfer function of offset DAC with and without mismatch

Just like for gain trimming, a general-purpose ADC would require a special calibration procedure to measure the ADC offset. In a high-speed link context, however, the ADC range is automatically centered around the mid-point of the input signal by continuously tracking its statistics. The same technique can be used to zero out the offsets of the two half-ADCs and appropriately stitch their ranges together without any dead zone. Similar statistical technique is applicable to cancelling out the offset mismatches between the multiple interleaved ADCs.

4.2.6. OUTPUT LATCHES AND FLIP-FLOPS

When the comparator detects a crossover of ramped down node HOLD+ with constant HOLD-, it deasserts COMP_OUT and asserts COMP_OUTB commanding a set of latches to store the current counter value as an ADC conversion result. There is only one slight complication – if HOLD+ were already lower than HOLD-, the latches would not get enabled during the current conversion cycle at all and hold previous conversion result, as shown in Figure 47.

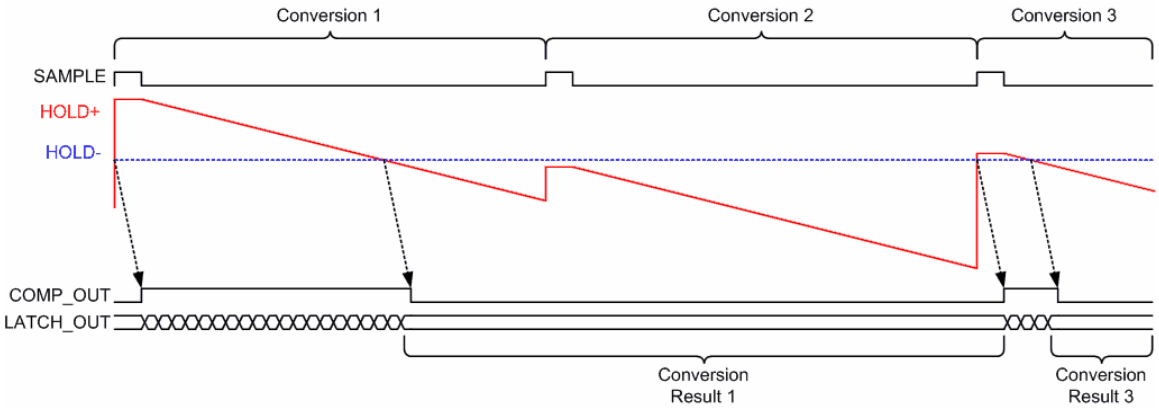


Figure 47. ADC operation without latch reset

In this case, from looking at the outputs of the latches, there is no way to tell if the conversion 2 result was the same as conversion 1 result, or if input voltage was out of ADC range. To solve this problem, we need to reset the latches to some known "out-of-range" state at the beginning of every conversion. The simplest way to accomplish this is to enable the latches for a fixed short time at the beginning of each conversion cycle, as shown in Figure 48.

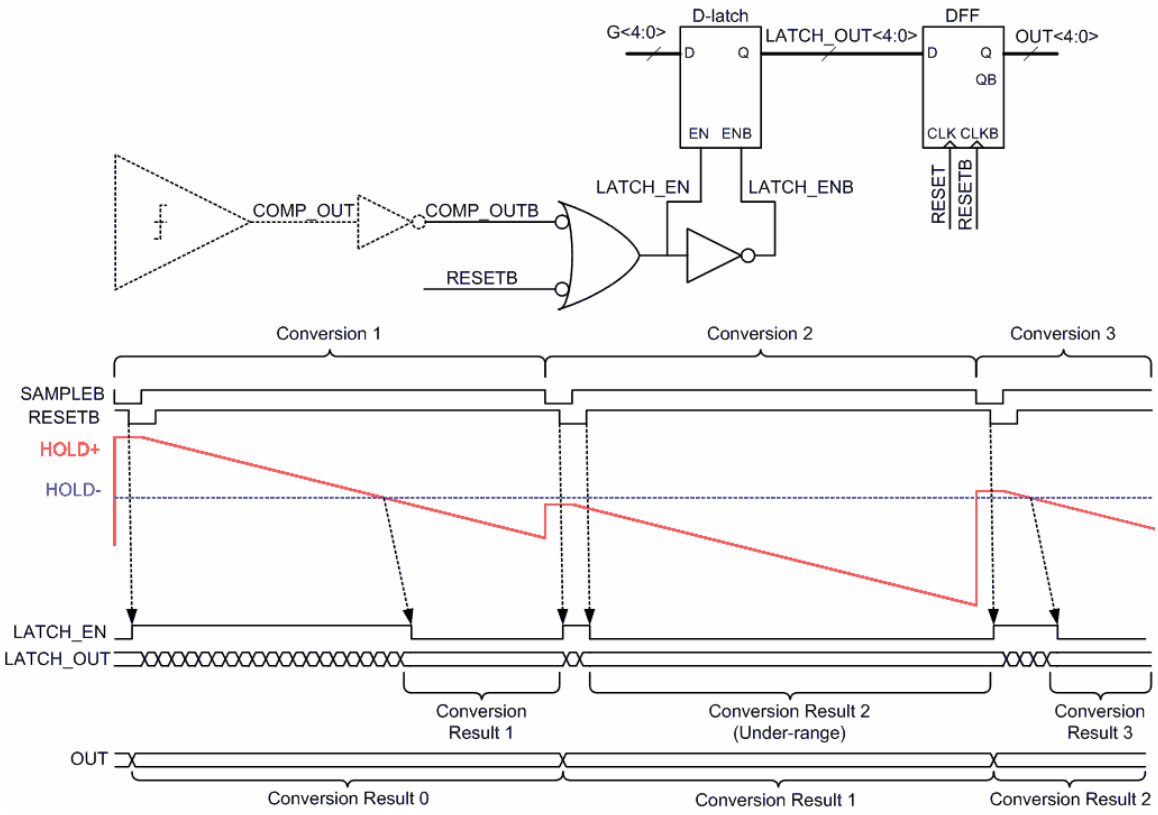


Figure 48. ADC operation with latch reset

Also shown is the bank of D-flip-flops that store the conversion results. But to output the results off-chip, we need to either build memory on-chip or decimate the data stream to a manageable bandwidth. On this chip, we have chosen decimation for its simplicity. Also, due to limited number of pins, we time-multiplex the results from two half-ADCs to the same set of pins as shown in Figure 49.

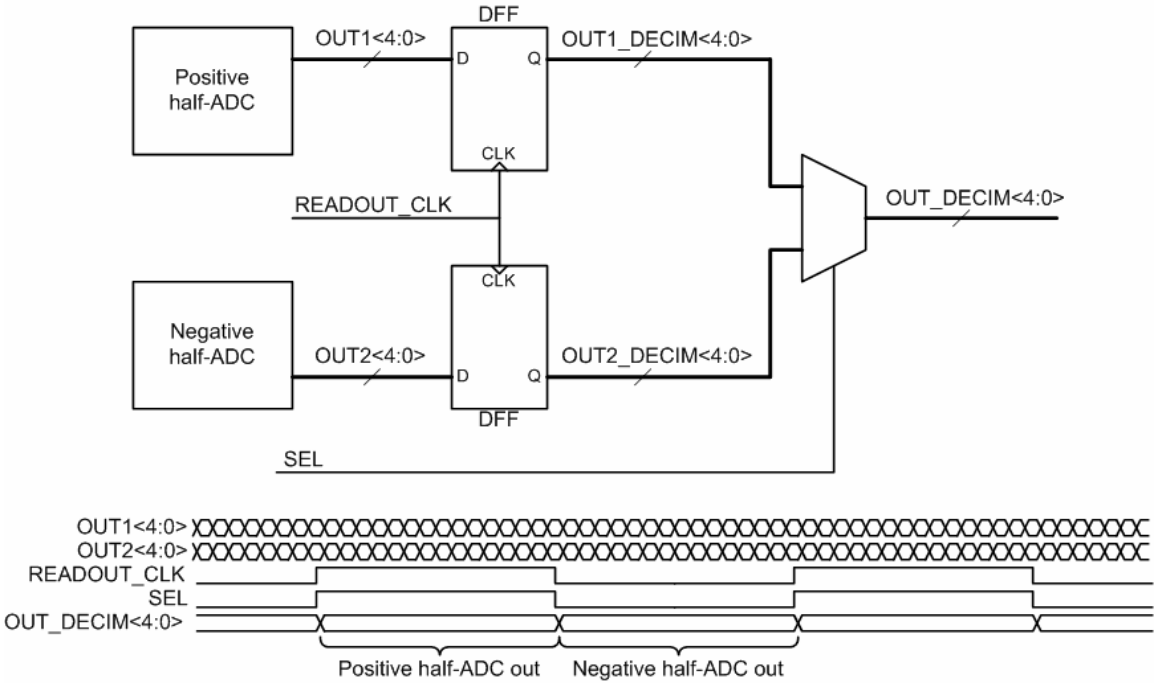


Figure 49. Output decimation and multiplexing

4.2.7. POWER EFFICIENCY ANALYSIS

Since the primary goal of this work is designing power-efficient ADCs, having described the entire system, let us now analyze its power consumption. In simulation this ADC consumes about 3mW of total power from a 1V supply.[1] Ideally it would provide 5 bits of resolution at 1.5625Gsps or 6 bits of resolution at 0.78125Gsps, but because of comparator overhead, the actual resolution is only around 20 levels instead of 32 levels for 5-bit mode and only around 52 levels instead of 64 levels for 6-bit mode. (Approximately 6 time units for each half-ADC are taken up by comparator delay and reset overhead.)

[1] Excluding PLL power, as PLL needs to be included on the chip anyway. The ADC presents about 80fF load to the clock tree, which results in clock power consumption of about **0.5mW**.

To explore the speed-power trade-off, let us break down the total power consumption by blocks as shown in Figure 50 and estimate how the power of each block scales with conversion frequency.

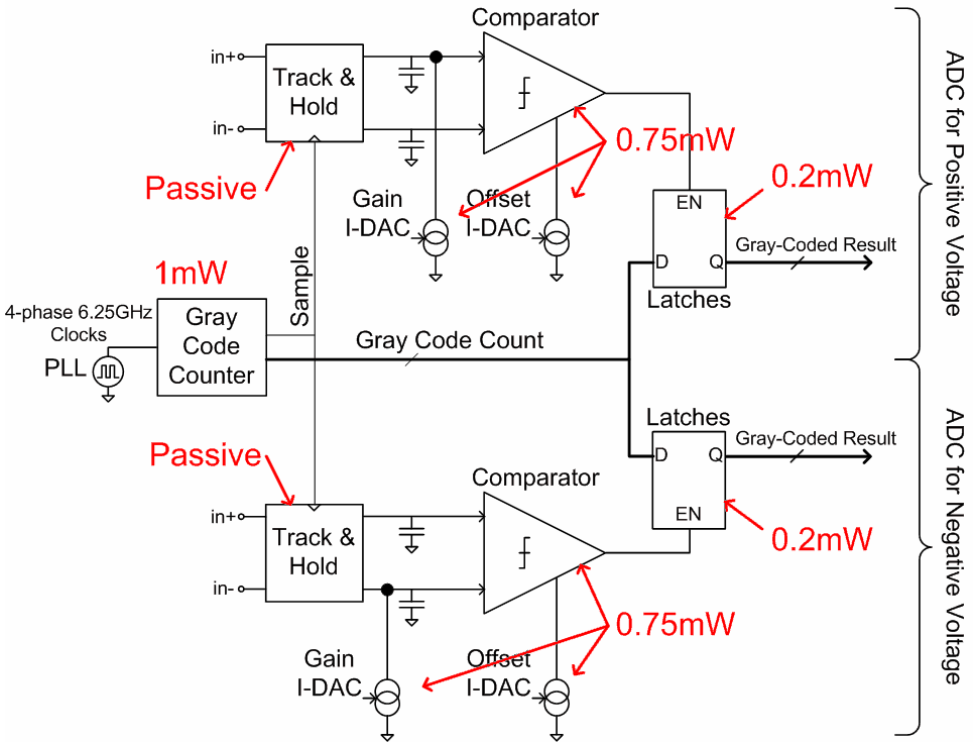


Figure 50. Power Consumption of Individual ADC Blocks

The set of output latches and registers for both sub-ADCs consumes a significant **0.4mW** of power. As mostly CV^2f -power, it is proportional to the frequency of Gray code transitions ($1/T_{LSB}$). Another digital circuit, a Gray code time-base generator itself, consumes even more substantial **1mW** power also proportional to $1/T_{LSB}$. We can assume T_{LSB} equal to the PLL phase step T_{step} as in this design, because this produces the highest conversion rate for a given power consumption. Thus, power consumed by the digital portion of the ADC is approximately

$$P_{dig} = \frac{E_{dig}}{T_{step}} = \frac{P_{dig0} \cdot T_{step0}}{T_{step}}, \tag{4.2}$$

where E_{dig} is energy consumed by digital circuits per T_{step}, and P_{dig0} is the simulated power consumption for nominal $T_{step0} = 40ps$.

The two comparators consume the largest percentage of total power – approximately **1.5mW**. If each comparator is scaled down in size by a factor $\alpha<1$ (transistor width W is decreased to αW, while length is kept constant; load resistors are increased from R to R/α, and tail bias currents are decreased from I to αI), its power consumption will be reduced at an expense of larger random offset and lower speed. While offset can be trimmed out digitally using DACs, lowering the comparator speed will limit the conversion rate. However, if power consumption can be improved significantly at an expense of only a small decrease in speed, it might be worth operating at a lower conversion rate with better power efficiency. To make this more specific, we simulated the comparator delay[2] as a function of its size for a fixed capacitive load of 5fF on the output of each stage to model interconnect parasitics.

A simple analytic delay model is RC-delay. Resistance R is the driving resistance of a comparator stage, and it scales as $1/\alpha$. Capacitance consists of two terms – a fixed parasitic capacitance C_p that does not scale and a self-load capacitance C that scales as α. Thus

[2] As we mentioned in Section 4.1. , the gain-bandwidth product of a comparator, not its delay, adds an overhead to the ADC conversion time. In general, gain-bandwidth product and delay can be completely decoupled (as, for example, in a distributed amplifier). However, for the chosen comparator topology, the bandwidth is inversely proportional to the delay, and delay is a good approximation for a comparator overhead.

$$t_{delay} = \frac{R_0}{\alpha}\left(C_p + \alpha C_0\right) = \frac{\tau_p}{\alpha} + \tau_0 . \tag{4.3}$$

From simulation results, the empirical parameters τ_p and τ_0 were found to be $\tau_p \approx 40ps$ and $\tau_0 \approx 91ps$ for the comparator gain of ~100. Delays found both from simulation and from this model are plotted in Figure 51.

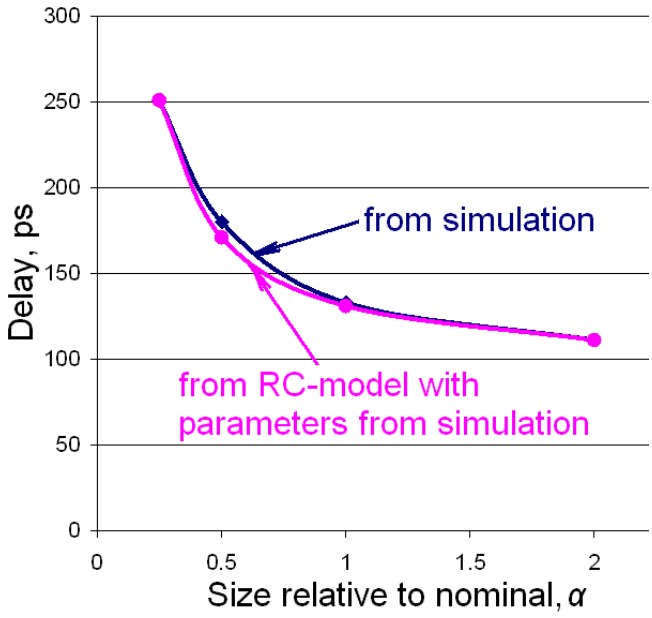


Figure 51. Comparator delay as a function of its size

It is obvious from the plot that as we increase comparator size, its speed does not improve beyond a certain point. So as we try to increase ADC speed beyond 1-2Gsps, a fixed comparator overhead of at least 100ps will make it impractical. On the other hand, if we can afford to decrease the conversion rate and thus allow slower comparator, comparator power

$$P_{comp} = P_{comp0}\frac{\tau_p}{t_{delay} - \tau_0} = \frac{E_{comp}}{t_{delay} - \tau_0} \tag{4.4}$$

(where P_{comp0} = 1.5mW is the power consumption of nominal-size comparators, and E_{comp} is the proportionality constant that has units of energy) can be significantly scaled down.

In that case, to achieve the required throughput, we have to interleave a larger number of ADCs. This raises the question about what interleave factor and individual ADC speed result in the lowest total power. Suppose that we need to sample an input signal every T_{sample}, but each ADC takes T_{conv} to convert, so we need to interleave $N = T_{conv}/T_{sample} = (T_{step} \cdot N_{levels} / 2 + t_{delay}) / T_{sample}$ ADCs. The total power consumed by the interleaved structure is then approximately

$$P_{total} \approx N(P_{comp} + P_{dig}) = N\left(\frac{E_{comp}}{t_{delay} - \tau_0} + \frac{E_{dig}}{T_{step}}\right) = N\left(\frac{E_{comp}}{NT_{sample} - N_{levels}T_{step}/2 - \tau_0} + \frac{E_{dig}}{T_{step}}\right). \quad (4.5)$$

This function is plotted versus interleave factor N in Figure 52 for $T_{sample} = 40ps$, $E_{comp} = P_{comp0} \cdot \tau_p \approx 1.5mW \cdot 40ps$, $E_{dig} = P_{dig0} \cdot T_{step0} \approx (1mW + 0.4mW) \cdot 40ps$, for N_{levels} = 20 (left plot) and N_{levels} = 52 (right plot). Here we have assumed for simplicity that a separate Gray counter is included in each ADC slice. In reality, one Gray counter can be shared by all interleaved ADCs. This would not reduce counter's power consumption significantly, however, since most of the power is required to drive the outputs, loading on which increases proportionally with the number of interleaved ADCs.

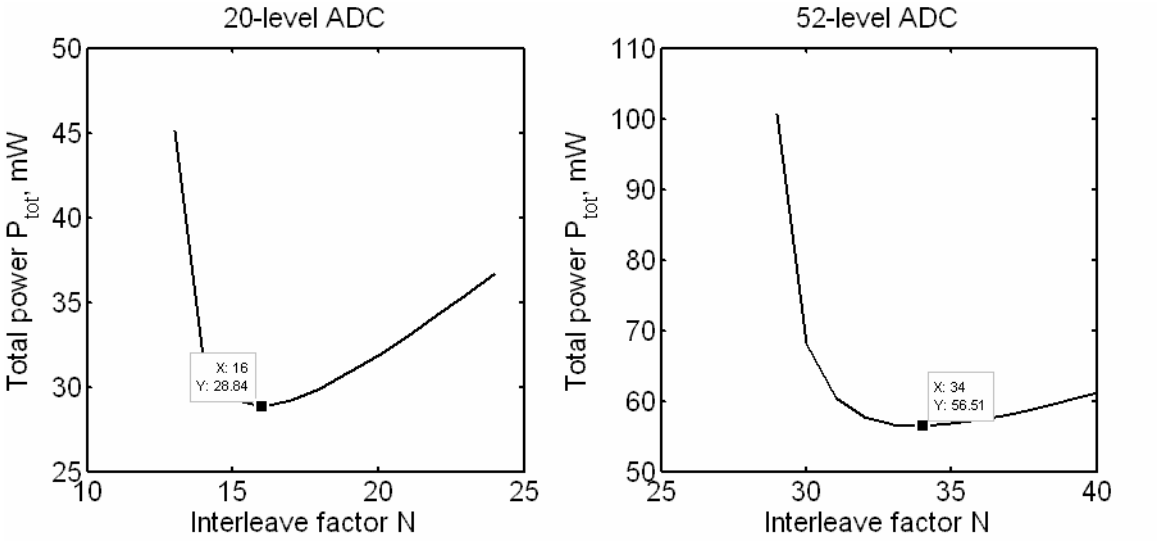


Figure 52. Total power consumption of an interleaved ADC running at T_{sample} = 40*ps* (25Gsps) for 20-level / 4.3 bit (left) and for 52-level / 5.7 bit (right) conversions

Although these power estimates are only approximate, we can see that there is an optimum number of interleaved ADCs. For 20-level conversion, for example, interleaving 16 individual ADCs results in the lowest total power consumption of 29mW. If we try to achieve the 25Gsps aggregate conversion rate with a smaller number of individual ADCs, faster comparator will come at an extremely large power cost. On the other hand, if too many ADCs are interleaved, the power saved by making comparator slower will not offset the increasing digital power. Interestingly, as it turned out, exactly 16 interleaved instances of our 20-level 1.5625Gsps ADCs will result in approximately the lowest power consumption. In high-resolution mode, 32 interleaved 52-level 0.78125Gsps will also be very close to the optimum interleaved design.

4.3. TEST RESULTS

The die photo of the single-slope ADC test chip is shown in Figure 53(a). As the top metal fill pattern made circuit features invisible, we scraped off the

upper metal layers, revealing layouts of the major circuit blocks, apparent in Figure 53(b). The corresponding fragment of the layout from a layout editor is shown in Figure 53(c).

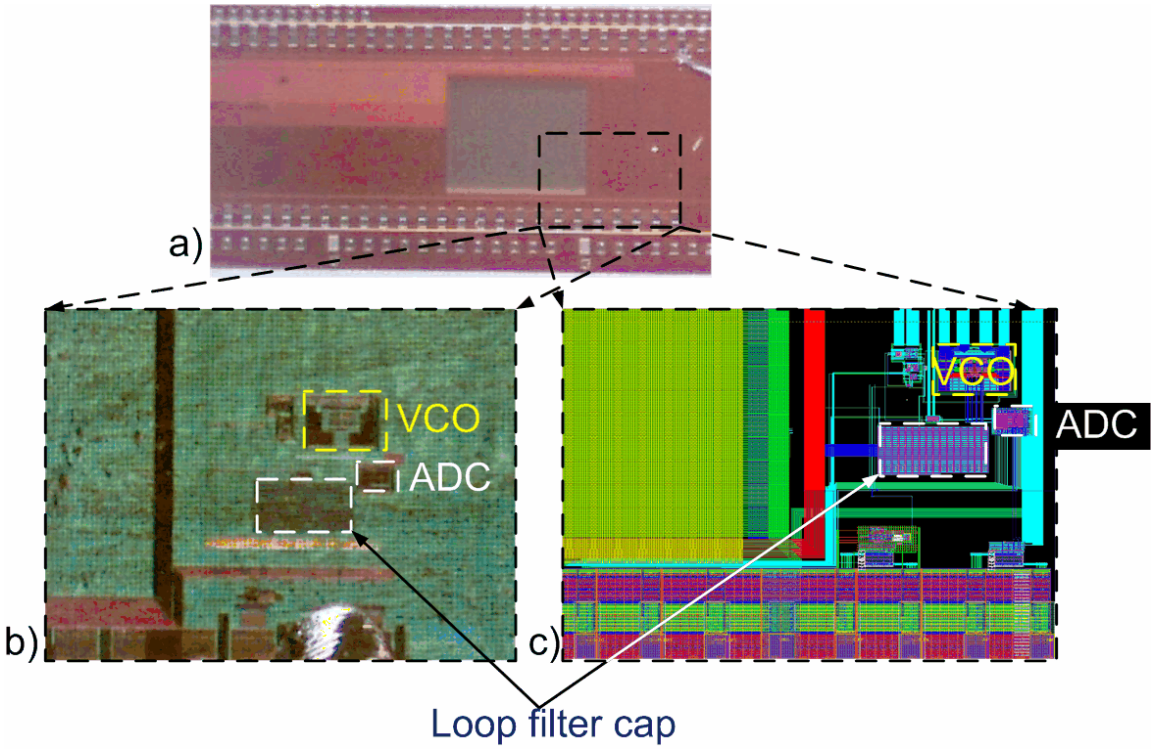


Figure 53. Die photos (a, b) and corresponding layout editor view (c) of the single-slope ADC test chip

Although the test chip was designed for TI's 45nm high-performance CMOS technology, the fabrication run was cancelled due to changes in TI's corporate strategy. Instead, the test chip was eventually fabricated without redesign in a low-power 45nm technology at TSMC. The corresponding shift in device parameters limited PLL frequency to 3.2GHz instead of the nominal 6.25GHz. Therefore, the ADC was limited to operating at 400Gsps in the 6-bit mode and at 800Gsps in the 5-bit mode. Other than this, the ADC worked mostly as expected.

4.3.1. TEST SETUP

The block diagram of the test setup is shown in Figure 54.

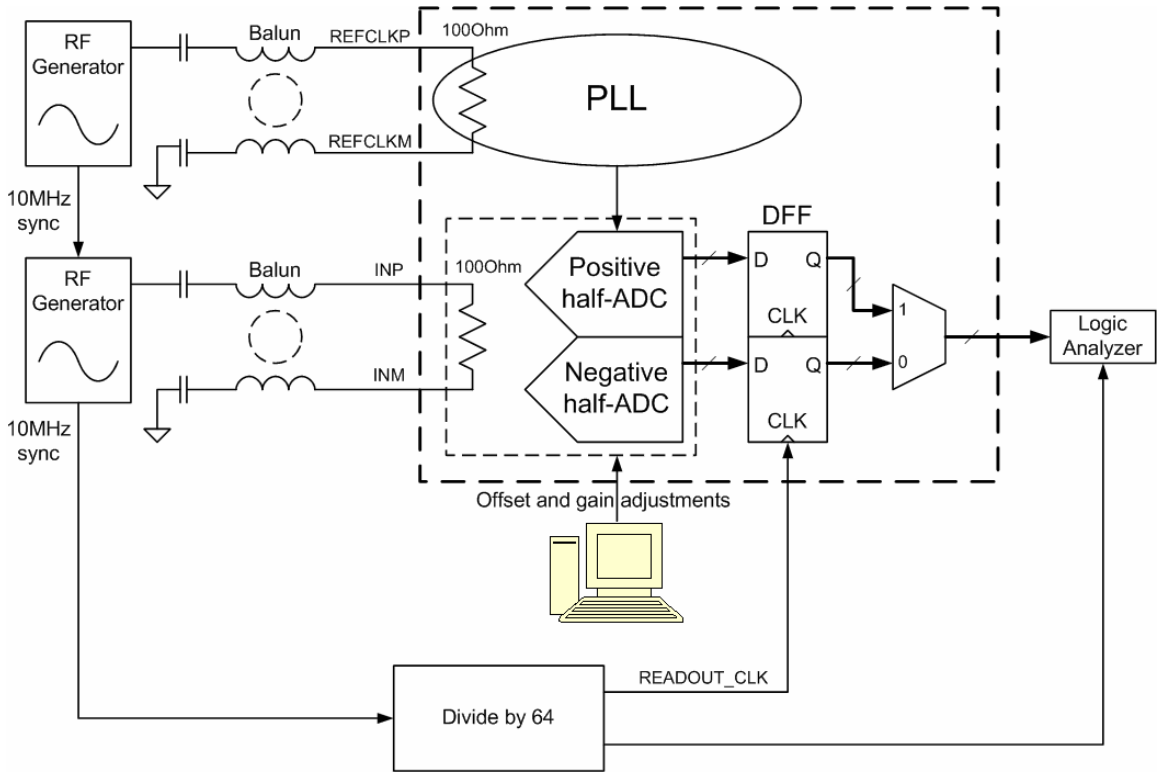


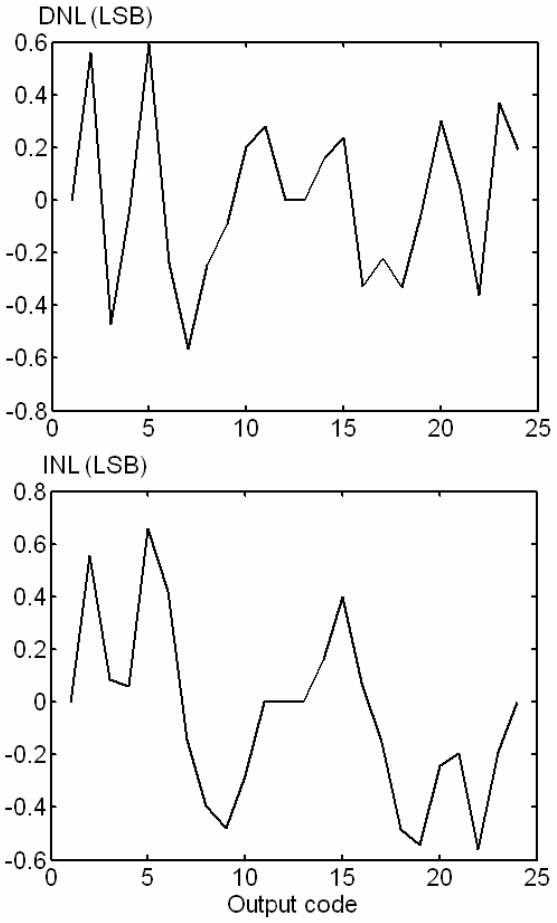
Figure 54. Chip test setup

The first RF signal source generated the reference clock for the PLL. Its single-ended output was converted to a differential signal using balancer-unbalancer (balun) on the test board. The on-die PLL was configured to multiply the reference frequency by 20, requiring a 160MHz reference clock for the maximum achievable conversion rate. The second RF signal source, locked to the first one using the instruments' 10MHz synchronization ports, generated the sinusoidal signal input to the ADC. The RF source's single-ended output was again converted to differential with an on-board balun. The distortion of the resulting differential signal was negligible compared to the expected distortion of the ADC itself.

The clocks for decimating the ADC outputs and selecting the results from either positive or negative half-ADC were derived from the RF sources' 10MHz sync outputs. To minimize noise from switching the output pins of the chip unnecessarily often, the 10MHz sync signal was divided down by 64 using an off-chip counter to obtain the 156.25kHz decimation clock.

4.3.2. STATIC LINEARITY MEASUREMENTS

The static integral and differential non-linearity (INL and DNL) were measured using a code density test **[31]** with a sufficiently low-frequency (100MHz) sinusoidal input. The amplitude of the sinusoid (400mV in the 5-bit mode and 600mV in the 6-bit mode) was significantly larger than the ADC input voltage range (approximately 100mV), so that non-linear top and bottom regions of the sinusoid were clipped. The probability density function (pdf) of the remaining portions was flat enough to make correcting for the non-uniform pdf of the sinusoid unnecessary – the DNL error resulting from non-linearity of the sinusoid was less than 0.04 LSB. The plots of the measured DNL and INL of the ADC are shown in Figure 55. Note that since the code density measurements were performed separately for positive and negative half-ADCs (and then joined in Figure 55), the gain and offset matching between them was implicit in post-processing. Separate gain and offset adjustments for the half-ADCs using on-chip DACs was exercised, however, as part of dynamic measurements, discussed in Section 4.3.3.


DNL (LSB)
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
0
5
10
15
20
25
INL (LSB)
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
0
5
10
15
20
25
Output code

a)

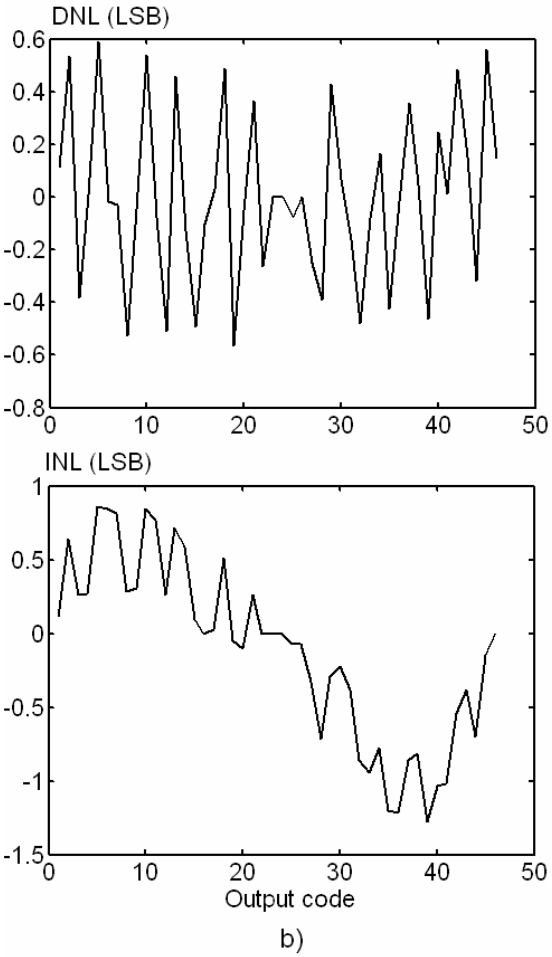


Figure 55. Differential (DNL) and integral (INL) non-linearity (a) in 5-bit 800Msps mode (b) in 6-bit 400Msps mode

As apparent from the plot, some of the codes were idle – the overhead due to T/H tracking phase and the finite bandwidth of the comparator. The overhead was quite significant: 8 codes out of 32 in a 5-bit mode and 18 codes out of 64 in a 6-bit mode. Therefore, the real number of bits was $\log_2(2^5 - 8) = 4.6$ in the 5-bit mode and $\log_2(2^6 - 18) = 5.5$ in the 6-bit mode. The differential non-linearity exhibited repetitive patterns, possibly due to timing mismatches between multiple clock phases. Nevertheless, DNL did not exceed 0.6LSB.

The integral non-linearity was as large as ±1LSB in a 6-bit mode, which prompted further investigation. As it turned out, INL was worse for lower comparator bias current and improved for higher bias current. As lower bias current meant lower gain-bandwidth product of the comparator, we attributed the non-linearity to malformed latch pulses produced by the comparator at high input voltages, as described in Section 4.1. Note that saturation of either the track-and-hold or the ramp generator circuits would have produced gain compression, which results in the opposite sign of non-linearity.

Based on DNL and INL measurements, we conclude that the two important improvements to ADC design would be more careful matching between the clock phases (using high-performance flip-flops instead of standard cells) and a comparator design with higher gain-bandwidth product.

4.3.3. DYNAMIC ADC PERFORMANCE

The dynamic performance of the ADC was characterized by measuring its signal to noise and distortion ratio (SNDR) as a function of sinusoidal input frequency. Since the output of the ADC was decimated to 156.25ksps, high-frequency sinusoids were subsampled at that frequency. To make sure that taking a fast Fourier transform (FFT) did not result in spectral leakage, the number of samples taken, $N_{samples}$, spanned an integer number of input signal cycles, N_{cycles}. More formally, the frequency of the input sinusoid, $f_{\sin}$, was chosen according to the relationship

$$f_{\sin} = \left(M + \frac{N_{cycles}}{N_{samples}} \right) f_{subsample} \ , \tag{4.6}$$

where M is the integer subsampling ratio, $M = \text{int}\left[\frac{f_{\sin}}{f_{subsample}}\right]$. N_{cycles} and $N_{samples}$ were also chosen to be mutually prime, 11 and 16384 respectively, to whiten the quantization noise.

Although no non-linearity correction was performed for each half-ADC, matching the gains and offsets of the two halves was critical to achieving acceptable spectral purity. The offsets were trimmed out by adjusting the on-chip offset DAC settings until 50% of all samples fell into the operating region of positive half-ADC and 50% of all samples fell into the operating region of negative half-ADC. In this way, the input range of the ADC was centered around the midpoint of the sinusoid. Then the gain DAC settings were adjusted until the input signal occupied about 95% of the operating ranges of both positive and negative half-ADCs. Although this algorithm was employed for a sinusoidal input, it would also work for DC-balanced data in a high-speed link.

After these adjustments, the outputs of the two half-ADCs were combined for post-processing. The resulting SNDR as a function of input frequency is shown in Figure 56.

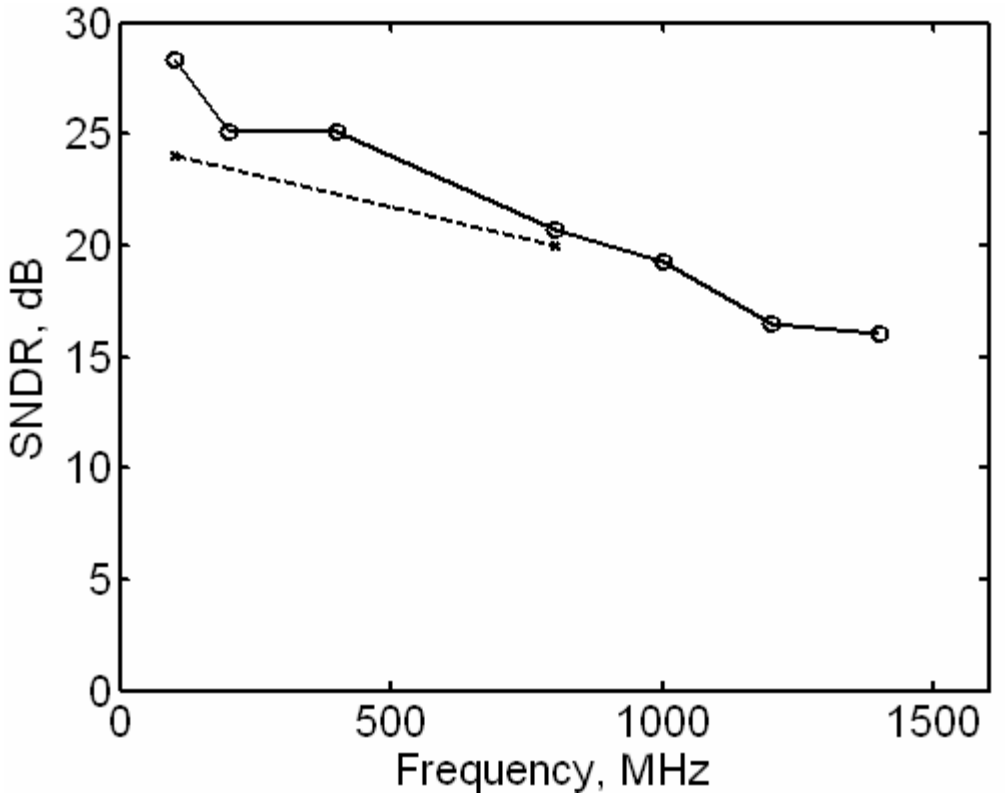


Figure 56. SNDR as a function of input signal frequency in 6-bit 400Msps mode (solid line) and 5-bit 800Msps mode (dashed line)

As apparent from the figure, ADC performance deteriorated with frequency, due to both clock jitter and dynamic distortion. Although the SNDR roll-off occurs at high enough frequencies for Nyquist-rate operation, interleaving requires higher so-called effective resolution bandwidth (ERBW) – the frequency, at which SNDR rolls off by 3dB. The ERBW of this ADC was 200MHz in the 6-bit mode and 800MHz in the 5-bit mode.

4.4. SUMMARY AND FUTURE WORK

In this chapter, a high-speed single-slope architecture was proposed. With a specific implementation of this architecture, we have shown it to be a feasible

alternative to more traditional high-speed architectures. Furthermore, small area and input capacitance of a single-slope ADC make it a good choice for an interleaved front-end of high-speed link receivers.

However, in a practical high-speed single-slope ADC, the overhead associated with the comparator turned out to be more significant than originally thought. Apart from improving the comparator design, the overhead can be reduced by interleaving a larger number of slower ADCs. Therefore, it would be advantageous to use one ADC, offset to cover the entire input voltage range, rather than two half-ADCs. It also becomes more important to minimize the digital power consumption to increase the optimal number of interleaved ADCs. Apart from improvements described above, future work includes building an interleaved ADC based on single-slope converters to achieve very high aggregate conversion rates.

CHAPTER 5. CONCLUSIONS

The primary goal of this work was to choose and optimize or develop an ADC topology for a high-speed link receiver. Such an ADC is required to resolve about 4-5 bits while operating at baud rates of the links, which are currently moving towards tens of Gbps. While the most conventional high-speed ADC architecture, FLASH, have been shown to be very efficient for the target resolution, it presents major difficulties with scaling a conversion rate beyond a certain point (currently several GSps in 90→45nm CMOS technologies). The difficulties are primarily due to limited gain-bandwidth product of the comparators and large input capacitance of the parallel comparator array that hinders interleaving.

Both pipeline and successive approximation ADCs, although generally much slower than FLASH, are more suitable for interleaving, allowing to achieve tens of GSps conversion rates. However, they tend to be more efficient for higher resolutions while being overly complex for 4-5 bits. Thus, modern high-speed links have the requirements that neither FLASH nor interleaved pipeline/SAR ADCs can meet in their optimal regime of operation.

These considerations have led us to the development of a high-speed single-slope ADC architecture specifically with high-speed link receiver as a target application. Like FLASH, this architecture works best at a relatively low resolution, 4-6 bits. At the same time, single-slope ADC occupies a small die area, has a small input capacitance, and contains a built-in track-and-hold, facilitating interleaving for high aggregate conversion rates. Without any linearity correction, single-slope ADCs are linear enough for 4-6 bits of resolution.

Leveraging the specificity of a high-speed link environment also solves some issues that single-slope ADCs do present. Although linear, single-slope ADC

does require offset and gain adjustments. However, in a high-speed link receiver, such adjustments can be carried out as part of a normal receiver adaptation based on signal statistics. A single-slope ADC also requires an accurate timing reference with a fine step size, much finer than the sampling period of an individual ADC. However, such timing reference must already be present in a link receiver with multiple single-slope ADCs interleaved for baud-rate conversions.

Our results indicate that a single-slope ADC, as an architecture tailored specifically for high-speed links, is capable of performance superior to that of more conventional ADC techniques applied to the same problem. This is an example of how novel circuit topologies, developed from scratch to leverage the specificity of a target application, can be well worth the risk and additional development time.

BIBLIOGRAPHY

[1] B. Casper *et al.*, "Future Microprocessor Interfaces - Analysis, Design, and Optimization," *IEEE Custom Integrated Circuits Conference*, pp. 479-486, Sept. 2007.

[2] T. Beukerna *et al.,* "A 6.4Gb/s CMOS SerDes Core with Feed-Forward and Decision Feedback Equalization," *IEEE J. Solid-State Circuits*, vol. 40, no. 12, pp. 2633–2645, Dec. 2005.

[3] K.T. Oshiro *et. al.*, "A 10-Gbps 83 mW GaAs HBT Equalizer/Detector for Coaxial Cable Channels," *IEEE Custom Integrated Circuits Conference*, pp. 347-350, Sept. 1998.

[4] A. J. Baker, "An Adaptive Cable Equalizer for Serial Digital Video Rates to 400Mb/s," *ISSCC Digest of Technical Papers*, pp. 174-175, Feb. 1996.

[5] M. H. Shakiba, "A 2.5Gb/s Adaptive Cable Equalizer," *ISSCC Digest of Technical Papers*, pp. 396-483, Feb. 1999.

[6] Jong-Sang Choi *et al.*, "A 0.18-μm CMOS 3.5-Gb/s continuous-time adaptive cable equalizer using enhanced low-frequency gain control method," *IEEE J. Solid-State Circuits*, vol. 39, no. 3, pp. 419–425, Mar. 2004.

[7] S. Gondi *et. al.*, "A 10Gb/s CMOS adaptive equalizer for backplane applications," *ISSCC Digest of Technical Papers*, pp. 328-601, Feb. 2005.

[8] G. Zhang and M. Green, "A 10 Gb/s BiCMOS Adaptive Cable Equalizer," *IEEE J. of Solid -State Circuits*, vol. 40, no. 11, pp. 2132-2140, Nov. 2005.

[9] J. Lee, "A 20Gb/s Adaptive Equalizer in 0.13μm CMOS Technology", *ISSCC Digest of Technical Papers*, pp. 273-282, Feb. 2006.

[10] B. Casper *et al.,* "A 20Gb/s Forwarded Clock Transceiver in 90nm CMOS," *ISSCC Digest of Technical Papers*, pp. 90-91, Feb. 2006.

[11] R. Palmer *et al.*, "A 14mW 6.25Gb/s Transceiver in 90nm CMOS for Serial Chip-to-Chip Communications," *ISSCC Dig. of Tech. Papers,* vol. 50, pp. 440-441, Feb. 2007.

[12] G. Balamurugan *et. al.*, "A Scalable 5-15Gbps, 14-75mW Low-Power I/O Transceiver in 65nm CMOS," *IEEE Symposium on VLSI Circuits*, vol. 43, pp. 270-271, June 2007.

[13] M. Harwood *et al.*, "A 12.5Gb/s SerDes in 65nm CMOS Using a Baud-Rate ADC with Digital Receiver Equalization and Clock Recovery," *ISSCC Dig. of Tech. Papers*, vol. 50, pp. 436-591, Feb. 2007.

[14] J. Zerbe *et al.*, "Equalization and Clock Recovery for a 2.5-10-Gb/s 2-PAM/4-PAM Backplane Transceiver Cell," *IEEE J. Solid-State Circuits*, vol. 38, no. 12, pp. 2121-2130, Dec. 2003.

[15] V. Stojanovic *et al.*, "Autonomous Dual-Mode (PAM2/4) Serial Link Transceiver With Adaptive Equalization and Data Recovery," *IEEE J. Solid-State Circuits*, vol. 40, no. 4, pp. 1012-1026, April 2005.

[16] E. L. Ginzton, W. R. Hewlett, J. H. Jasberg, and J. D. Noe, "Distributed Amplification," *Proc. IRE*, pp. 956-969, August 1948.

[17] G. McIver, "A Travelling Wave Transistor", *Proc. IEEE,* vol. 53, pp.1747-1748, Nov. 1965.

[18] E. H. Kopp, "A coupled mode analysis of the traveling-wave transistor," *Proc. IEEE (Letters),* vol. 54, pp. 1571-1572, Nov. 1966.

[19] W. Jutzi, "Uniform distribuled amplifier analysis with fast and slow waves," *Proc. IEEE(Letters),* vol. 56, pg. 6647, Jan. 1968.

[20] G. Kohn and R. Landauer, "Distributed Field-Effect Amplifiers", *Proc. IEEE,* vol. 56, pp.1136-1137, Jan. 1968.

[21] M.Farina and T. Rozzi, "A Theory of Distributed Amplifiers Exploiting Growing Waves," *Microwave Symposium Digest*, vol. 2, pp. 913-918, June 1997.

[22] W. Heinrich, "Distributed Equivalent-Circuit Model for Traveling-Wave FET Design," *IEEE Trans. Microwave Theory Tech.,* vol. MTT-35, no.5, pp. 487-491, May 1987.

[23] W. Heinrich, "Wave Propagation along GaAs-FET Electrodes," *Journal of Electromagnetic Waves and Application,* vol. 5, no. 4-5, pp. 403-417, 1991.

[24] A.J. Holden *et al.,* "Gallium Arsenide Travelling-Wave Field-Effect Transistor," *IEEE Trans. Electron Devices,* vol. ED-32, pp. 61-66, 1985.

[25] Behzad Razavi, *Principles of Data Conversion System Design*, IEEE Press, New York, 1995.

[26] Mikael Gustavsson *et al.*, *MOS Data Converters for Communications,* Kluwer Academic Publishers, Boston, 2000.

[27] R. Poujois and J. Borel, "A low drift fully integrated MOSFET operational amplifier," *IEEE J. Solid-State Circuits*, vol. 13, pp. 499 - 503, August 1978.

[28] J.T. Wu *et al.*, "A 100-MHz pipelined CMOS comparator," *IEEE J. Solid-State Circuits*, vol. 23, pp. 1379 - 1385, December 1988.

[29] D. W. Dobberpuhl, "Circuits and technology for Digital's StrongARM and ALPHA microprocessors [CMOS technology]," *17th IEEE Conf. Advanced Research in VLSI*, pp. 2–11, 1997.

[30] B. Zojer, K. Petschacher, and W. A. Luschnig, "A 6-bit/200-MHz Full Nyquist A/D converter," *IEEE J. Solid-state Circuits*, vol. SC-20, no. 3, pp. 780-786, June 1985.

[31] J. Doernberg, H. S. Lee, and D. A. Hodges, "Full-speed testing of A/D converters," *IEEE J. Solid-State Circuits*, vol. SC-19, no. 6, pp. 820-827, Dec. 1984.

[32] C. W. Mangelsdorf, "Improving bandwidth and error rate in flash converters," *IEEE Symposium on VLSI Circuits*, pp. 53-54, 1989.

[33] D. G. Knierim, "Thermometer-to-adjacent binary encoder," U.S. Patent no. 4733220, Mar. 22, 1988.

[34] C. L. Portmann and T. Meng, "Power-Efficient Metastability Error Reduction in CMOS Flash A/D Converters," *IEEE J. Solid-State Circuits*, vol. 31, no. 8, pp. 1132-1140, Aug. 1996.

[35] P. Scholtens and M. Vertregt, "A 6-bit 1.6-GSample/s flash ADC in 0.18-µm CMOS using averaging termination," *IEEE J. Solid-State Circuits*, vol. 37, no. 12, pp. 1599–1609, Dec. 2002.

[36] G. Van der Plas, S. Decoutere, and S. Donnay, "A 0.16pJ/conversion-step 2.5mW 1.25GS/s 4b ADC in a 90nm digital CMOS process," *IEEE ISSCC Dig. Tech. Papers*, vol. 49, pp. 566–567, Feb. 2006.

[37] B. Verbruggen *et al.*, "A 2.2mW 5b 1.75GS/s Folding Flash ADC in 90nm Digital CMOS," *ISSCC Dig. Tech. Papers*, vol. 51, pp. 252-253, Feb. 2008.

[38] J. Yuan and C. Svensson, "A 10-bit 5-MS/s successive approximation ADC cell used in a 70-MS/s ADC array in 1.2-µm CMOS," *IEEE J. Solid-State Circuits*, vol. 29, no. 8, pp. 866–872, Aug. 1994.

[39] D. Draxelmayr, "A 6b 600MHz 10mW ADC Array in Digital 90nm CMOS", *ISSCC Dig. of Tech. Papers*, pp. 264-265, Feb. 2004.

[40] B.P. Ginsburg and A.P. Chandrakasan, " Dual Time-Interleaved Successive Approximation Register ADCs for an Ultra-Wideband Receiver," *IEEE J. Solid-State Circuits*, vol. 42, no. 2, pp. 247 - 257, Feb. 2007.

[41] B.P. Ginsburg and A.P. Chandrakasan, "500-MS/s 5-bit ADC in 65-nm CMOS With Split Capacitor Array DAC," *IEEE J. of Solid-State Circuits*, vol. 42, no. 4, pp. 739 - 747, Apr. 2007.

[42] P. Schvan *et. al.*, "A 24GS/s 6b ADC in 90nm CMOS," *ISSCC Dig. of Tech. Papers,* pp. 544-545, Feb. 2008.

[43] K. Poulton, R. Neff, B. Setterberg, *et al.*,"A 20 GS/s 8b ADC with a 1MB memory in 0.18mm CMOS", *ISSCC Dig. of Tech. Papers,* pp. 318-319, Feb. 2003.

[44] S. G. Gupta *et al.*, "A 1 GS/s 11 b time-interleaved ADC in 0.13 µm CMOS," *ISSCC Dig. of Tech. Papers*, vol. 49, pp. 576–577, Feb. 2006.

[45] S. J. Jamal *et al.*, "A 10-bit 120-Msample/s time-interleaved analog-to-digital converter with digital background calibration," *IEEE J. Solid-State Circuits*, vol. 37, no. 12, pp. 1618–1627, Dec. 2002.

[46] D. Groeneveld *et al.*, "A self-calibration technique for monolithic high-resolution D/A converters," *IEEE J. Solid-State Circuits*, vol. 24, pp. 1517–1522, Dec. 1989.

[47] J. Bastos *et al.*, "A 12-bit intrinsic accuracy high-speed CMOS DAC," *IEEE J. Solid-State Circuits*, vol. 33, pp. 1959–1969, Dec. 1998.

[48] G. Van der Plas *et al.*, "A 14-bit intrinsic accuracy Q random-walk CMOS DAC," *IEEE J. Solid-State Circuits*, vol. 34, pp. 1708–1718, Dec. 1999.

[49] A. Bugeja *et al.*, "A 14-b 100-MS/s CMOS DAC designed for spectral performance," *IEEE J. Solid-State Circuits*, vol. 34, pp. 1719–1732, Dec. 1999.

[50] A. Bugeja and B.-S. Song, "A Self-Trimming 14-b 100-MS/s CMOS DAC," *IEEE J. Solid-State Circuits*, vol. 35, pp. 1841-1852, Dec. 2000.

[51] K. Doris *et al.*, *Wide-Bandwidth High Dynamic Range D/A Converters*, Springer 2006.

[52] A. Van den Bosch *et al.*, *Static and Dynamic Performance Limitations for High Speed D/A Converters*, Kluwer Academic Publishers, Boston, 2004.

[53] J. Savoj *et. al.*, "A New Technique for Characterization of Digital-to-Analog Converters in High-Speed Systems," *Design, Automation & Test in Europe Conference & Exhibition*, pp. 1-6, Apr. 2007.

[54] J. Savoj *et. al.*, "A 12-GS/s Phase-Calibrated CMOS Digital-to-Analog Converter," *IEEE Symposium on VLSI Circuits*, pp. 68-69, June 2007.

[55] C.-H. Lin and K. Bult, "A 10-b, 500-MSample/s CMOS DAC in 0.6 mm^2," *IEEE J. Solid-State Circuits,* vol. 33, pp. 1948–1958, Dec. 1998.

[56] A. van den Bosch *et. al.*, "A 10-bit 1-GSample/s Nyquist current-steering CMOS D/A converter," *IEEE J. Solid-State Circuits,* vol. 36, pp. 315-324, Mar. 2001.

[57] K. Doris *et. al.*, "A 12b 500MS/s DAC with > 70dB SFDR up to 120MHz in 0.18um CMOS," *ISSCC Dig. Tech. Papers*, pp. 116-117, Feb. 2005.

[58] W. Schofield *et. al.,* "A 16b 400MS/s DAC with <-80dBc IMD to 300MHz and <-160dBm/Hz noise power spectral density," *ISSCC Dig. Tech. Papers*, pp. 126-127, Feb. 2003.

[59] B. Schafferer and R. Adams, "A 3V CMOS 400mW 14b 1.4GS/s DAC for multi-carrier applications," *ISSCC Dig. Tech. Papers*, pp. 360-361, Feb. 2004.

[60] B. Jewett *et. al.*, "A 1.2GS/s 15b DAC for precision signal generation," *ISSCC Dig. Tech. Papers*, pp. 110-111, Feb. 2005.

[61] S. S. Mohan, M. del Mar Hershenson, S. P. Boyd, T. H. Lee, "Bandwidth Extension in CMOS with Optimized On-Chip Inductors," *IEEE J. Solid -State Circuits*, vol. 35, Mar. 2000.

[62] K. H. Mueller and M. Muller, "Timing recovery in digital synchronous data receivers," *IEEE Trans. Commun.*, vol. COM-24, pp. 516–530, 1976.

[63] M. Shinagawa *et al.*, "Jitter analysis of high-speed sampling systems", *IEEE J. of Solid-State Circuits*, vol. 25, pp. 220-224, Feb. 1990.

[64] T. H. Lee, *The Design of CMOS Radio-Frequency Integrated Circuits*. Cambridge, U.K.: Cambridge Univ. Press, 1998.

[65] E. Alon, "Measurement and Regulation of On-Chip Power Supply Noise," *Ph.D. Thesis*, Dec. 2006.

[66] T. Toifl *et al.*, "A 22-Gb/s PAM-4 Receiver in 90-nm CMOS SOI Technology," IEEE J. Solid-State Circuits, vol. 41, no. 4, pp. 954-965, Apr. 2006.