Techniques for Clock Skew Reduction over Intra-Die Process, Voltage, and Temperature Variations

by

JEFF MUELLER

M.B.A., University of Texas at Austin, 2001
M.S.E.E., University of Illinois at Urbana, 1995
B.S.E.E., University of Missouri at Rolla, 1991

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS OF THE DEGREE OF

DOCTOR OF PHILOSOPHY

in

The Faculty of Graduate Studies

(Electrical and Computer Engineering)

THE UNIVERSITY OF BRITISH COLUMBIA
(Vancouver)

September 2010

Abstract

Synchronous clock distribution continues to be the dominant timing methodology for very large scale integration circuit designs. As processes shrink, clock speeds increase, and die sizes grow, an increasingly larger percentage of the clock period is being lost to skew and jitter. New techniques to reduce clock skew and jitter must be deployed to utilize the faster clock frequencies possible with future process technologies, especially in the presence of on-chip process-voltage-temperature (PVT) variations.

This dissertation first proposes a *pre-silicon* design modification to symmetric clock buffers of traditional clock distribution networks. Specifically, clock performance is improved by targeting the critical clock edge (the edge activating rising edge-triggered flip-flops) while relaxing the requirements of the non-critical edge. This system uses alternating, asymmetric clock buffers to focus inverter resources on only one edge of the clock pulse; hence, it is called *Single Edge Clocking (SEC)*. A novel re-design of the traditional clock buffer is proposed as a drop-in replacement for existing clock distribution networks, yielding timing performance improvements of over 20% in latency and skew, and up to 30% in jitter; alternatively, these timing advantages could be traded off to reduce clock buffer area and power by 33% and 12%, respectively.

PVT variations, especially intra-die, increasingly upset the distribution of a synchronized clock signal, even in properly balanced clock tree networks. Hence, active clock deskewing becomes necessary to tune out unwanted clock skew after chip fabrication. This dissertation proposes a *post-silicon* autonomous deskewing system using tunable buffers to dynamically reduce clock skew. The operation of a specially designed phase detector is described, and four such phase detectors are used to construct a stable, autonomously locking *Quad Ring Tuning (QRT)* configuration that effectively links together four distributed Delay-Locked Loops (DLLs) without the need for any system-level controller. This cyclic, unidirectional, self-controlled, quad-DLL ring tuning technique is then implemented hierarchically to dynamically adjust clock signal delays across an entire chip during normal circuit operation. A simple form of the two-level QRT system is presented for a generic H-tree distribution network, demonstrating stable locking behavior and more than 50% average reduction in full-chip clock skew.

Preface

The research presented in this dissertation has been previously published in two Institute of Electrical and Electronics Engineers (IEEE) Conference papers and will soon be published in an IEEE Journal paper, as detailed below.

The research covered in Chapter 3 has been published in: J. Mueller and R. Saleh, “Single Edge Clock (SEC) distribution for improved latency, skew, and jitter performance,” *Proceedings of Very Large Scale Integration Design (VLSID) Conference*, pp. 214-9, Jan. 2008.

Most of the research covered in Chapter 4 has been published in: J. Mueller and R. Saleh, “A tunable clock buffer for intra-die PVT compensation in Single-Edge Clock (SEC) distribution networks,” *Proceedings of International Symposium on Quality Electronic Design (ISQED)*, pp. 572-7, Mar. 2008.

The research covered in Chapter 5 will be published in: J. Mueller and R. Saleh, "Autonomous, multi-level ring tuning scheme for post-silicon active clock deskewing over intra-die variations," to be published in *IEEE Transactions on Very Large Scale Integration (TVLSI) Systems, 14 pages,* 2010.

These three papers and this dissertation present research conducted by "the author", Jeff Mueller, in collaboration with "the supervisor", Dr. Resve Saleh, who provided consultation on research and editing assistance on publications.

Table of Contents

List of Tables

List of Figures

List of Symbols

β	beta of an inverter (=W_P/W_N)
η	degradation factor of SEC (=t_{negl}/t_{crit})
bbit1, bbit2	binary tuning bits for a tcp-INVf
DLL	delay locked loop
FF	flip-flop
INVf	inverter used as falling edge clock buffer in SEC
INVr	inverter used as rising edge clock buffer in SEC
INVt	inverter used as clock buffer in TC
ISF	impulse sensitivity function (a measure of jitter)
LT/LB/RB/RT	four quadrants of QRT (left-top, left-bottom, right-bottom, and right-top)
M.C.	Monte Carlo simulation
PLL	phase locked loop
PVT	variations created by process, voltage, and temperature
PW	pulse width of the clock signal
QRT	quad ring tuning (i.e. RT with four tuning zones)
QRT-2	single level QRT implemented at clock level-2
QRT-2&3	hierarchical QRT implemented at clock level-2 and level-3
r.s.a.	range sharing algorithm
RT	ring tuning
S_{th}	skew threshold of UDD
SBM	single branch model
SEC	single edge clocking
ss	step size of TB
s.w.a.	spinning wheel algorithm
tcp-INVf	tunable INVF buffer delayed by load capacitance
tsi-INVf	tunable INVf buffer delayed by a starved inverter
tbit1, tbit2, tbit3	thermometer code tuning bits for a tsi-INVf
t_{crit}	critical edge transition time of the clock signal
t_{negl}	neglected edge transition time of the clock signal
T_{clk}	period of the clock signal
TB	tunable buffer
TC	traditional clocking
TCG	thermometer code generator
UDD	up/down detector
W_N	width of the NMOS device of an inverter
W_P	width of the PMOS device of an inverter

Acknowledgements

Thanks to everyone who has helped me along the way!

Special thanks go to:	Supervisor: Dr. Resve Saleh
	Co-supervisor: Dr. Shahriar Mirabbasi
	Committee Members: Drs. Steve Wilton and John Madden
	Head's Nominee: Dr. Mark Greensteet
	Chair: Dr. Andre Ivanov.

Also, thanks to our SoC Lab experts: Dr. Roberto Rosales
Roozbeh Mehrabadi.

Thanks to my fellow SoC students, most notably: Dr. Dipanjan Sengupta
Dr. Xiongfei Meng
Dr. Peter Hallschmid.

Thanks to our funding sources:

Natural Sciences and Engineering Research Council of Canada (NSERC)
PMC-Sierra, Burnaby, BC Canada (including use of their design tools)
Canadian Microelectronics Corporation (CMC).

Finally, and most importantly, thank you to my family -- especially Mom and Dad who have always been there for me. And lastly, Thank You Dana for everything.

Chapter 1 : Introduction and Overview

1.1 Motivation

Synchronous clock delivery has always been a major design challenge in application specific integration circuits (ASICs). Without an adequate clock signal, synchronous circuits will experience setup time and/or hold time violations on flip-flops that may lead to logic errors, and they will consequently fail to operate as expected. Much of the effort in clock design is spent on reducing two key timing issues of *skew* and *jitter* [1]. *Skew* is defined as the difference in clock edge arrival times for two different clock wires (or clock paths). Skew is typically caused by mismatches in the branches of the clock distribution network. *Jitter* is defined as the difference in clock edge arrival times with respect to its expected position for a single clock wire (or clock path). Jitter is typically caused by circuit-induced noise effects on the clock signal. Today, significant area, power, and metal resources are required to distribute a synchronous, high-speed clock across a large die with minimal skew and jitter. Yet, the synchronous clock delivery problem is expected to be even more difficult in the future, as seen in Figure 1.1.

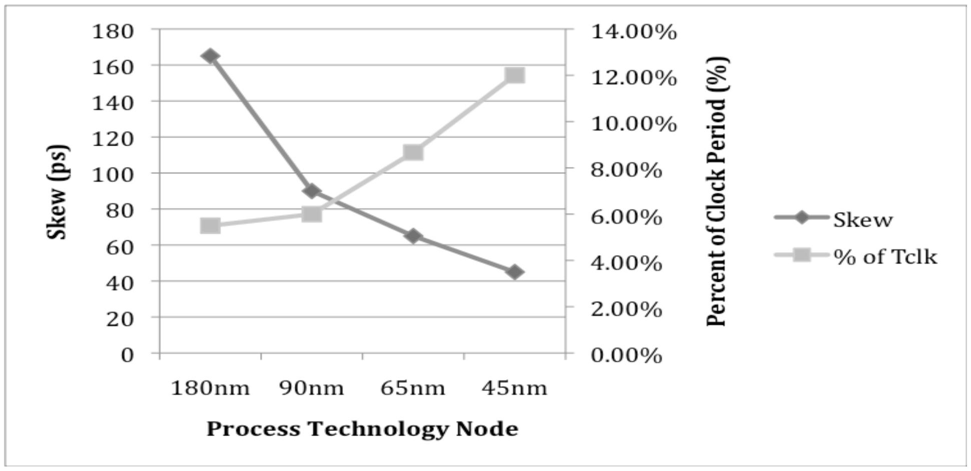


Figure 1.1 : Clock skew over process technology (from [2])

Table 1-1 : Clock Period Lost to Skew from Figure 1.1 Data

Technology	Clock Period	Skew	% Lost to Skew
180nm	3ns	165ps	5.5%
90nm	1.5ns	90ps	6.0%
65nm	750ps	65ps	8.7%
45nm	375ps	45ps	12.0%

Although the absolute level of clock skew is seen to be decreasing in the graph of Figure 1.1, the amount of clock skew as a percentage of clock period is in fact increasing for each new process technology node. Assuming a reasonable clock frequency of 333MHz for the 180nm node, and doubling the clock frequency for each subsequent node, the percent of clock period lost to skew is calculated for each process technology and displayed on Figure 1.1 and in Table 1-1. The percent of clock period lost to skew is seen doubling from the 90nm to the 45nm node. While these are somewhat conservative predictions, in reality, predictions for overall variability in process nodes beyond 45nm have been shown to further exacerbate the clock skew problem, as well as jitter. In fact, according to projections in [3], as processes scale further down into the nanometer range, larger percentages of the clock period will be lost to clock skew and clock jitter unless there are major changes in the clock design approaches. Therefore, new techniques to improve overall clock timing performance (clock skew and jitter) are needed.

1.2 Synchronous Clock Design Issues

The vast majority of present-day ASICs use synchronous clocks [1] driving single-edge triggered flip-flops (FFs). Furthermore, even the larger chip sizes of very large-scale integrated (VLSI) circuit designs, which in the future may adopt a *globally asynchronous* clock, will still rely on *locally synchronous* clocks for logical circuit operation [4]. Therefore, synchronous clock circuit design will be the focus of the research here since today it is a critical component of IC design and will continue to be so in the future.

The design and architecture of many commonly used clock distribution systems are thoroughly described in several papers [5][6]. Standard inverters, commonly referred to as clock buffers, are typically used to drive the clock signal from the root (the phase-locked loop or delay-locked loop source) to the leaves (the FFs) of a clock distribution network (e.g., H-tree, grid, hybrid) [7][8][9][10][11]. Three different forms of clock networks are illustrated in Figure 1.2, with the H-Tree in some form being the most common.

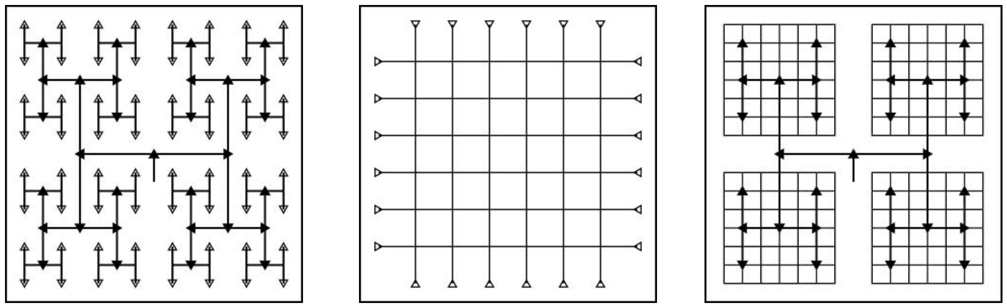


Figure 1.2 : Basic Clock Distribution Networks: H-tree, Grid, and Hybrid (from [7])

The primary goal of these different architectures is to reduce maximum clock skew and maximum jitter. For the purposes of this research, skew will be defined here as the difference in clock edge arrival times for two different clock branches, $\Delta t_{i\text{-}j}=t_i\text{-}t_j$. The arrival time for each unique clock branch will be measured at its terminal clock leaf and represented as t_i. Therefore, the *maximum clock skew* will be given by the absolute value maximum of $\Delta t_{i\text{-}j}=t_i\text{-}t_j$ for $i \neq j$, where i and j cover every clock leaf of the entire chip. Likewise, jitter will be defined as the difference in clock edge arrival time for a single clock branch. Therefore, the jitter at any given clock leaf is represented as $\Delta t_k = t_{k,max} - t_{k,min}$, and the *maximum jitter* is given by the absolute value maximum of Δt_k, where k covers every clock leaf of the entire chip. Illustrated representations of skew and jitter will be provided later in Section 2.1.4.

Skew can be caused by mismatches in the routing of the clock distribution network and also by intra-die process variations of one clock path, or branch, versus another clock branch. Ideally, the routing paths between the clock's root node and two different FFs should be laid out identically; but this option is often not possible and, hence, skew is created. Furthermore, the process variations, supply voltage fluctuations, and temperature gradients across the chip during its operation are increasingly significant factors in skew today [12][13][14]. Thus, both sources of clock skew (layout mismatches and process, voltage, and temperature variations) should be reduced. In addition, clock *latency*, which is the total path delay from the clock root to its leaves, where FFs are connected, is another useful timing metric since a lower latency often implies a lower skew.

Jitter is expected to be a significant source of problems for clock designers according to the projections of [3]. The effects of supply noise, coupling noise, and other noise sources are expected to increase in the future and significantly reduce the usable clock period due to jitter. At the same time the clock period will continue to shrink with each technology generation and further exacerbate the problem. Previous work on jitter, or phase noise as it is often called by analog designers, has lead to the metric of Impulse Sensitivity Function (ISF) [15], which relates jitter performance to clock edge transition slope. When the clock signal is stable (at a high or low level) or when the clock's non-critical edge is transitioning, then noise impulses are not a concern in terms of jitter. However, each transition of the clock's critical timing edge leaves it exposed to these noise impulses and therefore vulnerable to jitter. So, if the critical edge-rate of the clock signal can be sharpened, the jitter performance will likely improve. Other techniques to shield the clock and reduce noise injection levels can greatly improve the jitter as well.

1.3 PVT Variations

Circuit designers have long been aware of the three primary categories of variation mentioned earlier that affect circuit delays across a chip: *Process Parameters* for NMOS and PMOS devices, *Supply Voltage*, and *Temperature* -- collectively referred to as 'PVT'. Each one varies independently, although there may be correlations between them. These variations could generally be different from within a chip, from chip-to-chip, and from wafer-to-wafer due to integrated circuit (IC) wafer processing and in-situ system performance. However, the chip-to-chip variations (an example of which is shown in Figure 1.3) and wafer-to-wafer variations would equally impact all components and branches of a single chip's clock distribution network. Accordingly, they would not significantly contribute to clock skew within a given chip. Therefore, the greater concern for clock timing performance is due to *intra-chip PVT variation* [12].

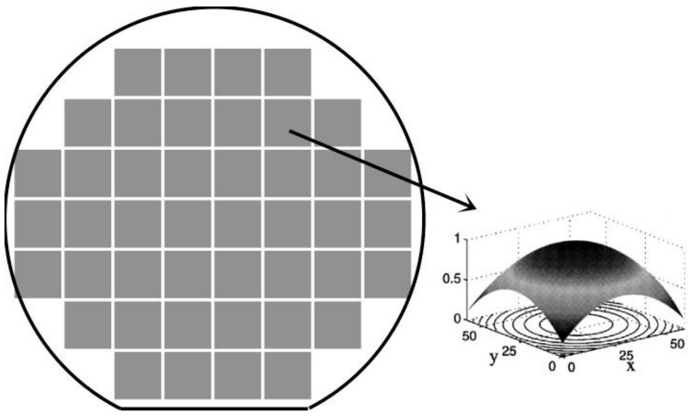


Figure 1.3 : Process variation over one die from a wafer (from [16])

The diagram of Figure 1.4 demonstrates an idealized representation of how P, V, and/or T could vary across an H-tree clock distribution network of a single chip. A variety of other gradients are possible by repositioning the center of the variation(s) elsewhere on the clock distribution tree, or by distorting the geometric shapes of the contour lines. These PVT variations could bring about time-invariant differences between the clock branches, or they could be time-varying during circuit operation. Typically, process variations tend to be static in nature but may drift slowly over time, while supply voltage and temperature variations are truly dynamic in nature and depend primarily on the layout of the power/ground grids, the floorplan of the blocks on the chip, and the activity levels of the blocks themselves. As a result, PVT variations must be expected throughout any clock network and will affect latency, skew, and jitter on the clock signal [17][18][19][20][21].

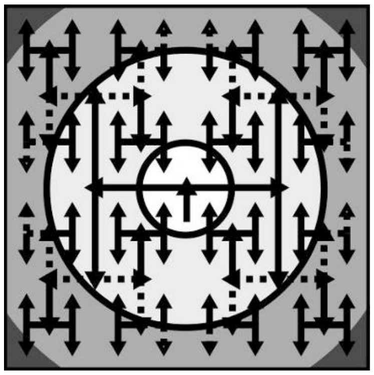


Figure 1.4 : Concentric circles representing P, V, and/or T variations (from [2])

Process variations can be further classified into two distinct categories based upon their degree of correlation across the chip. *Random variations* are considered uncorrelated spatially and therefore they can be modeled as random variables at every location on the chip. *Spatial variations* are highly-correlated across the chip and therefore are modeled as a spatial pattern or gradient across the chip, as in Figure 1.4. Some circuits (e.g., small, matched, localized blocks) may be more affected by random variations, while others (e.g., large, global, dispersed blocks) may be more affected by spatial variations; so, both random and spatial variations must be considered in clock timing analysis.

1.4 Skew and Jitter Reduction and Compensation

Various techniques involving clock buffer sizing and placement, along with clock distribution and power grid balancing, and symmetric layout have been used to keep PVT-induced clock skew and jitter under control [22][23][24][25][26]. Predictions for future process nodes shown earlier indicate that increases in clock skew and jitter will limit the speed gains (due to clock period reductions) that smaller device sizes may offer [3]. Therefore, new and better techniques to reduce clock skew and jitter must be developed and implemented. The main objective in this research is to *greatly reduce clock skew* without degrading clock jitter (or perhaps even improving it) in the presence of PVT variations, and this achievement constitutes the key contribution described in this dissertation.

The research described in the chapters to follow begins by proposing a new method which is minimally disruptive to the ASIC flow and which does not require any changes to the interconnect wires or any increase in the clock buffer area or power. Yet, this new *pre-silicon* method significantly improves the clock network's timing performance and reduces the effects of intra-chip PVT variations on the clock distribution. This new method is called *Single-Edge Clocking (SEC)*. It focuses clock buffer resources on the timing-critical clock edge to reduce skew and jitter by reducing latency and increasing the edge rate. SEC systems propagate asymmetrical clock signals to favor the clock's critical edge over its other, non-critical, edge. To implement the SEC method, two asymmetric, single-edge buffers are required to alternately propagate the critical edge through the clock network, and these asymmetric buffer designs are also described in detail.

The results will show that significant timing gains can be realized by replacing the single chain of equal rise/fall buffers with an alternating pattern of asymmetric rise/fall buffers that focus the majority of their current drive only on the one truly critical clock edge -- the edge that ultimately activates the leaf FFs. The other, non-critical, clock edge will be somewhat degraded. However, depending upon the circuit's clock period and timing requirements, that degradation may present no real liability to the circuit's overall performance; therefore, these significant timing gains can be made in nearly any synchronous clock network at little or no additional costs.

Such *pre-silicon* techniques work well to reduce clock skew by adjusting for design and/or layout related imbalances in the clock distribution network and its loads. In fact, the primary advantage of any *pre-silicon* technique is its ability to reduce clock skew prior to the major expense of reticle (or mask set) creation, but its primary disadvantage is its inability to address the causes of clock skew that only arise during circuit operation (after chip fabrication). As previously indicated, each processes shrink causes the on-chip or intra-die variability due to numerous PVT variations to grow, and these potentially sizeable post-silicon variations cause clock skew (and jitter) to rapidly consume an increasingly larger percentage of the clock period [12][13]. Since these full-chip variations are nearly impossible to predict and/or control during the design/layout phase, they cannot be addressed until after chip fabrication. Therefore, various types of *post-silicon* techniques will become necessary [22] as PVT variations continue to increase. It is expected that most future designs will incorporate some form of *post-silicon* clock adjustment (via tunable clock buffers, for example) to compensate for the majority of intra-die PVT variations [27][28].

A worthwhile direction for *post-silicon* skew compensation is to configure active (always-on) tuning loops that self-monitor for clock skews and appropriately adjust their tunable buffers [29]. This post-silicon adjustment necessitates the creation and proper insertion of adjustable or tunable components within the clock distribution network – namely, tunable clock buffers. It also requires methods of measuring skew differences of clocks at the appropriate points in the circuit, which are then fed back to the tunable buffers to enable the necessary adjustments. Such systems are commonly referred to as adaptive circuits or *active deskewing* circuits. As [28] clearly states, new scaling issues will cause more variability in circuits, and therefore designers will likely use more of these adaptive circuits. The work described in later chapters will focus on deskewing networks consisting of multiple clock zones without the availability of chip-wide reference clocks or signals. In particular, this research proposes a novel method of interlinking multiple instances of delay-locked loops (DLLs) into a stable ring configuration. This technique, called ring tuning (RT), will be presented in full for the post-silicon skew compensation of a typical H-tree clock distribution network.

1.5 Research Statement

Research Objective:

To build a better clock distribution network for ASICs that is able to automatically compensate for on-chip PVT variations in order to reduce clock skew and clock jitter with no significant increase in overall clock network area or power consumption.

Methodology:

- Develop a technique that optimizes only one edge of the clock signal -- single-edge clocking (SEC).
- Design asymmetric clock buffers to be used in conjunction with SEC.
- Extend the buffers to provide a tunable mechanism and determine appropriate locations and sizes of these SEC tunable buffers.
- Design a circuit to measure skew between clock edges at different points in the chip for use in automatic skew compensation.
- Develop an overall architecture for clock edge calibration (i.e., setting the adjustable clock buffer tuning values) in order to reduce clock skew.

1.6 Thesis Organization

The rest of this thesis is organized as follows. Chapter 2 provides the needed background and overview of clock distribution systems. Chapter 3 presents the new *Single Edge Clocking* method for pre-silicon skew reduction. Chapter 4 presents the tunable buffer for SEC, and introduces more specialized SEC applications. Chapter 5 presents the novel *Ring Tuning* architecture for post-silicon skew compensation. Chapter 6 summarizes the findings and recommends directions for future work.

Chapter 2 : Synchronous Clock Background

This introductory chapter provides a background and overview of synchronous clocking techniques and associated definitions for clock metrics. The key issues for clock design are presented, along with a few techniques for clock improvements based on previous work. Section 2.1 begins with a brief review of the clock function, starting with the basic concepts and continuing with clock networks and flip-flops. Section 2.2 discusses Process, Voltage, and Temperature (PVT) variations and their effect on the clock signal. Section 2.3 presents traditional clock buffer design concepts and more advanced adaptive deskewing compensation methodologies. Finally, Section 2.4 summarizes the chapter.

2.1 Clock Function and Design Issues

The clock circuit is the central heartbeat of most VLSI chips and is therefore a critical design component in both digital and analog circuits today. The clock source's primary function is to generate a periodic clock signal that regulates the timing of the entire chip [5]. Almost all circuits designed today are synchronous; although there are some exceptions that are *globally asynchronous,* they still make use of *locally synchronous* clocks [1][4], which is known as 'GALS'. Therefore, this thesis will focus only on the improvement of clocks in synchronous circuits.

Typical synchronous circuits rely upon a highly precise clock signal to trigger the computation of multiple combinational logic blocks which may be widely-distributed across an entire chip. The entire clock system can be divided into three main components: the clock source, the timing flip-flops (FFs), and the clock distribution network.

2.1.1 Clock Source

High-frequency clock signals are typically generated on-chip based on a low-frequency system clock provided at one of the chip input pins. Complex analog circuits designs, such as phase-locked loops (PLLs) or delay-locked loops (DLLs) are commonly used to produce a square-wave signal of the appropriate frequency for the entire chip's timing needs. This "source" clock signal is intended, ideally, to trigger each of the numerous FFs attached to the clock network at exactly the same instance in time. However, since the source clock is only available at the output of the PLL and the timing FFs are widely-distributed across the entire chip, a vast distribution network must be employed to deliver the clock signal ubiquitously and precisely.

2.1.2 Flip-Flop Timing Issues

The ultimate timing-related performance of a synchronous circuit depends upon the simultaneous triggering of all the timing FFs. As shown in Figure 2.1, the minimum clock period is ideally the sum of delays through the FFs, logic, interconnect and the FF setup time. However, this simultaneity may be difficult to accomplish since there are many differences (to be elaborated further in this chapter) through the buffers and interconnects from the clock source to the leaf nodes. The key is to identify and to compensate for these differences in the design of the clock distribution network. Hence, any technique to improve the timing accuracy of the final clock signal(s), which triggers the FF(s), will improve the timing performance of the entire chip.

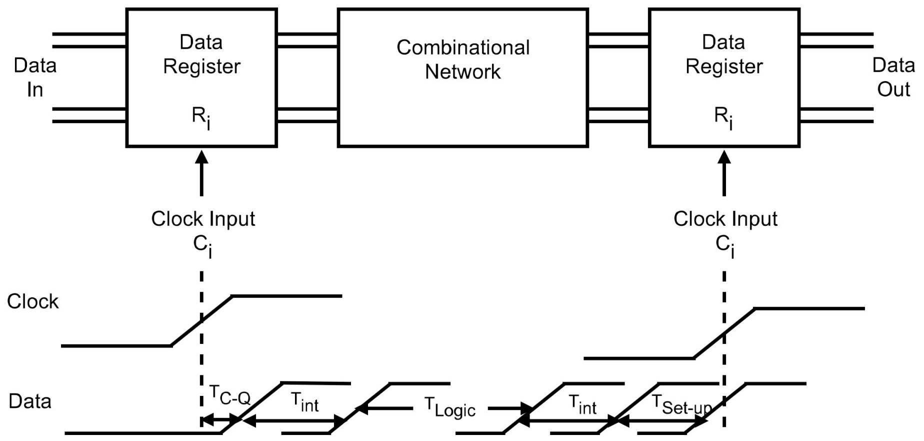


Figure 2.1 : Ideal Synchronous Clock Period (from [5])

The majority of circuit designs utilize single-edge triggered FFs, such as the one shown in Figure 2.2. Although implementations may vary, this type of FF can be viewed as a D-latch pair. By common definition, latches are *transparent* when their clock, or CK, line is *high*, while FFs are only triggered by a clock *edge*. As the diagrams show, the clock's low level sets up the Q output of the first D-latch, and the clock's subsequent rising edge actually triggers that input value to the FFs output. Therefore, this design is referred to as a *rising-edge triggered* FF. The only timing critical event is the clock's rising edge, provided that the clock's low level is long enough to pass the IN data through the first D-latch. The falling edge of the clock is only used to set up for the next rising edge that triggers the FF again, so the falling edge can occur at any time before the end of the clock period without FF error.

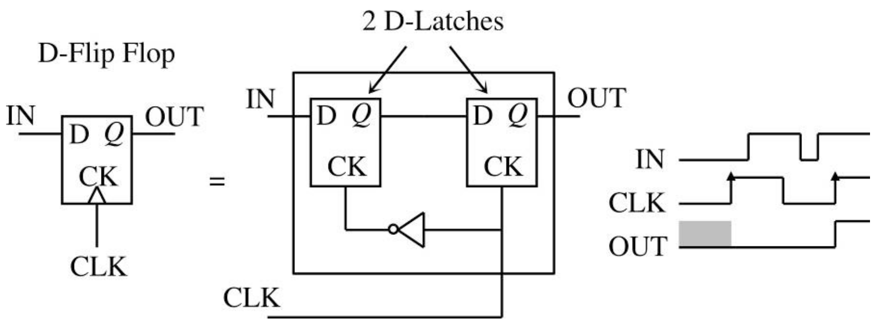


Figure 2.2 : Rising edge-triggered D Flip-flop (from [30])

For proper operation, the FF must satisfy certain setup and hold requirements, as shown in Figure 2.3. The FF setup time is defined as the time required for the input data signal to be valid before the incoming clock edge arrives to trigger its capture. Violations of the setup time are due to slow paths in the combinational logic. The setup time adds directly to the idealized clock period, shown earlier in Figure 2.1, and effectively slows down the operating frequency of the chip. The hold time is the time required for the input data to remain valid after the clock edge transition has occurred. Violations are due to fast paths in the logic circuit and they must be slowed down, typically by using delay buffers, to avoid timing problems. As a result, designers work to minimize both setup and hold times of FF’s.

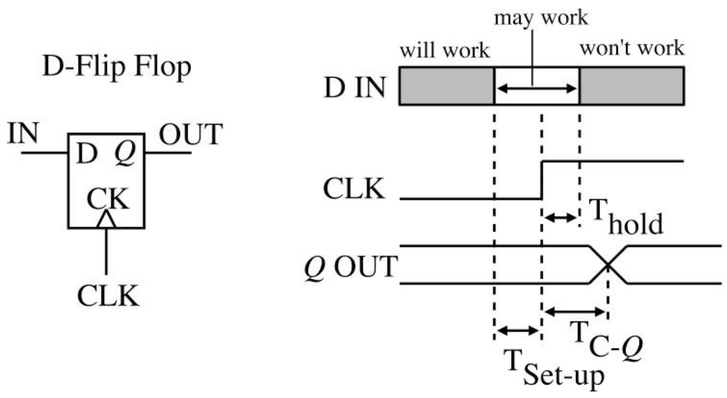


Figure 2.3 : Flip-flop setup and hold time definitions (from [30])

The effect of the clock arriving at different times to different FF’s may cause them to inadvertently capture incorrect data, which leads to a malfunction in an otherwise working logic design. In effect, the setup and hold times of the FFs are violated by late-arriving or early-arriving clock signals. To be safe, the clock period must be extended by the expected degree of clock edge variation to ensure that the design works properly. While the frequency of operation of the design may be reduced to fix set-up time violations, oftentimes the resulting slower frequency may not satisfy the design specification. Therefore, it is important to identify the sources of the clock edge uncertainty and to reduce this uncertainty as much as possible, as described in the sections to follow.

2.1.3 Clock Distribution Network Design

While a basic understanding of the clock source and leaf FFs is necessary, the research in this dissertation is exclusively focused on improving clock signal delivery via the clock distribution network. The background on this topic is covered in the section below.

2.1.3.1 Basics of Clock Networks

The clock distribution network is usually constructed as either a branching clock tree or some form of an interconnected clock tree/grid hybrid. Regardless of the chosen architecture, the typical clock distribution network is merely composed of two fundamental circuit elements: *buffers* and *wires*, as depicted in Figure 2.4 for a representative clock tree.

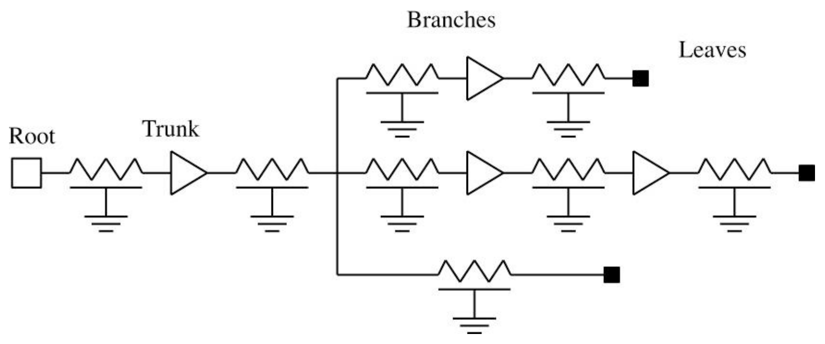


Figure 2.4 : Clock Tree Components [Root, Branches, and Leaves] (from [5])

The wires may use all metal layers, but the major trunks are typically routed using the upper layers of metal that tend to be wider and thicker, since large currents are expected to flow through the wires on every cycle. These metal wires are actually distributed RC lines which can be modeled as lumped RC circuits using a π-model for simulation purposes using the effective resistance and capacitance per unit length values. The layout and routing of the clock lines can be quite complex since they compete with power/ground lines and other global signals that also use the upper layers of metal.

In keeping with the "tree" metaphor, the final clock signals (outputs of the final clock buffers) are commonly referred to as the "leaf" nodes of the clock; likewise, the initial or source clock signal is referred to as the "root" node of the clock, and the pathway from the root to any leaf node is a clock "branch". A standard clock tree will have only one root, but multiple branches and leaves. It is the design (placement, sizing, routing, etc.) of these *buffers* and *wires* that ensures the accurate and precise delivery of the clock signal from its single source (root) to its numerous, widespread triggering nodes of FFs (leaves).

2.1.3.2 Basics of Clock Buffer Design

Clock buffers in CMOS circuits are predominantly large CMOS inverters, as shown in Figure 2.5. They must be large in order to propagate clock signals quickly to the target flops and to drive the large interconnect lines between them. These inverters are sized and placed throughout the clock tree according to various factors usually involving a trade-off between timing (including edge rates and skew), power, and area.

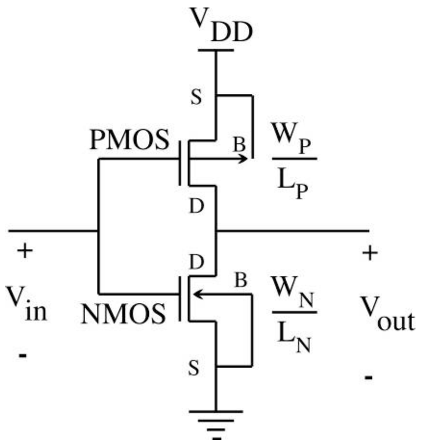


Figure 2.5 : Typical CMOS Inverter (from [30])

The principle design consideration for a CMOS inverter is the ratio of the PMOS width, W_P, to NMOS width, W_N [31]. The 50% of VDD crossing point for the timing response equations of the NMOS and PMOS transistors can be used as a measure of the propagation delays for the high-to-low and low-to-high transitions, respectively [30]. A first-order approach to creating a symmetrical clock inverter (i.e., the high-to-low and low-to-high propagation delays are equal) is by plugging the large-scale effective resistance values, R_{eff} (30kΩ/□ for PMOS, 12.5kΩ/□ for NMOS) into the propagation delay equation and setting the two equations equal to each other, with L_N=L_P, leading to:

$$W_P/W_N = 30k\Omega/12.5k\Omega = 2.4 \tag{2-1}$$

Therefore, if the clock buffers are designed to be symmetrical, the PMOS device is customarily 2.4 times larger than the NMOS device for equal rise/fall delays. In the past, this ratio resulted largely from the difference of mobility between holes (for PMOS) and electrons (for NMOS). Today, it is primarily due to differences in velocity saturation of the two types of devices. However, for the remainder of this dissertation, this 2.4X ratio will be rounded up to 2.5X for consistency with other publications. The 2.5X ratio provides a better balance between equal rise/fall times and optimal signal propagation delay through a chain of inverters. So, this practical value (from HSPICE models derived from actual process data) is more appropriate for use when designing inverters and clock buffers.

A secondary design consideration of the clock buffers is their input and output capacitances. The purpose of a clock buffer is to propagate the input signal to the output by properly driving the output load, but any buffer placed in a signal path, such as the clock tree, will necessarily affect the signal itself. Most significantly, a clock buffer will contain input and output capacitances that must be considered in the signal path's load calculations.

The input capacitance of a CMOS inverter is commonly referred to as the gate capacitance, since the gate terminal is its sole input connection. Basic first-order analysis of a typical CMOS inverter results in the following equation for the inverter's gate capacitance:

$$C_G = C_{Gn} + 2\ C_{OL} + C_{Gp} + 2\ C_{OL} = (C_{ox}L + 2\ C_{ol})(W_N + W_P) \tag{2-2}$$

Here, C_{Gn} and C_{Gp} are gate capacitances of NMOS and PMOS devices, respectively, and C_{OL} is the overlap capacitance. Since the first parenthesized term of this expression is a constant for a given process technology, the gate capacitance, or input capacitance, of a clock buffer is proportional to the sum of the transistor widths, or $(W_N + W_P)$.

The output capacitance, commonly referred to as the self-capacitance, of the same CMOS inverter can be analyzed to obtain:

$$C_{self} = C_{DBn} + C_{DBp} + 2\ C_{OL} + 2\ C_{OL} = C_{eff}\,(W_N + W_P) \tag{2-3}$$

where C_{DBn} and C_{DBp} are junction capacitances of NMOS and PMOS devices, respectively, and C_{eff} is the effective capacitance per width which is taken to be a constant for a given process technology. As a result, the output capacitance is also seen to be proportional to the sum of the transistor widths $(W_N + W_P)$, similar to the input capacitance.

2.1.3.3 Other Clock Design Issues

Other issues in clock design concern power and area, as well as noise issues that either affect the clock or are affected by the clock [5]. Predominantly, for power and area reduction, CMOS clock distribution networks have utilized clock inverters in place of (true, non-inverting) clock buffers, and balanced clock trees in place of high-load metal grids. Additionally, clock gating [30] is commonly used to minimize the number of FFs that are unnecessarily triggered, thereby reducing the overall clock power. Interconnect noise issues are typically dealt with by shielding and spacing techniques on the clock wires [6]. Power grid noise is addressed by placing decoupling capacitors close to the large clock buffers [30].

2.1.4 Clock Timing Metrics

This thesis will focus on a common configuration of clock distribution networks; namely, CMOS inverters driving metal wires in the H-tree configuration of Figure 2.6. This regular structure will also allow for a proper, but controlled, test vehicle for the comparison of traditional techniques with new methods to improve the timing characteristics of clocks that will be presented in this thesis.

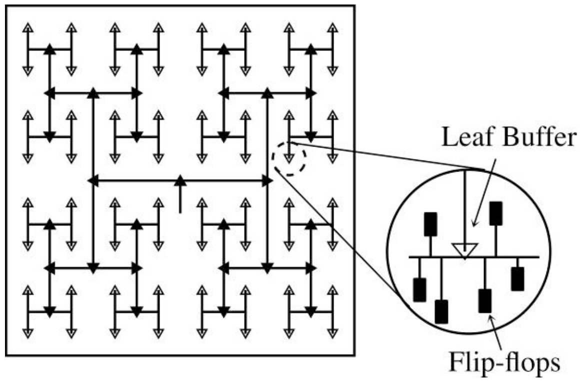


Figure 2.6 : Typical Synchronous Clock Network [H-tree and FFs] (from [7])

Three important metrics are used to assess the timing performance of a synchronous clock: latency, skew, and jitter, as shown in Figure 2.7. *Latency* is the total path delay from clock root to its leaves where flip-flops (FF) are connected. In Figure 2.7, latency would be measured as the total time for a clock signal to pass from the root buffer (#1) through all the branch wires and buffers (#1b, #2, #2b, and #3) and through the leaf buffer (#4). When two clock architectures both employ the same metal wire H-tree, have the same number of buffers, and roughly the same buffer area and power, yet one delivers the critical clock edge significantly earlier than the other, the faster method would be preferred since a lower latency often implies a lower skew.

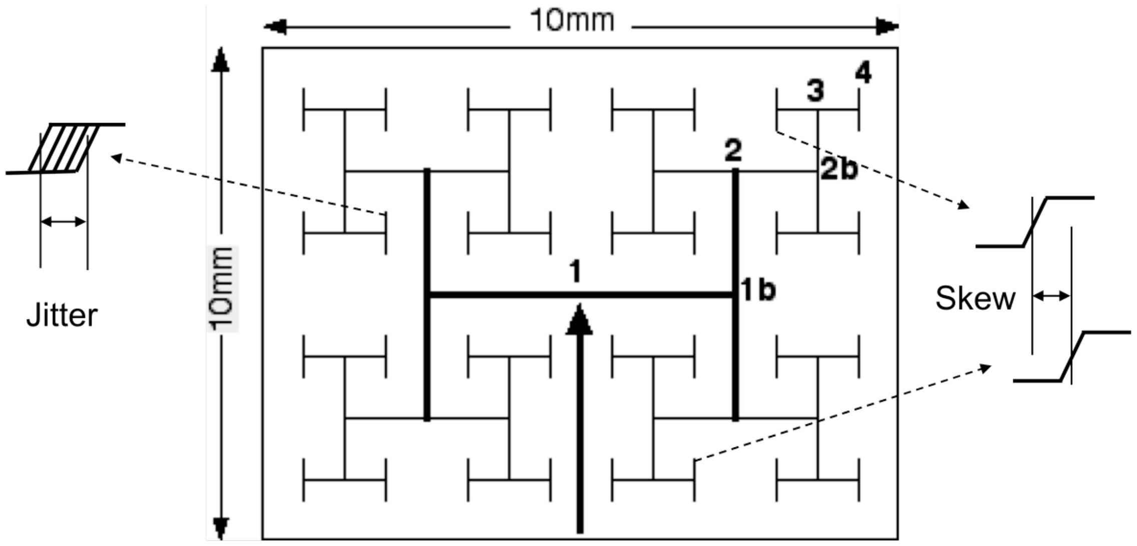


Figure 2.7 : Clock Latency, Skew, and Jitter. (#1 = root buffer and #4 = leaf buffer)

In this research no distinction between *local clock skew* (the maximum skew within a given clock zone inside of the chip) and *global clock skew* (the maximum skew within the entire chip) will be made. The design task will be to reduce full-chip clock skew; specifically, to minimize the *maximum clock skew* as defined in Section 1.2, as measured across each of the 64 leaf clocks.

Skew can be caused by mismatches in the clock distribution network and intra-die process variations of one clock path or branch versus another branch. In Figure 2.7, skew is depicted as the measured difference in latency, or delay, between *any two clock leaves*. Designers aim to minimize this value since it directly reduces the usable clock period by the clock skew. That is, it requires that the clock period be increased by at least the amount of skew. Ideally, the paths between the root node and two different FF's should be laid out identically, but this perfect symmetry is often not possible and consequently skew is created [32]. Perhaps even more importantly, the process variations, supply voltage fluctuations, and temperature gradients across the chip (i.e., PVT variations) are major factors in clock skew today.

Jitter is primarily caused by noise impulses on the clock signal, and is therefore very difficult to model and simulate using circuit simulators such as SPICE. In Figure 2.7, it is represented as the time-varying clock edge at *one clock leaf*. The amount of jitter also directly increases the clock period and therefore must be reduced as much as possible. Jitter is due to noise injection effects of the voltage supply, coupling to neighbors that are switching, and other sources such as substrate noise. These noise levels are expected to increase in the future and significantly reduce the usable clock period due to jitter [3], while at the same time the clock period will continue to shrink with each technology generation to further exacerbate the problem. As a result, designers will have to use a variety of methods to get jitter under control. The issue is that each transition of the critical edge of the clock may be exposed to these noise impulses and therefore vulnerable to jitter. Therefore, if the edge-rate of the clock signal can be sharpened, the jitter performance will likely improve. Previous work on jitter, or phase noise in the parlance of analog designers, has led to an associated metric called the Impulse Sensitivity Function (ISF), which relates jitter performance to clock edge transition slope [15].

2.2 PVT Variations and Skew

Clock distribution networks are *ideally* designed to be perfectly balanced; consequently, every clock branch is identical and no differences in latency exist between any two branches (i.e., zero skew); actually, this layout balance is difficult to achieve but designers and computer-aided design (CAD) tools have been able to reduce skew down to acceptable levels. However, in real-world designs, the variations due to process gradients, and voltage and temperature fluctuations will inevitably impact every clock branch of an actual operating chip differently [33]; therefore, techniques must be applied to the circuit models used for simulation to replicate these PVT variations to obtain early information about the degree of impact they may have on skew, and to propose clock design changes.

Programs such as HSPICE have built-in static models for simulating different process corners. In a typical process corner each of the contributing factors is listed as Fast (F), Typical (T), or Slow (S). Representation of the NMOS device is listed first, then the PMOS device (e.g., FS for fast-NMOS and slow-PMOS). The supply voltage variations can be VDD+10% (F), VDD (T), and VDD-10% (S). The temperature ranges can be low (F), nominal (T), and high (S). Best-case, worst-case and typical PVT corners can be established with various combinations of P, V and T.

The precise level of skew experienced across any clock distribution network requires highly-accurate models of random and spatial intra-chip variations [2][34][35], which are rarely available for every given process technology, fabrication site, and chip application. Therefore, *two different techniques of modeling the skew caused by PVT variations will be* *used in this research*. The first technique uses modifications in the circuit model parameters for P, V, and T in order to compare a clock with the fastest possible inverters to one with the slowest possible inverters -- this technique is referred to here as the *worst-case skew* metric. The second technique uses higher-level circuit parameters (such as transistor size or wire length) to mimic intra-die PVT variations that result in clock skew.

The *relative comparison*, which provides meaningful data of the robustness to skew of two clock distribution methods, can be obtained by simulating their *worst-case skew*. Here, the worst-case skew will be taken as the difference in latency, or path delay, of a clock branch simulated with Slowest corner (SS, 0.9V, 150^{o}C) parameters and with Fastest corner (FF, 1.1V, 0^{o}C) parameters in a 90nm CMOS process with a nominal supply voltage of 1.0V. This *worst-case skew* technique works well to compare two clock branches with two separate transient simulations; however, the difficulty of independently setting P, V, and T parameters for every component of a full-chip netlist leads to the usage of higher-level circuit parameters.

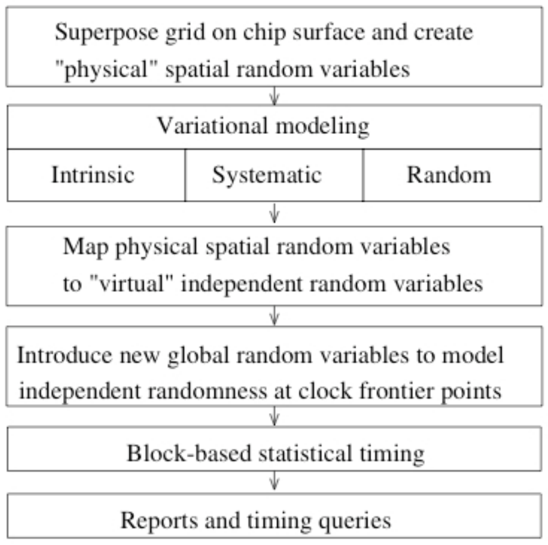

Superpose grid on chip surface and create "physical" spatial random variables
Variational modeling
Intrinsic
Systematic
Random
Map physical spatial random variables to "virtual" independent random variables
Introduce new global random variables to model independent randomness at clock frontier points
Block-based statistical timing
Reports and timing queries

Figure 2.8 : PVT Variation Skew Modeling (from [36])

The *higher-level circuit parameter technique* allows for easy and independent modification of every circuit component (transistors and wire segments) within a full-chip netlist. The full-chip schematic is partitioned into numerous sub-blocks and two Matlab created matrices (one for transistors and one for wire segments) are superimposed to create the PVT-induced variations. This higher-level variation technique can be used in the manner outlined in Figure 2.8 to model both *random variations* and *spatial variations* across the full-chip during a single circuit simulation, making it very useful for the Monte Carlo simulations carried out in a later chapter on the topic of ring tuning.

As seen in Chapter 5, both the worst-case skew technique (applied in Chapter 3 simulations) and the higher-level circuit parameter technique (applied in Chapter 5 simulations) result in comparable skew improvement results, confirming their effectiveness.

2.3 Skew Reduction Techniques

Understandably, there have been a wide variety of techniques developed to reduce skew over the years. Here, recently published techniques for skew reduction in clock circuits are summarized. Most attempts are applied in the circuit design phase (i.e., static), called *pre-silicon* techniques. Other attempts are invoked one time or multiple times during a circuit's operation (i.e., dynamic), called *post-silicon* techniques.

2.3.1 Pre-silicon Techniques

Pre-silicon techniques commonly involve a matching or balancing of the clock wires [37], the clock buffers, or both. Design adjustments to the clock distribution network are made based upon simulation and modeling of extracted data -- prior to chip tape-out.

2.3.1.1 Balancing Clock Distribution Wires

Many elaborate design techniques have been proposed and employed to reduce skew and/or jitter: grid tuning [38], wire snaking [8], delay insertion [23][39], mesh hybrids [24], and link insertion [25][26][40][41][42], to name a few. These techniques recognize that a perfectly balanced distribution network will minimize the variation between clock branches, and thereby reduce clock skew [43]. Unfortunately, the act of balancing out a distribution network typically involves the *addition* of metal, thereby increasing other undesirable effects. One such balancing technique is depicted in Figure 2.9. A typical H-tree (shown on the left) can be better balanced by adding square, metal links in a grid-like fashion (shown on the right).

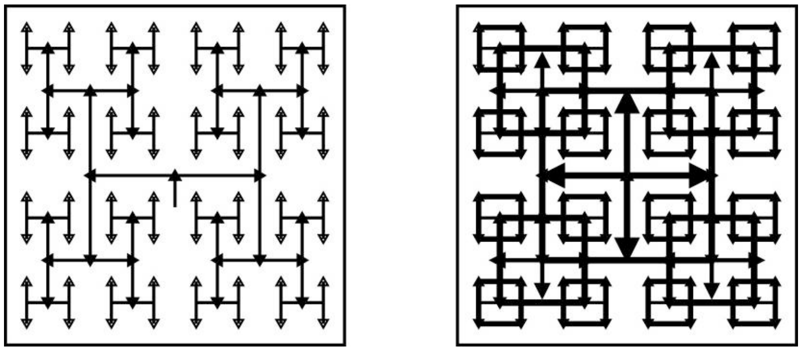


Figure 2.9 : Balanced H-tree for skew reduction (from [19])

This technique increases the upper layer metal wire area of the distribution network, which in all likelihood will increase its capacitive loading and its power dissipation, and may also necessitate the use of larger clock buffers, further increasing power dissipation and switching noise. It also creates more routing congestion for other global signals that require this metal layer. Therefore, this method of distribution network balancing through 'bulking-up' of the wires will come at a price, so it should only be used sparingly.

2.3.1.2 Balancing Clock Buffers

The clock buffers, or inverters, used to distribute the clock signal have been traditionally balanced in two ways.[1] First, they have been commonly designed as symmetric inverters in which the rising-edge and falling-edge transitions are designed to be equal-but-opposite, as explained previously. This clock signal symmetry has several advantages, such as 50% duty-cycle propagation, minimum combined switching time, and a single, edge-invariant inverter design.

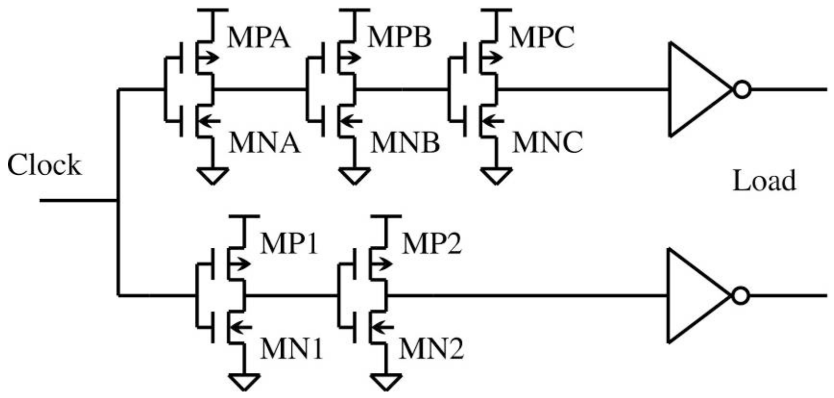


Figure 2.10 : Balanced Buffers for reduced skew (from [45])

The second method of balancing the clock buffers is described in [45] and is shown in Figure 2.10. The primary finding was to show that clock skew can be lessened by keeping the sum of the PMOS devices and the sum of the NMOS devices consistent in all clock branches – regardless of the number of inverters in each branch. By keeping the pull-up networks and pull-down networks equal strength across every clock branch, any PVT variations will more likely affect each branch similarly and the resulting clock skew will be minimized.

[1] Note that other more elaborate clock buffer techniques have been published to reduce skew, but their complex implementation makes them more of a new clocking architecture rather than just a balanced clock buffer technique [44].

One novel research topic in this dissertation proposes a method of using *asymmetric* inverters in place of symmetric ones for better timing performance with little to no power or area cost. *This asymmetric clock buffer design contravenes the first balanced buffer technique by using unequally weighted pull-up and pull-down individual inverters, but adheres to the second balanced buffer technique by maintaining equal pull-up/pull-down summations throughout every clock branch.*

These asymmetric buffers will be designed such that their W_P/W_N ratio will be different from the value used by traditional buffers (2.5), but they will keep (W_P+W_N) the same to ensure that their input and output capacitance effects are nearly identical to the traditional buffers that they are designed to replace. Subsequent design of a complete clock distribution tree involves a combination of the following three in-depth, inverter-design procedures:

- ♦ **Detailed load calculations** - The loading along the entire clock tree must be modeled accurately in order to properly design the drive sizes of the clock buffers. Inverter input and output loads, wire loads, and the terminal FF loads all must be considered.

- ♦ **Gate sizing for optimal path delay** - More realistic clock waveforms (not just step functions) and transistor models (not just effective channel resistance) must be used along with the detail load model from above when calculating signal and path delays.

- ♦ **Buffer insertion for very long wires** - The quadratic delay of long clock wires must be reduced to nearly linear delays with the proper insertion and sizing of additional buffers, or signal repeaters.

One last consideration regarding the clock distribution network design is, of course, the architecture chosen (tree, grid, hybrid, etc.). The balanced, fanout-of-four (FO4) H-tree architecture chosen for this research places several constraints on the placement and fanout of the clock buffers. Modified versions of the H-tree are implemented in practice since it is difficult to use the idealized structure directly on a design with irregular blocks. Clock grids tend to have much lower skews at the expense of area and power, and the hybrid methods attempt to balance the advantages and disadvantages of trees and grids. However, these networks introduce additional constraints depending on the chip being designed. These constraints tend to override some of the detailed calculations from above procedures, so they must be accounted for as well; hence, the idealized H-tree is the best vehicle for comparing the new techniques introduced in this thesis. Further details of the final clock distribution network design will be presented in the next chapter.

2.3.2 Post-silicon Techniques (Active De-skewing)

Pre-silicon techniques, such as those described in the previous section, work well to reduce clock skew by adjusting for design and/or layout related imbalances in the clock distribution network and its loads. However, on-chip PVT variations are nearly impossible to predict and/or control during the design/layout phase, they cannot be addressed until after chip fabrication. Therefore, some form of post-silicon clock adjustment of skew will be necessary to compensate for the majority of intra-die PVT variations. This post-silicon adjustment necessitates the creation of an adjustable or tunable component within the clock distribution network – namely, a tunable clock buffer. Furthermore, methods of measuring the skews at key points in the circuit must be developed along with reliable methods of feeding back the skew data to the tunable buffers for skew adjustment.

Research has been performed on tunable delay buffers for various applications [46], targeting the symmetric waveforms of traditional clocking. One-time programmable tunable clock buffers have been used to reduce skews by adjusting clock delays post-silicon based upon wafer probe or assembly final test data gathered from Automatic Test Equipment (ATE) or from a start-up characterization step [47][48]. However, the challenge of precisely measuring numerous internal clock signals during ATE test makes this technique difficult and costly. Furthermore, PVT variations may change over time during normal circuit operation, so an initial, single clock adjustment may not suffice over the device's lifetime usage.

A better approach for post-silicon skew compensation is to configure active (always-on) tuning loops that self-monitor for clock skews and appropriately adjust their tunable buffers. Such systems are commonly referred to as adaptive circuits or active deskewing circuits. As [28] clearly states, new scaling issues will cause more variability that affects circuit designs, and therefore designers will likely use more of these adaptive circuits.

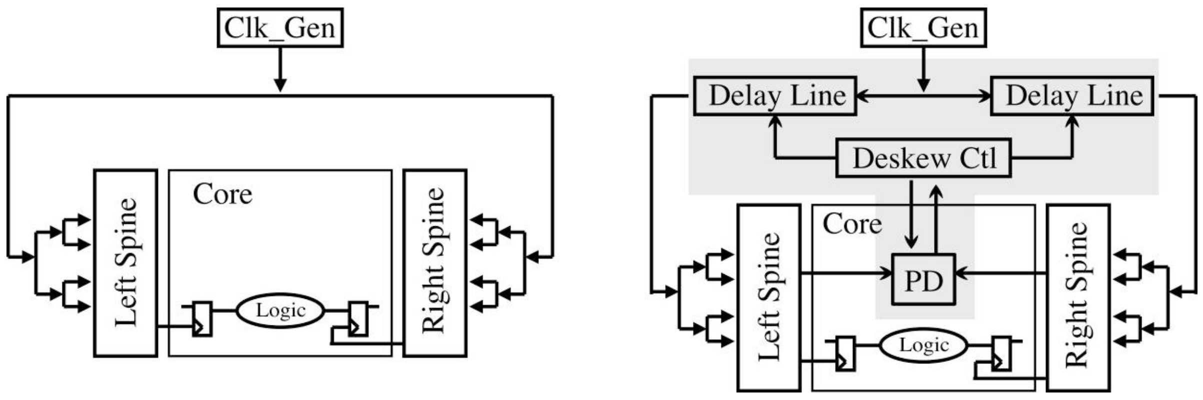


Figure 2.11 : A basic post-silicon skew compensation system (from [49])

Some previous implementations of active deskewing have been adequate at measuring and adjusting two clock regions (or tuning zones) with respect to each other across the entire chip [49][50], as in Figure 2.11. The clock distribution shown on the left side is made tunable by the addition of a phase detector, deskew control, and tunable delay line shown on the right side. However, the anticipated increasing levels of intra-die variations will require a finer resolution of adjustability consisting of many more than just two tunable clock zones, so the sharing of multiple clock zone phase information is critical.

2.3.2.1 Multiple Tuning Zone Synchronization

The traditional DLL architecture, which is commonly used to synchronize clock phases in an active deskewing system, consists of two clock signals: the *tunable clock* is adjusted via tunable buffers to align with a *reference clock*. As mentioned above, the synchronization of clock networks consisting of only two tuning zones is straightforward using a single DLL -- one zone is chosen as the reference, and the other zone is tuned to align with it. However, for clock networks of more than two tuning zones, referred to here as multi-zone networks, a more elaborate configuration of clock phase sharing must be employed. The simplest of the multi-zone phase sharing configurations is the use of a global reference clock. Having a single ubiquitous reference clock would allow for each tuning zone to align its tunable clock to a single, chip-wide clock phase. The multi-zone synchronization problem can therefore be reduced to multiple instances of the simple two-zone configuration -- one DLL for each zone aligns its tunable clock to the global reference clock. Unfortunately, generating that global reference clock with constant phase for use across the entire chip can be costly.

One such implementation of multi-zone synchronization via a global clock reference is given in [51]. A reference clock is routed in parallel with the main clock to be used for local synchronization, but it requires the addition of a second, identical, balanced clock H-tree exclusively for the reference clock distribution. Since the reference clock has no loads, this removes many constraints from the design of its clock tree allowing lower skew to be achieved. In addition, the drivers in the reference network can be smaller than those in the main network. Clock distribution networks are very costly in terms of die area, metal routing, and power consumption; therefore, adding a second clock distribution network is a costly addition that should be avoided if possible.

Another similar implementation is the use of return clocks for phase alignment as in [52][53][54]. Note that a *reference* clock is routed from the clock root to the clock leaves for phase comparison, whereas a *return* clock is routed from the clock leaves back to the clock root for phase comparison, but they both provide the two clocks necessary for multiple implementations of the simple two-zone deskewing system. Again, however, the additional distribution of a return clock can be quite challenging. In [54], the return clock is to be delivered via the clock shielding wires. The shielding wires are included in most clock distribution networks to isolate the clock signal from the rest of the chip; they are typically connected to ground to help attenuate any switching noise. Redesigning the shielding wires to double as clock returns presents additional complexity in the design.

A slight variant to the global clock reference method is the use of a global timing reference or global control voltage in the case of [55]. That system essentially converts the clock phase difference between the source clock and a return reference clock into a voltage level for use at multiple PLLs. However, distributing a precise, analog control voltage across an entire chip is also challenging, although solutions could be developed to address this issue. Such a voltage reference distribution would still be subject to the unavoidable intra-die PVT variations and various noise sources that would impact each tuning zone randomly, creating skew between zones for which the single, global control voltage could not be adjusted.

Based on the above issues, the work described in this dissertation will focus on deskewing networks consisting of more than two tunable clock zones without requiring chip-wide reference clocks or control signals. A model for multiple PLL clock distribution systems was created in [56] predicting that single PLL systems will be adequate for the 130nm process node, but that four PLLs will be superior for the 100nm node. Continuing with that trend further, multiple PLL (or DLL) clock distribution networks should be considered for future synchronous VLSI designs. Two such multi-zone clock-synchronizing configurations are examined below. The first is a series multi-module architecture, also referred to as a cyclic configuration, and the second is an acyclic configuration. Later, the research described in this dissertation will propose a new method of interlinking multiple instances of DLLs into a stable ring configuration that addresses problems of the previous two methods. This novel technique of *Ring Tuning (RT)* will be presented in Chapter 5 for the post-silicon skew compensation of a typical H-tree clock distribution network.

2.3.2.2 Cyclic Architecture

An early multiple PLL system is described in [57]. The schematic of their four PLL series architecture is shown in Figure 2.12. The master clock at the center is distributed to four delay elements that supply their tuned clocks to the four regions labeled A through D. The tuned clock from each region is then fed to two skew sensors (or phase detectors) for comparison with neighboring clocks. Two major issues exist with this proposed implementation: PVT variations on the signals and overall system stability.

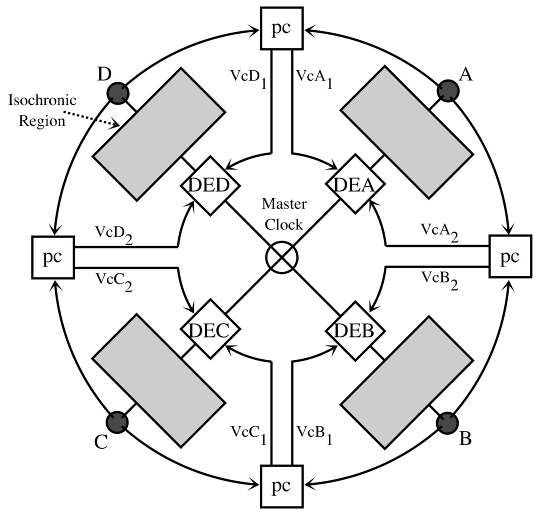


Figure 2.12 : Previous multiple PLL ring architecture (from [57])

Firstly, the specifics of distributing the primary clock signals, the analog control voltages, and the feedback clock signals across the entire chip are not addressed. With the high levels of intra-die variation of present and future process nodes, the chip-wide distribution problem must be specified and included in any clock skew results reported. Secondly, the issue of system stability is not adequately covered. As seen in the diagram of

Figure 2.12, each phase comparator (labeled 'pc') produces two control voltages (one for each of two neighboring delay elements), and consequently each delay element receives two different analog control voltages. How these separate voltages are handled so as not to result in overall system instability is not sufficiently discussed. In fact, three distinct types of stability must be considered for any cyclic architecture. For the purpose of this research, these three varieties of stability will be referred to as *Mode-lock*, *Mode-lock Oscillation*, and *Cyclic-Feedback Oscillation*.

Mode-lock: Traditionally, the term *mode-lock* refers to an undesired condition of multi-zone tuning, where one or more local phases may lock out of synchronization with the global phase. For the system depicted in Figure 2.12, it would be possible for zone D to lock 90˚ out of phase with zone A, zone C to lock 180˚ out of phase with zone A, and zone B to lock 270˚ out of phase with zone A, since traditional analog phase detectors have a 90˚ phase limit. This *mode-lock* condition is a perfectly valid and stable locking state of the cyclically connected multiple tuning zones; however, its phases are incorrectly locked, so the system will remain unsynchronized. *Mode-lock* is avoidable with the use of more complex phase detectors [58] [59]. However, in order to avoid these complex phase detectors, [60] recommends propagating the phase information in a single direction (acyclic instead of cyclic), which will be covered in the following section.

Mode-lock Oscillation: Note that in this dissertation, a new term, *mode-lock oscillation*, will be used to describe a multi-zone system instability that is closely related to the original problem of *mode-lock*. In *mode-lock oscillation,* the multiple tuning zones oscillate back-and-forth between two or more valid locking states, resulting in an unstable locking of the entire system. As shown in Figure 2.12, the tuned clock of one zone becomes a reference clock of its neighboring zone. This exchange of clocks (zone A tuned becomes zone B and zone D reference, zone B tuned becomes zone A and zone C reference, and so on) continues around the ring in a closed-loop or cyclic fashion. As a result, the adjustment of the tuned clock of one zone could (if the other zones are appropriately positioned) occur simultaneously as an opposite adjustment of its own reference clock – thereby negating the intended clock tuning step and causing the same zone to attempt a reverse tuning correction. Such a cyclic tuning structure could become 'stuck' in an infinite loop of continual oscillation between two valid adjacent tuning states. Oftentimes, this more difficult problem of *mode-lock oscillation* is simply (but incorrectly) referred to as *mode-lock*; nevertheless, both *mode-lock* and *mode-lock oscillation* must be avoided for stable synchronous clocking.

Cyclic-feedback Oscillation: The final stability concern for cyclically-connected multiple tuning zones arises from its complete cycle (or feedback) of phase sharing. Again, from Figure 2.12, any phase adjustment at zone A will be passed through its adjacent 'pc' block to zone D, and then to zone C, and then zone B, and finally back to zone A again. If the four zone phases were appropriately positioned, a 'traveling wave' of clock phases could essentially be passed around the cycle indefinitely. To stop this 'traveling wave' from maintaining oscillation, or instability, the cyclic loop should be designed to dampen the wave as it circles.

2.3.2.3 Acyclic Architecture

In order to break the cyclic configuration and eliminate its inherent problems of *mode-lock*, *mode-lock oscillation*, and *cyclic-feedback oscillation*, an acyclic configuration has been proposed. Two algorithms of acyclic phase averaging are presented in [61] and their favored technique is utilized in the Intel Itanium microprocessor [73]. The multiple

PLL clock architecture of [61] is shown in Figure 2.13. On the left is the generic H-tree clock distribution network populated with 16 PLLs synchronized via 24 phase detectors. On the right is a diagram of the acyclic phase alignment, starting with the reference PLL in the upper left corner of the chip and propagating across the entire chip to the lower right corner.

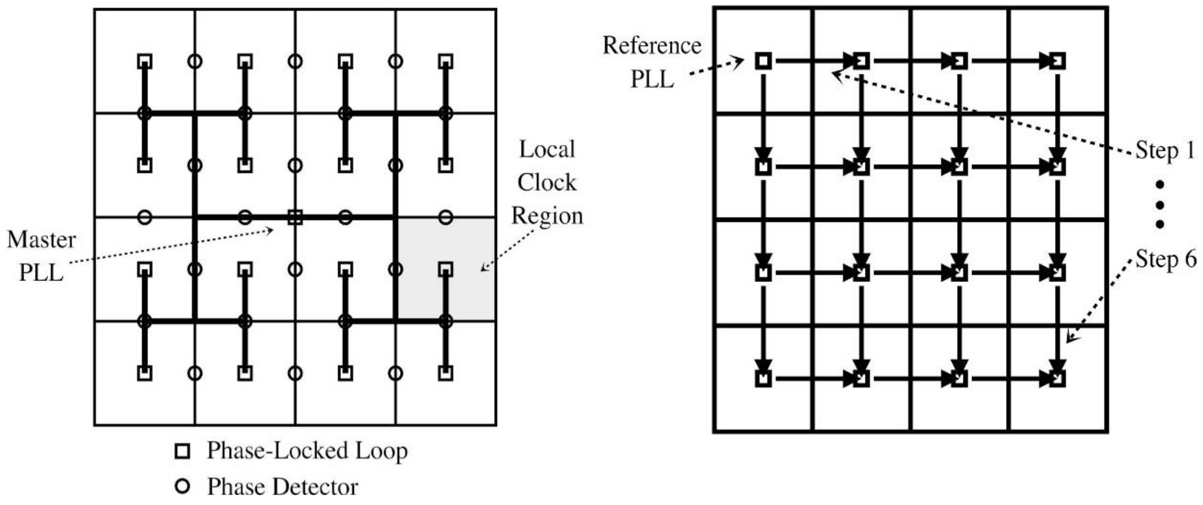


Figure 2.13 : Previous acyclic multiple PLL connections (from [61])

Such an acyclic phase distribution algorithm does avoid the three system instabilities discussed in the preceding section by eliminating the cyclic, multi-directional feedback loop of [57] and [58], but it brings up a new problem referred to as *Out of Range*.

Out of Range: As the phase information passes from one corner of the chip to the other, it is quite possible that the phase adjustments will accumulate, ultimately reaching the tuning limit of some of the tunable buffers. Using tunable buffers of larger ranges to avoid the limits is not necessarily desirable or even possible. Therefore, some method of global control must be established to adjust the entire clock system away from the limited tunable buffer(s). In fact, the algorithm proposed in [61] requires three controllers (global controller, local controller, and reference alignment controller) for proper operation of the active deskewing system. Designing these controllers and placing them throughout the chip are non-trivial tasks. If the deskewing clock regions could be configured such that they control themselves while avoiding the *out of range* problem, then all these system-level controllers would no longer be necessary, and the design and implementation of such a system would be simpler. This approach is pursued in the research, as described in detail in Chapter 5.

2.4 Background Summary

As described in this chapter, there are two opportunities to reduce clock skew:

Pre-silicon: by modifying the circuit design and/or layout, the clock distribution network can be better balanced and therefore more immune to PVT variation causing clock skew. The research to be described in the next chapter proposes a novel clock buffer design to be used for pre-silicon skew reduction, called *Single-edge Clocking (SEC)*. By moving from symmetric inverters to asymmetric inverters for clock buffers the critical clock edge can be delivered quicker and sharper, thereby reducing clock skew, while the non-critical clock edge is relaxed[2].

Post-silicon : by measuring the clock signals after chip fabrication, tunable elements can be adjusted to compensate for PVT variation causing clock skew. The research in later chapters proposes a novel, dynamic clock tuning scheme for active clock deskewing, called *Ring Tuning (RT)*. Simple adjustable clock buffers can be interlinked into a ring structure

[2] Note that previous work has recommended relaxed clock edges for switching noise reductions [62][63], but not for clock timing performance improvements, as intended here.

creating autonomous, stable, DLL-like clock zone tuning to compensate for real-time measured clock skew.

Chapter 3 : Single-Edge Clocking (SEC)[3]

The timing performance of a typical synchronous circuit is primarily dependent upon the simultaneous activation of the timing flip-flops (FFs) driven by the clock leaf buffers, as shown in the previous chapter. Furthermore, the activation of a typical edge-triggered FF is essentially governed by the single, critical edge (oftentimes the rising edge) of the clock signal. Hence, by delivering that critical, rising edge of the clock to all the leaf FFs faster (i.e., less latency) and sharper (i.e., quicker rise-time) the timing performance of the entire synchronous circuit can be improved.

The research described in this chapter will demonstrate a new technique to use asymmetrical clock buffers in an alternating pattern to propagate the critical edge of the clock signal both faster and sharper, resulting in clock latency, skew, and jitter performance improvements of over 20% versus traditional symmetric clocking. Furthermore, the clock buffer area and power consumption will remain virtually unchanged. Using these asymmetric clock buffers will result in pulse width accumulation, limiting their maximum frequency; however, by using a clock pulse at the clock root, this pulse width growth can be tolerated for clock frequencies typical in most ASIC circuits.

The clock distribution network for a hypothetical 10mm x 10mm square chip is used as the demonstration vehicle in this chapter using a full array of asymmetric SEC buffers, and its timing performance is compared with that of a traditional, symmetric clock distribution tree. First, however, the general timing tradeoffs of asymmetric, SEC design are described. To do so, a new timing metric called degradation factor will be defined in order

[3] The research covered in this chapter has been published in: J. Mueller and R. Saleh, "Single Edge Clock (SEC) distribution for improved latency, skew, and jitter performance," *Proc. of VLSID*, pp. 214-9, Jan. 2008.

to quantify the degree of asymmetry of any given clock buffer. Then, the details of the clock tree architecture and buffers will be described, and the simulation results will be compared to the traditional approach.

3.1 Improved Clock Period Utilization with SEC

Note that in the following discussion of this section, the traditional method of measuring a periodic signal's pulse width will be modified slightly to capture the degrading effect of the neglected clock edge, *tnegl*. As illustrated in the top diagram of Figure 3.1 the common definition of pulse width, *PW*, is measured at the 50% of Vdd crossing points. However, using this method to analyze the clock period would not directly include the degrading effects of the neglected clock edge. A modified method of defining the PW is shown in the bottom diagram of Figure 3.1. This modified method clearly demonstrates the clock period intervals consumed by the neglected and critical clock edges[4]. Furthermore, as the lower diagram indicates, the intended role of the 'safe' portion (i.e., to provide a constant voltage settling interval to insure that the neglected clock edge does not interfere with the next critical clock edge) is more explicitly displayed using this modified pulse width definition.

[4] Conveniently, this modified method of defining the pulse width matches with the HSPICE syntax for defining a voltage pulse source, thereby allowing for the simulation clock signal to be constructed directly from this clock period analysis.

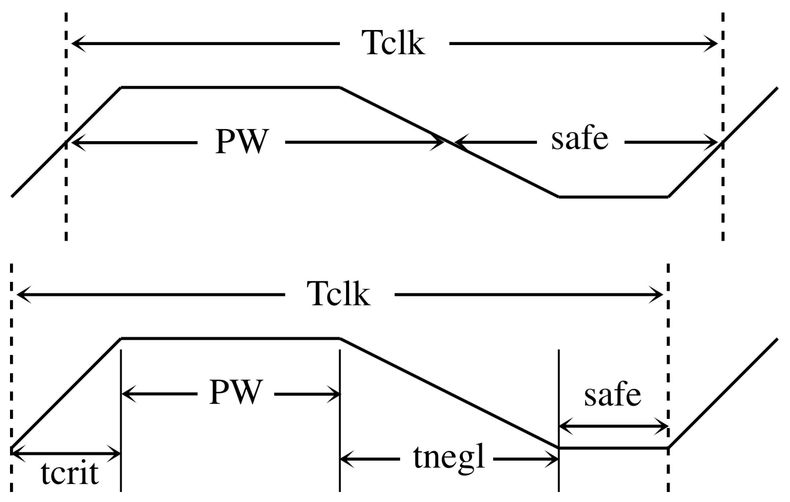


Figure 3.1 : Traditional and Modified Definition of Pulse Width

3.1.1 Performance Trade-off between Clock Edges

The motivating concept behind SEC clocking is best explained by examining the four segments of the clock period. The leading edge (rise or fall) is followed by the leading level (high or low), followed by the trailing edge (fall or rise), and the trailing level (low or high). In SEC designs, only one of these four segments is critical for timing performance -- the leading edge should have low skew and low jitter -- while the other three segments merely exist to 'setup' for the leading edge of the next clock period. Hence, more focus should be placed on the leading, critical edge, even if it is at the expense (or neglect) of the other three regions -- assuming, of course, that such degradation of the other regions does not affect the proper operation of the circuit. Furthermore, since the arrival of the critical clock edge at the final leaf FFs is the sole purpose of this clocking system, it can accurately be called single-edge clocking[5]. As such, the pulse width, or duty cycle, of the clock signal

[5] Note that double-edge triggered flip-flops cannot be supported in this approach. However, the dominant clocking scheme in most ASICs today employs single-edge triggered flip-flops.

is not important; therefore, this SEC technique will use a short pulse as the initial clock signal which is a notable difference compared to the traditional methods.

The clock period diagram of Figure 3.2 illustrates the SEC method by following a clock signal as it propagates through the clock network levels, starting with the initial pulse PW_0 at level l=0 and ending with PW_L at level l=L where the FFs are located. A sharp leading edge is propagated quickly and cleanly by each SEC buffer; however, the trailing edge is sluggish and delayed at each level, which forces the leading pulse width to grow and the trailing pulse width to shrink. This degradation is the effect of neglecting the trailing edge throughout a typical SEC distribution. Eventually, the trailing edge would exceed the clock period window, if this trend continued, and the clock signal would be lost.

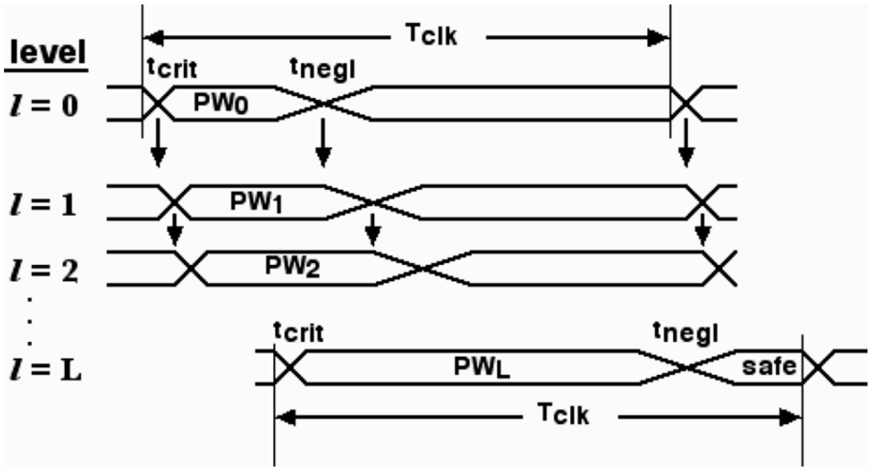


Figure 3.2 : Clock Period Utilization

Simple equations can be derived to bound the amount of trailing edge degradation allowable for a given clock period and distribution network. First, it is assumed that t_{crit} and t_{negl} (the rise/fall times of the critical and neglected clock edges, respectively) of Figure 3.2 are constant for all SEC levels. This assumption follows from the selection of asymmetric and complementary buffers, referred to here as INVr (strong rising edge, weak falling edge) and INVf (strong falling edge, weak rising edge), with equal degradation factors, to be defined shortly. The amount of pulse width (*PW*) growth through *l* SEC buffers (or levels) can be expressed as:

$$PW_l = PW_0 + \frac{1}{2}(t_{negl} - t_{crit})l \tag{3-1}$$

where $l=0,1,...L$.

For SEC evaluation, a new metric called the degradation factor is defined as, $\eta=(t_{negl}/t_{crit})$, where t_{negl} and t_{crit} are the transition times of the neglected clock edge and critical clock edge, respectively. Here, if $t_{negl}=t_{crit}$, then $\eta=1$, which is traditional (symmetric) clocking, so there is no pulse width growth and the duty cycle remains constant throughout. Effectively, η is a measure of how much of a departure the asymmetric buffers are from the traditional buffers. This directionally-neutral definition of degradation factor allows for its usage with INVr (strong PMOS) and INVf (strong NMOS) buffers alike.

The intent of this metric is to capture the degree of trade-off being made when focusing on the critical clock edge, which inevitably impairs the neglected clock edge. As mentioned previously, the critical edge of an INVr buffer is the rising edge of its output (the neglected edge is the falling edge); conversely, the critical edge of an INVf buffer is the falling edge of its output (the neglected edge is the rising edge). Furthermore, pairs of complementary INVr-INVf SEC buffers are chosen to have equal amounts of trade-off between their critical edge improvement and neglected edge impairment (i.e., complementary SEC buffer pairs have equal degradation factors as mentioned above).

It is important to realize that the initial clock pulse width, PW_0, must be large enough to propagate through the first buffer, but should be small enough to allow for as much clock period degradation as possible. Let T_{clk} be the clock period and let L represent the number of SEC levels. For proper operation, the initial (root clock) pulse width, PW_0, is bounded by[6]:

$$t_{crit} \leq PW_0 \leq 0.8T_{clk} - t_{crit}(\eta + 1) - \frac{1}{2} t_{crit}(\eta - 1)L \qquad (3\text{-}2)$$

A safeguard of 20% of the clock period (represented by the *0.8* multiplier above) has been added to account for generalized assumptions and simulation inaccuracies. Circuit design particulars may dictate using a higher guard-band for more conservative timing or a lower guard-band for more aggressive SEC performance improvements. Choosing a higher degradation factor will result in better critical edge performance, but more of the clock period will be degraded, so a trade-off must be made. The number of SEC levels, L, is limited by the clock distribution network; hence, the design challenge is to minimize PW_0 in order to maximize η.

The clock frequency limitation imposed by the neglected edge of the SEC technique, as expressed in Equation (3-2), can be summarized as follows. The minimum period (or maximum frequency) for a non-SEC clock signal, of any process technology, can be roughly defined as *twice the minimum pulse width* (assuming the conventional 50% duty cycle). Instead, the minimum period of an SEC system would be *twice that same minimum pulse width* ***plus*** *the amount of pulse-width growth through all clock levels due to the neglected edge, from Equation (3-1)* ***plus*** *a designer chosen guardband to insure that the neglected edge does not interfere with the critical edge*. The former term represents the desired trade-

[6] Note that the $t_{crit}(\eta+1)$ term, which equates to $(t_{crit}+t_{negl})$. in this equation is a result of the modified pulse width definition explained at the beginning of this section.

off from the neglected edge to the critical edge -- a more aggressive SEC system (higher η) will have improved timing performance but more pulse-width growth. The latter term represents an additional sacrifice that must be made due to the relaxed quality of the neglected edge. However, as later simulations will show, the two additional terms are small enough to permit the usage of SEC for most ASIC clock frequencies, which are typically well below the process technology's maximum frequency.

3.1.2 Practical Application of SEC

To illustrate these concepts further, full-chip H-tree simulations were performed. The H-tree used is the one shown in Figure 3.3, with a standard, balanced fanout-4, 4-level architecture. It is designed to deliver a supplied clock signal (as represented by the arrow pointing to #1) to 64 leaves covering the entire 10x10mm^2 chip. Buffers are positioned at the center of each H structure (#1, 2, and 3), along with extra buffers, or repeaters, to handle the long wire loads (#1b and 2b), and finally at the leaf ends to drive the FF loads (#4).

The distribution wires from the clock source to the 2nd level buffers are designed to be 2X wide (as indicated by the thick lines), and therefore are modeled as half the resistance per length of the remaining branches. Approximate 90nm process wire parameters[7] of $R_{wire}=300\Omega/mm$ and $C_{wire}=200fF/mm$ have been used in a π-segment wire model, and the FF loads at each leaf are modeled as 200fF capacitors.

[7] Note that this resistivity value of $R_{wire}=300\Omega/mm$ corresponds to lower level metal traces, which exhibit worst-case performance in these simulations. In later sections the results will be given with intermediate and upper level metals of $R_{wire}=30\Omega/mm$ and $R_{wire}=3\Omega/mm$, respectively.

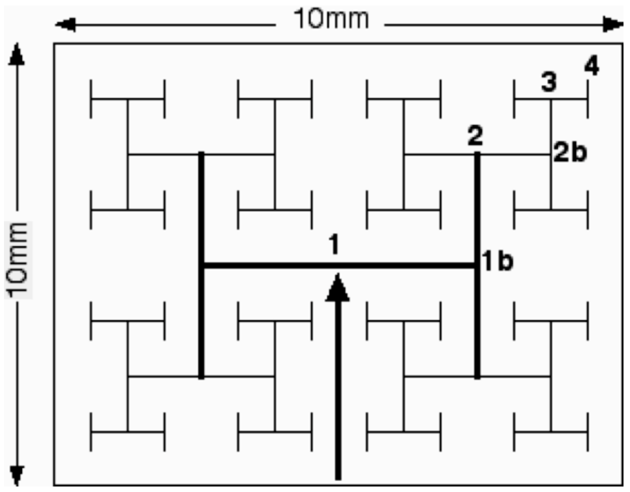


Figure 3.3 : 4-level H-tree

The experiments on the H-tree for different values of η used a T_{clk} of 4ns, a t_{crit} of 400ps (10% of T_{clk}), and 5 levels of SEC buffers in a typical 90nm process.[8] Plugging these values into Eqn. (3-2) results in the data shown in Table 3-1.

Table 3-1 : SEC buffer degradation factor choices

η	PW_0 min (ps)	PW_0 max (ps)
1	400	2400
1.5	400	1700
2	400	1000
2.3	400	580
2.5	**400**	**300**

[8] Note that the timing performance improvements of SEC over TC will hold for any clock period that provides adequate space for pulse width growth as captured in Equation (3-2). HSPICE simulations have shown this 5-level SEC H-tree circuit to operate properly for clock frequencies in excess of 1GHz (1ns clock period), but the period of 4ns has been chosen here for consistency with an ASIC design.

The first row of this table shows a minimum pulse width of 400ps leads to a maximum pulse width of 2400ps using Equation (3-2), which assumes a 20% guardband[9]. However, the most important results appear in the two last rows. The table indicates that a degradation factor of 2.5 is not feasible since $PW_0max < PW_0min$, and therefore a factor of 2.3 would be considered a practical limit. A more suitable factor of 1.5 with an initial clock pulse of 500ps will be used in further simulations. Based on this result, the size selection for the two complementary SEC buffers from the β modeling can be carried out as discussed in the next section. These two SEC buffers are then designed into a full-chip clock distribution network for the analyses described in the subsequent section.

3.2 SEC Buffer Design Constraints

With the given background of the design principles of traditional, symmetric clock buffers from Chapter 2, the "non-traditional" aspects behind SEC clock buffer design are now addressed. Most notable in this context is the need for two distinct SEC inverters in place of one single traditional inverter. Whereas the traditional buffer is symmetrical, so only one buffer design (called INVt) is necessary to propagate both the rising-edge and the falling-edge of the clock signal, the SEC buffers are asymmetrical, so two buffers are needed: one buffer (called INVr) to propagate the rising-edge and one buffer (called INVf) to propagate the falling-edge. The designs of these two complementary, asymmetric SEC buffers are very much a mirror-image process.

[9] A traditional clock distribution would not necessarily budget a 20% guardband for pulse width growth, so its maximum pulse width could be taken as 3600ps.

3.2.1 SEC Capacitance Constraint with Drop-in Compatibility

The SEC inverters are intended to integrate seamlessly into existing circuits by replacing the traditional, symmetrical inverters throughout the clock distribution network. Consequently, additional design constraints are placed on these special SEC buffers in order to make them compatible with traditional clock buffers. The most important of these constraints derives from the inverter capacitance modeling of the preceding chapter. By simply matching the SEC buffers' input and output capacitances and layout areas with those of traditional buffers, it will be possible to use these asymmetric SEC buffers as "drop-in" replacements for existing symmetric buffers. This interchangeability effectively simplifies their use by designers and minimizes their impact on automatic computer-aided design (CAD) tools.

From Equations (2-2) and (2-3) it has been shown that both the input and output capacitance of a CMOS inverter is proportional to the sum of the widths of its NMOS and PMOS devices. Therefore, in order to keep the input and output capacitances of these asymmetric inverters nearly equal to those of the traditional, symmetric inverter, *the sum of the transistor widths (W_N + W_P) should always be kept uniform*. The details behind the design of a typical SEC buffer pair are covered next.

3.2.2 SEC Asymmetrical Inverter Pairs

The degree of asymmetric skewing of an inverter is measured by that inverter's β, which is defined as the ratio of the width of the PMOS transistor to the width of the NMOS transistor, $\beta = W_P/W_N$. As explained in Section 2.3.1.2, the β of a traditional, symmetric inverter is β=2.5 (referred to as INVt). If the PMOS were made larger (i.e., wider channel) with respect to the NMOS, then the β would be skewed to a value larger than 2.5, and the

INVr buffer would be said to favor the rising edge of the output clock signal. Conversely, if the PMOS were made smaller with respect to the NMOS, the β would be less than 2.5, and the INVf buffer would favor the falling edge output. Now that the capacitive loading constraint of uniform *(W_N + W_P)* has been imposed on the new SEC buffer designs, equations governing the asymmetrical skewing of the clock buffers can be generated.

As previously mentioned, the asymmetry of these buffers will impact both speed (i.e., delay or latency) and edge rates (i.e., rise and fall transition times) of the clock signal. Therefore, either metric, delay or transition time, would be suitable for deriving these equations; here, transition time is chosen since it can be measured directly from the output signal alone (at the 10% to 90% levels) whereas delay would have to be measured with respect to some other reference point in time.

Figure 3.4 illustrates the impact that asymmetrically skewed buffers would have on a clock signal's transition times. The top waveform shows how a square-wave clock signal is passed by a symmetric, traditional inverter (INVt) without much distortion on either edge – just inversion. In the middle waveform, however, the INVr would produce a sharper rising edge at the output, but a much slower falling edge. Conversely, in the bottom waveform, the INVf would produce a sharper falling edge but a slower rising edge. From these waveforms, it is evident that one of *two possible asymmetric inverters would be necessary to replace a traditional, symmetric inverter* -- depending upon whether the rising or falling edge of the output signal was the critical edge. A model for selecting the precise β for each inverter is covered next.

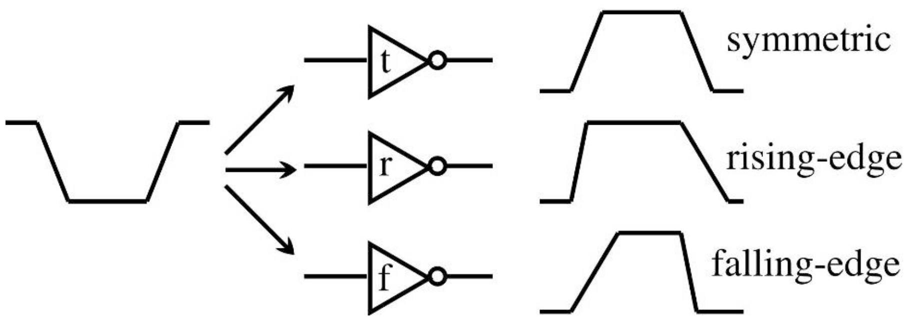


Figure 3.4 : Faster and sharper edges for SEC inverters (INVt versus INVr & INVf)

3.2.3 SEC Inverters' Beta Modeling

Circuit simulations using generic 90nm process models in HSPICE can be used to calculate the transitions times for a wide range of possible inverter β's. The results from such HSPICE simulations of the rise-time, *Trise*, and fall time, *Tfall*, (as measured at the 10% and 90% signal levels) are shown as the large data points in Figure 3.5 , respectively.[10]

Inverter β's ranging from 0.075 (strongly INVf) to 35 (strongly INVr) are plotted on the logarithmic x-axis. As mentioned above, the INVf inverters, on the left side of the plot, have faster *Tfall* values, but much slower *Trise* values. Conversely, the INVr inverters, on the right side, have faster *Trise* and slower *Tfall*. The symmetrical INVt (β=2.5) is indicated at the center of the plot where the *Trise* and *Tfall* curves intersect (i.e. *Trise=Tfall* as expected).

[10] Note that the rise and fall times (consequently, t_{crit} and t_{negl}) depend upon the input waveform and the inverters' load; therefore, these β simulations are carried out for the fully loaded clock distribution network with the assumed root clock input.

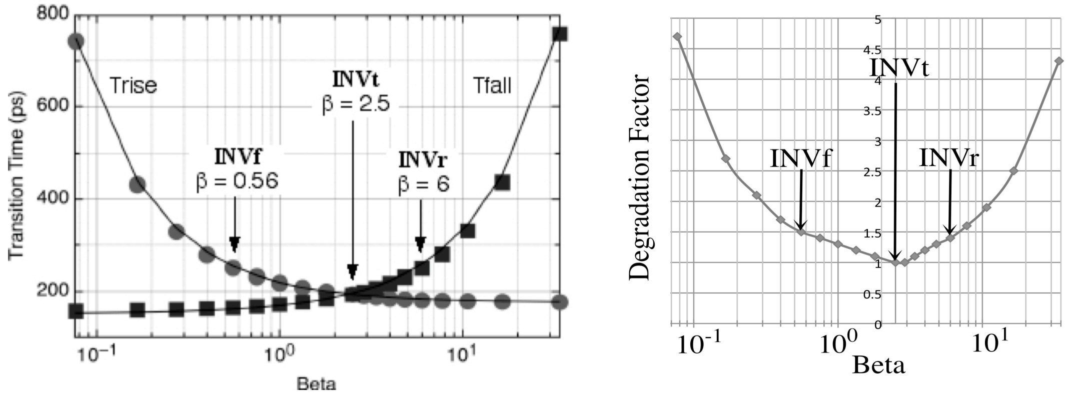


Figure 3.5 : *Trise* and *Tfall* (left) and η (right) versus buffer β

The graph on the right of Figure 3.5 plots the degradation factor ($\eta = t_{negl}/t_{crit}$) versus the clock buffer β. The y-axis is positioned at the INVt (β=2.5) inverter, so the INVf inverters are on the left of the axis, and the INVr inverters on the right. (These η values are simply calculated by dividing the appropriate T_{rise} and T_{fall} data from the graph on the left.)

Basic circuit analysis techniques can be used to derive equations for the two *Trise* and *Tfall* curves to validate the results. Using the general form of the Elmore delay [64]:

$$\tau_i = \sum_k C_k R_{ik}$$

Since *(W_N + W_P)* is held uniform the two equations for the transition times can be expressed in terms of the inverter's β as follows:

$$Trise = 2.15\ (\ 20.4\ (1/\beta) + 81.9\ )\quad (ps) \tag{3-3}$$

$$Tfall = 2.15\ (\ 8.5\ \beta + 70.0\ )\quad (ps) \tag{3-4}$$

Curves for the above two equations are superimposed as lines on the graph of Figure 3.5. As seen, they match the HSPICE simulation data extremely well[11]. Since these equations model the transition times of the inverters so accurately, they can be used to calculate the timing impact of skewing the inverter's β when designing SEC buffers[12].

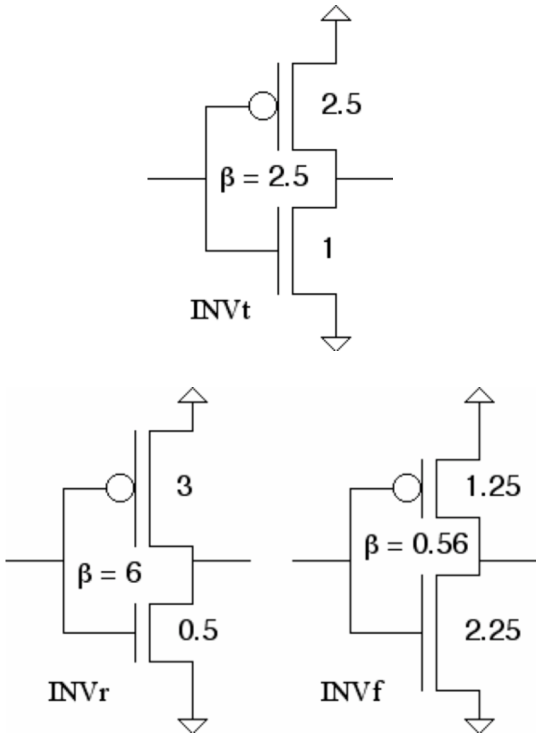


Figure 3.6 : Relative Transistor sizing for TC & SEC clock buffers

Using the curves of Figure 3.5, or the above equations for *Trise* and *Tfall*, two representative SEC inverters (INVf and INVr) have been designed to compare against a traditional inverter (INVt). As indicated on the graph, the INVf is chosen at a β=0.56 and the INVr at β=6.0. Schematics to illustrate the relative transistor size for these three

[11] Note that a factor of 2.15 has been included in these equations to fit the simulation data. This constant accounts for the difference between the Elmore definition of delay and the simulation transition times, as well as the use of the slow corner model in HSPICE as opposed to the nominal values for the hand calculation.

[12] Note that these equations and simulations are only valid for this generic 90nm process -- new equations would be computed, using these same techniques, for other process technologies.

inverters are shown in Figure 3.6. At the top is the traditional, symmetric inverter INVt with *$\beta=2.5$*; On the bottom left is the asymmetric, SEC inverter INVr in which the PMOS relative size is increased for stronger pull-up, with *$\beta=(3/0.5)=6.0$*; and, on the bottom right is the asymmetric, SEC inverter INVf in which the NMOS relative size is increased for stronger pull-down, with *$\beta=(1.25/2.25)=0.56$*. The relative width for each transistor is also given in the schematics of Figure 3.6. The sum of two transistor widths, *$(W_N + W_P)$*, is seen to be held uniform for all three inverters *$(1+2.5 = 0.5+3 = 2.25+1.25 = 3.5)$*, as explained previously for similar input and output capacitances.

3.2.4 Layout Area Considerations

One important topic related to the transistor sizing of the SEC inverters concerns the layout of the clock buffers. As previously mentioned, a primary concern in the design of these asymmetric SEC buffers is to allow them to be "drop-in" replaceable with the traditional buffers. With regard to the electrical characteristics of the input and output capacitive loading of the inverters, maintaining a uniform "sum of transistor widths" for the inverters would allow for their 'electrical' drop-in substitution anywhere within the clock tree. Furthermore, this same constraint, uniform *$(W_N + W_P)$*, can also be shown to allow for their 'physical' drop-in substitution directly in the actual layout of the chip.

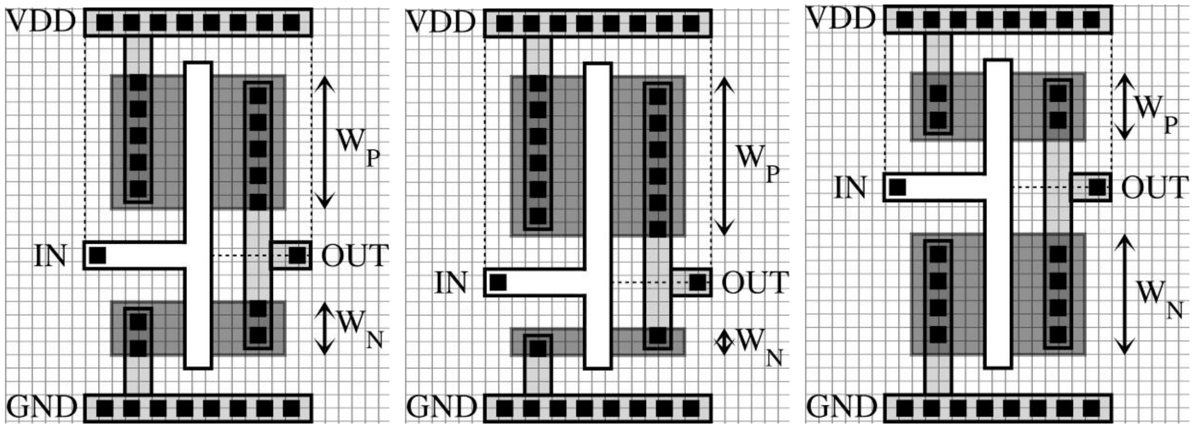


Figure 3.7 : Layout of three inverters (INVt, INVr, INVf)

Figure 3.7 shows a conventional layout for the three chosen inverters (INVt, INVr, and INVf). By orienting the widths of the PMOS and NMOS transistors of the inverters along the same axis, it can be seen that any size adjustments due to changes in the $\beta=W_p/W_n$ ratio will not impact the overall area or even footprint of the inverter -- provided of course that $(W_N + W_P)$ remains uniform. In particular, a clock tree could be designed in the traditional fashion with traditional clock buffers [65][66][67], all the way up until the final tape-out of the entire circuit, at which point the traditional buffers could be replaced with SEC buffers for improved clock timing performance. Furthermore, timing errors that may develop very near the end of the chip design and layout could be corrected by a simple substitution of a traditional buffer with an appropriately faster SEC buffer.

3.3 Analysis of Asymmetric SEC Buffers

3.3.1 Voltage Transfer Characteristics

The voltage transfer curve (VTC) is a very useful tool in analyzing the input-to-output operation of an inverter. The three inverters (INVt, INVr, and INVf have been simulated with a DC sweep of the input voltage in HSPICE to produce their VTCs (upper portion) and current plots (lower portion) as shown in Figure 3.8. INVt is at the top, INVr in the middle, and INVf at the bottom. Two simple observations can be made from these three VTC plots.

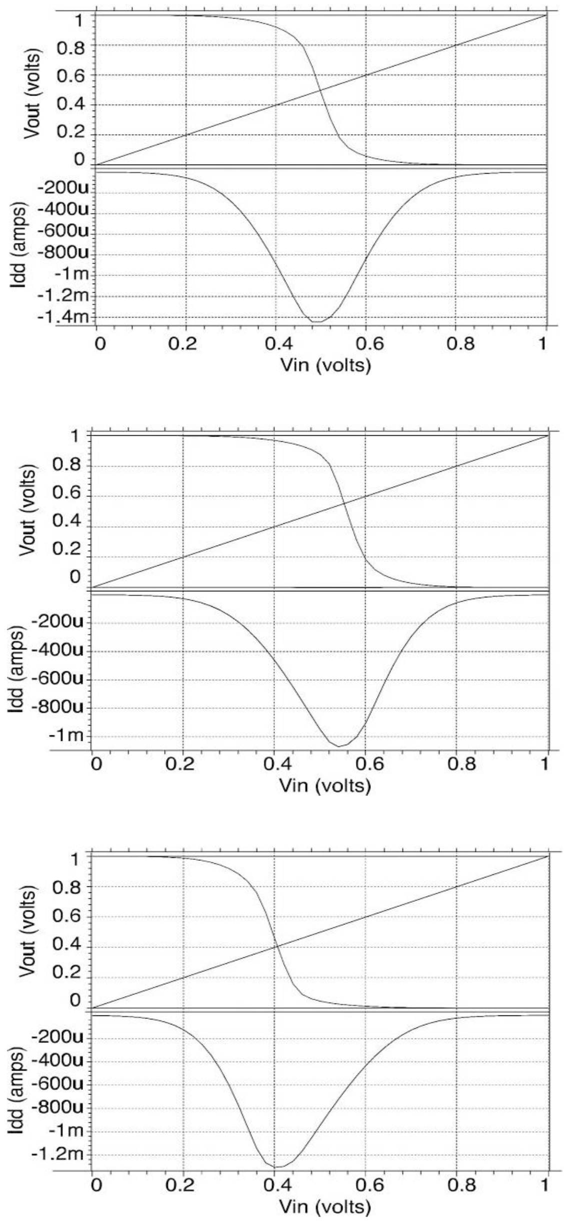


Figure 3.8 : VTC and current curves (INVt, INVr, INVf)

1.) The shapes of the three curves are all very similar. The 'high' horizontal region, its adjacent curve downward, the 'low' horizontal region, its adjacent curve upward, and even the slope of the middle linear transition region are nearly identical for all three graphs. This shape similarity is very desirable because it indicates that the three inverters will behave very similarly in operation.

2.) The input voltages at the switching point are all very different. INVr switches at a much higher input voltage than INVf, and INVt is near the middle point in between them. (The input voltage sweep of Vin=0 to Vdd is plotted to highlight the difference of the three VTC curves.) This difference in switching voltage is also very desirable because it permits the INVr buffer to switch faster and sharper to a falling-edge input, and INVf to switch faster and sharper to a rising-edge input.

The current plots (shown in the lower portion of each of the three graphs) of Figure 3.8 likewise illustrate how the asymmetric inverters, (INVr and INVf) have the same basic operation of the symmetric inverter (INVt), but merely shift the switching point to a higher or lower voltage, respectively. Note that while the peak current of the asymmetric inverters (1.06mA and 1.30mA) is actually less than the symmetric inverter (1.42mA), they each have one slightly wider side of their distribution, so the integrated current of all three inverters are about equal. This current distribution partially explains the fact that the power numbers of the different devices are about the same.

3.3.2 Noise Margins

The voltage levels of an input or output signal may vary significantly from the ideal inverter. Therefore, a practical inverter must be able to convert a 'less-than-ideal' input into a 'good-enough' output in the presence of noise; otherwise, it risks committing logic-like errors -- missed transitions, erroneous transitions, propagated glitches, etc. Such errors on the clock signal would likely cause disastrous errors in the circuit's operation and render the entire chip useless. Therefore, the proper inverting operation of the clock buffers must be ensured over the entire possible range of input and output signal levels -- this robustness to signal level noise is called the inverter's noise margin.

A common definition of noise margin involves the addition of noise to all nodes of the clock tree simultaneously. This metric is called the multiple-source noise margin, MSNM. The MSNM is defined as follows:

$$MSNM_H = V_{OH} - V_{IH} \tag{3-5}$$

$$MSNM_L = V_{IL} - V_{OL} \tag{3-6}$$

From the VTC curves of Figure 3.8 the MSNM for all three inverters are calculated in Table 3-2.[13]

Table 3-2 : Multiple-Source Noise Margins

	INVt	INVr	INVf
$MSNM_H$	0.33 V	0.26 V	0.45 V
$MSNM_L$	0.35 V	0.43 V	0.24 V

The skewing effect of the asymmetric buffers increases one side of the noise margin while decreasing the other side; and the worst-case noise margin represents the dominant

[13] Note that the V_{OH}, V_{OL}, V_{IH}, and V_{IL} data values are taken at the -1 slope points of the voltage transfer curves of Figure 3.8.

measure of the clock tree's noise tolerance. Consequently, the MSNM of a traditional tree will fall from 0.33V to 0.24V when using SEC buffers. However, the MSNM level is still nearly 1/4th of the power supply, so it is still adequate for error-free clock switching, provided, of course, that multiple noise occurrences on the clock branch can be kept to less than 0.25V.

3.4 Full-chip SEC Distribution Design

HSPICE simulations were carried out to compare the relative performance of the clock buffers of the traditional clocking, TC, and single-edge clocking, SEC, methods. The simulations were run with nominal process parameters initially, and then over PVT variations, for the previously described 10x10mm^2 chip design of Figure 3.3. Both the TC and SEC methods use exactly the same wire loading and wire process corners, since the purpose of this experiment is to compare the performance of the TC and SEC buffers head-to-head.

3.4.1 H-tree Buffer Sizing

Generally, clock buffers are sized as big as necessary to achieve the required skew performance [68], but not so big as to violate area and power constraints. As seen later, the total clock power of this design begins to grow unmanageably for buffer sizes of 200X and larger; therefore, the TC versus SEC performance comparisons are made at the 150X size; however, for a more complete comparison, all performance data are plotted over a wide range of clock buffer sizes (50X to 500X). The resulting graphs highlight the anticipated *similar behavior over varying absolute buffer size of the TC and SEC methods*.

For the TC clock network, all the buffers are INVt as in Figure 3.6 . For the SEC clock network, the buffers at levels #1, 2, and 3 are INVf and the buffers at levels #1b and 2b are INVr, also as in Figure 3.6 . The buffers at level #4 are always INVt (for both TC and SEC) to provide two relatively clean clock edges to the leaf FF loads. The absolute sizing of the buffers is applied identically to all buffers of the H-tree for both TC and SEC methods.

3.4.2 Simulation Model -- Single Branch Model

A simple, but essential, reduction of the complete full-chip clock distribution H-tree must be made in order to perform the necessary HSPICE simulations. As shown in Figure 3.3, there are a total of 95 clock buffers and 127 wires comprising the H-tree network (which contains 64 clock leaves and therefore 64 distinct clock branches).

For the simulations carried out here, only one complete clock branch is necessary -- provided of course that its loading is properly calculated and applied. Therefore, the complete architecture of Figure 3.3 is used to calculate the wire and buffer loading of one clock branch for simulations, which is shown in Figure 3.9. This simplified circuit model, which is called the single-branch model, SBM, is a string of six clock buffers, or inverters, connected by wires with every junction fully loaded to replicate the complete full-chip H-tree. The SBM represents any one of the 64 clock branches from root clock to leaf clock with FF load. As labeled in Figure 3.9, the INVf-INVr buffers are alternated (5 levels in total) and an INVt buffer is used at the leaf node to help improve the quality of the neglected clock edge that directly drives the FFs.

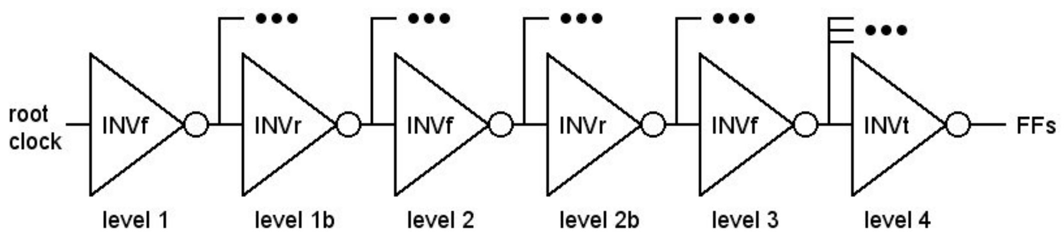


Figure 3.9 : SBM schematic

3.5 Full-chip SEC Performance Results

Figure 3.10 shows the timing performance of the TC design and two SEC designs (all at 150X buffer sizing) directly from HSPICE simulations using their SBM. The time-domain graphs aptly illustrate the distinctive TC and SEC operations -- specifically the latency improvement, neglected edges impairment, and pulse width growth of SEC versus TC. Waveforms for the input root clock and all branch buffers (#1, 1b, 2, 2b, 3, and 4 [in bold dot-dash]) are shown for the three different clocking methods (TC, SEC with η=1.5 (β=0.56 and 6.0), and SEC with η=2.3 (β=0.27 and 11)).

The top graph of Figure 3.10 is for the TC method -- all buffers are of type INVt, all of the leading and trailing edges have similar transition times, and the signal maintains its 50% duty cycle throughout. The middle graph of Figure 3.10 is for a safe SEC method (η=1.5) -- three INVf and two INVr are alternated, terminated by a INVt, the leading (critical) edges are sharper and earlier than those of the TC method, but the trailing (neglected) edges are slower for all but the final leaf clock; therefore, the duty cycle grows from the root to the leaf. The bottom graph of Figure 3.10 is for the practical limit of the SEC method (η=2.3) -- the INVf and INVr β's are spread further apart increasing the SEC characteristics, but the final leaf clock signal nearly consumes the entire clock period, increasing the risk of losing the clock signal prior to triggering the FFs.

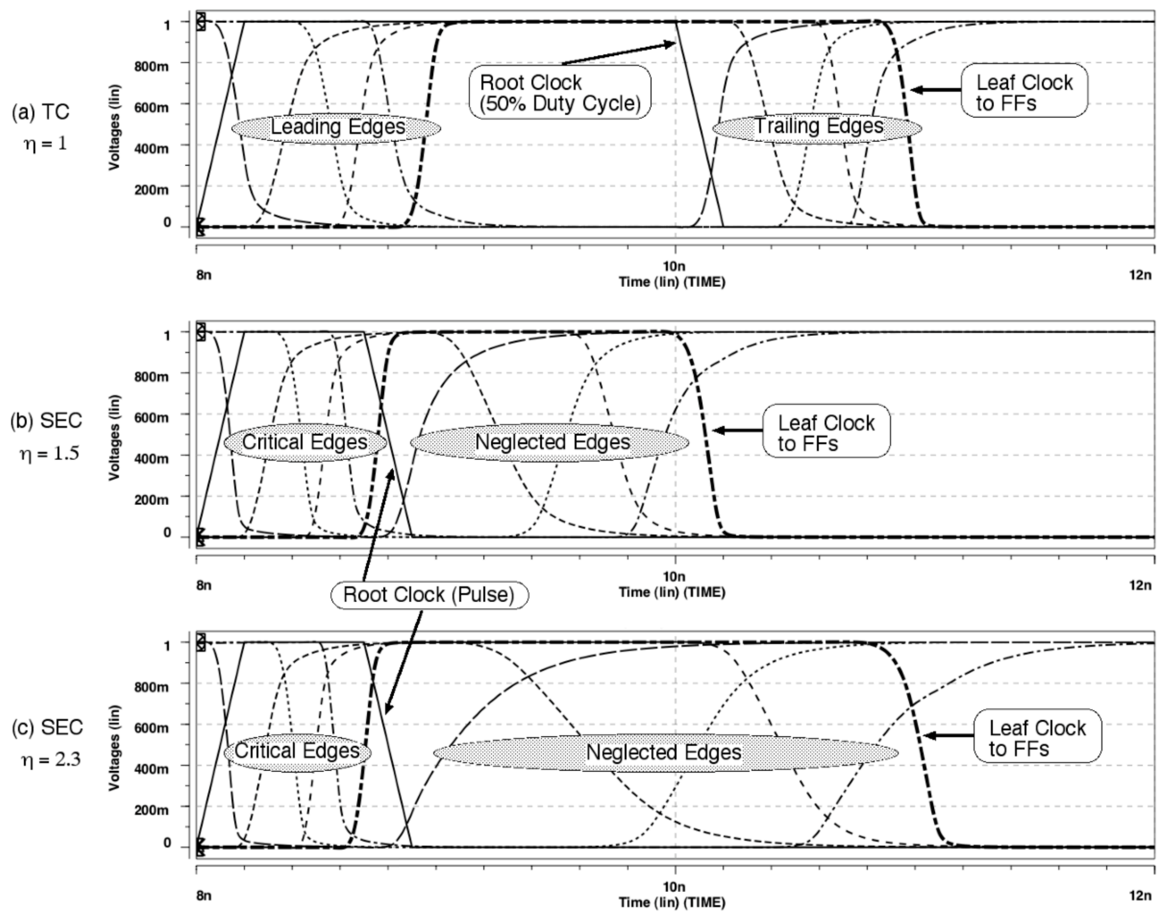


Figure 3.10 : Timing waveforms for TC and two choices of SEC

As mentioned previously, these HSPICE timing simulations were performed over a range of clock buffer sizing (50X to 500X) for the TC (η=1) and SEC (η=1.5) clock buffer designs. The SEC (η=2.3) clock signal is too degraded and is only shown in Figure 3.10 as an example of the potential risks of an aggressive use of buffer asymmetry. A closer look at the critical edge timing performance improvements of SEC over TC are presented in the following sections for latency, skew, and jitter.

3.5.1 Latency Improvements

Latency, or the total path delay from clock root to leaves, is not necessarily as critical for a synchronous system as are skew and jitter. However, when two clock architectures share the same wire H-tree, have the same number of buffers, roughly the same buffer area and power, yet one delivers the critical clock edge significantly earlier than the other, the faster method would be preferred as it tends to improve skew.

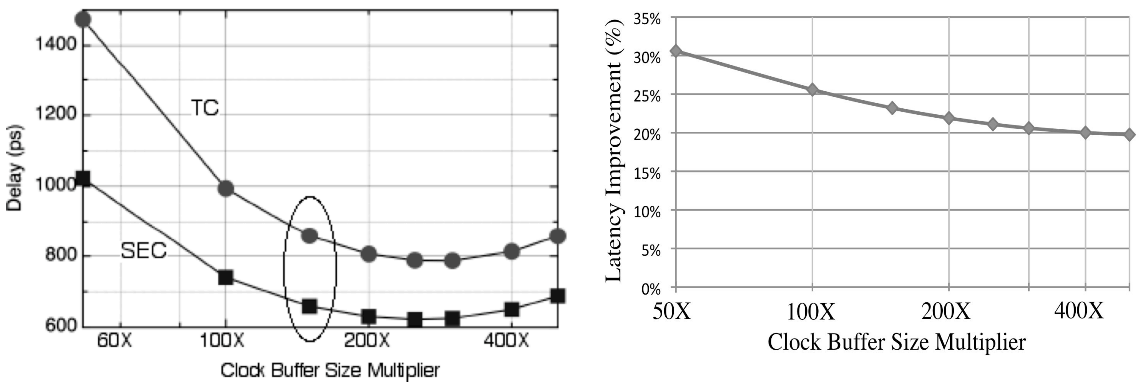


Figure 3.11 : Latency versus buffer size

As seen in the left graph of Figure 3.11 at the 150X size (circled), the SEC latency is nearly 200ps or 23% faster than the TC latency (659ps vs. 858ps) for R_{wire}=300Ω/mm. The similar shapes of the curves also demonstrate that capacitive self-loading of the inverters is behaving as expected for both TC and SEC; therefore, SEC is essentially a simple modification of the CMOS inverter instead of a major transformation. Consequently, similar behavior can be expected between the operation of TC (symmetric) and SEC (asymmetric) designs. The right graph of Figure 3.11 shows the percent improvement of SEC over TC for all simulated clock buffer sizes.

If the metal resistance is reduced to R_{wire}=30Ω/mm, for intermediate metal layers, then the latency improvement at the 150X size increases from 23% to 35%. At the higher levels of metal, with R_{wire}=3Ω/mm, the improvement is 38%. Clearly, this SEC approach is well suited for the clock distribution networks that are mostly routed at the upper layers of metal.

3.5.2 Skew Improvements

Skew, or the spatial variation of clock edges, can be caused by on-chip PVT variations of one clock path or branch versus another branch. While the precise level of skew experienced across any clock distribution network requires accurate models of random and systematic intra-die variations [2], a relative comparison of the degree of skew of two clock distribution methods can be obtained by measuring their worst-case skews. Two simulations were performed on the SBMs of the TC design and the SEC design; the first with Slowest (SS, 0.9V, 150^{o}C) PVT parameters[14], and the second with Fastest (FF, 1.1V, 0^{o}C) parameters. The difference of their slowest and fastest latencies (overall path delays from root to leaf) is a measure of their *worst-case skew*, as described in Section 2.2.

The left graph of Figure 3.12 shows that the worst-case skew for SEC is 25% less than that of TC (322ps vs. 431ps) at the 150X size for R_{wire}=300Ω/mm. It would not be accurate to state that SEC has 110ps less skew than TC, since these are unrealistic worst-case numbers and actual intra-die PVT variations would perhaps be half of these extremes. However, the *relative* improvement of 25% less skew for SEC versus TC is still a valid conclusion. So, if for example a TC distribution network has 200ps of skew, then the comparable SEC network could be expected to have only 150ps of skew -- a considerable

[14] The PVT parameters are listed in the order (NMOS-PMOS, Supply Voltage, Temperature); also, SS refers to Slow-Slow for NMOS-PMOS, while FF refers to Fast-Fast and TT refers to Typical-Typical.

improvement. The right graph of Figure 3.12 shows the percent improvements of SEC over TC for all simulated clock buffer sizes.

Furthermore, similar to the above latency numbers, if the metal resistivity is reduced to R_{wire}=30Ω/mm for intermediate metal layers, or to R_{wire}=3Ω/mm for upper metal layers, then the worst-case skew improvement increases from 25% to 36% for each.

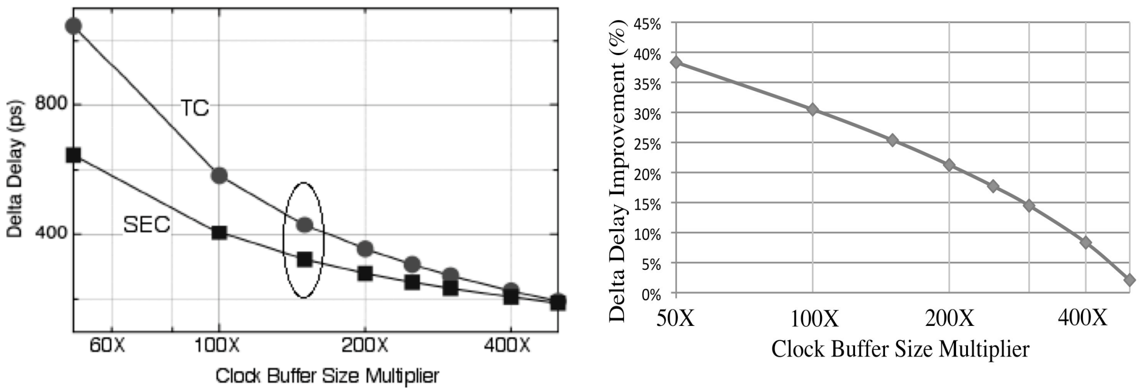


Figure 3.12 : Worst-case skew versus buffer size

3.5.3 Jitter Improvements

Jitter, or the temporal variation of clock edges, is primarily caused by noise impulses on the clock signal, and is therefore very difficult to simulate. However, previous work on jitter, or phase noise, has developed a metric of Impulse Sensitivity Function (ISF) which relates jitter performance to clock edge transition slope [15]. Each transition of the critical edge of the clock leaves is exposed to these noise impulses and, therefore, vulnerable to jitter. When the clock is not transitioning (at a high or low level) or when the neglected edge is transitioning, then noise impulses are not as much a concern for jitter. Therefore, to gain some insight into the level of jitter improvement that can be obtained, the transition times of the critical edges (at #1, 1b, 2, 2b, and 3) are averaged to compare SEC to TC.

As seen in the left graph of Figure 3.13, the amount of average transition time of the critical edge is 11% less for SEC than TC (323ps vs. 364ps) at the 150X size for R_{wire}=300Ω/mm. Calculating the rms value of ISF across an entire clock period, however, is a more accurate measure of jitter performance and, as derived in [15], that equation becomes a cubed relationship. Hence, the potential jitter improvement of SEC over TC would be (0.89^3), which is a full 30% improvement in jitter performance. The right graph of Figure 3.13 shows the percent improvements of SEC over TC for all simulated clock buffer sizes.

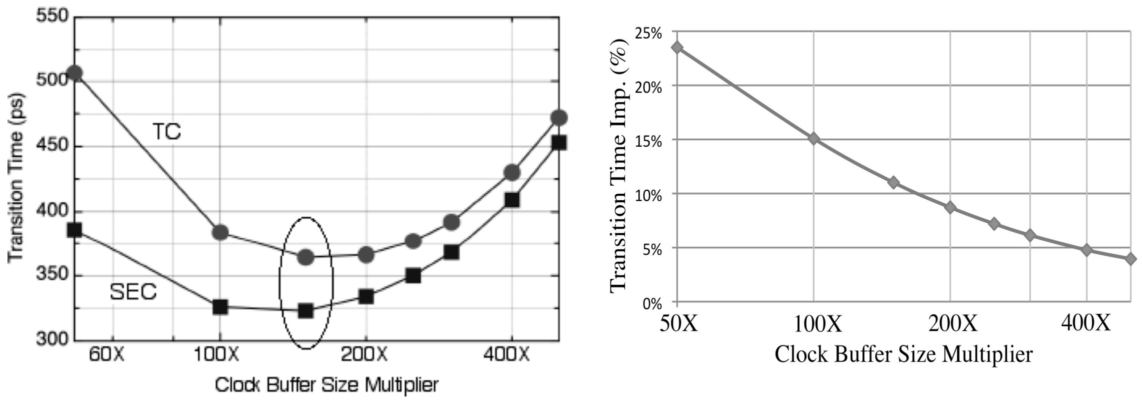


Figure 3.13 : Average transition time versus buffer size

Once again though, using R_{wire}=30Ω/mm for intermediate metal layers or R_{wire}=3Ω/mm for upper metal layers further increases the above 11% improvement to 29% or 32%, respectively. Hence, the chosen lower metal resistivity is clearly the conservative selection in reporting these results.

3.5.4 Power Consumption and Layout Area

Usually, there is a power and/or area tradeoff of a given method relative to another method, but this is not the case for SEC vs. TC. As the graph of Figure 3.14 shows, the average current consumption (and likewise the power consumption) of the TC and SEC designs are virtually identical for all clock buffer sizes (50X to 500X) and for all three PVT conditions: Fastest (FF, 1.1V, 0^{o}C), Nominal (TT, 1.0V, 75^{o}C), and Slowest (SS, 0.9V, 150^{o}C). In fact, they are close enough to be indiscernible in the graphs. This nearly identical power result is expected from the current comparisons of the inverters' voltage transfer curves and DC load lines, and these SEC buffers can be laid-out within the same area as the TC buffers by aligning their transistor widths. Therefore, *the significant timing improvements in latency, skew, and jitter can all be achieved using SEC without any significant increase in power consumption or layout area relative to TC.*

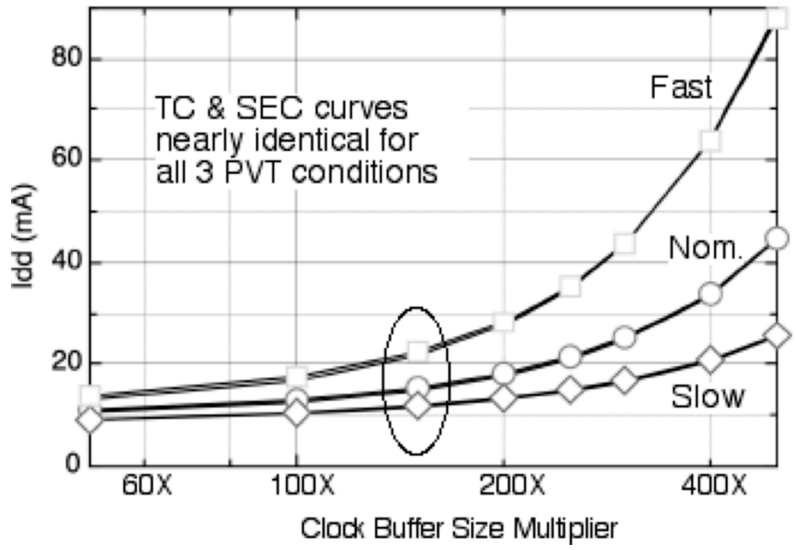


Figure 3.14 : Average Idd versus buffer size

Alternatively, the better timing performance described above could be traded-off to reduce clock buffer area and clock distribution power by essentially 'under-sizing' the clock buffers. As seen in the latency, skew, and jitter performance graphs of Figure 3.11, Figure 3.12, and Figure 3.13, *equal or better timing performance* can be obtained with the 100X SEC buffers as with the 150X TC buffers, resulting in a roughly 33% reduction in clock buffer gate area and a 12% reduction in total nominal clock distribution current (i.e., power) -- 14.7mA for 150X TC down to 12.9mA for 100X SEC from Figure 3.14. Therefore, the SEC technique not only allows for clock timing improvements, but it could also be used for clock power reduction (a significant portion of overall chip power consumption) over a traditional clock distribution network by more than 10% in this example.

3.6 Full-chip SEC Conclusions

By analyzing the clock signal waveforms for a typical ASIC synchronous circuit design, a valuable trade-off from the non-critical clock edge to the critical clock edge has been illustrated in Figure 3.2. The recognition that a single clock edge is critical for the FFs' timing has led to the concept of single-edge clocking, which implies the use of alternating, asymmetric buffers to deliver that one critical clock edge faster and sharper, while neglecting the other clock edge as an acceptable trade-off. The degree of that trade-off is captured in the metric called the degradation factor, η, defined as the ratio of the neglected clock edge to the critical clock edge, t_{negl}/t_{crit}. Equations have been derived to calculate acceptable degrees of trade-off for any given clock distribution network.

This chapter has also shown how to start with a traditional, symmetric clock inverter (or buffer) and adjust the pull-up and pull-down networks in order to create asymmetric inverters that improve the rising or falling edge of the clock signal. So long as the sum of the transistor widths (W_N + W_P) is held uniform, these new SEC inverters can be used interchangeably with the traditional inverters. The drive strength of the critical edge of the SEC inverter will be improved over the traditional inverter, but the input and output capacitances of the SEC inverter as seen by the rest of the circuit will remain virtually the same. However, since these SEC inverters target a single critical edge of the clock, SEC uses two complementary inverter designs (INVr and INVf) instead of just the traditional inverter (INVt). Furthermore, if these new SEC inverters are laid-out with their PMOS and NMOS widths along the same axis, then the asymmetric inverter's (INVr or INVf) footprint will be identical to the symmetric one's (INVt) – allowing for clock timing changes very late in the overall design process.[15]

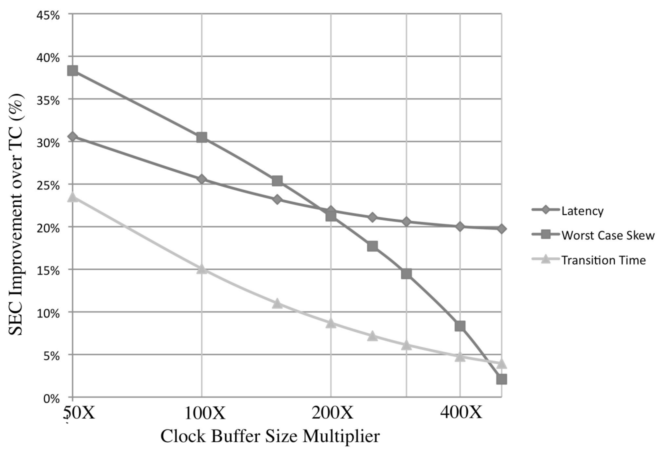


Figure 3.15 : Percent Improvement of SEC over TC

Finally, a hypothetical, but typical, clock distribution network was designed in both the traditional way (TC) and the new, asymmetric way (SEC). A comparison of HSPICE simulation data shows that *timing improvements of more than 20% are achievable for clock latency, skew, and jitter -- all without any increase in power consumption or layout area*. Conversely, significant power and area improvements can be made without any decrease in timing performance. The resulting percent improvements for SEC over TC for latency, worst case skew, and transition time are once again plotted in Figure 3.15. Significant timing improvements (15-35%) will be realized at typical clock buffer sizes by using SEC.

[15] Note that even if they are not laid-out in this orientation, they will still have equal area, just not the exact same footprint.

Chapter 4 : Tunable SEC Buffer Design and Applications[16]

The design of SEC buffers was covered in detail in Chapter 3, and a full-chip SEC implementation was shown to improve clock performance in a full H-tree by over 20%. In this chapter, two additional specific applications of these asymmetric buffers will be proposed for typical ASIC synchronous circuits. First, a delay adjustable clock buffer, or tunable buffer (TB), will be designed for use in SEC networks. Adjustability of the clock edge is a requirement for most post-silicon skew compensation schemes, so tunable buffers are expected to become necessary as PVT variations continue to increase. Then, a single-use application of SEC buffers will be proposed to correct circuit timing-errors, which may arise late in the design stage. Thanks to the interchangeability of traditional and SEC clock buffers, small but potentially catastrophic timing errors can be corrected immediately before the tape-out of a final circuit design by the drop-in replacement of a faster (or slower) SEC buffer.

4.1 Tunable SEC Buffers

The SEC technique now will be used in conjunction with a tunable clock buffer capable of adjusting its critical clock edge in an attempt to cancel out any detected clock skew. As with traditional clocking, TC, systems, a tunable buffer is ultimately needed for SEC to compensate for increasing PVT variations. In its most basic form, a delay buffer is created by slowing-down a typical buffer or inverter. Three common methods to slow down an inverter are: 1) adding load capacitance to the inverter output, 2) limiting (or 'starving') the inverter power and/or ground supplies, and 3) feeding-back the inverter output to

[16] Most of the research covered in this chapter has been published in: J. Mueller and R. Saleh, "A tunable clock buffer for intra-die PVT compensation in Single-Edge Clock (SEC) distribution networks," *Proc. of ISQED*, pp. 572-7, Mar. 2008.

contend with itself. Methods 1 and 2 will be investigated further in this chapter and will be referred to as tcp-INVf and tsi-INVf, respectively. Method 3 has been used in industry and is called a Clock Vernier Device (CVD) [69], but the node contention that exists at its own output proves to be too much for the neglected edge of the SEC technique, so it is not a viable option with SEC designs, and will therefore not be analyzed here.

For the particular clock network considered in this chapter, only the INVf buffers are made tunable; however, the same principles can be applied to INVr buffers, although, they will require larger areas for their dominant PMOS devices.

4.1.1 Tunable Buffer Design

The two most common types of delay buffers (capacitive load and starved inverter) will be described for the asymmetric clock buffers of the SEC technique. Selectable (or programmable) delay can be easily added to any inverter (TC or SEC) through the addition of various transmission gates; essentially, the design of an SEC tunable buffer parallels the design of a TC tunable buffer. However, the neglected edge of the SEC waveform creates a unique complication in its timing delay; therefore, only single-edge delay techniques will be compatible with SEC networks as shown in the following analysis.

4.1.1.1 Capacitive Load

Figure 4.1 shows a common delay buffer, which will be called a 'tcp-INVf'. This extremely common delay buffer will be compared later to a specially-designed, single-sided, starved inverter in order to highlight the particular needs of SEC networks. The capacitive load delay of tcp-INVf is created by adding selectable capacitance via transmission gates to the output of a typical clock buffer.

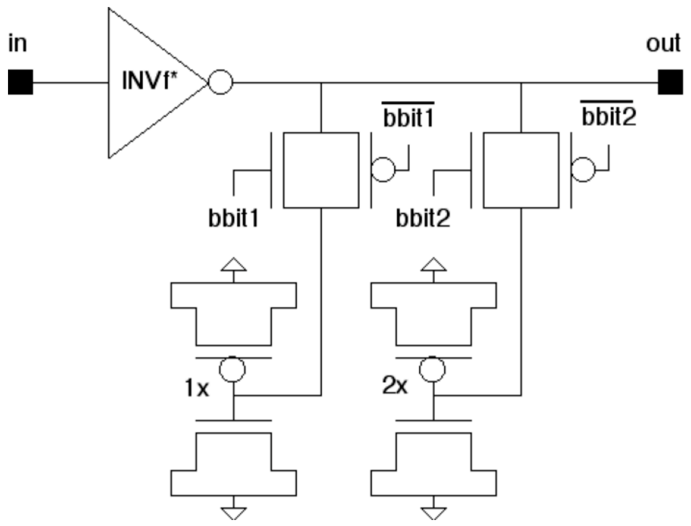


Figure 4.1 : Capacitive load delay buffer

In this case, the typical clock buffer is an INVf of the SEC technique, as shown in Chapter 3. The near linear delay behavior of the shunt capacitive loads allows for the two capacitors to be binary weighted (indicated as 1x/2x), thus providing 4 selectable delay steps in total {00, 10, 01, 11}, where the ordering implies settings for bbit1 and bbit2.

As capacitance is added to a clock line, the signal transitions will be slowed down on both rising and falling edges; that is, both the critical and neglected clock edges. Since the SEC technique, by itself, already relaxes the neglected edge considerably, any additional delays on that neglected edge could inadvertently push it beyond the clock period boundary. Therefore, one concern is that the tcp-INVf delay buffer will be somewhat limited in the SEC application.

4.1.1.2 Starved Inverter

The starved inverter method for adding delay has several configurations (analog or digital, footer and/or header). The version shown in Figure 4.2 will be called a 'tsi-INVf' design. For the tunable SEC approach, the fundamental goal is to control the delay of the critical edge while allowing the neglected edge to pass unaffected. The critical, falling edge in this case, is delayed by the NMOS footers connected to ground, but the neglected edge is passed 'normally', as with a standard SEC buffer. (The same could be said, in reverse, for a tsi-INVr tunable buffer.) Since this configuration only delays one edge of the clock signal, it will be called a single-sided starved inverter.

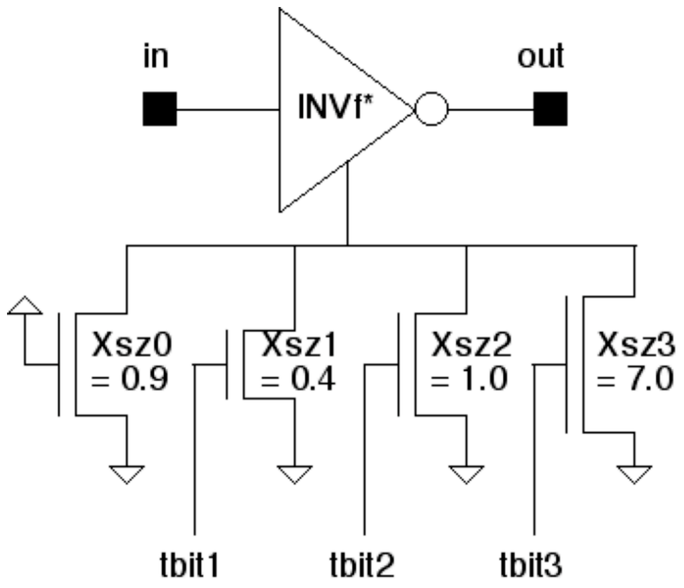


Figure 4.2 : Single-sided starved inverter delay buffer

These particular tunable delay buffers have been designed such that their critical edge has equal positive and negative tunability with respect to an untunable INVf buffer. The negative delays are made possible by a more aggressive β sizing of the INVf shown in Figure 4.1 and Figure 4.2. This centering technique is further explained in Section 4.2.2. However, their relative tuning delays are a valid metric for this SEC tunable buffer comparison, whether or not the centering option is ultimately utilized.

For the sizing listed in Figure 4.2, all possible combinations of tbits (thermometer code bits) and their corresponding delay values are shown in Table 4-1. Three of the codes: 011, 101, and 001 (as indicated in italics in the table) lie so close to the 111 code that they are of limited use.

Table 4-1 : Delay values for all possible tbit settings

tbit code	111	*011*	*101*	*001*	110	*010*	100	000
delay (ps)	-24.9	*-24.7*	*-24.3*	*-24.1*	-9.3	*-4.7*	+7.6	+25.7

For the purposes of SEC applications, a thermometer code is the most convenient way to adjust the delay. Therefore, the footers have been sized to allow for 4 steps of thermometer-code selectable delay {111, 110, 100, 000}, where the ordering implies settings for tbit1, tbit2 and tbit3. As a result, this tsi-INVf design has four, linear selectable delay settings, just like the preceding tcp-INVf design.

Sizing the NMOS footers of this tsi-INVf is rather straightforward. Let the relative size of each transistor of Figure 4.2 from left to right be *Xsz0, Xsz1, Xsz2, and Xsz3*, and let *Δt=±25ps* be the desired tuning range. The value of delay follows an inverse linear relationship with the sum of 'on' transistor sizes. First, the 'always-on' footer, *Xsz0*, is sized to set the maximum delay (for a tuning range of ±25ps), *Xsz0=0.9*.

Next, the other three footers are sized to sum to the minimum delay, sum *Xsz=Xsz0+Xsz1+Xsz2+Xsz3=9.3*. Then, those two values can be used to determine the y-intercept and slope of the linear equation, *y=mx+b*:

$$m = \frac{\left(\frac{1}{Xsz0} - \frac{1}{\sum Xsz}\right)}{(\#\,totalsteps - 1)} = \frac{\left(\frac{1}{0.9} - \frac{1}{9.3}\right)}{(4-1)} = 0.33 \tag{4-1}$$

$$b = \frac{1}{\sum Xsz} = \frac{1}{9.3} = 0.11 \tag{4-2}$$

All the footer sizes can then be taken from the inverse of that linear equation, as follows:

$$Xsz0 + Xsz1 = \frac{1}{m(2)+b} = 1.3 \Rightarrow Xsz1 = 0.4 \tag{4-3}$$

Similar calculations lead to *Xsz2=1.0* and *Xsz3=7.0*.

4.1.2 Tunable SEC Simulation Results

Using the sizing process described above, HSPICE simulations were performed on both the tcp-INVf and tsi-INVf delay buffers in order to compare their clock tuning accuracy. Both delay buffers were designed to have 4 linear tuning steps over a 50ps tuning range. The tcp-INVf used a 2-bit binary code, and the tsi-INVf uses a 3-bit thermometer code.

4.1.2.1 Single Branch Model (SBM)

The same single-branch model, SBM, constructed in Section 3.4.2 is used in the HSPICE simulations. It accurately models a standard, balanced fanout-of-4, 4-level H-tree for a hypothetical 10x10mm^2 chip. Initially, as per the SEC technique, the buffers at levels 1, 2, and 3 are INVf, the buffers at levels 1b and 2b are INVr, and the buffers at level 4 are always INVt. The 150X sizing of the buffers is applied identically to all buffers of the H-tree. Next, the clock edge tunability will be added to this standard SBM via the two chosen tunable SEC buffers; first tcp-INVf and then tsi-INVf.

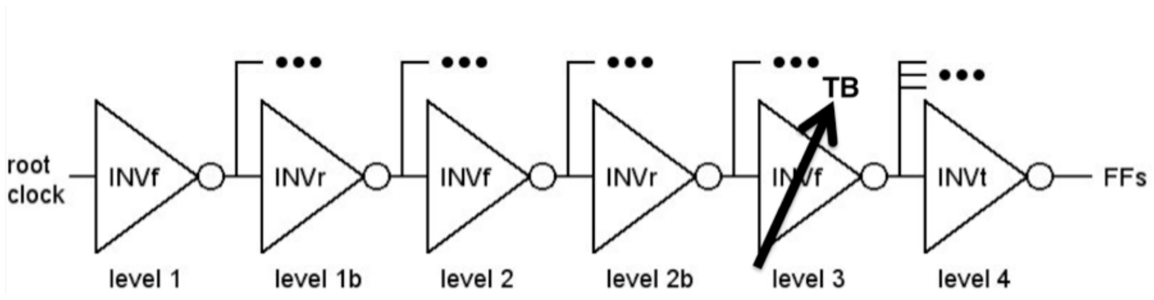


Figure 4.3 : SBM schematic with Tunable Buffer (TB)

The simple tunability scheme used here replaces the level-3 SEC buffer (INVf) with the new tunable SEC buffer (tcp-INVf or tsi-INVf), as shown in Figure 4.3. For the complete chip H-tree, this tuning configuration results in 16 zones of independent tunability, each with Δt=±25ps tuning range. Theoretically, if one zone was skewed 60ps ahead of another zone, the 'fast' zone could be tuned to +25ps and the 'slow' zone could be tuned to -25ps, thereby reducing the 60ps of skew down to just 10ps. Larger windows of tuning, such as ±50ps, are possible for both delay buffer designs with a simple resizing of their capacitance or their pass transistors.

4.1.2.2 Tuning Range Waveforms

Simulations were carried out in HSPICE to compare the familiar tcp-INVf with the new tsi-INVf in the context of this 16-zone tunable H-tree. To measure delay tunability, the level-3 buffers were replaced with tunable buffers, and the tcp-INVf and tsi-INVf buffers were simulated for all four delay settings. The transient response waveforms of Figure 4.4 show nine leaf clocks (output of the level 4 buffers from Figure 4.3) that drive the circuit's FFs. The bold waveform labeled "SEC Reference" is obtained by keeping the level-3 buffer as a fixed, non-tunable INVf in order to observe the standard SEC timing as a reference.

The top graph of Figure 4.4 demonstrates what happens with the neglected edges from the centered, tunable buffers. First note that due to the SEC β resizing to center the tunable critical edges about the reference (as explained in the Section 4.2.2), all of the tunable neglected edges are delayed with respect to the reference, an untunable INVf. Of primary significance, however, is that the tsi-INVf buffer delays only its critical edge, so all four of its neglected edges appear virtually identical, as noted in the graph.

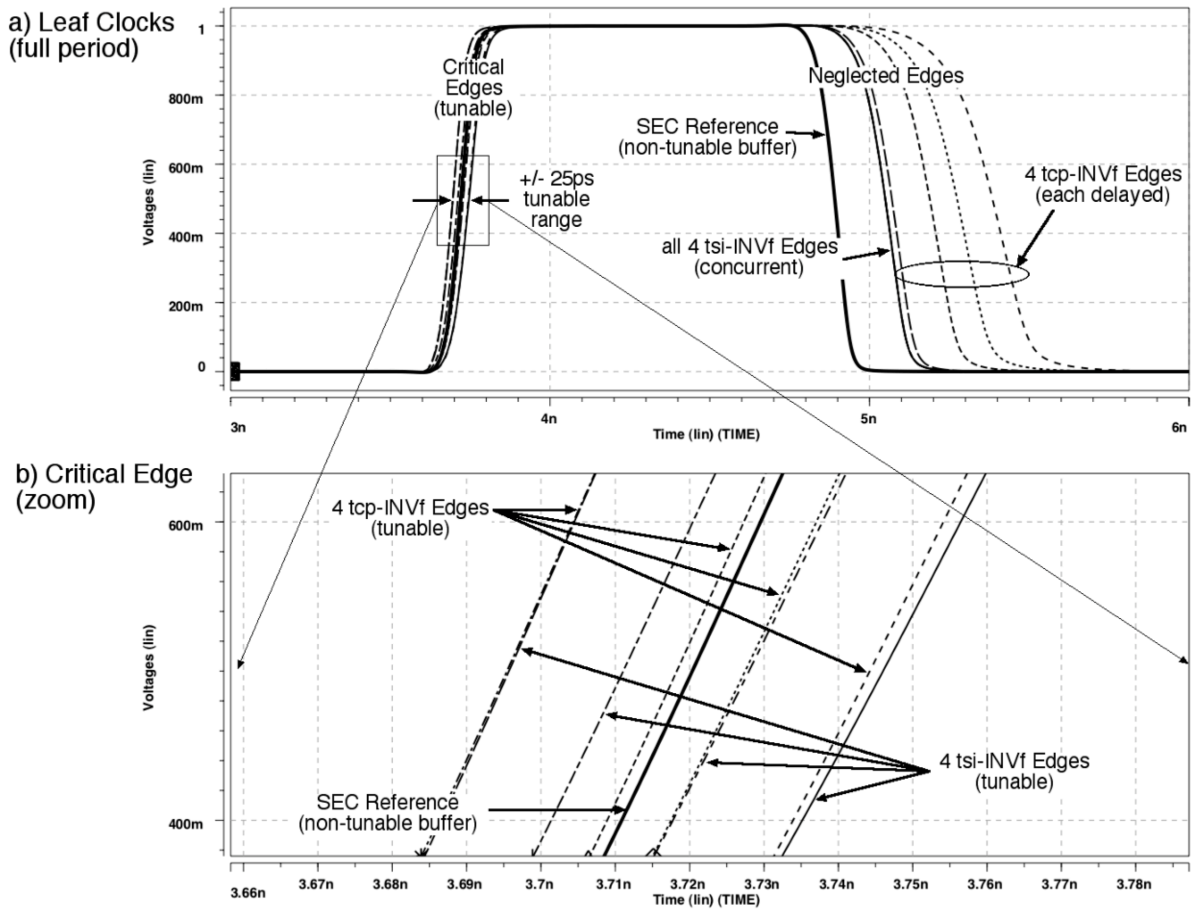


Figure 4.4 : Tsi and Tcp waveforms over 4 delay settings

Note that the tcp-INVf buffer delays both its critical and neglected edges, so four distinct neglected edges (one for each of the four delay settings) can be seen in the top graph. Since the tsi-INVf buffers do not degrade the neglected edges as much as the tcp-INVfs, the tsi-INVf buffers are able to operate with a smaller clock period (i.e., a higher clock frequency) than the tcp-INVfs, and therefore tsi-INVfs are more suitable for use in the SEC system.

A portion of the critical edges of the leaf clocks are shown with a higher resolution in the bottom graph of Figure 4.4. As noted on the graph, the "SEC Reference" is right in the center of the tunable range as intended -- 2 tcp-INVf or tsi-INVf settings can be used to tune a negative delay, and 2 tcp-INVf or tsi-INVf settings can be used to tune a positive delay. It is evident from the curves that the timing performance of the new tsi-INVf design is equal to, or better, than that of the more familiar tcp-INVf design (i.e., tsi-INVf waveform edges show more uniform step sizes than those of the tcp-INVf).

4.1.2.3 Area and Power Comparisons

Similar to the timing delays, the silicon area of the tsi-INVf design is much more 'invariant' to delay values than that of other tunable buffer designs. The area of the tsi-INVf design remains at a near constant 4X of a comparable, non-tunable buffer -- a useful characteristic if tuning ranges and/or values will be decided/changed late in the design process -- whereas the area of the tcp-INVf design will vary widely depending upon the tuning range (which directly affects its capacitance sizing).

As highlighted in Figure 4.4, the neglected edge of the SEC method causes difficulties for the tcp-INVf design and, therefore, its has a higher amount of crowbar current relative to the tsi-INVf design. But, even if the clock edges were better controlled, the tcp-INVf design would still consume significantly higher power due to its larger capacitance. What is important to note, however, is that these trade-offs do not exist for the tsi-INVf buffer. The tsi-INVf buffer has only a trivial increase of a few percentage points in average current versus the original, non-tunable SEC buffer. Therefore, from a nominal operational perspective, the tsi-INVf delay buffer design is the proper choice for SEC networks. Its performance over PVT variation is also very critical; thus, the tsi-INVf tunable buffer will now be analyzed over PVT variations.

4.1.3 PVT Variation Results

As PVT varies for both inter- and intra-die scenarios, it is important that the tunable delay buffer range continues to be centered on the reference edge, and also that the step size remains relatively constant for the entire range. Ideally, for this chosen Δt=±25ps buffer, the delay steps would be located at -25ps, -8.3ps, +8.3ps, and +25ps with respect to the reference. The plots of Figure 4.5 show the four tuning step locations for the Δt=±25ps range tsi-INVf buffers over the design target extremes of process, voltage, and temperature, respectively. As shown, the tsi-INVf design is able to maintain 40ps of tuning range with more or less uniform step sizes for every PVT condition.

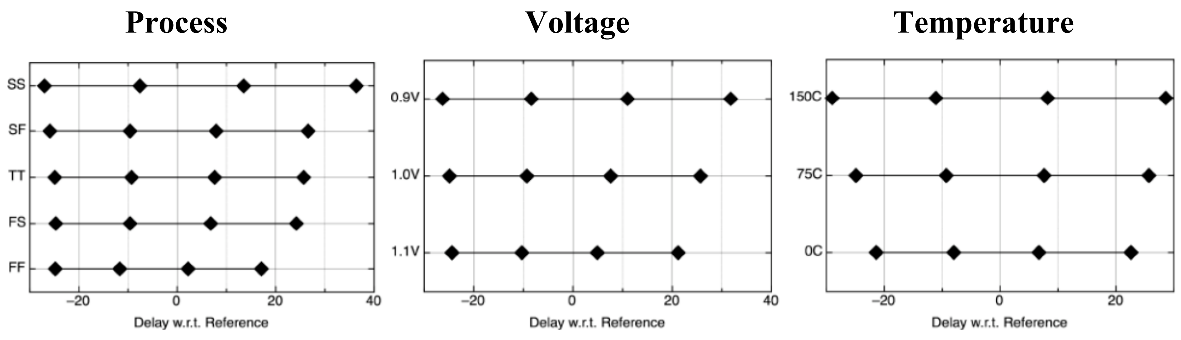


Figure 4.5 : Process, Voltage, and Temperature tuning ranges

Additionally, the post-tuning skew improvements from all of these PVT conditions are plotted in Figure 4.6. The clock skew pre-tuning is plotted on the x-axis, with the corresponding clock skew post-tuning on the y-axis. Each PVT variation case produces a slightly different sawtooth waveform. The results demonstrate that, for any skew between -30ps to +30ps, the tuning buffers can ideally reduce it to under 12ps. The buffers have been designed for this range, and the post-tuning skew could be reduced below this 12ps level if more tuning steps were added to the tsi-INVf design.

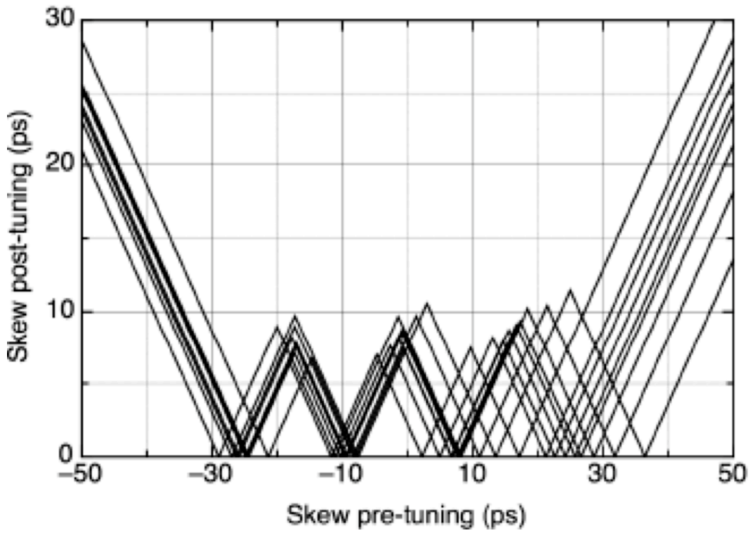


Figure 4.6 : Pre-tuning skew versus post-tuning skew

4.1.4 Tunable SEC Buffer Limitations and Conclusions

Two of the most common delay buffer designs (capacitive loading and single-sided starved inverter) have been applied to SEC buffers (tcp-INVf and tsi-INVf, respectively). The tsi-INVf delay buffers can only be controlled by thermometer code settings (not binary), so having more than 4 or 5 steps per buffer could be difficult, but two or more of these buffers could be used within a particular clock tree branch or path to provide for more tuning steps and a wider tuning range. More complicated methods of 'starving' the pull-down side of an inverter have been proposed, such as a PMOS vector controlling a current mirror [70] or a matrix of NMOS footers [71]; however, the single level, four NMOS footer design of tsi-INVf presented here was selected because it is capable of meeting the chosen tuning design targets with less power and area.

It has been demonstrated that clock skew in the presence of PVT variations can be effectively managed with SEC tunable clock buffers. The neglected edge of the SEC technique proves to be too sensitive for traditional tunable clock buffers (such as the tcp-INVf); therefore, the tsi-INVf tunable buffer, which delays only the critical edge, has been proposed. Its single-edge delay design makes it ideally suited to SEC applications; however, since only one clock edge is critical in the vast majority of synchronous circuits, the tsi-INVf buffer could work equally well in TC applications. HSPICE simulations have shown that it provides adequate tuning steps across a reasonable tuning range, while remaining largely insensitive to the unique SEC waveforms. Assuming that the proper tuning code can be determined on-chip (as covered in the following chapter), these tunable SEC buffers can be used to reduce clock skew significantly in the presence of intra-die PVT variation.

4.2 Individual SEC Buffers for Clock Edge Adjustments

The final application for SEC buffers takes advantage of their superior critical-edge performance to refine the clock signal timing in either TC or SEC networks. As explained previously, only the single clock edge that triggers the load FFs is critical for synchronous circuit timing. Therefore, any favorable adjustment to that critical edge will carry forward in the clock branch's timing regardless of the corresponding degraded impact on the neglected edge -- assuming of course that the buffer is not too asymmetrical as to lose the clock switching.

This individual application of an SEC buffer will be described for two distinct usages: first, for the fine tuning of a clock signal in any existing clock distribution network (TC or SEC); and second, for the centering of a tunable, or adjustable, buffer's branch timing with that of an untunable, or fixed, branch timing. In both cases, it is possible to fine-tune the critical clock edge of any synchronous circuit merely by adjusting the β of a single clock buffer, or inverter.

4.2.1 Single SEC Buffer Substitution

When one or more clock signals are not sufficiently synchronized at the clock tree's leaves, clock timing re-design (or re-timing) must be performed. Typically, that re-timing involves adding more capacitance on one or more clock branches, either by wire snaking [8] or increasing capacitive loading, in order to slow down the faster clock branch or branches. This technique of adding capacitance allows for timing adjustments in only one direction: slowing down or delaying the clock signal. Therefore, it is somewhat limiting and restrictive; it slows down the speed of the circuit, whereas the usual design goal is to speed up any given circuit.

On the other hand, selective replacement of TC buffers with asymmetric SEC buffers (or SEC buffers with even more asymmetric SEC buffers) allows for timing adjustments in both directions: slowing down or speeding up of the clock signal. Therefore, this method of clock timing re-design (or leaf clock synchronization) is far more powerful than traditional, unidirectional methods. The timing characteristics of an existing clock tree can be finely tuned by strategically converting select clock buffers to more asymmetric SEC buffers in order to reduce (or increase) a given clock branch's latency and thereby synchronize the clock leaf timing.

Previous data has shown that converting 5 buffers (in a six buffer string) from TC to SEC can reduce overall clock latency, or delay, by 200ps. Therefore, on average, each of the chosen SEC buffers (η=1.5) reduces clock delay by 40ps versus a TC buffer. From the timing plots versus β's of Chapter 3, it is quite easy to see how any delay reduction up to a maximum limit of ~40ps would be possible, simply by adjusting the β of a single clock buffer. A clock branch could be made faster by any amount up to ~40ps simply by tweaking the inverter layout (as mentioned previously) to favor the critical clock edge while slightly neglecting the other clock edge. Such an adjustment could prove immensely useful in tuning the critical clock signals of an entire clock distribution network very late in the design or layout stage[17].

One important drawback with the single buffer replacement is the resulting unbalancing of the buffer pull-ups and pull-downs as described in [45]. Again, this unbalancing of the clock branch that contains the single-buffer being converted to SEC has the effect of improving, or strengthening, the critical clock edge while degrading, or weakening, the other clock edge; unfortunately, it also unbalances this clock branch with respect to the other clock branches, potentially increasing the skewing effects of some intra-die PVT variations. However, this unbalancing may be acceptable, and even desirable, for a given clock distribution network, so the designer is advised to consider this trade-off when using an individual SEC buffer for critical clock edge fine-tuning.

[17] Recall from the previous chapter that properly orientated inverters can be modified from TC to SEC without any significant change in layout area or power consumption.

4.2.2 Centering Tunable Buffers with SEC

The second usage of individual SEC for clock timing refinement involves the SEC tunable buffer from the previous section. One common application of a tunable buffer would be to include it within one clock branch in order to allow for clock edge adjustments with respect to another untuned, or fixed, clock branch. Simply swapping one fixed buffer for a tunable buffer, however, will necessarily increase the latency, or delay, of the tunable clock branch, due to the larger loading of the tunable components, regardless of the tunable buffer architecture. This consequence automatically skews the two clock signals with respect to each other, and forces some type of timing re-design.

Similar to the re-timing explained in the preceding section, capacitive loads could be placed on the untuned, or fixed, clock branch to re-align the two clock edges, but that adjustment would have major repercussions throughout the rest of the clock tree timing. Another option would be to insert similar tunable buffers in all clock branches, but only use the tunable feature of the desired one. This method would slow the clock signal down in every branch, and would also require significant additional layout area for the unused tunable buffers. A better method of re-timing the two clock edges (tunable branch versus fixed branch) into synchronization would be to speed up the tunable clock buffer branch by an amount equal to its larger loading delay; thereby canceling-out its inherent timing mismatch and leaving all of the fixed clock branches unmodified.

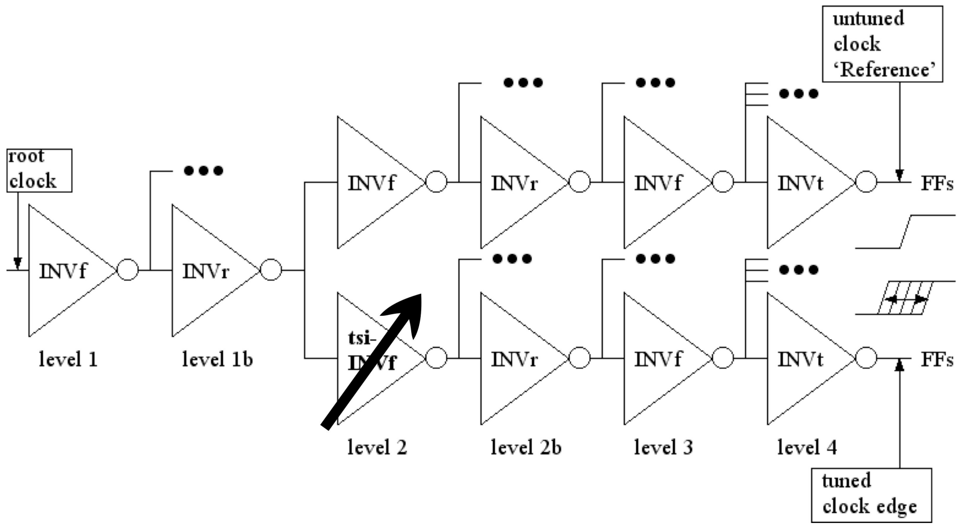


Figure 4.7 : Centering the SEC tunable buffer

Figure 4.7 represents two paths of a typical H-tree (shown in Chapter 3). Instead of having to replace all 'like' buffers (level-2 buffers in this case) in a distribution network with tunable buffers, it is possible to center a tunable branch to one or more non-tunable branches, without modifying the non-tunable branches at all. With traditional, symmetric clocking (TC) methods, adding tunability to a clock branch (the bottom one of Figure 4.7) would increase its latency by some amount inherent to the design of the delay buffer (even when it is tuned for no delay). This unwanted delay necessitates the addition of 'extra' delay within any non-tunable branch (the top one of Figure 4.7) in order to keep them nominally balanced/synchronized.

With the SEC method, the freedom of adjustable-β inverters allows for the range of the tunable branch to be easily centered on the non-tunable branch(es). Just as switching an INVt (β=2.5) to an INVf (β=0.56) reduces the delay through the inverter, lowering the β even further, say to tsi-INVf (β=0.21), can speed up the tunable branch with respect to the non-tunable, reference branch enough to center the tuning range. This centered tuning range allows both positive and negative tuning for skew compensation.

By centering the tunable range around a standard SEC buffer, not all buffers on a given level of the H-tree would need to be replaced with tunable buffers (TB) to keep the H-tree in balance. For example, in Figure 4.7, each level 1b buffer branches out to two level 2 buffers, one up and one down. A tuning scheme could be developed, as partially shown in Figure 4.7, in which both of the 'up' level-2 buffers are kept as regular SEC (non-tunable) buffers while both of the 'down' level-2 buffers are replaced with tunable buffers. Then, the 'down' buffers could be tuned (positively or negatively) to be in sync with the 'up' buffers.

4.2.3 Individual SEC Buffer Limitations and Conclusions

The individual β sizing of SEC buffers can be used for clock edge timing refinements with minimal impact to the rest of the clock distribution network. Most significantly, adjustments of a single clock buffer's β can be performed to either speed-up or slow-down the critical clock edge of any branch of a clock distribution tree. This SEC technique will impair the non-critical clock edge to some extent, but so long as the single buffer adjustment is made within a reasonable level of asymmetry, the impact on the neglected clock edge will be very minor and inconsequential to the overall circuit's timing. This single buffer adjustment has been proposed for clock branch fine-tuning to synchronize all of the clock tree leaf clocks, as well as for delay buffer centering to allow for both positive and negative delay tuning with a single tunable buffer.

4.3 SEC Application Limitations and Summary

The novel SEC method improves timing performance by trading off the clock period between the critical and neglected edges, but this method would primarily be useful in ASIC circuit designs operating at less than the maximum allowable frequency of a given CMOS process. This limitation is because the initial clock phase grows in duration as it propagates through the clock network. However, the allowable frequency can be extended by shortening the initial clock signal (root clock) to a *minimally-sized pulse* that is allowed to grow in size (due to the neglected clock edge) as it propagates throughout the clock tree. Therefore, a minor design change to generate this initial pulse clock is recommended when using the SEC approach.

Also, if both the pull-up and pull-down transistors of a buffer are on simultaneously, short-circuit current will flow from Vdd to Ground. Therefore, the slow transition of the neglected edge in the SEC method must be handled carefully. The smaller devices of these buffers naturally reduce the level of short-circuit current; however, SEC buffers should be terminated by a final symmetric TC buffer to clean up the neglected clock edge before driving the FFs.

The first application of the SEC method involved converting an entire clock distribution H-tree from TC to SEC. By shifting the clock buffer resources already available from equal rise/fall buffers to alternating fast-rise/fast-fall buffers, significant reductions in latency, skew, and jitter can be achieved with virtually no penalty in clock buffer area or power consumption. HSPICE simulations show that these new INVr and INVf buffers can be simply dropped into an existing TC H-tree buffer distribution network to effectively improve clock timing performance by over 20% or to reduce power by 12%. This new SEC technique could also be utilized in other clock distribution networks (grid, hybrid) with similar results, although this usage was not demonstrated in the research work presented here.

Next, the SEC buffer was given tunability by designing a single-sided delay buffer (tsi-INVf) that allows for four linear tuning steps. The single-sided feature was necessary to only target the critical clock edge while allowing the neglected clock edge to pass without further degradation. These tunable buffers will be necessary in adaptive skew compensation schemes that are becoming increasingly popular at reducing intra-die PVT variations, post-silicon. (One such scheme is presented in the following chapter.)

Finally, a simple but powerful application of individual SEC buffers to fine tune a selected branch of any clock network (TC or SEC) was described. This single SEC buffer does degrade the non-critical clock edge slightly, but since it is not a series of SEC buffers (such as the full-chip SEC application), the neglected edge impairment does not accumulate significantly. One specialized implementation of this individual SEC β sizing has been shown to allow for both positive and negative clock edge tuning with respect to an unmodified, non-tunable clock branch. Furthermore, if the new SEC buffer is chosen in accordance to the critical clock edge then the clock signal can be sped-up (i.e., using an INVr buffer to speed-up the rising edge); but, if it is chosen in opposition to the critical clock edge then the clock signal can be slowed down (i.e., using an INVf buffer to slow down the rising edge).

Chapter 5 : Ring Tuning for Post-Silicon Clock Deskewing[18]

In Chapter 3, a complete clock distribution H-tree was constructed using SEC buffers to improve clock performance in a *pre-silicon* context. In this chapter, the new SEC clock tree is outfitted with an *active* clock de-skewing system to further improve clock performance, but this time in the context of *post-silicon* adjustments. Note that the SEC buffers are not a necessary component of this deskewing system; however, since SEC narrows the skew range over traditional clock buffers, using SEC allows for a finer resolution in the deskewing system, ultimately resulting in lower clock skews. Alternately, fewer tuning bits may be required for an SEC implementation compared to a traditional implementation with the same tuning resolution.

This active clock deskewing mode is intended to operate autonomously only at start-up to monitor and adjust for any initial clock skew which may be caused by intra-die PVT variations. It should be disabled during normal circuit operation to prevent any possible ongoing clock edge adjustments.[19] It consists of multiple deskew loops (or DLLs) linked together in a *ring* fashion to share clock phase information across the entire chip, hence the name *Ring Tuning*.

This chapter is organized as follows. First, the overall context of deskewing systems will be revisited. Next, the individual components and DLL design for the particular deskew loop utilized in this research will be presented. Then, the design considerations for creating a complete *ring tuning* system from multiple deskew loops will be covered, including the

[18] The research covered in this chapter will be published in: J. Mueller and R. Saleh, "Autonomous, multi-level ring tuning scheme for post-silicon active clock deskewing over intra-die variations," to be published in *IEEE TVLSI,14 pages,* 2010.

[19] Note that this system is capable of automatically correcting for clock skews that may arise during normal circuit operation, but such tuning corrections are bound to result in interim clock skew and jitter during any *re*-tuning process. Therefore, depending upon the given circuit, it may be preferable to disable the autonomous skew tuning and to simply tolerate any clock skew that may arise during circuit operation.

possibility of using multiple ring tuning connections in a multi-level, hierarchical fashion. Finally, the simulation results of a representative full-chip clock implementation of both single-level and multi-level ring tuning will be presented.

5.1 Overview of Active Clock Deskewing

An overview of post-silicon, autonomous clock tuning will now be presented. As mentioned in Section 2.3.2, active clock deskewing loops have been used individually, much like a single DLL, to tune two clocks and/or clock zones of a chip [49], but as chip complexities increase, more than two clock zones must be tuned. Therefore, multiple deskewing loops have been serially connected together in a cyclic fashion [57], but their cyclic design leaves them susceptible to stability concerns such as *mode-lock*, *mode-lock oscillation,* and *cyclic-feedback oscillation,* as previously described in Section 2.3.2.2. Consequently, acyclic designs have been proposed to address the stability issues [61], but their requirement for global control to handle the *out of range* problem, as described in Section 2.3.2.3, makes them complicated to implement.

In this research, a new type of cyclic architecture will be proposed. It will be called *Ring Tuning*, and it will be designed to address the stability issues of *mode-lock*, *mode-lock oscillation*, and *cyclic-feedback oscillation* traditionally inherent in cyclic systems, while avoiding the *out of range* problem inherent in acyclic systems. Multiple instances of a deskew loop will be interlinked by their phase detectors in a unidirectional cyclic or *ring* structure to collectively provide skew control simultaneously across the entire chip. The *mode-lock* problem will be addressed with a wider skew-range converging phase detector. The *mode-lock oscillation* and *cyclic-feedback oscillation* problems will both be addressed by slowing down the tuning adjustments of the multiple deskew loops; this 'dampening' of the feedback adjustments can be obtained either by stepping through the multiple loops one-at-a-time or by widening the skew threshold of the phase detector, as further explained in Section 5.4.1.2.

This *ring tuning* design provides a stable means of tuning multiple, distributed clock zones into synchronization without the need for any global control, making it easier to implement and operate than previous designs. However, stringing together multiple tuning zones will become less efficient as the number of tuning zones increases. The skew bound of a *single-level ring tuning* system will rise linearly with the number of tuning zones. Therefore, a *hierarchical ring tuning* system will be proposed to manage this skew bound metric for systems with more tuning zones.

First, the fundamental sub-system of this multiple-zone, active clock deskewing configuration is described outside the context of *Ring Tuning*. This sub-system is called a *deskew loop*, and it closely resembles a traditional delay-locked loop (DLL).

5.2 Three Components for the Deskew Loops

The novel *ring tuning* design proposed in this chapter uses multiple instances of deskew loops (or digital DLLs) to autonomously perform post-silicon active clock deskewing. Each of these deskew loops will operate individually like a traditional DLL, so their stand-alone design will now be discussed, outside the context of *ring tuning*. Later, in Section 5.4, they will be linked together into the *ring tuning* configuration for stable self-tuning of the entire clock system. The three main components (i.e., Up/Down Detector, Thermometer Code Generator, and Tunable Buffer) of these specially designed deskew loops are shown in the block diagram of Figure 5.1.

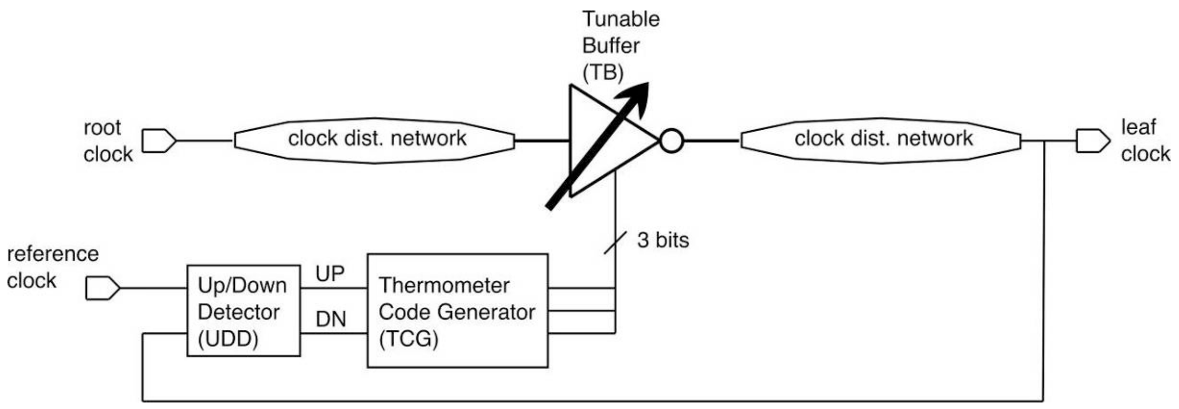


Figure 5.1 : DLL design for use in ring tuning

The Up/Down Detector (UDD) measures the skew between the *reference clock* and the *leaf clock* being tuned. Any skew of greater magnitude than UDD's threshold will trigger either an *UP* or *DOWN* signal to indicate that the bits of the Tunable Buffer (TB) should be adjusted. The *UP/DOWN* signal is converted via the Thermometer Code Generator (TCG) into a 3-bit thermometer code that is fed directly to the TB's three programmable bits. Since the purpose of this active clock deskewing system is to align two clock signals, much like a typical digital DLL, the sub-blocks of both are very similar and the two can be related block-by-block in terms of their functionality, as is done in the sections that follow.

5.2.1 Up/Down Detector (UDD)

UDD sub-block acts similarly to the phase detector of a typical DLL. The purpose of UDD is to send an *UP* signal to the remaining clock deskewing system when the delay of its TB should be adjusted *up*, and to send a *DOWN* signal when its TB should be adjusted *down,* thereby minimizing the skew between the two clock signals. There are two important design considerations regarding UDD: 1.) it must deliver accurate clock signals to UDD for comparison, and 2.) it must set the up and down thresholds of UDD precisely.

First, the UDD can only be as accurate as its input signals, but unlike a typical DLL, the two clocks being compared here are not available at the same location on the chip. Therefore, 'sense' lines from the two clocks (labeled *clk L* and *clk R* in Figure 5.2) must be carefully routed to the UDD in order to preserve their true timing edges relative to each other. As such, the UDD should be placed directly in the center of the sources of these two clock signals being measured, and proven buffer sizing and buffer insertion techniques should be used to create a mirrored signal delivery path with sufficient accuracy. As illustrated in Figure 5.2, the initial inverter is minimally sized to reduce the loading on the clock signal, but sequentially larger inverters are added to boost the drive strength adequately for the metal lines' loadings, indicated by the rectangles. Furthermore, all of the inverters used in this top schematic of Figure 5.2 are Single-Edge Clocking (SEC) buffers, since the SEC method results in reduced latency and skew, therefore higher accuracy, as explained in the preceding chapters and in [72].

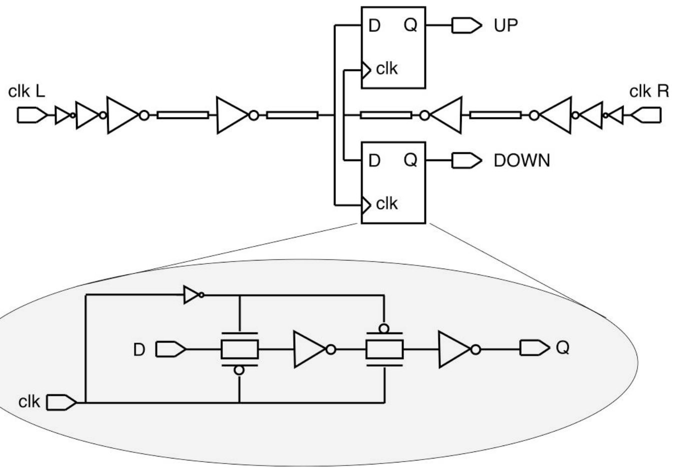


Figure 5.2 : UDD schematic

Second, the UDD must be both *accurate* at detecting skews greater than a designed threshold and *symmetrical* about the zero skew axis, such that the negative skew threshold is equal but opposite to the positive skew threshold. The accuracy of this UDD is obtained by using the setup time of a simple dynamic D-flip-flop (DFF) as UDD's threshold. By altering the size of the input transmission-gate (shown in the lower portion of Figure 5.2 on the left), the setup time of this DFF can be set to the desired value. Threshold adjustments in the range of picoseconds can be obtained in this fashion. Note that this very simple UDD design is susceptible to meta-stability problems, but it is used in this thesis for its simplicity and ease of simulation. For a production chip, a more robust phase comparator design such as in [49] or [73] should be used, and care must be taken to ensure that any additional latency of the phase comparator does not compromise the stability of the adaptation scheme.

The symmetry about zero skew is easily obtained by using one of these DFFs (*clk L* connected to the *D* input and *clk R* to the *clk* input) to check for negative skews and to trigger the *UP* signal whenever it is greater than the threshold, and using an exact duplicate but oppositely connected DFF (*clk L* connected to the *clk* input and *clk R* to the *D* input) to check for positive skews and to trigger the *DOWN* signal whenever it is greater than the threshold.

UDD's signals are important in guaranteeing DLL locking and system stability. This can be best illustrated by the graph of Figure 5.3. The *UP* and *DOWN* signals are both logical, DC signals, either logical high or logical low. When the measured skew (as plotted on the x-axis of Figure 5.3) is more than the negative threshold, S_{th-}, and less than the positive threshold, S_{th+}, then both *UP* and *DOWN* will be *low* and the system is considered *'locked'*. As with a typical DLL, when this system is in the locked state it will be in the *hold* mode, and no tuning adjustments will be made. If, however, the measured skew is greater in magnitude then either of the two thresholds, then either an *UP* or *DOWN* signal will be *high* in order to appropriately adjust the clock. As seen in the graph of Figure 5.3, the *UP* and *DOWN* signals can *never both* be high (or active) *simultaneously*, thereby eliminating any possible control contention.

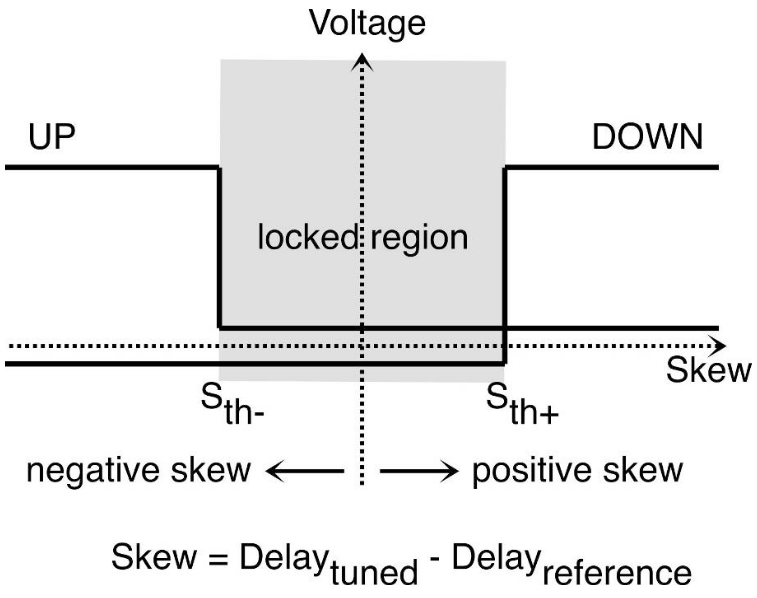


Figure 5.3 : UDD locking region

The *UP* and *DOWN* signals have been designed as DC (logical) values for two important reasons. First, since UDD is located at the middle of the two clocks being synchronized, some signal(s) will have to be routed to the TB in order to deliver UDD's determined tuning adjustments. Routing a *pulse-width signal* or an *analog voltage signal* (both commonly found in PLL or DLL designs) across a relatively large portion of the die would be difficult, whereas routing two DC logical signal lines is relatively easy. Second, having the output of UDD as DC logic signals allows for logical operations to be used on these control signals (if desired). Two or more UDDs could be logically combined to create a cumulative, more representative *UP/DOWN* control signal for its TB. For example, two UDDs could be connected across two pairs of clocks and a simple AND logic operation could be used to trigger the *UP/DOWN* signal for its TB *only when both* of the two UDDs *agree* on their measured skews, as depicted in Figure 5.4. This averaging or filtering of the signals reduces PVT variability impacts on the feedback and thereby helps to stabilize the overall system's operation. Once the *UP/DOWN* decision has been made by UDD(s), with or without logical combinations, those signals must be converted into bits for the loop system's tuning adjustments.

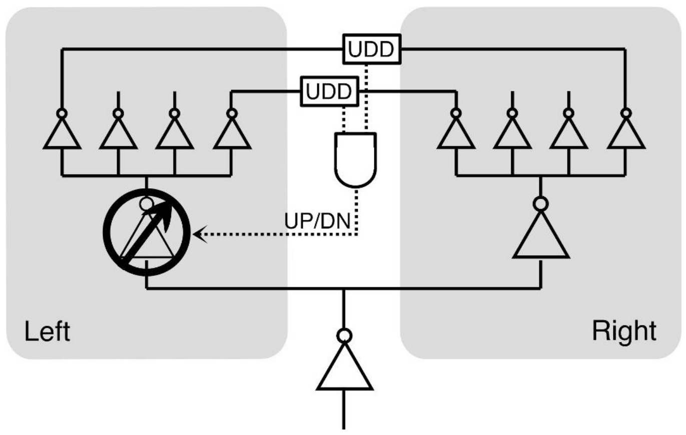


Figure 5.4 : Logical Operation on Multiple UDDs

5.2.2 Thermometer Code Generator (TCG)

TCG performs the same basic operation as the Counter/Digital Loop Filter in a typical DLL. TCG takes the control signals from UDD and converts them into a control word to adjust TB's programmable delay (much like a DLL's Digitally Controlled Delay Line). Additionally, the conversion time of these signals to a 3-bit thermometer code by TCG acts as a low-pass filter in the overall active deskewing loop, effectively providing stability to its operation. The basic design of this TCG has been taken from [74] and a simplistic representation of its operation is shown in Figure 5.5.

The general principle behind this [74] design is to connect two chains of buffers such that they are competing with each other over the *high* or *low* state of the three thermometer bits (*tbit3*, *tbit2*, and *tbit1*). As shown in Figure 5.5, the upper chain of buffers is fed by a *low* state and is triggered by the DOWN signal, while the lower chain of buffers is fed by a *high* state and is triggered by the UP signal. In this simplified example, *tbit3* is shown to be at a logic '0', *tbit1* is at a logic '1', and *tbit2* is somewhere in between a '0' and '1', as represented by the '0~1'. Note that buffers *d3* and *u3* both agree that *tbit3* should be '0', since the input to buffer *d3* is *low* and the input to buffer *u3* is *not high*. Likewise, buffers *u1* and *d1* agree that *tbit1* be '1', since *u1*'s input is *high* and *d1*'s input is *not low*. Therefore, the *tbit3* and *tbit1* lines will maintain their logic levels due to the charge held on the small resistor-capacitors shown in the figure. The state of *tbit2*, however, is in direct contention from the *d2* and *u2* buffers -- *d2* is trying to pull *tbit2 low*, while *u2* is trying to pull it *high*.

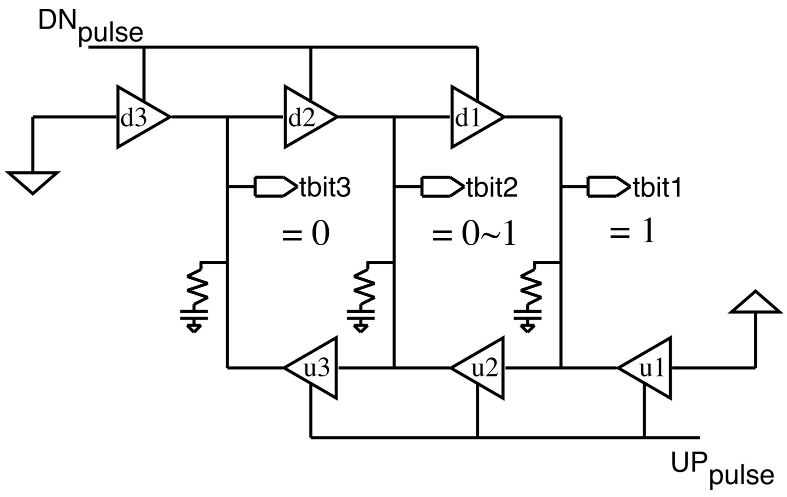


Figure 5.5 : TCG simplified functional schematic [74]

If more DOWN pulses were sent than UP pulses, then *d2* would eventually win its battle to pull *tbit2 low*. Conversely, more UP pulses would pull *tbit2 high*. In essence, this contended bit is acting like a common charge pump controlled by UP and DOWN pulses from a common phase detector. Once the state of *tbit2* has been settled, then the contended bit will shift, to *tbit1* if the DOWN signal wins or to *tbit3* if the UP signal wins, and *tbit2* will be held stable by its resistor-capacitor. This bit under contention can shift left and right through the buffer chains depending upon the UP and DOWN signals, and it therefore produces a true thermometer code.

In normal operation if only DOWN signals were sent, the three bits would progressively transition to *low* from *tbit3* to *tbit2* to *tbit1* (left-to-right). Likewise, if only UP signals were sent, the bits would transition to *high* in reverse order from *tbit1* to *tbit2* to *tbit3* (right-to-left). Thus, the three bits always maintain the thermometer code word structure (zeroes on the left and ones on the right). A transition point that is not fully *high* or not fully *low*, must exist somewhere in between the fixed *low* input on the left and the fixed *high* input on the right. The original paper [74] details the simple design techniques for resolving this '0~1' bit.

Another useful feature (not depicted in this figure) of this TCG design is the ability to lock-in the value of bits when their state is known, thereby removing the ongoing contention from as many bits as possible. As per the definition of a thermometer code, all bits to the left of the transition point must be *low* (or zero), and all bits to the right of it must be *high* (or one). If *tbit3* is *high*, then *tbit2* is locked *high*, and since *tbit2* is *high*, then *tbit1* is also locked *high*. On the other hand, if *tbit1* is *low*, then *tbit2* is locked *low*, and since *tbit2* is *low*, then *tbit3* is also locked *low*. This lock-in feature also helps to improve the stability of the TCG and to reduce its power consumption.

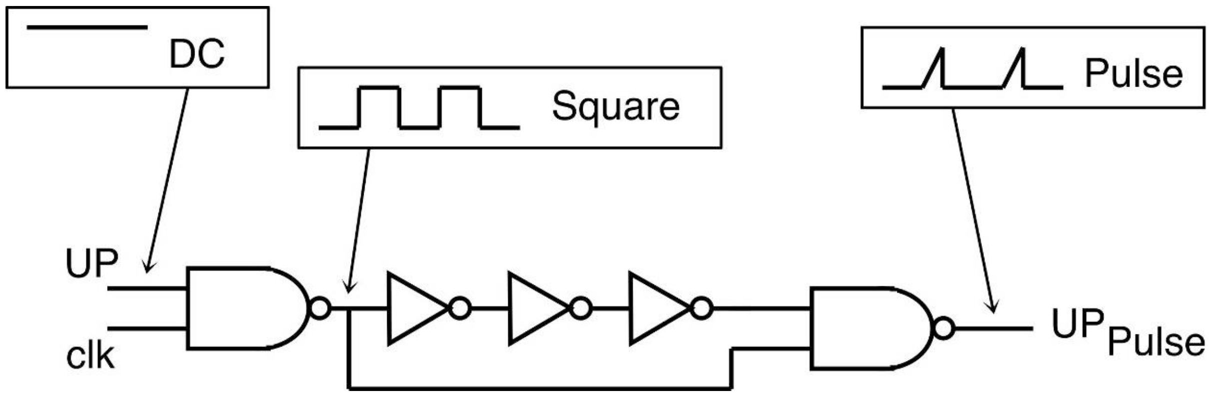


Figure 5.6 : Pulse generator for TCG

The primary concern in implementing this TCG design is to match the two competing buffer chains. Consequently, small-sized transistors can be used to minimize the area and power consumption. In fact, only 78 small transistors are used in the entire TCG design, so it can be easily placed right next to the TB with just one simple, additional change made to UDD's signals. The *UP/DOWN* DC signals must be converted into pulses with a simple clocked NAND pulse generator as shown in Figure 5.6. The pulse width is adjusted by the sizing of the delay inverters to properly control the TCG. A pulse that is too wide will overdrive the up (or down) buffer chain, and TCG will transition through the bits too quickly. Conversely, a pulse that is too narrow will under-drive the chain, and it will fail to transition the bits at all. The pulses for this implementation have been sized such that they step TCG bits up or down a single bit in about 2 clock cycles.

It is important to note that the UDD and TCG have been designed so that *tuning adjustments are carried out only one tuning step at a time to ensure system stability*. Even if a large skew of two (or more) TB tuning steps exists between the reference and tuned clocks, UDD+TCG would only adjust its TB one step, reevaluate the clock skew, adjust TB another step, reevaluate, and so on.

5.2.3 Tunable Buffer (TB)

The TB acts like the Digitally-Controlled Delay line of a typical DLL. It is simply a clock buffer, or inverter, that has some means to provide for an adjustable delay. The single-sided starved inverter configuration shown in Figure 4.2 and from [75] has been chosen for this implementation. Since a thermometer code is the most convenient way to adjust the delay in these active deskewing loops, the footers of these TBs have been sized to allow for four linear steps of a 3-bit thermometer-code selectable delay.

As PVT varies for both inter- and intra-die cases, it is important that the tunable delay buffer range continues to span the desired tuning range (50ps for this case), and also that the step size, *ss*, (defined as the distance between any two adjacent tuning steps) remains relatively constant. As explained later, this step size will be fundamental in designing the UDD's threshold, S_{th}, for system stability. Data has shown this *ss* value (nominally 16.7ps) to vary from ~13ps to ~20ps over PVT variations for this particular 50ps TB [75], which is sufficient for its use in this deskewing system.

5.3 Deskew Loop Design for Stable Operation

Now that the three basic components of the special deskew loop have been described, the deskew loop itself must be properly designed to ensure stable operation. Once again, the stable locking discussion of this section address the stand-alone operation of the deskew loop — outside the context of *ring tuning*. In Section 5.4, the special considerations due to the cyclic, *ring tuning* connection of multiple instances of these deskew loops will be presented.

Since these deskew loops will be widely distributed across the entire chip, they will all be subjected to various intra-die PVT variations. Furthermore, the multiple deskew loops used in ring tuning are all expected to have similar operating functionality. Therefore, these distributed deskew loops must be more robustly designed than a traditional, stand-alone DLL that operates individually and is locally positioned. Two elements of the deskew loop stability will be discussed here; the first one dealing with the loop delay of the deskew loop, and the second one dealing with the resolutions of the measurement (UDD) and adjustment (TB) blocks.

5.3.1 Deskew Loop Delay Analysis

The first design consideration for deskew loop stability involves the complete time delay of the DLL feedback loop. The relevant components of Figure 5.1 have been redrawn in Figure 5.7 to illustrate the DLL feedback loop. Derivation of complex Laplace Transfer functions for each component, followed by open-loop and closed-loop system analysis, is outside the scope of this research. However, the main consideration for this DLL's loop stability can be summarized as follows.

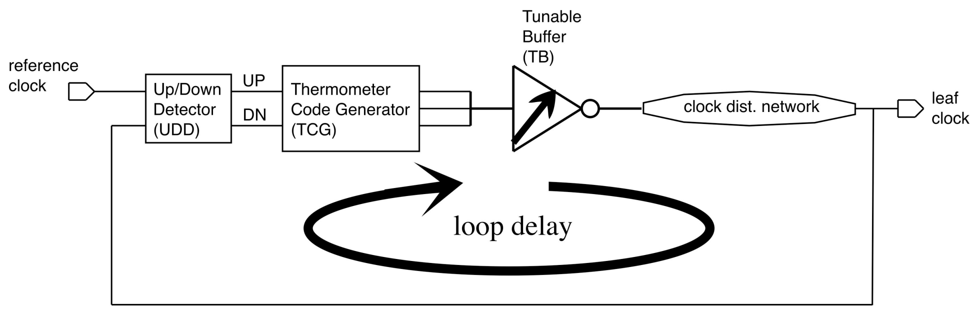


Figure 5.7 : Loop delay consideration for stability

Unstable operation may occur if adjustments to the TB are made based upon 'old' UDD measurement data. For example, if the UDD sends an UP signal, there will exist a finite period of time, called the loop delay, for that signal to propagate to the TCG, adjust the TCG up one step, propagate to the TB, adjust the TB up one step, propagate through the remaining clock distribution network, and finally be measured back at the UDD. During that loop delay time, the UDD may still be measuring the old clock skew data and may be sending another, superfluous, UP signal. *For stability, design of the deskew loop must ensure that the tuning effect of each TCG adjustment be measured before another tuning step is made.*

The particular deskew loop used in this research utilizes the TCG as a low-pass filter to address this loop delay concern. The combined TB, clock distribution network, plus UDD delay is designed to be less than a clock cycle, while the TCG step adjustment delay is designed to be two clock cycles (as described in Section 5.2.2). Therefore, accurate clock skew information will always be passed from the UDD to TCG faster than the TCG could act upon it. Hence, TB adjustments will not be made based upon 'old' measurement data.

Note that this analysis considers only the time-varying effects of *TB adjustments* on the measured clock skew. Any time-varying effects of *PVT variations* on clock skew are assumed to be much slower (of lower frequency) than the loop delay; therefore, the time varying effect of *PVT variations* can be safely ignored in this analysis.

5.3.2 Measurement and Adjustment Resolutions

Deskew loop stability may not exist if the adjustment resolution (the TB's step-size) is greater than the measurement resolution (the UDD's threshold). The skew threshold, S_{th}, of the UDD (as shown in Figure 5.3) is the design parameter that will ensure stable locking of the deskew loop over PVT variations.

Figure 5.8 shows two cases of UDD threshold values. On the left side, S_{th} is set to the theoretical minimum value of half of TB's step size*, (S_{th} = 0.5ss)*. On the right side, S_{th} is increased to 1.5 times that theoretical minimum, *(S_{th} = 0.75ss)*. Note that the 1.5X increase (for nominal models) is chosen here to encompass the expected PVT variations of this example – a design with less variation could employ a smaller increase while a design with more variation could employ a larger one. The graphs at the top of Figure 5.8 show the *UP/DOWN* signals plotted against the measured clock skew (similar to Figure 5.3). Below those graphs are three line diagrams for each showing the tunable bit transitions of TB as the active deskewing loop seeks to *lock* (highlighted by the shaded region in between -S_{th} and +S_{th}). As indicated by the arrows on these line diagrams, if the measured skew is outside of the locked region, then TB's tuning bits will be incremented or decremented *one step* by UDD+TCG to another (better) value of lesser skew. Since TBs used in this example are 3-bit linear, there are four possible delay tuning states (*000*, *001*, *011*, and *111*).

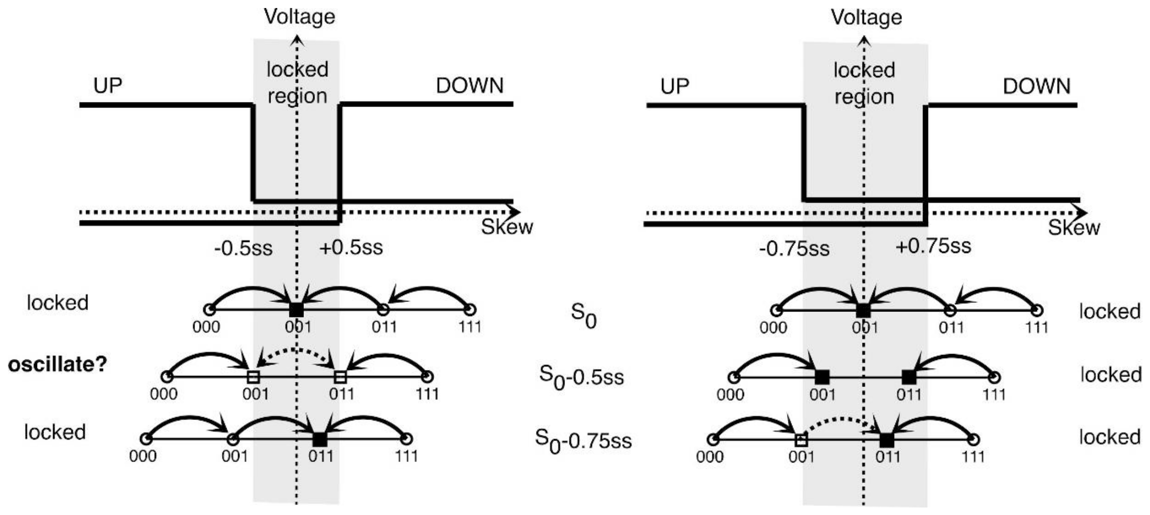


Figure 5.8 : Stable locking over PVT variations

For the first line diagram of Figure 5.8, the initial measured clock skew, S_0, happens to align with one of the tunable steps (*001*) right on the zero skew axis. In this case, both the left and right graphs operate similarly, and both obtain *lock* as follows:

- TB at *000*: UDD measures a negative skew greater than the threshold, so the *UP* signal is sent to TCG and TB is moved to *001*.
- TB at *001*: UDD measures no skew, so the system is **locked**.
- TB at *011*: UDD measures a positive skew greater than the threshold, so the *DOWN* signal is sent to TCG and TB is moved to *001*.
- TB at *111*: UDD measures a positive skew, sends *DOWN* signal, and TB is moved to *011*.

If however, the initial clock skew happens to be shifted to *(S_0-0.5ss)*, as depicted in the second line diagram, then the graph on the left will have **two** *unstable locking* states. These unstable states result when TB delay tuning steps (*001* and *011*, in this case) align right on UDD thresholds. System oscillation for the left side design may result as follows:

- TB at *000*: UDD measures a negative skew and TB is moved to *001*.
- TB at *001*: UDD *may or may not* measure negative skew, so the system may stay locked **or** *TB may be moved to 011*.
- TB at *011*: UDD *may or may not* measure positive skew, so the system may stay locked **or** *TB may be moved to 001*.
- TB at *111*: UDD measures a positive skew and TB is moved to *011*.

The uncertainty in these UDD threshold comparisons comes from *all possible* design and/or PVT variations that could potentially affect the measured clock signals, UDD thresholds, and TB step sizes. There are very many possibilities for even small variations in these comparisons, so some uncertainty must be expected. The design on the left side (S_{th}*=0.5ss*) does not contain any design margin for these variations, and erratic state jumping and/or full oscillation (between *001* and *011*) will almost certainly result. However, this problem can be easily resolved by designing S_{th}*=0.75ss* as shown in the graph on the right side of Figure 5.8. For this same initial skew case, the design on the right side does not have any *unstable locking* states, so it will lock properly and stay locked. In fact, as seen in the diagram, it will have **two** *valid locking* states, so TB may end up at either *001* or *011* – both within the chosen design window, and both acceptable for stable operation.

If the initial clock skew is further shifted to *(*S_0*-0.75ss)*, as depicted in the third line diagram of Figure 5.8, then the graph on the right will have **one** *unstable locking* state (*001* in this case). However, one unstable state is not enough to initiate oscillation, so if the system happens to jump out of that one unstable state (*001*), it will necessarily land in a stable locking state (*011*) that has sufficient design margin safely inside the opposite threshold. Therefore, for this third line diagram, both the left and right side will ultimately obtain stable *lock*.

Note that this technique of widening the thresholds to stabilize the DLL locking could result in *non-optimum* TB bit choices. As shown in the third line diagram on the right side of Figure 5.8, this stable system could very possibly maintain lock with the TB set to *001*, even though a setting of *011* would result in a smaller skew. (This system is not 'smart' enough to find that better state.) It will guarantee that the measured skew is within the chosen design threshold window before it locks, but it will not guarantee that the measured skew is at the absolute minimum. *Some system resolution is being traded off in order to* *ensure stable DLL locking over PVT variations* -- but clock stability is critical, and there are other ways to improve resolution, so this tradeoff is a very good one.

5.4 Multiple Deskew Loop Connection for Ring Tuning

Now that the design of the deskew loop has been presented, the method of connecting multiple deskew loops together to create *ring tuning* will be discussed. In Section 5.4.1, the motivation for moving from an *acyclic* connection to a *cyclic* connection (*ring tuning*), and ultimately to a *hierarchical cyclic* connection, will be presented. In Section 5.4.2, an example of single-level *ring tuning* for the generic H-tree will be described, and in Section 5.4.3 a second level of *ring tuning* will be added to the generic H-tree to create a complete hierarchical (multi-level) *ring tuning* system.

5.4.1 The Case for Ring Tuning over Acyclic Tuning

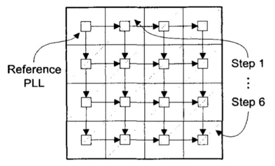
As previously described in Section 2.3.2.2, the *cyclic* connection for multiple zone deskewing has been abandoned by modern practitioners [61] in favor of an *acyclic* connection. The acyclic connection avoids the inherent cyclic stability concerns of *mode-lock*, *mock-lock oscillation*, and *cyclic-feedback oscillation*, as defined in Section 2.3.2.2. However, this acyclic connection creates a new problem of *out of range*, as defined in Section 2.3.2.3, which requires a global system controller. Here, in order to eliminate the need for any global controller, a new variety of cyclic connection is being proposed, called *Ring Tuning*.

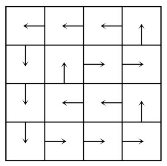
First, this cyclic *ring tuning* architecture will be compared to the acyclic design. Next, the design techniques for ensuring *ring tuning* stability (i.e., avoiding *mode-lock*, *mode-lock oscillation*, and *cyclic-feedback oscillation*) will be presented. Note that these three stability concerns are due to the cyclic connection of *ring tuning*, and they are in addition to the individual deskew loop stability issues discussed in Section 5.3. Finally, the metric of skew bound will be presented, which will lead to the proposed use of two, hierarchical *ring tuning* systems.

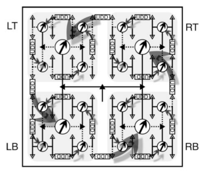
5.4.1.1 Avoiding the Global Controller

This section provides a comparison of two versions of the newly-proposed *Ring Tuning* method with the previously published acyclic tuning method from [61]. Table 5-1 contains the comparison data for the three methods. The first column lists the previous acyclic method described in Section 2.3.2.3. The second column lists a *single-level* implementation of the newly-proposed *ring tuning* method called "RT-3", since it is *ring tuning* applied only at clock level 3 on a total of 16 tuning zones. (The clock levels are shown in Figure 3.3, and the tuning zone selection process is explained in Section 5.4.2.1) The third column lists a hierarchical implementation of the newly proposed ring tuning method called "QRT-2&3", since it is two-level *quad-ring tuning* (or four zone ring tuning) at clock levels 2 and 3.

Table 5-1 : Comparison of Acyclic Method to Two Ring Tuning Methods







	Acyclic [61]	**RT-3**	**QRT-2&3**
# TB	16	16	4 + 16 = 20
# PD	24	16	4 + 16 = 20
separation	6	8	2' + 2 = 4'
mode-lock	no	fixed w/ s.w.a.	fixed w/ s.w.a.
out of range	fixed w/ r.s.a.	fixed w/ n/2 overlap	fixed w/ n/2 overlap
skew bound	0.5ss x 6 **= 3ss**	0.5ss x 8 = 4ss	1ss x 2' + 0.5ss x 2 **= 3ss**

s.w.a. = spinning wheel algorithm r.s.a. = range sharing algorithm

As the table indicates, each method uses roughly the same number of tunable buffers (TBs) and phase detectors (PDs). Likewise, each method has techniques to eliminate the oscillation problems associated with *mode-lock* and the *out of range* problem resulting from a finite number of tuning steps. However, it is the complexity of these solutions that may motivate the *ring tuning* method to be chosen over the acyclic method. First, the *spinning wheel algorithm (s.w.a.)* and the *range sharing algorithm (r.s.a.)* which are introduced in [61] must be summarized here:

Spinning wheel algorithm (s.w.a.) - The s.w.a. is relatively simple algorithm for passing phase information from one clock zone to its neighboring clock zone. In [61] it is proposed for the acyclic system shown in the first column of Table 5-1, but its general operation could be applied to a cyclic system, such as ring tuning, as well. Its sole purpose is to transition through all the clock zone boundaries one-at-a-time. For the acyclic design shown here it would begin at the reference PLL in the upper left and transition over-and-down in the direction of the arrows to the last tuning zone in the lower right, before starting over again in the upper left. For the ring tuning designs it would begin at one of the clock zones and then travel around the ring in the direction of the arrows completing the ring and then starting over. The s.w.a. adjusts each clock zone individually, while holding all other clock zones inactive, before proceeding to the next clock zone, thereby stepping through the progression of clock zones in a controlled order one-at-a-time.

Range sharing algorithm (r.s.a.) - The r.s.a. is an extension of the s.w.a. created specifically to address the out of range problem of the acyclic systems. The r.s.a. progresses one-at-a-time like the s.w.a., but it also checks for tuning buffers that have reached their tuning limit. If such an 'out of range' tuning buffer is found, then the r.s.a. will immediately return to the reference PLL zone and 'borrow' some of its tuning range. At that point, the r.s.a. will start over, progressing through the clock zones in order as before. Since the cyclic designs are naturally immune to the out of range problem, they would not require the r.s.a. For the comparison of this section, it is important to note that the *range sharing algorithm* is considerably more difficult to implement than the *spinning wheel algorithm*.

Returning to Table 5-1, it will be shown how both *ring tuning* methods (RT-3 and QRT-2&3) can utilize the s.w.a. to avoid the stability problems of *mode-lock oscillation* and *cyclic-feedback oscillation*. Furthermore, both *ring tuning* methods will be immune from the *out of range* problem provided that their tuning zones are allowed to overlap by n/2 tuning steps, where n=# of tuning zones. This relatively small overlap of tuning zones allows for a tuning zone which has reached its limit to still participate in the ring tuning process with its neighboring tuning zones. Since this n/2 overlap is a very minor design consideration for the TCG and TB blocks, it will not elaborated on further. Again, these two design techniques are easily implemented. But, the acyclic method requires the r.s.a. to avoid its *out of range* problem, and this r.s.a. is more difficult to implement. *Therefore, to eliminate the need for this complex r.s.a. and its accompanying global controller, the ring tuning method is pursued here.*

5.4.1.2 Avoiding Cyclic Instabilities

Now that a cyclic method (*ring tuning*) has been chosen to avoid the complex global controller required by the acyclic *out of range* problem, the inherent cyclic instabilities defined in Section 2.3.2.2, namely *mode-lock*, *mode-lock oscillation*, and *cyclic-feedback oscillation* must be addressed. Note that these three stability concerns are collectively listed as Mode-lock in Table 5-2.

First, the classic problem of *mode-lock* is easily addressed by the design of the phase discriminator (referred to as an Up/Down Detector from Section 5.2.1). As explained in Section 2.3.2.2, *mode-lock* occurs when the deskew loop phase detector incorrectly locks 90˚ out of phase with its neighbor. If all four quadrants of a multi-tuning system lock 90˚ out of phase, then the system will remain incorrectly locked. The solution to *mode-lock* is to design a phase detector that converges to the correct phase even when off by 90˚ or more. As seen in Figure 5.3, the simple, digital phase discriminator (UDD) will converge to the correct clock phase even when off by 359˚, so *mode-lock* will not be a concern with *ring tuning*.

Next, the sustained flip-flopping problem of *mode-lock oscillation* must be addressed. This back-and-forth oscillation is caused by the possibility of the deskew loop's *reference clock* being adjusted at the same time that its *tunable clock* is being adjusted. As mentioned previously, the *tuned clock* of one zone becomes the *reference clock* of its neighboring zone. There are two possible design fixes for *mode-lock oscillation*. 1.) The spinning wheel algorithm could be implemented for the *ring tuning*, thereby ensuring that no zone's tuned clock and reference clock can be concurrently adjusted. This fix solves *mode-lock oscillation* without impacting the overall tuning resolution, but it requires the s.w.a. controller. 2.) The skew threshold of the UDD could be increased by 2X to ensure that no clock adjustments are undertaken unless the measured skew is twice the tuning step size. Therefore, even if the tuned clock and reference clock both adjust concurrently, no back-and-forth oscillation can occur. This fix solves *mode-lock oscillation* without the need for any controller, but it worsens the overall tuning resolution by 2X.

Finally, the traveling wave problem of *cyclic-feedback oscillation* is addressed with the same design fixes for *mode-lock oscillation*. The *cyclic-feedback oscillation* may occur if an appropriately positioned 'wave' of clock phase is allowed to travel through the cyclic, ring connection without attenuation. This 'attenuation' of the traveling wave can be obtained by stepping through the *ring tuning* one step at a time with the s.w.a. Alternatively, it can be obtained by widening the UDD skew threshold to twice the tuning step size, resulting in a skew of two steps being adjusted with only one step. Either the s.w.a. or the UDD threshold increase will solve the *cyclic-feedback oscillation* problem, with the same trade-offs discussed above.

One additional aspect of ring tuning stability should be considered. In Section 5.3.1 the deskew loop delay was analyzed in order to ensure independent deskew loop stability. Now that multiple deskew loops will be joined in a ring, the delay (or latency) around the complete ring could impact overall system stability. Hence, attenuation of the adjustments around the cyclic ring is critical. One way of attenuating is to delay the adjustment decisions until a *complete ring delay* has been made; therefore, it will not over-steer the skew adjustment. Another way, described earlier as the UDD threshold widening, is to attenuate the skew adjustments with a gain less than one, so that any measured input phase must be greater than the output phase adjustment. This latter approach may take a while to reach convergence. This has been a rather informal stability analysis, and a more formal stability analysis is proposed for future work.

5.4.1.3 Improving Skew Bound Hierarchically

The primary metric for the comparison of Table 5-1 is the "skew bound". The skew bound is chosen as a measure of the worst-case skew allowed by a deskewing system. For these architectures it can be calculated as the theoretical resolution of the phase detectors multiplied by the maximum clock tuning zone separation. To create a fair comparison, it is assumed that each of the three methods employs phase detectors with the theoretical minimum resolution of *0.5ss*, where *ss* is the step size of the tunable buffers.

The acyclic method requires six steps to propagate from the top left clock reference to the bottom right clock tuning zone; therefore, its separation is 6 steps and its skew bound is *3ss*, as shown in the table. The ring tuning methods have a clock tuning zone separation equal to half of the number of tuning zones of the ring; therefore, RT-3's separation is 8 steps, and its skew bound is *4ss*. This skew bound is slightly worse than the acyclic method, but since it is obtained with fewer phase detectors and a simpler control algorithm (s.w.a. for RT-3 instead of r.s.a. for acyclic), this tradeoff may be acceptable for a given circuit design. However, if the number of clock zones were to increase above the 16 used here, the skew bound of the single-level ring tuning method would increase at a much faster rate than that of the acyclic method, so an improvement to the single-level ring tuning is needed.

One such improvement, compatible with the clock tree distributions used here, is *hierarchical ring tuning*. A second level of ring tuning has been added at clock level 2, and the single ring tuning of 16 zones (RT-3) has been divided into four ring tunings of four zones each, in order to create the QRT-2&3 method. The multi-level ring tuning of QRT-2&3 is shown to match the skew bound performance of the acyclic method. Calculating the clock zone separation and phase detector resolution of the multi-level system is not as straightforward as the other two since QRT-2&3 uses different clock zone sizes and different tuning step sizes for each level, but sizing adjustments relative to clock level 3 can be made. As shown in the table, QRT-2&3 can match the *3ss* skew bound of the acyclic method. The QRT-2&3 method uses four more tunable buffers and four less phase detectors than the acyclic method, but more importantly, the QRT-2&3 only requires the simpler s.w.a. control, while the acyclic method requires the more complex r.s.a. control.

The design details for single-level ring tuning at clock level 2 (QRT-2) and multi-level, hierarchical ring tuning (QRT-2&3) will now be presented in detail, followed by their simulation results.

5.4.2 Single-level Ring Tuning for H-tree

The specific implementation of this *ring tuning* structure is the generic 4-level H-tree clock distribution network with 64 clock leaves on a typical square chip used in previous chapters (see Figure 3.3). However, the principles of ring tuning can be used for autonomous clock deskewing with virtually any clock distribution network -- H-tree or non-H-Tree, square chip or not. *The primary goal of this hypothetical example is to demonstrate how to distribute the root clock signal to all of the 64 leaf clocks with a reduced clock skew.*

5.4.2.1 Independent Tuning Zone Selection

First, the concept of independent tuning zones must be considered. Tuning zones are quite simply created by the insertion of a tunable buffer (TB) within a clock branch. From Figure 5.1, it is clear that the portion of the clock distribution network on the 'upstream', or source, side (clock root side) of the TB is unaffected by its delay adjustments, whereas the entire clock network on the 'downstream', or terminal, side (clock leaf side) of the TB is affected by its delay settings. Therefore, it can be stated that *tuning zones* are created by the locations of TBs, and that these tuning zones continue from their point in the clock network all the way to the end clock leaves. The four diagrams of Figure 5.9 illustrate four possible tuning zone sets for the chosen H-tree, each created by the placement of a TB at a different clock level.

The number of tuning zones for a ring tuning system should now be considered. Since *ring tuning* strings all the zones of a chosen clock level together in a loop, a ring tuning implementation of Figure 5.9 (a) would be called RT-1a and would only link two tuning zones. A ring tuning implementation of Figure 5.9 (d) would be called RT-3 (also shown in the middle column of Table 5-1) and would link together 16 tuning zones. Because the tuning zone separation grows linearly with the number of tuning zones, the skew bound grows as well, so large rings of many tuning zones should be avoided for better skew control.

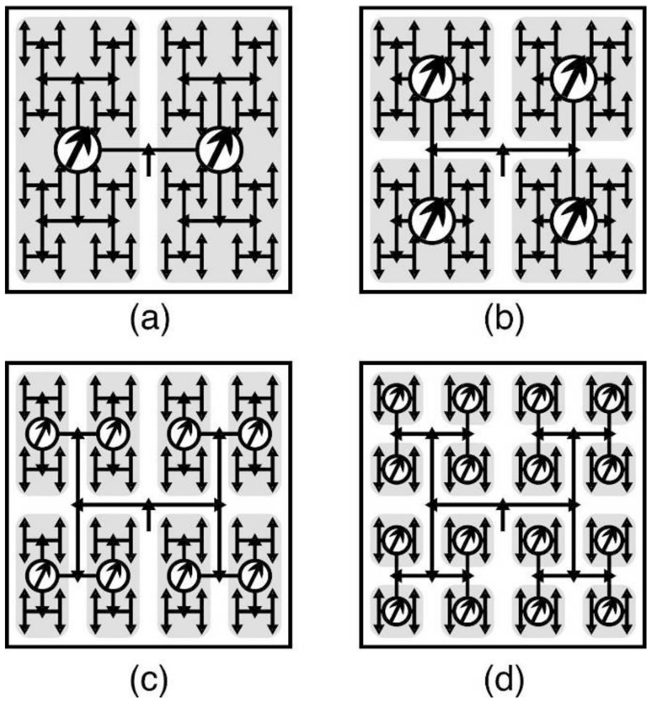


Figure 5.9 : Four Possible Tuning Zones TBs at (a) level-1a, (b) level-2, (c) 2a, (d) 3

The size of each tuning zone can be better represented by redrawing the H-tree distribution as a hierarchical tree structure as in Figure 5.10, with the root clock, or source, at the bottom (level-1) branching up to the leaf clocks, or terminals, at the top (level-4). If the TB were to be inserted at level-2 of this clock tree (as shown on the left side of the diagram), then its *tuning zone* would encompass all of the buffers (23 in total) and wires and chip area from that point upwards (as shaded in the diagram); similarly, if TB was inserted at level-3 (as shown on the right side), then its *tuning zone* would be considerably smaller (only 5 buffers plus wires).

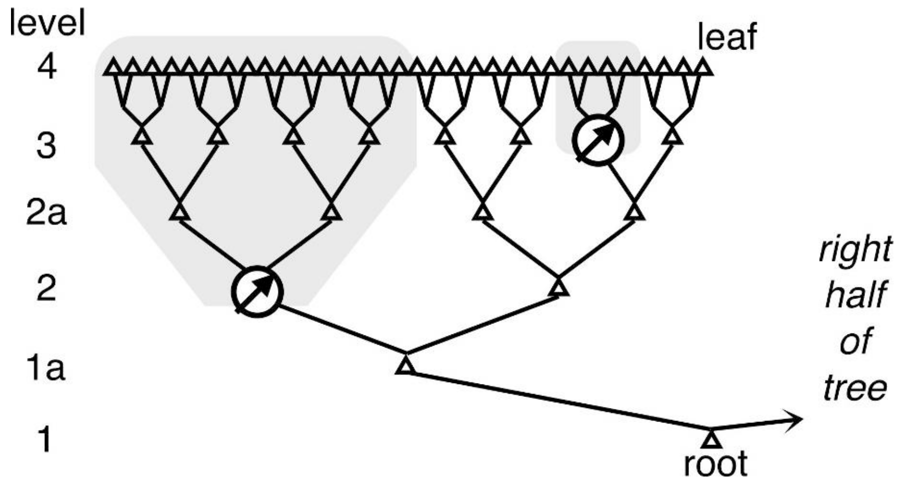


Figure 5.10 : H-tree hierarchy (left half only)

Therefore, the selection of tuning zones must balance both the desire for *simpler control* of *fewer, large* tuning zones with the desire for *higher resolution* of *more, small* tuning zones. Considering the nature of PVT variation across this hypothetical square chip along with the geometries of the H-tree clock buffer/wire network, *four tuning zones* (or quadrants) as defined by TB placement at clock level-2, Figure 5.9 (b), has been chosen for this example. Therefore, this particular implementation is called *Quad Ring Tuning (QRT)*.

5.4.2.2 Ring Tuning Connection for Four Zones

Once the tuning zones have been chosen and TBs have been placed for this single QRT case at clock level-2, as in Figure 5.9 (b), then the control of the four TBs must be established. Inclusion of four TBs allows for adjustability of four independent tuning zones, but each TB's tuning bits must be controlled somehow. This control stage is where the *Ring Tuning* technique is employed. As illustrated in Figure 5.1, one UDD will be used to control each TB (via TCG). The location of UDD should be chosen to minimize routing and maximize skew measurement accuracy.

Consider the two examples of UDD placement depicted in Figure 5.11. The four quadrants (or tuning zones) of the H-tree are designated LT, LB, RB, and RT for left-top, left-bottom, right-bottom, and right-top, respectively. On the **lower** half of that chip, clock signals from TBs at clock level-2 (LB and RB) are both routed to UDD near their midpoint. This implementation has two major drawbacks. 1.) Large metal wires must be *added, routed, and buffered* in order to deliver the two clock signals to the UDD. These additional wires and buffers (indicated with the thick, dotted lines in the figure) must be large to maintain accuracy of the UDD's skew measurement over PVT variations. 2.) The skew measurement is being performed at level-2 of the clock tree, but the relevant clock skew is at level-4, the clock leaves.

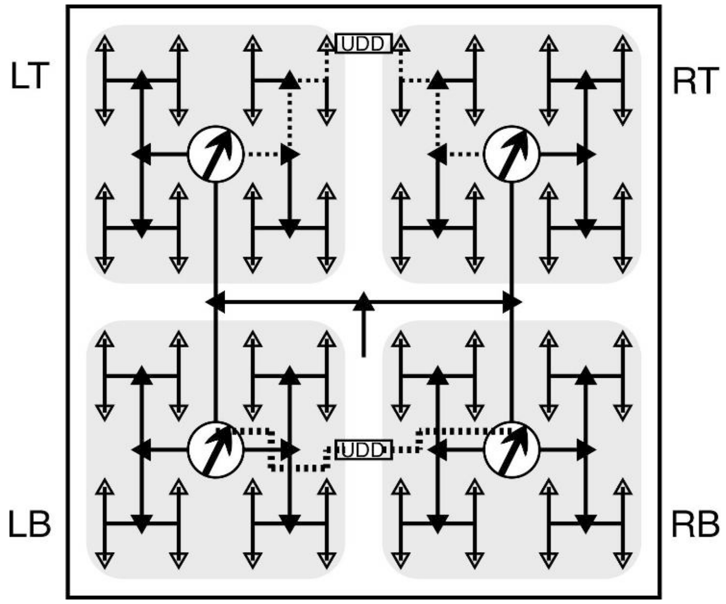


Figure 5.11 : UDD connections

A better implementation is shown in the **upper** half of the same diagram of Figure 5.11. UDD is placed at the level-4 clock leaves (one side from LT and the other from RT).

This choice solves both problems just mentioned. 1.) No new, *major* clock routing wires are needed -- the existing clock tree wires are sufficient (indicated with the thin dotted lines). 2.) The skew measurement is performed at the level-4 of the clock -- where true clock skew is defined. So, *UDDs will be placed across adjacent clock leaves of neighboring clock zones* to simplify the design and routing. While this can be done easily in the classic H-tree, designers will have to look for similar opportunities in other clock structures.

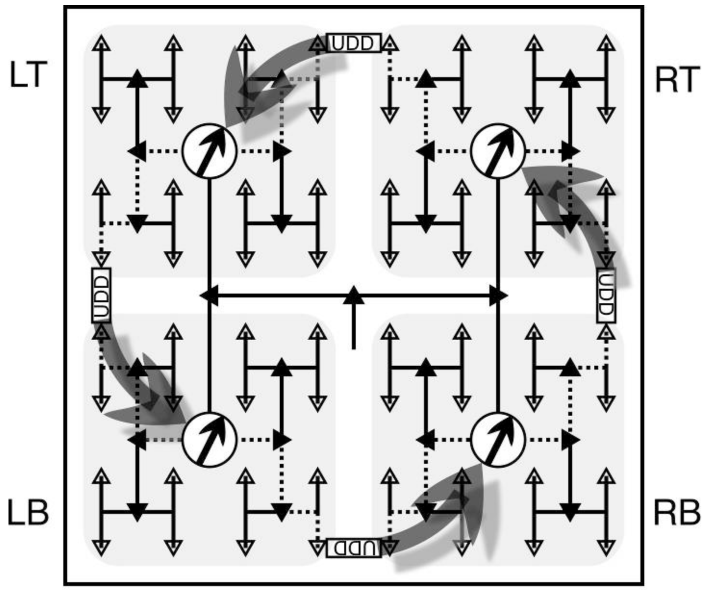


Figure 5.12 : QRT level-2 operation

Now that UDDs have been optimally located at the ends of adjacent clock leaves, one UDD for each TB will be placed spanning each shared boundary of the four tuning zones. They are all connected in a counter-clockwise ring to create the QRT structure, as depicted in Figure 5.12. The skew measured across LT/RT boundary by the top UDD is used to control TB of LT zone; LT/LB boundary skew controls LB TB; LB/RB controls RB; and finally RB/RT controls RT, which completes the ring.

There are four complete active clock deskewing loops or DLLs (as described in Figure 5.1) operating here, but because their control signals are derived from the skew across the *boundary of neighboring zones*, they are all *interlinked* in a *ring* formation, as indicated by the dotted lines and arrows of Figure 5.12. *The feedback (or tuned) clock of one zone becomes the reference clock of its counter-clockwise neighboring zone, and so on, all the way around the ring.*

In this example, the four UDDs are placed at the perimeter of the chip in order to encompass maximum intra-die PVT variation. Although, as described previously, UDDs could be placed anywhere along the zones' boundary; furthermore, multiple UDDs could be combined with logical operands (such as AND or OR) if desired. Since this hypothetical example is for a generic square chip without any special timing considerations or constraints of the internal circuit sub-blocks, positioning four single UDDs at the chip's perimeter as shown are sufficient.

5.4.3 Hierarchical Ring Tuning for H-tree

The basic principle of QRT has been explained using clock tuning zones at clock level-2 of the common H-tree, as depicted earlier in Figure 5.12. However, further analysis of this 4-level H-tree's structure reveals another possible implementation of QRT -- namely at clock level-3. Four additional, complete QRT structures can be located hierarchically within the initial QRT, as shown in Figure 5.13.

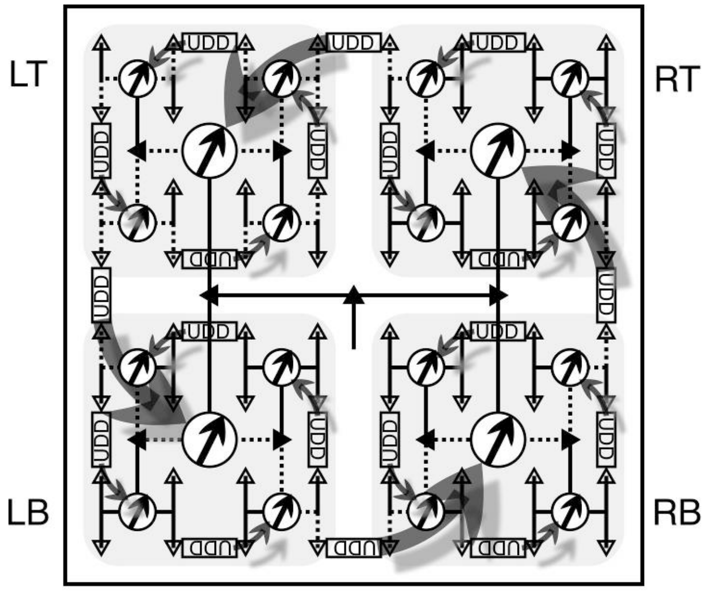


Figure 5.13 : QRT level-2&3 operation

Four distinct QRT 'sub-systems' for controlling the new TBs at clock level-3 can be seen (one *ring* inside each of the LT, LB, RB, and RT quadrants). Each one of these new, level-3 sub-system QRTs contains four UDDs at the clock leaf perimeter controlling four TBs via four TCGs, for a total of sixteen additional UDDs, TCGs, and TBs. The small and simple design of these three circuit sub-blocks allows them to be added at little additional cost.

Except for TB's tuning range and UDD's threshold, these four new QRTs at level-3 are designed, connected, and operated exactly like the existing QRT at level-2. The original, full-chip QRT (clock level-2) was designed for a tuning range of 50ps to cover PVT variations expected across the entire full-chip. However, these new sub-quadrant QRTs (clock level-3) span one-fourth of the chip area, and therefore they are designed with a smaller tuning range of 20ps. *Note that once the TB's tuning range and number of steps have been determined, its step size can be calculated,* ss=(tuning_range/(#steps-1))*, and then* *the corresponding UDD's threshold is designed to be* S_{th}=1.5ss*, as previously explained.* These (4+1) QRT structures all operate concurrently, working to tune the (16+4) TBs such that the (16+4) UDDs each obtain *lock*, even though they are all interrelated in some manner.

The same UDD threshold design technique that guarantees **one** or **two** *stable locking states* per active clock deskewing loop works to keep all of these (4+1) QRT systems collectively stable. By using this two-level method of QRT, the clock skews of the individual quadrants themselves are reduced by the level-3 QRT systems. Furthermore, the intra-quadrant skew improvement in turn *aids* the operation of the overall level-2 QRT system. Therefore, the *sub-skews* within each quadrant will be *reduced* and, additionally, the *full-chip clock skew* will also be *reduced*.

5.5 H-tree Ring Tuning Simulation Results

HSPICE simulations were carried out to compare the clock skew of two static H-trees (one using TC and one using SEC), and two active QRT systems (single-level and multilevel). Both of the QRT systems are complete SEC systems (i.e., they use SEC clock buffers instead of TC clock buffers), since SEC was already shown to outperform TC. It is important to note, as described in Section 2.2, that a TC versus SEC comparison was made previously by modeling PVT variations as worst-case process models, supply voltages, and operation temperatures; however, a completely different approach for PVT variations is used in this chapter. Here, the PVT variations are modeled at the higher level of *transistor strengths* and *wire loads*. This alternative PVT variation model further (and independently) confirms the reported clock timing improvements of SEC over TC, as demonstrated by the following results.

5.5.1 Full H-tree Netlist

The same generic fanout-4, 4-level H-tree for a hypothetical 10x10mm^2 chip from the preceding chapters was used in the simulations. For the two static clock tree cases, all the clock buffers were TC and SEC, respectively. For the two active deskewing cases, the QRT structures were designed into the H-tree at clock level-2 and level-3, as previously described.

Time-domain transient simulations were run until steady-state was achieved (just a few clock cycles for the static cases, but all DLLs must lock for the active cases). As presented in Section 1.2, skew will be defined here as the difference in clock edge arrival times for two different clock branches, $\Delta t_{i\text{-}j}=t_i\text{-}t_j$, and the goal will be to minimize the *maximum clock skew*, given by the absolute value maximum of $\Delta t_{i\text{-}j}=t_i\text{-}t_j$ for $i \neq j$, where i and j cover every clock leaf of the entire chip. When the range of delays for all 64 clock leaves is considered, this maximum clock skew will be referred to as *full-chip skew*. The resulting full-chip skews of the static SEC H-tree are compared to those of the static TC H-tree for confirmation of the improved clock performance of SEC over TC. Then, full-chip skews of the single and multi-level QRT cases are compared against those of the static SEC for demonstration of the improved clock performance of this active deskewing system over a static system.

As the HSPICE data presented below shows, even this very basic (only 3-bit TB) QRT system can be used to reduce clock skew by over 50% versus an untuned, static H-tree. Furthermore, the QRT ring consumes only *1mW* of power and each DLL consists of three major blocks: UDD (*20* small transistors[20]), TCG (*78* small transistors), and TB *(~4X* the size of a non-tunable clock buffer).

5.5.2 PVT Variation Modeling

Firstly, in order to tune away clock skew, clock signal delay variation must be inserted into the nominal H-tree clock distribution network schematic for HSPICE, which is initially created as perfectly balanced. This intra-die variation exists in real-world circuits due to PVT variations. Each branch of the clock tree was subjected to different PVT conditions; consequently, these PVT variations alter the clock signal's timing as it propagates from the single clock root to all of the 64 clock leaves, thereby creating clock skew across the die.

The sources of these PVT variations are numerous, but they can all be classified into two distinct types [2][17]: *spatial variations*, which vary across the surface of the die according to some spatial, or x-y, pattern (e.g., concentric circles or cone function), and *random variations*, which vary across the die's surface according to some random function (e.g., a Gaussian distribution). Certain PVT variations tend to follow a spatial pattern, while other PVT variations tend to follow a random pattern, so both types of variation will be combined for these simulations.

The random variations are the easiest to include in HSPICE's Monte Carlo (M.C.) simulations. Once a *mean* and *standard deviation* has been determined for a given parameter, the HSPICE *gauss()* function can be called to assign that parameter a Gaussian distributed random value for each M.C. simulation run. The spatial variations, however,

[20] Note that this count would become somewhat larger if a design with adequate robustness to meta-stability was used

must be calculated into a matrix via an outside tool (in this case MATLAB), and then superimposed atop the appropriately (x,y) segmented H-tree within the HSPICE netlist.

Traditional spatial variation modeling such as [76] would involve a multi-level partitioning of the circuit followed by a data analysis procedure such a kriging[21] to calculate each partition's correlation factor and construct a variogram (or structure function) that defines the spatial random process [77]. However, such an analysis is not suitable in this case since the entire QRT system is a relatively small number of elements that are widely dispersed across the entire chip, and since no particular process data is available for correlation analysis. Instead, the spatial random process in $\mathbf{R}^2$ space will be modeled with the general expression, $z = Z(x,y)$, from [77].

Previous work on spatial variation modeling [16][78] has produced surface plots of this $Z(x,y)$ function that closely resemble half sphere or cone shapes. Since the sharper peak of the cone function has been empirically shown to produce a more severe test of QRT's deskewing capability, the following cone function has been chosen for these spatial variation pattern:

$$z^2 = (x-a)^2 + (y-b)^2 \tag{5-1}$$

where,

(x,y) = two dimensional Euclidean coordinates of the chip

(a,b) = center of the cone function within that (x,y) space

{The six center locations chosen for these are shown in Figure 5.14*}*

z = relative variation of spatially modeled parameter at (x,y) location of the chip

[21] "Kriging" is a geostatistical term referring to techniques to interpolate the value, or surface function called a variogram, of a random field. It can be used in PVT variation modeling to construct a thorough surface function of one or more spatial variations across the surface of a chip.

{This z value is scaled to range between +10% and -10% of the nominal.}

It is important to note that unlike a traditional spatial variation analysis that attempts to precisely correlate process parameters by their proximity on the chip in order to model real-world variations, these spatial variation simulations are primarily constructed to introduce a maximum amount of clock skew in a chosen pattern (cone function) across the full-chip in order to stress test QRT's ability to tune away clock skew. Therefore, instead of modeling the fundamental process parameters (such as *Leff* or *Vth*), this analysis will only vary the two higher-level, composite parameters of transistor strengths and metal wire loads. The resulting clock skew patterns have been compared to the global skew maps of [79] to validate their usage here.

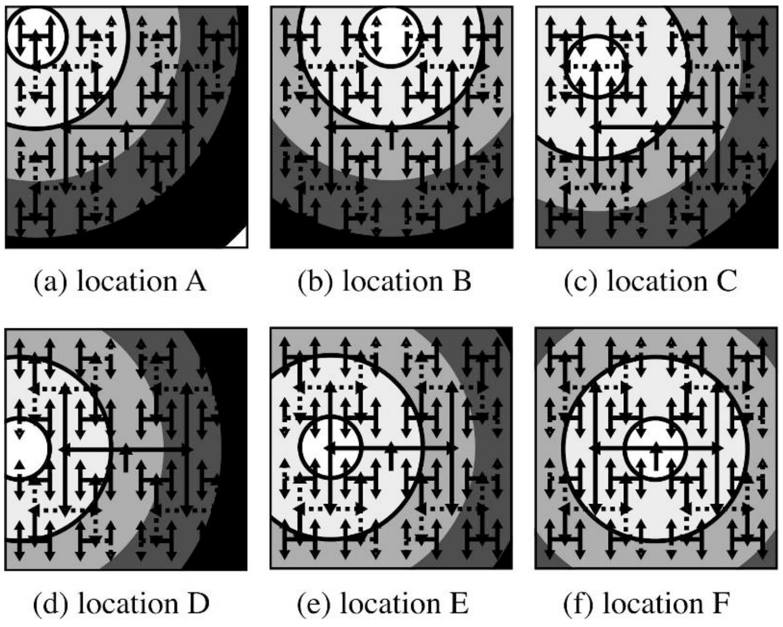


Figure 5.14 : Location of spatial variations

The cone function creates concentric circles of equal values centered at *(a,b)* which radiate from its center linearly. For a *non-inverted* cone, its *maximum* value will be at the center (or peak), and the minimum value will be at the outer (or largest) circle. The *inverted* cone is just the opposite – the *minimum* is at the center and the maximum is at the outer circle. Since the center of a PVT spatial variation could reasonably occur at any *(x,y)* location on the die, six representative locations (designated A, B, C, D, E, and F) have been chosen in the left-top quadrant of this hypothetical square chip as shown in Figure 5.14. Because the QRT method is virtually identical for each quadrant, the remaining three quadrants would be redundant to consider; therefore, these six spatial variations, plus their *inverses*, will be used to represent the spatial PVT variations across this entire die – twelve spatial variations over a single quadrant.

The mathematical models of PVT variations have been determined: *random variations* will be modeled with a Gaussian distribution and *spatial variations* will be modeled with cone functions (or concentric circles) and their inverses placed at six locations across a quadrant. Next, the parameters that will be varied must be chosen. A clock tree only consists of clock buffers and clock wires, so these two components are the only ones that need to be varied. The variation (random and spatial) for these simulations is only being included to create some initial clock skew in an otherwise *nominal* (ideally symmetrical and balanced) clock tree; so, simpler, higher-level parameters of the clock buffers and wires have been chosen to model these variations. Namely, the *M=xx multiplier* of HSPICE (which scales a transistor's channel width, leakage, capacitors, and resistors) has been chosen to vary the clock buffers, and the *length* and the *resistance* and *capacitance per unit length* have been chosen to vary the clock wires. These four variables along with the real circuit parameters they are intended to model are listed in Table 5-2. Additionally, the degree of spatial and random variation for each variable with respect to its nominal value (as a percentage) is included in the same table.

Table 5-2 : PVT variations used in simulations

	Real Circuit Parameters	HSPICE Model Variable	Spatial Variation	Random Variation
Transistors	Leff, Vth, Idrive, etc.	Multiplier (M=xx)	±10%	±5%
Metal Wires	clock tree RC load	Wire length	±5%	
	Metal resistivity	R_{wire}/length		±10%
	Metal capacitance	C_{wire}/length		±5%

As shown in Table 5-2, the HSPICE element multiplier (M=xx) variable was used in these simulations to model PVT variations that would impact the transistors by adjusting their effective size and thereby their strength. Importantly, these variations affect *all transistors in the entire netlist*, not just the clock buffer transistors. This transistor variation is made up of a spatial component that is varied ±10% and a random component that is varied ±5% about its nominal value.

Likewise, the metal wires used in the clock distribution are varied in both a spatial (±5% each for total resistance & capacitance) and random (±10% for resistance per length and ±5% for capacitance per length) component about their nominal values. For these simulations, the spatial pattern for the metal wires and transistors is chosen to be the same cone function (i.e., centered at the same *x,y* location); however, *the two are correlated such* *that weak transistors drive large wires (and strong transistors drive small wires)* to magnify their clock skew effect. This correlation between transistors strengths and wire sizes has been chosen based upon a sampling of simulations resulting in slightly larger clock skews, but other transistor/wire correlations are expected to produce similar skew results.

These levels of spatial and random variation have been chosen based upon previous research [2][17][18][19], and confirmed to produce typical levels of clock skew in this hypothetical H-tree case. The total PVT variation for a given parameter, *P*, is thus modeled by the following equation [80].

$$P = P_{nom} + \Delta P_{inter} + \Delta P_{spatial}(x_i, y_i) + \Delta P_{random,i} \tag{5-2}$$

where P_{nom} = the nominal value of this parameter

ΔP_{inter} = some amount of *inter-die* variation for this one particular die for this parameter

{This value is the same across the entire die; hence, it does not cause clock skew, and it is ignored here.}

$\Delta P_{spatial}$ = some amount of *intra-die* variation which is a function of location on the die (x,y) for this parameter

{This spatial variation is modeled as a cone and inverse cone function with six different centers each -- totaling twelve cases in one quadrant.}

ΔP_{random} = some amount of random variation for each device on this die for this parameter

{This random variation is modeled as a zero mean random variable.}

5.5.3 SEC-only Improvements

To confirm the SEC clock performance improvements reported Chapter 3, a full-chip SEC implementation is compared to a TC implementation in Table 5-3. As previously mentioned, the simulations used a fundamentally different PVT variation model than the one used in the preceding chapters, so this data represents a new, independent confirmation of the superior timing performance of SEC over TC. As the data shows, for every spatial variation pattern, SEC reduces full-chip skew by ~20% versus TC -- the same value as the preceding results. An illustration of the TC to SEC improvements is shown later in the clock skew surface plot of Figure 5.16(a) to (b).

Table 5-3 : Overall clock skew results for SEC versus TC

PVT Spatial Variation Pattern	full-chip skew (ps) *(mean of 35 Monte-Carlo runs each)*	
	TC	SEC
A	70.1	54.3
Ai	66.3	50.9
B	54.5	42.3
Bi	52.7	40.9
C	73.4	57.0
Ci	71.7	55.0
D	76.3	59.9
Di	73.3	57.4
E	74.1	58.6
Ei	72.6	57.2
F	34.1	27.5
Fi	33.0	26.6
Average Improvement	-	22%

The remainder of these simulation results will be comparing the new QRT method (single level and multi-level) with *no QRT*. Since both QRT systems utilize SEC buffers, the upcoming data table will compare the QRT methods to the untuned SEC method -- not the TC method.

5.5.4 SEC + QRT-2 Improvements

In order to demonstrate the locking operation of QRT in action, transient simulation graphs of one single HSPICE M.C. simulation run are provided in Figure 5.15. In Figure 5.15(a), a close-up view of the critical timing edge of the eight leaf clocks is being compared with the four UDDs of this system (refer to Figure 5.12) *before* the tuning has begun. Timing variations (or clock skew) of greater than 50ps can be seen in this top graph. The next four graphs, Figure 5.15(b) to (e), are individual plots of the *UP/DOWN* signals and the *TB3-TB1* thermometer bits for each of the four quadrants (LT, LB, RB, and RT, respectively). The important data from these four graphs have been reproduced in Table 5-4 for convenience. Finally, the bottom graph, Figure 5.15(f), is another close-up view of the same eight clock leaf edges *after* the tuning has *locked*. The edge variation (or clock skew) has been reduced to less than 20ps in about 10 clock cycles (~30ns-35ns) with the single level QRT technique.

The specific operation of QRT's tuning process can be understood by stepping through the data of Table 5-4 (taken directly from the four middle graphs of Figure 5.15). There are four rows, one for each of the four quadrants, and five columns of data taken at significant time points of the transient simulation. Each cell contains TB's bit state (e.g. *001*) and UDD's control signal (UP, DN, or neither = *lock*) at the chosen simulation times. Any changes to the bit states are highlighted in bold.

At startup (2ns column), all TBs' bits are at *001*; two quadrants need to be tuned down (LT and RT), and two quadrants up (LB and RB). By 10ns, LB quadrant has tuned up to *011*, and its measured clocks are within UDD's thresholds, so it is *locked*. RB quadrant has also tuned up to *011*, but it is still not in the lock region. RT quadrant is not yet able to tune down, but since its UDD is controlled by RB/RT relative skew, and since RB has just tuned up, RT quadrant is now in the lock region.

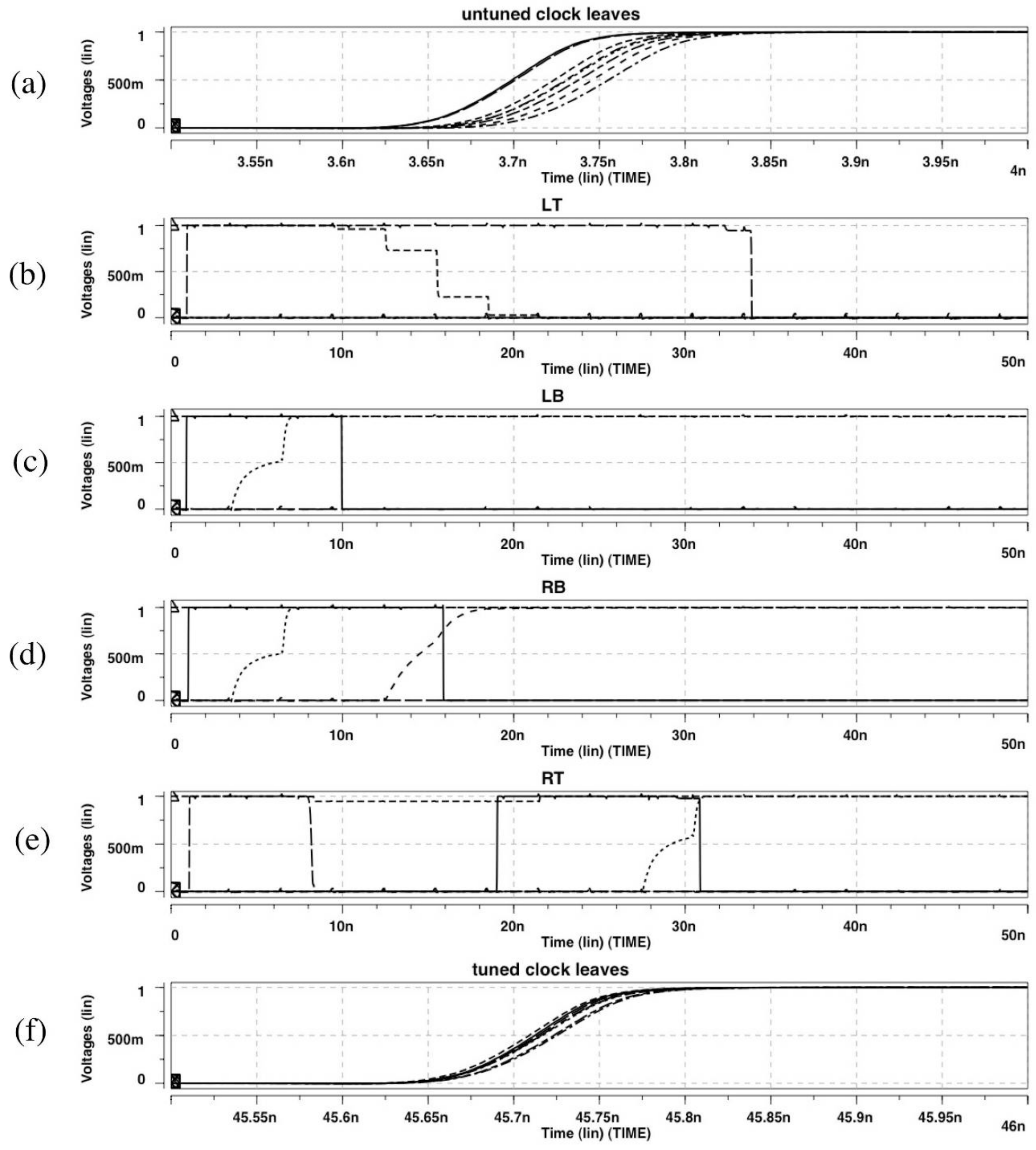


Figure 5.15 : Transient simulation waveforms of QRT-2 operation

Table 5-4 : Tunable buffer bits state and UP/DN signal over locking time

chip quadrant	UDD output TB setting	transient simulation time				
		2ns	10ns	20ns	32ns	35ns
LT	UP/DN	DN	DN	DN	DN	*lock*
	bits	001	001	**000**	000	000
LB	UP/DN	UP	*lock*	*lock*	*lock*	*lock*
	bits	001	**011**	011	011	011
RB	UP/DN	UP	UP	*lock*	*lock*	*lock*
	bits	001	**011**	**111**	111	111
RT	UP/DN	DN	*lock*	UP	*lock*	*lock*
	bits	001	001	001	**011**	011

* data taken from graphs of Figure 5.15

At 20ns, LT quadrant has tuned down to *000*, but it is still not enough to lock, so it would like to tune down further; unfortunately, it has run out of its downward tuning range, but nevertheless QRT will continue to operate properly. LB remains locked as before. RB has tuned up to *111*, and it is now *locked* too; and because RB has tuned up, RT's UDD has come out of lock, and now wants to tune up (following the RB move).

By 32ns RT has tuned up to *011* and it is now *locked*; and by RT tuning up, LT's UDD (which is controlled by the RT/LT relative skew) has now reached *lock*. Therefore, at 35ns (just over 10 clock cycles) all four quadrants are locked: LT at its maximum delay setting (*000*), RB at its minimum delay (*111*), and LB and RT at middle delays (*011*), and they all will *remain locked indefinitely*.

Table 5-5 : Overall clock skew results for QRT

Clock Skew Results for Single & Multi-level QRT

PVT Spatial Variation Pattern	no tuning full-chip skew (ps)	full-chip skew average % improvement	
		QRT-2	QRT-2&3
A	54.3	54%	59%
Ai	50.9	52%	60%
B	42.3	42%	56%
Bi	40.9	41%	56%
C	57.0	52%	57%
Ci	55.0	53%	49%
D	59.9	56%	63%
Di	57.4	55%	65%
E	58.6	55%	64%
Ei	57.2	56%	62%
F	27.5	0.5%	40%
Fi	26.6	-2.6%	40%
Average Improvement	-	**43%**	**56%**

Full-chip SEC clock skew results for single QRT at clock level-2 are provided in Table 5-5 column QRT-2. As previously described, six spatial variation patterns (A through F) along with their inverses (denoted with an “i” suffix) were chosen for the HSPICE simulations, totaling 12 rows of data in the table. For each pattern, 35 Monte Carlo transient simulations were run until steady-state was reached – a total of 420 complete simulation runs. For each simulation run, the full-chip skew is calculated from the clock arrival times of all 64 leaf clocks.

In *every* simulation run (excluding the F and Fi patterns which will be explained shortly) the full-chip skew has been dramatically reduced using a single-level QRT. The average percent improvement versus the untuned (no QRT) network is listed in column QRT-2 of the table. (The QRT-2&3 column will be discussed later with multi-level QRT simulations.) Even including the poor F and Fi results, the total average improvement for QRT-2 over all chosen PVT variations is shown at the bottom of the table to be *43%*.

For spatial patterns A through Ei, the QRT-2 technique reduces the full-chip skew by more than 50%. However, for the F and Fi patterns – both located at the exact center of the H-tree as shown in Figure 5.14(f), the spatial variation is symmetrically distributed across all four quadrants leading to minimal skew at the leaves located in the outermost ring of the clock tree. So, the chosen UDD placement with QRT-2 (shown in Figure 5.12) measures the low skew between the tuning zones, and thus no improvement is realized. However, there is intra-quadrant skew between internal leaves of the clock tree that goes undetected. This intra-quadrant skew is calculated using the same technique described in Section 1.2, but only considers the 16 clock leaves within each quadrant of the chip. As described next, multi-level QRT will help to address this symmetrical problem and significantly improve these clock skew results.

5.5.5 SEC + QRT-2&3 Improvements

As explained previously, four more QRT systems can be added at clock level-3 to create a multi-level QRT implementation with tuning at both levels 2 and 3, referred to here as QRT-2&3. In order to observe the behavior of single-level and multi-level QRT, surface plots of the clock leaf delays for all four clock tree designs are displayed in Figure 5.16. Figure 5.16(a) is from TC clock buffers without QRT (no tuning), Figure 5.16(b) is from

SEC clock buffers without QRT (no tuning), Figure 5.16(c) is from SEC single-level QRT at clock level-2, and Figure 5.16(d) is SEC multi-level QRT at clock level-2 and 3. The z-axis for each plot is the relative clock delay measured at the 64 clock leaves, and the x-y axes indicate the H-tree clock leaves' 64 spatial locations on the square chip.

The *full-chip skew,* as calculated from all 64 clock leaves and defined in Section 5.5.1, is explicitly labeled as 'skew' on each plot. Additionally, the *intra-quadrant skew*, calculated from only the 16 clock leaves within each quadrant, is also labeled on each of the four quadrants for each surface plot. *(Note that these clock delay surface plots are for the exact same simulation conditions as the data of* Figure 5.15 *and* Table 5-4.*)* As covered previously, the improvement from TC to SEC is evident in the ~20% reduction in skews from plots Figure 5.16(a) and (b).

In comparing the active tuning performance of QRT, only the SEC plots of Figure 5.16(b), (c) and (d) will be considered, since they all use the same SEC clock buffers. As the baseline (no tuning) plot of Figure 5.16(b) shows, the PVT variation of this particular simulation produces *64.5ps* of clock skew for a traditional H-tree distribution network. The spatial variation of this pattern (pattern C of Figure 5.14(c), in this case) is noticeably evident in the small delays for the LT quadrant, rising to the large delays of the RB quadrant.

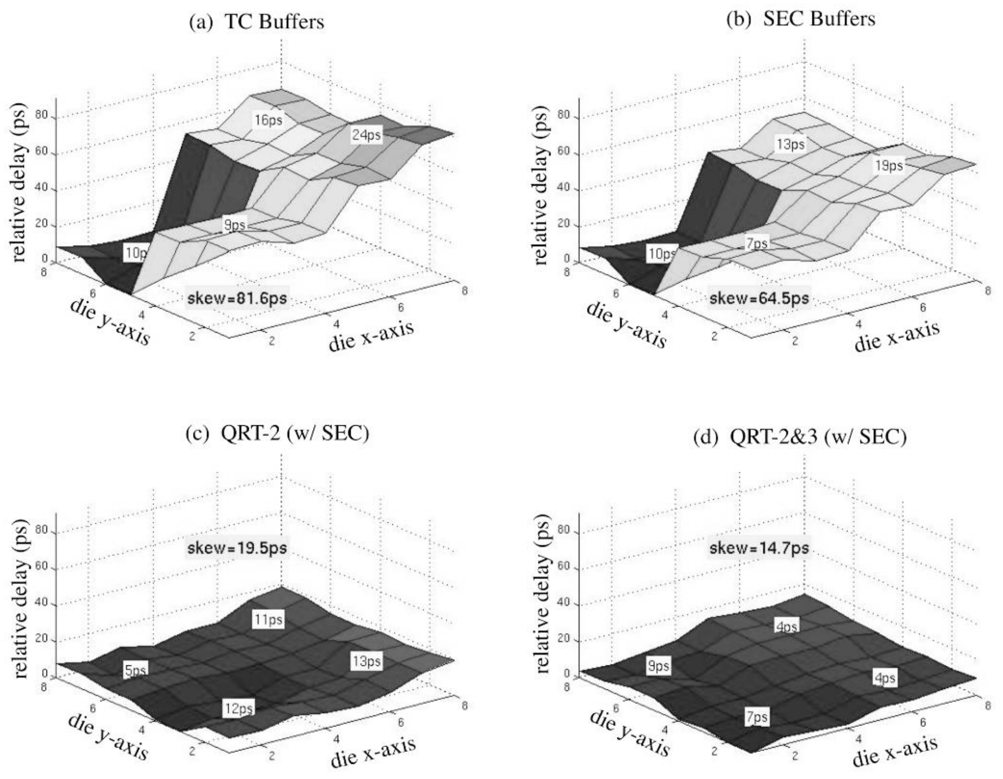


Figure 5.16 : Surface plots of H-tree leaf clocks for four clock designs

Once QRT-2 is implemented, the four clock zones (LT, LB, RB, and RT) will commence to autonomously tune themselves to an equal delay, and the full-chip skew will ultimately be reduced to *19.5ps* as shown in the plot of Figure 5.16(c). For this case, as previously shown in Table 5-4, after tuning to *lock* LT zone increased itself to maximum delay (*000*), RB zone decreased itself to minimum delay (*111*), while LB and RT remained in the middle (*011*). Since these TBs have a step-size of approximately 16ps, LT has gone up by *~16ps*, LB and RT have gone down by *~16ps*, and RB has gone down by *~32ps*, resulting in an overall relative change of *~48ps*, and a *flattening* of the full-chip clock delays is very evident in the surface plots from Figure 5.16(b) to (c).

Finally, when the QRT-2&3 (multi-level) is implemented, the sub-skew improvements within each quadrant can be seen in the plot Figure 5.16(d). Unfortunately, LT sub-skew does increase slightly from *5ps* to *9ps* (still within the QRT-3 threshold design window), but the remaining three quadrants all show significant improvements (*12ps* down to *7ps*, *11ps* and *13ps* both down to *4ps*). Furthermore, the flattening of each of these quadrants has *aided* with the flattening of the full chip, as evident in the reduction of full-chip skew from *19.5ps* for QRT-2 in Figure 5.16(c) down to *14.7ps* for QRT-2&3 in Figure 5.16(d).

Now that the detailed operations of the single-level and multi-level QRT systems have been presented, the final HSPICE simulation data can be analyzed. The same twelve spatial patterns described earlier were simulated over the same 35 HSPICE Monte Carlo transient runs to evaluate the associated clock skew improvements of this multi-level implementation (QRT-2&3) in comparison to the previous two designs (no tuning and QRT-2). Based on the results, the primary improvement realized by adding four QRT-3 systems at the quadrant level is a reduction in the sub-skew or intra-quadrant clock skews.

Table 5-6 : Sub-quadrant skew results for QRT

Sub-Quadrant Skew Results for Single & Multi-level QRT

PVT Spatial Variation Pattern	no tuning max quadrant skew (ps)	max quadrant skew average % improvement	
		QRT-2	QRT-2&3
A	14.4	-5%	35%
Ai	13.9	-1%	30%
B	15.8	-1%	36%
Bi	16.2	3%	42%
C	16.1	-5%	43%
Ci	15.1	-5%	42%
D	17.1	-3%	42%
Di	16.8	1%	45%
E	18.6	-5%	46%
Ei	17.4	-2%	45%
F	23.6	1%	55%
Fi	22.4	1%	51%
Average Improvement	-	-1.8%	**43%**

Since all four quadrants are being treated as equally important in this hypothetical example, only the maximum (or worst case) of the four quadrant sub-skews will be compared. Table 5-6 shows these data for all three SEC designs over the twelve spatial variation patterns. Since QRT-2 implementation does not operate *within* the quadrants themselves, no improvement would be expected over the baseline (no tuning) technique, and accordingly the table shows a minimal *-1.8%* average change. However, with the addition of the QRT-3 *within* each quadrant, the sub-skews are now dramatically improved, as shown by the *43%* total average at the bottom of column QRT-2&3. Therefore, smaller skews and tighter control of the clock edges can be obtained *within* each quadrant by the addition of QRT-3 (not possible with QRT-2 alone).

However, as stated at the outset, the purpose of this example is to reduce full-chip clock skew across the entire chip, not just intra-quadrant clock skew. As the data of Table 5-5 shows, the addition of QRT-3 helps to reduce full-chip skew as well. The previous *43%* improvement in clock skew from QRT-2 has improved to *56%* with the addition of the multi-level QRT-2&3. The explanation for this improvement was briefly addressed in the discussion of the two surface plots of Figure 5.16(c) and Figure 5.16(d). When the four QRT-3 systems are allowed to flatten out the skew *inside* of each of the four quadrants, then the single QRT-2 system will have a much better chance of flattening out the entire chip's clock delays.

Finally, it was previously pointed out that QRT-2 system alone cannot improve the F and Fi patterns due to their symmetrical pattern over the full chip, but as the data of Table 5-5 shows, the QRT-2&3 multi-level system can significantly improve the full-chip clock skew because that same symmetrical pattern effect is not seen at the individual quadrant level (QRT-3 level).

5.6 QRT Summary and Limitations

Controlling clock skew is vital to the functionality of synchronous circuit designs. As intra-die variations continue to increase, some form of post-silicon clock adjustability may become valuable for tuning away unwanted clock skew. However, performing full-chip clock tuning via ATE test or using several complex system-level controllers may prove too expensive for typical ASIC designs. Therefore, the autonomous clock tuning architecture of QRT has been developed to allow clock zones to essentially tune themselves.

This chapter first described the state of the art of multiple zone tuning systems which has progressed from the cyclic architecture to the acyclic architecture to avoid the stability problems inherent in cyclic designs, namely *mode-lock*, *mode-lock oscillation*, and *cyclic-feedback oscillation*. Then, it proposed a return to a cyclic architecture to avoid the *out of range* problem and its need for a global controller inherent with acyclic designs. This new cyclic architecture has been called *ring tuning*.

Ring tuning is a novel technique for reducing clock skew in typical synchronous circuits by linking together multiple deskew loops that are distributed across the entire chip. Implementing ring tuning for post-silicon, active clock deskewing allows the tunable clock zones of a chip to autonomously adjust their clock delays during normal circuit operation and to stably *lock-in* below a designer chosen skew threshold. A specific implementation, called QRT, was presented using a total of four deskew tuning loops. The problems of *mode-lock*, *mode-lock oscillation,* and *cyclic-feedback oscillation* have been addressed with the design of the UDD, and the problem of *out of range* has been addressed with the design of the TB and TCG. This QRT system has been shown to operate with stability without the need for any global controller, making it easier to implement than previous techniques. Then, a hierarchically implementation of multi-level QRT helped to overcome the linear growth of skew bound for larger numbers of tuning zones. This approach, called QRT-2&3, was shown to *reduce (on average) intra-quadrant clock skew by 43% and full-chip clock skew by 56%.*

The stability of this ring tuning technique is guaranteed by adjusting TBs only one step at a time and by widening UDD’s threshold in order to provide *at least one or more* valid locking state for all four DLLs. Therefore, the step-size of TB is the true limiting parameter in the resolution of ring tuning systems – provided of course that TB has sufficient tuning range to compensate for all expected skews. Even the very simple 3-bit (4-step) TB used in these simulations produced clock skew improvements of more than 50% using QRT. However, delay elements with higher resolutions have been presented [71], including a simple modification to this TB resulting in sixteen steps of monotonic tuning [70][81], and a 128-bit, 1ps-step TB [82]. Using such a high resolution TB here, would allow for the same tuning range to be covered with a smaller step size; consequently, increasing the resolution of QRT greatly.

Also, just like any DLL, this QRT system must run for several clock cycles to lock. During this *locking time*, the clock signals will be periodically adjusted until the skew thresholds are met (as demonstrated in Figure 5.15 and Table 5-4). Of the 420 single-level and 420 multi-level QRT PVT varying HSPICE simulations performed here, *all* of them ultimately stabilized. So, the single and multi-level QRT techniques have been demonstrated to stabilize and lock *consistently* and *quickly* over PVT variations.

Chapter 6 : Summary, Conclusions, and Future Work

Synchronous circuits comprised of flip-flops (FF) driven by a clock are still the preferred choice in most chips today, and therefore a reliable clock signal remains a critical design focus. However, the on-chip PVT (process-voltage-temperature) variations unquestionably impact the clock buffers and clock wires that comprise the clock distribution network. While designers try to maintain balance in the clock branches by adhering to well-established layout strategies, the PVT variations inherently unbalance the clock distribution network and subsequently result in excessive clock skew and jitter. As PVT variations are expect to increase in future process technologies, so too will the clock skew and jitter increase. Therefore, new techniques to control clock skew are the focus in this dissertation. In addition, improvements to clock jitter and clock latency are also addressed here.

Two options exist to reduce clock skew: a general approach that improves the clock distribution network on all chips via design and/or layout techniques prior to mask generation, called *pre-silicon methods*, and a chip-specific tailored approach that measures clock skew during chip operation and correspondingly tunes the clock signal arrival time to cancel the skew out, called *post-silicon methods*.

The pre-silicon techniques are relatively cheaper and easier than the post-silicon techniques, so they are always the first-line of defense in terms of skew reduction. However, as PVT variations continue to grow with shrinking process technologies, the limits of pre-silicon techniques are being reached. Furthermore, many of the PVT variations are experienced by the circuit only during its operation, so they are completely unpredictable in a pre-silicon context. Therefore, post-silicon techniques are becoming more prevalent for skew compensation. Novel techniques for both pre-silicon skew reduction and post-silicon skew compensation have been proposed in this research. In the sections below, we review the contributions, describe certain limitations and propose future work for each category.

6.1 Pre-silicon Skew Reduction

6.1.1 Summary of Contributions

The first key contribution was to design a new low-skew clocking scheme that is a drop-in replacement for the traditional clocking (TC) systems. It was based on the realization that, of the four segments that comprise a typical clock period (rising-edge, high-level, falling-edge, and low-level), only one segment (e.g., the rising-edge for a single-edge triggered FF systems) is truly critical to the timing performance of a synchronous system. Therefore, that rising-edge has been dubbed the *critical* clock edge, and a novel design technique, called *Single-Edge Clocking* (SEC), was proposed to improve that critical-edge segment while allowing the other three non-critical segments to gracefully degrade, but not fail.

The goal of the SEC technique is to create a sharp critical edge and push it as quickly as possible through the clock distribution network. This task requires the use of asymmetric clock buffers, called INVr and INVf, that create sharp rising-edge and falling-edge signals, respectively, by choosing suitable β values for each type of inverter. Consequently, the two buffers are alternated throughout the clock distribution network as the signal inverts through each stage. The equations for selecting the appropriate inverters' β's for a given clock distribution network and for sizing the SEC buffers to directly replace the TC buffers have been derived.

The maximum clock timing performance gain from this SEC technique was obtained by designing the full-chip clock distribution network completely using these alternating INVr and INVf buffers. *Simulation results on a typical clock H-tree have demonstrated timing improvements in clock latency, skew, and jitter of more than 20% for SEC over a traditional clock buffer network, without any significant increase in power or area.* Furthermore, a single buffer replacement option has been proposed which allows for fine adjustments in clock path timing by dropping in an SEC buffer in place of a TC buffer (or a more aggressive SEC buffer in place of a less aggressive one) to either speed-up or slow-down one clock path with respect to the others.

6.1.2 Limitations

The drawback for improving the critical clock edge in a full-chip SEC design is not a 1-to-1 tradeoff; therefore some other concessions and modifications must be made. In particular, the SEC system requires a clock pulse to initially drive the network. That is, the clock signal should first be reduced from a fifty percent duty cycle square wave to a minimum-sized clock pulse in order to allow for clock pulse growth during the clock signal propagation from root to leaf. Generating such a clock pulse from the clock source PLL is a relatively simple design modification with no real sacrifice in circuit operation.

Furthermore, the clock frequency of the SEC system would preferably be less than the process technology maximum to allow for additional clock period tradeoffs. In the example described in this dissertation, the clock for a 90nm technology is chosen to be 250MHz -- much less than the GHz range maximum frequency possible. Nevertheless, less-than-maximum clocks are quite typical in ASIC designs, so SEC should not be greatly limited by this frequency requirement. However, since it is only the pulse width growth problem of SEC (detailed in Section 3.1.1) which limits its maximum frequency, further research into techniques to fix this problem would enable SEC to reach its full potential.

The non-critical edge, or neglected edge, of the SEC buffers creates a couple of additional design concerns. First, the slow transitioning of that edge allows for simultaneous conduction of both the PMOS and NMOS transistors, thereby creating short-circuit current. Second, the resulting shift in the voltage transfer curves will reduce noise margin of these inverters in series. Neither of these two concerns is significant in nature, but they should both be monitored closely during the design and layout phases.

Finally, the slow transition of the neglected edge may create some additional timing or short-circuit considerations when driving the FFs at the leaves of the clock branches. Therefore, the clock pulse should be "cleaned-up" by terminating the clock branch with a traditional, symmetric buffer prior to the FFs. This terminal buffer, called INVt, was used in these research simulations; it sufficiently improved the neglected clock edge without significantly impacting the performance gains.

6.1.3 Proposed Future Work

Individual SEC buffers for fine clock adjustments: The concept of using individual SEC buffers for skew reduction from *both positive and negative delay adjustments* on critical clock paths has been proposed. However, further research to quantify possible gains (as well as potential losses) on industrial benchmark circuits and to incorporate SEC within existing commercial CAD tools and flows would be valuable follow-up work.

Maximum full-chip SEC contributions: The SEC systems and equations presented here have been kept conservative in order to provide proof-of-concept results and reasonable improvement levels. Even as clock speeds approach maximum allowable frequency of a given process technology, SEC should still provide some timing performance improvements, with minimal area and power costs, although the true limits of the SEC technique need to be explored further. Most interesting would be to precisely determine the maximum amount of SEC asymmetry allowable over various clock frequencies before the neglected clock edge begins to interfere with the critical clock edge. This research could include new techniques to limit or correct for the pulse width growth problem due to the neglected edge of the clock signal in SEC.

SEC for single-edge signaling: The usage of SEC buffers to deliver two clock signals to a centrally-located up/down detectors in the QRT research (summarized in the next section) is more of a low-skew *signaling* technique, not just a clocking technique. Therefore, these alternating SEC buffers may have uses in delivering *data* or *signals* faster and sharper than traditional buffers in certain applications, which should be explored.

6.2 Post-silicon Skew Compensation

6.2.1 Summary of Contributions

The main contribution in this area was to develop and implement a framework for post-silicon clock tuning to reduce skew in the presence of on-chip PVT variations. Previous cyclic methods of multiple clock zone tuning had stability problems (e.g., mode-lock, mode-lock oscillation, and cyclic-feedback oscillation), and acyclic methods had an out of range problem that requires a complex controller. The novel cyclic method presented here, called *ring tuning*, has been designed to overcome those stability and out of range problems, without the controller. Furthermore, its hierarchical implementation has been presented to reduce the skew bound growth problem of numerous tuning zones.

The first step in creating this autonomous de-skewing system was to design an adjustable, or tunable, clock buffer (TB) capable of handling the distinctive clock signal of the SEC system. The non-critical, or neglected, clock edge of the SEC method is already degraded by the asymmetric SEC inverters, so any additional delay on that edge could eventually lead to an invalid clock signal. Therefore, a single-sided starved inverter buffer, called tsi-INVf, was created with 3-bits of linear delay tuning on the *falling edge only* -- the rising edge is unaffected. Four of these buffers were positioned across the H-tree of a typical chip creating four independent clock-tuning zones.

Next, a novel linking of those four zones into a one-directional ring configuration was implemented via a newly-designed phase detector called an Up/Down Detector (UDD). The control signals from these UDDs were designed to be extremely simple in order to minimize their area and power but, perhaps more importantly, to guarantee tuning stability. Each UDD controls a TB to form four DLL-like deskew loops interlinked in a counterclockwise ring to allow for autonomous tuning of each clock zone with its neighbor. This complete scheme is called ring tuning, with a specific implementation of four tuning zones called, Quad Ring Tuning (QRT).

Furthermore, this QRT architecture was duplicated four times at a higher clock buffer level to create a multi-level implementation. Four QRT systems can operate at clock level-3 to keep 16 clock zones synchronized, while at the same time one QRT system operates at clock level-2 to keep those other four QRT systems synchronized. The concurrent tuning of all five QRT systems (composed of 20 DLL-like tuning loops) is kept stable by the slow and simple, one-step-at-a-time, unidirectional tuning algorithm. *A very basic implementation of this multi-level QRT system has been shown to reduce full-chip clock skew by more than 50% versus an H-tree without post-silicon tuning.*

6.2.2 Limitations

The resolution of the ring tuning system is limited by the tuning range and step-size of the TBs. The simple, single-sided starved inverter design used for these TBs is limited in their number of tuning steps (4-steps used here), so a better TB design should be pursued. Several alternative designs that could be compatible with the SEC clock signals have been published, so finding a suitable 128-step (or more) TB should not be a problem.

The simplistic, one-step-at-a-time, unidirectional tuning algorithm proposed here for tuning stability has one requirement for achieving 'lock'. The range of the TBs must be 1/2 max-skew[22] in the *positive* direction and 1/2 max-skew in the *negative* direction. Therefore, it is important that the tunable buffers initially startup at the middle tuning-step, which is satisfied by the TCG design used here. This locking range requirement is only slightly more stringent (by 1/2 step-size) than the ideal case, so it does not significantly impact the QRT performance.

Also, due to its simplistic operation, this algorithm tunes within the designed threshold but does not find the optimal tuning bit settings; therefore, it is expected that the skew budgets will be *met*, but not *minimized*. For instance, if the skew budget were 20 ps, it could be expected that all devices tune to less than 20 ps of skew, but that very many of those devices are within a range of 11 ps to 19 ps; perhaps a smarter tuning algorithm could possibly reduce many of those devices down into the 5 ps to 10 ps range. However, the goal

[22] Max-skew is defined as the maximum difference in clock arrival times across all the clock tuning zones.

of this QRT algorithm was to meet the skew targets without global control, so this sacrifice in 'achievable' final skew is the tradeoff that was exercised here.

The continuous tuning of all the clock zones allows for possible concurrent adjustments of the *reference* clock and *tuned* clock, which could result in tuning oscillation. Therefore, the UDD threshold of this simple QRT system must be doubled to 1.0*ss*[23], instead of the theoretical minimum of 0.5*ss*. Adding a very simple control method of stepping through the four adjustments one at a time (such as the spinning wheel algorithm) could eliminate this 2X penalty, but its implementation is left for further study.

Lastly, the maximum final skew due to clock zone separation of this QRT system has been given as twice the UDD threshold. A stable condition could potentially exist at three times the UDD threshold if one tuning zone has prematurely reached its tuning limit, but that state should never be arrived at unless some error exists in the tuning process, so it was not considered in the analysis.

6.2.3 Proposed Future Work

Investigate better tunable buffers: The resolution of the QRT de-skewing is limited by the range and step-size of the TBs. Better TBs are available and should be investigated to improve the results.

Proper ranges and step sizes for multi-level RT: The range and step-size for both levels of the multi-level QRT system used here were arrived at empirically. Statistics and circuit theory should be used to derive equations to quantify the proper tuning range and tuning step-sizes when using multi-level RT.

[23] From Chapter 5, *ss* is the step-size of the tunable buffers; defined as the temporal distance between any two adjacent tuning settings.

Improve the tuning algorithm: Another useful direction would be to explore better (but still simple) control of clock zone tuning adjustments in order to eliminate the 2X penalty mentioned above. An algorithm such as the spinning wheel algorithm[24] should allow for stable locking without the 2X penalty. It also would be interesting to explore methods to optimize the skew compensation. Specifically, do not just settle for locking the tuning once arriving below the skew budget, but rather seek the optimal value.

Carry-out formal stability analysis: In this research the stability analysis, both of the deskew loop and of the ring tuning, has been analyzed in a general, pragmatic fashion, and the stability verification has been provided through experimental results. More formal methods of stability analysis should be conducted to theoretically prove the design criteria required for a stable single-level and multi-level ring tuning system.

Other clock networks and industrial designs: The results were obtained purely through simulation using HSPICE on an H-tree. Other architectures could be used to validate claims that it is suitable for many types of clock design for the pre- and post-silicon techniques. Furthermore, various control algorithms that enable deskewing one-time only during start-up (static version) or always during normal circuit operation (dynamic version) or some combination of the two (an intelligent gated-tuning version) could be developed. It would also be useful to carry out a theoretical analysis to prove the stability of the multi-ring scheme in static and dynamic operation to support the experimental findings.

In order to truly demonstrate its feasibility and substantiate the improvements, the approach should be applied to an existing ASIC industrial design. New techniques typically gain widespread appeal if they are shown to work in silicon on a real design.

[24] A spinning wheel algorithm could measure and adjust one clock zone, then move around the wheel to measure and adjust the next clock zone, and so on *one-step-at-a-time*, all the way around the ring until all boundaries are 'locked' below the skew threshold. (Note that the QRT system used here *concurrently* measures and adjusts all of the clock zones, necessitating the 2X threshold increase for stability.)

References

[1] V. Oklobdzija, "Multi-GHz systems clocking," *Proc. of IEEE IC-ASIC*, pp. 701-6, 2003.

[2] A. Narasimhan and R. Sridhar, “Impact of variability on clock skew in H-tree clock networks,” *Proc. of ISQED*, pp. 458-66, Mar. 2007.

[3] A. V. Mule, E. N. Glytsis, T. K. Gaylord, and J. D. Meindl, “Electrical and optical clock distribution networks for gigascale microprocessors,” *IEEE Trans. on VLSI*, vol. 10, pp. 582-94, Oct. 2002.

[4] J. Muttersbach, T. Villiger, H. Kaeslin, N. Felber, and W. Fichtner, "Globally-asynchronous locally-synchronous architectures to simplify the design of on-chip systems," *Proc. of IEEE IC-ASIC/SOC Conf.*, pp. 317-21, 1999.

[5] E. G. Friedman, “Clock distribution networks in synchronous digital integrated circuits,” *Proc. of the IEEE*, vol. 89, pp. 665–692, May 2001.

[6] R. Escovar and R. Suaya, “Optimal design of clock trees for multigigahertz applications,” *IEEE Trans. on CAD*, vol. 23, pp. 329–345, Mar. 2004.

[7] C. Yeh, G. Wilke, H. Chen, S. Reddy, H. Nguyen, T. Miyoshi, W. Walker, and R. Murgai, “Clock distribution architectures: A comparative study,” *Proc. of ISQED*, pp. 85-91, 2006.

[8] R. Chaturvedi and J. Hu, “Buffered clock tree for high quality IC design,” *Proc. of ISQED*, pp. 381-6, 2004.

[9] A. Davoodi and A. Srivastava, “Variability-driven buffer insertion considering correlations,” *IEEE Inter. Conf. on Comp. Dsgn: VLSI in Comp. and Proc.*, pp. 425-30, Oct. 2005.

[10] D. Velenis, M. C. Papaefthymiou, and E. G. Friedman, “Clock tree layout design for reduced delay uncertainty,” *IEEE Inter. SOC Conf. Proc.*, pp. 179-80, Sep. 2004.

[11] M. R. Guthaus, D. Sylvester, and R. B. Brown, “Clock buffer and wire sizing using sequential programming,” *Proc. of ACM/IEEE DAC*, pp. 1041-6, Jul. 2006.

[12] M. Hashimoto, T. Yamamoto, and H. Onodera, “Statistical analysis of clock skew variation in H-tree structure,” *Proc. of ISQED*, pp. 402-7, Mar. 2005.

[13] A. Agarwal, D. Blaauw, and V. Zolotov, “Statistical clock skew analysis considering intra-die process variations,” *Proc. of ICCAD*, pp. 914-21, Nov. 2003.

[14] D. Harris, M. Horowitz, and D. Liu, "Timing analysis including clock skew," *IEEE Trans. on CAD of ICs and Sys.*, vol. 18, pp. 1608-18, Nov. 1999.

[15] A. Hajimiri, S. Limotyrakis, and T. H. Lee, “Jitter and phase noise in ring oscillators,” *IEEE JSSC*, vol. 34, pp. 790-804, Jun. 1999.

[16] B. Stine, D. Boning, and J. Chung, “Analysis and decomposition of spatial variation in integrated circuit processes and devices, “ *IEEE Trans. on Semi.*

Mfg., vol. 10, pp. 24-41, Feb. 1997.

[17] V. Mehrotra and D. Boning, “Technology scaling impact of variation on clock skew and interconnect delay,” *Proc. of IITC*, pp. 122-4, Jun. 2001.

[18] P. Zarkesh-Ha, T. Mule, and J. Meindl, “Characterization and modeling of clock skew with process variations,” *Proc. of CICC*, pp. 441-4, May 1999.

[19] S. Sauter, D. Schmitt-Landsiedel, R. Thewes, and W. Weber, “Effect of parameter variations at chip and wafer level on clock skews,” *IEEE Trans. on Semi. Mfg.*, vol. 13, pp. 395-400, Nov. 2000.

[20] S. Zanella, A. Nardi, A. Neviani, M. Quarantelli, S. Saxena, and C. Guardiani, “Analysis of the impact of process variations on clock skew,” *IEEE Trans. on Semi. Mfg.*, vol. 13, pp. 401-7, Nov. 2000.

[21] S. Sauter, D. Cousinard, R. Thewes, D. Schmitt-Landsiedel, and W. Weber, “Clock skew determination from parameter variations at chip and wafer level,” *Proc. of IWSM*, pp. 7-9, Jun. 1999.

[22] J.-L. Tsai, L. Zhang, and C. C.-P. Chen, “Statistical timing analysis driven post-silicon-tunable clock-tree synthesis,” *Proc. of ICCAD*, pp. 575-81, Nov. 2005.

[23] Y. Elboim, A. Kolodny, and R. Ginosar, “A clock-tuning circuit for system-on-chip,” *IEEE Trans. on VLSI*, vol. 11, pp. 616-26, Aug. 2003.

[24] G. R. Wilke and R. Murgai, “Design and analysis of ‘tree + local meshes’ clock architecture,” *Proc. of ISQED*, pp. 165-70, Mar. 2007.

[25] M. Mori, H. Chen, B. Yao, and C.-K. Cheng, “A multiple level network approach for clock skew minimization with process variations,” *Proc. of ASP-DAC*, pp. 263-268, Jan. 2004.

[26] B. Liu, A. B. Kahng, X. Xu, J. Hu, and G. Venkataraman, “A global minimum clock distribution network augmentation algorithm for guaranteed clock skew yield,” *Proc. of ASP-DAC*, pp. 24-31, Jan. 2007.

[27] J.-L. Tsai, D. Baik, C.-P. Chen, and K. Saluja, “A yield improvement methodology using pre- and post-silicon statistical clock scheduling,” *Proc. of ICCAD*, pp. 611-8, Nov. 2004.

[28] E. S. Fetzer, “Using adaptive circuits to mitigate process variations in a microprocessor design,” *IEEE Dsgn and Test of Comp.*, pp. 476-83, Nov.-Dec. 2006.

[29] E. Takahashi, Y. Kasai, M. Murakawa, and T. Higuchi, “A post-silicon clock timing adjustment using genetic algorithms,” *Symp. on VLSI Circuits*, pp. 13-16, Jun. 2003.

[30] D.A. Hodges, H.G. Jackson, R.A. Saleh, *Analysis and Design of Digital Integrated Circuits: In Deep Submicron Technology*, McGraw Hill, 2004.

[31] N. Hedenstierna and K. O. Jeppson, “CMOS circuit speed and buffer optimization,” *IEEE Trans. on CAD*, vol. CAD-6, pp. 270-281, Mar. 1987.

[32] E. Friedman and S. Powell, "Design and analysis of a hierarchical clock distribution system for synchronous standard cell/macrocell VLSI," *IEEE JSSC*, vol. SC-21, pp. 240-6, Apr. 1986.

[33] D. Harris and S. Naffziger, "Statistical clock skew modeling with data delay variations," *IEEE Trans. on VLSI*, vol. 9, pp. 888-98, Dec. 2001.

[34] A. Roy, N. Mahmoud, and M. H. Chowdhury, “Effects of coupling capacitance and inductance on delay uncertainty and clock skew,” *Proc. of DAC*, pp. 184-7, Jun. 2007.

[35] A. Rajaram, B. Lu, J. Hu, R. Mahapatra, and W. Guo, “Analytical bound for unwanted clock skew due to wire width variation,” *IEEE Trans. on CAD*, vol. 25, pp. 1869-76, Sep. 2006.

[36] R. Chen, L. Zhang, V. Zolotov, C. Viswewariah, and J. Xiong, "Static timing back to our roots," *Proc. of IEEE ASPDAC*, pp. 310-5, 2008.

[37] D. Velenis, M. C. Papaefthymiou, and E. G. Friedman, “Reduced delay uncertainty in high performance clock distribution networks,” *Proc. of IEEE DAT in Europe Conf.*, pp. 68-73, Mar. 2003.

[38] P. Restle, T. McNamara, D. Webber, P. Camporese, K. Eng, K. Jenkins, D. Allen, M. Rohn, M. Quaranta, D. Boerstler, C. Alpert, C. Carter, R. Bailey, J. Petrovic, B. Krauter, and B. McCredie, "A clock distribution network for microprocessors," *Proc. of IEEE Symp. on VLSI Circuits*, pp. 184-7, 2000.

[39] B. Taskin and I. Kourtev, “Delay insertion method in clock skew scheduling,” *IEEE Trans. on CAD*, vol. 25, pp. 651–63, Apr. 2006.

[40] A. Rajaram, J. Hu, and R. Mahapatra, “Reducing clock skew variability via crosslinks,” *IEEE Trans. on CAD*, vol. 25, pp. 1176–82, Jun. 2006.

[41] G. Venkataraman, N. Jayakumar, J. Hu, P. Li, S. Khatri, A. Rajaram, P. McGuinness, and C. Alpert, "Practical techniques to reduce skew and its variations in buffered clock networks," *Proc. of IEEE/ACM ICCAD*, pp. 591-595, 2005.

[42] R. Saeidi and N. Masoumi, "Clock skew reduction by link-region technique," *Proc. of IEEE MWSCAS*, pp. 213-6, 2006.

[43] Q. Zhu and W. Dai, “High-speed clock network sizing optimization based on distributed RC and lossy RLC interconnect models,” *IEEE Trans. on CAD of IC and Systems*, vol. 15, pp. 1106-18, Sept. 1996.

[44] M. Omana, D. Rossi, and C. Metra, "Low cost scheme for on-line clock skew compensation," *Proc. of IEEE VTS*, pp 90-5, 2005.

[45] M. Shoji, “Elimination of process-dependent clock skew in CMOS VLSI,” *IEEE JSSC*, vol. 21, pp. 875-880, Oct. 1986.

[46] N. Mahapatra, A. Tareen, and S. Garimella, “Comparison and analysis of delay elements,” *Proc. of IEEE MWSCAS*, pp. 473-6, 2002.

[47] A. Chakraborty, K. Duraisami, A. Sathanur, P. Sithambaram, L. Benini, A. Macii, E. Macii, and M. Poncino, “Dynamic thermal clock skew compensation using tunable delay buffers,” *Proc. of ISLPED*, pp. 162-7, Oct. 2006.

[48] A. Chakraborty, K. Duraisami, A. Sathanur, P. Sithambaram, L. Benini, A. Macii, E. Macii, and M. Poncino, "Dynamic thermal clock skew compensation using tunable delay buffers," *IEEE Trans. on VLSI*, vol. 16, pp. 639-49, Jun. 2008.

[49] G. Geannopoulos and X. Dai, “An adaptive digital deskewing circuit for clock distribution networks,” *Proc. of ISSCC*, pp. 400-1, Feb. 1998.

[50] G. Jung, S. Hong, D. Lee, J. Park, S. Yi, Y. Kwon, U. Cho, and S. Park, “Skew variation compensating technique for clock mesh networks,” *Proc. of APCCAS*, pp. 894-7, Nov. 2008.

[51] S. Tam, S. Rusu, U. Desai, R. Kim, J. Zhang, and I. Young, “Clock generation and distribution for the first IA-64 microprocessor,” *IEEE JSSC*, vol. 35, pp. 1545-52, Nov. 2000.

[52] A. Kapoor, N. Jayakumar, and S. P. Khatri, “A novel clock distribution and dynamic de-skewing methodology,” *Proc. of IEEE*, pp. 626-31, 2004.

[53] H. Lee, H. Quang, and D. Potter, "Design self-synchronized clock distribution networks in an SOC ASIC using DLL with remote clock feedback," *Proc. IEEE ASIC/SOC*, pp. 248-52, 2000.

[54] A. Kapoor, N. Jayakumar, and S. Khatri, "Dynamically de-skewable clock distribution methodology," *IEEE Trans. on VLSI*, vol. 16, pp. 1220-9, Sep. 2008.

[55] H. Mizuno and K. Ishibashi, "A noise-immune GHz-clock distribution scheme using synchronous distributed oscillators," *Proc. of IEEE ISSCC*, pp. 404-6, 1998.

[56] M. Saint-Laurent, P. Zarkesh-Ha, M. Swaminathan, and J. Meindl, “Optimal clock distribution with an array of phase-locked loops for multiprocessor chips,” *Proc. of IEEE MWSCAS*, pp. 454-7, Aug. 2001.

[57] D. Brueske and S. Embabi, “A dynamic clock synchronization technique for large systems,” *IEEE Trans. on CPMT-Part B*, vol. 17, pp. 350-61, Aug. 1994.

[58] V. Gutnik and A. Chandrakasan, “Active GHz clock network using distributed PLLs,” *IEEE JSSC*, vol. 35, pp. 1553-60, Nov. 2000.

[59] G. Pratt and J. Nguyen, “Distributed synchronous clocking,” *IEEE Trans. on Parallel and Distributed Systems*, vol. 6, pp. 314-28, Mar. 1995.

[60] M. Saint-Laurent and M. Swaminathan, “A multi-PLL clock distribution architecture for gigascale integration,” *Proc. of CSW on VLSI*, pp. 30-5, Apr. 2001.

[61] M. Saint-Laurent, M. Swaminathan, and J. Meindl, “On the micro-architectural impact of clock distribution using multiple PLLs,” *Proc. of ICCD*, pp. 214-20, Sep. 2001.

[62] E. Backenius and M. Vesterbacka, “Reduction of simultaneous switching noise in digital circuits,” *IEEE Norchip*, pp. 187-190, 2006.

[63] J. Castro, P. Parra, M. Valencia, and A. J. Acosta, “Asymmetric clock driver for improved power and noise performances,” *Proc. of ISQED*, pp. 893-6, Mar. 2007.

[64] Y. Ismail, E. Friedman, and J. Neves, "Equivalent Elmore delay for RLC trees," *IEEE Trans. on CAD of ICs and Sys.*, vol. 19, pp. 83-97, Jan. 2000.

[65] S. Greenberg, I. Bloch, M. Horwitz, and A. Maman, "Optimization of chip level clock tree performance by using simultaneous drivers and wire sizing,"

Proc. of IEEE ICECS, pp 419-23, 2004.

[66] J. Xi and W. Dai, "Buffer insertion and sizing under process variation for low power clock distribution," *Proc. of IEEE DAC*, pp. 491-6, 1995.

[67] M. Guthaus, D. Sylvester, and R. Brown, "Process-induced skew reduction in nominal zero-skew clock trees," *Proc. of IEEE ASPDAC*, pp. 84-9, 2006.

[68] D. Velenis, R. Sundaresha, and E. G. Friedman, "Buffer sizing for delay uncertainty induced by process variations," *Proc. of IEEE ICECS*, pp. 415-418, Dec. 2004.

[69] S. Naffziger, B. Stackhouse, T. Grutkowski, D. Josephson, J. Desai, E. Alon, and M. Horowitz, "The implementation of a 2-core, multi-threaded Itanium family processor," *IEEE JSSC*, vol. 41, pp. 197-209, Jan. 2006.

[70] M. Maymandi-Nejad and M. Sachdev, "A monotonic digitally controlled delay element," *IEEE JSSC*, vol. 40, pp. 2212–9, Nov. 2005.

[71] M. Saint-Laurent and M. Swaminathan, "A digitally adjustable resistor for path delay characterization in high-frequency microprocessors," *Proc. of SSMSD*, pp. 61–4, Feb. 2001.

[72] J. Mueller and R. Saleh, "Single edge clock (SEC) distribution for improved latency, skew, and jitter performance," *Proc. of VLSID*, pp. 214-9, Jan. 2008.

[73] P. Mahoney, E. Fetzer, B. Doyle, and S. Naffziger, "Clock distribution on a dual-core, multi-threaded Itanium-family processor," *Proc. of ISSCC*, pp. 292-3, Feb. 2005.

[74] G. Allan and J. Knight, "Mixed-signal thermometer filtering for low-complexity PLLs/DLLs," *Proc. of ISCAS*, pp. 2465-8, 2006.

[75] J. Mueller and R. Saleh, "A tunable clock buffer for intra-die PVT compensation in single-edge clock (SEC) distribution networks," *Proc. of ISQED*, pp. 572-7, Mar. 2008.

[76] A. Agarwal, D. Blaauw, and V. Zolotov, "Statistical timing analysis for intra-die process variations with spatial correlations," *Proc. of ICCAD*, pp. 900-7, Nov. 2003.

[77] F. Liu, "A general framework for spatial correlation modeling in VLSI design," *Proc. of DAC*, pp. 817-22, Jun. 2007.

[78] B. Hargreaves, H. Hult, and S. Reda, "Within-die process variations: how accurately can they be statistically modeled?" *Proc. of ASP-DAC*, pp. 524-30, Mar. 2008.

[79] M. Orshansky, L. Milor, P. Chen, K. Keutzer, and C. Hu, "Impact of spatial intrachip gate length variability on the performance of high-speed digital circuits," *IEEE Trans. on CAD of IC and Sys.*, vol. 21, pp. 544-53, May 2002.

[80] A. Srivastava, D. Sylvester, and D. Blaauw, *Statistical Analysis and Optimization for VLSI: Timing and Power*, Springer, p. 6, 2005.

[81] M. Maymandi-Nejad and M. Sachdev, "A digitally programmable delay element: Design and analysis," *IEEE Trans. on VLSI*, vol. 11, pp. 871-8, Oct. 2003.

[82] T. Fischer, J. Desai, B. Doyle, S. Naffziger, and B. Patella, "A 90-nm variable

frequency clock system for a power-managed Itanium architecture processor," *IEEE JSSC*, vol. 41, pp. 218-228, Jan. 2006.