Clocking Analysis, Implementation and Measurement Techniques for High-Speed Data Links—A Tutorial

Bryan Casper, *Member, IEEE*, and Frank O'Mahony, *Member, IEEE*

(Invited Paper)

Abstract—**The performance of high-speed wireline data links depend crucially on the quality and precision of their clocking infrastructure. For future applications, such as microprocessor systems that require terabytes/s of aggregate bandwidth, signaling system designers will have to become even more aware of detailed clock design tradeoffs in order to jointly optimize I/O power, bandwidth, reliability, silicon area and testability. The goal of this tutorial is to assist I/O circuit and system designers in developing intuitive and practical understanding of I/O clocking tradeoffs at all levels of the link hierarchy from the circuit-level implementation to system-level architecture.**

Index Terms—**Clock distribution, clock recovery, high-speed I/O, phase-locked loops.**

I. Introduction

TECHNOLOGY to enable digital transmission of data has transformed today's society by facilitating the widespread use of advanced computer and communication systems. Digital communications technology as well as integrated circuit scaling trends has enabled the industry to dramatically scale the bandwidth of high-loss networks such as DSL and Ethernet. Many of these networks are channel bandwidth limited and have had to leverage sophisticated equalization techniques to push well beyond the uncompensated channel bandwidth.

High-bandwidth shorter-length networks are becoming increasingly important in modern computer systems for the effective transfer of information between microprocessors, memory, peripheral components, network hubs, storage devices, etc. In the past, interfaces for microprocessor systems were primarily circuit limited due to process technology bandwidth constraints. As microprocessor system interface data rates have grown, the channel has started to hamper performance. To alleviate this bottleneck, microprocessor interfaces have adopted advanced communication techniques and optimized interconnect topologies previously used only in longer-haul networks. These advanced communications methods include such things as linear equalization, decision feedback equalization (DFE), modulation and coding techniques. Additionally, interconnect improvements such as transmission line termination and adoption of point-to-point topologies have helped enhance the performance of modern data links.

Manuscript received September 17, 2008. Current version published February 4, 2009. This paper was recommended by Associate Editor W. A. Serdijn.

The authors are with Circuit Research Laboratory, Intel Corporation, Hillsboro, OR 97124 USA (e-mail: bryan.k.casper@intel.com).

Digital Object Identifier 10.1109/TCSI.2008.931647

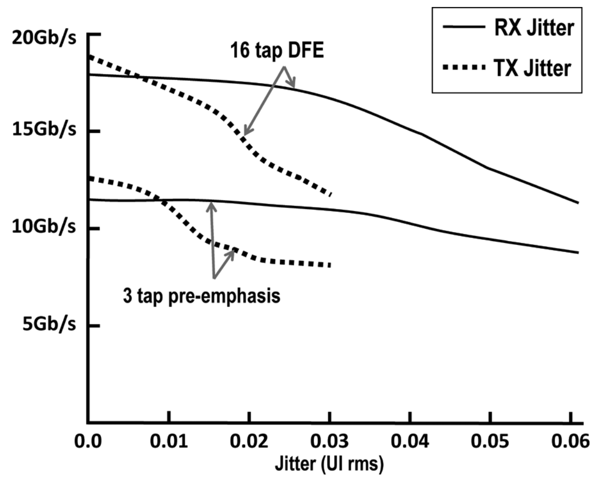


Fig. 1. Maximum data rate as a function of TX jitter magnitude for a data link with varying amounts of equalization complexity. Baseline parameters for link simulation include: Channel loss 15 dB@5 GHz, BER $= 10^{-12}$. 16 tap DFE link includes 3 tap TX pre-emphasis. RX input-referred noise $= 1$ mV rms. TX jitter $= 1/2$ ps rms, RX jitter $= 1$ ps rms unless otherwise specified. Jitter is normally distributed.

While these breakthroughs have helped advance the state-of-the-art in system interfaces, improved clocking methods have played a central role in the widespread adoption of multi-gigabit data links. Past methods of data link clocking such as common clock or synchronous architectures relied on a central clocking source as the synchronization signal to retime the outbound data at the transmitter (TX) as well as sample the input data at the receiver (RX). The maximum data rate of these legacy topologies was severely limited by clock interconnect bandwidth, uncompensated clock skew, channel latency, and jitter. The introduction of methods such as clock multiplication, forwarded clock recovery [1], embedded clock recovery [2], per-pin deskew [3], jitter filtering, and duty-cycle error correction have increased maximum data rates from a few hundred megabits per second (Mb/s) to many gigabits per second (Gb/s).

Well designed multi-Gb/s data link architectures employ a combination of advanced equalization and precise clocking to balance the goals of performance, power efficiency, and cost. Architectures overly focused on equalization of the channel and not sufficiently optimized for clock quality may suffer from sub-optimum tradeoffs between complexity, power efficiency and data rate [4]. Fig. 1 demonstrates the high sensitivity of jitter on maximum data rate using TX only equalization (3 tap pre-emphasis) or TX and RX equalization (16 tap DFE). This example demonstrates that jitter at the TX or RX has a significant impact on maximum achievable data rates. In this case, the maximum data rate is as sensitive to high-frequency TX jitter as it is to equalization choice or RX sampling jitter [9]. Practically, RX jitter also impacts I/O performance since long term RX sampling jitter magnitudes usually exceed that of the TX high frequency jitter. Therefore, it is essential that a proportional amount of design effort as well as power budget is focused on methods to reduce both TX and RX jitter magnitude.

Over the last few decades, process technology scaling has benefitted high-speed data link interfaces by providing increased transistor bandwidth, greater density and enhanced functionality. However, microprocessor interface design in particular is facing significant challenges, partly due to the need for integration on process technology optimized for digital functionality. Since a large portion of microprocessor silicon area is dominated by digital functionality and is usually optimized for cost, there is a tendency to limit the process feature set used for analog and mixed-signal functionality. In many cases, modern data link interfaces have to be designed with poor quality resistors, capacitors, and inductors while the transistors have suboptimum analog characteristics in terms of output impedance or matching. As the integration of on-die analog and digital functionality increases, power supply noise and its effect on jitter becomes progressively more significant. Additionally, system level considerations such as aggressive power management features, which are becoming increasingly common, can severely degrade supply noise and corresponding clock quality. In summary, historical scaling of transistor density has benefitted I/O bandwidth by enabling wider interfaces, more advanced equalization and higher data rates; however, the challenges associated with increased integration and bandwidth demands are causing ever more challenging conditions to optimize clock quality.

The intent of this paper is to introduce the fundamentals required for high-speed data link clocking design and analysis. We begin by introducing the most commonly implemented clock architectures required for synthesis, distribution and recovery of I/O clocks. Specifically, Section II will focus on two of the most important classes of I/O clock architectures: forwarded clock, embedded clock, and their corresponding variants. We will also introduce standard I/O clocking terminology in Section III by describing and defining essential clock quality and jitter metrics. Section IV provides an introduction to clock and jitter characterization techniques using both on-die integrated and external measurement methods. Section V of this tutorial paper will detail implementation aspects of high-speed I/O clocking design and we will overview specific methods and circuits used to perform clock synthesis, distribution and recovery. Section VI will conclude by discussing data link modeling techniques and demonstrating some examples of clock system sensitivities and interactions using advanced I/O modeling methods.

II. Forwarded and Embedded Clock Architectures

As mentioned in the introduction, common clock architectures were frequently used in past generations of data link interfaces. The challenge of this type of architecture was that I/O cycle time was limited by the sum of TX clock-to-q time, channel latency, RX setup time, and TX to RX clock skew and jitter. These timing constraints limited I/O data rates to a few hundred Mb/s. More refined clocking techniques to scale I/O rates beyond a few Gb/s have been referred to by some as “serial I/O” clocking schemes. However, this term is a misnomer in that these more advanced techniques are not only applied to narrow, serial interfaces but they may also be leveraged for wide, parallel interfaces with equivalent per-pin rates at much higher aggregate bandwidths. The two primary classes of multi-Gb/s clock architectures that embody these advanced clocking techniques are forwarded clock and embedded clock.

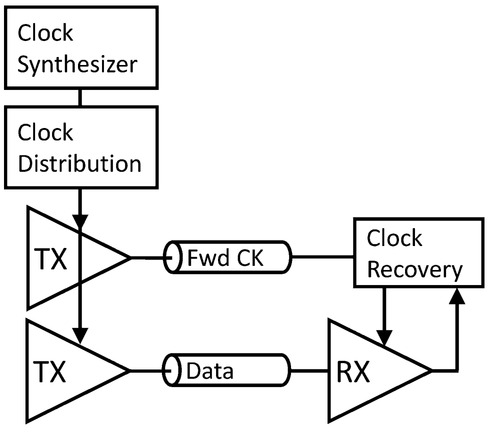


Fig. 2. Forwarded clock architecture.

A. Forwarded Clock (FC) Architectures

A diagram of the FC architecture is shown in Fig. 2. FC, sometimes referred to more specifically as source-synchronous clocking, uses a dedicated clock link sent from the TX to the RX. This architecture is most often used in wide, high aggregate bandwidth links where the cost and power overhead of the FC circuits are amortized across multiple links in the system. In many cases, FC architectures utilize matching between the clock and data circuits to minimize the impact of transmit induced jitter. In the ideal scenario, the FC TX and data TX share a common design and are synchronized by an identical synthesizer and clock distribution tree. This will not only save TX power and area, it also provides TX jitter tracking between data and the corresponding RX sampling clock. Practically, matched FC and data latency is usually not perfectly achieved which results in a degradation of clock recovery bandwidth. However, it is usually feasible to constrain latency mismatch to enable recovery bandwidths of hundreds of MHz [5].

The FC recovery unit attempts to center the receiver sample at the optimum point as measured by operating margin (i.e., time or voltage margin) or bit error rate (BER). Most FC recovery implementations allow for arbitrary length mismatch between the FC and data lines by detecting the optimum sampling phase and appropriately shifting the FC by the optimum amount. If the TX clock and data circuits are adequately matched and tolerant to voltage or temperature fluctuations, it may be sufficient to optimize the RX phase only during initialization or at some periodic interval. During the training period, it is necessary for

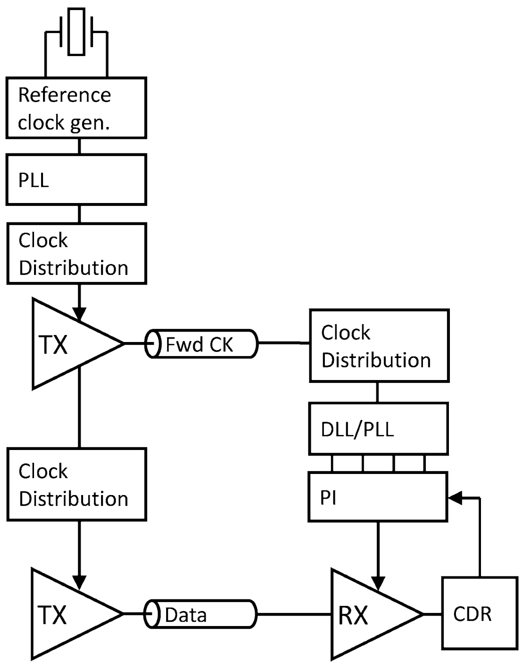


Fig. 3. DLL/PLL-based forwarded clock architecture.

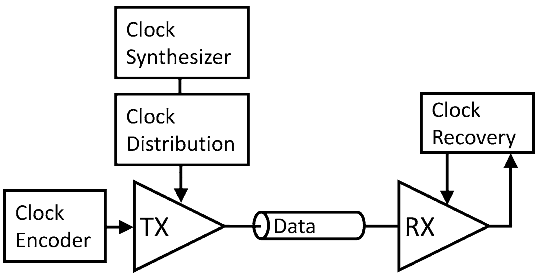


Fig. 4. Embedded clock architecture.

the data TX to send a pattern with a high enough edge probability to provide sufficient eye diagram shape information to the clock recovery unit. The periodicity of retraining is determined by the maximum frequency jitter or delay variation component to be compensated in either the TX, channel or RX. Some FC architectures use a continuous recovery scheme in an attempt to maximize the clock recovery bandwidth. In this case, the data TX must perform clock edge encoding of the data using a transition coding method (e.g., 8B/10B). Periodic recovery may also be referred to as per-pin deskew and is frequently utilized due to the advantages of low power and complexity without the burden of continuous clock edge encoding of data.

There are multiple categories of FC techniques; however, in this paper we discuss two of the most significant and readily used implementations.

DLL-Based Forwarded Clocking: FC recovery based on a delay-locked loop (DLL) circuit represents a very simple yet robust implementation. Fig. 3 demonstrates a specific example of a DLL-based recovery mechanism that receives the FC using a distribution network followed by a DLL which acts as a multiphase clock generator. The DLL and phase interpolator (PI) combination produces an adjustable phase clock generator with the same frequency as the incoming FC. As described previously, the clock and data recovery (CDR) logic adjusts the phase either continuously or periodically to center the RX sample at the optimum eye position. Advantages of DLLs are that they may be more simple, robust and stable than other types of multiphase clock generators such as phase-locked loops (PLL). Unlike a PLL, DLLs don't exhibit accumulation of internal phase errors. Additionally, DLLs facilitate high bandwidth tracking of the FC since they act as an all-pass clock phase filter which doesn't add appreciable latency to the clock path. However, the downside to this all-pass phase characteristic is that FC jitter is not only passed through to the PI but can also be amplified due to the finite bandwidth of the DLL delay line [4].

PLL-Based Forwarded Clocking: Clock recovery using a PLL rather than a DLL may provide advantages for applications that are sensitive to FC jitter amplification. Jitter amplification occurs as a result of high frequency jitter being exaggerated due to the lossy characteristics of the FC channel. Since a PLL has a low-pass phase transfer characteristic, the high frequency portion of the jitter that is not correlated to the data channel jitter is rejected, potentially resulting in superior clock recovery performance. However, this low-pass phase transfer characteristic may diminish useful jitter components (i.e., those that are correlated to the data channel jitter) which could result in suboptimum performance and lower clock recovery bandwidth. PLLs also have other disadvantages such as susceptibility to jitter accumulation and stability issues. Additionally, PLLs are usually more area intensive and complex than the equivalent DLL-based recovery solution [5], [6].

B. Embedded Clock (EC) Architecture

Some data links only require a narrow interface due to low aggregate bandwidth requirements of the targeted application. In cases such as this it is common to use EC recovery, such as that shown in Fig. 4, which doesn't incur the overhead of the FC link. EC interfaces also have the advantage of being highly modular since all TX and RX clocking circuitry may be fully contained within each transceiver cell. This *serial* link design method allows an arbitrary number of links to be instantiated in a design without significant overhead since there is little or no globally shared link circuitry. However, there are also many instances in which EC architectures are optimized for *parallel* operation that allow cross-link sharing of clock synthesizers, distribution and recovery circuits which saves power and area at the cost of design modularity. All EC based links require timing information to be encoded in the data to ensure a high probability of data edge transitions. Examples of data edge encoding include Manchester encoding, 8B/10B, 64B/66B, or even statistical coding methods such as pseudorandom bit sequence (PRBS) scrambling.

As with FC architectures, there are numerous varieties of EC recovery links. We focus this tutorial on two of the most common and useful architectures used for many years throughout the industry.

PI-Based (or Mixer-Based) Embedded Clocking: The RX clock recovery method for the PI-based architecture aligns the incoming data phase with a globally or locally synthesized clock using a frequency mixer circuit. In most cases, this frequency mixer is implemented as a multi-phase clock generator followed

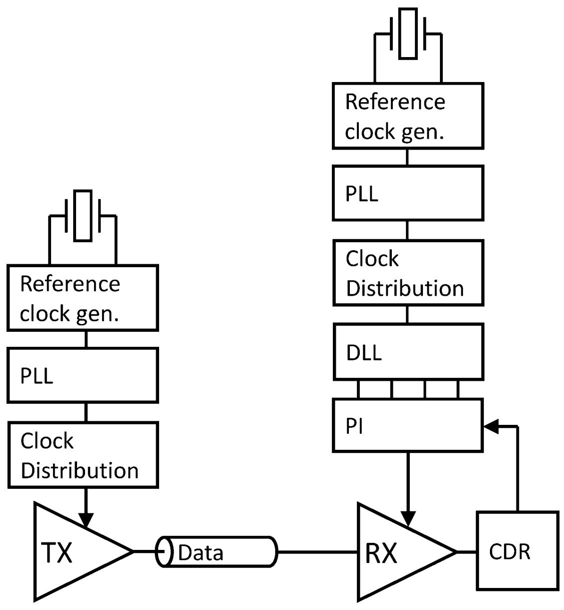


Fig. 5. PI-based (or mixer-based) embedded clock architecture.

by a PI and corresponding control logic. In the most common implementations of this architecture, the RX detects the phase of the data based on a phase detector which produces a binary indication of whether the input phase is before or after the sampling clock phase [2]. To determine the relative edge position of random data, it is necessary to sample the data edge along with the state of the data before and after the edge. This results in an RX architecture that must sample at twice the symbol rate, resulting in higher power, area and complexity than the equivalent FC architecture that doesn't necessarily require real-time extraction of edge information.

PI-based clock recovery is advantageous in area sensitive applications since much of the RX circuitry (in addition to the TX circuitry) shown in Fig. 5 may be shared globally across multiple links. For example, the PLL, DLL and distribution to produce the multi-phase clocks for the PI may be shared between adjacent links. Additionally, the CDR loop filter that controls the PI can be implemented using digital techniques. The combination of these features enables area efficient implementation. For this reason, this architecture is frequently leveraged for highly parallel EC applications. Another advantage of a globally shared PLL at the RX (as well as the TX) is that a higher proportion of power, area and design resources may be allocated for the central clock source since the overhead is amortized across multiple links. Moreover, any improvement in the clock quality of a globally shared resource usually enjoys a larger return on investment when compared with resources allocated to multiple instantiations of local circuitry.

There are multiple variations of the architecture shown in Fig. 5, some of which include generating multiple phases directly from either a DLL or a PLL whether it be local or global. Advantages of the embodiment shown in Fig. 5 is that only a single clock phase need be distributed to multiple links rather than distributing multiple clock phases directly from a global PLL with the associated power overhead and phase mismatch susceptibility. Furthermore, not only does a local DLL provide

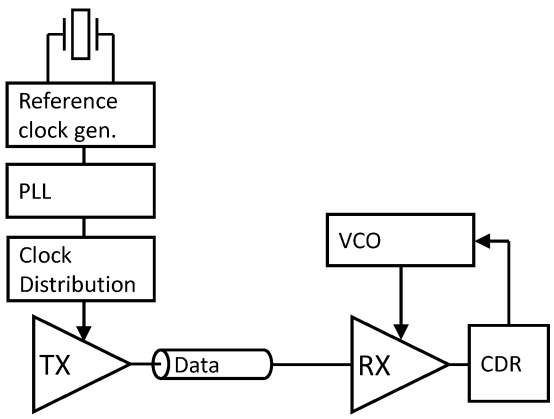


Fig. 6. VCO-based (or PLL-based) embedded clock architecture.

buffer bias settings for the delay line but it also may control neighboring clock buffers and the PI which will minimize the jitter and delay variation.

A key performance concern of the PI-based architecture is associated with PI output clock quality. Several nonidealities are the result of such PI clock quality issues including but not limited to: input phase non-linearity, input amplitude inconsistency and PI control feed-forward noise. Because of the susceptibility to these non-idealities, much of the design effort is usually focused on the PI design as well as the multi-phase clock generator [7].

Another design consideration for the PI-based architecture is the impact of two separate clock domains at the TX and RX. The architecture of Fig. 5 uses two different clock generators that may not have exactly the same frequency output. Because of this, the PI must act as a type of frequency mixer to align both the phase and frequency of the incoming data and the RX sampling clock. This frequency misalignment can be corrected but it usually results in additional jitter due to having to constantly rotate the PI output phase to emulate a frequency shift. Moreover, the CDR design may become more complex due to the addition of an integral controller in the feedback loop. An alternative embodiment of the architecture shown in Fig. 5 is to utilize the same crystal oscillator and reference clock generator for both the TX and RX. This reduces or eliminates frequency mismatch issues but the PI must still compensate for phase drift between the TX and RX PLLs.

VCO-Based (or PLL-Based) Embedded Clocking: This architecture uses a dedicated RX side voltage controlled oscillator (VCO) as the central clock recovery component as shown in Fig. 6. It is sometimes referred to as PLL-based clocking since it uses a VCO as the clock source, the RX functioning as a phase detector and the CDR circuit acting as a loop filter. The combination of these components forms a loop that locks to the mean phase of the incoming data. The fact that the loop must lock to a reference source that may not have consistent edge transitions is what causes this architecture to deviate from a conventional PLL design.

A primary motivation of VCO-based clocking use is due to the performance advantages of locking the RX clock synthesizer directly to the TX clock domain. Because of this, there is no requirement for intermediate frequency mixing or interpolation that may cause jitter and a corresponding performance degradation. Furthermore, any update to the phase of the RX sampling clock is filtered through the VCO loop filter which results in a minimum amount of clock glitches and jitter. As mentioned previously, another advantage to a VCO-based architecture is that the RX clocking is instantiated locally which permits design modularity.

There are also several drawbacks associated with the VCO-based clock topology. Like a conventional PLL, this architecture requires a loop filter (within the CDR in Fig. 6). A portion of the loop filter is frequently implemented using a capacitor which can consume a large area. Given that the VCO and loop filter is local and is not shared by neighboring links, cost of implementation may be high. Additionally, VCO-based clocking in the presence of multiple adjacent links may be subject to interference and coupling effects known as injection locking. Injection locking may cause significant jitter or even functional issues if not carefully considered.

III. Clock Quality and Jitter Terminology

As illustrated in Fig. 1, jitter can have a dominant impact on high-speed link performance. To optimize the performance of aggressive high-speed data links, it is crucial that designers and system architects understand jitter terminology and metrics as well as comprehend the sources and origin of jitter and its related consequences. Additionally, interoperability of link interfaces has become a necessity due to the wide proliferation of industry link standards. Because clock quality is a paramount concern when designing for interoperability, clock and jitter metrics and terminology are at the foundation of many industry standard link specifications. A challenge associated with understanding jitter terminology is that much of the jargon used in the industry to describe jitter phenomena is inconsistent, ambiguous and context dependent. The intent of this section is to establish a common vernacular of jitter and clock terms and to unambiguously and mathematically define key jitter metrics necessary for link design and analysis.

Generally, jitter is defined as the deviation of a data or clock edge from ideal timing. The origin of this timing uncertainty has many sources which can either be deterministic (i.e., predictable) or random. In high-speed data link applications, the two primary sources of jitter are either channel or circuit induced:

1) Channel induced jitter occurs due to interference caused by electromagnetic coupling either within a channel, which is known as intersymbol interference (ISI), or between channels, which is often referred to as co-channel interference (CCI). This interference is deterministic and affects both clock and data recovery for high-speed interfaces. Given the highly predictable nature of this jitter, it can be effectively modeled and anticipated as long as the channel characteristics are known [8]–[10]. The mitigation of these jitter sources is part of the interconnect design space and must be balanced with other important system criteria such as cost, density and performance.
2) Circuit induced sources of jitter are usually much more complex and diverse than channel induced jitter. Fundamental interference sources such as thermal and flicker noise randomly disturb normal circuit operation and cause voltage-induced timing noise. Another important jitter source results from power supply and substrate noise that injects timing noise and causes delay variation of clock circuits. As process technologies scale, device variation can cause delay variation and deterministic jitter effects such as duty-cycle distortion and PI nonlinearity. Jitter caused by circuit design defects could include such things as layout mismatch, locked loop update glitches, coupling effects, etc.

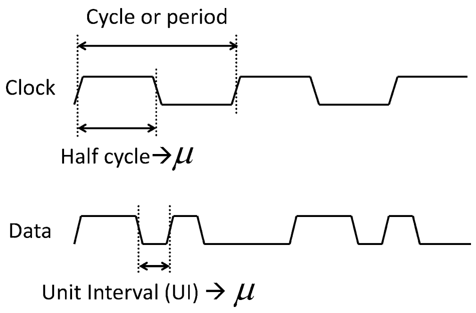


Fig. 7. Unifying symbols and terminology for clock and data signals.

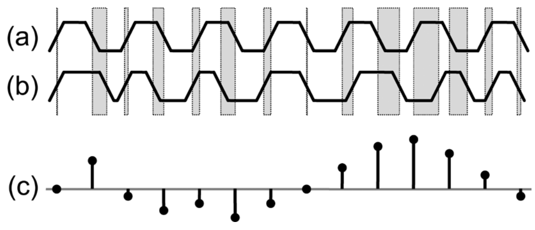


Fig. 8. Jitter sequence example in which the gray bars represent clock edge deviation from the ideal position. (a) Ideal clock, (b) Jittered clock, (c) Jitter sequence representation of ideal versus jittered clock phase.

For jitter terminology to be relevant for either clock or data signals, it is necessary to establish definitions that consolidate the mathematical representation of clock and data jitter. Fig. 7 shows the primitive dimension of a data signal which is the duration of a low (high) signal level or unit interval (UI). The same figure demonstrates the equivalent dimension of a clock signal consisting of the time from rising edge to falling edge (or vice versa) and may be referred to as the clock half cycle. To unify terminology, both the clock and data primitive time dimension will be designated as a UI and will be represented by the character μ_i in subsequent mathematical equations (subscript i represents the UI number of a clock sequence). Additionally, the jitter terms we demonstrate will be referenced to a clock signal even though all of the definitions could be applied equally well to a data signal (other than the fact that the presence of data signal edges is not guaranteed). A useful representation of signal timing uncertainty is known as a jitter sequence, which is the long term phase deviation with respect to an ideal clock. Fig. 8(c) gives an example of a jitter sequence graph in which the discrete x axis is the edge number while the y axis equals the absolute phase deviation. Mathematically, the jitter sequence is the cumulative sum of the UI in a k-length sequence as given by (1) in Table I. Other synonymous terms used to describe the jitter sequence or its corresponding magnitude and distribution

TABLE I
JITTER DEFINITIONS

Jitter sequence	$= \sum_{1}^{k} \mu_i \; ; k > 0$	(1)
Duty-Cycle Error	$= \left\| \dfrac{\overline{\mu}_{even} - \overline{\mu}_{odd}}{2\overline{\mu}} \right\|$	(2)
UI Jitter (1-UI Jitter)	$= \mu_i - \overline{\mu}$	(3)
UI-UI Jitter	$= \mu_{i+1} - \mu_i$	(4)
Period Jitter (2-UI Jitter)	$= \mu_{2i-1} + \mu_{2i} - 2\overline{\mu}$	(5)
N-UI Jitter	$= \sum_{k}^{k+\mathrm{N}-1} (\mu_i - \overline{\mu}) \; ; k > 0$	(6)
N-UI Jitter (rms)[a]	$= \dfrac{\Delta f}{f_o^{3/2}} \cdot 10^{\frac{L(\Delta f)}{20}} \cdot \sqrt{\mathrm{N}}$	(7)
Differential Jitter	$= \mu_{a_i} - \mu_{b_{i+M}}$	(8)

[a]Valid only for oscillators dominated by white Gaussian noise. S_ϕ is the phase noise spectral density at Δf (in dB), f_o is the frequency offset, and Δ_f is the fundamental clock frequency.

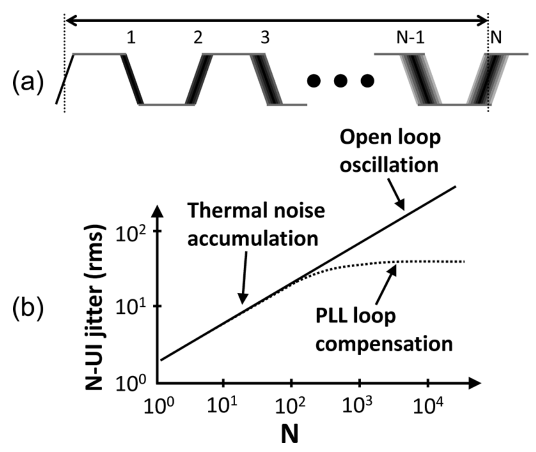


Fig. 9. N-UI jitter examples.

include: phase jitter, absolute jitter, long-term jitter and accumulated jitter.

One of the most basic metrics describing clock quality is duty-cycle error which quantifies the average shift in adjacent edges of a pattern and is defined by (2) in which $\bar{\mu}_{\text{even}}$ and $\bar{\mu}_{\text{odd}}$ signify the mean UI time of the even (i.e., i is even) and odd symbols, respectively. Duty-cycle error is highly destructive to circuit and channel operation [5] primarily due to the high frequency content of the timing noise. For instance, if this type of jitter is present at the data transmitter, it will cause the channel ISI to be amplified and will reduce equalizer effectiveness. Duty-cycle error is often referred to as *static* jitter in that it is deterministic and predictable, which means that it can be attenuated or eliminated using filters or closed-loop feedback correction.

Two basic and closely related jitter metrics used to specify the short-term or high-frequency aspects of jitter are UI jitter and UI-UI jitter defined by (3) and (4), respectively. UI jitter simply quantifies uncertainty of UI time (with no regard to chronological order of the UI). On the other hand, UI-UI jitter expresses the time difference of adjacent UIs. Generally, UI jitter is a key metric for design, analysis, and specification of data links due to historical factors and reduced measurement complexity. Closely related to UI jitter is period jitter which is used to quantify uncertainty in cycle time (equal to 2UI) of a clock and is defined by (5).

A closely related but even more general metric than UI or period jitter is that of N-UI jitter [11] which is defined by (6). This metric helps to quantify the frequency content of jitter and represents the effect of jitter accumulation on a jitter sequence. Fig. 9(a) shows an intuitive depiction of N-UI jitter using an example clock waveform. This illustration mimics the procedure for measuring this metric using an oscilloscope. By triggering the waveform on an arbitrary edge, the jitter magnitude or distribution is calculated as a function of N (number of edges from the trigger point). Fig. 9(b) demonstrates an example N-UI jitter plot based on normalized jitter of an open loop oscillator dominated by thermal noise. In this case, thermal noise causes Gaussian distributed jitter accumulation of the oscillator that equates to a slope of 1/2 on a log-log graph [11]. In other words, for every 4$\times$ increase of N, there will be a 2$\times$ increase in the standard deviation of accumulated jitter. As shown in the same example, closed loop clocking schemes (such as a PLL or CDR) can modify the N-UI jitter characteristic by reducing long term jitter up to the loop bandwidth.

It is common to depict jitter values using a single value in terms of peak-peak magnitude (pp) or standard deviation value (rms). However, the qualification of jitter values as either pp or rms implies the distribution is either truncated or purely Gaussian. In practice, jitter may be a combination of different distributions and it may be necessary to provide a more detailed description rather than use a single parameter. For example, it is appropriate to cite a pp value for jitter as long as the distribution is highly truncated and the parameter is accompanied by the sample quantity. In many cases it may be necessary to describe jitter based from a fit of a mixed distribution such as a dual-Dirac delta jitter separation technique [12].

Jitter is often measured and depicted using frequency domain metrics rather than in the time domain. For example, oscillators are frequently characterized based on phase noise magnitude at a given offset frequency from the carrier. Equation (7) demonstrates the equivalence of time and frequency domain metrics for thermal noise dominated oscillators. A more general time to frequency domain conversion method is available based on the Wiener-Khinchin theorem [69].

The preceding jitter terms are applicable to single clock or data waveforms and are widely applicable to a variety of data link applications. An additional class of jitter known as differential jitter describes comparative time uncertainty between two different clock or data waveforms as given by (8). In this analytical description, a and b refer to two separate signals while M indicates the relative edge delay between the waveforms. The metric of differential jitter is relevant in applications which are sensitive to the relative value of jitter rather than just the absolute magnitude. For example, differential jitter is an effective indicator of the circuit matching quality between an FC TX and a corresponding data TX. Furthermore, differential jitter between the input and output of a clock buffer may help to isolate the additive jitter contribution of that buffer.

Low-level jitter descriptions such as those just described and outlined in Table I are essential to the process of designing, analyzing and debugging clock circuit performance. High-level jitter and clock performance metrics used to characterize and specify transceivers and clock systems assist in the effective design and analysis of clock systems. Moreover, clock system performance metrics enable interoperability and compatibility of separately designed circuits and systems.

At the foundation of many industry standard specifications is a means to perform jitter compliance testing of separate components to guarantee interoperability. Some widely used specifications ensure compliance of each system component by restricting jitter related behavior and specifying criteria known as "jitter generation", "jitter tolerance" and "jitter transfer" [13], [14]. While specific definitions for these metrics have been established by standard bodies such as the International Telecommunication Union, we will provide more general definitions for these parameters that provide greater applicability to a wide variety of applications and systems.

Jitter Generation: Jitter generation is a measure of system intrinsic jitter in the absence of input jitter. The objective is to quantify the amount of jitter a component adds to a system in terms of magnitude, distribution and/or frequency content. This type of jitter specification is frequently applied to elements such as a PLL, TX or even a CDR. When specifying an element such as a TX, it is important to consider that the frequency content of the jitter may be just as important as the magnitude since high-frequency jitter that leads to pulsewidth distortion can cause channel ISI amplification and jitter enhancement. Furthermore, jitter frequencies much lower than the bandwidth of the CDR may be tracked and have little influence on signaling performance.

One example of a jitter generation specification that accounts for statistical and spectral behavior may consist of an N-UI jitter limit in which the magnitude or distributions are specified at various values of N. A more frequently used method is to limit accumulated jitter magnitude at different frequency bands by applying time-domain bandpass filters to the raw jitter sequence (or alternatively use similar filtering methods based on phase noise analysis or measurements).

Jitter Tolerance: This metric indicates the ability of a block or component to tolerate input jitter. Traditionally, jitter tolerance testing was applied to an RX or CDR using phase modulated data as the input source. The test ensures the target link BER is achieved when subjecting the input to a sinusoidal phase modulated source at specified frequency and amplitude criteria. Fig. 10 shows an example of a sinusoidal jitter tolerance limit specification that is based on the OTU1 specification [13].

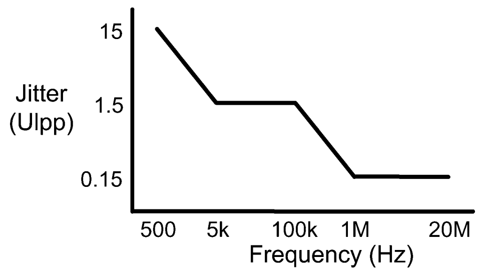


Fig. 10. OTU1 input sinusoidal jitter tolerance example.

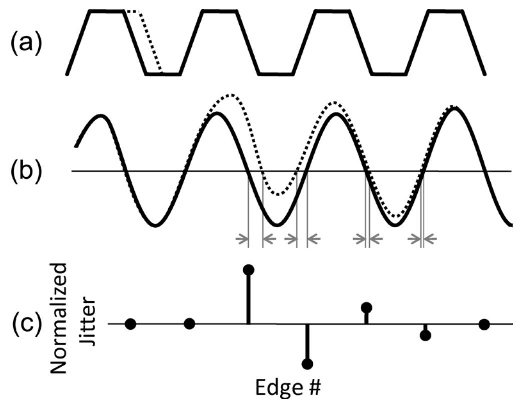


Fig. 11. Normalized jitter impulse response of a bandwidth limited circuit. (a) Ideal input clock waveform superimposed with clock incorporating jitter impulse stimulus. (b) Output clock waveforms using ideal clock versus jitter impulse clock. (c) Jitter impulse response.

Jitter Transfer: The transfer of jitter through clock system is characterized by comparing the ratio of output to input jitter as a function of frequency. An application of this metric is to limit successive amplification of jitter by a cascaded system with peaked responses. For example, the SONET specification [13] limits maximum transfer peaking of the PLL to less than 0.1 dB. In general, the system is assumed to be linear and is often depicted using a magnitude frequency response.

An alternative representation of a frequency domain jitter transfer function is that of a discrete time jitter impulse response [4], [70]. A benefit of depicting jitter transfer in the time domain is the simplicity of extracting the jitter impulse response through simulations or measurements. Fig. 11 shows an example of jitter impulse response characterization for a system with bandwidth limited response. The discrete time jitter impulse response is frequently normalized to the input jitter stimulus similar to the procedure used to represent a conventional system impulse response. The jitter impulse response is a useful metric for analysis and modeling of clock systems. For example, a clock system's effect (assuming it exhibits a linear, time-invariant phase response) on an input jitter sequence can be evaluated by convolving the jitter sequence with the jitter impulse response. Additionally, the jitter impulse response may be analyzed to produce intuitive metrics such as the factor which duty-

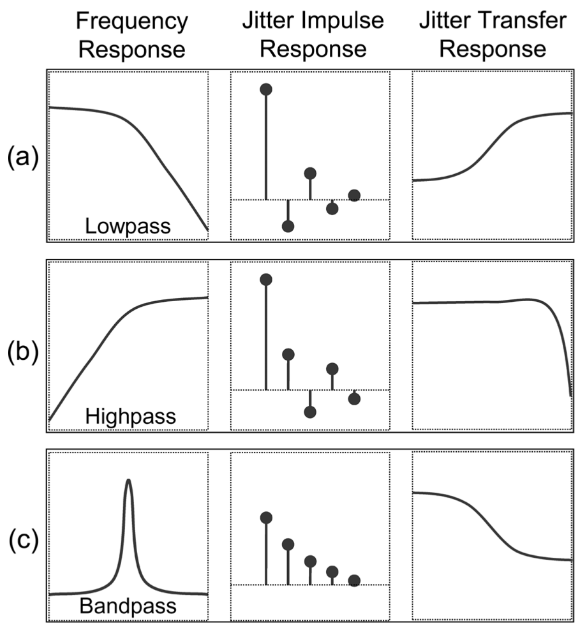


Fig. 12. Examples of relationships between frequency response and jitter transfer for (a) low-pass, (b) high-pass, and (c) bandpass.

cycle error is amplified (or attenuated) or the amount which normally distributed uncorrelated jitter is amplified by a clocking element. For instance, duty-cycle amplification (DCA) factor is defined by

$$\left|\sum_{1}^{\frac{k}{2}}(h_{2i-1}-h_{2i})\right| \tag{9}$$

in which h is the normalized jitter impulse response and k is the response length. Moreover, random jitter amplification (RJA) or the factor by which the standard deviation of the normally distributed uncorrelated jitter is amplified is given by

$$\sqrt{\sum_{1}^{k} h_i^2}. \tag{10}$$

There exists a wide variety of jitter impulse response classes, each of which is dependent on the component frequency response. Fig. 12 shows some of the most important classes of jitter impulse responses along with corresponding frequency response and jitter transfer response. The example shown in Fig. 11 and in Fig. 12(a) demonstrates a jitter impulse response that is a result of a bandwidth limited clock element such as a buffer or distribution interconnect. Though Fig. 12(a) shows a low-pass frequency response, the resulting jitter transfer response is similar to a high-pass jitter filter in that it amplifies high-frequency jitter. As shown in Fig. 12(b), a high-pass frequency response with a pole frequency much lower than the clock frequency (and the clock has a bandwidth limited waveform) results in a response that passes almost all frequencies of jitter excluding the Nyquist rate jitter otherwise known as duty-cycle error. Fig. 12(c) demonstrates a bandpass with the center frequency aligned with the fundamental clock frequency that filters destructive jitter harmonics and blocks high frequency jitter. Each of these filter functions are commonly encountered or used in the practical implementation of clock architectures outlined in this paper and a detailed understanding of the jitter transfer characteristics is essential to optimizing high-speed data link performance.

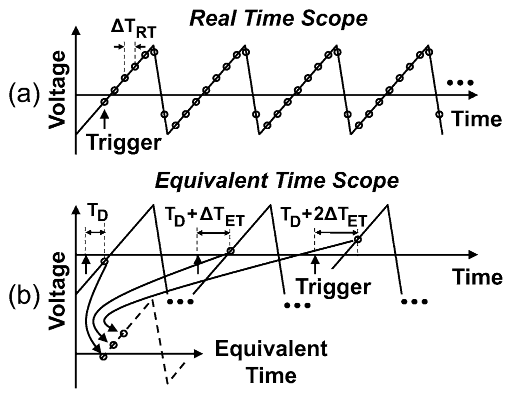


Fig. 13. Waveform sampling with (a) real time and (b) equivalent time oscilloscopes.

IV. Integrated and External Jitter Characterization Techniques

Because of the high sensitivity of link performance to clock quality, jitter measurement plays an important role in link engineering and validation. Accurate measurements are necessary to check that a component meets a required high-level jitter specification (e.g., jitter tolerance) and for confirming expected low-level jitter characteristics during debug and development. This section describes time- and frequency-domain measurement techniques for obtaining the clock and data jitter metrics described in Section III using both external and integrated measurement instruments.

A. Measurement Equipment and Limitations

Real- and Equivalent-Time Oscilloscopes: When taking time-domain jitter measurements, it is essential to understand the distinction between real-time (RT) and equivalent-time (ET) digital oscilloscopes [71].[1] Both types of scopes are commonly used for I/O clock and data measurements, but they are not always interchangeable within jitter measurement setups. Both time-domain scopes sample and digitize their inputs using high-speed analog-to-digital converter (ADC) front ends and provide waveform plots on voltage and time axes. The key difference is that an RT scope captures the waveform by taking a series of high-speed, sequential data samples whereas an ET scope reconstructs the waveform by sub-sampling a repeating waveform.

The sampling approach for both an RT and ET oscilloscope is illustrated in Fig. 13. The RT scope starts sampling at the trigger event, which can be based on the sampled waveform (e.g., start sampling when the signal crosses some voltage threshold) or another user-supplied or scope-supplied signal. The scope then takes a voltage sample every ΔT_{RT} following the trigger event, capturing the waveform during a specific period of time. This sequence of samples can be postprocessed

[1]We will not consider analog oscilloscopes since they have been largely replaced by digital oscilloscopes for high-speed link measurements.

to extract the clock edges and obtain a jitter sequence as defined in Section III. The measurement bandwidth of this type of oscilloscope is set by the analog bandwidth and sampling rate of the front-end ADC.[2] The sample interval, ΔT_{RT}, determines the Nyquist rate of the measurement, meaning that spectral content above $f_N = 1/(2\Delta T_{RT})$ will be aliased by the RT oscilloscope. The jitter noise floor for measurements made with the RT scope is primarily limited by its sampling jitter, which in commercial scopes can be <1 ps-rms [15].

The basic operation of an ET scope is shown in the bottom half of Fig. 13. Unlike an RT scope, the ET scope takes only a single sample following each trigger event (at time $T_D + n\Delta T_{ET}$ in Fig. 13), but can vary the sample delay to sweep across the waveform. Provided that the waveform is periodic with the trigger,[3] the waveform can be virtually reconstructed by combining samples as shown in the "Equivalent Time" plot of Fig. 13. Note that unlike an RT scope, the displayed waveform is not based on a contiguous series of samples but on a reordering of samples relative to the repeating trigger. This has several consequences in terms of scope resolution and bandwidth and in terms of the information contained in the waveforms. First, the time resolution is not limited by the maximum ADC sampling rate, but rather by the sampler aperture and timing resolution. In practice, ΔT_{ET} can be much less than ΔT_{RT}. As a result, the measurement bandwidth and/or voltage sampling resolution of ET scopes can be significantly higher than RT scopes.[4] However, because it sub-samples the waveform, it can only generate an aggregate picture of the repeating waveform and cannot be used to extract a jitter sequence unless the jitter is also periodic with the trigger. Thus, it cannot be used for measurements such as UI-UI jitter that require contiguous waveform snapshots. The jitter noise floor is again limited by the sample jitter, which in commercial scopes can be <200 fs [16].

Effect of Test Setups on Jitter: Many of the jitter measurements described in this section are fairly straightforward in concept. In many cases, the user can simply select from a set of predefined automated measurements. However, making accurate timing measurements for high-speed clock and data requires careful consideration of how the entire measurement setup—including I/O buffers, probes, connectors, in-line test components (e.g., amplifiers, filters or bias-Ts), cables and the instrument itself—impacts the final measurement. For example, we have already discussed how some basic knowledge of how RT and ET oscilloscopes work is necessary to understand the respective limitations in terms of aliasing and jitter correlation. It is also necessary to consider how all the signals involved in a measurement are being altered, in both linear and nonlinear aspects, and how that impacts the accuracy of the measurement.

As an example, consider the impact of a low-pass measurement channel (this could be the signal path to an external scope) on a phase noise and duty-cycle error measurement (Fig. 14). A phase noise measurement, which will be discussed in more detail later, considers only the jitter spectrum centered around the clock frequency. Relative to the clock frequency itself, the frequency span of that measurement will usually be quite narrow. For example the goal may be determining the phase noise at a 10 MHz offset from a 5 GHz carrier. As shown in Fig. 14, even if the clock being measured is beyond the channel bandwidth, it will attenuate both the clock and its sidebands about evenly, resulting in a reasonably accurate measurement (unless the filter behavior varies rapidly relative to the ± 10 MHz bandwidth of interest). However, duty-cycle error of a square-wave clock has components at DC and at all harmonics of the clock.[5] In this case, the measurement reflects spectral content within a relatively wide bandwidth. Since the filter characteristics change significantly over this range, obtaining an accurate measurement would require reducing the channel loss, equalizing for the channel by postprocessing the measurement (e.g., de-embedding by estimating the jitter impulse response), or using an alternative measurement method. Note that most time-domain jitter measurements require this type of consideration because they measure the wideband characteristics of the clock and are therefore sensitive to the measurement channel bandwidth, including the bandwidth of the equipment itself.

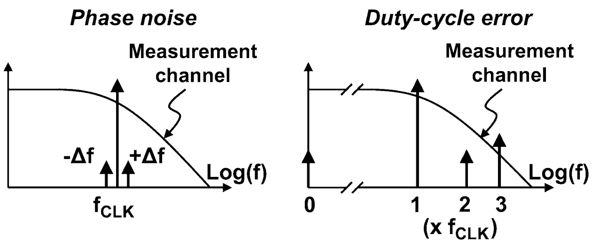


Fig. 14. Illustration showing the impact of a low-pass measurement channel on phase noise and duty-cycle error measurements.

On-Die Oscilloscope: One way to minimize or eliminate the measurement channel associated with probing on-chip data and clock signals is to integrate the measurement hardware with the link circuitry. Many transceivers are now capable of making link, interconnect, and circuit characterization measurements using on-die circuitry [17], [18]. This measurement capability is referred to here as an "on-die oscilloscope". A typical RX with on-die oscilloscope capability is shown in Fig. 15. It contains a variable-offset sampler and phase interpolator (already present in the PI-based RX) along with a pattern generator/checker and measurement loop control. The recovered clock serves as the reference trigger from which the PI can sweep the sampling point, similar to an ET oscilloscope. The key difference is that the voltage sampler, which is just the data sampler, is usually a 1-b quantizer as opposed to a high-resolution ADC used for ET scopes. Therefore, measuring voltage amplitude requires sweeping across the voltage range. Note that for embedded-clock links, it is not possible to sweep time and voltage without

[2]Suggested bandwidth guidelines for different I/O specifications can be found in [15].

[3]There are also common examples of ET scope measurements that do not require the wave to be periodic with the trigger as long as it has an underlying frequency reference that is. For example, an eye diagram can be generated based on a random data sequence that does not repeat at the trigger rate as long as the trigger period is a multiple of 1UI for the data.

[4]State-of-the-art RT scopes currently have a maximum bandwidth of 13 GHz and compared with 80 GHz for an ET scope [15], [16].

[5]This can be realized by viewing a square-wave clock with duty-cycle error as the superposition of a pulse train with an ideal clock such that the pulse train extends the duration of the clock pulses on one side. The clock itself has spectral components at the fundamental and odd harmonics. The pulse train has components at DC, the fundamental, and all harmonics of the clock.

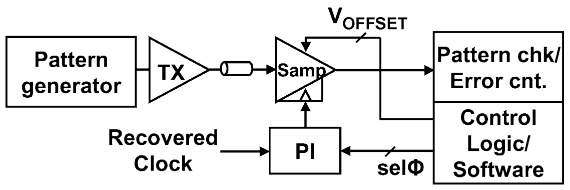


Fig. 15. Typical RX on-die oscilloscope architecture.

corrupting the timing information used to keep the clock aligned with data. However, auxiliary samplers and phase shifters can be used to sample the data without interfering with the clock recovery loops [19].

Another key advantage of on-die oscilloscopes for doing link characterization is that the measurements include the entire data channel and leverage the same circuitry used during normal link operation. In contrast, making similar measurements using external equipment either doesn't capture portions of the channel or requires invasive probing that impact the signal integrity. Also, it is not possible otherwise to make measurements that include the effects of the receiver circuitry (e.g., the data sampler aperture) on signal integrity and overall link performance.

A disadvantage of utilizing on-die circuitry is that nonlinearities in the voltage and phase offset circuits will be reflected in the measurements. The effect of these nonlinearities can be removed by calibrating the on-die circuits by using external equipment. However, calibration is not always an option, specifically if the test features are intended for self characterization for high-volume testing.

An important design decision for the on-die oscilloscope is determining the degrees of flexibility and integration for the auxiliary circuit components. The pattern generator/checker and loop controller can be integrated to provide completely automatic and self-contained link characterization. However, any or all of the three could also be external. It is often advantageous to have some portion of the measurement controller run through software to provide additional flexibility in defining and tuning the measurement algorithms. However, given the relatively slow speed of test interfaces (e.g., a JTAG scan interface) as well as interfaces to external pattern generators and checkers, higher degrees of integration can significantly reduce measurement times. There is a cost advantage to integration as well, since high-speed pattern generators and checkers carry a significant cost overhead.

The methods for using this on-die oscilloscope capability to measure BER eyes and differential clock-to-data jitter are described later in this section.[6]

B. Jitter Measurement Techniques

Duty-Cycle Error: Duty-cycle error (2) is a time-domain measurement that can be obtained using either an RT or ET oscilloscope. As mentioned in the previous example, it is a straightforward measurement unless the measurement channel loss is significant enough to amplify the duty-cycle error. For example, using this approach to measure the duty-cycle error of a 10 GHz clock using external equipment is extremely challenging and can require de-embedding to account for the channel loss.

An alternative approach is to measure the duty-cycle error by measuring only the DC portion of a differential clock. Assuming that the differential clock saturates internally, the duty-cycle can be deduced from the difference in the DC value between the two complementary clocks. This technique is widely used within integrated duty-cycle correctors [20]. Since the measurement itself filters the clock to get the DC component, it is not sensitive to the low-pass filter of the measurement channel. This technique was used successfully to measure the duty-cycle of a 10 GHz TX clock for a 20 Gb/s data link [5].

N-UI Jitter/UI Jitter/Period Jitter: N-UI jitter (6) quantifies the random jitter magnitude and degree of jitter autocorrelation for a clock. It can be measured using either an RT or ET oscilloscope using at least two different techniques. Note that UI jitter and period jitter correspond to N-UI jitter at $\mathrm{N} = 1$ and $\mathrm{N} = 2$, respectively, so they are also covered using these measurement techniques.

In one technique (Fig. 16), many clock sweeps are stored and viewed on the screen at once.[7] Since the measured clock also serves as the trigger for the scope, the sweeps will overlap almost exactly at the trigger point. Moving to the right along the time axis is equivalent to looking at how the jitter accumulates following the trigger. Measuring the jitter histogram (e.g., rms jitter) at each edge produces plots like the one shown for an open-loop VCO in Fig. 9(b) [11]. Note that the accuracy of this measurement is limited by the oscilloscope jitter noise floor. This limitation will manifest itself as a flat region (with magnitude equal to the jitter noise floor) on the low-N portion of the N-UI jitter plot. It is also important to note that for many ET scopes, the minimum time delay between the trigger and the first sample, TD, can be tens of nanoseconds. In this case, a delay line must be added to the trigger path to counterbalance the delay. However, a delay line of this length typically has significant jitter amplification at multi-GHz frequencies, resulting in pessimistic measurements.

A second method of measuring N-UI jitter is to use an RT scope to capture a typical jitter sequence. Unlike the previous method that superimposed multiple sweeps, for this method it is useful to just save a single long sweep following a single trigger event. The edge crossings can then be calculated from

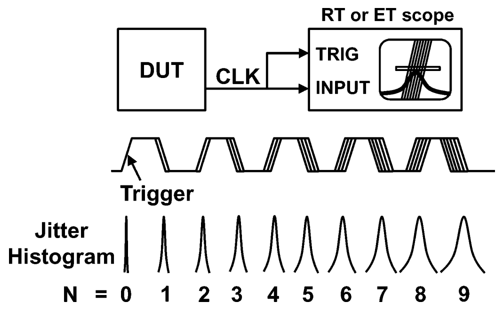


Fig. 16. N-UI clock jitter measurement setup.

[6]There are numerous other measurements that can be done using an on-die oscilloscope, including waveform capture and data sampler noise characterization [18].

[7]This requires putting a RT scope into a mode where it stores data from multiple sweeps. This is automatic for ET scopes.

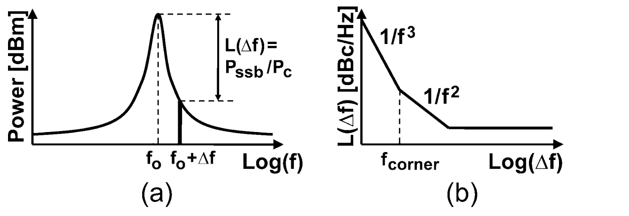


Fig. 17. (a) Power spectrum of a noisy clock with phase noise and (b) typical phase noise characteristics due to random noise.

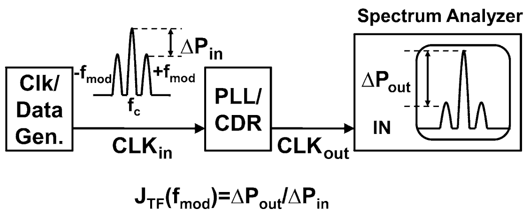


Fig. 18. PLL or CDR jitter transfer function measurement setup.

the waveform and used to generate a jitter sequence. From the jitter sequence, the N-UI jitter can be calculated using (1), where the reference clock is based on the average UI time. Although this post-processing capability (edge time extraction, N-UI calculation) is beyond what is usually available on commercial RT scopes, it can be done using auxiliary commercial or custom (e.g., programmed in MATLAB) jitter post-processing software. Compared to the previous method, this technique may be simpler to automate.

UI-UI Jitter: UI-UI jitter can only be measured using an RT oscilloscope because it requires measurement of a jitter sequence. Therefore, the test setup is identical to the latter N-UI jitter measurement setup above. From the jitter sequence, the UI-UI jitter can be calculated using (4). Typical values of UI-UI jitter for data links can easily be below the jitter noise floor of RT scopes, making it difficult or impossible to take this measurement accurately.

Phase Noise: Phase noise is a frequency-domain measure of the clock jitter. In data links, frequency-domain jitter measurements are commonly used to characterize synthesized and recovered clocks. In the frequency domain, a realistic noisy clock will have a skirt of noise around the center frequency, or "carrier," as shown in Fig. 17(a). Phase noise is calculated as the ratio of the sideband noise energy (measured at a certain frequency offset from the carrier) to the peak carrier energy.[8] It is expressed as decibels relative to the carrier, or dBc/Hz. The "/Hz" term comes from the fact that the sideband energy is integrated over a 1-Hz bandwidth. As an example, phase noise of a hypothetical clock could be described as "−80 dBc/Hz at 1 MHz offset." Phase noise is meant to characterize the magnitude of random components of the clock jitter, rather than the deterministic clock spurs. Spurs are also important clock metrics, but are quantified separately. Although theoretically random noise added to a clock causes both phase and amplitude fluctuations, the inherent limiting nature of most practical VCOs suppresses the amplitude noise close to the carrier [21].

Phase noise of an integrated oscillator due to random noise will generally contain three distinct regions [Fig. 17(b)] characterized by their slopes: 1) a $1/f^3$ region closest the carrier that is due to flicker (correlated) noise; 2) a $1/f^2$ region that is due to thermal noise; and 3) a flat region caused by white noise sources that have not been modulated by the carrier [21]. The frequency at which the $1/f^3$ and $1/f^2$ slopes meet is known as the 1/f or flicker noise corner.

[8]This definition is technically single-sideband phase noise, which is most commonly used. The dual-sideband phase noise is calculated using the sum of the positive and negative sidebands.

Phase noise can be measured in several different ways. One simple way is to measure its spectrum directly, as shown in Fig. 17(a). However, this technique requires the spectrum analyzer to operate at a setting with enough dynamic range to cover the carrier peak and the sidebands, which can reach or exceed 120 dB. Since the spectrum analyzer can trade off range for resolution, such a large range degrades the accuracy of the phase noise measurement. An improved and more commonly used approach is to mix the carrier down to an IF or baseband while attenuating the carrier peak to reduce the required dynamic range. [9] Some instruments can even lock to clock sources with significant amounts of slow drift and separate phase and amplitude modulation noise. Fortunately, complex phase noise measurements are automated by modern spectrum analyzers and signal source analyzers. Nevertheless, phase noise is a measurement that requires the user to have reasonable knowledge of how automation is done to avoid erroneous or suboptimal measurement results.

Jitter Transfer: The jitter transfer function quantifies how jitter is filtered by a clock element such as a PLL or embedded clock CDR. The measurement setup for jitter transfer is shown in Fig. 18. The phase of the clock or data source into the device under test (DUT) is modulated at a discrete frequency. Viewed in the frequency domain, this generates discrete sidebands, or spurs, around the fundamental. The relative magnitude of the spurs normalized to the input magnitude is the jitter transfer magnitude at the modulation frequency. Repeating this measurement for a range of modulation frequencies generates the jitter transfer plot.

Jitter Tolerance: Jitter tolerance for a CDR is a measurement of the amount of sinusoidal jitter modulation the link can tolerate without dropping below a specified BER. A typical setup for jitter tolerance is shown in Fig. 19, where the BER is measured using an integrated or external pattern checker. The instrument that is commonly used to generate modulated data patterns, check data patterns, and count errors is a bit error ratio tester (BERT). Similar to the jitter transfer measurement, data phase is modulated with a specific frequency and magnitude. For each modulation frequency of interest, the jitter amplitude is increased until the link BER drops below the specified threshold. The maximum amplitude at which the BER requirement is met indicates one point in the jitter tolerance curve.

[9]An interesting (if somewhat out-of-date) tutorial on phase noise and its measurement is "Understanding and Measuring Phase Noise in the Frequency Domain" (http://cp.literature.agilent.com/litweb/pdf/5952-8708.pdf). A more recent survey of measurement techniques is Agilent's "Advanced Phase Noise and Transient Measurement Techniques" (http://cp.literature.agilent.com/litweb/pdf/5989-7273EN.pdf).

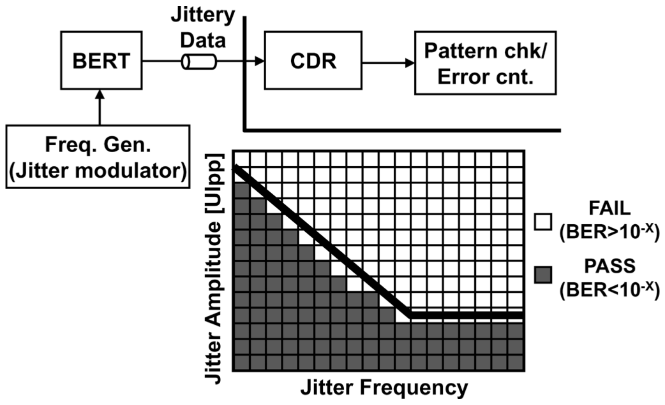


Fig. 19. Embedded clock CDR jitter tolerance measurement setup.

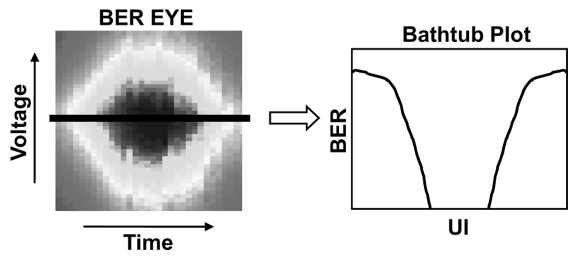


Fig. 20. Measured BER eye and timing bathtub plot.

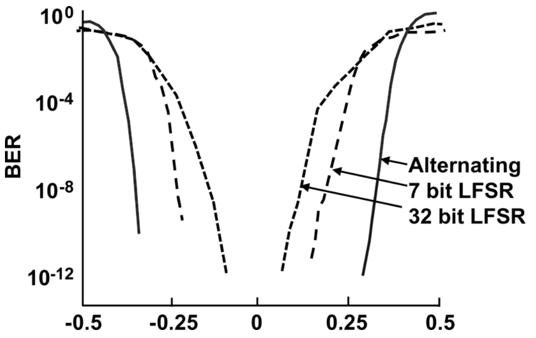


Fig. 21. A bathtub plot for embedded clock CDR obtained using on-die oscilloscope techniques [6]. Timing margins vary depending on the type of pattern into the CDR.

The complete jitter tolerance characteristic is obtained by repeating this measurement across multiple frequencies, as shown in Fig. 19.

BER Eye: BER eyes can be obtained using either integrated or external measurement equipment. The measurement can be done externally with a BERT, which recognizes and locks to the data pattern and then sweeps across the eye in time and voltage. At each point it characterizes the BER based on the known data pattern. External BER eye measurements are useful for TX compliance but may not be sufficiently accurate for full link characterization.

On-die oscilloscope BER eyes, however, can characterize the complete link performance and margins. To measure the eye diagram, the on-die oscilloscope sweeps across one equivalent-time eye in both the time and voltage axes. For each voltage-time setting, the BER is calculated using data from the pattern checker and error counter. The BER data is then plotted on voltage-time axes as shown in Fig. 20. As mentioned previously, if the link is an embedded clock link, an auxiliary sampler can be used with a phase shifter that is referenced to the recovered clock [19].

Bathtub Plot and Clock-Data Jitter: A timing bathtub plot contains a subset of the BER eye information. It can be produced based on the complete BER eye (as shown in Fig. 20) or can be measured by fixing the voltage offset and recording BER measurements as a function of phase position across at least 1UI. From the bathtub plot, one can determine the timing margin of the link or describe the differential clock-data jitter. The clock-data jitter will usually depend heavily upon the type of data pattern that is sent. Channel ISI produces jitter on the data waveform in any type of link. In addition, embedded clock link jitter is highly dependent on the activity factor of the data since it depends on data edges to for tracking. An example of this effect for a PLL-based EC link is shown in Fig. 21 [6]. Note that the clock-data jitter increases if the data patterns contain long strings of consecutive 1's or 0's. A useful measurement of clock-data jitter in forwarded clock links characterizes link timing uncertainty, or the clock-data jitter without the effects of data ISI. This can be measured by sending an alternating data pattern (e.g., ...00001111 . . . .) and sampling the BER around the transitioning data edge. This produces a jitter CDF, from which a clock-data jitter PDF can be calculated [5], [18].

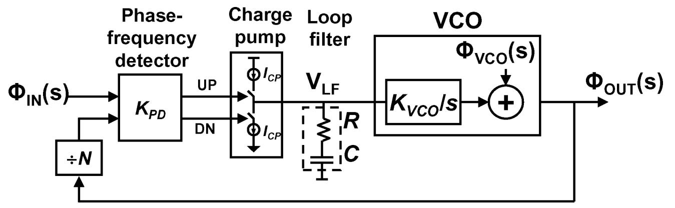
V. Circuits for Clock Synthesis, Distribution, and Recovery

Achieving high-quality clocks in an I/O system requires choosing the appropriate circuit implementations for system-level blocks. This section provides an overview of circuits that are typically used to accomplish the three main functions for I/O clocking: clock synthesis, distribution, and recovery. For each of these functions, we compare the merits of different implementation choices and cite relevant published examples. We also identify some areas of current circuit research that have attempted to solve some of the downsides of these common circuit implementations.

A. Clock Synthesis

PLLs: As described in Section II, synthesis of a clock from a reference clock or data sequence may be required at the TX and/or RX side of a link, depending on the type of link architecture. Clock synthesis is generally accomplished using a PLL, which has the ability to both multiply and filter a reference clock. Fig. 22 shows a block diagram for a conventional analog Type II 2nd order PLL. It consists of a phase-frequency detector to lock the PLL output phase and frequency to the reference clock, a charge pump to add or subtract charge from the loop filter, a VCO, and a clock divider to set the clock multiplication factor. Its designation as Type II refers to its ability to track both phase and frequency (which is a phase ramp) due to the presence of two integrators within the loop. Its designation as 2nd order refers to the two poles (with respect to phase) in the open loop transfer function, one from the loop filter and one from the VCO, which is an ideal phase integrator.

Since the PLL tracks the input phase, $\Phi_{\mathrm{IN}}(s)$, only within its closed-loop bandwidth, f_{PLL}, it essentially acts as a low-pass



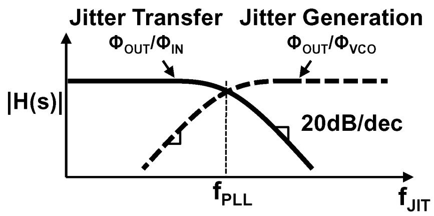


Fig. 22. Conventional analog Type II 2nd order PLL and jitter filtering transfer functions.

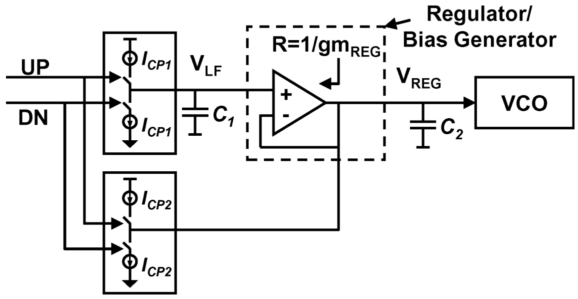


Fig. 23. Partial control loop for adaptive bandwidth PLL.

filter for reference clock jitter. Likewise, the PLL will only reject jitter of its own internal VCO below $\mathrm{f_{PLL}}$, making it a high-pass filter with respect to $\Phi_{\mathrm{VCO}}(\mathrm{s})$. The low-pass (jitter transfer) and high-pass (jitter generation) characteristics of a typical 2nd order PLL are shown in Fig. 22. In some cases, the PLL bandwidth will be part of the link specification.

Because the 2nd order PLL loop contains two integrators, it has an inherent possibility for instability and jitter peaking. To stabilize the loop, a zero must be added to the loop filter, typically by adding a resistor in series with the loop filter capacitance. The degree of stability of the PLL is characterized by the PLL damping coefficient (calculated based on loop parameters [21], [23]), which in turn specifies the amount of jitter peaking within its jitter transfer function. Often an additional pole is added to the loop filter to attenuate the voltage step caused by the charge pump, which further complicates stability [23].

Given the potentially high sensitivity of the overall link performance to PLL loop stability and bandwidth (Table II), it can be essential for the PLL characteristics to be insensitive to process, voltage and temperature (PVT). Specifically, the VCO gain, charge pump current, and passive component values can all be strong functions of PVT, resulting in poorly controlled loop dynamics. As described in [23]–[25], the characteristics of the PLL can be made to rely simply on the reference clock frequency and a ratio of capacitances by using the VCO bias voltage to control the charge pump currents and loop-filter zero $(= 1/(\mathrm{RC_1}))$ (Fig. 23). Note that this architecture explicitly separates the proportional and integral paths and also implements a third pole at the output of the op-amp.

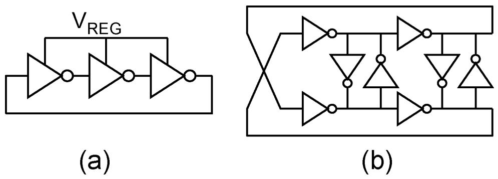


Fig. 24. Regulated-supply VCOs: (a) single-ended and (b) cross-coupled (pseudo-differential).

Given the numerous circuit implementations and references already available for integrated PLLs [21]–[25], this paper does not review general PLL circuit design. Instead, we describe implementation details and tradeoffs for several different types of VCOs that are typically used within data link PLLs. Some linear and binary phase detectors will also be presented when clock recovery circuits are discussed.

Oscillators: One of the most critical design decisions in link PLL design is the choice of an oscillator. Different types of oscillators provide different advantages and tradeoffs in terms of power, random and deterministic jitter, tuning range, and portability. The rest of this section highlights useful VCO topologies and their fundamental tradeoffs. Although this discussion centers on oscillators, the same delay cells (and associated biasing circuits) may be used for clock distribution and for tunable delay lines (e.g., within a DLL). Therefore, the characteristics of the delay cells discussed in this section directly relate to the later discussions on clock distribution and clock recovery loops.

One of the simplest VCOs to implement in CMOS is an inverter-based ring VCO, shown in Fig. 24. By tuning the control voltage, $\mathrm{V_{REG}}$, this type of oscillator can cover a wide tuning range (usually one order of magnitude or more), has power that scales quadratically with frequency (assuming a linear voltage regulator), and is easily portable to any CMOS process [26]. The key disadvantages of this ring VCO are its high VCO gain $(\mathrm{K_V = df_{VCO}/dV_{REG}})$—and hence high sensitivity to biasing noise—and its relatively poor phase noise compared with resonant-tank-based oscillators. It is important to note that VCOs that have a wide tuning range, like the inverter-based ring, will generally have the disadvantage of correspondingly high sensitivity to supply and bias noise. The relatively poor phase noise (for a fixed amount of power) is partially a result of the relatively low Q ($\sim$1–1.5) attainable with ring oscillators compared with integrated *LC* oscillators [27]. Common variations of this basic oscillator include using cross-coupled inverters and using current starving with degenerating devices to tune the delay.

Because of the high $\mathrm{K_V}$ of inverter-based delay cells, regulation must be carefully designed to minimize jitter due to power supply noise. This is especially true for PLLs since the VCO phase is the integral of $\mathrm{V_{REG}}$, including any noise on this node. A common rule of thumb given for the delay sensitivity of an inverter is $\sim$1%-delay/1%-supply [25], which is approximately valid near the nominal supply voltage for a given process. However, since $\mathrm{V_{REG}}$ must be tuned below $\mathrm{V_{SUPPLY}}$ to provide headroom for the regulator and account for frequency tuning range and PVT, the sensitivity can be far greater than this rule of thumb would indicate. Fig. 25 shows the delay sensitivity of a FO2 inverter as a function of $\mathrm{V_{REG}}$ with voltage noise normalized to the nominal supply for the process.[10] The results

[10]This simulation was done in a 45 nm CMOS process.

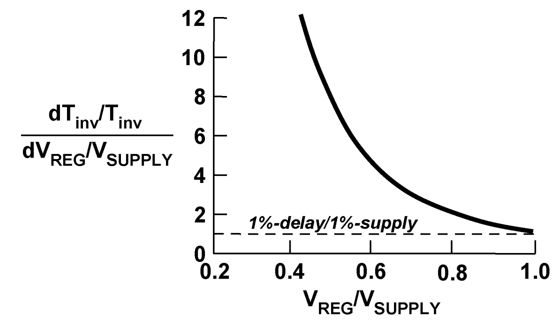


Fig. 25. Simulated sensitivity of inverter delay ($\mathrm{T_{INV}}$) to noise on the regulated voltage ($\mathrm{V_{REG}}$) normalized to the full supply voltage ($\mathrm{V_{SUPPLY}}$).

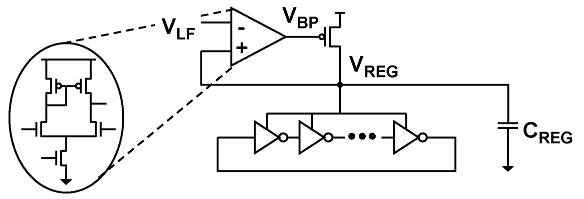


Fig. 26. Linear supply regulator for an inverter-based VCO.

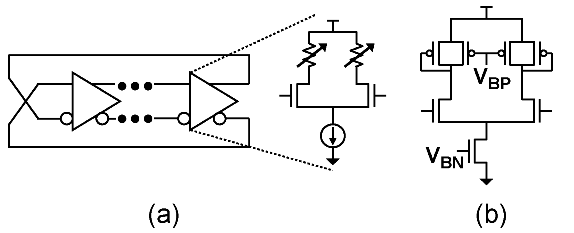


Fig. 27. (a) CML-based VCO with tunable load and (b) symmetric load delay cell.

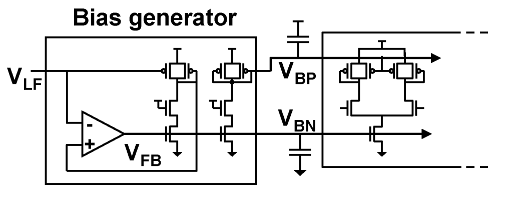


Fig. 28. Replica bias generation circuit for a symmetric load delay cells.

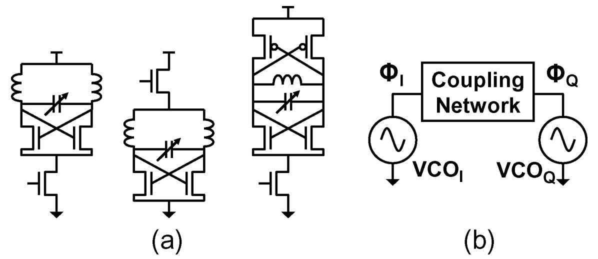


Fig. 29. (a) Several types of differential LC-VCOs and (b) quadrature LC-VCO with generalized coupling network.

for $\mathrm{V_{REG}} = \mathrm{V_{SUPPLY}}$ are consistent with the rule of thumb. However, the sensitivity increases dramatically as $\mathrm{V_{REG}}$ is reduced.[11]

The linear regulator shown in Fig. 26 is commonly used in PLL designs (and other inverter-based timing loops like DLLs) [25], [26], [28]. It acts as a unity gain buffer for $\mathrm{V_{LF}}$ while rejecting supply noise by the loop gain. Since optimal supply rejection relies on the pole at $\mathrm{V_{REG}}$ being approximately 5–10× lower than the pole at $\mathrm{V_{BP}}$, higher bandwidth op amp designs have been used, but these typically consume 3–5× more power than the VCO itself [29].

Another VCO that provides a wide tuning range and scales well with CMOS process is based on CML-based delay cells with active loads [Fig. 27(a)] [24], [31]. The symmetric-load (SL) delay cell shown in Fig. 27(b) uses a combination of saturated and diode-connected devices to form a voltage-tuned resistive load. The resistance of this load is tunable over a wide range using $\mathrm{V_{BP}}$ and maintains a symmetric *I–V* transfer function across the range of the output swing [24]. The magnitude of the delay gain with respect to $\mathrm{V_{BP}}$ is comparable to the gain of a regulated inverter delay cell to $\mathrm{V_{REG}}$. Like the inverter-based ring, it also has power that scales with frequency and has a comparable sensitivity to bias and supply noise. Also, like the inverter-based ring, this type of VCO requires regulation to generate the bias voltage and reject supply noise. However, in this case the bias regulator is not sourcing the current to the VCO. Instead it is generating the voltages that control the load impedance and current source (the latter is typically far less sensitive to noise). The bias generator in Fig. 28 acts as a unity-gain buffer (and filter) from $\mathrm{V_{LF}}$ to $\mathrm{V_{BP}}$. It also uses delay cell half-replicas to set the voltage swing equal to $\mathrm{V_{SUPPLY}} - \mathrm{V_{BP}}$ [24].

A third commonly used VCO for data links is an LC-VCO, with several implementations shown in Fig. 29. The VCO frequency is determined by the resonant frequency of the *LC* tank and is tuned by varying the capacitance. Given the square-root dependence of frequency with C and limited $\mathrm{C_{ON}/C_{OFF}}$ ratio of practical varactors, the tuning range of LC-VCOs is generally much smaller than that of RC-tuned ring VCOs. The tuning range is often extended beyond the range of the varactor by incorporating a digitally-controlled coarse tuning capacitor bank spaced such that the analog fine-tuned varactor spans any two coarse-tuning codes. This approach can achieve about 15–30% tuning range around the center frequency [32], [33]. Conversely, these VCOs are less sensitive to supply and bias noise because of this lower $\mathrm{K_V}$. They also enable lower phase noise for a given power due to the higher Q tank compared with RC-based ring VCOs [21]. Unlike the previously discussed ring VCOs, power does not scale as well with clock frequency. The power increases with the amount of negative transconductance required to offset tank losses. Since Q reduces as frequency is reduced, power can actually increase at lower frequencies. However, the upside of this feature is that the voltage swing is not a strong function of frequency, unlike the previously adaptive-biased VCOs.

There is a significant amount of research activity regarding clock synthesis and PLLs for data links. A large amount of effort is currently being focused on digitizing the PLL loop [34]–[36], which is motivated by several factors. Partially or completely digitizing the loop reduces or removes analog loop components—most notably the charge pump and loop filter.

[11]A logical question to ask is whether the rule of thumb holds if the voltage variation is instead normalized to $\mathrm{V_{REG}}$, meaning that the delay sensitivity is defined to be $(\mathrm{dT_{inv}/T_{inv}})/(\mathrm{dV_{REG}/V_{REG}})$. However, even with this definition the simulated sensitivity increases as $\mathrm{V_{REG}}$ reduces, reaching about $5\% - \mathrm{T_{INV}}/\% - \mathrm{V_{REG}}$ when $\mathrm{V_{REG}/V_{SUPPLY}} = 0.4$. This is not too surprising since the overdrive for the inverter devices ($\mathrm{V_{OD}} = \mathrm{V_{REG}} - \mathrm{V_{T(P,N)}}$) determines the inverter speed, and it is reducing at a faster rate (in terms of %) than $\mathrm{V_{REG}}$.

Digitizing the loop facilitates lower loop filter area, reduced sensitivity to analog device characteristics, easier portability between processes, and digital reconfigurability and calibration. Recent work has explored ways to solve some of the remaining challenges associated with digital PLLs, including reducing quantization jitter and implementing low-power, high-resolution time-to-digital converters and digitally-controlled oscillators (DCOs) [34], [36], [37]. Another recent area of interest has been in split-tuned ring oscillators. These oscillators have dual controls, one with high K_V and one with low K_V, to maintain the wide tuning range of RC-based rings while reducing the sensitivity to bias noise from the main control loop [38]–[40].

B. Clock Distribution

Moving the clock between circuit blocks within a data link with minimum jitter and power can be a significant challenge. Clock rates for state-of-the-art data links can range beyond 10 GHz with distribution distances of several millimeters when timing blocks are amortized across multiple data lanes. Although per-bit deskew is common for high-speed parallel links making lane-to-lane skew tolerable [3], clock jitter and duty-cycle error added by a poorly designed clock distribution network can degrade link performance. This section discusses circuit techniques and design methodologies for distributing high-speed clocks, and identifies tradeoffs of jitter generation, power, and complexity and provides two examples of clock network design.

Similar to the discussion on VCOs, the simplest distribution method in a CMOS process is using an unregulated inverter-based chain of buffers. Using inverter buffers makes the design simple to port and is power efficient since current is only drawn from the supply during clock transitions. A typical distribution network would consist of a series of buffers driving no more than a maximum length of interconnect, where the maximum length between buffers is determined by the clock frequency and *RC* characteristics of the line [41]. The primary disadvantage of unregulated inverters is their susceptibility to supply noise. Using the 1%-delay/1%-supply rule of thumb, one can quickly estimate the delay variation through an inverter-based buffer chain by adding up the total buffer delay (ignoring interconnect delay) in a chain and multiplying by the expected percentage of supply noise. This approach is valid so long as the total delay is a small fraction of the period of the expected supply noise.

Given on-chip supply noise in the range of 5-10%, it is not difficult to see how this jitter can be large relative to 1UI. For example, if we assume 20 ps/buffer delay, then each buffer adds 1-2 ps of jitter due to supply noise. If the jitter spectrum is within the bandwidth of the clock recovery mechanism, then this jitter will not impact link margins. Otherwise, it will directly degrade the link timing margin.

There are several methods of distributing the clock that trade off lower supply-noise sensitivity for higher power. For example, the same regulated delay cells described previously for inverter and CML-based VCOs can be used to distribute the clock. These have the advantage of regulating the delay of the clock buffers using the PLL or other timing loop, but they carry the disadvantages of increased power (since the bias generator must now support additional buffers) and the complexity of routing the control voltage/current to the regulated buffers. As shown in Fig. 25, regulated inverter delay cells actually have a higher inherent sensitivity to their control voltage than an unregulated inverter. The same is usually true for SL-based CML buffers as well. Therefore, distributing the control voltage/current to the buffers with inadequate shielding or bandwidth can potentially result in worse overall jitter than an unregulated inverter. CML buffers with fixed resistor loads are another option. Unlike buffers in a VCO, the delay of the clock distribution buffers does not need to be adjustable and the resistors can be fixed so that the *RC* delay is, to first order, not a function of supply noise. The main disadvantage of this approach is the relative power inefficiency of CML buffers, which draw power from the supply even when the clock is not switching, and the requirement of well-controlled resistors within the process technology.

Since minimizing delay through clock buffers will help to reduce the susceptibility of the network to supply noise, it is worthwhile to consider the extreme where the clock is distributed without repeaters. Given the availability of reasonable on-chip transmission lines [42], it is feasible to use these lines to distribute a multi-GHz clock around the chip up to several millimeters. For example, in [33] a 10-GHz global TX clock was sent nearly 3 mm using an open-drain buffer to drive differential transmission lines with on-chip termination. Since this approach involves low-swing clocks, care must be paid when designing the clock receiver to avoid introducing more jitter than was eliminated by using the repeaterless approach in the first place. This technique can also be used to distribute clocks that are driven externally, such as the 640-MHz reference clock in [43] which was sent 10 mm on chip and the 5-GHz forwarded clock in [5]. Note that because this type of distribution reuses the power of the external driver, it does not cost any additional on-chip power.

When designing a clock distribution network that balances power with jitter, it is necessary to also consider the effective bandwidth of the clock network and how much it amplifies input jitter and duty-cycle error. Take for example the clock tree shown in Fig. 30. The time-domain waveforms show the clock at the far end of the clock tree at 5.0, 7.5, and 10 GHz. Although the amplitude is clearly attenuated at higher frequencies, it is not clear how robust the network is to input jitter at each frequency. Indeed, a designer looking to minimize clock power could assume that the clock is well-behaved even up to 10 GHz and could be recovered at the far end using a level converter.

A better way to gauge the clock network bandwidth is to extract the jitter impulse response. In simulations, this is easily done by injecting a jitter impulse and tracking how the jitter progresses through the network.[12] The information can be further distilled by calculating the DCA of (9) or RJA of (10)

[12]This is done by simply modifying a clock waveform so that one edge is slightly offset from the ideal position. The edges of the jittery clock are then compared with the edges of a clock without jitter at various points along the clock network. The differential jitter between the jittery and clean clock is the jitter impulse response. This should then be normalized to the original input jitter.

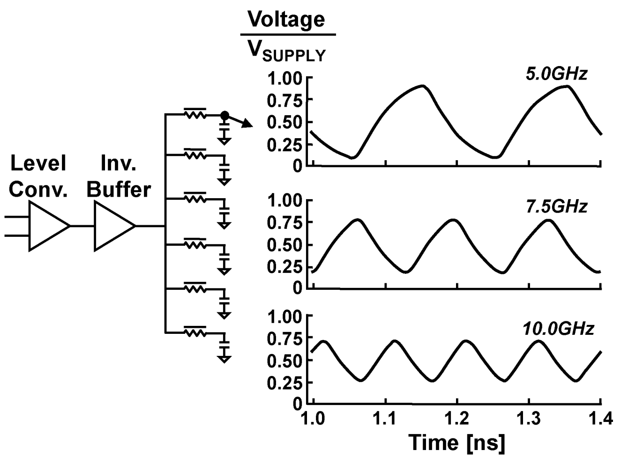


Fig. 30. Clock distribution and output waveforms with a low-swing differential input clock and inverter-driven clock tree driver.

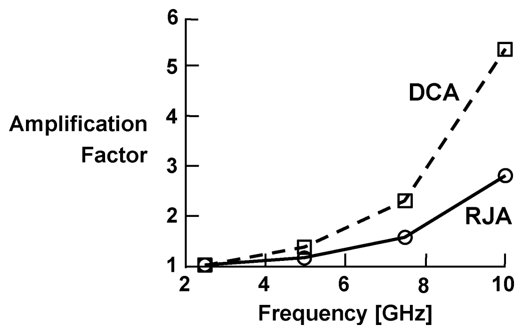


Fig. 31. Simulated random jitter amplification (RJA) and duty-cycle amplification (DCA) for the clock network in Fig. 30 as a function of frequency.

based on the normalized jitter impulse response as a function of frequency, as shown in Fig. 31. In this form, we see that random jitter is amplified by more than 1.6× above 7.5 GHz and duty-cycle error—which is jitter at the Nyquist rate of the clock—is amplified by more than 2.3× above 7.5 GHz. Also, jitter amplification increases very rapidly beyond the bandwidth of the clock network. These amplification factors need to be taken into account when modeling the system clock jitter.

Recent research and advances in clock distribution with application to data links have explored using resonance and embedded oscillators within the clock path to reduce power and jitter. Self-oscillating clock networks [44]–[46] and clock trees with tuned *LC* resonance [39], [46] have the potential to reduce power by recycling energy while also filtering clock jitter. Injection-locked oscillators within the clock path have also been demonstrated to filter and delay the clock [47], with one implementation even adaptively mixing between injection-locked and resonant buffers to optimize jitter [49].

C. Clock Recovery

In this section, clock recovery circuit implementations for the three different data link architectures discussed previously—forwarded clock (FC), PI-based embedded clock (EC) and VCO-based EC—are presented. Despite the differences in how these three architectures handle clock and data, the circuit components used to implement them overlap in many places. Once again, we will describe commonly used circuit implementation options and tradeoffs for the three architectures and finish by identifying recent research trends.

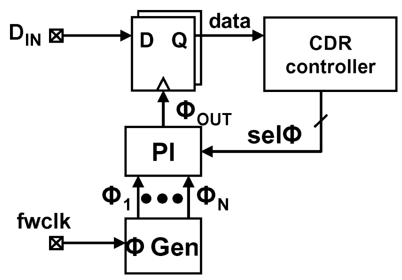


Fig. 32. Typical forwarded clock RX implementation.

Forwarded Clock: A typical implementation for a FC link is shown in Fig. 32. The forwarded clock from the TX is used to generate multiple phases at the sampling rate. These phases are then mixed by the PI to generate the sampling phase, Φ_{OUT}. Clock frequency and RX sampler bandwidth requirements can be relaxed by sampling with multiple phases within one clock cycle, meaning that there may be more than one PI or a multiphase generator following the PI [7]. The PI output phase is determined by the digital select code from the CDR controller, $\mathrm{sel}\Phi$, as described in Section II. It is common for an FC CDR controller to determine the optimal sampling point based on timing and voltage margins sweeps across the eye, similar to the on-die oscilloscope capability described in Section IV. The disadvantage of this type of algorithm is that it requires breaks in actual data transmission or requires an auxiliary data path with PI and samplers. Other methods of continuous CDR implementations will be discussed later for the PI-based EC architecture, although these can be used for FC CDRs as well.

There are several choices for how to implement the phase generator, including a delay-locked loop (DLL) [7], [50], PLL [43], multiplying DLL (MDLL) [51] and quadrature divider [20]. Only PLL and MDLL implementations are capable of clock multiplication and filtering clock jitter. These capabilities makes them particularly attractive when it is preferable to forward a sub-rate clock or if the forwarded clock has intolerable amounts of high-frequency jitter that is not correlated to the data jitter. Of these options, the DLL and PLL have found the most use based on recent FC (and EC) publications. Given that we have already discussed PLLs, we now take a closer look at the DLL and its implementation details. Although not covered in detail here, it is worth noting that the MDLL has interesting advantages over the PLL in terms of its jitter accumulation properties [51].

A conventional DLL architecture is shown in Fig. 33. It consists of a linear phase detector, charge pump, loop filter, and a tunable delay line (often a voltage controlled delay line, or VCDL). Unlike the PLL, the DLL has only a single integrator in its loop (contributed by the loop filter), making compensation in the loop filter unnecessary.[13] The DLL loop attempts to align two phases within the delay line so that the delay line maintains a latency that is some fraction of a clock period, such as $\mathrm{T_{CLK}}/2$ as in Fig. 33. Using a differential or pseudo-differential delay line

[13]For designs that use a bias generator following the loop filter [7], [24], stability should be considered. However, it is typically straightforward to achieve high loop phase margins.

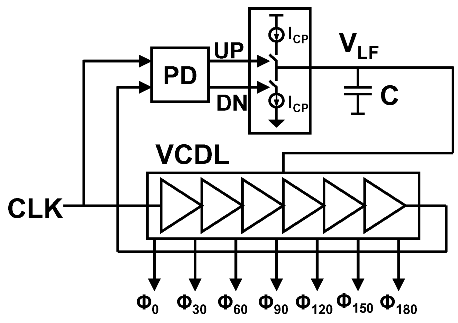

PD
UP
DN
I_{CP}
I_{CP}
V_{LF}
C
VCDL
CLK
Φ_0 Φ_{30} Φ_{60} Φ_{90} Φ_{120} Φ_{150} Φ_{180}

Fig. 33. Conventional analog delay-locked loop (DLL) with 6-stages/180° voltage-controlled delay line and complementary (180°) phase detector.

provides complementary phases so that phases in a half-cycle delay line cover a complete clock period. Unlike a PLL, the jitter transfer function of this DLL architecture is approximately all-pass [24].[14] The DLL bandwidth is a measure of the delay tracking rate as opposed to phase tracking rate. DLLs typically only require a phase-only detector since the output clocks are simply delayed versions of the input clock. Letting the loop come out of reset in its minimum delay state ensures that the loop will not reach lock at a sub-harmonic of the clock frequency.

The DLL delay line is typically a cascade of tunable delay buffers. The types of buffers and associated biasing schemes that are commonly used are the same described previously for VCOs. Regulated inverters and active-load CML are the dominant design choices based on recent data link publications. DLLs do not integrate noise on the delay line control, which somewhat relaxes the requirements for bias regulation.

A critical (and potentially catastrophic) design decision for delay lines is choosing the number of delay cells in the line. Intuitively, the implications of this decision can be understood by considering each delay buffer to be a single-pole amplifier and assuming small-signal behavior. Given N delay buffers per 180°, the amount of phase shift through each buffer is 180°/N. This phase delay at the clock frequency also specifies the 3 dB bandwidth of the buffer. For example, each buffer in a 4-stage/180° delay line will delay the clock by 45°. Therefore, the bandwidth of the buffer is *equal to the clock frequency*. Note that this relationship between clock frequency and buffer bandwidth does not depend on raw process speed since the buffer bandwidth is regulated to provide the appropriate delay. Keeping with our small-signal analysis, this means that signals near the clock frequency will be attenuated by 12 dB relative to the DC component. Based on the previous discussion of the impact of channel loss on jitter, this discrepancy in gain between low- and high-frequency signals will result in jitter being amplified through the delay line. Since most practical delay lines saturate the clock signal, this small-signal linear analysis is actually pessimistic, but the intuition that it provides is correct.[15] In practice, 4-stage/180° active-load CML delay lines provide marginal jitter amplification using active-load CML buffer (for example, duty-cycle error is amplified by

[14]There is actually some jitter peaking in DLLs as described by [52], but most practical designs minimize it by designing a suitably low DLL loop bandwidth.

[15]A similar argument can be made assuming exponential *RC* behavior. In that case, jitter amplifies less when the clock is allowed to fully saturate. Less saturation (due to lower N) results in more jitter amplification.

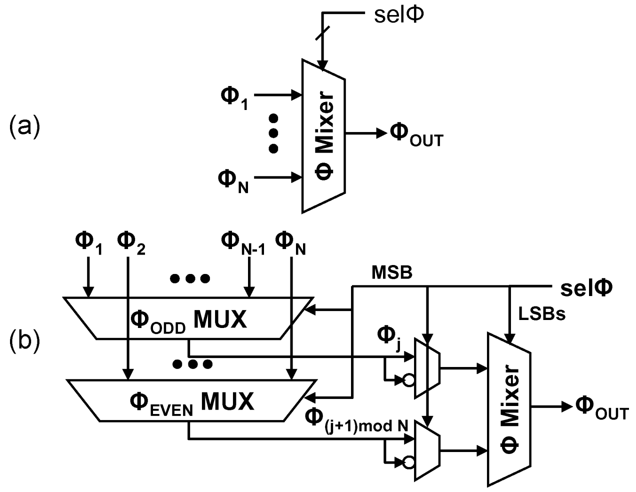


Fig. 34. Two digital phase interpolator implementations: (a) phase mixer with many phase inputs and (b) two-stage phase interpolator with coarse phase selection followed by a phase mixer with only two phase inputs.

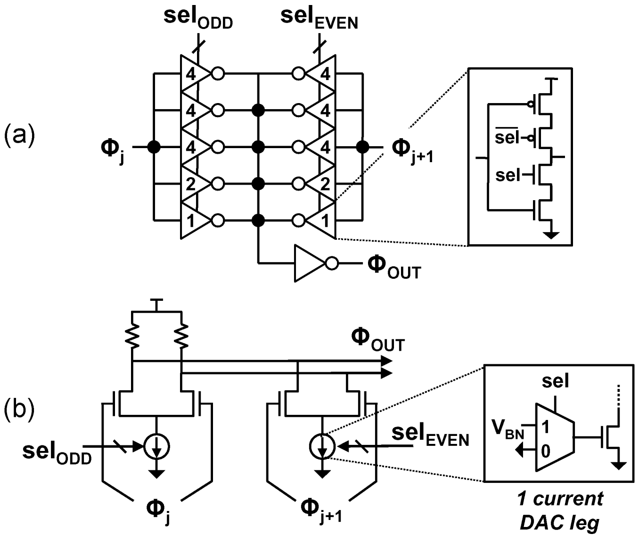


Fig. 35. (a) Inverter-based [50] and (b) CML-based [7] two-phase mixers.

1.1–1.2× per stage) whereas 6-stage/180 °CML delay lines show 5–10× less jitter amplification per stage. The acceptable number of stages, N, also depends on the type of delay cell used since the jitter amplification depends how quickly the buffer saturates when operating with a certain clock delay.

Next, we will consider options for the PI block. As shown in Fig. 34, the PI can be implemented as either a phase mixer or as a coarse-phase mux followed by a phase mixer. Two common phase mixer implementations based on inverters [50] and CML [7] are shown in Fig. 35. Both designs adjust the output phase by summing the two phases with different digitally-controlled weights. The advantage of adding a phase-select mux before the phase mixer is that it reduces the number of mixer input devices, which improves the mixer bandwidth. The main disadvantage of the mux-based design is that capacitive feedthrough due to overlap capacitance on the input devices has a nonnegligible impact on the output phase. This causes discrete phase jumps in the output phase when the de-emphasized phase is updated through the mux [7]. These discrete phase jumps do not happen with the mixer-only PI since the capacitive feedthrough

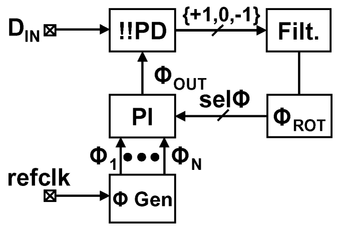


Fig. 36. Phase-interpolator (PI)-based embedded clock RX implementation with bang-bang phase detector.

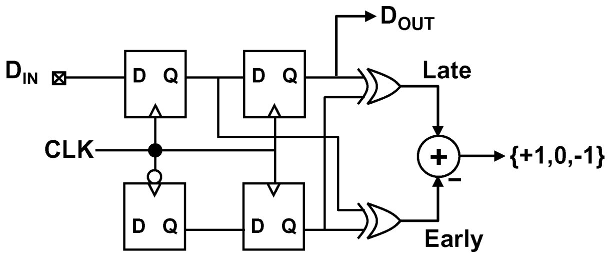


Fig. 37. A bang-bang phase detector with tri-state output.

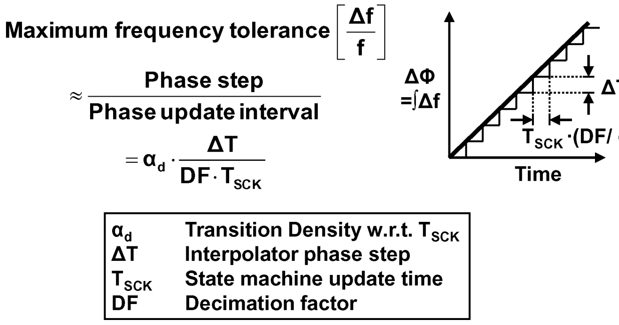

Maximum frequency tolerance [Δf/f]
≈ Phase step / Phase update interval
= α_d · ΔT / (DF·T_SCK)
ΔΦ =∫Δf
ΔT
T_SCK ·(DF/ α_d)
Time
α_d Transition Density w.r.t. T_SCK
ΔT Interpolator phase step
T_SCK State machine update time
DF Decimation factor

Fig. 38. Illustration and calculation to determine the maximum frequency tolerance for a PI-based embedded architecture with a bang-bang phase detector.

is always present from all phases to the output regardless of the PI phase setting. Mixer-only designs may be reasonable if only four phases are being mixed (e.g., quadrature inputs) [53], although even then they contain twice as many input devices as the mux-based design. The linearity of phase mixing is also very sensitive to the edge rate of the clock. Assuming ideal sinusoidal inputs, the output phase and magnitude will be the vector addition of the two mixed phases. However, linearly sweeping the magnitudes of the two phases such that the sum of their magnitudes is constant (similar to the designs in Fig. 35) will not result in equally spaced phases [39]. This nonlinearity is characteristic of the mixer-only designs due to their typically low bandwidth. Phase nonlinearity can also result when the clock edges are too fast relative to the separation of the two selected coarse phases [7], [54]. The PI circuitry is often a replica of the delay cell used in the ring VCO or VCDL so that the edge rate tracks the clock frequency across PVT. Despite using this technique, the nonlinearity is difficult to remove because of the nonlinear (exponential) slew rate of single-pole buffers.

PI-Based Embedded Clock (EC): The block diagram of a conventional PI-based EC receiver architecture is shown in Fig. 36. The architecture is identical to an FC RX with continuous deskew except that a reference clock is used instead of a forwarded clock from the TX. The CDR controller consists of a phase detector, loop filter, and phase rotator (Φ_{ROT}) to control the PI. The reference clock is generally sub-rate, so some clock multiplication, either by PLL [55] or MDLL [56], is required. The phase generator can be a cascade of clock loops, for example a PLL followed by a DLL or two cascaded PLLs [39]. As with the FC design, the bandwidth of the phase generator should be determined in part by the correlation between the reference clock and data jitter spectrum.

The most common EC CDRs architecture uses a bang-bang phase detector (!!PD) with that oversamples the data sequence to obtain information about clock-data tracking error. The bang-bang PD shown in Fig. 37 [57] generates a tri-state phase output, requires a full-rate clock (e.g., 10 GHz for 10 Gb/s), and generates the retimed data sequence. The top path generates the retimed data sequence, while the bottom path samples the data transitions, or edges. This PD generates an "early" or "late" ($+1$ or -1) signal for each data transition by comparing the edge sample to the previous and subsequent data samples. The PD also has a third "do nothing" state (0) that it outputs when there is no data transition. The basic idea of this PD can be extended to a sub-rate PD to reduce the clock rate below the data rate, which reduces power and allows for lower-frequency handling of the digital output [58].

One drawback of using a digital PD and digitally-controlled phase rotator is that the resulting clock will have quantization error at lock. In addition, latency in the PD can increase the quantization noise since it may take several clock cycles for the phase rotator to respond to phase errors at the input to the PD. To remedy this, it is common to use some amount of digital pre-filtering between the PD and the phase rotator.

In practical EC data links, the RX reference clock frequency will vary slightly from the TX clock either because they do not share the same clock source or due to the wander between two different PLLs that share the same reference clock. As a result, one of the key specifications for a PI-based EC CDR is the maximum frequency tolerance between the TX (data) and reference clock. A static frequency difference manifests itself as an accumulating phase difference between the reference clock and the incoming data rate. Since this architecture updates phase periodically by single discrete steps, increasing the frequency tolerance requires either increasing the phase step and/or reducing the update time (Fig. 38) [40]. However, increasing the phase step increases quantization jitter and reducing the update time increases power. Therefore, there is a fundamental tradeoff in this architecture between quantization jitter and frequency tolerance.

VCO-Based Embedded Clock (EC): The architecture for a VCO-based (or PLL-based) EC CDR is shown in Fig. 39. The key difference between this and the PI-based EC CDR is that there is no external reference clock provided. As such, the loop must be frequency locked and then phase locked to the incoming data sequence. As with a PI-based EC CDR, the phase detector must be able to correctly handle cases when no edges are present.

Two important specifications for this type of CDR are the lock range and capture range. The lock range is the frequency range

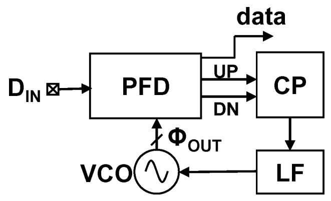


Fig. 39. VCO-based embedded clock RX.

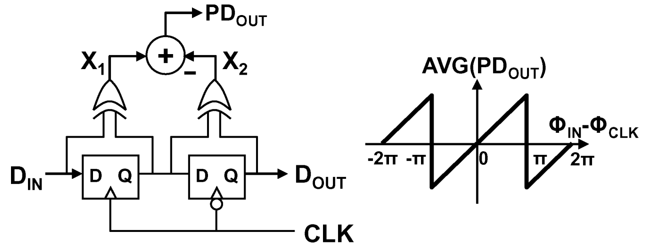


Fig. 40. Hogge phase detector and phase-to-voltage transfer characteristic.

over which the CDR can operate, and is typically limited by the tuning range of the internal oscillator when driven by the charge pump. The capture range is a measure of the maximum initial frequency difference between the VCO output and the data rate that still results in a locked loop. In the absence of a frequency detector, the capture range is about equal to the bandwidth of the CDR. However CDR bandwidth, which is usually part of the link design specification or standard, can be quite narrow relative to the expected variation of an integrated oscillator. To increase the capture range, some method of frequency locking the loop is necessary. This can be accomplished with an external reference [59], but it is preferably done using a "referenceless" method [32], [60].

PLL-based CDRs may use either a linear or bang-bang phase detector. The key tradeoff here is the reduced design complexity of a bang-bang PD compared with the generally better jitter characteristics of a linear PD [32]. As with the PI-based EC CDR, the impact of phase detector latency must be carefully considered during stability analysis and can be mitigated by reducing the CDR bandwidth.

The bang-bang PD shown previously in Fig. 37 generates quantized early/late/absent information to control the loop. It is commonly used in PLL-based CDRs that can tolerate its associated loop nonlinearity [57], [58], [60], [61] and can be analyzed using similar nonlinear techniques as PLLs with bang-bang phase detectors [71]. When a linear PD is required, the Hogge PD (Fig. 40) can be extremely useful [21], [62]. Like the bang-bang PD, it also has the benefit of generating the retimed data sequence within the PD and generates no signal when no edges are present on the incoming data stream (again using XOR logic on time-delayed samples). When there is a data edge, the detector generates pulses at X_1 and X_2. The pulse-width at X_1 is linearly proportional to the phase difference between clock and data. For the full-rate clocking example in Fig. 40, the pulsewidth will be one half clock cycle when the clock is sampled optimally between transitions and increases (decreases) if the clock is late (early). However, the pulsewidth at X_2 is ideally equal to one half clock cycle regardless of the clock-data phase relationship, thus providing a reference pulse to subtract from X_1. Note that the subtraction of X_2 from X_1 can be implemented within the charge pump. Although this example shows a full-rate example of a Hogge PD, it has been adapted for sub-rate clocking [63]. Because of the linear characteristics of the PD, the CDR can be analyzed using similar analysis as linear PLLs.

Recent Clock Recovery Research: Clock recovery continues to be an extremely active area of I/O research over the past several years, with a particular emphasis on low-power and scaling tolerant/portable designs. Some very low-power implementations have incorporated the phase tuning circuitry into the PLL phase detector [28] or charge pump [63]. For achieving extremely fine, process-insensitive phase steps, a delta-sigma modulator (DSM) can be used within the PLL feedback loop [65]. This architecture decouples the discrete phase LSB from the programmable phase resolution, and has been demonstrated with sub-picosecond resolution. Injection-locked oscillators have also found increasing use in link clocking. They have been used as programmable phase shifters by either varying the injection point in a ring VCO [56] or by varying the free-running frequency of the oscillator away from the injection frequency [48]. Injection-locked oscillators have also been proposed for clock recovery in burst-mode CDRs, providing advantages in both simplicity and lock speed [66]. Other recent work has proposed methods to remove the need for oversampling—and its associated power and area—for continuous clock-data tracking. For example, [67] demonstrated a "baud rate" CDR that uses the integral of the input data pattern to provide timing information. For process portability and tolerance as well as area reduction, digitizing the clock recovery loop has a distinct advantage. However, operating the entire digital proportional-integral loop filter at the full clock rate is usually not power efficient and/or feasible for high-speed applications. To solve this problem, both [32] and [61] digitize the integral controller, which can be clocked at a sub-rate without significantly impacting the loop dynamics, while operating the proportional path at the higher clock rate. In [32] this was done with a DSM-based linear phase detector (based on the Hogge), whereas [61] used the output of a bang-bang phase detector to control a simple DAC in the proportional path.

VI. Clock System Modeling and Performance Tradeoffs

To design and implement balanced I/O clock architectures, it is essential to not only understand low-level clock circuit implementation tradeoffs but to also comprehend how the circuit building blocks interact at the system level and their individual effect on aggregate link performance. Moreover, designing clock circuits and systems in a holistic manner will not only optimize the performance of high-speed data link interfaces but it will minimize power and cost. Essential to this holistic method is the incorporation of accurate clock and link modeling algorithms and tools as part of the overall design process [10].

There are numerous methods that may be used to model clock system components and architectures. The most primitive methods leverage results from independent block level (e.g., PLL, CDR, TX, RX, clock buffers) circuit simulations or specifications and cascade the jitter parameters together in a simplistic manner. These crude models disregard the complex statistical, nonlinear and spectral interaction of the clock circuits and system components. On the other hand, full system circuit simulation of data links can provide high accuracy with few simplifying assumptions. However, since modern I/O circuits and systems are highly complex and may contain many thousands or even millions of devices, detailed system level circuit simulations don't provide a realistic analysis framework primarily due to simulation time and computation complexity constraints. Furthermore, it may be burdensome to require complete transistor level circuit models when performing high-level system tradeoffs or sensitivity studies. The most accurate and useful clock modeling and analysis methodologies pursue an intermediate approach between the two aforementioned methods. In this section, we will give a brief overview of a behavioral approach that can be used to enable circuit and system level optimization and analysis of even the most complex clock topologies.

The analysis approach we demonstrate is based on a functional abstraction of the most critical low level circuit building blocks to accurately emulate clock phase (i.e., jitter sequence) behavior based on a number of inputs such as power supply noise, device noise, device mismatch, jitter transfer, input jitter sequence, etc [68]. The models from these primitive clock building blocks are assembled to emulate an entire data link architecture. Examples of these primitive clock building blocks may include: PLL, CDR, DLL, clock buffer, TX, RX, and channel. By just modeling the circuit's clock jitter sequence rather than emulating the entire clock waveforms, a significant decrease in simulation time and model complexity is achieved without loss of accuracy. However, in our simulation framework we choose to use complete voltage waveforms to model certain functions that require enhanced accuracy or are not conducive to edge-based modeling. For example, we model linear time-invariant channels and equalizer functions using a convolution-based approach for both the clock and data.

To demonstrate the utility of a system level model approach, we compare the performance and parameter sensitivities of two different clock architectures: FC DLL-based and EC PI-based. A combination of these two clock architectures is shown in Fig. 41 in which selecting connection (a) results in the FC mode and selection (b) results in the EC architecture. The systems were analyzed based on a data rate of 10 Gb/s with 30 cm FR4-based clock and data channels. Other baseline assumptions used for the analysis are listed in the caption of Fig. 41 and in the baseline column of Table II. An attempt was made to use practical and reasonable estimates for the clock architecture parameters to produce a realistic example.

The sensitivity results of Table II indicate first-order sensitivity of each parameter as it is modified from the given baseline value. Note that the baseline margin for the FC link of 55 ps (out of 100 ps UI) is much greater than the EC link margin of 18 ps. This is primarily due to the much higher effective clock recovery bandwidth of the FC architecture and the corresponding low sensitivity to TX jitter that is correlated between clock and data. On the other hand, EC recovery effectiveness depends solely on the RX CDR which has a bandwidth that is

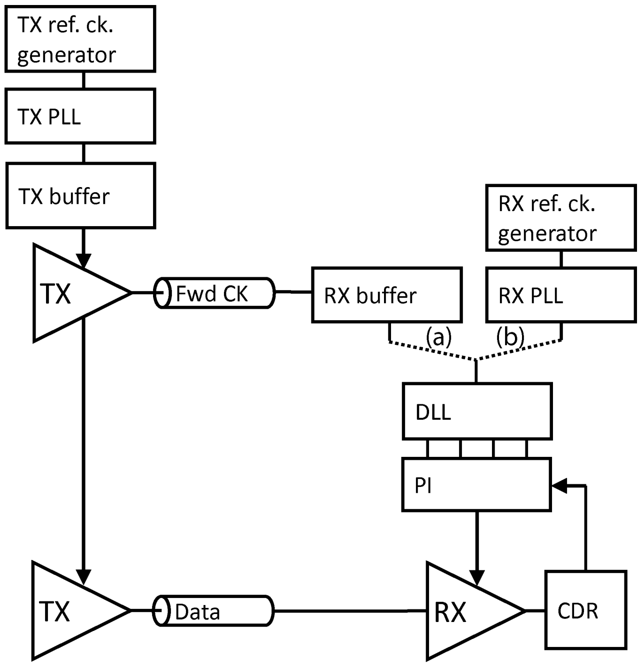


Fig. 41. Clock architecture comparison example. Connection (a) enables the forwarded clock DLL-based mode while (b) selects the embedded clock PI-based model. Assumptions used in the analysis: CK505 reference clock generator with spread spectrum clock enabled, 2nd order PLL, 2-tap data TX pre-emphasis, binary phase detector based CDR.

limited to a few MHz. This causes EC links to be sensitive to jitter frequencies above the CDR bandwidth and explains why the performance sensitivity to reference clock jitter, PLL accumulated jitter, PLL bandwidth and clock distribution jitter is 3 to 4 times greater than it is for FC.

There are some aspects of FC architectures that must be carefully optimized to produce sufficient performance and margin. For example, Table II demonstrates the effect of adding excess delay to the forwarded path (by adding an additional 2UI to the RX buffer latency) which hampers the clock recovery bandwidth provided by clock to data tracking. Another characteristic of FC methods that must be carefully suppressed is the effect of high-frequency jitter amplification across the low-pass FC channel. The addition of the RX buffer bandpass filter, as demonstrated in Table II, reverses the jitter amplification of the clock channel and produces enhanced BER and eye margin.

Though a high loop bandwidth CDR is not required for the FC architecture, it was enabled to force a symmetric comparison between the two architectures. One strong indicator of CDR bandwidth is the proportional controller gain which was set to a baseline value of 0.5 ps. As indicated by the relative sensitivities, the optimum value for the CDR (relative to the baseline) is actually higher for the EC architecture and lower for the FC architecture. This data agrees with intuition in that the primary mechanism for FC recovery is not the CDR (other than for low-frequency deskew) but rather the FC channel. Furthermore, the EC architecture is sensitive to CDR loop stability and dither jitter which is influenced by the CDR loop feedback latency and the PI phase resolution.

TABLE II
FORWARDED & EMBEDDED CLOCK ARCHITECTURE SENSITIVITIES

Parameter change	**Baseline change**	**Eye width change vs. baseline (units = 1ps or 0.01UI)**	
		Forwarded clock	**Embedded clock**
Baseline eye width		**55**	**18**
TX ref. ck. jitter (pp)	50ps→60ps	0	−4
TX PLL 1-UI jitter , rms (Gaussian jitter, accumulated)	0.5ps→0.75ps	−4	−12
TX PLL bandwidth	4MHz→6MHz	0	−5
TX buffer sinusoidal jitter @ 200MHz	±15ps→±18ps	−3	−10
TX buffer duty cycle error	1% →2%	−1	−1
RX buffer latency	1UI→3UI	−7	n/a
RX buffer band-pass filter (Q=3)	Disable→Enable	+3	n/a
RX PLL 1-UI jitter , rms (Gaussian jitter, accumulated)	0.5ps→0.75ps	n/a	−0
RX PLL bandwidth	4MHz→6MHz	n/a	−7
CDR loop latency	2UI→4UI	0	−2
CDR proportional controller gain	0.5ps→1ps	−2	+3
PI phase resolution	0.015UI→0.03UI	0	−1

VII. CONCLUSION

Data link performance will continue to scale by means of high-quality clock architecture and circuit solutions. Optimizing I/O clock systems requires accurate modeling techniques and detailed knowledge of circuit and architecture interactions and tradeoffs. Interoperability specifications of many high-speed data links require designers to be knowledgeable about jitter terminology as well as clock measurements techniques. To optimize and balance data link performance, power, and cost, it is crucial that circuit and system designers develop a deep understanding of clocking tradeoffs at all levels of the link hierarchy from the circuit-level implementation to system-level architecture.

ACKNOWLEDGMENT

The authors would like to thank J. Jaussi, M. Mansuri, G. Balamurugan, S. Shekhar, J. Kennedy, and R. Mooney for collaboration and helpful discussions.

REFERENCES

[1] *IEEE Scalable Coherent Interface (SCI)*, , IEEE , New York, 1956.
[2] J. Alexander, "Clock recovery from random binary signals," *Electron. Lett.*, vol. 11, no. 22, pp. 541–542, 1975.
[3] E. Yeung and M. A. Horowitz, "A 2.4 Gb/s/pin simultaneous bidirectional parallel link with per-pin skew compensation," *IEEE J. Solid-State Circuits*, vol. 35, no. 11, pp. 1619–1628, Nov. 2000.
[4] G. Balamurugan and N. Shanbag, "Modeling and mitigation of jitter in multi-Gbps source-synchronous I/O links," in *Proc. 21st Int. Conf. Computer Design*, 2003, pp. 254–260.
[5] B. Casper, J. Jaussi, F. O'Mahony, M. Mansuri, K. Canagasaby, J. Kennedy, E. Yeung, and R. Mooney, "A 20 Gb/s forwarded clock transceiver in 90 nm CMOS," in *ISSCC 2006 Dig. Tech. Papers*, pp. 90–91.
[6] J. Jaussi, B. Casper, M. Mansuri, F. O'Mahony, K. Canagasaby, J. Kennedy, and R. Mooney, "A 20 Gb/s embedded clock transceiver in 90 nm CMOS," in *ISSCC 2006 Digest of Tech. Papers*, pp. 340–341.
[7] S. Sidiropoulos and M. A. Horowitz, "A semidigital dual delay-locked loop," *IEEE J. Solid-State Circuits*, vol. 32, no. 11, pp. 1683–1692, Nov. 1997.
[8] B. Casper, M. Haycock, and R. Mooney, "An accurate and efficient analysis method for multi-Gb/s chip-to-chip signaling schemes," in *IEEE Symp. VLSI Circuits Dig. Tech. Papers*, 2002, pp. 54–57.
[9] V. Stojanović and M. Horowitz, "Modeling and analysis of high speed links," in *Proc. IEEE Custom Integrated Circuits Conf.*, 2003, pp. 589–594.
[10] B. Casper, G. Balamurugan, J. Jaussi, J. Kennedy, M. Mansuri, F. O'Mahony, and R. Mooney, "Future microprocessor interface: Design, analysis and optimization," in *Proc. IEEE Custom Integrated Circuits Conf.*, 2007, pp. 479–486.
[11] J. A. McNeill, "Jitter in ring oscillators," *IEEE J. Solid-State Circuits*, vol. 32, no. 6, pp. 870–879, Jun. 1997.
[12] Fibre Channel—Methodologies for Jitter and Signal Quality Specification [Online]. Available: http://www.t11.org/index.htm T11/04-101v5, Jun. 2004
[13] International Telecommunication Union G-Series Specifications [Online]. Available: www.itu.int
[14] Telcordia Technologies, Synchronous Optical Network (SONET) Transport Systems: Common Generic Criteria (A Module of TSGR, FR-440) GR-253-CORE, no. 3, Sep. 2000.
[15] Agilent Infiniium DSA/DSO90000A Series Data Sheet [Online]. Available: http://cp.literature.agilent.com/litweb/pdf/5989-7819EN.pdf
[16] Agilent 86100C Infiniium DCA-J Series Data Sheet [Online]. Available: http://cp.literature.agilent.com/litweb/pdf/5988-5235EN.pdf
[17] J. Zerbe, P. S. Chau, C. W. Werner, T. P. Thrush, H. J. Liaw, B. W. Garlepp, and K. S. Donnelly, "1.6 Gb/s/pin 4-PAM signaling and circuits for a multidrop bus," *IEEE J. Solid-State Circuits*, vol. 36, no. 5, pp. 752–760, May 2001.
[18] B. Casper, A. Martin, J. E. Jaussi, J. Kennedy, and R. Mooney, "8 Gb/s SBD link with on-die waveform capture," *IEEE J. Solid-State Circuits*, vol. 38, no. 12, pp. 2111–2120, Dec. 2003.
[19] 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, Apr. 2005.
[20] T. Lee *et al.*, "A 2.5 V CMOS delay-locked loop for an 18 Mbit, 500 MB/s DRAM," *IEEE J. Solid-State Circuits*, vol. 29, no. 12, pp. 1491–1496, Dec. 1994.
[21] T. Lee, *The Design of CMOS Radio-Frequency Integrated Circuits*. Cambridge: Cambridge University Press, 1998, ch. 15 and 17.
[22] B. Razavi, *Phase-Locking in High-Performance Systems: From Devices to Architectures*. Hoboken, NJ: Wiley, 2003.
[23] M. Mansuri and C.-K. K. Yang, "Jitter optimization based on phase-locked loop design parameters," *IEEE J. Solid-State Circuits*, vol. 37, no. 11, pp. 1375–1382, Nov. 2002.
[24] J. Maneatis, "Low-jitter process-independent DLL and PLL based self-biased techniques," *IEEE J. Solid-State Circuits*, vol. 31, no. 11, pp. 1723–1732, Nov. 1996.
[25] S. Sidiropoulos, D. Liu, J. Kim, G. Wei, and M. Horowitz, "Adaptive bandwidth DLL's and PLL's using regulated supply CMOS buffers," in *IEEE Symp. VLSI Circuits Dig. Tech. Papers*, 2000, pp. 124–127.
[26] V. von Kaenel, D. Aebischer, C. Piguet, and E. Dijkstra, "A 320 MHz, 1.5 mW @ 1.35 V CMOS PLL for microprocessor clock generation," *IEEE J. Solid-State Circuits*, vol. 31, no. 11, pp. 1715–1722, Nov. 1996.

[27] B. Razavi, "A study of phase noise in CMOS oscillators," *IEEE J. Solid-State Circuits*, vol. 31, no. 3, pp. 331–343, Mar. 1996.

[28] T. Toifl, C. Menolfi, P. Buckmann, M. Kossel, T. Morf, R. Reutemann, M. Reugg, M. Schmatz, and J. Weiss, "0.94 ps-rms-jitter 0.016 mm^2 2.5 GHz multi-phase generator PLL with 360 $^\circ$ digitally programmable phase shift for 10 Gb/s serial links," in *ISSCC 2005 Dig. Tech. Papers*, pp. 410–411.

[29] E. Alon, J. Kim, S. Pamarti, K. Chang, and M. Horowitz, "Replica compensated linear regulators for supply-regulated phase-locked loops," *IEEE J. Solid-State Circuits*, vol. 41, no. 2, pp. 413–424, Feb. 2006.

[30] A. Hajimiri, S. Limotyrakis, and T. Lee, "Jitter and phase noise in ring oscillators," *IEEE J. Solid-State Circuits*, vol. 34, no. 6, pp. 790–804, Jun. 1999.

[31] I. Young, J. Greason, J. Smith, and K. Wong, "A PLL clock generator with 5 to 110 MHz lock range for microprocessors," in *ISSCC 1992 Dig. Tech. Papers*, pp. 50–51.

[32] M. Perrott, "A 2.5-Gb/s multi-rate 0.25- μm CMOS clock and data recovery circuit utilizing a hybrid analog/digital loop filter and all-digital referenceless frequency acquisition," *IEEE J. Solid-State Circuits*, vol. 41, no. 12, pp. 2930–2944, Dec. 2006.

[33] F. O'Mahony, M. Mansuri, B. Casper, J. Jaussi, and R. Mooney, "A low-jitter PLL and repeaterless clock distribution network for a 20 Gb/s link," in *IEEE Symp. VLSI Circuits Dig. Tech. Papers*, Jun. 2006, pp. 36–37.

[34] R. B. Staszewski, J. Wallberg, S. Rezeq, C.-M. Hung, O. Eliezer, S. Vemulapalli, C. Fernando, K. Maggio, R. Staszewski, N. Barton, M.-C. Lee, P. Cruise, M. Entezari, K. Muhammad, and D. Leipold, "All-digital PLL and GSM/EDGE transmitter in 90 nm CMOS," in *ISSCC 2005 Dig. Tech. Papers*, pp. 316–317.

[35] A. Rylyakov, J. Tierno, G. English, D. Friedman, and M. Meghelli, "A wide power supply range (0.5 V-to-1.3 V) wide tuning range (500 MHz-to-8 GHz) all-static CMOS AD PLL in 65 nm SOI," in *ISSCC 2007 Dig. Tech. Papers*, pp. 172–173.

[36] C.-M. Hsu, M. Straayer, and M. Perrott, "A low-noise, wide-BW 3.66 Hz digital fractional-N frequency synthesizer with a noise-shaping time-to-digital converter and quantization noise cancellation," in *ISSCC 2008 Dig. Tech. Papers*, pp. 340–341.

[37] M. Lee and A. Abidi, "A 9 b, 1.25 ps resolution coarse-fine time-to-digital converter in 90 nm CMOS that amplifies a time residue," *IEEE J. Solid-State Circuits*, vol. 43, no. 4, pp. 769–777, Apr. 2008.

[38] S. Anand and B. Razavi, "A 2.75 Gb/s CMOS clock recovery circuit with broad capture range," in *ISSCC 2001 Dig. Tech. Papers*, pp. 214–215.

[39] J. Poulton, R. Palmer, A. M. Fuller, T. Greer, J. Eyles, W. J. Dally, and M. Horowitz, "A 14-mW 6.25-Gb/s transceiver in 90-nm CMOS," *IEEE J. Solid-State Circuits*, vol. 42, no. 12, pp. 2745–2757, Dec. 2007.

[40] P. Hanumolu, G.-Y. Wei, and U.-K. Moon, "A wide-tracking range clock and data recovery circuit," *IEEE J. Solid-State Circuits*, vol. 43, no. 2, pp. 425–439, Feb. 2008.

[41] R. Ho, K. Mai, and M. Horowitz, "The future of wires," *Proc. IEEE*, no. 4, pp. 490–504, Apr. 2001.

[42] B. Kleveland, T. H. Lee, and S. S. Wong, "50-GHz interconnect design in standard silicon," in *IEEE MTT-S Int. Microw. Symp. Dig.*, 1998, pp. 1913–1916.

[43] K. Chang, S. Pamarti, K. Kaviani, E. Alon, X. Shi, T. J. Chin, S. Jie, G. Yip, C. Madden, R. Schmitt, C. Yuan, F. Assaderaghi, and M. Horowitz, "Clocking and circuit design for a parallel IO on a first generation CELL processor," in *ISSCC 2005 Dig. Tech. Papers*, pp. 526–527.

[44] J. Wood, T. C. Edwards, and S. Lipa, "Rotary traveling-wave oscillator arrays: A new clock technology," *IEEE J. Solid-State Circuits*, vol. 36, no. 11, pp. 1654–1665, Nov. 2001.

[45] F. O'Mahony, C. P. Yue, M. Horowitz, and S. S. Wong, "A 10-GHz global clock distribution using coupled standing-wave oscillators," *IEEE J. Solid-State Circuits*, vol. 38, no. 11, pp. 1813–1820, Nov. 2003.

[46] S. Chan, P. Restle, T. Bucelot, S. Weitzel, J. Keaty, J. Liberty, B. Flachs, R. Volant, P. Kapusta, and J. Zimmerman, "A resonant global clock distribution for the cell broadband-engine processor," in *ISSCC 2008 Dig. Tech. Papers*, pp. 512–513.

[47] L. Zhang, B. Ciftcioglu, M. Huang, and H. Wu, "Injection-locked clocking: A new GHz clock distribution scheme," in *Proc. IEEE Custom Integrated Circuits Conf.*, 2006, pp. 785–788.

[48] F. O'Mahony, S. Shekhar, M. Mansuri, G. Balamurugan, J. Jaussi, J. Kennedy, B. Casper, D. J. Allstot, and R. Mooney, "A 27 Gb/s forwarded-clock I/O receiver using an injection-locked LC-DCO in 45 nm CMOS," in *ISSCC 2008 Dig. Tech. Papers*, pp. 452–453.

[49] L.-M. Lee and C.-K. K. Yang, "Adaptive low-jitter LC-based clock distribution," in *ISSCC 2007 Dig. Tech. Papers*, pp. 182–183.

[50] G.-Y. Wei, J. Kim, D. Liu, S. Sidiropoulos, and M. A. Horowitz, "A variable-frequency parallel I/O interface with adaptive power-supply regulation," *IEEE J. Solid-State Circuits*, vol. 35, no. 11, pp. 1600–1610, Nov. 2000.

[51] R. Farjad-Rad, W. Dally, H. T. Ng, R. Senthinathan, M. J. E. Lee, R. Rathi, and J. Poulton, "A low-power multiplying DLL for low-jitter multigigahertz clock generation in highly integrated digital chips," *IEEE J. Solid-State Circuits*, vol. 37, no. 12, pp. 1804–1812, Dec. 2002.

[52] M.-J. E. Lee, W. J. Dally, T. Greer, H.-T. Ng, R. Farjad-Rad, J. Poulton, and R. Senthinathan, "Jitter transfer characteristics of delay-locked loops—Theories and design techniques," *IEEE J. Solid-State Circuits*, vol. 38, no. 4, pp. 614–621, Apr. 2003.

[53] C. Kromer, G. Sialm, C. Menolfi, M. Schmatz, F. Ellinger, and H. Jackel, "A 25-Gb/s CDR in 90-nm CMOS for high-density interconnects," *IEEE J. Solid-State Circuits*, vol. 41, no. 12, pp. 2921–2929, Dec. 2006.

[54] B. W. Garlepp, K. S. Donnelly, J. Kim, P. S. Chau, J. L. Zerbe, C. Huang, C. V. Tran, C. L. Portmann, D. Stark, Y.-F. Chan, T. H. Lee, and M. A. Horowitz, "A portable digital DLL for high-speed CMOS interface circuits," *IEEE J. Solid-State Circuits*, vol. 34, no. 5, pp. 632–644, May 1999.

[55] C.-K. Yang and M. Horowitz, "A 0.8 μm CMOS 2.5 Gbps oversampling receiver and transmitter for serial links," *IEEE J. Solid-State Circuits*, vol. 31, no. 12, Dec. 1996.

[56] H.-T. Ng, R. Farjad-Rad, M.-J. E. Lee, W. J. Dally, T. Greer, J. Poulton, J. H. Edmondson, R. Rathi, and R. Senthinathan, "A second-order semidigital clock recovery circuit based on injection locking," *IEEE J. Solid-State Circuits*, vol. 38, no. 12, pp. 2101–2110, Dec. 2003.

[57] J. Alexander, "Clock recovery from random binary signals," *Electron. Lett.*, vol. 11, no. 22, pp. 541–542, 1975.

[58] J. Lee and B. Razavi, "A 40-Gb/s clock and data recovery circuit in 0.18- μm CMOS technology," *IEEE J. Solid-State Circuits*, vol. 38, no. 12, Dec. 2003.

[59] M. Meghelli, B. Parker, H. Ainspan, and M. Soyuer, "SiGe BiCMOS 3.3-V clock and data recovery circuits for 10 Gb/s serial transmission systems," *IEEE J. Solid-State Circuits*, vol. 35, no. 12, pp. 1992–1995, Dec. 2000.

[60] J. Savoj and B. Razavi, "A 10 Gb/s CMOS clock and data recovery circuit with a half-rate binary phase/frequency detector," *IEEE J. Solid-State Circuits*, vol. 38, no. 1, pp. 13–21, Jan. 2003.

[61] P. Hanumolu, M. G. Kim, G.-Y. Wei, and U.-K. Moon, "A 1.6 Gbps digital clock and data recovery circuit," in *Proc. IEEE Custom Integrated Circuits Conf.*, 2006, pp. 603–606.

[62] C. Hogge, "A self-correcting clock recovery circuit," *J. Lightwave Technol.*, vol. LT-3, pp. 1312–1314, Dec. 1985.

[63] J. Savoj and B. Razavi, "A 10-Gb/s CMOS clock and data recovery circuit with a half-rate linear phase detector," *IEEE J. Solid-State Circuits*, vol. 36, no. 5, pp. 761–768, May 2001.

[64] K.-L. J. Wong, H. Hatamkhani, M. Mansuri, and C.-K. K. Yang, "A 27-mW 3.6-Gb/s I/O transceiver," *IEEE J. Solid-State Circuits*, vol. 39, no. 4, pp. 602–612, Dec. 2003.

[65] P. Hanumolu, G.-Y. Wei, and U.-K. Moon, "A wide tracking range 0.2–4 Gbps clock and data recovery circuit," in *IEEE Symp. VLSI Circuits Dig. Tech. Papers*, 2006, pp. 88–89.

[66] J. Lee and M. Liu, "A 20 Gb/s burst-mode CDR circuit using injection-locking technique," in *ISSCC 2005 Dig. Tech. Papers*, pp. 46–47.

[67] A. Emami-Neyestanak, S. Palermo, H.-C. Lee, and M. A. Horowitz, "CMOS transceiver with baud rate clock recovery for optical interconnects," in *IEEE Symp. VLSI Circuits Dig. Tech. Papers*, 2004, pp. 410–413.

[68] A. Demir, E. Liu, A. Sangiovanni-Vincentelli, and I. Vassiliou, "Behavioral simulation techniques for phase/delay-locked systems," in *Proc. IEEE Custom Integrated Circuits Conf.*, 1994, pp. 453–456.

[69] W. Gardner, *Introduction to Random Processes*. New York: McGraw-Hill, 1990.

[70] W. Beyene, "Modeling and analysis techniques of jitter enhancement across high-speed interconnect systems," in *Proc. IEEE Electrical Performance of Electronic Packaging Conf.*, Oct. 2007, pp. 29–32.

[71] T. Lee, *Planar Microwave Engineering: A Practical Guide to Theory, Measurements and Circuits*. Cambridge, U.K.: Cambridge Univ. Press, 2004, ch. 19.

[72] R. C. Walker, "Designing bang-bang PLLs for clock and data recovery in serial data transmission systems," in *Phase-Locking in High-Performance Systems*, B. Razavi, Ed. New York: IEEE Press, 2003, pp. 34–45.

Bryan Casper (S'96–M'98) received the M.S. degree in electrical engineering from Brigham Young University, Provo, UT.

He is currently leading the high-speed signaling team of Intel's Circuit Research Laboratory, Hillsboro, OR. In 1998, he joined the Performance Microprocessor Division of Intel Corporation and worked on the development of Pentium and Xeon processors. Since 2000, he has been a Circuit Researcher responsible for the research, design, validation, and characterization of high-speed mixed-signal circuits and I/O systems.

Frank O'Mahony (S'99–M'03) received the B.S., M.S., and Ph.D. degrees in electrical engineering from Stanford University, Stanford, CA, in 1997, 2000, and 2004, respectively. His doctoral research focused on resonant clock distribution techniques for high-performance microprocessors.

Since 2003, he has been with Intel's Circuit Research Laboratory, Hillsboro, OR. His current research interests include high-speed and low-power data links, clock generation and distribution, and design techniques for low-noise, variation-tolerant clocking and signaling circuits.

Dr. O'Mahony was the recipient of the ISSCC 2003 Jack Kilby Award for Outstanding Student Paper.